FUJITSU SEMICONDUCTOR

CONTROLLER MANUAL

CM71-xxxxx-1E

FR60

32-BIT MICROCONTROLLER

MB91460 Series

User's Manual

Version 1.10
2007-03-09

FUJITSU

FUJITSU LIMITED

FR60

32-BIT MICROCONTROLLER

MB91460 Series

User's Manual

- The contents of this document are subject to change without notice.
  Customers are advised to consult with FUJITSU sales representatives before ordering.
- The information, such as descriptions of function and application circuit examples, in this document are presented solely for the purpose of reference to show examples of operations and uses of Fujitsu semiconductor device; Fujitsu does not warrant proper operation of the device with respect to use based on such information. When you develop equipment incorporating the device based on such information, you must assume any responsibility arising out of such use of the information. Fujitsu assumes no liability for any damages whatsoever arising out of the use of the information.
- Any information in this document, including descriptions of function and schematic diagrams, shall not be construed as license of the use or exercise of any intellectual property right, such as patent right or copyright, or any other right of Fujitsu or any third party or does Fujitsu warrant non-infringement of any third-party' s intellectual property right or other right by using such information. Fujitsu assumes no liability for any infringement of the intellectual property rights or other rights of third parties which would result from the use of information contained herein.
- The products described in this document are designed, developed and manufactured as contemplated for general use, including without limitation, ordinary industrial use, general office use, personal use, and household use, but are not designed, developed and manufactured as contemplated (1) for use accompanying fatal risks or dangers that, unless extremely high safety is secured, could have a serious effect to the public, and could lead directly to death, personal injury, severe physical damage or other loss (i.e., nuclear reaction control in nuclear facility, aircraft flight control, air traffic control, mass transport control, medical life support system, missile launch control in weapon system), or (2) for use requiring extremely high reliability (i.e., submersible repeater and artificial satellite).
  Please note that Fujitsu will not be liable against you and/or any third party for any claims or damages arising in connection with above-mentioned uses of the products.
- Any semiconductor devices have an inherent chance of failure. You must protect against injury, damage or loss from such failures by incorporating safety design measures into your facility and equipment such as redundancy, fire protection, and prevention of over-current levels and other abnormal operating conditions.
- If any products described in this document represent goods or technologies subject to certain restrictions on export under the Foreign Exchange and Foreign Trade Law of Japan, the prior authorization by Japanese government will be required for export of those products from Japan.

TOC

Chapter 1 Introduction

1. How to Handle the Device

■ Device Handling Instructions

This chapter describes latch-up prevention and pin termination.

- To set latch-up prevention

  Latch up may occur on CMOS ICs when the applied voltage for input terminals or output terminals is higher than V_{CC} or lower than V_{SS}, or a voltage higher than the maximum rating voltage is applied between V_{CC} and V_{SS}. Make sure not to apply a voltage higher than the maximum rating voltage since latch up may surge electric current and result in the thermal destruction of the device.

- Termination of unused pin

  An unused pin must be terminated by a pull-up or pull-down resistor externally, or by switching on the internal pull-up or pull-down resistor before enabling the pin inputs to avoid transverse current.

- Power-supply pin

  If multiple V_{CC} and V_{SS} exist, as a matter of device design, they are connected to each other to prevent an error when their voltage should be identical in the device. In order to reduce unnecessary radiation, prevent an strobe signal error due to upward ground level, and comply with total output current standard, be sure to externally connect them to power supply and ground. Give consideration to connect $V_{CC\ and}$ V_{SS} of the device from power supply at low impedance.

  Near the device, it is preferable to connect about 0.1uF ceramic capacitor as a bypass capacitor between V_{CC} and V_{SS}.

- Crystal-oscillator circuit

  Noise to X0 or X1 pin may cause an error. Make a design for printed board to closely allocate X0, X1, crystal oscillator (or ceramic oscillator), bypass capacitor towards ground and the device.

  It is recommended to make a printed board artwork which surrounds X0 and X1 pins using ground.

  The above recommendations also apply to the subclock oscillator pins X0A and X1A.

- NC and OPEN pin termination

  Do not terminate NC pin and OPEN pin to use.

- Mode pins (from MD0 to MD2)

  Connect pins from MD0 to MOD2 directly to V_{CC} or V_{SS} to use. To avoid entering test mode due to noise, make a short pattern length between each mode pin on printed board and V_{CC} or V_{SS} to connect pins at low impedance.

- At the time of power-on

  Immediately after power-on operation, be sure to reset $\overline{\text{INIT}}$ pin to initialize the setting (INIT). Immediately after power-on operation, to ensure the oscillation stabilization time required for oscillation circuit, hold “L”-level input to the pin during the oscillation stabilization time required for oscillation circuit. (INIT operation on the pin initializes the setting for oscillation stabilization time to minimum value.)

- Source oscillation input at the time of power-on

  At the time of power-on, be sure to input the clock until the oscillation stabilization wait is over.

- Caution: during the PLL clock operation

  Even if oscillator is disconnected or input is stopped while selecting PLL clock, self-excited oscillation circuit in the PLL may continue running at self-running frequency. This self-running operation is not covered by guarantee.

- For more specification about operating voltage, see the latest data sheet.
- Unused oscillator pins

  If an oscillator is not used (e.g. subclock oscillator pins X0A/X1A available, but not used), connect the X0 pin to ground and leave the X1 pin open.

2. Instruction for Users

■ Clock Controls

By inputting “L” to $\overline{\text{INIT}}$, ensure clock oscillation stabilization time.

■ Switching of dual-purpose port

Use PFR (Port function register) to switch between PORT and dual-purpose port.

■ Low-power-consumption mode

- For standby mode, enable synchronous standby (TBCR.SYNCS=“1”) and then use the following sequences.

```
(LDI    #value_of_standby, R0   ;   value_of_standby is a write data to STCR
(LDI    #_STCR, R12             ;   _STCR is the STCR address. (481H)
STB     R0, @R12                ;   Write to Standby Control Register (STCR).
LDUB    @R12, R0                ;   STCR read for synchronous standby.
LDUB    @R12, R0                ;   Dammy read STCR again.
NOP                             ;   NOP x 5 for timing adjustment.
NOP
NOP
NOP
NOP
```

In addition, after returning from standby, set I flag, ILM and ICR in order to branch to interrupt handler which triggered the return.

- If you use monitor debugger, you should avoid the following.
  - Do not set breakpoints for command sequence above.
  - Do not conduct stepwise execution for command sequence above.

■ Power-on sequence

Power-on and power-off sequence valid for MB91V460 Rev.A. Please review the datasheets of the flash devices for a valid power-on and power-off sequence on those devices.

Power-on sequence: (1) VDD5 , VDD35, HVDD5, VDD5R (2) AVCC, AVRH, V0-V3
Power-off sequence: (1) AVCC, AVRH, V0-V3 (2) VDD5 , VDD35, HVDD5, VDD5R

The power supply V3 for LCD must not exceed VDD5. The power-on of V3 should be carried out after power-on of VDD5R and VDD5. To power on analogue power supply AVCC and analogue signal, power VDD5R and VDD5 on before.

■ Power supply operating conditions

Power supply recommendation valid for MB91V460 Rev.A. Please review the datasheets of the flash devices for a recommendation of the power supply conditions on those devices.

[VDD5 = HVDD5 = AVCC] >= VDD35. This is the recommended condition.

■ Caution: PS register

Because some commands previously proceed PS register, interrupt processing routine may be broken during the use of debugger or displayed data on PS flag may be updated due to the following excecptional operations ((1) and (2)).

In each case, it is designed to correctly re-proceed operations after the return, the operation before and after EIT is carried out in accordance with the specification.

- **In immediately preceding DIV0U or DIV0S command,**
  - If interrupted by user,
  - If stepwise execution is carried out,
  - If data event or emulator menu made a break,
    The following operation may be generated.
    1. D0 or D1 flag is updated ahead.
    2. EIT processing routine (interruption by user or emulator) is carried out.
    3. After the return from EIT, it executes DIV0U or DIV0S command and then D0 or D1 flag are updated to the same value as 1.

- **If you execute each command of ORCCR, STILM, MOV Ri or PS to enable interruption with interruption by user generated, the following operation may be generated.**
    4. PS register is updated ahead.
    5. EIT processing routine (interruption by user) is carried out.
    6. After the return from EIT, it executes commands above, and then PS register is updated to the same value as 1.

■ Watchdog timer function

Watchdog timer function equipped with FR60 monitors the progress to ensure that program executes reset delay operation within a specified time and resets CPU if reset delay operation was not executed due to runaway of program. Once you enable watchdog timer function, it continues running until it is reset.

By way of exception, reset delay is automatically conducted under the condition where CPU program execution is stopped. For this exceptional condition, see “Chapter 20 Software Watchdog Timer (Page No.275)“.

■ Register against read/modify/write command

SMR register within UART cannot use read/modify/write command. To write in SMR register, write by Byte/ Half-word/Word in consideration with write control bit (bit-5, 4, 2, 0) rather than accessing by bit-by-bit.

■ Caution: writing to registers which include a status flag

Writing to a register including a status flag (in particular, interrupt request flag) in order to control the function, note that you should not clear status flag unintentionally.

That is, take care not to clear the flag for status bit and make control bit to be the expected value during the writing.

Especially, for control bits consisting of several bits, bit command is not available since single bit access is only acceptable for bit command, you should write into the both of control bit and status flag at the same time by Byte/Half-word/Word access. In this case, you should not clear other bits (bits of status flag) unintentionally.

The following shows registers which mostly include both of several bits and status flag.

- TBCR
- OSCR
- TWCR
- TCCS0, TCCS1
- ICS01
- TMCSR0, TMCSR1, TMCSR2, TMCSR3
- PCN00, PCN01, PCN02,...
- ADCSL0, ADCSL1
- CCR0, CCR1

Note: For bit command, you do not have to be careful since this matter has been already considered.

■ Caution: writing to registers which include a status flag

Writing to a register including a status flag (in particular, interrupt request flag) in order to control the function, note that the actual writing to the registers may be delayed. This is because of using write buffers on the busses to the resources which accept a write access from CPU immediately but can access the resource registers delayed.

In this case it can happen that within an ISR the interrupt request flag is cleared by writing to the register and the ISR is completed with RETI, but the interrupt request flag is still active and the ISR is executed again.

To synchronize the access to the resources on this architecture please follow this recommendation:

Use a read access (byte or halfword) to the RBSYNC address to synchronize the CPU operation (e.g. the interrupt acceptance of the CPU) to a preceding write access to the resources on R-bus (e.g. to an interrupt flag) on following addresses (0x0000-0x01FF, 0x0280-0x037F, 0x0400-0x063F and 0x0C00-0x0FFF).

Use a read access (byte or halfword) to the CBSYNC address to synchronize the CPU operation (e.g. the interrupt acceptance of the CPU) to a preceding write access to the CANs on D-bus (e.g. to an interrupt flag) on following addresses (0xC000-0xFFFF).

3. Caution: debug-related matters

■ Stepwise execution of RETI command

Under the circumstances where interruption is often generated when carrying out stepwise execution, only relevant interrupt processing routine is repeatedly executed after the stepwise execution of RETI. Therefore, main routine or low-level interruption program will not be executed.

To avoid this problem, do not proceed stepwise execution of RETI command.

Or, upon the time when no debug is needed for relevant interrupt routine, proceed the debug by prohibiting relevant interruptions.

■ Operand break

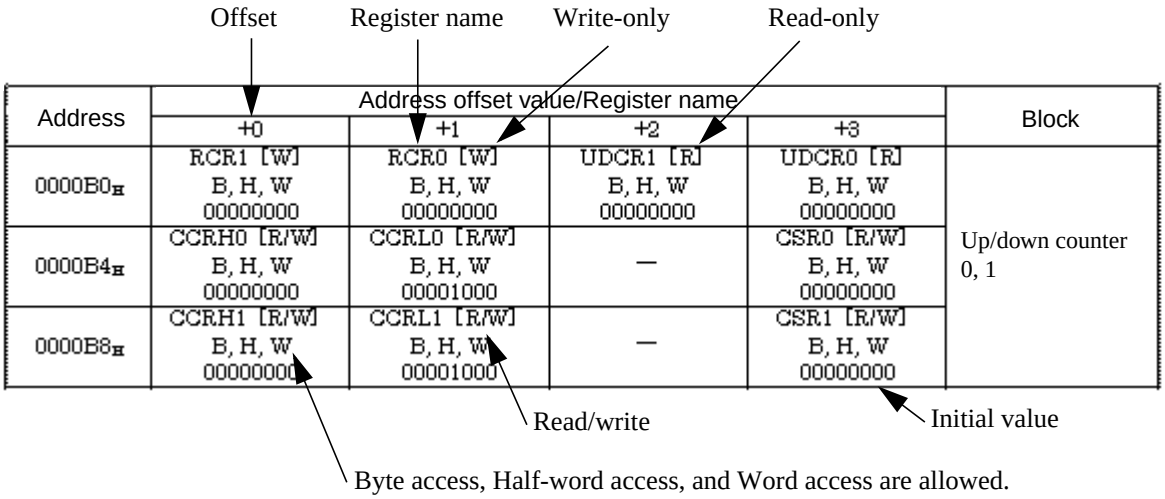
Do not set the access for area including system stack pointer address as the target for data event break.

4. How to Use This Document

■ Main terminology: This table shows main terminology used for FR60.

Term	Meaning
I-bus	32-bit-wide bus for internal instruction. Since FR60 series employ internal Harvard architecture, instruction and data are independent bus. For I-bus, Harverd/Prinston-bus-converter is connected.
D-bus	Internal 32-bit-wide data bus. For D-bus, bit search module, Harverd/Prinston-bus-converter, R-bus interface (32-bit⇔16-bit Bus-converter), and CAN modules are connected.
F-bus	Internal 32-bit-wide bus. F-bus is connected to embedded Flash/ROM and embedded RAM.
R-bus	Internal 16-bit-wide data bus. R-bus is connected to D-bus via R-bus-converter. For R-bus, peripheral function, I/O, clock generator and interrupt controller are connected.
X-bus	32-bit-wide address and data bus. Via bus-converter for external bus, it accesses to external bus.
Main clock ($F_{CL-MAIN}$)	This a clock which acts as a benchmark for LSI operation triggered by high-speed-side oscillation. This is connected to main clock oscillation stabilization timer and clock generator.
Subclock (F_{CL-SUB})	This a clock which acts as a benchmark for LSI operation triggered by low-speed-side oscillation. This is connected to sub oscillation stabilisation timer, real-time clock and clock generator.
Base clock (Φ)	At the maximum speed, base clock has the same cycle as source oscillation. In PLL of the clock generator, base clock has clock multiplied by 1, 2, 3, 4, 5, 6, 7 and 8 or clock divided by 2. Base clock is basis clock which generates CLKB, CLKP and CKLT in the clock generator.
CPU clock (CLKB)	CPU clock is the clock which is referred by CPU, embedded ROM, embedded RAM, bit search module and internal bus (I-bus, D-bus, F-bus and X-bus) operations. Generated from the base clock in the clock generator.
Peripheral clock (CLKP)	Peripheral clock is the clock which is referred by each peripheral function (peripheral functions other than bit search module and CAN) connected to R-bus and R-bus, clock control, interrupt controller, I/O port and external interrupt input d operations. Generated from the base clock in the clock generator.
External bus clock (CLKT)	External bus clock is the clock which is referred by external expansion bus interface connected to X-BUS and external clock output operations. Generated from the base clock in the clock generator.
CAN clock (CLKCAN)	CAN clock is the clock which is referred by the CAN modules. Generated from the non modulated PLL output clock to ensure operation within CAN network oscillation tolerances.
Main clock mode	Mode which runs based on main clock. This main clock mode has status such as main RUN, main sleep, main stop, oscillation stabilization wait RUN, oscillation stabilization wait reset and program reset.
Subclock mode	Mode which runs based on subclock. This subclock mode has status such as sub RUN, sub sleep, sub stop, subclock oscillation stabilization wait RUN and program reset.
Main RUN	Main RUN is the status which is in main clock mode and also all circuits are operable.
Sub RUN	Sub RUN is the status which is in subclock mode and also all circuits are operable.
Oscillation stabilization time	Upon the reset (INITX, RST), return from stop, return from PLL abnormal operation, generation of watchdog and during main clock stop, it takes oscillation stabilization time for main clock. Time base timer counts the time.
Main clock oscillation stabilization wait	Wait time until main clock oscillates after main clock stops in subclock mode. Main clock oscillation stabilization timer counts the time.

■ Access size and address position



There are three kinds of accesses such as Byte access, Half-word access and Word access. However, note that some registers have restricted access. For more information, see “3.2. I/O Map (Page No.24)” or “Detail Description of Register” in each chapter.

B,H,W	: Byte access, Half-word access, and Wordaccess are allowed.
B	: Byte access (Be sure to access by Byte.)
H	: Half-word access (Be sure to access by Half-word.)
W	: Word access (Be sure to access by Word.)
B, H	: Byte access, Half-word access only (Word access is not allowed.)
H,W	: Half-word access, Word access only (Byte access is not allowed.)

Reference

The following describes address position to access.

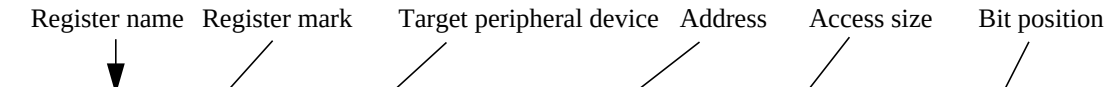
- In Word access, address becomes multiple of 4. (Lowest order 2 bits mandatorily become “00”.)
- In Half-word access, address becomes multiple of 2. (Lowest order 1 bit mandatorily becomes “0”.)
- In Byte access, address will not be changed.

Therefore, for example, make RCR0 register to use Half-word access,

For address 0B0H, RCR1+RCR0 register is accessed.

(When address offset is +1 and +2, (Example: RCR0+UDCR1) Half-word access is not allowed.)

■ About access size and bit position



(1) Counter control register (Higher byte)

This is the register (higher byte) which controls up/down counter operation.

CCRH0 (Up/down counter 0): address 00B4h (Access: Byte, Half-word, Word)

CCRH1 (Up/down counter 1): address 00B8h (Access: Byte, Half-word, Word)

15	14	13	12	11	10	9	8	bit
M16E/Reserved	CDCF	CFIE	CLKS	CMS1	CMS0	CES1	CES0	
0	0	0	0	0	0	0	0	Initial value
R/W *	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Attribute

bit15: Enable 16-bit mode

M16E (CCRH0 only)	Enable 16-bit mode
0	8-bit x 2-channel mode (8-bit mode)
1	16-bit x 1-channel mode (16-bit mode)

*: CCRH1: ReservedAlways write 0 for writing. The read value is indeterminate.

When access size changes, bit position changes.

- In the case that address offset value is +0 (Example: CCRH0 register)

Access size	Address	Bit position							
Byte	$0B4_H+0_H$	07	06	05	04	03	02	01	00
Half-word	$0B4_H+0_H$	15	14	13	12	11	10	09	08
Word	$0B4_H+0_H$	31	30	29	28	27	26	25	24
Bit name		M16E	CDCF	CFIE	CLKS	CMS1	CMS0	CES1	CES0

- In the case that address offset value is +1 (Example: CCRL0 register)

Access size	Address	Bit position							
Byte	$0B4_H+1_H$	07	06	05	04	03	02	01	00
Half-word	$0B4_H+0_H$	07	06	05	04	03	02	01	00
Word	$0B4_H+0_H$	23	22	21	20	19	18	17	16
Bit name		Reserved	CTUT	UCRE	RLDE	UDCC	CGSC	CGE1	CGE0

- In the case that address offset value is +2 (Example: UDCR1 register)

Access size	Address	Bit position							
Byte	$0B0_H+2_H$	07	06	05	04	03	02	01	00
Half-word	$0B0_H+2_H$	15	14	13	12	11	10	09	08
Word	$0B0_H+0_H$	15	14	13	12	11	10	09	08
Bit name		D15	D14	D13	D12	D11	D10	D9	D8

- In the case that address offset value is +3 (Example: UDCR 1 register)

Access size	Address	Bit position							
Byte	$0B0_H+3_H$	07	06	05	04	03	02	01	00
Half-word	$0B0_H+2_H$	07	06	05	04	03	02	01	00
Word	$0B0_H+0_H$	07	06	05	04	03	02	01	00
Bit name		D7	D6	D5	D4	D3	D2	D1	D0

■ Meaning of Bit Attribute Symbols

R : Readable
W : Writable
RM : Reading operation during read/modify/write operation.
"/" (Slash) R/W: Readable and writable. (The read value is the value written.)
"," (comma) R,W: Values are different between read and write. (The read value is different from the value written.)
R0 : The read value is "0".
R1 : The read value is "1".
W0 : Always write "0".
W1 : Always write "1".
(RM0) : read/modify/write operation reads "0".
(RM1) : read/modify/write operation reads "1".
RX : The read value is indeterminate. (Reserved bit or undefined bit)
WX : Writing does not affect the operation. (Undefined bit)

- **Example of how R/W is used**
  - R/W : Readable and writable. (The read value is the value written.)
  - R,W : Readable and writable. (The read value and written value are different.)
  - R,RM/W : Readable and writable. (The read value and written value are different. Read/modify/write command reads the value written.) Example: port data register
  - R(RM1),W : Readable and writable. (The read value and written value are different. Read/modify/write command reads 1.) Example: interrupt request flag
  - R/WX : Read-only (Read-only. Writing does not affect the operation.)
  - R1,W : Write-only (Write-only. The read value is 1.)
  - R0,W : Write-only (Write-only. The read value is 0.)
  - RX,W : Write-only (Write-only. The read value is indeterminate.)
  - R/W0 : Reserved bit (The written value is 0. The read value is the value written.)
  - R0/W0 : Reserved bit (The written value is 0. The read value is 0.)
  - R1,W0 : Reserved bit (The written value is 0. The read value is 1.)
  - RX,W0 : Reserved bit (The written value is 0. The read value is indeterminate.)
  - R/W1 : Reserved bit (The written value is 1. The read value is the value written.)
  - R1/W1 : Reserved bit (The written value is 1. The read value is 1.)
  - R0,W1 : Reserved bit (The written value is 1. The read value is 0.)
  - RX,W1 : Reserved bit (The written value is 1. The read value is indeterminate.)
  - RX/WX : Undefined bit (The read value is indeterminate. Writing does not affect the operation.)
  - R0/WX : Undefined bit (The read value is 0. Writing does not affect the operation.)

Chapter 2 MB91460 Rev.A/Rev.B Overview

1. Overview

MB91460 is a series of standard microcontrollers containing a range of I/O peripherals and bus control functions. MB91460 features a 32-bit RISC CPU (FR60 series) core and is suitable for embedded control applications requiring high-performance and high-speed CPU processing. MB91460 derivatives also contain up to 16 kByte instruction cache memory and other internal memories to improve the execution speed of the CPU.

MB91460 Rev.B has the same features as MB91460 Rev.A and adds some additional components in order to support infotainment applications. Which components and modules will be included in MB91460 Rev.B is not yet decided finally. So this document gives only a proposal at this stage of development.

MB91460 Rev.B : This series is presently being specified, and not available yet.

2. Features

2.1 FR60 CPU Core

- 32-bit RISC, load/store architecture, pipeline 5 stages
- Maximum operating frequency: Core clock = 100 MHz (device dependent)
  (Source oscillation= 4 MHz, multiplied by 25 (PLL clock multiplier method))
- General-purpose registers: 16 x 32 bits
- 16-bit fixed-length instruction (Base instruction)
- 32-bit linear address space: 4 Gbytes
- Instructions suitable for embedded application
  - Transfer command between memories
  - Bit-processing instruction
  - Barrel-shift instructions
- Instructions supporting C-language
- Function's enter command /exit command
- Multi-load/store command of register contents
- Assembler statement is also easily available
  Register's interlock function
- Multiplier's embedded application/command level support
  - Signed 32-bit multiplication: 5 cycles
  - Signed 16-bit multiplication: 3 cycles
- Interrupt (PC/PS are saved): 6 cycles (16 priority level)
- Harvard architecture enables simultaneous execution of program access and data access
- Memory protection function
- Embedded debug support
- Commands compatible with FR family

2.2 Instruction Cache

- 2 way set associative I-cache
  - Up to 4 kByte integrated

    - 4 words (16 bytes) per set
    - Variable capacity (4/2/1 kB)
    - Lock function enabling programs to be resident
    - Available as instruction RAM requiring no wait state when not used as an instruction cache
  - Direct mapped I-cache
    - Up to 16 kByte integrated
    - Variable capacity (16/8/4/2/1 kB)
    - Lock function enabling programs to be resident

2.3 Interrupt Controller

- A total of 17 external interrupt lines (1 nonmaskable interrupt pin, 8 normal interrupt pins, 8 interrupt pins shared (with peripheral inputs for Wake Up from STOP mode, e.g. CAN RX)
- Interrupts from internal peripherals (128 interrupt vectors)
- Priority levels programmable for normal interrupt lines excluding the nonmaskable one (16 levels)
- Capable of using the normal interrupt and nonmaskable interrupt pins for Wake Up from STOP mode

2.4 Internal Data RAM

- Up to 64 kBytes integrated
- Zero wait state for read/write access
- *Referenced as Data-RAM or D-RAM in this manual*

2.5 Internal Instruction/Data RAM

- Up to 64 kBytes integrated
- Zero wait state for read/write access of instructions
- One wait state for read/write access of data
- *Referenced as General-Purpose-RAM (GP-RAM) or I/D-RAM in this manual*

2.6 Embedded Instruction/Data Memory

- Up to 4 MByte (Flash or Mask ROM)
- Programmable wait state for read/write access
- Flash/ROM security

2.7 External Bus Interface

- 8 chip select areas with individual area size, data bus width selection (8, 16, 32-bit) and wait
- Address bus up to 32 bit wide
- Programmable auto-wait function or external wait input (RDY)
- Basic bus cycles : 2 cycles
- Prefetch function
- Burst access function

2.8 DMA Controller

- Four transfer modes supported: single/block, burst, continuous transfer, and fly-by
- 5 channels (4 channels for external-to-external transfer)

- 3 types of transfer sources (external pins/internal peripherals/and software)
- Up to 128 selectable internal transfer sources
- Addressing mode: Specifying up to 32-bit addresses (Increment/decrement/fixed)
- Transfer mode (Demand transfer/burst transfer/step transfer/block transfer)
- Fly-by transfer supported (between external I/O and memory)
- Transferred data size selectable from among 8, 16, and 32 bits

2.9 Infotainment extension (MB91460 Rev.B)

- Inter-IC sound bus (I2S)
  - master or slave operation
  - operation up to 2.5 MBit/s
- MOST support (MediaLB for controlling an external MOST IC is integrated)
  - Digital interface to external MOST controller
  - Frame sync pattern support
  - Scalable data rate for streaming, packet, control, isochronous
  - System-broadcast channel for administration
  - Broadcast support for synchronous data
- USB
  - USB 1.1 compliant up to 12Mbit/s (USB 2.0 tbd)
  - Configurable endpoints
  - Supports control, bulk, interrupt and isochronous transfer
  - built-in FIFO for all endpoints
  - clock and data recovery
- Flexray
  - Event or Time triggered protocol
  - Asynchronous or synchronous operation
  - Fault tolerant operation (2 channels supported)
  - Static and dynamic data transfer
  - Transfer rates up to 10 MBit/s
- Ethernet

MB91460 Rev.B : This series is presently being specified and not available yet.

2.10 Peripheral Function

- General-purpose port : Up to 288
  - N channel open drain port out of above: 8 (for I^2C)

- A/D converter : 32 channels (1 unit)
  - Series-parallel type
  - Resolution: 10 bits
  - Minimum conversion time: 3us
  - Single conversion mode
  - Continuous conversion mode
  - Stop conversion mode
  - Activation by software or external trigger can be selected
  - Reload timer 7 and A/D Converter co-operate

- D/A converter : 2 channels
  - R-2R type
  - Resolution: 10 bits
  - Conversion rate: 0.45us (when 20 pF load is applied)
  - Conversion rate: 2us (when 100 pF load is applied)

- Alarm comparator : 2 channels
  - Monitors an external voltage and generates an interrupt in case of a voltage lower or higher than the defined thresholds
  - Status is readable, interrupts can be masked separately

- External interrupt input : 16 channels
  - Can be programmed to be edge sensitive or level sensitive
  - Interrupt mask and request pending bits per channel
  - 6 channels combined with CAN RX for wakeup

- Non maskable interrupt (NMI) : 1 channel
  - Highest priority of all user interrupts

- Bit search module (using REALOS)
  - Function to search the first bit position of “1”, “0”, “Changed” from MSB (most significant bit) within 1 word

- Up/down counter : 16 bits x 2 channels (8 bits x 4 channels)
  - Timer mode, up/down count mode, phase difference mode (x2, x4)
  - Includes clock prescaler ($f_{RES}/2^1$, $f_{RES}/2^3$)

- Reload timer : 16 bits x 8 channels

  - 16-bit reload counter
  - Includes clock prescaler ($f_{RES}/2^1$, $f_{RES}/2^3$, $f_{RES}/2^5$, $f_{RES}/2^6$, $f_{RES}/2^7$)

- Free-run timer : 16 bits x 8 channels
  - 16-bit free running counter, signals an interrupt when overflow or match with compare register
  - Includes prescaler ($f_{RES}/2^2$, $f_{RES}/2^4$, $f_{RES}/2^5$, $f_{RES}/2^6$)
  - Timer data register has R/W access

- PPG : 16 bit x 16 channels
  - 16 bit down counter, cycle and duty setting registers
  - Interrupt at triggering, cycle or duty match
  - PWM operation and one-shot operation
  - Internal prescaler allows $f_{RES}/2^0$, $f_{RES}/2^2$, $f_{RES}/2^4$, $f_{RES}/2^6$ as counter clock
  - Can be triggered by software, reload timer or external trigger
  - Reload timer 0/1 available as trigger for PPG 0/1/2/3
  - Reload timer 2/3 available as trigger for PPG 4/5/6/7
  - Reload timer 4/5 available as trigger for PPG 8/9/10/11
  - Reload timer 6/7 available as trigger for PPG 12/13/14/15
  - External trigger for PPG 0/8 (shared)
  - External trigger for PPG 1/9 (shared)
  - External trigger for PPG 2/10 (shared)
  - External trigger for PPG 3/11 (shared)
  - External trigger for PPG 4/12 (shared)
  - External trigger for PPG 5/13 (shared)
  - External trigger for PPG 6/14 (shared)
  - External trigger for PPG 7/15 (shared)

- Input capture : 16 bits x 8 channels
  - Rising edge, falling edge or rising & falling edge sensitive
  - Free-run timer 0 available as trigger for input capture 0/1
  - Free-run timer 1 available as trigger for input capture 2/3
  - Free-run timer 4 available as trigger for input capture 4/5
  - Free-run timer 5 available as trigger for input capture 6/7

- Output compare : 16 bits x 8 channels
  - Signals an interrupt when a match with of 16-bit IO timer occurs
  - An output signal can be generated
  - Free-run timer 2 available as trigger for output compare 0/1
  - Free-run timer 3 available as trigger for output compare 2/3
  - Free-run timer 6 available as trigger for output compare 4/5
  - Free-run timer 7 available as trigger for output compare 6/7

- LIN-USART (LIN=Local Interconnect Network) : 16 channels
  - Full-duplex double buffer system (4 ch with 16 byte RX/TX FIFO buffer each)
  - With parity/without parity selectable
  - 1 or 2 stop bits selectable
  - 7 or 8 bits data length selectable
  - NRZ type transfer format
  - Asynchronous /synchronous communications selectable
  - Master-slave communication function (multiprocessor mode)
  - Dedicated baud rate prescaler is embedded in each channel
  - External clock is able to use as transfer clock
  - Parity error, frame error, and overrun error detecting functions
  - SPI compatible
  - LIN master and slave
  - LIN USART 0/8 and ICU 0 co-operate (for LIN sync field in slave mode)
  - LIN USART 1/9 and ICU 1 co-operate (for LIN sync field in slave mode)
  - LIN USART 2/10 and ICU 2 co-operate (for LIN sync field in slave mode)
  - LIN USART 3/11 and ICU 3 co-operate (for LIN sync field in slave mode)
  - LIN USART 4/12 and ICU 4 co-operate (for LIN sync field in slave mode)
  - LIN USART 5/13 and ICU 5 co-operate (for LIN sync field in slave mode)
  - LIN USART 6/14 and ICU 6 co-operate (for LIN sync field in slave mode)
  - LIN USART 7/15 and ICU 7 co-operate (for LIN sync field in slave mode)

- CAN : 6 channels
  - Supports CAN protocol version 2.0 part A and B
  - Bit rates up to 1 Mbit/s
  - Up to 128 message objects
  - Each message object has its own identifier mask
  - Programmable FIFO mode (cocatenation of message objects)
  - Maskable interrupt
  - Disabled Automatic Retransmission mode for Time Triggered CAN applications
  - Programmable loop-back mode for self-test operation

- I^2C (400k fast mode) : 4 channels
  - Master or slave transmission
  - Arbitration function
  - Clock synchronization function
  - Slave address and general call address detect function
  - Transfer direction detect function
  - Start condition repeat generation and detection function
  - Bus error detect function
  - Compatible to I2C standard and fast mode specification (operation up to 400 kHz, 10 bit addressing)
  - Includes clock divider functionality
  - SCL and SDA lines include optional noise filter. The noise filter allows the suppression of spikes in

the range of 1 to 1.5 cycles of the resource clock (CLKP)

- PFM (pulse frequency modulator) : 16 bits x 1 channel
  - 16-bit reload timers for generating high/low pulse waveforms
  - Includes clock prescaler ($f_{RES}/2^1$, $f_{RES}/2^3$, $f_{RES}/2^5$, $f_{RES}/2^6$, $f_{RES}/2^7$)

- Sound Generator : 1 channel
  - 8-bit PWM signal is mixed with tone frequency from 16-bit reload counter
  - PWM clock by internal prescaler: $f_{RES}/2^0$, $f_{RES}/2^1$, $f_{RES}/2^2$, $f_{RES}/2^3$, $f_{RES}/2^4$
  - Tone frequency: PWM frequency / 2 / (reload value + 1)

- Stepper Motor Controller : 6 channels
  - Four high current outputs for each channel
  - Two synchronized 8/10-bit PWMs per channel
  - Internal prescaling for PMW clock: $f_{RES}/1$, $f_{RES}/4$, $f_{RES}/5$, $f_{RES}/6$, $f_{RES}/8$, $f_{RES}/10$, $f_{RES}/12$, $f_{RES}/16$

- LCD controller
  - 4 common / 40 segment
  - Display: Up to 160 cells (for 1/4 duty cycle)
  - Duty cycle: Selectable from options: 1/2, 1/3 and 1/4
  - Bias: Fixed at 1/3
  - Frame period: Selectable from four options. (for clock, peripheral clock or subclock is selectable)
  - Driver: Built-in (for internal divided resistors), or external divided resistors can be connected to the V0-V3 pins
  - Data memory: Built-in 16-byte data memory for display
  - Stop mode: Enable LCD display in the sub-stop mode
  - Blank display: Selectable
  - Pin: The SEG0-39 of COM0-4 pin usage can be switched between general and specialized purposes
  - Others: External divided resistors can be also used to shut off the current when LCD is deactivated

- Timebase/watchdog timer (26 bits)
  - Adjustable watchdog timer interval (between 2^{20} and 2^{26} system clock cycles)

- RC oscillator watchdog timer (16 bits)

- Real-time clock (counts during stop mode)
  - RTC module can be clocked either from 32 kHz crystal, 4 MHz quartz or from the RC Oscillator
  - Facility to correct oscillation deviation (subclock calibration)
  - Read/write accessible second/minute/ hour registers
  - Can signal interrupts every halfsecond/second/ minute/hour/day
  - Internal clock divider and prescaler provide exact 1s clock based on a 4 MHz or a 32 kHz clock input
  - Prescaler value for 4 MHz is $0F423F_H$

    - Prescaler value for 32 kHz is $001FFF_H$

- Clock monitor (clock output function): 1 channel

- Clock supervisor
    - Monitors external 32kHz and 4MHz for fails (e.g. crystal breaks)
    - Switches in case of fail to an available recovery clock (subclock or RC clock)

- Clock modulator (reduction of EME)

- Subclock calibration
    - Calibration of the RTC timer in 32 kHz or RC oscillator operation, based on the more accurate 4 MHz quartz is possible

- Main oscillation stabilisation timer
    - 23 bit counter for main oscillation stabilisation wait when running in sub clock mode
    - Generates an interrupt when stabilisation time has elapsed

- Sub oscillation stabilisation timer
    - 15 bit counter for sub oscillation stabilisation wait when running in main clock mode
    - Generates an interrupt when stabilisation time has elapsed

- Low-consumption modes: SLEEP/Sub-Run/RTC/STOP function

- Package: BGA660-03EK (Evaluation device MB91V460)

- CMOS 0.35um technology (Evaluation device MB91V460)

- Power supply (Evaluation device MB91V460):
    - Single power supply 3.3V
    - Single power supply 5V

3. MB91460 Series Product Lineup

Feature	MB91V460 Rev.A	MB91V460 Rev.B	MB91F467DA
Core frequency	80 MHz	100 MHz	96 MHz
Resource frequency	40 MHz	50 MHz	48 MHz
Technology	0.35um	0.18um	0.18um
Watchdog	yes	yes	yes
Watchdog (RC osc. based)	yes (disengageable)	yes (disengageable)	yes
Bit Search	yes	yes	yes
Reset Input (INITX)	yes	yes	yes
Clock Modulator	yes	yes	yes
Low Power Mode	yes	yes	yes
DMA	5 ch	5 ch	5 ch
MAC (uDSP)	no	tbd	no
MMU/MPU	MPU (16 ch) [1)]	MPU (16 ch) [1)]	MPU (8 ch) [1)]
Flash	Emulation SRAM	Emulation SRAM	1 MByte
Flash Protection	n.a.	n.a.	yes
D-bus RAM	64 kByte	64 kByte	32 kByte
I/D-bus RAM	64 kByte	64 kByte	32 kByte
I-bus RAM / I-Cache	16 kByte	16 kByte	8 kByte
Boot-ROM / BI-ROM	4 kByte	4 kByte	4 kByte
RTC	1 ch	1 ch	1 ch
Free Running Timer	8 ch	8 ch	8 ch
ICU	8 ch	8 ch	8 ch
OCU	8 ch	8 ch	4 ch
Reload Timer	8 ch	8 ch	8 ch
PPG 16-bit	16 ch	16 ch	12 ch
PFM 16-bit	1 ch	1 ch	1 ch
Sound Generator	1 ch	1 ch	1 ch
Up/Down Counter (8/16-bit)	4 ch (8-bit) / 2 ch (16-bit)	4 ch (8-bit) / 2 ch (16-bit)	3 ch (8-bit) / 1 ch (16-bit)
C_CAN (32 MsgBuf)	6 ch (128msg)	6 ch (128msg)	3 ch (32msg)
LIN-USART	4 ch + 4 ch FIFO + 8 ch	4 ch + 4 ch FIFO + 8 ch	1 ch + 4 ch FIFO
I2C (400k)	4 ch	4 ch	3 ch
FR external bus	yes (32bit addr, 32bit data)	yes (32bit addr, 32bit data)	yes (26bit addr, 32bit data)
External Interrupts	16 ch	16 ch	10 ch
NMI Interrupts	1 ch	1 ch	no
SMC	6 ch	6 ch	6 ch
LCD controller (40x4)	1 ch	1 ch	no

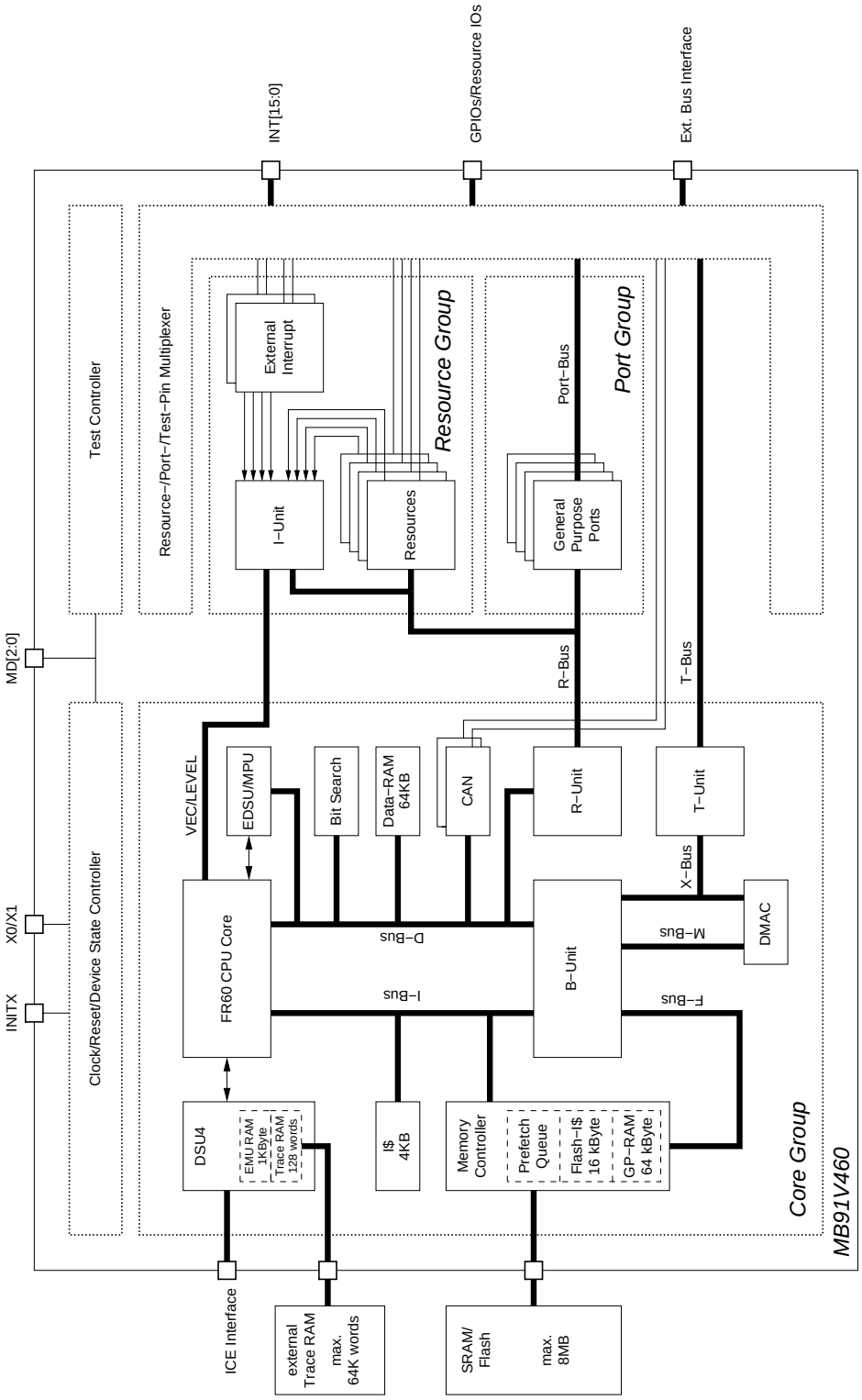
Feature	MB91V460 Rev.A	MB91V460 Rev.B	MB91F467DA
ADC (10 bit)	32 ch	32 ch	24 ch
Alarm Comparator	2 ch	2 ch	1 ch
Supply Supervisor	yes	yes	yes
Clock Supervisor	yes	yes	yes
Main clock oscillator	4 MHz	4 MHz	4 MHz
Sub clock oscillator	32 kHz	32 kHz	32 kHz
RC Oscillator	100 kHz	100 kHz / 2 MHz	100 kHz / 2 MHz
PLL	x 25	x 25	x 25
DSU4	yes	yes	no
EDSU	yes (32 BP)	yes (32 BP)	yes (16 BP)
JTAG Boundary Scan	no	tbd	no
Supply Voltage	3V / 5V	tbd	3V / 5V
Regulator	yes	tbd	yes
Power Consumption	n.a.	n.a.	< 1 W
Temperatur Range (Ta)	0..70 C	0..70 C	-40..105 C
Package	PGA/BGA	PGA/BGA	QFP208
Power on to PLL run	< 20 ms	< 20 ms	< 20 ms
Flash Download Time	n.a.	n.a.	< 30 sec (2M)
I^2S	no	yes	no
MediaLB (MOST interface)	no	yes	no
USB	no	yes	no
Flexray	no	yes	no

MB91V460 Rev.B : This series is presently being specified and not available yet.
1) The Memory protection Unit (MPU) is a part of the EDSU functionality

4. Block Diagram

The following illustration shows the block diagram of MB91460 series.

Figure 4-1 Block Diagram MB91460 Series



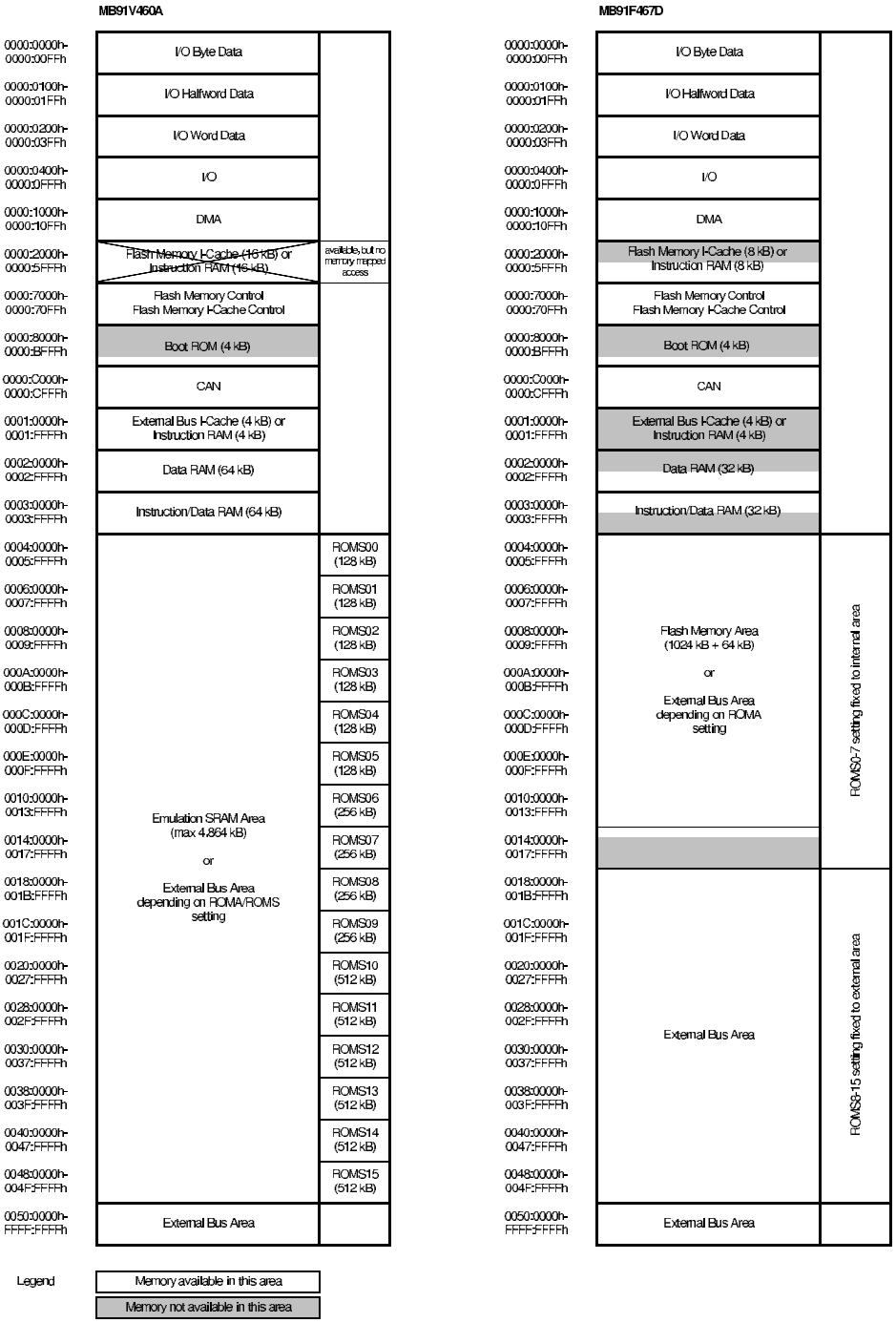
4.Block Diagram

Chapter 3 MB91460 Series Basic Information

This chapter describes MB91460 series basic information including Memory- and I/O map, interrupt vector table, pin function list, circuit type and pin state table for each device mode.

1. Memory Map

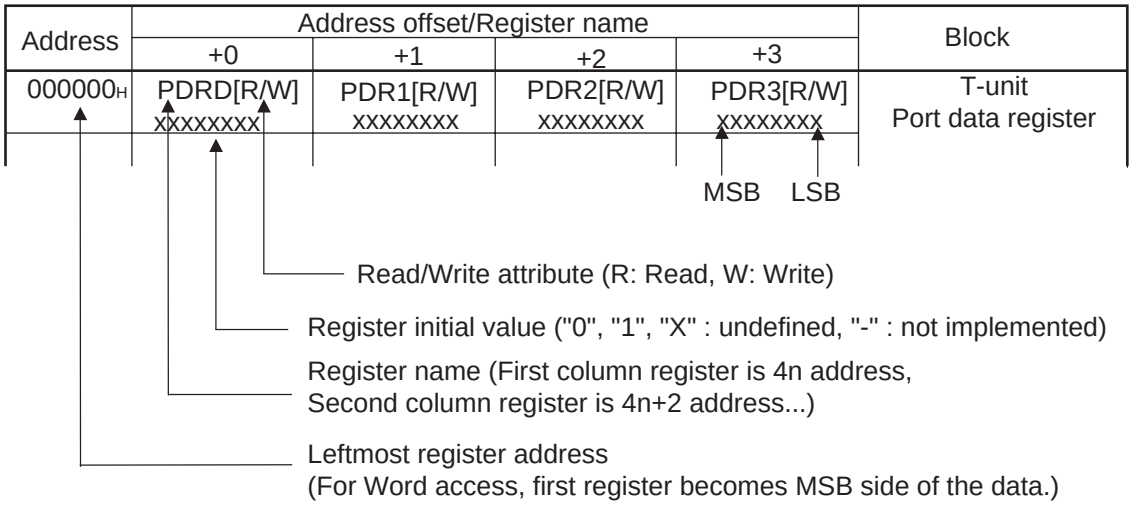
Figure 1-1 Memory Map



2. I/O Map

This section shows the association between memory space and each register of peripheral resources.

- Table convention



Note: Bit value of register shows initial values as follows.

- "1": Initial value is "1".
- "0": Initial value is "0".
- "X": Initial value is indeterminate.
- "N/A": No physical register exists in the position.

Do not use other data access attributes to access data.

Table 2-1 I/O Map

Address	Register				Block
	+0	+1	+2	+3	
000000_H	PDR00 [R/W] XXXXXXXX	PDR01 [R/W] XXXXXXXX	PDR02 [R/W] XXXXXXXX	PDR03 [R/W] XXXXXXXX	R-bus Port Data Register
000004_H	PDR04 [R/W] XXXXXXXX	PDR05 [R/W] XXXXXXXX	PDR06 [R/W] XXXXXXXX	PDR07 [R/W] XXXXXXXX	
000008_H	PDR08 [R/W] XXXXXXXX	PDR09 [R/W] XXXXXXXX	PDR10 [R/W] XXXXXXXX	PDR11 [R/W] XXXXXXXX	
$00000C_H$	PDR12 [R/W] XXXXXXXX	PDR13 [R/W] XXXXXXXX	PDR14 [R/W] XXXXXXXX	PDR15 [R/W] XXXXXXXX	
000010_H	PDR16 [R/W] XXXXXXXX	PDR17 [R/W] XXXXXXXX	PDR18 [R/W] XXXXXXXX	PDR19 [R/W] XXXXXXXX	
000014_H	PDR20 [R/W] XXXXXXXX	PDR21 [R/W] XXXXXXXX	PDR22 [R/W] XXXXXXXX	PDR23 [R/W] XXXXXXXX	
000018_H	PDR24 [R/W] XXXXXXXX	PDR25 [R/W] XXXXXXXX	PDR26 [R/W] XXXXXXXX	PDR27 [R/W] XXXXXXXX	
$00001C_H$	PDR28 [R/W] XXXXXXXX	PDR29 [R/W] XXXXXXXX	PDR30 [R/W] XXXXXXXX	PDR31 [R/W] XXXXXXXX	
000020_H	PDR32 [R/W] XXXXXXXX	PDR33 [R/W] XXXXXXXX	PDR34 [R/W] XXXXXXXX	PDR35 [R/W] XXXXXXXX	
000024_H - $00002C_H$	reserved				
000030_H	EIRR0 [R/W] 00000000	ENIR0 [R/W] 00000000	ELVR0 [R/W] 00000000 00000000		Ext. INT 0-7 NMI
000034_H	EIRR1 [R/W] 00000000	ENIR1 [R/W] 00000000	ELVR1 [R/W] 00000000 00000000		Ext. INT 8-15
000038_H	DICR [R/W] - - - - - - - 0	HRCL [R/W] 0 - - 11111	RBSYNC *1		DLYI/I-unit
$00003C_H$	reserved				
000040_H	SCR00 [R/W,W] 00000000	SMR00 [R/W,W] 00000000	SSR00 [R/W,R] 00001000	RDR00/TDR00 [R/W] 00000000	USART (LIN) 0
000044_H	ESCR00 [R/W] 00000X00	ECCR00 [R/W,R,W] 000000XX	res.		

Address	Register				Block
	+0	+1	+2	+3	
000048_H	SCR01 [R/W,W] 00000000	SMR01 [R/W,W] 00000000	SSR01 [R/W,R] 00001000	RDR01/TDR01 [R/W] 00000000	USART (LIN) 1
$00004C_H$	ESCR01 [R/W] 00000X00	ECCR01 [R/W,R,W] 000000XX	res.		
000050_H	SCR02 [R/W,W] 00000000	SMR02 [R/W,W] 00000000	SSR02 [R/W,R] 00001000	RDR02/TDR02 [R/W] 00000000	USART (LIN) 2
000054_H	ESCR02 [R/W] 00000X00	ECCR02 [R/W,R,W] 000000XX	res.		
000058_H	SCR03 [R/W,W] 00000000	SMR03 [R/W,W] 00000000	SSR03 [R/W,R] 00001000	RDR03/TDR03 [R/W] 00000000	USART (LIN) 3
$00005C_H$	ESCR03 [R/W] 00000X00	ECCR03 [R/W,R,W] 000000XX	res.		
000060_H	SCR04 [R/W,W] 00000000	SMR04 [R/W,W] 00000000	SSR04 [R/W,R] 00001000	RDR04/TDR04 [R/W] 00000000	USART (LIN) 4 with FIFO
000064_H	ESCR04 [R/W] 00000X00	ECCR04 [R/W,R,W] 000000XX	FSR04 [R] - - - 00000	FCR04 [R/W] 0001 - 000	
000068_H	SCR05 [R/W,W] 00000000	SMR05 [R/W,W] 00000000	SSR05 [R/W,R] 00001000	RDR05/TDR05 [R/W] 00000000	USART (LIN) 5 with FIFO
$00006C_H$	ESCR05 [R/W] 00000X00	ECCR05 [R/W,R,W] 000000XX	FSR05 [R] - - - 00000	FCR05 [R/W] 0001 - 000	
000070_H	SCR06 [R/W,W] 00000000	SMR06 [R/W,W] 00000000	SSR06 [R/W,R] 00001000	RDR06/TDR06 [R/W] 00000000	USART (LIN) 6 with FIFO
000074_H	ESCR06 [R/W] 00000X00	ECCR06 [R/W,R,W] 000000XX	FSR06 [R] - - - 00000	FCR06 [R/W] 0001 - 000	
000078_H	SCR07 [R/W,W] 00000000	SMR07 [R/W,W] 00000000	SSR07 [R/W,R] 00001000	RDR07/TDR07 [R/W] 00000000	USART (LIN) 7 with FIFO
$00007C_H$	ESCR07 [R/W] 00000X00	ECCR07 [R/W,R,W] 000000XX	FSR07 [R] - - - 00000	FCR07 [R/W] 0001 - 000	

Address	Register				Block
	+0	+1	+2	+3	
000080_H	BGR100 [R/W] 00000000	BGR000 [R/W] 00000000	BGR101 [R/W] 00000000	BGR001 [R/W] 00000000	Baudrate Generator USART (LIN) 0-7
000084_H	BGR102 [R/W] 00000000	BGR002 [R/W] 00000000	BGR103 [R/W] 00000000	BGR003 [R/W] 00000000	
000088_H	BGR104 [R/W] 00000000	BGR004 [R/W] 00000000	BGR105 [R/W] 00000000	BGR005 [R/W] 00000000	
$00008C_H$	BGR106 [R/W] 00000000	BGR006 [R/W] 00000000	BGR107 [R/W] 00000000	BGR007 [R/W] 00000000	
000090_H	PWC20 [R/W] - - - - - - XX XXXXXXXX		PWC10 [R/W] - - - - - - XX XXXXXXXX		Stepper Motor 0
000094_H	res.	res.	PWS20 [R/W] -0000000	PWS10 [R/W] --000000	
000098_H	PWC21 [R/W] - - - - - - XX XXXXXXXX		PWC11 [R/W] - - - - - - XX XXXXXXXX		Stepper Motor 1
$00009C_H$	res.	res.	PWS21 [R/W] -0000000	PWS11 [R/W] --000000	
$0000A0_H$	PWC22 [R/W] - - - - - - XX XXXXXXXX		PWC12 [R/W] - - - - - - XX XXXXXXXX		Stepper Motor 2
$0000A4_H$	res.	res.	PWS22 [R/W] -0000000	PWS12 [R/W] --000000	
$0000A8_H$	PWC23 [R/W] - - - - - - XX XXXXXXXX		PWC13 [R/W] - - - - - - XX XXXXXXXX		Stepper Motor 3
$0000AC_H$	res.	res.	PWS23 [R/W] -0000000	PWS13 [R/W] --000000	
$0000B0_H$	PWC24 [R/W] - - - - - - XX XXXXXXXX		PWC14 [R/W] - - - - - - XX XXXXXXXX		Stepper Motor 4
$0000B4_H$	res.	res.	PWS24 [R/W] -0000000	PWS14 [R/W] --000000	
$0000B8_H$	PWC25 [R/W] - - - - - - XX XXXXXXXX		PWC15 [R/W] - - - - - - XX XXXXXXXX		Stepper Motor 5
$0000BC_H$	res.	res.	PWS25 [R/W] -0000000	PWS15 [R/W] --000000	
$0000C0_H$	res.	PWC0 [R/W] -00000--	res.	PWC1 [R/W] -00000--	Stepper Motor Control 0-5
$0000C4_H$	res.	PWC2 [R/W] -00000--	res.	PWC3 [R/W] -00000--	
$0000C8_H$	res.	PWC4 [R/W] -00000--	res.	PWC5 [R/W] -00000--	
$0000CC_H$	reserved				

Address	Register				Block
	+0	+1	+2	+3	
$0000D0_H$	IBCR0 [R/W] 00000000	IBSR0 [R] 00000000	ITBAH0 [R/W] - - - - - - 00	ITBAL0 [R/W] 00000000	I2C 0
$0000D4_H$	ITMKH0 [R/W] 00 - - - - 11	ITMKL0 [R/W] 11111111	ISMK0 [R/W] 01111111	ISBA0 [R/W] - 0000000	
$0000D8_H$	res.	IDAR0 [R/W] 00000000	ICCR0 [R/W] - 0011111	res.	
$0000DC_H$	IBCR1 [R/W] 00000000	IBSR1 [R] 00000000	ITBAH1 [R/W] - - - - - - 00	ITBAL1 [R/W] 00000000	I2C 1
$0000E0_H$	ITMKH1 [R/W] 00 - - - - 11	ITMKL1 [R/W] 11111111	ISMK1 [R/W] 01111111	ISBA1 [R/W] - 0000000	
$0000E4_H$	res.	IDAR1 [R/W] 00000000	ICCR1 [R/W] - 0011111	res.	
$0000E8_H$	LCDCMR [R/W] - - - - 0000	LCR0 [R/W] 00010000	LCR1H [R/W] - - - - - - 00	LCR1L [R/W] 00000000	LCD Controller
$0000EC_H$	VRAM00 [R/W] XXXXXXXX	VRAM01 [R/W] XXXXXXXX	VRAM02 [R/W] XXXXXXXX	VRAM03 [R/W] XXXXXXXX	
$0000F0_H$	VRAM04 [R/W] XXXXXXXX	VRAM05 [R/W] XXXXXXXX	VRAM06 [R/W] XXXXXXXX	VRAM07 [R/W] XXXXXXXX	
$0000F4_H$	VRAM08 [R/W] XXXXXXXX	VRAM09 [R/W] XXXXXXXX	VRAM10 [R/W] XXXXXXXX	VRAM11 [R/W] XXXXXXXX	
$0000F8_H$	VRAM12 [R/W] XXXXXXXX	VRAM13 [R/W] XXXXXXXX	VRAM14 [R/W] XXXXXXXX	VRAM15 [R/W] XXXXXXXX	
$0000FC_H$	VRAM16 [R/W] XXXXXXXX	VRAM17 [R/W] XXXXXXXX	VRAM18 [R/W] XXXXXXXX	VRAM19 [R/W] XXXXXXXX	
000100_H	GCN10 [R/W] 00110010 00010000		res.	GCN20 [R/W] - - - - 0000	PPG Control 0-3
000104_H	GCN11 [R/W] 00110010 00010000		res.	GCN21 [R/W] - - - - 0000	PPG Control 4-7
000108_H	GCN12 [R/W] 00110010 00010000		res.	GCN22 [R/W] - - - - 0000	PPG Control 8-11
000110_H	PTMR00 [R] 11111111 11111111		PCSR00 [W] XXXXXXXX XXXXXXXX		PPG 0
000114_H	PDUT00 [W] XXXXXXXX XXXXXXXX		PCNH00 [R/W] 0000000 -	PCNL00 [R/W] 000000 - 0	
000118_H	PTMR01 [R] 11111111 11111111		PCSR01 [W] XXXXXXXX XXXXXXXX		PPG 1
$00011C_H$	PDUT01 [W] XXXXXXXX XXXXXXXX		PCNH01 [R/W] 0000000 -	PCNL01 [R/W] 000000 - 0	

Address	Register				Block
	+0	+1	+2	+3	
000120_H	PTMR02 [R] 11111111 11111111		PCSR02 [W] XXXXXXXX XXXXXXXX		PPG 2
000124_H	PDUT02 [W] XXXXXXXX XXXXXXXX		PCNH02 [R/W] 0000000 -	PCNL02 [R/W] 000000 - 0	
000128_H	PTMR03 [R] 11111111 11111111		PCSR03 [W] XXXXXXXX XXXXXXXX		PPG 3
$00012C_H$	PDUT03 [W] XXXXXXXX XXXXXXXX		PCNH03 [R/W] 0000000 -	PCNL03 [R/W] 000000 - 0	
000130_H	PTMR04 [R] 11111111 11111111		PCSR04 [W] XXXXXXXX XXXXXXXX		PPG 4
000134_H	PDUT04 [W] XXXXXXXX XXXXXXXX		PCNH04 [R/W] 0000000 -	PCNL04 [R/W] 000000 - 0	
000138_H	PTMR05 [R] 11111111 11111111		PCSR05 [W] XXXXXXXX XXXXXXXX		PPG 5
$00013C_H$	PDUT05 [W] XXXXXXXX XXXXXXXX		PCNH05 [R/W] 0000000 -	PCNL05 [R/W] 000000 - 0	
000140_H	PTMR06 [R] 11111111 11111111		PCSR06 [W] XXXXXXXX XXXXXXXX		PPG 6
000144_H	PDUT06 [W] XXXXXXXX XXXXXXXX		PCNH06 [R/W] 0000000 -	PCNL06 [R/W] 000000 - 0	
000148_H	PTMR07 [R] 11111111 11111111		PCSR07 [W] XXXXXXXX XXXXXXXX		PPG 7
$00014C_H$	PDUT07 [W] XXXXXXXX XXXXXXXX		PCNH07 [R/W] 0000000 -	PCNL07 [R/W] 000000 - 0	
000150_H	PTMR08 [R] 11111111 11111111		PCSR08 [W] XXXXXXXX XXXXXXXX		PPG 8
000154_H	PDUT08 [W] XXXXXXXX XXXXXXXX		PCNH08 [R/W] 0000000 -	PCNL08 [R/W] 000000 - 0	
000158_H	PTMR09 [R] 11111111 11111111		PCSR09 [W] XXXXXXXX XXXXXXXX		PPG 9
$00015C_H$	PDUT09 [W] XXXXXXXX XXXXXXXX		PCNH09 [R/W] 0000000 -	PCNL09 [R/W] 000000 - 0	
000160_H	PTMR10 [R] 11111111 11111111		PCSR10 [W] XXXXXXXX XXXXXXXX		PPG 10
000164_H	PDUT10 [W] XXXXXXXX XXXXXXXX		PCNH10 [R/W] 0000000 -	PCNL10 [R/W] 000000 - 0	
000168_H	PTMR11 [R] 11111111 11111111		PCSR11 [W] XXXXXXXX XXXXXXXX		PPG 11
$00016C_H$	PDUT11 [W] XXXXXXXX XXXXXXXX		PCNH11 [R/W] 0000000 -	PCNL11 [R/W] 000000 - 0	

Address	Register				Block
	+0	+1	+2	+3	
000170_H	P0TMCSRH [R/W] - 0 - 000 - 0	P0TMCSRL [R/W] - - - 00000	P1TMCSRH [R/W] - 0 - 000 - 0	P1TMCSRL [R/W] - - - 00000	Pulse Frequency Modulator
000174_H	P0TMRLR [W] XXXXXXXX XXXXXXXX		P0TMR [R] XXXXXXXX XXXXXXXX		
000178_H	P1TMRLR [W] XXXXXXXX XXXXXXXX		P1TMR [R] XXXXXXXX XXXXXXXX		
$00017C_H$	reserved				
000180_H	res.	ICS01 [R/W] 00000000	res.	ICS23 [R/W] 00000000	Input Capture 0-3
000184_H	IPCP0 [R] XXXXXXXX XXXXXXXX		IPCP1 [R] XXXXXXXX XXXXXXXX		
000188_H	IPCP2 [R] XXXXXXXX XXXXXXXX		IPCP3 [R] XXXXXXXX XXXXXXXX		
$00018C_H$	OCS01 [R/W] - - - 0 - - 00 0000 - - 00		OCS23 [R/W] - - - 0 - - 00 0000 - - 00		Output Compare 0-3
000190_H	OCCP0 [R/W] XXXXXXXX XXXXXXXX		OCCP1 [R/W] XXXXXXXX XXXXXXXX		
000194_H	OCCP2 [R/W] XXXXXXXX XXXXXXXX		OCCP3 [R/W] XXXXXXXX XXXXXXXX		
000198_H	SGCRH [R/W] 0000 - - 00	SGCRL [R/W] - - 0 - - 000	SGFR [R/W, R] XXXXXXXX XXXXXXXX		Sound Generator
$00019C_H$	SGAR [R/W] 00000000	res.	SGTR [R/W] XXXXXXXX	SGDR [R/W] XXXXXXXX	
$0001A0_H$	ADERH [R/W] 00000000 00000000		ADERL [R/W] 00000000 00000000		A/D Converter
0001A4	ADCS1 [R/W] 00000000	ADCS0 [R/W] 00000000	ADCR1 [R] 000000XX	ADCR0 [R] XXXXXXXX	
$0001A8_H$	ADCT1 [R/W] 00010000	ADCT0 [R/W] 00101100	ADSCH [R/W] - - - 00000	ADECH [R/W] - - - 00000	
$0001AC_H$	res.	ACSR0 [R/W] 011XXX00	res.	ACSR1 [R/W] 011XXX00	Alarm Comparator 0-1
$0001B0_H$	TMRLR0 [W] XXXXXXXX XXXXXXXX		TMR0 [R] XXXXXXXX XXXXXXXX		Reload Timer 0 (PPG 0-1)
$0001B4_H$	reserved		TMCSRH0 [R/W] - - - 00000	TMCSRL0 [R/W] 0 - 000000	

Address	Register				Block
	+0	+1	+2	+3	
$0001B8_H$	TMRLR1 [W] XXXXXXXX XXXXXXXX		TMR1 [R] XXXXXXXX XXXXXXXX		Reload Timer 1 (PPG 2-3)
$0001BC_H$	reserved		TMCSRH1 [R/W] - - - 00000	TMCSRL1 [R/W] 0 - 000000	
$0001C0_H$	TMRLR2 [W] XXXXXXXX XXXXXXXX		TMR2 [R] XXXXXXXX XXXXXXXX		Reload Timer 2 (PPG 4-5)
$0001C4_H$	reserved		TMCSRH2 [R/W] - - - 00000	TMCSRL2 [R/W] 0 - 000000	
$0001C8_H$	TMRLR3 [W] XXXXXXXX XXXXXXXX		TMR3 [R] XXXXXXXX XXXXXXXX		Reload Timer 3 (PPG 6-7)
$0001CC_H$	reserved		TMCSRH3 [R/W] - - - 00000	TMCSRL3 [R/W] 0 - 000000	
$0001D0_H$	TMRLR4 [W] XXXXXXXX XXXXXXXX		TMR4 [R] XXXXXXXX XXXXXXXX		Reload Timer 4 (PPG 8-9)
$0001D4_H$	reserved		TMCSRH4 [R/W] - - - 00000	TMCSRL4 [R/W] 0 - 000000	
$0001D8_H$	TMRLR5 [W] XXXXXXXX XXXXXXXX		TMR5 [R] XXXXXXXX XXXXXXXX		Reload Timer 5 (PPG 10-11)
$0001DC_H$	reserved		TMCSRH5 [R/W] - - - 00000	TMCSRL5 [R/W] 0 - 000000	
$0001E0_H$	TMRLR6 [W] XXXXXXXX XXXXXXXX		TMR6 [R] XXXXXXXX XXXXXXXX		Reload Timer 6 (PPG 12-13)
$0001E4_H$	reserved		TMCSRH6 [R/W] - - - 00000	TMCSRL6 [R/W] 0 - 000000	
$0001E8_H$	TMRLR7 [W] XXXXXXXX XXXXXXXX		TMR7 [R] XXXXXXXX XXXXXXXX		Reload Timer 7 (PPG 14-15) (ADC)
$0001EC_H$	reserved		TMCSRH7 [R/W] - - - 00000	TMCSRL7 [R/W] 0 - 000000	
$0001F0_H$	TCDT0 [R/W] XXXXXXXX XXXXXXXX		res.	TCCS0 [R/W] 00000000	Free Running Timer 0 (ICU 0-1)
$0001F4_H$	TCDT1 [R/W] XXXXXXXX XXXXXXXX		res.	TCCS1 [R/W] 00000000	Free Running Timer 1 (ICU 2-3)

Address	Register				Block
	+0	+1	+2	+3	
$0001F8_H$	TCDT2 [R/W] XXXXXXXX XXXXXXXX		res.	TCCS2 [R/W] 00000000	Free Running Timer 2 (OCU 0-1)
$0001FC_H$	TCDT3 [R/W] XXXXXXXX XXXXXXXX		res.	TCCS3 [R/W] 00000000	Free Running Timer 3 (OCU 2-3)
000200_H	DMACA0 [R/W] 00000000 0000XXXX XXXXXXXX XXXXXXXX				DMAC
000204_H	DMACB0 [R/W] 00000000 00000000 XXXXXXXX XXXXXXXX				
000208_H	DMACA1 [R/W] 00000000 0000XXXX XXXXXXXX XXXXXXXX				
$00020C_H$	DMACB1 [R/W] 00000000 00000000 XXXXXXXX XXXXXXXX				
000210_H	DMACA2 [R/W] 00000000 0000XXXX XXXXXXXX XXXXXXXX				
000214_H	DMACB2 [R/W] 00000000 00000000 XXXXXXXX XXXXXXXX				
000218_H	DMACA3 [R/W] 00000000 0000XXXX XXXXXXXX XXXXXXXX				
$00021C_H$	DMACB3 [R/W] 00000000 00000000 XXXXXXXX XXXXXXXX				
000220_H	DMACA4 [R/W] 00000000 0000XXXX XXXXXXXX XXXXXXXX				
000224_H	DMACB4 [R/W] 00000000 00000000 XXXXXXXX XXXXXXXX				
000228_H - $00023C_H$	reserved				
000240_H	DMACR [R/W] 0 - - 00000	reserved			
000244_H - $00024C_H$	reserved				
000250_H	DMATEST0 [R/W] XXXXXXXX 00000000 00000000 0000XXXX				DMA Test (do not use)
000254_H	DMATEST1 [R] XXXXXXXX XXXXX000 00000000 00000000				

Address	Register				Block
	+0	+1	+2	+3	
000258_H - $00027C_H$	reserved				
000280_H	SCR08 [R/W,W] 00000000	SMR08 [R/W,W] 00000000	SSR08 [R/W,R] 00001000	RDR08/TDR08 [R/W] 00000000	USART (LIN) 8
000284_H	ESCR08 [R/W] 00000X00	ECCR08 [R/W,R,W] 000000XX	res.		
000288_H	SCR09 [R/W,W] 00000000	SMR09 [R/W,W] 00000000	SSR09 [R/W,R] 00001000	RDR09/TDR09 [R/W] 00000000	USART (LIN) 9
$00028C_H$	ESCR09 [R/W] 00000X00	ECCR09 [R/W,R,W] 000000XX	res.		
000290_H	SCR10 [R/W,W] 00000000	SMR10 [R/W,W] 00000000	SSR10 [R/W,R] 00001000	RDR10/TDR10 [R/W] 00000000	USART (LIN) 10
000294_H	ESCR10 [R/W] 00000X00	ECCR10 [R/W,R,W] 000000XX	res.		
000298_H	SCR11 [R/W,W] 00000000	SMR11 [R/W,W] 00000000	SSR11 [R/W,R] 00001000	RDR11/TDR11 [R/W] 00000000	USART (LIN) 11
$00029C_H$	ESCR11 [R/W] 00000X00	ECCR11 [R/W,R,W] 000000XX	res.		
$0002A0_H$	SCR12 [R/W,W] 00000000	SMR12 [R/W,W] 00000000	SSR12 [R/W,R] 00001000	RDR12/TDR12 [R/W] 00000000	USART (LIN) 12
$0002A4_H$	ESCR12 [R/W] 00000X00	ECCR12 [R/W,R,W] 000000XX	res.		
$0002A8_H$	SCR13 [R/W,W] 00000000	SMR13 [R/W,W] 00000000	SSR13 [R/W,R] 00001000	RDR13/TDR13 [R/W] 00000000	USART (LIN) 13
$0002AC_H$	ESCR13 [R/W] 00000X00	ECCR13 [R/W,R,W] 000000XX	res.		

Address	Register				Block
	+0	+1	+2	+3	
$0002B0_H$	SCR14 [R/W,W] 00000000	SMR14 [R/W,W] 00000000	SSR14 [R/W,R] 00001000	RDR14/TDR14 [R/W] 00000000	USART (LIN) 14
$0002B4_H$	ESCR14 [R/W] 00000X00	ECCR14 [R/W,R,W] 000000XX	res.		
$0002B8_H$	SCR15 [R/W,W] 00000000	SMR15 [R/W,W] 00000000	SSR15 [R/W,R] 00001000	RDR15/TDR15 [R/W] 00000000	USART (LIN) 15
$0002BC_H$	ESCR15 [R/W] 00000X00	ECCR15 [R/W,R,W] 000000XX	res.		
$0002C0_H$	BGR108 [R/W] 00000000	BGR008 [R/W] 00000000	BGR109 [R/W] 00000000	BGR009 [R/W] 00000000	Baudrate Generator USART (LIN) 8-15
$0002C4_H$	BGR110 [R/W] 00000000	BGR010 [R/W] 00000000	BGR111 [R/W] 00000000	BGR011 [R/W] 00000000	
$0002C8_H$	BGR112 [R/W] 00000000	BGR012 [R/W] 00000000	BGR113 [R/W] 00000000	BGR013 [R/W] 00000000	
$0002CC_H$	BGR114 [R/W] 00000000	BGR014 [R/W] 00000000	BGR15 [R/W] 00000000	BGR015 [R/W] 00000000	
$0002D0_H$	res.	ICS45 [R/W] 00000000	res.	ICS67 [R/W] 00000000	Input Capture 4-7
$0002D4_H$	IPCP4 [R] XXXXXXXX XXXXXXXX		IPCP5 [R] XXXXXXXX XXXXXXXX		
$0002D8_H$	IPCP6 [R] XXXXXXXX XXXXXXXX		IPCP7 [R] XXXXXXXX XXXXXXXX		
$0002DC_H$	OCS45 [R/W] - - - 0 - - 00 0000 - - 00		OCS67 [R/W] - - - 0 - - 00 0000 - - 00		Output Compare 4-7
$0002E0_H$	OCCP4 [R/W] XXXXXXXX XXXXXXXX		OCCP5 [R/W] XXXXXXXX XXXXXXXX		
$0002E4_H$	OCCP6 [R/W] XXXXXXXX XXXXXXXX		OCCP7 [R/W] XXXXXXXX XXXXXXXX		
$0002E8_H$ - $0002EC_H$	reserved				
$0002F0_H$	TCDT4 [R/W] XXXXXXXX XXXXXXXX		res.	TCCS4 [R/W] 00000000	Free Running Timer 4 (ICU 4-5)

Address	Register				Block
	+0	+1	+2	+3	
$0002F4_H$	TCDT5 [R/W] XXXXXXXX XXXXXXXX		res.	TCCS5 [R/W] 00000000	Free Running Timer 5 (ICU 6-7)
$0002F8_H$	TCDT6 [R/W] XXXXXXXX XXXXXXXX		res.	TCCS6 [R/W] 00000000	Free Running Timer 6 (OCU 4-5)
$0002FC_H$	TCDT7 [R/W] XXXXXXXX XXXXXXXX		res.	TCCS7 [R/W] 00000000	Free Running Timer 7 (OCU 6-7)
000300_H	UDRC1 [W] 00000000	UDRC0 [W] 00000000	UDCR1 [R] 00000000	UDCR0 [R] 00000000	Up/Down Counter 0-1
000304_H	UDCCH0 [R/W] 00001000	UDCCL0 [R/W] 00000000	res.	UDCS0 [R/W] 00000000	
000308_H	UDCCH1 [R/W] 00001000	UDCCL1 [R/W] 00000000	res.	UDCS1 [R/W] 00000000	
$00030C_H$	reserved				
000310_H	UDRC3 [W] 00000000	UDRC2 [W] 00000000	UDCR3 [R] 00000000	UDCR2 [R] 00000000	Up/Down Counter 2-3
000314_H	UDCCH2 [R/W] 00001000	UDCCL2 [R/W] 00000000	res.	UDCS2 [R/W] 00000000	
000318_H	UDCCH3 [R/W] 00001000	UDCCL3 [R/W] 00000000	res.	UDCS3 [R/W] 00000000	
$00031C_H$	reserved				
000320_H	GCN13 [R/W] 00110010 00010000		res.	GCN23 [R/W] - - - - 0000	PPG Control 12-15
000324_H - $00032C_H$	reserved				
000330_H	PTMR12 [R] 11111111 11111111		PCSR12 [W] XXXXXXXX XXXXXXXX		PPG 12
000334_H	PDUT12 [W] XXXXXXXX XXXXXXXX		PCNH12 [R/W] 0000000 -	PCNL12 [R/W] 000000 - 0	
000338_H	PTMR13 [R] 11111111 11111111		PCSR13 [W] XXXXXXXX XXXXXXXX		PPG 13
$00033C_H$	PDUT13 [W] XXXXXXXX XXXXXXXX		PCNH13 [R/W] 0000000 -	PCNL13 [R/W] 000000 - 0	

Address	Register				Block
	+0	+1	+2	+3	
000340_H	PTMR14 [R] 11111111 11111111		PCSR14 [W] XXXXXXXX XXXXXXXX		PPG 14
000344_H	PDUT14 [W] XXXXXXXX XXXXXXXX		PCNH14 [R/W] 0000000 -	PCNL14 [R/W] 000000 - 0	
000348_H	PTMR15 [R] 11111111 11111111		PCSR15 [W] XXXXXXXX XXXXXXXX		PPG 15
$00034C_H$	PDUT15 [W] XXXXXXXX XXXXXXXX		PCNH15 [R/W] 0000000 -	PCNL15 [R/W] 000000 - 0	
000350_H - $00035C_H$	reserved				
000360_H	res.	DACR [R/W] - - - - - 000	res.	res.	D/A Converter
000364_H	DADR0 [R/W] - - - - - - XX XXXXXXXX		DADR1 [R/W] - - - - - - XX XXXXXXXX		
000368_H	IBCR2 [R/W] 00000000	IBSR2 [R] 00000000	ITBAH2 [R/W] - - - - - - 00	ITBAL2 [R/W] 00000000	I2C 2
$00036C_H$	ITMKH2 [R/W] 00 - - - - 11	ITMKL2 [R/W] 11111111	ISMK2 [R/W] 01111111	ISBA2 [R/W] - 0000000	
000370_H	res.	IDAR2 [R/W] 00000000	ICCR2 [R/W] - 0011111	res.	
000374H	IBCR3 [R/W] 00000000	IBSR3 [R] 00000000	ITBAH3 [R/W] - - - - - - 00	ITBAL3 [R/W] 00000000	I2C 3
000378_H	ITMKH3 [R/W] 00 - - - - 11	ITMKL3 [R/W] 11111111	ISMK3 [R/W] 01111111	ISBA3 [R/W] - 0000000	
$00037C_H$	res.	IDAR3 [R/W] 00000000	ICCR3 [R/W] - 0011111	res.	
000380_H - $00038C_H$	reserved				
000390_H	ROMS [R/W] XXXXXXXX XXXXXXXX		res.		ROM Select Register
000394_H - $0003BC_H$	reserved				
$0003C0_H$	reserved				I-Cache
$0003C4_H$	reserved			ISIZE [R/W] - - - - - - 10	

Address	Register				Block
	+0	+1	+2	+3	
$0003D8_H$ - $0003E0_H$	reserved				
$0003E4_H$	reserved			ICHCR [R/W] 0 - 000000	I-Cache
$0003E8_H$ - $0003EC_H$	reserved				
$0003F0_H$	BSD0 [W] XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX				Bit Search Module
$0003F4_H$	BSD1 [R/W] XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX				
$0003F8_H$	BSDC [W] XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX				
$0003FC_H$	BSRR [R] XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX				
000400_H - $00043C_H$	reserved				

Address	Register				Block
	+0	+1	+2	+3	
000440_H	ICR00 [R/W] ---11111	ICR01 [R/W] ---11111	ICR02 [R/W] ---11111	ICR03 [R/W] ---11111	Interrupt Control Unit
000444_H	ICR04 [R/W] ---11111	ICR05 [R/W] ---11111	ICR06 [R/W] ---11111	ICR07 [R/W] ---11111	
000448_H	ICR08 [R/W] ---11111	ICR09 [R/W] ---11111	ICR10 [R/W] ---11111	ICR11 [R/W] ---11111	
$00044C_H$	ICR12 [R/W] ---11111	ICR13 [R/W] ---11111	ICR14 [R/W] ---11111	ICR15 [R/W] ---11111	
000450_H	ICR16 [R/W] ---11111	ICR17 [R/W] ---11111	ICR18 [R/W] ---11111	ICR19 [R/W] ---11111	
000454_H	ICR20 [R/W] ---11111	ICR21 [R/W] ---11111	ICR22 [R/W] ---11111	ICR23 [R/W] ---11111	
000458_H	ICR24 [R/W] ---11111	ICR25 [R/W] ---11111	ICR26 [R/W] ---11111	ICR27 [R/W] ---11111	
$00045C_H$	ICR28 [R/W] ---11111	ICR29 [R/W] ---11111	ICR30 [R/W] ---11111	ICR31 [R/W] ---11111	
000460_H	ICR32 [R/W] ---11111	ICR33 [R/W] ---11111	ICR34 [R/W] ---11111	ICR35 [R/W] ---11111	
000464_H	ICR36 [R/W] ---11111	ICR37 [R/W] ---11111	ICR38 [R/W] ---11111	ICR39 [R/W] ---11111	
000468_H	ICR40 [R/W] ---11111	ICR41 [R/W] ---11111	ICR42 [R/W] ---11111	ICR43 [R/W] ---11111	
$00046C_H$	ICR44 [R/W] ---11111	ICR45 [R/W] ---11111	ICR46 [R/W] ---11111	ICR47 [R/W] ---11111	
000470_H	ICR48 [R/W] ---11111	ICR49 [R/W] ---11111	ICR50 [R/W] ---11111	ICR51 [R/W] ---11111	
000474_H	ICR52 [R/W] ---11111	ICR53 [R/W] ---11111	ICR54 [R/W] ---11111	ICR55 [R/W] ---11111	
000478_H	ICR56 [R/W] ---11111	ICR57 [R/W] ---11111	ICR58 [R/W] ---11111	ICR59 [R/W] ---11111	
$00047C_H$	ICR60 [R/W] ---11111	ICR61 [R/W] ---11111	ICR62 [R/W] ---11111	ICR63 [R/W] ---11111	
000480_H	RSRR [R/W] 10000000	STCR [R/W] 00110011	TBCR [R/W] 00XXXX00	CTBR [W] XXXXXXXX	Clock Control Unit
000484_H	CLKR [R/W] ----0000	WPR [W] XXXXXXXX	DIVR0 [R/W] 00000011	DIVR1 [R/W] 00000000	
000488_H	CTEST [R/W] XXXX00XX	res.	res.	res.	C-Unit Test (do not use)

Address	Register				Block
	+0	+1	+2	+3	
$00048C_H$	PLLDIVM [R/W] - - - - 0000	PLLDIVN [R/W] - - 000000	PLLDIVG [R/W] - - - - 0000	PLLMULG [W] 00000000	PLL Clock Gear Unit
000490_H	PLLCTRL [R/W] - - - - 0000	res.	res.	res.	
000494_H	OSCC1 [R/W] - - - - - 010	OSCS1 [R/W] 00001111	OSCC2 [R/W] - - - - - 010	OSCS2 [R/W] 00001111	Main/Sub Oscillator Control (do not use)
000498_H	PORTEN [R/W] - - - - - - 00	res.	res.	res.	Port Input Enable Control
$0004A0_H$	res.	WTCER [R/W] - - - - - - 00	WTCR [R/W] 00000000 000 - 00 - 0		Real Time Clock (Watch Timer)
$0004A4_H$	WTBR [R/W] - - - - - - - - - - - XXXXX XXXXXXXX XXXXXXXX				
$0004A8_H$	WTHR [R/W] - - - 00000	WTMR [R/W] - - 000000	WTSR [R/W] - - 000000	res.	
$0004AC_H$	CSVTR [R/W] (do not use)	CSVCR [R/W] 00011100	CSCFG [R/W] 0X000000	CMCFG [R/W] 00000000	Clock-Supervisor / Selector / Monitor
$0004B0_H$	CUCR [R/W] - - - - - - - - - - - 0 - - 00		CUTD [R/W] 10000000 00000000		Calibration Unit of Sub Oscillation
$0004B4_H$	CUTR1 [R] - - - - - - - - 00000000		CUTR2 [R] 00000000 00000000		
$0004B8_H$	CMPR [R/W] - - 000010 11111101		res.	CMCR [R/W] - 001 - - 00	Clock Modulation
$0004BC_H$	CMT1 [R/W] 00000000 1 - - - 0000		CMT2 [R/W] - - 000000 - - 000000		
$0004C0_H$	CANPRE [R/W] - - 000000	CANCKD [R/W] - - 000000	res.	res.	CAN Clock Control
$0004C4_H$	LVSEL [R/W] 00000111	LVDET [R/W] - 0000 - 00	HWWDE [R/W] - - - - - - 00	HWWD [R/W,W] 00011000	LV Detection / Hardware-Watchdog
$0004C8_H$	OSCRH [R/W] 000 - - 001	OSCRL [R/W] - - - - - 000	WPCRH [R/W] 000 - - 001	WPCRL [R/W] - - - - - - 00	Main-/Sub-Oscillation Stabilisation Timer
$0004CC_H$	OSCCR [R/W] - - - - - - 00	res.	REGSEL [R/W] - - 000110	REGCTR [R/W] - - - 0 - - 00	Main/Sub-Oscillation Standby Control / Main/Sub Regulator Control

Address	Register				Block
	+0	+1	+2	+3	
$0004D0_H$	C340R [R/W] - - - - - - - 0	res.	EISSRH [R/W] 00000000	EISSRL [R/W] 00000000	340 Compatibility Mode (do not use)
$0004D4_H$	SHDE [R/W] 0 - - - - - - -	res.	EXTE [R/W] 00000000	EXTF [R/W] 00000000	Supply Shut Down Mode
$0004D8_H$	EXTLV [R/W] 00000000 00000000		res.	res.	
$0004DC_H$ - $00063C_H$	reserved				

Address	Register				Block
	+0	+1	+2	+3	
000640_H	ASR0 [R/W] 00000000 00000000		ACR0 [R/W] 1111**00 00000000		External Bus Unit
000644_H	ASR1 [R/W] XXXXXXXX XXXXXXXX		ACR1 [R/W] XXXXXXXX XXXXXXXX		
000648_H	ASR2 [R/W] XXXXXXXX XXXXXXXX		ACR2 [R/W] XXXXXXXX XXXXXXXX		
$00064C_H$	ASR3 [R/W] XXXXXXXX XXXXXXXX		ACR3 [R/W] XXXXXXXX XXXXXXXX		
000650_H	ASR4 [R/W] XXXXXXXX XXXXXXXX		ACR4 [R/W] XXXXXXXX XXXXXXXX		
000654_H	ASR5 [R/W] XXXXXXXX XXXXXXXX		ACR5 [R/W] XXXXXXXX XXXXXXXX		
000658_H	ASR6 [R/W] XXXXXXXX XXXXXXXX		ACR6 [R/W] XXXXXXXX XXXXXXXX		
$00065C_H$	ASR7 [R/W] XXXXXXXX XXXXXXXX		ACR7 [R/W] XXXXXXXX XXXXXXXX		
000660_H	AWR0 [R/W] 01111111 11111*11		AWR1 [R/W] XXXXXXXX XXXXXXXX		
000664_H	AWR2 [R/W] XXXXXXXX XXXXXXXX		AWR3 [R/W] XXXXXXXX XXXXXXXX		
000668_H	AWR4 [R/W] XXXXXXXX XXXXXXXX		AWR5 [R/W] XXXXXXXX XXXXXXXX		
$00066C_H$	AWR6 [R/W] XXXXXXXX XXXXXXXX		AWR7 [R/W] XXXXXXXX XXXXXXXX		
000670_H	MCRA [R/W] XXXXXXXX	MCRB [R/W] XXXXXXXX	reserved		
000674_H	reserved				
000678_H	IOWR0 [R/W] XXXXXXXX	IOWR1 [R/W] XXXXXXXX	IOWR2 [R/W] XXXXXXXX	IOWR3 [R/W] XXXXXXXX	
$00067C_H$	reserved				
000680_H	CSER [R/W] 00000001	CHER [R/W] 11111111	res.	TCR [R/W] 0000****	
000684_H	RCRH [R/W] 00XXXXXX	RCRL [R/W] XXXX0XXX	reserved		
000688_H - $0007F8_H$	reserved				
ACR0[11:10] depends on Modevector fetch information on buswidth TCR[3:0] INIT value = 0000, keeps value after RST					

Address	Register				Block
	+0	+1	+2	+3	
$0007FC_H$	res.	MODR [W] XXXXXXXX	res.	res.	Mode Register
000800_H - $000BFC_H$	reserved				DSU4 / RTM
$000C00_H$	TVCTW [W] XXXXXXXX	TVCTR [R] - - XXXXXX	res.	IOS [R/W] - - - - - - - 0	I-Unit Test (do not use)
$000C04_H$ - $000CFC_H$	reserved				
$000D00_H$	PDRD00 [R] XXXXXXXX	PDRD01 [R] XXXXXXXX	PDRD02 [R] XXXXXXXX	PDRD03 [R] XXXXXXXX	R-bus Port Data Direct Read Register
$000D04_H$	PDRD04 [R] XXXXXXXX	PDRD05 [R] XXXXXXXX	PDRD06 [R] XXXXXXXX	PDRD07 [R] XXXXXXXX	
$000D08_H$	PDRD08 [R] XXXXXXXX	PDRD09 [R] XXXXXXXX	PDRD10 [R] XXXXXXXX	PDRD11 [R] XXXXXXXX	
$000D0C_H$	PDRD12 [R] XXXXXXXX	PDRD13 [R] XXXXXXXX	PDRD14 [R] XXXXXXXX	PDRD15 [R] XXXXXXXX	
$000D10_H$	PDRD16 [R] XXXXXXXX	PDRD17 [R] XXXXXXXX	PDRD18 [R] XXXXXXXX	PDRD19 [R] XXXXXXXX	
$000D14_H$	PDRD20 [R] XXXXXXXX	PDRD21 [R] XXXXXXXX	PDRD22 [R] XXXXXXXX	PDRD23 [R] XXXXXXXX	
$000D18_H$	PDRD24 [R] XXXXXXXX	PDRD25 [R] XXXXXXXX	PDRD26 [R] XXXXXXXX	PDRD27 [R] XXXXXXXX	
$000D1C_H$	PDRD28 [R] XXXXXXXX	PDRD29 [R] XXXXXXXX	PDRD30 [R] XXXXXXXX	PDRD31 [R] XXXXXXXX	
$000D20_H$	PDRD32 [R] XXXXXXXX	PDRD33 [R] XXXXXXXX	PDRD34 [R] XXXXXXXX	PDRD35 [R] XXXXXXXX	
$000D24_H$ - $000D3C_H$	reserved				

Address	Register				Block
	+0	+1	+2	+3	
$000D40_H$	DDR00 [R/W] 00000000	DDR01 [R/W] 00000000	DDR02 [R/W] 00000000	DDR03 [R/W] 00000000	R-bus Port Direction Register
$000D44_H$	DDR04 [R/W] 00000000	DDR05 [R/W] 00000000	DDR06 [R/W] 00000000	DDR07 [R/W] 00000000	
$000D48_H$	DDR08 [R/W] 00000000	DDR09 [R/W] 00000000	DDR10 [R/W] 00000000	DDR11 [R/W] 00000000	
$000D4C_H$	DDR12 [R/W] 00000000	DDR13 [R/W] 00000000	DDR14 [R/W] 00000000	DDR15 [R/W] 00000000	
$000D50_H$	DDR16 [R/W] 00000000	DDR17 [R/W] 00000000	DDR18 [R/W] 00000000	DDR19 [R/W] 00000000	
$000D54_H$	DDR20 [R/W] 00000000	DDR21 [R/W] 00000000	DDR22 [R/W] 00000000	DDR23 [R/W] 00000000	
$000D58_H$	DDR24 [R/W] 00000000	DDR25 [R/W] 00000000	DDR26 [R/W] 00000000	DDR27 [R/W] 00000000	
$000D5C_H$	DDR28 [R/W] 00000000	DDR29 [R/W] 00000000	DDR30 [R/W] 00000000	DDR31 [R/W] 00000000	
$000D60_H$	DDR32 [R/W] 00000000	DDR33 [R/W] 00000000	DDR34 [R/W] 00000000	DDR35 [R/W] 00000000	
$000D64_H$ - $000D7C_H$	reserved				
$000D80_H$	PFR00 [R/W] 11111111	PFR01 [R/W] 11111111	PFR02 [R/W] 11111111	PFR03 [R/W] 11111111	R-bus Port Function Register
$000D84_H$	PFR04 [R/W] 11111111	PFR05 [R/W] 11111111	PFR06 [R/W] 11111111	PFR07 [R/W] 11111111	
$000D88_H$	PFR08 [R/W] 11111111	PFR09 [R/W] 11111111	PFR10 [R/W] 11111111	PFR11 [R/W] 00000000	
$000D8C_H$	PFR12 [R/W] 00000000	PFR13 [R/W] 00000000	PFR14 [R/W] 00000000	PFR15 [R/W] 00000000	
$000D90_H$	PFR16 [R/W] 00000000	PFR17 [R/W] 00000000	PFR18 [R/W] 00000000	PFR19 [R/W] 00000000	
$000D94_H$	PFR20 [R/W] 00000000	PFR21 [R/W] 00000000	PFR22 [R/W] 00000000	PFR23 [R/W] 00000000	
$000D98_H$	PFR24 [R/W] 00000000	PFR25 [R/W] 00000000	PFR26 [R/W] 00000000	PFR27 [R/W] 00000000	
$000D9C_H$	PFR28 [R/W] 00000000	PFR29 [R/W] 00000000	PFR30 [R/W] 00000000	PFR31 [R/W] 00000000	
$000DA0_H$	PFR32 [R/W] 00000000	PFR33 [R/W] 00000000	PFR34 [R/W] 00000000	PFR35 [R/W] 00000000	

Address	Register				Block
	+0	+1	+2	+3	
$000DA4_H$ - $000DBC_H$	reserved				
$000DC0_H$	EPFR00 [R/W] 00000000	EPFR01 [R/W] 00000000	EPFR02 [R/W] 00000000	EPFR03 [R/W] 00000000	R-bus Port Extra Function Register
$000DC4_H$	EPFR04 [R/W] 00000000	EPFR05 [R/W] 00000000	EPFR06 [R/W] 00000000	EPFR07 [R/W] 00000000	
$000DC8_H$	EPFR08 [R/W] 00000000	EPFR09 [R/W] 00000000	EPFR10 [R/W] 00000000	EPFR11 [R/W] 00000000	
$000DCC_H$	EPFR12 [R/W] 00000000	EPFR13 [R/W] 00000000	EPFR14 [R/W] 00000000	EPFR15 [R/W] 00000000	
$000DD0_H$	EPFR16 [R/W] 00000000	EPFR17 [R/W] 00000000	EPFR18 [R/W] 00000000	EPFR19 [R/W] 00000000	
$000DD4_H$	EPFR20 [R/W] 00000000	EPFR21 [R/W] 00000000	EPFR22 [R/W] 00000000	EPFR23 [R/W] 00000000	
$000DD8_H$	EPFR24 [R/W] 00000000	EPFR25 [R/W] 00000000	EPFR26 [R/W] 00000000	EPFR27 [R/W] 00000000	
$000DDC_H$	EPFR28 [R/W] 00000000	EPFR29 [R/W] 00000000	EPFR30 [R/W] 00000000	EPFR31 [R/W] 00000000	
$000DE0_H$	EPFR32 [R/W] 00000000	EPFR33 [R/W] 00000000	EPFR34 [R/W] 00000000	EPFR35 [R/W] 00000000	
$000DE4_H$ - $000DFC_H$	reserved				

Address	Register				Block
	+0	+1	+2	+3	
$000E00_H$	PODR00 [R/W] 00000000	PODR01 [R/W] 00000000	PODR02 [R/W] 00000000	PODR03 [R/W] 00000000	R-bus Port Output Drive Select Register
$000E04_H$	PODR04 [R/W] 00000000	PODR05 [R/W] 00000000	PODR06 [R/W] 00000000	PODR07 [R/W] 00000000	
$000E08_H$	PODR08 [R/W] 00000000	PODR09 [R/W] 00000000	PODR10 [R/W] 00000000	PODR11 [R/W] 00000000	
$000E0C_H$	PODR12 [R/W] 00000000	PODR13 [R/W] 00000000	PODR14 [R/W] 00000000	PODR15 [R/W] 00000000	
$000E10_H$	PODR16 [R/W] 00000000	PODR17 [R/W] 00000000	PODR18 [R/W] 00000000	PODR19 [R/W] 00000000	
$000E14_H$	PODR20 [R/W] 00000000	PODR21 [R/W] 00000000	PODR22 [R/W] 00000000	PODR23 [R/W] 00000000	
$000E18_H$	PODR24 [R/W] 00000000	PODR25 [R/W] 00000000	PODR26 [R/W] 00000000	PODR27 [R/W] 00000000	
$000E1C_H$	PODR28 [R/W] 00000000	PODR29 [R/W] 00000000	PODR30 [R/W] 00000000	PODR31 [R/W] 00000000	
$000E20_H$	PODR32 [R/W] 00000000	PODR33 [R/W] 00000000	PODR34 [R/W] 00000000	PODR35 [R/W] 00000000	
$000E24_H$ - $000E3C_H$	reserved				
$000E40_H$	PILR00 [R/W] 00000000	PILR01 [R/W] 00000000	PILR02 [R/W] 00000000	PILR03 [R/W] 00000000	R-bus Port Input Level Select Register
$000E44_H$	PILR04 [R/W] 00000000	PILR05 [R/W] 00000000	PILR06 [R/W] 00000000	PILR07 [R/W] 00000000	
$000E48_H$	PILR08 [R/W] 00000000	PILR09 [R/W] 00000000	PILR10 [R/W] 00000000	PILR11 [R/W] 00000000	
$000E4C_H$	PILR12 [R/W] 00000000	PILR13 [R/W] 00000000	PILR14 [R/W] 00000000	PILR15 [R/W] 00000000	
$000E50_H$	PILR16 [R/W] 00000000	PILR17 [R/W] 00000000	PILR18 [R/W] 00000000	PILR19 [R/W] 00000000	
$000E54_H$	PILR20 [R/W] 00000000	PILR21 [R/W] 00000000	PILR22 [R/W] 00000000	PILR23 [R/W] 00000000	
$000E58_H$	PILR24 [R/W] 00000000	PILR25 [R/W] 00000000	PILR26 [R/W] 00000000	PILR27 [R/W] 00000000	
$000E5C_H$	PILR28 [R/W] 00000000	PILR29 [R/W] 00000000	PILR30 [R/W] 00000000	PILR31 [R/W] 00000000	
$000E60_H$	PILR32 [R/W] 00000000	PILR33 [R/W] 00000000	PILR34 [R/W] 00000000	PILR35 [R/W] 00000000	

Address	Register				Block
	+0	+1	+2	+3	
$000E64_H$ - $000E7C_H$	reserved				
$000E80_H$	EPILR00 [R/W] 00000000	EPILR01 [R/W] 00000000	EPILR02 [R/W] 00000000	EPILR03 [R/W] 00000000	R-bus Port Extra Input Level Select Register
$000E84_H$	EPILR04 [R/W] 00000000	EPILR05 [R/W] 00000000	EPILR06 [R/W] 00000000	EPILR07 [R/W] 00000000	
$000E88_H$	EPILR08 [R/W] 00000000	EPILR09 [R/W] 00000000	EPILR10 [R/W] 00000000	EPILR11 [R/W] 00000000	
$000E8C_H$	EPILR12 [R/W] 00000000	EPILR13 [R/W] 00000000	EPILR14 [R/W] 00000000	EPILR15 [R/W] 00000000	
$000E90_H$	EPILR16 [R/W] 00000000	EPILR17 [R/W] 00000000	EPILR18 [R/W] 00000000	EPILR19 [R/W] 00000000	
$000E94_H$	EPILR20 [R/W] 00000000	EPILR21 [R/W] 00000000	EPILR22 [R/W] 00000000	EPILR23 [R/W] 00000000	
$000E98_H$	EPILR24 [R/W] 00000000	EPILR25 [R/W] 00000000	EPILR26 [R/W] 00000000	EPILR27 [R/W] 00000000	
$000E9C_H$	EPILR28 [R/W] 00000000	EPILR29 [R/W] 00000000	EPILR30 [R/W] 00000000	EPILR31 [R/W] 00000000	
$000EA0_H$	EPILR32 [R/W] 00000000	EPILR33 [R/W] 00000000	EPILR34 [R/W] 00000000	EPILR35 [R/W] 00000000	
$000EA4_H$ - $000EBC_H$	reserved				

Address	Register				Block
	+0	+1	+2	+3	
$000EC0_H$	PPER00 [R/W] 00000000	PPER01 [R/W] 00000000	PPER02 [R/W] 00000000	PPER03 [R/W] 00000000	R-bus Port Pull-Up/Down Enable Register
$000EC4_H$	PPER04 [R/W] 00000000	PPER05 [R/W] 00000000	PPER06 [R/W] 00000000	PPER07 [R/W] 00000000	
$000EC8_H$	PPER08 [R/W] 00000000	PPER09 [R/W] 00000000	PPER10 [R/W] 00000000	PPER11 [R/W] 00000000	
$000ECC_H$	PPER12 [R/W] 00000000	PPER13 [R/W] 00000000	PPER14 [R/W] 00000000	PPER15 [R/W] 00000000	
$000ED0_H$	PPER16 [R/W] 00000000	PPER17 [R/W] 00000000	PPER18 [R/W] 00000000	PPER19 [R/W] 00000000	
$000ED4_H$	PPER20 [R/W] 00000000	PPER21 [R/W] 00000000	PPER22 [R/W] 00000000	PPER23 [R/W] 00000000	
$000ED8_H$	PPER24 [R/W] 00000000	PPER25 [R/W] 00000000	PPER26 [R/W] 00000000	PPER27 [R/W] 00000000	
$000EDC_H$	PPER28 [R/W] 00000000	PPER29 [R/W] 00000000	PPER30 [R/W] 00000000	PPER31 [R/W] 00000000	
$000EE0_H$	PPER32 [R/W] 00000000	PPER33 [R/W] 00000000	PPER34 [R/W] 00000000	PPER35 [R/W] 00000000	
$000EE4_H$ - $000EFC_H$	reserved				
$000F00_H$	PPCR00 [R/W] 11111111	PPCR01 [R/W] 11111111	PPCR02 [R/W] 11111111	PPCR03 [R/W] 11111111	R-bus Port Pull-Up/Down Control Register
$000F04_H$	PPCR04 [R/W] 11111111	PPCR05 [R/W] 11111111	PPCR06 [R/W] 11111111	PPCR07 [R/W] 11111111	
$000F08_H$	PPCR08 [R/W] 11111111	PPCR09 [R/W] 11111111	PPCR10 [R/W] 11111111	PPCR11 [R/W] 11111111	
$000F0C_H$	PPCR12 [R/W] 11111111	PPCR13 [R/W] 11111111	PPCR14 [R/W] 11111111	PPCR15 [R/W] 11111111	
$000F10_H$	PPCR16 [R/W] 11111111	PPCR17 [R/W] 11111111	PPCR18 [R/W] 11111111	PPCR19 [R/W] 11111111	
$000F14_H$	PPCR20 [R/W] 11111111	PPCR21 [R/W] 11111111	PPCR22 [R/W] 11111111	PPCR23 [R/W] 11111111	
$000F18_H$	PPCR24 [R/W] 11111111	PPCR25 [R/W] 11111111	PPCR26 [R/W] 11111111	PPCR27 [R/W] 11111111	
$000F1C_H$	PPCR28 [R/W] 11111111	PPCR29 [R/W] 11111111	PPCR30 [R/W] 11111111	PPCR31 [R/W] 11111111	
$000F20_H$	PPCR32 [R/W] 11111111	PPCR33 [R/W] 11111111	PPCR34 [R/W] 11111111	PPCR35 [R/W] 11111111	

Address	Register				Block
	+0	+1	+2	+3	
$000F24_H$ - $000F3C_H$	reserved				
001000_H	DMASA0 [R/W] XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX				DMAC
001004_H	DMADA0 [R/W] XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX				
001008_H	DMASA1 [R/W] XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX				
$00100C_H$	DMADA1 [R/W] XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX				
001010_H	DMASA2 [R/W] XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX				
001014_H	DMADA2 [R/W] XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX				
001018_H	DMASA3 [R/W] XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX				
$00101C_H$	DMADA3 [R/W] XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX				
001020_H	DMASA4 [R/W] XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX				
001024_H	DMADA4 [R/W] XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX				
001028_H - $006FFC_H$	reserved				
007000_H	FMCS [R/W] 01101000	res.	FCHCR [R/W] - - - - - - 00 10000011		Flash Memory/ I-Cache Control Register
007004_H	FMWT [R/W] 11111111 11111111		res.	FMPS [R/W] - - - - - 000	
007008_H	FMAC [R] 00000000 00000000 00000000 00000000				
$00700C_H$	FCHA0 [R/W] - - - - - - - - - 0000000 00000000 00000000				I-Cache Non-cacheable area setting Register
007010_H	FCHA1 [R/W] - - - - - - - - - 0000000 00000000 00000000				
007014_H - $007FFC_H$	reserved				

Address	Register				Block
	+0	+1	+2	+3	
008000_H - $00BFFC_H$	MB91V460 Boot-ROM size is 4kB : $00B000_H$ - $00BFFC_H$ (instruction access is 1 waitcycle, data access is 1 waitcycle)				Boot ROM 16 kB
$00C000_H$	CTRLR0 [R/W] 00000000 00000001		STATR0 [R/W] 00000000 00000000		CAN 0 Control Register
$00C004_H$	ERRCNT0 [R] 00000000 00000000		BTR0 [R/W] 00100011 00000001		
$00C008_H$	INTR0 [R] 00000000 00000000		TESTR0 [R/W] 00000000 X0000000		
$00C00C_H$	BRPER0 [R/W] 00000000 00000000		CBSYNC0 *2		
$00C010_H$	IF1CREQ0 [R/W] 00000000 00000001		IF1CMSK0 [R/W] 00000000 00000000		CAN 0 IF 1 Register
$00C014_H$	IF1MSK20 [R/W] 11111111 11111111		IF1MSK10 [R/W] 11111111 11111111		
$00C018_H$	IF1ARB20 [R/W] 00000000 00000000		IF1ARB10 [R/W] 00000000 00000000		
$00C01C_H$	IF1MCTR0 [R/W] 00000000 00000000		res.		
$00C020_H$	IF1DTA10 [R/W] 00000000 00000000		IF1DTA20 [R/W] 00000000 00000000		
$00C024_H$	IF1DTB10 [R/W] 00000000 00000000		IF1DTB20 [R/W] 00000000 00000000		
$00C028_H$ - $00C02C_H$	reserved				
$00C030_H$	*IF1DTA20 [R/W]* *00000000 00000000*		*IF1DTA10 [R/W]* *00000000 00000000*		
$00C034_H$	*IF1DTB20 [R/W]* *00000000 00000000*		*IF1DTB10 [R/W]* *00000000 00000000*		
$00C038_H$ - $00C03C_H$	reserved				

Address	Register				Block
	+0	+1	+2	+3	
$00C040_H$	IF2CREQ0 [R/W] 00000000 00000001		IF2CMSK0 [R/W] 00000000 00000000		CAN 0 IF 2 Register
$00C044_H$	IF2MSK20 [R/W] 11111111 11111111		IF2MSK10 [R/W] 11111111 11111111		
$00C048_H$	IF2ARB20 [R/W] 00000000 00000000		IF2ARB10 [R/W] 00000000 00000000		
$00C04C_H$	IF2MCTR0 [R/W] 00000000 00000000		res.		
$00C050_H$	IF2DTA10 [R/W] 00000000 00000000		IF2DTA20 [R/W] 00000000 00000000		
$00C054_H$	IF2DTB10 [R/W] 00000000 00000000		IF2DTB20 [R/W] 00000000 00000000		
$00C058_H$ - $00C05C_H$	reserved				
$00C060_H$	*IF2DTA20 [R/W]* *00000000 00000000*		*IF2DTA10 [R/W]* *00000000 00000000*		
$00C064_H$	*IF2DTB20 [R/W]* *00000000 00000000*		*IF2DTB10 [R/W]* *00000000 00000000*		
$00C068_H$ - $00C07C_H$	reserved				

Address	Register				Block
	+0	+1	+2	+3	
$00C080_H$	TREQR20 [R] 00000000 00000000		TREQR10 [R] 00000000 00000000		CAN 0 Status Flags
$00C084_H$	TREQR40 [R] 00000000 00000000		TREQR30 [R] 00000000 00000000		
$00C088_H$	TREQR60 [R] 00000000 00000000		TREQR50 [R] 00000000 00000000		
$00C08C_H$	TREQR80 [R] 00000000 00000000		TREQR70 [R] 00000000 00000000		
$00C090_H$	NEWDT20 [R] 00000000 00000000		NEWDT10 [R] 00000000 00000000		
$00C094_H$	NEWDT40 [R] 00000000 00000000		NEWDT30 [R] 00000000 00000000		
$00C098_H$	NEWDT60 [R] 00000000 00000000		NEWDT50 [R] 00000000 00000000		
$00C09C_H$	NEWDT80 [R] 00000000 00000000		NEWDT70 [R] 00000000 00000000		
$00C0A0_H$	INTPND20 [R] 00000000 00000000		INTPND10 [R] 00000000 00000000		
$00C0A4_H$	INTPND40 [R] 00000000 00000000		INTPND30 [R] 00000000 00000000		
$00C0A8_H$	INTPND60 [R] 00000000 00000000		INTPND50 [R] 00000000 00000000		
$00C0AC_H$	INTPND80 [R] 00000000 00000000		INTPND70 [R] 00000000 00000000		
$00C0B0_H$	MSGVAL20 [R] 00000000 00000000		MSGVAL10 [R] 00000000 00000000		
$00C0B4_H$	MSGVAL40 [R] 00000000 00000000		MSGVAL30 [R] 00000000 00000000		
$00C0B8_H$	MSGVAL60 [R] 00000000 00000000		MSGVAL50 [R] 00000000 00000000		
$00C0BC_H$	MSGVAL80 [R] 00000000 00000000		MSGVAL70 [R] 00000000 00000000		
$00C0C0_H$ - $00C0FC_H$	reserved				

Address	Register				Block
	+0	+1	+2	+3	
$00C100_H$	CTRLR1 [R/W] 00000000 00000001		STATR1 [R/W] 00000000 00000000		CAN 1 Control Register
$00C104_H$	ERRCNT1 [R] 00000000 00000000		BTR1 [R/W] 00100011 00000001		
$00C108_H$	INTR1 [R] 00000000 00000000		TESTR1 [R/W] 00000000 X0000000		
$00C10C_H$	BRPER1 [R/W] 00000000 00000000		CBSYNC1 *2		
$00C110_H$	IF1CREQ1 [R/W] 00000000 00000001		IF1CMSK1 [R/W] 00000000 00000000		CAN 1 IF 1 Register
$00C114_H$	IF1MSK21 [R/W] 11111111 11111111		IF1MSK11 [R/W] 11111111 11111111		
$00C118_H$	IF1ARB21 [R/W] 00000000 00000000		IF1ARB11 [R/W] 00000000 00000000		
$00C11C_H$	IF1MCTR1 [R/W] 00000000 00000000		res.		
$00C120_H$	IF1DTA11 [R/W] 00000000 00000000		IF1DTA21 [R/W] 00000000 00000000		
$00C124_H$	IF1DTB11 [R/W] 00000000 00000000		IF1DTB21 [R/W] 00000000 00000000		
$00C128_H$ - $00C12C_H$	reserved				
$00C130_H$	*IF1DTA21 [R/W]* *00000000 00000000*		*IF1DTA11 [R/W]* *00000000 00000000*		
$00C134_H$	*IF1DTB21 [R/W]* *00000000 00000000*		*IF1DTB11 [R/W]* *00000000 00000000*		
$00C138_H$ - $00C13C_H$	reserved				

Address	Register				Block
	+0	+1	+2	+3	
$00C140_H$	IF2CREQ1 [R/W] 00000000 00000001		IF2CMSK1 [R/W] 00000000 00000000		CAN 1 IF 2 Register
$00C144_H$	IF2MSK21 [R/W] 11111111 11111111		IF2MSK11 [R/W] 11111111 11111111		
$00C148_H$	IF2ARB21 [R/W] 00000000 00000000		IF2ARB11 [R/W] 00000000 00000000		
$00C14C_H$	IF2MCTR1 [R/W] 00000000 00000000		res.		
$00C150_H$	IF2DTA11 [R/W] 00000000 00000000		IF2DTA21 [R/W] 00000000 00000000		
$00C154_H$	IF2DTB11 [R/W] 00000000 00000000		IF2DTB21 [R/W] 00000000 00000000		
$00C158_H$ - $00C15C_H$	reserved				
$00C160_H$	*IF2DTA21 [R/W]* *00000000 00000000*		*IF2DTA11 [R/W]* *00000000 00000000*		
$00C164_H$	*IF2DTB21 [R/W]* *00000000 00000000*		*IF2DTB11 [R/W]* *00000000 00000000*		
$00C168_H$ - $00C17C_H$	reserved				

Address	Register				Block
	+0	+1	+2	+3	
$00C180_H$	TREQR21 [R] 00000000 00000000		TREQR11 [R] 00000000 00000000		CAN 1 Status Flags
$00C184_H$	TREQR41 [R] 00000000 00000000		TREQR31 [R] 00000000 00000000		
$00C188_H$	TREQR61 [R] 00000000 00000000		TREQR51 [R] 00000000 00000000		
$00C18C_H$	TREQR81 [R] 00000000 00000000		TREQR71 [R] 00000000 00000000		
$00C190_H$	NEWDT21 [R] 00000000 00000000		NEWDT11 [R] 00000000 00000000		
$00C194_H$	NEWDT41 [R] 00000000 00000000		NEWDT31 [R] 00000000 00000000		
$00C198_H$	NEWDT61 [R] 00000000 00000000		NEWDT51 [R] 00000000 00000000		
$00C19C_H$	NEWDT81 [R] 00000000 00000000		NEWDT71 [R] 00000000 00000000		
$00C1A0_H$	INTPND21 [R] 00000000 00000000		INTPND11 [R] 00000000 00000000		
$00C1A4_H$	INTPND41 [R] 00000000 00000000		INTPND31 [R] 00000000 00000000		
$00C1A8_H$	INTPND61 [R] 00000000 00000000		INTPND51 [R] 00000000 00000000		
$00C1AC_H$	INTPND81 [R] 00000000 00000000		INTPND71 [R] 00000000 00000000		
$00C1B0_H$	MSGVAL21 [R] 00000000 00000000		MSGVAL11 [R] 00000000 00000000		
$00C1B4_H$	MSGVAL41 [R] 00000000 00000000		MSGVAL31 [R] 00000000 00000000		
$00C1B8_H$	MSGVAL61 [R] 00000000 00000000		MSGVAL51 [R] 00000000 00000000		
$00C1BC_H$	MSGVAL81 [R] 00000000 00000000		MSGVAL71 [R] 00000000 00000000		
$00C1C0_H$ - $00C1FC_H$	reserved				

Address	Register				Block
	+0	+1	+2	+3	
$00C200_H$	CTRLR2 [R/W] 00000000 00000001		STATR2 [R/W] 00000000 00000000		CAN 2 Control Register
$00C204_H$	ERRCNT2 [R] 00000000 00000000		BTR2 [R/W] 00100011 00000001		
$00C208_H$	INTR2 [R] 00000000 00000000		TESTR2 [R/W] 00000000 X0000000		
$00C20C_H$	BRPER2 [R/W] 00000000 00000000		CBSYNC2 *2		
$00C210_H$	IF1CREQ2 [R/W] 00000000 00000001		IF1CMSK2 [R/W] 00000000 00000000		CAN 2 IF 1 Register
$00C214_H$	IF1MSK22 [R/W] 11111111 11111111		IF1MSK12 [R/W] 11111111 11111111		
$00C218_H$	IF1ARB22 [R/W] 00000000 00000000		IF1ARB12 [R/W] 00000000 00000000		
$00C21C_H$	IF1MCTR2 [R/W] 00000000 00000000		res.		
$00C220_H$	IF1DTA12 [R/W] 00000000 00000000		IF1DTA22 [R/W] 00000000 00000000		
$00C224_H$	IF1DTB12 [R/W] 00000000 00000000		IF1DTB22 [R/W] 00000000 00000000		
$00C228_H$ - $00C22C_H$	reserved				
$00C230_H$	*IF1DTA22 [R/W]* *00000000 00000000*		*IF1DTA12 [R/W]* *00000000 00000000*		
$00C234_H$	*IF1DTB22 [R/W]* *00000000 00000000*		*IF1DTB12 [R/W]* *00000000 00000000*		
$00C238_H$ - $00C23C_H$	reserved				

Address	Register				Block
	+0	+1	+2	+3	
$00C240_H$	IF2CREQ2 [R/W] 00000000 00000001		IF2CMSK2 [R/W] 00000000 00000000		CAN 2 IF 2 Register
$00C244_H$	IF2MSK22 [R/W] 11111111 11111111		IF2MSK12 [R/W] 11111111 11111111		
$00C248_H$	IF2ARB22 [R/W] 00000000 00000000		IF2ARB12 [R/W] 00000000 00000000		
$00C24C_H$	IF2MCTR2 [R/W] 00000000 00000000		res.		
$00C250_H$	IF2DTA12 [R/W] 00000000 00000000		IF2DTA22 [R/W] 00000000 00000000		
$00C254_H$	IF2DTB12 [R/W] 00000000 00000000		IF2DTB22 [R/W] 00000000 00000000		
$00C258_H$ - $00C25C_H$	reserved				
$00C260_H$	*IF2DTA22 [R/W]* *00000000 00000000*		*IF2DTA12 [R/W]* *00000000 00000000*		
$00C264_H$	*IF2DTB22 [R/W]* *00000000 00000000*		*IF2DTB12 [R/W]* *00000000 00000000*		
$00C268_H$ - $00C27C_H$	reserved				

Address	Register				Block
	+0	+1	+2	+3	
$00C280_H$	TREQR22 [R] 00000000 00000000		TREQR12 [R] 00000000 00000000		CAN 2 Status Flags
$00C284_H$	TREQR42 [R] 00000000 00000000		TREQR32 [R] 00000000 00000000		
$00C288_H$	TREQR62 [R] 00000000 00000000		TREQR52 [R] 00000000 00000000		
$00C28C_H$	TREQR82 [R] 00000000 00000000		TREQR72 [R] 00000000 00000000		
$00C290_H$	NEWDT22 [R] 00000000 00000000		NEWDT12 [R] 00000000 00000000		
$00C294_H$	NEWDT42 [R] 00000000 00000000		NEWDT32 [R] 00000000 00000000		
$00C298_H$	NEWDT62 [R] 00000000 00000000		NEWDT52 [R] 00000000 00000000		
$00C29C_H$	NEWDT82 [R] 00000000 00000000		NEWDT72 [R] 00000000 00000000		
$00C2A0_H$	INTPND22 [R] 00000000 00000000		INTPND12 [R] 00000000 00000000		
$00C2A4_H$	INTPND42 [R] 00000000 00000000		INTPND32 [R] 00000000 00000000		
$00C2A8_H$	INTPND62 [R] 00000000 00000000		INTPND52 [R] 00000000 00000000		
$00C2AC_H$	INTPND82 [R] 00000000 00000000		INTPND72 [R] 00000000 00000000		
$00C2B0_H$	MSGVAL22 [R] 00000000 00000000		MSGVAL12 [R] 00000000 00000000		
$00C2B4_H$	MSGVAL42 [R] 00000000 00000000		MSGVAL32 [R] 00000000 00000000		
$00C2B8_H$	MSGVAL62 [R] 00000000 00000000		MSGVAL52 [R] 00000000 00000000		
$00C2BC_H$	MSGVAL82 [R] 00000000 00000000		MSGVAL72 [R] 00000000 00000000		
$00C2C0_H$ - $00C2FC_H$	reserved				

Address	Register				Block
	+0	+1	+2	+3	
$00C300_H$	CTRLR3 [R/W] 00000000 00000001		STATR3 [R/W] 00000000 00000000		CAN 3 Control Register
$00C304_H$	ERRCNT3 [R] 00000000 00000000		BTR3 [R/W] 00100011 00000001		
$00C308_H$	INTR3 [R] 00000000 00000000		TESTR3 [R/W] 00000000 X0000000		
$00C30C_H$	BRPER3 [R/W] 00000000 00000000		CBSYNC3 *2		
$00C310_H$	IF1CREQ3 [R/W] 00000000 00000001		IF1CMSK3 [R/W] 00000000 00000000		CAN 3 IF 1 Register
$00C314_H$	IF1MSK23 [R/W] 11111111 11111111		IF1MSK13 [R/W] 11111111 11111111		
$00C318_H$	IF1ARB23 [R/W] 00000000 00000000		IF1ARB13 [R/W] 00000000 00000000		
$00C31C_H$	IF1MCTR3 [R/W] 00000000 00000000		res.		
$00C320_H$	IF1DTA13 [R/W] 00000000 00000000		IF1DTA23 [R/W] 00000000 00000000		
$00C324_H$	IF1DTB13 [R/W] 00000000 00000000		IF1DTB23 [R/W] 00000000 00000000		
$00C328_H$ - $00C32C_H$	reserved				
$00C330_H$	*IF1DTA23 [R/W]* *00000000 00000000*		*IF1DTA13 [R/W]* *00000000 00000000*		
$00C334_H$	*IF1DTB23 [R/W]* *00000000 00000000*		*IF1DTB13 [R/W]* *00000000 00000000*		
$00C338_H$ - $00C33C_H$	reserved				

Address	Register				Block
	+0	+1	+2	+3	
$00C340_H$	IF2CREQ3 [R/W] 00000000 00000001		IF2CMSK3 [R/W] 00000000 00000000		CAN 3 IF 2 Register
$00C344_H$	IF2MSK23 [R/W] 11111111 11111111		IF2MSK13 [R/W] 11111111 11111111		
$00C348_H$	IF2ARB23 [R/W] 00000000 00000000		IF2ARB13 [R/W] 00000000 00000000		
$00C34C_H$	IF2MCTR3 [R/W] 00000000 00000000		res.		
$00C350_H$	IF2DTA13 [R/W] 00000000 00000000		IF2DTA23 [R/W] 00000000 00000000		
$00C354_H$	IF2DTB13 [R/W] 00000000 00000000		IF2DTB23 [R/W] 00000000 00000000		
$00C358_H$ - $00C35C_H$	reserved				
$00C360_H$	*IF2DTA23 [R/W]* *00000000 00000000*		*IF2DTA13 [R/W]* *00000000 00000000*		
$00C364_H$	*IF2DTB23 [R/W]* *00000000 00000000*		*IF2DTB13 [R/W]* *00000000 00000000*		
$00C368_H$ - $00C37C_H$	reserved				

Address	Register				Block
	+0	+1	+2	+3	
$00C380_H$	TREQR23 [R] 00000000 00000000		TREQR13 [R] 00000000 00000000		CAN 3 Status Flags
$00C384_H$	TREQR43 [R] 00000000 00000000		TREQR33 [R] 00000000 00000000		
$00C388_H$	TREQR63 [R] 00000000 00000000		TREQR53 [R] 00000000 00000000		
$00C38C_H$	TREQR83 [R] 00000000 00000000		TREQR73 [R] 00000000 00000000		
$00C390_H$	NEWDT23 [R] 00000000 00000000		NEWDT13 [R] 00000000 00000000		
$00C394_H$	NEWDT43 [R] 00000000 00000000		NEWDT33 [R] 00000000 00000000		
$00C398_H$	NEWDT63 [R] 00000000 00000000		NEWDT53 [R] 00000000 00000000		
$00C39C_H$	NEWDT83 [R] 00000000 00000000		NEWDT73 [R] 00000000 00000000		
$00C3A0_H$	INTPND23 [R] 00000000 00000000		INTPND13 [R] 00000000 00000000		
$00C3A4_H$	INTPND43 [R] 00000000 00000000		INTPND33 [R] 00000000 00000000		
$00C3A8_H$	INTPND63 [R] 00000000 00000000		INTPND53 [R] 00000000 00000000		
$00C3AC_H$	INTPND83 [R] 00000000 00000000		INTPND73 [R] 00000000 00000000		
$00C3B0_H$	MSGVAL23 [R] 00000000 00000000		MSGVAL13 [R] 00000000 00000000		
$00C3B4_H$	MSGVAL43 [R] 00000000 00000000		MSGVAL33 [R] 00000000 00000000		
$00C3B8_H$	MSGVAL63 [R] 00000000 00000000		MSGVAL53 [R] 00000000 00000000		
$00C3BC_H$	MSGVAL83 [R] 00000000 00000000		MSGVAL73 [R] 00000000 00000000		
$00C3C0_H$ - $00C3FC_H$	reserved				

Address	Register				Block
	+0	+1	+2	+3	
$00C400_H$	CTRLR4 [R/W] 00000000 00000001		STATR4 [R/W] 00000000 00000000		CAN 4 Control Register
$00C404_H$	ERRCNT4 [R] 00000000 00000000		BTR4 [R/W] 00100011 00000001		
$00C408_H$	INTR4 [R] 00000000 00000000		TESTR4 [R/W] 00000000 X0000000		
$00C40C_H$	BRPER4 [R/W] 00000000 00000000		CBSYNC4 *2		
$00C410_H$	IF1CREQ4 [R/W] 00000000 00000001		IF1CMSK4 [R/W] 00000000 00000000		CAN 4 IF 1 Register
$00C414_H$	IF1MSK24 [R/W] 11111111 11111111		IF1MSK14 [R/W] 11111111 11111111		
$00C418_H$	IF1ARB24 [R/W] 00000000 00000000		IF1ARB14 [R/W] 00000000 00000000		
$00C41C_H$	IF1MCTR4 [R/W] 00000000 00000000		res.		
$00C420_H$	IF1DTA14 [R/W] 00000000 00000000		IF1DTA24 [R/W] 00000000 00000000		
$00C424_H$	IF1DTB14 [R/W] 00000000 00000000		IF1DTB24 [R/W] 00000000 00000000		
$00C428_H$ - $00C42C_H$	reserved				
$00C430_H$	*IF1DTA24 [R/W]* *00000000 00000000*		*IF1DTA14 [R/W]* *00000000 00000000*		
$00C434_H$	*IF1DTB24 [R/W]* *00000000 00000000*		*IF1DTB14 [R/W]* *00000000 00000000*		
$00C438_H$ - $00C43C_H$	reserved				

Address	Register				Block
	+0	+1	+2	+3	
$00C440_H$	IF2CREQ4 [R/W] 00000000 00000001		IF2CMSK4 [R/W] 00000000 00000000		CAN 4 IF 2 Register
$00C444_H$	IF2MSK24 [R/W] 11111111 11111111		IF2MSK14 [R/W] 11111111 11111111		
$00C448_H$	IF2ARB24 [R/W] 00000000 00000000		IF2ARB14 [R/W] 00000000 00000000		
$00C44C_H$	IF2MCTR4 [R/W] 00000000 00000000		res.		
$00C450_H$	IF2DTA14 [R/W] 00000000 00000000		IF2DTA24 [R/W] 00000000 00000000		
$00C454_H$	IF2DTB14 [R/W] 00000000 00000000		IF2DTB24 [R/W] 00000000 00000000		
$00C458_H$ - $00C45C_H$	reserved				
$00C460_H$	*IF2DTA24 [R/W]* *00000000 00000000*		*IF2DTA14 [R/W]* *00000000 00000000*		
$00C464_H$	*IF2DTB24 [R/W]* *00000000 00000000*		*IF2DTB14 [R/W]* *00000000 00000000*		
$00C468_H$ - $00C47C_H$	reserved				

Address	Register				Block
	+0	+1	+2	+3	
$00C480_H$	TREQR24 [R] 00000000 00000000		TREQR14 [R] 00000000 00000000		CAN 4 Status Flags
$00C484_H$	TREQR44 [R] 00000000 00000000		TREQR34 [R] 00000000 00000000		
$00C488_H$	TREQR64 [R] 00000000 00000000		TREQR54 [R] 00000000 00000000		
$00C48C_H$	TREQR84 [R] 00000000 00000000		TREQR74 [R] 00000000 00000000		
$00C490_H$	NEWDT24 [R] 00000000 00000000		NEWDT14 [R] 00000000 00000000		
$00C494_H$	NEWDT44 [R] 00000000 00000000		NEWDT34 [R] 00000000 00000000		
$00C498_H$	NEWDT64 [R] 00000000 00000000		NEWDT54 [R] 00000000 00000000		
$00C49C_H$	NEWDT84 [R] 00000000 00000000		NEWDT74 [R] 00000000 00000000		
$00C4A0_H$	INTPND24 [R] 00000000 00000000		INTPND14 [R] 00000000 00000000		
$00C4A4_H$	INTPND44 [R] 00000000 00000000		INTPND34 [R] 00000000 00000000		
$00C4A8_H$	INTPND64 [R] 00000000 00000000		INTPND54 [R] 00000000 00000000		
$00C4AC_H$	INTPND84 [R] 00000000 00000000		INTPND74 [R] 00000000 00000000		
$00C4B0_H$	MSGVAL24 [R] 00000000 00000000		MSGVAL14 [R] 00000000 00000000		
$00C4B4_H$	MSGVAL44 [R] 00000000 00000000		MSGVAL34 [R] 00000000 00000000		
$00C4B8_H$	MSGVAL64 [R] 00000000 00000000		MSGVAL54 [R] 00000000 00000000		
$00C4BC_H$	MSGVAL84 [R] 00000000 00000000		MSGVAL74 [R] 00000000 00000000		
$00C4C0_H$ - $00C4FC_H$	reserved				

Address	Register				Block
	+0	+1	+2	+3	
$00C500_H$	CTRLR5 [R/W] 00000000 00000001		STATR5 [R/W] 00000000 00000000		CAN 5 Control Register
$00C504_H$	ERRCNT5 [R] 00000000 00000000		BTR5 [R/W] 00100011 00000001		
$00C508_H$	INTR5 [R] 00000000 00000000		TESTR5 [R/W] 00000000 X0000000		
$00C50C_H$	BRPER5 [R/W] 00000000 00000000		CBSYNC5 *2		
$00C510_H$	IF1CREQ5 [R/W] 00000000 00000001		IF1CMSK5 [R/W] 00000000 00000000		CAN 5 IF 1 Register
$00C514_H$	IF1MSK25 [R/W] 11111111 11111111		IF1MSK15 [R/W] 11111111 11111111		
$00C518_H$	IF1ARB25 [R/W] 00000000 00000000		IF1ARB15 [R/W] 00000000 00000000		
$00C51C_H$	IF1MCTR5 [R/W] 00000000 00000000		res.		
$00C520_H$	IF1DTA15 [R/W] 00000000 00000000		IF1DTA25 [R/W] 00000000 00000000		
$00C524_H$	IF1DTB15 [R/W] 00000000 00000000		IF1DTB25 [R/W] 00000000 00000000		
$00C528_H$ - $00C52C_H$	reserved				
$00C530_H$	*IF1DTA25 [R/W]* *00000000 00000000*		*IF1DTA15 [R/W]* *00000000 00000000*		
$00C534_H$	*IF1DTB25 [R/W]* *00000000 00000000*		*IF1DTB15 [R/W]* *00000000 00000000*		
$00C538_H$ - $00C53C_H$	reserved				

Address	Register				Block
	+0	+1	+2	+3	
$00C540_H$	IF2CREQ5 [R/W] 00000000 00000001		IF2CMSK5 [R/W] 00000000 00000000		CAN 5 IF 2 Register
$00C544_H$	IF2MSK25 [R/W] 11111111 11111111		IF2MSK15 [R/W] 11111111 11111111		
$00C548_H$	IF2ARB25 [R/W] 00000000 00000000		IF2ARB15 [R/W] 00000000 00000000		
$00C54C_H$	IF2MCTR5 [R/W] 00000000 00000000		res.		
$00C550_H$	IF2DTA15 [R/W] 00000000 00000000		IF2DTA25 [R/W] 00000000 00000000		
$00C554_H$	IF2DTB15 [R/W] 00000000 00000000		IF2DTB25 [R/W] 00000000 00000000		
$00C558_H$ - $00C55C_H$	reserved				
$00C560_H$	*IF2DTA25 [R/W]* *00000000 00000000*		*IF2DTA15 [R/W]* *00000000 00000000*		
$00C564_H$	*IF2DTB25 [R/W]* *00000000 00000000*		*IF2DTB15 [R/W]* *00000000 00000000*		
$00C568_H$ - $00C57C_H$	reserved				

Address	Register				Block
	+0	+1	+2	+3	
$00C580_H$	TREQR25 [R] 00000000 00000000		TREQR15 [R] 00000000 00000000		CAN 5 Status Flags
$00C584_H$	TREQR45 [R] 00000000 00000000		TREQR35 [R] 00000000 00000000		
$00C588_H$	TREQR65 [R] 00000000 00000000		TREQR55 [R] 00000000 00000000		
$00C58C_H$	TREQR85 [R] 00000000 00000000		TREQR75 [R] 00000000 00000000		
$00C590_H$	NEWDT25 [R] 00000000 00000000		NEWDT15 [R] 00000000 00000000		
$00C594_H$	NEWDT45 [R] 00000000 00000000		NEWDT35 [R] 00000000 00000000		
$00C598_H$	NEWDT65 [R] 00000000 00000000		NEWDT55 [R] 00000000 00000000		
$00C59C_H$	NEWDT85 [R] 00000000 00000000		NEWDT75 [R] 00000000 00000000		
$00C5A0_H$	INTPND25 [R] 00000000 00000000		INTPND15 [R] 00000000 00000000		
$00C5A4_H$	INTPND45 [R] 00000000 00000000		INTPND35 [R] 00000000 00000000		
$00C5A8_H$	INTPND65 [R] 00000000 00000000		INTPND55 [R] 00000000 00000000		
$00C5AC_H$	INTPND85 [R] 00000000 00000000		INTPND75 [R] 00000000 00000000		
$00C5B0_H$	MSGVAL25 [R] 00000000 00000000		MSGVAL15 [R] 00000000 00000000		
$00C5B4_H$	MSGVAL45 [R] 00000000 00000000		MSGVAL35 [R] 00000000 00000000		
$00C5B8_H$	MSGVAL65 [R] 00000000 00000000		MSGVAL55 [R] 00000000 00000000		
$00C5BC_H$	MSGVAL85 [R] 00000000 00000000		MSGVAL75 [R] 00000000 00000000		
$00C5C0_H$ - $00EFFC_H$	reserved				

Address	Register				Block
	+0	+1	+2	+3	
$00F000_H$	BCTRL [R/W] - - - - - - - - - - - - - - - - 11111100 00000000				EDSU / MPU
$00F004_H$	BSTAT [R/W] - - - - - - - - - - - - - - - - 000 00000000 10 - - 000000				
$00F008_H$	BIAC [R] 00000000 00000000 00000000 00000000				
$00F00C_H$	BOAC [R] 00000000 00000000 00000000 00000000				
$00F010_H$	BIRQ [R/W] 00000000 00000000 00000000 00000000				
$00F014_H$ - $00F01C_H$	reserved				
$00F020_H$	BCR0 [R/W] - - - - - - - - 00000000 00000000 00000000				
$00F024_H$	BCR1 [R/W] - - - - - - - - 00000000 00000000 00000000				
$00F028_H$	BCR2 [R/W] - - - - - - - - 00000000 00000000 00000000				
$00F02C_H$	BCR3 [R/W] - - - - - - - - 00000000 00000000 00000000				
$00F030_H$	BCR4 [R/W] - - - - - - - - 00000000 00000000 00000000				
$00F034_H$	BCR5 [R/W] - - - - - - - - 00000000 00000000 00000000				
$00F038_H$	BCR6 [R/W] - - - - - - - - 00000000 00000000 00000000				
$00F03C_H$	BCR7 [R/W] - - - - - - - - 00000000 00000000 00000000				
$00F040_H$ - $00F07C_H$	reserved				

Address	Register				Block
	+0	+1	+2	+3	
$00F080_H$	BAD0 [R/W] XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX				EDSU / MPU
$00F084_H$	BAD1 [R/W] XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX				
$00F088_H$	BAD2 [R/W] XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX				
$00F08C_H$	BAD3 [R/W] XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX				
$00F090_H$	BAD4 [R/W] XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX				
$00F094_H$	BAD5 [R/W] XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX				
$00F098_H$	BAD6 [R/W] XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX				
$00F09C_H$	BAD7 [R/W] XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX				
$00F0A0_H$	BAD8 [R/W] XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX				
$00F0A4_H$	BAD9 [R/W] XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX				
$00F0A8_H$	BAD10 [R/W] XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX				
$00F0AC_H$	BAD11 [R/W] XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX				
$00F0B0_H$	BAD12 [R/W] XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX				
$00F0B4_H$	BAD13 [R/W] XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX				
$00F0B8_H$	BAD14 [R/W] XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX				
$00F0BC_H$	BAD15 [R/W] XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX				

Address	Register				Block
	+0	+1	+2	+3	
$00F0C0_H$	BAD16 [R/W] XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX				EDSU / MPU
$00F0C4_H$	BAD17 [R/W] XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX				
$00F0C8_H$	BAD18 [R/W] XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX				
$00F0CC_H$	BAD19 [R/W] XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX				
$00F0D0_H$	BAD20 [R/W] XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX				
$00F0D4_H$	BAD21 [R/W] XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX				
$00F0D8_H$	BAD22 [R/W] XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX				
$00F0DC_H$	BAD23 [R/W] XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX				
$00F0E0_H$	BAD24 [R/W] XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX				
$00F0E4_H$	BAD25 [R/W] XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX				
$00F0E8_H$	BAD26 [R/W] XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX				
$00F0EC_H$	BAD27 [R/W] XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX				
$00F0F0_H$	BAD28 [R/W] XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX				
$00F0F4_H$	BAD29 [R/W] XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX				
$00F0F8_H$	BAD30 [R/W] XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX				
$00F0FC_H$	BAD31 [R/W] XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX				
$00F100_H$ - $00FFFC_H$	reserved				

Address	Register				Block
	+0	+1	+2	+3	
010000_H - $013FFC_H$	Cache TAG way 1 (010000_H - $0107FC_H$)				2 way set associative I-Cache 4kB
014000_H - $017FFC_H$	Cache TAG way 2 (014000_H - $0147FC_H$)				
018000_H - $01BFFC_H$	Cache RAM way 1 (018000_H - $0187FC_H$)				
$01C000_H$ - $01FFFC_H$	Cache RAM way 2 ($01C000_H$ - $01C7FC_H$)				

Address	Register				Block
	+0	+1	+2	+3	
020000_H - $02FFFC_H$	MB91V460 D-RAM size is 64kB : 020000_H - $02FFFC_H$ (data access is 0 waitcycles)				D-RAM 64 kB
030000_H - $03FFFC_H$	MB91V460 I-/D-RAM size is 64kB : 030000_H - $03FFFC_H$ (instruction access is 0 waitcycles, data access is 1 waitcycle)				I-/D-RAM 64 kB
040000_H - $05FFFC_H$	ROMS00 area (128kB)				
060000_H - $07FFFC_H$	ROMS01 area (128kB)				
080000_H - $09FFFC_H$	ROMS02 area (128kB)				
$0A0000_H$ - $0BFFFC_H$	ROMS03 area (128kB)				
$0C0000_H$ - $0DFFFC_H$	ROMS04 area (128kB)				
$0E0000_H$ - $0FFFF4_H$	ROMS05 area (128kB)				
$0FFFF8_H$	FMV [R] $06\ 00\ 00\ 00_H$				Fixed Reset/Mode Vector
$0FFFFC_H$	FRV [R] $00\ 00\ BF\ F8_H$				
100000_H - $13FFFC_H$	ROMS06 area (256kB)				
140000_H - $17FFFC_H$	ROMS07 area (256kB)				
180000_H - $1BFFFC_H$	ROMS08 area (256kB)				
$1C0000_H$ - $1FFFFC_H$	ROMS09 area (256kB)				

Address	Register				Block
	+0	+1	+2	+3	
200000_H - $27FFFC_H$	ROMS10 area (512kB)				
280000_H - $2FFFFC_H$	ROMS11 area (512kB)				
300000_H - $37FFFC_H$	ROMS12 area (512kB)				
380000_H - $3FFFFC_H$	ROMS13 area (512kB)				
400000_H - $47FFFC_H$	ROMS14 area (512kB)				
480000_H - $4FFFFC_H$	ROMS15 area (512kB)				
Write operations to address $0FFFF8_H$ and $0FFFFC_H$ are not possible. When reading these addresses, the values shown above will be read.					

Notes:

*1 Use a read access (byte or halfword) to this address to synchronize the CPU operation (e.g. the interrupt acceptance of the CPU) to a preceding write access to the resources on R-bus (e.g. to an interrupt flag) on following addresses (0x0000-0x01FF, 0x0280-0x037F, 0x0400-0x063F and 0x0C00-0x0FFF).

*2 Use a read access (byte or halfword) to this address to synchronize the CPU operation (e.g. the interrupt acceptance of the CPU) to a preceding write access to the CANs on D-bus (e.g. to an interrupt flag) on following addresses (0xC000-0xFFFF).

3. Interrupt Vector Table

This section shows the allocation of interrupt and interrupt vector/interrupt register.

Interrupt	Interrupt number		Interrupt level[*1]		Interrupt vector[*2]		
	Decimal	Hexa-decimal	Setting Register	Register address	Offset	Default Vector address	RN[*8]
Reset	0	00	-	-	0x3FC	0x000FFFFC	
Mode vector	1	01	-	-	0x3F8	0x000FFFF8	
System reserved	2	02	-	-	0x3F4	0x000FFFF4	
System reserved	3	03	-	-	0x3F0	0x000FFFF0	
System reserved	4	04	-	-	0x3EC	0x000FFFEC	
CPU supervisor mode (INT #5 instruction) [*6]	5	05	-	-	0x3E8	0x000FFFE8	
Memory Protection exception [*6]	6	06	-	-	0x3E4	0x000FFFE4	
Co-processor fault trap [*5]	7	07	-	-	0x3E0	0x000FFFE0	
Co-processor error trap [*5]	8	08	-	-	0x3DC	0x000FFFDC	
INTE instruction [*5]	9	09	-	-	0x3D8	0x000FFFD8	
Instruction break exception [*5]	10	0A	-	-	0x3D4	0x000FFFD4	
Operand break trap [*5]	11	0B	-	-	0x3D0	0x000FFFD0	
Step trace trap [*5]	12	0C	-	-	0x3CC	0x000FFFCC	
NMI interrupt (tool)[*5]	13	0D	-	-	0x3C8	0x000FFFC8	
Undefined instruction exception	14	0E	-	-	0x3C4	0x000FFFC4	
NMI request	15	0F	F_H fixed		0x3C0	0x000FFFC0	
External Interrupt 0	16	10	ICR00	0x440	0x3BC	0x000FFFBC	0, 16
External Interrupt 1	17	11			0x3B8	0x000FFFB8	1, 17
External Interrupt 2	18	12	ICR01	0x441	0x3B4	0x000FFFB4	2, 18
External Interrupt 3	19	13			0x3B0	0x000FFFB0	3, 19
External Interrupt 4	20	14	ICR02	0x442	0x3AC	0x000FFFAC	20
External Interrupt 5	21	15			0x3A8	0x000FFFA8	21
External Interrupt 6	22	16	ICR03	0x443	0x3A4	0x000FFFA4	22
External Interrupt 7	23	17			0x3A0	0x000FFFA0	23

External Interrupt 8	24	18	ICR04	0x444	0x39C	0x000FFF9C	
External Interrupt 9	25	19			0x398	0x000FFF98	
External Interrupt 10	26	1A	ICR05	0x445	0x394	0x000FFF94	
External Interrupt 11	27	1B			0x390	0x000FFF90	
External Interrupt 12	28	1C	ICR06	0x446	0x38C	0x000FFF8C	
External Interrupt 13	29	1D			0x388	0x000FFF88	
External Interrupt 14	30	1E	ICR07	0x447	0x384	0x000FFF84	
External Interrupt 15	31	1F			0x380	0x000FFF80	
Reload Timer 0	32	20	ICR08	0x448	0x37C	0x000FFF7C	4, 32
Reload Timer 1	33	21			0x378	0x000FFF78	5, 33
Reload Timer 2	34	22	ICR09	0x449	0x374	0x000FFF74	34
Reload Timer 3	35	23			0x370	0x000FFF70	35
Reload Timer 4	36	24	ICR10	0x44A	0x36C	0x000FFF6C	36
Reload Timer 5	37	25			0x368	0x000FFF68	37
Reload Timer 6	38	26	ICR11	0x44B	0x364	0x000FFF64	38
Reload Timer 7	39	27			0x360	0x000FFF60	39
Free Run Timer 0	40	28	ICR12	0x44C	0x35C	0x000FFF5C	40
Free Run Timer 1	41	29			0x358	0x000FFF58	41
Free Run Timer 2	42	2A	ICR13	0x44D	0x354	0x000FFF54	42
Free Run Timer 3	43	2B			0x350	0x000FFF50	43
Free Run Timer 4	44	2C	ICR14	0x44E	0x34C	0x000FFF4C	44
Free Run Timer 5	45	2D			0x348	0x000FFF48	45
Free Run Timer 6	46	2E	ICR15	0x44F	0x344	0x000FFF44	46
Free Run Timer 7	47	2F			0x340	0x000FFF40	47
CAN 0	48	30	ICR16	0x450	0x33C	0x000FFF3C	
CAN 1	49	31			0x338	0x000FFF38	
CAN 2	50	32	ICR17	0x451	0x334	0x000FFF34	
CAN 3	51	33			0x330	0x000FFF30	
CAN 4	52	34	ICR18	0x452	0x32C	0x000FFF2C	
CAN 5	53	35			0x328	0x000FFF28	
USART (LIN) 0 RX	54	36	ICR19	0x453	0x324	0x000FFF24	6, 48
USART (LIN) 0 TX	55	37			0x320	0x000FFF20	7, 49
USART (LIN) 1 RX	56	38	ICR20	0x454	0x31C	0x000FFF1C	8, 50
USART (LIN) 1 TX	57	39			0x318	0x000FFF18	9, 51

USART (LIN) 2 RX	58	3A	ICR21	0x455	0x314	0x000FFF14	52
USART (LIN) 2 TX	59	3B			0x310	0x000FFF10	53
USART (LIN) 3 RX	60	3C	ICR22	0x456	0x30C	0x000FFF0C	54
USART (LIN) 3 TX	61	3D			0x308	0x000FFF08	55
System reserved	62	3E	ICR23 *4	0x457	0x304	0x000FFF04	
Delayed Interrupt	63	3F			0x300	0x000FFF00	
System reserved *3	64	40	(ICR24)	(0x458)	0x2FC	0x000FFEFC	
System reserved *3	65	41			0x2F8	0x000FFEF8	
USART (LIN, FIFO) 4 RX	66	42	ICR25	0x459	0x2F4	0x000FFEF4	10, 56
USART (LIN, FIFO) 4 TX	67	43			0x2F0	0x000FFEF0	11, 57
USART (LIN, FIFO) 5 RX	68	44	ICR26	0x45A	0x2EC	0x000FFEEC	12, 58
USART (LIN, FIFO) 5 TX	69	45			0x2E8	0x000FFEE8	13, 59
USART (LIN, FIFO) 6 RX	70	46	ICR27	0x45B	0x2E4	0x000FFEE4	60
USART (LIN, FIFO) 6 TX	71	47			0x2E0	0x000FFEE0	61
USART (LIN, FIFO) 7 RX	72	48	ICR28	0x45C	0x2DC	0x000FFEDC	62
USART (LIN, FIFO) 7 TX	73	49			0x2D8	0x000FFED8	63
I2C 0 / I2C 2	74	4A	ICR29	0x45D	0x2D4	0x000FFED4	
I2C 1 / I2C 3	75	4B			0x2D0	0x000FFED0	
USART (LIN) 8 RX	76	4C	ICR30	0x45E	0x2CC	0x000FFECC	64
USART (LIN) 8 TX	77	4D			0x2C8	0x000FFEC8	65
USART (LIN) 9 RX	78	4E	ICR31	0x45F	0x2C4	0x000FFEC4	66
USART (LIN) 9 TX	79	4F			0x2C0	0x000FFEC0	67
USART (LIN) 10 RX	80	50	ICR32	0x460	0x2BC	0x000FFEBC	68
USART (LIN) 10 TX	81	51			0x2B8	0x000FFEB8	69
USART (LIN) 11 RX	82	52	ICR33	0x461	0x2B4	0x000FFEB4	70
USART (LIN) 11 TX	83	53			0x2B0	0x000FFEB0	71
USART (LIN) 12 RX	84	54	ICR34	0x462	0x2AC	0x000FFEAC	72
USART (LIN) 12 TX	85	55			0x2A8	0x000FFEA8	73
USART (LIN) 13 RX	86	56	ICR35	0x463	0x2A4	0x000FFEA4	74
USART (LIN) 13 TX	87	57			0x2A0	0x000FFEA0	75
USART (LIN) 14 RX	88	58	ICR36	0x464	0x29C	0x000FFE9C	76
USART (LIN) 14 TX	89	59			0x298	0x000FFE98	77
USART (LIN) 15 RX	90	5A	ICR37	0x465	0x294	0x000FFE94	78
USART (LIN) 15 TX	91	5B			0x290	0x000FFE90	79

Input Capture 0	92	5C	ICR38	0x466	0x28C	0x000FFE8C	80
Input Capture 1	93	5D			0x288	0x000FFE88	81
Input Capture 2	94	5E	ICR39	0x467	0x284	0x000FFE84	82
Input Capture 3	95	5F			0x280	0x000FFE80	83
Input Capture 4	96	60	ICR40	0x468	0x27C	0x000FFE7C	84
Input Capture 5	97	61			0x278	0x000FFE78	85
Input Capture 6	98	62	ICR41	0x469	0x274	0x000FFE74	86
Input Capture 7	99	63			0x270	0x000FFE70	87
Output Compare 0	100	64	ICR42	0x46A	0x26C	0x000FFE6C	88
Output Compare 1	101	65			0x268	0x000FFE68	89
Output Compare 2	102	66	ICR43	0x46B	0x264	0x000FFE64	90
Output Compare 3	103	67			0x260	0x000FFE60	91
Output Compare 4	104	68	ICR44	0x46C	0x25C	0x000FFE5C	92
Output Compare 5	105	69			0x258	0x000FFE58	93
Output Compare 6	106	6A	ICR45	0x46D	0x254	0x000FFE54	94
Output Compare 7	107	6B			0x250	0x000FFE50	95
Sound Generator	108	6C	ICR46	0x46E	0x24C	0x000FFE4C	
Pulse Frequ. Modulator	109	6D			0x248	0x000FFE48	
System reserved	110	6E	ICR47 *4	0x46F	0x244	0x000FFE44	
System reserved	111	6F			0x240	0x000FFE40	
Prog. Pulse Gen. 0	112	70	ICR48	0x470	0x23C	0x000FFE3C	15, 96
Prog. Pulse Gen. 1	113	71			0x238	0x000FFE38	97
Prog. Pulse Gen. 2	114	72	ICR49	0x471	0x234	0x000FFE34	98
Prog. Pulse Gen. 3	115	73			0x230	0x000FFE30	99
Prog. Pulse Gen. 4	116	74	ICR50	0x472	0x22C	0x000FFE2C	100
Prog. Pulse Gen. 5	117	75			0x228	0x000FFE28	101
Prog. Pulse Gen. 6	118	76	ICR51	0x473	0x224	0x000FFE24	102
Prog. Pulse Gen. 7	119	77			0x220	0x000FFE20	103
Prog. Pulse Gen. 8	120	78	ICR52	0x474	0x21C	0x000FFE1C	104
Prog. Pulse Gen. 9	121	79			0x218	0x000FFE18	105
Prog. Pulse Gen. 10	122	7A	ICR53	0x475	0x214	0x000FFE14	106
Prog. Pulse Gen. 11	123	7B			0x210	0x000FFE10	107
Prog. Pulse Gen. 12	124	7C	ICR54	0x476	0x20C	0x000FFE0C	108
Prog. Pulse Gen. 13	125	7D			0x208	0x000FFE08	109

Prog. Pulse Gen. 14	126	7E	ICR55	0x477	0x204	0x000FFE04	110
Prog. Pulse Gen. 15	127	7F			0x200	0x000FFE00	111
Up/Down Counter 0	128	80	ICR56	0x478	0x1FC	0x000FFDFC	
Up/Down Counter 1	129	81			0x1F8	0x000FFDF8	
Up/Down Counter 2	130	82	ICR57	0x479	0x1F4	0x000FFDF4	
Up/Down Counter 3	131	83			0x1F0	0x000FFDF0	
Real Time Clock	132	84	ICR58	0x47A	0x1EC	0x000FFDEC	
Calibration Unit	133	85			0x1E8	0x000FFDE8	
A/D Converter 0	134	86	ICR59	0x47B	0x1E4	0x000FFDE4	14, 112
-	135	87			0x1E0	0x000FFDE0	
Alarm Comparator 0	136	88	ICR60	0x47C	0x1DC	0x000FFDDC	
Alarm Comparator 1	137	89			0x1D8	0x000FFDD8	
Low Voltage Detection	138	8A	ICR61	0x47D	0x1D4	0x000FFDD4	
-	139	8B			0x1D0	0x000FFDD0	
Timebase Overflow	140	8C	ICR62	0x47E	0x1CC	0x000FFDCC	
PLL Clock Gear	141	8D			0x1C8	0x000FFDC8	
DMA Controller	142	8E	ICR63	0x47F	0x1C4	0x000FFDC4	
Main/Sub OSC stability wait	143	8F			0x1C0	0x000FFDC0	
Boot Security vector *7	144	90	-	-	0x1BC	0x000FFDBC	
Used by the INT instruction.	145 to 255	91 to FF	-	-	0x1B8 to 0x000	0x000FFDB8 to 0x000FFC00	

Table 3-1 Interrupt Vector Table

Notes:

*1 The ICRs are located in the interrupt controller and set the interrupt level for each interrupt request. An ICR is provided for each interrupt request.
*2 The vector address for each EIT (exception, interrupt or trap) is calculated by adding the listed offset to the table base register value (TBR). The TBR specifies the top of the EIT vector table. The addresses listed in the table are for the default TBR value (0x000FFC00). The TBR is initialized to this value by a reset. After execution of the internal boot ROM TBR is set to 0x000FFC00.
*3 Used by REALOS
*4 ICR23 and ICR47 can be exchanged by setting the REALOS compatibility bit (addr 0x0C03 : IOS[0])
*5 System reserved
*6 Memory Protection Unit (MPU) support
*7 Only for MB91V460. Please see Chapter 53 Fixed Mode-Reset Vector / BOOT-ROM (Page No.985) for boot security vectors used on flash devices.
*8 RN resource number used for DMA operation. No number means that this resource interrupt cannot be used to trigger a DMA transfer.

4. Package

■ BGA-660P-M02 package (BGA660-03EK): MB91V460

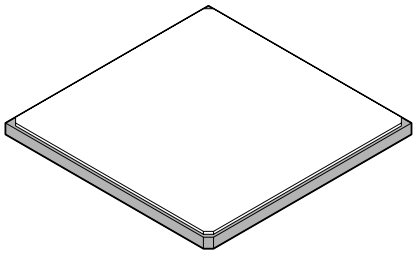
Figure 4-1 External Dimension of BGA660-03EK

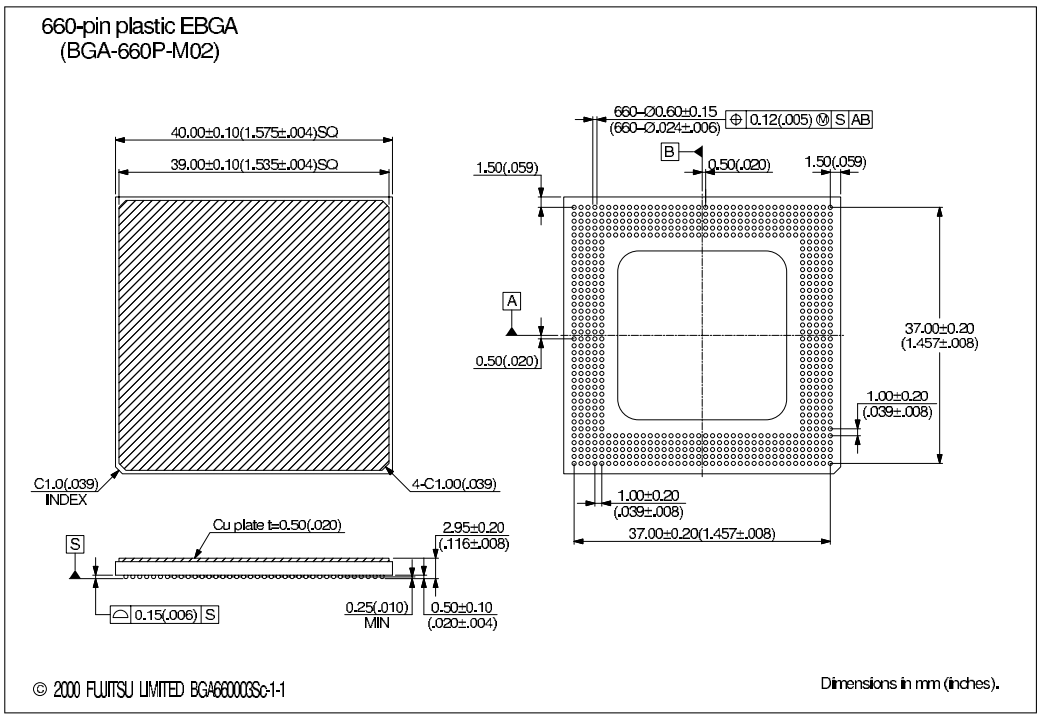
FUJITSU SEMICONDUCTOR
DATA SHEET

ENHANCED BALL GRID ARRAY PACKAGE
660 PIN PLASTIC

To Top / Package Lineup / Package Index

BGA-660P-M02

660-pin plastic EBGA  (BGA-660P-M02)	Ball pitch	1.00 mm
	Ball matrix	38 × 38
	Package width × package length	40.00 × 40.00 mm
	Sealing method	Resin sealed
	Ball size	∅ 0.6 mm
	Mounting height	3.15 mm MAX
	Weight	14.2g

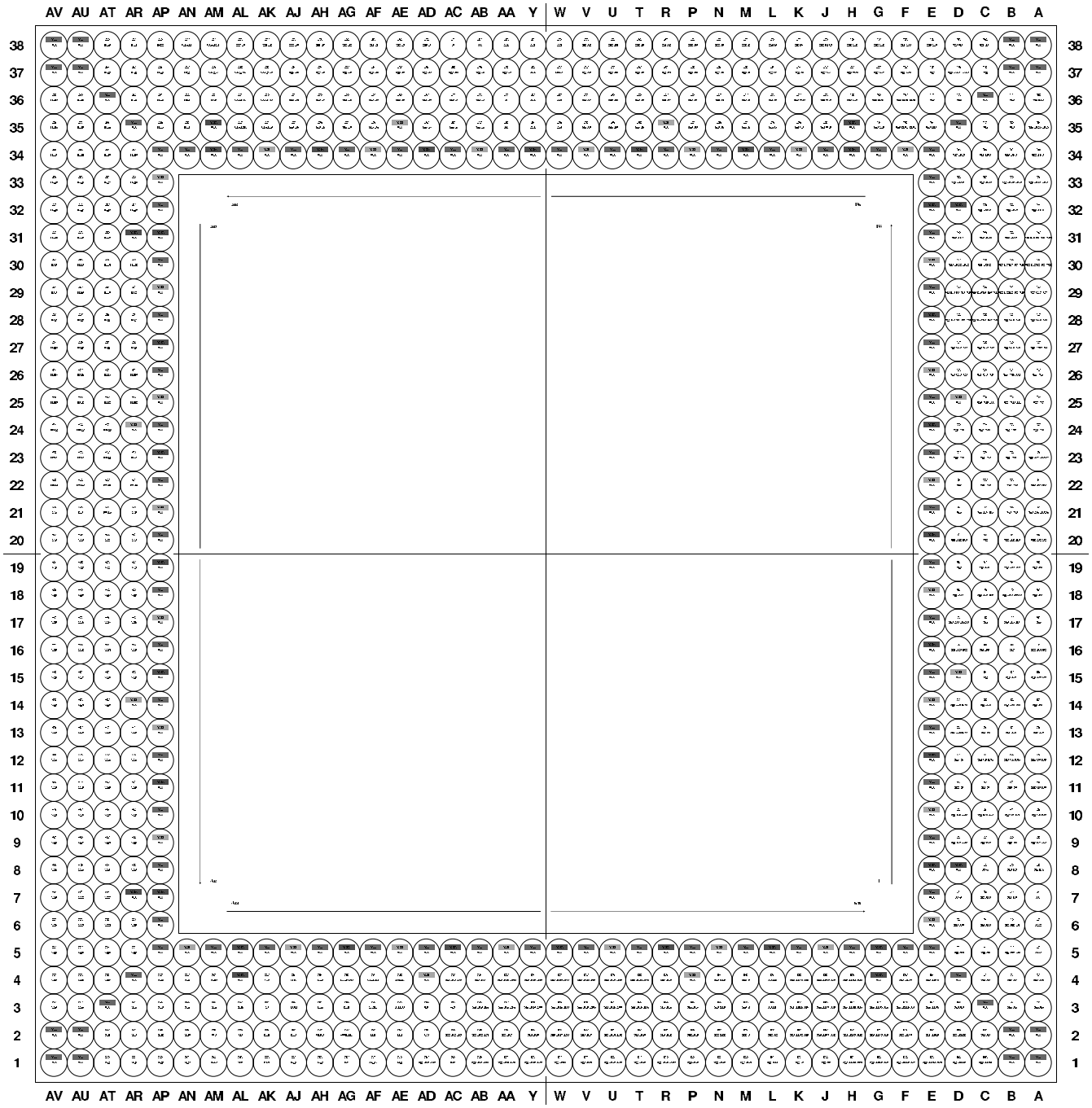


0011

5. Pin Assignment Diagram

■ MB91V460 (BGA660 package)

Figure 5-1 Pin Assignment Diagram of BGA660-03EK



6. Pin Definitions

JEDEC	Pin (INNER)	Pad	I/O	Function	PFR=1	EPFR=1	Special	Type	Pull Up/ Dwn	CMOS/ CMOS Hyst/ Auto / TTL	Input Stop	Usage	Output
AL38	315	262	P00_7	D31	D31	-	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA
AJ37	314	261	P00_6	D30	D30	-	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA
AJ36	311	259	P00_5	D29	D29	-	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA
AJ35	310	258	P00_4	D28	D28	-	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA
AH36	309	257	P00_3	D27	D27	-	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA
AH35	308	256	P00_2	D26	D26	-	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA
AK38	307	255	P00_1	D25	D25	-	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA
AJ38	305	253	P00_0	D24	D24	-	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA
AH37	304	252	P01_7	D23	D23	-	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA
AG37	302	251	P01_6	D22	D22	-	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA
AG35	300	249	P01_5	D21	D21	-	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA
AG36	299	250	P01_4	D20	D20	-	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA
AF35	298	247	P01_3	D19	D19	-	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA
AF36	297	248	P01_2	D18	D18	-	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA
AH38	295	246	P01_1	D17	D17	-	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA
AF37	294	244	P01_0	D16	D16	-	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA
AG38	293	245	P02_7	D15	D15	-	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA
AE37	292	243	P02_6	D14	D14	-	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA
AE36	289	241	P02_5	D13	D13	-	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA
AD37	288	240	P02_4	D12	D12	-	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA
AD36	287	239	P02_3	D11	D11	-	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA
AC37	286	238	P02_2	D10	D10	-	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA
AF38	285	237	P02_1	D9	D9	-	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA
AE38	283	236	P02_0	D8	D8	-	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA
AD35	282	234	P03_7	D7	D7	-	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA
AC35	280	233	P03_6	D6	D6	-	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA
AB37	278	231	P03_5	D5	D5	-	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA
AC36	277	232	P03_4	D4	D4	-	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA
AA37	276	229	P03_3	D3	D3	-	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA
AB36	275	230	P03_2	D2	D2	-	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA
AD38	273	228	P03_1	D1	D1	-	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA
AC38	271	227	P03_0	D0	D0	-	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA
AB35	272	226	X1	-	-	-	-	TO00_1	-	OSC	Stop	X1	-
AA36	270	225	X0	-	-	-	-	TO00_0	-	OSC	Stop	X0	-
AB38	267	222	X1A	-	-	-	-	TO01_1	-	OSC	Stop	X1A	-
AA35	266	221	X0A	-	-	-	-	TO01_0	-	OSC	Stop	X0A	-
W37	257	211	MONCLK	-	-	-	-	TC10_0	-	-	no	MONCLK	8mA
V35	256	212	P04_7	A31	A31	-	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA
V38	255	209	P04_6	A30	A30	-	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA
U38	253	207	P04_5	A29	A29	-	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA

V37	252	208	P04_4	A28	A28	-	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA
V36	251	205	P04_3	A27	A27	-	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA
U37	250	206	P04_2	A26	A26	-	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA
U36	249	203	P04_1	A25	A25	-	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA
U35	248	204	P04_0	A24	A24	-	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA
T35	246	202	P05_7	A23	A23	-	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA
T38	243	200	P05_6	A22	A22	-	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA
T37	242	199	P05_5	A21	A21	-	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA
R38	241	198	P05_4	A20	A20	-	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA
R37	240	197	P05_3	A19	A19	-	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA
T36	239	196	P05_2	A18	A18	-	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA
R36	237	195	P05_1	A17	A17	-	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA
P37	236	194	P05_0	A16	A16	-	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA
N37	234	193	P06_7	A15	A15	-	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA
P38	233	192	P06_6	A14	A14	-	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA
N38	231	190	P06_5	A13	A13	-	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA
P35	230	189	P06_4	A12	A12	-	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA
P36	229	188	P06_3	A11	A11	-	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA
N35	228	187	P06_2	A10	A10	-	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA
N36	227	186	P06_1	A9	A9	-	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA
M37	226	185	P06_0	A8	A8	-	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA
L37	224	184	P07_7	A7	A7	-	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA
M38	221	182	P07_6	A6	A6	-	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA
M35	220	181	P07_5	A5	A5	-	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA
L38	219	180	P07_4	A4	A4	-	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA
L35	218	179	P07_3	A3	A3	-	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA
M36	217	178	P07_2	A2	A2	-	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA
L36	215	177	P07_1	A1	A1	-	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA
K37	214	176	P07_0	A0	A0	-	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA
J37	212	175	P08_7	RDY	RDY	-	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA
K38	211	174	P08_6	BRQ	BRQ	-	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA
J38	209	172	P08_5	BGRNTX	BGRNTX	-	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA
K35	208	171	P08_4	RDX	RDX	-	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA
K36	207	170	P08_3	WRX3	WRX3	-	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA
J35	206	169	P08_2	WRX2	WRX2	-	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA
J36	205	168	P08_1	WRX1	WRX1	-	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA
H37	202	167	P08_0	WRX0	WRX0	-	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA
G37	200	165	P09_7	CSX7	CSX7	-	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA
H38	199	164	P09_6	CSX6	CSX6	-	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA
G38	197	163	P09_5	CSX5	CSX5	-	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA
H36	196	162	P09_4	CSX4	CSX4	-	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA
F38	195	161	P09_3	CSX3	CSX3	-	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA
G35	194	160	P09_2	CSX2	CSX2	-	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA
E38	193	159	P09_1	CSX1	CSX1	-	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA
F37	192	158	P09_0	CSX0	CSX0	-	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA

E37	190	157	P10_7	-	-	-	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA
G36	189	156	P10_6	MCLKE	MCLKE	^	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA
F36	187	154	P10_5	MCLKI	MCLKI	/MCLKI	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA
F35	186	153	P10_4	MCLKO	MCLKO	/MCLKO	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA
D38	185	152	P10_3	WEX	WEX	^	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA
E35	184	151	P10_2	BAAX	BAAX	^	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA
C38	183	150	P10_1	ASX	ASX	^	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA
D37	178	147	P10_0	SYSCLK	SYSCLK	/SYSCLK	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA
E36	177	148	P11_7	-	-	-	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA
C37	176	145	P11_6	-	-	-	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA
D36	175	146	P11_5	-	-	-	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA
B36	171	143	P11_4	-	-	-	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA
C35	170	142	P11_3	-	-	-	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA
B35	169	141	P11_2	-	-	-	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA
C34	168	139	P11_1	IOWRX	IOWRX	-	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA
A36	166	138	P11_0	IORDX	IORDX	-	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA
D34	165	137	P12_7	DEOP3	DEOP3	-	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA
A35	164	136	P12_6	DEOTX3	DEOTX3	DEOP3	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA
D33	163	135	P12_5	DACKX3	DACKX3	^	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA
B34	161	134	P12_4	DREQ3	DREQ3	^	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA
C33	160	133	P12_3	DEOP2	DEOP2	-	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA
B33	159	132	P12_2	DEOTX2	DEOTX2	DEOP2	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA
C32	158	131	P12_1	DACKX2	DACKX2	^	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA
A34	156	130	P12_0	DREQ2	DREQ2	^	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA
B32	155	129	P13_7	DEOP1	DEOP1	-	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA
A33	154	128	P13_6	DEOTX1	DEOTX1	DEOP1	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA
B31	153	127	P13_5	DACKX1	DACKX1	^	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA
D31	149	125	P13_4	DREQ1	DREQ1	^	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA
C31	148	124	P13_3	DEOP0	DEOP0	-	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA
D30	147	123	P13_2	DEOTX0	DEOTX0	DEOP0	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA
C30	146	122	P13_1	DACKX0	DACKX0	^	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA
A32	144	120	P13_0	DREQ0	DREQ0	^	-	TP04_0	U/D	CH / A / TTL	Stop	TTL (extbus)	4mA
B30	143	119	P14_7	-	ICU7/TIN7	TIN7	TTG15/7	TP00_0	U/D	CH / A	Stop	-	4mA
A31	142	118	P14_6	-	ICU6/TIN6	TIN6	TTG14/6	TP00_0	U/D	CH / A	Stop	-	4mA
B29	141	117	P14_5	-	ICU5/TIN5	TIN5	TTG13/5	TP00_0	U/D	CH / A	Stop	-	4mA
C29	138	116	P14_4	-	ICU4/TIN4	TIN4	TTG12/4	TP00_0	U/D	CH / A	Stop	-	4mA
D29	137	115	P14_3	-	ICU3/TIN3	TIN3	TTG11/3	TP00_0	U/D	CH / A	Stop	-	4mA
C28	136	114	P14_2	-	ICU2/TIN2	TIN2	TTG10/2	TP00_0	U/D	CH / A	Stop	-	4mA
D28	135	113	P14_1	-	ICU1/TIN1	TIN1	TTG9/1	TP00_0	U/D	CH / A	Stop	-	4mA
A30	134	112	P14_0	-	ICU0/TIN0	TIN0	TTG8/0	TP00_0	U/D	CH / A	Stop	-	4mA
A29	132	111	P15_7	-	OCU7	TOT7	-	TP00_0	U/D	CH / A	Stop	-	4mA
B28	131	110	P15_6	-	OCU6	TOT6	-	TP00_0	U/D	CH / A	Stop	-	4mA
B27	129	108	P15_5	-	OCU5	TOT5	-	TP00_0	U/D	CH / A	Stop	-	4mA
D27	127	107	P15_4	-	OCU4	TOT4	-	TP00_0	U/D	CH / A	Stop	-	4mA
C27	126	106	P15_3	-	OCU3	TOT3	-	TP00_0	U/D	CH / A	Stop	-	4mA

D26	125	105	P15_2	-	OCU2	TOT2	-	TP00_0	U/D	CH / A	Stop	-	4mA
C26	124	104	P15_1	-	OCU1	TOT1	-	TP00_0	U/D	CH / A	Stop	-	4mA
A28	122	102	P15_0	-	OCU0	TOT0	-	TP00_0	U/D	CH / A	Stop	-	4mA
B26	121	101	P16_7	-	PPG15	ATGX	-	TP00_0	U/D	CH / A	Stop	-	4mA
A27	120	100	P16_6	-	PPG14	PFM	-	TP00_0	U/D	CH / A	Stop	-	4mA
B25	119	99	P16_5	-	PPG13	SGO	-	TP00_0	U/D	CH / A	Stop	-	4mA
C25	116	97	P16_4	-	PPG12	SGA	-	TP00_0	U/D	CH / A	Stop	-	4mA
B24	115	96	P16_3	-	PPG11	^	-	TP00_0	U/D	CH / A	Stop	-	4mA
C24	114	95	P16_2	-	PPG10	^	-	TP00_0	U/D	CH / A	Stop	-	4mA
B23	113	94	P16_1	-	PPG9	^	-	TP00_0	U/D	CH / A	Stop	-	4mA
A26	112	93	P16_0	-	PPG8	^	-	TP00_0	U/D	CH / A	Stop	-	4mA
A25	110	92	P17_7	-	PPG7	-	-	TP00_0	U/D	CH / A	Stop	-	4mA
D24	109	91	P17_6	-	PPG6	-	-	TP00_0	U/D	CH / A	Stop	-	4mA
D23	107	90	P17_5	-	PPG5	-	-	TP00_0	U/D	CH / A	Stop	-	4mA
B22	105	88	P17_4	-	PPG4	-	-	TP00_0	U/D	CH / A	Stop	-	4mA
C23	104	89	P17_3	-	PPG3	-	-	TP00_0	U/D	CH / A	Stop	-	4mA
B21	103	86	P17_2	-	PPG2	-	-	TP00_0	U/D	CH / A	Stop	-	4mA
C22	102	87	P17_1	-	PPG1	-	-	TP00_0	U/D	CH / A	Stop	-	4mA
A24	100	85	P17_0	-	PPG0	-	-	TP00_0	U/D	CH / A	Stop	-	4mA
D22	99	82	P18_7	-	-	-	-	TP00_0	U/D	CH / A	Stop	-	4mA
A23	98	83	P18_6	-	SCK7	ZIN3/CK7	-	TP00_0	U/D	CH / A	Stop	-	4mA
C21	97	80	P18_5	-	SOT7	BIN3	-	TP00_0	U/D	CH / A	Stop	-	4mA
A22	94	79	P18_4	-	SIN7	AIN3	-	TP00_0	U/D	CH / A	Stop	-	4mA
D21	93	77	P18_3	-	-	-	-	TP00_0	U/D	CH / A	Stop	-	4mA
A21	92	78	P18_2	-	SCK6	ZIN2/CK6	-	TP00_0	U/D	CH / A	Stop	-	4mA
D20	91	75	P18_1	-	SOT6	BIN2	-	TP00_0	U/D	CH / A	Stop	-	4mA
B20	90	76	P18_0	-	SIN6	AIN2	-	TP00_0	U/D	CH / A	Stop	-	4mA
C20	89	73	P19_7	-	-	-	-	TP00_0	U/D	CH / A	Stop	-	4mA
A20	88	74	P19_6	-	SCK5	CK5	-	TP00_0	U/D	CH / A	Stop	-	4mA
C19	87	71	P19_5	-	SOT5	^	-	TP00_0	U/D	CH / A	Stop	-	4mA
A19	86	72	P19_4	-	SIN5	^	-	TP00_0	U/D	CH / A	Stop	-	4mA
D19	85	70	P19_3	-	-	-	-	TP00_0	U/D	CH / A	Stop	-	4mA
B19	84	69	P19_2	-	SCK4	CK4	-	TP00_0	U/D	CH / A	Stop	-	4mA
D18	83	68	P19_1	-	SOT4	^	-	TP00_0	U/D	CH / A	Stop	-	4mA
A18	82	67	P19_0	-	SIN4	^	-	TP00_0	U/D	CH / A	Stop	-	4mA
A17	80	65	P20_7	-	-	-	-	TP00_0	U/D	CH / A	Stop	-	4mA
B18	79	64	P20_6	-	SCK3	ZIN1/CK3	-	TP00_0	U/D	CH / A	Stop	-	4mA
C18	78	63	P20_5	-	SOT3	BIN1	-	TP00_0	U/D	CH / A	Stop	-	4mA
B17	77	62	P20_4	-	SIN3	AIN1	-	TP00_0	U/D	CH / A	Stop	-	4mA
C17	76	61	P20_3	-	-	-	-	TP00_0	U/D	CH / A	Stop	-	4mA
D17	75	60	P20_2	-	SCK2	ZIN0/CK2	-	TP00_0	U/D	CH / A	Stop	-	4mA
D16	73	58	P20_1	-	SOT2	BIN0	-	TP00_0	U/D	CH / A	Stop	-	4mA
A16	70	57	P20_0	-	SIN2	AIN0	-	TP00_0	U/D	CH / A	Stop	-	4mA
B16	69	55	P21_7	-	-	-	-	TP00_0	U/D	CH / A	Stop	-	4mA
A15	68	56	P21_6	-	SCK1	CK1	-	TP00_0	U/D	CH / A	Stop	-	4mA

B15	67	53	P21_5	-	SOT1	^	-	TP00_0	U/D	CH / A	Stop	-	4mA
C16	66	54	P21_4	-	SIN1	^	-	TP00_0	U/D	CH / A	Stop	-	4mA
C15	64	52	P21_3	-	-	-	-	TP00_0	U/D	CH / A	Stop	-	4mA
B14	63	51	P21_2	-	SCK0	CK0	-	TP00_0	U/D	CH / A	Stop	-	4mA
B13	61	49	P21_1	-	SOT0	^	-	TP00_0	U/D	CH / A	Stop	-	4mA
A14	60	50	P21_0	-	SIN0	^	-	TP00_0	U/D	CH / A	Stop	-	4mA
A13	58	47	P22_7	-	SCL1	-	-	TP02_0	-	CH / A	Stop	I2C	3mA
D14	57	46	P22_6	-	SDA1	-	INT15	TP02_0	-	CH / A	Stop	I2C	3mA
C14	56	45	P22_5	-	SCL0	-	-	TP02_0	-	CH / A	Stop	I2C	3mA
D13	55	44	P22_4	-	SDA0	-	INT14	TP02_0	-	CH / A	Stop	I2C	3mA
C13	54	42	P22_3	-	TX5	-	-	TP00_0	U/D	CH / A	Stop	-	4mA
B12	53	43	P22_2	-	RX5	-	INT13	TP00_0	U/D	CH / A	Stop	-	4mA
B11	51	41	P22_1	-	TX4	-	-	TP00_0	U/D	CH / A	Stop	-	4mA
A12	48	39	P22_0	-	RX4	-	INT12	TP00_0	U/D	CH / A	Stop	-	4mA
D12	47	38	P23_7	-	TX3	-	-	TP00_0	U/D	CH / A	Stop	-	4mA
A11	46	37	P23_6	-	RX3	-	INT11	TP00_0	U/D	CH / A	Stop	-	4mA
D11	45	36	P23_5	-	TX2	-	-	TP00_0	U/D	CH / A	Stop	-	4mA
C12	44	35	P23_4	-	RX2	-	INT10	TP00_0	U/D	CH / A	Stop	-	4mA
C11	42	34	P23_3	-	TX1	-	-	TP00_0	U/D	CH / A	Stop	-	4mA
B10	41	33	P23_2	-	RX1	-	INT9	TP00_0	U/D	CH / A	Stop	-	4mA
B9	39	32	P23_1	-	TX0	-	-	TP00_0	U/D	CH / A	Stop	-	4mA
A10	38	31	P23_0	-	RX0	-	INT8	TP00_0	U/D	CH / A	Stop	-	4mA
A9	36	29	P24_7	-	INT7	-	SCL3	TP02_0	-	CH / A	Stop	I2C	3mA
D10	35	28	P24_6	-	INT6	-	SDA3	TP02_0	-	CH / A	Stop	I2C	3mA
C10	34	26	P24_5	-	INT5	-	SCL2	TP02_0	-	CH / A	Stop	I2C	3mA
D9	33	27	P24_4	-	INT4	-	SDA2	TP02_0	-	CH / A	Stop	I2C	3mA
C9	32	25	P24_3	-	INT3	-	-	TP00_0	U/D	CH / A	Stop	-	4mA
B8	29	24	P24_2	-	INT2	-	-	TP00_0	U/D	CH / A	Stop	-	4mA
B7	27	23	P24_1	-	INT1	-	-	TP00_0	U/D	CH / A	Stop	-	4mA
A8	26	20	P24_0	-	INT0	-	-	TP00_0	U/D	CH / A	Stop	-	4mA
D2	688	576	P25_7	-	SMC2M5	-	-	TP05_0	-	CH / A	Stop	SMC / AN	30mA
E3	687	575	P25_6	-	SMC2P5	-	-	TP05_0	-	CH / A	Stop	SMC / AN	30mA
C1	685	573	P25_5	-	SMC1M5	-	-	TP05_0	-	CH / A	Stop	SMC / AN	30mA
E4	684	572	P25_4	-	SMC1P5	-	-	TP05_0	-	CH / A	Stop	SMC / AN	30mA
D1	683	571	P25_3	-	SMC2M4	-	-	TP05_0	-	CH / A	Stop	SMC / AN	30mA
F4	682	570	P25_2	-	SMC2P4	-	-	TP05_0	-	CH / A	Stop	SMC / AN	30mA
E2	680	568	P25_1	-	SMC1M4	-	-	TP05_0	-	CH / A	Stop	SMC / AN	30mA
F2	678	567	P25_0	-	SMC1P4	-	-	TP05_0	-	CH / A	Stop	SMC / AN	30mA
F3	679	566	P26_7	-	SMC2M3	AN31	-	TP05_0	-	CH / A	Stop	SMC / AN	30mA
G3	677	565	P26_6	-	SMC2P3	AN30	-	TP05_0	-	CH / A	Stop	SMC / AN	30mA
E1	675	562	P26_5	-	SMC1M3	AN29	-	TP05_0	-	CH / A	Stop	SMC / AN	30mA
G2	674	563	P26_4	-	SMC1P3	AN28	-	TP05_0	-	CH / A	Stop	SMC / AN	30mA
F1	673	560	P26_3	-	SMC2M2	AN27	-	TP05_0	-	CH / A	Stop	SMC / AN	30mA
H2	672	561	P26_2	-	SMC2P2	AN26	-	TP05_0	-	CH / A	Stop	SMC / AN	30mA
H4	668	558	P26_1	-	SMC1M2	AN25	-	TP05_0	-	CH / A	Stop	SMC / AN	30mA

J4	666	557	P26_0	-	SMC1P2	AN24	-	TP05_0	-	CH / A	Stop	SMC / AN	30mA
H3	667	556	P27_7	-	SMC2M1	AN23	-	TP05_0	-	CH / A	Stop	SMC / AN	30mA
J3	665	555	P27_6	-	SMC2P1	AN22	-	TP05_0	-	CH / A	Stop	SMC / AN	30mA
G1	663	552	P27_5	-	SMC1M1	AN21	-	TP05_0	-	CH / A	Stop	SMC / AN	30mA
J2	662	553	P27_4	-	SMC1P1	AN20	-	TP05_0	-	CH / A	Stop	SMC / AN	30mA
H1	661	550	P27_3	-	SMC2M0	AN19	-	TP05_0	-	CH / A	Stop	SMC / AN	30mA
K2	660	551	P27_2	-	SMC2P0	AN18	-	TP05_0	-	CH / A	Stop	SMC / AN	30mA
K3	657	547	P27_1	-	SMC1M0	AN17	-	TP05_0	-	CH / A	Stop	SMC / AN	30mA
K4	656	548	P27_0	-	SMC1P0	AN16	-	TP05_0	-	CH / A	Stop	SMC / AN	30mA
L3	655	545	ALARM_1	-	-	-	-	TA02_0	-	AN IN	-	Ana In	-
L4	654	546	ALARM_0	-	-	-	-	TA02_0	-	AN IN	-	Ana In	-
A5	20	14	P28_7	-	AN15	-	DA1	TP01_0	-	CH / A	Stop	AN / DA	4mA
B5	17	13	P28_6	-	AN14	-	DA0	TP01_0	-	CH / A	Stop	AN / DA	4mA
C7	16	12	P28_5	-	AN13	-	-	TP03_0	U/D	CH / A	Stop	AN	4mA
C6	14	10	P28_4	-	AN12	-	-	TP03_0	U/D	CH / A	Stop	AN	4mA
D6	13	11	P28_3	-	AN11	-	-	TP03_0	U/D	CH / A	Stop	AN	4mA
A4	12	8	P28_2	-	AN10	-	-	TP03_0	U/D	CH / A	Stop	AN	4mA
D5	11	9	P28_1	-	AN9	-	-	TP03_0	U/D	CH / A	Stop	AN	4mA
A3	10	6	P28_0	-	AN8	-	-	TP03_0	U/D	CH / A	Stop	AN	4mA
B4	5	4	P29_7	-	AN7	-	-	TP03_0	U/D	CH / A	Stop	AN	4mA
C5	4	3	P29_6	-	AN6	-	-	TP03_0	U/D	CH / A	Stop	AN	4mA
B3	3	2	P29_5	-	AN5	-	-	TP03_0	U/D	CH / A	Stop	AN	4mA
C4	2	1	P29_4	-	AN4	-	-	TP03_0	U/D	CH / A	Stop	AN	4mA
C2	690	578	P29_3	-	AN3	-	-	TP03_0	U/D	CH / A	Stop	AN	4mA
D3	689	577	P29_2	-	AN2	-	-	TP03_0	U/D	CH / A	Stop	AN	4mA
J1	653	544	P29_1	-	AN1	-	-	TP03_0	U/D	CH / A	Stop	AN	4mA
L2	650	543	P29_0	-	AN0	-	-	TP03_0	U/D	CH / A	Stop	AN	4mA
K1	651	542	P30_7	-	V3	-	-	TP08_0	-	CH / A	Stop	LCD V3	4mA
M2	648	541	P30_6	-	V2	-	-	TP07_0	-	CH / A	Stop	LCD V0,V1,V2	4mA
M4	646	540	P30_5	-	V1	-	-	TP07_0	-	CH / A	Stop	LCD V0,V1,V2	4mA
M3	645	538	P30_4	-	V0	-	-	TP07_0	-	CH / A	Stop	LCD V0,V1,V2	4mA
N4	644	539	P30_3	-	COM3	-	-	TP06_0	-	CH / A	Stop	LCD COM/SEG	4mA
N3	643	536	P30_2	-	COM2	-	-	TP06_0	-	CH / A	Stop	LCD COM/SEG	4mA
L1	641	534	P30_1	-	COM1	-	-	TP06_0	-	CH / A	Stop	LCD COM/SEG	4mA
N2	640	535	P30_0	-	COM0	-	-	TP06_0	-	CH / A	Stop	LCD COM/SEG	4mA
M1	639	532	P31_7	-	SEG39	-	-	TP06_0	-	CH / A	Stop	LCD COM/SEG	4mA
P2	638	533	P31_6	-	SEG38	-	-	TP06_0	-	CH / A	Stop	LCD COM/SEG	4mA
P3	635	529	P31_5	-	SEG37	-	-	TP06_0	-	CH / A	Stop	LCD COM/SEG	4mA
R2	634	530	P31_4	-	SEG36	-	-	TP06_0	-	CH / A	Stop	LCD COM/SEG	4mA
R3	633	527	P31_3	-	SEG35	-	-	TP06_0	-	CH / A	Stop	LCD COM/SEG	4mA
T2	632	528	P31_2	-	SEG34	-	-	TP06_0	-	CH / A	Stop	LCD COM/SEG	4mA
N1	631	526	P31_1	-	SEG33	-	-	TP06_0	-	CH / A	Stop	LCD COM/SEG	4mA
P1	629	524	P31_0	-	SEG32	-	-	TP06_0	-	CH / A	Stop	LCD COM/SEG	4mA
R4	628	525	P32_7	-	SEG31	-	-	TP06_0	-	CH / A	Stop	LCD COM/SEG	4mA
T4	626	523	P32_6	-	SEG30	SCK15	-	TP06_0	-	CH / A	Stop	LCD COM/SEG	4mA

U2	624	522	P32_5	-	SEG29	SOT15	-	TP06_0	-	CH / A	Stop	LCD COM/SEG	4mA
T3	623	521	P32_4	-	SEG28	SIN15	-	TP06_0	-	CH / A	Stop	LCD COM/SEG	4mA
V2	622	520	P32_3	-	SEG27	-	-	TP06_0	-	CH / A	Stop	LCD COM/SEG	4mA
U3	621	519	P32_2	-	SEG26	SCK14	-	TP06_0	-	CH / A	Stop	LCD COM/SEG	4mA
R1	619	517	P32_1	-	SEG25	SOT14	-	TP06_0	-	CH / A	Stop	LCD COM/SEG	4mA
U4	618	516	P32_0	-	SEG24	SIN14	-	TP06_0	-	CH / A	Stop	LCD COM/SEG	4mA
T1	617	515	P33_7	-	SEG23	-	-	TP06_0	-	CH / A	Stop	LCD COM/SEG	4mA
V3	616	514	P33_6	-	SEG22	SCK13	-	TP06_0	-	CH / A	Stop	LCD COM/SEG	4mA
U1	613	512	P33_5	-	SEG21	SOT13	-	TP06_0	-	CH / A	Stop	LCD COM/SEG	4mA
V4	612	511	P33_4	-	SEG20	SIN13	-	TP06_0	-	CH / A	Stop	LCD COM/SEG	4mA
V1	611	510	P33_3	-	SEG19	-	-	TP06_0	-	CH / A	Stop	LCD COM/SEG	4mA
W4	610	509	P33_2	-	SEG18	SCK12	-	TP06_0	-	CH / A	Stop	LCD COM/SEG	4mA
W2	609	508	P33_1	-	SEG17	SOT12	-	TP06_0	-	CH / A	Stop	LCD COM/SEG	4mA
W3	608	507	P33_0	-	SEG16	SIN12	-	TP06_0	-	CH / A	Stop	LCD COM/SEG	4mA
W1	607	506	P34_7	-	SEG15	-	-	TP06_0	-	CH / A	Stop	LCD COM/SEG	4mA
Y3	606	505	P34_6	-	SEG14	SCK11	-	TP06_0	-	CH / A	Stop	LCD COM/SEG	4mA
Y1	605	503	P34_5	-	SEG13	SOT11	-	TP06_0	-	CH / A	Stop	LCD COM/SEG	4mA
Y4	604	504	P34_4	-	SEG12	SIN11	-	TP06_0	-	CH / A	Stop	LCD COM/SEG	4mA
Y2	603	501	P34_3	-	SEG11	-	-	TP06_0	-	CH / A	Stop	LCD COM/SEG	4mA
AA4	602	502	P34_2	-	SEG10	SCK10	-	TP06_0	-	CH / A	Stop	LCD COM/SEG	4mA
AA1	601	499	P34_1	-	SEG9	SOT10	-	TP06_0	-	CH / A	Stop	LCD COM/SEG	4mA
AB1	599	497	P34_0	-	SEG8	SIN10	-	TP06_0	-	CH / A	Stop	LCD COM/SEG	4mA
AA2	598	498	P35_7	-	SEG7	-	-	TP06_0	-	CH / A	Stop	LCD COM/SEG	4mA
AA3	597	495	P35_6	-	SEG6	SCK9	-	TP06_0	-	CH / A	Stop	LCD COM/SEG	4mA
AB2	596	496	P35_5	-	SEG5	SOT9	-	TP06_0	-	CH / A	Stop	LCD COM/SEG	4mA
AB3	595	493	P35_4	-	SEG4	SIN9	-	TP06_0	-	CH / A	Stop	LCD COM/SEG	4mA
AB4	594	494	P35_3	-	SEG3	-	-	TP06_0	-	CH / A	Stop	LCD COM/SEG	4mA
AC4	592	492	P35_2	-	SEG2	SCK8	-	TP06_0	-	CH / A	Stop	LCD COM/SEG	4mA
AC1	589	489	P35_1	-	SEG1	SOT8	-	TP06_0	-	CH / A	Stop	LCD COM/SEG	4mA
AC2	588	490	P35_0	-	SEG0	SIN8	-	TP06_0	-	CH / A	Stop	LCD COM/SEG	4mA
AD1	587	487	INITX	-	-	-	-	TC02_0	Up	CH (old)	no	MCU control	-
AD2	586	488	RSTX	-	-	-	-	TC01_0	Up	CH (old)	no	MCU control	-
AC3	585	486	HSTX	-	-	-	-	TC00_0	-	CH (old)	no	MCU control	-
AD3	583	484	NMIX	-	-	-	-	TC01_0	Up	CH (old)	no	MCU control	-
AE2	582	485	MD_2	-	-	-	-	TC02_0	-	CH (old)	no	MCU control	-
AF2	580	483	MD_1	-	-	-	-	TC02_0	-	CH (old)	no	MCU control	-
AE1	579	481	MD_0	-	-	-	-	TC02_0	-	CH (old)	no	MCU control	-
AF1	577	479	ADC_SEL	-	-	-	-	TC00_0	-	CH (old)	no	MCU control	-
AE4	576	480	ALARM_SEL	-	-	-	-	TC00_0	-	CH (old)	no	MCU control	-
AE3	575	477	SRAM_SFX	-	-	-	-	TC00_0	-	CH (old)	no	MCU control	-
AF4	574	478	FSC_DISABLE	-	-	-	-	TC00_0	-	CH (old)	no	MCU control	-
AF3	573	475	CSV_KILL	-	-	-	-	TC00_0	-	CH (old)	no	MCU control	-
AG2	572	476	HWWDG_KILL	-	-	-	-	TC00_0	-	CH (old)	no	MCU control	-
AH2	570	474	FIX_ENX	-	-	-	-	TC00_0	-	CH (old)	no	MCU control	-
AG4	566	472	EDSU_BREAKX	-	-	-	-	TC01_0	Up	CH (old)	no	MCU control	-

AG1	567	471	ICLK	-	-	-	-	TE22_0	-	C	no	Tool	8mA
AH1	565	469	ICD_3	-	-	-	-	TE21_0	-	C	no	Tool	4mA
AH4	564	470	ICD_2	-	-	-	-	TE21_0	-	C	no	Tool	4mA
AG3	563	467	ICD_1	-	-	-	-	TE21_0	-	C	no	Tool	4mA
AH3	561	465	ICD_0	-	-	-	-	TE21_0	-	C	no	Tool	4mA
AJ2	560	466	ICS_2	-	-	-	-	TE10_0	-	-	no	Tool	4mA
AK2	558	464	ICS_1	-	-	-	-	TE10_0	-	-	no	Tool	4mA
AJ1	557	462	ICS_0	-	-	-	-	TE10_0	-	-	no	Tool	4mA
AK1	555	460	BREAK	-	-	-	-	TE01_0	-	C	no	Tool	-
AJ4	554	461	PLEVEL	-	-	-	-	TE02_0	-	C	no	Tool	-
AJ3	553	458	EXRAM	-	-	-	-	TE00_0	-	C	no	Tool	-
AK4	552	459	TRSTX	-	-	-	-	TE03_0	-	C	no	Tool	-
AK3	551	457	TCLK	-	-	-	-	TE11_0	-	-	no	Tool	8mA
AL2	548	456	TOEX	-	-	-	-	TE10_0	-	-	no	Tool	4mA
AM2	546	454	TWRX	-	-	-	-	TE10_0	-	-	no	Tool	4mA
AL1	545	453	TCE1X	-	-	-	-	TE10_0	-	-	no	Tool	4mA
AM1	543	450	TADSCX	-	-	-	-	TE10_0	-	-	no	Tool	4mA
AL3	542	451	TAD_15	-	-	-	-	TE10_0	-	-	no	Tool	4mA
AN1	541	448	TAD_14	-	-	-	-	TE10_0	-	-	no	Tool	4mA
AM4	540	449	TAD_13	-	-	-	-	TE10_0	-	-	no	Tool	4mA
AP1	539	446	TAD_12	-	-	-	-	TE10_0	-	-	no	Tool	4mA
AN2	538	447	TAD_11	-	-	-	-	TE10_0	-	-	no	Tool	4mA
AP2	536	445	TAD_10	-	-	-	-	TE10_0	-	-	no	Tool	4mA
AM3	535	444	TAD_9	-	-	-	-	TE10_0	-	-	no	Tool	4mA
AN3	533	443	TAD_8	-	-	-	-	TE10_0	-	-	no	Tool	4mA
AN4	532	442	TAD_7	-	-	-	-	TE10_0	-	-	no	Tool	4mA
AR1	531	441	TAD_6	-	-	-	-	TE10_0	-	-	no	Tool	4mA
AP4	530	440	TAD_5	-	-	-	-	TE10_0	-	-	no	Tool	4mA
AT1	529	439	TAD_4	-	-	-	-	TE10_0	-	-	no	Tool	4mA
AR2	524	436	TAD_3	-	-	-	-	TE10_0	-	-	no	Tool	4mA
AP3	523	435	TAD_2	-	-	-	-	TE10_0	-	-	no	Tool	4mA
AT2	522	434	TAD_1	-	-	-	-	TE10_0	-	-	no	Tool	4mA
AR3	521	433	TAD_0	-	-	-	-	TE10_0	-	-	no	Tool	4mA
AU3	517	432	TDT_68	-	-	-	-	TE20_0	-	C	no	Tool	4mA
AT4	516	430	TDT_67	-	-	-	-	TE20_0	-	C	no	Tool	4mA
AU4	515	431	TDT_66	-	-	-	-	TE20_0	-	C	no	Tool	4mA
AT5	514	428	TDT_65	-	-	-	-	TE20_0	-	C	no	Tool	4mA
AV3	512	426	TDT_64	-	-	-	-	TE20_0	-	C	no	Tool	4mA
AR5	511	427	TDT_63	-	-	-	-	TE20_0	-	C	no	Tool	4mA
AV4	510	424	TDT_62	-	-	-	-	TE20_0	-	C	no	Tool	4mA
AR6	509	425	TDT_61	-	-	-	-	TE20_0	-	C	no	Tool	4mA
AU5	507	423	TDT_60	-	-	-	-	TE20_0	-	C	no	Tool	4mA
AT6	506	421	TDT_59	-	-	-	-	TE20_0	-	C	no	Tool	4mA
AU6	505	422	TDT_58	-	-	-	-	TE20_0	-	C	no	Tool	4mA
AT7	504	419	TDT_57	-	-	-	-	TE20_0	-	C	no	Tool	4mA

AV5	502	417	TDT_56	-	-	-	-	TE20_0	-	C	no	Tool	4mA
AU7	501	420	TDT_55	-	-	-	-	TE20_0	-	C	no	Tool	4mA
AV6	500	416	TDT_54	-	-	-	-	TE20_0	-	C	no	Tool	4mA
AU8	499	418	TDT_53	-	-	-	-	TE20_0	-	C	no	Tool	4mA
AR8	495	415	TDT_52	-	-	-	-	TE20_0	-	C	no	Tool	4mA
AT8	494	412	TDT_51	-	-	-	-	TE20_0	-	C	no	Tool	4mA
AR9	493	413	TDT_50	-	-	-	-	TE20_0	-	C	no	Tool	4mA
AT9	492	410	TDT_49	-	-	-	-	TE20_0	-	C	no	Tool	4mA
AV7	490	408	TDT_48	-	-	-	-	TE20_0	-	C	no	Tool	4mA
AU9	489	409	TDT_47	-	-	-	-	TE20_0	-	C	no	Tool	4mA
AV8	488	406	TDT_46	-	-	-	-	TE20_0	-	C	no	Tool	4mA
AU10	487	407	TDT_45	-	-	-	-	TE20_0	-	C	no	Tool	4mA
AT10	484	404	TDT_44	-	-	-	-	TE20_0	-	C	no	Tool	4mA
AR10	483	405	TDT_43	-	-	-	-	TE20_0	-	C	no	Tool	4mA
AT11	482	402	TDT_42	-	-	-	-	TE20_0	-	C	no	Tool	4mA
AR11	481	403	TDT_41	-	-	-	-	TE20_0	-	C	no	Tool	4mA
AV9	480	401	TDT_40	-	-	-	-	TE20_0	-	C	no	Tool	4mA
AV10	478	400	TDT_39	-	-	-	-	TE20_0	-	C	no	Tool	4mA
AU11	477	399	TDT_38	-	-	-	-	TE20_0	-	C	no	Tool	4mA
AU12	475	398	TDT_37	-	-	-	-	TE20_0	-	C	no	Tool	4mA
AR12	473	396	TDT_36	-	-	-	-	TE20_0	-	C	no	Tool	4mA
AT12	472	395	TDT_35	-	-	-	-	TE20_0	-	C	no	Tool	4mA
AR13	471	394	TDT_34	-	-	-	-	TE20_0	-	C	no	Tool	4mA
AT13	470	392	TDT_33	-	-	-	-	TE20_0	-	C	no	Tool	4mA
AV11	468	390	TDT_32	-	-	-	-	TE20_0	-	C	no	Tool	4mA
AU13	467	391	TDT_31	-	-	-	-	TE20_0	-	C	no	Tool	4mA
AV12	466	388	TDT_30	-	-	-	-	TE20_0	-	C	no	Tool	4mA
AU14	465	389	TDT_29	-	-	-	-	TE20_0	-	C	no	Tool	4mA
AT14	462	386	TDT_28	-	-	-	-	TE20_0	-	C	no	Tool	4mA
AU15	461	385	TDT_27	-	-	-	-	TE20_0	-	C	no	Tool	4mA
AT15	460	383	TDT_26	-	-	-	-	TE20_0	-	C	no	Tool	4mA
AU16	459	384	TDT_25	-	-	-	-	TE20_0	-	C	no	Tool	4mA
AV13	458	382	TDT_24	-	-	-	-	TE20_0	-	C	no	Tool	4mA
AV14	456	381	TDT_23	-	-	-	-	TE20_0	-	C	no	Tool	4mA
AR15	455	380	TDT_22	-	-	-	-	TE20_0	-	C	no	Tool	4mA
AR16	453	379	TDT_21	-	-	-	-	TE20_0	-	C	no	Tool	4mA
AU17	451	377	TDT_20	-	-	-	-	TE20_0	-	C	no	Tool	4mA
AT16	450	376	TDT_19	-	-	-	-	TE20_0	-	C	no	Tool	4mA
AU18	449	375	TDT_18	-	-	-	-	TE20_0	-	C	no	Tool	4mA
AT17	448	374	TDT_17	-	-	-	-	TE20_0	-	C	no	Tool	4mA
AV15	446	371	TDT_16	-	-	-	-	TE20_0	-	C	no	Tool	4mA
AR17	445	372	TDT_15	-	-	-	-	TE20_0	-	C	no	Tool	4mA
AV16	444	369	TDT_14	-	-	-	-	TE20_0	-	C	no	Tool	4mA
AT18	443	370	TDT_13	-	-	-	-	TE20_0	-	C	no	Tool	4mA
AV17	440	368	TDT_12	-	-	-	-	TE20_0	-	C	no	Tool	4mA

AR18	439	367	TDT_11	-	-	-	-	TE20_0	-	C	no	Tool	4mA
AV18	438	366	TDT_10	-	-	-	-	TE20_0	-	C	no	Tool	4mA
AR19	437	365	TDT_9	-	-	-	-	TE20_0	-	C	no	Tool	4mA
AU19	436	364	TDT_8	-	-	-	-	TE20_0	-	C	no	Tool	4mA
AT19	435	363	TDT_7	-	-	-	-	TE20_0	-	C	no	Tool	4mA
AV19	434	362	TDT_6	-	-	-	-	TE20_0	-	C	no	Tool	4mA
AT20	433	361	TDT_5	-	-	-	-	TE20_0	-	C	no	Tool	4mA
AV20	432	360	TDT_4	-	-	-	-	TE20_0	-	C	no	Tool	4mA
AR20	431	359	TDT_3	-	-	-	-	TE20_0	-	C	no	Tool	4mA
AU20	430	358	TDT_2	-	-	-	-	TE20_0	-	C	no	Tool	4mA
AR21	429	357	TDT_1	-	-	-	-	TE20_0	-	C	no	Tool	4mA
AV21	428	356	TDT_0	-	-	-	-	TE20_0	-	C	no	Tool	4mA
AV22	426	354	EMRAM	-	-	-	-	TE00_0	-	C	no	Tool	-
AU21	425	353	ECSX	-	-	-	-	TE10_0	-	-	no	Tool	4mA
AT21	424	351	EWRX_3	-	-	-	-	TE10_0	-	-	no	Tool	4mA
AU22	423	352	EWRX_2	-	-	-	-	TE10_0	-	-	no	Tool	4mA
AT22	422	349	EWRX_1	-	-	-	-	TE10_0	-	-	no	Tool	4mA
AR22	421	350	EWRX_0	-	-	-	-	TE10_0	-	-	no	Tool	4mA
AR23	419	348	EECSX	-	-	-	-	TE10_0	-	-	no	Tool	4mA
AV23	416	346	EEWEX	-	-	-	-	TE10_0	-	-	no	Tool	4mA
AU23	415	345	EEOEX	-	-	-	-	TE10_0	-	-	no	Tool	4mA
AV24	414	344	EEBEX_3	-	-	-	-	TE10_0	-	-	no	Tool	4mA
AU24	413	343	EEBEX_2	-	-	-	-	TE10_0	-	-	no	Tool	4mA
AT23	412	342	EEBEX_1	-	-	-	-	TE10_0	-	-	no	Tool	4mA
AT24	410	340	EEBEX_0	-	-	-	-	TE10_0	-	-	no	Tool	4mA
AU25	409	339	EEA_20	-	-	-	-	TE10_0	-	-	no	Tool	4mA
AU26	407	338	EEA_19	-	-	-	-	TE10_0	-	-	no	Tool	4mA
AV25	406	336	EEA_18	-	-	-	-	TE10_0	-	-	no	Tool	4mA
AV26	404	335	EEA_17	-	-	-	-	TE10_0	-	-	no	Tool	4mA
AR25	403	334	EEA_16	-	-	-	-	TE10_0	-	-	no	Tool	4mA
AT25	402	333	EEA_15	-	-	-	-	TE10_0	-	-	no	Tool	4mA
AR26	401	332	EEA_14	-	-	-	-	TE10_0	-	-	no	Tool	4mA
AT26	400	330	EEA_13	-	-	-	-	TE10_0	-	-	no	Tool	4mA
AU27	399	331	EEA_12	-	-	-	-	TE10_0	-	-	no	Tool	4mA
AU28	397	329	EEA_11	-	-	-	-	TE10_0	-	-	no	Tool	4mA
AV27	394	328	EEA_10	-	-	-	-	TE10_0	-	-	no	Tool	4mA
AR27	393	327	EEA_9	-	-	-	-	TE10_0	-	-	no	Tool	4mA
AV28	392	326	EEA_8	-	-	-	-	TE10_0	-	-	no	Tool	4mA
AR28	391	325	EEA_7	-	-	-	-	TE10_0	-	-	no	Tool	4mA
AT27	390	324	EEA_6	-	-	-	-	TE10_0	-	-	no	Tool	4mA
AT28	388	322	EEA_5	-	-	-	-	TE10_0	-	-	no	Tool	4mA
AU29	387	321	EEA_4	-	-	-	-	TE10_0	-	-	no	Tool	4mA
AU30	385	320	EEA_3	-	-	-	-	TE10_0	-	-	no	Tool	4mA
AV29	384	319	EEA_2	-	-	-	-	TE10_0	-	-	no	Tool	4mA
AV30	382	317	EEA_1	-	-	-	-	TE10_0	-	-	no	Tool	4mA

AR29	381	316	EEA_0	-	-	-	-	TE10_0	-	-	no	Tool	4mA
AT29	380	315	EED_31	-	-	-	-	TE20_0	-	C	no	Tool	4mA
AR30	379	314	EED_30	-	-	-	-	TE20_0	-	C	no	Tool	4mA
AT30	378	313	EED_29	-	-	-	-	TE20_0	-	C	no	Tool	4mA
AU31	375	311	EED_28	-	-	-	-	TE20_0	-	C	no	Tool	4mA
AU32	373	310	EED_27	-	-	-	-	TE20_0	-	C	no	Tool	4mA
AV31	372	309	EED_26	-	-	-	-	TE20_0	-	C	no	Tool	4mA
AV32	370	307	EED_25	-	-	-	-	TE20_0	-	C	no	Tool	4mA
AT31	369	306	EED_24	-	-	-	-	TE20_0	-	C	no	Tool	4mA
AV33	368	305	EED_23	-	-	-	-	TE20_0	-	C	no	Tool	4mA
AR32	367	304	EED_22	-	-	-	-	TE20_0	-	C	no	Tool	4mA
AV34	366	303	EED_21	-	-	-	-	TE20_0	-	C	no	Tool	4mA
AU33	365	302	EED_20	-	-	-	-	TE20_0	-	C	no	Tool	4mA
AU34	363	300	EED_19	-	-	-	-	TE20_0	-	C	no	Tool	4mA
AT32	362	301	EED_18	-	-	-	-	TE20_0	-	C	no	Tool	4mA
AT33	360	299	EED_17	-	-	-	-	TE20_0	-	C	no	Tool	4mA
AR33	359	297	EED_16	-	-	-	-	TE20_0	-	C	no	Tool	4mA
AV35	358	298	EED_15	-	-	-	-	TE20_0	-	C	no	Tool	4mA
AR34	357	295	EED_14	-	-	-	-	TE20_0	-	C	no	Tool	4mA
AV36	356	296	EED_13	-	-	-	-	TE20_0	-	C	no	Tool	4mA
AU35	351	292	EED_12	-	-	-	-	TE20_0	-	C	no	Tool	4mA
AT34	350	291	EED_11	-	-	-	-	TE20_0	-	C	no	Tool	4mA
AU36	349	290	EED_10	-	-	-	-	TE20_0	-	C	no	Tool	4mA
AT35	348	289	EED_9	-	-	-	-	TE20_0	-	C	no	Tool	4mA
AT37	344	288	EED_8	-	-	-	-	TE20_0	-	C	no	Tool	4mA
AR36	343	287	EED_7	-	-	-	-	TE20_0	-	C	no	Tool	4mA
AR37	342	286	EED_6	-	-	-	-	TE20_0	-	C	no	Tool	4mA
AP36	341	285	EED_5	-	-	-	-	TE20_0	-	C	no	Tool	4mA
AT38	339	283	EED_4	-	-	-	-	TE20_0	-	C	no	Tool	4mA
AP35	338	282	EED_3	-	-	-	-	TE20_0	-	C	no	Tool	4mA
AR38	337	281	EED_2	-	-	-	-	TE20_0	-	C	no	Tool	4mA
AN35	336	280	EED_1	-	-	-	-	TE20_0	-	C	no	Tool	4mA
AP37	334	279	EED_0	-	-	-	-	TE20_0	-	C	no	Tool	4mA
AN36	333	278	IHIT_3	-	-	-	-	TE10_0	-	-	no	Tool	4mA
AN37	332	277	IHIT_2	-	-	-	-	TE10_0	-	-	no	Tool	4mA
AM36	331	276	IHIT_1	-	-	-	-	TE10_0	-	-	no	Tool	4mA
AP38	329	275	IHIT_0	-	-	-	-	TE10_0	-	-	no	Tool	4mA
AM37	328	274	FLASH_RDY	-	-	-	-	TE00_0	-	C	no	Tool	-
AN38	327	273	FLASH_ALE	-	-	-	-	TE10_0	-	-	no	Tool	4mA
AL37	326	272	FLASH_SYNC	-	-	-	-	TE10_0	-	-	no	Tool	4mA
AL35	322	269	FLASH_ATDIN	-	-	-	-	TE10_0	-	-	no	Tool	4mA
AL36	321	268	FLASH_EQIN	-	-	-	-	TE10_0	-	-	no	Tool	4mA
AK35	320	267	FLASH_RD32	-	-	-	-	TE10_0	-	-	no	Tool	4mA
AK36	319	266	FLASH_BYTEX	-	-	-	-	TE10_0	-	-	no	Tool	4mA
AM38	317	265	FLASH_SECUR	-	-	-	-	TE10_0	-	-	no	Tool	4mA

AK37	316	263	FLASH_FRSTX	-	-	-	-	TE10_0	-	-	no	Tool	4mA	
0.0	637	531	VDD5	-	-	-	-	TS02_0	-	-	-	VDD 5V	-	
0.0	615	513	VDD5	-	-	-	-	TS02_0	-	-	-	VDD 5V	-	
0.0	591	491	VDD5	-	-	-	-	TS02_0	-	-	-	VDD 5V	-	
0.0	569	473	VDD5	-	-	-	-	TS02_0	-	-	-	VDD 5V	-	
AR7	313	260	VDD5	-	-	-	-	TS02_0	-	-	-	VDD 5V	-	
AP11	291	242	VDD5	-	-	-	-	TS02_0	-	-	-	VDD 5V	-	
AP19	269	224	VDD5	-	-	-	-	TS02_0	-	-	-	VDD 5V	-	
AP23	245	201	VDD5	-	-	-	-	TS02_0	-	-	-	VDD 5V	-	
AP31	223	183	VDD5	-	-	-	-	TS02_0	-	-	-	VDD 5V	-	
AM35	201	166	VDD5	-	-	-	-	TS02_0	-	-	-	VDD 5V	-	
AH34	179	149	VDD5	-	-	-	-	TS02_0	-	-	-	VDD 5V	-	
AD34	172	144	VDD5	-	-	-	-	TS02_0	-	-	-	VDD 5V	-	
T34	152	126	VDD5	-	-	-	-	TS02_0	-	-	-	VDD 5V	-	
H34	130	109	VDD5	-	-	-	-	TS02_0	-	-	-	VDD 5V	-	
H35	118	98	VDD5	-	-	-	-	TS02_0	-	-	-	VDD 5V	-	
E32	96	81	VDD5	-	-	-	-	TS02_0	-	-	-	VDD 5V	-	
E28	72	59	VDD5	-	-	-	-	TS02_0	-	-	-	VDD 5V	-	
E20	50	40	VDD5	-	-	-	-	TS02_0	-	-	-	VDD 5V	-	
E12	28	22	VDD5	-	-	-	-	TS02_0	-	-	-	VDD 5V	-	
D8	6	5	VDD5	-	-	-	-	TS02_0	-	-	-	VDD 5V	-	
M5	642	537	VSS5	-	-	-	-	TS00_0	-	-	-	VSS	-	
T5	620	518	VSS5	-	-	-	-	TS00_0	-	-	-	VSS	-	
Y5	600	500	VSS5	-	-	-	-	TS00_0	-	-	-	VSS	-	
AD5	578	482	VSS5	-	-	-	-	TS00_0	-	-	-	VSS	-	
AJ34	306	254	VSS5	-	-	-	-	TS00_0	-	-	-	VSS	-	
AE34	284	235	VSS5	-	-	-	-	TS00_0	-	-	-	VSS	-	
AA34	268	223	VSS5	-	-	-	-	TS00_0	-	-	-	VSS	-	
W34	254	210	VSS5	-	-	-	-	TS00_0	-	-	-	VSS	-	
R34	232	191	VSS5	-	-	-	-	TS00_0	-	-	-	VSS	-	
L34	210	173	VSS5	-	-	-	-	TS00_0	-	-	-	VSS	-	
G34	188	155	VSS5	-	-	-	-	TS00_0	-	-	-	VSS	-	
B37	167	140	VSS5	-	-	-	-	TS00_0	-	-	-	VSS	-	
E31	145	121	VSS5	-	-	-	-	TS00_0	-	-	-	VSS	-	
E27	123	103	VSS5	-	-	-	-	TS00_0	-	-	-	VSS	-	
E23	101	84	VSS5	-	-	-	-	TS00_0	-	-	-	VSS	-	
E19	81	66	VSS5	-	-	-	-	TS00_0	-	-	-	VSS	-	
E15	59	48	VSS5	-	-	-	-	TS00_0	-	-	-	VSS	-	
E11	37	30	VSS5	-	-	-	-	TS00_0	-	-	-	VSS	-	
C3	9	7	VSS5	-	-	-	-	TS00_0	-	-	-	VSS	-	
0.0	681	569	HVDD5	-	-	-	-	TS02_0	-	-	-	VDD 5V	-	
0.0	671	559	HVDD5	-	-	-	-	TS02_0	-	-	-	VDD 5V	-	

0.0	659	549	HVDD5	-	-	-	-	TS02_0	-	-	-	VDD 5V	-
B2	686	574	HVSS5	-	-	-	-	TS00_0	-	-	-	VSS	-
F5	676	564	HVSS5	-	-	-	-	TS00_0	-	-	-	VSS	-
H5	664	554	HVSS5	-	-	-	-	TS00_0	-	-	-	VSS	-
E9	25	21	AVSS	-	-	-	-	TA03_0	-	AVSS	-	AVSS	-
C8	23	19	AVRL	-	-	-	-	TA01_0	-	AVRH/L	-	AVRH/L	-
A7	24	18	AVRH5	-	-	-	-	TA01_0	-	AVRH/L	-	AVRH/L	-
D7	21	17	AVCC5	-	-	-	-	TA00_0	-	AVCC	-	AVCC	-
A6	22	16	AVRH3	-	-	-	-	TA01_0	-	AVRH/L	-	AVRH/L	-
B6	19	15	AVCC3	-	-	-	-	TA00_0	-	AVCC	-	AVCC	-
Y38	261	216	VDD5R	-	-	-	-	TA00_0	-	AVCC	-	AVCC	-
W36	260	215	VDD5R	-	-	-	-	TA00_0	-	AVCC	-	AVCC	-
W38	259	214	VDD5R	-	-	-	-	TA00_0	-	AVCC	-	AVCC	-
W35	258	213	VDD5R	-	-	-	-	TA00_0	-	AVCC	-	AVCC	-
AA38	265	220	VCC3C	-	-	-	-	TA10_0	-	ANA OUT	-	VCC3C	-
Y35	264	219	VCC3C	-	-	-	-	TA10_0	-	ANA OUT	-	VCC3C	-
Y37	263	218	VCC3C	-	-	-	-	TA10_0	-	ANA OUT	-	VCC3C	-
Y36	262	217	VCC3C	-	-	-	-	TA10_0	-	ANA OUT	-	VCC3C	-
0.0	562	468	VDD3	-	-	-	-	TS01_0	-	-	-	VDD	-
0.0	549	455	VDD3	-	-	-	-	TS01_0	-	-	-	VDD	-
0.0	527	437	VDD3	-	-	-	-	TS01_0	-	-	-	VDD	-
0.0	497	414	VDD3	-	-	-	-	TS01_0	-	-	-	VDD	-
0.0	474	397	VDD3	-	-	-	-	TS01_0	-	-	-	VDD	-
0.0	463	387	VDD3	-	-	-	-	TS01_0	-	-	-	VDD	-
0.0	452	378	VDD3	-	-	-	-	TS01_0	-	-	-	VDD	-
J5	420	347	VDD3	-	-	-	-	TS01_0	-	-	-	VDD	-
N5	411	341	VDD3	-	-	-	-	TS01_0	-	-	-	VDD	-
U5	389	323	VDD3	-	-	-	-	TS01_0	-	-	-	VDD	-
AD4	376	312	VDD3	-	-	-	-	TS01_0	-	-	-	VDD	-
AE5	354	294	VDD3	-	-	-	-	TS01_0	-	-	-	VDD	-
AN5	324	271	VDD3	-	-	-	-	TS01_0	-	-	-	VDD	-
AP9	323	270	VDD3	-	-	-	-	TS01_0	-	-	-	VDD	-
AH5	556	463	VSS3	-	-	-	-	TS00_0	-	-	-	VSS	-
AK5	544	452	VSS3	-	-	-	-	TS00_0	-	-	-	VSS	-
AT3	528	438	VSS3	-	-	-	-	TS00_0	-	-	-	VSS	-
AU2	513	429	VSS3	-	-	-	-	TS00_0	-	-	-	VSS	-
AP8	491	411	VSS3	-	-	-	-	TS00_0	-	-	-	VSS	-
AP12	469	393	VSS3	-	-	-	-	TS00_0	-	-	-	VSS	-
AP16	447	373	VSS3	-	-	-	-	TS00_0	-	-	-	VSS	-

AP20	427	355	VSS3	-	-	-	-	TS00_0	-	-	-	VSS	-
AP24	405	337	VSS3	-	-	-	-	TS00_0	-	-	-	VSS	-
AP28	383	318	VSS3	-	-	-	-	TS00_0	-	-	-	VSS	-
AP30	371	308	VSS3	-	-	-	-	TS00_0	-	-	-	VSS	-
AT36	355	293	VSS3	-	-	-	-	TS00_0	-	-	-	VSS	-
AU37	340	284	VSS3	-	-	-	-	TS00_0	-	-	-	VSS	-
AL34	318	264	VSS3	-	-	-	-	TS00_0	-	-	-	VSS	-

7. I/O Circuit Type

The table below describes the circuit types which are used on the evaluation device MB91V460 Rev.A. Please refer to the datasheets for information about the circuit type of each pin used on the flash devices.

Type	Pull Up/ Pull Down	CMOS/ CMOS Hyst./ Automotive/ TTL	Input Stop	Usage	Output drive
TP00_0	Up / Dn	CH / A	yes	-	4 mA
TP01_0	-	CH / A	yes	A/D in / D/A out	4 mA
TP02_0	-	CH / A	yes	I2C	3 mA
TP03_0	Up / Dn	CH / A	yes	A/D in	4 mA
TP04_0	Up / Dn	CH / A / TTL	yes	External Bus	4 mA
TP05_0	-	CH / A	yes	SMC / A/D in	30 mA
TP06_0	-	CH / A	yes	LCD COM/SEG	4 mA
TP07_0	-	CH / A	yes	LCD V0/V1/V2	4 mA
TP08_0	-	CH / A	yes	LCD V3	4 mA
TC00_0	-	CH	no	MCU control	-
TC01_0	Up	CH	no	MCU control	-
TC02_0	Up	CH	no	MCU control	-
TC02_1	-	CH	no	MD0/MD1/MD2	-
TC10_0	-	-	no	MONCLK	8 mA
TA00_0	-	-	-	AVCC	-
TA01_0	-	-	-	AVRH/AVRL	-
TA02_0	-	-	-	A/D in	-
TA03_0	-	-	-	AVSS	-
TA10_0	-	-	-	VDD 3V	-
TO00_0	-	-	yes	X0	-
TO00_1	-	-	yes	X1	-
TO01_0	-	-	yes	X0A	-
TO01_1	-	-	yes	X1A	-
TE00_0	-	C	no	Tool	-
TE01_0	Dn (ctrl)	C	no	Tool	-
TE02_0	Up	C	no	Tool	-
TE03_0	Up	C	no	Tool	-
TE10_0	-	-	no	Tool	4 mA

TE11_0	-	-	no	Tool	8 mA
TE20_0	-	C	no	Tool	4 mA
TE21_0	Dn (ctrl)	C	no	Tool	4 mA
TE22_0	-	C	no	Tool	8 mA
TS00_0	-	-	-	VSS	
TS01_0	-	-	-	VDD	
TS02_0	-	-	-	VDD 5V	

8. Pin State Table

Explanation of the meaning of words and phrases used in the pin state table according to the chosen mode.

- Input enable: It is possible to input a signal to the device.
- Input fixed: To prevent leakage by floating inputs, the input level is fixed to "0" internally.
- Hi-Z: The pin is put in a high impedance state.
- State hold: The state (input/output) of the pin immediately before changing the state is maintained. In case of output, the output value of the pin is maintained.

Table 8-1 Pin State Table

								Device State					
JEDEC	Pin (INNER)	Pad	Port Name	Function	PFR=1	EPFR=1	Special	at Reset (INIT, RST)	after Reset and Vector Fetch	at SLEEP	at STOP HIZ = 0	at STOP HIZ = 1	Remarks
AL38	315	262	P00_7	D31	D31	-	-	Output Hi-Z, Input enabled	Output Hi-Z, Input enabled	State hold	Output: State hold, Input fixed	Output: HiZ, Input fixed	
AJ37	314	261	P00_6	D30	D30	-	-						
AJ36	311	259	P00_5	D29	D29	-	-						
AJ35	310	258	P00_4	D28	D28	-	-						
AH36	309	257	P00_3	D27	D27	-	-						
AH35	308	256	P00_2	D26	D26	-	-						
AK38	307	255	P00_1	D25	D25	-	-						
AJ38	305	253	P00_0	D24	D24	-	-						
AH37	304	252	P01_7	D23	D23	-	-	Output Hi-Z, Input enabled	Output Hi-Z, Input enabled	State hold	Output: State hold, Input fixed	Output: HiZ, Input fixed	
AG37	302	251	P01_6	D22	D22	-	-						
AG35	300	249	P01_5	D21	D21	-	-						
AG36	299	250	P01_4	D20	D20	-	-						
AF35	298	247	P01_3	D19	D19	-	-						
AF36	297	248	P01_2	D18	D18	-	-						
AH38	295	246	P01_1	D17	D17	-	-						
AF37	294	244	P01_0	D16	D16	-	-						
AG38	293	245	P02_7	D15	D15	-	-	Output Hi-Z, Input enabled	Output Hi-Z, Input enabled	State hold	Output: State hold, Input fixed	Output: HiZ, Input fixed	
AE37	292	243	P02_6	D14	D14	-	-						
AE36	289	241	P02_5	D13	D13	-	-						
AD37	288	240	P02_4	D12	D12	-	-						
AD36	287	239	P02_3	D11	D11	-	-						
AC37	286	238	P02_2	D10	D10	-	-						
AF38	285	237	P02_1	D9	D9	-	-						
AE38	283	236	P02_0	D8	D8	-	-						

AD35	282	234	P03_7	D7	D7	-	-	Output Hi-Z, Input enabled	Output Hi-Z, Input enabled	State hold	Output: State hold, Input fixed	Output: HiZ, Input fixed	
AC35	280	233	P03_6	D6	D6	-	-						
AB37	278	231	P03_5	D5	D5	-	-						
AC36	277	232	P03_4	D4	D4	-	-						
AA37	276	229	P03_3	D3	D3	-	-						
AB36	275	230	P03_2	D2	D2	-	-						
AD38	273	228	P03_1	D1	D1	-	-						
AC38	271	227	P03_0	D0	D0	-	-						
AB35	272	226	X1	-	-	-	-	Oscillator ON	Oscillator ON	Oscillator ON	Oscillator STOP if OSCD1 is set	Oscillator STOP if OSCD1 is set	
AA36	270	225	X0	-	-	-	-						
AB38	267	222	X1A	-	-	-	-	Oscillator ON	Oscillator ON	Oscillator ON	Oscillator STOP if OSCD2 is set	Oscillator STOP if OSCD2 is set	
AA35	266	221	X0A	-	-	-	-						
W37	257	211	MONCLK	-	-	-	-	Output Hi-Z	Output HiZ	State hold	State hold	State hold	
V35	256	212	P04_7	A31	A31	-	-	Output Hi-Z, Input enabled	Ext.Bus mode: Address Output Single Chip mode: Output Hi-Z, Input enabled	State hold	Output: State hold, Input fixed	Output: HiZ, Input fixed	Single Chip mode: Mode
V38	255	209	P04_6	A30	A30	-	-						
U38	253	207	P04_5	A29	A29	-	-						
V37	252	208	P04_4	A28	A28	-	-						
V36	251	205	P04_3	A27	A27	-	-						
U37	250	206	P04_2	A26	A26	-	-						
U36	249	203	P04_1	A25	A25	-	-						
U35	248	204	P04_0	A24	A24	-	-						
T35	246	202	P05_7	A23	A23	-	-	Output Hi-Z, Input enabled	Ext.Bus mode: Address Output Single Chip mode: Output Hi-Z, Input enabled	State hold	Output: State hold, Input fixed	Output: HiZ, Input fixed	
T38	243	200	P05_6	A22	A22	-	-						
T37	242	199	P05_5	A21	A21	-	-						
R38	241	198	P05_4	A20	A20	-	-						
R37	240	197	P05_3	A19	A19	-	-						
T36	239	196	P05_2	A18	A18	-	-						
R36	237	195	P05_1	A17	A17	-	-						
P37	236	194	P05_0	A16	A16	-	-						
N37	234	193	P06_7	A15	A15	-	-	Output Hi-Z, Input enabled	Ext.Bus mode: Address Output Single Chip mode: Output Hi-Z, Input enabled	State hold	Output: State hold, Input fixed	Output: HiZ, Input fixed	
P38	233	192	P06_6	A14	A14	-	-						
N38	231	190	P06_5	A13	A13	-	-						
P35	230	189	P06_4	A12	A12	-	-						
P36	229	188	P06_3	A11	A11	-	-						
N35	228	187	P06_2	A10	A10	-	-						
N36	227	186	P06_1	A9	A9	-	-						
M37	226	185	P06_0	A8	A8	-	-						
L37	224	184	P07_7	A7	A7	-	-	Output Hi-Z, Input enabled	Ext.Bus mode: Address Output Single Chip mode: Output Hi-Z, Input enabled	State hold	Output: State hold, Input fixed	Output: HiZ, Input fixed	
M38	221	182	P07_6	A6	A6	-	-						
M35	220	181	P07_5	A5	A5	-	-						
L38	219	180	P07_4	A4	A4	-	-						
L35	218	179	P07_3	A3	A3	-	-						
M36	217	178	P07_2	A2	A2	-	-						
L36	215	177	P07_1	A1	A1	-	-						
K37	214	176	P07_0	A0	A0	-	-						

J37	212	175	P08_7	RDY	RDY	-	-	Output Hi-Z, Input enabled	Output Hi-Z, Input enabled	State hold	Output: State hold, Input fixed	Output: HiZ, Input fixed	
K38	211	174	P08_6	BRQ	BRQ	-	-						
J38	209	172	P08_5	BGRNTX	BGRNTX	-	-		Ext.Bus mode: Control Output				
K35	208	171	P08_4	RDX	RDX	-	-						
K36	207	170	P08_3	WRX3	WRX3	-	-		Single Chip mode: Output Hi-Z, Input enabled				
J35	206	169	P08_2	WRX2	WRX2	-	-						
J36	205	168	P08_1	WRX1	WRX1	-	-						
H37	202	167	P08_0	WRX0	WRX0	-	-						
G37	200	165	P09_7	CSX7	CSX7	-	-	Output Hi-Z, Input enabled	Ext.Bus mode: Chip Select Output	State hold	Output: State hold, Input fixed	Output: HiZ, Input fixed	
H38	199	164	P09_6	CSX6	CSX6	-	-						
G38	197	163	P09_5	CSX5	CSX5	-	-						
H36	196	162	P09_4	CSX4	CSX4	-	-		Single Chip mode: Output Hi-Z, Input enabled				
F38	195	161	P09_3	CSX3	CSX3	-	-						
G35	194	160	P09_2	CSX2	CSX2	-	-						
E38	193	159	P09_1	CSX1	CSX1	-	-						
F37	192	158	P09_0	CSX0	CSX0	-	-						
E37	190	157	P10_7	-	-	-	-	Output Hi-Z, Input enabled	Output Hi-Z, Input enabled	State hold	Output: State hold, Input fixed	Output: HiZ, Input fixed	
G36	189	156	P10_6	MCLKE	MCLKE	^	-		Ext.Bus mode: MCLKE Output Single Chip mode: Output Hi-Z, Input enabled				
F36	187	154	P10_5	MCLKI	MCLKI	/MCLKI	-		Output Hi-Z, Input enabled				
F35	186	153	P10_4	MCLKO	MCLKO	/MCLKO	-		Ext.Bus mode: Control/ Clock Output				
D38	185	152	P10_3	WEX	WEX	^	-						
E35	184	151	P10_2	BAAX	BAAX	^	-		Single Chip mode: Output Hi-Z, Input enabled				
C38	183	150	P10_1	ASX	ASX	^	-						
D37	178	147	P10_0	SYSCLK	SYSCLK	/SYSCLK	-						
E36	177	148	P11_7	-	-	-	-	Output Hi-Z, Input enabled	Output Hi-Z, Input enabled	State hold	Output: State hold, Input fixed	Output: HiZ, Input fixed	
C37	176	145	P11_6	-	-	-	-						
D36	175	146	P11_5	-	-	-	-						
B36	171	143	P11_4	-	-	-	-						
C35	170	142	P11_3	-	-	-	-						
B35	169	141	P11_2	-	-	-	-						
C34	168	139	P11_1	IOWRX	IOWRX	-	-						
A36	166	138	P11_0	IORDX	IORDX	-	-						

D34	165	137	P12_7	DEOP3	DEOP3	-	-	Output Hi-Z, Input enabled	Output Hi-Z, Input enabled	State hold	Output: State hold, Input fixed	Output: HiZ, Input fixed	
A35	164	136	P12_6	DEOTX3	DEOTX3	DEOP3	-						
D33	163	135	P12_5	DACKX3	DACKX3	^	-						
B34	161	134	P12_4	DREQ3	DREQ3	^	-						
C33	160	133	P12_3	DEOP2	DEOP2	-	-						
B33	159	132	P12_2	DEOTX2	DEOTX2	DEOP2	-						
C32	158	131	P12_1	DACKX2	DACKX2	^	-						
A34	156	130	P12_0	DREQ2	DREQ2	^	-						
B32	155	129	P13_7	DEOP1	DEOP1	-	-	Output Hi-Z, Input enabled	Output Hi-Z, Input enabled	State hold	Output: State hold, Input fixed	Output: HiZ, Input fixed	
A33	154	128	P13_6	DEOTX1	DEOTX1	DEOP1	-						
B31	153	127	P13_5	DACKX1	DACKX1	^	-						
D31	149	125	P13_4	DREQ1	DREQ1	^	-						
C31	148	124	P13_3	DEOP0	DEOP0	-	-						
D30	147	123	P13_2	DEOTX0	DEOTX0	DEOP0	-						
C30	146	122	P13_1	DACKX0	DACKX0	^	-						
A32	144	120	P13_0	DREQ0	DREQ0	^	-						
B30	143	119	P14_7	-	ICU7/ TIN7	TIN7	TTG15/7	Output Hi-Z, Input enabled	Output Hi-Z, Input enabled	State hold	Output: State hold, Input fixed	Output: HiZ, Input fixed	
A31	142	118	P14_6	-	ICU6/ TIN6	TIN6	TTG14/6						
B29	141	117	P14_5	-	ICU5/ TIN5	TIN5	TTG13/5						
C29	138	116	P14_4	-	ICU4/ TIN4	TIN4	TTG12/4						
D29	137	115	P14_3	-	ICU3/ TIN3	TIN3	TTG11/3						
C28	136	114	P14_2	-	ICU2/ TIN2	TIN2	TTG10/2						
D28	135	113	P14_1	-	ICU1/ TIN1	TIN1	TTG9/1						
A30	134	112	P14_0	-	ICU0/ TIN0	TIN0	TTG8/0						
A29	132	111	P15_7	-	OCU7	TOT7	-	Output Hi-Z, Input enabled	Output Hi-Z, Input enabled	State hold	Output: State hold, Input fixed	Output: HiZ, Input fixed	
B28	131	110	P15_6	-	OCU6	TOT6	-						
B27	129	108	P15_5	-	OCU5	TOT5	-						
D27	127	107	P15_4	-	OCU4	TOT4	-						
C27	126	106	P15_3	-	OCU3	TOT3	-						
D26	125	105	P15_2	-	OCU2	TOT2	-						
C26	124	104	P15_1	-	OCU1	TOT1	-						
A28	122	102	P15_0	-	OCU0	TOT0	-						
B26	121	101	P16_7	-	PPG15	ATGX	-	Output Hi-Z, Input enabled	Output Hi-Z, Input enabled	State hold	Output: State hold, Input fixed	Output: HiZ, Input fixed	
A27	120	100	P16_6	-	PPG14	PFM	-						
B25	119	99	P16_5	-	PPG13	SGO	-						
C25	116	97	P16_4	-	PPG12	SGA	-						
B24	115	96	P16_3	-	PPG11	^	-						
C24	114	95	P16_2	-	PPG10	^	-						
B23	113	94	P16_1	-	PPG9	^	-						
A26	112	93	P16_0	-	PPG8	^	-						

A25	110	92	P17_7	-	PPG7	-	-	Output Hi-Z, Input enabled	Output Hi-Z, Input enabled	State hold	Output: State hold, Input fixed	Output: HiZ, Input fixed	
D24	109	91	P17_6	-	PPG6	-	-						
D23	107	90	P17_5	-	PPG5	-	-						
B22	105	88	P17_4	-	PPG4	-	-						
C23	104	89	P17_3	-	PPG3	-	-						
B21	103	86	P17_2	-	PPG2	-	-						
C22	102	87	P17_1	-	PPG1	-	-						
A24	100	85	P17_0	-	PPG0	-	-						
D22	99	82	P18_7	-	-	-	-	Output Hi-Z, Input enabled	Output Hi-Z, Input enabled	State hold	Output: State hold, Input fixed	Output: HiZ, Input fixed	
A23	98	83	P18_6	-	SCK7	ZIN3/ CK7	-						
C21	97	80	P18_5	-	SOT7	BIN3	-						
A22	94	79	P18_4	-	SIN7	AIN3	-						
D21	93	77	P18_3	-	-	-	-						
A21	92	78	P18_2	-	SCK6	ZIN2/ CK6	-						
D20	91	75	P18_1	-	SOT6	BIN2	-						
B20	90	76	P18_0	-	SIN6	AIN2	-						
C20	89	73	P19_7	-	-	-	-	Output Hi-Z, Input enabled	Output Hi-Z, Input enabled	State hold	Output: State hold, Input fixed	Output: HiZ, Input fixed	
A20	88	74	P19_6	-	SCK5	CK5	-						
C19	87	71	P19_5	-	SOT5	^	-						
A19	86	72	P19_4	-	SIN5	^	-						
D19	85	70	P19_3	-	-	-	-						
B19	84	69	P19_2	-	SCK4	CK4	-						
D18	83	68	P19_1	-	SOT4	^	-						
A18	82	67	P19_0	-	SIN4	^	-						
A17	80	65	P20_7	-	-	-	-	Output Hi-Z, Input enabled	Output Hi-Z, Input enabled	State hold	Output: State hold, Input fixed	Output: HiZ, Input fixed	
B18	79	64	P20_6	-	SCK3	ZIN1/ CK3	-						
C18	78	63	P20_5	-	SOT3	BIN1	-						
B17	77	62	P20_4	-	SIN3	AIN1	-						
C17	76	61	P20_3	-	-	-	-						
D17	75	60	P20_2	-	SCK2	ZIN0/ CK2	-						
D16	73	58	P20_1	-	SOT2	BIN0	-						
A16	70	57	P20_0	-	SIN2	AIN0	-						
B16	69	55	P21_7	-	-	-	-	Output Hi-Z, Input enabled	Output Hi-Z, Input enabled	State hold	Output: State hold, Input fixed	Output: HiZ, Input fixed	
A15	68	56	P21_6	-	SCK1	CK1	-						
B15	67	53	P21_5	-	SOT1	^	-						
C16	66	54	P21_4	-	SIN1	^	-						
C15	64	52	P21_3	-	-	-	-						
B14	63	51	P21_2	-	SCK0	CK0	-						
B13	61	49	P21_1	-	SOT0	^	-						
A14	60	50	P21_0	-	SIN0	^	-						

A13	58	47	P22_7	-	SCL1	-	-	Output Hi-Z, Input enabled	Output Hi-Z, Input enabled	State hold	Output: State hold, Input fixed	Output: HiZ, Input fixed	STOP: Input enabled if PFR is set (for Ext.INT)
D14	57	46	P22_6	-	SDA1	-	INT15						
C14	56	45	P22_5	-	SCL0	-	-						STOP: Input enabled if PFR is set (for Ext.INT)
D13	55	44	P22_4	-	SDA0	-	INT14						
C13	54	42	P22_3	-	TX5	-	-						STOP: Input enabled if PFR is set (for Ext.INT)
B12	53	43	P22_2	-	RX5	-	INT13						
B11	51	41	P22_1	-	TX4	-	-						STOP: Input enabled if PFR is set (for Ext.INT)
A12	48	39	P22_0	-	RX4	-	INT12						
D12	47	38	P23_7	-	TX3	-	-						STOP: Input enabled if PFR is set (for Ext.INT)
A11	46	37	P23_6	-	RX3	-	INT11						
D11	45	36	P23_5	-	TX2	-	-						STOP: Input enabled if PFR is set (for Ext.INT)
C12	44	35	P23_4	-	RX2	-	INT10						
C11	42	34	P23_3	-	TX1	-	-						STOP: Input enabled if PFR is set (for Ext.INT)
B10	41	33	P23_2	-	RX1	-	INT9						
B9	39	32	P23_1	-	TX0	-	-						STOP: Input enabled if PFR is set (for Ext.INT)
A10	38	31	P23_0	-	RX0	-	INT8						

A9	36	29	P24_7	-	INT7	-	SCL3	Output Hi-Z, Input enabled	Output Hi-Z, Input enabled	State hold	Output: State hold, Input fixed	Output: HiZ, Input fixed	STOP: Input enabled if PFR is set (for Ext.INT)
D10	35	28	P24_6	-	INT6	-	SDA3						
C10	34	26	P24_5	-	INT5	-	SCL2						
D9	33	27	P24_4	-	INT4	-	SDA2						
C9	32	25	P24_3	-	INT3	-	-						
B8	29	24	P24_2	-	INT2	-	-						
B7	27	23	P24_1	-	INT1	-	-						
A8	26	20	P24_0	-	INT0	-	-						
D2	688	576	P25_7	-	SMC2M5	-	-	Output Hi-Z, Input enabled	Output Hi-Z, Input enabled	State hold	Output: State hold, Input fixed	Output: HiZ, Input fixed	
E3	687	575	P25_6	-	SMC2P5	-	-						
C1	685	573	P25_5	-	SMC1M5	-	-						
E4	684	572	P25_4	-	SMC1P5	-	-						
D1	683	571	P25_3	-	SMC2M4	-	-						
F4	682	570	P25_2	-	SMC2P4	-	-						
E2	680	568	P25_1	-	SMC1M4	-	-						
F2	678	567	P25_0	-	SMC1P4	-	-						
F3	679	566	P26_7	-	SMC2M3	AN31	-	Output Hi-Z, Input enabled	Output Hi-Z, Input enabled	State hold	Output: State hold, Input fixed	Output: HiZ, Input fixed	
G3	677	565	P26_6	-	SMC2P3	AN30	-						
E1	675	562	P26_5	-	SMC1M3	AN29	-						
G2	674	563	P26_4	-	SMC1P3	AN28	-						
F1	673	560	P26_3	-	SMC2M2	AN27	-						
H2	672	561	P26_2	-	SMC2P2	AN26	-						
H4	668	558	P26_1	-	SMC1M2	AN25	-						
J4	666	557	P26_0	-	SMC1P2	AN24	-						
H3	667	556	P27_7	-	SMC2M1	AN23	-	Output Hi-Z, Input enabled	Output Hi-Z, Input enabled	State hold	Output: State hold, Input fixed	Output: HiZ, Input fixed	
J3	665	555	P27_6	-	SMC2P1	AN22	-						
G1	663	552	P27_5	-	SMC1M1	AN21	-						
J2	662	553	P27_4	-	SMC1P1	AN20	-						
H1	661	550	P27_3	-	SMC2M0	AN19	-						
K2	660	551	P27_2	-	SMC2P0	AN18	-						
K3	657	547	P27_1	-	SMC1M0	AN17	-						
K4	656	548	P27_0	-	SMC1P0	AN16	-						
L3	655	545	ALARM_1	-	-	-	-	analog input	analog input	State hold	State hold	State hold	
L4	654	546	ALARM_0	-	-	-	-						
A5	20	14	P28_7	-	AN15	-	DA1	Output Hi-Z, Input enabled	Output Hi-Z, Input enabled	State hold	Output: State hold, Input fixed	Output: HiZ, Input fixed	
B5	17	13	P28_6	-	AN14	-	DA0						
C7	16	12	P28_5	-	AN13	-	-						
C6	14	10	P28_4	-	AN12	-	-						
D6	13	11	P28_3	-	AN11	-	-						
A4	12	8	P28_2	-	AN10	-	-						
D5	11	9	P28_1	-	AN9	-	-						
A3	10	6	P28_0	-	AN8	-	-						

B4	5	4	P29_7	-	AN7	-	-	Output Hi-Z, Input enabled	Output Hi-Z, Input enabled	State hold	Output: State hold, Input fixed	Output: HiZ, Input fixed	
C5	4	3	P29_6	-	AN6	-	-						
B3	3	2	P29_5	-	AN5	-	-						
C4	2	1	P29_4	-	AN4	-	-						
C2	690	578	P29_3	-	AN3	-	-						
D3	689	577	P29_2	-	AN2	-	-						
J1	653	544	P29_1	-	AN1	-	-						
L2	650	543	P29_0	-	AN0	-	-						
K1	651	542	P30_7	-	V3	-	-	Output Hi-Z, Input enabled	Output Hi-Z, Input enabled	State hold	Output: State hold, Input fixed	Output: HiZ, Input fixed	
M2	648	541	P30_6	-	V2	-	-						
M4	646	540	P30_5	-	V1	-	-						
M3	645	538	P30_4	-	V0	-	-						
N4	644	539	P30_3	-	COM3	-	-						
N3	643	536	P30_2	-	COM2	-	-						
L1	641	534	P30_1	-	COM1	-	-						
N2	640	535	P30_0	-	COM0	-	-						
M1	639	532	P31_7	-	SEG39	-	-	Output Hi-Z, Input enabled	Output Hi-Z, Input enabled	State hold	Output: State hold, Input fixed	Output: HiZ, Input fixed	
P2	638	533	P31_6	-	SEG38	-	-						
P3	635	529	P31_5	-	SEG37	-	-						
R2	634	530	P31_4	-	SEG36	-	-						
R3	633	527	P31_3	-	SEG35	-	-						
T2	632	528	P31_2	-	SEG34	-	-						
N1	631	526	P31_1	-	SEG33	-	-						
P1	629	524	P31_0	-	SEG32	-	-						
R4	628	525	P32_7	-	SEG31	-	-	Output Hi-Z, Input enabled	Output Hi-Z, Input enabled	State hold	Output: State hold, Input fixed	Output: HiZ, Input fixed	
T4	626	523	P32_6	-	SEG30	SCK15	-						
U2	624	522	P32_5	-	SEG29	SOT15	-						
T3	623	521	P32_4	-	SEG28	SIN15	-						
V2	622	520	P32_3	-	SEG27	-	-						
U3	621	519	P32_2	-	SEG26	SCK14	-						
R1	619	517	P32_1	-	SEG25	SOT14	-						
U4	618	516	P32_0	-	SEG24	SIN14	-						
T1	617	515	P33_7	-	SEG23	-	-	Output Hi-Z, Input enabled	Output Hi-Z, Input enabled	State hold	Output: State hold, Input fixed	Output: HiZ, Input fixed	
V3	616	514	P33_6	-	SEG22	SCK13	-						
U1	613	512	P33_5	-	SEG21	SOT13	-						
V4	612	511	P33_4	-	SEG20	SIN13	-						
V1	611	510	P33_3	-	SEG19	-	-						
W4	610	509	P33_2	-	SEG18	SCK12	-						
W2	609	508	P33_1	-	SEG17	SOT12	-						
W3	608	507	P33_0	-	SEG16	SIN12	-						

W1	607	506	P34_7	-	SEG15	-	-	Output Hi-Z, Input enabled	Output Hi-Z, Input enabled	State hold	Output: State hold, Input fixed	Output: HiZ, Input fixed	
Y3	606	505	P34_6	-	SEG14	SCK11	-						
Y1	605	503	P34_5	-	SEG13	SOT11	-						
Y4	604	504	P34_4	-	SEG12	SIN11	-						
Y2	603	501	P34_3	-	SEG11	-	-						
AA4	602	502	P34_2	-	SEG10	SCK10	-						
AA1	601	499	P34_1	-	SEG9	SOT10	-						
AB1	599	497	P34_0	-	SEG8	SIN10	-						
AA2	598	498	P35_7	-	SEG7	-	-	Output Hi-Z, Input enabled	Output Hi-Z, Input enabled	State hold	Output: State hold, Input fixed	Output: HiZ, Input fixed	
AA3	597	495	P35_6	-	SEG6	SCK9	-						
AB2	596	496	P35_5	-	SEG5	SOT9	-						
AB3	595	493	P35_4	-	SEG4	SIN9	-						
AB4	594	494	P35_3	-	SEG3	-	-						
AC4	592	492	P35_2	-	SEG2	SCK8	-						
AC1	589	489	P35_1	-	SEG1	SOT8	-						
AC2	588	490	P35_0	-	SEG0	SIN8	-						
AD1	587	487	INITX	-	-	-	-	Input enabled					
AD2	586	488	RSTX	-	-	-	-						
AC3	585	486	HSTX	-	-	-	-						
AD3	583	484	NMIX	-	-	-	-						
AE2	582	485	MD_2	-	-	-	-	Input enabled					
AF2	580	483	MD_1	-	-	-	-						
AE1	579	481	MD_0	-	-	-	-						

Chapter 4 CPU Architecture

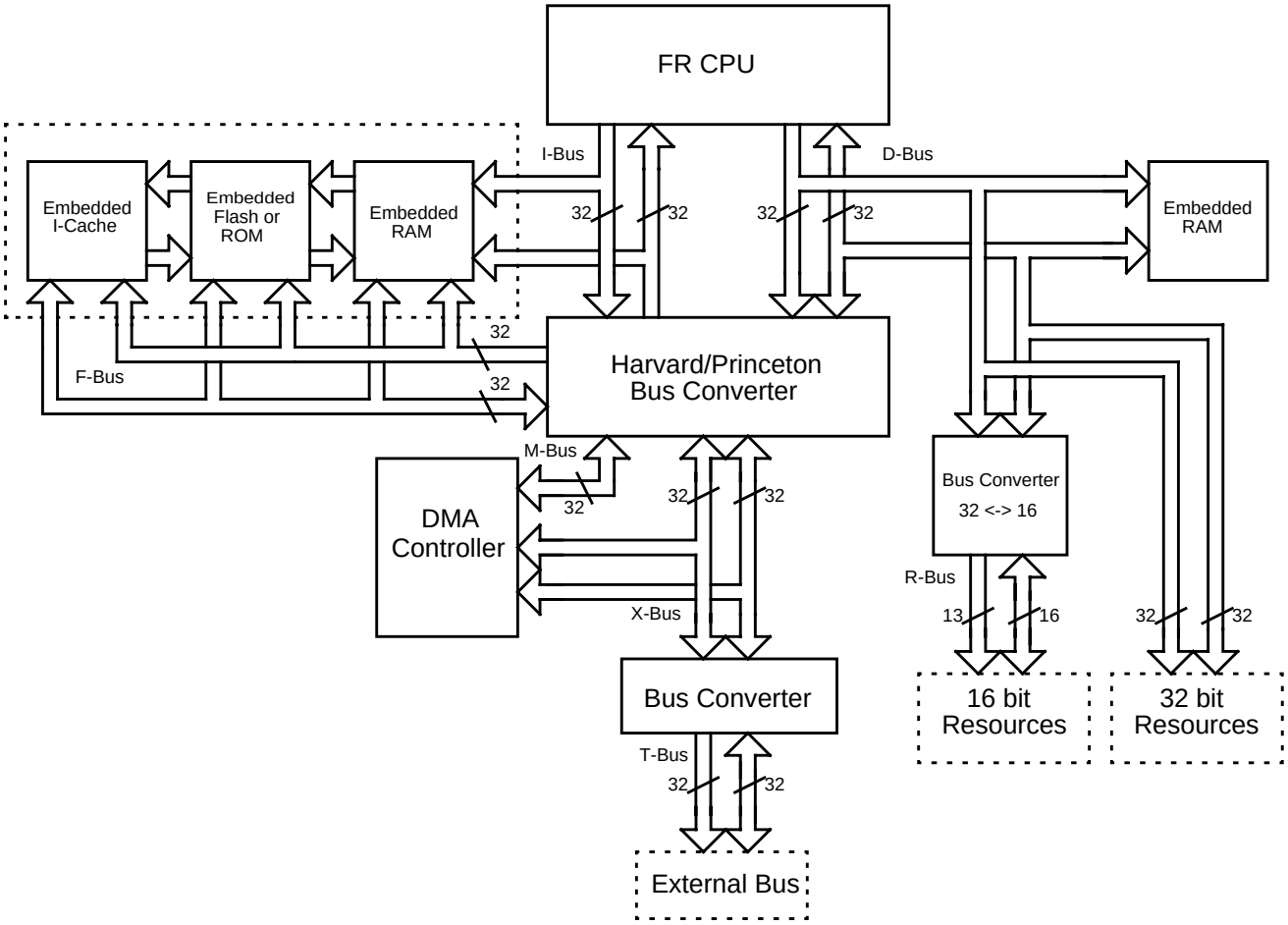
This chapter describes the architecture of FR60 family CPU.

1. Overview

The CPUs of the FR60 family series employ RISC architecture and advanced function instructions for embedded application.

CPU of FR60 family employs Harvard architecture whose instruction bus and data bus are independent. “32-bit/16-bit bus converter” realizes the interface between CPU and peripheral functions. “Harvard/Princeton bus converter” connects both of I-bus and D-bus and realizes the interface between CPU and bus controller.

Figure 1-1 Connection Diagram of Internal Architecture



2. Features

■ Features of internal architecture

- RISC architecture
- Base instruction: 1 instruction/1 cycle
- 32-bit architecture
- General-purpose register: 32-bit x 16
- 4GB of linear memory space
- Equipped with multiplier.
  - •32-bit x 32-bit multiplication: 5 cycles
  - •16-bit x 16-bit multiplication: 3 cycles
- Enhanced interrupt processing function
  - •High-speed respond (6 cycles)
  - •Support of multiple interrupts
  - •Level mask function (16 levels)
- Enhanced instruction for I/O operation
  - •Transfer instruction between memories
  - •Bit-processing instruction
- Highly efficient code
- Length of base instruction words: 16 bits
- Standby mode (Low power consumption mode)
- Sleep/Stop
- Setting function of clock division ratios

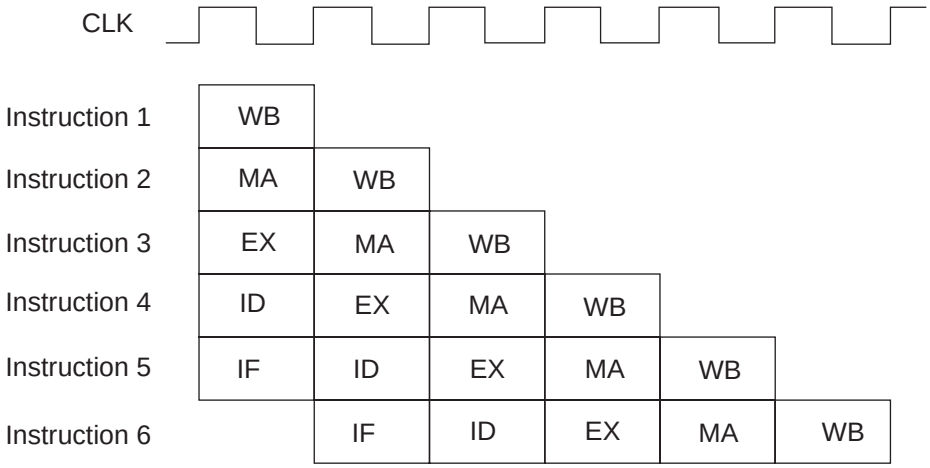
3. CPU

The CPU realizes the compact implementation of a 32-bit RISC FR architecture.

It employs a 5-stage instruction pipeline method to execute 1 instruction per 1 cycle.
This pipeline consists of the following stages.

- Instruction fetch (IF): outputs instruction address to fetch instruction.
- Instruction decode (ID): decodes fetched instruction and reads register.
- Execution (EX): executes operation.
- Memory access (MA): loads data for memory or accesses stored data.
- Write back to register (WB): writes back data to registers.

Figure 3-1 Instruction Pipeline



No instruction is executed in random order. If instruction A enters into pipeline before instruction B, instruction A always reaches to write-back stage before instruction B.

1 instruction is executed per 1 cycle.

However, to execute the instruction, multiple cycles are required for load/store instruction with memory wait, branch instruction without delay slot and multi-cycle instruction. In addition, a slow instruction degrades instruction execution speed.

4. 32-bit/16-bit Bus Converter

This converter generates the interface between D-bus which executes 32-bit high-speed access and R-bus which executes 16-bit access in order to realize data access from CPU to peripheral functions.

If 32-bit access comes from CPU, this converter converts the access into two 16-bit accesses to access to R bus. Some peripheral functions have restrictions of access width.

5. Harvard/Princeton Bus Converter

This converter realizes interface between instruction access and data access of CPU, to realize smooth interface with external bus.

The CPU employs the Harvard architecture whose instruction bus and data bus are independent while it employs single-bus Princeton architecture for bus controller to control external bus. This bus converter prioritizes instruction accesses and data accesses of CPU, and executes access control to bus controller. This always optimizes access sequence to external bus.

6. Instruction Overview

The FR60 family supports logic operation, bit operation and direct addressing instruction optimized for embedded application as well as general RISC instruction system. Instruction-set list is shown in the appendix. Since each instruction is 16-bit length (some instruction is 32-bit or 48-bit length), it enables you to generate compact program code.

Instruction sets are grouped into the following a through f function groups.

■ Arithmetic Operation

This group consists of standard arithmetic operation instruction (addition, subtraction and comparison) and shift instruction (logic shift and arithmetic shift). For addition and subtraction, the operation with carry used for multiple word length operation and the operation useful for address calculation without changing flag value are allowed.

In addition, it includes 32-bit x 32-bit and 16-bit x 16-bit multiplication instruction as well as 32-bit/32-bit step division instruction.

It provides transfer instruction of immediate value which sets immediate value to register, and transfer instruction between registers.

All arithmetic instructions are operated using general-purpose register and multiply & divide register within CPU.

■ Load and Store

Load/store is the instruction to read and write to memory. This is also used for read and write to peripheral functions (I/O) within chip.

Load and store consist of 3 type access lengths including byte, half-word and word. In addition to general register-indirect memory addressing, some instructions allow register-indirect memory addressing with displacement or with register increment/decrement.

■ Branch

This is the instruction for branch, call, interrupt and return. Branch instruction consists of instructions with and without delay slot. For more information of branch instruction, see “Chapter 7 Branch Instruction (Page No.129)”.

■ Logical Operation and Bit Operation

Logical operation instruction allows the logical operation of AND, OR and EOR between general-purpose registers or general-purpose register and memory (and I/O). Bit operation instruction allows the direct operation of data of memory (and I/O). Memory addressing is general register indirect.

■ Direct Addressing

Direct addressing instruction is the instruction to access between I/O and general-purpose register, or between I/O and memory. By directly instructing I/O address rather than register indirect, it enables high-speed and high-efficient access. Some instructions allow register-indirect memory addressing with register increment/decrement.

■ Others

This is the instruction which executes flag setting, stack operation, sign extension and zero extension within PS register. It provides the function entrance/exit which supports high-level language, and register multi-load/store instruction.

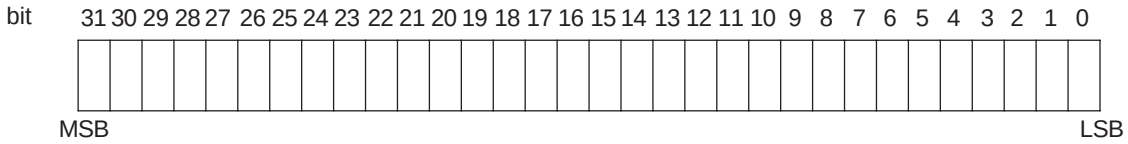
7. Data Structure

FR60 has two data allocations as follows.

■ Bit Ordering

FR60 uses little endian as bit ordering.

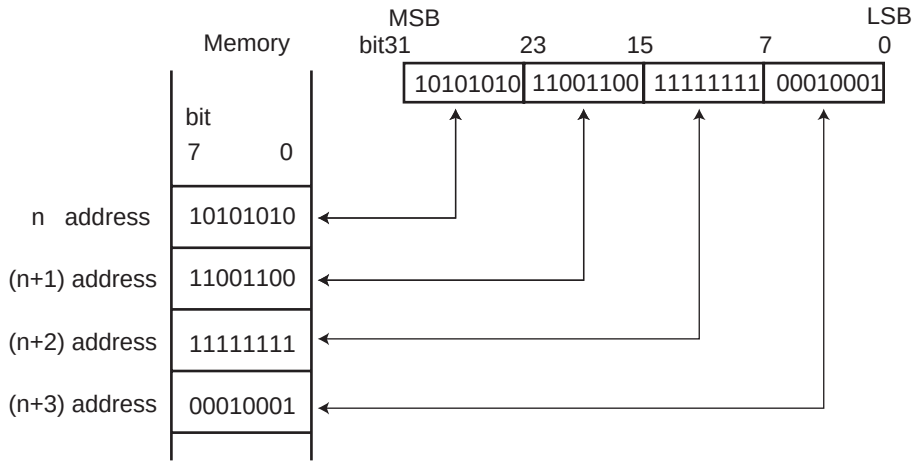
Figure 7-1 Bit Structure of Bit Ordering



■ Byte Ordering

FR60 uses big endian as byte ordering.

Figure 7-2 Bit Structure of Byte Ordering



8. Word Alignment

Since instructions and data are accessed by byte, allocated addresses vary by instruction length or data width.

■ Program Access

FR60 program is required to be allocated in addresses multiplied by 2.

PC's bit0 is cleared for instruction execution upon the PC update.

(PC bit 0 may be set when odd address is specified for branching address, however, it is invalid. Since the instruction is required to be allocated in addresses multiplied by 2, there is no odd address exception.)

■ Data Access

FR60 provides the following alignment for addresses depending on data width when executing data access.

- Word access : Address is multiplied by 4. (Lowest order 2 bits are mandatorily 00.)
- Half-word access : Address is multiplied by 2. (Lowest order bit is mandatorily 0.)
- Byte access: : Address is multiplied by 1.

  Upon the word and half-word data accesses, some bits mandatorily become 0 for computing results of effective address. For example, in the case of addressing mode of @(R13, Ri), register value before addition is used as is (even if lowest order bit is 1), and lower bits of addition results are masked. Register values before computing are not masked.

[Example] LD @(R13, R2), R0

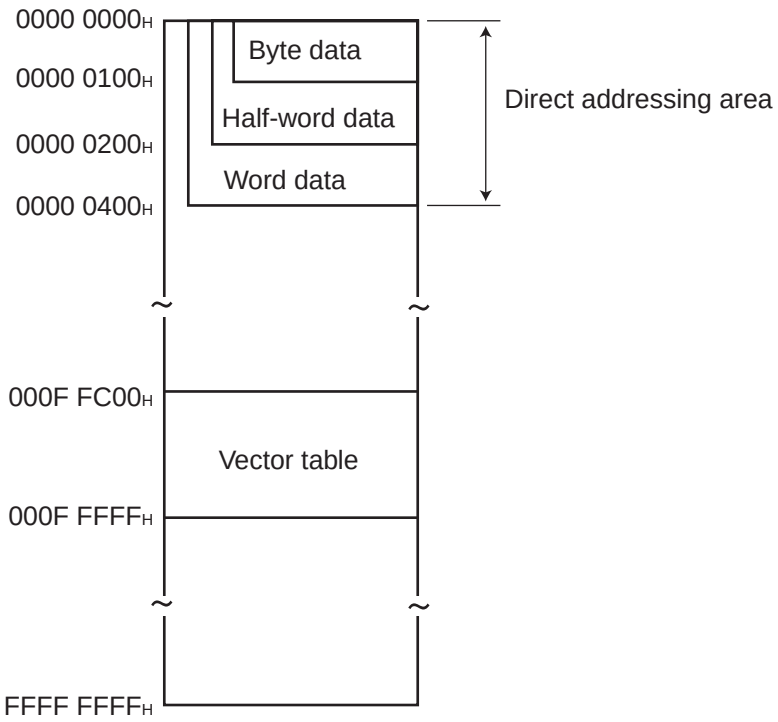
R13	00002222	
R2	00000003	
+)		
Addition results	00002225_H	
	↓	Lower 2 bits are mandatorily masked.
Address pin	00002224_H	

9. Addressing

Address space is 32-bit linear.

■ Map

Figure 9-1 Map



FR60's logical address space is 4GB (2^{32} addresses), CPU accesses the data linearly.

■ Direct Addressing Area

The following areas are used for I/O.

These spaces are referred to as direct addressing area where you can specify direct operand address by the instruction.

These direct areas vary by data size to be accessed.

- Byte access : 0 - $0FF_H$
- Half-word access : 0 - $1FF_H$
- Word access : 0 - $3FF_H$

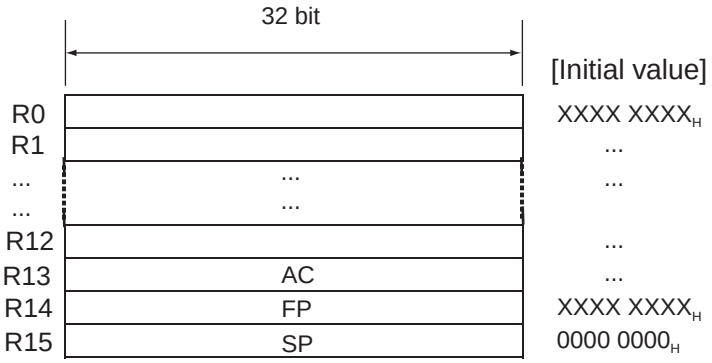
9.Addressing

Chapter 5 CPU Registers

1. General-purpose Registers

Registers R0 through R15 are general-purpose registers. These registers are used for accumulator and memory access pointers on various operations.

Figure 1-1 General-purpose Registers



Of 16 registers, the following registers are reserved for special application.

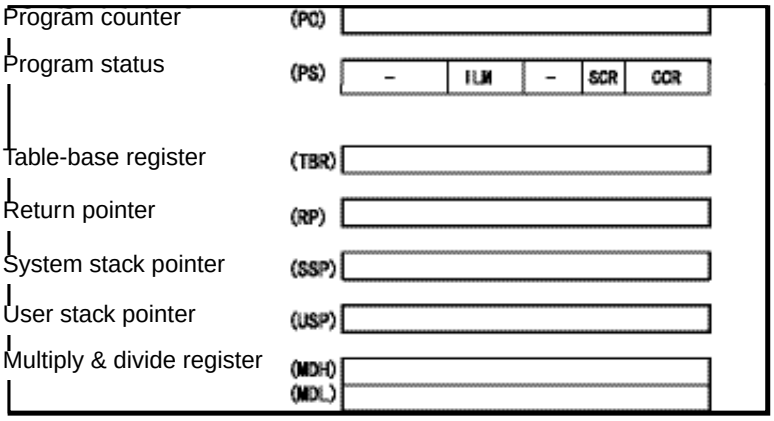
- R13: Virtual accumulator
- R14: Frame pointer
- R15: Stack pointer

Initial values by reset are indeterminate for R0 through R14. Initial value by reset is 00000000_H (SSP value) for R15.

2. Dedicated Registers

Dedicated registers consist of program counter (PC), program status (PS), table-base register (TBR), return pointer (RP), system stack pointer (SSP), user stack pointer (USP) and multiply & divide register (MDH/MDL).

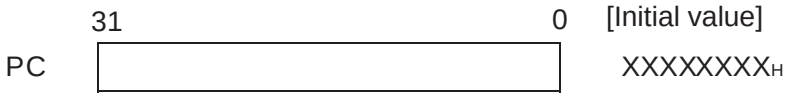
Figure 2-1 Dedicated Registers List



2.1 PC: Program Counter

Program Counter (PC) consists of 32 bits.

Figure 2-2 Bit Structure of Program Counter (PC)



Program counter (PC) indicates active instruction address.

Upon the execution of the instruction, program counter (PC)'s bit 0 is cleared.

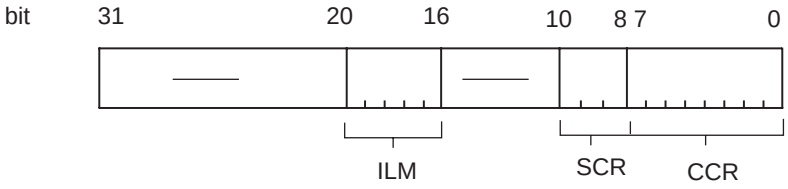
2.2 PS: Program Status Register

Program status register (PS) is the register to hold program status which consists of three parts including ILM, SCR and CCR.

All undefined bits are reserved bit. Upon the reading, "0" is always read. Writing is invalid.

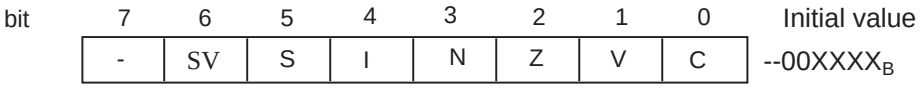
Program status register (PS) consists of condition code register (CCR), system condition code register (SCR) and interrupt level mask register (ILM).

Figure 2-3 Bit Structure of Program Status (PS)



■ CCR: Condition Code Register

Figure 2-4 Structure of Condition Code Register (CCR)



- [Bit 6] SV: SuperVisor flag

  This bit specifies supervisor mode of the CPU/MPU

SV	Description
0	CPU/MPU is in User Mode
1	CPU/MPU is in Superviser Mode

  This bit becomes "0" by reset.

  Signals SVMODE to the EDSU/MPU. Can be triggered by the INT #5 instruction (sets the SV flag and executes INT#5 ISR) or by using ORCCR to set, resp. ANDCCR to clear the SV-flag. Hardware behaviour by execution of INT #5 is same as executing other INT instructions except that the SV flag is set.

  For the complete overview about the Memory Protection Unit (MPU) please refer to chapter "MPU / EDSU" in the MB91V460 Hardware Manual.

- [Bit 5] S: Stack flag

  This bit specifies stack pointer.

S	Description
0	Uses R15 as SSP. Upon generating EIT, this bit automatically becomes "0". (Note that the value saved in stack is the value before clear.)

1	Uses R15 as USP.

This bit becomes “0” by reset.

After using R15 as USP, write “0” before executing RETI instruction.

- [Bit 4] I: Interrupt-enable flag

  This bit enables and disables user interrupt request.

I	Description
0	Disables user interrupt.
	Upon executing INT instruction, this bit becomes “0”.
	(Note that the value saved in stack is the value before clear.)
1	Enables user interrupt. Mask processing of user interrupt request is controlled by the value which is held in ILM.

This bit becomes “0” by reset.

- [Bit 3] N: Negative flag

  This bit indicates the sign when operation results is deemed as integer represented by two’s-complement numbers.

N	Description
0	It indicates that operation result is positive value.
1	It indicates that operation result is negative value.

- [Bit 2] Z: Zero flag

  It indicates whether operation result is 0 or not.

Z	Description
0	It indicates that operation result is other than 0.
1	It indicates that operation result is 0.

- [Bit 1] V: Overflow flag

  This bit deems that operand used for operation as integer represented by two’s-complement numbers, and indicates whether overflow was generated or not as the result of operation results.

V	Description
0	It indicates that overflow was not generated as the result of operation.
1	It indicates that overflow was generated as the result of operation.

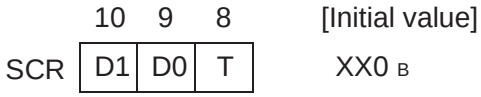
- [Bit 0] C: Carry flag

  This bit indicates whether carry or borrow from highest-order bit was generated or not as the result of operation.

Value	Description
0	It indicates that neither carry nor borrow is generated.
1	It indicates that either carry or borrow is generated.

■ SCR: System Condition Code Register

Figure 2-5 Structure of System Condition Code Register (SCR)



This section describes each bit structure of system condition code register (SCR).

- [Bit 10, 9] D1 and D0: Step division flag

  D1 and D0 bits hold intermediate data during the execution of step division.

  Do not modify data during the execution of division processing.

  If other processes is executed during the execution of step division, step division is assured to be restarted by saving

and returning PS register value.

Initial status by reset is indeterminate for D1 and D0 bits.

Upon executing DIV0S instruction, these bits are set by referring to dividend and divisor.

Upon executing DIV0U instruction, these bits mandatorily become “00”.

- [Bit 8] T: Step trace trap flag

  This bit is the flag to specify whether to enable step trace trap or not.

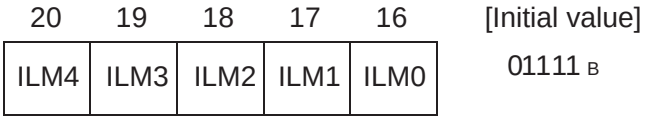
T value	Description
0	Disables step trace trap.
1	Enables step trace trap. In this case, all user interrupts are disabled.

This bit is initialized to “0” by reset.

The function of step trace trap is used for emulator. During the use of emulator, you cannot use this bit for user program.

■ ILM: Interrupt Level Mask Register

Figure 2-6 Register Structure of Interrupt Level Mask Register (ILM)



- This is the register to hold interrupt level mask value. This bit uses the value held in ILM as level mask.
- ILM indicates corresponding interrupt level from interrupt requests entered in CPU.
- Interrupt requests are accepted only if it’s priority is higher than the level.
- For level value, the highest priority is 0 (00000_B), and the lowest priority is 31 (11111_B).
- Program has some restrictions on configurable data.
  - When original value is between 16 and 31,

    Configurable new values are the value between 16 and 31.

    If you execute the instruction to set the value between 0 and 15, “specified value +16” value is set.
  - When original value is between 0 and 15,

    You can set any value between 0 and 31.

    These values are initialized to 15 (01111_B) by reset.

■ Caution: PS Register

Since some instructions have already processed PS register in advance, the following exception operations may break interrupt processing routine during the use of debugger, or update PS flag data.

In either cases, after returning from EIT, it is designed to execute the correct process so that operations before and after EIT will be processed in accordance with specification.

- **At instruction right before DIV0U/DIV0S instruction, the following 1. to 3. operation may be executed.**
  - If user interrupt is received,
  - If step execution is executed,
  - If data event or emulator menu is broken,

  1. D0 or D1 flag is updated in first.
  2. EIT processing routine (user interrupt or emulator) is executed.
  3. After returning from EIT, it executes DIV0U/DIV0S instruction and updates D0/D1 flag to the same value as 1.
- **When user interrupt is generated, if you execute each instruction of ORCCR, STILM, MOV Ri or PS to enable interrupt, the following operations are generated.**
  1. Updates PS register in first.
  2. Executes EIT processing routine (user interrupt).
  3. After returning from EIT, executes the instruction above and updates PS register to the same value as 1.

Note: For EIT, See “Chapter 6 EIT: Exceptions, Interrupts and Traps (Page No.121)”.

2.3 TBR: Table-base Register

Table-base register (TBR) consists of 32 bits.

Figure 2-7 Bit Structure of Table-base Register (TBR)

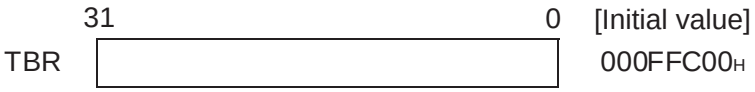
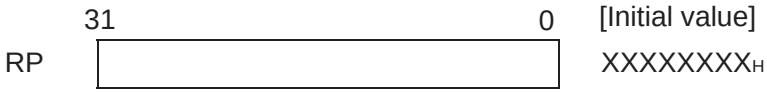


Table-base register holds head address of vector table used for EIT processes.
Vector address is made by adding offset value specified in TBR and EIT each.

2.4 RP: Return Pointer

Return pointer (RP: Return Pointer) consists of 32 bits.

Figure 2-8 Bit Structure of Return Pointer (RP)

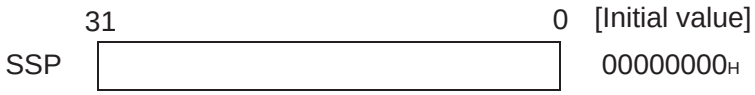


Return pointer (RP) holds addresses returned from sub routines.
Upon executing CALL, PC values are set in this RP.
Upon executing RET, RP data are set in this PC.

2.5 SSP: System Stack Pointer

System stack pointer (SSP) is used for the pointer which receives EIT and indicates stack to save/return data for return operation.
System stack pointer (SSP) consists of 32 bits.

Figure 2-9 Bit Structure of System Stack Pointer (SSP)

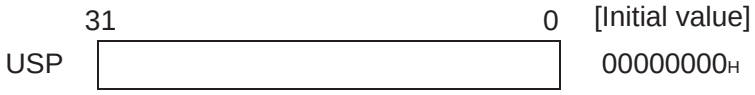


When S flag is “0”, it works as R15. You can explicitly specify SSP.
Upon generating EIT, it is used for the pointer which specifies the stack to save PS and PC.
During the EIT process, this pointer reduces the value by 8, and adds 8 to the value during the return from EIT by executing RETI instruction.
System stack pointer (SSP) works as general-purpose register R15 when S flag within CCR is “0”.

2.6 USP: User Stack Pointer

User Stack Pointer (USP) consists of 32 bits.

Figure 2-10 Bit Structure of User Stack Pointer (USP)



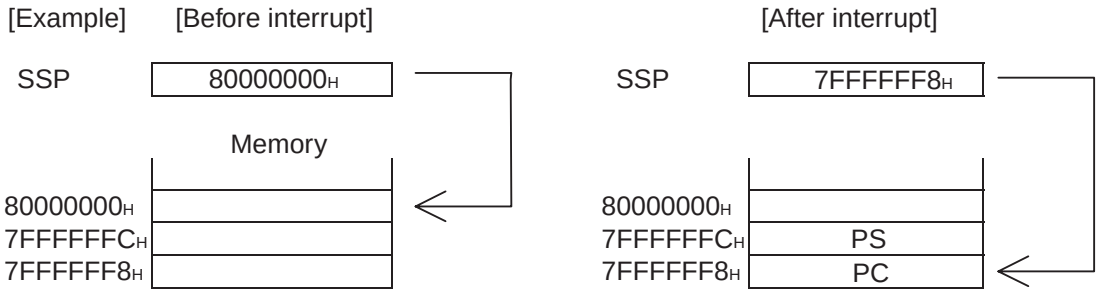
When S flag is “1”, this pointer works as R15.

You can explicitly specify USP.

You can not use it for RETI instruction.

This pointer saves and returns PC and PS values at the position where system stack pointer (SSP) indicates. After interrupt, it stores PC in address where SSP indicates, and PS in (SSP+4) address.

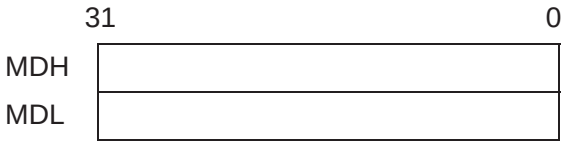
Figure 2-11 Interrupt Stack



2.7 MDH, MDL: Multiply & Divide Register

Multiply & Divide register (MDH/MDL) consists of 32 bits.

Figure 2-12 Bit Structure of Multiply & Divide Register (MDH/MDL)



This is the register for multiplication and division and consists of 32 bits.

Initial value by reset is indeterminate.

■ At the executing multiplication

When 32 bits x 32 bits multiplication, operation results of 64 bits are stored in multiplication/division store register as the following allocation.

- MDH: Upper 32 bits
- MDL: Lower 32 bits

When 16 bits x 16 bits multiplication, results are stored as follows.

- MDH: Indeterminate.
- MDL: Results of 32 bits

■ At the executing division

Upon starting operation, dividend is stored in MDL.

By computing division by executing DIV0S/DIV0U, DIV1, DIV2, DIV3, or DIV4S instruction, results are stored in MDL and MDH.

- MDH: Remainder
- MDL: Quotient

Chapter 6 EIT: Exceptions, Interrupts and Traps

1. Overview

EIT means that some events interrupt current program to execute other programs. EIT stands for Exception, Interrupt and Trap.

- **Exception is**

  the event which is generated in association with active context. It is returned to the instruction which triggered the exception.

- **Interrupt is**

  the event which is generated regardless of active context. Interrupt is triggered by hardware.

- **Trap is**

  the event which is generated in association with active context. Some trap is specified by program such as system call. It is returned to the instruction following the instruction which triggered the trap.

2. Features

- Supports multiple interrupts.
- Level mask function for interrupt (User can use 15 level.)
- Trap instruction (INT)
- EIT for emulator trigger (hardware/software)

3. EIT Trigger

- Reset
- User interrupt (peripheral function and external interrupt)
- Delayed interrupt
- Undefined-instruction exception
- Trap instruction (INT)
- Trap instruction (INTE)
- Step trace trap
- Coprocessor absent trap
- Coprocessor error trap
- CPU supervisor mode
- Memory protection violation

4. Return from EIT

To return from EIT, use RETI instruction.

5. EIT Interrupt Level

Interrupt level is between 0 and 31, and controlled with 5 bits.

Table 5-1 Interrupt Level of EIT

Level		Description	Remarks
Binary	Decimal		
00000	0	(Reserved for system)	
...	...	...	
...	...	...	
00011	3	(Reserved for system)	
00100	4	INTE instruction Step trace trap	If original value of ILM is between 16 and 31, these values are not configurable to ILM by program.
00101	5	(Reserved for system)	
...	...	...	
...	...	...	
01110	14	(Reserved for system)	
01111	15	NMI (for users)	
10000	16	Interrupt	When ILM is set, user Interrupt is disabled.
10001	17	Interrupt	
...	...	...	
...	...	...	
11110	30	Interrupt	
11111	31	N/A	When ICR is set, Interrupt is disabled.

Only 16 through 31 levels are operable.

Undefined-instruction exception, coprocessor absent trap, coprocessor error trap and INT instruction are not affected by interrupt level. Also, ILM is not changed by interrupt level.

6. EIT Vector Table

For EIT vector table, see the chapter of “3. Interrupt Vector Table (Page No.73)”.

Vector for EIT is between address which table-base register [TBR] indicates and a 1 kByte area.

Its size is 4 bytes per one vector. For vector number/vector address/trigger, see “3. Interrupt Vector Table (Page No.73)”.

Address arithmetic is as follows.

Vector address = [TBR] + Offset value = [TBR] + {$03FC_H$ - 4 x Vector number (No.)}

Lower two bits as the result of addition are always used for “00”.

$000FFC00_H$ through $000FFFFF_H$ areas are initial values of vector table by reset.

If you rewrite the TBR value, the mode and reset vectors always use the fixed address of $000FFFF8_H$, $000FFFFC_H$.

7. Multiple EIT Processing

If multiple EITs are generated at the same time, CPU repeats the operation which selects one of the EIT to accept, and then executes EIT sequence, and detects EIT again. If there is no EIT to accept upon detecting EIT, CPU executes instruction of the last accepted EIT handler. Therefore, if multiple EITs are generated at the same time, execution sequence of each EIT handler is determined by the following two parameters.

■ Priority Level of Receipt of EIT Triggers

Priority level of receipt of EIT triggers means the sequence to select which EIT triggers to execute by saving PS and PC in order to update PC and masking other triggers where appropriate.

EIT does not always mean first-in first-out handler.

Table 7-1 Priority Level of Receipt of EIT Triggers and Mask for Other Triggers

Priority for accepting EITs	EIT	Masking of other EITs
1	Reset	All EITs are cleared
2	Instruction Break	Other EITs are canceled (ILM = 4)
3	INTE instruction	Other EITs are canceled (ILM = 4)
4	Undefined instruction exception	Other EITs are canceled (I-flag = 0)
5	INT instruction / Coprocessor exceptions	I-flag = 0
6	Memory protection violation	I-flag = 0
7	User interrupt	ILM = level of accepted INT
8	NMI (user)	ILM = 15
9	NMI (emulator)	Other EITs are canceled (ILM = 4)
10	Step Trace trap	Other EITs are canceled (ILM = 4)
11	Operand Break	Other EITs are canceled (ILM = 4)

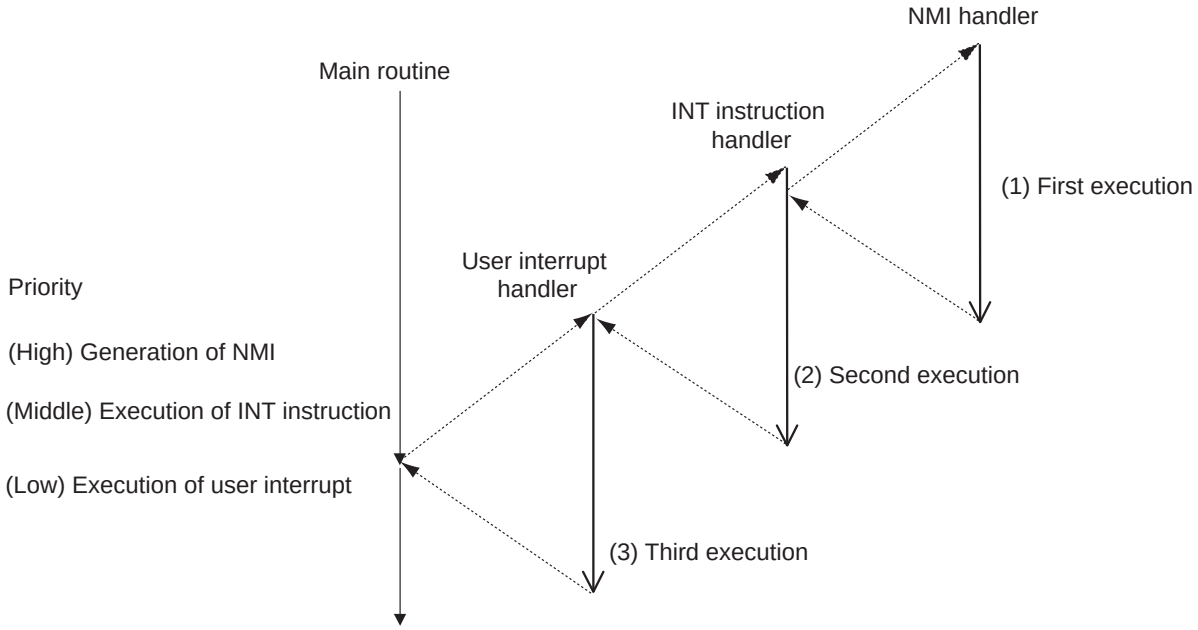
■ How to Mask Other Triggers upon the Receipt

Table below shows the execution sequence of each handler for multiple EIT triggers generated at the same time when considering mask processes for other triggers upon the receipt of EIT triggers.

Table 7-2 Execution Sequence of EIT Handlers

Handler execution Priority	EIT	Masking of other EITs
1	Reset	All EITs are cleared
2	Undefined instruction exception	Other EITs are canceled (I-flag = 0)
3	Instruction Break	Other EITs are canceled (ILM = 4)
4	INTE instruction	Other EITs are canceled (ILM = 4)
5	NMI (emulator)	Other EITs are canceled (ILM = 4)
6	Step Trace trap	Other EITs are canceled (ILM = 4)
7	Operand Break	Other EITs are canceled (ILM = 4)
8	NMI (user)	ILM = 15
9	Memory protection violation	I-flag = 0
10	INT instruction / Coprocessor exceptions	I-flag = 0
11	User interrupt	ILM = level of accepted INT

Figure 7-1 Multiple EITs Process



8. Operation

In the following sections, note that source “PC” means instruction address which detected each EIT trigger.

Similarly, “address of next instruction” means the following addresses based on the instruction which detected the EIT.

- When LDI is 32: PC+6
- When LDI is 20, and it is COPOP, COPLD, COPST or COPSV: PC+4
- For other instructions: PC+2

8.1 User Interrupt operation

If user interrupt request occurs, it determines whether to receive its request or not in the following sequence.

■ How to determine whether to receive interrupt request or not

1. Selects the interrupt which holds the highest priority level (the smallest number) by comparing interrupt request levels generated at the same time.
   For the level to be compared, it uses the value which ICR holds corresponding to maskable interrupt.
2. Selects the interrupt request which has the earliest interrupt number if multiple interrupt requests with the same priority level are generated.
3. Where “Interrupt level ≧ Level mask value”, the interrupt request is masked without receipt. Where “Interrupt level < Level mask value”, it goes forward to Step 4.
4. When selected interrupt request is maskable interrupt, if I flag is 0, its Interrupt request is masked without receipt and if I flag is 1, it goes forward to Step 5.
5. If conditions above are satisfied, interrupt requests are received between instruction processes.
   If user interrupt requests are received upon detecting EIT requests, CPU executes the following operations according to the Interrupt number for the interrupt request received.

■ Operation

1. The contents of the program status (PS) are saved to the system stack.
2. The address of the next instruction is saved to the system stack.
3. The value of the system stack pointer (SSP) is reduced by 8.
4. The value (level) of the accepted interrupt is stored in the “ILM”.
5. The value “0” is written to the “S” flag in the condition code register (CCR) in the program status (PS).
6. The vector address of the accepted interrupt is stored in the program counter (PC).

After the interrupt sequence, EIT is checked again before executing the main program handler’s instruction. If any receivable EIT is generated at this time, the CPU goes to EIT process sequence.

8.2 Operation of INT Instruction

INT No. u8 instruction is operated as follows.
Branches to interrupt handler of vector specified in u8.

■ Operation

1. The contents of the program status (PS) are saved to the system stack.
2. The address of the next instruction is saved to the system stack.
3. The value of the system stack pointer (SSP) is reduced by 8.
4. The value “0” is written to the “I” flag in the condition code register (CCR) in the program status (PS).
5. The value “0” is written to the “S” flag in the condition code register (CCR) in the program status (PS).
6. The value ‘TBR+3FC$_H$-4 x u8’ is stored in the program counter (PC).

8.3 Operation of INTE Instruction

INTE instruction is operated as follows.
Branches to vector interrupt handler of vector number 9.

■ Operation

1. The contents of the program status (PS) are saved to the system stack.
2. The address of the next instruction is saved to the system stack.
3. The value of the system stack pointer (SSP) is reduced by 8.
4. The value (level) “4” is stored in the “ILM”.
5. The value “0” is written to the “S” flag in the condition code register (CCR) in the program status (PS).
6. The value ‘TBR+3D8$_H$’ is stored in the program counter (PC).

During the execution of step, EIT is not generated by INTE.
Since INTE instruction is used for the device, do not use it.

8.4 Operation of Step Trace Trap

If you set T flag at SCR within PS and enable step trace trap function, step trace trap is generated with each executing instruction.

■ Condition for detecting step trace trap

T flag = 1

Instructions are other than delayed branch command.

During the execution of instructions other than INTE instructions or step trace trap process routines.

If conditions above are satisfied, it is broken between instruction operations.

■ Operation

1. The contents of the program status (PS) are saved to the system stack.
2. The address of the next instruction is saved to the system stack.
3. The value of the system stack pointer (SSP) is reduced by 8.
4. The value (level) “4” is stored in the “ILM”.
5. The value “0” is written to the “S” flag in the condition code register (CCR) in the program status (PS).
6. The value ‘TBR+3CC$_H$’ is stored in the program counter (PC).

If you set T flag to enable step trace trap, user interrupt is disabled.

In addition, EIT will not be generated by INTE instruction.

FR60 generates traps from next instruction to instruction which set T flag.

8.5 Operation of Undefined-instruction Exception

If any undefined instruction is detected upon decoding instruction, undefined-instruction exception is generated.

■ Condition for detecting undefined-instruction exception

- Upon the decoding instruction, undefined instruction is detected.
- It is out of delayed slot. (It is not the instruction which is right after delay branch instruction.)

If conditions above are satisfied, undefined-instruction exception will be generated.

■ Operation

1. The contents of the program status (PS) are saved to the system stack.
2. The address of the instruction that caused the undefined instruction exception is saved to the system stack.
3. The value of the system stack pointer (SSP) is reduced by 8.
4. The value “0” is written to the “S” flag in the condition code register (CCR) in the program status (PS).
5. The value ‘TBR+3C4$_H$’ is stored in the program counter (PC).

Address of the instruction which detected undefined-instruction exception is saved as PC.

8.6 Coprocessor Absent Trap

If you execute coprocessor instruction for unmounted coprocessor, coprocessor absent trap is generated.

■ Operation

1. The contents of the program status (PS) are saved to the system stack.
2. The address of the instruction that caused the undefined instruction exception is saved to the system stack.
3. The value of the system stack pointer (SSP) is reduced by 8.
4. The value “0” is written to the “S” flag in the condition code register (CCR) in the program status (PS).
5. The value ‘TBR+3E0$_H$’ is stored in the program counter (PC).

8.7 Coprocessor Error Trap

If error occurs during the use of coprocessor, coprocessor error trap is generated when you execute coprocessor instruction in order to operate the coprocessor next time.

■ Operation

1. The contents of the program status (PS) are saved to the system stack.
2. The address of the instruction that caused the undefined instruction exception is saved to the system stack.
3. The value of the system stack pointer (SSP) is reduced by 8.
4. The value “0” is written to the “S” flag in the condition code register (CCR) in the program status (PS).
5. The value ‘TBR+3DC$_H$’ is stored in the program counter (PC).

8.8 Operation of RETI Instruction

RETI instruction is the instruction which returns from EIT process routine.

■ Operation

1. Load data from stack indicated by (R15)* to the program counter (PC).
2. Increment R15+4 and store to R15.
3. Load data from stack indicated by (R15)* to the program status (PS).
4. Increment R15+4 and store to R15.

RETI instruction should be executed with S flag “0”.

9. Caution

- Since INTE instruction is used for the Debug Support Unit (DSU), do not use it in any application.
- Delay slot for branch instruction has restrictions on EIT.

See “Chapter 7 Branch Instruction (Page No.129)”.

Chapter 7 Branch Instruction

FR60 can instruct the operation with and without delay slot for branch instruction.

1. Branch Instruction with Delay Slot

- Branch instruction with delay slot

JMP:D @Ri	CALL:D label12	CALL:D @Ri	RET:D
BRA:D label9	BNO:D label9	BEQ:D label9	BNE:D label9
BC:D label9	BNC:D label9	BN:D label9	BP:D label9
BV:D label9	BNV:D label9	BLT:D label9	BGE:D label9
BLE:D label9	BGT:D label9	BLS:D label9	BHI:D label9

2. Operation of Branch Instruction with Delay Slot

Operation with delay slot executes the instruction located in the next address where a branch instruction exists (referred to as delay slot) and then branches before executing branched instructions.

Since it executes delay slot instruction before branch operation, apparent execution rate becomes 1 cycle. Instead, if delay slot has no valid instruction, NOP instruction must be entered.

- Example

```
;          Sequence of instruction
           ADD     R1,     R2     ;
           BRA:D   LABEL          ; Branch instruction
           MOV     R2,     R3     ; Delay slot ..... To be executed before the branch.
           ...
LABEL :    ST      R3,     @R4    ; Branched instruction
```

In conditional branch instruction, regardless of whether branch parameter is approved or not, instructions located in delay slot are executed.

In delay branch instruction, execution sequence of some instructions seems opposite, however, it only applies to updating process on the PC. Any other operation (register update/look-up) is executed in the order of description.

3. Actual Example (with Delay Slot)

3.1 JMP:D @Ri / CALL:D @Ri Instruction

Ri referred in JMP:D @Ri / CALL:D @Ri instruction remains intact even if instructions within delay slot update Ri.

- Example

```
LDI:32   #Label,   R0
JMP:D    @R0             ; Branches to Label.
LDI:8    #0,       R0    ; Not affect any branched address.
...
```

3.2 RET:D Instruction

RP referred in RET:D instruction remains intact even if instructions within delay slot update RP.

- Example

```
RET:D                    ; Branches to the address previously specified in RP.
MOV      R8,       RP    ; Not affect any return operation.
...
```

3.3 Bcc:D rel Instruction

Flag referred in Bcc:D rel instruction also remains unaffected by instructions within delay slot.

- Example

```
ADD      #1,       R0    ; Change of flag
BC:D     Overflow         ; Branches in accordance with the execution result of instructions above.
ANDCCR #0                 ; This flag update is not referred in branch instruction above.
...
```

3.4 CALL:D Instruction

When RP is referred using the instruction within delay slot of CALL:D instruction, the data updated by CALL:D instruction is read out.

- Example

```
CALL:D   Label             ; Branches by updating RP.
MOV      RP,       R0      ; Transfers RP based on the execution results of CALL: D above.
...
```

4. Restrictions on Branch Instruction with Delay Slot

4.1 Available Instructions for Delay Slot

Instructions which meet the following requirements can only be executed in delay slot.

- 1-cycle instruction
- Non-branch instruction
- Instruction which does not affect any operation even if its sequence is changed.

“1-cycle instruction” indicates instructions whose number of cycles column in the instruction list table is described with “1”, “a”, “b”, “c” or “d”.

4.2 Step Trace Trap

Step trace trap is not generated between the execution of branch instruction with delay slot and delay slot.

4.3 Interrupt

Interrupt is not acceptable between the execution of branch instruction with delay slot and delay slot.

4.4 Undefined-instruction Exception

If undefined instruction exists in delay slot, undefined instruction-exception is not generated. In this case, undefined instruction works as NOP instruction.

5. Branch Instruction without Delay Slot

- Branch instruction without delay slot:

JMP @Ri	CALL label12	CALL @Ri	RET
BRA label9	BNO label9	BEQ label9	BNE label9
BC label9	BNC label9	BN label9	BP label9
BV label9	BNV label9	BLT label9	BGE label9
BLE label9	BGT label9	BLS label9	BHI label9

6. Operation of Branch Instruction without Delay Slot

Operation without delay slot executes instructions in the order of instructions and never executes the instruction located in the next address where a branch instruction exists before branch.

- Example

```
;         Sequence of instruction
          ADD     R1,     R2    ;
          BRA:D   LABEL         ; Branch instruction (without delay slot)
          MOV     R2,     R3    ; Not to be executed.
          ...
LABEL     ST      R3,     @R4   ; Branched instruction
```

The number of execution cycles of branch instruction without delay slot is 2 cycles for branch and 1 cycle for no branch.

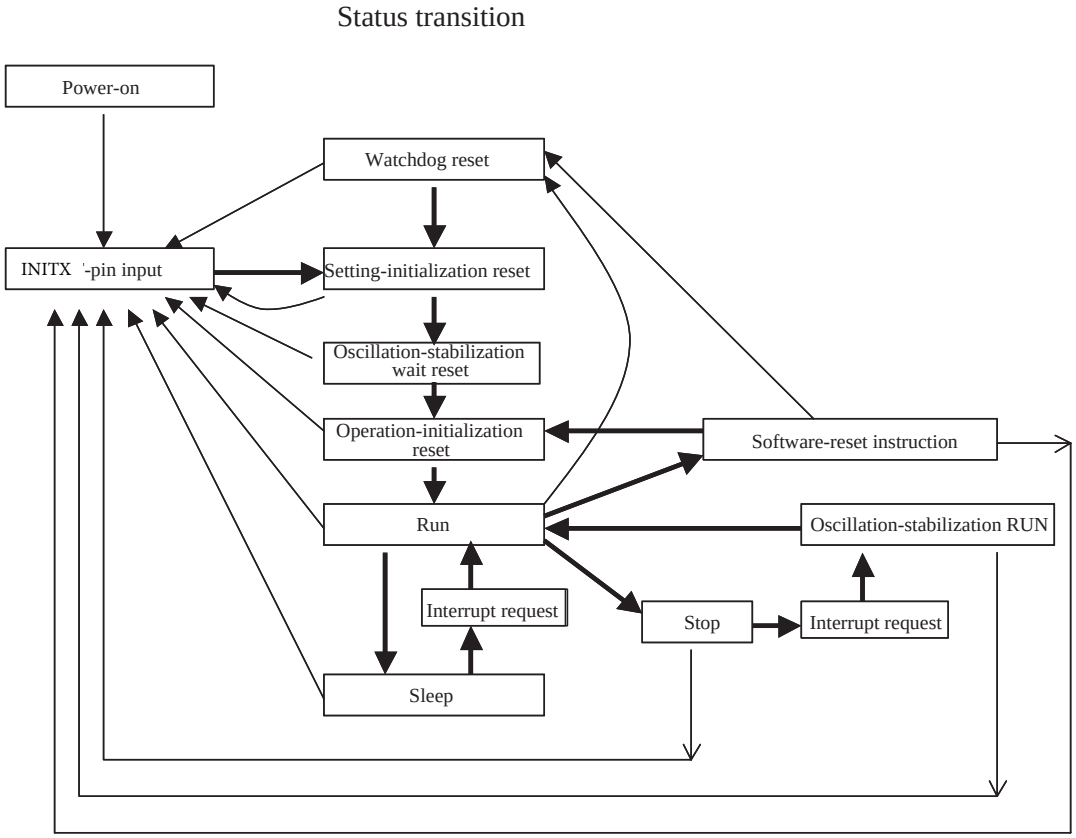
Unlike the branch instruction with delay slot where NOP is described because appropriate instruction cannot be entered, it can increase efficiency of instruction code.

Select the operation with delay slot when valid instruction can be set in delay slot. Otherwise select the operation without delay slot. This selection enables FR60 to satisfy both of execution rate and code efficiency.

Chapter 8 Device State Transition

1. Overview

MB91460 basically has devices state and flow as shown below.
For more information, see “3. State Transition Diagram (Page No.134)”.



2. Features

■ Device state

- RUN (Normal operation): State where the program is executed.
- SLEEP: State where the program is stopped (peripheral circuits are operating).
- STOP: State where the device is stopped.
- Oscillation-stabilization-wait RUN: State to return from the STOP to the RUN state (waiting until clock oscillation is stabilized).
- Oscillation-stabilization-wait reset: State for waiting until the clock oscillation is stabilized after INIT.
- Operation-initialization reset (RST): State where the program is initialized.
- Setting-initialization reset (INIT): State where all settings are initialized.

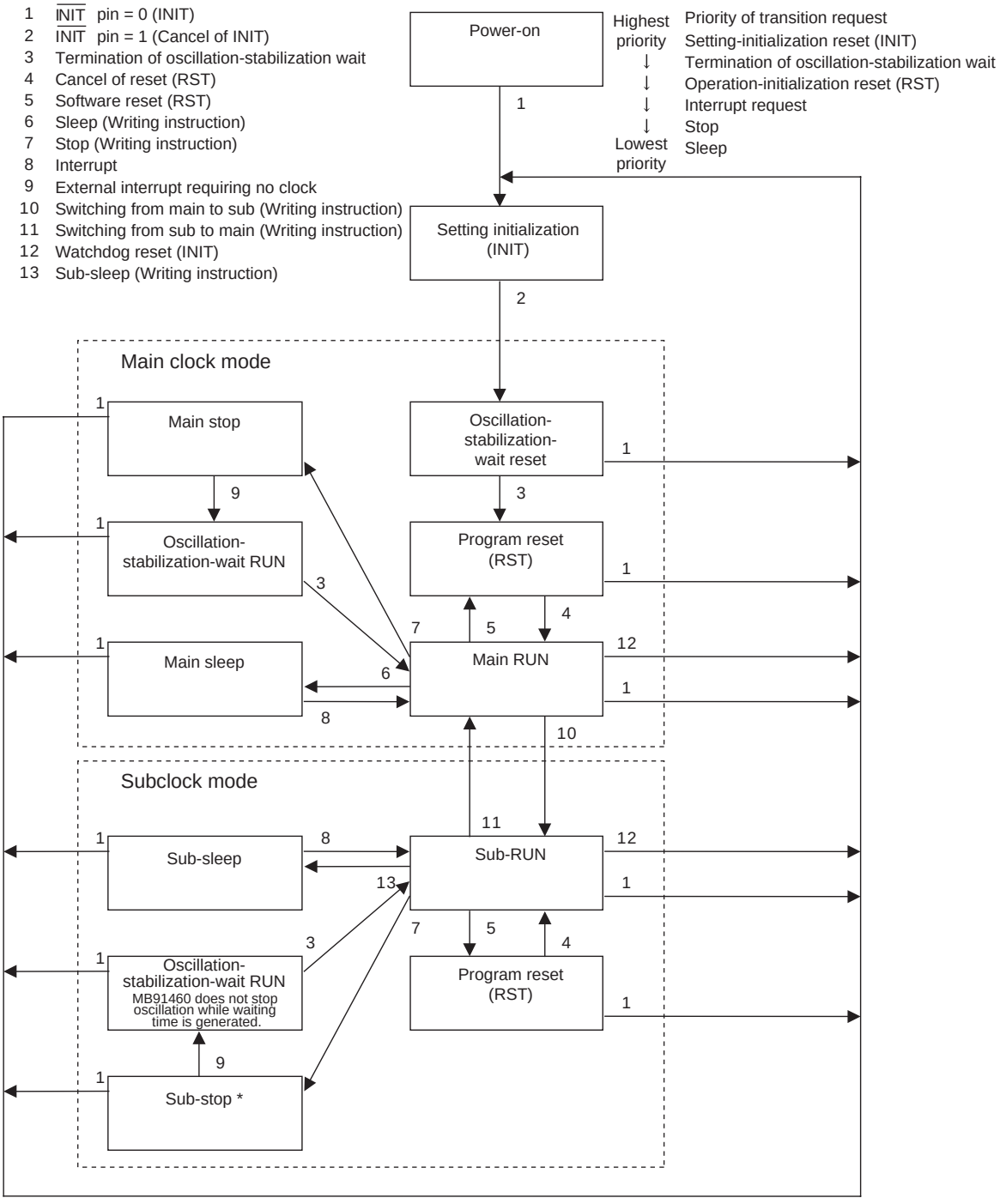
■ Standby mode (Low-power-consumption mode)

SLEEP and STOP above are standby modes.

3. State Transition Diagram

This section describes state transition.

Figure 3-1 State Transition of MB91460 Series



* To switch clock source between main and sub, in the RUN status where switched clock is stable, switch clock source register (CLKR) between bit 1 and 0 (CLKS1 and CLKS0 bit).

3.1 RUN (Normal Operation)

This is the state where program is executed with all clocks and all circuits are enabled.

This state has various paths for a state transition. However, if the synchronous reset mode is selected the state transition operations for some requests are different from normal reset mode. For more information, see the chapter of “Chapter 9 Reset (Page No.139)”.

3.2 SLEEP

This is the state where only the CPU's program execution is stopped and the peripheral circuits are operating. Embedded memories and internal/external buses are stopped unless the DMA controller requests them. This state is entered by program operation.

- Upon generation of valid interrupt requests the SLEEP state is cancelled and the RUN mode (Normal operation) is entered.
- Upon generation of the setting-initialization reset request by the external INITX pin the setting-initialization reset state (INIT) is entered.
- Upon generation of the operation-initialization reset request (external RSTX or software reset) the operation-initialization reset state (RST) is entered.

3.3 STOP

All internal circuits are stopped, all internal clocks and the PLL is stopped. Main/Sub oscillation and RC oscillation (which can be connected to the Real Time Clock (RTC) can be stopped by setting the related registers). This state is entered by program operation.

Additionally high impedance for external pins can be enabled by setting the related register.

- Upon generation of specific valid interrupt requests (requiring no clock), active oscillation timer interrupt or main clock oscillation stabilization timer interrupt request the oscillation stabilization wait RUN state is entered.
- Upon generation of the setting-initialization reset request by the external INITX pin the setting-initialization reset state (INIT) is entered.
- Upon generation of the operation-initialization reset request (external RSTX or software reset) the operation-initialization reset state (RST) is entered.

The Real Time Clock (RTC) can be supplied in the STOP mode with either main or sub oscillation clock, if the control bits for oscillation disable (OSCDx of the STCR register) are not set to disable.

The Real Time Clock (RTC) can be supplied in the STOP mode with the RC oscillation clock, if the control bit for oscillation enable (RCE of the CSVCR register) is not set to disable.

3.4 Oscillation-stabilization-wait RUN

All internal circuits are stopped except for clock generation control parts (timebase counter and device state control parts). All internal clocks are stopped while oscillation circuits and enabled main PLL is operated. This state is entered automatically after the return from STOP.

- High-impedance control of external pins by STOP is cancelled.
- After the configured oscillation-stabilization-wait time has passed the RUN (Normal operation) state is entered.
- Upon generation of the setting-initialization reset request by the external INITX pin the setting-initialization reset state (INIT) is entered.
- Upon generation of the operation-initialization reset request (external RSTX or software reset) the operation-initialization reset state (RST) is entered.

3.5 Oscillation-stabilization-wait Reset

This is the state where the device is stopped. This state is entered upon a setting-initialization reset (INIT).

All internal circuits are stopped except for clock generation control parts (timebase counter and device state control parts). All internal clocks are stopped while oscillation circuits and main PLL (if enabled) are operating.

- High-impedance control of external pins by STOP is cancelled.
- For internal circuits this state outputs operation-initialization reset (RST).
- After configured oscillation-stabilization-wait time has passed the oscillation-stabilization-wait reset state is entered.
- Upon generation of the setting-initialization reset request by the external INITX pin the setting-initialization reset state (INIT) is entered.

3.6 Operation-initialization Reset (RST)

This is the state where the program execution is initialized. Upon receipt of the operation-initialization reset (external RSTX pin or software reset) request or the termination of the oscillation-stabilization-wait reset (RST) this state is active.

CPU's program is stopped and program counter is initialized. All peripheral circuits are initialized except for some peripheral circuits. All internal clocks, oscillation circuits and enabled main PLL are operating.

- For internal circuits this state asserts operation-initialization reset (RST).
- Upon clear of request of operation-initialization reset (RST) this state transits to the RUN (normal operation) state and executes the operation-initialization reset sequence. Upon returning from setting-initialization reset (INIT) this state executes the setting-initialization reset sequence.
- Upon generation of the setting-initialization reset request by the external INITX pin the setting-initialization reset state (INIT) is entered.

3.7 Setting-initialization Reset (INIT)

This is the state where all settings are initialized. Upon receipt of request of setting-initialization reset (INIT) this state is active.

CPU's program is stopped and program counter is initialized. All peripheral circuits are initialized. Oscillation circuits are operating while the main PLL is stopped. All internal clocks are operating except while "L" level is input to the external INITX pin.

- For internal circuits this state asserts the setting-initialization reset (INIT) and the operation-initialization reset (RST).
- Upon clear of the setting-initialization reset (INIT) request this state cancels the setting-initialization reset state and then enters to the oscillation-stabilization-wait reset. After that it executes the operation-initialization reset sequence.

3.8 Priority of Each Request of State Transition

In any state, each request of the state transition is subject to the following priority:

[Highest priority]	Request of setting-initialization reset (INIT)
↓	Termination of oscillation-stabilization-wait time (This is generated only in status of oscillation-stabilization-wait reset and oscillation-stabilization-wait RUN.)
↓	Request of operation-initialization reset (RST)
↓	Request of valid interrupt (This is generated only in RUN, sleep or stop status.)
↓	Request of stop mode (Writing in register) (This is generated only in RUN status.)
[Lowest priority]	Request of sleep mode (Writing in register) (This is generated only in RUN status.)

3.State Transition Diagram

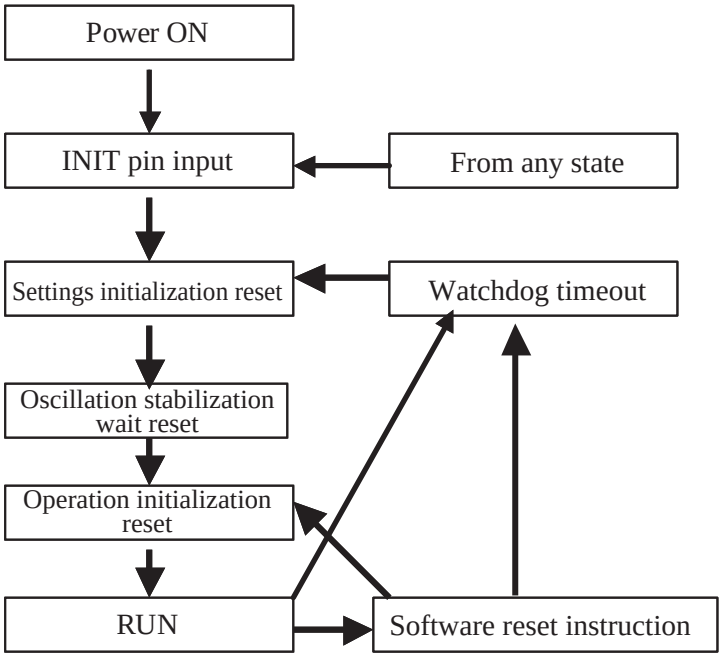
Chapter 9 Reset

1. Overview

When a reset is triggered, the device halts the program and all hardware operation, and then initializes all states. This state is called a reset.

When the reset trigger condition is removed, the device changes from this initialized state to restart the program and hardware operation. The series of steps from removal of the reset condition until operation starts is called the reset cancellation sequence.

Figure 1-1 Flow of Reset Operation



2. Features

- **Types of reset**
  - INIT pin input: Settings initialization reset (INIT)
  - RST pin input: Operation initialization reset (RST)
  - HSTX pin input: Settings initialization reset (INIT)
  - Watchdog reset: Settings initialization reset (INIT)*
  - Software reset: Operation initialization reset (RST)
  - Low Voltage reset: Setting initialization reset (INIT)

*: Although a watchdog reset triggers the same settings initialization reset (INIT) as the INITX pin input, it does not initialize the oscillation stabilization time selection bits (OS[1:0]) and reset cause flags (INIT, HSTB, WDOG, ERST, SRST and LINIT).

- **Cause of reset can be determined**
  - The cause of the previous reset is stored in a series of flags (INIT, HSTB, WDOG, ERST, SRST and LINIT) in the RSRR register.
- **Operation after reset condition is removed**
  - Operation mode: Determined by the mode pins and mode data.

- A settings initialization reset (INIT) is followed by an operation reset (RST) after the oscillation stabilization time elapses.

3. Configuration

Figure 3-1 Configuration Diagram

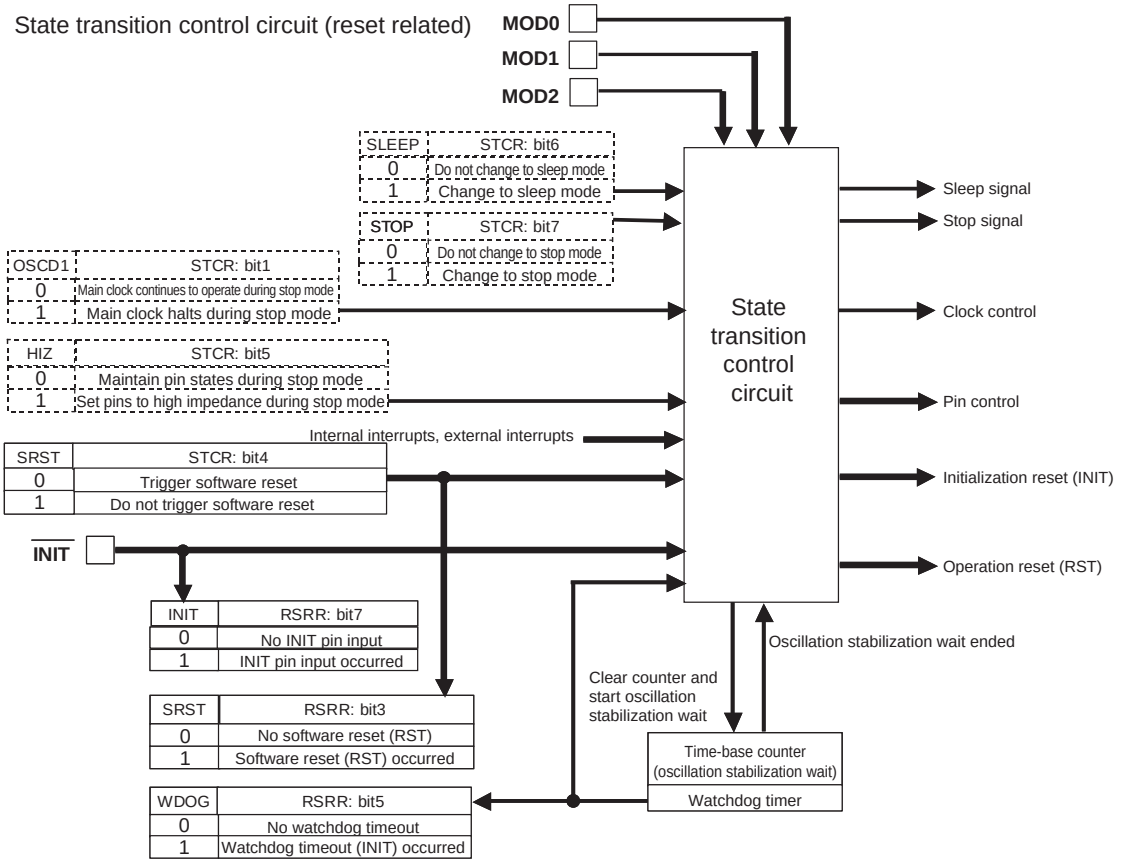


Figure 3-2 Register List

RSRR/STCR

Address	Bit 7	6	5	4	3	2	1	0		
00480H	INIT	HSTB	WDOG	ERST	SRST	LINIT	WT1	WT0	RSRR	(Reset cause)
00481H	STOP	SLEEP	HIZ	SRST	OS1	OS0	OSCD2	OSCD1	STCR	(Standby control)

4. Registers

4.1 RSRR: Reset Cause Register

Stores the cause of the previous reset, and sets the period and activation control for the watchdog timer.

- **RSRR: Address 0480h (Access: Byte, Half-word)**

7	6	5	4	3	2	1	0	bit
INIT	HSTB	WDOG	ERST	SRST	LINIT	WT1	WT0	
1	0	0	0	0	0	0	0	Initial value (INIT pin input)
-	-	-	X	X	-	0	0	Initial value (Watchdog reset)
X	X	X	-	-	X	0	0	Initial value (Software reset)
R/WX	R/WX	R/WX	R/WX	R/WX	R/WX	R/W	R/W	Attribute

Note: See “Meaning of Bit Attribute Symbols (Page No.10)” for details of the attributes.

Reading the reset request cause returns the reset cause flags and then clears the flag values to “0”.
If multiple resets occur prior to reading the register, the resulting flag values contain the bitwise OR of the flags for each reset. That is, more than one flag may be set to “1”.

- Bit7: Initialization reset occurred flag

  Indicates whether a reset (INIT) was triggered by $\overline{\text{INIT}}$ pin input.

INIT	Meaning
0	No INIT has been triggered by the $\overline{\text{INIT}}$ pin input.
1	INIT has been triggered by the $\overline{\text{INIT}}$ pin input or by the hardware watchdog.

  The initialization reset occurred flag (INIT) is cleared to “0” after reading.
  See “Chapter 21 Hardware Watchdog Timer (Page No.287)” for details.

- Bit6: Hardware Standby reset occurred flag

  Indicates whether a reset (INIT) was triggered by $\overline{\text{HST}}$ pin input.

HSTB	Meaning
0	No INIT has been triggered by the $\overline{\text{HST}}$ pin input.
1	INIT has been triggered by the $\overline{\text{HST}}$ pin input.

  The hardware standby reset occurred flag (HSTB) is cleared to “0” after reading.

- Bit5: Watchdog reset occurred flag

  Indicates whether a reset (INIT) was triggered by the watchdog timer.

WDOG	Meaning
0	No INIT has been triggered by the watchdog timer.
1	INIT has been triggered by the watchdog timer.

  The watchdog reset occurred flag (WDOG) is cleared to “0” after reading.

- Bit4: External reset occurred flag

  Indicates whether a reset (RST) was triggered by the $\overline{\text{RST}}$ pin input.

ERST	Meaning
0	No RST has been triggered by the $\overline{\text{RST}}$ pin input.
1	RST has been triggered by the $\overline{\text{RST}}$ pin input.

  The external reset occurred flag (ERST) is cleared to “0” after reading.

- Bit3: Software reset occurred flag

Indicates whether a software reset has been triggered by writing to the software reset bit (STCR.SRST).

SRST	Meaning
0	No RST has been triggered by a software reset.
1	RST has been triggered by a software reset.

The software reset occurred flag (SRST) is cleared to “0” after reading.

- Bit2: Low voltage reset occurred flag

  Indicates whether a reset (INIT) was triggered by the low voltage detection.

LINIT	Meaning
0	No INIT has been triggered by the low voltage detection.
1	INIT has been triggered by the low voltage detection.

The low voltage reset occurred flag (LINIT) is cleared to “0” after reading.

- Bit1-0: Watchdog period selection

  The watchdog period selection bits (WT[1:0]) can set the period of the watchdog timer to the following:
  (F x 2^{20} to 2^{21}, F x 2^{22} to 2^{23}, F x 2^{24} to 2^{25}, F x 2^{26} to 2^{27})
  See “Chapter 20 Software Watchdog Timer (Page No.275)” for details.

4.2 STCR: Standby Control Register

This register is used for software reset control (changing to standby mode, pin control in stop mode, and clock oscillation halted in stop mode), and specifies the oscillation stabilization wait time.

Note: See also “Chapter 10 Standby (Page No.155)”.

- **STCR: Address 0481h (Access: Byte, Half-word)**

7	6	5	4	3	2	1	0	bit
STOP	SLEEP	HIZ	SRST	OS1	OS0	OSCD2	OSCD1	
0	0	1	1	0	0	1	1	Initial value (INIT pin input)
0	0	1	1	X	X	1	1	Initial value (Watchdog reset)
0	0	X	1	X	X	X	X	Initial value (Software reset)
R/W	R/W	R/W	R1,W	R/W	R/W	RX/WX	R/W	Attribute

Note: See “Meaning of Bit Attribute Symbols (Page No.10)” for details of the attributes.

- Bit7: Stop mode
  Writing “1” to the stop mode bit (STOP) changes to stop mode.
  See “Chapter 10 Standby (Page No.155)” for details.
- Bit6: Sleep mode
  Writing “1” to the sleep mode bit (SLEEP) changes to sleep mode.
  See “Chapter 10 Standby (Page No.155)” for details.
- Bit5: High impedance mode
  Writing “1” to the high impedance mode bit (HIZ) sets pin to high impedance (Hi-z) during stop mode.
  See “Chapter 10 Standby (Page No.155)” for details.
- Bit4: Software reset
  Writing “0” to the software reset bit triggers a software reset.

SRST	Operation
0	Trigger a software reset
1	Do not trigger a software reset

  - Note that negative logic is used.
  - The read value is always “1”.
- Bit3-2: Oscillation stabilization time selection
  The oscillation stabilization time selection bits (OS[1:0]) set the oscillation stabilization time as follows:
  ($F2 \times 2^1$, $F2 \times 2^{11}$, $F2 \times 2^{16}$, $F2 \times 2^{22}$)
  The count is supplied by the timebase counter.
  Initialized to “00” ($F2 \times 2^1$, main clock) by a reset triggered by INIT pin input.
  See “Chapter 19 Timebase Timer (Page No.265)” for details.
- Bit1: Halt sub clock oscillation
  Writing “1” to the halt sub clock oscillation bit (OSCD2) halts the oscillation of the sub clock during stop mode.
- Bit0: Halt main clock oscillation
  Writing “1” to the halt main clock oscillation bit (OSCD1) halts the oscillation of the main clock during stop mode.

4.3 MOD: Mode Pins

These pins specify the location of the mode vector and reset vector that are read after the MCU is reset.

Mode pins			Mode name	Reset vector Access area	Remarks
MD2	MD1	MD0			
0	0	0	Internal ROM mode vector	Internal	
0	0	1	External ROM mode vector	External	

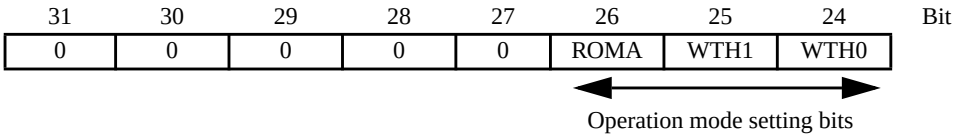
4.4 Mode Vector

The data written to the mode register (MODR) by the mode vector fetch operation is called the mode data. (The mode register is an internal register and cannot be written to or read from directly.)

After the mode register is set, the MCU operates in accordance with the modes (bus mode and access mode) set in this register.

The mode data is set by all types of reset. Setting the mode data from the user program is not possible.

- **Mode Vector: Address 000FFF8h (Access: Byte, Half-word, Word)**



- Bit32-27: Reserved bits

  Always set these bits to “00000”.

  If a value other than “00000” is set, the operation of the MCU is not guaranteed.

- Bit26: Internal ROM enable

  Specifies whether to enable the internal ROM area.

ROMA	Function	Remarks
0	External ROM mode	Enables the external ROM area.
1	Internal ROM mode	Enables the internal ROM area.

  Always set to “1”.

- Bit25-24: Bus width setting

  This sets the bus width for external bus mode.

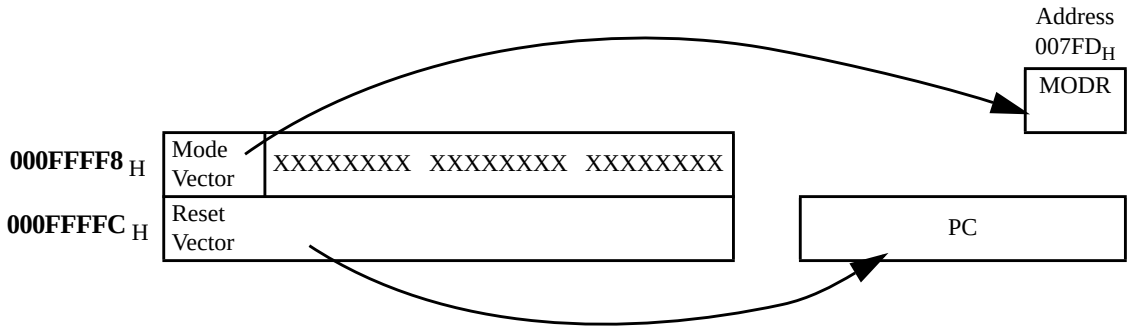
WTH1	WTH0	Function	Remarks
0	0	8-bit bus width	
0	1	16-bit bus width	
1	0	32-bit bus width	
1	1	Single chip mode	

- Bit23-0: Undefined bits

4.5 Reset Vector

The MCU starts program execution from the address specified by the mode vector.

Initial value to load into PC.



4.6 Device Mode Overview

The following table gives an overview about supported device mode combinations on the MB91460 series:

Mode pins			Mode/Reset Vector access area	FIXED Mode/Reset Vector	ROMA	ROM access area	WTH[1:0]	Bus width	Remarks
MD2	MD1	MD0							
0	0	0	Internal	Yes	1	Internal	10	32bit	Fixed Mode Data is 0x06
				No	0	External	00	8bit	
							01	16bit	
							10	32bit	
							11	Single	Setting not supported
					1	Internal	00	8bit	
							01	16bit	
							10	32bit	
							11	Single	
0	0	1	External	-	0	External	00	8bit	
							01	16bit	
							10	32bit	
							11	Single	Setting not supported
					1	Internal	00	8bit	
							01	16bit	
							10	32bit	
							11	Single	Setting not supported

Remarks:

- On the MB91460 series the ROM area is from 0x0004:0000 up to 0xFFFF:FFFF

5. $\overline{\text{INIT}}$ Pin Input (INIT: Settings Initialization Reset)

5.1 Trigger

The pin is used to trigger a settings initialization reset.

A settings initialization reset (INIT) request remains active while the pin remains at the “L” level. Keep the “L” level for the main oscillation stabilisation time.

5.2 Releasing the Reset Request

Inputting an “H” level to the pin after the main oscillation stabilisation time releases the settings initialization reset (INIT) request.

5.3 Flag

When an pin request triggers a settings initialization reset (INIT), the settings initialization reset flag (RSRR.INIT) is set to “1”.

5.4 Reset Level

This reset has the maximum reset level and initializes all settings. This type of reset is called the settings initialization reset (INIT)

A settings initialization reset (INIT) triggered by $\overline{\text{INIT}}$ pin input has the highest priority of all resets and has priority over all other inputs, operations, and states.

When a settings initialization reset (INIT) occurs, it is followed by an operation reset (RST) after the oscillation stabilization time elapses.

5.5 Initialization Triggered by $\overline{\text{INIT}}$ Pin Input (INIT)

- Device operation mode (bus mode and external bus width setting)
- All internal clock related settings (clock source selection, main PLL control, division setting)
- All settings relating to the external bus CS0 area
- All other settings related to pin states
- All areas initialized by an operation reset (RST)
  - Program operation
  - CPU and internal bus
  - Peripheral circuit register contents
  - I/O port settings
  - Device operation mode (bus mode and external bus width setting)

5.6 Reset Cancellation Sequence

After the cancellation (removal) of the settings initialization reset (external INITX pin) request the device performs the following operations in the sequence listed.

1. Removal of settings initialization reset (INIT)
2. Set operation reset (RST) state and start internal clock
3. Clear operation reset (RST) state and change to normal operation (RUN)
4. Read mode vector from address $000FFFF8_H$
5. Write mode vector to MODR (mode register)
6. Read reset vector from address $000FFFFC_H$
7. Write reset vector in PC (program counter)

   Start program execution from the address specified by PC (program counter)

Note: See explanation in “5. Operation (Page No.254)”.

6. Watchdog Reset (INIT: Settings Initialization Reset)

6.1 Trigger

Writing to the watchdog timer control register (RSRR) starts the watchdog timer. Once started, a watchdog reset request is generated unless “$A5_H$” and “$5A_H$” are written to the watchdog reset delay register (WPR) within the time specified by the watchdog period selection bits (RSRR.WT[1:0]).

6.2 Releasing the Reset Request

The watchdog reset request invokes a settings initialization reset (INIT). The watchdog reset request is released after the request is received and the settings initialization reset (INIT) generated, or when an operation reset (RST) occurs.

6.3 Flag

When watchdog reset request triggers a settings initialization reset (INIT), the watchdog timeout flag (RSRR.WDOG) is set to “1”.

6.4 Reset Level

This reset has the maximum reset level and initializes all settings. This type of reset is called the settings initialization reset (INIT).

When a settings initialization reset (INIT) occurs, it is followed by an operation reset (RST) after the oscillation stabilization time elapses.

6.5 Initialization Triggered by Watchdog Reset (INIT)

Same as for a reset triggered by an $\overline{\text{INIT}}$ pin input.

However, the oscillation stabilization time selection bits (STCR.OS[1:0]) and reset cause flags (INIT, WDOG, SRST) are not initialized and retain their existing values.

6.6 Reset Cancellation Sequence

Same as for INIT pin input.

(See “Chapter 20 Software Watchdog Timer (Page No.275)” for details.)

7. Software Reset (RST: Operation Initialization Reset)

7.1 Trigger

Writing “0” to the software reset bit (STCR.SRST) generates a software reset request.
A software reset requests an operation reset (RST).

7.2 Releasing the Reset Request

The software reset request is released after the request is received and the operation reset (RST) generated.

7.3 Flag

When software reset request triggers an operation reset (RST), the software reset flag (RSRR.SRST) is set to “1”.

7.4 Reset Level

This is a normal level reset which only initializes the program and is called an operation reset (RST).
The following section lists the main items initialized by an operation reset (RST):

7.5 Items Initialized by Operation Reset (RST)

- Program operation
- CPU and internal bus
- Content of registers in peripheral circuits
- I/O port settings
- Device operation mode (bus mode and external bus width setting)

7.6 Reset Cancellation Sequence

After cancellation (removal) of the operation reset (RST) request, the device performs the following operations in the sequence listed.

1. Removal of operation reset (RST) and change to RUN state
2. Read mode vector from address $000FFFF8_H$
3. Write mode vector to MODR (mode register)
4. Read reset vector from address $000FFFFC_H$
5. Write reset vector to the PC (program counter)
6. Start program execution from the address specified by the PC (program counter)

8. Reset Operation Modes

The following two different modes can be used for an operation reset (RST):

- Normal (asynchronous) reset mode
- Synchronous reset mode

Which mode to use is specified by the synchronous reset operation enable bit (TBCR.SYNCR).

Pin input resets and watchdog resets always use normal reset mode.

For software resets, either normal reset mode or synchronous reset mode can be selected.

8.1 Normal (Asynchronous) Reset Mode

Normal reset operation refers to the mode when the device goes to the operation reset (RST) state immediately after an operation reset (RST) request occurs.

For a normal reset, the device changes to the reset (RST) state immediately after a reset (RST) request is received regardless of the current state of internal bus access.

In normal reset mode, the result on any bus operation that is in progress at the time the device changes state is not guaranteed. However, acceptance of the operation reset (RST) request is guaranteed.

Setting the synchronous reset operation enable bit (TBCR.SYNCR) to “0” specifies normal reset mode.

Normal reset mode is the default setting after a settings initialization reset (INIT).

8.2 Synchronous Reset Operation

Synchronous reset operation refers to the mode when the device does not go to the operation reset (RST) state after a operation reset (RST) request until after all bus access has halted.

In synchronous reset mode, the device does not go to the reset (RST) state when a reset (RST) request is received if internal bus access is still in progress.

When such a reset request is received, a sleep request is issued to the internal bus. The device does not change to the operation reset (RST) state until all buses have shutdown operation and changed to sleep mode.

In synchronous reset mode, the results of bus operations are guaranteed because the device does not change state until all bus access has halted.

However, if bus access should not halt for some reason, no requests can be received while bus operation continues. In such a case, the settings initialization reset (INIT) remains available at any time.

The following lists cases in which bus access may not stop:

If bus wait is enabled due to continuous input of RDY (ready request) to the external expansion bus interface.

9. MCU Operation Mode

After release of a reset, the MCU starts operation in the mode specified by the mode pins and mode data.

Operation mode	Bus mode	Single chip mode Internal ROM/external bus mode External ROM/external bus mode
	Access mode	32-bit bus width 16-bit bus width 8-bit bus width

9.1 Bus Modes and Access Modes

■ Bus mode

The bus mode controls internal ROM operation and the external access function. The bus mode is specified by the mode setting pins (MD2, MD1, MD0) and internal ROM enable bit (Mode-Vector. ROMA).

The FR60 has the following three bus modes.

- Single chip mode

  In this mode, internal I/O, internal RAM, and internal ROM are available but access to other areas is disabled. External pins are used either by the peripheral functions or as general-purpose ports. Pins cannot be used as bus pins. This mode can not be used when using the fixed mode/reset vector as implemented on most of the MB91460 series devices.

- Internal ROM, external bus mode

  In this mode, internal I/O, internal RAM, and internal ROM are available, and access to areas for which external access is enabled results in access to the external area. Some external pins function as bus pins.

- External ROM, external bus mode

  In this mode, internal I/O and internal RAM are available but access to internal ROM is prohibited. Access to internal ROM areas and areas for which external access is enabled results in access to the external area. Some external pins function as bus pins.

■ Access mode

The access mode controls the width of the external data bus and is set by the WTH[1:0] bits in the mode data.

10. Caution

- $\overline{\text{INIT}}$ pin input

  Ensure that a settings initialization reset (INIT) is applied to this pin when the power is turned on. Also, after turning on the power, ensure a sufficient oscillation stabilization wait time is provided for the oscillation circuit by holding the input to the pin at the “L” level for the required time.

Note: The INIT reset triggered by $\overline{\text{INIT}}$ pin input initializes the oscillation stabilization wait time to its minimum value.

- Watchdog reset

  When a settings initialization reset (INIT) is triggered by a watchdog reset request, the oscillation stabilization time is not initialized. Also, in main RUN or sub RUN mode, no oscillation stabilization wait occurs in response to a watchdog reset request if the main clock is not halted.

- Software reset

  If “1” (synchronous reset mode) is set to the synchronous reset operation enable bit (TBCR.SYNCR) when an operation reset (RST) is triggered by a software reset request, the operation reset (RST) does not occur until all bus access halts. Accordingly, there may be a long delay before the operation reset (RST) occurs, depending on the bus usage.

- Settings initialization reset (INIT)

  A settings initialization reset (INIT) invokes an operation reset (RST) after the oscillation stabilization wait time elapses.

- Reset cause flags (INIT), (HSTB), (WDOG), (ERST), (SRST) and (LINIT)
  - Reading the reset cause register clears all the reset cause flags to “0”.
  - If more than one reset occurs before the reset cause register is read, the flag values are ORed and more than one flag may be set to “1”.

- Reset mode

  A settings initialization reset (INIT) initializes the reset mode to normal reset mode.

- DMA controller

  As the DMA controller halts any transfer when a request is received, it does not cause any delay in changing device state.

- Pin states during a reset

  See “8. Pin State Table (Page No.96)” for details about pin states during a reset.

10.Caution

Chapter 10 Standby

1. Overview

Two standby modes (low power consumption modes) are available.

- Sleep mode: Stops the program
- Stop mode: Shuts down the device

Note: It is possible to keep the Real Time Clock active in STOP mode (see chapter RTC).

2. Features

■ Sleep mode

- Device state in sleep mode:
  - Halts the program.
  - CPU program execution only stops. Peripheral functions can continue to operate.
  - The internal memory and internal bus halt.
- Transition to sleep mode:
  - Sleep mode is invoked by the program.
- Recovery from sleep mode:
  - Generation of a valid interrupt request ends sleep mode (returns to normal operation)
  - An INIT pin input or generation of a watchdog reset invokes an initialization reset (INIT) followed by an operation reset (RST).

■ Stop mode

- Device state in stop mode:
  - The overall device halts.
  - Internal circuits halt (with some exceptions)
  - Internal clock signals halt (with some exceptions)
  - Whether or not the oscillation circuit halts can be controlled by a setting (programmable).
  - All external pins can be set to high impedance (programmable, excludes some pins)
- Transition to stop mode:
  - Stop mode is invoked by the program.
- Recovery from stop mode:
  - The following four interrupt requests change the device to the oscillation stabilization wait state.
    - •External level-detect or edge-detect interrupt
    - •Interrupt generated by oscillation stabilization wait timer for the main clock when oscillation not halted.
    - •Interrupt generated by oscillation stabilization wait timer for the sub clock when oscillation not halted.
    - •Real time clock interrupt when oscillation not halted.
  - Input to the INITX pin invokes an initialization reset (INIT) and then an operation reset (RST).

3. Configuration

Figure 3-1 Configuration Diagram

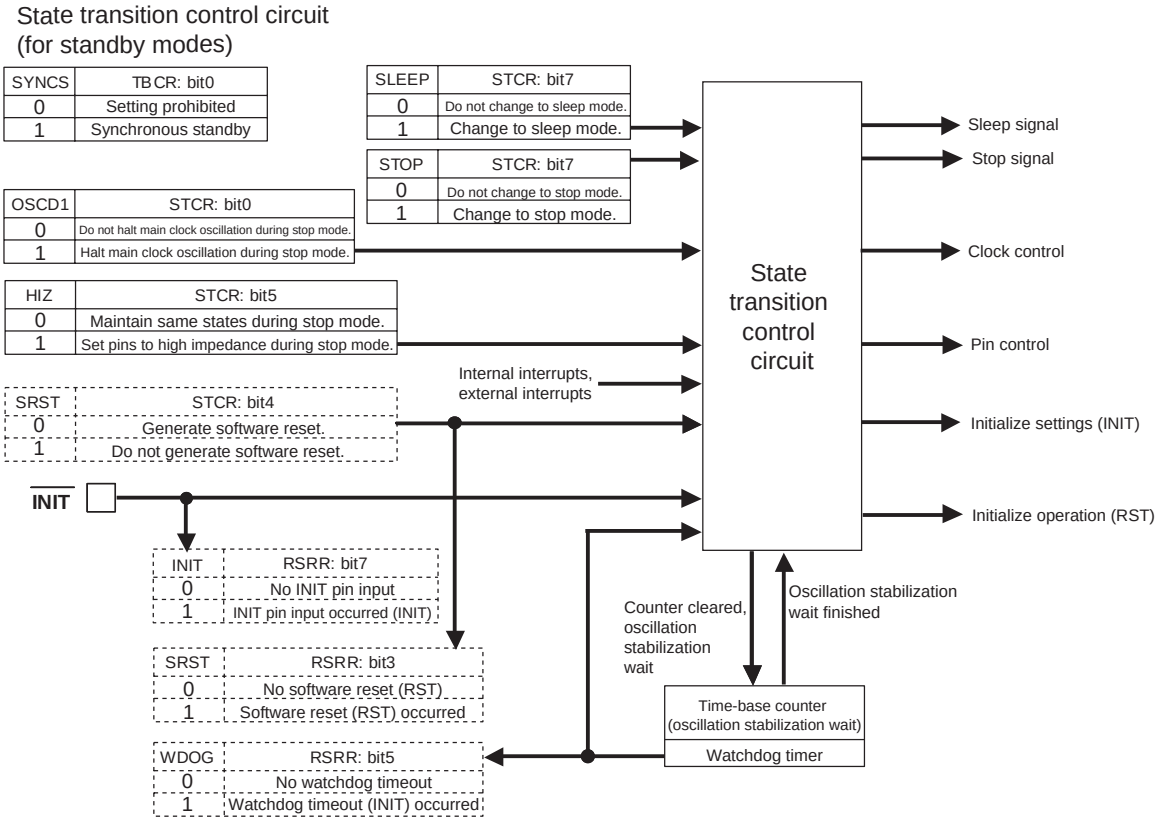


Figure 3-2 Register List

Standby control

Address	Bit	7	6	5	4	3	2	1	0		
00481H		STOP	SLEEP	HIZ	SRST	OS1	OS0	OSCD2	OSCD1	STCR	(Standby control)
00482H		TBIF	TBIE	TBC2	TBC1	TBC0	---	SYNCR	SYNCR	TBCR	(Time-base counter control)

4. Registers

4.1 STCR: Standby Control Register

Used to control transition to the stop and sleep standby modes, and to specify the pin states and whether to halt the oscillation during stop mode.

Note: See “Chapter 9 Reset (Page No.139)” also.

- **STCR: Address 0481h (Access: Byte)**

7	6	5	4	3	2	1	0	bit
STOP	SLEEP	HIZ	SRST	OS1	OS0	OSCD2	OSCD1	
0	0	1	1	0	0	1	1	Initial value (INITX pin input)
0	0	1	1	X	X	1	1	Initial value (Watchdog reset)
0	0	X	1	X	X	X	X	Initial value (Software reset)
R/W	R/W	R/W	R1, W	R/W	R/W	R/W	R/W	Attribute

(See “Meaning of Bit Attribute Symbols (Page No.10)” for details of the attributes.)

- Bit7: Stop mode

STOP	Operation
0	Does not change to stop mode.
1	Changes to stop mode.

  - Goes to “0” when a reset (INIT pin input or software reset) occurs or on recovery from stop mode.
  - Going directly from main PLL operation to stop mode is prohibited. (See “8. Caution (Page No.165)”.)
- Bit6: Sleep mode

SLEEP	Operation
0	Does not change to sleep mode.
1	Changes to sleep mode.

  - If this bit and the stop mode bit (STOP) bit are set to “1” at the same time, the device goes to stop mode.
  - Goes to “0” when a reset (INIT pin input or software reset) occurs or on recovery from sleep mode.
- Bit5: High impedance mode

HIZ	Operation
0	Maintain same pin states when changing to stop mode.
1	Set pin outputs to high impedance (Hi-z) during stop mode.

  - The default setting is high impedance.
- Bit4: Software reset (SRST)
  - Setting this bit to “0” invokes a software reset.
- Bit3-2: Oscillation stabilization time selection (OS[1:0])
  - Setting these bits in the range “00”-“11” sets the oscillation stabilization time to use after recovering from stop mode.
    An INIT pin input reset or watchdog reset initialize this setting to its initial value.
    (See “Chapter 18 Timebase Counter (Page No.249)”.)
- Bit1: Sub clock oscillation halt

OSCD2	Operation of sub clock during stop mode
0	Continue oscillation
1	Halt oscillation

- Bit0: Main clock oscillation halt

OSCD1	Operation of main clock during stop mode
0	Continue oscillation
1	Halt oscillation

4.2 TBCR: Timebase timer control register

This register controls the timebase timer interrupts and the options for resets and standby operation.

Note: See also “Chapter 19 Timebase Timer (Page No.265)”.

- **TBCR: Address 0482h (Access: Byte)**

7	6	5	4	3	2	1	0	bit
TBIF	TBIE	TBC2	TBC1	TBC0	---	SYNCR	SYNCS	
0	0	X	X	X	X	0	0	Initial value (INIT pin, watchdog)
0	0	X	X	X	X	X	X	Initial value (Software reset)
R(RM1),W	R/W	R/W	R1,W	R/W	RX/WX	RX/WX	R/W	Attribute

(See “Meaning of Bit Attribute Symbols (Page No.10)” for details of the attributes.)

- Bit7: Interrupt request flag for timebase timer
  - This flag goes to “1” when a timebase timer interrupt request occurs
- Bit6: Interrupt request enable for the timebase timer
  - Writing “1” to this bit enables timebase timer interrupt requests.
- Bit5-3: Interval time selection for timebase timer
  - Writing a value in the range “000”-“111” to these bits selects the interval time for the timebase timer. (F x 2^{11}, x 2^{12}, x 2^{13}, x 2^{22}, x 2^{23}, x 2^{24}, x 2^{25}, x 2^{26})
- Bit2: Reserved Writing does not affect the operation. The read value is undefined.
- Bit1: Enable synchronous reset operation
  - Selects a normal reset “0” or a synchronous reset “1”.
- Bit0: Enable synchronous standby operation

SYNCS	Operation
0	Normal reset operation (Not permitted on this model).
1	Enable synchronous standby operation (always set this before changing to a standby mode).

5. Operation

5.1 Sleep Mode

■ Entering sleep mode

Writing “1” to the sleep mode bit (STCR.SLEEP) changes to sleep mode. The device remains in this mode until an event occurs to wakeup the device from sleep mode.
(See “8. Caution (Page No.165)”.)

■ Device state in sleep mode

- CPU program execution stops. (Peripheral functions continue to operate.)
- The internal memory and internal bus halt.
- Circuits that halt during sleep mode
  - Bit search module
  - All internal memory (inclusive I-cache)
  - Internal/external bus
- Circuits that do not halt during sleep mode
  - Oscillation circuit, main PLL (if enabled)
  - Clock generation control circuit
  - Interrupt controller
  - External interrupts
  - DMA
  - Peripherals

■ Recovery and other items

- Generation of an interrupt request that is currently enabled changes the device back to RUN mode. (Restores normal operation.)
- An INIT pin input or generation of a watchdog reset invokes an initialization reset (INIT) followed by an operation reset (RST).

5.2 Stop mode

■ Entering stop mode

Writing “1” to the stop mode bit (STCR.STOP) changes to stop mode.
The device remains in this mode until an event occurs to wakeup the device from stop mode.
(See “8. Caution (Page No.165)”.)

■ Device state in stop mode

- The overall device halts (internal circuits halt and the internal clock signals halt).
- Circuits that halt during stop mode
  All internal circuits except those listed below.
- Circuits that do not halt during stop mode
  - Oscillation circuits that are not specified to be halted
    - Oscillation circuit for main clock (if not disabled)
    - Oscillation circuit for sub clock (if not disabled)
    - Main PLL circuit if oscillation circuit for main clock is enabled and PLL circuit is enabled and main regulator is kept enabled.
  - Peripheral functions that are driven directly by the oscillation and which have not been specified to be halted.
    - Real Time Clock (if not disabled) and main or sub clock oscillation is enabled and the RTC clock source is set to the enabled oscillation
    - LCDC (if LCD display enabled for sub-stop mode and subclock selected as the clock source.)
- Pin states (High impedance or maintain previous state)
  - When pin outputs are set to go to high impedance during stop mode
    - High impedance output: Pins that are set as general purpose ports and pins that have been selected for use by peripheral functions.
  - When pin outputs are set to maintain their previous states during stop mode
    - Maintain previous state: Pins that are set as general purpose ports and pins that have been selected for use by peripheral functions.
  - When set as external interrupts
    - Input available state:

      Pins set as external interrupt inputs using level detection or edge detection.
      (Whether the pin output during stop mode has been set to either high impedance or maintain previous state.)

■ Recovery and other items

- Any of the following interrupt requests cause the device to go to the oscillation stabilization wait RUN state and then to change back to RUN mode after the oscillation stabilization time elapses (return to normal operation).
  - External interrupts set to level detection or edge detection and that do not require a specific clock.
  - Real Time Clock interrupt (if operating)
- An INIT pin input or generation of a watchdog reset invokes an initialization reset (INIT) followed by an operation reset (RST) after the oscillation stabilization time.

6. Settings

Table 6-1 Settings Required to Change to Sleep Mode

Setting	Setting register	Setting procedure*
Interrupt settings	(See the chapter for each peripheral function.)	—
Synchronous standby settings	Timebase timer control register (TBCR)	See 7.1
Change to sleep mode	Standby control register (STCR)	See 7.1
Operational restrictions	(See “8. Caution (Page No.165)”.)	—

*:For the setting procedure, refer to the section indicated by the number.

Table 6-2 Settings Required to Change to Stop Mode

Setting	Setting register	Setting procedure*
Selects the oscillation stabilization wait time	(See “Chapter 18 Timebase Counter (Page No.249)”.)	—
Interrupt settings	(See the chapter for each peripheral function.)	—
Synchronous standby settings	Timebase timer control register (TBCR)	See 7.2
Change to stop mode	Standby control register (STCR)	See 7.2
Operational restrictions	(See “8. Caution (Page No.165)”.)	—

*: For the setting procedure, refer to the section indicated by the number.

7. Q&A

7.1 How do I change to sleep mode?

Before you can change to sleep mode, you must first set the synchronous standby operation enable bit (TBCR.SYNCS).

Operation	Synchronous standby operation enable bit (SYNCS)
To enable synchronous standby operation	Set to “1”.

Note: Setting (SYSNCS=“0”) is prohibited.

Set using the sleep mode bit (STCR.SLEEP).

Operation	Sleep mode bit (SLEEP)
When you do not want to change to sleep mode	Set to “0”.
To change to sleep mode	Set to “1”.

Note: Some restrictions apply when changing to sleep mode. See “8. Caution (Page No.165)” for details.

7.2 How do I change to stop mode?

- When operating on the main PLL clock, the operating clock must be set to the main clock divided by two.
  See "7.3 How do I select the operating clock source? (Page No.202)" for details about changing the operating clock.
- Before you can change to stop mode, you must first set the synchronous standby operation enable bit (TBCR.SYNCS). See section 7.1.
- Set using the stop mode bit (STCR.STOP).

Operation	Stop mode bit (STOP)
When you do not want to change to stop mode	Set to "0".
To change to stop mode	Set to "1".

Note: Some restrictions apply when changing to stop mode. See "8. Caution (Page No.165)" for details.

7.3 How do I set pins to high impedance (Hi-z) during stop mode?

Set using the high impedance mode bit (STCR.HIZ).

Operation	High impedance mode bit (HIZ)
When you do not want to set pins to high impedance during stop mode	Set to "0".
To set pins to high impedance during stop mode	Set to "1".

Note: Some ports do not go to high impedance in some circumstances. (See "5.2 Stop mode (Page No.160)".)

7.4 How do I halt the main clock oscillation during stop mode?

Use the main clock oscillation stop bit (STCR.OSCD1).

Operation	Main clock oscillation stop bit (OSCD1)
When you do not want to halt the main clock oscillation in stop mode	Set to "0".
To halt the main clock oscillation during stop mode	Set to "1".

7.5 How do I recover from sleep mode?

Two methods are available to recover from sleep mode.

- Generation of a valid interrupt request changes to RUN mode (restores normal operation).
  If using interrupt processing, remember to set the I flag (I), interrupt level mask register (ILM), and interrupt control register (ICR).
- An INIT pin input or generation of a watchdog reset invokes an initialization reset (INIT) followed by an operation reset (RST).

7.6 How do I recover from stop mode?

The following events end stop mode:

- The following four interrupts change the device to the oscillation stabilization wait state.
  - External level-detect interrupt or edge-detect interrupt.
  - Oscillation stabilization wait timer for the main clock when oscillation not halted.
  - Sub oscillation stabilisation timer when oscillation not halted.
  - Real time clock when oscillation not halted.

  If using interrupt processing, remember to set the I flag (I), interrupt level mask register (ILM), and interrupt control register (ICR).
- Input to the INIT pin invokes an initialization reset (INIT) followed by an oscillation stabilization delay and then an operation reset (RST).

  In the case of an INIT pin input, an oscillation stabilization wait is required, depending on the width of the INIT pin input.

  See also “Chapter 13 Clock Control (Page No.189)” and “Chapter 18 Timebase Counter (Page No.249)”.

7.Q&A

8. Caution

- Points to note when changing to sleep mode

  When changing to sleep mode, set the synchronous standby operation enable bit (TBCR.SYNCS= “1”).
  Also, in order to change to sleep mode with synchronous standby operation enabled, the STCR register must be read after writing to the SLEEP bit. Always use the following sequence.

```
(LDI    #value_of_sleep, R0)  ; value_of_sleep contains the write data for STCR.
(LDI    #_STCR, R12)          ; _STCR is the address of STCR (481H).
 STB    R0, @R12              ; Write to standby control register (STCR).
 LDUB   @R12, R0              ; STCR read required for synchronous standby.
 LDUB   @R12, R0              ; Second dummy read to STCR.
 NOP                          ; NOP x 5 required for timing
 NOP
 NOP
 NOP
 NOP
```

- Points to note when changing to stop mode

  When changing to sleep mode, set the synchronous standby operation enable bit (TBCR.SYNCS= “1”).
  Also, in order to change to stop mode with synchronous standby operation enabled, the STCR register must be read after writing to the STOP bit. Always use the following sequence.

```
(LDI    #value_of_stop, R0)   ; value_of_stop contains the write data for STCR.
(LDI    #_STCR, R12)          ; _STCR is the address of STCR (481H).
 STB    R0, @R12              ; Write to standby control register (STCR).
 LDUB   @R12, R0              ; STCR read required for synchronous standby.
 LDUB   @R12, R0              ; Second dummy read to STCR.
 NOP                          ; NOP x 5 required for timing
 NOP
 NOP
 NOP
 NOP
```

- When the main PLL is selected as the operation clock source

  When the main PLL is selected as the operation clock source, change the operation clock source selection to main clock divided by two before changing to stop mode.
  See “Chapter 13 Clock Control (Page No.189)” for details.
  The restrictions that apply to the clock divide ratio setting are the same as for normal operation. Also, you do not necessarily have to halt the PLL oscillation.

- If interrupts are disabled in the interrupt control register (ICR=“00011111B”), the device will not recover from stop or sleep mode when an interrupt occurs.
- Pin high impedance control in stop mode

  Setting the high impedance bit (STCR.HIZ) to “1” sets pin outputs to high impedance during stop mode. If the high impedance bit (STCR.HIZ) is set to “0”, pins retain the states they have prior to entering stop mode.
  See “8. Pin State Table (Page No.96)” for details such as the operation of specific pins.

8.Caution

Chapter 11 Memory Controller

1. Overview

This module combines the interfaces to the F-Bus memory resources, FLASH and General Purpose RAM (also referenced as I/D-RAM). These memories can be combined CODE and DATA storage. While code fetch is possible in general via the F-Bus at the FR core, due to performance reasons the code fetch is accellerated by a direct I-Bus connection in MB91460 series MCUs.

For FLASH access the interface contains an instruction cache and data read buffer. A prefetch mechanism removes CPU internal code fetch latencies for linear code.

In addition the module includes the definition of the Fixed Mode Vector (FMV) and the Fixed Reset Vector (FRV), depending on the device mode.

2. FLASH Interface

- Wait timing
- Generation of FLASH control signals ATDIN and EQIN for synchronous access.
  (this version supports independent timing configuration of ADTIN, EQIN and Wait)
- Generation of CEX, WEX and OEX
- Handling of 32 or 64 bit read mode and 16 or 32 bit read/write mode for programming
- Support of external SRAM for emulation devices with 1:1 timing transparency (same wait cycles)
- Measures for FLASH macro test and parallel programming support

3. General Purpose RAM

- Zero wait cycle access (code), one wait cycle access (data) to shared code/data memory (up to 64 kByte), also referenced as I/D-RAM

4. Instruction Cache and Data Buffer

- Up to 16 kByte Instruction cache (4k word entries, one way direct mapped, prefetch miss option)
- Size configuration for the evaluation device (0, 4, 8 and 16 kB)
- 1 or 2 dword (32 or 64 bit) data read buffer **(not available on MB91460 series)**

5. Prefetch

- Prefetch of consecutive instruction word address to the cache buffer
- Prefetch is canceled in case of prefetch miss (branch or data access), thus it works without any penalties in the prefetch miss case.
- The FLASH macro needs to support FLASH access cycle cancelation at any point, that means it may not affect the timing of the next complete access cycle (no special recovery condition required from previous access cancelation).

6. Fixed Mode and Reset Vectors

- Mode vector address: 0x000ffff8; return 0x06000000 for internal vector mode

- Reset vector address: 0x000ffffc; return 0x00030000 at RAM execution mode (jump to test program) or return 0x0000bff8 in any other case (jump to Boot ROM)
- If FMCS_FIXE is switched off, the FLASH memory can be accessed on addresses 0x000ffff8 and 0x000ffffc. FIXE is set at reset.

7. Registers

● List of FLASH-IF Registers

Table 7-1 FLASH-IF Registers Summary

Address	Register				Block
	+0	+1	+2	+3	
7000_H	FMCS [R/W] 01101000	FMCR [R/W] ----0000	FCHCR [R/W] ------00 1000011		FLASH_IF
7004_H	FMWT [R/W] 11111111 01111111[1] 11111111 01011101[2]		FMWT2[R/W] -101----	FMPS [R/W] -----000	
7008_H	FMAC [R] 00000000 00000000 00000000 00000000				
$700C_H$	FCHA0 -------- -0000000 00000000 00000000				
7010_H	FCHA1 -------- -0000000 00000000 00000000				

Remarks:

- Read and write access to all registers is byte, halfword and word.
- FMCR and FMWT2 registers are not available on MB91V460
- The initial values of the waitcycle setting register FMWT differs between the flash-less evaluation device MB91V460 and devices with embedded flash memory

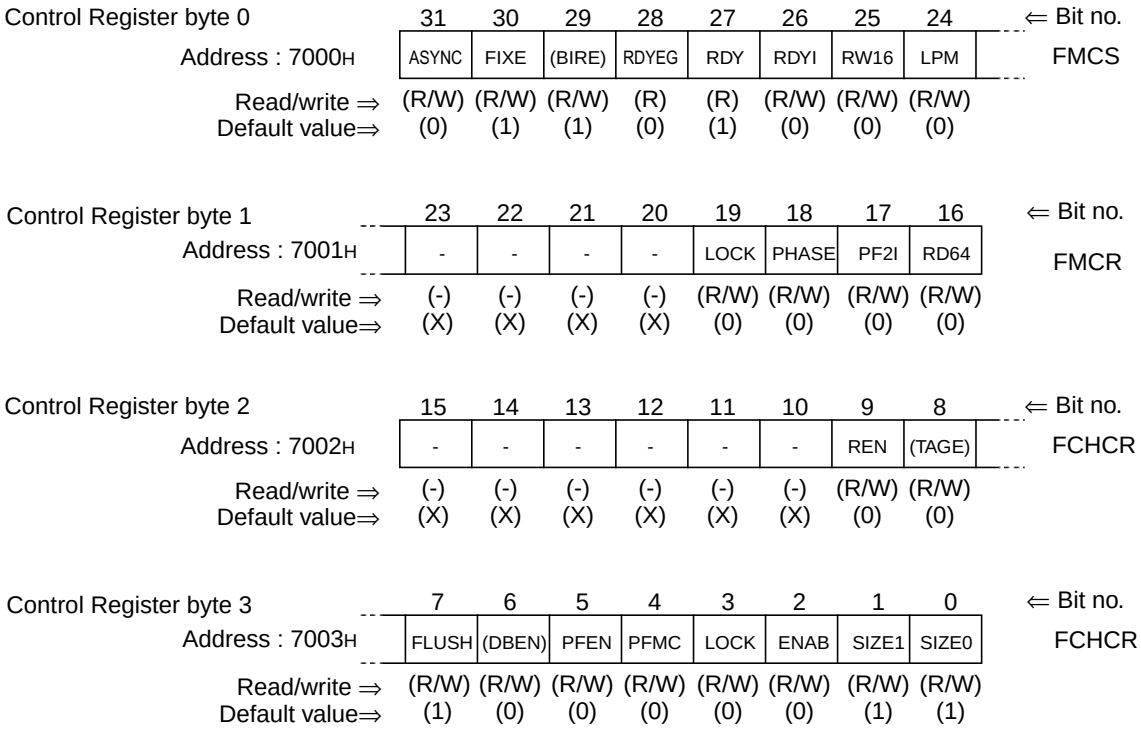
Notes:

1) Initial value on MB91V460

2) Initial value on MB91F467DA

8. Explanations of Registers

● FLASH Interface Control Register



FLASH Memory Control and Status Register (FMCS)

- **BIT[31]: ASYNC - ASYNChronous access enable**

0	Synchronous FLASH access (default)
1	Asynchronous FLASH access

The ASYNC bit is cleared at reset, which enables the fast synchronous FLASH access mode by default. To switch to asynchronous mode, set this bit (however it is basically not recommended to set this bit, neither in read nor in write access).

- **BIT[30]: FIXE - FIXed reset and mode vector Enable**

0	Disable FMV/FRV and enable FLASH access at mode vector address
1	Output the fixed mode or reset vector at address hit (default)

The FIXE bit is set by default.

To enable FLASH access on address 0x000ffff8 and 0x000ffffc, clear this bit.

- **BIT[29]: BIRE - Burn-In ROM Enable**

0	Disable Burn-In ROM and enable FLASH access at Burn-In ROM address
1	Enable access to the Burn-In ROM (default)

The BIRE bit is a reserved bit and should not be used.

- **BIT[28]: RDYEG - RDY status hold and qittation register**

0	Auto algorithm not started or started and not compleated (default)
1	The FLASH auto-algorithm has been completed since last register read access.

The RDYEG bit is cleared after reset.

The bit is set after 0->1 transition of FMCS_RDY. The bit shows that the RDY signal of the FLASH is or was active (compleated auto-algorithm). The RDYEG bit is cleared automatically at read access to address 0x7000.

The RDYEG bit is read-only status information.

Remark: The function of this status bit is not guaranteed when running the CPU on frequencies lower than 1MHz.

- **BIT[27]: RDY - FLASH RDY status of auto-algorithm**

This bit shows the status of the RDY line of the FLASH macro. RDY is used to signalize the state of the FLASH macro in case of an auto-algoriythm was started (e.g. sector erase, chip erase). If RDY returns to ’1’, the auto-algorithm has been completed.

The RDY bit is read-only status information.

Remark: The function of this status bit is not guaranteed when running the CPU on frequencies lower than 1MHz.

- **BIT[26]: RDYI - RDY output force**

0	Inactive (default)
1	Force the RDY output to ’1’

This bit is reserved for FLASH test. Do not set this bit.

- **BIT[25]: RW16 - 16 bit Read/Write enable to FLASH**

0	32 bit read and write access to FLASH is enabled (default)
1	16 bit read and write access to FLASH is enabled

This bit is cleared after reset. There is a 32 bit read and write access to the FLASH memory enabled by default.

Setting of the RW16 bit implies switching from 32 bit into 16 bit mode. When it is intended to write data to the flash memory (or while chip erase or sector erase) then code fetch from the flash memory is not supported.

Important remark: To maintain data consistency it is strongly recommended to disable the instruction cache while writing to the FLASH memory and to flush the instruction cache (FLUSH=1) after completing the write procedure to the FLASH memory.

Important remark: It is not allowed to switch between the 16 bit, the 32 bit and the 64 bit mode while reading instructions or data from the FLASH memory.

- **BIT[24]: LPM - Low Power Mode**

0	Low power mode off (default)
1	Low power mode enabled

This bit is cleared after reset. The low power mode is switched off by default.

If LPM=0, CEX is permanently asserted to ’0’ (active). This enables fastest possible FLASH access timing.

Setting this bit to ’1’ enables the low power mode. CEX is asserted low only in case of FLASH access. In between the FLASH macro is in stand-by mode.

Remark: On the MB91460 series with embedded FLASH memories it is not necessary to use this setting since the FLASH memory supports an “automatic sleep mode” which puts the FLASH automatically in a low power consumption state when not accessed.

FLASH Memory Control Register (FMCR)

The FMCR register is not available on the evaluation device MB91V460.

- **BIT[19]: LOCK - ALEH auto-update lock**

0	ALEH setting auto update is enabled (default)
1	ALEH setting auto update is disabled

FLASH memories embedded on the MB91460 series require a certain timing between ATDIN falling edge and EQIN rising edge. This timing is named tALEH and has usually the same length as the ATDIN duration.

By writing the setting of ATDIN length to the FMWT.ATD[2:0] bits, the FMWT2.ALEH[2:0] bits will be updated automatically to the same setting. To avoid this automatic update it is possible to set the ALEH LOCK bit.

It is also possible to apply a different setting to the FMWT2.ALEH[2:0] bits by writing first to the FMWT.ATD[2:0] bits and second to the FMWT2.ALEH[2:0] bits.

- **BIT[18]: PHASE - ATDIN/EQIN clock phase**

0	ATDIN/EQIN generation is in phase with the core clock (default)
1	ATDIN/EQIN generation is inverted to the core clock

At lower core clock frequencies it can be beneficial to change the ATDIN/EQIN generation to inverted core clock to save a waitcycle compared to the generation of these signals in phase with the core clock.

It is recommended to always refer to the setting requirements of ATDIN, EQIN and waitcycles for each product which are provided by Fujitsu (see the related datasheets).

(PHASE setting is not available on MB91460 series)

- **BIT[17]: PF2I - Prefetch 32 bit (2 instructions) only**

0	Prefetch 64 bit (default)
1	Prefetch 32 bit only

When switching on 64 bit read mode (RD64=1) then prefetch will be performed on instruction address IA+8 (when current access is aligned at IA+0) and on instruction address IA+4 (when current access is aligned at IA+4). However, the setting of PF2I=1 in the 64 bit read mode will cause a prefetch only on next instruction address IA+4 (independent of current access alignment is IA+0 or IA+4).

Usually prefetching 64 bit is superior to 32 bit only, however it can be the case on strong fragmented code that the performance deteriorates due to replacement of cache entries. In this case it can be sensible to switch to 32 bit prefetch only.

- **BIT[16]: RD64 - Enable 64 bit read mode**

0	64 bit read mode is disabled (default)
1	64 bit read mode is enabled

Some embedded FLASH memories supports switching the 64 bit read mode to increase the access performance. Please contact Fujitsu if this feature is available on the product you are using.

This bit is cleared after reset. The 32 bit read and write access to the FLASH memory is enabled by default.

Setting of the RD64 bit implies switching from 32 bit into 64 bit mode. Writing data to the flash memory is not supported in the 64 bit read only mode.

Important remark: It is not allowed to switch between the 16 bit, the 32 bit and the 64 bit mode while reading instructions or data from the FLASH memory.

FLASH Cache Control Register (FCHCR)

- **BIT[9]: REN - Non-cacheable area Range Enable**

0	FCHA1 defines address mask (default)
1	FCHA1 defines second point for the non-cacheable address range from FCHA0 to FCHA1

The bit is cleared after reset. The address defined in FCHA0 is combined with a bit mask defined in FCHA1 to define the non-cacheable area.

If the REN bit is set, the non-cacheable area is defined by two points. The non-cacheable range is from addresses greather than or equal to FCHA0 up to addresses less than or equal to FCHA1.

- **BIT[8]: TAGE - TAG RAM access Enable**

0	Memory mapped TAG RAM access disabled (default)
1	Memory mapped TAG RAM access enabled

The bit is set to 0 after reset.

(TAG RAM access is not available on MB91460 series).

- **BIT[7]: FLUSH - Flush instruction cache entries**

0	Flushing the instruction cache entries has been completed
1	Actually flushing the instruction cache entries

This bit is set after reset.

If the FLUSH bit is set, the instruction cache entries are flushed sequentially. During this initialization the cache is disabled. The initialization has a duration of 1 clock cycle per cache entry. The number of valid entries depends on the configured cache size.

After completion (all entries are flushed) the FLUSH bit is cleared by hardware.

Writing a ’1’ to this bit triggers the flushing of the cache entries.

Important remark: It is not allowed to set the cache size configuration (FCHCR.SZ[1:0]) and FLUSH at the same time (same write access). Always first set the size configuration before flushing the cache.

- **BIT[6]: DBEN - Data Buffer ENable**

0	Buffering of read data is disabled (default)
1	Buffering of read data is enabled

This bit is cleared after reset. The read data buffer is disabled by default.

Setting the DBEN bit enables the data read buffer. This is useful to speed up reading of data structures of 8 or 16 bit operands. There is one word data buffer implemented. If the same 32 or 64 bit word address is accessed consecutively, the data is read from the buffer.

(Data buffer is not available on MB91460 series)

- **BIT[5]: PFEN - PreFetch ENable**

0	Prefetch of instructions is disabled (default)
1	Prefetch of instructions is enabled

This bit is cleared after reset. The prefetch of instructions is disabled by default.

Setting the PFEN bit enables the code prefetch from the next word on instruction address IA+4. Prefetch eliminates any latency in the code fetch path of the MCU to the FLASH memory for linear code.

When switching on 64 bit read mode (RD64=1) then prefetch will be performed on instruction address IA+8 (when current access is aligned at IA+0) and on instruction address IA+4 (when current access is aligned at IA+4). However, the setting of PF2I=1 in the 64 bit read mode will cause a prefetch only on next instruction address IA+4 (independent of current access alignment is IA+0 or IA+4).

A running prefetch cycle can be directly taken over from a matching instruction access. If there is no instruction access in between, the prefetched instruction word is stored in cache memory. If there is an FLASH access (code or data) to an address different from the prefetch address, the prefetch cycle is canceled immediately.

- **BIT[4]: PFMC - Prefetch Miss Cache enable**

0	Standard cache algorithm (default)
1	Prefetch misses are cached only

This bit is cleared after reset. The prefetch miss cache is disabled by default. The instruction cache uses the standard algorithm of writing cache entries for each accessed instruction word from FLASH.

Setting the PFMC bit switches to a second write algorithm for cache entries. This algorithm writes only this instruction words to the cache, which are causing prefetch miss conditions.

The FR CPU requests approximately one instruction word (which contains two 16 bit instruction codes) in two clock cycles. If the FLASH data throughput (one word in two cycles) is sufficient for the needs of the CPU, the PFMC option is useful in most cases.

If the FLASH access time is two clock cycles, normally no wait states are generated when the next instruction word is requested from a consecutive address and prefetch is enabled. Thus, caching such linear code segments in conjunction with prefetch may not improve the code fetch performance, which is at the optimum already. More interesting is to improve the situation for branches in the code, where prefetch could not remove the latency of accessing it. If FPMC is set to '1', the cache algorithm stores only these FLASH accesses, which have caused a wait condition due to a prefetch miss condition (not mached predicted address).

The effect of this algorithm is, that the restricted amount of cache entries is utilized more efficiently. Usually the same performance can be reached with half the cache size. Or, in other words, the cache is as same efficient as it would have the doubled size.

The efficiency of the PFMC algorithm depends on the structure of the application.

- **BIT[3]: LOCK - Global lock of cache entries**

0	Write of cache entries enabled (default)
1	Writing of cache entries is disabled, the cache contents is locked

This bit is cleared during reset. The cache entries are writable by default.

If the LOCK bit is set, no new entries can be written to cache memory. The old contents of cache entries remains in memory. There is only a global lock feature for all cache entries.

- **BIT[2]: ENAB - Instruction cache enable**

0	The instruction cache is disabled (default)
1	Enable the instruction cache

This bit is cleared after reset. By default the instruction cache is disabled.

If the ENAB bit is set, the instruction cache is switched on. The instruction cache is dedicated to FLASH access only.

The cache is utilized by the prefetch algorithm as prefetch buffer. Hence prefetch can be used in an unbuffered form with cache disabled.

Cache miss did not cause code fetch penalties. The FLASH access is started in parallel, independent from cache hit or miss evaluation.

(If the cache is disabled, the cache entries and the TAG RAM contents can be accessed memory mapped. This feature is disabled in this version of the interface, see the explanaition of the TAGE bit.)

- **BIT[1:0]: SZ[1:0] - Cache size configuration**

00	0kByte - Cache disabled
01	4kByte (1024 entries)
10	8kByte (2048 entries)
11	16kByte (4096 entries) (default)

The cache size is set to ’11’ after reset.

The cache size can be configured on the evaluation device (EVA).

Remark: The number of cache entries determines the TAG initialization period at device startup, see the explanaition of the FLUSH bit above.

Important remark: On products with less than 16kByte instruction cache it is recommended to set the size configuration in the system startup according to the available cache size.

Important remark: It is not allowed to set the cache size configuration (FCHCR.SZ[1:0]) and FLUSH at the same time (same write access). Always first set the size configuration before flushing the cache.

● FLASH Memory Wait Timing Register (FMWT)

Wait Timing Register byte 0

	31	30	29	28	27	26	25	24	⇐ Bit no.
Address : 7004H	WTP1	WTP0	WEXH1	WEXH0	WTC3	WTC2	WTC1	WTC0	FMWT
Read/write ⇒	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	
Default value⇒	(1)	(1)	(1)	(1)	(1)	(1)	(1)	(1)	

Wait Timing Register byte 1

	23	22	21	20	19	18	17	16	⇐ Bit no.
Address : 7005H	(FRAM)	ATD2	ATD1	ATD0	EQ3	EQ2	EQ1	EQ0	FMWT
Read/write ⇒	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	
Default value⇒	(0)	(1)	(1/0)	(1)	(1)	(1)	(1/0)	(1)	

Wait Timing Register byte 2

	15	14	13	12	11	10	9	8	⇐ Bit no.
Address : 7006H	-	ALEH2	ALEH1	ALEH0	-	-	-	-	FMWT2
Read/write ⇒	(-)	(1)	(0)	(1)	(-)	(-)	(-)	(-)	
Default value⇒	(X)	(R/W)	(R/W)	(R/W)	(X)	(X)	(X)	(X)	

Wait Timing Register byte 3

	7	6	5	4	3	2	1	0	⇐ Bit no.
Address : 7007H	-	-	-	-	-	PS2	PS1	PS0	FMPS
Read/write ⇒	(-)	(-)	(-)	(-)	(-)	(R/W)	(R/W)	(R/W)	
Default value⇒	(X)	(X)	(X)	(X)	(X)	(0)	(0)	(0)	

Remarks:

- ATD[2:0] setting is 0x7 on MB91V460 and 0x5 on MB91F467DA
- EQ[3:0] setting is 0xF on MB91V460 and 0xD on MB91F467DA
- FMWT2 is not available on MB91V460

- **BIT[31:30]: WTP[1:0] - Wait cycles for FLASH in page access**

WTP is set to 3 after reset.

WTP controls the wait timing of the FLASH access in case of page hit for Page Mode FLASH. The WTP configuration is in units of clock cycles. The value of WTP should be set to the intra page access time (cycle time) of the FLASH memory in number of clock cycles, subtracted by one.

The setting is used if the page size PS[2:0] is set different to 0.

- **BIT[29:28]: WEXH[1:0] - Minimum WEX High timing requirement**

WEXH is set to 3 after reset. The minimum high time duration of WEX is 5 cycles by default.

Setting an other value reduces the WEX high time to 2 fixed cycles + WEXH.

- **BIT[27:24]: WTC[3:0] - Wait cycles for FLASH memory access**

WTC is set to 15 after reset.

WTC controls the wait timing of the FLASH access. The WTC configuration is in units of clock cycles. The value of WTC should be set to the access time (cycle time) of the FLASH memory in number of clock cycles, subtracted by one.

- **BIT[23]: FRAM - Wait cycles for F-Bus general purpose RAM memory access**

FRAM is set to 0 after reset.

This is a reserved bit. This version on MB91V460 has no configurable wait timing to F-Bus RAM, it operates with fixed 0 wait states RAM access.

- **BIT[22:20]: ATD[2:0] - Duration of the ATDIN signal for FLASH memory access**

MB91V460: ATD is set to 7 after reset. ATD defaults to 4 clock cycles.

MB91F467DA: ATD is set to 5 after reset. ATD defaults to 3 clock cycles.

ATD controls the timing of the ATDIN signal for FLASH access. The ATD configuration is in units of half clock cycles. The effective high duration of ATDIN equals to tATDIN=(ATD+1)*0.5 clock cycles.

- **BIT[19:16]: EQ[3:0] - Duration of the EQIN signal for FLASH memory access**

MB91V460: EQ is set to 15 after reset. EQ defaults to 8 clock cycles.

MB91F467DA: EQ is set to 13 after reset. EQ defaults to 7 clock cycles.

EQ controls the timing of the EQIN signal for FLASH access. The EQ configuration is in units of half clock cycles. The effective high duration of EQIN equals to tEQIN=(EQ+1)*0.5 clock cycles.

- **BIT[14:12]: ALEH[2:0] - Duration of the ALEH time for FLASH memory access**

MB91V460: not available

MB91F467DA: ALEH is set to 5 after reset. ALEH defaults to 3 clock cycles.

ALEH controls the timing of the ATDIN falling edge to EQIN rising edge for FLASH access.

The EQ configuration is in units of half clock cycles. The effective duration of ALEH equals to tALEH=(ALEH+1)*0.5 clock cycles.

Important remark: ALEH[2:0] is updated automatically to the same value as ATD[2:0] when writing to ATD[2:0]. Usually the ALEH time equals the ATD time, so there is normarlly no reason to update ALEH[2:0] in particular.

Even though it is possible to program ALEH[2:0] with a different value than ATD[2:0] by:

- Writing a different value to ALEH[2:0] after writing to ATD[2:0], or
- Setting the FMCR.LOCK bit to disable the auto update

FLASH access cycle waveform

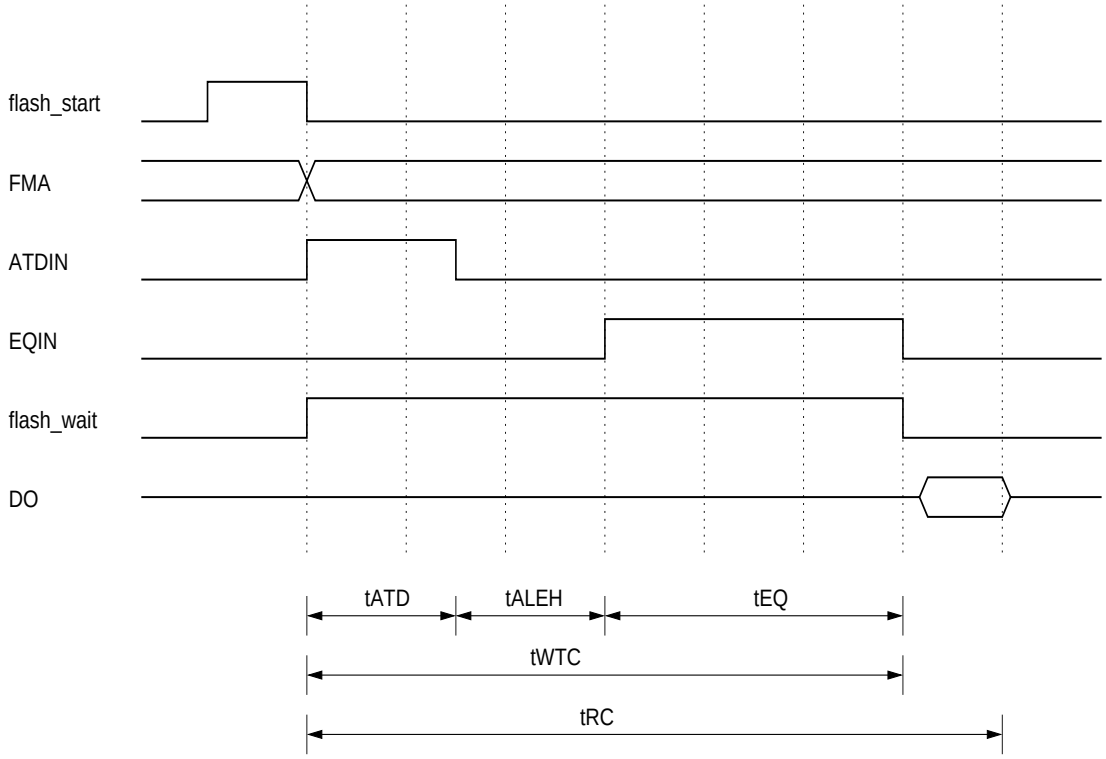


Figure 8-1 Timing of a FLASH access cycle

Figure 8-1 shows the example of a FLASH access cycle. In the FMWT register the three parts of the FLASH timing tATD, tALEH, tEQ and tWTC can be configured independently. The table below lists the configuration values for this example.

Symbol	Length	Setup
tATD	1.5 cycles	ATD=2
tALEH	1.5 cycles	ALEH=2
tEQ	3 cycles	EQ=5
tWTC	6 cycles	WTC=6

The resulting FLASH access cycle (tRC) time is 7 cycles (WTC+1).

- **BIT[2:0]: PS[2:0] - Page size definition for Page Mode FLASH**

PS is set to 0 after reset. Page Mode FLASH is disabled by default.

This setting defines the page size to 2^PS in number of bytes.

E.g. for Am29PL320D/MBM29PL3200 with a page size of 16 byte the value of PS has to be set to 4.

(Embedded FLASH memories on MB91460 series do not support page mode)

- FLASH Memory Adddress Check register (FMAC)

FMAC [R]				
Address	**+0**	**+1**	**+2**	**+3**
7008_H	--------	-0000000	00000000	00000000

This register captures the address at the begin of a FLASH access cycle for test purposes.
The register could be read only.

- Non-cacheable area definition

The non-cacheable area definition registers FCHA0 and FCHA1 define the FLASH region not to be cached. Not used bits are read back as zero. The tables below define the initial values of the registers. The point defined by FCHA0 = 0 and mask bits off is located outside the FLASH region, thus initially the whole FLASH region will be cached.

FCHA0 [R/W]				
Address	**+0**	**+1**	**+2**	**+3**
$700C_H$	--------	-0000000	00000000	00000000

FCHA1 [R/W]				
Address	**+0**	**+1**	**+2**	**+3**
7010_H	--------	-0000000	00000000	00000000

If the FCHCR_REN bit is cleared, the address range is defined by the address given by FCHA0, masked with the bits set to ’1’ in FCHA1.

Example 1 (Point and mask range definition):
- FCHCR_REN = 0
- FCHA0 = 0x000F:A300
- FCHA1 = 0x0000:FFFF

The non-cacheable area is defined from 0x000F:0000 to 0x000F:FFFF.

Example 2 (Point to point range definition):
- FCHCR_REN = 1
- FCHA0 = 0x000F:A300
- FCHA1 = 0x000F:F7FF

The non-cacheable area is defined from 0x000F:A300 to 0x000F:F7FF.

Chapter 12 Instruction Cache

This chapter describes the instruction cache memory included in MB91460 family members and its operation.

1. General description

The instruction cache is a fast local memory for temporary storage. Once an instruction is accessed to be fetched from external slower memory, the instruction cache holds the instruction code inside to increase the speed of accessing the same code from then on. The instruction cache data RAM and tag RAM are made directly read/write-accessible by software by setting the RAM mode. To turn off the instruction cache after turning it on once, be sure to use the subroutine shown in Section 4.2.4 "Settings for handling the I-Cache".

2. Main body structure

- FR basic instruction length: 2 bytes
- Block arrangement system: 2-way set associative system
- Block 128 blocks per way
  16 bytes per block (= 4 sub-blocks)
  4 bytes per sub-block (= 1 bus access unit)

Figure 2-1 Instruction Cache Structure

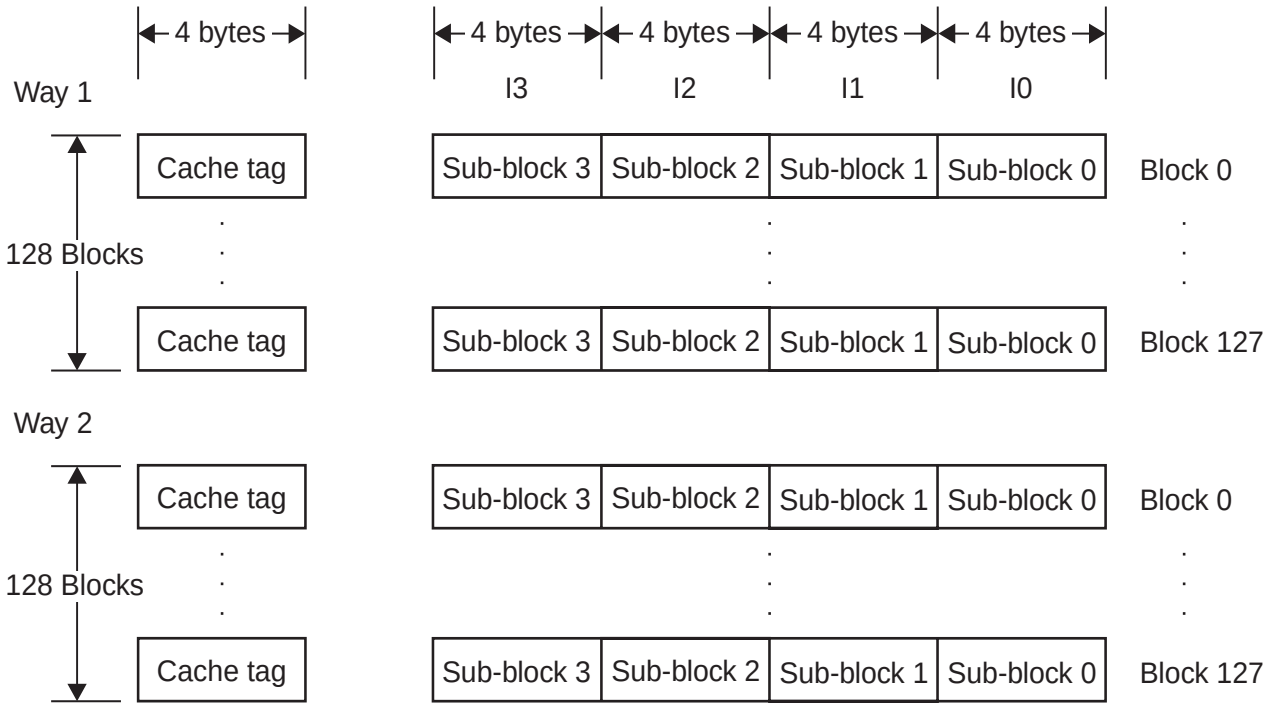
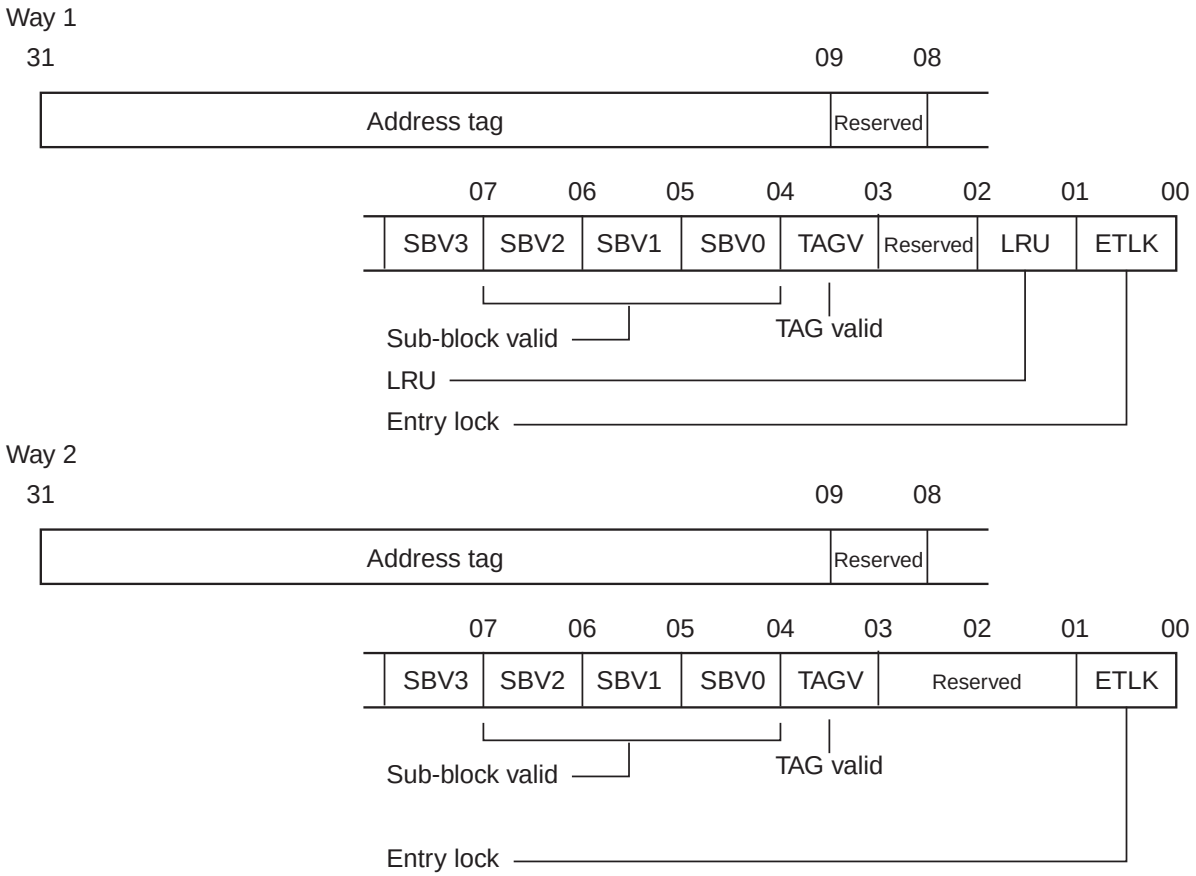


Figure 2-2 Instruction Cache Tag



[Bits 31 to 9] Address tag

This area stores the upper 23 bits of the memory address of the instruction cached in the corresponding block. For example, memory address IA of the instruction data stored in sub-block k in block i is obtained from the following equation:

IA = address tag x 2^{11} + **i** x 2^{4} + **k** x 2^{2}

The address tag is used to check for a match with the instruction address requested for access by the CPU. The CPU and cache behave as follows depending on the result of the cache check:

1) When the requested instruction data exists in the cache (hit), the cache transfers the data to the CPU within the cycle.

2) When the requested instruction data does not exist in the cache (miss), the CPU and cache obtain the data loaded by external access at the same time.

[Bits 7 to 4] **SBV3** to **SBV0**: Sub-block valid bits

When SBV* contains "1", the corresponding sub-block holds the current instruction data at the address located by the tag. Each sub-block usually holds two instructions (including immediate-value transfer instructions).

[Bit 3] **TAGV**: **TAG** valid bit

This bit indicates whether the address tag value is valid. When the bit contains "0", the corresponding block is invalid regardless of the settings of the sub-block validation bits. (The

FLUSHbit is set to "0" when the cache is flushed.)

[Bit 1] **LRU** bit (way 1 only)

This bit exists only in the instruction cache tag in way 1. The bit indicates way 1 or 2 as the way containing the last entry accessed in the selected set. When set to "1", the LRU bit indicates that the entry of the set in way 1 is the last entry accessed. When set to "0", it indicates that the one in way 2 is the last entry accessed.

[Bit 0] **ETLK**: Entry lock bit

This bit is used to lock all the entries in the block corresponding to the tag in the cache. When set to "1", the entries are locked and are not updated when a cache miss occurs. Note, however, that invalid sub-blocks are updated. If a cache miss occurs with both of ways 1 and 2 in the entry lock states, access to external memory takes place after losing one cycle used for evaluating the cache miss.

● Control register structure

ISIZE (8 bit)	7	6	5	4	3	2	1	0	**Initial value**
0000 03C7$_H$	-	-	-	-	-	-	SIZE1	SIZE0	---- --10$_B$
	-	-	-	-	-	-	R/W	R/W	

ISIZE [bits 1 to 0] : **SIZE1** and **SIZE0**

These bits set the cache capacity. The combination of the settings determines the cache size, IRAM size, and the address map in RAM mode as shown in Figure I-CACHE-3. When the cache size is changed, be sure to flush the cache and release the entry lock before turning on the cache.

CACHE Size Register

SIZE1	SIZE0	Size
0	0	1 KB
0	1	2 KB
1	0	4 KB (initial value)
1	1	Setting disabled

ICHCR [bits 7 to 0] :

The ICHCR (I-Cache Control Register) controls the operation of the instruction cache. Writing a value to the ICHCR has no effect on the caching of any instruction fetched within three cycles that follow.

ICHCR (8 bit)	7	6	5	4	3	2	1	0	**Initial value**
0000 03E7$_H$	RAM	-	GBLK	ALFL	EOLK	ELKR	FLUSH	ENAB	0-00 0000$_B$
	R/W	-	R/W	R/W	R/W	R/W	R/W	R/W	

[Bit 7] **RAM**: RAM Mode

Setting this bit to "1" causes the cache to operate in RAM mode. By placing the cache in RAM mode, the cache RAM is mapped as shown in Figure I-CACHE-3 while the cache is enabled with the ENAB bit set to "1".

[Bit 5] **GBLK**: Global lock bit

This bit locks all of the current entries in the cache. Setting the GBLK bit to "1" prevents the valid entries in the cache from being updated when a cache miss occurs. Note, however, that invalid sub-blocks are updated then. Fetching of instruction data in the global lock state is performed in the same way as not in that state.

[Bit 4] **ALFL**: Auto lock fail bit

This bit (ALFL) is set to 1 if locking is attempted on an entry that is already locked. If, during entry autolock, an entry update is attempted on an entry that is already locked, no new entry is locked in the instruction cache regardless of what the user intends. Reference this bit for debugging of a program or similar purpose. Clear this bit by writing 0 to it.

[Bit 3] **EOLK**: Entry auto lock bit

This bit enables or disables auto-locking for each entry in the instruction cache. An entry accessed (but resulting in a miss) with the EOLK bit containing "1" is locked when the entry lock bit in the cache tag is set to "1" by hardware. Once locked, the entry is not updated at any cache miss that follows until it is unlocked. Note, however, that invalid sub-blocks are updated. To ensure the entry lock, set this bit after flushing.

[Bit 2] **ELKR**: Entry lock release bit

This bit specifies the clearing of the entry lock bits in all cache tags. When the ELKR bit is set to "1" in a cycle, the entry lock bits in all cache tags are cleared to "0" in the next cycle. Note that the content of this bit is retained for only one clock cycle; it is cleared to "0" in the clock cycle that follows.

[Bit 1] **FLUSH**: Flush bit

This bit specifies the flushing of the instruction cache. Setting the FLUSH bit to "1" flushes the cache. Note that the content of this bit is retained for only one clock cycle; it is cleared to "0" in the clock cycle that follows.

[Bit 0] **ENAB**:Enable bit

This bit enables or disables the instruction cache. Setting the ENAB bit to "0" disables the cache, where the CPU directly accesses external memory to request instructions without cache intervention. While disabled, the cache retains its contents.

Figure 2-3 I-Cache Address Map

Figure I-CACHE-3 RAM Address Map

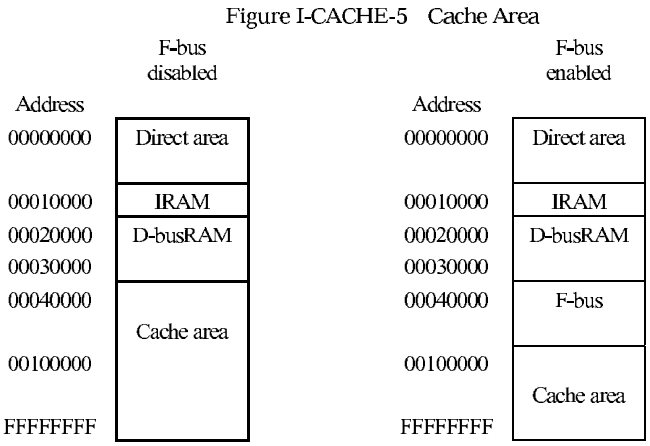
Address	Cache off RAM off	Cache off RAM on	Cache on 4K RAM off	Cache on 4K RAM on	Cache on 2K RAM off	Cache on 2K RAM on	Cache on 1K RAM off	Cache on 1K RAM on
00010000		TAG (way1)		TAG (way1)		TAG (way1)		TAG(way1)
00010200								[TAG way1]
00010400						[TAG way1]		[TAG way1]
00010600								[TAG way1]
00010800		[TAG way1]		[TAG way1]		[TAG way1]		[TAG way1]
...		...		...		...		...
								[TAG way1]
00014000		TAG (way2)		TAG (way2)		TAG (way2)		TAG(way2)
00014200								[TAG way2]
00014400						[TAG way2]		[TAG way2]
00014600								[TAG way2]
00014800		[TAG way2]		[TAG way2]		[TAG way2]		[TAG way2]
...				...		...		...
								[TAG way2]
00018000	IRAM (way1)	$RAM(IRAM) (way1)		$RAM (way1)	[IRAM way1]	$RAM (way1)		$RAM(way1)
00018200							IRAM (way1)	IRAM (way1)
00018400					IRAM (way1)	IRAM (way1)		
00018600								
00018800	[IRAM way1]	[$RAM way1]		[$RAM way1]	[IRAM way1]	[$/I way1]	[IRAM way]	[$/I way1]
...	...	...		...	...	...	...	...
0001C000	IRAM (way2)	$RAM(IRAM) (way2)		$RAM (way2)	[IRAM way1]	$RAM (way2)	-	$RAM(way2)
0001C200							IRAM (way2)	IRAM (way2)
0001C400					IRAM (way2)	IRAM (way2)		
0001C600								
0001C800	[IRAM way2]	[$RAM way2]		[$RAM way2]	[IRAM way2]1	[$/I way2]	[IRAM way2]	[$/I way2]
...	...	...		...	...	...	...	...
00020000	Square brackets [ ] enclose an area for mirroring.							

	TAGRAM			$RAM	
00010000		<- Entry at address 00x	00018000		Instruction at address 000 (SBV0)
00010004		<- Mirror of 00x	00018004		Instruction at address 004 (SBV1)
00010008			00018008		Instruction at address 008 (SBV2)
0001000C			0001800C		Instruction at address 00C (SBV3)
00010010		<- Entry at address 01x	00018010		Instruction at address 000 (SBV0)
00010014		<- Mirror of 01x	00018014		Instruction at address 004 (SBV1)
00010018			00018018		Instruction at address 008 (SBV2)
0001001C			0001801C		Instruction at address 00C (SBV3)
00010020			00018020		

Figure I-CACHE-4 Memory Allocation by Cache Capacity

Address	Cache 4K	Cache 2K	Cache 1K	Cache off
000	$RAM (WAY1)	$RAM	$RAM	IRAM
200			IRAM	
400		IRAM		
600				
...		...	...	
000	$RAM (WAY2)	$RAM	$RAM	IRAM
200			IRAM	
400		IRAM		
600				

Figure 2-4 I-Cacheable Area



3. Operating mode conditions

● Cache status in various operating modes

The table below indicates the prevailing state for disable and flush when the associated bit is changed by bit manipulation instruction, etc.

		Immediately after a Reset	Disable (FNAB=0)	Flushed
Cache Memory		Contents undefined	The preceding state is held. Rewriting is impossible while the cache is disabled.	The preceding state is held.
Tag	Address Tag	Contents undefined	The preceding state is held. Rewriting is impossible while the cache is disabled.	The preceding state is held.
	Sub-block Valid Bit	Contents undefined	The preceding state is held. Rewriting is impossible while the cache is disabled.	The preceding state is held.
	LRU Bit	Contents undefined	The preceding state is held. Rewriting is impossible while the cache is disabled.	The preceding state is held.
	Entry Lock Bit	Contents undefined	The preceding state is held. Rewriting is impossible while the cache is disabled.	The preceding state is held. (entry lock release is required)
	TAG Valid Bit	Contents undefined	The preceding state is held. Flushingis possible while the cache is disabled.	All entries are invalid.
RAM		Normal Mode	The preceding state is held. Flushing is possible while the cache is disabled.	The preceding state is held.
Control Register	Global Lock	Unlock	The preceding state is held. Rewriting is possible while the cache is disabled.	The preceding state is held.
	Auto lock Fail	No fail	The preceding state is held. Rewriting is possible while the cache is disabled.	The preceding state is held.
	Entry Auto Lock	Unlock	The preceding state is held. Rewriting is possible while the cache is disabled.	The preceding state is held.
	Entry Lock Release	No release	The preceding state is held. Rewriting is possible while the cache is disabled.	The preceding state is held.
	Enable	Disable	Disabled	The preceding state is held.
	Flush	Not flushed	The preceding state is held. Rewriting is possible while the cache is disabled.	Flushed in cycle following memory accessing. Reverts to 0 subsequently.

● Cache Entry Update

Cache entries are updated as shown in the following table.

	Unlock	Lock
Hit	Not updated	Not updated.
Miss	The memory data is loaded, and the cache entry data is updated.	Not updated at tag miss. Updated when sub-block invalid.

4. Cacheable areas in the instruction cache

- The instruction cache can cache data only in external bus space.
- Even when the contents of external memory are updated by DMA transfer, the instruction cache does not refresh its contents to be coherent with the new contents of the memory. In this case, flush the cache to give it coherence.
- Each chip select area can be set as a non-cacheable area. The penalty for this is one cycle compared to the cache off state. (See Section 5.5 "Setting Chip Select Areas" in Chapter 5 "EXTERNAL INTERFACE".)

5. Settings for handling the I-Cache

1) Initializing

To use the I-Cache, first, clear the cache contents. Erase the old data by setting the register **FLUSH** bit and **ELKR** bit to **1**.

```
ldi   #0x000003e7,r0        //  I-Cache control register address
ldi   #0B00000110,r1        //  FLUSH bit (bit 1)
                            //  ELKR bit (bit 2)
stb   r1,@r0                //  Writing to register
```

The cache is now initialized.

2) Enabling (turning ON) cache

To enable the I-Cache, set the ENAB bit to 1.

```
ldi   #0x000003e7,r0        // I-Cache control register address
ldi   #0B00000001,r1        // ENAB bit (bit 0)
stb   r1,@r0                // Writing to register
```

All subsequently-accessed instructions will be cached.

Cache can be validated and initialized at the same time.

```
ldi   #0x000003e7,r0        // I-Cache control register address
ldi   #0B00000111,r1        // ENAB bit (bit 0)
                            //  FLUSH bit (bit 1)
                            //  ELKR bit (bit 2)
stb   r1,@r0                //  Writing to register
```

3) Disabling (turning OFF) cache

To disable the I-Cache, set the ENAB bit to 0.

```
ldi   #0x000003e7,r0          // I-Cache control register address
ldi   #0B00000000,r1          // ENAB bit (bit 0)
stb   r1,@r0                  //  Writing to register
```

In the resultant state (same as state prevailing after reset), there appears to be no cache. The cache can be turned off if the processing may experience problems due to cache overhead

4) Locking all cached instructions

To lock all the currently-cached instructions in the I-Cache, set the register **GBLK** bit to 1. The **ENAB** bit must also be set to 1. If it is not, the cache is turned OFF, so instructions locked in the cache cannot be used.

```
ldi   #0x000003e7,r0          //  I-Cache control register address
ldi   #0B00100001,r1          //  ENAB bit (bit 0)
                              //  GBLK bit (bit 5)
stb   r1,@r0                  //  Writing to register
```

5) Locking specific cached instructions

To lock a specific group of instructions (e.g., subroutines) in the cache, set the EOLK bit to 1 before executing such instructions.

Instructions locked in this manner are accessed rapidly as if using high-speed internal ROM.

```
ldi   #0x000003e7,r0          //  I-Cache control register address
ldi   #0B00001001,r1          //  ENAB bit (bit 0)
                              //  EOLK bit (bit 3)
stb   r1,@r0                  //  Writing to register
```

The above instruction lock becomes effective starting with the instruction next to the stb instruction although it depends on the memory wait count. Set the EOLK bit to 0 when the group of instructions which want to lock is ended.

```
ldi   #0x000003e7,r0          //  I-Cache control register address
ldi   #0B00000101,r1          // ENAB bit (bit 0)
                              //  EOLK bit (bit 3)
stb   r1,@r0                  //  Writing to register
```

6) Unlocking cached instructions

To release the lock, proceed as follows.

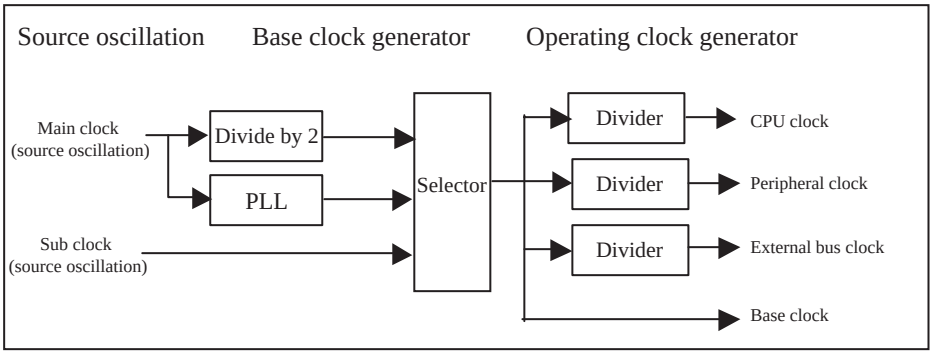
```
ldi   #0x000003e7,r0          // I-Cache control register address
ldi   #0B00000000,r1          //Cash disabled
stb   r1,@r0                  // Writing to register
ldi   #0B00000100,r1          // ELKR bit (bit 2)
stb   r1,@r0                  //  Writing to register
```

Only lock information is released; locked instructions are replaced sequentially with new instructions according to the state of the LRU bit.

Chapter 13 Clock Control

1. Overview

The clock control circuit consists of the source oscillator, base clock generator, and operating clock generator. The circuit supports a range of clock speeds from the high speed clock (100MHz maximum) to the low speed clock (32.768kHz).



2. Features

■ Source oscillation

- Main clock ($F_{CL-MAIN}$): 4MHz
  Input from the X0/X1 pins and used as the high speed clock
- Subclock (F_{CL-SUB}): 32.768kHz
  Input from the X0A/X1A pins and used as the low speed clock
- Subclock (F_{CL-SUB}): 100kHz
  RC oscillator and used as the low speed clock

■ Base clock (F): Selectable from 3 different clocks

- Main PLL (programmable) : $F_{CL-MAIN}$ x (MxN)\M
- Main clock divided by 2 : $F_{CL-MAIN}$ divided by 2
- Subclock : F_{CL-SUB}

■ Operating clocks: Selectable from 16 different speeds

- **CPU clock (CLKB):** F**/1, /2, /3, /4, /5, /6, /7, /8, ..., /16**

  The clock used by the CPU, internal memory, and internal buses. The circuits that use this clock are as follows.
  - CPU, internal RAM, internal ROM, bit search module,
  - I bus, D bus, F bus, X bus
  - On-chip debug support unit (DSU)
- **Peripheral clock (CLKP):** F**/1, /2, /3, /4, /5, /6, /7, /8, ..., /16**

  The clock used by the peripheral functions and peripheral bus. The circuits that use this clock are as follows.
  - Peripheral bus
  - Clock controller (bus interface unit only)
  - Interrupt controller
  - I/O ports
  - External interrupt inputs, UART, 16-bit timer, and similar peripheral functions

- **External bus clock (CLKT):** F**/1, /2, /3, /4, /5, /6, /7, /8, ..., /16**

  The clock used by the external bus expansion interface. The circuits that use this clock are as follows.
  - External bus expansion interface
  - External CLK output

3. Configuration

Figure 3-1 Configuration Diagram

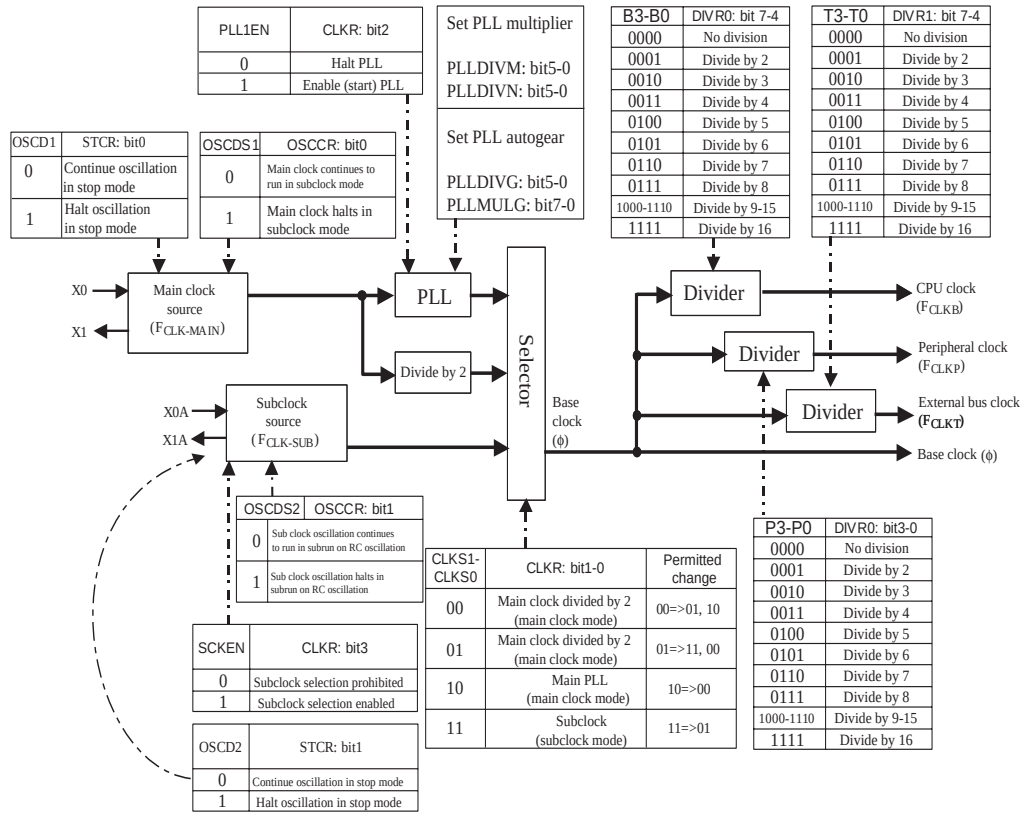


Figure 3-2 Register List

Clock Control

Address	Bit 7	6	5	4	3	2	1	0		
00486H	B3	B2	B1	B0	P3	P2	P1	P0	DIVR0	(Operating clock division setting 0)
00487H	T3	T2	T1	T0	---	---	---	---	DIVR1	(Operating clock division setting 1)
00484H	---	---	---	---	SCKEN	PLL1EN	CLKS1	CLKS0	CLKR	(Clock source control)
0048AH	---	---	---	---	---	---	OSCDS2	OSCDS1	OSCCR	(Oscillation control)
00481H	STOP	SLEEP	HIZ	SRST	OS1	OS0	OSCD2	OSCD1	STCR	(Standby control)

4. Registers

4.1 CLKR: Clock Source Control Register

Selects the clock source for the base clock used to run the MCU and controls the PLL.

- **CLKR: Address 0484h (Access: Byte)**

7	6	5	4	3	2	1	0	bit
-	-	-	-	SCKEN	PLL1EN	CLKS1	CLKS0	
X	X	X	X	0	0	0	0	Initial value (INIT pin input, watchdog reset)
X	X	X	X	X	X	X	X	Initial value (Software reset)
R/W0	R/W0	R/W0	R/W0	R/W	R/W	R/W	R/W	Attribute

(See "Meaning of Bit Attribute Symbols (Page No.10)" for details of the attributes.)

- Bit7-4: Reserved bitAlways write "0" to this bit. The read value is the value written.
- Bit3: Subclock select enable

SCKEN	Function
0	Prohibit subclock selection [Initial value]
1	Enable subclock selection

- Modifying the subclock selection enable bit (SCKEN) while the subclock is selected as the clock source (CLKS[1:0]="11") is prohibited (result is not guaranteed). Only modify the setting when the main clock is selected.
  (See the explanation for the clock source selection bits (CLKS[1:0]) for details of how to change the clock source.)

- Bit2: Enable main PLL operation

PLL1EN	Function
0	Halt main PLL (Initial value)
1	Enable main PLL operation

- Modifying the main PLL operation enable bit (PLL1EN) while the main PLL is selected as the clock source (CLKS[1:0]="10") is prohibited.
- Modifying the main PLL operation enable bit (PLL1EN) while the clock autogear function is active (gear up or gear down) is prohibited. Always check the gear status flags before changing the PLL state (see chapter "Clock Auto Gear Up/Down" on P. 213).
- If the main clock oscillation is halted (STCR.OSCD1="1"), the main PLL halts during stop mode even if the PLL enable bit (PLL1EN) is set to "1". If main PLL operation is enabled (PLL1EN="1"), the main clock operates using the PLL after recovering from stop mode.

  (See the explanation for the clock source selection bits (bits 1-0:) for details of changing the clock source.)

- Bit1-0: Clock source selection

CLKS1	CLKS0	Clock source setting	Mode
0	0	Main clock input from X0/X1 divided by 2 (Initial value)	Main clock mode
0	1	Main clock input from X0/X1 divided by 2	Main clock mode
1	0	Main PLL	Main clock mode
1	1	Subclock	Subclock mode

- When changing the clock mode, the value of CLKS0 may not be modified while CLKS1 is "1".

The table below lists the cases in which the CLKS1- CLKS0 bits may be modified.

- There is no setting to select the subclock divided by 2.

- After setting “11B” (subclock), insert one or more NOP instructions.
- Selecting the subclock as the clock source is prohibited while the subclock selection enable bit (SCKEN) is “0”. (See table for details.)

Table 4-1 Cases When the CLKS1 and CLKS0 Bits May or May Not be Modified

Modify permitted	Modify not permitted
“00” → “01” or “10”	“00” → “11”
“01” → “11” or “00”	“01” → “10”
“10” → “00”	“10” → “01” or “11”
“11” → “01”	“11” → “00” or “10”

Example: To select the subclock after an INIT reset, first write “01_B” and then write “11_B” (subclock).

(See “8. Caution (Page No.205)”.)

4.2 DIVR0: Clock Division Setting Register 0

Sets the division ratio for the clocks used for internal device operation.

DIVR0: Address 0486h (Access: Byte, Half-word)

7	6	5	4	3	2	1	0	bit
B3	B2	B1	B0	P3	P2	P1	P0	
0	0	0	0	0	0	1	1	Initial value (INIT pin input, watchdog reset)
X	X	X	X	X	X	X	X	Initial value (software reset)
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Attribute

(See “Meaning of Bit Attribute Symbols (Page No.10)” for details of the attributes.)

- Sets up the clock for the CPU and internal buses (CLKB), and the clock for the peripheral circuits and peripheral bus (CLKP).

- Bit7-4: CLKB division selection

B3-B0	CPU clock (CLKB) division ratio
0000	Φ/1 (initial value)
0001	Φ/2
0010	Φ/3
0011	Φ/4
0100	Φ/5
0101	Φ/6
0110	Φ/7
0111	Φ/8
1000	Φ/9
1001	Φ/10
1010	Φ/11
1011	Φ/12
1100	Φ/13
1101	Φ/14
1110	Φ/15
1111	Φ/16

  - **Do not change the division ratio with B3-B0 if current CLKB frequency is equal or above 80MHz !**
  - Sets the clock division ratio for the clock used by the CPU, internal memory, and internal buses (CLKB). The 16 options listed in the table are available.
  - Do not set a division ratio that exceeds the maximum operating frequency of the device.

- Bit3-0: CLKP division selection

P3-P0	Peripheral clock (CLKP) division ratio
0000	Φ/1
0001	Φ/2
0010	Φ/3
0011	Φ/4 (initial value)
0100	Φ/5
0101	Φ/6
0110	Φ/7
0111	Φ/8
1000	Φ/9
1001	Φ/10

1010	Φ/11
1011	Φ/12
1100	Φ/13
1101	Φ/14
1110	Φ/15
1111	Φ/16

- Sets the clock division ratio for the clock used by the peripheral circuits and peripheral bus (CLKP). The 16 options listed in the table are available.
- Do not set a division ratio that exceeds the maximum operating frequency of the MCU.

4.3 DIVR1: Clock Division Setting Register 1

Sets the division ratio for the clocks used for internal device operation.

- **DIVR1: Address 0487h (Access: Byte, Half-word)**

7	6	5	4	3	2	1	0	bit
T3	T2	T1	T0	–	–	–	–	
0	0	0	0	0	0	0	0	Initial value (INIT pin input, watchdog reset)
X	X	X	X	X	X	X	X	Initial value (software reset)
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Attribute

(See “Meaning of Bit Attribute Symbols (Page No.10)” for details of the attributes.)

Sets the clock division ratio (relative to the base clock) for the clock used by the external bus interface (CLKT).

- Bit7-4: CLKT division selection

T3-T0	External bus clock (CLKT) division ratio
0000	Φ/1 (initial value)
0001	Φ/2
0010	Φ/3
0011	Φ/4
0100	Φ/5
0101	Φ/6
0110	Φ/7
0111	Φ/8
1000	Φ/9
1001	Φ/10
1010	Φ/11
1011	Φ/12
1100	Φ/13
1101	Φ/14
1110	Φ/15
1111	Φ/16

  - Sets the clock division ratio for the clock used by the external bus interface (CLKT). The 16 options listed in the table are available.
  - Do not set a division ratio that exceeds the maximum operating frequency of the device.
  - If you modify the CLKP division selection bits, the new division ratio applies from the next clock after the setting is modified.
- Bit3-0: Reserved bit Always write “0” to this bit. The read value is the value written.

4.4 CSCFG: Clock Source Configuration Register

This register controls the main clock oscillation in subclock mode

- **CSCFG: Address 04AEh (Access: Byte)**

7	6	5	4	3	2	1	0	bit
EDSUEN	PLLLOCK	RCSEL	MONCKI	CSC3	CSC2	CSC1	CSC0	
0	X	0	0	0	0	0	0	Initial value (INIT pin input, watchdog reset)
X	X	X	X	X	X	X	X	Initial value (software reset)
R/W	R	R/W	R/W	R/W	R/W	R/W	R/W	Attribute

(See “Meaning of Bit Attribute Symbols (Page No.10)” for details of the attributes.)

- bit7: EDSU/MPU Enable

EDSUEN	Function
0	EDSU/MPU is (clock) disabled [Initial value]
1	EDSU/MPU is (clock) enabled

- bit6: PLL Lock

PLLLOCK	Function
0	PLL is in the un-locked state
1	PLL is in the locked state

- bit5: RC Oscillator Selector

RCSEL	Function
0	RC oscillation is set to 100 kHz [Initial value]
1	RC oscillation is set to 2 MHz

The selected oscillation frequency is supplied to Clock Control Unit (for Subclock Operation) and Flash Security Unit (change the oscillation to 2 MHz for faster CRC generation). Hardware Watchdog (RC based Watchdog), Real Time Clock, Calibration Unit, LCD and Clock Supervisor Module are always supplied with 100 kHz independent of this setting.

- bit4: Clock Monitor MONCLK inverter

MONCKI	Function
0	MONCLK mark level is low [Initial value]
1	MONCLK mark level is high

See chapter “Clock Monitor (Page No.943)” about information about this function.

- Bit3-0: Clock Source Selection

CSC3-CSC0	Function
--00	Real Time Clock is sourced by Main Oscillation
--01	Real Time Clock is sourced by Sub Oscillation
--10	Real Time Clock is sourced by RC Oscillation
--11	Setting prohibited
-0--	Subclock Calibration is sourced by Sub Oscillation

-1--	Subclock Calibration is sourced by RC Oscillation
0---	LCD Controller is sourced by Sub Oscillation
1---	LCD Controller is sourced by RC Oscillation

4.5 OSCCR: Oscillation Control Register

This register controls the main clock oscillation in subclock mode

- **OSCCR: Address 04CCh (Access: Byte)**

7	6	5	4	3	2	1	0	bit
–	–	–	–	–	–	OSCDS2	OSCDS1	
X	X	X	X	0	0	0	0	Initial value (INIT pin input, watchdog reset)
X	X	X	X	X	X	X	X	Initial value (software reset)
RX/WX	RX/WX	RX/WX	RX/WX	RX/WX	RX/WX	R/W	R/W	Attribute

(See “Meaning of Bit Attribute Symbols (Page No.10)” for details of the attributes.)

- bit7-2: Undefined bit

  Writing does not affect the operation. The read value is undefined.

- bit1: Halt sub clock oscillation in subrun mode on RC oscillation.

OSCDS2	Operation when written to	Read value meaning
0	Does not halt sub clock oscillation during subrun on RC oscillation.	Sub clock mode can be selected after the oscillation stabilization time elapses.
1	Halt sub clock oscillation during subrun on RC oscillation.	Selecting sub clock mode is prohibited.

  - When the sub clock oscillation is selected (CSVCR.SCKS=’0’), specifying the sub clock oscillation to halt during subrun mode. Setting OSCDS2=“1” is prohibited.

(See “8. Caution (Page No.205)”.)

- bit0: Halt main clock oscillation in subrun mode.

OSCDS1	Operation when written to	Read value meaning
0	Does not halt main clock oscillation during subrun mode.	Main clock mode can be selected after the oscillation stabilization time elapses.
1	Halt main clock oscillation during subrun mode.	Selecting main clock mode is prohibited.

  - When the main clock oscillation is selected (CLKS[1:0]=“00”, “01”, “10”), specifying the main clock to halt during subclock mode (OSCDS1=“1”) is prohibited.

(See “8. Caution (Page No.205)”.)

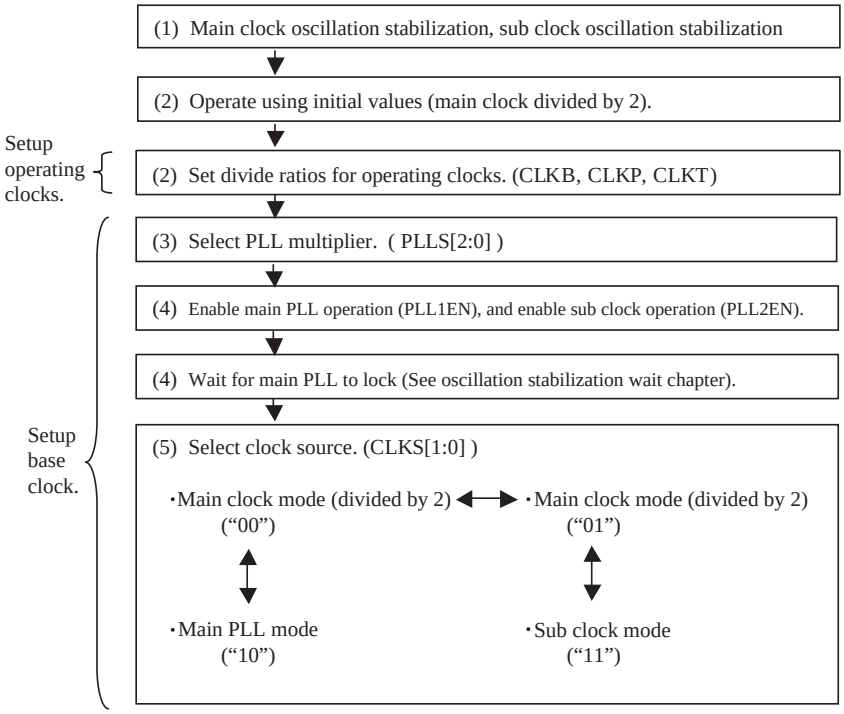
  - The following table lists the settings which are necessary to operate with the different clock sources. The second statement of the table is, when the clock sources can become switched off.

	CLKS[1:0]	SCKS	switch off Main oscillation	switch off sub oscillation
Main oscillation	00, 01, 10	-	prohibited	prohibited
Sub oscillation	11	0	OSCDS1 = 1	prohibited
RC oscillation	11	1	OSCDS1 = 1	OSCDS2 = 1

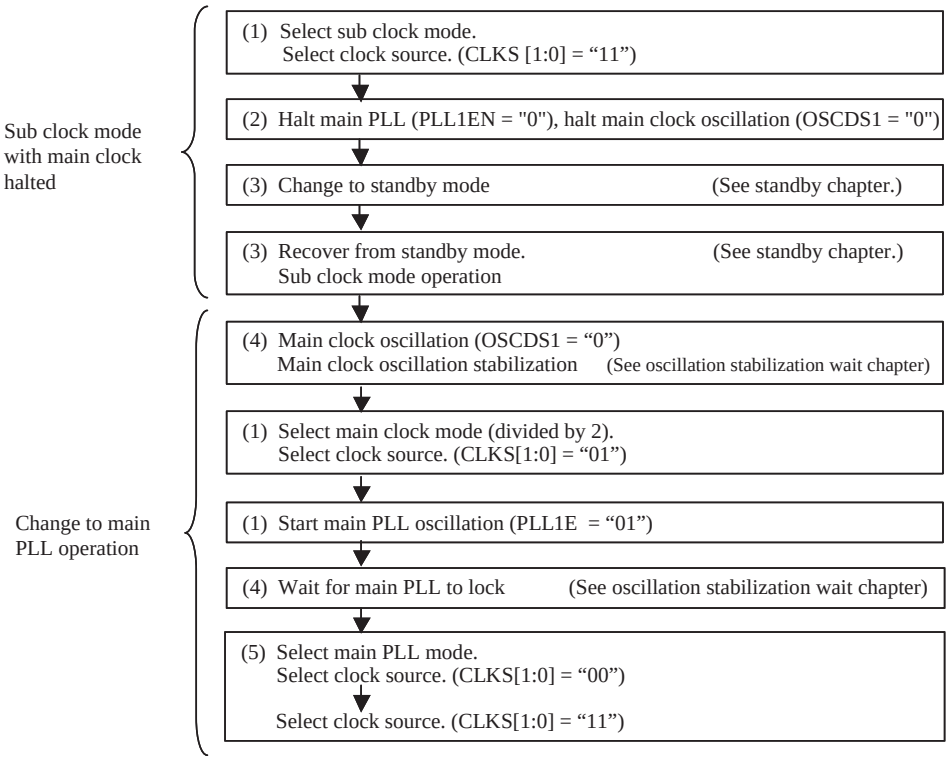
5. Operation

This section describes how to setup and switch between clocks.

5.1 Clock Setup Sequence (Example)



5.2 Halting and Restarting the Main Clock Oscillation During Subclock Mode (Example)



5.3 Notes

■ Main PLL control

After initialization, the main PLL oscillation is halted. While halted, the output of the main PLL cannot be selected as the clock source.

After the program starts, first set the multiplier for the main PLL that you want to use as the clock source and then, after allowing a time for the main PLL to lock, change the clock source. The recommended method for waiting for the main PLL to lock is to use the timebase timer interrupt.

You cannot halt the main PLL while the output of the main PLL is selected as the clock source.

Writing to the register has no effect. If you wish to stop the main PLL such as when changing to stop mode, first select the main clock divided by 2 as the clock source and then halt the main PLL.

If the main clock oscillation is set to halt during stop mode by the main clock oscillation stop bit (STCR.OSCD1="1"), the main PLL stops automatically when the MCU changes to stop mode and you do not need to halt the main PLL (CLKR.PLL1EN="0") explicitly beforehand. The main PLL also restarts automatically on recovering from stop mode. When the oscillation is not set to halt during stop mode (STCR.OSCD1="0"), the main PLL does not stop automatically. In this case, halt the main PLL explicitly (CLKR.PLL1EN="0") before changing to stop mode.

■ Main PLL multiplier

When changing the main PLL multiplier setting to a value different to the initial value, set this before or at the same time as you enable the main PLL after program execution starts. After changing the multiplier setting, wait for the main PLL lock time before switching the clock source. The recommended method for waiting for the main PLL to lock is to use the timebase timer interrupt.

To modify the main PLL multiplier setting during normal operation, first change the clock source to something other than the main PLL. As in the above case, after changing the multiplier setting, wait for the main PLL lock time before changing the clock source.

The main PLL multiplier setting can be changed while the main PLL is in use. In this case, the MCU automatically goes to the oscillation stabilization wait state after the multiplier setting is modified and program execution halts for the time specified as the oscillation stabilization wait time. Program execution doe not halt when changing to a clock source other than the main PLL.

■ Clock division

The clocks used to drive the internal operation of the device allow division ratios relative to the base clock to be set independently for each clock. This function allows the optimum operating frequency to be selected for each circuit.

The division ratios are set in the operating clock division setting registers (DIVR0 and DIVR1). These registers contain 4-bit settings that specify the ratio for each clock. The division ratio relative to the base clock = (register value+1). The duty ratio is always 50, even if an odd numbered division ratio is set.

If a setting is modified, the new setting applies from the next rising edge of the clock.

The division ratio settings are not initialized by an operation reset (RST) and the settings from before the reset are maintained. The ratio settings are only initialized by a settings initialization reset (INIT). When changing the clock source from its initial setting to a high speed clock, always set the division ratio first.

Device operation is not guaranteed if the result of the clock source selection, main PLL multiplier setting, and division ratio setting is a frequency that is higher than the maximum permitted frequency. Please take care with these settings. (In particular, take care with the sequence in which you change clock source settings.)

6. Settings

Table 6-1 Settings for Operating at 1/2 of the Main Clock

Setting	Setting register	Setting procedure*
Clock source selection	Clock source control register (CLKR)	See 7.3

*: For the setting procedure, refer to the section indicated by the number.

Table 6-2 Settings for Operating Using the Main PLL

Setting	Setting register	Setting procedure*
Main PLL operation enable	Clock source control register (CLKR)	See 7.1
Clock source selection		See 7.3

*: For the setting procedure, refer to the section indicated by the number.

Table 6-3 Settings for Operating Using the Subclock

Setting	Setting register	Setting procedure*
Subclock selection enable	Clock source control register (CLKR)	See 7.1
Clock source selection		See 7.3

*: For the setting procedure, refer to the section indicated by the number.

Table 6-4 Settings for Selecting the Division Ratio for the Operating Clocks

Setting	Setting register	Setting procedure*
Clock source selection	Clock source control register (CLKR)	See 7.3
Operating clock division ratio selection	Operating clock division setting registers (DIVR0, DIVR1)	See 7.4

*: For the setting procedure, refer to the section indicated by the number.

7. Q & A

7.1 How do I enable or disable clock operation?

- There is no operation enable bit for the main clock. Main clock operation is always enabled.
  (Halting the oscillation in subclock mode or stop mode is handled separately.)
- Main PLL operation is enabled by the main PLL operation enable bit (CLKR.PLL1EN).

Operation	Main PLL operation enable bit (PLL1EN)
To halt the main PLL	Set to “0”.
To enable operation of the main PLL	Set to “1”.

  Initially, the PLL is halted and therefore PLL operation must be enabled and the PLL started after setting the PLL multiplier ratio.
- The subclock on the MB91460 does not halt and therefore no corresponding operation enable bit is provided.
  Instead, the subclock selection enable bit (CLKR.SCKEN) is used.

Operation	Subclock selection enable bit (SCKEN)
Subclock selection prohibited	Set to “0”.
To enable selection of the subclock	Set to “1”.

7.2 How do I select the main PLL multiplier ratio?

- The PLL multiplier can be set by using the PLL interface registers PLLDIVM and PLLDIVN (see chapter "PLL Interface" on P. 207).

7.3 How do I select the operating clock source?

Use the clock source selection bits (CLKR.CLKS[1:0]) to select main clock divided by 2, main PLL, or the subclock as the operating clock source.

Operating clock source	Clock source selection bits (CLKS[1:0])
To change from the initial value to the main clock divided by 2	Set initial values “00” or “01”.
To change from the initial value to the main PLL	Change from the initial values “00” to “10”.
To change from the initial value to the subclock	First change from the initial values “00” to “01”, and then to “11”.
To change from the subclock to the main clock divided by 2	Change from “11” to “01”.
To change from the subclock to main PLL	First change from “11” to “01”, next set “00”, and then set “10”.
To change from main PLL to the main clock divided by 2	Change from “10” to “00”.
To change from main PLL to the subclock	First change from “10” to “00”, next set “01”, and then set “11”.

7.4 How do I set the operation clock division ratios?

- CPU clock setting

The CPU clock setting is set using the CLKB division ratio selection bits (DIVR0.B[3:0]).

PLL multiplier ratio	CLKB division ratio selection bits(B[3:0])	Example frequency	
		When F_Φ = 32MHz	When F_Φ = 16MHz
To select no division	Set to “0000”.	F_{CLKB} = 32.0MHz	F_{CLKB} = 16.0MHz
To select divide by 2	Set to “0001”.	F_{CLKB} = 16.0MHz	F_{CLKB} = 8.00MHz
To select divide by 3	Set to “0010”.	F_{CLKB} = 10.6MHz	F_{CLKB} = 5.33MHz
To select divide by 4	Set to “0011”.	F_{CLKB} = 8.00MHz	F_{CLKB} = 4.00MHz
To select divide by 5	Set to “0100”.	F_{CLKB} = 6.40MHz	F_{CLKB} = 3.20MHz
To select divide by 6	Set to “0101”.	F_{CLKB} = 5.33MHz	F_{CLKB} = 2.66MHz
To select divide by 7	Set to “0110”.	F_{CLKB} = 4.57MHz	F_{CLKB} = 2.28MHz
To select divide by 8	Set to “0111”.	F_{CLKB} = 4.00MHz	F_{CLKB} = 2.00MHz
To select divide by 16	Set to “1111”.	F_{CLKB} = 2.00MHz	F_{CLKB} = 1.00MHz

- Peripheral clock setting

The peripheral clock setting is set using the CLKP division ratio selection bits (DIVR0.P[3:0]).

PLL multiplier ratio	CLKP division ratio selection bits (P[3:0])	Example frequency	
		When F_Φ = 32MHz	When F_Φ = 16MHz
To select no division	Set to “0000”.	F_{CLKP} = 32.0MHz	F_{CLKP} = 16.0MHz
To select divide by 2	Set to “0001”.	F_{CLKP} = 16.0MHz	F_{CLKP} = 8.00MHz
To select divide by 3	Set to “0010”.	F_{CLKP} = 10.6MHz	F_{CLKP} = 5.33MHz
To select divide by 4	Set to “0011”.	F_{CLKP} = 8.00MHz	F_{CLKP} = 4.00MHz
To select divide by 5	Set to “0100”.	F_{CLKP} = 6.40MHz	F_{CLKP} = 3.20MHz
To select divide by 6	Set to “0101”.	F_{CLKP} = 5.33MHz	F_{CLKP} = 2.66MHz
To select divide by 7	Set to “0110”.	F_{CLKP} = 4.57MHz	F_{CLKP} = 2.28MHz
To select divide by 8	Set to “0111”.	F_{CLKP} = 4.00MHz	F_{CLKP} = 2.00MHz
To select divide by 16	Set to “1111”.	F_{CLKP} = 2.00MHz	F_{CLKP} = 1.00MHz

- Setting for the external bus clock

The setting for the external bus clock is set using the CLKT division ratio selection bits (DIVR1.T[3:0]).

PLL multiplier ratio	CLKT division ratio selection bits (T[3:0])	Example frequency	
		When F_Φ = 32MHz	When F_Φ = 16MHz
To select no division	Set to “0000”.	F_{CLKT} = 32.0MHz	F_{CLKT} = 16.0MHz
To select divide by 2	Set to “0001”.	F_{CLKT} = 16.0MHz	F_{CLKT} = 8.00MHz
To select divide by 3	Set to “0010”.	F_{CLKT} = 10.6MHz	F_{CLKT} = 5.33MHz
To select divide by 4	Set to “0011”.	F_{CLKT} = 8.00MHz	F_{CLKT} = 4.00MHz
To select divide by 5	Set to “0100”.	F_{CLKT} = 6.40MHz	F_{CLKT} = 3.20MHz
To select divide by 6	Set to “0101”.	F_{CLKT} = 5.33MHz	F_{CLKT} = 2.66MHz
To select divide by 7	Set to “0110”.	F_{CLKT} = 4.57MHz	F_{CLKT} = 2.28MHz
To select divide by 8	Set to “0111”.	F_{CLKT} = 4.00MHz	F_{CLKT} = 2.00MHz
To select divide by 16	Set to “1111”.	F_{CLKT} = 2.00MHz	F_{CLKT} = 1.00MHz

7.5 How do I halt the main clock in sub clock mode?

Set using the “halt main clock oscillation in subclock mode” bit (OSCCR.OSCDS1).

Operation in subclock mode	Halt main clock oscillation in subclock mode bit (OSCDS1)
To not halt the main clock	Set to “0”.
To halt the main clock	Set to “1”.

(See “8. Caution (Page No.205)”.)

7.6 How do I halt the sub clock in sub clock on RC oscillator mode?

Set using the “halt main clock oscillation in subclock mode” bit (OSCCR.OSCDS2).

Operation in subclock mode	Halt sub clock oscillation in subclock on RC oscillation mode bit (OSCDS2)
To not halt the sub clock	Set to “0”.
To halt the sub clock	Set to “1”.

(See “8. Caution (Page No.205)”.)

7.7 How do I halt the sub clock in main clock mode?

Operation in mainclock mode	Halt sub clock oscillation in main mode bit (OSCD2)
To not halt the sub clock	Set to “0”.
To halt the sub clock	Set to “1”.

8. Caution

- Operation is not guaranteed if the clock source selection, main PLL multiplier setting, and division ratio setting result in a frequency that exceeds the maximum.

- Take care with the sequence in which you set or modify the clock source selection.

- When the main clock oscillation is set to halt during subclock mode (OSCDS1 = “1”), selecting the main clock (CLKS[1:0])=“00”, “01”, or “10”) is prohibited. To select the main clock, set (OSCDS1=“0”) and then change to the main clock after waiting for the main clock oscillation to stabilize. Use the main clock oscillation stabilization wait timer to provide the wait time. See “Chapter 22 Main Oscillation Stabilisation Timer (Page No.293)” for details.

- When the sub clock oscillation is set to halt during subclock mode (OSCDS2 = “1”), selecting the sub clock (CSVCR.SCKS=“0”) is prohibited. To select the sub clock, set (OSCDS2=“0”) and then change to the sub clock after waiting for the sub clock oscillation to stabilize. Use the sub clock oscillation stabilization wait timer to provide the wait time. See “Chapter 23 Sub Oscillation Stabilisation Timer (Page No.303)” for details.

- When the main clock oscillation is halted (OSCDS1 = “1”) or the sub clock oscillation is halted (OSCDS2 = “1”) an oscillation stabilization wait time (for main clock or subclock) is also required if a reset (INIT) occurs that switches the clock source to the main clock. In this case, operation after the reset is not guaranteed if the wait time set in the oscillation stabilization time selection bits (STCR.OS[1:0]) does not satisfy the oscillation stabilization time requirement for the main clock.
  Always set the oscillation stabilization time selection bits (STCR.OS[1:0]) to a value that provides an adequate oscillation stabilization time for the main clock.
  In the case of an INIT reset triggered by the $\overline{\text{INIT}}$ pin, the “L” level input must be maintained for long enough for the main clock oscillation to stabilize.
  See “Chapter 18 Timebase Counter (Page No.249)” and “Chapter 22 Main Oscillation Stabilisation Timer (Page No.293)” for details of the oscillation stabilization wait.

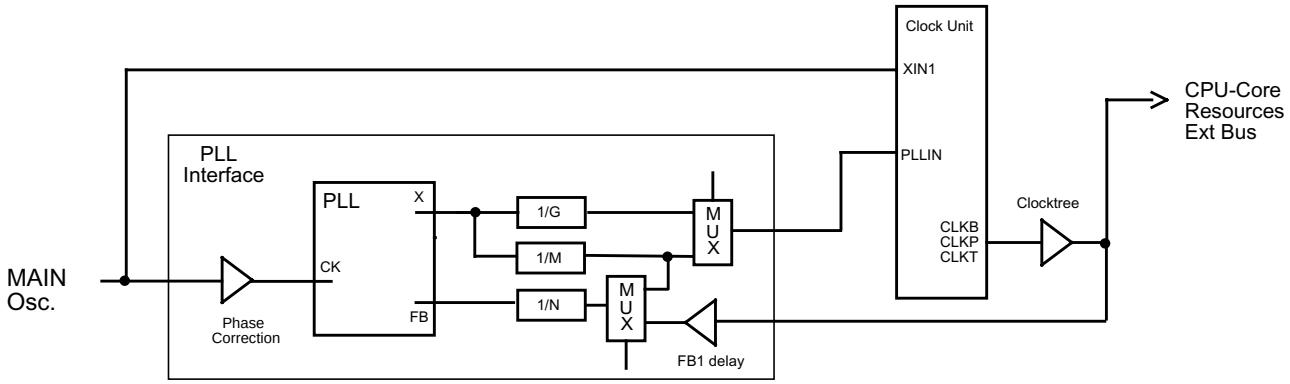
- When changing to stop mode, the main PLL must either be halted or de-selected. Either set the main clock oscillation halt bit (STCR.OSCD1 = “1”) to halt automatically or change the operating clock to main clock divided by two before changing to stop mode.

8.Caution

Chapter 14 PLL Interface

1. Overview

- This blockdiagram (simplified) shows the integration of the PLL and the PLL Interface with the multiplier control logic (1/M, 1/N for basic frequency multiplication and 1/G for clock auto gear).



2. Features

- Free programmable divide-by-M counter in the range of 1..16
- Free programmable divide-by-N counter in the range of 1..64
- Clock auto gear up/down function to prevent voltage drops and surges

3. Frequency calculation

- CLKB frequency is determined by :

  f(CLKB) = [Main-Osc * (PLLDIVM_DVM+1) *(PLLDIVN_DVN+1)] / [(PLLDIVM_DVM+1) * (DIVR0_B+1)]

- CLKP frequency is determined by :

  f(CLKP) = [Main-Osc * (PLLDIVM_DVM+1) *(PLLDIVN_DVN+1)] / [(PLLDIVM_DVM+1) * (DIVR0_P+1)]

- CLKT frequency is determined by :

  f(CLKT) = [Main-Osc * (PLLDIVM_DVM+1) *(PLLDIVN_DVN+1)] / [(PLLDIVM_DVM+1) * (DIVR1_T+1)]

4. Registers

4.1 PLL Control Registers

Controls the PLL multiplier ratio (divide-by-M and divide-by-N) and the automatic clock gear up/down function.

- **PLLDIVM: Address 048Ch (Access: Byte, Halfword, Word)**

7	6	5	4	3	2	1	0	bit
-	-	-	-	DVM3	DVM2	DVM1	DVM0	
0	0	0	0	0	0	0	0	Initial value (INIT pin input, watchdog reset)
0	0	0	0	X	X	X	X	Initial value (Software reset)
R0/W0	R0/W0	R0/W0	R0/W0	R/W	R/W	R/W	R/W	Attribute

(See “Meaning of Bit Attribute Symbols (Page No.10)” for details of the attributes.)

- Bit7-4: Reserved bits. Always write “0” to these bits.
- Bit3-0: PLL divide-by-M selection

DVM3-DVM0	PLL output divided-by-M (generates Φ: Base clock)
0000	Source (F_{CL-PLL}) : 1 (no division)
0001	Source (F_{CL-PLL}) : 2 (division by 2)
0010	Source (F_{CL-PLL}) : 3 (division by 3)
0011	Source (F_{CL-PLL}) : 4 (division by 4)
0100	Source (F_{CL-PLL}) : 5 (division by 5)
0101	Source (F_{CL-PLL}) : 6 (division by 6)
0110	Source (F_{CL-PLL}) : 7 (division by 7)
0111	Source (F_{CL-PLL}) : 8 (division by 8)
.....	.....
1111	Source (F_{CL-PLL}) : 16 (division by 16)

(Note) Even though it is possible to select no division ratio (:1) for the divide-by-M counter it is not recommended. The resulting output clock will have an odd clock duty ratio (direct PLL output). Always select at least a division ratio > 1 and an even division ratio (:2, :4, :6, etc.).

(Note) Even though it is possible to select an odd division ratio (:3, :5, :7, etc.) for the divide-by-M counter it is not recommended. The resulting output clock will have an odd clock duty ratio. Always select an even division ratio (:2, :4, :6, etc.).

(Note) The register value can not be changed once PLL is selected as clock source (CLKS[1:0]=”10”).

(Note) It is strongly recommended to disable the PLL (CLKR.PLL1EN=0) while or after changing the PLLDIVM and PLLDIVN registers and to enable the PLL (CLKR.PLL1EN=1) afterwards.

- **PLLDIVN: Address 048Dh (Access: Byte, Halfword, Word)**

7	6	5	4	3	2	1	0	bit
-	-	DVN5	DVN4	DVN3	DVN2	DVN1	DVN0	
0	0	0	0	0	0	0	0	Initial value (INIT pin input, watchdog reset)
0	0	X	X	X	X	X	X	Initial value (Software reset)
R0/WX	R0/WX	R/W	R/W	R/W	R/W	R/W	R/W	Attribute

(See “Meaning of Bit Attribute Symbols (Page No.10)” for details of the attributes.)

- Bit7-6: Reserved bits.The read value is always “0”.
- Bit5-0: PLL divide-by-N selection

DVN5-DVN0	Φ: Base clock divide-by-N (feedback to PLL)
000000	Base clock ($F_{CL-MAIN}$) : 1 (no division)
000001	Base clock ($F_{CL-MAIN}$) : 2 (division by 2)
000010	Base clock ($F_{CL-MAIN}$) : 3 (division by 3)
000011	Base clock ($F_{CL-MAIN}$) : 4 (division by 4)
000100	Base clock ($F_{CL-MAIN}$) : 5 (division by 5)
000101	Base clock ($F_{CL-MAIN}$) : 6 (division by 6)
000110	Base clock ($F_{CL-MAIN}$) : 7 (division by 7)
000111	Base clock ($F_{CL-MAIN}$) : 8 (division by 8)
.....	.....
111111	Base clock ($F_{CL-MAIN}$) : 64 (division by 64)

(Note) The register value can not be changed once PLL is selected as clock source (CLKS[1:0]=”10”).

(Note) It is strongly recommended to disable the PLL (CLKR.PLL1EN=0) while or after changing the PLLDIVM and PLLDIVN registers and to enable the PLL (CLKR.PLL1EN=1) afterwards.

- **PLLDIVG: Address 048Eh (Access: Byte, Halfword, Word)**

7	6	5	4	3	2	1	0	bit
-	-	-	-	DVG3	DVG2	DVG1	DVG0	
0	0	0	0	0	0	0	0	Initial value (INIT pin input, watchdog reset)
0	0	0	0	X	X	X	X	Initial value (Software reset)
R0/W0	R0/W0	R0/W0	R0/W0	R/W	R/W	R/W	R/W	Attribute

(See “Meaning of Bit Attribute Symbols (Page No.10)” for details of the attributes.)

- Bit7-4: Reserved bits.Always write “0” to these bits.
- Bit3-0: PLL auto gear start/end divide-by-G selection

DVG3-DVG0	PLL output divided-by-G start/end frequency (generates Φ: Base clock)
0000	Auto gear disabled (inital value)
0001	Source (F_{CL-PLL}) : 2 (division by 2)
0010	Source (F_{CL-PLL}) : 3 (division by 3)
0011	Source (F_{CL-PLL}) : 4 (division by 4)
0100	Source (F_{CL-PLL}) : 5 (division by 5)
0101	Source (F_{CL-PLL}) : 6 (division by 6)
0110	Source (F_{CL-PLL}) : 7 (division by 7)
0111	Source (F_{CL-PLL}) : 8 (division by 8)
.....	.....
1111	Source (F_{CL-PLL}) : 16 (division by 16)

(Note) See chapter 6. Clock Auto Gear Up/Down for detailed information on how to use this function.

(Note) Even though it is possible to select an odd division ratio (:3, :5, :7, etc.) for the divide-by-G counter it is not recommended. Always select an even division ratio (:2, :4, :6, etc.).

(Note) The register value can not be changed once PLL is selected as clock source (CLKS[1:0]=”10”).

- **PLLMULG: Address 048Fh (Access: Byte, Halfword, Word)**

7	6	5	4	3	2	1	0	bit
MLG7	MLG6	MLG5	MLG4	MLG3	MLG2	MLG1	MLG0	
0	0	0	0	0	0	0	0	Initial value (INIT pin input, watchdog reset)
X	X	X	X	X	X	X	X	Initial value (Software reset)
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Attribute

(See "Meaning of Bit Attribute Symbols (Page No.10)" for details of the attributes.)

- Bit7-0: PLL auto gear divide-by-G step multiplier selection

MLG5-MLG0	Divide-by-G step multiplier
00000000	Divide-by-G step x 1 (multiply by 1)
00000001	Divide-by-G step x 2 (multiply by 2)
00000010	Divide-by-G step x 3 (multiply by 3)
00000011	Divide-by-G step x 4 (multiply by 4)
00000100	Divide-by-G step x 5 (multiply by 5)
00000101	Divide-by-G step x 6 (multiply by 6)
00000110	Divide-by-G step x 7 (multiply by 7)
00000111	Divide-by-G step x 8 (multiply by 8)
.....	.....
11111111	Divide-by-G step x 256 (multiply by 256)

(Note) See chapter 6. Clock Auto Gear Up/Down for detailed information on how to use this function.

(Note) The register value can not be changed once PLL is selected as clock source (CLKS[1:0]="10").

- **PLLCTRL: Address 0490h (Access: Byte, Halfword, Word)**

7	6	5	4	3	2	1	0	bit
-	-	-	-	IEDN	GRDN	IEUP	GRUP	
0	0	0	0	0	0	0	0	Initial value (INIT pin input, watchdog reset)
0	0	0	0	X	X	X	X	Initial value (Software reset)
R/W	R/W	R/W	R/W	R/W	RM1/W	R/W	RM1/W	Attribute

(See "Meaning of Bit Attribute Symbols (Page No.10)" for details of the attributes.)

- Bit7-4: Reserved bitThe read value is always "0".
- Bit3: Interrupt Enable Gear DOWN.

IEDN	Function
0	Gear DOWN interrupt disabled [Initial value]
1	Gear DOWN interrupt enabled

- Bit2: Interrupt Flag Gear DOWN.

GRDN	Function
0	Gear DOWN interrupt not active [Initial value]
1	Gear DOWN interrupt active

- While switching from clock source PLL to clock source oscillator this flag is set when the divide-by-G

counter reaches the programmed end value.

- This bit is read as “1” at a Read-Modify-Write instructions. Writing “1” has no effect.

- Bit1: Interrupt Enable Gear UP.

IEUP	Function
0	Gear UP interrupt disabled [Initial value]
1	Gear UP interrupt enabled

- Bit2: Interrupt Flag Gear UP.

GRUP	Function
0	Gear UP interrupt not active [Initial value]
1	Gear UP interrupt active

- While switching from clock source oscillator to clock source PLL this flag is set when the divide-by-G counter reaches the end value defined by the divide-by-M counter.
- This bit is read as “1” at a Read-Modify-Write instructions. Writing “1” has no effect.

5. Recommended Settings

PLL Input (CK) [MHz]	Frequency Parameter		Clockgear Parameter		PLL Output (X) [MHz]	Core base Clock [MHz]
	DIVM	DIVN	DIVG	MULG		
4	2	25	16	24	200	100
4	2	24	16	24	192	96
4	2	23	16	24	184	92
4	2	22	16	24	176	88
4	2	21	16	20	168	84
4	2	20	16	20	160	80
4	2	19	16	20	152	76
4	2	18	16	20	144	72
4	2	17	16	16	136	68
4	2	16	16	16	128	64
4	2	15	16	16	120	60
4	2	14	16	16	112	56
4	2	13	16	12	104	52
4	2	12	16	12	96	48
4	2	11	16	12	88	44
4	4	10	16	24	160	40
4	4	9	16	24	144	36
4	4	8	16	24	128	32
4	4	7	16	24	112	28
4	6	6	16	24	144	24
4	8	5	16	28	160	20
4	10	4	16	32	160	16
4	12	3	16	32	144	12

- Important remark: Not all settings which are shown in this table are available for all devices. Please consult the available datasheet for each device for the maximum allowed PLL output and the allowed maximum frequencies for each clock domain (CLKB, CLKP and CLKT) respectively.

6. Clock Auto Gear Up/Down

To avoid voltage drops and surges when switching the clock source from oscillator to high frequency PLL/ DLL output (or vice versa), a clock smooth gear-up and gear-down circuitry is implemented with the PLL interface.

The main functionality is implemented using two divide-by counters (divide-by-M and divide-by-G counter), where one supplies the PLL feedback always with the target frequency (divide-by-M counter), and the other (divide-by-G counter) which increases the frequency from a programmable frequency division given by the divide-by-G setting (DIVG) up to the target frequency given by the divide-by-M setting (DIVM), or decreases the frequency from the divide-by-M setting (DIVM) down to the programmable end frequency (DIVG).

In this sense only a setting of DIVG > DIVM is a valid clock gear specification to scale the system clock from slower frequencies to faster frequencies (when gearing up) and from faster frequencies to slower ones (when gearing down).

The frequency steps are performed in multiple of the PLL output frequency, e,g, the setting of: Oscillator = 4 MHz, M = 2, N = 20 (which is a frequency multiplication of M * N = 40 with PLL output = 160 MHz and frequency output to C-Unit = 80 MHz).

The gear divider can be set to any even divider, in this example it is G = 20, which causes the following gear-up when switching from oscillator to PLL:

1. step : 1 cycle of 8.0 MHz (8.0 MHz equals 20 cycles of the PLL output)
2. step : 2 cycles of 8.4 MHz (8.4 MHz equals 19 cycles of the PLL output)
3. step : 3 cycles of 8.8 MHz (8.8 MHz equals 18 cycles of the PLL output)
:
17. step : 17 cycles of 40.0 MHz (40.0 MHz equals 4 cycles of the PLL output)
18. step : 18 cycles of 53.3 MHz (53.3 MHz equals 3 cycles of the PLL output)
19. step : 19 cycles of 80.0 MHz (80.0 MHz equals 2 cycles of the PLL output)
    -> Target frequency reached by transition to last step (here from 18. to 19.)

Each step can be multiplied by setting a multiplication value in the gear multiplier register. The duration from generating the start frequency up to reaching the target frequency can be calculated by the following formula:

$$\text{duration} = \text{mul} \cdot t \cdot \left(\sum_{k=1}^{i} k \cdot (i - k + 1) - \sum_{k=j+1}^{i} k \cdot (i - k + 1) \right)$$

this equals to (resolved closed arithmetic series of the first sum term):

$$\text{duration} = \text{mul} \cdot t \cdot \left(\frac{i \cdot (i+1) \cdot (i+2)}{6} - \sum_{k=j+1}^{i} k \cdot (i-k+1) \right)$$

with i = G ; j = G - M ; mul = MULG ; t = 1/f(pllout)

For the above given setting this equals 1483 PLL output clock cycles with a duration from the start frequency to the target frequency of 9262500 ps (about 9.3 us).

7. Caution

When using the clock auto-gear function it is strongly recommended to make use of the gear up and gear down flags (PLLCTRL.GRUP, PLLCTRL.GRDN) to evaluate the current state of this function to avoid malfunctions in the clock system due to setting changes prior to completion.

Procedure example:

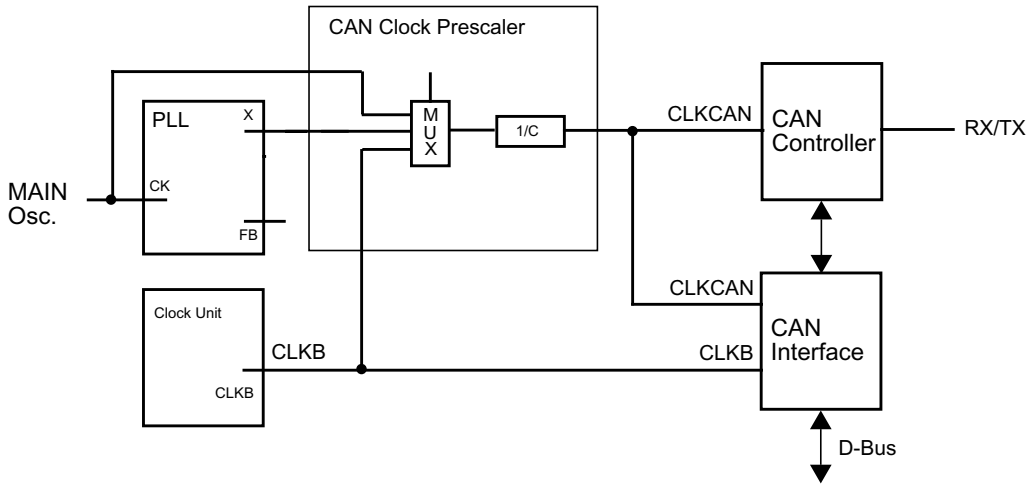
- **Set the PLL interface registers (PLLDIVN, PLLDIVM, PLLDIVG, PLLMULG) according to the selected frequency and gear duration**
- **Switch on the PLL (CLKR.PLL1EN='1')**
- **If you want to receive interrupts after gearing up or down, also enable the corresponding interrupt enables (PLLCTRL.IEUP, PLLCTRL.IEDN)**
- **Wait for the PLL stabilization time**
- **Set the base clock division registers (DIVR0, DIVR1)**
- **Switch the clock source to the PLL (CLKR.CLKS "00"-> "10")**
- **Wait for the PLLCTRL.GRUP gear up flag (either by polling or by interrupt) before switching the clock source back to oscillation or confirm the setting of PLLCTRL.GRUP='1' before changing bits in the CLKR register**
- **Switch the clock source to Oscillator (CLKR.CLKS "10"-> "00")**
- **Wait for PLLCTRL.GRDN gear down flag (either by polling or by interrupt) before switching the clock source back to PLL or confirm the setting of PLLCTRL.GRDN='1' before changing bits in the CLKR register**
- **Switch off the PLL (CLKR.PLL1EN='0')**

7.Caution

Chapter 15 CAN Clock Prescaler

1. Overview

- This blockdiagram (simplified) shows the integration of the CAN and the CAN Interface with the CAN clock prescaler logic (1/C) and clock source selector.



- **Remark: If the CLKCAN source is set either to main oscillator or to PLL output then the clock for the CAN is not influenced by the clock modulation. If the CLKCAN source is set core clock CLKB then the clock for the CAN is also modulated (if the clock modulator is enabled).**

2. Features

- CAN clock source selectable out of Main Oscillation, Base Clock (CLKB) and PLL output
- Free programmable divide-by-C counter in the range of 1..16
- Individual clock disable function for each CAN controller

3. Registers

3.1 CAN Clock Control Register

Controls the CAN clock source, the clock division ratio and the clock disable.

- **CANPRE: Address 04C0h (Access: Byte)**

7	6	5	4	3	2	1	0	bit
-	-	CPCKS1	CPCKS0	DVC3	DVC2	DVC1	DVC0	
0	0	0	0	0	0	0	0	Initial value (INIT pin input, watchdog reset)
0	0	X	X	X	X	X	X	Initial value (Software reset)
R0/W0	R0/W0	R/W	R/W	R/W	R/W	R/W	R/W	Attribute

(See “Meaning of Bit Attribute Symbols (Page No.10)” for details of the attributes.)

- Bit7-6: Reserved bitAlways write “0” to these register bits.
- Bit5-4: CAN Prescaler ClocK Selection

CPCKS1-CPCKS0	Prescaler clock source
00	CLKB, core clock (inital)
01	PLL output
10	reserved
11	Main oscillation

- Bit3-0: Source clock Divide-by-C selection

DVC3-DVC0	Source clock divided-by-C (generates CLKCAN)
0000	Source clock : 1 (no division)
0001	Source clock : 2 (division by 2)
0010	Source clock : 3 (division by 3)
0011	Source clock : 4 (division by 4)
0100	Source clock : 5 (division by 5)
0101	Source clock : 6 (division by 6)
0110	Source clock : 7 (division by 7)
0111	Source clock : 8 (division by 8)
.....	.....
1111	Source clock : 16 (division by 16)

(Note) Do not exceed the specified upper frequency limit of CLKCAN (e.g. 20 MHz) e.g. by setting prescaler values exceeding this limit, or by switching the prescaler clock source to a higher frequency clock without switching previously the prescaler values to higher division rates.

(Note) If prescaler source is selected to PLL output: Even though it is possible to select no division ratio (:1) for the divide-by-C counter it is not recommended. The resulting output clock will have an odd clock duty ratio (direct PLL output can have up to 90:10 duty). Always select at least a division ratio > 1.

- **CANCKD: Address 04C1h (Access: Byte)**

7	6	5	4	3	2	1	0	bit
-	-	CANCKD5	CANCKD4	CANCKD3	CANCKD2	CANCKD1	CANCKD0	
0	0	0	0	0	0	0	0	Initial value (INIT pin input, watchdog reset)

0	0	0	0	X	X	X	X	Initial value (Software reset)
R/W0	R/W0	R/W	R/W	R/W	R/W	R/W	R/W	Attribute

(See “Meaning of Bit Attribute Symbols (Page No.10)” for details of the attributes.)

- Bit7-6: Reserved bitAlways write “0” to these register bits.
- Bit5-0: CAN clock disable

CANCKD5-CANCKD0	Function
-----0	CAN controller 0 is enabled
-----1	CAN controller 0 is disabled
----0-	CAN controller 1 is enabled
----1-	CAN controller 1 is disabled
---0--	CAN controller 2 is enabled
---1--	CAN controller 2 is disabled
--0---	CAN controller 3 is enabled
--1---	CAN controller 3 is disabled
-0----	CAN controller 4 is enabled
-1----	CAN controller 4 is disabled
0-----	CAN controller 5 is enabled
1-----	CAN controller 5 is disabled

3.Registers

Chapter 16 Clock Supervisor

This section gives an overview of the Clock Supervisor. Purpose of the Clock Supervisor is the supervision of the main and sub oscillation clock. In case of main oscillation clock failure the Clock Supervisor control logic will take action, i.e. switching to an internal RC-oscillation clock, depending on the operation mode set in the control register.

1. Overview Clock Supervisor

The Clock Supervisors purpose is the supervision of the main and sub oscillation clocks. In case of a clock failure (main clock and/or sub-clock) it can be replaced by an on-chip RC-oscillation clock, depending on the configuration.

If a clock the MCU currently uses, fails for a certain time (20-80µs for main clock / 160-640µs for sub-clock) the MCU is reset and the reset cause can be checked after reset vector fetch.

If the sub-clock is failing while the MCU is in main clock mode reset can be delayed until the transition to sub-clock mode or no reset will be initiated. The user can choose the behaviour with a control bit in the Clock Supervisor Control Register.

There are two independent supervisors one for the main clock and one for the sub-clock. They can be enabled/ disabled seperately.

Main clock and sub-clock supervisor are disabled and re-enabled automatically if the corresponding oscillator is disabled and re-enabled.

If the MCU changes to stop mode, the RC-oscillator is automatically disabled. It will be enabled again upon wake-up from stop mode.

There are two status bits in the Clock Supervisor Control Register which indicate the failure of the main clock and sub-clock. These bits can be available at two port pins (device dependent).

Single clock devices can use the RC-oscillation clock as sub-clock.

2. Clock Supervisor Register

This section lists the Clock Supervisor Control Register and describes the function of each bit in detail.

■ Clock Supervisor Control Register (CSVCR)

The Clock Supervisor Control Register (CSVCR) sets the operation mode of the Clock Supervisor. Figure 2-1 shows the configuration of the Clock Supervisor Control Register.

Figure 2-1 Configuration Clock Supervisor Control Register (CSVCR)

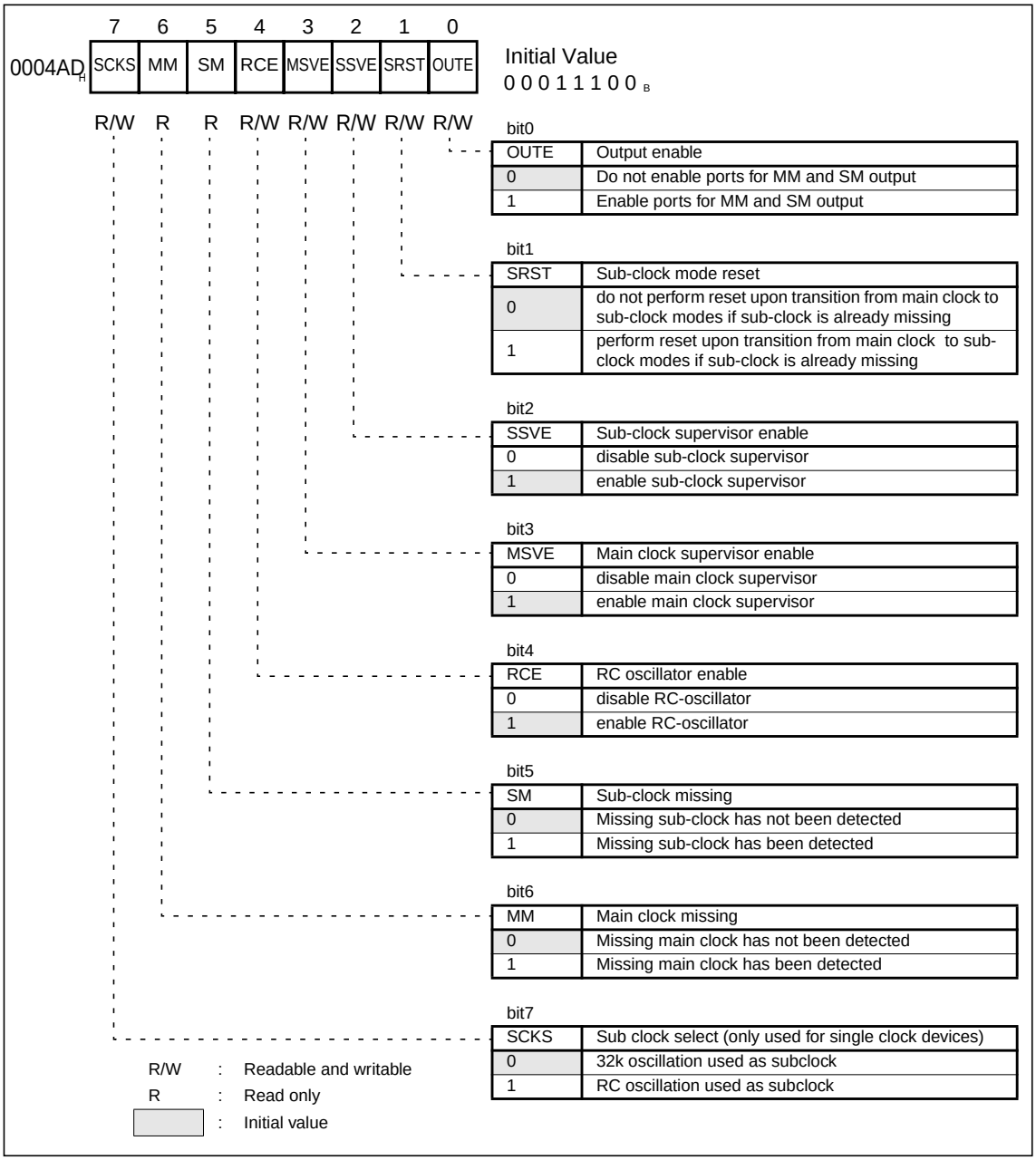


Table 2-1 describes the function of each bit of the Clock Supervisor Control Register (CSVCR).

Table 2-1 Functional Description of each bit of the Clock Supervisor Control Register

Bit	Name	Function
7	SCKS (Sub-clock select)	This bit is to select between 32k external oscillation and internal RC oscillation as subclock. If this bit is 0 then the external 32k oscillation is used as subclock, if it's 1 then the internal RC oscillation is used as subclock.This bit is cleared to '0' by Power-On reset or external reset. Other types of reset will not affect this bit.
6	MM (Main clock missing)	If this bit is 1, the main clock supervisor has detected that the main oscillation clock coming from X0, X1 is missing, e.g. by a broken crystal. If this bit is 0, a missing main clock has not been detected. Writing this bit has no effect. This bit is cleared to '0' by Power-On reset or external reset. Other types of reset will not affect this bit.
5	SM (Sub-clock missing)	If this bit is 1, the sub-clock supervisor has detected that the sub oscillation clock coming from X0A, X1A is missing, e.g. by a broken crystal. If this bit is 0, a missing sub-clock has not been detected. Writing this bit has no effect. This bit is cleared to '0' by Power-On reset or external reset. Other types of reset will not affect this bit.
4	RCE (RC-oscillator enable)	Setting this bit to 1 enables the RC-oscillator. Do not disable the RC-oscillator by setting this bit to 0 while the main or sub-clock supervisors are still enabled. First the supervisors must be disabled, then it must be checked that MM and SM are '0', then RCE can be set to '0'. If either MM or SM bits are '1', RCE must not be set to '0'. This bit is set to '1' by Power-On reset or external reset. Other types of reset will not affect this bit.
3	MSVE (Main clock supervisor enable)	Setting this bit to 1 enables the main clock supervisor. When this bit is set to 1, the RC-oscillator must have been enabled by the RCE bit for at least 100µs. This bit is set to '1' by Power-On reset only. Other types of reset will not affect this bit.
2	SSVE (Sub-clock supervisor enable)	Setting this bit to 1 enables the sub-clock supervisor. When this bit is set to 1, the RC-oscillator must have been enabled by the RCE bit for at least 100µs. This bit is set to '1' by Power-On reset only. Other types of reset will not affect this bit.
1	SRST (Sub-clock mode reset)	If this bit is set to 1, a reset is performed upon transition from main/PLL clock mode to sub-clock mode if the sub-clock is already missing. If this bit is set to 0, no reset is performed in this case. This bit is cleared to '0' by Power-On reset or external reset. Other types of reset will not affect this bit.
0	OUTE (Output enable)	This bit can be used as an output enable to output the signals MM (bit 3 of CSVCR) and SM (bit 4 of CSVCR) to port pins. If this bit is set to '1', the ports are enabled for MM and SM output. This bit is cleared to '0' by Power-On reset or external reset. Other types of reset will not affect this bit.

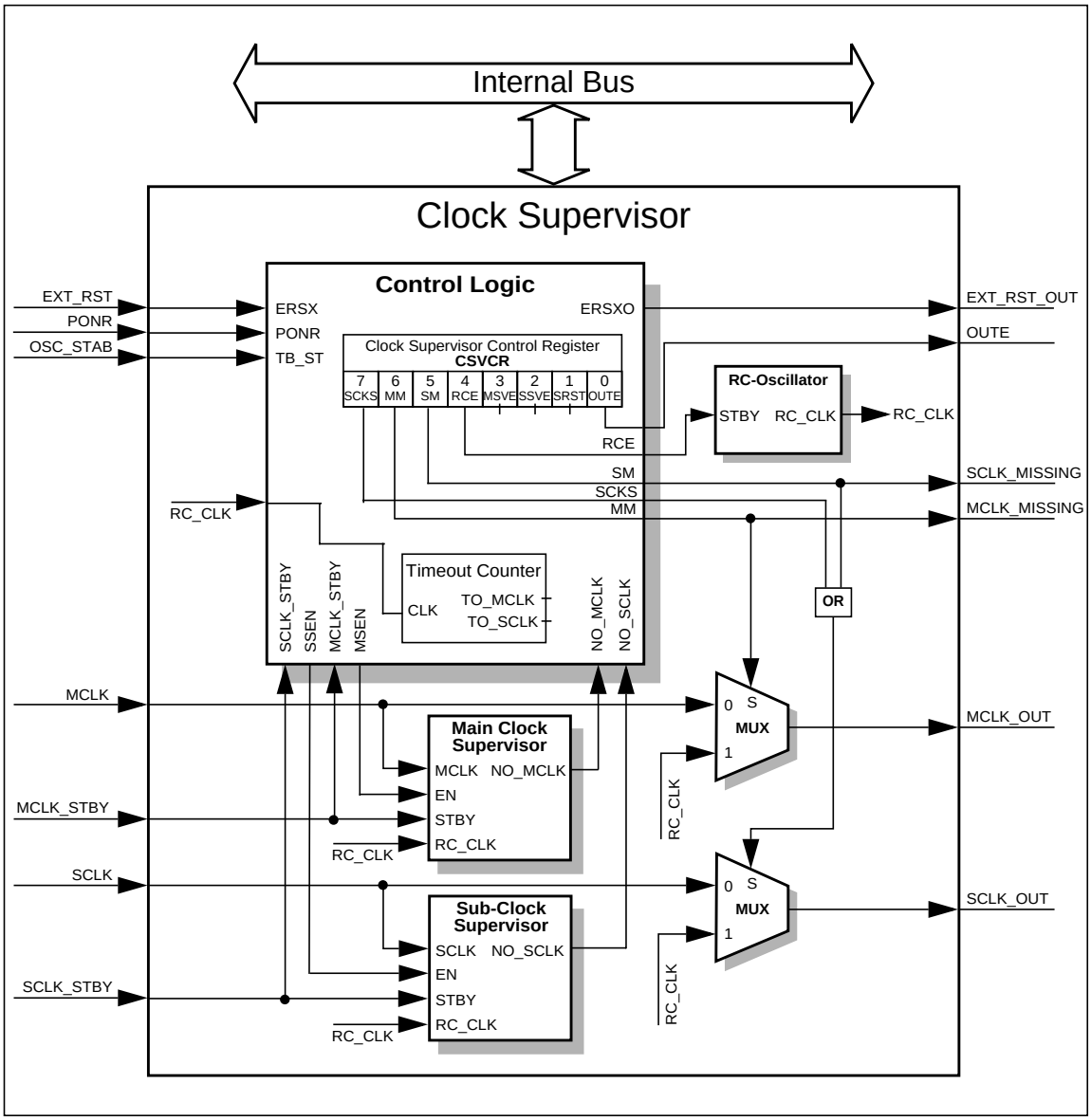
3. Block Diagram Clock Supervisor

This section presents a block diagram of the Clock Supervisor. The building blocks of the Clock Supervisor are:

- Main Clock Supervisor
- Sub-Clock Supervisor
- Control Logic
- RC-Oscillator

■ Block Diagram Clock Supervisor

Figure 3-1 Bock Diagram Clock Supervisor



Remark: SCLK_OUT and MCLK_OUT can be observed using the Clock Monitor Module. SCLK_MISSING and MCLK_MISSING can be programmed to device specific outputs (see the datasheet of the used device for the information which pins are used) by setting OUTE=1.

4. Operation Modes

This section describes all operation modes of the Clock Supervisor.

■ Operation mode with initial settings

In case the clock supervisor control register (CSVCR) is not configured at the beginning of the user program, the RC-oscillator, the main clock supervisor and the sub-clock supervisor is enabled.

- The RC-oscillator is enabled at power-on.
- The main clock supervisor is enabled after the ’oscillation stabilisation wait time’ with the rising edge of signal OSC_STAB or in case the main clock is missing before the completion of the ’oscillation stabilisation wait time’, after the ’main clock timeout’ (TO_MCLK) from the timeout counter. The timeout counter is clocked with RC-oscillation clock. If the main clock is missing from power-on, the power-on reset state is never left, which in this case is a safe state. The user must make sure with external pull-up/pull-down resistors that all relevant signal are pulled to the correct level.
- The sub-clock supervisor is enabled after the completion of the ’sub-clock timeout’ (TO_SCLK) from the timeout counter. The timeout counter is clocked with RC-oscillation clock.
- If the main clock stops while the main clock supervisor is enabled, the main clock is replaced with the RC-oscillation clock, the MM bit is set to ’1’ and reset (EXT_RST_OUT) is asserted.
- If the sub-clock stops and the sub-clock supervisor is enabled, the behaviour depend on whether the MCU is in main clock mode or in sub-clock mode. If the sub-clock stops in sub-clock mode, the RC-oscillation clock divided by two substitutes the sub-clock, the SM bit is set to ’1’ and reset (EXT_RST_OUT) is asserted. If the sub-clock stops in main clock mode, the RC-oscillation clock divided by two substitutes the sub-clock, the SM bit is set to ’1’ and no reset occurs upon transition to sub-clock mode, since the SRST bit has its initial value of ’0’.

Figure 4-1 Timing Diagram: Initial settings, main clock missing during power-on reset

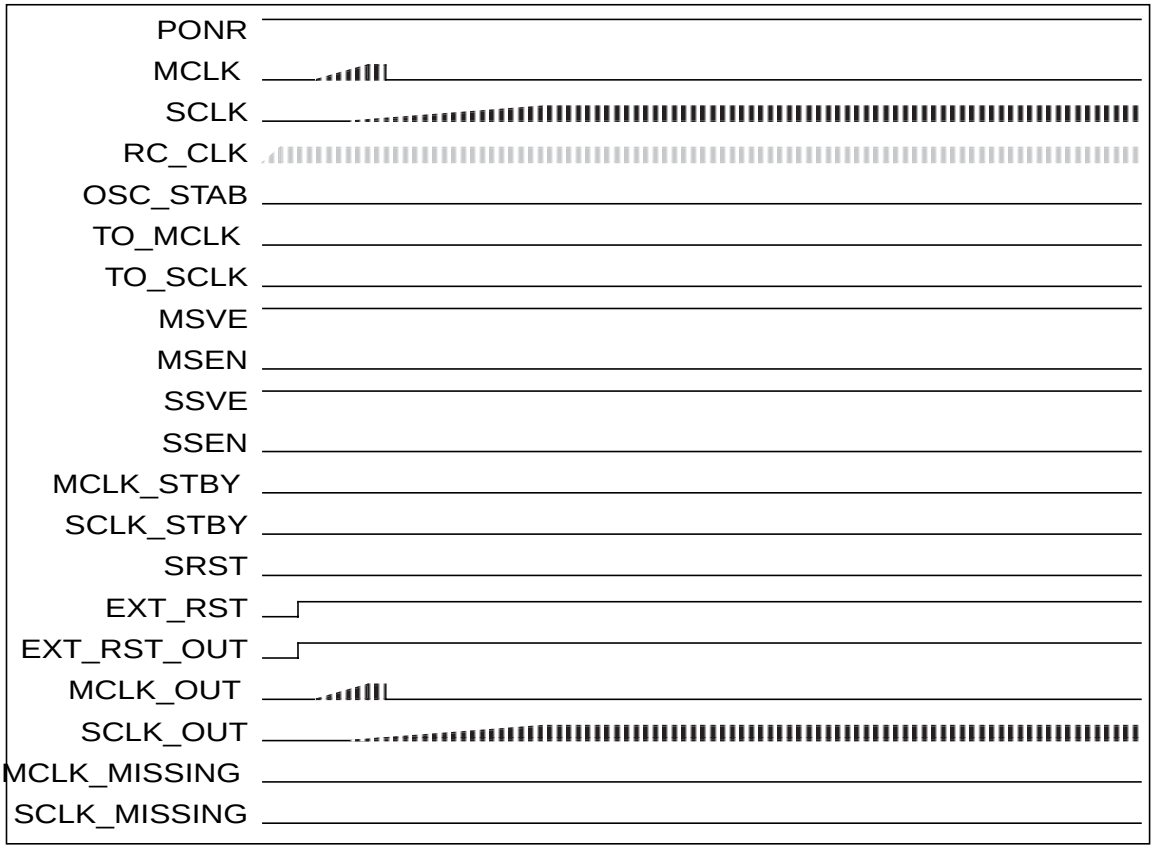


Figure 4-2 Timing Diagram: Initial settings, main clock missing during 'oscillation stabilisation wait time'

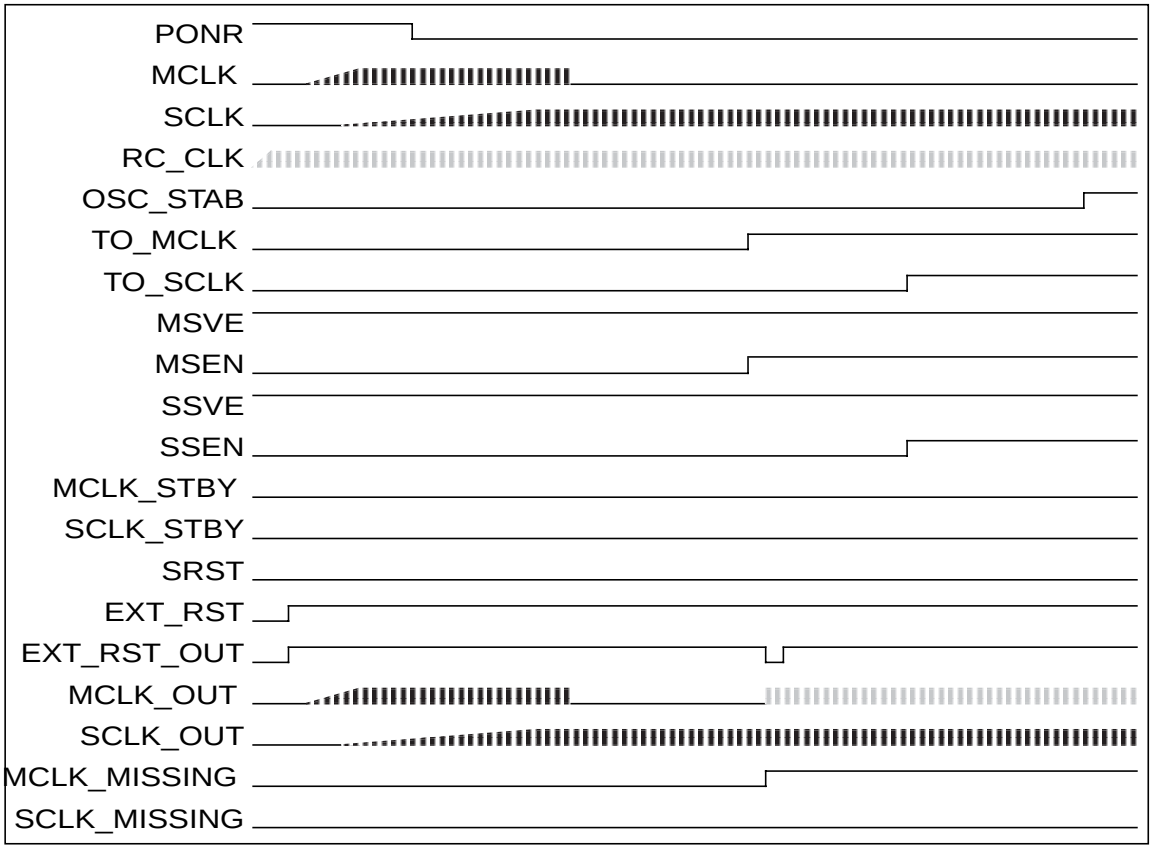


Figure 4-3 Timing Diagram: Initial settings, main clock missing after 'oscillation stabilisation wait time'

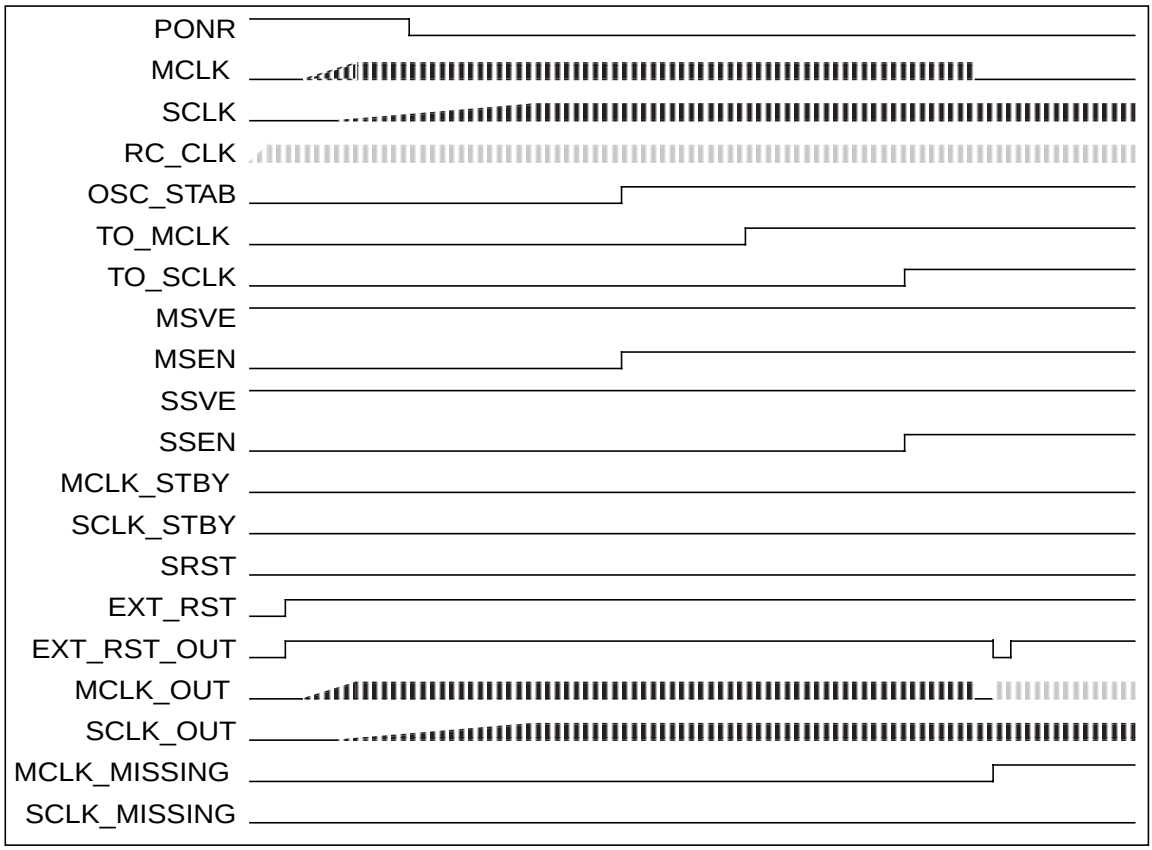


Figure 4-4 Timing Diagram: Initial settings, sub-clock missing before timeout

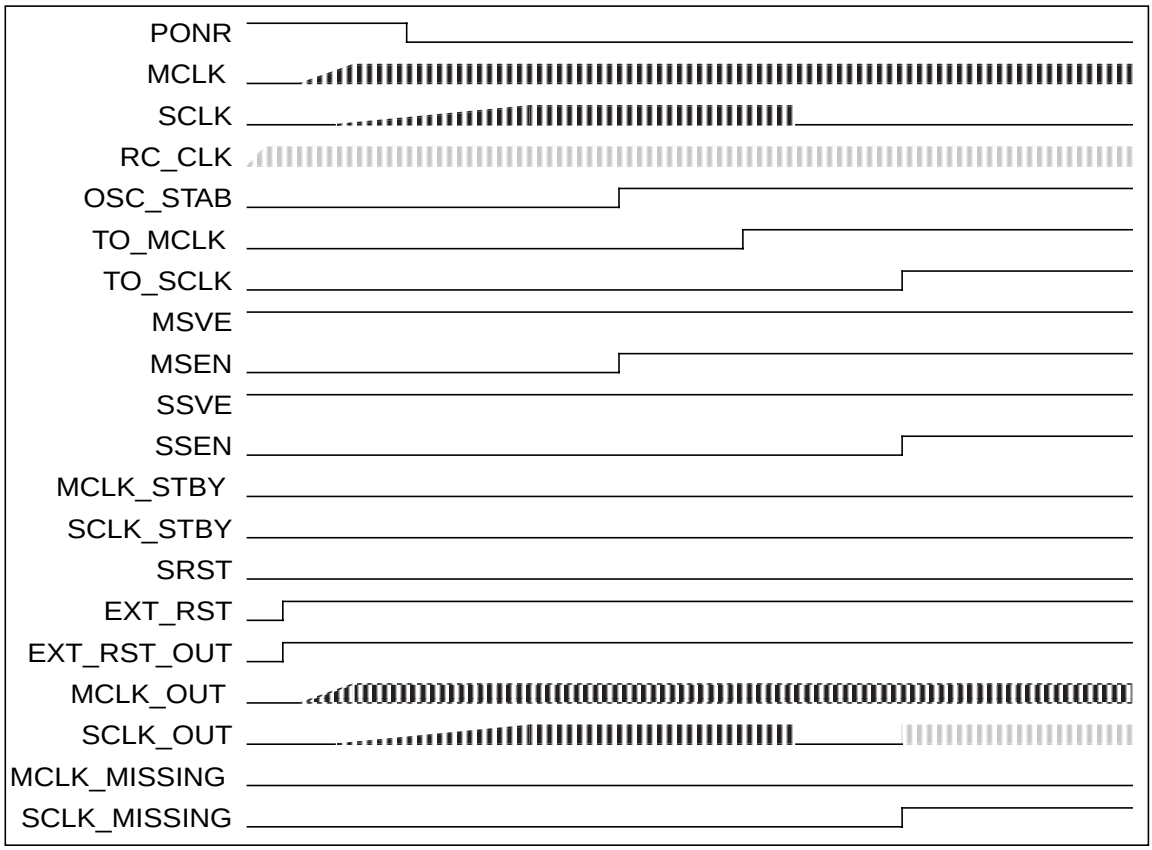
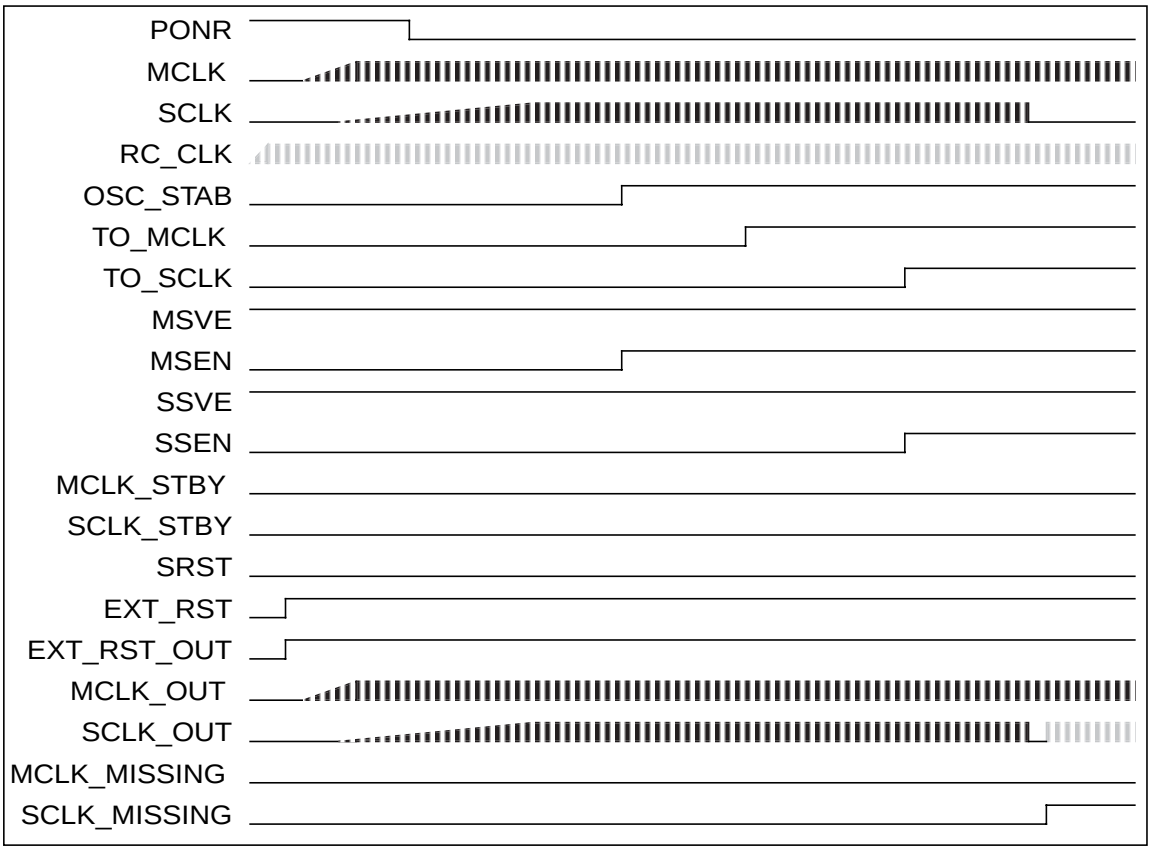


Figure 4-5 Timing Diagram: Initial settings, sub-clock missing after timeout

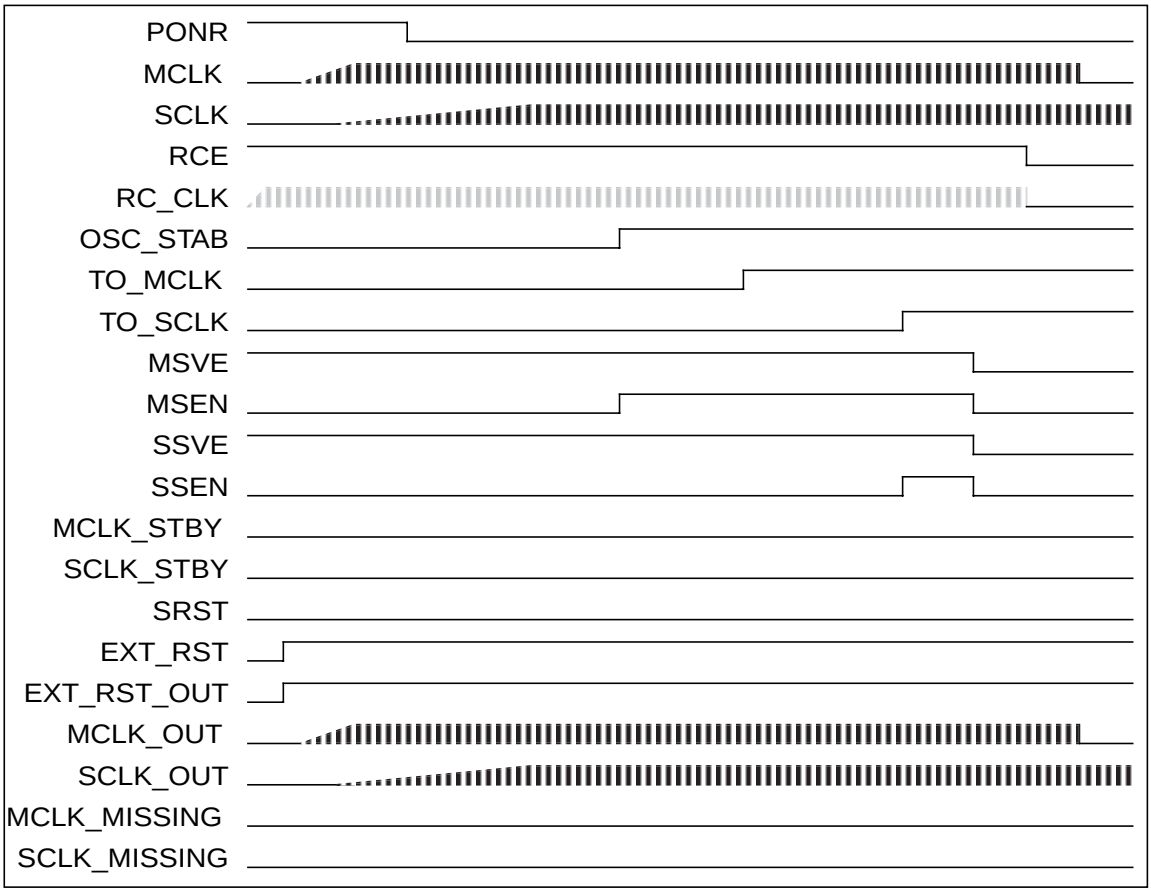


■ Disabling the RC-oscillator and the clock supervisors

The initial point of this scenario is that the RC-oscillator and main clock or sub-clock supervisor is enabled.

- The RC-oscillator can be disabled by setting bit RCE (bit 4 of CSVCR) to ’0’. First disable the main clock and sub-clock supervisor. Do not disable the RC-oscillator while either the main clock or sub-clock supervisor is still enabled. Then check that both SM and MM (bit 5 and bit 6 of CSVCR) are still ’0’. Disable the RC-oscillator by setting RCE to ’0’. If either SM or MM bit is ’1’, RCE must not be set to ’0’.
- The main clock supervisor is disabled by setting MSVE (bit 3 of CSVCR) to ’0’.
- The sub-clock supervisor is disabled by setting SSVE (bit 2 of CVSVR) to ’0’.

Figure 4-6 Timing Diagram: Disabling the RC-oscillator and the clock supervisors

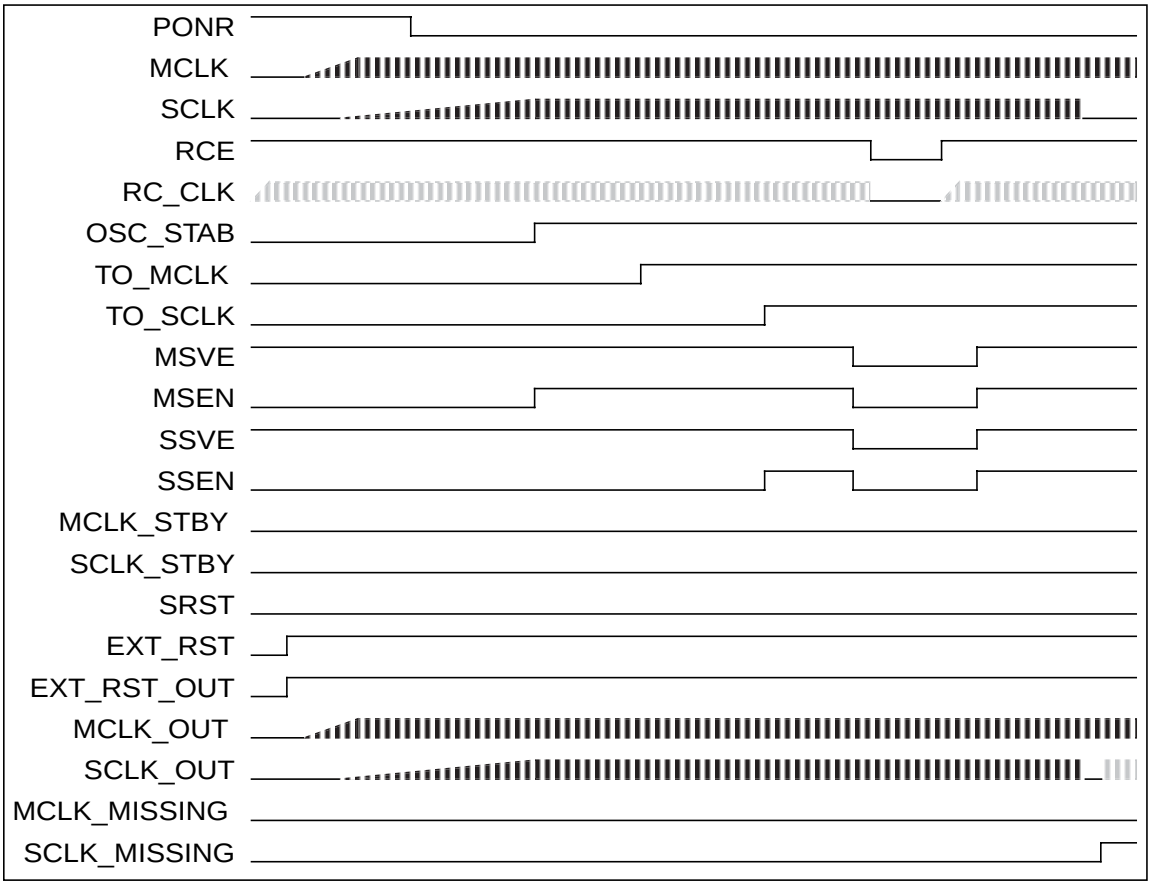


■ Re-enabling the RC-oscillator and the clock supervisors

The initial point of this scenario is that the RC-oscillator and both main clock and sub-clock supervisor are disabled.

- The RC-oscillator can be enabled by setting RCE (bit 4 of CSVCR) to ’1’.
- The main clock supervisor is enabled by setting MSVE (bit 3 of CSVCR) to ’1’. Enabling of the main clock supervisor must only take place 100 µs after the RC-oscillator is enabled. The software has to take care that this time constraint is met.
- The sub- clock supervisor is enabled by setting SSVE (bit 2 of CSVCR) to ’1’. Enabling of the sub-clock supervisor must only take place 100 µs after the RC-oscillator is enabled. The software has to take care that this time constraint is met.

Figure 4-7 Timing Diagram: Re-enabling the RC-oscillator and the clock supervisors



■ Sub-clock modes

The main clock supervisor is automatically disabled in sub-clock modes. The enable bit MSVE remains unchanged. At transition from sub-clock mode to main clock mode the main clock supervisor is enabled after the ’oscillation stabilisation wait time’ with the rising edge of signal OSC_STAB or in case the main clock is missing before the completion of the ’oscillation stabilisation wait time’, after the ’main clock timeout’ (TO_MCLK) from the timeout counter. The timeout counter is clocked with RC-oscillation clock.

■ Changing the behaviour upon transition to sub-clock mode if the sub-clock has already stopped in main clock mode

If the sub-clock has stopped in main clock mode and this was detected by the sub-clock supervisor, the behaviour upon transition to sub-clock mode depends on the state of the SRST bit.

- If SRST is set to ’0’ (initial value), reset is not asserted at the transition to sub-clock mode. The transition is performed using the RC-oscillation clock as sub-clock. In this case it is recommended to check the SM bit before the transition to sub-clock mode to get the information if sub-clock or RC-oscillation clock is used.
- If SRST is set to ’1’, reset is asserted at the transition to sub-clock mode.

The following timing diagrams (Figure 4-8, Figure 4-9, Figure 4-10) illustrate this behaviour.

Figure 4-8 Timing Diagram: Sub-clock missing in main clock mode, SRST=0

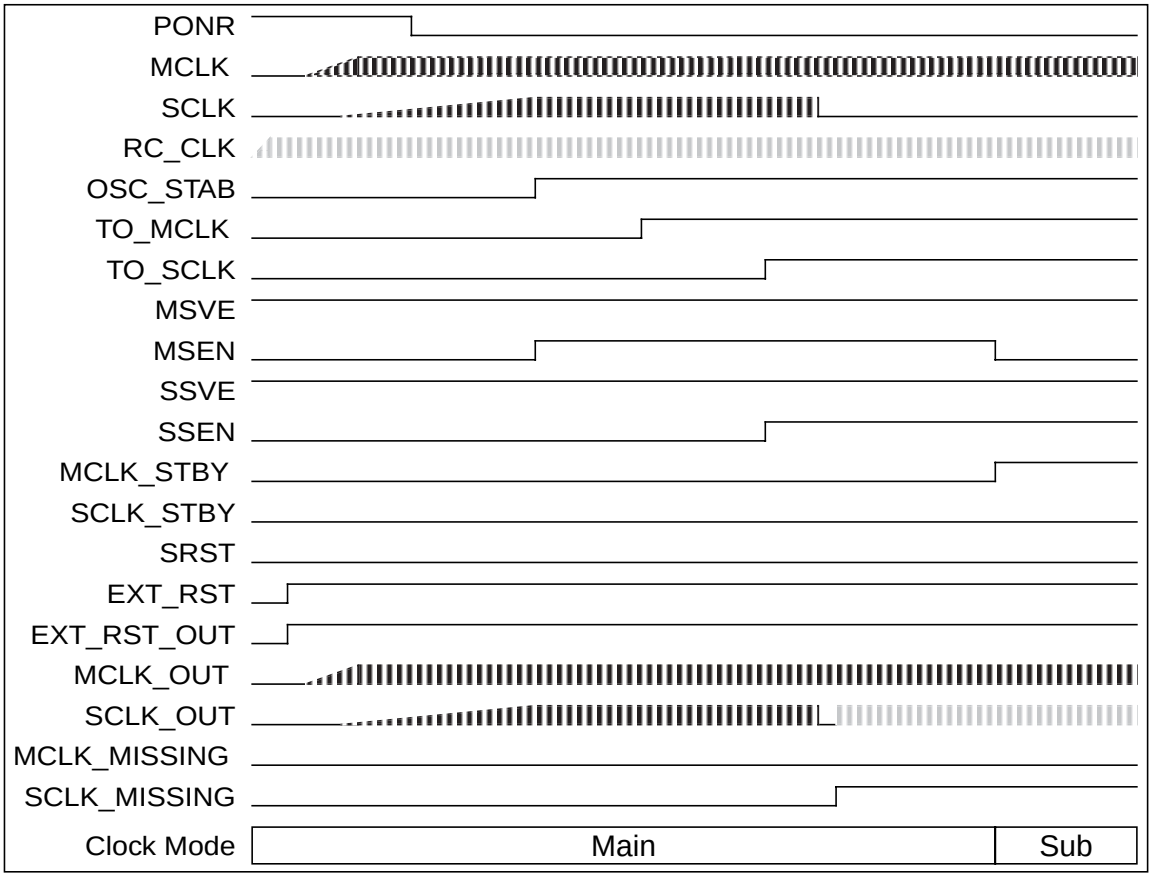


Figure 4-9 Timing Diagram: Sub-clock missing in main clock mode, SRST=1

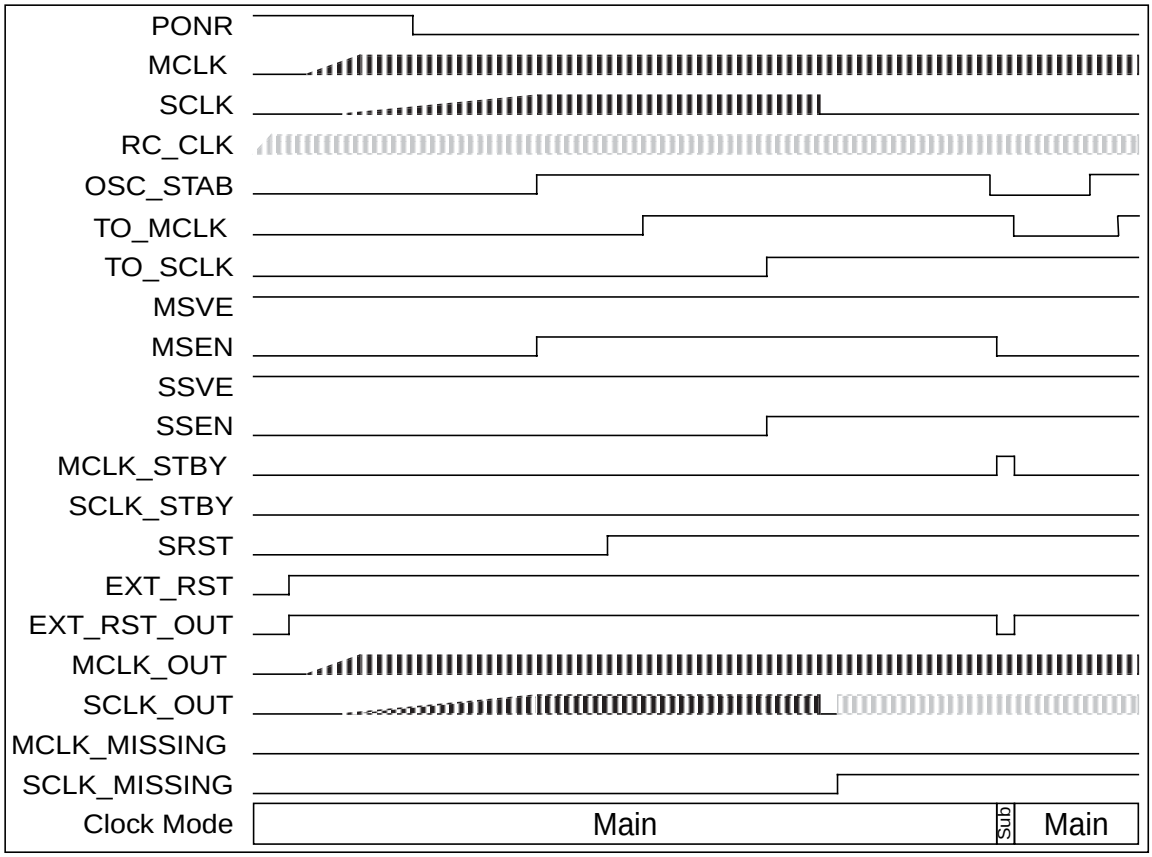
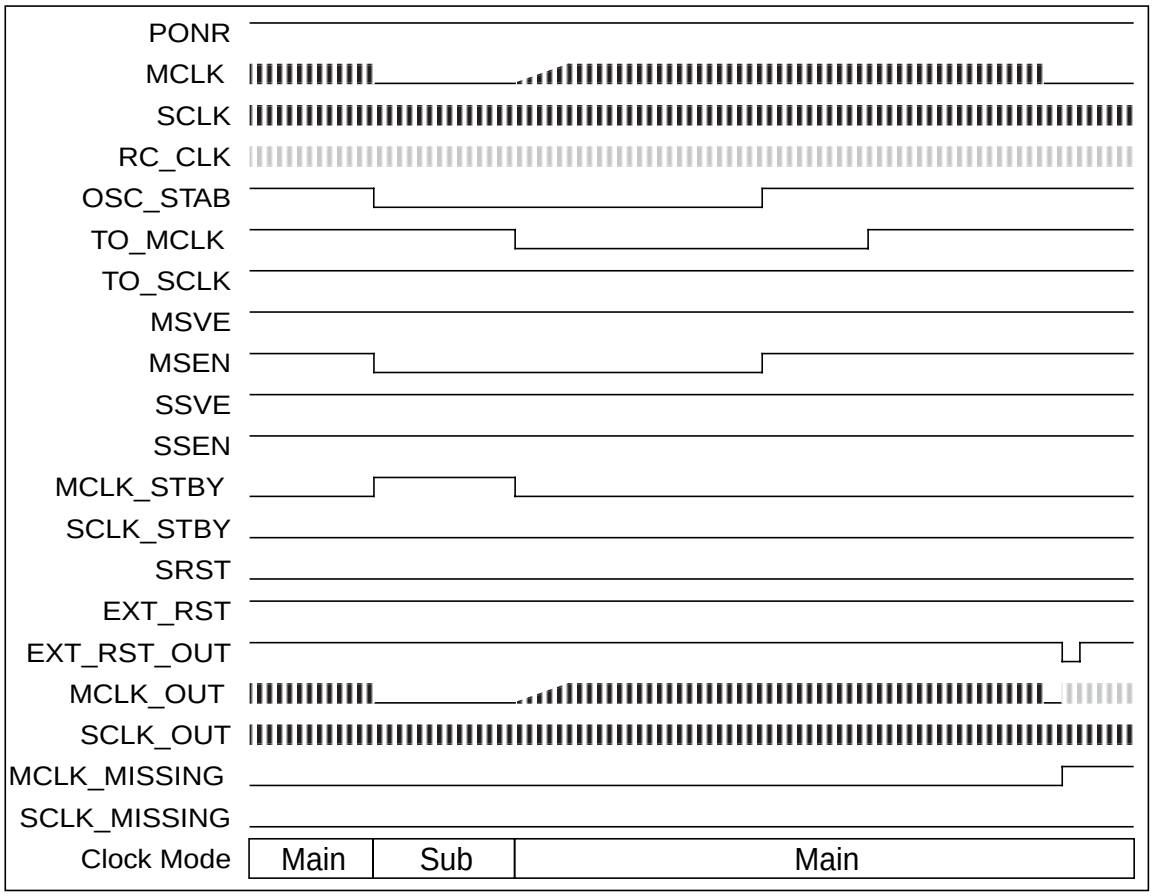


Figure 4-10 Timing Diagram: Waking up from sub-clock mode

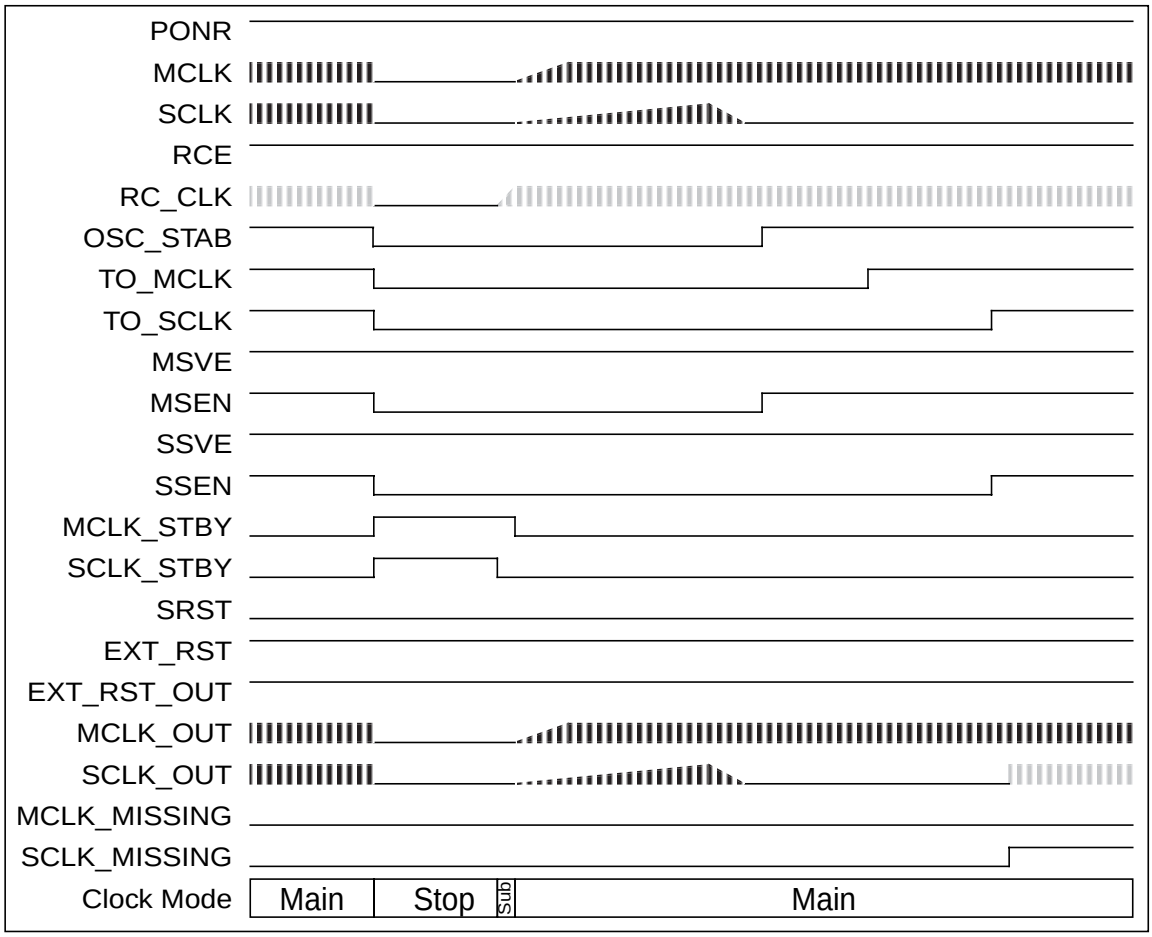


■ Stop mode

If RC-oscillator, main clock and sub-clock supervisors are enabled, they will be automatically disabled at transition into stop mode. The corresponding enable bits in the clock supervisor control register remain unchanged. So after wake-up from stop mode the RC-oscillator and the clock supervisors will be enabled again. If the corresponding enable bits are set to ’0’, the RC-oscillator and the clock supervisors will stay disabled after wake-up from stop mode.

- The RC-oscillator is enabled immediately after wake-up from stop mode.
- The main clock supervisor is enabled after the ’oscillation stabilisation wait time’ with the rising edge of the signal OSC_STAB or in case the main clock is missing after wake-up from stop mode, after the ’main clock timeout’ (TO_MCLK) from the timeout counter. The timeout counter is clocked with RC-oscillation clock.
- The sub-clock supervisor is enabled after the ’sub-clock timeout’ (TO_SCLK) from the timeout counter which is clocked with the RC-oscillation clock.

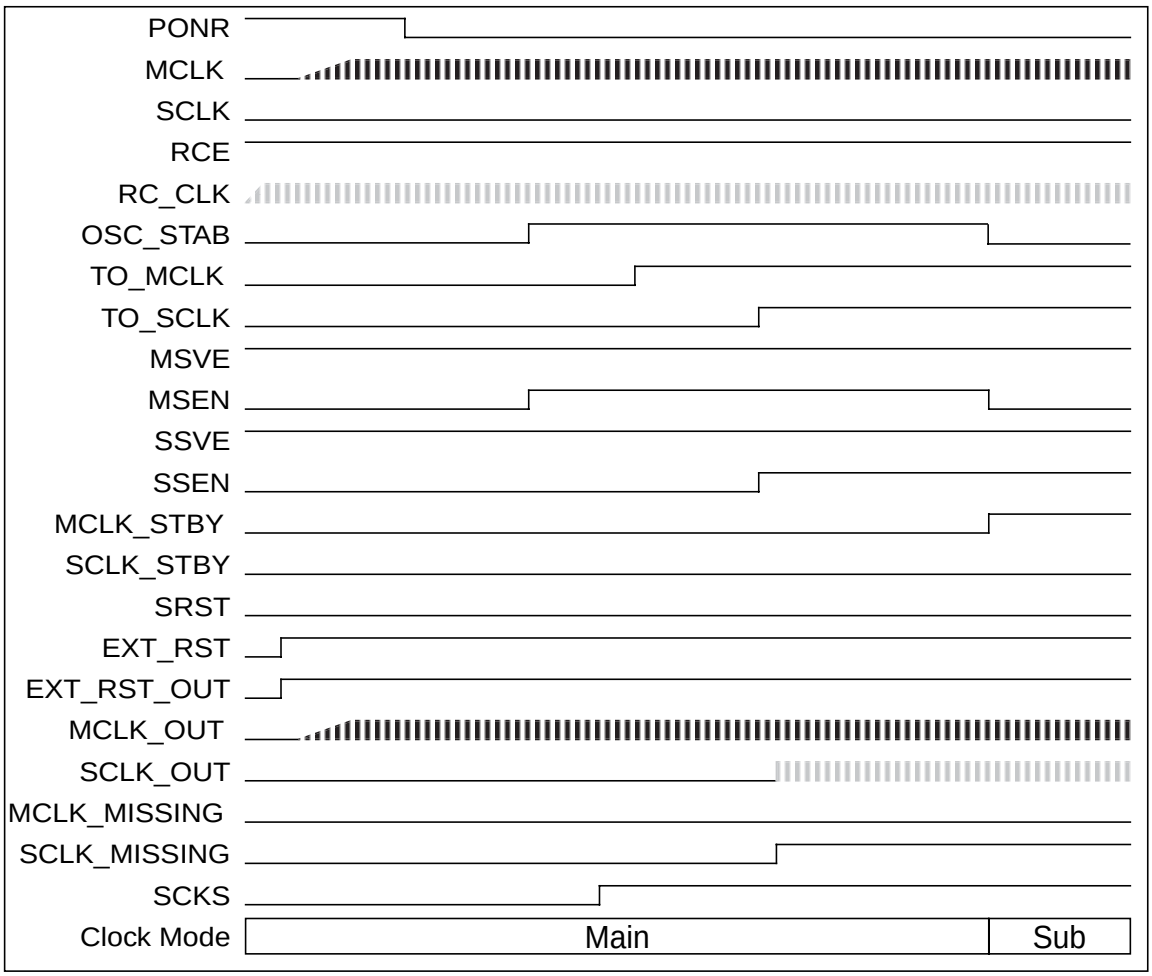
Figure 4-11 Timing Diagram: Waking up from stop mode



■ Operation with single clock device

In a single clock device the sub-clock supervisor can provide the RC-oscillation clock as sub-clock. To enable this feature, SCKS bit (bit7 of CSVCR) must be set to ’1’ (refer to Table 2-1for precautions when modifying this bit) and SRST must be ’0’ (initial value). Before the transition to sub-clock mode, it has to be confirmed by software that the sub-clock has been substituted by the RC-oscillation clock. This can be accomplished by checking that SM bit (bit 5 of CSVCR) is ’1’ and RCE bit (bit 4 of CSVCR) is ’1’.

Figure 4-12 Timing Diagram: Sub-clock mode with single clock device



■ Check if reset was asserted by the Clock Supervisor

To find out whether the Clock Supervisor has asserted reset , the software must check the reset cause by reading the WDTC register at address $A8_H$. If ERST (bit 4 of WDTC) is set, the cause was either external reset at the RSTX pin or the clock supervisor. If neither SM bit nor MM bit (bit 5 and bit 6 of CSVCR) is set, reset cause was the external reset. If SM is ’1’ the reset cause is a missing sub-clock and if MM is ’1’ the reset cause is a missing main clock.

Chapter 17 Clock Modulator

This chapter provides an overview of the Clock Modulator and its features. It describes the register structure and operation of the Clock Modulator.

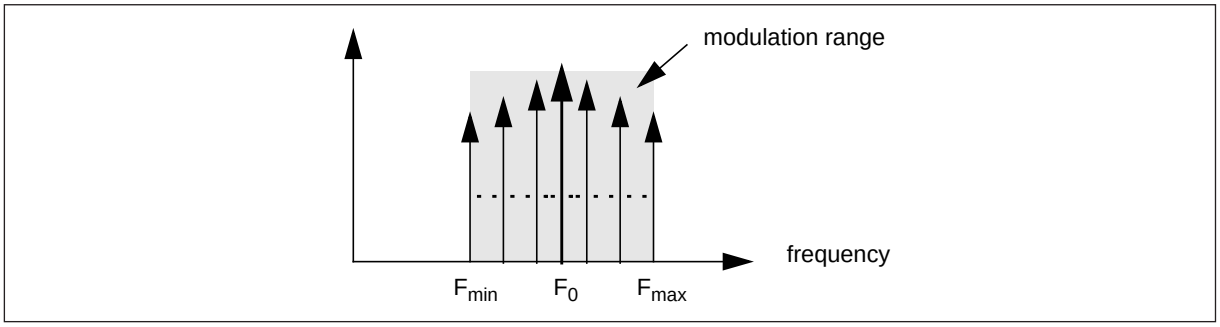
1. Overview

The clock modulator is intended for the reduction of electromagnetic interference - EMI, by spreading the spectrum of the clock signal over a wide range of frequencies.

The module is fed with an unmodulated reference clock with frequency F0, provided by the PLL circuit. This reference clock is frequency modulated, controlled by a random signal.

The mean frequency of the modulated clock is equal to the reference clock frequency F0.

Figure 1-1 Frequency spectrum of the modulated clock (fundamentals only)



- Modulation degree and frequency resolution in frequency modulation mode

Maximum and minimum frequencies (F_{max} and F_{min}) of the modulated clock are defined by the modulation degree parameter. Furthermore the resolution of the modulation range is selectable in 7 steps from low (1) to high (7). Higher resolution implies a finer granularity of discrete frequencies in the spectrum of the modulated clock but less possible modulation degrees.

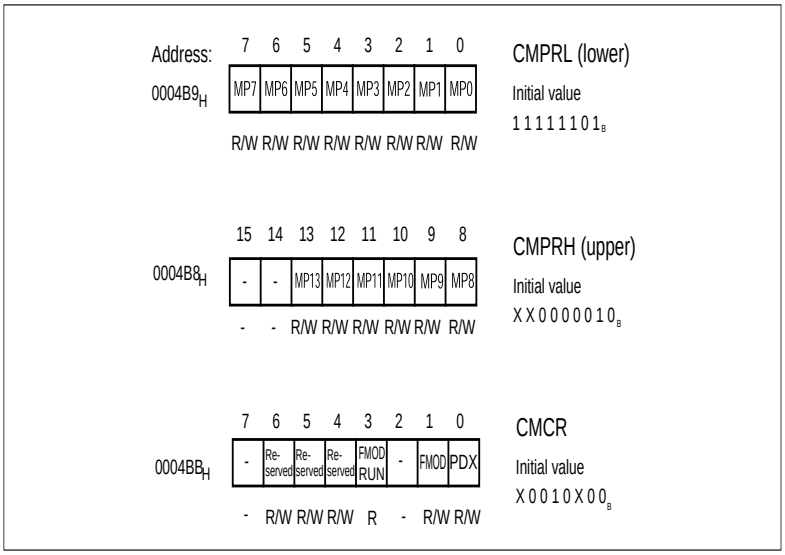
In general the highest possible frequency resolution combined with the highest possible modulation degree results in the highest EMI reduction. But for some cases lower modulation degrees may result in a better EMI behavior. Please refer to the table of possible settings in Table 2-4 Modulation Parameter settings.

2. Clock Modulator Registers

This section lists the clock modulator registers and describes the function of each register in detail.

- Clock modulator registers

Figure 2-1 Clock modulator registers



● Clock Modulator Control Register (CMCR)

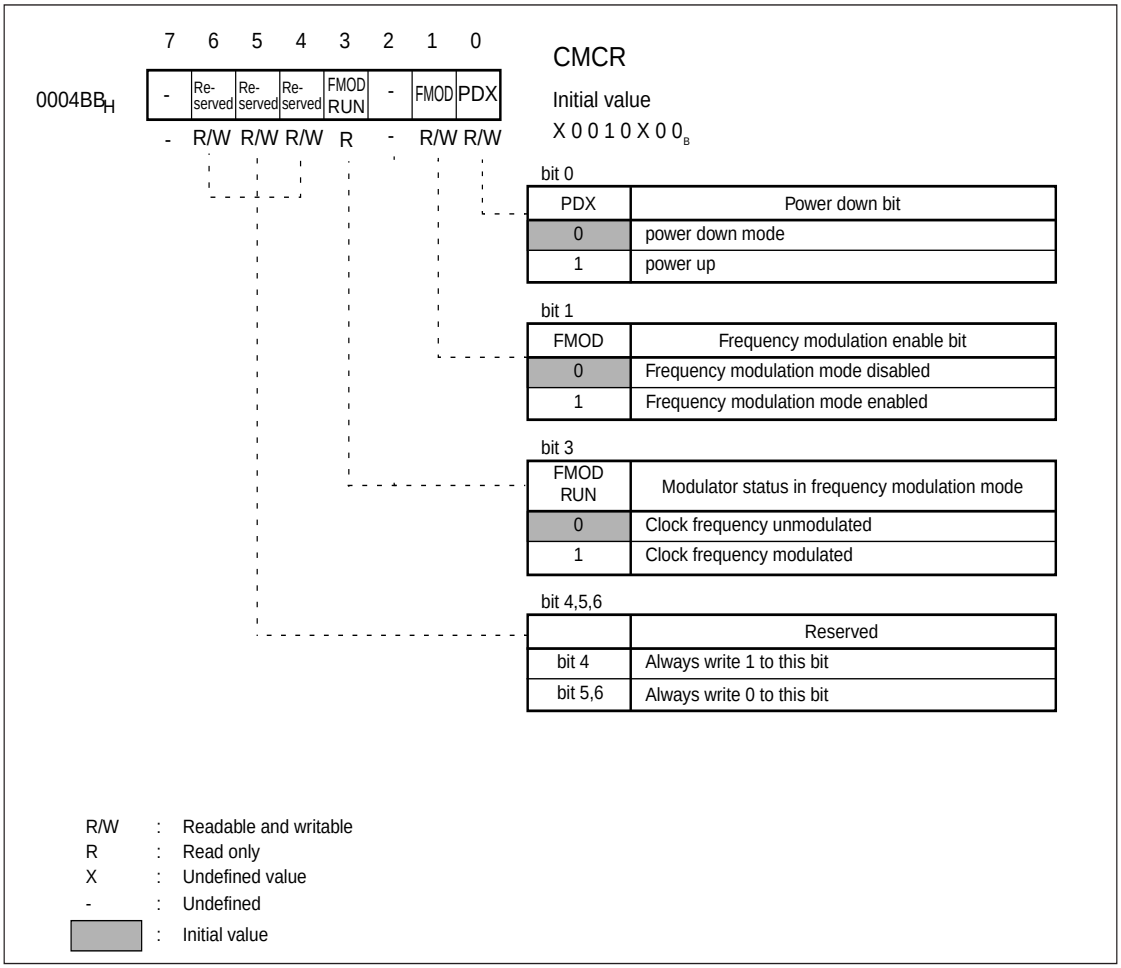
The Control Register (CMCR) has the following functions:

Set the modulator to power down mode

Modulator enable/disable in frequency modulation mode

Indicates the status of the modulator

Figure 2-2 Configuration of the clock modulator control register (CMCR)



The bits FMODRUN, FMOD, PDX control or indicate the status of the frequency modulation mode. Frequency modulation mode needs some additional configuration (CMPR register).

- Clock modulator control register contents

Table 2-1 Function of each bit of the clock modulator control register (1 / 2)

Bit name		Function
bit7	undefined	
bit 6 to 5	Reserved	Always write 0 to this bit.
bit 4	Reserved	Always write 1 to this bit.
bit 3	FMOD RUN: Modulator status in frequency modulation mode bit	"0": MCU is running with unmodulated clock "1": MCU is running with frequency modulated clock • FMODRUN indicates the status of the modulator output clock in frequency modulation mode (FMOD=1). If the output clock is frequency modulated, MODRUN is set to 1, otherwise MODRUN is set to 0. • After enabling the frequency modulation mode by setting FMOD to 1, the modulator is calibrated. During this time, the clock is unmodulated. Therefore it takes several us before the output clock switches to modulated clock and the FMODRUN bit is set to 1. The calibration time depends on the frequency of the oscillator. At oscillator (Fc) = 4MHz : Calibration time = 64.00us (calibration time = 256/Fc). • During normal operation, after calibration is finished, the clock is not switched to unmodulated clock anymore. • Due to the synchronization of the FMOD signal and the synchronized switching to unmodulated clock, it takes less than 9 x T0 (input clock period) before FMODRUN changes to 0 and the clock switches to unmodulated clock after the modulator is disabled. • The FMODRUN bit is read only. Writing to FMODRUN has no effect. • Before changing the parameter register CMPR, the modulator must be disabled -> FMOD=0 and FMODRUN=0.
bit 2	Undefined	

Table 2-1 Function of each bit of the clock modulator control register (2 / 2)

Bit name		Function
bit 1	FMOD: Frequency modulation enable bit	"0": Frequency modulation disabled. "1": Frequency modulation enabled. • To enable the modulator in frequency modulation mode, FMOD must be set to 1. • Before the modulator can be enabled, the PLL must deliver a stable reference clock (PLL lock time must be elapsed). • The specified PLL frequency range for frequency modulation mode is 15 MHz to 25 MHz. • Each PLL output frequency offers a set of possible modulation parameters. The selected setting (CMPR register) and the PLL frequency must match. Please refer to the CMPR register description. • Whenever the PLL output frequency is changed or the PLL is switched off e.g. in power down modes, the modulator must be disabled before -> FMOD=0 and FMODRUN=0. • Before the modulator can be enabled, it must be switched from power down to active mode by setting PDX to 1. And the startup time of 6us must be awaited. Please refer to the application note for a description of the recommended startup sequence. • Before the modulator can be enabled in frequency modulation mode, a proper setting must be selected via the parameter register CMPR. • After enabling the frequency modulation mode by setting FMOD to 1, the modulator is calibrated. During this time, the clock is unmodulated. Therefore the output clock does not switch immediately to modulated clock. The status of the clock (frequency modulated / unmodulated) is indicated by the FMODRUN status bit. Please refer to the FMODRUN bit description. • Due to the synchronization of the FMOD signal and the synchronized switching to unmodulated clock, it takes less than 9 x T0 (input clock period) before the clock switches to unmodulated clock after the modulator is disabled. The modulator can be disabled at any time. • Before changing the parameter register CMPR, the modulator must be disabled -> FMOD=0 and FMODRUN=0.
bit 0	PDX: Power down bit	"0": Power down mode "1": Power up • PDX is the power down signal for the modulator. Before the frequency modulation mode can be enabled, this bit must be set to 1 and the startup time of 6us must be awaited. Please refer to the application note for a description of the recommended startup sequence. • Before switching to power down mode (PDX=0), the modulator must be disabled -> FMOD=0 and FMODRUN=0.

In the Table below the modulator states are summarized:

Table 2-2 States of the modulator

		FMOD	PDX	FMODRUN (read only)
modulator disabled		0	0	0
modulator power on, waiting modulator startup time (> 6 us)		0	1	0

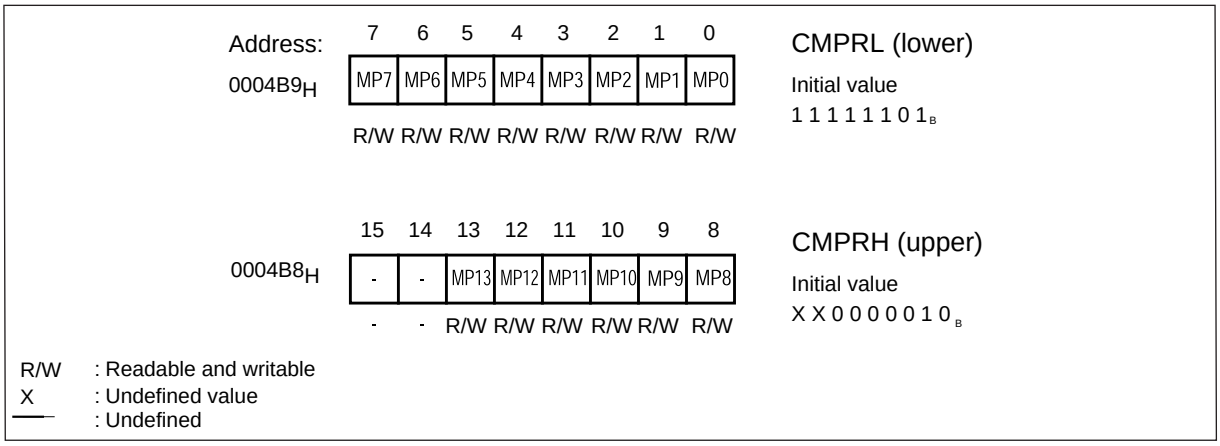
Table 2-2 States of the modulator

		FMOD	PDX	FMODRUN (read only)
modulator enabled in frequency modulation mode, modulator is calibrating, modulation not active		1	1	0
modulator is running in frequency modulation mode modulation is active		1	1	1
	others not allowed			

● Clock Modulation Parameter Register (CMPR)

The Modulation Parameter Register (CMPR) determines the modulation degree in frequency modulation mode.

Figure 2-3 Modulation parameter register



- The modulation parameter determines the degree of modulation and the maximal and minimal occurring frequencies in the modulated clock. Please refer to the application note for a description of an approach to select the optimal setting.
- Each set of possible modulation parameters refers to a particular PLL frequency. The PLL frequency and the selected parameter must match. Please refer to the following table of possible settings.
- The modulation parameter affects only the frequency modulation mode.

Note:

The modulation parameter must be changed only when the modulator is disabled and the RUN flag is 0 (FMOD=0, FMODRUN=0).

● Modulation parameter register contents

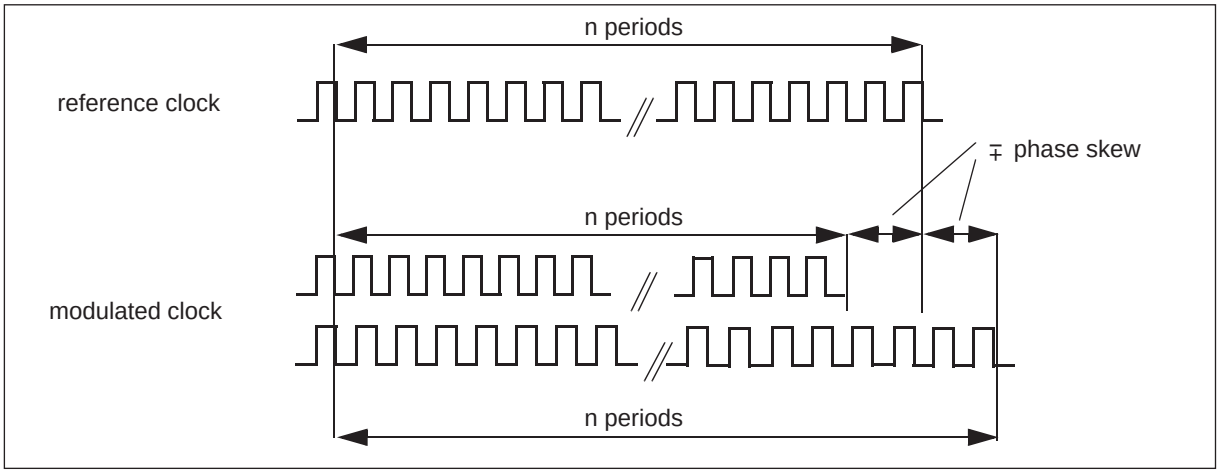
Table 2-3 Function of each bit of the modulation parameter register (CMPR)

Bit name		Function
bit 15, 14	Undefined	

Table 2-3 Function of each bit of the modulation parameter register (CMPR)

Bit name		Function
bit 13 to 0	MP13 to 0: Modulation Parameter bits	Depending on the PLL frequency the following modulation parameter settings are possible. The corresponding CMPR register value is stated in the most right column.

F0:	Frequency of unmodulated input clock (PLL frequency)
T0:	Period of unmodulated input clock (PLL clock period)
resolution:	resolution of frequencies in the modulated clock. low (1) to high (7)
F_{min}:	minimal frequency occurring in the frequency modulated clock
F_{max}:	maximal frequency occurring in the frequency modulated clock
phase skew:	The maximal phase shift of the modulated clock relative to the unmodulated reference clock in terms of clock periods of the unmodulated clock. Example: phase skew=1.44 In worst case, a sequence of n periods of the modulated clock can be 1.44*T0 shorter or 1.44*T0 longer than a sequence of n periods of the unmodulated reference clock. n can be any number > 50 periods
phase skew 50:	phase skew for sequences with n<= 50 periods
CMPR:	register setting of the CMPR register



Note: NOT ALL SETTINGS ARE ALLOWED ON EVERY DEVICE!

Please consider the actual maximal allowed clock frequency of the MCU (refer to the data sheet).

The table below shows the recommended setting for several MCU clocks and modulation parameters:

Table 2-4 Modulation Parameter settings

F0 (MHz)	resolution	mod degree	F_{min} (MHz)	F_{max} (MHz)	+/- phase skew 50 [periods]	+/- phase skew min/max [periods]	CMPR

Please refer to the datasheet of each device about modulation parameter settings.

3. Application Note

Startup/stop sequence for frequency modulation mode.
Modulation parameter for frequency modulation mode.

● Recommended startup sequence for frequency modulation mode

start	1.	Switch modulator from power down to power up mode PDX=1
	2.	Switch on PLL
	3.	Wait PLL lock time (refer to the MCM flag description in the CLOCK chapter of the hardware manual). At the same time the modulator starts up.
	4.	Set CMPR register to a proper setting
	5.	Enable frequency modulation mode FMOD=1 After the calibration is finished, the clock switches from unmodulated to modulated clock and the FMODRUN flag changes to 1
		... running...
stop	6.	Disable modulator FMOD=0
	7.	Wait until FMODRUN changes to 0
	8.	Switch to power down mode PDX=0
	9.	Disable PLL, switch to power down mode, etc.

Note:

Do not enable the modulator before the PLL lock time has elapsed. Do not disable the PLL while the modulator is running.

● Modulation parameter for frequency modulation mode

It is not possible to recommend a particular modulation parameter setting to achieve a particular reduction in EMI. The best setting depends much on the actual application, the whole system and the requirements.

In order to find the optimal modulation parameter setting in frequency modulation mode, the following approach is

recommended.

1.	define the required PLL frequency based on performance needs	e.g. 16 MHz
2.	determine the maximal allowed clock frequency of the MCU	e.g. 32 MHz
3.	choose the setting with the highest resolution and the highest modulation degree, whose maximal frequency is below the maximal allowed clock frequency of the MCU.	e.g. resolution:7, degree:2, CMPR=0x05F2 (F_{max}= 30.34 MHz)
4.	perform EMI measurements	
5.	if the EMI measurements does not fulfill the requirements, you may either	
	reduce the modulation degree at the same frequency resolution (this may improve the reduction in the upper frequency band > 100 MHz, but decrease the reduction of the fundamental < 100 MHz)	e.g. resolution:7, degree:1, CMPR=0x03F9
	or increase the modulation degree at a lower frequency resolution (this may improve the reduction of the fundamental < 100 MHz, but worsen the reduction in the upper frequency band > 100 MHz)	or e.g. resolution:5, degree:3, CMPR=0x0771
6.	repeat item 3) with the new setting and continue until the best settings is identified	

- Recommended settings

The following table lists some example conditions for PLL clock and maximal allowed MCU clock frequency and the recommended clock modulator setting:

Table 3-1 Some example conditions for PLL clock

F0 PLL clock frequency	maximal allowed MCU clock frequency (refer to the data sheet)	clock modulator setting			
		resolution	modulation degree	F_{max}	CMPR

Please refer to the datasheet of each device about modulation parameter settings.

Chapter 18 Timebase Counter

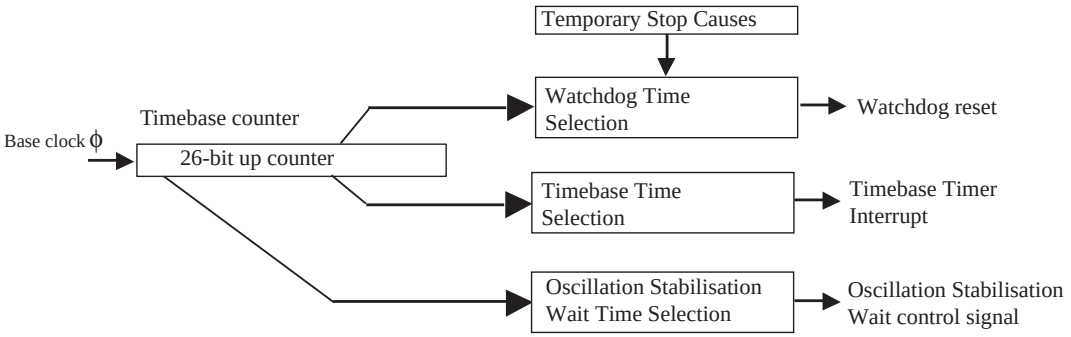
1. Overview

The timebase counter is a 26-bit up-counter that counts the subclock or the main clock divided by two.

When recovering from a state in which the selected clock source for the MCU has been, or may have been, halted, the MCU automatically changes to the oscillation stabilization wait state to avoid any unstable output from the oscillator.

During the oscillation stabilization wait time, supply of internal and external clocks is halted and only the timebase counter continues to operate until the time set by the oscillation stabilization wait time setting has elapsed.

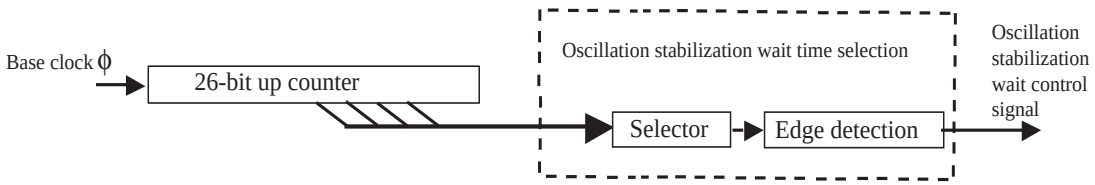
Figure 1-1 Timebase Counter (overview diagram)



This diagram is just an overview. Each part is explained in detail in the respective chapter.

The timebase counter, timebase timer, and watchdog timer are collectively called the watchdog control unit.

Figure 1-2 Timebase counter when used to generate the oscillation stabilisation wait



2. Features

2.1 Timebase Counter (when used to generate the oscillation stabilization wait)

Type : 26-bit up-counter

Number : 1

Clock source : Base clock (depending on clock source selected in CLKR register, $F_{Main-CL/2}$, Main PLL, subclock)

Clear : Cleared automatically when changing to oscillation stabilization wait state.

2.2 Events that Invoke an Oscillation Stabilization Wait

■ Events that invoke an oscillation stabilization wait using the timebase counter

● Wait time after a settings initialization: Invoked automatically (timebase counter)

- $\overline{\text{INIT}}$ Initial oscillation stabilization wait after pin input
- Watchdog reset

- If the main clock oscillation has not been halted: Oscillation stabilization wait not required
- If the main clock oscillation has halted: Oscillation stabilization wait is required

Example: If a watchdog reset occurs during subclock mode with main clock oscillation halted

● Wait time after recovering from stop mode: Invoked automatically (timebase counter)

- Stop mode cases when clock oscillation circuit is halted:
  - The oscillation stabilization wait time for the intended oscillation circuit is required
  - Wait time for main PLL to lock is required (if main PLL is used)
- Stop mode cases when clock oscillation circuit is not halted:

  Oscillation stabilization wait is not required unless the clock oscillation (main/PLL) has been halted.

● When recovering from abnormal state with main PLL selected

Automatically goes to the oscillation stabilization wait state to allow time for the main PLL to lock.

■ **Events that invoke an oscillation stabilization wait using other than the timebase counter**

- Wait time after power on: Provided by pin input
- Wait time after changing from subclock to main clock: Using the main oscillation stabilization wait timer to generate this time is recommended.
- When recovering from main clock oscillation halted: Enabling the main clock oscillation and waiting for oscillation to stabilize is required.
- Main PLL lock wait time (for main clock operation): Using the timebase timer interrupt to generate this time is recommended.
  - A wait time is required after the main PLL operation is enabled.
  - A wait time is required after the main PLL multiplier setting is changed.

3. Configuration

Figure 3-1 Configuration Diagram of the timebase counter used to generate the oscillation stabilisation wait time

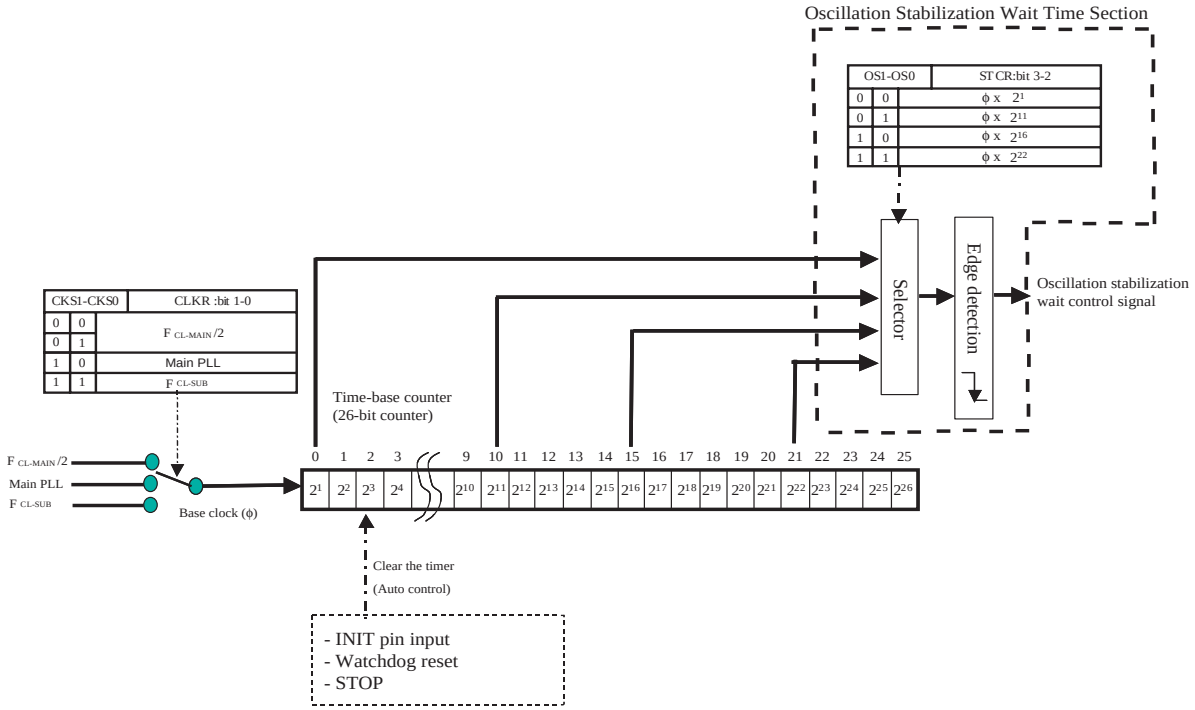


Figure 3-2 Register List

Time-base counter

Address	Bit	7	6	5	4	3	2	1	0	
00481H		STOP	SLEEP	HIZ	SRST	OS1	OS0	OSCD2	OSCD1	STCR

(Standby control)

Clock control

Address	Bit	7	6	5	4	3	2	1	0	
00484H		---	---	---	---	SCKEN	PLL1EN	CLKS1	CLKS0	CLKR

(Clock source control)

4. Registers

4.1 STCR: Standby Control Register

Controls transition to standby modes, pin states during stop mode, whether to halt the clock during stop mode, the oscillation stabilization wait time, and software reset.

Note: See also “Chapter 10 Standby (Page No.155)” and “Chapter 20 Software Watchdog Timer (Page No.275)” chapters.

- **STCR: Address 0481h (Access: Byte)**

7	6	5	4	3	2	1	0	bit
STOP	SLEEP	HIZ	SRST	OS1	OS0	OSCD2	OSCD1	
0	0	1	1	0	0	1	1	Initial value (INITX pin input)
0	0	1	1	X	X	1	1	Initial value (Watchdog)
0	0	X	1	X	X	X	X	Initial value (Software reset)
R/W	R/W	R/W	R1,W	R/W	R/W	RX,W	R/W	Attribute

(See “Meaning of Bit Attribute Symbols (Page No.10)” for details of the attributes.)

- Bit7: Stop mode (STOP)
  - Setting “1” changes to stop mode.
- Bit6: Sleep mode (SLEEP)
  - Setting “1” changes to sleep mode.
  - If this bit and the stop mode bit (STOP) bit are set to “1” at the same time, the device goes to stop mode.
- Bit5: High impedance mode (HIZ)
  - Setting “0” specifies that pins maintain the same states they have on entering stop mode.
  - Setting “1” specifies that pin outputs go to high impedance (Hi-z) during stop mode.
- Bit4: Software reset (SRST)
  - Setting “0” triggers a software reset.
  - Note that negative logic is used.
- Bit3-2: Oscillation stabilization time selection

OS[1:0]	The oscillation stabilization wait time after a reset (INIT) or on recovering from stop mode.		
	Oscillation stabilization wait time	When using main clock (For a 4.0MHz main clock)	When using subclock (For a 32.768kHz subclock)
00	$\Phi \times 2^{1}$	1.00µs	61µs
01	$\Phi \times 2^{11}$	1.0ms	62.5ms
10	$\Phi \times 2^{16}$	32ms	2.0s
11	$\Phi \times 2^{22}$	2s	128s

  - F2: Main clock divided by two or subclock
  - In the case of a reset triggered by an $\overline{\text{INIT}}$ pin input, operation defaults to “00” (Φ x 2^{1}, main clock).
  - In the case of other resets or on recovering from stop mode, the specified clock (main or sub) and oscillation stabilization wait time (OS[1:0]) are used.
  - The count is performed by the timebase counter.
- Bit1: Sub clock oscillation halt (OSCD2)

  Setting “1” specifies that the sub clock oscillation halts in stop mode.
- bit0: Main clock oscillation halt (OSCD1)

  Setting “1” specifies that the main clock oscillation halts in stop mode.

(See “8. Caution (Page No.263)”.)

4.2 CLKR: Clock Source Control Register

Selects the clock source for the base clock used to run the MCU and controls the PLL.

Note: See also the “Chapter 13 Clock Control (Page No.189)”.

- **CLKR: Address 0484h (Access: Byte)**

7	6	5	4	3	2	1	0	bit
-	-	-	-	SCKEN	PLL1EN	CLKS1	CLKS0	
X	X	X	X	0	0	0	0	Initial value (INIT pin input)
X	X	X	X	X	X	X	X	Initial value (Software reset)
R/W0	R/W0	R/W0	R/W0	R/W	R/W	R/W	R/W	Attribute

(See “Meaning of Bit Attribute Symbols (Page No.10)” for details of the attributes.)

- Bit7-4: Reserved bit Always write “0” to this bit. The read value is the value written.
- Bit3: Subclock selection enable (SCKEN)
  - Setting this bit to “1” enables the subclock to be selected.
- Bit2: Main PLL operation enable (PLL1EN)
  - Setting this bit to “1” starts main PLL operation. Main PLL can be selected as the operating clock after the main PLL has locked.
- Bit1-0: Clock source selection

CLKS1	CLKS0	Clock source setting	Mode
0	0	The main clock input from X0/X1 divided by 2 (initial value)	Main clock mode
0	1	The main clock input from X0/X1 divided by 2	Main clock mode
1	0	Main PLL	Main clock mode
1	1	Subclock	Subclock mode

When changing the clock mode, the value of CLKS0 cannot be modified if CLKS1 is “1”

The table below lists the cases when the CLKS1 - CLKS0 bits may or may not be modified.

Table 4-1 Cases When the CLKS1 and CLKS0 Bits May or May Not be Modified

Modify permitted	Modify not permitted
“00” -> “01” or “10”	“00” -> “11”
“01” -> “11” or “00”	“01” -> “10”
“10” -> “00”	“10” -> “01” or “11”
“11” -> “01”	“11” -> “00” or “10”

Example: To select the subclock after an INIT reset, first write “01_B” and then write “11_B” (subclock)

The clock source for the timebase counter during the oscillation stabilization wait time is set by the clock source selection bits.

CLKS1	CLKS0	Clock source for timebase counter during oscillation stabilization wait time	Mode
0	0	The main clock input from X0/X1 divided by 2 (initial value)	Main clock mode
0	1		
1	0		
1	1	Subclock	Subclock mode

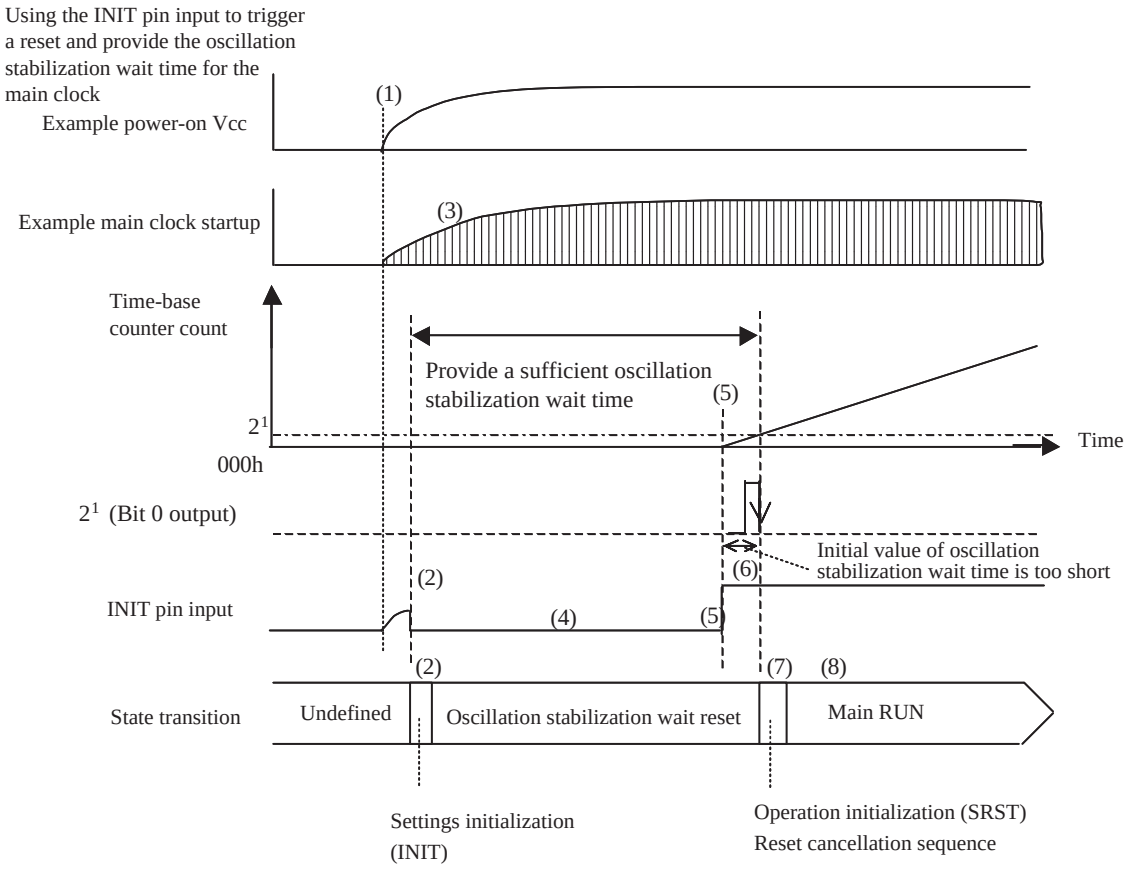
5. Operation

This section describes the events that trigger an oscillation stabilization wait and the operation in each case.

5.1 $\overline{\text{INIT}}$ Pin Input

An oscillation stabilization wait is required after power on. As the wait time provided by the initialized timebase counter is too short, the $\overline{\text{INIT}}$ pin input must be held at the “L” level.

Figure 5-1 Using the Width of the Pin Input to Provide the Oscillation Stabilization Wait Time



(1) Power turned on
(2) Start INIT pin input (Settings initialization reset)
(3) Main clock oscillation starting
(4) INIT pin input (to provide a sufficient time for the main clock oscillation to stabilize)
(5) INIT pin input removed. The timebase counter is initialized and starts counting.
(6) Oscillation stabilization wait time provided by timebase timer/counter (Initial value = minimum value)
(If the INITX pin input (4) is not maintained, the wait time is too short.)
(7) Operation initialization reset, reset cancellation sequence
(8) Main RUN

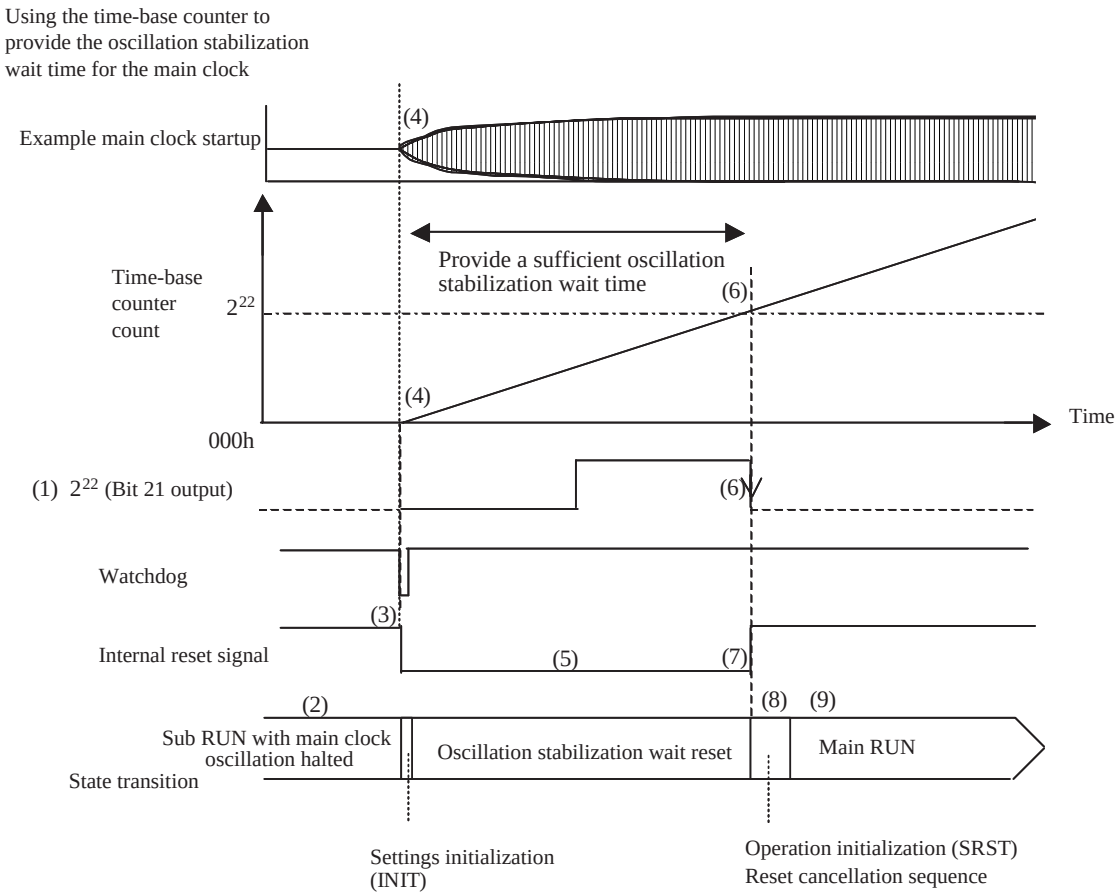
■ $\overline{\text{INIT}}$ Pin input when main clock running

The device goes to the operation initialization reset (RST) state automatically after the minimum oscillation stabilization wait time elapses.

5.2 Watchdog Reset (The specified oscillation stabilization wait time is generated automatically)

If a watchdog reset occurs while the main clock oscillation is halted, the oscillation stabilization wait time is generated automatically. (See figure below.)

Figure 5-2 Watchdog Reset when Main Clock Halted (Sub RUN)



(1) Oscillation stabilization wait time selection (Example: Main clock divided by two x 2^{22})
(Set the interval time beforehand to provide an adequate oscillation stabilization wait time.)

(2) Sub RUN with main clock oscillation halted

(3) Watchdog reset occurs

(4) Main clock oscillation starts
The timebase counter is cleared and starts counting.

(5) Oscillation stabilization wait

(6) Time set as the timebase timer interval time (time set in (1))

(7) Reset released, operation initialization (SRST)

(8) Operation initialization, reset sequence

(9) Main RUN

Note: If a watchdog reset occurs when the main clock oscillation is halted during subclock mode (subclock is being used as clock source) by the main clock oscillation halt bit (OSCCR.OSCDS1), the device changes to the oscillation stabilization wait state after the settings initialization reset (INIT) is released. The device then changes to the operation initialization reset (RST) state after the oscillation stabilization wait time elapses.

■ Watchdog reset when main clock operating

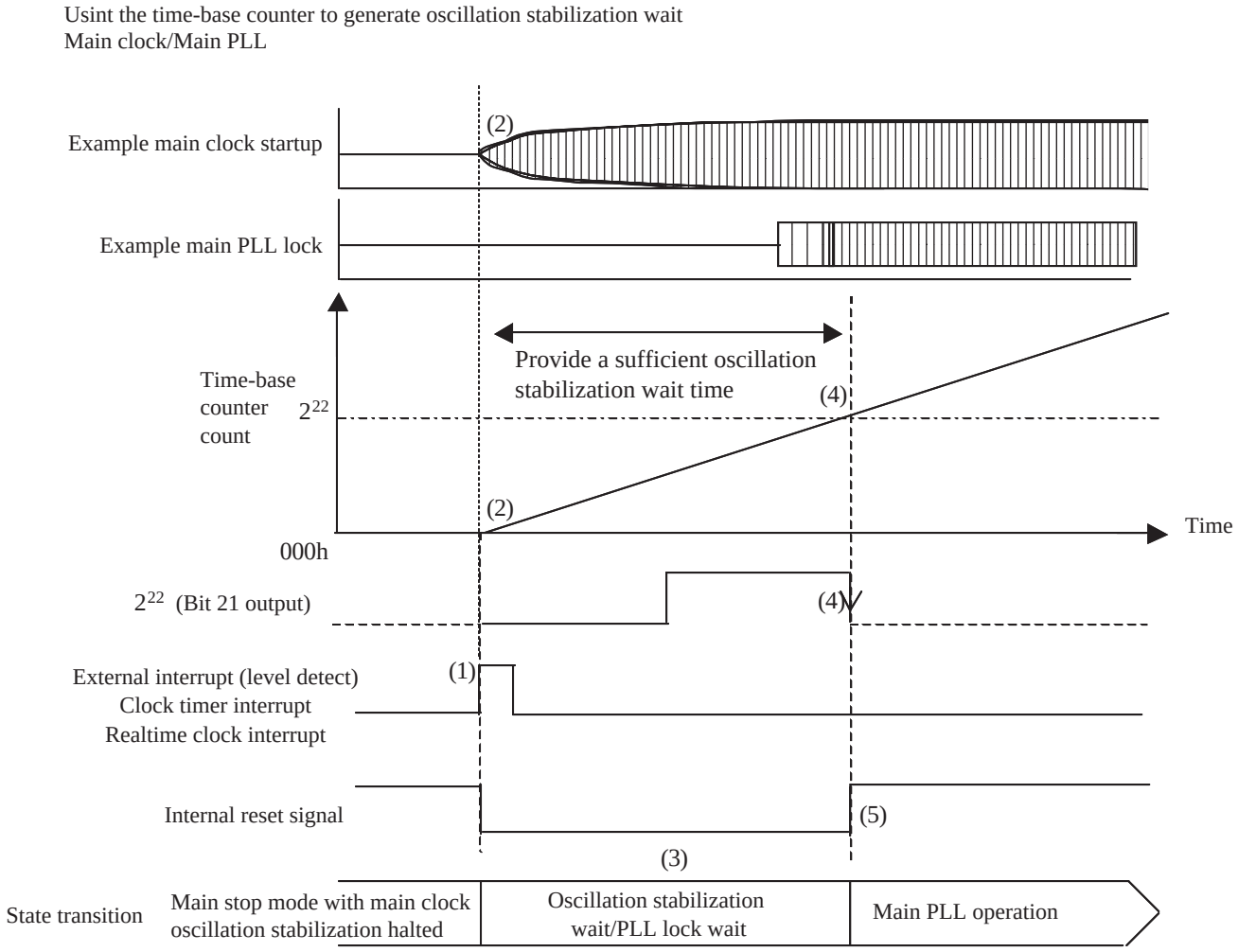
Although no oscillation stabilization wait is required in this case, the specified wait time is generated automatically.

5.3 Recovering from Stop Mode via an Interrupt

■ When changing from main PLL operation to stop mode with the main clock oscillation halted (STCR.OSCD[2:1]="11"):

The main oscillation circuit generates the selected oscillation stabilization time automatically.

Figure 5-3 Recovering from Stop Mode with the Main Clock Halted to Main PLL Operation via an Interrupt



(1) Enabled interrupt is generated (end stop mode)
(2) The timebase counter is cleared automatically and then starts counting.
(3) Oscillation stabilization wait time (specified value)
(Set the interval time beforehand to provide an adequate oscillation stabilization wait time.)
(4) Interval time for timebase counter
(5) Main PLL operation

■ **When changing to stop mode without halting the clock oscillation circuit (main PLL/main/sub):**

Although no oscillation stabilization wait is required in this case, a wait is generated automatically. Accordingly, it is recommended that you set the interval time to its minimum value before changing to stop mode.

- When recovering from stop mode, the device goes to the oscillation stabilization wait state immediately after stop mode is released.
- The next state after the oscillation stabilization wait completes depends on what triggered recovery from stop mode.
- If recovery was triggered by an enabled external interrupt, sub oscillation stabilisation timer interrupt, or main oscillation stabilization wait timer interrupt, the device goes to the normal operating state (RUN).

Note: If the main PLL continues to operate in stop mode, changing to stop mode with the main PLL clock set as the active clock is not permitted. Always set the active clock to the main clock divided by two or the subclock beforehand.)

5.4 The lock wait time for the main PLL must be generated by software.

■ **Wait time after main PLL operation enabled:**

Using the timebase timer interrupt is recommended

However, the main PLL must not be selected as the clock source.

■ **Wait time after main PLL multiplier modified:**

Using the timebase timer interrupt is recommended

However, the main PLL must not be selected as the clock source.

See “Chapter 19 Timebase Timer (Page No.265)” for details.

5.5 Generating an Oscillation Stabilization Wait when Changing from Subclock Mode to Main Clock Mode

■ **When main clock continues to run during subclock mode:**

- If not using main PLL after changing clock: No oscillation stabilization wait time
- If using main PLL after changing clock: Main PLL lock wait is required.
  (Using the timebase timer interrupt is recommended. See “5.3 Recovering from Stop Mode via an Interrupt (Page No.256)”.)

■ **When main clock halts during subclock mode:**

- A main clock oscillation stabilization wait is required before changing clock.
  (Use the oscillation stabilization wait timer for the main clock. See “Chapter 22 Main Oscillation Stabilisation Timer (Page No.293)”.)
- When using the main PLL: A further wait is required for the main PLL to lock.
  (Using the timebase timer interrupt is recommended. See “5.3 Recovering from Stop Mode via an Interrupt (Page No.256)”.)

5.6 When Recovering from an Abnormal State with the Main PLL Selected

When the main PLL is set as the clock source and a problem of some sort occurs in main PLL control (such as the multiplier setting being changed or the main PLL enable bit modified during main PLL operation), the device goes to the oscillation stabilization wait state automatically to provide the main PLL lock time. The device then goes to normal operating mode after the oscillation stabilization wait elapses.

5.7 Types of Oscillation Stabilization Wait

■ Timebase counter

Automatically provides a count for the oscillation stabilization wait time.

When a trigger occurs to change the device to the oscillation stabilization wait state, the timebase counter is cleared and then starts counting the specified oscillation stabilization wait time.

■ “L” level input to $\overline{\text{INIT}}$ pin

When device operation is restarted by inputting an “L” level to the $\overline{\text{INIT}}$ pin while the oscillation is halted (the three cases listed below), the width of the “L” level input provides the stabilization time required by the oscillation circuit.

- $\overline{\text{INIT}}$ pin input after power turned on.
- $\overline{\text{INIT}}$ pin input when oscillation halted during stop mode
- $\overline{\text{INIT}}$ pin input when subclock selected as clock source and main clock oscillation halted.

■ Timebase timer

Using the timebase timer to generate the main PLL lock time is recommended.

See “Chapter 20 Software Watchdog Timer (Page No.275)” for details.

■ Main oscillation stabilization wait timer

Used when restarting the main clock while operating in subclock mode.

See “Chapter 22 Main Oscillation Stabilisation Timer (Page No.293)” for details.

■ Sub oscillation stabilisation wait timer

Used when restarting the subclock while operating in main clock mode.

See “Chapter 23 Sub Oscillation Stabilisation Timer (Page No.303)” for details.

5.8 Whether or not a Stabilization Wait is Required for Each State Transition

See figure below.

Reset due to INIT pin input

State	PLL	Main	Sub	Operation after INIT signal input	Is oscillation stabilization wait required?	Remarks
Wait time after power on	x	x	x	Operate using initial value (main clock (1/2))	Must be provided by width of INIT pin input	The automatic oscillation stabilization wait (minimum value) is too short
Main clock running (= main clock oscillation running)	x	O	O	Operate using initial value (main clock (1/2))	Oscillation stabilization wait not required	Uses the automatic oscillation stabilization wait (initial value = minimum value)
Main PLL running (=main PLL oscillation running)	O	O	O			
Main stop (main clock/main PLL oscillation running)	O/x	O	O			
Sub clock running (main clock/main PLL oscillation running)	O/x	O	O			
Sub stop (main clock oscillation running)	O/x	O	O			
From main clock (1/2) running to main stop (main clock halted)	x	x	O	Operate using initial value (main clock (1/2))	Must be provided by width of INIT pin input	The automatic oscillation stabilization wait (initial value = minimum value) is too short
From main PLL running to main stop (main clock halted)	x	x	O			
Sub clock running (main clock halted)	x	x	O			
Sub stop/sub sleep (main clock halted)	x	x	O			
Main clock oscillation stabilization wait in progress	x	x	O/x			

Watchdog reset

State	PLL	Main	Sub	Operation after watchdog reset	Is oscillation stabilization wait required?	Remarks
Main clock running (= main clock oscillation running)	x	O	O	Operate using initial value (main clock (1/2))	Oscillation stabilization wait not required	Uses automatic oscillation stabilization wait. (The oscillation stabilization wait time is not initialized.)
Main PLL running (=main PLL oscillation running)	O	O	O			
Sub clock running (main clock/main PLL oscillation running)	O/x	O	O			
Sub clock running (main clock halted)	x	x	O	Operate using initial value (main clock (1/2))	Main clock oscillation stabilization wait is required	Uses automatic oscillation stabilization wait. Set an appropriate wait time setting.

Recovery from stop mode via an interrupt

State	PLL	Main	Sub	Operation after recovering from stop mode via an interrupt	Is oscillation stabilization wait required?	Remarks
Main stop (main clock oscillation continues during stop mode, main PLL running) *	O	O	O	Previous operation state (main clock (1/2))	When changing to main PLL after recovering, a main PLL lock wait time is required.	Uses automatic oscillation stabilization wait. Set wait time setting to OS[1:0] ="11" to provide the main PLL lock wait time.
Main stop (main clock oscillation continues during stop mode, main PLL halted)	x	O	O	Previous operation state (main clock (1/2))	Oscillation stabilization wait not required	Uses automatic oscillation stabilization wait. Set wait time setting to minimum value.
Main stop (main clock oscillation halted during stop mode (automatic))	x	x	O	Previous operation state (main clock (1/2))	Main clock oscillation and main PLL lock wait required.	Uses automatic oscillation stabilization wait. Set an appropriate wait time setting.
Sub stop (main clock and main PLL running)	O/x	O	O	Previous operation state (sub)	Oscillation stabilization wait not required	Uses automatic oscillation stabilization wait. Set wait time setting to minimum value. (However, a oscillation stabilization wait for the main clock is required before changing to main clock operation.)
Sub stop (main clock halted)	x	x	O			

* The active clock must be set to the main clock divided by two before changing to stop mode.

Main clock oscillation enable

State	PLL	Main	Sub	Main clock oscillation enabled	Is oscillation stabilization wait required?	Remarks
Sub clock running (main clock halted)	x	x	O	Start main clock oscillation	Main clock oscillation stabilization wait is required	Using the main clock oscillation stabilization wait timer to generate the time is recommended.

Main PLL oscillation enable

State	PLL	Main	Sub	Main PLL oscillation enable	Is oscillation stabilization wait required?	Remarks
Sub clock running (main clock running)	x	O	O	Start main clock oscillation/Change PLL multiplier setting	Main PLL lock wait required	Using the time-base timer to generate the main PLL lock wait time is recommended.
Main clock (1/2) running	x	O	O			

O: Oscillation running, x: Oscillation halted

6. Settings

Table 6-1 Settings Required to Specify the Oscillation Stabilization Wait Time

Setting	Setting register	Setting procedure*
Oscillation stabilization wait time setting	Standby control register (STCR)	See 7.1.

*: For the setting procedure, refer to the section indicated by the number.

Table 6-2 Settings Required to Setup an INITX Pin Reset

Setting	Setting item	Setting procedure
INITX pin input	Refer to the oscillator parameters and the reset parameters in the Data Sheet.	–

- Settings required to specify the oscillation stabilization wait time for the main clock
  See “Chapter 22 Main Oscillation Stabilisation Timer (Page No.293)”.
- Settings required to specify the PLL lock wait time
  See “Chapter 19 Timebase Timer (Page No.265)”.

7. Q&A

7.1 How do I setup the oscillation stabilization wait time that is generated automatically?

Use the oscillation stabilization wait time selection bits (STCR.OS[1:0]). (The following lists likely scenarios and the required settings.)

Scenario	Oscillation stabilization wait time selection bits (OS[1:0])	Example oscillation stabilization wait time after a reset (INIT) or on recovering from stop mode		
		Oscillation stabiliza-tion wait time	4.0MHz Main clock running	32.768kHz Subclock running
To not halt the main PLL or oscillator during stop mode (No oscillation stabilization wait time required)	Set “00”.	$\Phi 2 \times 2^{1}$	1.00µs	61µs
To not stop the oscillator during external clock input or stop mode (Main PLL lock wait time)	Set “01”.	$\Phi 2 \times 2^{11}$	1.0ms	62.5ms
When using an oscillator with a fast stabilization time such as a ceramic resonator (Oscillation stabilization wait time (medium))	Set to “10”.	$\Phi 2 \times 2^{16}$	32ms	2.0s
When using a standard quartz oscillator (Oscillation stabilization wait time (long))	Set to “11”.	$\Phi 2 \times 2^{22}$	2s	128s

- F2: Main clock divided by 2, or subclock
- In the case of an $\overline{\text{INIT}}$ pin input, operation defaults to “00” (F2 x 2^{1}=main clock divided by 4).
- For other resets and when recovering from stop mode, the operation is in accordance with the specified clock (main or sub) and oscillation stabilization wait time (OS[1:0]) setting.
- The count is performed by the timebase counter.
- Once the time is selected, it is not initialized except by a settings initialization triggered by the external $\overline{\text{INIT}}$ pin.

7.2 How do I set the oscillation stabilization wait time without generating it automatically?

The settings described below for various cases are required.

State (before transition)	Oscillation			Condition (after transition)	Is oscillation stabilization wait required?	To set the oscillation stabilization wait time
	PLL	Main	Sub			
Wait time after power on	×	×	×	Operation after INIT signal input defaults to main clock (1/2) (Initial value)	Main clock oscillation stabilization wait is required	As the automatic oscillation stabilization wait (minimum value) is too short, the width of the INIT pin must be sufficient to provide the stabilization time.
Subclock operation (main clock halted)	×	×	O			
Sub sleep, sub stop (main clock halted)	×	×	O			
Main clock oscillation stabilization wait	×	×	O/×			

State (before transition)	Oscillation			Main clock oscillation enabled	Is oscillation stabilization wait required?	To set the oscillation stabilization wait time
	PLL	Main	Sub			
Subclock operation (main clock halted)	×	×	O	Main clock running	Main clock oscillation stabilization wait is required	Using the main clock oscillation stabilization wait timer to generate the time is recommended.

State (before transition)	Oscillation			Main PLL running and enabled	Is oscillation stabilization wait required?	To set the oscillation stabilization wait time
	PLL	Main	Sub			
Subclock running (main clock running)	×	O	O	Start main PLL oscillation/ Change PLL multiplier setting	Main PLL lock wait required	Using the timebase timer to generate the main PLL lock wait time is recommended.
Main clock (1/2) running	×	O	O			

O: Oscillation running,
×: Oscillation halted

7.3 What is the clear timing for the timebase counter?

- The timebase counter is only cleared automatically by INIT pin input.
  The timebase counter automatically starts counting after being cleared.
- The timebase counter can also be cleared by software.
  See “Chapter 19 Timebase Timer (Page No.265)” for details.

8. Caution

- Clock source
  If the clock selected as the clock source is not stable, an oscillation stabilization wait time is required.

- Oscillation stabilization wait time
  The wait time set in the oscillation stabilization time selection bits (STCR.OS[1:0]) is not initialized by any reset except a reset triggered by the external INITX pin input, the RC based watchdog or the Clock Supervisor. For other resets including settings initialization resets (timebase counter based watchdog reset) and operation initialization resets (RST), the wait time set prior to the reset is used.

- Watchdog reset (Timebase Counter based watchdog)
  Although an oscillation stabilization wait time is not required if a watchdog reset occurs while the main clock is running (main or sub), a wait time is generated automatically. In this case, the oscillation stabilization wait time (STCR.OS[1:0]) is not initialized.

- “L” level input to $\overline{\text{INIT}}$ pin
  As the oscillation stabilization wait time is initialized to its minimum value when an initialization is triggered by an $\overline{\text{INIT}}$ pin input, the wait time in this case is too short. Ensure the $\overline{\text{INIT}}$ pin input width is long enough to provide the oscillation stabilization wait time.
  In the following three cases, maintain the $\overline{\text{INIT}}$ pin input at the “L” level for long enough to provide the oscillation stabilization wait time required by the oscillation circuit.
  - $\overline{\text{INIT}}$ pin input after turning on the power
  - $\overline{\text{INIT}}$ pin input after oscillation halted in stop mode
  - $\overline{\text{INIT}}$ pin input when subclock selected as the clock source and main clock oscillation halted

  (Accordingly, to stabilize the oscillation of both the main and subclocks, input an “L” level to the $\overline{\text{INIT}}$ pin for long enough to provide a sufficient oscillation stabilization time for both the main and subclocks.)

- Main PLL lock wait
  If enabling the main PLL from the halted state after program execution starts, the main PLL must not be used until after sufficient time has elapsed for the main PLL to lock.
  Similarly, when changing the multiplier setting for the main PLL when the PLL is running, the new main PLL clock must not be used until sufficient time has elapsed for the main PLL to lock.
  Using the timebase timer interrupt to generate the main PLL lock wait time is recommended.

- Cases when oscillation stabilization wait is not required
  Although no oscillation stabilization wait is required when recovering via an interrupt from main stop or sub stop mode when the main clock oscillation has not been halted, the oscillation stabilization wait is generated automatically. Setting the wait time to its minimum value prior to entering stop mode is recommended.

8.Caution

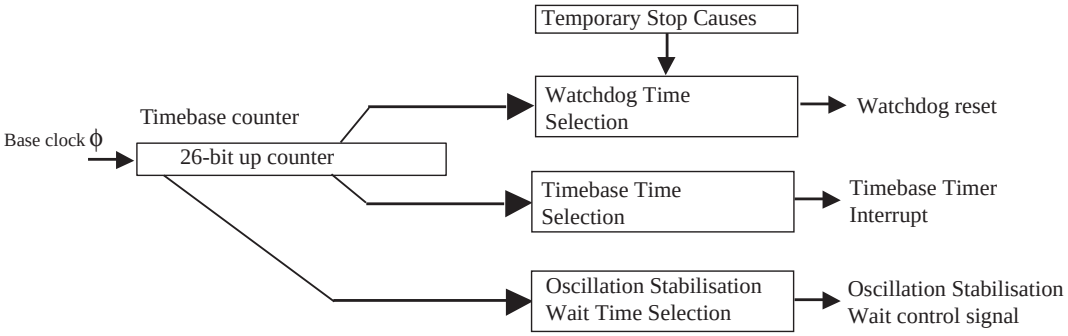
Chapter 19 Timebase Timer

1. Overview

The timebase timer is a selector that uses the output from a 26-bit timebase counter using the base clock (F).

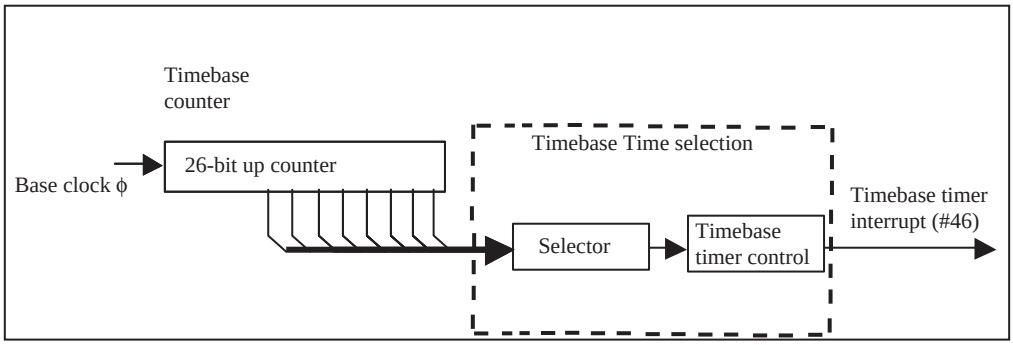
The timebase timer is an interval-interrupt generating timer that is used to acquire main PLL lock wait time and to count a long time.

Figure 1-1 Timebase Counter (overview diagram)



This diagram is just an overview. Each part is explained in detail in the respective chapter.
The timebase counter, timebase timer, and watchdog timer are collectively called the watchdog control unit.

Figure 1-2 Timebase Counter used to generate the Timebase Timer Interrupt



2. Features

■ Timebase timer (TBT)

- Type : Detects timebase timer bit output and generates an interval interrupt.
- Quantity : 1
- Interval time: 8 types (Timebase timer bit output)

  Period = $2^{11}/F_F$, $2^{12}/F_F$, $2^{13}/F_F$, $2^{22}/F_F$, $2^{23}/F_F$, $2^{24}/F_F$,

  $2^{25}/F_F$, $2^{26}/F_F$
- Operation start/stop: Always in operation (Can be replaced by interrupt request enable control)
- Timebase counter clear: Continuously writes “A5”“5A” in the timebase counter clear register CTBR using the software.

3. Configuration

Figure 3-1 Configuration Diagram of the Timebase Timer

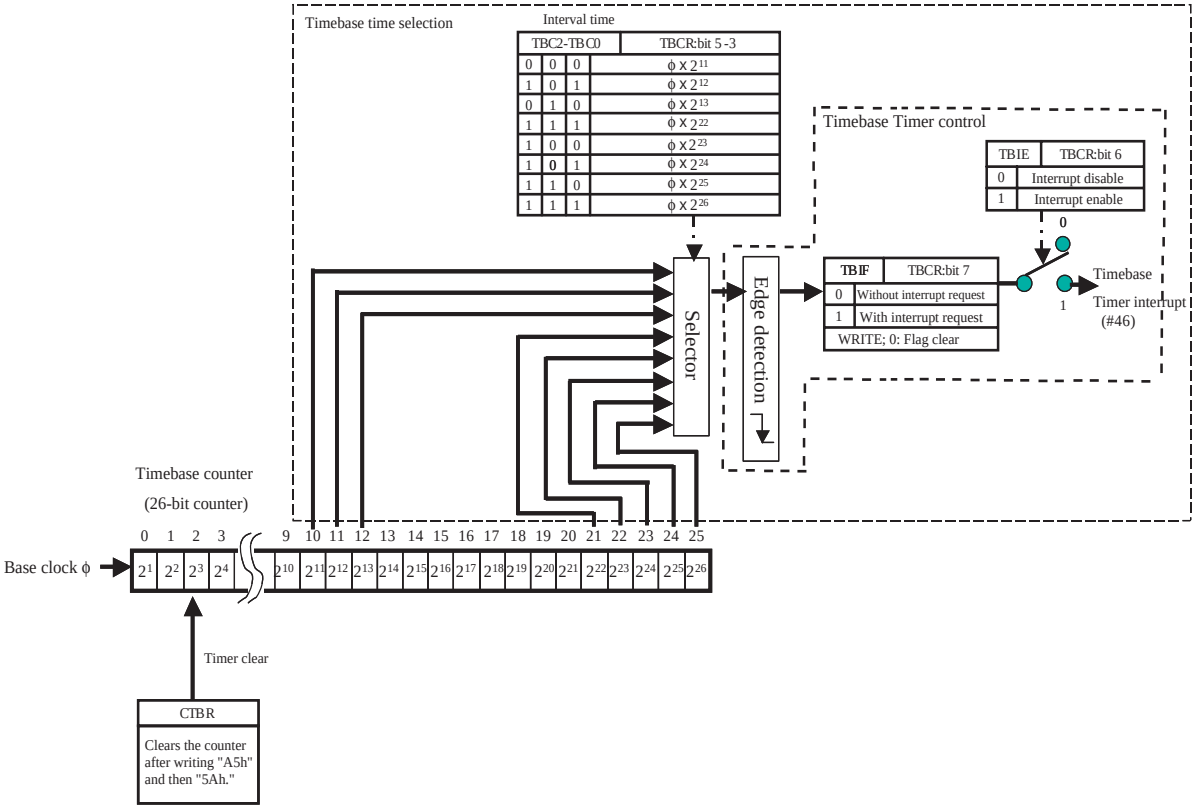
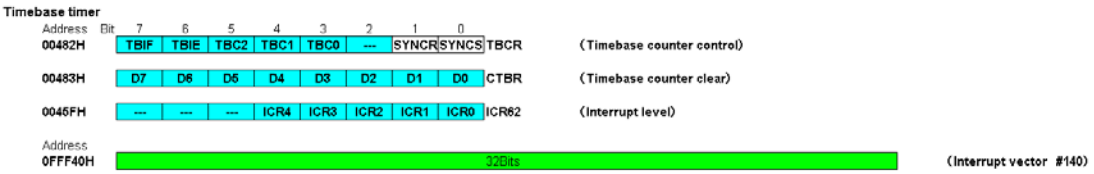


Figure 3-2 List of Registers



4. Register

4.1 TBCR: Timebase Timer Control Register

This register is used to set timebase timer interrupt control, reset/ standby operation option etc.

Note: Refer also to “Chapter 10 Standby (Page No.155)”.

- **TBCR: Address 0482h (Access: Byte)**

7	6	5	4	3	2	1	0	bit
TBIF	TBIE	TBC2	TBC1	TBC0	---	SYNCR	SYNCS	
0	0	X	X	X	X	0	0	Initial value (INIT terminal input, watchdog reset)
0	0	X	X	X	X	X	X	Initial value (the software reset)
R(RM1),W	R/W	R/W	R/W	R/W	RX/WX	RX/WX	R/W	Attribute

(Refer to “Meaning of Bit Attribute Symbols (Page No.10)” for the attributes.)

- Bit7: Timebase timer interrupt request flag

TBIF	Operation	
	Read	Write
0	With no interrupt request	Flag is cleared
1	With interrupt request (The interval time set by the timebase timer has elapsed)	Writing does not affect operation

  - An interrupt request is generated if the timebase timer interrupt request enable bit is “1”, and if the timebase timer interrupt request flag is “1”.

- Bit6: Timebase timer interrupt request enable

TBIE	Operation
0	Disabling the timebase timer interrupt request
1	Enabling the timebase timer interrupt request

- Bit5-3: Selecting the timebase timer interval time

TBC2-TBC0	Interval time	Example	
		While the main clock operates (4.0MHz, PLL8 multiply)	While the subclock operates (32.768kHz)
000	$\Phi \times 2^{11}$	64.0µs	62.5ms
001	$\Phi \times 2^{12}$	128µs	125ms
010	$\Phi \times 2^{13}$	256µs	250ms
011	$\Phi \times 2^{22}$	131ms	128s
100	$\Phi \times 2^{23}$	262ms	256s
101	$\Phi \times 2^{24}$	524ms	512s
110	$\Phi \times 2^{25}$	1048ms	1024s
111	$\Phi \times 2^{26}$	2097ms	2048s

  - Be sure to set the interval time before an interrupt.
    (Oscillation stability wait time used when returning to the stop caused by an interrupt)

- Bit2: Reserved bit

  Writing does not affect the operation. The read value is indefinite.

- Bit1: Enabling the synchronous reset operation

SYNCR	Operation
0	Ordinary reset operation
1	Synchronous reset operation enable

  - Ordinary operation reset: Immediately resets the operation initialization when the operation initialization reset (RST) request is generated.
    Synchronous reset: Resets the operation initialization after all accesses to the bus have stopped.
- Bit0: Synchronous standby operation enable

SYNCS	Operation
0	Ordinary reset operation (In this product, any setting is prohibited)
1	Synchronous standby operation enable (Be sure to set before making the transition to standby)

4.2 CTBR: Timebase Counter Clear Register

This register is used to initialize the timebase counter.

- **CTBR: Address 0483h (Access: Byte)**

7	6	5	4	3	2	1	0	bit
D7	D6	D5	D4	D3	D2	D1	D0	
X	X	X	X	X	X	X	X	Initial value
RX/W	RX/W	RX/W	RX/W	RX/W	RX/W	RX/W	RX/W	Attribute

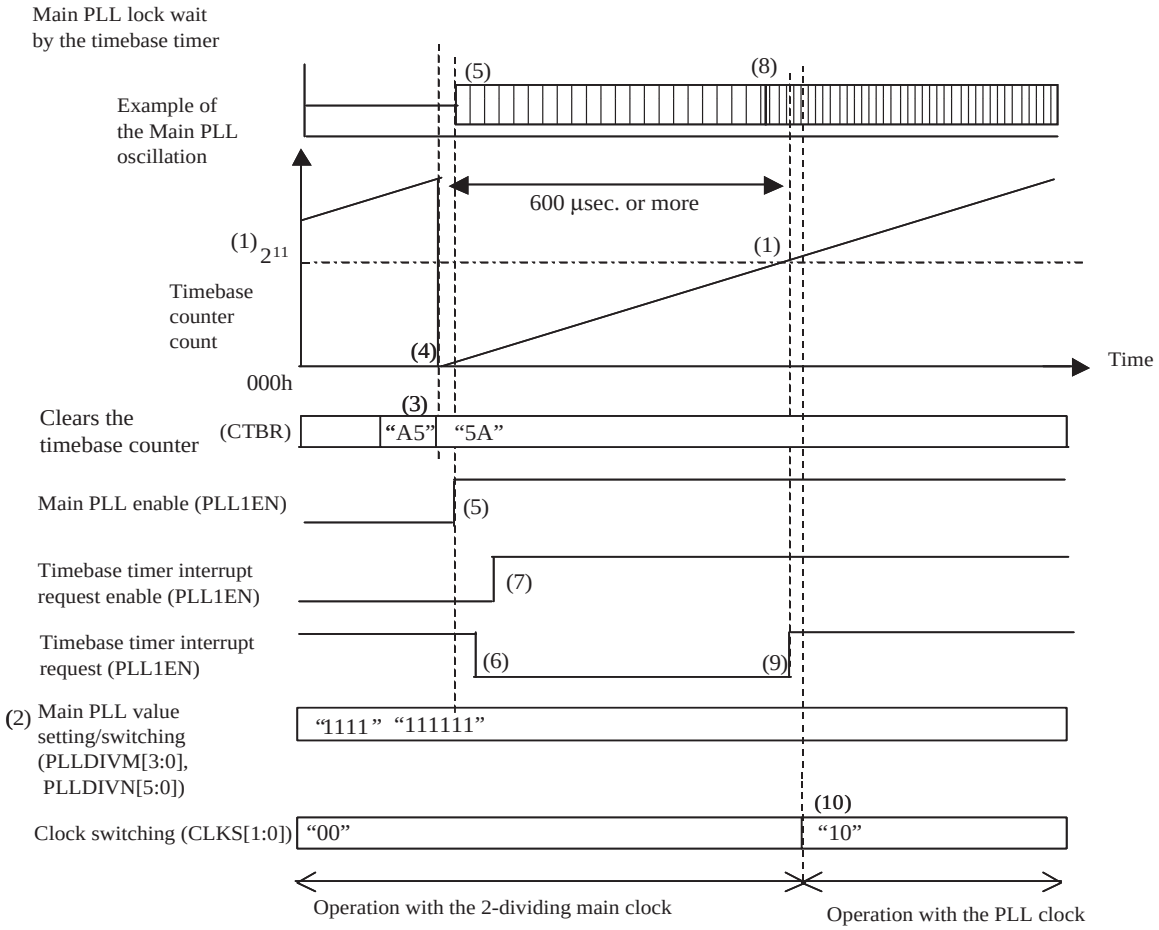
(Refer to "Meaning of Bit Attribute Symbols (Page No.10)" for the attributes)

- Continuously writing "A5$_H$", "5A$_H$" in the timebase counter clear register clears the timebase counter immediately after writing "5A$_H$". (All bits are "0")
  There is no time restrictions between "A5$_H$" and "5A$_H$", but if "A5$_H$" is written followed by the one other than "5A$_H$", you should write "A5$_H$" again. If not, the timebase counter cannot be cleared even if "5A$_H$" is written.
- The read value is indefinite.
- Clearing the timebase counter using the timebase counter clear register temporarily modifies the relevant items shown below.
  - Oscillation stability wait interval
  - Watchdog timer period
  - Timebase timer period

5. Operation

Timebase timer operation is described.

5.1 Timebase Timer Interrupt Example (Main PLL Lock Wait)



(1) Selecting the interval value in the timebase timer
(2) Selecting the main PLL value (Setting / Switching)
(3) Writing data in the timebase counter clear register in the order of "A5" and "5A"
(4) Writing "5A" clears the timebase counter and causes the count to start from"0"
(5) Enables the main PLL to operate
(6) Clears the timebase timer interrupt request using the software
(7) Setting the timebase timer interrupt request enable bit to "1"
(8) The main PLL locks
(9) A timebase timer interrupt occurs when the timebase timer interval time has elapsed
(10) Setting the main PLL to the operation clock

6. Setting

Table 6-1 Setting Required for the Timebase Timer

Setting	Setting register	Setting method*
Setting the interval time	Timebase timer control register control register (TBCR)	Refer to 7.1
Timebase counter clear	Timebase counter clear register (CTBR)	Refer to 7.5

*: Refer to the number for more information on the setting method.

Table 6-2 Setting Required for Interrupting the Timebase Timer

Setting	Setting register	Setting method*
Setting the timebase timer interrupt vector and interrupt level	Refer to “Chapter 24 Interrupt Control (Page No.315)”	Refer to 7.6
Setting the main clock oscillation stability wait timer interrupt Interrupt request clear Interrupt request enable	Timebase timer control register control register (TBCR)	Refer to 7.7

*: Refer to the number for more information on the setting method.

7. Q & A

7.1 What are the types of interval time used in the timebase timer (and the timebase counter used by the timebase timer) and how to select them?

There are eight types of interval time, and they are set using the interval selection bit (TBCR.TBC[2:0]).

Timebase timer Interval time	Interval selection bit (TBC[2:0])	Example Interval time		
		FΦ =2MHz	FΦ = 32MHz	FΦ = 32.768kHz
How to select $\Phi \times 2^{11}$?	Set the value to “000”	1.024ms	64µs	62.5ms
How to select $\Phi \times 2^{12}$?	Set the value to “001”	2.048ms	128µs	125ms
How to select $\Phi \times 2^{13}$?	Set the value to “010”	4.096ms	256µs	256ms
How to select $\Phi \times 2^{22}$?	Set the value to “011”	2.097s	131ms	128s
How to select $\Phi \times 2^{23}$?	Set the value to “100”	4.194s	262ms	256s
How to select $\Phi \times 2^{24}$?	Set the value to “101”	8.388s	524ms	512s
How to select $\Phi \times 2^{25}$?	Set the value to “110”	16.77s	1.04s	1024s
How to select $\Phi \times 2^{26}$?	Set the value to “111”	33.55s	2.09s	2048s

F: This is the base clock. (Refer to “Chapter 13 Clock Control (Page No.189)”.)

7.2 What Is the count clock of the timebase counter?

The count clock is a base clock. Refer to “Chapter 13 Clock Control (Page No.189)”.

7.3 How to operate the timebase timer?

The timebase timer is always operating. (Setting is unnecessary.)

However, to use interval interrupt, interrupt setting is required.

7.4 How is the timebase timer (=timebase counter) operation stopped?

It cannot be stopped.

7.5 How is the timebase counter (=timebase timer) cleared?

If you write $\{A5_H\}$ and $\{5A_H\}$ successively in the timebase counter clear register CTBR, the timebase counter is cleared immediately after $\{5A_H\}$. (All bits are “0”.)

However, if the timebase counter is cleared, the watchdog timer is affected. (Refer to “8. Caution (Page No.273)”)

7.6 How about the interrupt-associated registers?

Setting timebase timer’s interrupt vector and interrupt level

The relationship between the interrupt level and vector is shown in the following table.

Refer to “Chapter 24 Interrupt Control (Page No.315)” for more information on the interrupt level and interrupt vector.

Interrupt vector (default)	Interrupt level setting bit (ICR[4:0])
#140 Address: 0FFDCCh	Interrupt level register (ICR62) Address: 047Eh

The interrupt request flag (TBCR.TBIF) cannot automatically be cleared. As a result, clear it by the software before returning from an interrupt service. (Write “0” in the interrupt request flag (TBIF).)

7.7 What are the interrupt types?

One type of interrupt is available, and an interrupt is generated when the interval time that is set using the interval selection bit (TBCR.TBC[2:0]) has elapsed. (Selection is unnecessary.)

7.8 How is an interrupt enabled?

Interrupt request enable and interrupt request flag
Setting interrupt enable is conducted using the interrupt request enable bit (TBCR.TBIE).

	Interrupt request enable bit (TBIE)
Interrupt disable	Set the value to “0”
Interrupt enable	Set the value to “1”

Clearing the interrupt request is performed using the interrupt request bit (TBCR.TBIF).

	Interrupt request bit (TBIF)
Interrupt request clear	“0” is written

8. Caution

- The main PLL needs the PLL lock wait time after operation enable and after modifying the rate of multiply.
  We recommend that this main PLL lock wait time be acquired using the timebase interrupt.
  The lockup time of PLL is approximately 600us. Make sure that the PLL lock wait time is set to a value a little larger than 600us.

- Regarding the interval setting
  - When modifying the timebase timer interval time, set the interrupt request enable bit (TBIE) to “0” in advance to disable an interrupt.
  - The timebase counter is always counting. Clear the timebase counter before enabling an interrupt to acquire an accurate interval interrupt time using the timebase timer.

  (If not, an interrupt request may be generated immediately after an interrupt enable.)

- About clearing the timebase counter using a program
  - If you write data in the timebase counter clear register CTBR in the order of “$A5_H$” and “$5A_H$” the timebase counter is cleared immediately after writing “$5A_H$”. (All bits are “0.”)
  - Although there are no restrictions on the write timings for “$A5_H$” and “$5A_H$” writing “$A5_H$” followed by a one other than “$5A_H$” the clearing operation is not performed if “$A5_H$” is not written again even if “$5A_H$” is written.
  - If the timebase counter is cleared, the reset signal to the watchdog is generated with a delay once.

- About clearing the timebase counter by the hardware
  The timebase counter is cleared by the STOP mode and the setting initialization reset ($\overline{\text{INIT}}$ pin input, watchdog reset). (All bits are “0.”)

- In the stop mode
  When returning to the interrupt from a stop, the timebase counter is used to acquire the clock oscillation stability wait time. As a result, there is a possibility to unintentionally generate timebase timer’s interval interrupt. Therefore, disable the timebase timer interrupt not to use the timebase timer before the stop is set.

8.Caution

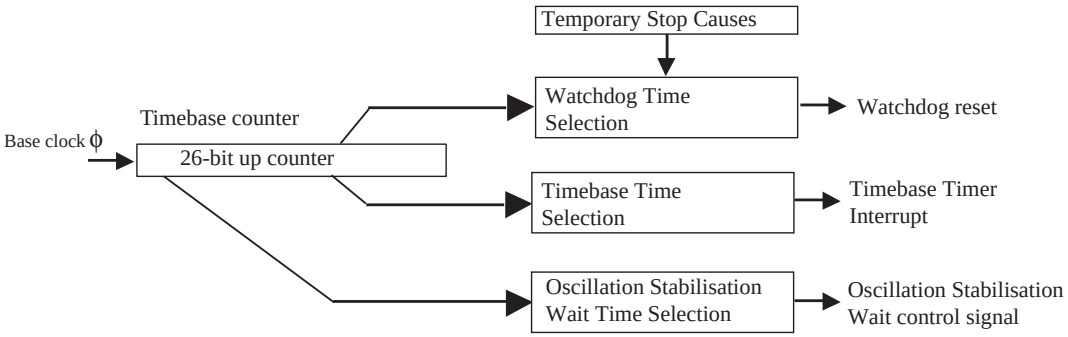
Chapter 20 Software Watchdog Timer

1. Overview

The software watchdog timer consists of a selector that uses the output from a 26-bit timebase counter using the base clock (F) and a one-bit counter.

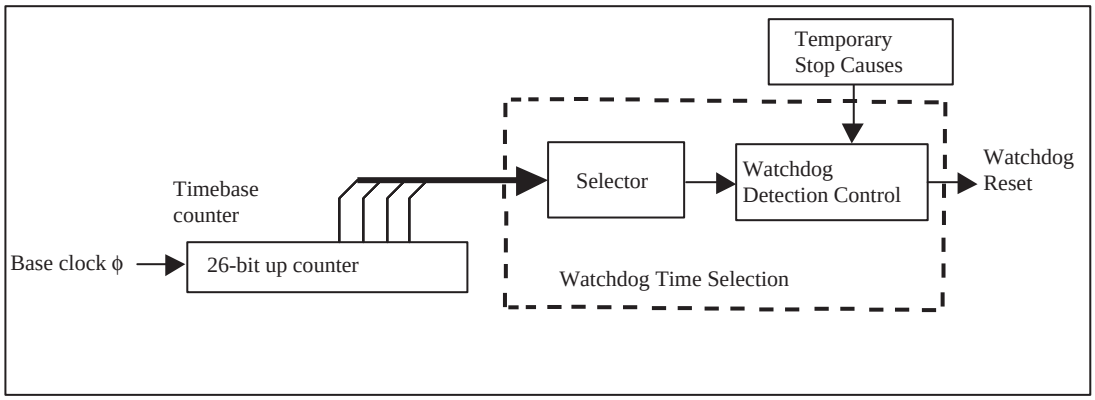
The watchdog timer generates the watchdog reset (initial setting reset) if the generation delay operation (an interval watchdog reset) is disabled due to problems such as program runaway.

Figure 1-1 Timebase Counter (overview diagram)



This diagram is just an overview. Each part is explained in detail in the respective chapter.
The timebase counter, timebase timer, and watchdog timer are collectively called the watchdog control unit.

Figure 1-2 Timebase Counter used to generate the Watchdog Reset



2. Features

■ Watchdog timer

- Type : Generates the watchdog reset (INIT) with the overflow from one-bit counter
- Quantity : 1
- Count clock (interval time): Bit output from the timebase timer
  4 types $2^{20}/F_F$, $2^{22}/F_F$, $2^{24}/F_F$, $2^{26}/F_F$
  (Can be set only once after the reset (RST).)
- Clearing 1-bit counter:
  Successively writes “A5”“5A” to watchdog reset generation delay register WPR by the software.

- Operation start/stop: This timer starts to operate once it writes data to the watchdog control register RSRR for the first time after the reset (RST). This timer stops only by the reset (RST).

3. Configuration

Figure 3-1 Configuration Diagram of the Watchdog Timer

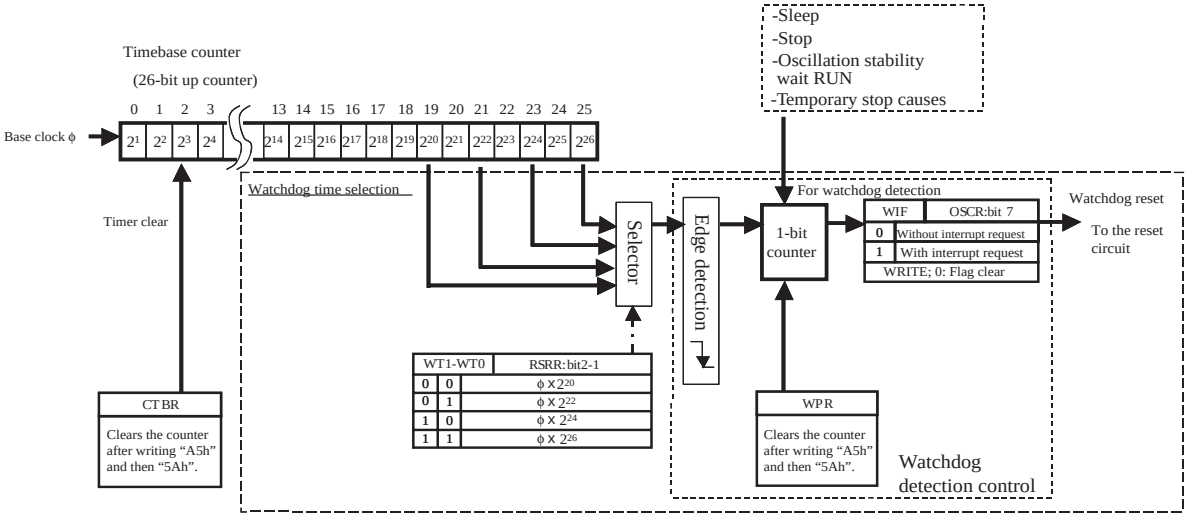
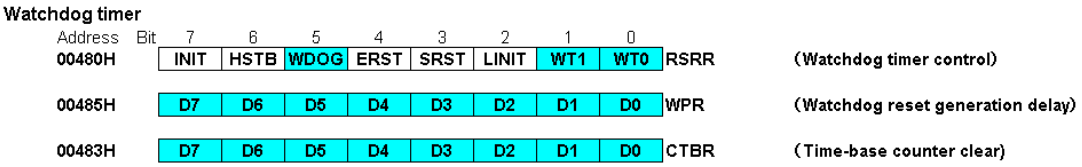


Figure 3-2 List of Registers



4. Register

4.1 RSRR: Watchdog Timer Control Register

This register is used to set watchdog timer periods, and execute the startup control.
(This register also functions as the reset cause register that stores previously generated reset causes.)

Note: Refer also to “Chapter 9 Reset (Page No.139)”.

- **RSRR: Address 0480h (Access: Byte, Half-word)**

7	6	5	4	3	2	1	0	bit
INIT	HSTB	WDOG	ERST	SRST	LINIT	WT1	WT0	
1	0	0	0	0	0	0	0	Initial value ($\overline{\text{INIT}}$ pin input)
-	-	-	X	X	-	0	0	Initial value (Watchdog reset)
X	X	X	-	-	X	0	0	Initial value (Software reset)
R/WX	R/WX	R/WX	R/WX	R/WX	R/WX	R/W	R/W	Attribute

(For the attributes, refer to “Meaning of Bit Attribute Symbols (Page No.10)”.)

The watchdog timer starts once it writes the watchdog timer control register.

- Bit7: Initialization reset occurred flag

  Indicates whether a reset (INIT) was triggered by $\overline{\text{INIT}}$ pin input.

INIT	Meaning
0	No INIT has been triggered by the $\overline{\text{INIT}}$ pin input.
1	INIT has been triggered by the $\overline{\text{INIT}}$ pin input.

  The initialization reset occurred flag (INIT) is cleared to “0” after reading.

- Bit6: Hardware Standby reset occurred flag

  Indicates whether a reset (INIT) was triggered by $\overline{\text{HST}}$ pin input.

HSTB	Meaning
0	No INIT has been triggered by the $\overline{\text{HST}}$ pin input.
1	INIT has been triggered by the $\overline{\text{HST}}$ pin input.

  The hardware standby reset occurred flag (HSTB) is cleared to “0” after reading.

- Bit5: Watchdog reset occurred flag

  Indicates whether a reset (INIT) was triggered by the watchdog timer.

WDOG	Meaning
0	No INIT has been triggered by the watchdog timer.
1	INIT has been triggered by the watchdog timer.

  The watchdog reset occurred flag (WDOG) is cleared to “0” after reading.

- Bit4: External reset occurred flag

  Indicates whether a reset (RST) was triggered by the $\overline{\text{RST}}$ pin input.

ERST	Meaning
0	No RST has been triggered by the $\overline{\text{RST}}$ pin input.
1	RST has been triggered by the $\overline{\text{RST}}$ pin input.

  The external reset occurred flag (ERST) is cleared to “0” after reading.

- Bit3: Software reset occurred flag

  Indicates whether a software reset has been triggered by writing to the software reset bit (STCR.SRST).

SRST	Meaning
0	No RST has been triggered by a software reset.

1	RST has been triggered by a software reset.

The software reset occurred flag (SRST) is cleared to “0” after reading.

- Bit2: Low voltage reset occurred flag

Indicates whether a reset (INIT) was triggered by the low voltage detection.

LINIT	Meaning
0	No INIT has been triggered by the low voltage detection.
1	INIT has been triggered by the low voltage detection.

The low voltage reset occurred flag (LINIT) is cleared to “0” after reading.

- Bit1-0: Watchdog interval time selection

WT1	WT0	The minimum writing interval required for WPR so that the watchdog timer may not be reset	Interval between the time when WPR is last written with $5A_H$ and when the watchdog is reset
		(Interval time of the timebase counter selection bit)	(Watchdog interval time)
0	0	$\Phi \times 2^{20}$ (Initial value)	$\Phi \times 2^{20}$ to $\Phi \times 2^{21}$
0	1	$\Phi \times 2^{22}$	$\Phi \times 2^{22}$ to $\Phi \times 2^{23}$
1	0	$\Phi \times 2^{24}$	$\Phi \times 2^{24}$ to $\Phi \times 2^{25}$
1	1	$\Phi \times 2^{26}$	$\Phi \times 2^{26}$ to $\Phi \times 2^{27}$

(Φ: Base clock)

- A total of four watchdog interval times are available to be selected.
- Only the data firstly written after a reset is valid, and the other data sets are invalid.
- Watchdog interval time selection bit can be read to know the set value.

Note: For more information on bits used for timers other than the watchdog timer, refer to “Chapter 9 Reset (Page No.139)”.

4.2 WPR: Watchdog Reset Generation Postponement Register

This register is used to postpone the generation of watchdog reset.

- **WPR: Address 0485h (Access: Byte)**

7	6	5	4	3	2	1	0	bit
D7	D6	D5	D4	D3	D2	D1	D0	
X	X	X	X	X	X	X	X	Initial value (INIT)
X	X	X	X	X	X	X	X	Initial value (RST)
RX,W	RX,W	RX,W	RX,W	RX,W	RX,W	RX,W	RX,W	Attribute

(Refer to "Meaning of Bit Attribute Symbols (Page No.10)" for the attributes.)

- If "A5$_H$" and "5A$_H$" are successively written in the watchdog reset generation postponement register and immediately after writing "5A$_H$" the 1-bit counter used to detect the watchdog is set to "0" to postpone the generation of a watchdog reset.
  Although there are no restrictions on the write timings for "A5$_H$" and "5A$_H$", if "A5$_H$" and a value other than "5A$_H$" are written, "A5$_H$" must be written again. If not, writing "5A$_H$" does not set the 1-bit counter to "0".
- The read value is indefinite.
- Both "A5$_H$" and "5A$_H$" must be written within the specified interval as shown below to prevent the watchdog reset from being generated. The intervals are shown in the following table according to the watchdog interval time selection bit (RSRR.WT[1:0]).

WT1	WT0	Minimum interval required for writing data in WPR
0	0	Within $\Phi \times 2^{20}$ (Initial value)
0	1	Within $\Phi \times 2^{22}$
1	0	Within $\Phi \times 2^{24}$
1	1	Within $\Phi \times 2^{26}$

4.3 CTBR: Timebase Counter Clear Register

This register is used to initialize the timebase counter.

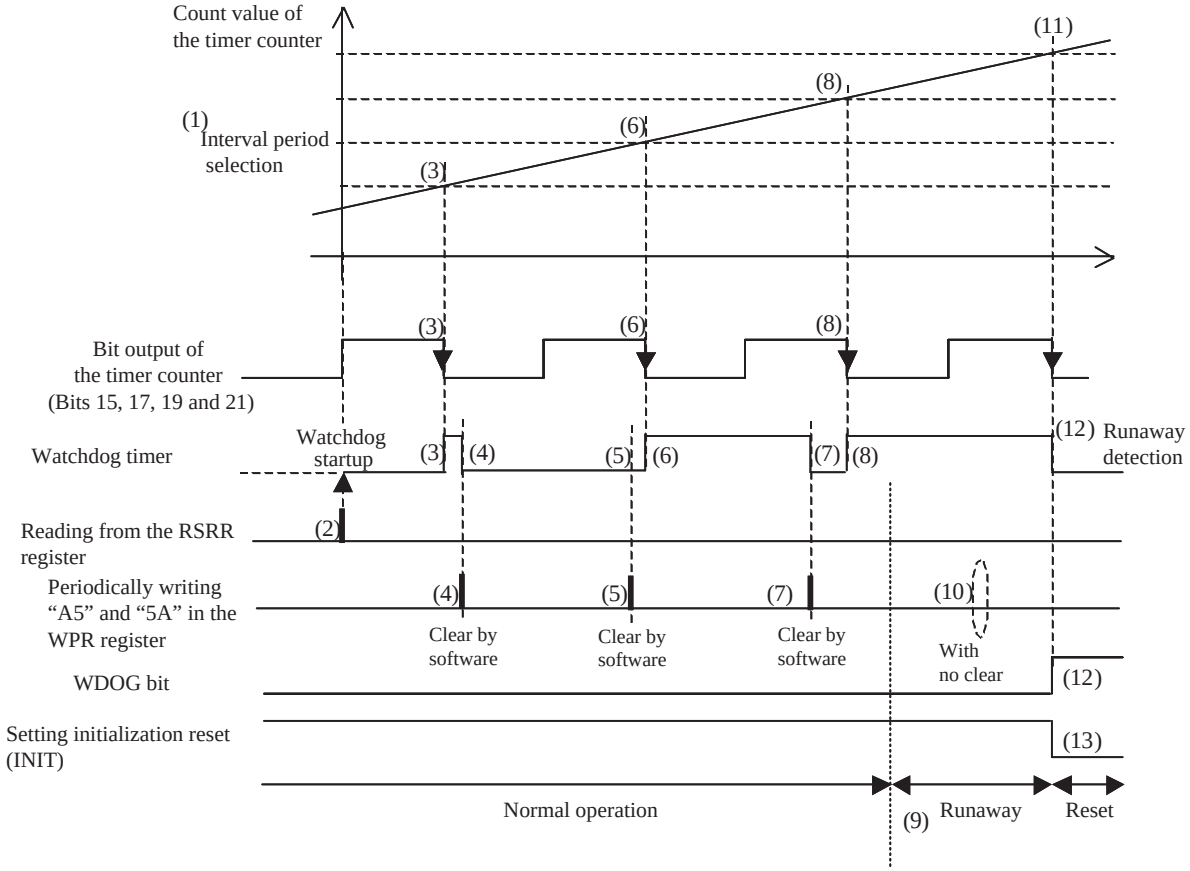
- **CTBR: Address 0483h (Access: Byte)**

For more information, refer to "Chapter 19 Timebase Timer (Page No.265)".

5. Operation

This section describes the watchdog operation.

5.1 Watchdog (Detecting Runaway)



(1) Setting interval time
(2) Watchdog startup (Watchdog timer clear)
(3) Interval signal output from the timebase counter. The watchdog timer counts.
(4) Within the interval time, by software periodic writing to the WPR register with "A5" and "5A" has been performed. The watchdog timer clears.
(5) Within the interval time, by software periodic writing to the WPR register with "A5" and "5A" has been performed. The watchdog timer clears.
(6) Interval signal output from the timebase counter. The watchdog timer counts.
(7) Within the interval time, by software periodic writing to the WPR register with "A5" and "5A" has been performed. Watchdog timer clears.
(8) Interval signal output from the timebase counter. The watchdog timer counts.
(9) MCU runs away (the runaway of MCU is assumed).
(10) Within interval time, by software writing to the WPR register with "A5" and "5A" has not been performed.
(11) Interval signal output from the timebase counter. The watchdog timer counts.
(12) Runaway is detected, WODOG flag has been changed to "1".
(13) Watchdog reset (INIT) has been generated.

5.2 Starting the Watchdog Timer and Setting the Watchdog Timer Period

The watchdog timer starts once it first writes data to the RSRR (Reset cause register/Watchdog timer control register) after the reset (RST). At this time, Bits 1 and 0 (WT1 and WT0 bits) set the watchdog timer interval time. Only the setting for the interval time executed first after the reset is valid, and the other settings executed at a later time are invalid.

5.3 Postponing the Generation of a Watchdog Reset

Once watchdog timer is started, it is necessary that the WPR (watchdog reset generation postponement register) should be written periodically with $\{A5_H\}$ and $\{5A_H\}$ in this order by software. This operation is used to set the 1-bit counter for detecting the watchdog reset to “0”.

5.4 Confirming that the Watchdog Reset has been Generated

The 1-bit counter for detecting the watchdog reset is set at the falling edge of the output of the timebase counter where an interval is set. In addition, if the second falling edge is detected while the 1-bit counter is set, the request for the setting initialization reset (INIT) is generated as the watchdog reset.

5.5 Temporarily Stopped Watchdog Timer (Automatic Generation Postponement)

The watchdog timer resets the 1-bit counter used for detecting the watchdog reset to “0” as initialization while CPU program operation is stopped. In this state, the generation of the watchdog reset is postponed. The states where programs stop running are concretely shown below.

- Sleep
- Stop
- Oscillation stability wait RUN
- Is in break when using the emulator debugger and monitor debugger, only if DSU4 is mounted
- Is in break when using the embedded debug support unit (only if EDSU and EMMODE is enabled)
- Period between the time when executing the INTE command and when executing RETI, only if DSU4 is mounted
- Step trace trap (break per each command with PS register T flag=“1”), only if DSU4 is mounted

In addition, clearing the timebase counter simultaneously initializes the 1-bit counter used for detecting the watchdog reset, thus causing the reset timing of the watchdog to be postponed.

5.6 Stopping the Watchdog Timer

Once the watchdog timer is started, the watchdog timer operation cannot be stopped until the initialization reset (RST) is generated.

The watchdog timer is stopped under these states shown below where the operation initialization reset (RST) is generated until it is restarted by software.

- Operation initialization reset (RST)
- Setting initialization reset (INIT)
- Oscillation stability wait reset

6. Setting

Table 6-1 Setting Required for Using the Watchdog Timer

Setting	Setting register	Setting method
Interval time setting	Watchdog timer control register (RSRR)	Refer to 7.1
Startup of the watchdog		Refer to 7.2

*: Refer to the number for more information on the setting method.

Table 6-2 Setting Required for Delaying the Generation of the Watchdog

Setting	Setting register	Setting method
Setting required for delay the generation of the watchdog reset	Watchdog reset generation delay register (WPR)	Refer to 7.3

*: Refer to the number for more information on the setting method.

Table 6-3 Setting Required for Checking the Generation of the Watchdog

Setting	Setting register	Setting method
Watchdog generation check	Watchdog timer control register (RSRR)	Refer to 7.5

*: Refer to the number for more information on the setting method.

7. Q & A

7.1 What are the types of watchdog interval time and how are they selected?

There are four types of the interval period, and they are set using the interval selection bit (RSRR.WT[1:0]).

Watchdog Interval time	Interval Selection bit (WT[1:0])	Example) Interval Time		
		FΦ =80.0MHz	FΦ = 2.00MHz	FΦ = 32.768kHz
To select $\Phi \times 2^{20}$	Set the value to "00"	13.1 ms	0.524 s	32.0 s
To select $\Phi \times 2^{22}$	Set the value to "01"	52.4 ms	2.097 s	128.0 s
To select $\Phi \times 2^{24}$	Set the value to "10"	209.7 ms	8.388 s	512.0 s
To select $\Phi \times 2^{26}$	Set the value to "11"	838.8 ms	33.554 s	2048.0 s

Note: F: Base clock. (Refer to "Chapter 13 Clock Control (Page No.189)".)
Only the data sets first written after the reset ($\overline{\text{INIT}}$ pin input, watchdog reset, software reset) are valid, and the other data sets are invalid.

7.2 How is the watchdog operation started (set to valid)?

Writing data in the watchdog timer control register RSRR causes the watchdog timer to be started (set to valid). Writing data in the interval selection bit (RSRR.WT[1:0]) causes the watchdog to be started.

7.3 How can we check that the watchdog reset has been generated?

If the watchdog reset flag (RSRR.WDOG) is set to "1", the watchdog reset has been generated.

7.4 How is the watchdog stopped?

The watchdog cannot be stopped by the software.

The watchdog can be stopped only with the reset ($\overline{\text{INIT}}$ pin input, watchdog reset).

7.5 How do I clear the watchdog timer (1-bit counter)?

Successively writing "$A5_H$" and "$5A_H$" in the watchdog reset generation postponement register WPR causes the 1-bit counter used for detecting the watchdog to be cleared immediately after writing "$5A_H$". In this state, the reset timing of the watchdog can be postponed.

In addition, if the timebase timer is cleared, the 1-bit counter used for detecting the watchdog is simultaneously reset.

8. Caution

- Although the watchdog interval time corresponds to the one twice as long as the watchdog 1-bit counter, the watchdog timer clear operation only clears the 1-bit counter used for detecting the watchdog. As a result, the time margin to clear the watchdog timer is different from the interval time.

  Watchdog interval time selection

WT1-WT0	Time margin to clear the watchdog timer	Interval time during which the watchdog reset is generated
00	$\Phi \times 2^{20}$ (Initial value)	$\Phi \times 2^{20}$ to $\Phi \times 2^{21}$
01	$\Phi \times 2^{22}$	$\Phi \times 2^{22}$ to $\Phi \times 2^{23}$
10	$\Phi \times 2^{24}$	$\Phi \times 2^{24}$ to $\Phi \times 2^{25}$
11	$\Phi \times 2^{26}$	$\Phi \times 2^{26}$ to $\Phi \times 2^{27}$

- The watchdog timer is started once data is written in the watchdog timer control register.
- The watchdog timer control register is also the reset cause register and the status (INIT, HSTB, WDOG, ERST, SRST and LINIT) is set to “0” when it is read.
- The watchdog reset holds the oscillation stability wait time.
  (Refer to “Chapter 18 Timebase Counter (Page No.249)”.)
- The watchdog reset from the main RUN or the sub-RUN where the main clock oscillation is in process cannot have the oscillation stability wait time because the main clock is oscillating.
- Refer to “Chapter 19 Timebase Timer (Page No.265)” for the method of clearing the timebase counter that is the count source for the watchdog timer.
- Clearing the timebase counter causes the watchdog reset timing to be postponed once.
- MB91V460A and MB91461: The watchdog timer will be retriggered / cleared automatically by DMA during D-bus access.

8.Caution

Chapter 21 Hardware Watchdog Timer

1. Overview

The hardware watchdog timer (R/C oscillation based) provides a system reset if an internal watchdog timer is not cleared within the postponement duration.

● Hardware watchdog timer

This watchdog timer starts counting after the setting initialization reset (INIT) automatically. Clearing the counter in the postponement duration is necessary to continue running an application. Otherwise if the counter is not cleared within the postponement duration, e.g. due to infinite loop in the application, this module provides a reset signal (initialisation reset, INIT), which width is typical 20us (2 RC clock cycles at typical 100kHz).

If the CPU is in a standby mode as described below, this watchdog timer stops:

• SLEEP mode: the CPU stops, the peripherals run.
• STOP mode: the CPU and the peripherals stop.
• RTC mode: the CPU and the peripherals stop but the RTC module and the oscillator run.

If one of the below condition occurs, the watchdog counter is cleared:

• Writing “0” to CL bit in the HWWD register
• Initialisation Reset (INIT)
• Operational Reset (RST)
• Oscillation stops
• Transition to the SLEEP/RTC/STOP mode

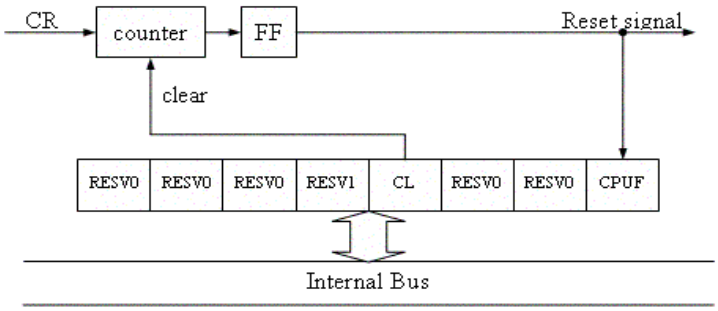
2. Configuration

Hardware watchdog timer consists of two sub-blocks:

- Watchdog timer
- Timer control and status register

● Block diagram of the hardware watchdog timer

Figure 2-1 Block Diagram of hardware watchdog timer



Watchdog timer

This is a timer to supervise CPU operation. The counter needs to be cleared periodically after releasing the reset.

Hardware watchdog timer control status register

This register has the reset flag and clear bit for the counter.

Occuring of the watchdog reset

If the counter has not been cleared periodically, this module provides a setting initialization reset (INIT). The width of internal reset signal is 63 times the system base clock. After the watchdog reset the normal system reset procedure starts. For more details about this procedure, see the corresponding section in the device state description.

3. Register

3.1 Hardware watchdog timer control and status register

Hardware watchdog timer control status register (with reset flag and clear bit).

- **HWWD: Address 04C7h (Access: Byte)**

7	6	5	4	3	2	1	0	bit
RESV0	RESV0	RESV0	RESV1	CL	RESV0	RESV0	CPUF	
0	0	0	1	1	0	0	0	Initial value (INIT pin input, watchdog reset)
0	0	0	1	1	0	0	X	Initial value (Software reset)
R/W0	R/W0	R/W0	R/W1	W	R/W0	R/W0	R/W	Attribute

(See “Meaning of Bit Attribute Symbols (Page No.10)” for details of the attributes.)

- Bit7-5: Reserved bits. Always write “0” to these bits.
- Bit4: Reserved bit. Always write “1” to this bit.
- Bit3: CL (counter clear).

CL	Function
0	By writing ‘0’ the watchdog timer is cleared
1	Writing ‘1’ has no effect

This bit is write only, it is always read as ‘1’.

- Bit2-1: Reserved bits. Always write “0” to these bits.
- Bit0: CPUF (CPU reset Flag).

CPUF	Function
0	Watchdog reset not triggered
1	Watchdog reset triggered (overflow of watchdog timer occured)

This bit is initialized by external reset input (INITX) or clock supervisor reset, but not by internal reset. Writing ‘0’ clears this bit, writing ‘1’ has no effect.

3.2 Hardware watchdog timer duration register

Hardware watchdog timer duration register (elongation of the trigger duration).

- **HWWDE: Address 04C6h (Access: Byte)**

7	6	5	4	3	2	1	0	bit
-	-	-	-	-	-	ED1	ED0	
-	-	-	-	-	-	0	0	Initial value (INIT pin input, watchdog reset)
-	-	-	-	-	-	0	0	Initial value (Software reset)
RX/W0	RX/W0	RX/W0	RX/W0	RX/W0	RX/W0	R/W	R/W	Attribute

(See “Meaning of Bit Attribute Symbols (Page No.10)” for details of the attributes.)

- Bit7-2: Reserved bits. Always write “0” to these bits.
- Bit1-0: ED (Elongate watchdog duration).

ED1-0	Function
00	The watchdog period is 2^16 RC clock cycles [initial setting]
01	The watchdog period is 2^17 RC clock cycles *)
10	The watchdog period is 2^18 RC clock cycles *)
11	The watchdog period is 2^19 RC clock cycles *)

*) This setting is not available on MB91V460.

4. Functions

If the watchdog timer is not cleared periodically, a setting initialization reset (INIT) occurs. In this case the value of registers in CPU is not guaranteed.

● Function of the hardware watchdog timer

After releasing INITX the hardware watchdog timer starts immediately without stabilization time. If the timer is not cleared periodically, setting initialization (INIT) reset occurs.

● Period of the hardware watchdog timer

The timer width is 16-bit. Since the RC oscillator is used as clock source of the hardware watchdog timer, the duration of the timer deviates with the RC oscillator accuracy:

	ED1-0	Min.	Typ.	Max.
RC oscillation cycle (u s)		5	10	20
Watchdog term (ms)	00	327.68	655.36	1310.72
	01	655.36	1310.72	2621.44
	10	1310.72	2621.44	5242.88
	11	2621.44	5242.88	10485.76

Note: The watchdog duration elongation (ED1-0) setting is not available on MB91V460. In that case ED1-0 is always “00”.

5. Caution

- **Software disabling is not possible**

The watchdog timer starts counting immediately after reset (release of INITX). Software cannot stop the counting.

- **Hardware disabling is only possible on the evaluation device MB91V460**

The watchdog timer can be permanently disabled by setting the corresponding jumper of the evaluation board (this is not possible on flash devices with this watchdog timer). So always ensure correct configuration of the evalution system to reflect the behaviour of the flash device.

- Postponement of reset

In order to postpone the watchdog reset, the clearing of the watchdog timer is necessary. Whenever the CL bit of register is set to ‘0’ (there is no minimum writing limitation), the timer is cleared and the occurrance of reset is postponed. Just writing to the register without setting CL to ‘0’ does not clear the timer.

- Timer stop and clear

In modes where the CPU does not work (SLEEP, STOP or RTC mode), the timer is cleared first then the counting is stopped.

- During DMA transfer

During DMA transfer between D-bus modules, the writing ‘0’ to CL bit is not possible. Thus, if the transfer time is more than 328ms (calculated from the fastest frequency of the RC oscillator as minimum period), a reset occurs.

- Duration setting

Unlike on MB91V460 Rev.A it is possible on flash devices to elongate the duration of the watchdog reset.

- RC clock frequency

Unlike on MB91V460 Rev.A it is possible on flash devices to change the RC clock frequency to 2MHz. Even though the watchdog timer is always operated with a frequency of 100kHz (10us) typical.

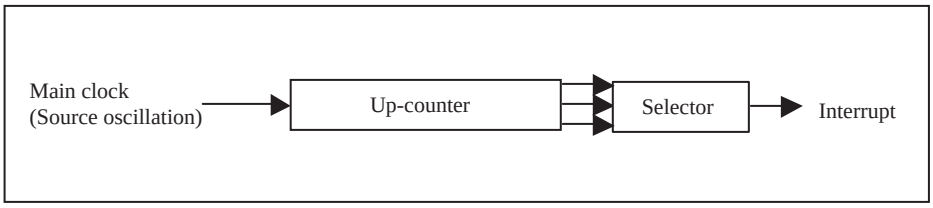
Chapter 22 Main Oscillation Stabilisation Timer

1. Overview

The main clock oscillation stabilisation timer is a 23-bit counter that counts the main clock. This timer does not affect the selection of clock source operated by MCU/dividing setting.

This timer is mainly used for acquiring main clock oscillation stability wait time to resume main clock oscillation after the main clock oscillation has been stopped (OSCCR.OSCDS1=1) while the subclock is being operated.

In addition, this timer is best suited for interval timers or system clocks for real time OS.



2. Features

- Type : 23-bit free run counter
- Quantity : 1
- Clock source : Main clock (source oscillation) --- Period = $1/F_{CL-MAIN}$
- Interval time : 3 types
  Period = $2^{12}/F_{CL-MAIN}$, $2^{17}/F_{CL-MAIN}$, $2^{23}/F_{CL-MAIN}$,
  (1.0ms, 32.7ms, 2s / main clock 4MHz)
- Cause of timer clear: (Software, overflow, reset (INIT))
- Operation start/stop : Can be operated/stopped by the software.
- Interrupt : Main clock oscillation stability wait interrupt (Interval interrupt)
- Count value : Cannot read/write. (Clear only)

3. Configuration

Figure 3-1 Configuration Diagram

Main clock oscillation stability wait timer

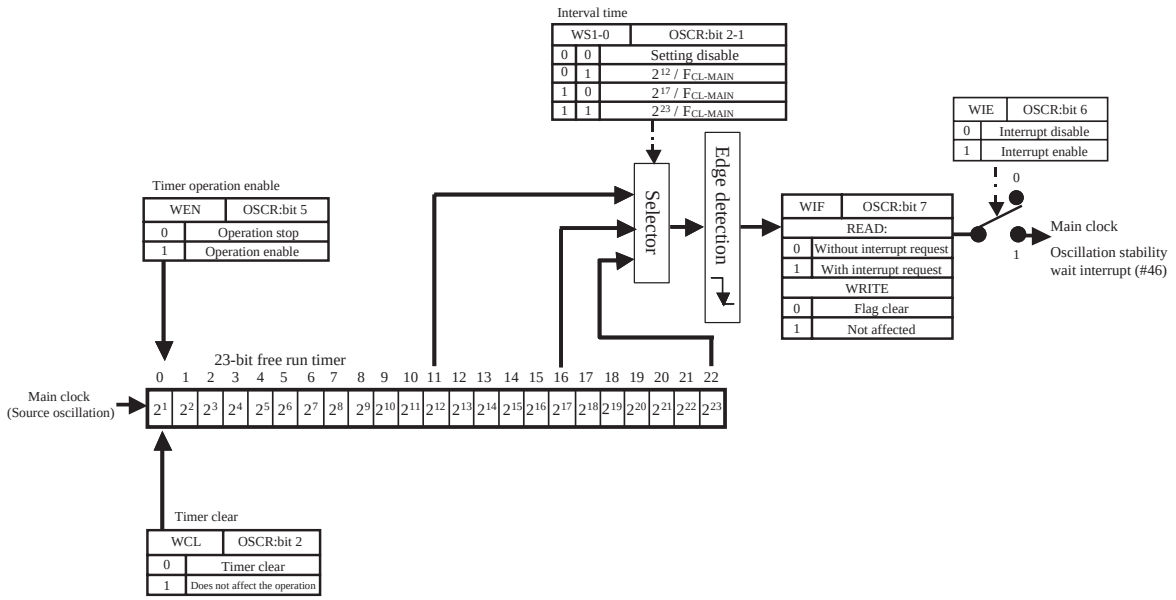
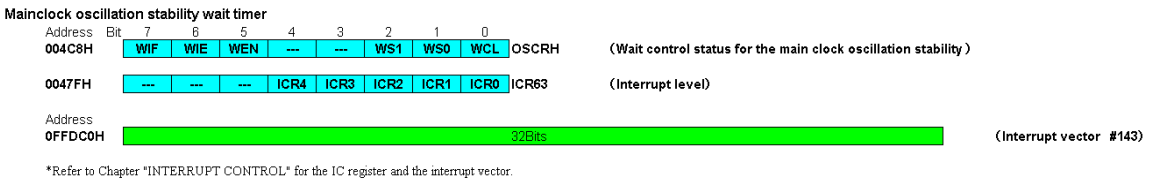


Figure 3-2 List of Registers



Note: Refer to “Chapter 24 Interrupt Control (Page No.315)” for the ICR register and the interrupt vector.

4. Register

4.1 OSCRH: Control Register for the Main Clock Oscillation Stability Wait Timer

This register is used to select the interval time, clear the timer, control the interrupt, control the timer such as stop, and confirm the state of the timer.

- **OSCRH: Address 04C8h (Access: Byte)**

7	6	5	4	3	2	1	0	bit
WIF	WIE	WEN	–	–	WS1	WS0	WCL	
0	0	0	X	X	0	0	1	Initial value (INIT terminal input, watchdog reset)
X	X	X	X	X	X	X	X	Initial value (Software reset)
R(RM1),W	R/W	R/W	RX/W0	RX/W0	R/W	R/W	R1,W	Attribute

(For the attributes, refer to "Meaning of Bit Attribute Symbols (Page No.10)".)

- Bit7: Timer interrupt request flag

WIF	Read Operation	Write Operation
0	Without interrupt request	Clears interrupt request flags
1	With interrupt request	Writing does not affect operation

  - The timer interrupt request flag bit is set to "1" at the falling edge of the selected interval period output.

- Bit6: Interrupt request enable

WIE	Operation
0	Interrupt request disable
1	Interrupt request enable

  - If the timer interrupt request flag (WIF) is set to "1" while the interrupt request enable (WIE) is "1" an interrupt request is immediately generated.

- Bit5: Timer operation enable

WEN	Operation
0	Stops timer operation
1	Enables timer operation

- Bit4-3: Reserved bit Be sure to write "0". The read value is "0".
- Bit2-1: Interval period selection

WS1	WS0	Interval period (At 4MHz)
0	0	Setting prohibted
0	1	$2^{12}/F_{CL\text{-}MAIN}$ (1.0ms)
1	0	$2^{17}/F_{CL\text{-}MAIN}$ (32.7ms)
1	1	$2^{23}/F_{CL\text{-}MAIN}$ (2.0s)

  - The reset does not initialize. Be sure to set it after the startup.

- Bit0: Timer clear

WCL	Operation
0	Clears the main clock oscillation stability wait timer
1	Writing does not affect operation

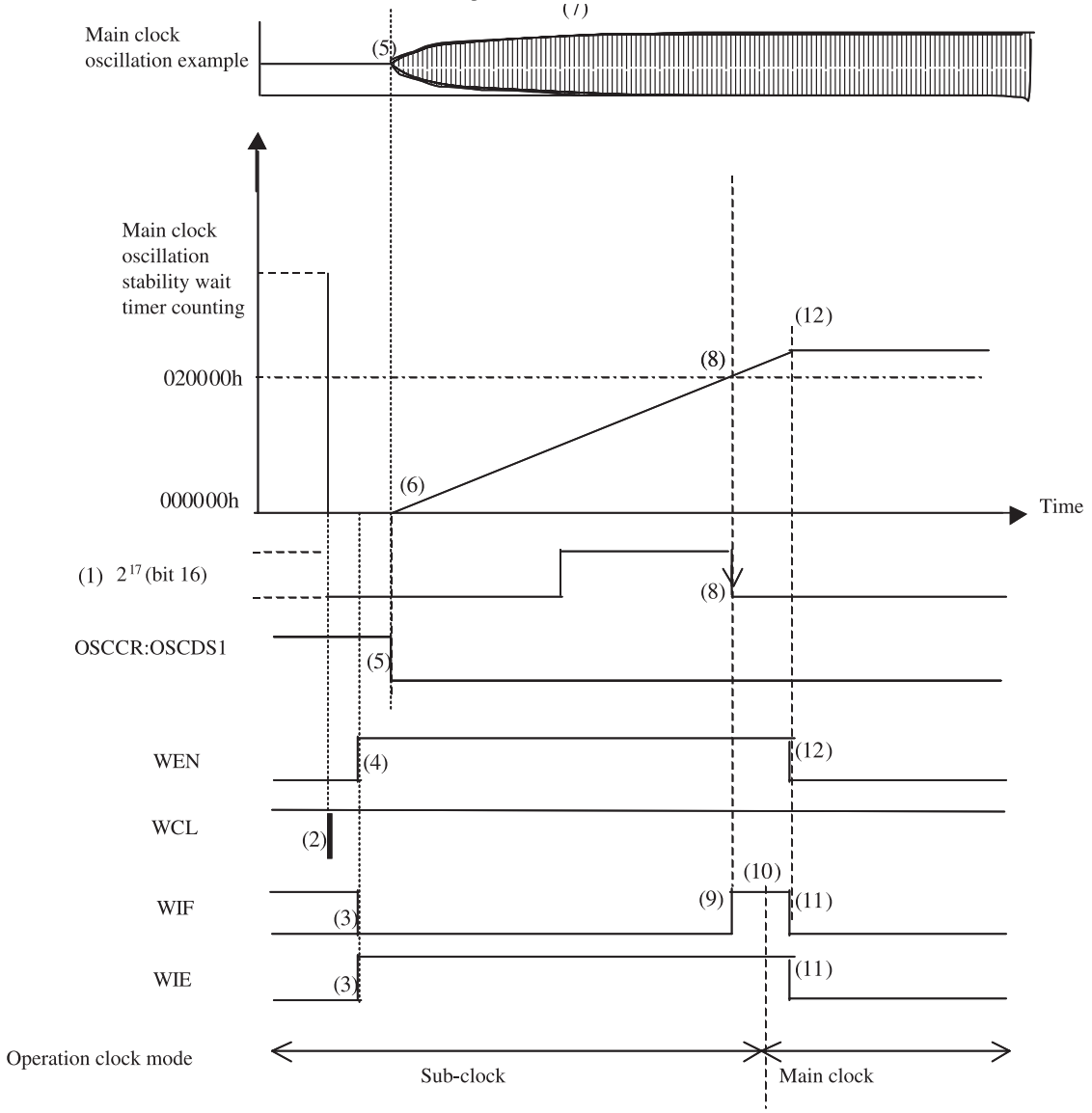
  - The timer is also cleared by INITX terminal input and watchdog reset.

(Refer to "8. Caution (Page No.301)".)

5. Operation

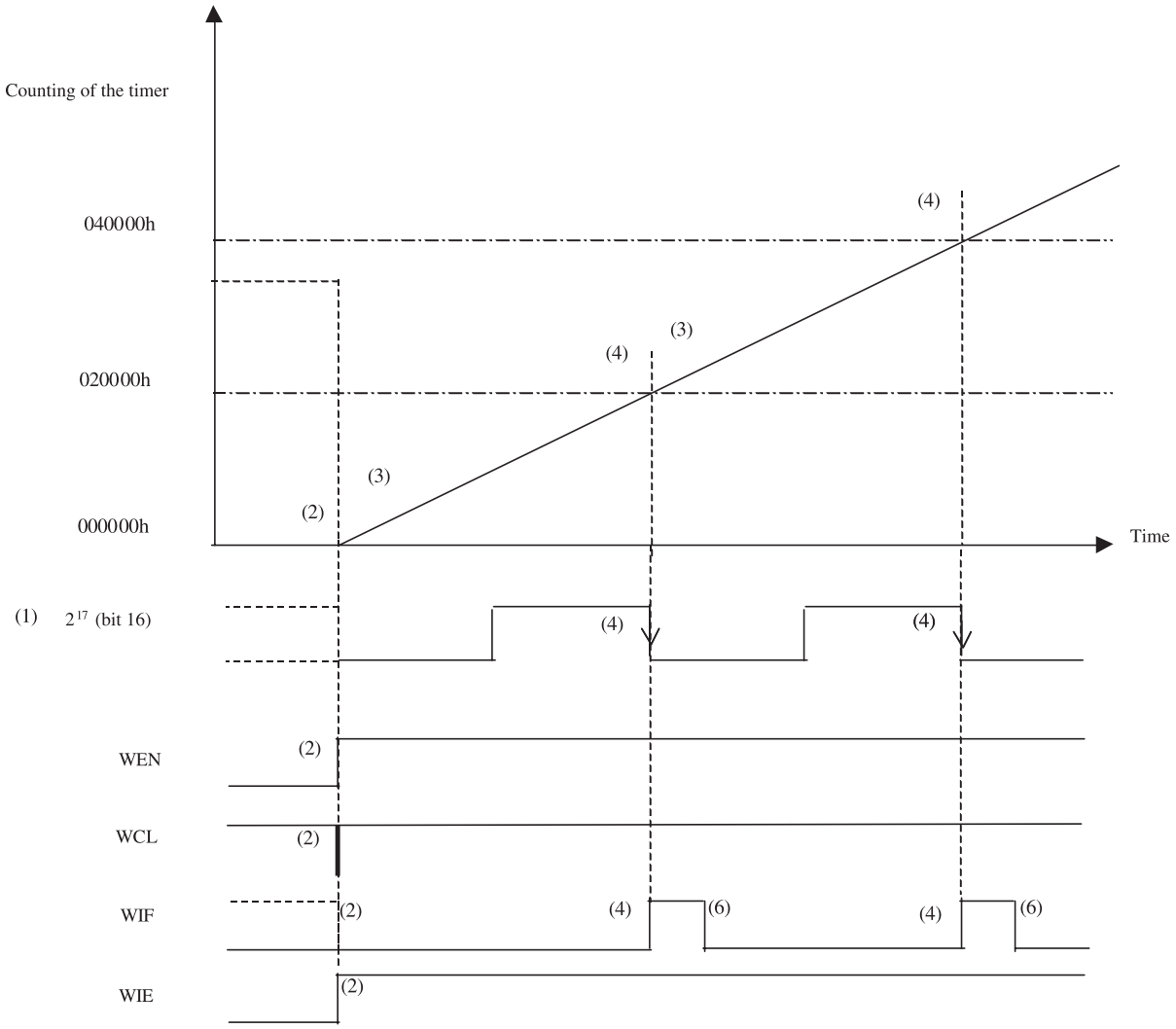
This section describes the main clock oscillation stability wait timer operation.

5.1 Main Clock Oscillation Stability Wait



(1) Selects the interval time. (WS[1:0]) (In this example, $2^{17}/F_{CL\text{-}MAIC}$ is selected.)
(2) Sets timer clear (WCL="0") by the software.
(3) Sets flag clear (WIF="0") and interrupt request enable (WIE="1") by the software.
(4) Sets timer count enable (WEN="1") by the software.
(5) Releases main clock stop (OSCCR.OSCDS1="0") while the subclock is in operation by the software, and starts main clock oscillation.
(6) Starts counting (The timer counts up using the main clock (source oscillation).)
(7) Stabilizes the main clock oscillation.
(8) The selected interval time is used. (Detects the falling edge of the dividing 2^{17}.)
(9) Generates a main clock oscillation stability wait interrupt.
(10) Processing caused by an interrupt (Software): Operation clock switching (Sub-RUN => main RUN)
(11) Disables interrupt request (WIE="0") and clears interrupt request (WIF="0").
(12) Stops counting (WEN="0").

5.2 Interval Interrupt



(1) Selects the interval time (WS[1:0]). (In this example, $2^{17}/F_{CL\text{-}MAIC}$ is selected.)

(2) Clears the timer (WCL="0"), clears flags (WIF="0"), enables interrupt request (WIE="1"), enables timer count (WEN="1") by the software.

(3) The timer counts up using the main clock (source oscillation).

(4) Generates interval interrupt at the selected interval time (Falling of the dividing 2^{17}).

(5) Processing caused by an interrupt (Software): Clears interrupt request (WIF="0").

(6) Repeats Items (3) to (5)

6. Setting

Figure 6-1 Settings Required for Using the Main Clock Oscillation Stability Wait Timer

Setting	Setting register	Setting method*
Setting interval time	Main clock oscillation stability wait timer control register (OSCRH)	7.1
Count clear		7.4
Counting operation start		7.3

*: Refer to the number for more information on the setting method.

Figure 6-2 Settings Required for Enabling the Main Clock Oscillation Stability Wait Timer Interrupt

Setting	Setting register	Setting method*
Sets the main clock oscillation stability wait timer interrupt vector and Sets free run timer interrupt level	Refer to "Chapter 24 Interrupt Control (Page No.315)".	7.5
Sets the main clock oscillation stability wait timer interrupt Clears interrupt request Enables interrupt request	The main clock oscillation stability wait timer control register (OSCRH)	7.7

*: Refer to the number for more information on the setting method.

Figure 6-3 Settings Required for Stopping the Main Clock Oscillation Stability Wait Timer

Setting	Setting register	Setting method*
Sets the main clock oscillation stability wait timer stop	The main clock oscillation stability wait timer control register (OSCRH)	7.8

*: Refer to the number for more information on the setting method.

7. Q & A

7.1 What are the types of interval time (wait time) and how are they selected?

There are 3 types of interval time, and they are set with the interval selection bit (OSCRH.WS[0:1]).

Interval time	Count period	Interval (Wait time) Example
	Interval selection bit (WS[1:0])	At F_{CLKP} = 4.00MHz
To set the value to $2^{12}/F_{CL-MAIN}$	Set the value to “01”	1.00ms
To set the value to $2^{17}/F_{CL-MAIN}$	Set the value to “10”	32.7ms
To set the value to$2^{23}/F_{CL-MAIN}$	Set the value to “11”	2.00 s

Note: Setting (WS[1:0]=“00”) is prohibited.

7.2 How do I select the count clock?

The count clock is the main clock (source oscillation). (Cannot be selected.)

7.3 How is the main clock oscillation stability wait timer count operation enabled/ disabled?

Sets with the timer operation enable bit (OSCRH.WEN).

Operation	Timer operation enable bit (WEN)
To stop the main clock oscillation stability wait timer	Set the value to “0”
To start the main clock oscillation stability wait timer	Set the value to “1”

7.4 How is the main clock oscillation stability wait timer cleared?

The following methods are used to clear the main clock oscillation stability wait timer.

- Sets with the clear bit (OSCRH.WCL).

Operation	Clear bit (WCL)
To clear the main clock oscillation stability wait timer	Writes “1”

- Performs a reset.

  Clears the free run timer with the operation initialization reset (INIT terminal input, watchdog reset).

  (Value is held without being cleared even if a software reset is performed.)
- The overflow (Next of “FFFFFFh”) of the main clock oscillation stability wait timer causes the count value to be reset to “000000_H”.

7.5 What happens with the interrupt-associated registers?

Setting the interrupt vector and interrupt level of the main clock oscillation stability wait timer

The relationship between the interrupt level and the interrupt vector is shown in the following table.

Refer to “Chapter 24 Interrupt Control (Page No.315)” for the interrupt level and interrupt vector.

Interrupt vector (Default)	Interrupt level setting bit (ICR4-ICR0)
#143 Address: 0FFDC0h	Interrupt level register (ICR63) Address: 047Fh

As the interrupt request flag (OSCRH.WIF) is not automatically cleared, clear it before returning to interrupt processing by the software. (Writes “0” in the WIF bit.)

7.6 What are the types of interrupt?

There is one type of interrupt called the main clock oscillation stability wait timer interrupt.
(Selection is unnecessary.)

7.7 how is an interrupt enabled?

Interrupt request enable and interrupt request flag
Setting the interrupt enable is performed with the interrupt request enable bit (OSCRH.WIE).

	Interrupt request enable bit (WIE)
Interrupt disable	Set the value to “0”
Interrupt enable	Set the value to “1”

Clearing an interrupt request is performed with the interrupt request bit (OSCRH.WIF).

	Interrupt request bit (WIF)
Interrupt request clear	Writes “0”

7.8 How is the main clock oscillation stability wait timer stopped counting?

Sets with the timer operation enable bit (OSCRH.WEN). Refer to 7.3.

In addition, if the MCU stops the main clock while the subclock is being operated, the main clock oscillation stability wait timer also stops counting.

8. Caution

- To wait until the main clock oscillation stability is attained while the subclock is in operation, it is necessary to acquire wait time using the main clock oscillation stability wait timer.
  (An unstable clock may be supplied to the entire device, and normal operation is not guaranteed if the MCU operation mode is switched from the sub-RUN to the main RUN mode without waiting until the main clock oscillation becomes stable.)

- The value for the oscillation stability wait time is an estimated value because the oscillation period of the main clock oscillation is unstable for the beginning immediately after the oscillation has started.

- If the main clock oscillation stops, the main clock oscillation stability wait interrupt (interval interrupt) is not generated either because the main clock oscillation stability wait timer stops. The main clock oscillation should be enabled for processing that uses the main clock oscillation stability wait interrupt (interval interrupt).

- The flag is set to “1” (flag setting preference) if the timer interrupt request (WIF=“1”) and the writing operation where “0” is written by software in the flag occur simultaneously.

- The main clock oscillation stability wait timer is counted up with the main clock. As a result, in the following state, the counting of the timer used to stop the main clock oscillation also stops.
  - If the timer operation enable bit (OSCRH.WEN) is “0”, the timer stops counting.
  - If the main clock is stopped in the stop mode (STCR.OSCD1=“1”), the timer stops counting from the moment the stop mode is activated.
  - If the main clock oscillation is stopped (OSCCR.OSCDS1=“1”) during subclock operation, the timer stops while the subclock is in operation.

- If you want to enable (WIE=“1”) the interrupt request after the reset is released, and the interval time to be modified, be sure to simultaneously set the interrupt request flag (WIF) and the clear bit (WCL) to “0” beforehand.

- The timer interrupt request bit (WIF), timer interrupt request enable bit (WIE), timer enable bit (WEN) and timer clear bit (WCL) are initialized using the setting initialization reset (INIT terminal input, watchdog reset).

- Be sure to set the interval selection bit (WS[1:0]) after startup (after setting initialization reset) by the software.

- The main clock oscillation stability wait timer control register should be initialized (to set the initial value) only with the setting initialization reset (INIT terminal input, watchdog reset) because the software reset does not initialize the register and the current value is held.

- If the counter clear (WPCR.WCL=“0”) and the overflow for the selected bit occur simultaneously, the interrupt request flag (WIF) is not set to “1”.

8.Caution

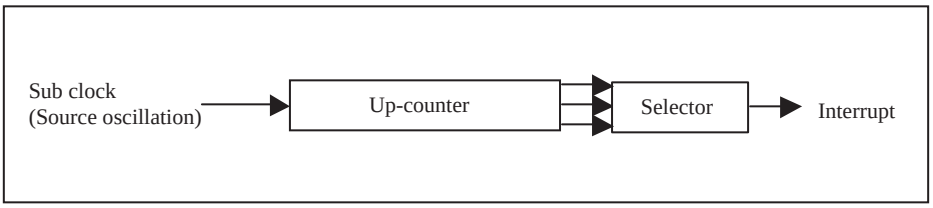
Chapter 23 Sub Oscillation Stabilisation Timer

1. Overview

The sub oscillation stabilisation timer is a 15-bit counter that is counted up with the subclock. This timer does not affect the selection/dividing setting of the MCU operating clock.

This timer is used to acquire subclock oscillation stability wait time if the subclock oscillation is resumed mainly when the subclock oscillation is stopped while the main clock is in operation.

This timer is can be used for acquiring sub clock oscillation stability wait time to resume sub clock oscillation after the sub clock oscillation has been stopped (OSCCR.OSCDS2=1) while the RC oscillator is being operated.



2. Features

- Type : 15-bit free run counter
- Quantity : 1
- Clock source : Subclock (source oscillation) --- Period = $1/F_{CL-SUB}$ = 1/32.768kHz
- Interval time : 4 types

  Period = $2^{10}/F_{CL-SUB}$, $2^{13}/F_{CL-SUB}$, $2^{14}/F_{CL-SUB}$, $2^{15}/F_{CL-SUB}$,

  (31.25ms, 0.25s, 0.50s, 1.00s)
- Timer clear cause: (Software, overflow, reset (INIT))
- Interrupt : clock interrupt (interval interrupt)
- Count value : Cannot read and write (Clear only)

3. Configuration

Figure 3-1 Configuration Diagram

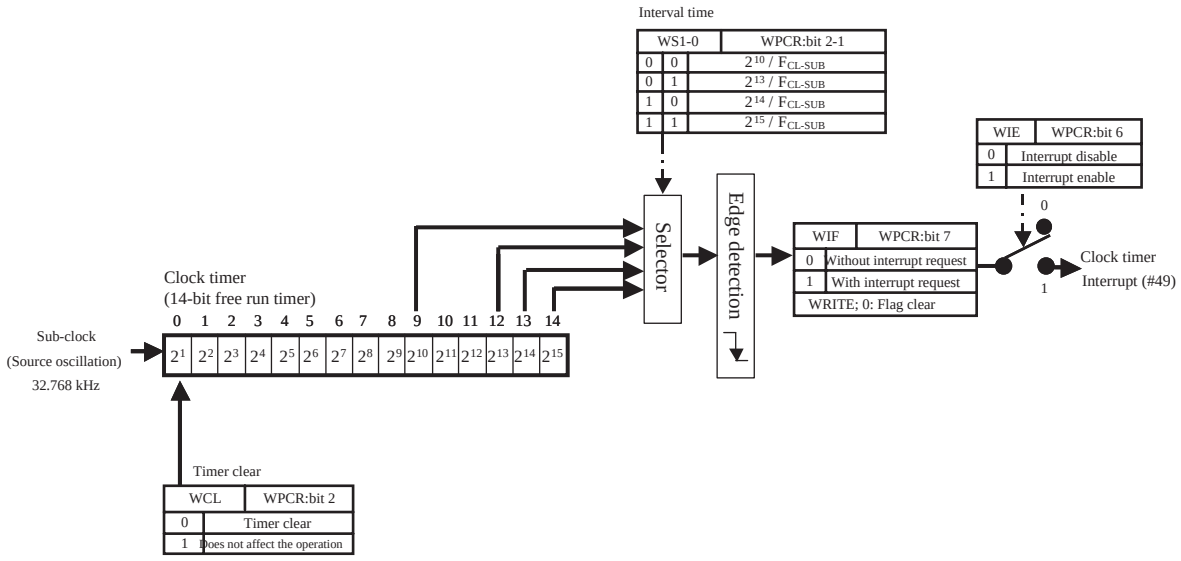
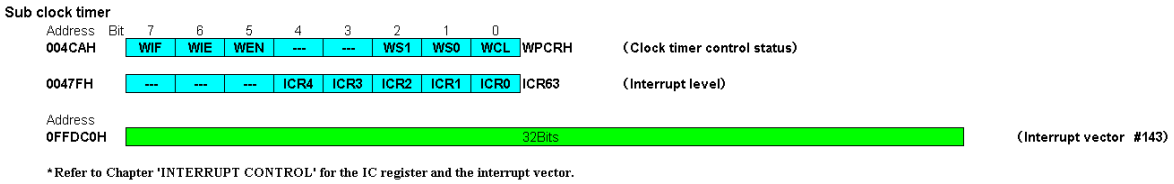


Figure 3-2 List of Registers



Note: For the ICR register and interrupt vector, refer to “Chapter 24 Interrupt Control (Page No.315)”.

4. Register

4.1 WPCRH: Sub oscillation stabilisation timer Control Register

This register is used to select interval time, clear the timer, control interrupt, control timer stop etc., and confirm the states.

- **WPCRH: Address 04CAh (Access: Byte)**

7	6	5	4	3	2	1	0	bit
WIF	WIE	WEN	–	–	WS1	WS0	WCL	
0	0	0	X	X	0	0	1	Initial value (At INIT) *1
R(RM1),W	R/W	R/W	RX/W0	RX/W0	R/W	R/W	R1,W	Attribute

(For the attributes, refer to "Meaning of Bit Attribute Symbols (Page No.10)".)

(Refer to "8. Caution (Page No.313)".)

- Bit7: Sub oscillation stabilisation timer interrupt request flag

WIF	Read Operation	Write Operation
0	Without interrupt request	Clears the interrupt request flag
1	With interrupt request	Writing does not affect operation

  - The sub oscillation stabilisation timer interrupt request flag bit is set to "1" at the falling edge of the selected interval period output.
- Bit6: Interrupt request enable

WIE	Operation
0	Interrupt request is prohibited
1	Interrupt request enable

  - If the interrupt request enable bit is set to "1" an interrupt request is enabled.
  - If the sub oscillation stabilisation timer interrupt request flag is (WIF="1"), and if the interrupt request enable bit (WIE) is set to "1", an interrupt request is immediately generated.
- Bit5: Timer operation enable

WEN	Operation
0	Stops timer operation
1	Enables timer operation

- Bit4-3: Reserved bit Be sure to write "0". The read value is "0".
- Bit2-1: Interval period selection

WS1	WS0	Interval period (F_{CL-SUB}= 32.768kHz)
0	0	$2^{10}/F_{CL-SUB}$ (31.25ms)
0	1	$2^{13}/F_{CL-SUB}$ (0.25s)
1	0	$2^{14}/F_{CL-SUB}$ (0.50s)
1	1	$2^{15}/F_{CL-SUB}$ (1.00s)

- Bit0: Timer clear

WCL	Operation
0	Clears the sub oscillation stabilisation timer.
1	Writing does not affect write operation.

  - The timer is also cleared by INITX terminal input and watchdog reset.

Notes 1: Initial value can be set using the setting initialization reset (INIT terminal input,

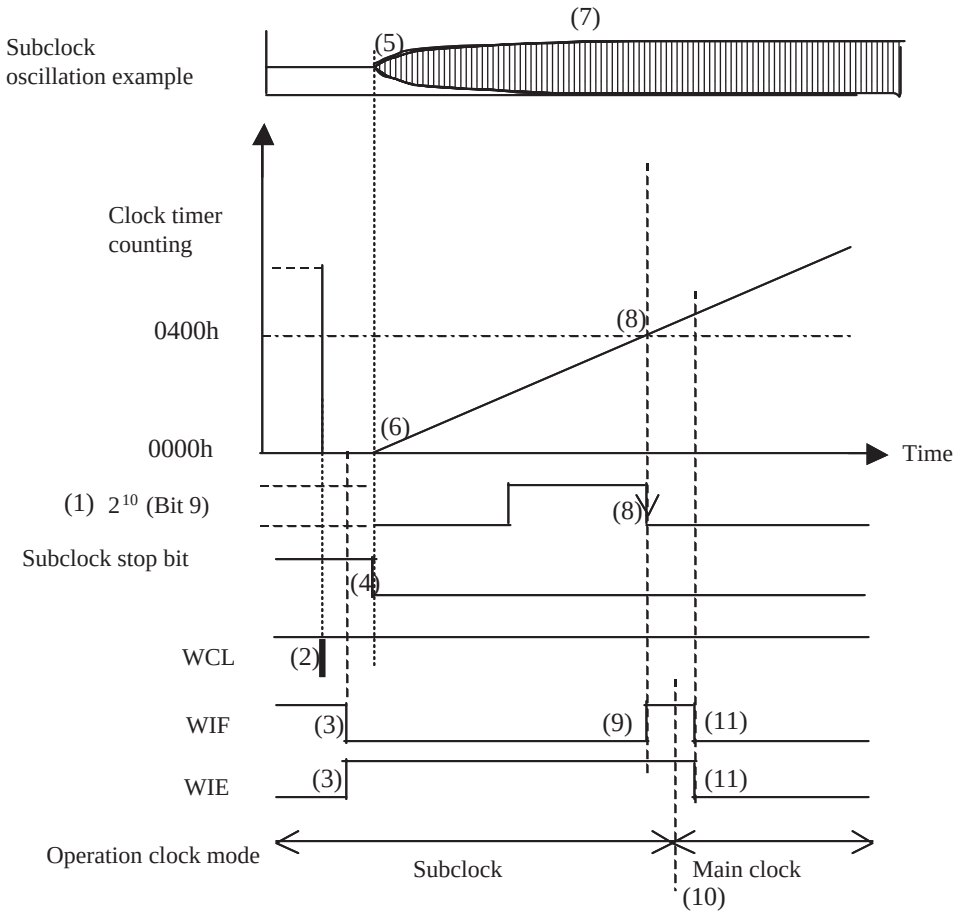
watchdog reset), but the operation initialization reset (Software reset) holds the current value instead of initializing it.

2: If you set the interrupt request enable (WIE="1"), and the interval period selection (WS[1:0]) after canceling the reset, be sure to simultaneously set the timer interrupt request flag (WIF) and the timer clear (WCL) "0".

5. Operation

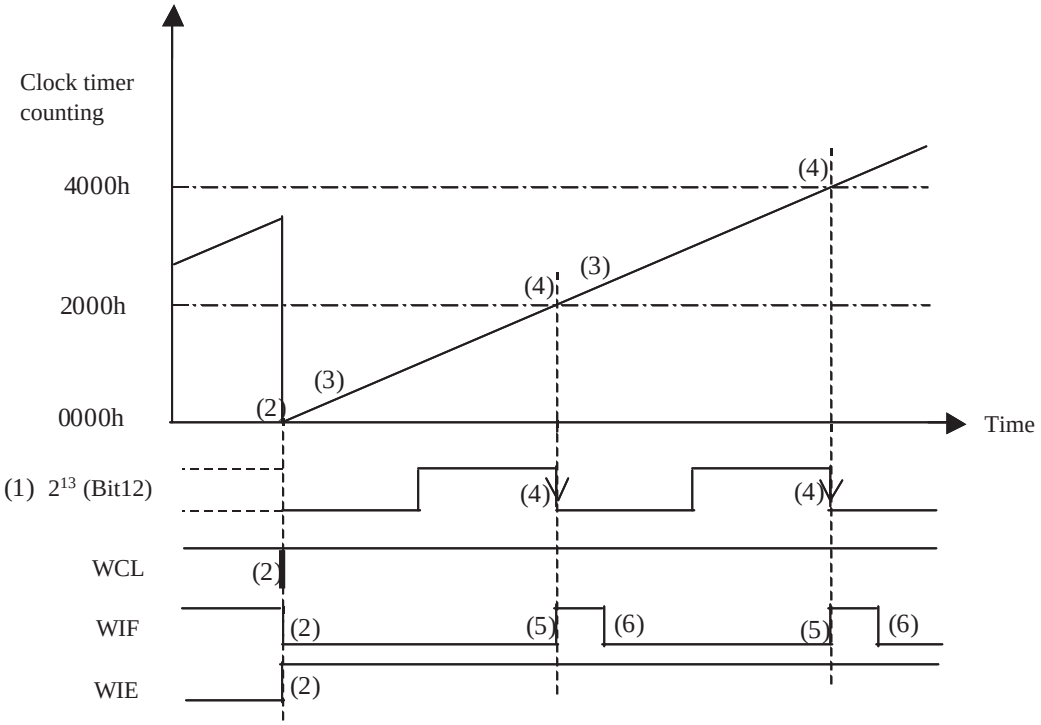
5.1 Subclock Oscillation Stability Wait Interrupt

Figure 5-1 Reference



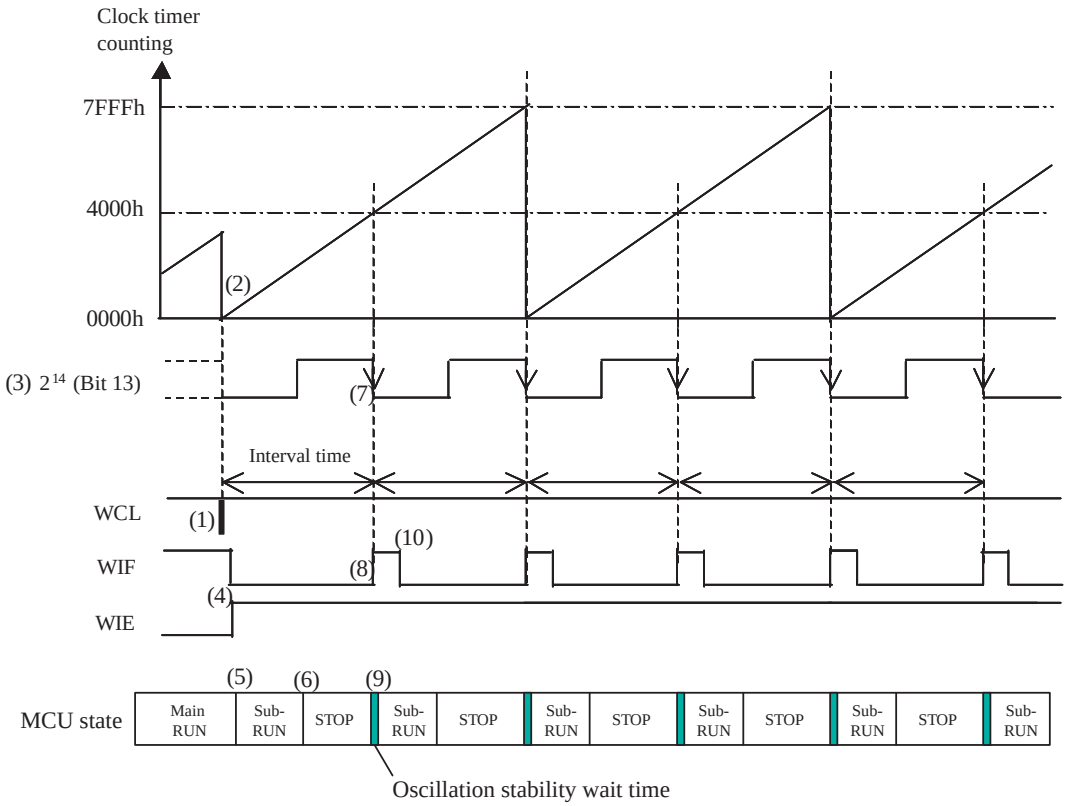
(1) Selects the interval (WS[1:0]) (In this example, $2^{10}/F_{CL\text{-}SUB}$ is selected.)
(2) Sets the timer so that it is cleared (WCL="0") by software.
(3) Sets the flag clear (WIF="0") and the interrupt request enable (WIE="1") by software.
(4) Sets the subclock stop release (OSCCR.OSCDS1="0") while the subclock is in operation by software.
(5) The subclock oscillation starts.
(6) Counts up with the subclock (source oscillation).
(7) Make the subclock oscillation stable.
(8) Makes the interval time be the selected time. (Detects the falling of 2^{10}dividing.)
(9) If the flag (WIF) becomes "1", the subclock oscillation stability wait interrupt request is generated.
(10) Processing cause by an interrupt (Software): Switching the operation clock (Sub-RUN => main RUN)
(11) Interrupt request disable (WIE="0") and the interrupt request clear (WIF="0").

5.2 Interval Interrupt (Clock Interrupt)



(1) Selects the interval time. (WS[1:0]) (In this example, $2^{13}/F_{CL\text{-}SUB}$ is selected.)
(2) Sets the timer clear (WCL="0"), flag clear (WIF="0") and interrupt request enable (WIE="1") by the software.
(3) The timer counts up with the subclock (Source oscillation).
(4) Makes the interval time be the selected time. (Detects the fall of 2^{13}.)
(5) If the flag (WIF) is set to "1", interval interrupt request (Clock interrupt request) is generated.
(6) Processing caused by an interrupt (Software): The interrupt request clear (WIF="0")
(Arbitrary processing such as clock counting)
(7) Repeats Items (3) to (6).

5.3 Returning from the Stop Mode due to Interval Operation (Clock Interrupt)



(1) The sub oscillation stabilisation timer is cleared by software. (Writes “0” to WCL.)
(2) Counts up the sub oscillation stabilisation timer with the subclock.
(3) Selects the interval time. (In this example, 0.5 second: Selects WS[1:0]=“10”.)
(4) Sets the flag clear (WIF=“0”) and sub oscillation stabilisation timer interrupt enable (WIE=“1”) by the software.
(5) Switches the MCU operation from the main RUN to sub-RUN.
(6) Switches to the stop mode.
(7) Makes the interval time be the selected time. (0.5 second)
(8) The interrupt request flag (WIF) is set to “1”.
(9) As the interrupt request is enabled (WIE=“1”), returns from the stop mode to sub-RUN.
(10) Clears the interrupt request flag by software. (Writes “0” to the WIF.)
(11) Repeats Items (6) from (10).

6. Setting

Table 6-1 Settings Required for Using the Sub oscillation stabilisation timer

Setting	Setting register	Setting method*
Setting the interval time	Sub oscillation stabilisation timer control register (WPCRH)	Refer to 7.1.
Count clear		Refer to 7.4.

*: Refer to the number for more information on the setting method.

Table 6-2 Items Required for Enabling the Sub oscillation stabilisation timer Interrupt

Setting	Setting register	Setting method*
Setting the interrupt vector and the free run timer level of the sub oscillation stabilisation timer	Refer to "Chapter 24 Interrupt Control (Page No.315)".	Refer to 7.5.
Setting the sub oscillation stabilisation timer interrupt Clearing the interrupt request Enabling the interrupt request	Sub oscillation stabilisation timer control register (WPCRH)	Refer to 7.7.

*: Refer to the number for more information on the setting method.

7. Q & A

7.1 What are the types of interval time (wait time) and how are they selected?

There are three types of interval time, and they are set with the interval selection bit (WPCRH.WS[1:0]).

Interval time	Count period	Interval (Wait time) Example
	Interval selection bit (WS[1:0])	F_{CL-SUB} = 32.768kHz
To set the interval time to $2^{10}/F_{CL-SUB}$	Set the value to “00”.	31.25ms
To set the interval time to $2^{13}/F_{CL-SUB}$	Set the value to “01”.	0.25s
To set the interval time to $2^{14}/F_{CL-SUB}$	Set the value to “10”.	0.50s
To set the interval time to $2^{15}/F_{CL-SUB}$	Set the value to “11”.	1.00s

7.2 How is the count clock selected?

The count clock is the subclock (source oscillation).

7.3 How is the sub oscillation stabilisation timer cleared?

The following methods are available to clear the sub oscillation stabilisation timer.

- Sets the clear bit (WPCRH.WCL).

Operation	Clear bit (WCL)
To clear the sub oscillation stabilisation timer	Writes “1”

- Performs a reset.

  Clears the 15-bit free run timer with the initialization reset (INIT terminal input, watchdog reset).

Note: The operation initialization reset (Software reset) holds the count of a 15-bit free run timer.

- The overflow of the sub oscillation stabilisation timer (Next count-up for “FFFFh”) causes the count value to be reset to “0000_H”.

7.4 What are interrupt-associated registers?

Setting the interrupt vector and the interrupt level of the sub oscillation stabilisation timer

The relationship between the interrupt level and the vector is shown in the following table.

Refer to “Chapter 24 Interrupt Control (Page No.315)” for more information on the interrupt level and the interrupt vector.

Interrupt vector (Default)	Interrupt level setting bit (ICR[4:0])
#143 Address: 0FFDC0h	Interrupt level register (ICR63) Address: 047Fh

As the interrupt request flag (WPCRH.WIF) is not automatically cleared, clear it before returning from the interrupt processing by the software. (Writes “0” to the WIF bit.)

7.5 What are the types of interrupt?

There is one type for the interrupt, and it is generated with the interval time (Subclock oscillation stability wait).

7.6 How is the interrupt enabled?

The interrupt request enable and the interrupt request flag

The interrupt enable is set with the interrupt request enable bit (WPCRH.WIE).

	Interrupt request enable bit (WIE)
Interrupt disable	Set the value to “0”
Interrupt enable	Set the value to “1”

The interrupt request is cleared with the interrupt request bit (WPCRH.WIF).

	Interrupt request bit (WIF)
Interrupt request clear	Writes “0”

7.7 How is the sub oscillation stabilisation timer stopped counting?

Sets with the timer operation enable bit (WPCRH.WEN). Refer to 7.3.

In addition, if the MCU stops the sub clock while the mainclock is being operated, the sub clock oscillation stability wait timer also stops counting.

8. Caution

- If the setting request (WIF="1") of the timer interrupt request flag and the writing timing where "0" is written to the flag by the software occur simultaneously, the flag is set to "1".

- If the interrupt request is enabled (WIE="1") after defeating a reset, and if the interval time is changed, be sure to simultaneously set "0" to the interrupt request enable flag (WIF) and the clear bit (WCL).

- Read-modify-write
  The interrupt request flag (WIF) is always read as "1" with the Read-modify-write.

- The setting initialization reset (INIT terminal input, watchdog reset) initializes the values of the timer interrupt request bit (WIF), timer interrupt request enable bit (WIE), timer enable bit (WEN) and timer clear bit (WCL) to "0", but cannot initialize the interval period selection bit (WS[1:0]). Be sure to set it by the software.

- Setting the initial value of the sub oscillation stabilisation timer control register is possible using the initialization reset (INIT terminal input, watchdog reset), but the operation initialization reset (Software reset) holds the current value instead of initializing the value of the sub oscillation stabilisation timer control register.

- The value for the oscillation stability wait time is an estimated value because the oscillation period of the main clock oscillation is unstable for the beginning immediately after the oscillation has started.

- An unstable clock may be supplied to the entire device, and normal operation is not guaranteed if the subclock is made to oscillate starting from subclock stopped state, and if the MCU operation mode is switched from the main RUN to the sub-RUN mode without waiting until the subclock oscillation becomes stable. Be sure to acquire the subclock oscillation stability wait time using the sub oscillation stabilisation timer, etc. (If the main clock is selected as the clock source, the oscillation stability wait time for the subclock may not be acquired.)

- The value for the oscillation stability wait time is an estimated value because the oscillation period of the subclock is unstable for the beginning immediately after it has started.

- As the sub oscillation stabilisation timer stops while the subclock stops oscillating, a clock interrupt (interval interrupt) is not generated either. If processing using the clock interrupt (interval interrupt) is performed, enable the subclock oscillation. (Do not stop the subclock oscillation).

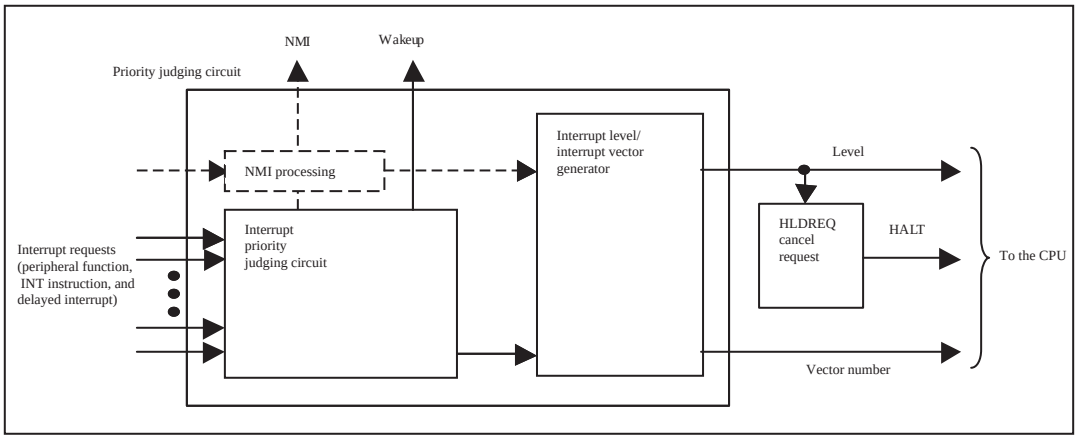
- The sub oscillation stabilisation timer counts up with the subclock. As a result, the timer stops counting because the subclock stops oscillating under the following conditions.
  - If the subclock is set that it stops in the stop mode (Subclock oscillation enable bit* ="1"), and then the mode is switched to the stop mode, the sub oscillation stabilisation timer stops counting while in the stop mode.
  - If you want the sub oscillation stabilisation timer to continue counting while in stop mode, set the subclock oscillation enable bit to "0" before switching the mode to the stop mode.
  - If the subclock stop bit ="1" while in the subclock, and if the subclock is specified so that it stops oscillating while the subclock is in operation, the sub oscillation stabilisation timer stops, too, while the subclock is in operation.

8.Caution

Chapter 24 Interrupt Control

1. Overview

Interrupt control manages interrupt reception and arbitration.



2. Features

- Functions
  - Detection of interrupt requests
  - Priority determination (determined by level and number)
  - Interrupt level propagation of the factor of the priority to the CPU
  - Interrupt number propagation of the factor of the priority to the CPU
  - Request (to the CPU) to return from stop mode by a valid interrupt (Wakeup)
- Interrupt level
  - Reserved for System: level 0 to 14
  - MNI : level 15
  - Interrupt : level 16 to 31
  - Interrupt disable : level 32
    (As the interrupt level goes up, the number goes down.)
- Number of interrupt triggers
  - NMI : 1
  - Interrupt from peripheral functions: 128
  - Delayed interrupt : 1
  - Reserved for system (for REALOS): 2
  - INT instruction : 111

3. Configuration

Figure 3-1 Configuration Diagram

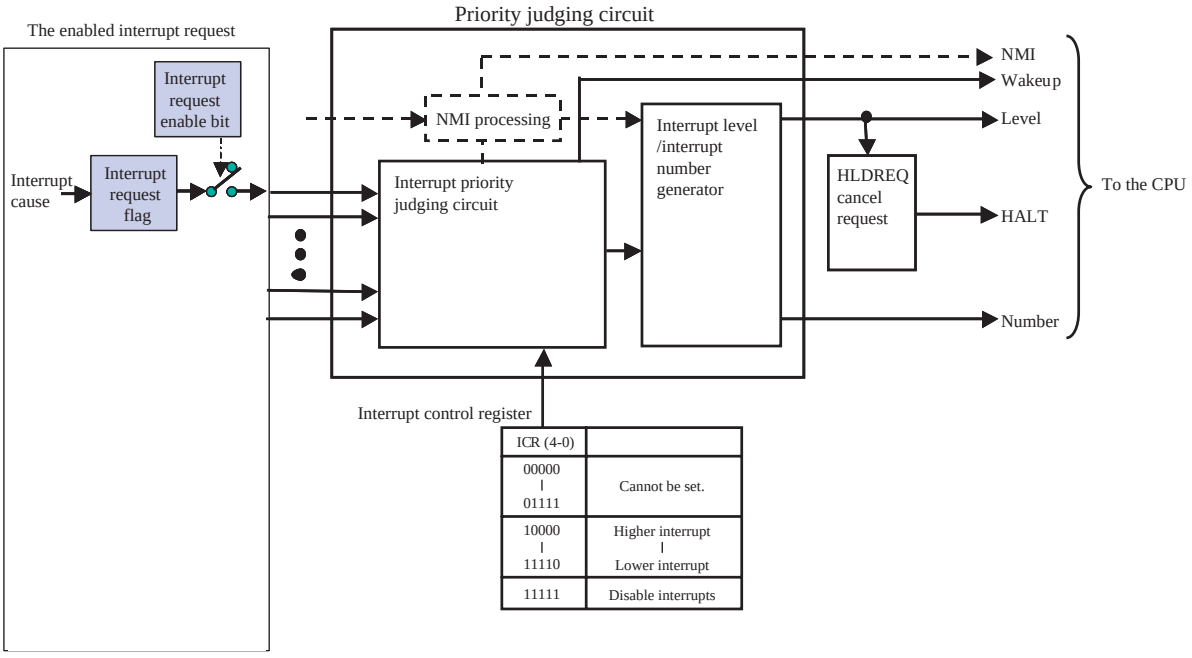
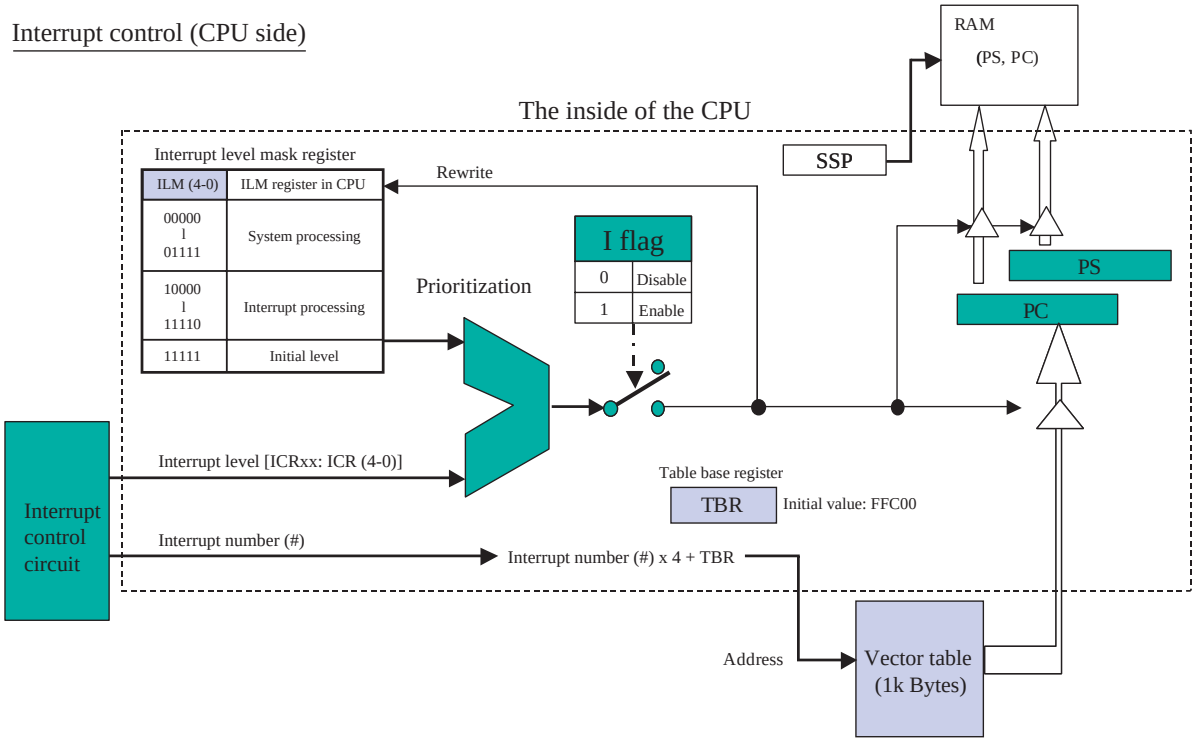


Figure 3-2 Configuration Diagram



4. Registers

4.1 ICR: Interrupt Control Register

The register that specifies the interrupt level of an interrupt request.

ICR00	#16	External Interrupt 0	: Address 0440_H	(Access: Byte)
	#17	External Interrupt 1		
ICR01	#18	External Interrupt 2	: Address 0441_H	(Access: Byte)
	#19	External Interrupt 3		
ICR02	#20	External Interrupt 4	: Address 0442_H	(Access: Byte)
	#21	External Interrupt 5		
ICR03	#22	External Interrupt 6	: Address 0443_H	(Access: Byte)
	#23	External Interrupt 7		
ICR04	#24	External Interrupt 8	: Address 0444_H	(Access: Byte)
	#25	External Interrupt 9		
ICR05	#26	External Interrupt 10	: Address 0445_H	(Access: Byte)
	#27	External Interrupt 11		
ICR06	#28	External Interrupt 12	: Address 0446_H	(Access: Byte)
	#29	External Interrupt 13		
ICR07	#30	External Interrupt 14	: Address 0447_H	(Access: Byte)
	#31	External Interrupt 15		
ICR08	#32	Reload Timer 0	: Address 0448_H	(Access: Byte)
	#33	Reload Timer 1		
ICR09	#34	Reload Timer 2	: Address 0449_H	(Access: Byte)
	#35	Reload Timer 3		
ICR10	#36	Reload Timer 4	: Address $044A_H$	(Access: Byte)
	#37	Reload Timer 5		
ICR11	#38	Reload Timer 6	: Address $044B_H$	(Access: Byte)
	#39	Reload Timer 7		
ICR12	#40	Free Run Timer 0	: Address $044C_H$	(Access: Byte)
	#41	Free Run Timer 1		
ICR13	#42	Free Run Timer 2	: Address $044D_H$	(Access: Byte)
	#43	Free Run Timer 3		
ICR14	#44	Free Run Timer 4	: Address $044E_H$	(Access: Byte)
	#45	Free Run Timer 5		
ICR15	#46	Free Run Timer 6	: Address $044F_H$	(Access: Byte)
	#47	Free Run Timer 7		
ICR16	#48	CAN 0	: Address 0450_H	(Access: Byte)
	#49	CAN 1		
ICR17	#50	CAN 2	: Address 0451_H	(Access: Byte)
	#51	CAN 3		
ICR18	#52	CAN 4	: Address 0452_H	(Access: Byte)
	#53	CAN 5		
ICR19	#54	USART (LIN) 0 RX	: Address 0453_H	(Access: Byte)
	#55	USART (LIN) 0 TX		
ICR20	#56	USART (LIN) 1 RX	: Address 0454_H	(Access: Byte)
	#57	USART (LIN) 1 TX		
ICR21	#58	USART (LIN) 2 RX	: Address 0455_H	(Access: Byte)
	#59	USART (LIN) 2 TX		
ICR22	#60	USART (LIN) 3 RX	: Address 0456_H	(Access: Byte)
	#61	USART (LIN) 3 TX		
ICR23	#62	System reserved	: Address 0457_H (*2)	(Access: Byte)
	#63	Delayed Interrupt		
ICR24	#64	System reserved (*1)	: Address 0458_H	(Access: Byte)
	#65	System reserved (*1)		
ICR25	#66	USART (LIN, FIFO) 4 RX	: Address 0459_H	(Access: Byte)
	#67	USART (LIN, FIFO) 4 TX		
ICR26	#68	USART (LIN, FIFO) 5 RX	: Address $045A_H$	(Access: Byte)
	#69	USART (LIN, FIFO) 5 TX		
ICR27	#70	USART (LIN, FIFO) 6 RX	: Address $045B_H$	(Access: Byte)
	#71	USART (LIN, FIFO) 6 TX		
ICR28	#72	USART (LIN, FIFO) 7 RX	: Address $045C_H$	(Access: Byte)
	#73	USART (LIN, FIFO) 7 TX		

ICR29	#74	I2C 0 / I2C 2	: Address $045D_H$	(Access: Byte)
	#75	I2C 1 / I2C 3		
ICR30	#76	USART (LIN) 8 RX	: Address $045E_H$	(Access: Byte)
	#77	USART (LIN) 8 TX		
ICR31	#78	USART (LIN) 9 RX	: Address $045F_H$	(Access: Byte)
	#79	USART (LIN) 9 TX		
ICR32	#80	USART (LIN) 10 RX	: Address 0460_H	(Access: Byte)
	#81	USART (LIN) 10 TX		
ICR33	#82	USART (LIN) 11 RX	: Address 0461_H	(Access: Byte)
	#83	USART (LIN) 11 TX		
ICR34	#84	USART (LIN) 12 RX	: Address 0462_H	(Access: Byte)
	#85	USART (LIN) 12 TX		
ICR35	#86	USART (LIN) 13 RX	: Address 0463_H	(Access: Byte)
	#87	USART (LIN) 13 TX		
ICR36	#88	USART (LIN) 14 RX	: Address 0464_H	(Access: Byte)
	#89	USART (LIN) 14 TX		
ICR37	#90	USART (LIN) 15 RX	: Address 0465_H	(Access: Byte)
	#91	USART (LIN) 15 TX		
ICR38	#92	Input Capture 0	: Address 0466_H	(Access: Byte)
	#93	Input Capture 1		
ICR39	#94	Input Capture 2	: Address 0467_H	(Access: Byte)
	#95	Input Capture 3		
ICR40	#96	Input Capture 4	: Address 0468_H	(Access: Byte)
	#97	Input Capture 5		
ICR41	#98	Input Capture 6	: Address 0469_H	(Access: Byte)
	#99	Input Capture 7		
ICR42	#100	Output Compare 0	: Address $046A_H$	(Access: Byte)
	#101	Output Compare 1		
ICR43	#102	Output Compare 2	: Address $046B_H$	(Access: Byte)
	#103	Output Compare 3		
ICR44	#104	Output Compare 4	: Address $046C_H$	(Access: Byte)
	#105	Output Compare 5		
ICR45	#106	Output Compare 6	: Address $046D_H$	(Access: Byte)
	#107	Output Compare 7		
ICR46	#108	Sound Generator	: Address $046E_H$	(Access: Byte)
	#109	Phase Frequ. Modulator		
ICR47	#110	System reserved	: Address $046F_H$ (*2)	(Access: Byte)
	#111	System reserved		
ICR48	#112	Prog. Pulse Gen. 0	: Address 0470_H	(Access: Byte)
	#113	Prog. Pulse Gen. 1		
ICR49	#114	Prog. Pulse Gen. 2	: Address 0471_H	(Access: Byte)
	#115	Prog. Pulse Gen. 3		
ICR50	#116	Prog. Pulse Gen. 4	: Address 0472_H	(Access: Byte)
	#117	Prog. Pulse Gen. 5		
ICR51	#118	Prog. Pulse Gen. 6	: Address 0473_H	(Access: Byte)
	#119	Prog. Pulse Gen. 7		
ICR52	#120	Prog. Pulse Gen. 8	: Address 0474_H	(Access: Byte)
	#121	Prog. Pulse Gen. 9		
ICR53	#122	Prog. Pulse Gen. 10	: Address 0475_H	(Access: Byte)
	#123	Prog. Pulse Gen. 11		
ICR54	#124	Prog. Pulse Gen. 12	: Address 0476_H	(Access: Byte)
	#125	Prog. Pulse Gen. 13		
ICR55	#126	Prog. Pulse Gen. 14	: Address 0477_H	(Access: Byte)
	#127	Prog. Pulse Gen. 15		
ICR56	#128	Up/Down Counter 0	: Address 0478_H	(Access: Byte)
	#129	Up/Down Counter 1		
ICR57	#130	Up/Down Counter 2	: Address 0479_H	(Access: Byte)
	#131	Up/Down Counter 3		
ICR58	#132	Real Time Clock	: Address $047A_H$	(Access: Byte)
	#133	Calibration Unit		
ICR59	#134	A/D Converter 0	: Address $047B_H$	(Access: Byte)
	#135	-		
ICR60	#136	Alarm Comparator 0	: Address $047C_H$	(Access: Byte)
	#137	Alarm Comparator 1		

ICR61	#138	Low Voltage Detection	: Address $047D_H$	(Access: Byte)
	#139	SMC Comparator 0-5		
ICR62	#140	Timebase Overflow	: Address $047E_H$	(Access: Byte)
	#141	PLL Clock Gear		
ICR63	#142	DMA Controller	: Address $047F_H$	(Access: Byte)
	#143	Main/Sub OSC stability wait		

(*1) : Used by REALOS

(*2): ICR23 and ICR47 can be exchanged by setting the REALOS compatibility bit (addr 0x0C03 : IOS[0])

ICR (Interrupt Control Register) is a register in the interrupt controller, and it specifies the interrupt level for each interrupt request. ICR corresponds to each of interrupt request input. ICR is mapped to the I/O space.

- **ICR00 – ICR63**

7	6	5	4	3	2	1	0	bit
–	–	–	ICR4	ICR3	ICR2	ICR1	ICR0	
–	–	–	1	1	1	1	1	Initial value
RX/WX	RX/WX	RX/WX	R/WX	R/W	R/W	R/W	R/W	Attribute

(About attributes, see “Meaning of Bit Attribute Symbols (Page No.10)”.)

- Bit 7-5: Undefined. Writing does not affect the operation. The read value is indeterminate.
- Bit 4-0: Interrupt level setting bits

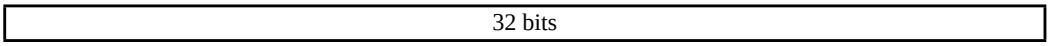
ICR4-ICR0 bits	Interrupt level	Description
0000-01110	0-14	Reserved for system (cannot to be set)
01111	15	NMI
10000	16	The highest level
10001	17	(High)
10010	18	
10011	19	
10100	20	
10101	21	
10110	22	
10111	23	
11000	24	
11001	25	
11010	26	
11011	27	
11100	28	
11101	29	(Low)
11110	30	The lowest level
11111	31	Disable interrupts

- The interrupt level setting bit specifies the interrupt level of the corresponding interrupt request.
- When the interrupt level set to the interrupt control register is the same as, or higher than the level mask value set to the ILM register of the CPU, the interrupt request is masked by the CPU side.

4.2 Interrupt Vector

Interrupt vector that corresponds to a vector number (#) with TBR register set to 0FFC00h (initial value):

#00	: Address	FFFBCh
#01	: Address	FFFB8h
≀	≀	≀
#07	: Address	FFFE0h
≀	≀	≀
#63	: Address	FFF00h
≀	≀	≀
#143	: Address	FFDC0h

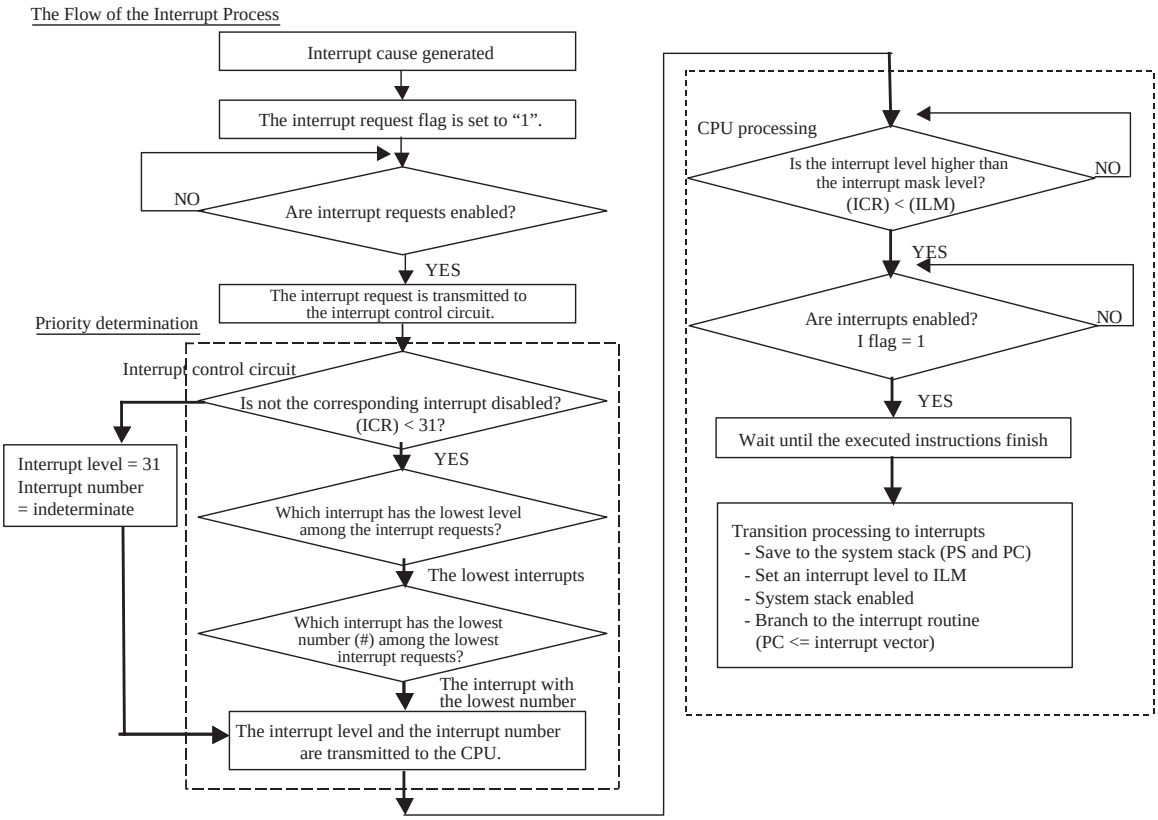


- Set the address of each interruption handling routine to the corresponding vector.
- The address of a vector = TBR (table vector register) + {3FCH - 4 x vector number (#)}
- EIT used by system (#0-#14)

Interrupt number	Interrupt level (fixed)	Factor
#0	0	Reset vector
#1	1	Mode vector
#2-#4	–	Reserved for system
#5	5	CPU Supervisor Mode
#6	6	Memory Protection Exception
#7	7	Coprocessor absence trap
#8	8	Coprocessor error trap
#9	9	INTE instruction
#10	10	Instruction break exception
#11	11	Operand break trap
#12	12	Step trace trap
#13	13	MNI request (TOOL)
#14	14	Undefined-instruction exception
#15	15	NMI request

5. Operation

The following section explains priority determination operation of interrupt control.



■ Priority determination

- The interrupt control circuit selects the highest priority factor from those that have been generated simultaneously, and outputs the factor's interrupt level (ICR) and interrupt number (#) to the CPU.
- The priority level criteria of an interrupt cause are the following conditions.
  - The value of the interrupt level is not 31. (31 is “interrupt disable”)
  - The factors with the smallest interrupt level.
  - Among these, the factor that has the smallest interrupt number.
- If nothing is applicable by the above-mentioned criteria, interrupt level 31 (11111_B) is sent to the CPU. In this case, the interrupt number is indeterminate.

6. Setting

Table 6-1 Setting Required to Use Interrupts

Setting	Setting Registers	Setting Procedure
Setting the interrupt level	Interrupt control registers (ICR00 to ICR63)	See 7.1
Clearing the interrupt request flags	See the corresponding chapter for each peripheral function.	–
Enabling interrupt requests	See the corresponding chapter for each peripheral function.	–
I flag setting	CCR register	See 7.5

*: For the setting procedure, refer to the section indicated by the number.

Table 6-2 Setting that Requires the Setting within Interrupt Processing

Setting	Setting Registers	Setting Procedure
Clearing the interrupt request flags	See the corresponding chapter for each peripheral function.	–

7. Q & A

7.1 How can I set interrupt levels?

Set by Interrupt control registers (ICR00 to ICR63).

It is necessary to set interrupt levels in advance to the control registers of the applicable interrupts.

	Interrupt control registers ICR00 to ICR63
How to specify the highest level	Set 16.
How to specify a level	Set any level (from 16 to 30).
How to specify the lowest level	Set 30.
When interrupt is not used	Set 31 (interrupt disable).

- Since the bit of the interrupt control register (ICR[4]) is fixed to “1”, 0 to 15 cannot be set to a register.

7.2 How do I enable interrupts?

To enable interrupts, all of the following three settings should be set:

- Set the value 16 to 30 to the applicable register in the interrupt control registers (ICR00-ICR63).
- Set the interrupt request enabling bit of the applicable peripheral function to “1” (enable) (See the chapter for the corresponding peripheral function).
- Set the interrupt enabling flag (I) to “1.”

7.3 How do I disable interrupts?

To disable interrupts, at least one of the following three settings should be set:

- Set the value 31 to the applicable register in the interrupt control registers (ICR00-ICR63).
- Set the interrupt request enabling bit of the applicable peripheral function to “0” (disable).
- Set the interrupt enabling flag (I) to “0” (disable all interrupts.)

7.4 How can I set an I flag?

–>In C:

I flag is set to “1” (interrupt enable) by writing __EI();.
I flag is set to “0” (interrupt disable) by writing __DI();.

Two underscores

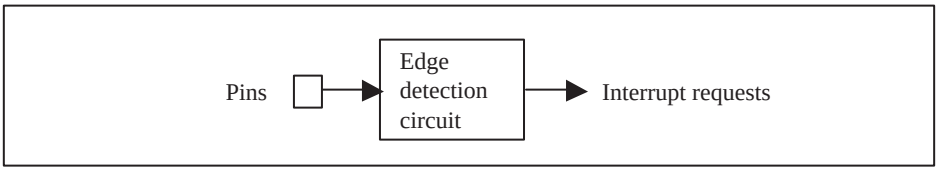
8. Caution

Interrupt request flags are not cleared automatically. Make sure to clear them in the interrupt process.
(They are usually cleared by writing “0” to the bit of the interrupt request flag, however, there are some exceptions depending on the type of peripheral functions. See the chapter for the corresponding peripheral function.)

Chapter 25 External Interrupt

1. Overview

External interrupt detects a signal input to an external interrupt input pin, and generates an interrupt request.

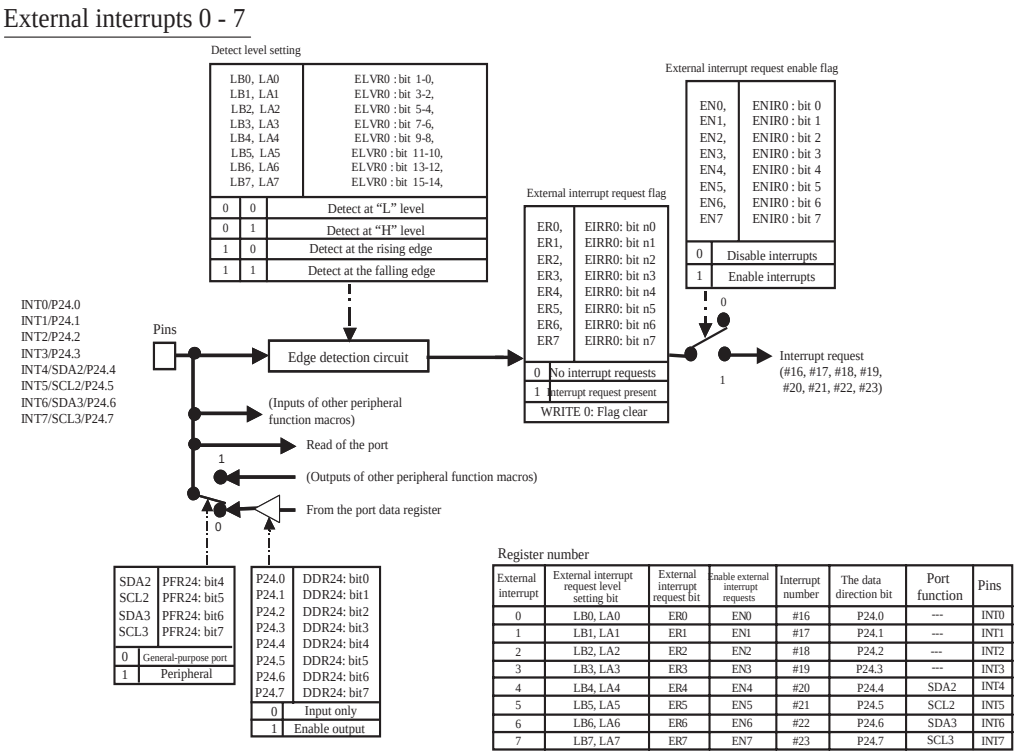


2. Features

- Quantity : 16 (INT input -- 16 channels: INT0-INT15)
- Interrupt levels: 4 levels
  - “L” level
  - “H” level
  - Rising edge
  - Falling edge

3. Configuration

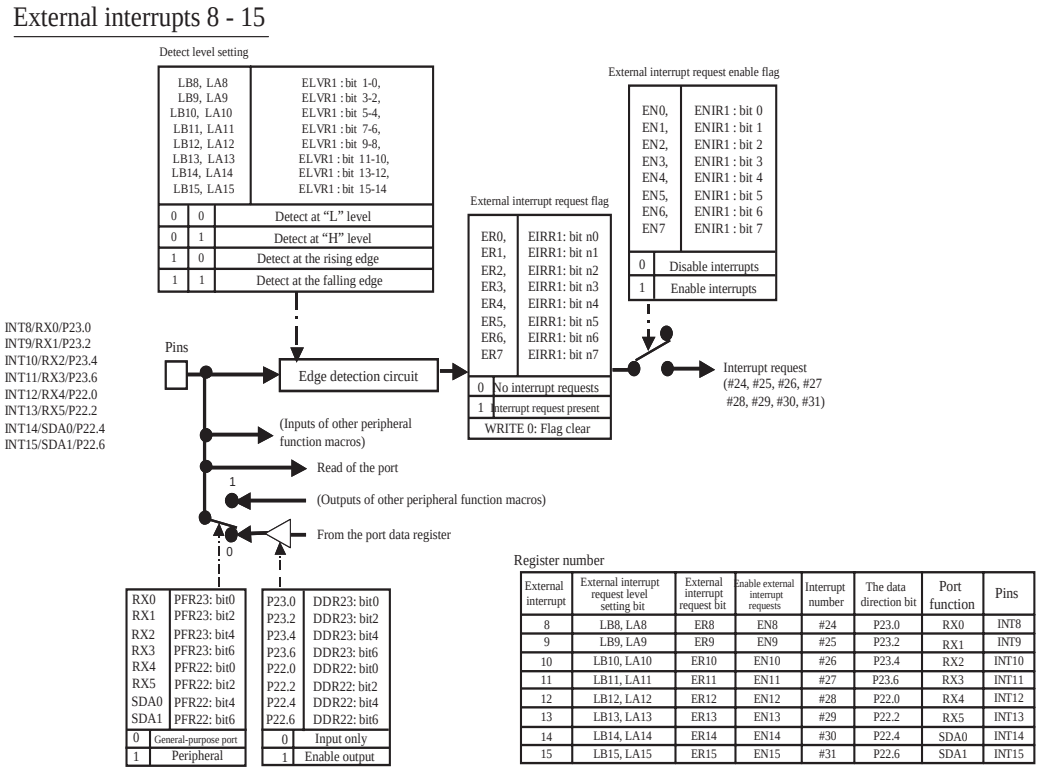
Figure 3-1 Configuration Diagram



Register number

External interrupt	External interrupt request level setting bit	External interrupt request bit	Enable external interrupt requests	Interrupt number	The data direction bit	Port function	Pins
0	LB0, LA0	ER0	EN0	#16	P24.0	---	INT0
1	LB1, LA1	ER1	EN1	#17	P24.1	---	INT1
2	LB2, LA2	ER2	EN2	#18	P24.2	---	INT2
3	LB3, LA3	ER3	EN3	#19	P24.3	---	INT3
4	LB4, LA4	ER4	EN4	#20	P24.4	SDA2	INT4
5	LB5, LA5	ER5	EN5	#21	P24.5	SCL2	INT5
6	LB6, LA6	ER6	EN6	#22	P24.6	SDA3	INT6
7	LB7, LA7	ER7	EN7	#23	P24.7	SCL3	INT7

Figure 3-2 Configuration Diagram



Register number

External interrupt	External interrupt request level setting bit	External interrupt request bit	Enable external interrupt requests	Interrupt number	The data direction bit	Port function	Pins
8	LB8, LA8	ER8	EN8	#24	P23.0	RX0	INT8
9	LB9, LA9	ER9	EN9	#25	P23.2	RX1	INT9
10	LB10, LA10	ER10	EN10	#26	P23.4	RX2	INT10
11	LB11, LA11	ER11	EN11	#27	P23.6	RX3	INT11
12	LB12, LA12	ER12	EN12	#28	P22.0	RX4	INT12
13	LB13, LA13	ER13	EN13	#29	P22.2	RX5	INT13
14	LB14, LA14	ER14	EN14	#30	P22.4	SDA0	INT14
15	LB15, LA15	ER15	EN15	#31	P22.6	SDA1	INT15

Figure 3-3 Register List

External interrupt (0-7)

Address	Bit 7	6	5	4	3	2	1	0		
00030H	EN7	EN6	EN5	EN4	EN3	EN2	EN1	EN0	EIRR0	(External interrupt cause 0)
00031H	EN7	EN6	EN5	EN4	EN3	EN2	EN1	EN0	ENIR0	(External interrupt enable 0)

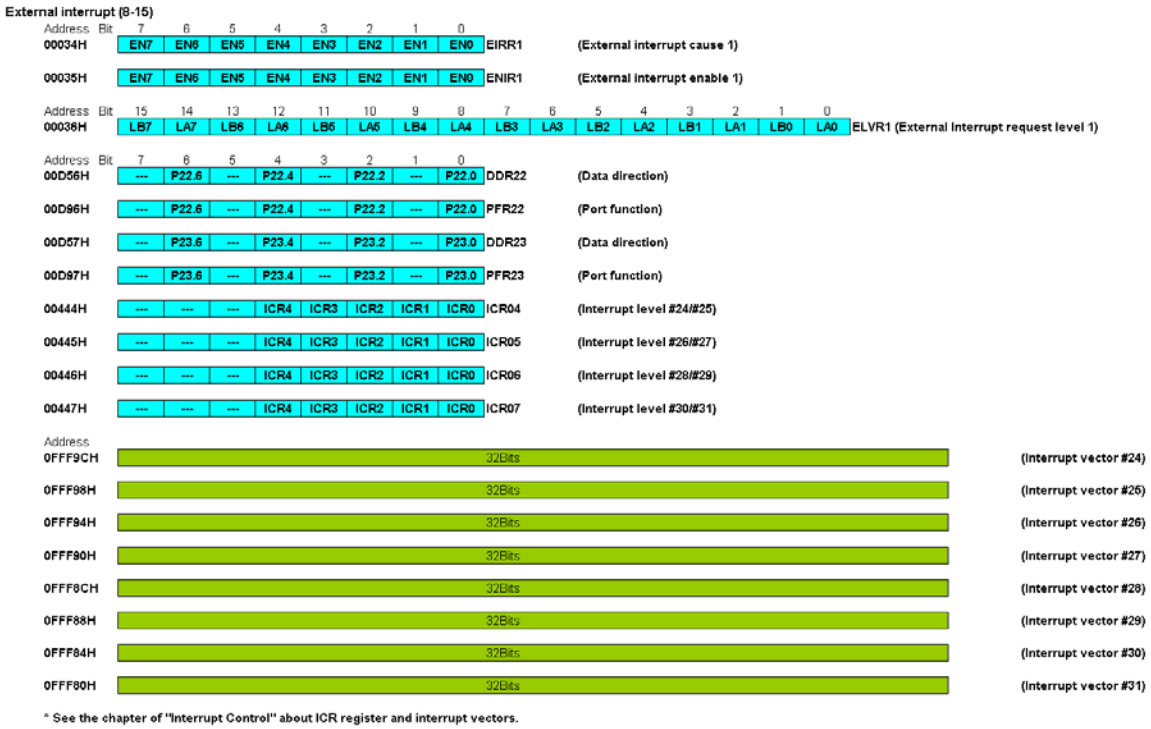
	Bit 15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
00032H	LB7	LA7	LB6	LA6	LB5	LA5	LB4	LA4	LB3	LA3	LB2	LA2	LB1	LA1	LB0	LA0	ELVR0 (External interrupt request level 0)

	Bit 7	6	5	4	3	2	1	0		
00D58H	P24.7	P24.6	P24.5	P24.4	P24.3	P24.2	P24.1	P24.0	DDR24	(Data direction)
00D98H	P24.7	P24.6	P24.5	P24.4	P24.3	P24.2	P24.1	P24.0	PFR24	(Port function)
00440H	---	---	---	ICR4	ICR3	ICR2	ICR1	ICR0	ICR00	(Interrupt level #16/#17)
00441H	---	---	---	ICR4	ICR3	ICR2	ICR1	ICR0	ICR01	(Interrupt level #18/#19)
00442H	---	---	---	ICR4	ICR3	ICR2	ICR1	ICR0	ICR02	(Interrupt level #20/#21)
00443H	---	---	---	ICR4	ICR3	ICR2	ICR1	ICR0	ICR03	(Interrupt level #22/#23)

0FFFBCH	32Bits	(Interrupt vector #16)
0FFFB8H	32Bits	(Interrupt vector #17)
0FFFB4H	32Bits	(Interrupt vector #18)
0FFFB0H	32Bits	(Interrupt vector #19)
0FFFACH	32Bits	(Interrupt vector #20)
0FFFA8H	32Bits	(Interrupt vector #21)
0FFFA4H	32Bits	(Interrupt vector #22)
0FFFA0H	32Bits	(Interrupt vector #23)

* See the chapter of "Interrupt Control" about ICR register and interrupt vectors.

Figure 3-4 Register List



Note: See “Chapter 24 Interrupt Control (Page No.315)” about ICR register and interrupt vectors.

4. Registers

4.1 ELVR: Interrupt Request Level Register

The register that selects request detection of external interrupts.

- **ELVR0 (INT0-INT7): Address 032H (access: Half-word, Word)**

15	14	13	12	11	10	9	8	Bit
LB7	LA7	LB6	LA6	LB5	LA5	LB4	LA4	
0	0	0	0	0	0	0	0	Initial value
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Attribute

7	6	5	4	3	2	1	0	Bit
LB3	LA3	LB2	LA2	LB1	LA1	LB0	LA0	
0	0	0	0	0	0	0	0	Initial value
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Attribute

(About attributes, see “Meaning of Bit Attribute Symbols (Page No.10)”.)

- **ELVR1 (INT8-INT15): Address 036H (access: Half-word, Word)**

15	14	13	12	11	10	9	8	Bit
LB15	LA15	LB14	LA14	LB13	LA13	LB12	LA12	
0	0	0	0	0	0	0	0	Initial value
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Attribute

7	6	5	4	3	2	1	0	Bit
LB11	LA11	LB10	LA10	LB9	LA9	LB8	LA8	
0	0	0	0	0	0	0	0	Initial value
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Attribute

(About attributes, see “Meaning of Bit Attribute Symbols (Page No.10)”.)

Interrupt request level bits (LBn, LAn) are registers that select request detection.

2 bits (LBn, LAn) are assigned to each external interrupt INTn.

LBn	LAn	Description
0	0	Detect “L” level and generate an interrupt request.
0	1	Detect “H” level and generate an interrupt request.
1	0	Detect the rise and generate an interrupt request.
1	1	Detect the fall and generate an interrupt request.

When the request input is a level (LAn, LBn = “00” or “01”), and when the INTn pin input is the valid level, the corresponding bit (ERn) will be re-set to “1” even if the external interrupt request bit (ERn) is set to “0”.

Note: n = 0 to 15

4.2 EIRR: Interrupt Request Register

Status bit of a request of an external interrupt.

- **EIRR0 (INT0-INT7): Address 030H (access: Byte, Half-word, Word)**

7	6	5	4	3	2	1	0	Bit
ER7	ER6	ER5	ER4	ER3	ER2	ER1	ER0	
0	0	0	0	0	0	0	0	Initial value
R (RM1), W	R (RM1), W	R (RM1), W	R (RM1), W	R (RM1), W	R (RM1), W	R (RM1), W	R (RM1), W	Attribute

(About attributes, see “Meaning of Bit Attribute Symbols (Page No.10)”.)

- **EIRR1 (INT8-INT15): Address 034H (access: Byte, Half-word, Word)**

7	6	5	4	3	2	1	0	Bit
ER15	ER14	ER13	ER12	ER11	ER10	ER9	ER8	
0	0	0	0	0	0	0	0	Initial value
R (RM1), W	R (RM1), W	R (RM1), W	R (RM1), W	R (RM1), W	R (RM1), W	R (RM1), W	R (RM1), W	Attribute

(About attributes, see “Meaning of Bit Attribute Symbols (Page No.10)”.)

An external interrupt request bit (ERn) indicates the corresponding external interrupt request.

ERn	Description	
	Read value	Write value
0	No external interrupt request present	Clear external interrupt factor bits
1	External interrupt request present	No effect on operation

Note: n = 0 to 15

4.3 ENIR: Interrupt Request Enable Register

Enable bit of external interrupt requests.

- **ENIR0 (INT0-INT7): Address 031H (access: Byte, Half-word, Word)**

7	6	5	4	3	2	1	0	Bit
EN7	EN6	EN5	EN4	EN3	EN2	EN1	EN0	
0	0	0	0	0	0	0	0	Initial value
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Attribute

(About attributes, see “Meaning of Bit Attribute Symbols (Page No.10)”.)

- **ENIR1 (INT8-INT15): Address 035H (access: Byte, Half-word, Word)**

7	6	5	4	3	2	1	0	Bit
EN15	EN14	EN13	EN12	EN11	EN10	EN9	EN8	
0	0	0	0	0	0	0	0	Initial value
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Attribute

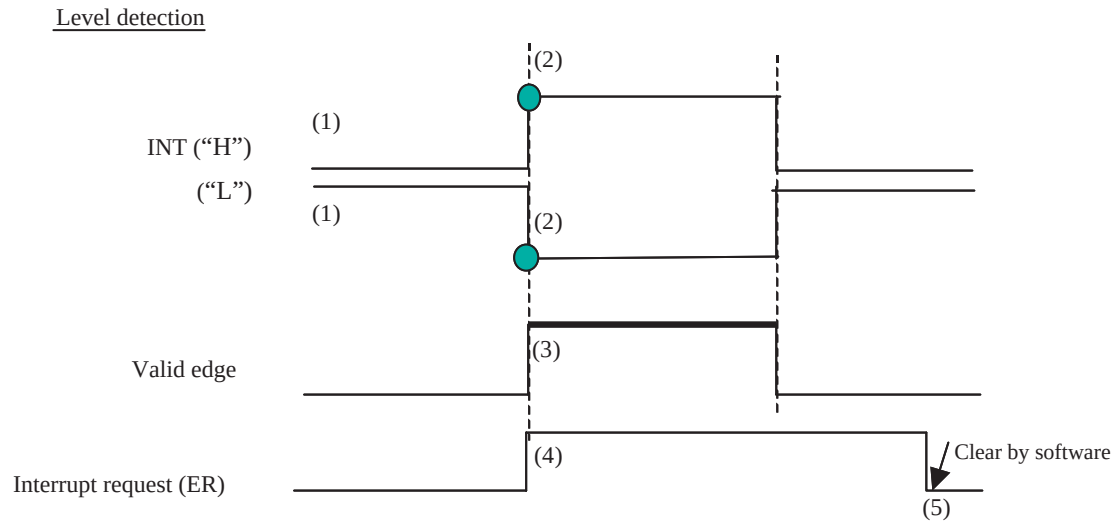
(About attributes, see “Meaning of Bit Attribute Symbols (Page No.10)”.)

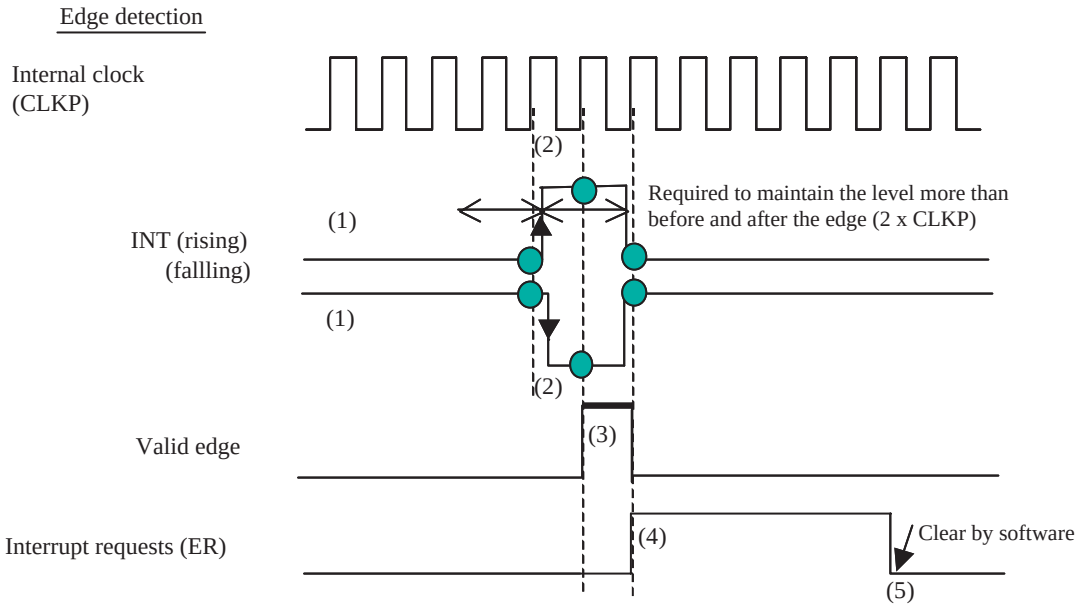
An external interrupt request enable bit (ENn) enables the corresponding external interrupt request.

ERn	Description
0	External interrupt request output disable
1	External interrupt request output enable

Note: n = 0 to 15

5. Operation





(1) External interrupt signal (INT) input
(2) Detect interrupt signals (level/edge).
(3) Valid edge signal (2xCLKP above required)
(4) An interrupt request generated.
(5) The interrupt request is cleared with software.

Remark: When waking up from STOP mode with edge detection enabled a minimum pulse pulse width (> 50ns) of the INT signal trigger must be fulfilled.

6. Setting

Table 6-1 Setting Required in Order to Use External Interrupts

Setting	Setting Registers	Setting Procedures*
Setting of detect level	External interrupt request level setting register (ELVR0 - ELVR1)	See 7.1
Set INT pin as the input.	Data direction register (DDR22, DDR23, DDR24) Port function register (PFR22, PFR23, PFR24)	See 7.2
External interrupt	External inputs →Inputs the signal to INT0 - INT15 pins.	–

Note: For the setting procedure, refer to the section indicated by the number.

7. Q & A

7.1 What are the types and setting procedures of detect levels?

There are 4 types of detect levels: “L” level, “H” level, rise, and fall

Carry out in Detection level bit (ELVR0. LBx, LAx) x = 0-7, and (ELVR1. LBx, LAx) x = 8-15.

Operation mode	Detection level bit (LBn, LAn) n = 0-15
Use as “L” level detection	Sets to “00”
Use as “H” level detection	Sets to “01”
Use as rise detection	Sets to “10”
Use as fall detection	Sets to “11”

7.2 How do I set INT pin as the input?

Use data direction registers (DDR22, DDR23, DDR24).

Use port function register (PFR22, PFR23, PFR24).

Operation	Data Direction bits	Setting	Port Function bit	Setting
To use INT0 pin input	DDR24.0	Set to “0”	PFR24.0	Set to “1”
To use INT1 pin input	DDR24.1	Set to “0”	PFR24.1	Set to “1”
To use INT2 pin input	DDR24.2	Set to “0”	PFR24.2	Set to “1”
To use INT3 pin input	DDR24.3	Set to “0”	PFR24.3	Set to “1”
To use INT4 pin input	DDR24.4	Set to “0”	PFR24.4	Set to “1”
To use INT5 pin input	DDR24.5	Set to “0”	PFR24.5	Set to “1”
To use INT6 pin input	DDR24.6	Set to “0”	PFR24.6	Set to “1”
To use INT7 pin input	DDR24.7	Set to “0”	PFR24.7	Set to “1”
To use INT8 pin input	DDR23.0	Set to “0”	PFR23.0	Set to “1”
To use INT9 pin input	DDR23.2	Set to “0”	PFR23.2	Set to “1”
To use INT10 pin input	DDR23.4	Set to “0”	PFR23.4	Set to “1”
To use INT11 pin input	DDR23.6	Set to “0”	PFR23.6	Set to “1”
To use INT12 pin input	DDR22.0	Set to “0”	PFR22.0	Set to “1”
To use INT13 pin input	DDR22.2	Set to “0”	PFR22.2	Set to “1”
To use INT14 pin input	DDR22.4	Set to “0”	PFR22.4	Set to “1”
To use INT15 pin input	DDR22.6	Set to “0”	PFR22.6	Set to “1”

Remark: Even though the external interrupt can be even used with setting DDR=0 and PFR=0 (general purpose port input mode), the input line will be disabled when setting STOP mode with HIZ.

7.3 What interrupt registers are used?

Setting of interrupt vectors of external interrupts, and interrupt levels

The relationship among external interrupt numbers, interrupt levels, and vectors is shown in the table below.

See “Chapter 24 Interrupt Control (Page No.315)” about the details of interrupt levels and interrupt vectors.

	Interrupt vectors (default)	Interrupt level setting bits (ICR[4:0])
INT0	#16 Address: 0FFFBCh	Interrupt level register (ICR00) Address: 00440h
INT1	#17 Address: 0FFFB8h	
INT2	#18 Address: 0FFFB4h	Interrupt level register (ICR01) Address: 00441h
INT3	#19 Address: 0FFFB0h	
INT4	#20 Address: 0FFFACh	Interrupt level register (ICR02) Address: 00442h
INT5	#21 Address: 0FFFA8h	
INT6	#22 Address: 0FFFA4h	Interrupt level register (ICR03) Address: 00443h
INT7	#23 Address: 0FFFA0h	
INT8	#24 Address: 0FFF9Ch	Interrupt level register (ICR04) Address: 00444h
INT9	#25 Address: 0FFF98h	
INT10	#26 Address: 0FFF94h	Interrupt level register (ICR05) Address: 00445h
INT11	#27 Address: 0FFF90h	
INT12	#28 Address: 0FFF8Ch	Interrupt level register (ICR06) Address: 00446h
INT13	#29 Address: 0FFF88h	
INT14	#30 Address: 0FFF84h	Interrupt level register (ICR07) Address: 00447h
INT15	#31 Address: 0FFF80h	

7.4 Interrupt types

Interrupt causes are limited to external interrupts. There is no bit for selection.

7.5 How do I enable, disable, and clear interrupts?

Enable flag for interrupt requests, interrupt request flag

Use interrupt enabling bits (ENIR0.ENx. x = 0-7) and (ENIR1.ENx. x = 8-15) to enable interrupts.

	Interrupt enabling bit (En [n = 0-15])
To disable interrupt requests	Sets to “0”
To enable interrupt requests	Sets to “1”

Use interrupt request bits (EIRR0.ERx. x = 0-7) and (EIRR1.ERx. x = 8-15) to clear interrupt requests.

	Interrupt request bit (Ern [n = 0-15])
To clear interrupt requests	Writes “0”

8. Caution

- When the request input is a level (LAn, LBn = “00” or “01”) and when the INT pin input is the set active level, the corresponding bit (ERn) will be re-set to “1” even if the external interrupt request bit (ERn) is set to “0”.

Note: n = 0 to 15

- Before enabling the external interrupt request with ENn = “1” it is recommended to clear the external interrupt request bit (set ERn to “0”) to avoid interrupts caused by previous matches of the input trigger (the IRQ flag is set independently of the setting of ENn).

Note: n = 0 to 15

- Before going into standby (stop mode), make sure to disable unused external interrupts (ENn = “0”).

Note: n = 0 - 15

- Minimum 3x CLKP (peripheral clock) is required for the pulse width to detect the edge presence when the request level is set to the edge request.

- When waking up from STOP mode with edge detection enabled a minimum pulse width (> 50ns) of the INT signal trigger must be fulfilled.

- The interrupt request to the interrupt controller remains active even if an external interrupt request is input from the external interrupt pin INTn and canceled afterward, since the interrupt request flag (ERn) is present. To cancel the interrupt request to the interrupt controller, the interrupt request flag must be cleared (ERn = “0”) with software. (See the diagram in “5. Operation (Page No.331)”)

Note: n = 0 to 15

8.Caution

Chapter 26 DMA Controller

1. Overview of the DMA Controller (DMAC)

The DMA controller (DMAC) is a module that implements DMA (Direct Memory Access) transfer on FR family devices. When this module is used to control DMA transfer, various kinds of data can be transferred at high speed by bypassing the CPU, enhancing system performance.

■ Hardware Configuration

The DMA controller (DMAC) consists mainly of the following blocks:

Five independent DMA channels

- 5-channel independent access control circuit
- 32-bit address registers (reload specifiable, two registers for each channel)
- 16-bit transfer count register (reload specifiable, one register for each channel)
- 4-bit block count register (one for each channel)
- Up to 128 internal transfer request sources
- External transfer request input pins: DREQ0, DREQ1, DREQ2, DREQ3 (for ch0-3 only)
- External transfer request acceptance output pins: DACK0, DACK1, DACK2, DACK3 (for ch0-3 only)
- DMA end output pins: DEOP0, DEOP1, DEOP2, DEOP3 (for ch0-3 only)
- Fly-by transfer (memory to I/O and I/O to memory) (for ch0-3 only)
- 2-cycle transfer

■ Main Functions

The following are the main functions related to data transfer by the DMA controller (DMAC):

Data can be transferred independently over multiple channels (5 channels)

  - Priority (ch.0>ch.1>ch.2>ch.3>ch.4)
  - The order can be rotated between ch.0 and ch.1.
  - DMAC start sources

- External dedicated pin input (edge detection/level detection for ch0-3 only)
- Built-in peripheral requests (shared interrupt requests, including external interrupts)
- Software request (register write)

  - Transfer mode

- Demand transfer, burst transfer, step transfer, and block transfer
- Addressing mode: 32-bit full addressing (increment/decrement/fixed)

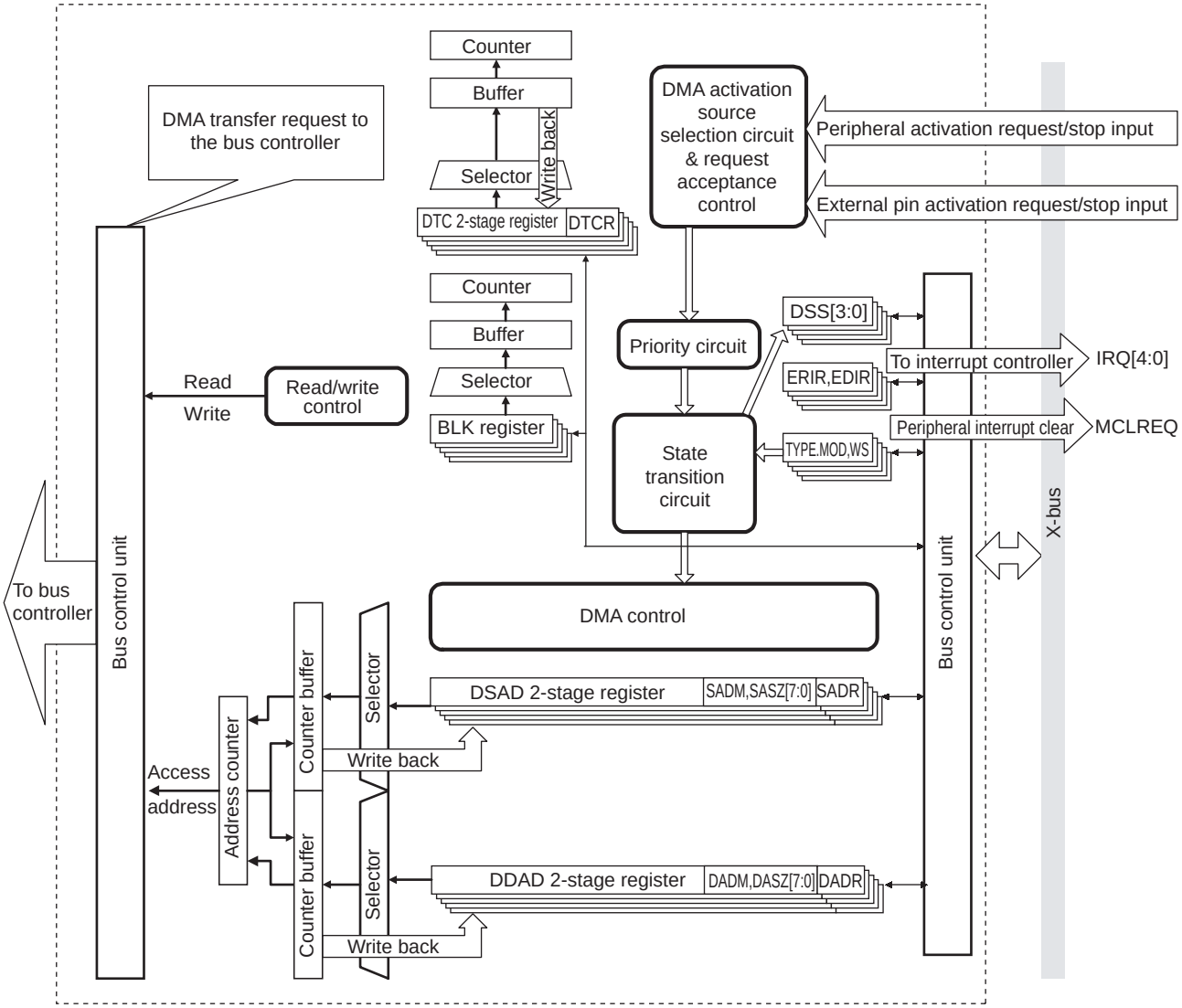
  The address increment/decrement range is from -255 to +255.

- Data types: Byte, halfword, and word length
- Single shot/reload selectable

■ Block Diagram

Figure 1-1"Block Diagram of the DMA Controller (DMAC)" is a block diagram of the DMA controller (DMAC).

Figure 1-1 Block Diagram of the DMA Controller (DMAC)



2. DMA Controller (DMAC) Registers

This section describes the configuration and functions of the registers used by the DMA controller (DMAC).

■ DMA Controller (DMAC) registers

Figure 2-1"DMA Controller (DMAC) Registers" shows the registers of the DMA controller (DMAC).

Figure 2-1 DMA Controller (DMAC) Registers

■ Notes on Setting Registers

When the DMA controller (DMAC) is set, some bits need to be set while DMA is stopped. If they are set while DMA is in progress (during transfer), correct operation cannot be guaranteed.

An asterisk following a bit when its function is described later indicates that the operation of the bit is affected if it is set during DMAC transfer. Rewrite this bit while DMAC transfer is stopped (start is disabled or temporarily stopped).

If the bit is set while DMA transfer start is disabled (when DMAE of DMACR=0, or DENB of DMACA=0), the setting takes effect when start is enabled.

If the bit is set while DMA transfer is temporarily stopped (DMAH[3:0] of DMACR not equal to 0000_B or PAUS of DMACA=1), the setting takes effect when temporary stopping is canceled.

2.1 Control/Status Registers A (DMACA0 to 4)

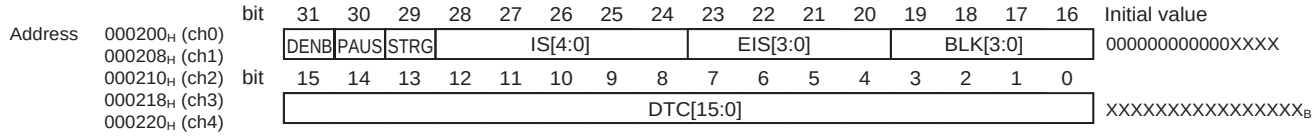
Control/status registers A (DMACA0 to 4) control the operation of the DMAC channels. There is a separate register for each channel.

This section describes the configuration and functions of control/status registers A (DMACA0 to 4).

■ Bit Configuration of Control/Status Registers A (DMACA0 to 4)

Figure 2-2"Bit Configuration of Control/Status Registers A (DMACA0 to 4)" shows the bit configuration of control/status registers A (DMACA0 to 4).

Figure 2-2 Bit Configuration of Control/Status Registers A (DMACA0 to 4)



■ Detailed Bit of Control/Status Registers A (DMACA0 to 4)

The following describes the functions of the bits of control/status registers A (DMACA0 to 4).

[Bit 31] DENB (Dma ENaBle): DMA operation enable bit

This bit, which corresponds to a transfer channel, is used to enable and disable DMA transfer.

The activated channel starts DMA transfer when a transfer request is generated and accepted.

All transfer requests that are generated for a deactivated channel are disabled.

When the transfer on an activated channel reaches the specified count, this bit is set to 0 and transfer stops.

The transfer can be forced to stop by writing 0 to this bit. Be sure to stop a transfer forcibly (0 write) only after temporarily stopping DMA using the PUAS bit (Bit30 of DMACA). If the transfer is forced to stop without first temporarily stopping DMA, DMA stops but the transferred data cannot be guaranteed. Check whether DMA is stopped using the DSS[2:0] bits [Bit18-16 of DMACB].

DENB	Function
0	Disables operation of DMA on the corresponding channel (initial value).
1	Enables operation of DMA on the corresponding channel.

- If a stop request is accepted during reset: Initialized to 0.
- This bit is readable and writable.
- If the operation of all channels is disabled by Bit15 (DMAE bit) of the DMAC all-channel control register (DMACR), writing 1 to this bit is disabled and the stopped state is maintained. If the operation is disabled by the above bit while it is enabled by this bit, 0 is written to this bit and the transfer is stopped (forced stop).

[Bit 30] PAUS (PAUSe)*: Temporary stop instruction

This bit temporarily stops DMA transfer on the corresponding channel. If this bit is set, DMA transfer is not performed before this bit is cleared (While DMA is stopped, the DSS bits are $1xx_B$.

If this bit is set before starting, DMA transfer continues to be temporarily stopped.

New transfer requests that occur while this bit is set are accepted, but no transfer starts before this bit is cleared (See 3.9"Operation from Starting to End/Stopping").

PAUS	Function
0	Enables operation of the corresponding channel DMA (initial value)
1	Temporarily stops DMA on the corresponding channel.

- When reset: Initialized to 0.
- This bit is readable and writable.

[Bit 29] STRG (Software TRiGger): Transfer request

This bit generates a DMA transfer request for the corresponding channel. If 1 is written to this bit, a transfer request is generated when write operation to the register is completed and transfer on the corresponding channel is started.

However, if the corresponding channel is not activated, operations on this bit are disabled.

If starting by a write operation to the DMAE bit and a transfer request occurring due to this bit are simultaneous, the transfer request is enabled and transfer is started. If writing of 1 to the PAUS bit and a transfer request occurring due to this bit are simultaneous, the transfer request is enabled, but DMA transfer is not started before 0 is written to the PAUS bit.

STRG	Function
0	Disabled
1	DMA starting request

- When reset: Initialized to 0.
- The read value is always 0.
- Only a write value of 1 is valid. If 0 is write, operation is not affected.

[Bits 28 to 24] IS4 to 0 (Input Select)*: Transfer source selection

These bits select the source of a transfer request note that the software transfer request by the STRG bit function is always valid regardless of the setting of these bits. As listed in Table 2-1 "Settings for Transfer Request Sources".

Table 2-1 Settings for Transfer Request Sources

IS	EIS	RN	Function	Transfer stop request
00000	-	-	Activation by hardware prohibited	not available
00001 - 01101	-	-	Setting prohibited - Setting prohibited	
01110	-	-	External DMA-pin high level or rising edge	
01111	-	-	External DMA-pin low level or falling edge	
10000	0000	0	External Interrupt 0	-
10001	0000	1	External Interrupt 1	-
10010	0000	2	External Interrupt 2	-
10011	0000	3	External Interrupt 3	-
10100	0000	4	Reload Timer 0	-
10101	0000	5	Reload Timer 1	-
10110	0000	6	USART (LIN) 0 RX	available
10111	0000	7	USART (LIN) 0 TX	-
11000	0000	8	USART (LIN) 1 RX	available
11001	0000	9	USART (LIN) 1 TX	-
11010	0000	10	USART (LIN, FIFO) 4 RX	available
11011	0000	11	USART (LIN, FIFO) 4 TX	-
11100	0000	12	USART (LIN, FIFO) 5 RX	available
11101	0000	13	USART (LIN, FIFO) 5 TX	-
11110	0000	14	A/D Converter	-
11111	0000	15	Programmable Pulse Generator (PPG) 0	-

- When reset: IS4-0 is initialized to 00000_B.
- When reset: EIS3-0 is initialized to 0000_B.
- These bits are readable and writable.

Notes:

- If DMA start resulting from an interrupt from a peripheral function is set (IS=1xxxx$_B$), disable interrupts from the selected peripheral function with the ICR register.
- If demand transfer mode is selected, only IS[4:0]=01110$_B$, 01111$_B$ can be set. Starting by other sources is disabled.
- External request input is valid only for CH0, 1, and 2. External request input cannot be selected for CH2, CH3 and 4. Whether level detection or edge detection is used is determined by the mode setting. Level detection is selected for demand transfer. For all other cases, edge detection is selected.

[Bits 23 to 20] EIS3 to 0 (Extended Input Select)*: Extended Transfer Source Selection

These bits select the source of a transfer request note that the software transfer request by the STRG bit function is always valid regardless of the setting of these bits. As listed in Table 2-2 "Settings for Extended Transfer Request Sources".

Table 2-2 Settings for Extended Transfer Request Sources

IS	EIS	RN	Function	Transfer stop request
10000	0001	16	External Interrupt 0	-
10001	0001	17	External Interrupt 1	-
10010	0001	18	External Interrupt 2	-
10011	0001	19	External Interrupt 3	-
10100	0001	20	External Interrupt 4	-
10101	0001	21	External Interrupt 5	-
10110	0001	22	External Interrupt 6	-
10111	0001	23	External Interrupt 7	-
11000	0001	24	res.	-
11001	0001	25	res.	-
11010	0001	26	res.	-
11011	0001	27	res.	-
11100	0001	28	res.	-
11101	0001	29	res.	-
11110	0001	30	res.	-
11111	0001	31	res.	-
10000	0010	32	Reload Timer 0	-
10001	0010	33	Reload Timer 1	-
10010	0010	34	Reload Timer 2	-
10011	0010	35	Reload Timer 3	-
10100	0010	36	Reload Timer 4	-
10101	0010	37	Reload Timer 5	-

Table 2-2 Settings for Extended Transfer Request Sources

IS	EIS	RN	Function	Transfer stop request
10110	0010	38	Reload Timer 6	-
10111	0010	39	Reload Timer 7	-
11000	0010	40	Free Run Timer 0	-
11001	0010	41	Free Run Timer 1	-
11010	0010	42	Free Run Timer 2	-
11011	0010	43	Free Run Timer 3	-
11100	0010	44	Free Run Timer 4	-
11101	0010	45	Free Run Timer 5	-
11110	0010	46	Free Run Timer 6	-
11111	0010	47	Free Run Timer 7	-
10000	0011	48	USART (LIN) 0 RX	available
10001	0011	49	USART (LIN) 0 TX	-
10010	0011	50	USART (LIN) 1 RX	available
10011	0011	51	USART (LIN) 1 TX	-
10100	0011	52	USART (LIN) 2 RX	available
10101	0011	53	USART (LIN) 2 TX	-
10110	0011	54	USART (LIN) 3 RX	available
10111	0011	55	USART (LIN) 3 TX	-
11000	0011	56	USART (LIN, FIFO) 4 RX	available
11001	0011	57	USART (LIN, FIFO) 4 TX	-
11010	0011	58	USART (LIN, FIFO) 5 RX	available
11011	0011	59	USART (LIN, FIFO) 5 TX	-
11100	0011	60	USART (LIN, FIFO) 6 RX	available
11101	0011	61	USART (LIN, FIFO) 6 TX	-
11110	0011	62	USART (LIN, FIFO) 7 RX	available
11111	0011	63	USART (LIN, FIFO) 7 TX	-
10000	0100	64	USART (LIN) 8 RX	available
10001	0100	65	USART (LIN) 8 TX	-
10010	0100	66	USART (LIN) 9 RX	available
10011	0100	67	USART (LIN) 9 TX	-
10100	0100	68	USART (LIN) 10 RX	available
10101	0100	69	USART (LIN) 10 TX	-

Table 2-2 Settings for Extended Transfer Request Sources

IS	EIS	RN	Function	Transfer stop request
10110	0100	70	USART (LIN) 11 RX	available
10111	0100	71	USART (LIN) 11 TX	-
11000	0100	72	USART (LIN) 12 RX	available
11001	0100	73	USART (LIN) 12 TX	-
11010	0100	74	USART (LIN) 13 RX	available
11011	0100	75	USART (LIN) 13 TX	-
11100	0100	76	USART (LIN) 14 RX	available
11101	0100	77	USART (LIN) 14 TX	-
11110	0100	78	USART (LIN) 15 RX	available
11111	0100	79	USART (LIN) 15 TX	-
10000	0101	80	Input Capture 0	-
10001	0101	81	Input Capture 1	-
10010	0101	82	Input Capture 2	-
10011	0101	83	Input Capture 3	-
10100	0101	84	Input Capture 4	-
10101	0101	85	Input Capture 5	-
10110	0101	86	Input Capture 6	-
10111	0101	87	Input Capture 7	-
11000	0101	88	Output Compare 0	-
11001	0101	89	Output Compare 1	-
11010	0101	90	Output Compare 2	-
11011	0101	91	Output Compare 3	-
11100	0101	92	Output Compare 4	-
11101	0101	93	Output Compare 5	-
11110	0101	94	Output Compare 6	-
11111	0101	95	Output Compare 7	-
10000	0110	96	Programmable Pulse Generator 0	-
10001	0110	97	Programmable Pulse Generator 1	-
10010	0110	98	Programmable Pulse Generator 2	-
10011	0110	99	Programmable Pulse Generator 3	-
10100	0110	100	Programmable Pulse Generator 4	-
10101	0110	101	Programmable Pulse Generator 5	-

Table 2-2 Settings for Extended Transfer Request Sources

IS	EIS	RN	Function	Transfer stop request
10110	0110	102	Programmable Pulse Generator 6	-
10111	0110	103	Programmable Pulse Generator 7	-
11000	0110	104	Programmable Pulse Generator 8	-
11001	0110	105	Programmable Pulse Generator 9	-
11010	0110	106	Programmable Pulse Generator 10	-
11011	0110	107	Programmable Pulse Generator 11	-
11100	0110	108	Programmable Pulse Generator 12	-
11101	0110	109	Programmable Pulse Generator 13	-
11110	0110	110	Programmable Pulse Generator 14	-
11111	0110	111	Programmable Pulse Generator 15	-
10000	0111	112	ADC 0	-
10001	0111	113	res.	-
10010	0111	114	res.	-

- When reset: IS4-0 is initialized to 00000_B.
- When reset: EIS3-0 is initialized to 0000_B.
- These bits are readable and writable.

[Bits 19 to 16] BLK3 to 0 (BLocK size): Block size specification

These bits specify the block size for block transfer on the corresponding channel. The value specified by these bits becomes the number of words in one transfer unit (more exactly, the repetition count of the data width setting). If block transfer will not be performed, set 01_H (size 1). This register value is ignored during demand transfer. The size becomes 1.

BLK	Function
$XXXX_B$	Block size of the corresponding channel

- When reset: Not initialized.
- These bits are readable and writable.
- If 0 is specified for all bits, the block size becomes 16 words.
- During reading, the block size is always read (reload value).

[Bits 15 to 00] DTC (Dma Terminal Count register)*: Transfer count register

The DTC register stores the transfer count. Each register has 16-bit length.

All registers have a dedicated reload register. When the register is used for a channel that is enabled to reload the transfer count register, the initial value is automatically written back to the register when the transfer is

completed.

DTC	Function
$XXXX_B$	Transfer count for the corresponding channel

When DMA transfer is started, data in this register is stored in the counter buffer of the DMA-dedicated transfer counter and is decremented by 1 (subtraction) after each transfer unit. When DMA transfer is completed, the contents of the counter buffer are written back to this register and then DMA ends. Thus, the transfer count value during DMA operation cannot be read.

- When reset: Not initialized.
- These bits are readable and writable. Always access DTC using halfword length or word length.
- During reading, the count value is read. The reload value cannot be read.

2.2 Control/Status Registers B (DMACB0 to 4)

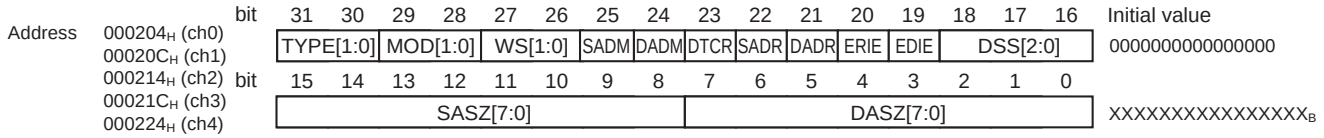
Control/status registers B (DMACB0 to 4) control the operation of each DMAC channel and exist independently for each channel.

This section describes the configuration of control/status registers B (DMACB0 to 4) and their functions.

■ Bit Configuration of Control/Status Register B (DMACB0 to 4)

Figure 2-3"Bit Configuration of Control/Status Registers B (DMACB0 to 4)" shows the bit configuration of control/status registers B (DMACB0 to 4).

Figure 2-3 Bit Configuration of Control/Status Registers B (DMACB0 to 4)



■ Detailed Bit of Control/Status Register B (DMACB0 to 4)

The following describeds the functions of the bits of control status register B (DMACB0 to 4).

[Bits 31 to 30] TYPE (TYPE)*: Transfer type setting

These bits are the transfer type setting bits and set the type of operation for the corresponding channel.

- 2-cycle transfer mode: In this mode, the transfer source address (DMASA) and transfer destination address (DMADA) are set and transfer is performed by repeating the read operation and write operation for the number of times specified by the transfer count. All areas can be specified as a transfer source or transfer destination (32-bit address).
- Fly-by transfer mode: In this mode, external <--> external transfer is performed in one cycle by setting a memory address as the transfer destination address (DMADA). Be sure to specify an external area for the

memory address.

Table 2-3 Settings for the Transfer Types

TYPE	Function
00_B	2-cycle transfer (initial value)
01_B	Fly-by: Memory --> I/O transfer
10_B	Fly-by: I/O --> memory transfer
11_B	Setting disabled

- When reset: Initialized to 00_B.
- These bits are readable and writable.

[Bits 29, 28] MOD (MODe)*: Transfer mode setting

These bits are the transfer mode setting bits and set the operating mode of the corresponding channel.

Table 2-4 Settings for Transfer Modes

MOD	Function
00_B	Block/step transfer mode (initial value)
01_B	Burst transfer mode
10_B	Demand transfer mode
11_B	Setting disabled

- When reset: Initialized to 00_B.
- These bits are readable and writable.

[Bits 27 to 26] WS (Word Size)*: Transfer data width selection

These bits are the transfer data width selection bits and are used to select the transfer data width of the corresponding channel. Transfer operations are repeated in units of the data width specified in this register for as many times as the specified count.

Table 2-5 Selection of the Transfer Data Width

WS	Function
00_B	Byte-width transfer (initial value)
01_B	Halfword-width transfer
10_B	Word-width transfer
11_B	Setting disabled

- When reset: Initialized to 00_B.
- These bits are readable and writable.

[Bit 25] SADM (Source-ADdr. Count-Mode select)*: Transfer source address count mode specification

This bit specifies the address processing of the transfer source address of the corresponding channel in each transfer operation.

An address increment is added or an address decrement is subtracted after each transfer operation according to the specified transfer source address count width (SASZ). When the transfer is completed, the next access address is written to the corresponding address register (DMASA).

As a result, the transfer source address register is not updated until DMA transfer is completed.

To make the address always the same, specify 0 or 1 for this register and make the address count width (SAAZ and DASZ) equal to 0.

SADM	Function
0	Increments transfer source address. (initial value)
1	Decrements the transfer source address.

- When reset: Initialized to 0.
- This bit is readable and writable.

[Bit 24] DADM (Destination-ADdr. Count-Mode select)*: Transfer destination address count mode specification

This bit specifies the address processing for the transfer destination address of the corresponding channel in each transfer operation.

An address increment is added or an address decrement is subtracted after each transfer operation according to the specified transfer destination address count width (DASZ). When the transfer is completed, the next access address is written to the corresponding address register (DMADA).

As a result, the transfer destination address register is not updated until the DMA transfer is completed.

To make the address always the same, specify 0 or 1 for this register and make the address count width (SASZ, DASZ) equal to 0.

DADM	Function
0	Increments the transfer source address. (initial value)
1	Decrements the transfer source address.

- When reset: Initialized to 0.
- This bit is readable and writable.

[Bit 23] DTCR (DTC-reg. Reload)*: Transfer count register reload specification

This bit controls reloading of the transfer count register for the corresponding channel.

If reload operation is enabled by this bit, the count register value is restored to its initial value after the transfer is completed then DMAC stops and then waiting starts for new transfer requests (an activation request by STRG or IS setting). If this bit is 1, the DENB bit is not cleared.

DENB=0 or DMAE=0 must be set to stop the transfer. In either case, the transfer is forcibly stopped.

If reloading of the counter is disabled, a single shot operation occurs. In single shot operation, operation stops after the transfer is completed even if reload is specified in the address register. The DENB bit is also cleared in this case.

DTCR	Function
0	Disables transfer count register reloading (initial value)
1	Enables transfer count register reloading.

- When reset: Initialized to 0.
- This bit is readable and writable.

[Bit 22] SADR (Source-ADdr.-reg. Reload)*: Transfer source address register reload specification

This bit controls reloading of the transfer source address register for the corresponding channel.

If this bit enables the reload operation, the transfer source address register value is restored to its initial value after the transfer is completed.

If reloading of the counter is disabled, a single shot operation occurs. In single shot operation, operation stops after the transfer is completed even if reload is specified in the address register. The address register value also stops in this case while the initial value is being reloaded.

If this bit disables the reload operation, the address register value when the transfer is completed is the address to be accessed next to the final address. When address increment is specified, the next address is an incremented address.

SADR	Function
0	Disables transfer source address register reloading. (initial value)
1	Enables transfer source address register reloading.

- When reset: Initialized to 0.
- This bit is readable and writable.

[Bit 21] DADR (Dest.-ADdr.-reg. Reload)*: Transfer destination address register reload specification

This bit controls reloading of the transfer destination address register for the corresponding channel.

If this bit enables reloading, the transfer destination address register value is restored to its initial value after the transfer is completed.

The details of other functions are the same as those described for Bit22 (SADR).

DADR	Function
0	Disables transfer destination address register reloading. (initial value)
1	Enables transfer destination address register reloading.

- When reset: Initialized to 0.
- This bit is readable and writable.

[Bit 20] ERIE (ERror Interrupt Enable)*: Error interrupt output enable

This bit controls the occurrence of an interrupt for termination after an error occurs. The nature of the error that occurred is indicated by DSS2 to 0. Note that an interrupt occurs only for specific termination causes and not for all termination causes (Refer to bits DSS2 to 0, which are Bits 18 to 16).

ERIE	Function
0	Disables error interrupt request output. (initial value)
1	Enables error interrupt request output.

- When reset: Initialized to 0.
- This bit is readable and writable.

[Bit 19] EDIE (EnD Interrupt Enable)*: End interrupt output enable

This bit controls the occurrence of an interrupt for normal termination.

EDIE	Function
0	Disables end interrupt request output. (initial value)
1	Enables end interrupt request output.

- When reset: Initialized to 0.
- This bit is readable and writable.

[Bits 18 to 16] DSS2 to 0 (DMA Stop Status)*: Transfer stop source indication

These bits indicate a code (end code) of 3 bits that indicates the source of stopping or termination of DMA transfer on the corresponding channel. For a list of end codes, see Table 2-6"End Codes".

Table 2-6 End Codes

DSS	Function	Interrupt
000_B	Initial value	None
$x01_B$	Address error (underflow/overflow)	Error
$x10_B$	Transfer stop request	Error
$x11_B$	Normal end	End
$1xx_B$	DMA stopped temporarily (due, for example, to DMAH, PAUS bit, and an interrupt)	None

A transfer stop request is set only when it is requested by a peripheral device or the external pin DSTP function is used.

The Interrupt column indicates the type of interrupts that can occur.

- When reset: Initialized to 000_B.
- These bits can be cleared by writing 000_B to them.
- These bits are readable and writable. Note that the only valid written value is 000.

[Bits 15 to 8] SASZ (Source Addr count SiZe)*: Transfer source address count size specification

These bits specify the increment or decrement width for the transfer source address (DMASA) of the corresponding channel in each transfer operation. The value set by these bits becomes the address increment/decrement for each transfer unit. The address increment/decrement conforms to the instruction in the transfer source address count mode (SADM).

SASZ	Function
XX_H	Specify the increment/decrement width of the transfer source address. 0 to 255

- When reset: Not initialized
- These bits are readable and writable.

[Bits 7 to 0] DASZ (Des Addr count SiZe)*: Transfer destination address count size specification

These bits specify the increment or decrement width for the transfer destination address (DMADA) of the corresponding channel in each transfer operation. The value set by these bits becomes the address increment/decrement for each transfer unit. The address increment/decrement conforms to the instruction in the transfer destination address count mode (DADM).

DASZ	Function
XX_H	Specify the increment/decrement width of the transfer destination address. 0 to 255

- When reset: Not initialized
- These bits are readable and writable.

2.3 Transfer Source/Transfer Destination Address Setting Registers (DMASA0 to 4/ DMADA0 to 4)

The transfer source/transfer destination address setting registers (DMASA0 to 4/DMADA0 to 4) control the operation of the DMAC channels. There is a separate register for each channel.

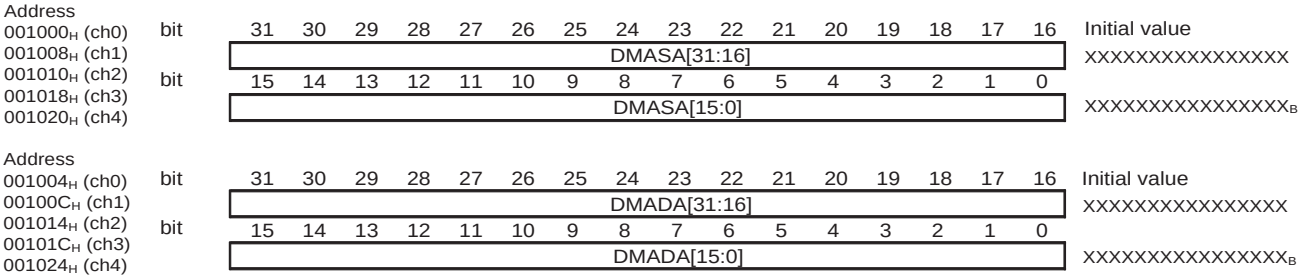
This section describes the configuration and functions of the transfer source/transfer destination address setting registers (DMASA0 to 4/DMADA0 to 4).

■ Bit Configuration of Transfer Source/Transfer Destination Address Setting Registers (DMASA0 to 4/DMADA0 to 4)

The transfer source/transfer destination address setting registers (DMASA0 to 4/DMADA0 to 4) are a group of registers that store the transfer source/transfer destination addresses. Each register is 32 bits length.

Figure 2-4"Bit Configuration of the Transfer Source/Transfer Destination Address Setting Registers (DMASA0 to 4/DMADA0 to 4)" shows the bit configuration of the transfer source/transfer destination address setting registers (DMASA0 to 4/DMADA0 to 4).

Figure 2-4 Bit Configuration of the Transfer Source/Transfer Destination Address Setting Registers (DMASA0 to 4/DMADA0 to 4)



Detailed Bit of Transfer Source/Transfer
Destination Address Setting Register (DMASA0 to 4/DMADA0 to 4)

The following describes the functions of the bits of each transfer source/transfer destination address setting register (DMASA0 to 4/DMADA0 to 4).

[Bits 31 to 0] DMASA (DMA Source Addr)*: Transfer source address setting

These bits set the transfer source address.

[Bits 31 to 0] DMADA (DMA Destination Addr)*: Transfer destination address setting

These bits set the transfer destination address.

If DMA transfer is activated, data in this register is stored in the counter buffer of the DMA-dedicated address counter and then the address is calculated according to the settings for the transfer operation. When the DMA transfer is completed, the contents of the counter buffer are written back to this register and then DMA ends. Thus, the address counter value during DMA operation cannot be read.

All registers have a dedicated reload register. When the register is used for a channel that is enabled for reloading of the transfer source/transfer destination address register, the initial value is automatically written back to the register when the transfer is completed. Other address registers are not affected.

- When reset: Not initialized.
- These bits are readable and writable. For this register, be sure to access these bits as 32-bit data.
- If these bits are read during transfer, the address before the transfer is read. If they are read after transfer, the next access address is read. Because the reload value cannot be read, it is not possible to read the transfer address in real time.

Note:

Do not set any of the DMAC's registers using this register. DMA transfer is not possible for the DMAC's registers themselves.

2.4 DMAC All-Channel Control Register (DMACR)

The DMAC all-channel control register (DMACR) controls the operation of the all five DMAC channels. Be sure to access this register using byte length.

This section describes the configuration and functions of the DMAC all-channel control register (DMACR).

■ Bit Configuration of DMAC All-Channel Control Register (DMACR)

Figure 2-5"Bit Configuration of the DMAC All-Channel Control Register (DMACR)" shows the bit configuration of the DMAC all-channel control register (DMACR).

Figure 2-5 Bit Configuration of the DMAC All-Channel Control Register (DMACR)

		bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	Initial value
Address	000240_H		DMAE	-	-	PM01	DMAH[3:0]				-	-	-	-	-	-	-	-	0XX00000XXXXXXXX
		bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
			-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	$XXXXXXXXXXXXXXXX_B$

■ Detailed Bit of DMAC All-Channel Control Register (DMACR)

The following describes the bit functions of the DMAC all-channel control register (DMACR) bits.

[Bit 31] DMAE (DMA Enable): DMA operation enable

This bit controls the operation of all DMA channels.

If DMA operation is disabled with this bit, transfer operations on all channels are disabled regardless of the start/stop settings for each channel and the operating status. Any channel carrying out transfer cancels the requests and stops transfer at a block boundary. All start operations on each channel in a disabled state are disabled.

If this bit enables DMA operation, start/stop operations are enabled for each channels. Simply enabling DMA operation with this bit does not activate each channel.

DMA operation can be forced to stop by writing 0 to this bit. However, be sure to force stopping (0 write) only after temporarily stopping DMA using the DMAH[3:0] bits [Bit27 to 24 of DMACR]. If forced stopping is carried out without first temporarily stopping DMA, DMA stops, but the transfer data cannot be guaranteed. Check whether DMA is stopped using the DSS[2:0] bits [Bit18 to 16 of DMACB].

DMAE	Function
0	Disables DMA transfer on all channels. (initial value)
1	Enables DMA transfer on all channels.

- When reset: Initialized to 0.
- This bit is readable and writable.

[Bit 28] PM01 (Priority mode ch0,1 robine): Channel priority rotation

This bit is set to alternate priority for each transfer between Channel0 and Channel1.

PM01	Function
0	Fixes the priority. (ch0 > ch1)(initial value)
1	Alternates priority. (ch1 > ch0)

- When reset: Initialized to 0.
- This bit is readable and writable.

[Bits 27 to 24] DMAH (DMA Halt): DMA temporary stop

These bits control temporary stopping of all DMA channels. If these bits are set, DMA transfer is not performed on any channel before these bits are cleared.

When DMA transfer is activated after these bits are set, all channels remain temporarily stopped.

Transfer requests that occur on channels for which DMA transfer is enabled (DENB=1) while these bits are set are all enabled. The transfer can be started by clearing all these bits.

DMAH	Function
0000_B	Enables the DMA operation on all channels. (initial value)
Other than 0000_B	Temporarily stops DMA operation on all channels.

- When reset: Initialized to 0.
- These bits are readable and writable.

[Bits 30, 29, and 23 to 0] (Reserved): Unused bits

These bits are unused.

- A read value is undefined.

2.5 Other Functions

The MB91460 series has the DACK, DEOP, and DREQ pins, which can be used for external transfer. These pins can also be used as general-purpose ports.

■ Pin Function of the DACK, and DEOP, and DREQ pins

To use the DACK, DEOP, or DREQ pins for external transfer, a switch must be made from the port function to the DMA pin function.

To make the switch, set the PFR register.

3. DMA Controller (DMAC) Operation

A DMA controller (DMAC) is built into all FR family devices. The FR family DMAC is a multi-functional DMAC that controls data transfer at high speed without the use of CPU instructions.

This section describes the operation of the DMAC.

■ Principal Operations

- Functions can be set for each transfer channel independently.
- Once starting has been enabled, a channel starts transfer operation only after a specified transfer request has been detected.
- After a transfer request is detected, a DMA transfer request is output to the bus controller and the bus right is acquired by the bus controller before the transfer is started.
- The transfer is carried out as a sequence conforming to the mode settings made independently for the channel being used.

■ Transfer Mode

Each DMA channel performs transfer according to the transfer mode set by the MOD[1:0] bits of its DMACB register.

● Block/step transfer

Only a single block transfer unit is transferred in response to one transfer request. DMA then stops requesting the bus controller for transfer until the next transfer request is received.

The block transfer unit is the specified block size (BLK[3:0] of DMACA).

● Burst transfer

Transfer in response to one transfer request is carried out continuously for the number of times in the specified transfer count is reached.

The specified transfer count is the transfer count (BLK[3:0] of DMACA X DTC[15:0] of DMACA) X block size.

● Demand transfer

Transfer is carried out continuously until the transfer request input (detected with a level at the DREQ pin) from an external device or a specified transfer count is reached.
The specified transfer count in a demand transfer is the specified transfer count (DTC[15:0] of DMACA). The block size is always 1 and the register value is ignored.

■ Transfer Type

● 2-cycle transfer (normal transfer)

The DMA controller operates using a read operation and a write operation as its unit of operation.

Data is read from an address in the transfer source register and then written to another address in the transfer destination register.

● Fly-by transfer (memory --> I/O)

The DMA controller operates using a read operation as its unit of operation.

If DMA transfer is performed when fly-by transfer is set, DMA issues a fly-by transfer (read) request to the bus controller and the bus controller lets the external interface carry out the fly-by transfer (read).

- Fly-by transfer (I/O --> memory)

The DMA controller operates using a write operation as its unit of operation.

Otherwise, operation is the same as fly-by transfer (memory --> I/O) operation.

Access areas used for MB91460 series fly-by transfer must be external areas.

■ Transfer Address

The following types of addressing are available and can be set independently for each channel transfer source and transfer destination.

The method for specifying the address setting register (DMASA/DMADA) for a 2-cycle transfer and the method for a fly-by transfer are different.

- Specifying the address for a 2-cycle transfer

The value read from a register (DMASA/DMADA) in which an address has been set in advance is used as the address for access. After receiving a transfer request, DMA stores the address from the register in the temporary storage buffer and then starts transfer.

After each transfer (access) operation, the next access address is generated (increment/decrement/fixed selectable) by the address counter and then written to the temporary storage buffer. Because the contents of the temporary storage buffer are written back to the register (DMASA/DMADA) after each block transfer unit is completed, the address register (DMASA/DMADA) value is updated after each block transfer unit is completed, making it impossible to determine the address in real time during transfer.

- Specifying the address for a fly-by transfer

In a fly-by transfer, the value read from the transfer destination address register (DMADA) is used as the address for access. The transfer source address register (DMASA) is ignored. Be sure to specify an external area as the address to be set.

After receiving a transfer request, DMA stores the address from the register in the temporary storage buffer and then starts transfer.

After each transfer (access) operation, the next access address is generated (increment/decrement/fixed selectable) by the address counter and then written to the temporary storage buffer. Because the contents of this temporary storage buffer are written back to the register (DMADA) after each block transfer unit is completed, the address register (DMADA) value is updated after each block transfer unit is completed, making it impossible to determine the address in real time during transfer.

■ Transfer Count and Transfer End

- Transfer count

The transfer count register is decremented (-1) after each block transfer unit is completed. When the transfer count register becomes 0, counting for the specified transfer ends, and the transfer stops with the end code displayed or is reactivated *.

Like the address register, the transfer count register value is updated only after each block transfer unit.

*: If transfer count register reloading is disabled, the transfer ends. If reloading is enabled, the register value is initialized and then waits for transfer (DTCR of DMACB)

- Transfer end

Listed below are the sources for transfer end. When transfer ends, a source is indicated as the end code (DSS[2:0] of DMACB).

- End of the specified transfer count (DMACA:BLK[3:0] x DMACA:DTC[15:0]) => Normal end
- A transfer stop request from a peripheral circuit or the external pin (DSTP) occurred => Error
- An address error occurred => Error
- A reset occurred => Reset

The transfer stop source is indicated (DSS) and the transfer end interrupt or error interrupt for the end source is generated.

3.1 Setting a Transfer Request

The following three types of transfer requests are provided to activate DMA transfer:

- External transfer request pin
- Built-in peripheral request
- Software request

Software requests can always be used regardless of the settings of other requests.

■ External Transfer Request Pin

A transfer request is generated by input to the input pin prepared for a channel.

The MB91460 series supports channels 0-3 (DREQ0-3).

If the input is valid at this point, the following sources are selected depending on the settings for the transfer type and the start source:

● Edge detection

If the transfer type is block, step, or burst transfer, select edge detection:

- Falling edge detection: Set with the transfer source selection register. When IS[4:0] of DMACA=01110_B.
- Rising edge detection: Set with the transfer source selection register. When IS[4:0] of DMACA=01111_B.

● Level detection

If the transfer type is demand transfer, select level detection:

- H level detection: Set with the transfer source selection register. When IS[4:0] of DMACA=01110_B.
- L level detection: Set with the transfer source selection register. When IS[4:0] of DMACA=01111_B.

■ Built-in Peripheral Request

A transfer request is generated by an interrupt from the built-in peripheral circuit.

For each channel, set the peripheral's interrupt by which a transfer request is generated (When IS[4:0] of DMACA=1xxxx.)

The built-in peripheral request cannot be used together with an external transfer request.

Note:

Because an interrupt request used in a transfer request seems like an interrupt request to the CPU, disable interrupts from the interrupt controller (ICR register).

■ Software Request

A transfer request is generated by writing to the trigger bit of a register (STRG of DMACA).

The software request is independent of the external transfer request pin and built-in peripheral request and can

always be caused.

If a software request occurs together with a start (transfer enable) request, the transfer is started by immediate output of a DMA transfer request to the bus controller.

3.2 Transfer Sequence

The transfer type and the transfer mode that determine, for example, the operation sequence after DMA transfer has started can be set independently for each channel (Settings for TYPE[1:0] and MOD[1:0] of DMACB).

■ Selection of the Transfer Sequence

The following sequence can be selected with a register setting:

- Burst 2-cycle transfer
- Demand 2-cycle transfer
- Block/step 2-cycle transfer
- Burst fly-by transfer
- Demand fly-by transfer
- Block/step fly-by transfer

● Burst 2-cycle transfer

In a burst 2-cycle transfer, as many transfers as specified by the transfer count are performed continuously for one transfer source. For a 2-cycle transfer, all 32-bit areas can be specified using a transfer source/transfer destination address.

A peripheral transfer request, software transfer request, or external pin (DREQ) edge input detection request can be selected as the transfer source.

Table 3-1 Specifiable transfer addresses (burst 2-cycle transfer)

Transfer source addressing	Direction	Transfer destination addressing
All 32-bit areas specifiable	=>	All 32-bit areas specifiable

The following are some features of a burst transfer:

When one transfer request is received, transfer is performed continuously until the transfer count register reaches 0.

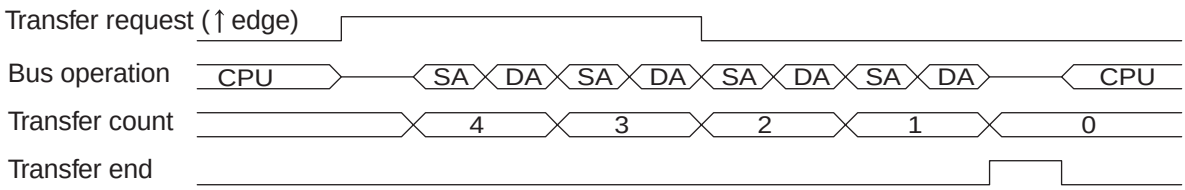
The transfer count is the transfer count x block size (BLK[3:0] of DMACA x DTC[15:0] of DMACA).

Another request occurring during transfer is ignored.

If the reload function of the transfer count register is enabled, the next request is accepted after transfer ends.

If a transfer request for another channel with a higher priority is received during transfer, the channel is switched at the boundary of the block transfer unit. Processing resumes only after the transfer request for the other channel is cleared.

Figure 3-1 Example of burst transfer for a start on an external pin rising edge, number of blocks =1, and transfer count = 4



● Burst fly-by transfer

A burst fly-by transfer has the same features as a 2-cycle transfer except that the transfer area can only be external areas, and the transfer unit is read (memory --> I/O) or write (I/O --> memory) only.

Table 3-2 Specifiable transfer addresses (burst fly-by transfer)

Transfer source addressing	Direction	Transfer destination addressing
Specification not required (invalid)	None	External area

● Demand Transfer 2-Cycle Transfer

A demand transfer sequence is generated only if H level or L level of an external pin is selected as a transfer request. Select the level with IS[3:0] of DMACA.

The following are some features of a continuous transfer:

The following are some features of a continuous transfer:

- Each transfer operation of a transfer request is checked. While the external input level is within the range of the specified transfer request levels, transfer is performed continuously without the request being cleared. If the external input changes, the request is cleared and the transfer stops at the transfer boundary. This operation is repeated for the number of times specified by the transfer count.
- Otherwise, operations are the same as those of a burst transfer.

Figure 3-2 Example of demand transfer for a start with the external pin at H level, number of blocks = 1, and transfer count = 3

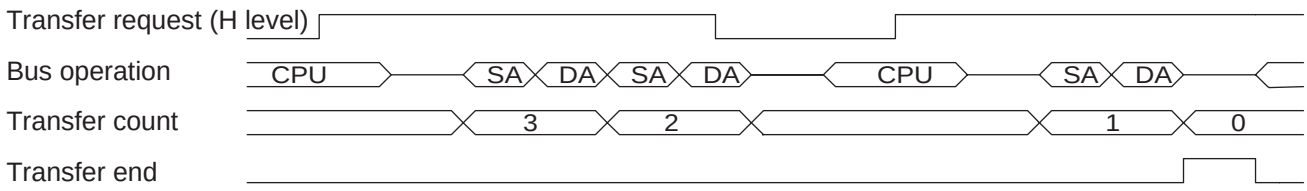


Table 3-3 Specifiable transfer addresses (demand transfer 2-cycle transfer)

Transfer source address	Direction	Transfer destination addressing
External area	=>	External area
External area	=>	Built-in IO
External area	=>	Built-in RAM
Built-in IO	=>	External area
Built-in RAM	=>	External area

Note:

For a demand transfer, be sure to set an external area address for the transfer source or transfer destination or both. Since DMA transfer is adjusted to the external bus timing in demand transfer mode, access to external areas is always needed.

- Demand transfer fly-by transfer

A demand transfer fly-by transfer has the same features as a 2-cycle transfer except that the transfer area can only be external areas, and the transfer unit is read (memory --> I/O) or write (I/O --> memory) only.

Table 3.2-3 Specifiable transfer addresses (demand transfer fly-by transfer)

Transfer source addressing	Direction	Transfer destination addressing
Specification not required (invalid)	=>	External area

- Step/block transfer 2-cycle transfer

For a step/block transfer (Transfer for each transfer request is performed as many times as the specified block count), all 32-bit areas can be specified as the transfer source/transfer destination address.

Table 3-4 Specifiable transfer addresses (step/block transfer 2-cycle transfer)

Transfer source addressing	Direction	Transfer destination addressing
All 32-bit areas specifiable	=>	All 32-bit areas specifiable

[Step transfer]

If 1 is set as the block size, a step transfer sequence is generated.

The following are some features of a step transfer:

- If a transfer request is received, the transfer request is cleared after one transfer operation and then the transfer is stopped (The DMA transfer request to the bus controller is canceled).
- Another request occurring during transfer is ignored.

- If a transfer request for another channel with a higher priority is received during transfer, the channel is switched after the transfer is stopped and then restarted. Priority in a step transfer is valid only if transfer requests occur simultaneously.

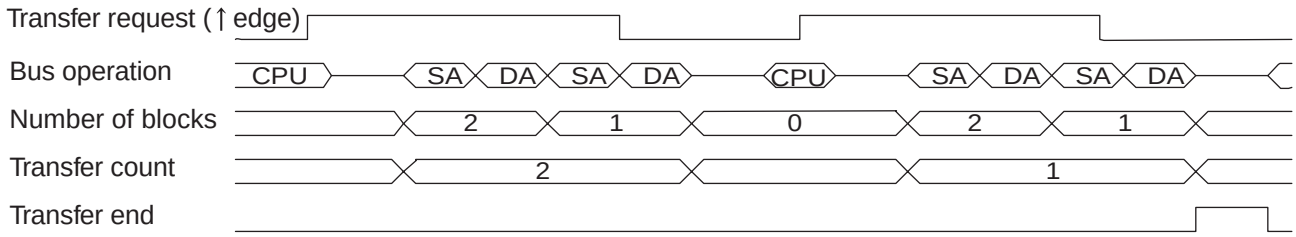
[Block transfer]

If any value other than 1 is specified as the block size, a block transfer sequence is generated.

The following are some features of a block transfer:

- The block transfer has the same features as those of a step transfer except that one transfer unit consists of multiple transfer cycle counts (number of blocks).

Figure 3-4 Example of block transfer for a start for an external pin on a rising edge, number of blocks = 2, and transfer count = 2



● Step/block transfer fly-by transfer

This transfer has the same features as those of a 2-cycle transfer except that the transfer area can only be external areas, and the transfer unit is read (memory --> I/O) or write (I/O --> memory) only.

Table 3-5 Specifiable transfer addresses (step/block transfer fly-by transfer)

Transfer source addressing	Direction	Transfer destination addressing
Specification not required (invalid)	None	External area

3.3 General Aspects of DMA Transfer

This section describes the block size for DMA transfers and the reload operation.

■ Block Size

- The unit and increment for transfer data is a set of (the number set in the block size specification register x data width) data.
- Since the amount of data transferred in one transfer cycle is determined by the value specified as the data width, one transfer unit is consists of the number of transfer cycles for the specified block size.
- If a transfer request with a higher priority is received during transfer or if a temporary stop request for a transfer occurs, the transfer stops only at the transfer unit boundary, whether or not the transfer is a block transfer. This arrangement makes it possible to protect data for which division or temporary stopping is not desirable. However, if the block size is large, response time decreases.
- Transfer stops immediately only when a reset occurs, in which case the data being transferred cannot be guaranteed.

■ Reload Operation

In this module, the following three types of reloading can be set for each channel:

● Transfer count register reloading

After transfer is performed the specified number of times, the initial value is set in the transfer count register again and waiting for a start request starts.

Set this type of reloading when the entire transfer sequence is to be performed repeatedly.

If reload is not specified, the count register value remains 0 after the transfer is performed the specified number of times and no further transfer is performed.

● Transfer source address register reloading

After transfer is performed the specified number of times, the initial value is set in the transfer source address register again.

Set this type of reloading when transfer is to be repeated from a fixed area in the transfer source address area.

If reload is not specified, the transfer source address register value after the transfer is performed the specified number of times becomes the next address. Use this type when the address area is not fixed.

● Transfer destination address register reloading

After transfer is performed the specified number of times, the initial value is set in the transfer destination address register again.

Set this type of reloading when transfer is to be repeated to a fixed area in the transfer destination address area.

(The processing hereafter is the same as described in "Transfer source address register reloading" above.)

- If only reloading of the transfer source/transfer destination register is enabled, restart after transfer is performed the specified number of times is not implemented and only the values of each address register are set.

● Special examples of operating mode and the reload operation

- If transfer is performed in continuous transfer mode by external pin input level detection and transfer count register reloading is used, transfer continues by reloading even though transfer ends during continuous input. Also in this case, an end code is set.
- If it is preferable that processing stops when data transfer ends and starts after input is detected again, do not specify reload.
- For a transfer in burst, block, or step transfer mode, transfer stops temporarily after reload when data transfer ends. Transfer does not start until new transfer request input is detected.

3.4 Addressing Mode

Specify the transfer destination/transfer source address independently for each transfer channel.

■ Address Register Specifications

The following two methods are provided to specify an address register. The method specified depends on the transfer sequence.

- In 2-cycle transfer mode, set the transfer source address in the transfer source address setting register (DMASA) and the transfer destination address in the transfer destination address setting register (DMADA).
- In fly-by transfer mode, specify the memory address in the transfer destination address setting register (DMADA). In this case, the value in the transfer source address setting register (DMASA) is ignored.

■ Features of the Address Register

This register has the maximum 32-bit length. With 32-bit length, all space in the memory map can be accessed.

■ Function of the Address Register

- The address register is read in each access operation and the read value is sent to the address bus.
- At the same time, the address for the next access is calculated by the address counter and the address register is updated using the calculated address.
- For address calculation, increment or decrement is selected independently for each channel, transfer destination, and transfer source. The address increment/decrement width is specified by the address count size register (SASZ/DASZ of DMACB).
- If reloading is not enabled, the address resulting from the address calculation of the last address remains in the address register when the transfer ends.
- If reloading is enabled, the initial value of the address is reloaded.

Notes:

- If an overflow or underflow occurs as a result of 32-bit length full address calculation, an address error is detected and transfer on the relevant channel is stopped. Refer to the description for the items related to the end code.
- Do not set any of the DMAC's registers as the address register.
- For demand transfer, be sure to set an address in an external area for the transfer source, transfer destination, or both.
- Do not let the DMAC transfer data to any of the DMAC's registers.

3.5 Data Types

Select the data length (data width) transferred in one transfer operation from the following:

- Byte
- Halfword
- Word

■ Data Length (Data width)

Since the word boundary specification is also observed in DMA transfer, different low-order bits are ignored if an address with a different data length is specified for the transfer destination/transfer source address.

- Byte: The actual access address and the addressing match.
- Halfword: The actual access address has 2-byte length starting with 0 as the lowest-order bit.
- Word: The actual access address has a 4-byte length starting with 00 as the lowest-order 2 bits.

If the lowest-order bits in the transfer source address and transfer destination address are different, the addresses as set are output on the internal address bus. However, each transfer target on the bus is accessed after the addresses are corrected according to the above rules.

3.6 Transfer Count Control

Specify the transfer count within the range of the maximum 16-bit length (1 to 65536).

■ Transfer Count Control

Set the transfer count value in the transfer count register (DTC of DMACA).

The register value is stored in the temporary storage buffer when the transfer starts and is decremented by the transfer counter. When the counter value becomes 0, end of transfer end for the specified count is detected, and the transfer on the channel is stopped or waiting for a restart request starts (when reload is specified).

The following are some features of the group of transfer count registers:

- Each register has 16-bit length.
- All registers have a dedicated reload register.
- If transfer is activated when the register value is 0, transfer is performed 65536 times.

■ Reload Operation

- The reload operation can be used only if reloading is enabled in a register that allows reloading.
- When transfer is activated, the initial value of the count register is saved in the reload register.
- If the transfer counter counts down to 0, end of transfer is reported and the initial value is read from the reload register and written to the count register.

3.7 CPU Control

When a DMA transfer request is accepted, DMA issues a transfer request to the bus controller.

The bus controller passes the right to use the internal bus to DMA at a break in bus operation and DMA transfer starts.

■ DMA Transfer and Interrupts

- During DMA transfer, interrupts are generally not accepted until the transfer ends.
- If a DMA transfer request occurs during interrupt processing, the transfer request is accepted and interrupt processing is stopped until the transfer is completed.
- If, as an exception, an NMI request or an interrupt request with a higher level than the hold suppress level set by the interrupt controller occurs, DMAC temporarily cancels the transfer request via the bus controller at a transfer unit boundary (one block) to temporarily stop the transfer until the interrupt request is cleared. In the meantime, the transfer request is retained internally. After the interrupt request is cleared, DMAC reissues a transfer request to the bus controller to acquire the right to use the bus and then restarts DMA transfer.

■ Suppressing DMA

When an interrupt source with a higher priority occurs during DMA transfer, an FR family device interrupts the DMA transfer and branches to the relevant interrupt routine. This feature is valid as long as there are any interrupt requests. When all interrupt sources are cleared, the suppression feature no longer works and the DMA transfer is restarted by the interrupt processing routine. Thus, if you want to suppress restart of DMA transfer after clearing interrupt sources in the interrupt source processing routine at a level that interrupts DMA transfer, use the DMA suppress function. The DMA suppress function can be activated by writing any value other than 0 to the DMAH[3:0] bits of the DMA all-channel control register and can be stopped by writing 0 to these bits.

This function is mainly used in the interrupt processing routines. Before the interrupt sources in an interrupt processing routine are cleared, the DMA suppress register is incremented by 1. If this is done, then no DMA transfer is performed. After interrupt processing, decrement the DMAH[3:0] bits by 1 before returning. If multiple interrupts have occurred, DMA transfer continues to be suppressed since the DMAH[3:0] bits are not 0 yet. If a single interrupt has occurred, the DMAH[3:0] bits become 0. DMA requests are then enabled immediately.

Note:

- Since the register has only four bits, this function cannot be used for multiple interrupts exceeding 15 levels.
- Be sure to assign the priority of the DMA tasks at a level that is at least 15 levels higher than other interrupt levels.

3.8 Hold Arbitration

When a device is operating in external bus extended mode, an external hold function can be used. The relationship between external hold requests and DMA transfer requests by this module when the hold function can be used is described below.

■ DMA Transfer Request during External Hold

DMA transfer is started when an external bus area is accessed, DMA transfer is temporarily stopped. When the external hold is released, DMA transfer is restarted.

■ External Hold Request During DMA Transfer

The device is externally held. When an external bus area is accessed by DMA transfer, DMA transfer is temporarily stopped. When the external hold is released, DMA transfer is restarted.

■ Simultaneous Occurrence of a DMA Transfer Request and an External Hold Request

The device is externally held and internal DMA transfer is started. When an external bus area is accessed by DMA transfer, DMA transfer is temporarily stopped. When the external hold is released, DMA transfer is restarted.

3.9 Operation from Starting to End/Stopping

Starting of DMA transfer is controlled independently for each channel, but before transfer starts, the operation of all channels needs to be enabled. This section describes operation from starting to end/stopping.

■ Operation Start

- Enabling operation for all channels

Before activating each DMAC channel, operation for all channels needs to be enabled in advance with the DMA operation enable bit (DMAE of DMACR). All start settings and transfer requests that occurred before operation is enabled are invalid.

- Starting transfer

The transfer operation can be started by the operation enable bit of the control register for each channel. If a transfer request to an activated channel is accepted, the DMA transfer operation is started in the specified mode.

- Starting from a temporary stop

If a temporary stop occurs before starting with channel-by-channel or all-channel control, the temporary stopped state is maintained even though the transfer operation is started. If transfer requests occur in the meantime, they are accepted and retained. When temporary stopping is released, transfer is started.

■ Transfer Request Acceptance and Transfer

Sampling for transfer requests set for each channel starts after starting.

If edge detection is selected for the external pin start source and a transfer request is detected, the request is retained within DMAC until the clear conditions are met (when the external pin start source is selected for block, step, or burst transfer).

If level detection or peripheral interrupt start is selected for the external pin start source, DMAC continues the transfer until all transfer requests are cleared. When they are cleared, DMAC stops the transfer after one transfer unit (demand transfer or peripheral interrupt start).

Since peripheral interrupts are handled as level detection, use interrupt clear by DMA to handle the interrupts.

Transfer requests are always accepted while other channel requests are being accepted and transfer performed. The channel that will be used for transfer is determined for each transfer unit after priority has been checked.

■ Clearing Peripheral Interrupts by DMA

This DMA has a function that clears peripheral interrupts. This function works when peripheral interrupt is selected as the DMA start source (when IS[4:0]=1xxxx$_B$).

Peripheral interrupts are cleared only for the set start sources. That is, only the peripheral functions set by IS[4:0] are cleared.

The timing for clearing an interrupt depends on the transfer mode (See Section 4."Operation Flowcharts").

- Block/step transfer: If block transfer is selected, a clear signal is generated after one block (step) transfer.
- Burst transfer: If burst transfer is selected, a clear signal is generated after transfer is performed the specified number of times.
- Demand transfer: Since only start requests from external pins are supported in demand transfer, no clear signal is generated.

■ Temporary Stopping

DMA transfer is stopped temporary in the following cases:

- Setting of temporary stopping by writing to the control register (Set independently for each channel or all channels simultaneously)

If temporary stopping is set using the temporary stop bit, transfer on the corresponding channel is stopped until release of temporary stopping is set again. You can check the DSS bits for temporary stopping.

- NMI/hold suppress level interrupt processing

If an NMI request or an interrupt request with a higher level than the hold suppress level occurs, all channels on which transfer is in progress are temporarily stopped at the boundary of the transfer unit and the bus right is returned to give priority to NMI/interrupt processing. Transfer request accepted during NMI/interrupt processing are retained, initiating a wait for completion of NMI processing.

Channels for which requests are retained restart transfer after NMI/interrupt processing is completed.

■ Operation End/Stopping

The end of DMA transfer is controlled independently for each channel. It is also possible to disable operation for all channels at once.

- Transfer end

If reloading is disabled, transfer is stopped, "Normal end" is displayed as the end code, and all transfer requests are disabled after the transfer count register becomes 0 (Clear the DENB bit of DMACA).

If reloading is enabled, the initial value is reloaded, "Normal end" is displayed as the end code, and a wait for transfer requests starts after the transfer count register becomes 0 (Do not clear the DENB bit of DMACA).

- Disabling all channels

If the operation of all channels is disabled with the DMA operation enable bit DMAE, all DMAC operations, including operations on active channels, are stopped. Then, even if the operation of all channels is enabled again, no transfer is performed unless a channel is restarted. In this case, no interrupt whatever occurs.

■ Stopping Due To an Error

In addition to normal end after transfer for the number of times specified, stopping as the result of various types of errors and the forced stopping are provided.

- Transfer stop requests from peripheral circuits

Depending on the peripheral circuit that outputs a transfer request, a transfer stop request is issued when an error is detected (Example: Error when data is received at or sent from a communications system peripheral).

The DMAC, when it receives such a transfer stop request, displays "Transfer stop request" as the end code and stops the transfer on the corresponding channel.

Table 3-6 Stopping due to an Error

IS	EIS	Function	Transfer stop request
10110	0000	USART 0 RX *1	Yes
11000	0000	USART 1 RX *1	Yes
11010	0000	USART 4 RX *1	Yes
11100	0000	USART 5 RX *1	Yes
10000	0011	USART 0 RX *1	Yes
10010	0011	USART 1 RX *1	Yes
10100	0011	USART 2 RX *1	Yes
10110	0011	USART 3 RX *1	Yes
11000	0011	USART 4 RX *1	Yes
11010	0011	USART 5 RX *1	Yes
11100	0011	USART 6 RX *1	Yes
11110	0011	USART 7 RX *1	Yes
10000	0100	USART 8 RX *1	Yes
10010	0100	USART 9 RX *1	Yes
10100	0100	USART 10 RX *1	Yes
10110	0100	USART 11 RX *1	Yes
11000	0100	USART 12 RX *1	Yes
11010	0100	USART 13 RX *1	Yes
11100	0100	USART 14 RX *1	Yes
11110	0100	USART 15 RX *1	Yes
others	others		None

*1 : A transfer stop request is issued when an error is detected

For details of the conditions under which a transfer stop request is generated, see the specifications for each peripheral circuit.

■ Occurrence of an Address Error

If inappropriate addressing, as shown below in parenthesis, occurs in an addressing mode, an address error is detected (if an overflow or underflow occurs in the address counter when a 32-bit address is specified).

If an address error is detected, "An address error occurred" is displayed as the end code and transfer on the corresponding channel is stopped.

3.10 DMAC Interrupt Control

Independent of peripheral interrupts that become transfer requests, interrupts can also be output for each DMAC channel.

■ DMAC Interrupt Control

The following interrupts can be output for each DMAC channel:

- Transfer end interrupt: Occurs only when operation ends normally.
- Error interrupt: Transfer stop request due to a peripheral circuit (error due to a peripheral)
- Error interrupt: Occurrence of address error (error due to software)

All of these interrupts are output according to the meaning of the end code.

An interrupt request can be cleared by writing 000_B to DSS2 to 0 (end code) of DMACS. Be sure to clear the end code by writing 000_B before restarting.

If reloading is enabled, the transfer is automatically restarted. At this point, however, the end code is not cleared and is retained until a new end code is written when the next transfer ends.

Since only one end source can be displayed in an end code, the result after considering the order of priority is displayed when multiple sources occur simultaneously. The interrupt that occurs at this point conforms to the displayed end code.

The following shows the priority for displaying end codes (in order of decreasing priority):

- Reset
- Clearing by writing 000_B
- Peripheral stop request or external pin input (DSTP) stop request
- Normal end
- Stopping when address error detected
- Channel selection and control

■ DMA Transfer during Sleep

- The DMAC can also operate in sleep mode.
- If you anticipate operations during sleep mode, note the following:
  - Since the CPU is stopped, DMAC registers cannot be rewritten. Make settings before sleep mode is entered.
  - The sleep mode is released by an interrupt. Thus, if a peripheral interrupt is selected as the DMAC start source, interrupts must be disabled by the interrupt controller.
- If you do not want to release sleep mode with a DMAC end interrupt, disable these interrupts.

3.11 Channel Selection and Control

Up to five channels can be simultaneously set as transfer channels. In general, an independent function can be set for each channel.

■ Priority Among Channels

Since DMA transfer is possible only on one channel at a time, priority must be set for the channels.

Two modes, fixed and rotation, are provided as the priority settings and can be selected for each channel group (described later).

- Fixed mode

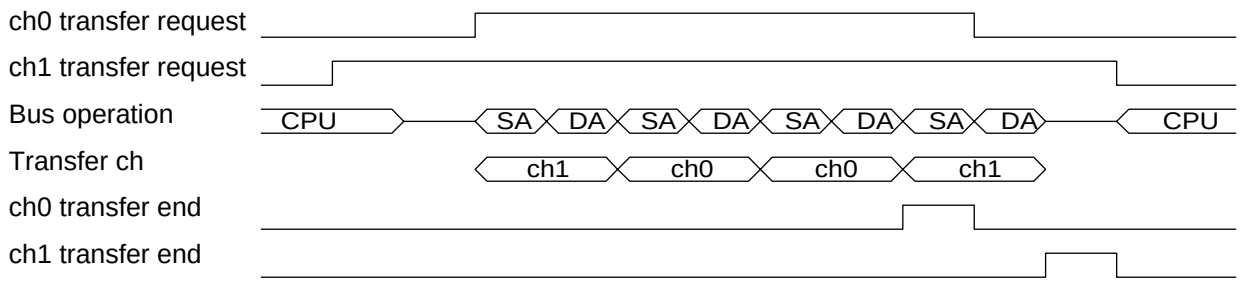
The order of priority is fixed by channel number, with priority decreasing from channel 0 to channel 4:

(ch.0 > ch.1 > ch.2 > ch.3 > ch.4)

If a transfer request with a higher priority is received during a transfer, the transfer channel becomes the channel with the higher priority when the transfer for the transfer unit (number set in the block size specification register x data width) ends.

When higher priority transfer is completed, transfer is restarted on the previous channel.

Figure 3-5 Timing Example in Fixed Mode

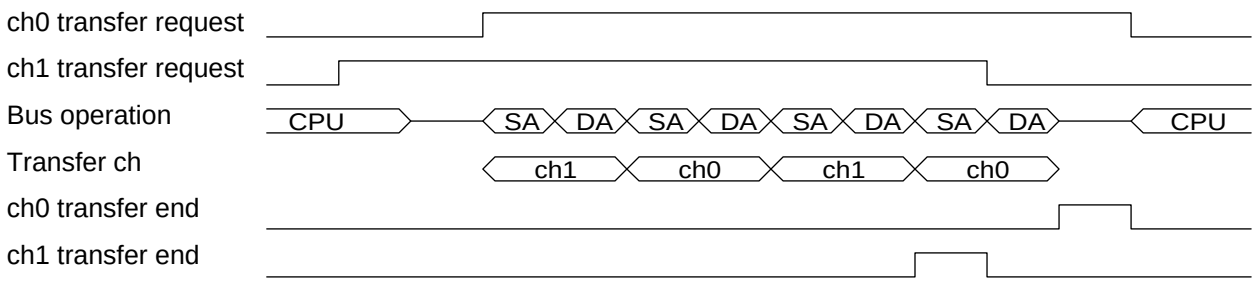


- Rotation mode (ch.0 to ch.1 only)

When operation is enabled, the initial states have the same order that they would have in fixed mode, but at the end of each transfer operation, the priority of the channels is reversed. Thus, if more than one transfer request is output at the same time, the channel is switched after each transfer unit.

This mode is effective when continuous or burst transfer is set.

Figure 3-6 Timing Example in Rotation Mode



■ Channel Group

The order of priority is set as shown in the following table.

MODE	Priority	Remarks
Fixed	ch0 > ch1	–
Rotation	ch0 > ch1 ↑ ↓ ch0 < ch1	The initial state is the top row. If transfer occurs for the top row, the priority is reversed.

3.12 Supplement on External Pin and Internal Operation Timing

This section provides supplementary information about external pins and internal operation timing.

■ Minimum Effective Pulse Width of the DREQ pin Input

Only channels 0-3 are applicable for the MB91460 series.

In all transfer modes for burst, step, block, and demand transfers, the minimum width required is five system clock cycles (5 cycles of CLKT).

Note:

DACK output does not indicate acceptance of DREQ input. DREQ input is always accepted if DMA is enabled but transfer has not started. Therefore, it is not necessary to retain DREQ input until DACK output is asserted (except in demand transfer mode).

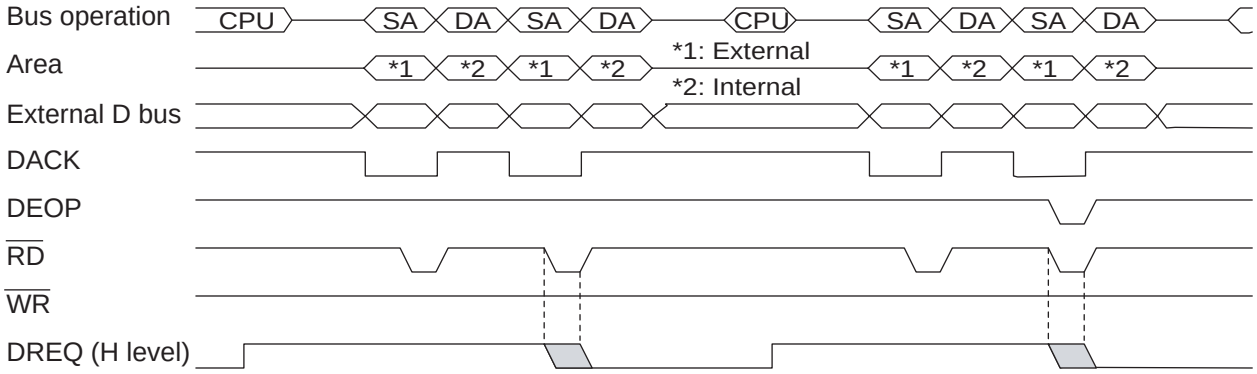
■ Negate Timing of the DREQ Pin Input when a Demand Transfer Request is Stopped

● For 2-cycle transfer

For a demand transfer, be sure to set an address in an external area for the transfer source, the transfer destination, or both.

- If the transfer type is external <--> external: Negate before the last sense timing of the clock in the L section of the external $\overline{WRn}$ pin output when accessing the transfer source for the last DMA transfer (section where DACK = L and $\overline{WRn}$ = L). If DREQ is negated later than this, a DMA request may be sensed, resulting in negation until the next transfer.
- If the transfer type is external <--> internal: Negate before the last sense timing of the clock in the L section of the external $\overline{RD}$ pin output when accessing the transfer source for the last DMA transfer (Section where DACK = L and $\overline{RD}$ = L). If DREQ is negated later than this, a DMA request may be sensed, resulting in negation until the next transfer

Figure 3-7 Negate timing example of the DREQ pin input for 2-cycle external transfer --> internal transfer



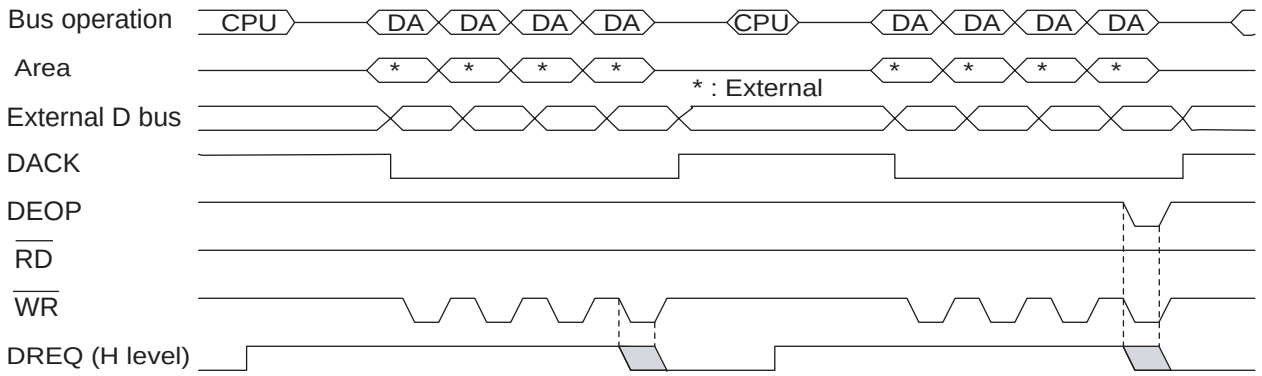
- If the transfer is internal <--> external: Negate before the last sense timing of the clock in the L section of the external $\overline{WRn}$ pin output when accessing the transfer source for the last DMA transfer (Section of DACK = 1and $\overline{WRn}$ = L). If DREQ is negated later than this, a DMA request may be sensed, resulting in negation until the next transfer.

● For fly-by (read/write) transfer

For a demand transfer, be sure to set an address in an external area for the transfer destination.

- For fly-by (read) transfer: After the $\overline{IOWR}$ pin output for the last DMA transfer goes to the H level, negate DREQ while the external $\overline{RD}$ pin output is at the L level. (section where DACK=L & $\overline{RD}$=L). If DREQ is negated later than this, the negation may continue until the next transfer.
- For fly-by (write) transfer: After the external $\overline{WRn}$ pin output for the last DMA transfer goes to the H level, negate DREQ while $\overline{IORD}$ is at the L level. (section where DACK=L & $\overline{RD}$=L). If DREQ is negated later than this, the negation may continue until the next transfer.

Figure 3-8 Negate timing example of the DREQ pin input for fly-by (write) transfer



■ Timing of the DREQ Pin Input for Continuing Transfer over the Same Channel

● For burst, step, block, and demand transfers

Operation in which transfer is continued over the same channel by the DREQ pin input cannot be guaranteed. If DREQ is reasserted at the fastest timing to clear requests retained internally after the transfer ends, at least one system clock cycle (one CLK output cycle) is provided to detect transfer requests for other channels. If, as a result, a transfer request for another channel with a higher priority is detected, transfer on that channel will be started.

Even if DREQ is reasserted earlier, it is ignored because the transfer has not been completed. If no transfer requests for other channels occur, transfer over the same channel is restarted by reasserting DREQ when the DACK pin output is asserted.

■ Timing of DACK Pin Output

The DACK output of this DMAC indicates that transfer with respect to an accepted transfer request is being performed.

The output of DACK is basically synchronized with the address output of external bus access timing. To use DACK output, it is necessary to enable the DACK output with a port.

■ Timing of the DEOP Pin Output

- The DEOP output of this DMA indicates that DMA transfer for the specified number of times of the accepted channel has been completed.
- DEOP output is output when access to an external area of the last transfer block starts. Thus, if any value other than 1 is set (block transfer mode) as the block size, DEOP is output when the last data of the last block is transferred. In this case, the acceptance of the next DREQ is already started even during transfer (before DEOP output) if the DACK pin output is asserted.
- The DEOP output is synchronized with $\overline{RD}$ and $\overline{WRn}$ of external bus access timing. However, if the transfer source/transfer destination is internal access, DEOP is not output. To use DEOP output, it is necessary to enable the DEOP output using the port register.

■ If an External Pin Transfer Request is Reentered During Transfer

● For burst, step, and block transfers

While the DACK signal is asserted within the DMAC, the next transfer request, if it is entered, is disabled. However, since operation of the external bus control unit and operation of the DMAC are not completely synchronous, the circuit must be initialized to create DREQ pin input using DACK and DEOP output to enable transfer requests by using DREQ input.

● For a demand transfer

If reloading of the transfer count register is specified when transfer for as many transfers as specified has been completed, another transfer request is accepted.

■ If Another Transfer Request Occurs During Block Transfer

No request is detected before the transfer of the specified blocks is completed. At the block boundaries, transfer requests accepted at that time are evaluated and then transfer on the channel with the highest priority is performed.

■ Transfer Between External I/O and External Memory

As targets of transfer by the DMAC, external I/O and external memory are not distinguished. Specify an external I/O as a fixed external address.

To perform fly-by transfer, set the address of external memory in the transfer destination address register. For external I/O, use DACK output and the signal decoded by the read signal $\overline{RD}$ or write signal $\overline{WRn}$ pin.

■ AC Characteristics of DMAC

DREQ pin input, DACK pin output, and DEOP pin output are provided as the external pins related to the DMAC,. Output timing is synchronized with external bus access (refer to the AC standard for the DMAC).

4. Operation Flowcharts

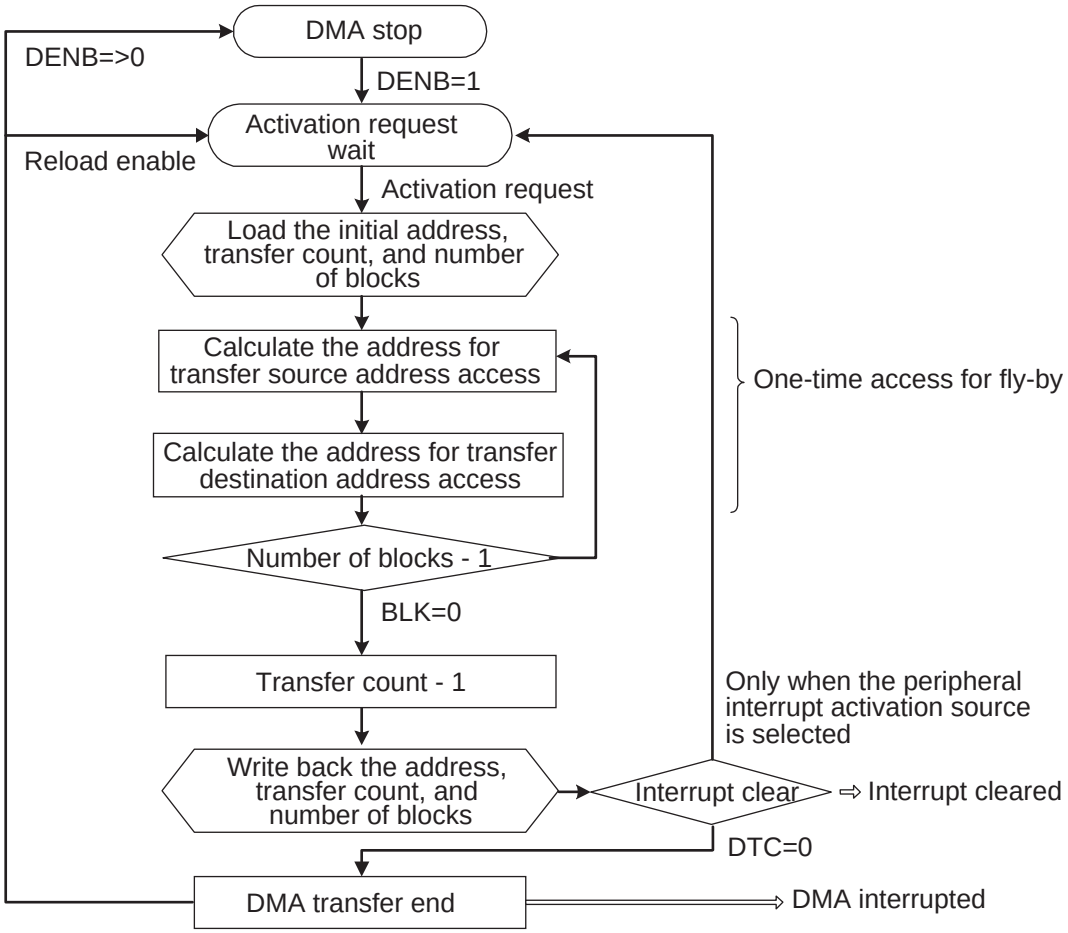
This section contains operation flowcharts for the following transfer modes:

- Block transfer
- Burst transfer
- Demand transfer

■ Block Transfer

Figure 4-1"Operation Flowchart for Block Transfer" shows the flowchart for block transfer.

Figure 4-1 Operation Flowchart for Block Transfer

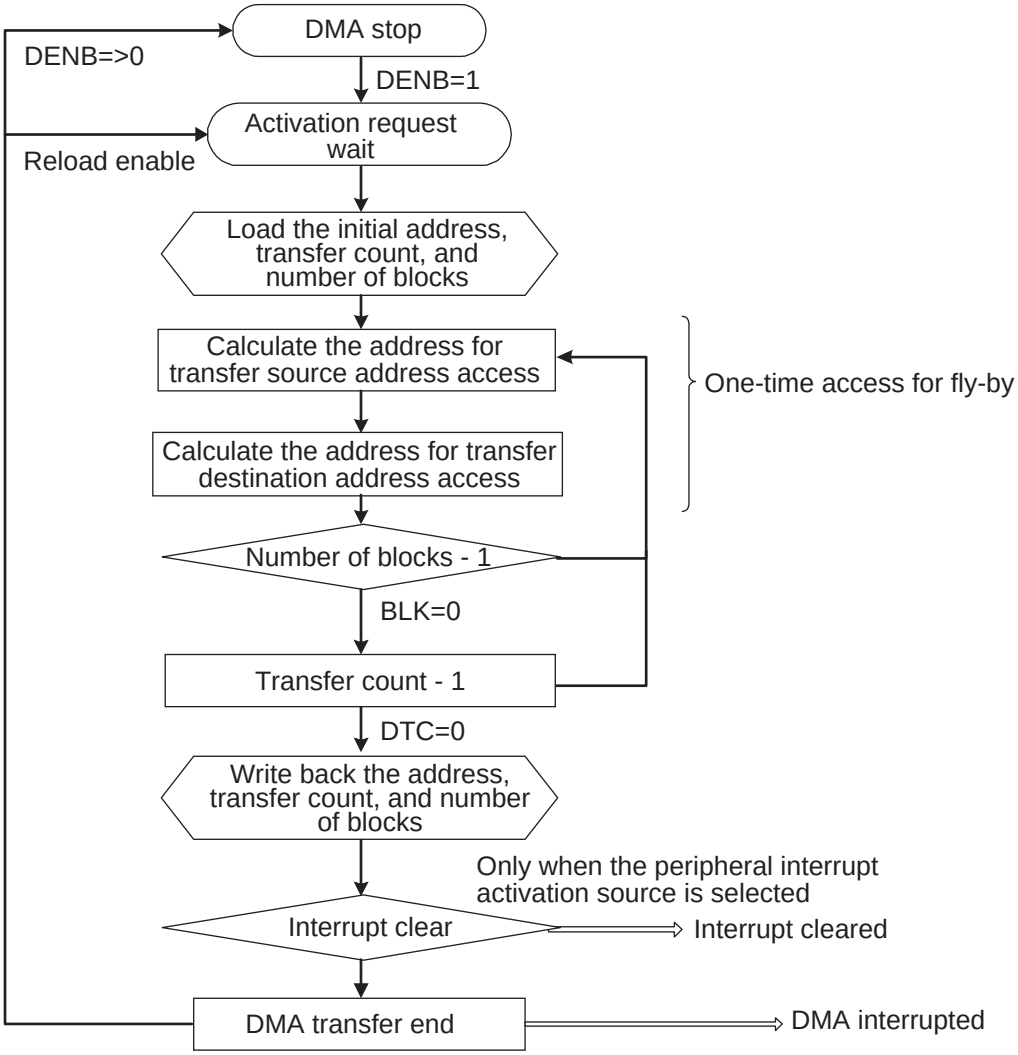


Block transfer
- Can be activated by all activation sources (selection).
- Can access to all areas.
- The number of blocks can be set.
- Interrupt clear is issued when transfer of the specified number of blocks is completed.
- The DMA interrupt is issued when transfer for the number of times specified is completed.

■ Burst Transfer

Figure 4-2"Operation Flowchart for Burst Transfer" shows the operation flowchart for burst transfer.

Figure 4-2 Operation Flowchart for Burst Transfer

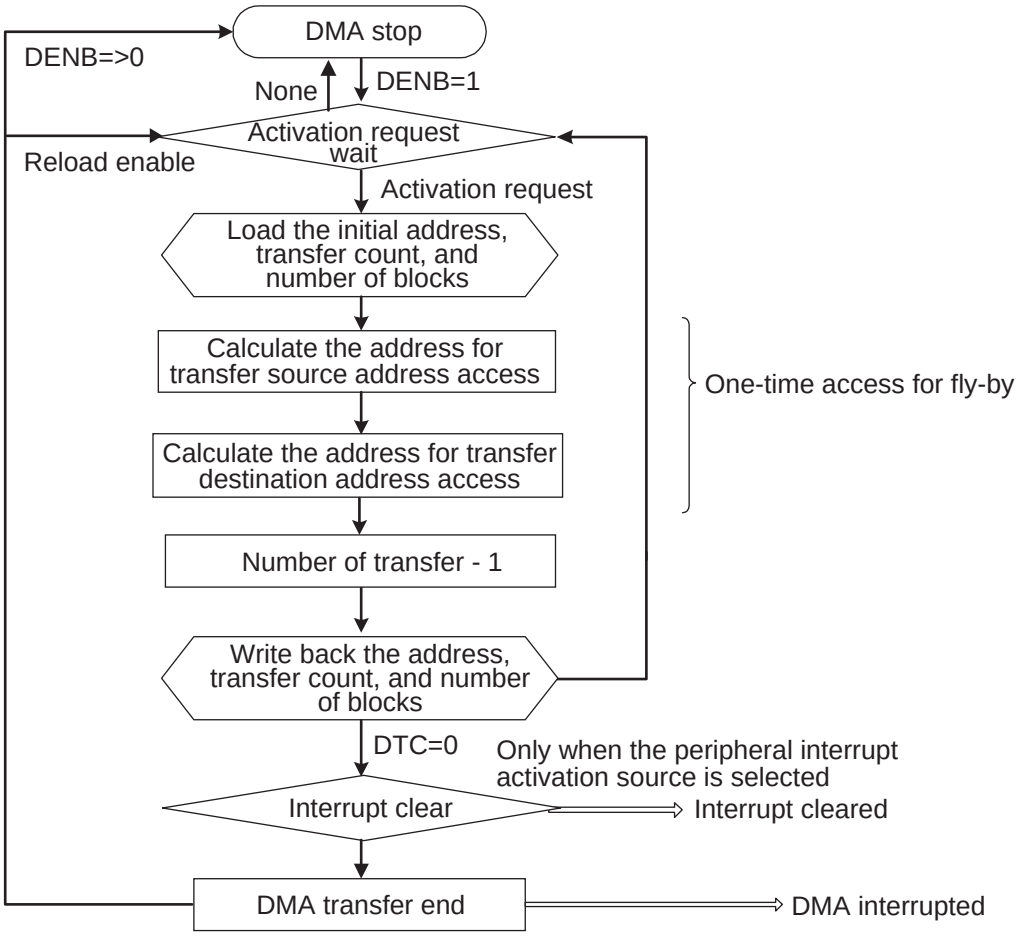


Burst transfer
- Can be activated by all activation sources (selection).
- Can access to all areas.
- The number of blocks can be set.
- Interrupt clear and the DMA interrupt are issued when transfer for the number of times specified is completed.

■ Demand Transfer

Figure 4-3"Operation Flowchart for Demand Transfer" shows the operation flowchart for demand transfer.

Figure 4-3 Operation Flowchart for Demand Transfer



Demand transfer

- Only requests (level detection) from the external pin (DREQ) are accepted. Activation by other sources is disabled.
- Access to an external area is required (since access to an external area becomes the next activation source).
- The number of blocks is always 1, regardless of the settings.
- Interrupt clear and the DMA interrupt are issued when transfer for the number of times specified is completed.

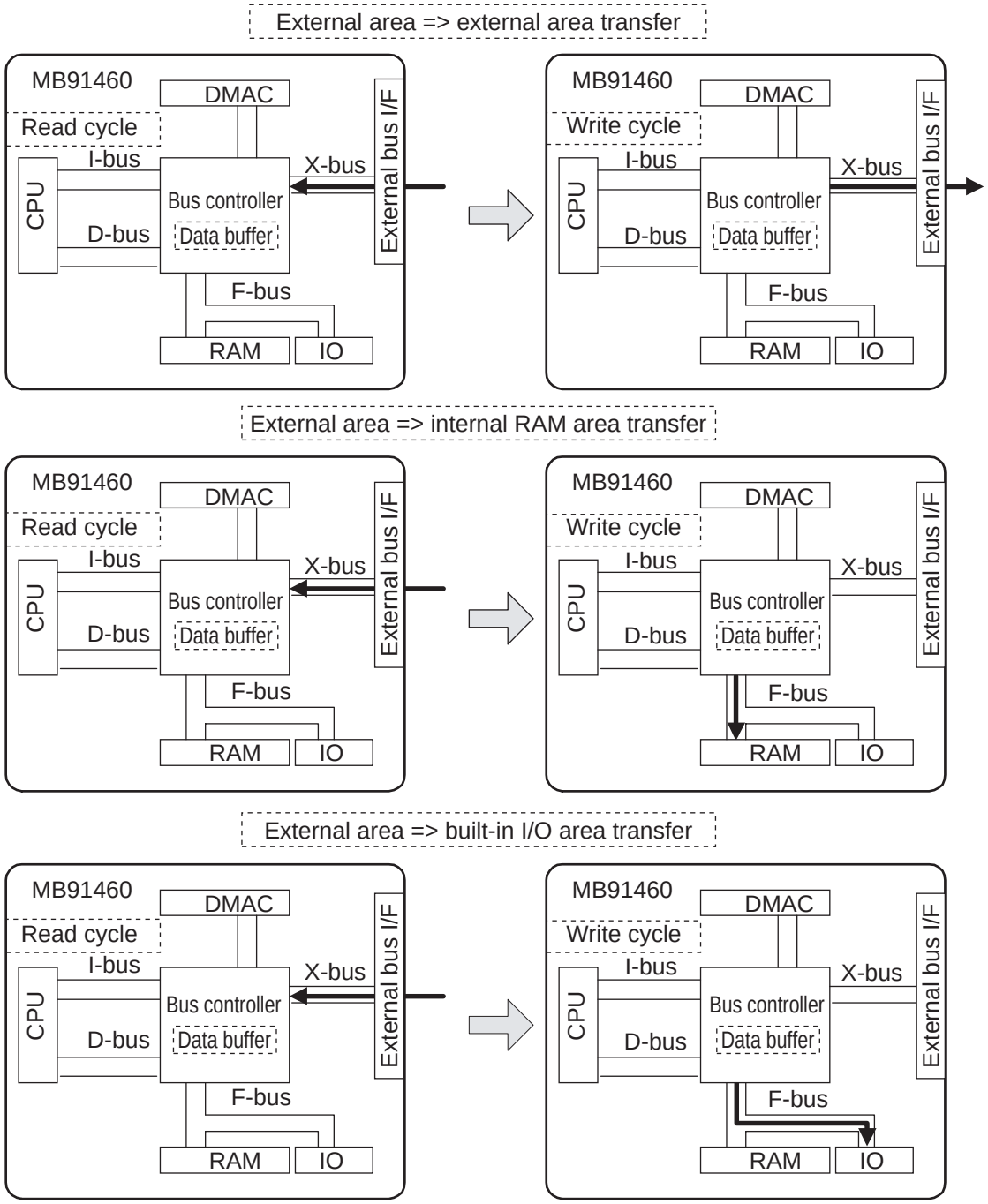
5. Data Bus

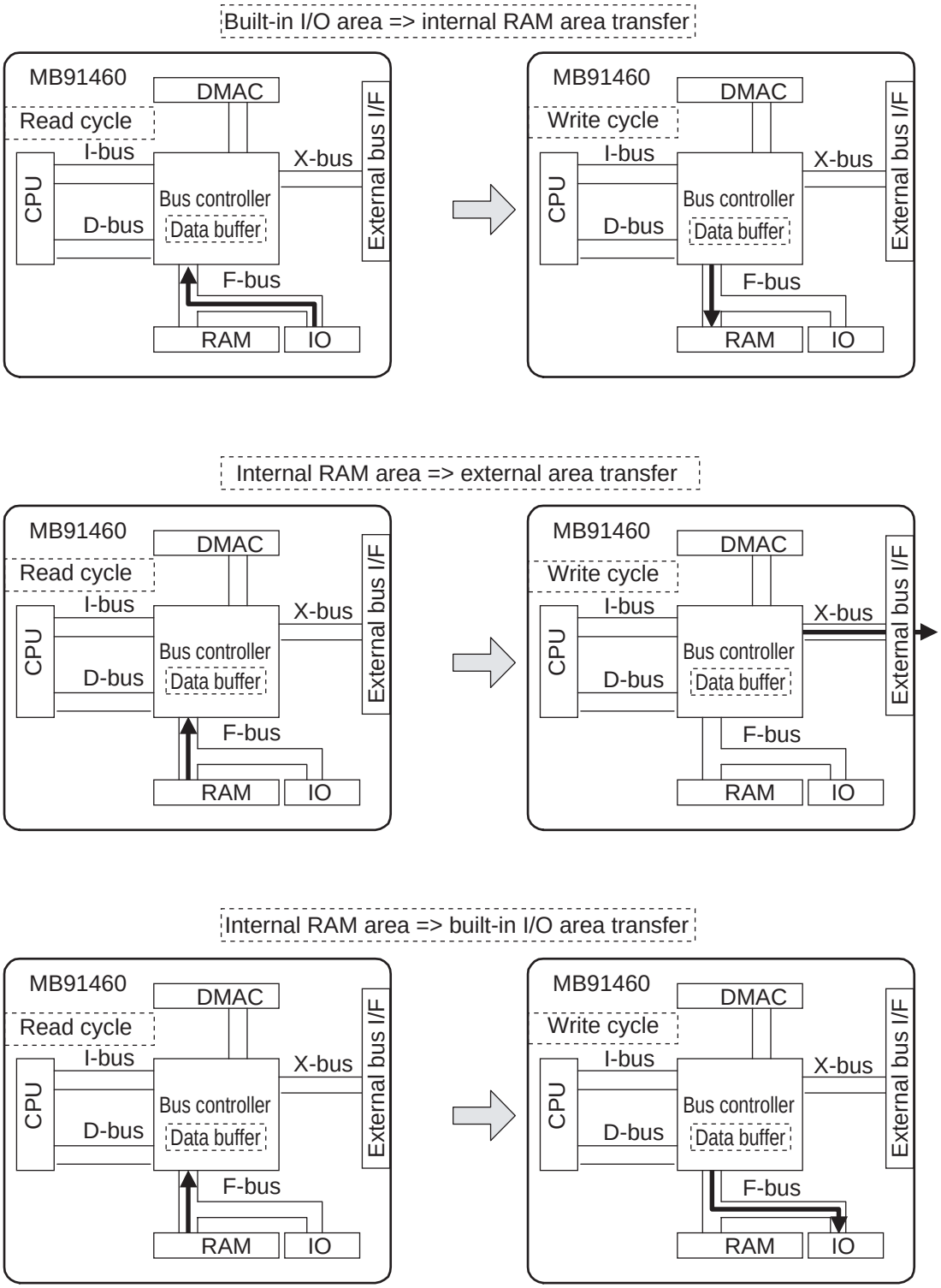
This section shows the flow of data during 2-cycle transfer and fly-by transfer.

■ Flow of Data During 2-Cycle Transfer

Figure 14.5-1 shows examples of six types of transfer during 2-cycle transfer.

Figure 5-1 Examples of 2-Cycle Transfer (Continued on next page)

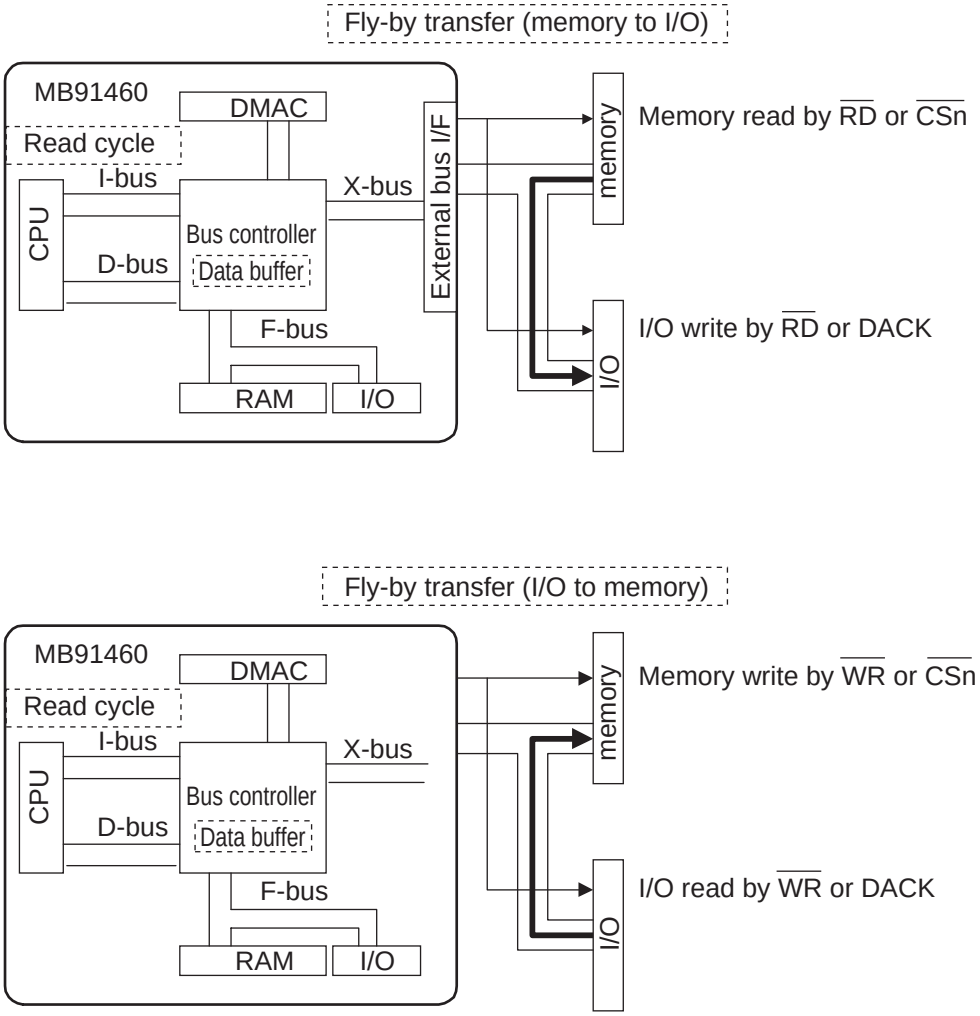




■ Flow of Data During Fly-By Transfer

Figure 5-2"Examples of Fly-By Transfer" shows examples of two types of transfer during fly-by transfer.

Figure 5-2 Examples of Fly-By Transfer



6. DMA External Interface

This section provides operation timing charts for the DMA external interface.

■ DMA External Interface Pins

DMA channels 0-3 have the following DMA-dedicated pins (DREQ, DACK, and DEOP):

- DREQ: DMA transfer request input pin for demand transfer. A transfer is requested with an input.
- DACK: This pin becomes active (L output) when DMA accesses an external area via the external interface.
- DEOP: This pin becomes active (L output) in synchronization with the last access to complete DMA transfer.
- $\overline{\text{IORD}}$: This signal becomes active when the direction I/O -> memory is selected for fly-by transfer.
- $\overline{\text{IOWR}}$: This signal becomes active when the direction memory -> I/O is selected for fly-by transfer.

Note:

Refer to 4.10 "DMA Access Operation" for the operation example of DMA external interface.

6.1 Input Timing of the DREQx Pin

The DREQx pin is a DMA start request signal. If the pin is also used as a port, enable the DREQ input using the PFR register. This section shows the input timing of the DREQx pin.

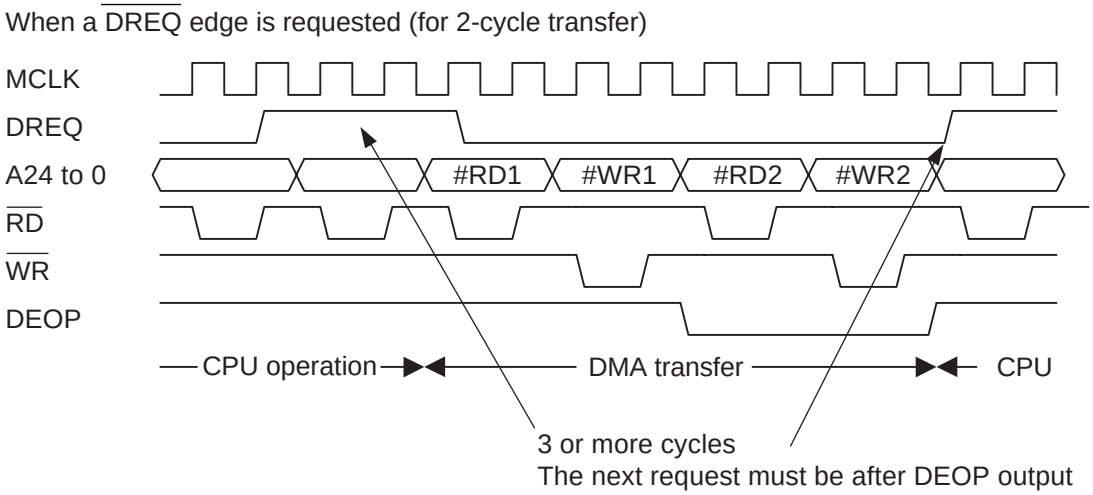
■ Timing of Transfer Other Than Demand Transfer

For transfer other than demand transfer, set the DMA start source to edge detection. Although there is no rule for rise/fall timing, use three or more clock cycles as the holding time the DREQ signal. To make another transfer request, enter the request after the DMA transfer is completed (make a request after DEOP is output).

If a request is made before DEOP is output, it may be ignored.

Figure 6-1"Timing Chart for Transfer Other Than the Demand Transfer" shows the timing chart for transfer other than demand transfer.

Figure 6-1 Timing Chart for Transfer Other Than the Demand Transfer

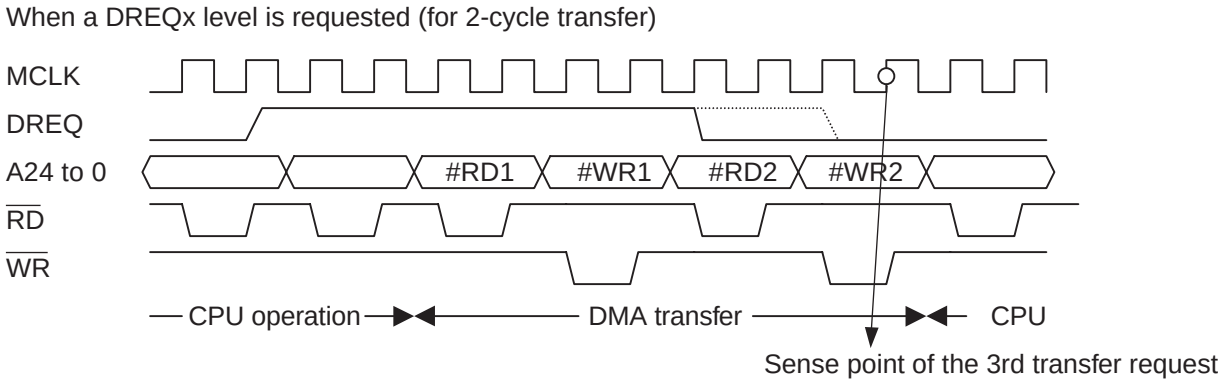


■ Timing of Demand Transfer

For demand transfer, set the DMA start source to level detection. Although there is no rule for starting, synchronize with $\overline{RD}$/$\overline{WRn}$ of the DMA transfer when stopping a transfer. The sense timing is the rise of MCLK in the final external access.

Figure 6-2"Timing Chart for Demand Transfer" shows the timing chart for demand transfer.

Figure 6-2 Timing Chart for Demand Transfer



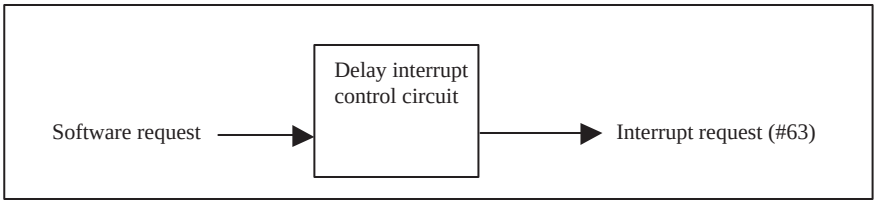
Note:

In this case, because 2-cycle transfer is used and the transfer source and transfer destination are an external area, negate from the fall of #RD2 to before the final MCLK rise of #WR2 to stop the two DMA transfer operations.

Chapter 27 Delayed Interrupt

1. Overview

The delayed interrupt, or the delayed interrupt module is used to generate an interrupt used for task switching.



2. Features

- Type: Interrupt request bit (There is no interrupt request enable bit)
- Quantity: 1
- Other:
  - The software generates/releases interrupt request.
  - Real time OS uses the delayed interrupt for task switching.

3. Configuration

Figure 3-1 Configuration Diagram

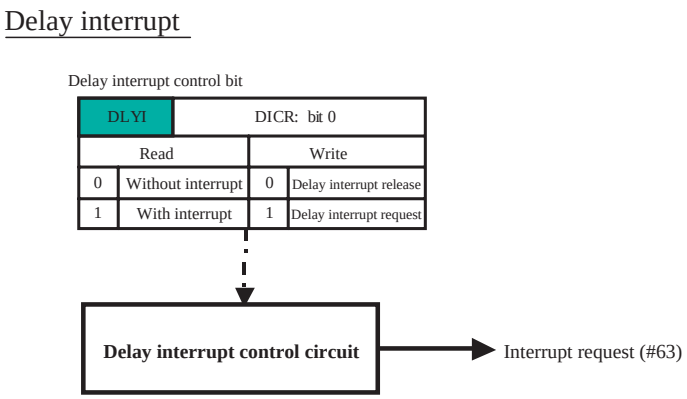
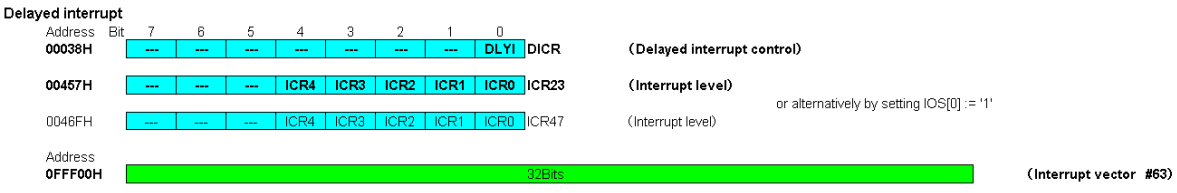


Figure 3-2 List of Registers



4. Register

4.1 DICR: Delayed Interrupt Control Register

This register controls to generate/clear the delayed interrupt.

- **DICR: Address 0038h (Access: Byte)**

7	6	5	4	3	2	1	0	bit
–	–	–	–	–	–	–	DLYI	
–	–	–	–	–	–	–	0	Initial value
RX/WX	RX/WX	RX/WX	RX/WX	RX/WX	RX/WX	RX/WX	R/W	Attribute

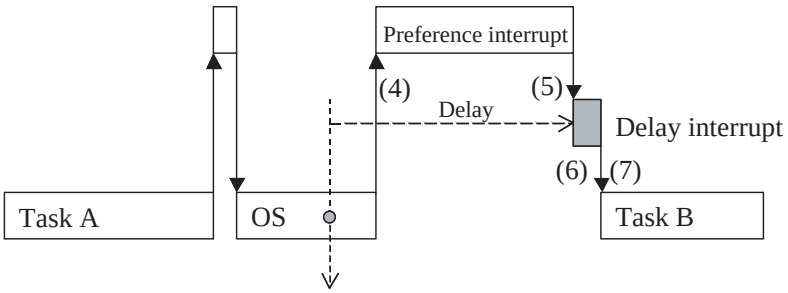
(Refer to "Meaning of Bit Attribute Symbols (Page No.10)" for the attributes.)

- Bit7-1: Undefined: Writing does not affect operation. The read value is undefined.
- Bit0: Delayed interrupt control bit

DLYI	Read operation	Write operation
0	No delayed interrupt request	Delayed interrupt request clear
1	Delayed interrupt request	Delayed interrupt request generation

5. Operation

Delayed interrupt service



(1) In OS, a request for task B dispatch is generated
(2) OS sets the delayed interrupt return destination (dispatch destination)
(3) OS sets the delayed interrupt (delayed interrupt generation)
(4) When OS returns, the interrupt with the highest priority sequence takes place, because an interrupt service is prohibited in OS
(5) When the interrupt with the highest priority is completed, delayed interrupt takes place
(6) In delayed interrupt, delayed interrupt is released
(7) Returned from the delayed interrupt (dispatched to task B)

6. Setting

Table Setting required for the delayed interrupt generation/clear

Table 6-1 Setting required for the delay interrupt generation/clear

Setting	Setting register	Setting method*
Vector for delay interrupt	Refer to “Chapter 24 Interrupt Control (Page No.315)”	Refer to 7.1
Delayed interrupt setting. Generating interrupt request/Releasing interrupt request	Delayed interrupt control register (DICR)	Refer to 7.2

*: Refer to the number for the setting method.

7. Q & A

7.1 What are interrupt-associated registers?

Setting for the delay interrupt vector and interrupt level

The relationship between the delay interrupt level and the delay interrupt vector is shown in the following table.

Refer to “Chapter 24 Interrupt Control (Page No.315)” for more information on the interrupt level and interrupt vector.

Interrupt vector (default)	Interrupt level setting bit (ICR[4:0])
#63 Address: 0FFF00h	Interrupt level register (ICR23) Address: 00457h

The interrupt request bit (DICR.DLYI) cannot automatically be released, and it should be released by the software before returning from an interrupt service. (“0” is written for DLYI bit)

Remark: For REALOS compatibility reasons, ICR23 and ICR47 can be exchanged by setting the REALOS compatibility bit (addr 0x0C03 : IOS[0]) if necessary.

7.2 How is the interrupt request generated/cleared?

The delayed interrupt request bit (DICR.DLYI) performs this function.

	Interrupt request enable bit (DLYI)
Clearing an interrupt request	Sets the value to “0”
Generating an interrupt request	Sets the value to “1”

The delayed interrupt does not have an interrupt request enable bit.

8. Caution

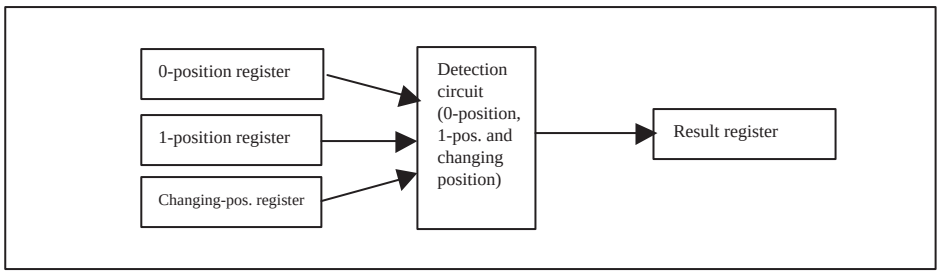
- The delay interrupt request bit is the same as general interrupt request flags. It should be used to clear delayed interrupt request bit in an interrupt routine in addition to switching tasks.
- The delayed interrupt function can use real time OS (REALOS). As a result, the delayed interrupt function is prohibited in a piece of user software when using real time OS.

8.Caution

Chapter 28 Bit Search

1. Overview

The bit search module is used to detect 0, 1 or changing position for data written in specific registers.



2. Features

- Function: Detects the first changing position by scanning data written in data register from MSB to LSB.
  - 0 detection Detects the first '0' changing position.
  - 1 detection Detects the first '1' changing position.
  - Changing position detection Detects the position where data first changes.
- Quantity: 1
- Other: Can read internal data.
  (This can be used to restore the previous state when it is used for bit search during interrupt service or handler.)

3. Configuration

Figure 3-1 Configuration Diagram

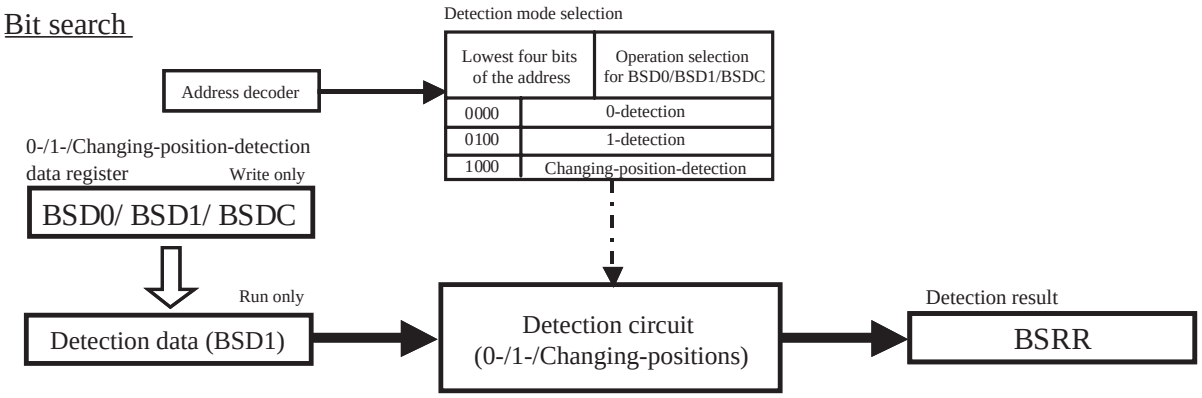
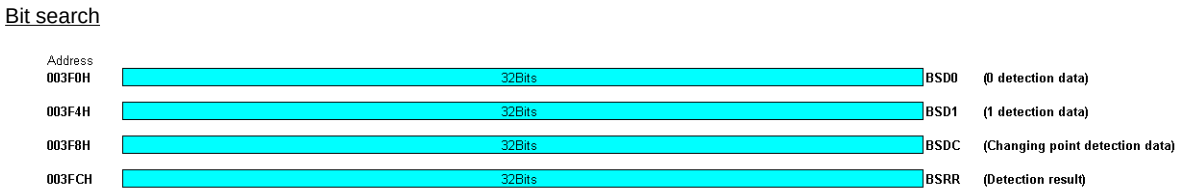


Figure 3-2 List of Registers

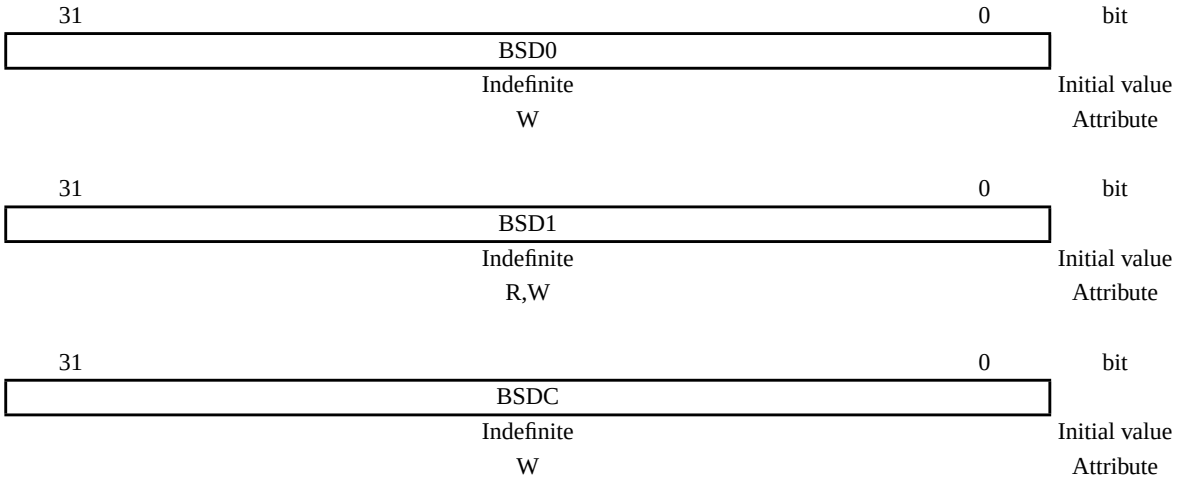


4. Register

4.1 BSD0: 0 Detection Register / BSD1:1 Detection Register / BSDC: Changing position Detection Data Register

This is a register for setting the bit search detection data.

- **BSD0: Address $03F0_H$ (Access: Word)**
- **BSD1: Address $03F4_H$ (Access: Word)**
- **BSDC: Address $03F8_H$ (Access: Word)**



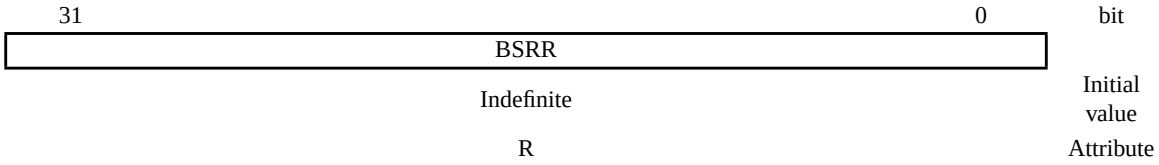
(For the attributes, refer to “Meaning of Bit Attribute Symbols (Page No.10)”.)

- Write data used to detect 0, 1 and changing position in each of the registers BSD0, BSD1 and BSDC.
- The result is stored in the detection result register BSRR.
  - During 0 detection, the position where “0” is first detected is stored for data written in the order of MSB(bit31) to LSB(bit0).
  - During 1 detection, the position where “1” is first detected is stored for data written in the order of MSB(bit31) to LSB(bit0).
  - During change position detection, the position where a value different from MSB(bit31) is first detected is stored for data written in the order of bit30 to LSB(bit0).
- The register BSD0 used for 0 detection and the BSRC register used for changing position detection are write-only. The value during read operation is indefinite.
- Data saved in the bit search can be read if the register BSR1 used to detect 1 is read.
  Previous detection result can be restored by re-writing previously read data in the BSR1 used for detecting 1. This applies to the processes for the 0 detection and the changing position detection.This function can be used to restore a specific state when using a bit search in processing such as interrupt handler.

4.2 BSRR: Detection Result Register

This register is used to read a bit search result.

- **BSRR: Address 03FC$_H$ (Access: Word)**

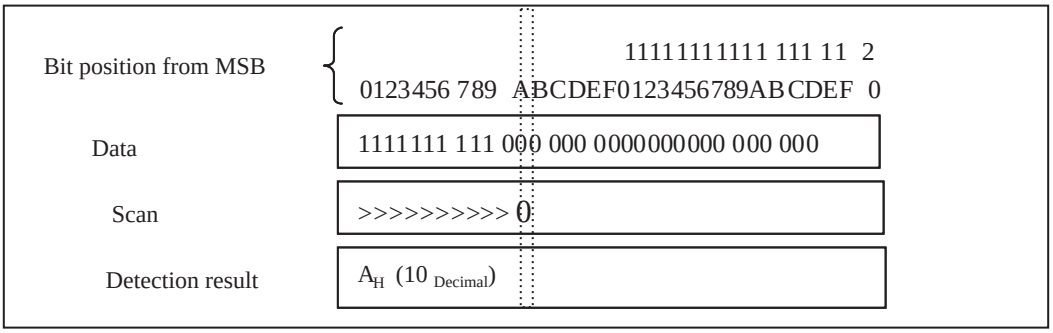


(For the attributes, refer to “Meaning of Bit Attribute Symbols (Page No.10)”.)

- Detection result for data written in the 0 detection register BSD0, the 1-detection register BSD1 and the changing-position-detection register BSDC can be read. Data last written can be read. However, the type of result cannot be identified: Information on 0 detection,1 detection or changing position detection is not included.
  A 0 can be read at detection position bit31(MSB), and continues reading 31 at detection position bit0(LSB) by adding 1 at the next position toward bit0(LSB). A value of 32 is read when not detected.
- The detection result register is read-only, and a write operation is invalid.

5. Operation

5.1 Zero detection

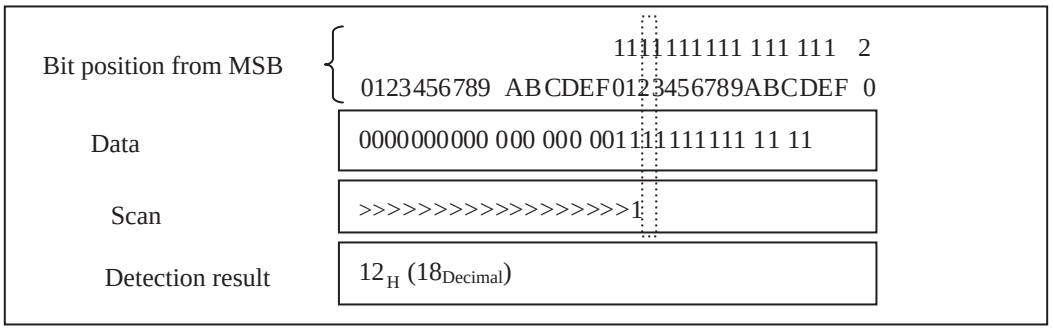


(1) Bit position from MSB
(2) Written data (Starts to search once data is written.)
(3) Detects “0” by scanning from MSB.
(4) Detected bit position
(5) Detection result
If ‘0’ does not exist (That is, numeric value is $FFFFFFFF_H$), ‘32’ is returned as detection result.

- Execution example

Write data	Read value	(Decimal notation)
$11111111111111111111000000000000_B$ ($FFFFF000_H$)	Æ	20
$11111000010010011110000010101010_B$ ($F849E0AA_H$)	Æ	5
$10000000000000101010101010101010_B$ ($8002AAAA_H$)	Æ	1
$11111111111111111111111111111111_B$ ($FFFFFFFF_H$)	Æ	32

5.2 One Detection

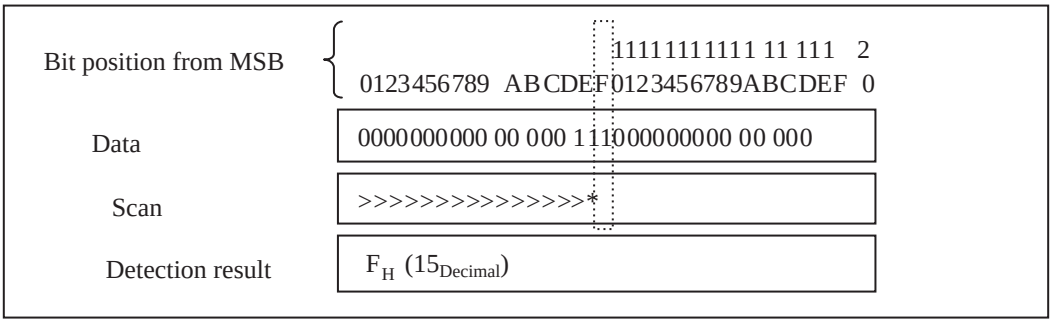


(1) Bit position from MSB
(2) Written data (Detection operation starts once data is written.)
(3) Detect “1” scan starting with the MSB.
(4) Detected bit position
(5) Detection result
If ‘1’ does not exist (That is, if numeric value is 00000000_H), value of ‘32’ is returned as detection result.

- Execution example

Write data	Read value	(Decimal notation)
$00100000000000000000000000000000_B$ (20000000_H)	Æ	2
$00000001001000110100010101100111_B$ (01234567_H)	Æ	7
$00000000000000111111111111111111_B$ ($0003FFFF_H$)	Æ	14
$00000000000000000000000000000001_B$ (00000001_H)	Æ	31
$00000000000000000000000000000000_B$ (00000000_H)	Æ	32

5.3 Changing Position Detection



(1) Bit position from MSB
(2) Written data (Detection starts once data is written.)
(3) Detects the changing position by scanning from MSB.
(4) Detected bit position
(5) Detection result
A value of '32' is returned as detection result if changing position does not exist.
A value of '0' is not returned as detection result for changing position detection.

- Execution example

Write data	Read value	(Decimal notation)
$00100000000000000000000000000000_B$ (20000000_H)	Æ	2
$00000001001000110100010101100111_B$ (01234567_H)	Æ	7
$00000000000000111111111111111111_B$ ($0003FFFF_H$)	Æ	14
$00000000000000000000000000000001_B$ (00000001_H)	Æ	31
$00000000000000000000000000000000_B$ (00000000_H)	Æ	32
$11111111111111111111000000000000_B$ ($FFFFF000_H$)	Æ	20
$11111000010010011110000010101010_B$ ($F849E0AA_H$)	Æ	5
$10000000000000101010101010101010_B$ ($8002AAAA_H$)	Æ	1
$11111111111111111111111111111111_B$ ($FFFFFFFF_H$)	Æ	32

Table 5-1 The Relationship Between the Bit Position and the Value to be Returned (Decimal Notation)

Detected bit position	Return value	Detected bit position	Return value	Detected bit position	Return value	Detected bit position	Return value
31	0	23	8	15	16	7	24
30	1	22	9	14	17	6	25
29	2	21	10	13	18	5	26
28	3	20	11	12	19	4	27
27	4	19	12	11	20	3	28
26	5	18	13	10	21	2	29
25	6	17	14	9	22	1	30
24	7	16	15	8	23	0	31
						Nonexistent	32

6. Setting

Table 6-1 Settings Required for “Zero” Position Detection

Setting	Setting register	Setting method *
Data write & scan start	“Zero” position detection data register (BSD0)	Refer to 7.1.
Converted value read	Detection result register (BSRR)	Refer to 7.2.

*: For detailed description contents, refer to the reference destination number.

Table 6-2 Setting Required for Using “One” Position Detection

Setting	Setting register	Setting method *
Data write & scan start	“One” position detection data register (BSD1)	Refer to 7.1.
Converted value read	Detection result register (BSRR)	Refer to 7.2.

*: For detailed description contents, refer to the reference destination number.

Table 6-3 Setting Required for Using Changing Position Detection

Setting	Setting register	Setting method *
Data write & scan start	Changing position detection data register (BSDC)	Refer to 7.1.
Converted value read	Detection result register (BSRR)	Refer to 7.2.

*: For detailed description contents, refer to the reference destination number.

7. Q & A

7.1 How is data written?

Writes data with the detection data registers (BSD0, BSD1, BSDC).

Operation mode	Detection data register
“Zero” position detection write	Writes data in (BSD0)
“One” position detection write	Writes data in (BSD1)
Changing position detection write	Writes data in (BSDC)

7.2 How is scanning started?

Scanning is started once data is written in the detection data registers (BSD0, BSD1, BSDC).

7.3 How is a result read?

The detection result register (BSRR) is read.

7.4 How is the previous bit search state restored?

The following restoration processes are performed.

If you need to restore the previous bit search state after a bit search has been executed in an interrupt handler.

1) Reads data from the one detection data register, and saves the contents. (evacuation)

2) The Bit search is used.

3) Writes data evacuated in Item 1) in the one detection data register. (restoration)

Using the above procedures, the value to be read next from the detection result register is the one that was written in the bit search executed in 1) or before.

The bit search state can be correctly restored using the above procedures even if the 0-detection, 1-detection or changing-position-detection data register has been written last.

8. Caution

The following are the remarks on using the bit search module.

- The macros are for REALOS(OS), and the user cannot use them when using REALOS.
- If the relevant detection is not found, a detection result of 32(decimal), 10(hexadecimal) or 10000(binary) is returned.
- A value of “0” is not returned for the changing position detection.
- The data registers (0-detection/1-detection/ changing-position-detection) is a write-only, and accessed by word.
  However, the 1-detection read address is assigned to an internal data register for restoration so that you can restore previous bit search state. (Refer to “7.3 How is a result read? (Page No.396)”.)
- The 0-detection register BSD0, 1-detection register BSD1 and changing-position-detection register BSDC are included in one register in terms of the structure. The operation is selected with the lowest four bits of the accessing address.

8.Caution

Chapter 29 MPU / EDSU

1. Overview

Memory Protection Unit (MPU) and Embedded Debug Support Unit (EDSU) for MB91460 series.

Remark: The MPU/EDSU module features a clock disable function. For enabling the MPU/EDSU module it is necessary to set the EDSUEN bit in the CSCFG register. See chapter "CSCFG: Clock Source Configuration Register (Page No.196)" for further information.

The features are scalable in units of 'Comparator Groups'. The number of this Groups can be defined from one to eight. Features of one Comparator Group are listed below:

- A total number of 4 Breakpoints, could be programmed to:
  - 4 Instruction Address Breakpoints
  - 4 Operand Address Breakpoints (programmable on datasize and access type)
  - 2 Operand Address Breakpoints and 2 Instruction Address Breakpoints
  - 2 Operand Address Breakpoints and 2 Data Value Breakpoints
- 2 Masks possible to assign (reduces the number of breakpoints)
- 2 Range Functions
- Break Trigger programmable on resource interrupts
- MPU functionality
  - User and SuperVisor permission for read/write/execute
  - Default permissions for the whole MCU address range
  - Permission definition for two address ranges per Comparator Group (8 Groups result in 16 MPU Channels)
  - Can detect DMA accesses on the D-Bus and Resource address regions
  - Register set is locked in User mode
  - Dynamic configuration possible, privileged configuration with INT #5 is not interruptible
  - A permission violation causes an MPUPV trap
- Capture register for Instruction Address and Operand Address (for MPU and Operand Break)
- Capture information for MPU channel index, DMA flag, Operand Size and Access Type

2. Features

One Comparator Group offers up to 4 Breakpoints. One Group consists of two full-featured range comparators with the option to use two point registers as mask information. The following features could be partially mixed-up:

4 Instruction Address Breakpoints

Up to 4 instruction address breakpoints can be defined.

Two instruction breakpoints can be masked. The other two registers can operate as mask registers then. Also maskable is a break address range made with two points and one mask register.

Two absolute address ranges for instruction breakpoints can be defined where 2 or 4 out of 4 instruction breakpoint registers are assigned for the range.

4 Operand Address Breakpoints

Up to 4 operand address breakpoints can be defined.

Two operand breakpoints can be masked. The other two registers can operate as mask registers then. Also maskable is a break address range made with two points and one mask register.

Two absolute address ranges for operand breakpoints can be defined where 2 or 4 out of 4 operand breakpoint registers are assigned for the range.

Operand breaks can be selected for datasizes: byte, halfword and word on access types: read, read-modify-write and write.

2 Operand Data Value Breakpoints

Up to 2 operand data value breakpoints can be defined.

The definition of one data value range is possible.

One data value breakpoint can be masked by defining the other point as mask register.

The Operand Address and Data Value Breakpoints can be switched to a combined trigger condition.

Memory protection

Two channels/ranges could be defined to operate in memory protection mode.

Possible is the protection of two Operand Address ranges, two Instruction Address Ranges or a combination of one Operand and one Instruction Address range.

Read/write or execute permissions could be defined for each channel, both for the normal User and the SuperVisor mode.

3. Break Functions

3.1 Instruction address break

The instruction address point break is the most basic break that occurs when an instruction is fetched at the address specified by the break address data registers BAD[3:0]. Setting the CTC[1:0] bits of the control register BCR0 to ’00’ provides this mode. The bits EP[3:0] in BCR0 enable the break points.

Up to 4 instruction breakpoints from channels 0 to 3 can be set. All instruction break events are ORed into instruction break exception requests to the CPU.

2 of the break address registers can operate as mask registers (BAD0, BAD2) for masking the instruction address which is being fetched. Mask register BAD0 can be assigned either to BAD1 (same channel) or BAD2/3 (opposite channel), mask register BAD2 can be assigned either to BAD3 or BAD0/1.

Normally Instruction break address and mask information reside in the same channel. So BAD3 contains the instruction break address and BAD2 the address mask information. The channel is enabled with EP3. The same applies for channel BAD1 (address), BAD0 (mask) and EP1 (enable).

But some cases require enabling point 2 (EP2) or the range function (ER1). Then BAD2 holds Instruction Address information and could not carry the address mask. In that cases (when EP2 or ER1 are set) the mask information is taken from the opposite BAD0 register. The same applies for EP0 and ER0 - which enables the use of the opposite BAD2 register for the mask information.

Example:	CTC	00	Type: Instruction Address Break
	EP1	1	Enable break point address BAD1
	EM0	1	Set mask BAD0 for break address BAD1
	BAD1	0x12345678	Set break address
	BAD0	0x00000FFF	Set break mask

Break occurs at 0x12345000 to 0x12345FFF

On break at BAD[3:0] the respective flags BD[3:0] in the break interrupt request register BIRQ will be set to ’1’. They have to be reset by software in the instruction break routine.

Channels 0 and 1 (BAD0, BAD1) can be set up to function as address range match. Setting the ER0 bit of the control register BCR0 to ’1’ provides this mode. BAD0 is the lower address and BAD1 is the upper address for address comparison. In this mode the mask register BAD2 will mask both channels 0 and 1, if the mask feature is enabled by EM0 = ’1’.

Alternatively channels 2 and 3 (BAD2, BAD3) can be set up to function as address range match. Setting the ER1 bit of the control register BCR0 to ’1’ provides this mode. BAD2 is the lower address and BAD3 is the upper address for address comparison. In this mode the mask register BAD0 will mask both channels 2 and 3, if the mask feature is enabled by EM1 = ’1’.

Example:	CTC	00	Type: Instruction Address Break
	EP0	1	Enable break point on BAD0
	EP1	1	Enable break point on BAD1
	ER0	1	Enable address range function on BAD0, BAD1
	EM0	1	Enable address mask function on BAD0, BAD1
	BAD0	0x12345200	Set lower break address
	BAD1	0x12345300	Set upper break address
	BAD2	0xF0000000	Set break mask

Break occurs at 0x02345200 to 0x02345300,or
at 0x12345200 to 0x12345300,or
at 0x22345200 to 0x22345300, etc.
The resulting setting of the BD[1:0] status bits indicates the point, respective the area in which the break has occured.

Table 3-1 Instruction Break Detection Status Bits (BD)

BD1	BD0	
0	1	Match on point (instruction address == 0x12345200), or Match on point (instruction address == 0x22345200), etc
1	0	Match on point (instruction address == 0x12345300), or Match on point (instruction address == 0x22345300), etc
1	1	Match on range (0x12345200 < instruction address < 0x12345300), or Match on range (0x22345200 < instruction address < 0x22345300), etc

In the instruction address break mode the following important point has to be considered:
To precisely determine the instruction address where a break occurs, use the PC value saved on the stack during entry to the instruction break interrupt service routine.

3.2 Operand address break

The operand break function causes a break for the data access address which can be specified by the the operand address break registers BAD[3:0]. Setting the CTC[1:0] bits of the control register BCR0 to ’01’ provides this mode. The bits EP[3:0] in BCR0 enable the break points.
Up to 4 breakpoints from channels 0 to 3 can be set. All operand break events are ORed into a operand break exception interrupt request to the CPU.
For the address mask function the same applies, what is stated in section 3.1 for the Instruction Address Break.

Example:	CTC	01	Type: Operand Address Break
	EP1	1	Enable break point address BAD1
	EM0	1	Set mask BAD0 for break address BAD1
	BAD1	0x12345678	Set break address
	BAD0	0x00000FFF	Set break mask

Break occurs at 0x12345000 to 0x12345FFF

On break at BAD[3:0] the respective flags BD[3:0] in the break interrupt request register BIRQ will be set to ’1’. They have to be reset by software in the operand break exception routine.

Channels 0 and 1 (BAD0, BAD1) can be set up to function as address range match. Setting the ER0 bit of the control register BCR0 to ’1’ provides this mode. BAD0 is the lower address and BAD1 is the upper address for address comparison. In this mode the mask register BAD2 will mask both channels 0 and 1, if the mask feature is enabled by EM0 = ’1’.
Alternatively channels 2 and 3 (BAD2, BAD3) can be set up to function as address range match. Setting the ER1 bit of the control register BCR0 to ’1’ provides this mode. BAD2 is the lower address and BAD3 is the upper address for address comparison. In this mode the mask register BAD0 will mask both channels 2 and 3, if the mask feature is enabled by EM1 = ’1’.

Example:	CTC	01	Type: Operand Address Break
	EP0	1	Enable break point on BAD0
	EP1	1	Enable break point on BAD1
	ER0	1	Enable address range function on BAD0, BAD1
	EM0	1	Enable address mask function on BAD0, BAD1
	BAD0	0x12345200	Set lower break address
	BAD1	0x12345300	Set upper break address
	BAD2	0xF0000000	Set break mask

Break occurs at 0x02345200 to 0x02345300,or
at 0x12345200 to 0x12345300,or
at 0x22345200 to 0x22345300, etc.

The resulting setting of the BD[1:0] status bits indicates the point, respective the area in which the break has occured.

Table 3-2 Operand Break Detection Status Bits (BD)

BD1	BD0	
0	1	Match on point (operand address == 0x12345200), or Match on point (operand address == 0x22345200), etc
1	0	Match on point (operand address == 0x12345300), or Match on point (operand address == 0x22345300), etc
1	1	Match on range (0x12345200 < operand address < 0x12345300), or Match on range (0x22345200 < operand address < 0x22345300), etc

The access data length and read/write break attributes can also be specified by the control register BCR0, bits OBS[1:0] and OBT[1:0]. When the mask function is disabled by setting EM1 = EM0 = 0 (all bits effective), the relationship between breakpoint setting, and break by access address is shown below:

Table 3-3 Operand size and operand address relations

Access data length	Access address	Address set in BOA0, BOA1			
		4n + 0	4n + 1	4n + 2	4n + 3
8 bit	4n + 0	Hit	-	-	-
	4n + 1	-	Hit	-	-
	4n + 2	-	-	Hit	-
	4n + 3	-	-	-	Hit
16 bit	4n + 0	Hit	Hit	-	-
	4n + 1	Hit	Hit	-	-
	4n + 2	-	-	Hit	Hit
	4n + 3	-	-	Hit	Hit

Table 3-3 Operand size and operand address relations

Access data length	Access address	Address set in BOA0, BOA1			
		4n + 0	4n + 1	4n + 2	4n + 3
32 bit	4n + 0	Hit	Hit	Hit	Hit
	4n + 1	Hit	Hit	Hit	Hit
	4n + 2	Hit	Hit	Hit	Hit
	4n + 3	Hit	Hit	Hit	Hit

In Operand address break mode the Operand Address, causing the break is captured in the BOAC register. Additional BIAC holds the instruction address of the instruction, which was executed one cycle before the break causing data operation. This is normally the instruction, which has caused the data transfer.

In the operand address break mode the following important points have to be considered:

1) In the FR family architecture, if data access is performed with misalignment, the lower address bit 0 will be ignored for halfword and the lower address bits 0 and 1 for word access. The mask register could be programmed accordingly.

2) The EDSU operand break does not always occur immediately after completion of execution of the instruction causing the break event.

3) Please see also information at chapter 3.4 Using operand with data break

3.3 Data value break

The data value break causes a break if specified data is read or written at a data access to an address specified by the CPU. The data can be specified by the data value break registers BAD0 and BAD1. Setting the CTC[1:0] bits of the control register BCR0 to ’11’ provides this mode. The bits EP0 and EP1 in BCR0 enable the break condition.

Up to 2 break points from channels 0 to 1 can be set. All data value break events are ORed into a operand break exception to the CPU.

1 mask register (BAD0) is available for masking the data value (stored in BAD1) and 1 mask register (BAD2) is available for masking the operand address (BAD3) which is being accessed. Mask registers BAD2 and BAD0 can be enabled with EM1 and EM0.

The data on which a break should be executed must be masked by a data-mask on the bus, requiring 32-bit setting considering the address and data length (see table below). This is required due to the byte position of the operand is dependent from the operand address. The setting of data length of the control register BCR0 OBS[1:0] could be configured to all ignored. The data length is controllable by mask setting to the BAD0 register implicitely.

On break at BAD[1:0] the respective flags BD[1:0] in the break interrupt request register BIRQ will be set to ’1’. They have to be reset by software in the operand break exception routine.

In Operand data value break mode the Operand Address, causing the break is captured in the BOAC register. Additional BIAC holds the instruction address of the instruction, which was executed one cycle before the break causing data operation. This is normally the instruction, which has caused the data transfer.

In the data value break mode the following important points have to be considered:

1) The data value break is also executed for matching DMA transfers. This could lead to unexpected behaviour due to parallel processes. The filter bits FDMA and FCPU could be set for dedicated investigations.

2) The EDSU data break does not always occur immediately after completion of execution of the instruction causing the break event.

3) Please see also information at chapter 3.4 Using operand with data break

Table 3-4

Access data length	Address set to BAD3/2	MASK set to BAD0	Position of valid data in BAD1/0 (indicated by *)	Remarks
8 bit	4n + 0	0x00FFFFFF	**-- ----	
	4n + 1	0xFF00FFFF	--** ----	
	4n + 2	0xFFFF00FF	---- **--	
	4n + 3	0xFFFFFF00	---- --**	
16 bit	4n + 0	0x0000FFFF	**** ----	Possibly intended to use address mask in BAD3 for address bit 0
	4n + 1	0x0000FFFF	**** ----	
	4n + 2	0xFFFF0000	---- ****	
	4n + 3	0xFFFF0000	---- ****	
32 bit	4n + 0	0x00000000	**** ****	Possibly intended to use address mask in BAD3 for address bits 1 and 0; Data mask not required, two channels could be used
	4n + 1	0x00000000	**** ****	
	4n + 2	0x00000000	**** ****	
	4n + 3	0x00000000	**** ****	

Notes:

1) The mask values for the BAD0 register in the table are a minimum set of bits. Setting more masking bits permits masking bits not needed to be compared with transfer data.

2) "Position of valid data in BAD1, BAD0" provides an 8-bit hexadecimal image for MSB on the left and LSB on the right. Data at bit positions indicated by * in the BAD1, BAD0 registers is compared with data on the data bus, according to the access data length and access address.

3.4 Using operand with data break

Using operand address with data value break together is enabled with setting both EP3 and EP1, and/or both EP2 and EP0 together with setting the bit COMB = ’1’ for the data value break mode set with CTC = ’11’.

In other words: a break in channel 0 will occur at a match on operand address in BAD2 and a match on data value in BAD0. A break in channel 1 will occur at a match on operand address in BAD3 and a match on data value in BAD1. It is not possible to mix them vice versa.

On break both BD0 and BD2, respective BD1 and BD3 are set. They have to be reset by software in the operand break exception routine.

Table 3-5 Operand address and data value break combinations

EP3/2	EP1/0	COMB	Function
0	0	0	No break detection
0	1	0	Independent data break (match value on any operand address)
1	0	0	Independent Operand break (match operand address)
1	1	0	Independent Data break and Operand break
0	0	1	No break detection
0	1	1	No break detection
1	0	1	No break detection
1	1	1	Data value break (match both operand address and value)

3.5 Memory Protection

Due to the availability of address range comparators for the operand and instruction addresses the wish is obvios, to use the same comparator hardware as a memory protection unit (MPU).

Following table list the possible type configurations and its feasibility to be used for memory protection. The number of break points and MPU channels is valid for 8 comparator groups implemented.

Table 3-6 Comparator Type Configuration

CTC	CMP1 Input	CMP0 Input	Max. Break Points (MPE=0)	Max. MPU Channels (MPE=1)
00	IA	IA	32 Instruction breaks	16 ranges with execute permissions
01	OA	OA	32 Operand breaks	16 ranges with read/ write permissions
10	OA	IA	16 + 16 IA/OA breaks	8 ranges with read/ write and execute permissions or 8+8 independent ranges
11	OA	DT	16 data value breaks	-

Additional to the given hardware there were made some extensions to provide the user with a more likely configuration of read, write and execute permissions instead of the more bus applicable definition of the operand break size and type definitions OBS/OBT, including read-modify-write. With the introduction of the SuperVisor mode a definition of User and SuperVisor permissions is possible.

Permissions can be set for the comparator channel CMP1 and CMP0 separately, indicated by the symbol index.

Table 3-7 Meaning of the permission config bits

Symbol	Data Mode (OA)	Instruction Mode (IA)
SRX[1:0]	SuperVisor Read permission	SuperVisor eXecute permission
SW[1:0]	SuperVisor Write permission	-
URX[1:0]	User Read permission	User eXecute permission
UW[1:0]	User Write permission	-

At each time an instruction is executed or an operand is accessed, the actual valid permissions were evaluated. This evaluation is divided into operand access (OA-based) and code execution (IA-based).

For each part the highest priority region hit is searched for. The highest channel number has the highest priority (strict priority scheme). If a channel hit was found, the permissions defined for this channel will apply. If no channel hit was detected, the default permissions apply.

After the actual permissions are evaluated (valid for actual data access, if any, and actual instruction) the permissions were checked. If the execute permission is not set or if the read or write permissions do not fit to the type of the actual access, a protection violation will be indicated. This causes a CPU trap to the memory protection violation MPUPV handler routine. The CPU switches directly to SuperVisor mode in this case.

The config register space of the EDSU is protected against random access in User mode. Only in SuperVisor mode or Emulation mode the register file enables write access. For configuration a system interrupt INT #5 was defined, which switches in SuperVisor mode (SV bit remains set during the execution of the INT #5-ISR). Except debugger interrupts by the emulator and NMI the SuperVisor ISR is not interruptible.

Exceptions caused by the memory protection and the break unit for debugging are separated. In that way the memory protection functionality can be debugged itself.

3.6 Break Factors

Summary of the internal break factors and the executed events:

Break on instruction address	->	Causes Instruction Break
Break on operand address	->	Causes Operand Break
Break on data value	->	Causes Operand Break
Resource Interrupt (BREAK)	->	Causes Tool NMI
Step Trace Trap	->	Causes Step Trace Trap
Execution of the INTE instr.	->	Causes INTE
Execution of INT #5	->	CPU SuperVisor Mode
Memory protection exception	->	Causes MPUPV Trap

Break factors and corresponding interrupt numbers and vectors:

Table 3-8 Interrupt numbers and vectors of break factors

Interrupt	Interrupt number		Interrupt level		Interrupt vector	
	Decimal	Hexa-decimal	Setting Register	Register address	Offset	Default Vector address
CPU supervisor mode (INT #5 instruction)	5	05	-	-	0x3E8	0x000FFFE8
Memory protection exception	6	06	-	-	0x3E4	0x000FFFE4
INTE instruction	9	09	-	-	0x3D8	0x000FFFD8
Instruction break exception	10	0A	-	-	0x3D4	0x000FFFD4
Operand break exception	11	0B	-	-	0x3D0	0x000FFFD0
Step trace trap	12	0C	-	-	0x3CC	0x000FFFCC
NMI interrupt (tool)	13	0D	-	-	0x3C8	0x000FFFC8

4. Registers

4.1 List of EDSU Registers

Table 4-1 EDSU Registers Summary

Address	Register				Block
	+0	+1	+2	+3	
$F000_H$	BCTRL [R/W] -------- -------- 11111100 00000000				EDSU
$F004_H$	BSTAT [R/W0][1] -------- -----000 00000000 10--0000				
$F008_H$	BIAC [R] 00000000 00000000 00000000 00000000				
$F00C_H$	BOAC [R] 00000000 00000000 00000000 00000000				
$F010_H$	BIRQ [R/W] 00000000 00000000 00000000 00000000				
$F014_H$...$F01F_H$	reserved -				
$F020_H$	BCR0 [R/W] -------- 00000000 00000000 00000000				
$F024_H$	BCR1 [R/W] -------- 00000000 00000000 00000000				
$F028_H$	BCR2 [R/W] -------- 00000000 00000000 00000000				
$F02C_H$	BCR3 [R/W] -------- 00000000 00000000 00000000				
$F030_H$	BCR4 [R/W] -------- 00000000 00000000 00000000				
$F034_H$	BCR5 [R/W] -------- 00000000 00000000 00000000				
$F038_H$	BCR6 [R/W] -------- 00000000 00000000 00000000				
$F03C_H$	BCR7 [R/W] -------- 00000000 00000000 00000000				
$F040_H$...$F07F_H$	reserved -				

Table 4-1 EDSU Registers Summary

Address	Register				Block
	+0	+1	+2	+3	
$F080_H$	BAD0 [R/W] XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX				EDSU
$F084_H$	BAD1 [R/W] XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX				
$F088_H$	BAD2 [R/W] XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX				
$F08C_H$	BAD3 [R/W] XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX				
$F090_H$	BAD4 [R/W] XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX				
$F094_H$	BAD5 [R/W] XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX				
$F098_H$	BAD6 [R/W] XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX				
$F09C_H$	BAD7 [R/W] XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX				
$F0A0_H$	BAD8 [R/W] XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX				
$F0A4_H$	BAD9 [R/W] XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX				
$F0A8_H$	BAD10 [R/W] XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX				
$F0AC_H$	BAD11 [R/W] XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX				
$F0B0_H$	BAD12 [R/W] XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX				
$F0B4_H$	BAD13 [R/W] XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX				
$F0B8_H$	BAD14 [R/W] XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX				
$F0BC_H$	BAD15 [R/W] XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX				

Table 4-1 EDSU Registers Summary

Address	Register				Block
	+0	+1	+2	+3	
$F0C0_H$	BAD16 [R/W] XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX				EDSU
$F0C4_H$	BAD17 [R/W] XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX				
$F0C8_H$	BAD18 [R/W] XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX				
$F0CC_H$	BAD19 [R/W] XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX				
$F0D0_H$	BAD20 [R/W] XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX				
$F0D4_H$	BAD21 [R/W] XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX				
$F0D8_H$	BAD22 [R/W] XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX				
$F0DC_H$	BAD23 [R/W] XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX				
$F0E0_H$	BAD24 [R/W] XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX				
$F0E4_H$	BAD25 [R/W] XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX				
$F0E8_H$	BAD26 [R/W] XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX				
$F0EC_H$	BAD27 [R/W] XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX				
$F0F0_H$	BAD28 [R/W] XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX				
$F0F4_H$	BAD29 [R/W] XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX				
$F0F8_H$	BAD30 [R/W] XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX				
$F0FC_H$	BAD31 [R/W] XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX				

1.RMW - read returns '1' for each flag, for write only '0' (clear) is supported.

Remark: Read and write access to all registers is byte, halfword and word.

4.2 Explanations of Registers

● EDSU Control Register (BCTRL)

EDSU Control Register byte 2

	15	14	13	12	11	10	9	8	⇐ Bit no.
Address : F002H	SR	SW	SX	UR	UW	UX	FCPU	FDMA	
Read/write ⇒	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	
Default value⇒	(1)	(1)	(1)	(1)	(1)	(1)	(0)	(0)	

EDSU Control Register byte 3

	7	6	5	4	3	2	1	0	⇐ Bit no.
Address : F003H	EEMM	PFD	SINT1	SINT0	EINT1	EINT0	EINTT	EINTR	
Read/write ⇒	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	
Default value⇒	(0)	(0)	(0)	(0)	(0)	(0)	(0)	(0)	

Default Permission Register

The default permission register defines the lowest priority access permissions for the whole memory and I/O address range of the MCU. Lowest priority means, that the default permissions take effect for all address regions, which are NOT covered by any dedicated channel configuration, operating in MPU mode. Default read, write and execute permissions could be defined for the super visior mode (SV=1) and the normal user mode (SV=0). The super visor mode (SV) is indicated by bit 6 of the CCR in the program status word of the CPU. After the INIT condition all permissions are set (access allowed).

BIT[15]: SR - Super visor default Read permission register

0	Super visor is not permitted to read data
1	Super visor is permitted to read data (default)

BIT[14]: SW - Super visor default Write permission register

0	Super visor is not permitted to write data
1	Super visor is permitted to write data (default)

BIT[13]: SX - Super visor default eXecute permission register

0	Super visor is not permitted to execute code
1	Super visor is permitted to execute code (default)

BIT[12]: UR - User default Read permission register

0	User is not permitted to read data
1	User is permitted to read data (default)

BIT[11]: UW - User default Write permission register

0	User is not permitted to write data
1	User is permitted to write data (default)

BIT[10]: UX - User default eXecute permission register

0	User is not permitted to execute code
1	User is permitted to execute code (default)

CPU and DMA Filter Option Register

BIT[9]: FCPU - Filter CPU access

0	Trigger on CPU accesses (default)
1	Do not trigger on CPU accesses

FCPU controls the filter operation for CPU accesses triggered by operand compare channels (Operand address break, data value break and memory data protection).

If FCPU is set to ’1’, all CPU acesses are masked out. If set to ’0’ CPU acesses can cause break function.

BIT[8]: FDMA - Filter DMA access

0	Trigger on DMA accesses (default)
1	Do not trigger on DMA accesses

FDMA controls the filter operation for DMA accesses triggered by operand compare channels (Operand address break, data value break and memory data protection).

If FDMA is set to ’1’, all DMA acesses are masked out. If set to ’0’ DMA acesses can cause break function.

Important Note for FDMA: Only DMA accesses over D-Bus were detected. The operands for an explicite DMA trigger condition have to be located in the D-Bus address area (This is the case for D-bus RAM, CAN and all R-Bus resources in the MB91460 family). Otherwise the DMA transfer could not be recognized by the EDSU. This function was mainly intendet to disable the trigger on DMA accesses (filter out the operand change condition by DMA), complete address range DMA trigger conditions are not supported.

Enable Interrupt Register

BIT[7]: EEMM - Enable Emulation Mode

0	Disable emulation mode (default)

1	Enable emulation mode

If EEMM is set to ’1’ then the emulation mode is entered during Step Trace Mode and EDSU exceptions Instruction Break, Operand Break and Tool NMI. During emulation mode the Watchdog Timer (WDT) is disabled. EDSU triggered emulation mode is left with the RETI instruction.

Set to ’0’ disables emulation mode function. The WDT is not stopped during Step Trace and EDSU exceptions.

BIT[6]: PFD - Phantom Filter Disable

0	Instruction break detection uses phantome filter (default)
1	Phantome Filter disabled

The default (PFD=0) is to use policies to filter out phantom interrupts and wrong status bits, which may be set in addition.

- The instruction fetched, after RETI was executed, is normally the instruction on which the break point was set. Fetch is repeated after processing the breakpoint handler ISR before executing the instruction at the break point. The filter avoids that the trigger of the break condition will be repeated.
- Not granted Instruction Break exceptions are timed out
  - — Pre-fetched, but not executed commands
  - — Commands after delayed slot instruction
- Consecutive break conditions which are pre-fetched are not allowed to set flags. Only the instruction at which the break condition occures at first time can set status bits accordingly.
- Nested Instruction breaks are not allowed (break within the break handler ISR)

BIT[5:4]: SINT[1:0] - Select resource INTerrupt source

SINT[1:0]	MB91V460	Resource
00	Tool NMI by interrupt on source 0 selected (default)	UART 0 RX / UART 0 TX
01	Tool NMI by interrupt on source 1 selected	UART 1 RX / UART 1 TX
10	Tool NMI by interrupt on source 2 selected	UART 2 RX / UART 2 TX
11	Tool NMI by interrupt on source 3 selected	CAN 0 / CAN 1

SINT[1:0]	MB91F467DA	Resource
00	Tool NMI by interrupt on source 0 selected (default)	UART 2 RX / UART 2 TX
01	Tool NMI by interrupt on source 1 selected	UART 4 RX / UART 4 TX
10	Tool NMI by interrupt on source 2 selected	UART 5 RX / UART 5 TX
11	Tool NMI by interrupt on source 3 selected	CAN 0 / CAN 1

SINT1 and SINT0 select the active resource interrupt source.

BIT[3]: EINT1 - Enable extended INTerrupt 1

0	Disable extended interrupt source 1 (default)
1	Enable extended interrupt source 1

If EINT1 is set to ’1’ then a Tool NMI will be generated on an extended interrupt event at source channel 1. Set to ’0’ disables this function.

Remark: EINT1 interrupt source is not available for the MB91460 series.

BIT[2]: EINT0 - Enable extended INTerrupt 0

0	Disable extended interrupt source 0 (default)
1	Enable extended interrupt source 0

If EINT0 is set to ’1’ then a Tool NMI will be generated on an extended interrupt event at source channel 0. Set to ’0’ disables this function.

Remark: EINT1 and EINT0 can be used for indicating a signal line event which can be used for generating a BREAK function. The sources of these interrupts are hardwired in the MCU and can be for example: external interrupt ports, general purpose I/O port pins, other resources, etc. This has to be defined in the device specification.

BIT[1]: EINTT - Enable INTerrupt on Transmit

0	Disable transmit interrupt source channels 0 to 3 (default)
1	Enable transmit interrupt source channels 0 to 3

If EINTT is set to ’1’ then a Tool NMI will be generated on a transmit interrupt event at source channels 0 to 3 set by TXINT[1:0]. Setting EINTT to ’0’ disables this function.

Remark: If SINT[1:0] is set to “11” this bit enables the interrupt of CAN channel 1 (CAN has one interrupt request for both reception and transmission).

BIT[0]: EINTR - Enable INTerrupt on Receive

0	Disable receive interrupt source channels 0 to 3 (default)
1	Enable receive interrupt source channels 0 to 3

If EINTR is set to ’1’ then a Tool NMI will be generated on a receive interrupt event at source channels 0 to 3 set by RXINT[1:0]. Setting EINTR to ’0’ disables this function.

Remark: If SINT[1:0] is set to “11” this bit enables the interrupt of CAN channel 0 (CAN has one interrupt request for both reception and transmission).

● EDSU Status Register (BSTAT)

EDSU Status Register byte 2

	15	14	13	12	11	10	9	8	⇐ Bit no.
Address : F006H	IDX4	IDX3	IDX2	IDX1	IDX0	CDMA	CSZ1	CSZ0	
Read/write ⇒	(R)	(R)	(R)	(R)	(R)	(R)	(R)	(R)	
Default value⇒	(0)	(0)	(0)	(0)	(0)	(0)	(0)	(0)	

EDSU Status Register byte 3

	7	6	5	4	3	2	1	0	⇐ Bit no.
Address : F007H	CRW1	CRW0	PV	RST	INT1	INT0	INTT	INTR	
Read/write ⇒	(R)	(R)	(R/W)	(R/W)	(R/W)	(R/W)	(R)	(R)	
Default value⇒	(0)	(0)	(0)	(1)	(0)	(0)	(0)	(0)	

BIT[15:11]: IDX[4:0] - Channel Index Indication of MPUPV Trigger

In the case of triggering a memory protection violation (MPUPV), the index of the channel pair 0...15 is saved in The IDX register, which caused the trigger. The channel pairs are normally used as range comparators.

If no MPU chanel has detected a hit on its address range, the default permissions apply. If the default permissions are violated, IDX is set to the value 16 (overrun). If the permissions of a matching MPU channel are violated, IDX shows the index of the appropriate break detection bits BIRQ_BD[31:0]. The break detection bits belonging tho this comparator are BD[2*IDX] and BD[2*IDX+1].

In case of multiple range hits and/or trigger conditions, the channel with the highest priority trigger condittion is indicated by IDX[4:0]. The priority raises with the channel index.

IDX	Description
0-15	Points to the channel number of the last protection violation
16	The last protection violation was caused by the violation of the default permissions

The channel index indication register can be read only.

Access Type Capture Register

In case of a trap caused by a memory protection violation or an operand/data value break condition, the status bits [12:8] capture type information about the break causing operand access. In case of a memory protection fault due to the violation of execution permissions, this information is also captured, regardless if there was an active operand access or not.

Access type capture register are read only.

BIT[10]: CDMA - Capture DMA Indication

0	The operand access was executed by the CPU
1	The operand access was executed by the DMA controller

BIT[9:8]: CSZ[1:0] - Capture Operand Size

00	The operand has a bit size of 8
01	The operand has a bit size of 16
10	The operand has a bit size of 32
11	reserved

BIT[7:6]: CRW[1:0] - Capture Operand Access Type

00	The operand has been read
01	The operand has been read by read-modify-write indicated
10	The operand has been written
11	no operand access

BIT[5]: PV - Protection Violation Detection

0	There was no protection violation on read, write and execute permissions
1	A protection violation (MPUPV) has been occured

If this bit is set after a protection violation, a MPUPV trap is indicated to the CPU. The occurance of a protection violation means, that there was a read or write access to a defined address region, which was not permitted or code was executed without execute permissions for this address region. As consequence the CPU switches to super visor mode (SV=1) and calls the handler routine for interrupt number #6 (see table 3-8).

This bit should be cleared by writing ’0’ in the MPUPV trap handler routine.

BIT[4]: RST - Operation Initialization Reset (RST) Detection

The reset operation of FRex family is divided into two levels, setting initialization reset (INIT) and operation initialization reset (RST). When INIT occurs, RST occures at the same time implicitely.

0	Operation Reset was not triggered since last BSTAT read or clear
1	Operation Reset was triggered since last BSTAT read or clear

The RST bit is read only, any write access to this bit will be ignored. RST is cleared after BSTAT is read (read from any byte address within the 32 bit word). RST has same behaviour for read and read-modify-write access.

The RST bit can be used for reset detection. It is set in any case of operation initialization reset is triggered. Debug monitor software can use this to detect if the communication device to the debugger front end needs to be re-configured after an operation reset. This is important for debugging of boot procedures and soft reset handling. After reading the EDSU status word the RST bit is cleared automatically.

Break Interrupt Register

BIT[3]: INT1 - INTerrupt on extended source 1

0	Interrupt on extended source channel 1 not detected (default)
1	Interrupt on extended source channel 1 detected

INT1 reflects the status of the extended interrupt source channel 1. It is set to ’1’ if a high level on the extended interrupt signal line has been occured. The status of ’1’ is stored until cleared by software.

Writing ’0’ resets the INT1 bit to ’0’. Writing ’1’ to this bit is ignored. On a Read Modify Write instruction INT1 is read as ’1’.

BIT[2]: INT0 - INTerrupt on extended source 0

0	Interrupt on extended source channel 0 not detected (default)
1	Interrupt on extended source channel 0 detected

INT0 reflects the status of the extended interrupt source channel 0. It is set to ’1’ if a high level on the extended interrupt signal line has been occured. The status of ’1’ is stored until cleared by software.

Writing ’0’ resets the INT0 bit to ’0’. Writing ’1’ to this bit is ignored. On a Read Modify Write instruction INT0 is read as ’1’.

BIT[1]: INTT - INTerrupt on Transmit source

0	Interrupt on transmit source not detected (default)
1	Interrupt on transmit source channel detected

INTT reflects the status of the transmit interrupt source channels 0 to 3 (can be selected by TXINT[1:0]). It is set to ’1’ on a high level on the transmit interrupt signal line and ’0’ on a low level on the signal line.

This bit is read-only. It can be set to ’0’ by clearing the appropriate interrupt bit in the selected resource.

Remark: If SINT[1:0] is set to “11” this bit indicates the interrupt of CAN channel 1 (CAN has one interrupt request for both reception and transmission).

BIT[0]: INTR - INTerrupt on Receive source

0	Interrupt on receive source not detected (default)
1	Interrupt on receive source channel detected

INTR reflects the status of the receive interrupt source channels 0 to 3 (can be selected by RXINT[1:0]). It is set to ’1’ on a high level on the receive interrupt signal line and ’0’ on a low level on the signal line.

This bit is read-only. It can be set to ’0’ by clearing the appropriate interrupt bit in the selected resource.

Remark: If SINT[1:0] is set to “11” this bit indicates the interrupt of CAN channel 1 (CAN has one interrupt request for both reception and transmission).

● EDSU Instruction Address Capture Register (BIAC)

BIAC [R]				
Address	**+0**	**+1**	**+2**	**+3**
$F008_H$	00000000	00000000	00000000	00000000

This register captures the address of the instruction (IA), which has caused the protection violation or the operand/ data value break. This register could be read only.

● EDSU Operand Address Capture Register (BOAC)

BOAC [R]				
Address	**+0**	**+1**	**+2**	**+3**
$F00C_H$	00000000	00000000	00000000	00000000

This register captures the address of the operand access (OA), which has caused the protection violation or the operand/data value break. This register could be read only.

● EDSU Break Detection Interrupt Request Register (BIRQ)

BIRQ [R/W]				
Address	**+0**	**+1**	**+2**	**+3**
$F010_H$	00000000	00000000	00000000	00000000

BIRQ collects all break detection bits of all channels, regardless of the type configuration of each channel. The actual implementation consists of 8 groups of channels, that are 32 single point channels totally.

Each group of channels consists of 4 channels and 4 bits for break detection in the BIRQ register. Each group has two comparator pairs. Each pair consists of two point comparators which could build a range comparator by setting the range enable bit. Such a range comparator pair is connected to the instruction address, operand address or the data value information - selected by the comparator type configuration.

For detection of combined operand address and data value breaks two of such comparator pairs are combined together. Than the break detection (BD) bits are set only if both conditions are matching simultaneously.

BIT[31:0]: BD[31:0] - Break Detection register

0	Break factor not detected (default)
1	Break factor detected on channel according the bit position [31:0]

BD[31:0] reflects the status of the break detection. It is set to ’1’ at match with BAD31...BAD0 accordingly (and if the mask condition is satisfied, if enabled by EM1/0). For bit pairs [31:30], [29:28], ..., [1:0] range matches could apply, if the range function using two points is enabled by ER1/0.

Break factors could be

- instruction address break,

- operand address break,
- data value break,
- combined operand address and data value break and
- memory protection violation.

Writing ’0’ resets the BD[31:0] bits to ’0’. Writing ’1’ to these bits is ignored. On a Read Modify Write instruction all BD bits are read as ’1’.

BD1/BD0 setting at enabled address range function (also valid for the other pairs of BD bits in neighbourhood):
If the operand address range function is enabled with ER0 in addition to the point enables EP1 and EP0, then the BD1 and BD0 detection bits are set in the following manner:

Table 4-2 BD Coding for Match on Start/Endpoint or Range

BD1	BD0	Compare value: Instruction, Operand Address, Data Value
0	0	No match (Default)
0	1	Match on point (compare value == BAD0)
1	0	Match on point (compare value == BAD1)
1	1	Match on range (BAD0 < compare value < BAD1)

● EDSU Channel Configuration Register (BCR0...BCR7)

EDSU Ch. Config Register 0, byte 0

	31	30	29	28	27	26	25	24	⇐ Bit no.
Address : F020H	-	-	-	-	-	-	-	-	
Read/write ⇒	(-)	(-)	(-)	(-)	(-)	(-)	(-)	(-)	
Default value⇒	(X)	(X)	(X)	(X)	(X)	(X)	(X)	(X)	

EDSU Ch. Config Register 0, byte 1

	23	22	21	20	19	18	17	16	⇐ Bit no.
Address : F021H	SRX1	SW1	SRX0	SW0	URX1	UW1	URX0	UW0	
Read/write ⇒	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	
Default value⇒	(0)	(0)	(0)	(0)	(0)	(0)	(0)	(0)	

EDSU Ch. Config Register 0, byte 2

	15	14	13	12	11	10	9	8	⇐ Bit no.
Address : F022H	MPE	COMB	CTC1	CTC0	OBS1	OBS0	OBT1	OBT0	
Read/write ⇒	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	
Default value⇒	(0)	(0)	(0)	(0)	(0)	(0)	(0)	(0)	

EDSU Ch. Config Register 0, byte 3

	7	6	5	4	3	2	1	0	⇐ Bit no.
Address : F023H	EP3	EP2	EP1	EP0	EM1	EM0	ER1	ER0	
Read/write ⇒	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	
Default value⇒	(0)	(0)	(0)	(0)	(0)	(0)	(0)	(0)	

For each group of four channels one channel configuration register (BCR0...BCR7) is implemented. It holds the configuration set for the according group of channels. The following table shows the relationship, which channel configuration, break point address/data registers and break detection bits belong together.

Table 4-3 Relationship of BCR, BAD and BIRQ registers

Group Config	Address/Data	BADx Usage	Point	Mask	Combination		BIRQ
BCR0	BAD0	Point0, Mask0	EP0	EM0	range 0 ER0	OA0	BD0
	BAD1	Point1	EP1			OA1	BD1
	BAD2	Point2, Mask1	EP2	EM1	range 1 ER1	DT0	BD2
	BAD3	Point3	EP3			DT1	BD3
BCR1	BAD4	Point0, Mask0	EP0	EM0	range 0 ER0	OA0	BD4
	BAD5	Point1	EP1			OA1	BD5
	BAD6	Point2, Mask1	EP2	EM1	range 1 ER1	DT0	BD6
	BAD7	Point3	EP3			DT1	BD7
BCR2	BAD8	Point0, Mask0	EP0	EM0	range 0 ER0	OA0	BD8
	BAD9	Point1	EP1			OA1	BD9
	BAD10	Point2, Mask1	EP2	EM1	range 1 ER1	DT0	BD10
	BAD11	Point3	EP3			DT1	BD11

Table 4-3 Relationship of BCR, BAD and BIRQ registers

Group Config	Address/Data	BADx Usage	Point	Mask	Combination		BIRQ
BCR3	BAD12	Point0, Mask0	EP0	EM0	range 0 ER0	OA0	BD12
	BAD13	Point1	EP1			OA1	BD13
	BAD14	Point2, Mask1	EP2	EM1	range 1 ER1	DT0	BD14
	BAD15	Point3	EP3			DT1	BD15
BCR4	BAD16	Point0, Mask0	EP0	EM0	range 0 ER0	OA0	BD16
	BAD17	Point1	EP1			OA1	BD17
	BAD18	Point2, Mask1	EP2	EM1	range 1 ER1	DT0	BD18
	BAD19	Point3	EP3			DT1	BD19
BCR5	BAD20	Point0, Mask0	EP0	EM0	range 0 ER0	OA0	BD20
	BAD21	Point1	EP1			OA1	BD21
	BAD22	Point2, Mask1	EP2	EM1	range 1 ER1	DT0	BD22
	BAD23	Point3	EP3			DT1	BD23
BCR6	BAD24	Point0, Mask0	EP0	EM0	range 0 ER0	OA0	BD24
	BAD25	Point1	EP1			OA1	BD25
	BAD26	Point2, Mask1	EP2	EM1	range 1 ER1	DT0	BD26
	BAD27	Point3	EP3			DT1	BD27
BCR7	BAD28	Point0, Mask0	EP0	EM0	range 0 ER0	OA0	BD28
	BAD29	Point1	EP1			OA1	BD29
	BAD30	Point2, Mask1	EP2	EM1	range 1 ER1	DT0	BD30
	BAD31	Point3	EP3			DT1	BD31

Group of Channels, Permission Definition Register

The permission definition registers are valid only for the group of channels operating in MPU mode. This is the case if MPE is set to ’1’. If the group does not operate in MPU mode, the permission configuration is not required (don’t care).

Normally MPU channels operate in range mode for the address definitions.

The type of the permission, which could be set-up, depends on the comparator type configuration (CTC) for each comparator pair. MPU channels could be configured either to check instruction addresses (IA) or operand addresses (OA). IA ranges could be used to define exceute permissions. OA ranges could be used to define read and write permissions.

The comparator type for MPU usage could be set to

- CTC=0: both IA ranges define execute permissions,
- CTC=1: both OA ranges define read/write permissions and

- CTC=2: IA range 0 defines execute permissions and OA range 1 defines read/write permissions.

Data value (DT) detection by setting CTC=3 is not possible to use in MPU mode.

Permission configurations exist for read, write and execute for two CPU modes, the super visor mode and the user mode. Super visor permissions are valid for SV=1 and user permissions are valid for SV=0.

BIT[23]: SRX1 - Super visor Read/eXecute permission register for range 1

Setting valid for CTC == 0 (Instruction address range comparator):

0	Super visor has no execute permission on address range 1(default)
1	Super visor has execute permission on address range 1

Setting valid for CTC == 1 or CTC == 2 (Operand address range comparator):

0	Super visor has no read permission on address range 1 (default)
1	Super visor has read permission on address range 1

BIT[22]: SW1 - Super visor Write permission register for range 1

Setting valid for CTC == 1 or CTC == 2 (Operand address range comparator):

0	Super visor has no write permission on address range 1 (default)
1	Super visor has write permission on address range 1

BIT[21]: SRX0 - Super visor Read/eXecute permission register for range 0

Setting valid for CTC == 0 or CTC == 2 (Instruction address range comparator):

0	Super visor has no execute permission on address range 0 (default)
1	Super visor has execute permission on address range 0

Setting valid for CTC == 1 (Operand address range comparator):

0	Super visor has no read permission on address range 0 (default)
1	Super visor has read permission on address range 0

BIT[20]: SW1 - Super visor Write permission register for range 0

Setting valid for CTC == 1 (Operand address range comparator):

0	Super visor has no write permission on address range 0 (default)
1	Super visor has write permission on address range 0

BIT[19]: URX1 - User Read/eXecute permission register for range 1

Setting valid for CTC == 0 (Instruction address range comparator):

0	User has no execute permission on address range 1(default)
1	User has execute permission on address range 1

Setting valid for CTC == 1 or CTC == 2 (Operand address range comparator):

0	User has no read permission on address range 1 (default)
1	User has read permission on address range 1

BIT[18]: UW1 - User Write permission register for range 1

Setting valid for CTC == 1 or CTC == 2 (Operand address range comparator):

0	User has no write permission on address range 1 (default)
1	User has write permission on address range 1

BIT[17]: URX0 - User Read/eXecute permission register for range 0

Setting valid for CTC == 0 or CTC == 2 (Instruction address range comparator):

0	User has no execute permission on address range 0 (default)
1	User has execute permission on address range 0

Setting valid for CTC == 1 (Operand address range comparator):

0	User has no read permission on address range 0 (default)
1	User has read permission on address range 0

BIT[16]: UW1 - User Write permission register for range 0

Setting valid for CTC == 1 (Operand address range comparator):

0	User has no write permission on address range 0 (default)
1	User has write permission on address range 0

Group of Channels, Mode Configuration Register

BIT[15]: MPE - Memory Protection Enable

0	The group of channels operates as debug interface and defines breakpoints (default)

1	The group of channels operates in memory protection mode

Some restrictions apply with the setting of the MPE bit.

MPE=0 (break unit):

- permission registers are don’t care (BCRx bits [23:16])

MPE=1 (memory protection unit):

- OBS and OBT should be set to ’3’ (BCRx bits [11:8], any size and any type)
- CTC should not be set to ’3’ (BCRx bits [13:12], data value check not supported in this mode)

BIT[14]: COMB - Channel Combination Enable

0	No combination between channels (default)
1	Combination between channels is effective

Depending on the MPE configuration bit the COMB feature has different meaning.

(A) COMB=1 and MPE=0 (break unit, combined operand address and data value break):

The break detection conditions are combined before setting the BIRQ_BD bits and signalizing an operand break condition. Setting the COMB bit is required for defining a data value break on a specific operand address. If the COMB bit is set to ’1’, both conditions, matching operand address (OA) and matching data value (DT), are required to be true. Setting the COMB bit makes only sense in the OA/DT mode, defined by CTC=3.

The AND-combination is effective between channels 3 (OA1) and 1 (DT1) and between channels 2 (OA0) and 0 (DT0). It is assumed that no range operation is defined (ER1=ER0=0).

```
BIRQ_BD3 = BIRQ_BD1 = BD3 && BD1;
BIRQ_BD2 = BIRQ_BD0 = BD2 && BD0;
```

If channels 3 and 2 define an operand address range (OA1:OA0) by setting ER1=1 and/or channels 1 and 0 define a data value range (DT1:DT0) by setting ER0=1, the break detection bits of each channel are AND-combined with the ORed channels of the opposite range comparator break detection outputs.

```
BIRQ_BD3 = BD3 && (BD1 || BD0);
BIRQ_BD2 = BD2 && (BD1 || BD0);
BIRQ_BD1 = BD1 && (BD3 || BD2);
BIRQ_BD0 = BD0 && (BD3 || BD2);
```

This offers the same interpretation of the BIRQ break detction bits (see table 4-2 for coding of match on start point, range or end point) as it would be the case for range detection with COMB=0. BD3 and BD2 hold the coding for the operand address (OA) match, whereas BD1 and BD0 hold the coding for the data value (DT) match. The COMB bit set to ’1’ ensures that both conditions, the OA match and the DT match must be true to set the appropriate BD bit in the end.

If the COMB bit is set to ’0’ all break detection bits are passed to the BIRQ register in it’s original form. The comparator channels match conditions are independent from each other.

(B) COMB=1 and MPE=1 (memory protection unit, combined rwx permissions on single range):

In memory protection mode the COMB bit has the meaning of combined data read/write and code execute permissions, set for the same address range. The setting is only meaningful for the combination of operand address (OA) comparators on channels 3 and 2 and instruction address (IA) comparators on channels 1 and 2 in the mode CTC=2.

The COMB bit set to ’1’ causes the IA comparator CMP0 to use the same BADx point definitions as the OA comparator CMP1. Point 3 and Point 2 define the address range for both comparators CMP0 and CMP1. This has the effect that the entry of Point 0/Mask 0 is not allocated for the Point set-up and could be used for masking either one or both comparators. The Point 1 entry is not useable in this case.

If the COMB bit is set to ’0’ both comparators have independent address configurations. The comparators can either define read/write permissions for data protection or define execute permissions for code protection. Each comparator can define an address region by a range between two points (ER=1) or by one point with a mask (EM=1).

BIT[13:12]: CTC[1:0] - Comparator Type Config

CTC	CMP1	CMP0	Break Function	MPU Function
0	IA	IA	4 instruction break points	2 regions for code protection (x permissions)
1	OA	OA	4 operand break points	2 regions for data protection (rw permissions)
2	OA	IA	2 instruction break points + 2 operand break points	1 region for code protection (x permissions) and 1 region for data protection (rw permissions) or 1 region for combined code and data protection (rwx permissions)
3	OA	DT	2 operand break points + 2 data value breaks, normally with combination	not applicable

One group of channels contains 2 range comparator blocks. Each comparator block can detect a range hit between two points or two independent point hits. The point configuration is stored in dedicated BADx registsers for each channel (4 BADx registers for each group of channels).

The comparator type configuration (CTC) controls the input multiplexing of the compare value for each of the two range comparator blocks CMP1 and CMP0. CMP1 combines the break detection channels 3 and 2. The compare value for CMP1 can be assigned either to the instruction address (IA) or to the operand address (OA). CMP0 combines the break detection channels 1 and 0. The compare value for CMP0 can be assigned to the instruction address (IA), to the operand address (OA) or to the data value (DT). The table above defines the input compare values for CMP1 and CMP0, depending on the CTC setting.

In addition a mask for each comparator block could be defined (see the definition of the EM bits later). In this case the BADx register, which contains the mask information, is not available for the point configuration. Thus the usage of the mask feature restricts the number of points or channels, which are available in total.

BIT[11:8]: OBS[1:0], OBT[1:0] - Operand Break Size / Operand Break Type register 1

Datasize		
OBS1	OBS0	
0	0	Byte (Default)
0	1	Halfword
1	0	Word

Access type		
OBT1	OBT0	
0	0	Read (Default)
0	1	Read-Modify-Write
1	0	Write

Datasize		
OBS1	**OBS0**	
1	1	All (Byte, Hword, Word)

Access type		
OBT1	**OBT0**	
1	1	All (Read, RMW, Write)

The operand break size register OBS configures the datasize and the operand break type register OBT configures the access type if the channel is configured to operand address break or data value break detection.

Setting to 'all' in datasize will cause detection of byte, halfword and word data sizes. Setting to 'all' in access type will cause detection of Read, Read-Modify-Write and Write access types.

Enable Break Point Register

BIT[7]: EP3 - Enable break Point 3 register

0	Break point 3 register is disabled (default)
1	Break point 3 register is enabled

If EP3 is enabled then the input value of CMP1 will be compared with the point 3 register content (BAD index = 3+group offset, BAD3 for group 0 channel 3, BAD7 for group 1 channel 3, ...).

The input value and the point value is masked if the mask function is enabled by EM1. On a compare match a break exception will be executed. CTC and MPE control the selection of the input value and the type of the break exception.

BIT[6]: EP2 - Enable break Point 2 register

0	Break point 2 register is disabled (default)
1	Break point 2 register is enabled

If EP2 is enabled then the input value of CMP1 will be compared with the point 2 register content (BAD index = 2+group offset, BAD2 for group 0 channel 2, BAD6 for group 1 channel 2, ...).

The input value and the point value is masked if the mask function is enabled by EM1. On a compare match a break exception will be executed. CTC and MPE control the selection of the input value and the type of the break exception.

EP2 controls in addition to enabling and allocating point 2 the selection of the mask register. Point 2 is also the default place for storing the CMP1 mask value. But, if point 2 is enabled, the mask could not be stored there and the mask input of CMP1 switches to point 0 (to the opposite comparator).

BIT[5]: EP1 - Enable break Point 1 register

0	Break point 1 register is disabled (default)
1	Break point 1 register is enabled

If EP1 is enabled then the input value of CMP0 will be compared with the point 1 register content (BAD index = 1+group offset, BAD1 for group 0 channel 1, BAD5 for group 1 channel 1, ...).

The input value and the point value is masked if the mask function is enabled by EM0. On a compare match a break exception will be executed. CTC and MPE control the selection of the input value and the type of the break exception.

BIT[4]: EP0 - Enable break Point 0 register

0	Break point 0 register is disabled (default)
1	Break point 0 register is enabled

If EP0 is enabled then the input value of CMP0 will be compared with the point 0 register content (BAD index = 0+group offset, BAD0 for group 0 channel 0, BAD4 for group 1 channel 0, ...).

The input value and the point value is masked if the mask function is enabled by EM0. On a compare match a break exception will be executed. CTC and MPE control the selection of the input value and the type of the break exception.

EP0 controls in addition to enabling and allocating point 0 the selection of the mask register. Point 0 is also the default place for storing the CMP0 mask value. But, if point 0 is enabled, the mask could not be stored there and the mask input of CMP0 switches to point 2 (to the opposite comparator).

If memory protection is enabled (MPE=1) in conjunction with the combination bit set (COMB=1), the address range is defined by point 3 and point 2 and is valid for both comparators COMB1 and COMB0. So the points 1 and 0 are not required for the range definition of CMP0, independent from the point enable EP0 and EP1, which normally are set in this case. Thus point 0 could be used for storing the mask value for both comparators CMP1 and CMP0 and the exception described in the paragraph above did not apply for this case.

Enable Mask And Range Register

BIT[3]: EM1 - Enable Mask for CMP1

0	Mask function for CMP1 is disabled (default)
1	Mask function for CMP1 is enabled

If EM1 is enabled the comparator CMP1 matches only these bit positions, which are set to ’0’ and are not masked by the mask register. All inputs for points and the compare value itself are OR-combined with the value from the mask register. The compare operations point match or range detection are derived based on these OR-masked values.

The selection of the appropriate BADx register (point 2 or 0) for the mask value depends on EP2 and ER1. If at least one of both bits are enabled, the mask usage switches to point 0 due to the allocation of point 2. Otherwise the default mask stored in point 2 applies for CMP1.

BIT[2]: EM0 - Enable Mask for CMP0

0	Mask function for CMP0 is disabled (default)
1	Mask function for CMP0 is enabled

If EM0 is enabled the comparator CMP0 matches only these bit positions, which are set to ’0’ and are not masked by the mask register. All inputs for points and the compare value itself are OR-combined with the value from the mask register. The compare operations point match or range detection are derived based on these OR-masked values.

The selection of the appropriate BADx register (point 0 or 2) for the mask value depends on EP0 and ER0. If at least one of both bits are enabled, the mask usage switches to point 2 due to the allocation of point 0. Otherwise the default mask stored in point 0 applies for CMP0. If MPE=1 and COMB=1 the mask is taken from point 0, regardless of the setting of EP0 and ER0.

BIT[1]: ER1 - Enable Range for CMP1

0	Range detection CMP1 (channels 2-3) is disabled (default)
1	Range detection CMP1 (channels 2-3) is enabled

If ER1 is enabled then the registers BADx, point 3 and point 2 will be used for range comparison:

Point 2 <= Compare Value <= Point 3.

If a mask is set with EM1 then both point registers will be masked with the mask register content.

Point 3 and Point 2 are taken from BAD[x+3] and BAD[x+2], the mask is stored in Point 0, BAD[x+0].

The 'x' is the group offset and calculates by the group index multiplied with 4.

BIT[0]: ER0 - Enable Range for CMP0

0	Range detection CMP0 (channels 0-1) is disabled (default)
1	Range detection CMP0 (channels 0-1) is enabled

If ER0 is enabled then the registers BADx, point 1 and point 0 will be used for range comparison:

Point 0 <= Compare Value <= Point 1.

If a mask is set with EM0 then both point registers will be masked with the mask register content.

In the special case of MPE=1 together with COMB=1, Point 1 and Point 0 are taken from the opposite channel BAD[x+3] and BAD[x+2] and the mask is stored in Point 0, BAD[x+0]. Otherwise Point 1 and Point 0 are taken from BAD[x+1] and BAD[x+0], the mask is stored in Point 2, BAD[x+2].

The 'x' is the group offset and calculates by the group index multiplied with 4.

- Break Address/Data register (BAD0...BAD31)

The BADx registers define 32 break point addresses, data values or mask information for the 8 groups of channels. For each group of channels there are 4 dedicated BAD registers. BAD0, BAD1, BAD2 and BAD3 belong to Group 0, BAD4, BAD5, BAD6 and BAD7 belong to Group 1 and so on. The functionality described below for the registers of group 0 is representative for all the other groups too. The index of the BADx registers has to be incremented by 4 for each of the next group indexes.

BAD0 (BAD4, BAD8, ..., BAD28) [R/W]				
Address	**+0**	**+1**	**+2**	**+3**
$F080_H$	XXXXXXXX	XXXXXXXX	XXXXXXXX	XXXXXXXX

This register sets the 32 bit comparison value for break point 0 of CMP0. In range mode (set with ER0) the register value of BAD0 functions as lower address limit. In addition BAD0 could be used as mask register.

In the special case of MPE=1 and COMB=1 BAD0 is not used for the point definition. CMP0 gets its point configuration then from BAD2.

BAD1 (BAD5 BAD9, ..., BAD29) [R/W]				
Address	**+0**	**+1**	**+2**	**+3**
$F084_H$	XXXXXXXX	XXXXXXXX	XXXXXXXX	XXXXXXXX

This register sets the 32 bit comparison value for break point 1 of CMP0. In range mode (set with ER0) the register value of BAD1 functions as upper address limit.

In the special case of MPE=1 and COMB=1 BAD1 is not used for the point definition. CMP0 gets its point configuration then from BAD3.

BAD2 (BAD6, BAD10, ..., BAD30) [R/W]				
Address	**+0**	**+1**	**+2**	**+3**
$F088_H$	XXXXXXXX	XXXXXXXX	XXXXXXXX	XXXXXXXX

This register sets the 32 bit comparison value for break point 2 of CMP1. In range mode (set with ER1) the register value of BAD2 functions as lower address limit. In addition BAD2 could be used as mask register.

BAD3 (BAD7, BAD11, ..., BAD31) [R/W]				
Address	**+0**	**+1**	**+2**	**+3**
$F08C_H$	XXXXXXXX	XXXXXXXX	XXXXXXXX	XXXXXXXX

This register sets the 32 bit comparison value for break point 3 of CMP1. In range mode (set with ER1) the register value of BAD3 functions as upper address limit.

5. Quick Reference

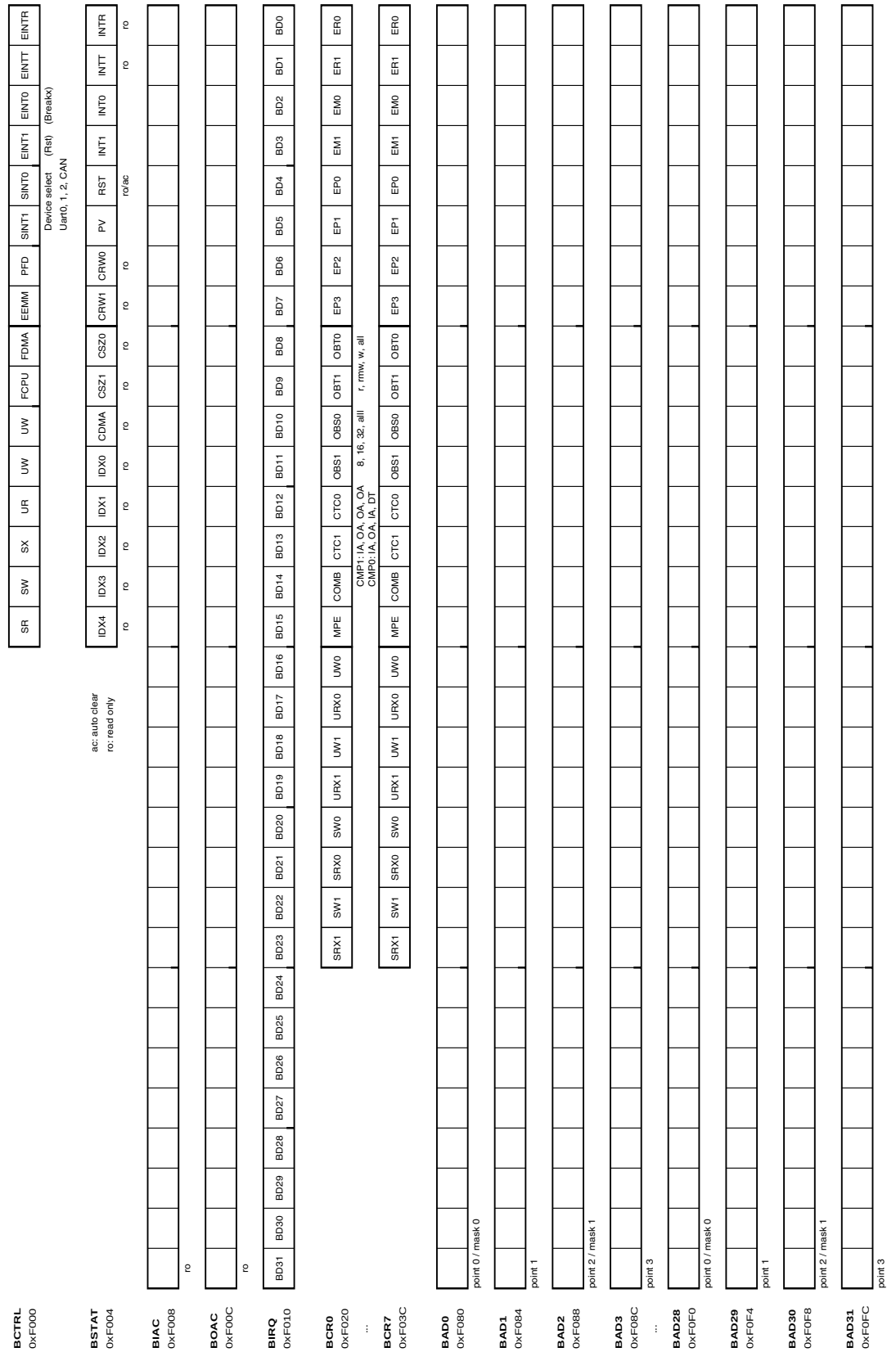


Figure 5-1 Register Quick Reference

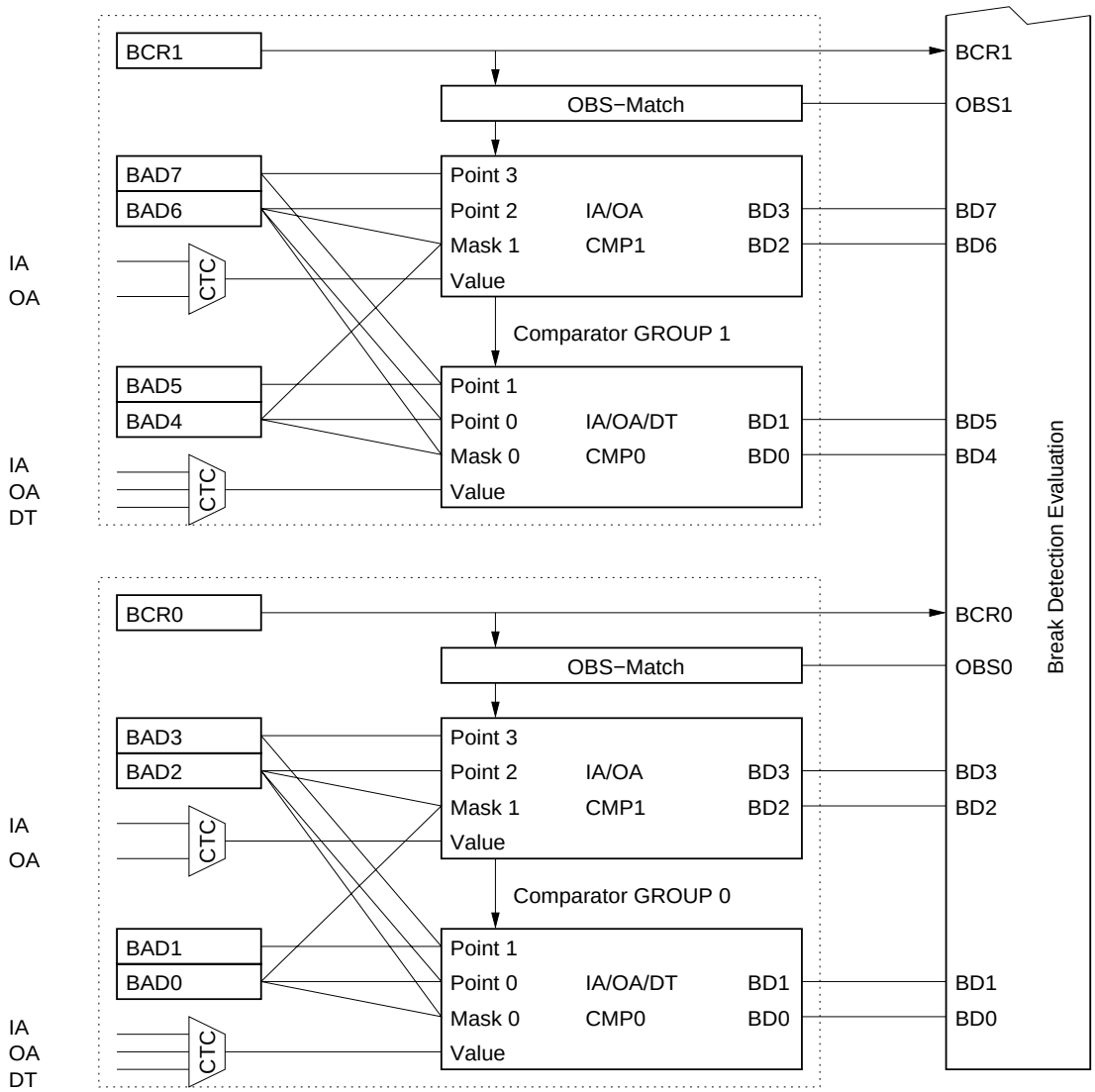


Figure 5-2 Comparator Group Structure (drawn for two groups)

Chapter 30 I/O Ports

1. I/O Ports Functions

For enabling the resource functions, please refer to section .....

Pin Name	I/O Signal	Circuit Type	Function
INITX	INITX	TC02_0	Reset input, low active
RSTX	RSTX	TC01_0	Reset input, low active
HSTX	HSTX	TC00_0	Hardware Standby input, low active
NMIX	NMIX	TC01_0	Non-maskable Interrupt input, low active
MD2	MD2	TC02_1	Mode Pin 2
MD1	MD1	TC02_1	Mode Pin 1
MD0	MD0	TC02_1	Mode Pin 0
X0	X0	TO00_0	Main Oscillation input
X1	X1	TO00_1	Main Oscillation output
X0A	X0A	TO01_0	Sub Oscillation input
X1A	X1A	TO01_1	Sub Oscillation output
ALARM0	ALARM0	TA02_0	ALARM comparator input channel 0
ALARM1	ALARM1	TA02_0	ALARM comparator input channel 1
EBREAKX	EBREAKX	TC01_0	EDSU BREAK input, low active
MONCLK	MONCLK	TC10_0	Clock Monitor Output
Port 00			
P00_7	P00_7	TP04_0	General purpose I/O. This function is enabled in the single-chip mode or by setting the corresponding PFR to ‘0’.
	D31		I/O pin for bit 31 of the external data bus. This function is enabled when the external bus is enabled.
P00_6	P00_6	TP04_0	General purpose I/O. This function is enabled in the single-chip mode or by setting the corresponding PFR to ‘0’.
	D30		I/O pin for bit 30 of the external data bus. This function is enabled when the external bus is enabled.
P00_5	P00_5	TP04_0	General purpose I/O. This function is enabled in the single-chip mode or by setting the corresponding PFR to ‘0’.
	D29		I/O pin for bit 29 of the external data bus. This function is enabled when the external bus is enabled.
P00_4	P00_4	TP04_0	General purpose I/O. This function is enabled in the single-chip mode or by setting the corresponding PFR to ‘0’.
	D28		I/O pin for bit 28 of the external data bus. This function is enabled when the external bus is enabled.

Pin Name	I/O Signal	Circuit Type	Function
P00_3	P00_3	TP04_0	General purpose I/O. This function is enabled in the single-chip mode or by setting the corresponding PFR to ‘0’.
	D27		I/O pin for bit 27 of the external data bus. This function is enabled when the external bus is enabled.
P00_2	P00_2	TP04_0	General purpose I/O. This function is enabled in the single-chip mode or by setting the corresponding PFR to ‘0’.
	D26		I/O pin for bit 26 of the external data bus. This function is enabled when the external bus is enabled.
P00_1	P00_1	TP04_0	General purpose I/O. This function is enabled in the single-chip mode or by setting the corresponding PFR to ‘0’.
	D25		I/O pin for bit 25 of the external data bus. This function is enabled when the external bus is enabled.
P00_0	P00_0	TP04_0	General purpose I/O. This function is enabled in the single-chip mode or by setting the corresponding PFR to ‘0’.
	D24		I/O pin for bit 24 of the external data bus. This function is enabled when the external bus is enabled.
Port 01			
P01_7	P01_7	TP04_0	General purpose I/O. This function is enabled in the single-chip mode or by setting the corresponding PFR to ‘0’.
	D23		I/O pin for bit 23 of the external data bus. This function is enabled when the external bus is enabled.
P01_6	P01_6	TP04_0	General purpose I/O. This function is enabled in the single-chip mode or by setting the corresponding PFR to ‘0’.
	D22		I/O pin for bit 22 of the external data bus. This function is enabled when the external bus is enabled.
P01_5	P01_5	TP04_0	General purpose I/O. This function is enabled in the single-chip mode or by setting the corresponding PFR to ‘0’.
	D21		I/O pin for bit 21 of the external data bus. This function is enabled when the external bus is enabled.
P01_4	P01_4	TP04_0	General purpose I/O. This function is enabled in the single-chip mode or by setting the corresponding PFR to ‘0’.
	D20		I/O pin for bit 20 of the external data bus. This function is enabled when the external bus is enabled.
P01_3	P01_3	TP04_0	General purpose I/O. This function is enabled in the single-chip mode or by setting the corresponding PFR to ‘0’.
	D19		I/O pin for bit 19 of the external data bus. This function is enabled when the external bus is enabled.
P01_2	P01_2	TP04_0	General purpose I/O. This function is enabled in the single-chip mode or by setting the corresponding PFR to ‘0’.
	D18		I/O pin for bit 18 of the external data bus. This function is enabled when the external bus is enabled.
P01_1	P01_1	TP04_0	General purpose I/O. This function is enabled in the single-chip mode or by setting the corresponding PFR to ‘0’.
	D17		I/O pin for bit 17 of the external data bus. This function is enabled when the external bus is enabled.

Pin Name	I/O Signal	Circuit Type	Function
P01_0	P01_0	TP04_0	General purpose I/O. This function is enabled in the single-chip mode or by setting the corresponding PFR to ‘0’.
	D16		I/O pin for bit 16 of the external data bus. This function is enabled when the external bus is enabled.
Port 02			
P02_7	P02_7	TP04_0	General purpose I/O. This function is enabled in the single-chip mode or by setting the corresponding PFR to ‘0’.
	D15		I/O pin for bit 15 of the external data bus. This function is enabled when the external bus is enabled.
P02_6	P02_6	TP04_0	General purpose I/O. This function is enabled in the single-chip mode or by setting the corresponding PFR to ‘0’.
	D14		I/O pin for bit 14 of the external data bus. This function is enabled when the external bus is enabled.
P02_5	P02_5	TP04_0	General purpose I/O. This function is enabled in the single-chip mode or by setting the corresponding PFR to ‘0’.
	D13		I/O pin for bit 13 of the external data bus. This function is enabled when the external bus is enabled.
P02_4	P02_4	TP04_0	General purpose I/O. This function is enabled in the single-chip mode or by setting the corresponding PFR to ‘0’.
	D12		I/O pin for bit 12 of the external data bus. This function is enabled when the external bus is enabled.
P02_3	P02_3	TP04_0	General purpose I/O. This function is enabled in the single-chip mode or by setting the corresponding PFR to ‘0’.
	D11		I/O pin for bit 11 of the external data bus. This function is enabled when the external bus is enabled.
P02_2	P02_2	TP04_0	General purpose I/O. This function is enabled in the single-chip mode or by setting the corresponding PFR to ‘0’.
	D10		I/O pin for bit 10 of the external data bus. This function is enabled when the external bus is enabled.
P02_1	P02_1	TP04_0	General purpose I/O. This function is enabled in the single-chip mode or by setting the corresponding PFR to ‘0’.
	D9		I/O pin for bit 9 of the external data bus. This function is enabled when the external bus is enabled.
P02_0	P02_0	TP04_0	General purpose I/O. This function is enabled in the single-chip mode or by setting the corresponding PFR to ‘0’.
	D8		I/O pin for bit 8 of the external data bus. This function is enabled when the external bus is enabled.
Port 03			
P03_7	P03_7	TP04_0	General purpose I/O. This function is enabled in the single-chip mode or by setting the corresponding PFR to ‘0’.
	D7		I/O pin for bit 7 of the external data bus. This function is enabled when the external bus is enabled.
P03_6	P03_6	TP04_0	General purpose I/O. This function is enabled in the single-chip mode or by setting the corresponding PFR to ‘0’.
	D6		I/O pin for bit 6 of the external data bus. This function is enabled when the external bus is enabled.

Pin Name	I/O Signal	Circuit Type	Function
P03_5	P03_5	TP04_0	General purpose I/O. This function is enabled in the single-chip mode or by setting the corresponding PFR to ‘0’.
	D5		I/O pin for bit 5 of the external data bus. This function is enabled when the external bus is enabled.
P03_4	P03_4	TP04_0	General purpose I/O. This function is enabled in the single-chip mode or by setting the corresponding PFR to ‘0’.
	D4		I/O pin for bit 4 of the external data bus. This function is enabled when the external bus is enabled.
P03_3	P03_3	TP04_0	General purpose I/O. This function is enabled in the single-chip mode or by setting the corresponding PFR to ‘0’.
	D3		I/O pin for bit 3 of the external data bus. This function is enabled when the external bus is enabled.
P03_2	P03_2	TP04_0	General purpose I/O. This function is enabled in the single-chip mode or by setting the corresponding PFR to ‘0’.
	D2		I/O pin for bit 2 of the external data bus. This function is enabled when the external bus is enabled.
P03_1	P03_1	TP04_0	General purpose I/O. This function is enabled in the single-chip mode or by setting the corresponding PFR to ‘0’.
	D1		I/O pin for bit 1 of the external data bus. This function is enabled when the external bus is enabled.
P03_0	P03_0	TP04_0	General purpose I/O. This function is enabled in the single-chip mode or by setting the corresponding PFR to ‘0’.
	D0		I/O pin for bit 0 of the external data bus. This function is enabled when the external bus is enabled.
Port 04			
P04_7	P04_7	TP04_0	General purpose I/O. This function is enabled in the single-chip mode or by setting the corresponding PFR to ‘0’.
	A31		I/O pin for bit 31 of the external address bus. This function is enabled when the external bus is enabled.
P04_6	P04_6	TP04_0	General purpose I/O. This function is enabled in the single-chip mode or by setting the corresponding PFR to ‘0’.
	A30		I/O pin for bit 30 of the external address bus. This function is enabled when the external bus is enabled.
P04_5	P04_5	TP04_0	General purpose I/O. This function is enabled in the single-chip mode or by setting the corresponding PFR to ‘0’.
	A29		I/O pin for bit 29 of the external address bus. This function is enabled when the external bus is enabled.
P04_4	P04_4	TP04_0	General purpose I/O. This function is enabled in the single-chip mode or by setting the corresponding PFR to ‘0’.
	A28		I/O pin for bit 28 of the external address bus. This function is enabled when the external bus is enabled.
P04_3	P04_3	TP04_0	General purpose I/O. This function is enabled in the single-chip mode or by setting the corresponding PFR to ‘0’.
	A27		I/O pin for bit 27 of the external address bus. This function is enabled when the external bus is enabled.

Pin Name	I/O Signal	Circuit Type	Function
P04_2	P04_2	TP04_0	General purpose I/O. This function is enabled in the single-chip mode or by setting the corresponding PFR to ‘0’.
	A26		I/O pin for bit 26 of the external address bus. This function is enabled when the external bus is enabled.
P04_1	P04_1	TP04_0	General purpose I/O. This function is enabled in the single-chip mode or by setting the corresponding PFR to ‘0’.
	A25		I/O pin for bit 25 of the external address bus. This function is enabled when the external bus is enabled.
P04_0	P04_0	TP04_0	General purpose I/O. This function is enabled in the single-chip mode or by setting the corresponding PFR to ‘0’.
	A24		I/O pin for bit 24 of the external address bus. This function is enabled when the external bus is enabled.
Port 05			
P05_7	P05_7	TP04_0	General purpose I/O. This function is enabled in the single-chip mode or by setting the corresponding PFR to ‘0’.
	A23		I/O pin for bit 23 of the external address bus. This function is enabled when the external bus is enabled.
P05_6	P05_6	TP04_0	General purpose I/O. This function is enabled in the single-chip mode or by setting the corresponding PFR to ‘0’.
	A22		I/O pin for bit 22 of the external address bus. This function is enabled when the external bus is enabled.
P05_5	P05_5	TP04_0	General purpose I/O. This function is enabled in the single-chip mode or by setting the corresponding PFR to ‘0’.
	A21		I/O pin for bit 21 of the external address bus. This function is enabled when the external bus is enabled.
P05_4	P05_4	TP04_0	General purpose I/O. This function is enabled in the single-chip mode or by setting the corresponding PFR to ‘0’.
	A20		I/O pin for bit 20 of the external address bus. This function is enabled when the external bus is enabled.
P05_3	P05_3	TP04_0	General purpose I/O. This function is enabled in the single-chip mode or by setting the corresponding PFR to ‘0’.
	A19		I/O pin for bit 19 of the external address bus. This function is enabled when the external bus is enabled.
P05_2	P05_2	TP04_0	General purpose I/O. This function is enabled in the single-chip mode or by setting the corresponding PFR to ‘0’.
	A18		I/O pin for bit 18 of the external address bus. This function is enabled when the external bus is enabled.
P05_1	P05_1	TP04_0	General purpose I/O. This function is enabled in the single-chip mode or by setting the corresponding PFR to ‘0’.
	A17		I/O pin for bit 17 of the external address bus. This function is enabled when the external bus is enabled.
P05_0	P05_0	TP04_0	General purpose I/O. This function is enabled in the single-chip mode or by setting the corresponding PFR to ‘0’.
	A16		I/O pin for bit 16 of the external address bus. This function is enabled when the external bus is enabled.
Port 06			

Pin Name	I/O Signal	Circuit Type	Function
P06_7	P06_7	TP04_0	General purpose I/O. This function is enabled in the single-chip mode or by setting the corresponding PFR to ‘0’.
	A15		I/O pin for bit 15 of the external address bus. This function is enabled when the external bus is enabled.
P06_6	P06_6	TP04_0	General purpose I/O. This function is enabled in the single-chip mode or by setting the corresponding PFR to ‘0’.
	A14		I/O pin for bit 14 of the external address bus. This function is enabled when the external bus is enabled.
P06_5	P06_5	TP04_0	General purpose I/O. This function is enabled in the single-chip mode or by setting the corresponding PFR to ‘0’.
	A13		I/O pin for bit 13 of the external address bus. This function is enabled when the external bus is enabled.
P06_4	P06_4	TP04_0	General purpose I/O. This function is enabled in the single-chip mode or by setting the corresponding PFR to ‘0’.
	A12		I/O pin for bit 12 of the external address bus. This function is enabled when the external bus is enabled.
P06_3	P06_3	TP04_0	General purpose I/O. This function is enabled in the single-chip mode or by setting the corresponding PFR to ‘0’.
	A11		I/O pin for bit 11 of the external address bus. This function is enabled when the external bus is enabled.
P06_2	P06_2	TP04_0	General purpose I/O. This function is enabled in the single-chip mode or by setting the corresponding PFR to ‘0’.
	A10		I/O pin for bit 10 of the external address bus. This function is enabled when the external bus is enabled.
P06_1	P06_1	TP04_0	General purpose I/O. This function is enabled in the single-chip mode or by setting the corresponding PFR to ‘0’.
	A9		I/O pin for bit 9 of the external address bus. This function is enabled when the external bus is enabled.
P06_0	P06_0	TP04_0	General purpose I/O. This function is enabled in the single-chip mode or by setting the corresponding PFR to ‘0’.
	A8		I/O pin for bit 8 of the external address bus. This function is enabled when the external bus is enabled.
Port 07			
P07_7	P07_7	TP04_0	General purpose I/O. This function is enabled in the single-chip mode or by setting the corresponding PFR to ‘0’.
	A7		I/O pin for bit 7 of the external address bus. This function is enabled when the external bus is enabled.
P07_6	P07_6	TP04_0	General purpose I/O. This function is enabled in the single-chip mode or by setting the corresponding PFR to ‘0’.
	A6		I/O pin for bit 6 of the external address bus. This function is enabled when the external bus is enabled.
P07_5	P07_5	TP04_0	General purpose I/O. This function is enabled in the single-chip mode or by setting the corresponding PFR to ‘0’.
	A5		I/O pin for bit 5 of the external address bus. This function is enabled when the external bus is enabled.

Pin Name	I/O Signal	Circuit Type	Function
P07_4	P07_4	TP04_0	General purpose I/O. This function is enabled in the single-chip mode or by setting the corresponding PFR to ‘0’.
	A4		I/O pin for bit 4 of the external address bus. This function is enabled when the external bus is enabled.
P07_3	P07_3	TP04_0	General purpose I/O. This function is enabled in the single-chip mode or by setting the corresponding PFR to ‘0’.
	A3		I/O pin for bit 3 of the external address bus. This function is enabled when the external bus is enabled.
P07_2	P07_2	TP04_0	General purpose I/O. This function is enabled in the single-chip mode or by setting the corresponding PFR to ‘0’.
	A2		I/O pin for bit 2 of the external address bus. This function is enabled when the external bus is enabled.
P07_1	P07_1	TP04_0	General purpose I/O. This function is enabled in the single-chip mode or by setting the corresponding PFR to ‘0’.
	A1		I/O pin for bit 1 of the external address bus. This function is enabled when the external bus is enabled.
P07_0	P07_0	TP04_0	General purpose I/O. This function is enabled in the single-chip mode or by setting the corresponding PFR to ‘0’.
	A0		I/O pin for bit 0 of the external address bus. This function is enabled when the external bus is enabled.
Port 08			
P08_7	P08_7	TP04_0	General purpose I/O. This function is enabled in the single-chip mode or by setting the corresponding PFR to ‘0’.
	RDY		Input pin for external wait (if RDY enabled for the corresponding CS area). This function is enabled when the external bus is enabled.
P08_6	P08_6	TP04_0	General purpose I/O. This function is enabled in the single-chip mode or by setting the corresponding PFR to ‘0’.
	BRQ		Input pin for external bus request (if sharing enabled for the corresponding CS area). This function is enabled when the external bus is enabled.
P08_5	P08_5	TP04_0	General purpose I/O. This function is enabled in the single-chip mode or by setting the corresponding PFR to ‘0’.
	BGRNTX		Output pin for external bus granted (if sharing enabled for the corresponding CS area). This function is enabled when the external bus is enabled.
P08_4	P08_4	TP04_0	General purpose I/O. This function is enabled in the single-chip mode or by setting the corresponding PFR to ‘0’.
	RDX		Output pin for external bus read strobe. This function is enabled when the external bus is enabled.
P08_3	P08_3	TP04_0	General purpose I/O. This function is enabled in the single-chip mode or by setting the corresponding PFR to ‘0’.
	WRX3		Output pin for external bus write strobe. This function is enabled when the external bus is enabled.
P08_2	P08_2	TP04_0	General purpose I/O. This function is enabled in the single-chip mode or by setting the corresponding PFR to ‘0’.
	WRX2		Output pin for external bus write strobe. This function is enabled when the external bus is enabled.

Pin Name	I/O Signal	Circuit Type	Function
P08_1	P08_1	TP04_0	General purpose I/O. This function is enabled in the single-chip mode or by setting the corresponding PFR to ‘0’.
	WRX1		Output pin for external bus write strobe. This function is enabled when the external bus is enabled.
P08_0	P08_0	TP04_0	General purpose I/O. This function is enabled in the single-chip mode or by setting the corresponding PFR to ‘0’.
	WRX0		Output pin for external bus write strobe. This function is enabled when the external bus is enabled.
Port 09			
P09_7	P09_7	TP04_0	General purpose I/O. This function is enabled in the single-chip mode or by setting the corresponding PFR to ‘0’.
	CSX7		Output pin for external bus chip select area 7. This function is enabled when the external bus is enabled.
P09_6	P09_6	TP04_0	General purpose I/O. This function is enabled in the single-chip mode or by setting the corresponding PFR to ‘0’.
	CSX6		Output pin for external bus chip select area 6. This function is enabled when the external bus is enabled.
P09_5	P09_5	TP04_0	General purpose I/O. This function is enabled in the single-chip mode or by setting the corresponding PFR to ‘0’.
	CSX5		Output pin for external bus chip select area 5. This function is enabled when the external bus is enabled.
P09_4	P09_4	TP04_0	General purpose I/O. This function is enabled in the single-chip mode or by setting the corresponding PFR to ‘0’.
	CSX4		Output pin for external bus chip select area 4. This function is enabled when the external bus is enabled.
P09_3	P09_3	TP04_0	General purpose I/O. This function is enabled in the single-chip mode or by setting the corresponding PFR to ‘0’.
	CSX3		Output pin for external bus chip select area 3. This function is enabled when the external bus is enabled.
P09_2	P09_2	TP04_0	General purpose I/O. This function is enabled in the single-chip mode or by setting the corresponding PFR to ‘0’.
	CSX2		Output pin for external bus chip select area 2. This function is enabled when the external bus is enabled.
P09_1	P09_1	TP04_0	General purpose I/O. This function is enabled in the single-chip mode or by setting the corresponding PFR to ‘0’.
	CSX1		Output pin for external bus chip select area 1. This function is enabled when the external bus is enabled.
P09_0	P09_0	TP04_0	General purpose I/O. This function is enabled in the single-chip mode or by setting the corresponding PFR to ‘0’.
	CSX0		Output pin for external bus chip select area 0. This function is enabled when the external bus is enabled.
Port 10			
P10_7	P10_7	TP04_0	General purpose I/O. This function is enabled in the single-chip mode or by setting the corresponding PFR to ‘0’.

Pin Name	I/O Signal	Circuit Type	Function
P10_6	P10_6	TP04_0	General purpose I/O. This function is enabled in the single-chip mode or by setting the corresponding PFR to ‘0’.
	MCLKE		Output pin for external bus memory clock enable. This function is enabled when the external bus is enabled.
P10_5	P10_5	TP04_0	General purpose I/O. This function is enabled in the single-chip mode or by setting the corresponding PFR to ‘0’.
	MCLKI		Input pin for external bus memory clock. This function is enabled when the external bus is enabled.
	$\overline{\text{MCLKI}}$		Input pin for external bus memory clock (inverted input). This function is enabled when the external bus is enabled.
P10_4	P10_4	TP04_0	General purpose I/O. This function is enabled in the single-chip mode or by setting the corresponding PFR to ‘0’.
	MCLKO		Output pin for external bus memory clock. This function is enabled when the external bus is enabled.
	$\overline{\text{MCLKO}}$		Output pin for external bus memory clock (inverted output). This function is enabled when the external bus is enabled.
P10_3	P10_3	TP04_0	General purpose I/O. This function is enabled in the single-chip mode or by setting the corresponding PFR to ‘0’.
	WEX		Output pin for external bus write strobe. This function is enabled when the external bus is enabled.
P10_2	P10_2	TP04_0	General purpose I/O. This function is enabled in the single-chip mode or by setting the corresponding PFR to ‘0’.
	BAAX		Output pin for external bus burst access. This function is enabled when the external bus is enabled.
P10_1	P10_1	TP04_0	General purpose I/O. This function is enabled in the single-chip mode or by setting the corresponding PFR to ‘0’.
	ASX		Output pin for external bus address strobe. This function is enabled when the external bus is enabled.
P10_0	P10_0	TP04_0	General purpose I/O. This function is enabled in the single-chip mode or by setting the corresponding PFR to ‘0’.
	SYSCLK		Output pin for external bus clock. This function is enabled when the external bus is enabled.
	$\overline{\text{SYSCLK}}$		Output pin for external bus clock (inverted output). This function is enabled when the external bus is enabled.
Port 11			
P11_7	P11_7	TP04_0	General purpose I/O.
P11_6	P11_6	TP04_0	General purpose I/O.
P11_5	P11_5	TP04_0	General purpose I/O.
P11_4	P11_4	TP04_0	General purpose I/O.
P11_3	P11_3	TP04_0	General purpose I/O.
P11_2	P11_2	TP04_0	General purpose I/O.
P11_1	P11_1	TP04_0	General purpose I/O.
	IOWRX		Output pin for DMA memory to I/O fly-by transfer.

Pin Name	I/O Signal	Circuit Type	Function
P11_0	P11_0	TP04_0	General purpose I/O.
	IORDX		Output pin for DMA I/O to memory fly-by transfer.
Port 12			
P12_7	P12_7	TP04_0	General purpose I/O.
	DEOP3		Output pin for DMA end of transfer.
P12_6	P12_6	TP04_0	General purpose I/O.
	DEOTX3		Input pin for DMA transfer stop request.
	DEOP3		Output pin for DMA end of transfer.
P12_5	P12_5	TP04_0	General purpose I/O.
	DACKX3		Output pin for DMA transfer request acknowledgement.
P12_4	P12_4	TP04_0	General purpose I/O.
	DREQ3		Input pin for DMA transfer request.
P12_3	P12_3	TP04_0	General purpose I/O.
	DEOP2		Output pin for DMA end of transfer.
P12_2	P12_2	TP04_0	General purpose I/O.
	DEOTX2		Input pin for DMA transfer stop request.
	DEOP2		Output pin for DMA end of transfer.
P12_1	P12_1	TP04_0	General purpose I/O.
	DACKX2		Output pin for DMA transfer request acknowledgement.
P12_0	P12_0	TP04_0	General purpose I/O.
	DREQ2		Input pin for DMA transfer request.
Port 13			
P13_7	P13_7	TP04_0	General purpose I/O.
	DEOP1		Output pin for DMA end of transfer.
P13_6	P13_6	TP04_0	General purpose I/O.
	DEOTX1		Input pin for DMA transfer stop request.
	DEOP1		Output pin for DMA end of transfer.
P13_5	P13_5	TP04_0	General purpose I/O.
	DACKX1		Output pin for DMA transfer request acknowledgement.
P13_4	P13_4	TP04_0	General purpose I/O.
	DREQ1		Input pin for DMA transfer request.
P13_3	P13_3	TP04_0	General purpose I/O.
	DEOP0		Output pin for DMA end of transfer.
P13_2	P13_2	TP04_0	General purpose I/O.
	DEOTX0		Input pin for DMA transfer stop request.
	DEOP0		Output pin for DMA end of transfer.
P13_1	P13_1	TP04_0	General purpose I/O.
	DACKX0		Output pin for DMA transfer request acknowledgement.
P13_0	P13_0	TP04_0	General purpose I/O.
	DREQ0		Input pin for DMA transfer request.
Port 14			

Pin Name	I/O Signal	Circuit Type	Function
P14_7	P14_7	TP00_0	General purpose I/O.
	ICU7		Data sample input pin for input capture ICU 7.
	TIN7		Event input pin for the reload timer RLT 7.
	TTG15/7		Event input pin for the programmable pulse generators PPG 15 and PPG 7.
P14_6	P14_6	TP00_0	General purpose I/O.
	ICU6		Data sample input pin for input capture ICU 6.
	TIN6		Event input pin for the reload timer RLT 6.
	TTG14/6		Event input pin for the programmable pulse generators PPG 14 and PPG 6.
P14_5	P14_5	TP00_0	General purpose I/O.
	ICU5		Data sample input pin for input capture ICU 5.
	TIN5		Event input pin for the reload timer RLT 5.
	TTG13/5		Event input pin for the programmable pulse generators PPG 13 and PPG 5.
P14_4	P14_4	TP00_0	General purpose I/O.
	ICU4		Data sample input pin for input capture ICU 4.
	TIN4		Event input pin for the reload timer RLT 4.
	TTG12/4		Event input pin for the programmable pulse generators PPG 12 and PPG 4.
P14_3	P14_3	TP00_0	General purpose I/O.
	ICU3		Data sample input pin for input capture ICU 3.
	TIN3		Event input pin for the reload timer RLT 3.
	TTG11/3		Event input pin for the programmable pulse generators PPG 11 and PPG 3.
P14_2	P14_2	TP00_0	General purpose I/O.
	ICU2		Data sample input pin for input capture ICU 2.
	TIN2		Event input pin for the reload timer RLT 2.
	TTG10/2		Event input pin for the programmable pulse generators PPG 10 and PPG 2.
P14_1	P14_1	TP00_0	General purpose I/O.
	ICU1		Data sample input pin for input capture ICU 1.
	TIN1		Event input pin for the reload timer RLT 1.
	TTG9/1		Event input pin for the programmable pulse generators PPG 9 and PPG 1.
P14_0	P14_0	TP00_0	General purpose I/O.
	ICU0		Data sample input pin for input capture ICU 0.
	TIN0		Event input pin for the reload timer RLT 0.
	TTG8/0		Event input pin for the programmable pulse generators PPG 8 and PPG 0.
Port 15			
P15_7	P15_7	TP00_0	General purpose I/O.
	OCU7		Waveform output pin for output compare OCU 7.
	TOT7		Output pin for the reload timer RLT 7.
P15_6	P15_6	TP00_0	General purpose I/O.
	OCU6		Waveform output pin for output compare OCU 6.
	TOT6		Output pin for the reload timer RLT 6.

Pin Name	I/O Signal	Circuit Type	Function
P15_5	P15_5	TP00_0	General purpose I/O.
	OCU5		Waveform output pin for output compare OCU 5.
	TOT5		Output pin for the reload timer RLT 5.
P15_4	P15_4	TP00_0	General purpose I/O.
	OCU4		Waveform output pin for output compare OCU 4.
	TOT4		Output pin for the reload timer RLT 4.
P15_3	P15_3	TP00_0	General purpose I/O.
	OCU3		Waveform output pin for output compare OCU 3.
	TOT3		Output pin for the reload timer RLT 3.
P15_2	P15_2	TP00_0	General purpose I/O.
	OCU2		Waveform output pin for output compare OCU 2.
	TOT2		Output pin for the reload timer RLT 2.
P15_1	P15_1	TP00_0	General purpose I/O.
	OCU1		Waveform output pin for output compare OCU 1.
	TOT1		Output pin for the reload timer RLT 1.
P15_0	P15_0	TP00_0	General purpose I/O.
	OCU0		Waveform output pin for output compare OCU 0.
	TOT0		Output pin for the reload timer RLT 0.
Port 16			
P16_7	P16_7	TP00_0	General purpose I/O.
	PPG15		Waveform output pin for programmable pulse generator PPG 15.
	ATGX		A/D converter external trigger input.
P16_6	P15_6	TP00_0	General purpose I/O.
	PPG14		Waveform output pin for programmable pulse generator PPG 14.
	PFM		Waveform output pin for pulse frequency modulator PFM.
P16_5	P15_5	TP00_0	General purpose I/O.
	PPG13		Waveform output pin for programmable pulse generator PPG 13.
	SGO		Waveform output pin for sound generator SG.
P16_4	P15_4	TP00_0	General purpose I/O.
	PPG12		Waveform output pin for programmable pulse generator PPG 12.
	SGA		Amplitude output pin for sound generator SG.
P16_3	P15_3	TP00_0	General purpose I/O.
	PPG11		Waveform output pin for programmable pulse generator PPG 11.
P16_2	P15_2	TP00_0	General purpose I/O.
	PPG10		Waveform output pin for programmable pulse generator PPG 10.
P16_1	P15_1	TP00_0	General purpose I/O.
	PPG9		Waveform output pin for programmable pulse generator PPG 9.
P16_0	P15_0	TP00_0	General purpose I/O.
	PPG8		Waveform output pin for programmable pulse generator PPG 8.
Port 17			

Pin Name	I/O Signal	Circuit Type	Function
P17_7	P17_7	TP00_0	General purpose I/O.
	PPG7		Waveform output pin for programmable pulse generator PPG 7.
P17_6	P17_6	TP00_0	General purpose I/O.
	PPG6		Waveform output pin for programmable pulse generator PPG 6.
P17_5	P17_5	TP00_0	General purpose I/O.
	PPG5		Waveform output pin for programmable pulse generator PPG 5.
P17_4	P17_4	TP00_0	General purpose I/O.
	PPG4		Waveform output pin for programmable pulse generator PPG 4.
P17_3	P17_3	TP00_0	General purpose I/O.
	PPG3		Waveform output pin for programmable pulse generator PPG 3.
P17_2	P17_2	TP00_0	General purpose I/O.
	PPG2		Waveform output pin for programmable pulse generator PPG 2.
P17_1	P17_1	TP00_0	General purpose I/O.
	PPG1		Waveform output pin for programmable pulse generator PPG 1.
P17_0	P17_0	TP00_0	General purpose I/O.
	PPG0		Waveform output pin for programmable pulse generator PPG 0.
Port 18			
P18_7	P18_7	TP00_0	General purpose I/O.
P18_6	P18_6	TP00_0	General purpose I/O.
	SCK7		Clock I/O pin for LIN-USART 7.
	ZIN3		8-bit reset input pin of the Up-/Down Counter UDC 2/3.
	CK7		Input for the 16-bit I/O Timer FRT 7.
P18_5	P18_5	TP00_0	General purpose I/O.
	SOT7		Serial data output pin for LIN-USART 7.
	BIN3		8-bit down-count input pin of the Up-/Down Counter UDC 2/3.
P18_4	P18_4	TP00_0	General purpose I/O.
	SIN7		Serial data input pin for LIN-USART 7.
	AIN3		8-bit up-count input pin of the Up-/Down Counter UDC 2/3.
P18_3	P18_3	TP00_0	General purpose I/O.
P18_2	P18_2	TP00_0	General purpose I/O.
	SCK6		Clock I/O pin for LIN-USART 6.
	ZIN2		8/16-bit reset input pin of the Up-/Down Counter UDC 2/3.
	CK6		Input for the 16-bit I/O Timer FRT 6.
P18_1	P18_1	TP00_0	General purpose I/O.
	SOT6		Serial data output pin for LIN-USART 6.
	BIN2		8/16-bit down-count input pin of the Up-/Down Counter UDC 2/3.
P18_0	P18_0	TP00_0	General purpose I/O.
	SIN6		Serial data input pin for LIN-USART 6.
	AIN2		8/16-bit up-count input pin of the Up-/Down Counter UDC 2/3.
Port 19			

Pin Name	I/O Signal	Circuit Type	Function
P19_7	P19_7	TP00_0	General purpose I/O.
P19_6	P19_6	TP00_0	General purpose I/O.
	SCK5		Clock I/O pin for LIN-USART 5.
	CK5		Input for the 16-bit I/O Timer FRT 5.
P19_5	P19_5	TP00_0	General purpose I/O.
	SOT5		Serial data output pin for LIN-USART 5.
P19_4	P19_4	TP00_0	General purpose I/O.
	SIN5		Serial data input pin for LIN-USART 5.
P19_3	P19_3	TP00_0	General purpose I/O.
P19_2	P19_2	TP00_0	General purpose I/O.
	SCK4		Clock I/O pin for LIN-USART 4.
	CK4		Input for the 16-bit I/O Timer FRT 4.
P19_1	P19_1	TP00_0	General purpose I/O.
	SOT4		Serial data output pin for LIN-USART 4.
P19_0	P19_0	TP00_0	General purpose I/O.
	SIN4		Serial data input pin for LIN-USART 4.
Port 20			
P20_7	P20_7	TP00_0	General purpose I/O.
P20_6	P20_6	TP00_0	General purpose I/O.
	SCK3		Clock I/O pin for LIN-USART 3.
	ZIN1		8-bit reset input pin of the Up-/Down Counter UDC 0/1.
	CK3		Input for the 16-bit I/O Timer FRT 3.
P20_5	P20_5	TP00_0	General purpose I/O.
	SOT3		Serial data output pin for LIN-USART 3.
	BIN1		8-bit down-count input pin of the Up-/Down Counter UDC 0/1.
P20_4	P20_4	TP00_0	General purpose I/O.
	SIN3		Serial data input pin for LIN-USART 3.
	AIN1		8-bit up-count input pin of the Up-/Down Counter UDC 0/1.
P20_3	P20_3	TP00_0	General purpose I/O.
P20_2	P20_2	TP00_0	General purpose I/O.
	SCK2		Clock I/O pin for LIN-USART 2.
	ZIN0		8/16-bit reset input pin of the Up-/Down Counter UDC 0/1.
	CK2		Input for the 16-bit I/O Timer FRT 2.
P20_1	P20_1	TP00_0	General purpose I/O.
	SOT2		Serial data output pin for LIN-USART 2.
	BIN0		8/16-bit down-count input pin of the Up-/Down Counter UDC 0/1.
P20_0	P20_0	TP00_0	General purpose I/O.
	SIN2		Serial data input pin for LIN-USART 2.
	AIN0		8/16-bit up-count input pin of the Up-/Down Counter UDC 0/1.

Pin Name	I/O Signal	Circuit Type	Function
Port 21			
P21_7	P21_7	TP00_0	General purpose I/O.
P21_6	P21_6	TP00_0	General purpose I/O.
	SCK1		Clock I/O pin for LIN-USART 1.
	CK1		Input for the 16-bit I/O Timer FRT 1.
P21_5	P21_5	TP00_0	General purpose I/O.
	SOT1		Serial data output pin for LIN-USART 1.
P21_4	P21_4	TP00_0	General purpose I/O.
	SIN1		Serial data input pin for LIN-USART 1.
P21_3	P21_3	TP00_0	General purpose I/O.
P21_2	P21_2	TP00_0	General purpose I/O.
	SCK0		Clock I/O pin for LIN-USART 0.
	CK0		Input for the 16-bit I/O Timer FRT 0.
P21_1	P21_1	TP00_0	General purpose I/O.
	SOT0		Serial data output pin for LIN-USART 0.
P21_0	P21_0	TP00_0	General purpose I/O.
	SIN0		Serial data input pin for LIN-USART 0.
Port 22			
P22_7	P22_7	TP02_0	General purpose I/O.
	SCL1		Serial clock I/O pin for I2C 1.
P22_6	P22_6	TP02_0	General purpose I/O.
	SDA1		Serial data I/O pin for I2C 1.
	INT15		External interrupt request input pin for INT 15.
P22_5	P22_5	TP02_0	General purpose I/O.
	SCL0		Serial clock I/O pin for I2C 0.
P22_4	P22_4	TP02_0	General purpose I/O.
	SDA0		Serial data I/O pin for I2C 0.
	INT14		External interrupt request input pin for INT 14.
P22_3	P22_3	TP00_0	General purpose I/O.
	TX5		Transmission output pin for CAN 5.
P22_2	P22_2	TP00_0	General purpose I/O.
	RX5		Reception input pin for CAN 5.
	INT13		External interrupt request input pin for INT 13.
P22_1	P22_1	TP00_0	General purpose I/O.
	TX4		Transmission output pin for CAN 4.
P22_0	P22_0	TP00_0	General purpose I/O.
	RX4		Reception input pin for CAN 4.
	INT12		External interrupt request input pin for INT 12.
Port 23			

Pin Name	I/O Signal	Circuit Type	Function
P23_7	P23_7	TP00_0	General purpose I/O.
	TX3		Transmission output pin for CAN 3.
P23_6	P23_6	TP00_0	General purpose I/O.
	RX3		Reception input pin for CAN 3.
	INT11		External interrupt request input pin for INT 11.
P23_5	P23_5	TP00_0	General purpose I/O.
	TX2		Transmission output pin for CAN 2.
P23_4	P23_4	TP00_0	General purpose I/O.
	RX2		Reception input pin for CAN 2.
	INT10		External interrupt request input pin for INT 10.
P23_3	P23_3	TP00_0	General purpose I/O.
	TX1		Transmission output pin for CAN 1.
P23_2	P23_2	TP00_0	General purpose I/O.
	RX1		Reception input pin for CAN 1.
	INT9		External interrupt request input pin for INT 9.
P23_1	P23_1	TP00_0	General purpose I/O.
	TX0		Transmission output pin for CAN 0.
P23_0	P23_0	TP00_0	General purpose I/O.
	RX0		Reception input pin for CAN 0.
	INT8		External interrupt request input pin for INT 8.
Port 24			
P24_7	P24_7	TP02_0	General purpose I/O.
	INT7		External interrupt request input pin for INT 7.
	SCL3		Serial clock I/O pin for I2C 3.
P24_6	P24_6	TP02_0	General purpose I/O.
	INT6		External interrupt request input pin for INT 6.
	SDA3		Serial data I/O pin for I2C 3.
P24_5	P24_5	TP02_0	General purpose I/O.
	INT5		External interrupt request input pin for INT 5.
	SCL2		Serial clock I/O pin for I2C 2.
P24_4	P24_4	TP02_0	General purpose I/O.
	INT4		External interrupt request input pin for INT 4.
	SDA2		Serial data I/O pin for I2C 2.
P24_3	P24_3	TP00_0	General purpose I/O.
	INT3		External interrupt request input pin for INT 3.
P24_2	P24_2	TP00_0	General purpose I/O.
	INT2		External interrupt request input pin for INT 2.
P24_1	P24_1	TP00_0	General purpose I/O.
	INT1		External interrupt request input pin for INT 1.
P24_0	P24_0	TP00_0	General purpose I/O.
	INT0		External interrupt request input pin for INT 0.

Pin Name	I/O Signal	Circuit Type	Function
Port 25			
P25_7	P25_7	TP05_0	General purpose I/O.
	SMC2M5		PWM output 2M (-) stepper motor controller 5.
P25_6	P25_6	TP05_0	General purpose I/O.
	SMC2P5		PWM output 2P (+) stepper motor controller 5.
P25_5	P25_5	TP05_0	General purpose I/O.
	SMC1M5		PWM output 1M (-) stepper motor controller 5.
P25_4	P25_4	TP05_0	General purpose I/O.
	SMC1P5		PWM output 1P (+) stepper motor controller 5.
P25_3	P25_3	TP05_0	General purpose I/O.
	SMC2M4		PWM output 2M (-) stepper motor controller 4.
P25_2	P25_2	TP05_0	General purpose I/O.
	SMC2P4		PWM output 2P (+) stepper motor controller 4.
P25_1	P25_1	TP05_0	General purpose I/O.
	SMC1M4		PWM output 1M (-) stepper motor controller 4.
P25_0	P25_0	TP05_0	General purpose I/O.
	SMC1P4		PWM output 1P (+) stepper motor controller 4.
Port 26			
P26_7	P26_7	TP05_0	General purpose I/O.
	SMC2M3		PWM output 2M (-) stepper motor controller 3.
	AN31		Analog input pin 31 for the A/D converter 1.
P26_6	P26_6	TP05_0	General purpose I/O.
	SMC2P3		PWM output 2P (+) stepper motor controller 3.
	AN30		Analog input pin 30 for the A/D converter 1.
P26_5	P26_5	TP05_0	General purpose I/O.
	SMC1M3		PWM output 1M (-) stepper motor controller 3.
	AN29		Analog input pin 29 for the A/D converter 1.
P26_4	P26_4	TP05_0	General purpose I/O.
	SMC1P3		PWM output 1P (+) stepper motor controller 3.
	AN28		Analog input pin 28 for the A/D converter 1.
P26_3	P26_3	TP05_0	General purpose I/O.
	SMC2M2		PWM output 2M (-) stepper motor controller 2.
	AN27		Analog input pin 27 for the A/D converter 1.
P26_2	P26_2	TP05_0	General purpose I/O.
	SMC2P2		PWM output 2P (+) stepper motor controller 2.
	AN26		Analog input pin 26 for the A/D converter 1.
P26_1	P26_1	TP05_0	General purpose I/O.
	SMC1M2		PWM output 1M (-) stepper motor controller 2.
	AN25		Analog input pin 25 for the A/D converter 1.

Pin Name	I/O Signal	Circuit Type	Function
P26_0	P26_0	TP05_0	General purpose I/O.
	SMC1P2		PWM output 1P (+) stepper motor controller 2.
	AN24		Analog input pin 24 for the A/D converter 1.
Port 27			
P27_7	P27_7	TP05_0	General purpose I/O.
	SMC2M1		PWM output 2M (-) stepper motor controller 1.
	AN23		Analog input pin 23 for the A/D converter 1.
P27_6	P27_6	TP05_0	General purpose I/O.
	SMC2P1		PWM output 2P (+) stepper motor controller 1.
	AN22		Analog input pin 22 for the A/D converter 1.
P27_5	P27_5	TP05_0	General purpose I/O.
	SMC1M1		PWM output 1M (-) stepper motor controller 1.
	AN21		Analog input pin 21 for the A/D converter 1.
P27_4	P27_4	TP05_0	General purpose I/O.
	SMC1P1		PWM output 1P (+) stepper motor controller 1.
	AN20		Analog input pin 20 for the A/D converter 1.
P27_3	P27_3	TP05_0	General purpose I/O.
	SMC2M0		PWM output 2M (-) stepper motor controller 0.
	AN19		Analog input pin 19 for the A/D converter 1.
P27_2	P27_2	TP05_0	General purpose I/O.
	SMC2P0		PWM output 2P (+) stepper motor controller 0.
	AN18		Analog input pin 18 for the A/D converter 1.
P27_1	P27_1	TP05_0	General purpose I/O.
	SMC1M0		PWM output 1M (-) stepper motor controller 0.
	AN17		Analog input pin 17 for the A/D converter 1.
P27_0	P27_0	TP05_0	General purpose I/O.
	SMC1P0		PWM output 1P (+) stepper motor controller 0.
	AN16		Analog input pin 16 for the A/D converter 1.
Port 28			
P28_7	P28_7	TP01_0	General purpose I/O.
	AN15		Analog input pin 15 for the A/D converter 1.
	DA1		Analog output pin 1 for the D/A converter 1.
P28_6	P28_6	TP01_0	General purpose I/O.
	AN14		Analog input pin 14 for the A/D converter 1.
	DA0		Analog output pin 0 for the D/A converter 1.
P28_5	P28_5	TP03_0	General purpose I/O.
	AN13		Analog input pin 13 for the A/D converter 1.
P28_4	P28_4	TP03_0	General purpose I/O.
	AN12		Analog input pin 12 for the A/D converter 1.

Pin Name	I/O Signal	Circuit Type	Function
P28_3	P28_3	TP03_0	General purpose I/O.
	AN11		Analog input pin 11 for the A/D converter 1.
P28_2	P28_2	TP03_0	General purpose I/O.
	AN10		Analog input pin 10 for the A/D converter 1.
P28_1	P28_1	TP03_0	General purpose I/O.
	AN9		Analog input pin 9 for the A/D converter 1.
P28_0	P28_0	TP03_0	General purpose I/O.
	AN8		Analog input pin 8 for the A/D converter 1.
Port 29			
P29_7	P29_7	TP03_0	General purpose I/O.
	AN7		Analog input pin 7 for the A/D converter 1.
P29_6	P29_6	TP03_0	General purpose I/O.
	AN6		Analog input pin 6 for the A/D converter 1.
P29_5	P29_5	TP03_0	General purpose I/O.
	AN5		Analog input pin 5 for the A/D converter 1.
P29_4	P29_4	TP03_0	General purpose I/O.
	AN4		Analog input pin 4 for the A/D converter 1.
P29_3	P29_3	TP03_0	General purpose I/O.
	AN3		Analog input pin 3 for the A/D converter 1.
P29_2	P29_2	TP03_0	General purpose I/O.
	AN2		Analog input pin 2 for the A/D converter 1.
P29_1	P29_1	TP03_0	General purpose I/O.
	AN1		Analog input pin 1 for the A/D converter 1.
P29_0	P29_0	TP03_0	General purpose I/O.
	AN0		Analog input pin 0 for the A/D converter 1.
Port 30			
P30_7	P30_7	TP08_0	General purpose I/O.
	V3		Analog input pin external reference voltage V3 LCD controller.
P30_6	P30_6	TP07_0	General purpose I/O.
	V2		Analog input pin external reference voltage V2 LCD controller.
P30_5	P30_5	TP07_0	General purpose I/O.
	V1		Analog input pin external reference voltage V1 LCD controller.
P30_4	P30_4	TP07_0	General purpose I/O.
	V0		Analog input pin external reference voltage V0 LCD controller.
P30_3	P30_3	TP06_0	General purpose I/O.
	COM3		Common driver output pin 3 LCD controller.
P30_2	P30_2	TP06_0	General purpose I/O.
	COM2		Common driver output pin 2 LCD controller.
P30_1	P30_1	TP06_0	General purpose I/O.
	COM1		Common driver output pin 1 LCD controller.

Pin Name	I/O Signal	Circuit Type	Function
P30_0	P30_0	TP06_0	General purpose I/O.
	COM0		Common driver output pin 0 LCD controller.
Port 31			
P31_7	P31_7	TP06_0	General purpose I/O.
	SEG39		Segment driver output pin 39 LCD controller.
P31_6	P31_6	TP06_0	General purpose I/O.
	SEG38		Segment driver output pin 38 LCD controller.
P31_5	P31_5	TP06_0	General purpose I/O.
	SEG37		Segment driver output pin 37 LCD controller.
P31_4	P31_4	TP06_0	General purpose I/O.
	SEG36		Segment driver output pin 36 LCD controller.
P31_3	P31_3	TP06_0	General purpose I/O.
	SEG35		Segment driver output pin 35 LCD controller.
P31_2	P31_2	TP06_0	General purpose I/O.
	SEG34		Segment driver output pin 34 LCD controller.
P31_1	P31_1	TP06_0	General purpose I/O.
	SEG33		Segment driver output pin 33 LCD controller.
P31_0	P31_0	TP06_0	General purpose I/O.
	SEG32		Segment driver output pin 32 LCD controller.
Port 32			
P32_7	P32_7	TP06_0	General purpose I/O.
	SEG31		Segment driver output pin 31 LCD controller.
P32_6	P32_6	TP06_0	General purpose I/O.
	SEG30		Segment driver output pin 30 LCD controller.
	SCK15		Clock I/O pin for LIN-USART 15.
P32_5	P32_5	TP06_0	General purpose I/O.
	SEG29		Segment driver output pin 29 LCD controller.
	SOT15		Serial data output pin for LIN-USART 15.
P32_4	P32_4	TP06_0	General purpose I/O.
	SEG28		Segment driver output pin 28 LCD controller.
	SIN15		Serial data input pin for LIN-USART 15.
P32_3	P32_3	TP06_0	General purpose I/O.
	SEG27		Segment driver output pin 27 LCD controller.
P32_2	P32_2	TP06_0	General purpose I/O.
	SEG26		Segment driver output pin 26 LCD controller.
	SCK14		Clock I/O pin for LIN-USART 14.
P32_1	P32_1	TP06_0	General purpose I/O.
	SEG25		Segment driver output pin 25 LCD controller.
	SOT14		Serial data output pin for LIN-USART 14.

Pin Name	I/O Signal	Circuit Type	Function
P32_0	P32_0	TP06_0	General purpose I/O.
	SEG24		Segment driver output pin 24 LCD controller.
	SIN14		Serial data input pin for LIN-USART 14.
Port 33			
P33_7	P33_7	TP06_0	General purpose I/O.
	SEG23		Segment driver output pin 23 LCD controller.
P33_6	P33_6	TP06_0	General purpose I/O.
	SEG22		Segment driver output pin 22 LCD controller.
	SCK13		Clock I/O pin for LIN-USART 13.
P33_5	P33_5	TP06_0	General purpose I/O.
	SEG21		Segment driver output pin 21 LCD controller.
	SOT13		Serial data output pin for LIN-USART 13.
P33_4	P33_4	TP06_0	General purpose I/O.
	SEG20		Segment driver output pin 20 LCD controller.
	SIN13		Serial data input pin for LIN-USART 13.
P33_3	P33_3	TP06_0	General purpose I/O.
	SEG19		Segment driver output pin 19 LCD controller.
P33_2	P33_2	TP06_0	General purpose I/O.
	SEG18		Segment driver output pin 18 LCD controller.
	SCK12		Clock I/O pin for LIN-USART 12.
P33_1	P33_1	TP06_0	General purpose I/O.
	SEG17		Segment driver output pin 17 LCD controller.
	SOT12		Serial data output pin for LIN-USART 12.
P33_0	P33_0	TP06_0	General purpose I/O.
	SEG16		Segment driver output pin 16 LCD controller.
	SIN12		Serial data input pin for LIN-USART 12.
Port 34			
P34_7	P34_7	TP06_0	General purpose I/O.
	SEG15		Segment driver output pin 15 LCD controller.
P34_6	P34_6	TP06_0	General purpose I/O.
	SEG14		Segment driver output pin 14 LCD controller.
	SCK11		Clock I/O pin for LIN-USART 11.
P34_5	P34_5	TP06_0	General purpose I/O.
	SEG13		Segment driver output pin 13 LCD controller.
	SOT11		Serial data output pin for LIN-USART 11.
P34_4	P34_4	TP06_0	General purpose I/O.
	SEG12		Segment driver output pin 12 LCD controller.
	SIN11		Serial data input pin for LIN-USART 11.
P34_3	P34_3	TP06_0	General purpose I/O.
	SEG11		Segment driver output pin 11 LCD controller.

Pin Name	I/O Signal	Circuit Type	Function
P34_2	P34_2	TP06_0	General purpose I/O.
	SEG10		Segment driver output pin 10 LCD controller.
	SCK10		Clock I/O pin for LIN-USART 10.
P34_1	P34_1	TP06_0	General purpose I/O.
	SEG9		Segment driver output pin 9 LCD controller.
	SOT10		Serial data output pin for LIN-USART 10.
P34_0	P34_0	TP06_0	General purpose I/O.
	SEG8		Segment driver output pin 8 LCD controller.
	SIN10		Serial data input pin for LIN-USART 10.
Port 35			
P35_7	P35_7	TP06_0	General purpose I/O.
	SEG7		Segment driver output pin 7 LCD controller.
P35_6	P35_6	TP06_0	General purpose I/O.
	SEG6		Segment driver output pin 6 LCD controller.
	SCK9		Clock I/O pin for LIN-USART 9.
P35_5	P35_5	TP06_0	General purpose I/O.
	SEG5		Segment driver output pin 5 LCD controller.
	SOT9		Serial data output pin for LIN-USART 9.
P35_4	P35_4	TP06_0	General purpose I/O.
	SEG4		Segment driver output pin 4 LCD controller.
	SIN9		Serial data input pin for LIN-USART 9.
P35_3	P35_3	TP06_0	General purpose I/O.
	SEG3		Segment driver output pin 3 LCD controller.
P35_2	P35_2	TP06_0	General purpose I/O.
	SEG2		Segment driver output pin 2 LCD controller.
	SCK8		Clock I/O pin for LIN-USART 8.
P35_1	P35_1	TP06_0	General purpose I/O.
	SEG1		Segment driver output pin 1 LCD controller.
	SOT8		Serial data output pin for LIN-USART 8.
P35_0	P35_0	TP06_0	General purpose I/O.
	SEG0		Segment driver output pin 0 LCD controller.
	SIN8		Serial data input pin for LIN-USART 8.

2. I/O Circuit Types

2.1 I/O Cell List MB91V460

Type	Input			Analog Line	Output Driver	Comment
	Pull Up / Down (50 kOhm)	CMOS (C) CMOS Hysteresis (CH) Automotive (AH)	Input Stop			
TP00_0	Up/Down switch	CH / AH switch	Stop	-	4 mA	-
TP01_0	-	CH / AH switch	Stop	In/Out	4 mA	ADC / DAC
TP02_0	-	CH / AH switch	Stop	-	3 mA	I2C
TP03_0	Up/Down switch	CH / AH switch	Stop	Input	4 mA	AN
TP04_0	Up/Down switch	CH / AH / TTL Switch	Stop	-	4 mA	External Bus
TP05_0	-	CH / AH switch	Stop	Input	30 mA	SMC / ADC
TP06_0	-	CH / AH switch	Stop	-	4 mA	LCD COM / SEG
TP07_0	-	CH / AH switch	Stop	In/Out	4 mA	LCD V0 / V1 / V2
TP08_0	-	CH / AH switch	Stop	In/Out	4 mA	LCD V3
TC00_0	-	CCH	-	-	-	HSTX
TC01_0	Up	CCH	-	-	-	RSTX / NMIX EBREAKX
TC02_0	Up	CCH	-	-	-	INITX
TC02_1	-	CCH	-	-	-	MD
TC10_0	-	-	-	-	8 mA	MONCLK
TO00_0	-	-	Stop	-	-	4 MHz Oscillator
TO00_1	-	-	Stop	-	-	4 MHz Oscillator
TO01_0	-	-	Stop	-	-	32 kHz Oscillator
TO01_1	-	-	Stop	-	-	32 kHz Oscillator
TA02_0	-	-	-	Input	-	ALARM

Note: This table shows the I/O cells used for MB91V460. Please refer to the appropriate datasheet for I/O circuits used on flash devices.

Note: The most I/O cells have programmable input levels (CH / AH [/ TTL]). About the programming, see section Port Input Level Selection on page 500.

Note: For the programming of input pull-up or pull-down function, see section Programmable Pull-Up/Pull Down Resistors on page 503.

2.2 I/O Input Voltages (VIL/VIH)

There are various types of input stages. The following table lists the input voltages VIL / VIH.

Type	Description	VIL	VIH
C	CMOS input	0.25 x VDD	0.65 x VDD
CH	CMOS Hysteresis Trigger input (I/O Port)	0.3 x VDD	0.7 x VDD
CCH	CMOS Hysteresis Trigger input (Control)	0.2 x VDD	0.8 x VDD
AH	CMOS Automotive Hysteresis Trigger input	0.5 x VDD	0.8 x VDD
TTL	TTL Input	0.8 V	2.1 V

3. Port Register Settings

3.1 General Rules

For all ports, the following rules are valid:

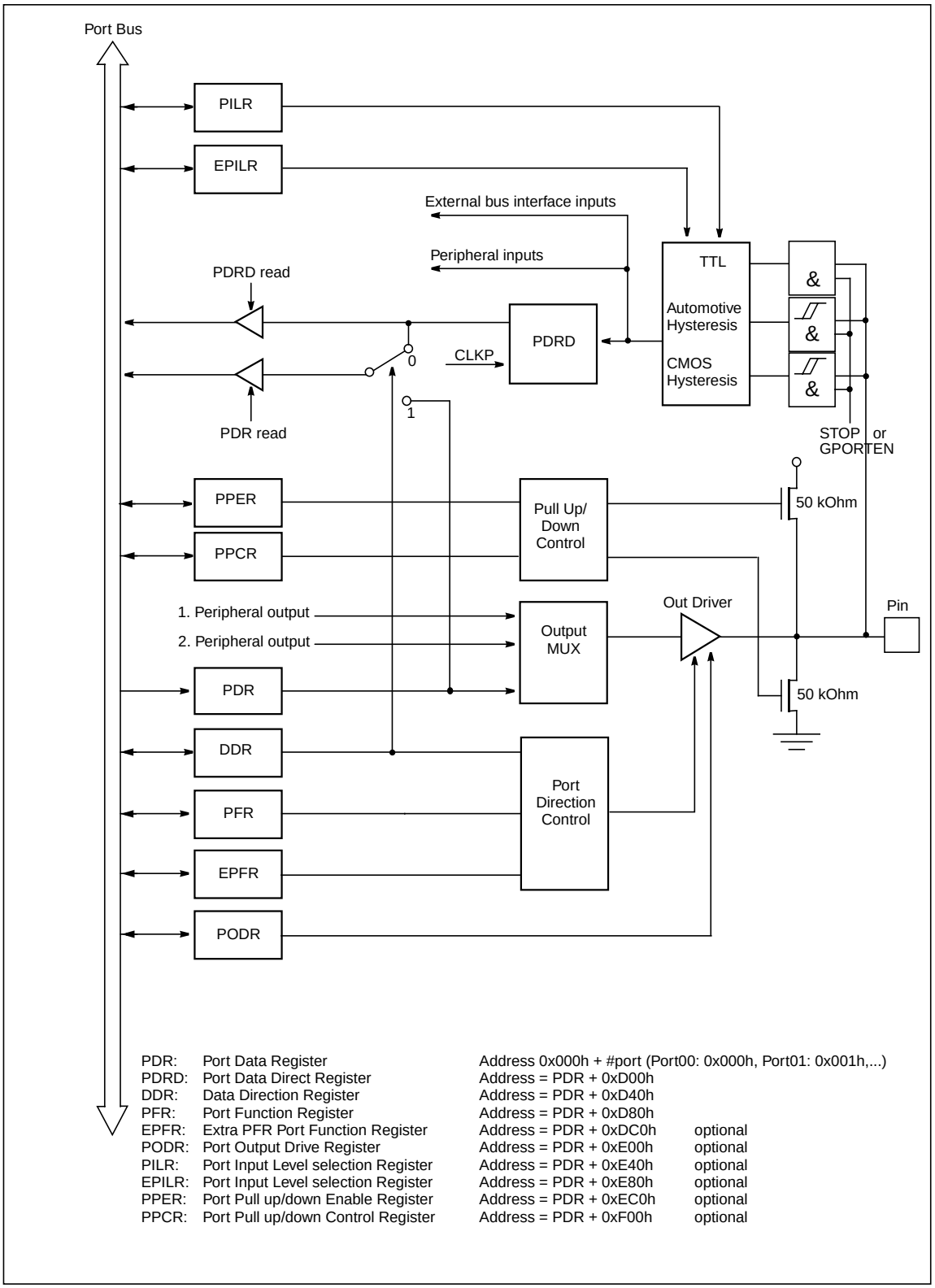
1. **All port inputs are disabled by default** to avoid transverse current floating before the ports are configured by software. After configuring each port pin according to its function it is necessary to enable the port inputs with the global port enable (PORTEN.GPORTEN). See section Port Input Enable on page 459.
2. Each port has a Port Data Register direct read (PDRD) to sample the pin data with CLKP. This register is read-only.
3. Each port has a Data Direction Register (DDR) to switch the port's input/output direction. After reset, all ports are input (DDR=0x00).
   - Port Input mode (PFR = "0" and DDR = "0")

     PDRD read : Reads the sampled pin data.
     PDR read : Reads the sampled pin data.
     PDR write : Writes the PDR setting value, has no effect on the pin value.
   - Port Output mode (PFR = "0" and DDR = "1")

     PDRD read : Reads the sampled pin data.
     PDR read : Reads the PDR register value.
     PDR write : Writes the PDR setting value to the corresponding external pins.
4. On a **Read-Modify-Write** instruction (bit operations) always the PDR register is read independent of the Data Direction Register (DDR) settings.
5. Certain ports have a Port Function Register (PFR) and an Extra Port Function Register (EPFR). To enable the function determined by EPFR='1' it is necessary to also set PFR='1'. On MB91V460 the behaviour of setting EPFR='1' and PFR='0' equals the port input/output mode (reserved for future use).
6. Each port has a Port Input Level Register (PILR) to bit-wise select the input level (CMOS-Hysteresis / Automotive [/ TTL]). The default value depends on the function of the port.
   The input level can be set in every device mode. See section Port Input Level Selection on page 500.
7. Certain ports have programmable Pull-Ups/Pull-Downs (50 kOhm) which are enabled bit-wise by their Pull-Up/Pull-Down Enable Registers (PPER) and Pull-Up/Pull-Down Control Registers (PPCR). See section Programmable Pull-Up/Pull Down Resistors on page 503.
8. Each port has one or two Port Function Registers: PFR and, if necessary, Extra PFR (EPFR). Together, they can serve up to 3 resource I/O's per pin. See section Port Function Register Setup on page 460.
9. Port setup controlled by the MD[2:0] pins and the mode register MODR overwrites the setup in the port registers. E.g. External Bus Mode overwrites port register setup. The external bus signal output can be disabled by setting the PFR of the pin to port mode (PFR='0).
10. **Resource input** lines are generally connected to the pin and are enabled by setting the appropriate functionality inside the resource. There are exceptions which are listed in Port Function Register Setup on page 460.
11. **External Interrupt input** lines are always connected to the pin and are enabled in the External Interrupt unit.
12. **In STOP mode** (STCR:STOP set and STCR:HIZ not set) all pins keep their state (input or output depending on the configuration before entering the STOP mode) and the input stages and lines are internally fixed to avoid transverse current. External interrupt input pins are not fixed if the corresponding pin is selected by using the PFR='1' setting and the corresponding external interrupt is enabled with the ENIR0, resp. ENIR1 registers. Pull-Ups and Pull-Downs are enabled.
13. **In STOP-HIZ mode** (STCR:STOP and STCR:HIZ set) all pins are switching to input (high impedance state) and all input stages and lines are internally fixed to avoid floating. External interrupt input pins are not fixed if the corresponding pin is selected by using the PFR='1' setting and the corresponding external interrupt is enabled with the ENIR0, resp. ENIR1 registers. Pull-Ups and Pull-Downs are disabled.

14. **Resource output** lines are enabled by setting the corresponding PFR and/or EPFR bit in the port. Details see section Port Function Register Setup on page 460. LIN-USART outputs (SOT) must be enabled additionally by setting the SOE bit in the LIN-USART control.
15. **Resource bidirectional** signals (e.g. SCK of the LIN-USART) are enabled by setting the corresponding PFR and/or EPFR bit in the port. The signal direction is controlled by the setup of the resource, e.g. via the output enable bits. Details see section Port Function Register Setup on page 460.

3.2 I/O Port Block Diagram



PDR:	Port Data Register	Address 0x000h + #port (Port00: 0x000h, Port01: 0x001h,...)	
PDRD:	Port Data Direct Register	Address = PDR + 0xD00h	
DDR:	Data Direction Register	Address = PDR + 0xD40h	
PFR:	Port Function Register	Address = PDR + 0xD80h	
EPFR:	Extra PFR Port Function Register	Address = PDR + 0xDC0h	optional
PODR:	Port Output Drive Register	Address = PDR + 0xE00h	optional
PILR:	Port Input Level selection Register	Address = PDR + 0xE40h	optional
EPILR:	Port Input Level selection Register	Address = PDR + 0xE80h	optional
PPER:	Port Pull up/down Enable Register	Address = PDR + 0xEC0h	optional
PPCR:	Port Pull up/down Control Register	Address = PDR + 0xF00h	optional

3.3 Port Input Enable

This section describes the Port Input Enable function.

■ **PORTEN:** Port Input Enable.

	Addr	7	6	5	4	3	2	1	0	initial
PORTEN	0498h	-	-	-	-	-	-	CPORTEN	GPORTEN	---- --00
		-	-	-	-	-	-	R/W	R/W	

All port inputs are disabled by default to avoid transverse current floating in the IO input stages and the subsequent logic. After configuring all ports according to their functional specification (input level, output drive, pull-up or pull-down resistor, etc.) it is mandatory to globally enable the inputs by setting the port input enable bit.

GPORTEN 0 - The inputs of all ports are disabled.
1 - The inputs of all ports are enabled.

CPORTEN 0 - The inputs of the bootloader communication ports are disabled.
1 - The inputs of the bootloader communication ports are enabled.

■ **General I/0 of the external bus interface:**

- Output lines:

  The external bus interface is initialized by default after power-on.
  Hence, the address lines and the relevant control lines are set to output and high after power-on.

- Input lines:

  The input lines of the external bus interface are enabled independent from GPORTEN by the following constellations:
  External Vector Fetch (MD2-0=001):
  At Vector Fetch: D31-24 are enabeld
  After Vector Fetch:

Value of Mode-Vector (WTH1-0)	Enabled Data Lines of the external bus
00 (byte)	D31-24
01 (halfword)	D31-16
10 (word)	D31-00
11 (single)	none

  Internal Vector Fetch (MD2-0=000):
  After Vector Fetch:
  D31-00 are enabled, because of Fixed-Mode-Vector-WTH1-0 = 10 (word).

3.4 Port Function Register Setup

This section describes the Port Function Registers of each port.

■ **P00:** The functions of Port 00 are controlled by PFR00

	Addr	7	6	5	4	3	2	1	0	initial
PFR00	0D80h	PFR00.7	PFR00.6	PFR00.5	PFR00.4	PFR00.3	PFR00.2	PFR00.1	PFR00.0	1111 1111
EPFR00	0DC0h	-	-	-	-	-	-	-	-	---- ----
		R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	

If the external bus interface is enabled (by mode pins MD[2:0] or mode vector), **P00[7:0]** is input/output for data lines **D[31:24]**. Otherwise, the port can be used as general purpose port.

PFR00.7 0 - Port is in general purpose port mode.
1 - Port is in external bus mode (if external bus is enabled otherwise general purpose port).

PFR00.6 0 - Port is in general purpose port mode.
1 - Port is in external bus mode (if external bus is enabled otherwise general purpose port).

PFR00.5 0 - Port is in general purpose port mode.
1 - Port is in external bus mode (if external bus is enabled otherwise general purpose port).

PFR00.4 0 - Port is in general purpose port mode.
1 - Port is in external bus mode (if external bus is enabled otherwise general purpose port).

PFR00.3 0 - Port is in general purpose port mode.
1 - Port is in external bus mode (if external bus is enabled otherwise general purpose port).

PFR00.2 0 - Port is in general purpose port mode.
1 - Port is in external bus mode (if external bus is enabled otherwise general purpose port).

PFR00.1 0 - Port is in general purpose port mode.
1 - Port is in external bus mode (if external bus is enabled otherwise general purpose port).

PFR00.0 0 - Port is in general purpose port mode.
1 - Port is in external bus mode (if external bus is enabled otherwise general purpose port).

■ **P01:** The functions of Port 01 are controlled by PFR01

	Addr	7	6	5	4	3	2	1	0	initial
PFR01	0D81h	PFR01.7	PFR01.6	PFR01.5	PFR01.4	PFR01.3	PFR01.2	PFR01.1	PFR01.0	1111 1111
EPFR01	0DC1h	-	-	-	-	-	-	-	-	---- ----
		R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	

If the external bus interface is enabled (by mode pins MD[2:0] or mode vector), **P01[7:0]** is input/output for data lines **D[23:16]**. Otherwise, the port can be used as general purpose port.

PFR01.7 0 - Port is in general purpose port mode.
1 - Port is in external bus mode (if external bus is enabled otherwise general purpose port).

PFR01.6 0 - Port is in general purpose port mode.
1 - Port is in external bus mode (if external bus is enabled otherwise general purpose port).

PFR01.5 0 - Port is in general purpose port mode.
1 - Port is in external bus mode (if external bus is enabled otherwise general purpose port).

PFR01.4 0 - Port is in general purpose port mode.
1 - Port is in external bus mode (if external bus is enabled otherwise general purpose port).

PFR01.3 0 - Port is in general purpose port mode.
1 - Port is in external bus mode (if external bus is enabled otherwise general purpose port).

PFR01.2 0 - Port is in general purpose port mode.
1 - Port is in external bus mode (if external bus is enabled otherwise general purpose port).

PFR01.1 0 - Port is in general purpose port mode.
1 - Port is in external bus mode (if external bus is enabled otherwise general purpose port).

PFR01.0 0 - Port is in general purpose port mode.
1 - Port is in external bus mode (if external bus is enabled otherwise general purpose port).

■ **P02:** The functions of Port 02 are controlled by PFR02

	Addr	7	6	5	4	3	2	1	0	initial
PFR02	0D82h	PFR02.7	PFR02.6	PFR02.5	PFR02.4	PFR02.3	PFR02.2	PFR02.1	PFR02.0	1111 1111
EPFR02	0DC2h	-	-	-	-	-	-	-	-	---- ----
		R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	

If the external bus interface is enabled (by mode pins MD[2:0] or mode vector), **P02[7:0]** is input/output for data lines **D[15:8]**. Otherwise, the port can be used as general purpose port.

PFR02.7 0 - Port is in general purpose port mode.
1 - Port is in external bus mode (if external bus is enabled otherwise general purpose port).

PFR02.6 0 - Port is in general purpose port mode.
1 - Port is in external bus mode (if external bus is enabled otherwise general purpose port).

PFR02.5 0 - Port is in general purpose port mode.
1 - Port is in external bus mode (if external bus is enabled otherwise general purpose port).

PFR02.4 0 - Port is in general purpose port mode.
1 - Port is in external bus mode (if external bus is enabled otherwise general purpose port).

PFR02.3 0 - Port is in general purpose port mode.
1 - Port is in external bus mode (if external bus is enabled otherwise general purpose port).

PFR02.2 0 - Port is in general purpose port mode.
1 - Port is in external bus mode (if external bus is enabled otherwise general purpose port).

PFR02.1 0 - Port is in general purpose port mode.
1 - Port is in external bus mode (if external bus is enabled otherwise general purpose port).

PFR02.0 0 - Port is in general purpose port mode.
1 - Port is in external bus mode (if external bus is enabled otherwise general purpose port).

■ **P03:** The functions of Port 03 are controlled by PFR03

	Addr	7	6	5	4	3	2	1	0	initial
PFR03	0D83h	PFR03.7	PFR03.6	PFR03.5	PFR03.4	PFR03.3	PFR03.2	PFR03.1	PFR03.0	1111 1111
EPFR03	0DC3h	-	-	-	-	-	-	-	-	---- ----
		R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	

If the external bus interface is enabled (by mode pins MD[2:0] or mode vector), **P03[7:0]** is input/output for data lines **D[7:0]**. Otherwise, the port can be used as general purpose port.

PFR03.7 0 - Port is in general purpose port mode.
1 - Port is in external bus mode (if external bus is enabled otherwise general purpose port).

PFR03.6 0 - Port is in general purpose port mode.
1 - Port is in external bus mode (if external bus is enabled otherwise general purpose port).

PFR03.5 0 - Port is in general purpose port mode.
1 - Port is in external bus mode (if external bus is enabled otherwise general purpose port).

PFR03.4 0 - Port is in general purpose port mode.
1 - Port is in external bus mode (if external bus is enabled otherwise general purpose port).

PFR03.3 0 - Port is in general purpose port mode.
1 - Port is in external bus mode (if external bus is enabled otherwise general purpose port).

PFR03.2 0 - Port is in general purpose port mode.
1 - Port is in external bus mode (if external bus is enabled otherwise general purpose port).

PFR03.1 0 - Port is in general purpose port mode.
1 - Port is in external bus mode (if external bus is enabled otherwise general purpose port).

PFR03.0 0 - Port is in general purpose port mode.
1 - Port is in external bus mode (if external bus is enabled otherwise general purpose port).

■ **P04:** The functions of Port 04 are controlled by PFR04

	Addr	7	6	5	4	3	2	1	0	initial
PFR04	0D84h	PFR04.7	PFR04.6	PFR04.5	PFR04.4	PFR04.3	PFR04.2	PFR04.1	PFR04.0	1111 1111
EPFR04	0DC4h	-	-	-	-	-	-	-	-	---- ----
		R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	

If the external bus interface is enabled (by mode pins MD[2:0] or mode vector), **P04[7:0]** is input/output for address lines **A[31:24]**. Otherwise, the port can be used as general purpose port.

PFR04.7 0 - Port is in general purpose port mode.
1 - Port is in external bus mode (if external bus is enabled otherwise general purpose port).

PFR04.6 0 - Port is in general purpose port mode.
1 - Port is in external bus mode (if external bus is enabled otherwise general purpose port).

PFR04.5 0 - Port is in general purpose port mode.
1 - Port is in external bus mode (if external bus is enabled otherwise general purpose port).

PFR04.4 0 - Port is in general purpose port mode.
1 - Port is in external bus mode (if external bus is enabled otherwise general purpose port).

PFR04.3 0 - Port is in general purpose port mode.
1 - Port is in external bus mode (if external bus is enabled otherwise general purpose port).

PFR04.2 0 - Port is in general purpose port mode.
1 - Port is in external bus mode (if external bus is enabled otherwise general purpose port).

PFR04.1 0 - Port is in general purpose port mode.
1 - Port is in external bus mode (if external bus is enabled otherwise general purpose port).

PFR04.0 0 - Port is in general purpose port mode.
1 - Port is in external bus mode (if external bus is enabled otherwise general purpose port).

■ **P05:** The functions of Port 05 are controlled by PFR05

	Addr	7	6	5	4	3	2	1	0	initial
PFR05	0D85h	PFR05.7	PFR05.6	PFR05.5	PFR05.4	PFR05.3	PFR05.2	PFR05.1	PFR05.0	1111 1111
EPFR05	0DC5h	-	-	-	-	-	-	-	-	---- ----
		R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	

If the external bus interface is enabled (by mode pins MD[2:0] or mode vector), **P05[7:0]** is input/output for address lines **A[23:16]**. Otherwise, the port can be used as general purpose port.

PFR05.7 0 - Port is in general purpose port mode.
1 - Port is in external bus mode (if external bus is enabled otherwise general purpose port).

PFR05.6 0 - Port is in general purpose port mode.
1 - Port is in external bus mode (if external bus is enabled otherwise general purpose port).

PFR05.5 0 - Port is in general purpose port mode.
1 - Port is in external bus mode (if external bus is enabled otherwise general purpose port).

PFR05.4 0 - Port is in general purpose port mode.
1 - Port is in external bus mode (if external bus is enabled otherwise general purpose port).

PFR05.3 0 - Port is in general purpose port mode.
1 - Port is in external bus mode (if external bus is enabled otherwise general purpose port).

PFR05.2 0 - Port is in general purpose port mode.
1 - Port is in external bus mode (if external bus is enabled otherwise general purpose port).

PFR05.1 0 - Port is in general purpose port mode.
1 - Port is in external bus mode (if external bus is enabled otherwise general purpose port).

PFR05.0 0 - Port is in general purpose port mode.
1 - Port is in external bus mode (if external bus is enabled otherwise general purpose port).

■ **P06:** The functions of Port 06 are controlled by PFR06

	Addr	7	6	5	4	3	2	1	0	initial
PFR06	0D86h	PFR06.7	PFR06.6	PFR06.5	PFR06.4	PFR06.3	PFR06.2	PFR06.1	PFR06.0	1111 1111
EPFR06	0DC6h	-	-	-	-	-	-	-	-	---- ----
		R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	

If the external bus interface is enabled (by mode pins MD[2:0] or mode vector), **P06[7:0]** is input/output for address lines **A[15:8]**. Otherwise, the port can be used as general purpose port.

PFR06.7 0 - Port is in general purpose port mode.
1 - Port is in external bus mode (if external bus is enabled otherwise general purpose port).

PFR06.6 0 - Port is in general purpose port mode.
1 - Port is in external bus mode (if external bus is enabled otherwise general purpose port).

PFR06.5 0 - Port is in general purpose port mode.
1 - Port is in external bus mode (if external bus is enabled otherwise general purpose port).

PFR06.4 0 - Port is in general purpose port mode.
1 - Port is in external bus mode (if external bus is enabled otherwise general purpose port).

PFR06.3 0 - Port is in general purpose port mode.
1 - Port is in external bus mode (if external bus is enabled otherwise general purpose port).

PFR06.2 0 - Port is in general purpose port mode.
1 - Port is in external bus mode (if external bus is enabled otherwise general purpose port).

PFR06.1 0 - Port is in general purpose port mode.
1 - Port is in external bus mode (if external bus is enabled otherwise general purpose port).

PFR06.0 0 - Port is in general purpose port mode.
1 - Port is in external bus mode (if external bus is enabled otherwise general purpose port).

■ **P07:** The functions of Port 07 are controlled by PFR07

	Addr	7	6	5	4	3	2	1	0	initial
PFR07	0D87h	PFR07.7	PFR07.6	PFR07.5	PFR07.4	PFR07.3	PFR07.2	PFR07.1	PFR07.0	1111 1111
EPFR07	0DC7h	-	-	-	-	-	-	-	-	---- ----
		R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	

If the external bus interface is enabled (by mode pins MD[2:0] or mode vector), **P07[7:0]** is input/output for address lines **A[7:0]**. Otherwise, the port can be used as general purpose port.

PFR07.7 0 - Port is in general purpose port mode.
1 - Port is in external bus mode (if external bus is enabled otherwise general purpose port).

PFR07.6 0 - Port is in general purpose port mode.
1 - Port is in external bus mode (if external bus is enabled otherwise general purpose port).

PFR07.5 0 - Port is in general purpose port mode.
1 - Port is in external bus mode (if external bus is enabled otherwise general purpose port).

PFR07.4 0 - Port is in general purpose port mode.
1 - Port is in external bus mode (if external bus is enabled otherwise general purpose port).

PFR07.3 0 - Port is in general purpose port mode.
1 - Port is in external bus mode (if external bus is enabled otherwise general purpose port).

PFR07.2 0 - Port is in general purpose port mode.
1 - Port is in external bus mode (if external bus is enabled otherwise general purpose port).

PFR07.1 0 - Port is in general purpose port mode.
1 - Port is in external bus mode (if external bus is enabled otherwise general purpose port).

PFR07.0 0 - Port is in general purpose port mode.
1 - Port is in external bus mode (if external bus is enabled otherwise general purpose port).

■ **P08:** The functions of Port 08 are controlled by PFR08

	Addr	7	6	5	4	3	2	1	0	initial
PFR08	0D88h	PFR08.7	PFR08.6	PFR08.5	PFR08.4	PFR08.3	PFR08.2	PFR08.1	PFR08.0	1111 1111
EPFR08	0DC8h	-	-	-	-	-	-	-	-	---- ----
		R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	

If the external bus interface is enabled (by mode pins MD[2:0] or mode vector), **P08[7:0]** is input/output for external bus control signals **RDY, BRQ, BGRNTX, RDX, WRX[3:0]**. Otherwise, the port can be used as general purpose port.

PFR08.7 0 - Port is in general purpose port mode.
1 - Port is in external bus mode (if external bus is enabled otherwise general purpose port):

External bus function is **RDY**

PFR08.6 0 - Port is in general purpose port mode.
1 - Port is in external bus mode (if external bus is enabled otherwise general purpose port):

External bus function is **BRQ**

PFR08.5 0 - Port is in general purpose port mode.
1 - Port is in external bus mode (if external bus is enabled otherwise general purpose port):

External bus function is **BGRNTX**

PFR08.4 0 - Port is in general purpose port mode.
1 - Port is in external bus mode (if external bus is enabled otherwise general purpose port):

External bus function is **RDX**

PFR08.3 0 - Port is in general purpose port mode.
1 - Port is in external bus mode (if external bus is enabled otherwise general purpose port):

External bus function is **WRX3**

PFR08.2 0 - Port is in general purpose port mode.
1 - Port is in external bus mode (if external bus is enabled otherwise general purpose port):

External bus function is **WRX2**

PFR08.1 0 - Port is in general purpose port mode.
1 - Port is in external bus mode (if external bus is enabled otherwise general purpose port):

External bus function is **WRX1**

PFR08.0 0 - Port is in general purpose port mode.
1 - Port is in external bus mode (if external bus is enabled otherwise general purpose port):

External bus function is **WRX0**

■ **P09:** The functions of Port 09 are controlled by PFR09

	Addr	7	6	5	4	3	2	1	0	initial
PFR09	0D89h	PFR09.7	PFR09.6	PFR09.5	PFR09.4	PFR09.3	PFR09.2	PFR09.1	PFR09.0	1111 1111
EPFR09	0DC9h	-	-	-	-	-	-	-	-	---- ----
		R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	

If the external bus interface is enabled (by mode pins MD[2:0] or mode vector), **P09[7:0]** is input/output for external bus control signals **CSX[7:0]**. Otherwise, the port can be used as general purpose port.

PFR09.7 0 - Port is in general purpose port mode.
1 - Port is in external bus mode (if external bus is enabled otherwise general purpose port).

PFR09.6 0 - Port is in general purpose port mode.
1 - Port is in external bus mode (if external bus is enabled otherwise general purpose port).

PFR09.5 0 - Port is in general purpose port mode.
1 - Port is in external bus mode (if external bus is enabled otherwise general purpose port).

PFR09.4 0 - Port is in general purpose port mode.
1 - Port is in external bus mode (if external bus is enabled otherwise general purpose port).

PFR09.3 0 - Port is in general purpose port mode.
1 - Port is in external bus mode (if external bus is enabled otherwise general purpose port).

PFR09.2 0 - Port is in general purpose port mode.
1 - Port is in external bus mode (if external bus is enabled otherwise general purpose port).

PFR09.1 0 - Port is in general purpose port mode.
1 - Port is in external bus mode (if external bus is enabled otherwise general purpose port).

PFR09.0 0 - Port is in general purpose port mode.
1 - Port is in external bus mode (if external bus is enabled otherwise general purpose port).

■ **P10:** The functions of Port 10 are controlled by PFR10 and EPFR10

	Addr	7	6	5	4	3	2	1	0	initial
PFR10	0D8Ah	-	PFR10.6	PFR10.5	PFR10.4	PFR10.3	PFR10.2	PFR10.1	PFR10.0	-111 1111
EPFR10	0DCAh	-	-	EPFR10.5	EPFR10.4	-	-	-	EPFR10.0	--00 ---0
		R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	

If the external bus interface is enabled (by mode pins MD[2:0] or mode vector), **P10[7:0]** is input/output for external bus control signals **MCLKE, MCLKI, MCLKO, WEX, BAAX, ASX, SYSCLK**. Otherwise, the port can be used as general purpose port.

PFR10.6 0 - Port is in general purpose port mode.
1 - Port is in external bus mode (if external bus is enabled otherwise general purpose port):

External bus function is **MCLKE**

PFR10.5 0 - Port is in general purpose port mode.
1 - Port is in external bus mode (if external bus is enabled otherwise general purpose port):

EPFR10.5 0 - External bus function is **MCLKI**
1 - External bus function is **$\overline{\text{MCLKI}}$** (inverted input)

PFR10.4 0 - Port is in general purpose port mode.
1 - Port is in external bus mode (if external bus is enabled otherwise general purpose port):

EPFR10.4 0 - External bus function is **MCLKO**
1 - External bus function is **$\overline{\text{MCLKO}}$** (inverted output)

PFR10.3 0 - Port is in general purpose port mode.
1 - Port is in external bus mode (if external bus is enabled otherwise general purpose port):

External bus function is **WEX**

PFR10.2 0 - Port is in general purpose port mode.
1 - Port is in external bus mode (if external bus is enabled otherwise general purpose port):

External bus function is **BAAX**

PFR10.1 0 - Port is in general purpose port mode.
1 - Port is in external bus mode (if external bus is enabled otherwise general purpose port):

External bus function is **ASX**

PFR10.0 0 - Port is in general purpose port mode.
1 - Port is in external bus mode (if external bus is enabled otherwise general purpose port):

EPFR10.4 0 - External bus function is **SYSCLK**
1 - External bus function is **$\overline{\text{SYSCLK}}$** (inverted output)

■ **P11:** The functions of Port 11 are controlled by PFR11

	Addr	7	6	5	4	3	2	1	0	initial
PFR11	0D8Bh	-	-	-	-	-	-	PFR11.1	PFR11.0	---- --00
EPFR11	0DCBh	-	-	-	-	-	-	-	-	---- ----
		R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	

P11[7:0] is input/output for DMA control signals **IOWRX, IORDX**. Otherwise, the port can be used as general purpose port.

PFR11.1 0 - Port is in general purpose port mode.
1 - Port is in DMA function mode:

DMA function is **IOWRX** output

PFR11.0 0 - Port is in general purpose port mode.
1 - Port is in DMA function mode:

DMA function is **IORDX** output

■ **P12:** The functions of Port 12 are controlled by PFR12 and EPFR12

	Addr	7	6	5	4	3	2	1	0	initial
PFR12	0D8Ch	PFR12.7	PFR12.6	PFR12.5	PFR12.4	PFR12.3	PFR12.2	PFR12.1	PFR12.0	0000 0000
EPFR12	0DCCh	-	EPFR12.6	-	-	-	EPFR12.2	-	-	-0-- -0--
		R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	

P12[7:0] is input/output for DMA control signals **DEOP, DEOTX, DACKX, DREQ** for DMA channels 2 and 3. Otherwise, the port can be used as general purpose port.

PFR12.7 0 - Port is in general purpose port mode.
1 - Port is in DMA function mode:

DMA function is **DEOP3** output

PFR12.6 0 - Port is in general purpose port mode.
1 - Port is in DMA function mode:

EPFR12.6 0 - DMA function is **DEOTX3** input
1 - DMA function is **DEOP3** output

PFR12.5 0 - Port is in general purpose port mode.
1 - Port is in DMA function mode:

DMA function is **DACKX3** output

PFR12.4 0 - Port is in general purpose port mode.
1 - Port is in DMA function mode:

DMA function is **DREQ3** input

PFR12.3 0 - Port is in general purpose port mode.
1 - Port is in DMA function mode:

DMA function is **DEOP2** output

PFR12.2 0 - Port is in general purpose port mode.
1 - Port is in DMA function mode:

EPFR12.2 0 - DMA function is **DEOTX2** input
1 - DMA function is **DEOP2** output

PFR12.1 0 - Port is in general purpose port mode.
1 - Port is in DMA function mode:

DMA function is **DACKX2** output

PFR12.0 0 - Port is in general purpose port mode.
1 - Port is in DMA function mode:

DMA function is **DREQ2** input

■ **P13:** The functions of Port 13 are controlled by PFR13 and EPFR13

	Addr	7	6	5	4	3	2	1	0	initial
PFR13	0D8Dh	PFR13.7	PFR13.6	PFR13.5	PFR13.4	PFR13.3	PFR13.2	PFR13.1	PFR13.0	0000 0000
EPFR13	0DCDh	-	EPFR13.6	-	-	-	EPFR13.2	-	-	-0-- -0--
		R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	

P13[7:0] is input/output for DMA control signals **DEOP, DEOTX, DACKX, DREQ** for DMA channels 0 and 1. Otherwise, the port can be used as general purpose port.

PFR13.7 0 - Port is in general purpose port mode.
1 - Port is in DMA function mode:

DMA function is **DEOP1** output

PFR13.6 0 - Port is in general purpose port mode.
1 - Port is in DMA function mode:

EPFR13.6 0 - DMA function is **DEOTX1** input
1 - DMA function is **DEOP1** output

PFR13.5 0 - Port is in general purpose port mode.
1 - Port is in DMA function mode:

DMA function is **DACKX1** output

PFR13.4 0 - Port is in general purpose port mode.
1 - Port is in DMA function mode:

DMA function is **DREQ1** input

PFR13.3 0 - Port is in general purpose port mode.
1 - Port is in DMA function mode:

DMA function is **DEOP0** output

PFR13.2 0 - Port is in general purpose port mode.
1 - Port is in DMA function mode:

EPFR13.2 0 - DMA function is **DEOTX0** input
1 - DMA function is **DEOP0** output

PFR13.1 0 - Port is in general purpose port mode.
1 - Port is in DMA function mode:

DMA function is **DACKX0** output

PFR13.0 0 - Port is in general purpose port mode.
1 - Port is in DMA function mode:

DMA function is **DREQ0** input

■ **P14:** The functions of Port 14 are controlled by PFR14 and EPFR14

	Addr	7	6	5	4	3	2	1	0	initial
PFR14	0D8Eh	PFR14.7	PFR14.6	PFR14.5	PFR14.4	PFR14.3	PFR14.2	PFR14.1	PFR14.0	0000 0000
EPFR14	0DCEh	EPFR14.7	EPFR14.6	EPFR14.5	EPFR14.4	EPFR14.3	EPFR14.2	EPFR14.1	EPFR14.0	0000 0000
		R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	

P14[7:0] is input/output for Input Capture inputs **ICU[7:0]**, Reload Timer triggers **TIN[7:0]** and PWM inputs **TTG[15:0]**. Otherwise, the port can be used as general purpose port.

PFR14.7 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

Resource function is **TIN7** and **TTG15/7** input, and

EPFR14.7 0 - Resource function is **ICU7** input
1 - ICU7 is internally connected to LSYN of LIN-UART 7/15

PFR14.6 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

Resource function is **TIN6** and **TTG14/6** input, and

EPFR14.6 0 - Resource function is **ICU6** input
1 - ICU6 is internally connected to LSYN of LIN-UART 6/14

PFR14.5 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

Resource function is **TIN5** and **TTG13/5** input, and

EPFR14.5 0 - Resource function is **ICU6** input
1 - ICU5 is internally connected to LSYN of LIN-UART 5/13

PFR14.4 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

Resource function is **TIN4** and **TTG12/4** input, and

EPFR14.4 0 - Resource function is **ICU4** input
1 - ICU4 is internally connected to LSYN of LIN-UART 4/12

PFR14.3 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

Resource function is **TIN3** and **TTG11/3** input, and

EPFR14.3 0 - Resource function is **ICU3** input
1 - ICU3 is internally connected to LSYN of LIN-UART 3/11

PFR14.2 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

Resource function is **TIN2** and **TTG10/2** input, and

EPFR14.2 0 - Resource function is **ICU2** input
1 - ICU2 is internally connected to LSYN of LIN-UART 2/10

PFR14.1 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

Resource function is **TIN1** and **TTG9/1** input, and

EPFR14.1 0 - Resource function is **ICU1** input
1 - ICU1 is internally connected to LSYN of LIN-UART 1/9

PFR14.0 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

Resource function is **TIN0** and **TTG8/0** input, and

EPFR14.0 0 - Resource function is **ICU0** input
1 - ICU0 is internally connected to LSYN of LIN-UART 0/8

■ **P15:** The functions of Port 15 are controlled by PFR15 and EPFR15

	Addr	7	6	5	4	3	2	1	0	initial
PFR15	0D8Fh	PFR15.7	PFR15.6	PFR15.5	PFR15.4	PFR15.3	PFR15.2	PFR15.1	PFR15.0	0000 0000
EPFR15	0DCFh	EPFR15.7	EPFR15.6	EPFR15.5	EPFR15.4	EPFR15.3	EPFR15.2	EPFR15.1	EPFR15.0	0000 0000
		R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	

P15[7:0] is input/output for Output Compare outputs **OCU[7:0]** and Reload Timer outputs **TOT[7:0]**. Otherwise, the port can be used as general purpose port.

PFR15.7 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

EPFR15.7 0 - Resource function is **OCU7** output
1 - Resource function is **TOT7** output

PFR15.6 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

EPFR15.6 0 - Resource function is **OCU6** output
1 - Resource function is **TOT6** output

PFR15.5 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

EPFR15.5 0 - Resource function is **OCU5** output
1 - Resource function is **TOT5** output

PFR15.4 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

EPFR15.4 0 - Resource function is **OCU4** output
1 - Resource function is **TOT4** output

PFR15.3 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

EPFR15.3 0 - Resource function is **OCU3** output
1 - Resource function is **TOT3** output

PFR15.2 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

EPFR15.2 0 - Resource function is **OCU2** output
1 - Resource function is **TOT2** output

PFR15.1 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

EPFR15.1 0 - Resource function is **OCU1** output
1 - Resource function is **TOT1** output

PFR15.0 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

EPFR15.0 0 - Resource function is **OCU0** output
1 - Resource function is **TOT0** output

■ **P16:** The functions of Port 16 are controlled by PFR16 and EPFR16

	Addr	7	6	5	4	3	2	1	0	initial
PFR16	0D90h	PFR16.7	PFR16.6	PFR16.5	PFR16.4	PFR16.3	PFR16.2	PFR16.1	PFR16.0	0000 0000
EPFR16	0DD0h	EPFR16.7	EPFR16.6	EPFR16.5	EPFR16.4	-	-	-	-	0000 ----
		R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	

P16[7:0] is input/output for Programmable Pulse Generator outputs **PPG[15:8]**, external ADC trigger **ATGX**, Pulse Frequency Modulator output **PFM**, and Sound Generator outputs **SGO/SGA**. Otherwise, the port can be used as general purpose port.

PFR16.7 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

EPFR16.7 0 - Resource function is **PPG15** output
1 - Resource function is **ATGX** input

PFR16.6 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

EPFR16.6 0 - Resource function is **PPG14** output
1 - Resource function is **PFM** output

PFR16.5 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

EPFR16.5 0 - Resource function is **PPG13** output
1 - Resource function is **SGO** output

PFR16.4 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

EPFR16.4 0 - Resource function is **PPG12** output
1 - Resource function is **SGA** output

PFR16.3 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

Resource function is **PPG11** output

PFR16.2 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

Resource function is **PPG10** output

PFR16.1 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

Resource function is **PPG9** output

PFR16.0 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

Resource function is **PPG8** output

■ **P17:** The functions of Port 17 are controlled by PFR17

	Addr	7	6	5	4	3	2	1	0	initial
PFR17	0D91h	PFR17.7	PFR17.6	PFR17.5	PFR17.4	PFR17.3	PFR17.2	PFR17.1	PFR17.0	0000 0000
EPFR17	0DD1h	-	-	-	-	-	-	-	-	---- ----
		R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	

P17[7:0] is input/output for Programmable Pulse Generator outputs **PPG[15:8]**. Otherwise, the port can be used as general purpose port.

PFR17.7 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

Resource function is **PPG7** output

PFR17.6 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

Resource function is **PPG6** output

PFR17.5 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

Resource function is **PPG5** output

PFR17.4 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

Resource function is **PPG4** output

PFR17.3 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

Resource function is **PPG3** output

PFR17.2 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

Resource function is **PPG2** output

PFR17.1 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

Resource function is **PPG1** output

PFR17.0 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

Resource function is **PPG0** output

■ **P18:** The functions of Port 18 are controlled by PFR18 and EPFR18

	Addr	7	6	5	4	3	2	1	0	initial
PFR18	0D92h	-	PFR18.6	PFR18.5	PFR18.4	-	PFR18.2	PFR18.1	PFR18.0	-000 -000
EPFR18	0DD2h	-	EPFR18.6	EPFR18.5	-	-	EPFR18.2	EPFR18.1	-	-00- -00-
		R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	

P18[7:0] is input/output for LIN-UART serial communication signals **SCK, SOT, SIN** of channels 6 and 7, Up-/Down-Counter inputs **ZIN, BIN, AIN** of channels 2 and 3, and Free Run Timer FRT inputs **CK** of channels 6 and 7. Otherwise, the port can be used as general purpose port.

PFR18.6 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

EPFR18.6 0 - Resource function is **SCK7** input/output
1 - Resource function is **ZIN3** and **CK7** input

PFR18.5 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

EPFR18.5 0 - Resource function is **SOT7** output
1 - Resource function is **BIN3** input

PFR18.4 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

Resource function is **SIN7** and **AIN3** input

PFR18.2 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

EPFR18.2 0 - Resource function is **SCK6** input/output
1 - Resource function is **ZIN2** and **CK6** input

PFR18.1 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

EPFR18.1 0 - Resource function is **SOT6** output
1 - Resource function is **BIN2** input

PFR18.0 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

Resource function is **SIN6** and **AIN2** input

■ **P19:** The functions of Port 19 are controlled by PFR19 and EPFR19

	Addr	7	6	5	4	3	2	1	0	initial
PFR19	0D93h	-	PFR19.6	PFR19.5	PFR19.4	-	PFR19.2	PFR19.1	PFR19.0	-000 -000
EPFR19	0DD3h	-	EPFR19.6	-	-	-	EPFR19.2	-	-	-0-- -0--
		R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	

P19[7:0] is input/output for LIN-UART serial communication signals **SCK, SOT, SIN** of channels 4 and 5, and Free Run Timer FRT inputs **CK** of channels 4 and 5. Otherwise, the port can be used as general purpose port.

PFR19.6 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

EPFR19.6 0 - Resource function is **SCK5** input/output
1 - Resource function is **CK5** input

PFR19.5 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

Resource function is **SOT5** output

PFR19.4 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

Resource function is **SIN5** input

PFR19.2 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

EPFR19.2 0 - Resource function is **SCK4** input/output
1 - Resource function is **CK4** input

PFR19.1 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

Resource function is **SOT4** output

PFR19.0 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

Resource function is **SIN4** input

■ **P20:** The functions of Port 20 are controlled by PFR20 and EPFR20

	Addr	7	6	5	4	3	2	1	0	initial
PFR20	0D94h	-	PFR20.6	PFR20.5	PFR20.4	-	PFR20.2	PFR20.1	PFR20.0	-000 -000
EPFR20	0DD4h	-	EPFR20.6	EPFR20.5	-	-	EPFR20.2	EPFR20.1	-	-00- -00-
		R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	

P20[7:0] is input/output for LIN-UART serial communication signals **SCK, SOT, SIN** of channels 2 and 3, Up-/Down-Counter inputs **ZIN, BIN, AIN** of channels 0 and 1, and Free Run Timer FRT inputs **CK** of channels 2 and 3. Otherwise, the port can be used as general purpose port.

PFR20.6 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

EPFR20.6 0 - Resource function is **SCK3** input/output
1 - Resource function is **ZIN1** and **CK3** input

PFR20.5 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

EPFR20.5 0 - Resource function is **SOT3** output
1 - Resource function is **BIN1** input

PFR20.4 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

Resource function is **SIN3** and **AIN1** input

PFR20.2 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

EPFR20.2 0 - Resource function is **SCK2** input/output
1 - Resource function is **ZIN0** and **CK2** input

PFR20.1 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

EPFR20.1 0 - Resource function is **SOT2** output
1 - Resource function is **BIN0** input

PFR20.0 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

Resource function is **SIN2** and **AIN0** input

■ **P21:** The functions of Port 21 are controlled by PFR21 and EPFR21

	Addr	7	6	5	4	3	2	1	0	initial
PFR21	0D95h	-	PFR21.6	PFR21.5	PFR21.4	-	PFR21.2	PFR21.1	PFR21.0	-000 -000
EPFR21	0DD5h	-	EPFR21.6	-	-	-	EPFR21.2	-	-	-0-- -0--
		R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	

P21[7:0] is input/output for LIN-UART serial communication signals **SCK, SOT, SIN** of channels 0 and 1, and Free Run Timer FRT inputs **CK** of channels 0 and 1. Otherwise, the port can be used as general purpose port.

PFR21.6 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

EPFR21.6 0 - Resource function is **SCK1** input/output
1 - Resource function is **CK1** input

PFR21.5 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

Resource function is **SOT1** output

PFR21.4 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

Resource function is **SIN1** input

PFR21.2 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

EPFR21.2 0 - Resource function is **SCK0** input/output
1 - Resource function is **CK0** input

PFR21.1 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

Resource function is **SOT0** output

PFR21.0 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

Resource function is **SIN0** input

■ **P22:** The functions of Port 22 are controlled by PFR22

	Addr	7	6	5	4	3	2	1	0	initial
PFR22	0D96h	PFR22.7	PFR22.6	PFR22.5	PFR22.4	PFR22.3	PFR22.2	PFR22.1	PFR22.0	0000 0000
EPFR22	0DD6h	-	-	-	-	-	-	-	-	---- ----
		R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	

P22[7:0] is input/output for I2C serial communication signals **SCL, SDA** of channels 0 and 1, CAN serial communication signals **TX, RX** of channels 4 and 5, and External Interrupt Triggers **INT[15:12]**. Otherwise, the port can be used as general purpose port.

PFR22.7 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

Resource function is **SCL1** open drain

PFR22.6 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

Resource function is **SDA1** open drain, and **INT15** input

Remark: This pin supports external interrupt wake up from STOP-HIZ mode. Because of this the internal input line is not forced to low in STOP-HIZ mode if the PFR is set to ‘1’ and interrupt is enabled with ENIR1.EN15 set to ‘1’.

PFR22.5 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

Resource function is **SCL0** open drain

PFR22.4 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

Resource function is **SDA0** open drain, and **INT14** input

Remark: This pin supports external interrupt wake up from STOP-HIZ mode. Because of this the internal input line is not forced to low in STOP-HIZ mode if the PFR is set to ‘1’ and interrupt is enabled with ENIR1.EN14 set to ‘1’.

PFR22.3 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

Resource function is **TX5** output

PFR22.2 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

Resource function is **RX5** input, and **INT13** input

Remark: This pin supports external interrupt wake up from STOP-HIZ mode. Because of this the internal input line is not forced to low in STOP-HIZ mode if the PFR is set to ‘1’ and interrupt is enabled with ENIR1.EN13 set to ‘1’.

PFR22.1 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

Resource function is **TX4** output

PFR22.0 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

Resource function is **RX4** input, and **INT12** input

Remark: This pin supports external interrupt wake up from STOP-HIZ mode. Because of this the internal input line is not forced to low in STOP-HIZ mode if the PFR is set to ‘1’ and interrupt is enabled with ENIR1.EN12 set to ‘1’.

Remark: It is generally possible to use input only resource functions (like e.g. INT, ICU, CAN.RX, UART.SIN) also in the Port I/O input mode (PFR=’0’ and DDR=’0’). In that case the internal input line is forced to low in STOP-HIZ mode.

■ **P23:** The functions of Port 23 are controlled by PFR23

	Addr	7	6	5	4	3	2	1	0	initial
PFR23	0D97h	PFR23.7	PFR23.6	PFR23.5	PFR23.4	PFR23.3	PFR23.2	PFR23.1	PFR23.0	0000 0000
EPFR23	0DD7h	-	-	-	-	-	-	-	-	---- ----
		R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	

P23[7:0] is input/output for CAN serial communication signals **TX, RX** of channels 0 to 3, and External Interrupt Triggers **INT[11:8]**. Otherwise, the port can be used as general purpose port.

PFR23.7 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

Resource function is **TX3** output

PFR23.6 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

Resource function is **RX3** input, and **INT11** input

Remark: This pin supports external interrupt wake up from STOP-HIZ mode. Because of this the internal input line is not forced to low in STOP-HIZ mode if the PFR is set to ‘1’ and interrupt is enabled with ENIR1.EN11 set to ‘1’.

PFR23.5 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

Resource function is **TX2** output

PFR23.4 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

Resource function is **RX2** input, and **INT10** input

Remark: This pin supports external interrupt wake up from STOP-HIZ mode. Because of this the internal input line is not forced to low in STOP-HIZ mode if the PFR is set to ‘1’ and interrupt is enabled with ENIR1.EN10 set to ‘1’.

PFR23.3 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

Resource function is **TX1** output

PFR23.2 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

Resource function is **RX1** input, and **INT9** input

Remark: This pin supports external interrupt wake up from STOP-HIZ mode. Because of this the internal input line is not forced to low in STOP-HIZ mode if the PFR is set to ‘1’ and interrupt is enabled with ENIR1.EN9 set to ‘1’.

PFR23.1 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

Resource function is **TX0** output

PFR23.0 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

Resource function is **RX0** input, and **INT8** input

Remark: This pin supports external interrupt wake up from STOP-HIZ mode. Because of this the internal input line is not forced to low in STOP-HIZ mode if the PFR is set to ‘1’ and interrupt is enabled with ENIR1.EN8 set to ‘1’.

Remark: It is generally possible to use input only resource functions (like e.g. INT, ICU, CAN.RX, UART.SIN) also in the Port I/O input mode (PFR=’0’ and DDR=’0’). In that case the internal input line is forced to low in STOP-HIZ mode.

■ **P24:** The functions of Port 24 are controlled by PFR24

	Addr	7	6	5	4	3	2	1	0	initial
PFR24	0D98h	PFR24.7	PFR24.6	PFR24.5	PFR24.4	PFR24.3	PFR24.2	PFR24.1	PFR24.0	0000 0000
EPFR24	0DD8h	-	-	-	-	-	-	-	-	---- ----
		R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	

P24[7:0] is input/output for I2C serial communication signals **SCL, SDA** of channels 2 and 3, and External Interrupt Triggers **INT[7:0]**. Otherwise, the port can be used as general purpose port.

PFR24.7 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

Resource function is **SCL3** open drain, and **INT7** input

Remark: This pin supports external interrupt wake up from STOP-HIZ mode. Because of this the internal input line is not forced to low in STOP-HIZ mode if the PFR is set to '1' and interrupt is enabled with ENIR0.EN7 set to '1'.

PFR24.6 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

Resource function is **SDA3** open drain, and **INT6** input

Remark: This pin supports external interrupt wake up from STOP-HIZ mode. Because of this the internal input line is not forced to low in STOP-HIZ mode if the PFR is set to '1' and interrupt is enabled with ENIR0.EN6 set to '1'.

PFR24.5 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

Resource function is **SCL2** open drain, and **INT5** input

Remark: This pin supports external interrupt wake up from STOP-HIZ mode. Because of this the internal input line is not forced to low in STOP-HIZ mode if the PFR is set to '1' and interrupt is enabled with ENIR0.EN5 set to '1'.

PFR24.4 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

Resource function is **SDA2** open drain, and **INT4** input

Remark: This pin supports external interrupt wake up from STOP-HIZ mode. Because of this the internal input line is not forced to low in STOP-HIZ mode if the PFR is set to '1' and interrupt is enabled with ENIR0.EN4 set to '1'.

PFR24.3 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

Resource function is **INT3** input

Remark: This pin supports external interrupt wake up from STOP-HIZ mode. Because of this the internal input line is not forced to low in STOP-HIZ mode if the PFR is set to '1' and interrupt is enabled with ENIR0.EN3 set to '1'.

PFR24.2 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

Resource function is **INT2** input

Remark: This pin supports external interrupt wake up from STOP-HIZ mode. Because of this the internal input line is not forced to low in STOP-HIZ mode if the PFR is set to ‘1’ and interrupt is enabled with ENIR0.EN2 set to ‘1’.

PFR24.1 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

Resource function is **INT1** input

Remark: This pin supports external interrupt wake up from STOP-HIZ mode. Because of this the internal input line is not forced to low in STOP-HIZ mode if the PFR is set to ‘1’ and interrupt is enabled with ENIR0.EN1 set to ‘1’.

PFR24.0 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

Resource function is **INT0** input

Remark: This pin supports external interrupt wake up from STOP-HIZ mode. Because of this the internal input line is not forced to low in STOP-HIZ mode if the PFR is set to ‘1’ and interrupt is enabled with ENIR0.EN0 set to ‘1’.

Remark: It is generally possible to use input only resource functions (like e.g. INT, ICU, CAN.RX, UART.SIN) also in the Port I/O input mode (PFR=’0’ and DDR=’0’). In that case the internal input line is forced to low in STOP-HIZ mode.

■ **P25:** The functions of Port 25 are controlled by PFR25

	Addr	7	6	5	4	3	2	1	0	initial
PFR25	0D99h	PFR25.7	PFR25.6	PFR25.5	PFR25.4	PFR25.3	PFR25.2	PFR25.1	PFR25.0	0000 0000
EPFR25	0DD9h	-	-	-	-	-	-	-	-	---- ----
		R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	

P25[7:0] is input/output for Stepper Motor PWM output signals and Comparator Inputs **SMC2M, SMC2P, SMC1M, SMC1P** of channels 4 and 5. Otherwise, the port can be used as general purpose port.

PFR25.7 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

Resource function is **SMC2M5** output, CMP5 input if selected

PFR25.6 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

Resource function is **SMC2P5** output, CMP5 input if selected

PFR25.5 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

Resource function is **SMC1M5** output, CMP5 input if selected

PFR25.4 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

Resource function is **SMC1P5** output, CMP5 input if selected

PFR25.3 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

Resource function is **SMC2M4** output, CMP4 input if selected

PFR25.2 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

Resource function is **SMC2P4** output, CMP4 input if selected

PFR25.1 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

Resource function is **SMC1M4** output, CMP4 input if selected

PFR25.0 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

Resource function is **SMC1P4** output, CMP4 input if selected

■ **P26:** The functions of Port 26 are controlled by PFR26 and EPFR26

	Addr	7	6	5	4	3	2	1	0	initial
PFR26	0D9Ah	PFR26.7	PFR26.6	PFR26.5	PFR26.4	PFR26.3	PFR26.2	PFR26.1	PFR26.0	0000 0000
EPFR26	0DDAh	EPFR26.7	EPFR26.6	EPFR26.5	EPFR26.4	EPFR26.3	EPFR26.2	EPFR26.1	EPFR26.0	0000 0000
		R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	

P26[7:0] is input/output for Stepper Motor PWM output signals and Comparator Inputs **SMC2M, SMC2P, SMC1M, SMC1P** of channels 2 and 3, and A/D converter analogue inputs **AN[31:24]**. Otherwise, the port can be used as general purpose port.

PFR26.7 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

EPFR26.7 0 - Resource function is **SMC2M3** output, CMP3 and/or AN31 input if selected
1 - Resource function is **AN31** input

PFR26.6 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

EPFR26.6 0 - Resource function is **SMC2P3** output, CMP3 and/or AN30 input if selected
1 - Resource function is **AN30** input

PFR26.5 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

EPFR26.5 0 - Resource function is **SMC1M3** output, CMP3 and/or AN29 input if selected
1 - Resource function is **AN29** input

PFR26.4 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

EPFR26.4 0 - Resource function is **SMC1P3** output, CMP3 and/or AN28 input if selected
1 - Resource function is **AN28** input

PFR26.3 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

EPFR26.3 0 - Resource function is **SMC2M2** output, CMP2 and/or AN27 input if selected
1 - Resource function is **AN27** input

PFR26.2 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

EPFR26.2 0 - Resource function is **SMC2P2** output, CMP2 and/or AN26 input if selected
1 - Resource function is **AN26** input

PFR26.1 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

EPFR26.1 0 - Resource function is **SMC1M2** output, CMP2 and/or AN25 input if selected
1 - Resource function is **AN25** input

PFR26.0 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

EPFR26.0 0 - Resource function is **SMC1P2** output, CMP2 and/or AN24 input if selected
1 - Resource function is **AN24** input

■ **P27:** The functions of Port 27 are controlled by PFR27 and EPFR27

	Addr	7	6	5	4	3	2	1	0	initial
PFR27	0D9Bh	PFR27.7	PFR27.6	PFR27.5	PFR27.4	PFR27.3	PFR27.2	PFR27.1	PFR27.0	0000 0000
EPFR27	0DDBh	EPFR27.7	EPFR27.6	EPFR27.5	EPFR27.4	EPFR27.3	EPFR27.2	EPFR27.1	EPFR27.0	0000 0000
		R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	

P27[7:0] is input/output for Stepper Motor PWM output signals and Comparator Inputs **SMC2M, SMC2P, SMC1M, SMC1P** of channels 0 and 1, and A/D converter analogue inputs **AN[23:16]**. Otherwise, the port can be used as general purpose port.

PFR27.7 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

EPFR27.7 0 - Resource function is **SMC2M1** output, CMP1 and/or AN23 input if selected
1 - Resource function is **AN23** input

PFR27.6 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

EPFR27.6 0 - Resource function is **SMC2P1** output, CMP1 and/or AN22 input if selected
1 - Resource function is **AN22** input

PFR27.5 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

EPFR27.5 0 - Resource function is **SMC1M1** output, CMP1 and/or AN21 input if selected
1 - Resource function is **AN21** input

PFR27.4 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

EPFR27.4 0 - Resource function is **SMC1P1** output, CMP1 and/or AN20 input if selected
1 - Resource function is **AN20** input

PFR27.3 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

EPFR27.3 0 - Resource function is **SMC2M0** output, CMP0 and/or AN19 input if selected
1 - Resource function is **AN19** input

PFR27.2 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

EPFR27.2 0 - Resource function is **SMC2P0** output, CMP0 and/or AN18 input if selected
1 - Resource function is **AN18** input

PFR27.1 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

EPFR27.1 0 - Resource function is **SMC1M0** output, CMP0 and/or AN17 input if selected
1 - Resource function is **AN17** input

PFR27.0 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

EPFR27.0 0 - Resource function is **SMC1P0** output, CMP0 and/or AN16 input if selected
1 - Resource function is **AN16** input

■ **P28:** The functions of Port 28 are controlled by PFR28

	Addr	7	6	5	4	3	2	1	0	initial
PFR28	0D9Ch	PFR28.7	PFR28.6	PFR28.5	PFR28.4	PFR28.3	PFR28.2	PFR28.1	PFR28.0	0000 0000
EPFR28	0DDCh	-	-	-	-	-	-	-	-	---- ----
		R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	

P28[7:0] is input/output for A/D converter analogue inputs **AN[15:8]**, and D/A converter analogue outputs **DA[1:0]**. Otherwise, the port can be used as general purpose port.

PFR28.7 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

Resource function is **AN15** input and/or **DA1** output if D/A converter output is enabled

PFR28.6 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

Resource function is **AN14** input and/or **DA0** output if D/A converter output is enabled

PFR28.5 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

Resource function is **AN13** input

PFR28.4 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

Resource function is **AN12** input

PFR28.3 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

Resource function is **AN11** input

PFR28.2 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

Resource function is **AN10** input

PFR28.1 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

Resource function is **AN9** input

PFR28.0 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

Resource function is **AN8** input

■ **P29:** The functions of Port 29 are controlled by PFR29

	Addr	7	6	5	4	3	2	1	0	initial
PFR29	0D9Dh	PFR29.7	PFR29.6	PFR29.5	PFR29.4	PFR29.3	PFR29.2	PFR29.1	PFR29.0	0000 0000
EPFR29	0DDDh	-	-	-	-	-	-	-	-	---- ----
		R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	

P29[7:0] is input/output for A/D converter analogue inputs **AN[7:0]**. Otherwise, the port can be used as general purpose port.

PFR29.7 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

Resource function is **AN7** input

PFR29.6 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

Resource function is **AN6** input

PFR29.5 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

Resource function is **AN5** input

PFR29.4 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

Resource function is **AN4** input

PFR29.3 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

Resource function is **AN3** input

PFR29.2 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

Resource function is **AN2** input

PFR29.1 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

Resource function is **AN1** input

PFR29.0 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

Resource function is **AN0** input

■ **P30:** The functions of Port 30 are controlled by PFR30

	Addr	7	6	5	4	3	2	1	0	initial
PFR30	0D9Eh	PFR30.7	PFR30.6	PFR30.5	PFR30.4	PFR30.3	PFR30.2	PFR30.1	PFR30.0	0000 0000
EPFR30	0DDEh	-	-	-	-	-	-	-	-	---- ----
		R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	

P30[7:0] is input/output for LCD controller reference voltage analogue inputs **V[3:0]**, and LCD controller common driver outputs **COM[3:0]**. Otherwise, the port can be used as general purpose port.

PFR30.7 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

Resource function is **V3** input

PFR30.6 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

Resource function is **V2** input

PFR30.5 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

Resource function is **V1** input

PFR30.4 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

Resource function is **V0** input

PFR30.3 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

Resource function is **COM3** output

PFR30.2 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

Resource function is **COM2** output

PFR30.1 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

Resource function is **COM1** output

PFR30.0 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

Resource function is **COM0** output

■ **P31:** The functions of Port 31 are controlled by PFR31

	Addr	7	6	5	4	3	2	1	0	initial
PFR31	0D9Fh	PFR31.7	PFR31.6	PFR31.5	PFR31.4	PFR31.3	PFR31.2	PFR31.1	PFR31.0	0000 0000
EPFR31	0DDFh	-	-	-	-	-	-	-	-	---- ----
		R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	

P31[7:0] is input/output for LCD controller segment driver outputs **SEG[39:32]**. Otherwise, the port can be used as general purpose port.

PFR31.7 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

Resource function is **SEG39** output

PFR31.6 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

Resource function is **SEG38** output

PFR31.5 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

Resource function is **SEG37** output

PFR31.4 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

Resource function is **SEG36** output

PFR31.3 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

Resource function is **SEG35** output

PFR31.2 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

Resource function is **SEG34** output

PFR31.1 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

Resource function is **SEG33** output

PFR31.0 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

Resource function is **SEG32** output

■ **P32:** The functions of Port 32 are controlled by PFR32 and EPFR32

	Addr	7	6	5	4	3	2	1	0	initial
PFR32	0DA0h	PFR32.7	PFR32.6	PFR32.5	PFR32.4	PFR32.3	PFR32.2	PFR32.1	PFR32.0	0000 0000
EPFR32	0DE0h	-	EPFR32.6	EPFR32.5	EPFR32.4	-	EPFR32.2	EPFR32.1	EPFR32.0	-000 -000
		R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	

P32[7:0] is input/output for LCD controller segment driver outputs **SEG[31:24]**, and LIN-UART serial communication signals **SCK, SOT, SIN** of channels 14 and 15. Otherwise, the port can be used as general purpose port.

PFR32.7 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

Resource function is **SEG31** output

PFR32.6 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

EPFR32.6 0 - Resource function is **SEG30** output
1 - Resource function is **SCK15** input/output

PFR32.5 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

EPFR32.5 0 - Resource function is **SEG29** output
1 - Resource function is **SOT15** output

PFR32.4 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

EPFR32.4 0 - Resource function is **SEG28** output
1 - Resource function is **SIN15** input

PFR32.3 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

Resource function is **SEG27** output

PFR32.2 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

EPFR32.2 0 - Resource function is **SEG26** output
1 - Resource function is **SCK14** input/output

PFR32.1 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

EPFR32.1 0 - Resource function is **SEG25** output
1 - Resource function is **SOT14** output

PFR32.0 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

EPFR32.0 0 - Resource function is **SEG24**output
1 - Resource function is **SIN14** input

■ **P33:** The functions of Port 33 are controlled by PFR33 and EPFR33

	Addr	7	6	5	4	3	2	1	0	initial
PFR33	0DA1h	PFR33.7	PFR33.6	PFR33.5	PFR33.4	PFR33.3	PFR33.2	PFR33.1	PFR33.0	0000 0000
EPFR33	0DE1h	-	EPFR33.6	EPFR33.5	EPFR33.4	-	EPFR33.2	EPFR33.1	EPFR33.0	-000 -000
		R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	

P33[7:0] is input/output for LCD controller segment driver outputs **SEG[23:16]**, and LIN-UART serial communication signals **SCK, SOT, SIN** of channels 12 and 13. Otherwise, the port can be used as general purpose port.

PFR33.7 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

Resource function is **SEG23** output

PFR33.6 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

EPFR33.6 0 - Resource function is **SEG22** output
1 - Resource function is **SCK13** input/output

PFR33.5 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

EPFR33.5 0 - Resource function is **SEG21** output
1 - Resource function is **SOT13** output

PFR33.4 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

EPFR33.4 0 - Resource function is **SEG20** output
1 - Resource function is **SIN13** input

PFR33.3 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

Resource function is **SEG19** output

PFR33.2 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

EPFR33.2 0 - Resource function is **SEG18** output
1 - Resource function is **SCK12** input/output

PFR33.1 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

EPFR33.1 0 - Resource function is **SEG17** output
1 - Resource function is **SOT12** output

PFR33.0 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

EPFR33.0 0 - Resource function is **SEG16** output
1 - Resource function is **SIN12** input

■ **P34:** The functions of Port 34 are controlled by PFR34 and EPFR34

	Addr	7	6	5	4	3	2	1	0	initial
PFR34	0DA2h	PFR34.7	PFR34.6	PFR34.5	PFR34.4	PFR34.3	PFR34.2	PFR34.1	PFR34.0	0000 0000
EPFR34	0DE2h	-	EPFR34.6	EPFR34.5	EPFR34.4	-	EPFR34.2	EPFR34.1	EPFR34.0	-000 -000
		R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	

P34[7:0] is input/output for LCD controller segment driver outputs **SEG[15:8]**, and LIN-UART serial communication signals **SCK, SOT, SIN** of channels 10 and 11. Otherwise, the port can be used as general purpose port.

PFR34.7 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

Resource function is **SEG15** output

PFR34.6 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

EPFR34.6 0 - Resource function is **SEG14** output
1 - Resource function is **SCK11** input/output

PFR34.5 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

EPFR34.5 0 - Resource function is **SEG13** output
1 - Resource function is **SOT11** output

PFR34.4 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

EPFR34.4 0 - Resource function is **SEG12** output
1 - Resource function is **SIN11** input

PFR34.3 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

Resource function is **SEG11** output

PFR34.2 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

EPFR34.2 0 - Resource function is **SEG10** output
1 - Resource function is **SCK10** input/output

PFR34.1 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

EPFR34.1 0 - Resource function is **SEG9** output
1 - Resource function is **SOT10** output

PFR34.0 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

EPFR34.0 0 - Resource function is **SEG8** output
1 - Resource function is **SIN10** input

■ **P35:** The functions of Port 35 are controlled by PFR35 and EPFR35

	Addr	7	6	5	4	3	2	1	0	initial
PFR35	0DA3h	PFR35.7	PFR35.6	PFR35.5	PFR35.4	PFR35.3	PFR35.2	PFR35.1	PFR35.0	0000 0000
EPFR35	0DE3h	-	EPFR35.6	EPFR35.5	EPFR35.4	-	EPFR35.2	EPFR35.1	EPFR35.0	-000 -000
		R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	

P35[7:0] is input/output for LCD controller segment driver outputs **SEG[7:0]**, and LIN-UART serial communication signals **SCK, SOT, SIN** of channels 8 and 9. Otherwise, the port can be used as general purpose port.

PFR35.7 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

Resource function is **SEG7** output

PFR35.6 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

EPFR35.6 0 - Resource function is **SEG6** output
1 - Resource function is **SCK9** input/output

PFR35.5 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

EPFR35.5 0 - Resource function is **SEG5** output
1 - Resource function is **SOT9** output

PFR35.4 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

EPFR35.4 0 - Resource function is **SEG4** output
1 - Resource function is **SIN9** input

PFR35.3 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

Resource function is **SEG3** output

PFR35.2 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

EPFR35.2 0 - Resource function is **SEG2** output
1 - Resource function is **SCK8** input/output

PFR35.1 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

EPFR35.1 0 - Resource function is **SEG1** output
1 - Resource function is **SOT8** output

PFR35.0 0 - Port is in general purpose port mode.
1 - Port is in resource function mode:

EPFR35.0 0 - Resource function is **SEG0** output
1 - Resource function is **SIN8** input

3.5 Port Input Level Selection

The input levels of each port can be programmed bit-wise between CMOS Hysteresis type A and B, Automotive Hysteresis and TTL level.

CMOS Hysteresis type A:	VIL=0.3 x VDD	VIH=0.7 x VDD
CMOS Hysteresis type B:	VIL=0.2 x VDD	VIH=0.8 x VDD
Automotive Hysteresis:	VIL=0.5 x VDD	VIH=0.8 x VDD
TTL	VIL=0.8 V	VIH=2.1 V

For setup, the Port Input Level Registers (PILR, EPILR) of each port are used.

Table 3-1 Settings of the MB91Fxxx (GP00 - GP35)

PILRx.y	EPILRx.y	Port Input Level
0 (default)	0 (default)	**CMOS Hysteresis A**
1	0	**Automotive Hysteresis**
0	1	**TTL**
1	1	**CMOS Hysteresis B**

Table 3-2 Settings of the MB91V460A (GP00 - GP13)

PILRx.y	EPILRx.y	Port Input Level
0 (default)	0 (default)	**CMOS Hysteresis A**
1	0	**Automotive Hysteresis**
0	1	**TTL**
1	1	**n.a.**

Table 3-3 Settings of the MB91V460A (GP14 - GP35))

PILRx.y	EPILRx.y	Port Input Level
0 (default)	0 (default)	**CMOS Hysteresis A**
1	0	**Automotive Hysteresis**
0	1	**n.a.**
1	1	**n.a.**

EPILR14-EPILR35 is not implemented.

	Addr	7	6	5	4	3	2	1	0	initial
PILR00	0E40h	PILR00.7	PILR00.6	PILR00.5	PILR00.4	PILR00.3	PILR00.2	PILR00.1	PILR00.0	0000 0000
PILR01	0E41h	PILR01.7	PILR01.6	PILR01.5	PILR01.4	PILR01.3	PILR01.2	PILR01.1	PILR01.0	0000 0000
PILR02	0E42h	PILR02.7	PILR02.6	PILR02.5	PILR02.4	PILR02.3	PILR02.2	PILR02.1	PILR02.0	0000 0000
PILR03	0E43h	PILR03.7	PILR03.6	PILR03.5	PILR03.4	PILR03.3	PILR03.2	PILR03.1	PILR03.0	0000 0000

PILR04	0E44h	PILR04.7	PILR04.6	PILR04.5	PILR04.4	PILR04.3	PILR04.2	PILR04.1	PILR04.0	0000 0000
PILR05	0E45h	PILR05.7	PILR05.6	PILR05.5	PILR05.4	PILR05.3	PILR05.2	PILR05.1	PILR05.0	0000 0000
PILR06	0E46h	PILR06.7	PILR06.6	PILR06.5	PILR06.4	PILR06.3	PILR06.2	PILR06.1	PILR06.0	0000 0000
PILR07	0E47h	PILR07.7	PILR07.6	PILR07.5	PILR07.4	PILR07.3	PILR07.2	PILR07.1	PILR07.0	0000 0000
PILR08	0E48h	PILR08.7	PILR08.6	PILR08.5	PILR08.4	PILR08.3	PILR08.2	PILR08.1	PILR08.0	0000 0000
PILR09	0E49h	PILR09.7	PILR09.6	PILR09.5	PILR09.4	PILR09.3	PILR09.2	PILR09.1	PILR09.0	0000 0000
PILR10	0E4Ah	PILR10.7	PILR10.6	PILR10.5	PILR10.4	PILR10.3	PILR10.2	PILR10.1	PILR10.0	0000 0000
PILR11	0E4Bh	PILR11.7	PILR11.6	PILR11.5	PILR11.4	PILR11.3	PILR11.2	PILR11.1	PILR11.0	0000 0000
PILR12	0E4Ch	PILR12.7	PILR12.6	PILR12.5	PILR12.4	PILR12.3	PILR12.2	PILR12.1	PILR12.0	0000 0000
PILR13	0E4Dh	PILR13.7	PILR13.6	PILR13.5	PILR13.4	PILR13.3	PILR13.2	PILR13.1	PILR13.0	0000 0000
PILR14	0E4Eh	PILR14.7	PILR14.6	PILR14.5	PILR14.4	PILR14.3	PILR14.2	PILR14.1	PILR14.0	0000 0000
PILR15	0E4Fh	PILR15.7	PILR15.6	PILR15.5	PILR15.4	PILR15.3	PILR15.2	PILR15.1	PILR15.0	0000 0000
PILR16	0E50h	PILR16.7	PILR16.6	PILR16.5	PILR16.4	PILR16.3	PILR16.2	PILR16.1	PILR16.0	0000 0000
PILR17	0E51h	PILR17.7	PILR17.6	PILR17.5	PILR17.4	PILR17.3	PILR17.2	PILR17.1	PILR17.0	0000 0000
PILR18	0E52h	PILR18.7	PILR18.6	PILR18.5	PILR18.4	PILR18.3	PILR18.2	PILR18.1	PILR18.0	0000 0000
PILR19	0E53h	PILR19.7	PILR19.6	PILR19.5	PILR19.4	PILR19.3	PILR19.2	PILR19.1	PILR19.0	0000 0000
PILR20	0E54h	PILR20.7	PILR20.6	PILR20.5	PILR20.4	PILR20.3	PILR20.2	PILR20.1	PILR20.0	0000 0000
PILR21	0E55h	PILR21.7	PILR21.6	PILR21.5	PILR21.4	PILR21.3	PILR21.2	PILR21.1	PILR21.0	0000 0000
PILR22	0E56h	PILR22.7	PILR22.6	PILR22.5	PILR22.4	PILR22.3	PILR22.2	PILR22.1	PILR22.0	0000 0000
PILR23	0E57h	PILR23.7	PILR23.6	PILR23.5	PILR23.4	PILR23.3	PILR23.2	PILR23.1	PILR23.0	0000 0000
PILR24	0E58h	PILR24.7	PILR24.6	PILR24.5	PILR24.4	PILR24.3	PILR24.2	PILR24.1	PILR24.0	0000 0000
PILR25	0E59h	PILR25.7	PILR25.6	PILR25.5	PILR25.4	PILR25.3	PILR25.2	PILR25.1	PILR25.0	0000 0000
PILR26	0E5Ah	PILR26.7	PILR26.6	PILR26.5	PILR26.4	PILR26.3	PILR26.2	PILR26.1	PILR26.0	0000 0000
PILR27	0E5Bh	PILR27.7	PILR27.6	PILR27.5	PILR27.4	PILR27.3	PILR27.2	PILR27.1	PILR27.0	0000 0000
PILR28	0E5Ch	PILR28.7	PILR28.6	PILR28.5	PILR28.4	PILR28.3	PILR28.2	PILR28.1	PILR28.0	0000 0000
PILR29	0E5Dh	PILR29.7	PILR29.6	PILR29.5	PILR29.4	PILR29.3	PILR29.2	PILR29.1	PILR29.0	0000 0000
PILR30	0E5Eh	PILR30.7	PILR30.6	PILR30.5	PILR30.4	PILR30.3	PILR30.2	PILR30.1	PILR30.0	0000 0000
PILR31	0E5Fh	PILR31.7	PILR31.6	PILR31.5	PILR31.4	PILR31.3	PILR31.2	PILR31.1	PILR31.0	0000 0000
PILR32	0E60h	PILR32.7	PILR32.6	PILR32.5	PILR32.4	PILR32.3	PILR32.2	PILR32.1	PILR32.0	0000 0000
PILR33	0E61h	PILR33.7	PILR33.6	PILR33.5	PILR33.4	PILR33.3	PILR33.2	PILR33.1	PILR33.0	0000 0000
PILR34	0E62h	PILR34.7	PILR34.6	PILR34.5	PILR34.4	PILR34.3	PILR34.2	PILR34.1	PILR34.0	0000 0000
PILR35	0E63h	PILR35.7	PILR35.6	PILR35.5	PILR35.4	PILR35.3	PILR35.2	PILR35.1	PILR35.0	0000 0000
		R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	

	Addr	7	6	5	4	3	2	1	0	initial
EPILR00	0E80h	EPILR00.7	EPILR00.6	EPILR00.5	EPILR00.4	EPILR00.3	EPILR00.2	EPILR00.1	EPILR00.0	0000 0000
EPILR01	0E81h	EPILR01.7	EPILR01.6	EPILR01.5	EPILR01.4	EPILR01.3	EPILR01.2	EPILR01.1	EPILR01.0	0000 0000
EPILR02	0E82h	EPILR02.7	EPILR02.6	EPILR02.5	EPILR02.4	EPILR02.3	EPILR02.2	EPILR02.1	EPILR02.0	0000 0000
EPILR03	0E83h	EPILR03.7	EPILR03.6	EPILR03.5	EPILR03.4	EPILR03.3	EPILR03.2	EPILR03.1	EPILR03.0	0000 0000
EPILR04	0E84h	EPILR04.7	EPILR04.6	EPILR04.5	EPILR04.4	EPILR04.3	EPILR04.2	EPILR04.1	EPILR04.0	0000 0000
EPILR05	0E85h	EPILR05.7	EPILR05.6	EPILR05.5	EPILR05.4	EPILR05.3	EPILR05.2	EPILR05.1	EPILR05.0	0000 0000
EPILR06	0E86h	EPILR06.7	EPILR06.6	EPILR06.5	EPILR06.4	EPILR06.3	EPILR06.2	EPILR06.1	EPILR06.0	0000 0000
EPILR07	0E87h	EPILR07.7	EPILR07.6	EPILR07.5	EPILR07.4	EPILR07.3	EPILR07.2	EPILR07.1	EPILR07.0	0000 0000
EPILR08	0E88h	EPILR08.7	EPILR08.6	EPILR08.5	EPILR08.4	EPILR08.3	EPILR08.2	EPILR08.1	EPILR08.0	0000 0000
EPILR09	0E89h	EPILR09.7	EPILR09.6	EPILR09.5	EPILR09.4	EPILR09.3	EPILR09.2	EPILR09.1	EPILR09.0	0000 0000
EPILR10	0E8Ah	EPILR10.7	EPILR10.6	EPILR10.5	EPILR10.4	EPILR10.3	EPILR10.2	EPILR10.1	EPILR10.0	0000 0000
EPILR11	0E8Bh	EPILR11.7	EPILR11.6	EPILR11.5	EPILR11.4	EPILR11.3	EPILR11.2	EPILR11.1	EPILR11.0	0000 0000
EPILR12	0E8Ch	EPILR12.7	EPILR12.6	EPILR12.5	EPILR12.4	EPILR12.3	EPILR12.2	EPILR12.1	EPILR12.0	0000 0000
EPILR13	0E8Dh	EPILR13.7	EPILR13.6	EPILR13.5	EPILR13.4	EPILR13.3	EPILR13.2	EPILR13.1	EPILR13.0	0000 0000
EPILR14	0E8Eh	-	-	-	-	-	-	-	-	---- ----
EPILR15	0E8Fh	-	-	-	-	-	-	-	-	---- ----
EPILR16	0E90h	-	-	-	-	-	-	-	-	---- ----
EPILR17	0E91h	-	-	-	-	-	-	-	-	---- ----
EPILR18	0E92h	-	-	-	-	-	-	-	-	---- ----
EPILR19	0E93h	-	-	-	-	-	-	-	-	---- ----
EPILR20	0E94h	-	-	-	-	-	-	-	-	---- ----
EPILR21	0E95h	-	-	-	-	-	-	-	-	---- ----
EPILR22	0E96h	-	-	-	-	-	-	-	-	---- ----
EPILR23	0E97h	-	-	-	-	-	-	-	-	---- ----
EPILR24	0E98h	-	-	-	-	-	-	-	-	---- ----
EPILR25	0E99h	-	-	-	-	-	-	-	-	---- ----
EPILR26	0E9Ah	-	-	-	-	-	-	-	-	---- ----
EPILR27	0E9Bh	-	-	-	-	-	-	-	-	---- ----
EPILR28	0E9Ch	-	-	-	-	-	-	-	-	---- ----
EPILR29	0E9Dh	-	-	-	-	-	-	-	-	---- ----
EPILR30	0E9Eh	-	-	-	-	-	-	-	-	---- ----
EPILR31	0E9Fh	-	-	-	-	-	-	-	-	---- ----
EPILR32	0EA0h	-	-	-	-	-	-	-	-	---- ----
EPILR33	0EA1h	-	-	-	-	-	-	-	-	---- ----

EPILR34	0EA2h	-	-	-	-	-	-	-	-	---- ----
EPILR35	0EA3h	-	-	-	-	-	-	-	-	---- ----
		R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	

3.6 Programmable Pull-Up/Pull Down Resistors

The Ports listed in the following table have 50 kOhm Pull-Up resistors, which can be enabled bit-wise. The function is enabled by the Port Pull Enable Registers (PPER) and controlled by the Port Pull Control Register (PPCR). The PPCR selects Pull-Up or Pull-Down. The Pull-Up/Pull-Down are disabled automatically in the STOP-HiZ mode (STCR:STOP and STCR:HIZ set).
For 91F467D the PU/PD are NOT disabled automatically by hardware.

Bit	**Port Pull-Up/Pull-Down Enable Registers**	
	0 (default)	**1**
PPERx.y	Pull-Up/Pull-Down disabled	Pull-Up/Pull-Down enabled

	Addr	7	6	5	4	3	2	1	0	initial
PPER00	0EC0h	PPER00.7	PPER00.6	PPER00.5	PPER00.4	PPER00.3	PPER00.2	PPER00.1	PPER00.0	0000 0000
PPER01	0EC1h	PPER01.7	PPER01.6	PPER01.5	PPER01.4	PPER01.3	PPER01.2	PPER01.1	PPER01.0	0000 0000
PPER02	0EC2h	PPER02.7	PPER02.6	PPER02.5	PPER02.4	PPER02.3	PPER02.2	PPER02.1	PPER02.0	0000 0000
PPER03	0EC3h	PPER03.7	PPER03.6	PPER03.5	PPER03.4	PPER03.3	PPER03.2	PPER03.1	PPER03.0	0000 0000
PPER04	0EC4h	PPER04.7	PPER04.6	PPER04.5	PPER04.4	PPER04.3	PPER04.2	PPER04.1	PPER04.0	0000 0000
PPER05	0EC5h	PPER05.7	PPER05.6	PPER05.5	PPER05.4	PPER05.3	PPER05.2	PPER05.1	PPER05.0	0000 0000
PPER06	0EC6h	PPER06.7	PPER06.6	PPER06.5	PPER06.4	PPER06.3	PPER06.2	PPER06.1	PPER06.0	0000 0000
PPER07	0EC7h	PPER07.7	PPER07.6	PPER07.5	PPER07.4	PPER07.3	PPER07.2	PPER07.1	PPER07.0	0000 0000
PPER08	0EC8h	PPER08.7	PPER08.6	PPER08.5	PPER08.4	PPER08.3	PPER08.2	PPER08.1	PPER08.0	0000 0000
PPER09	0EC9h	PPER09.7	PPER09.6	PPER09.5	PPER09.4	PPER09.3	PPER09.2	PPER09.1	PPER09.0	0000 0000
PPER10	0ECAh	PPER10.7	PPER10.6	PPER10.5	PPER10.4	PPER10.3	PPER10.2	PPER10.1	PPER10.0	0000 0000
PPER11	0ECBh	PPER11.7	PPER11.6	PPER11.5	PPER11.4	PPER11.3	PPER11.2	PPER11.1	PPER11.0	0000 0000
PPER12	0ECCh	PPER12.7	PPER12.6	PPER12.5	PPER12.4	PPER12.3	PPER12.2	PPER12.1	PPER12.0	0000 0000
PPER13	0ECDh	PPER13.7	PPER13.6	PPER13.5	PPER13.4	PPER13.3	PPER13.2	PPER13.1	PPER13.0	0000 0000
PPER14	0ECEh	PPER14.7	PPER14.6	PPER14.5	PPER14.4	PPER14.3	PPER14.2	PPER14.1	PPER14.0	0000 0000
PPER15	0ECFh	PPER15.7	PPER15.6	PPER15.5	PPER15.4	PPER15.3	PPER15.2	PPER15.1	PPER15.0	0000 0000
PPER16	0ED0h	PPER16.7	PPER16.6	PPER16.5	PPER16.4	PPER16.3	PPER16.2	PPER16.1	PPER16.0	0000 0000
PPER17	0ED1h	PPER17.7	PPER17.6	PPER17.5	PPER17.4	PPER17.3	PPER17.2	PPER17.1	PPER17.0	0000 0000
PPER18	0ED2h	PPER18.7	PPER18.6	PPER18.5	PPER18.4	PPER18.3	PPER18.2	PPER18.1	PPER18.0	0000 0000
PPER19	0ED3h	PPER19.7	PPER19.6	PPER19.5	PPER19.4	PPER19.3	PPER19.2	PPER19.1	PPER19.0	0000 0000
PPER20	0ED4h	PPER20.7	PPER20.6	PPER20.5	PPER20.4	PPER20.3	PPER20.2	PPER20.1	PPER20.0	0000 0000

PPER21	0ED5h	PPER21.7	PPER21.6	PPER21.5	PPER21.4	PPER21.3	PPER21.2	PPER21.1	PPER21.0	0000 0000
PPER22	0ED6h	PPER22.7	PPER22.6	PPER22.5	PPER22.4	PPER22.3	PPER22.2	PPER22.1	PPER22.0	0000 0000
PPER23	0ED7h	PPER23.7	PPER23.6	PPER23.5	PPER23.4	PPER23.3	PPER23.2	PPER23.1	PPER23.0	0000 0000
PPER24	0ED8h	PPER24.7	PPER24.6	PPER24.5	PPER24.4	PPER24.3	PPER24.2	PPER24.1	PPER24.0	0000 0000
PPER25	0ED9h	PPER25.7	PPER25.6	PPER25.5	PPER25.4	PPER25.3	PPER25.2	PPER25.1	PPER25.0	0000 0000
PPER26	0EDAh	PPER26.7	PPER26.6	PPER26.5	PPER26.4	PPER26.3	PPER26.2	PPER26.1	PPER26.0	0000 0000
PPER27	0EDBh	PPER27.7	PPER27.6	PPER27.5	PPER27.4	PPER27.3	PPER27.2	PPER27.1	PPER27.0	0000 0000
PPER28	0EDCh	PPER28.7	PPER28.6	PPER28.5	PPER28.4	PPER28.3	PPER28.2	PPER28.1	PPER28.0	0000 0000
PPER29	0EDDh	PPER29.7	PPER29.6	PPER29.5	PPER29.4	PPER29.3	PPER29.2	PPER29.1	PPER29.0	0000 0000
PPER30	0EDEh	PPER30.7	PPER30.6	PPER30.5	PPER30.4	PPER30.3	PPER30.2	PPER30.1	PPER30.0	0000 0000
PPER31	0EDFh	PPER31.7	PPER31.6	PPER31.5	PPER31.4	PPER31.3	PPER31.2	PPER31.1	PPER31.0	0000 0000
PPER32	0EE0h	PPER32.7	PPER32.6	PPER32.5	PPER32.4	PPER32.3	PPER32.2	PPER32.1	PPER32.0	0000 0000
PPER33	0EE1h	PPER33.7	PPER33.6	PPER33.5	PPER33.4	PPER33.3	PPER33.2	PPER33.1	PPER33.0	0000 0000
PPER34	0EE2h	PPER34.7	PPER34.6	PPER34.5	PPER34.4	PPER34.3	PPER34.2	PPER34.1	PPER34.0	0000 0000
PPER35	0EE3h	PPER35.7	PPER35.6	PPER35.5	PPER35.4	PPER35.3	PPER35.2	PPER35.1	PPER35.0	0000 0000
		R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	

Bit	**Port Pull-Up/Pull-Down Control Registers**	
	0	**1 (default)**
PPCRx.y	Pull Down is selected	Pull-Up is selected

	Addr	7	6	5	4	3	2	1	0	initial
PPCR00	0F00h	PPCR00.7	PPCR00.6	PPCR00.5	PPCR00.4	PPCR00.3	PPCR00.2	PPCR00.1	PPCR00.0	1111 1111
PPCR01	0F01h	PPCR01.7	PPCR01.6	PPCR01.5	PPCR01.4	PPCR01.3	PPCR01.2	PPCR01.1	PPCR01.0	1111 1111
PPCR02	0F02h	PPCR02.7	PPCR02.6	PPCR02.5	PPCR02.4	PPCR02.3	PPCR02.2	PPCR02.1	PPCR02.0	1111 1111
PPCR03	0F03h	PPCR03.7	PPCR03.6	PPCR03.5	PPCR03.4	PPCR03.3	PPCR03.2	PPCR03.1	PPCR03.0	1111 1111
PPCR04	0F04h	PPCR04.7	PPCR04.6	PPCR04.5	PPCR04.4	PPCR04.3	PPCR04.2	PPCR04.1	PPCR04.0	1111 1111
PPCR05	0F05h	PPCR05.7	PPCR05.6	PPCR05.5	PPCR05.4	PPCR05.3	PPCR05.2	PPCR05.1	PPCR05.0	1111 1111
PPCR06	0F06h	PPCR06.7	PPCR06.6	PPCR06.5	PPCR06.4	PPCR06.3	PPCR06.2	PPCR06.1	PPCR06.0	1111 1111
PPCR07	0F07h	PPCR07.7	PPCR07.6	PPCR07.5	PPCR07.4	PPCR07.3	PPCR07.2	PPCR07.1	PPCR07.0	1111 1111
PPCR08	0F08h	PPCR08.7	PPCR08.6	PPCR08.5	PPCR08.4	PPCR08.3	PPCR08.2	PPCR08.1	PPCR08.0	1111 1111
PPCR09	0F09h	PPCR09.7	PPCR09.6	PPCR09.5	PPCR09.4	PPCR09.3	PPCR09.2	PPCR09.1	PPCR09.0	1111 1111
PPCR10	0F0Ah	PPCR10.7	PPCR10.6	PPCR10.5	PPCR10.4	PPCR10.3	PPCR10.2	PPCR10.1	PPCR10.0	1111 1111
PPCR11	0F0Bh	PPCR11.7	PPCR11.6	PPCR11.5	PPCR11.4	PPCR11.3	PPCR11.2	PPCR11.1	PPCR11.0	1111 1111

PPCR12	0F0Ch	PPCR12.7	PPCR12.6	PPCR12.5	PPCR12.4	PPCR12.3	PPCR12.2	PPCR12.1	PPCR12.0	1111 1111
PPCR13	0F0Dh	PPCR13.7	PPCR13.6	PPCR13.5	PPCR13.4	PPCR13.3	PPCR13.2	PPCR13.1	PPCR13.0	1111 1111
PPCR14	0F0Eh	PPCR14.7	PPCR14.6	PPCR14.5	PPCR14.4	PPCR14.3	PPCR14.2	PPCR14.1	PPCR14.0	1111 1111
PPCR15	0F0Fh	PPCR15.7	PPCR15.6	PPCR15.5	PPCR15.4	PPCR15.3	PPCR15.2	PPCR15.1	PPCR15.0	1111 1111
PPCR16	0F10h	PPCR16.7	PPCR16.6	PPCR16.5	PPCR16.4	PPCR16.3	PPCR16.2	PPCR16.1	PPCR16.0	1111 1111
PPCR17	0F11h	PPCR17.7	PPCR17.6	PPCR17.5	PPCR17.4	PPCR17.3	PPCR17.2	PPCR17.1	PPCR17.0	1111 1111
PPCR18	0F12h	PPCR18.7	PPCR18.6	PPCR18.5	PPCR18.4	PPCR18.3	PPCR18.2	PPCR18.1	PPCR18.0	1111 1111
PPCR19	0F13h	PPCR19.7	PPCR19.6	PPCR19.5	PPCR19.4	PPCR19.3	PPCR19.2	PPCR19.1	PPCR19.0	1111 1111
PPCR20	0F14h	PPCR20.7	PPCR20.6	PPCR20.5	PPCR20.4	PPCR20.3	PPCR20.2	PPCR20.1	PPCR20.0	1111 1111
PPCR21	0F15h	PPCR21.7	PPCR21.6	PPCR21.5	PPCR21.4	PPCR21.3	PPCR21.2	PPCR21.1	PPCR21.0	1111 1111
PPCR22	0F16h	PPCR22.7	PPCR22.6	PPCR22.5	PPCR22.4	PPCR22.3	PPCR22.2	PPCR22.1	PPCR22.0	1111 1111
PPCR23	0F17h	PPCR23.7	PPCR23.6	PPCR23.5	PPCR23.4	PPCR23.3	PPCR23.2	PPCR23.1	PPCR23.0	1111 1111
PPCR24	0F18h	PPCR24.7	PPCR24.6	PPCR24.5	PPCR24.4	PPCR24.3	PPCR24.2	PPCR24.1	PPCR24.0	1111 1111
PPCR25	0F19h	PPCR25.7	PPCR25.6	PPCR25.5	PPCR25.4	PPCR25.3	PPCR25.2	PPCR25.1	PPCR25.0	1111 1111
PPCR26	0F1Ah	PPCR26.7	PPCR26.6	PPCR26.5	PPCR26.4	PPCR26.3	PPCR26.2	PPCR26.1	PPCR26.0	1111 1111
PPCR27	0F1Bh	PPCR27.7	PPCR27.6	PPCR27.5	PPCR27.4	PPCR27.3	PPCR27.2	PPCR27.1	PPCR27.0	1111 1111
PPCR28	0F1Ch	PPCR28.7	PPCR28.6	PPCR28.5	PPCR28.4	PPCR28.3	PPCR28.2	PPCR28.1	PPCR28.0	1111 1111
PPCR29	0F1Dh	PPCR29.7	PPCR29.6	PPCR29.5	PPCR29.4	PPCR29.3	PPCR29.2	PPCR29.1	PPCR29.0	1111 1111
PPCR30	0F1Eh	PPCR30.7	PPCR30.6	PPCR30.5	PPCR30.4	PPCR30.3	PPCR30.2	PPCR30.1	PPCR30.0	1111 1111
PPCR31	0F1Fh	PPCR31.7	PPCR31.6	PPCR31.5	PPCR31.4	PPCR31.3	PPCR31.2	PPCR31.1	PPCR31.0	1111 1111
PPCR32	0F20h	PPCR32.7	PPCR32.6	PPCR32.5	PPCR32.4	PPCR32.3	PPCR32.2	PPCR32.1	PPCR32.0	1111 1111
PPCR33	0F21h	PPCR33.7	PPCR33.6	PPCR33.5	PPCR33.4	PPCR33.3	PPCR33.2	PPCR33.1	PPCR33.0	1111 1111
PPCR34	0F22h	PPCR34.7	PPCR34.6	PPCR34.5	PPCR34.4	PPCR34.3	PPCR34.2	PPCR34.1	PPCR34.0	1111 1111
PPCR35	0F23h	PPCR35.7	PPCR35.6	PPCR35.5	PPCR35.4	PPCR35.3	PPCR35.2	PPCR35.1	PPCR35.0	1111 1111
		R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	

Note: PPCR Register bits can only be written, if the attached PPER register bit is low (resistors disabled).

3.7 Programmable Port Output Drive

The Ports listed in the following table have a programmable output drive option, which can be enabled bit-wise. The function is enabled by the Port Output Drive Registers (PODR).

Bit	**Port Output Drive Registers**	
	0 (default)	**1**
PODRx.y	5 mA output drive	2 mA output drive

	Addr	7	6	5	4	3	2	1	0	initial
PODR00	0E00h	PODR00.7	PODR00.6	PODR00.5	PODR00.4	PODR00.3	PODR00.2	PODR00.1	PODR00.0	0000 0000
PODR01	0E01h	PODR01.7	PODR01.6	PODR01.5	PODR01.4	PODR01.3	PODR01.2	PODR01.1	PODR01.0	0000 0000
PODR02	0E02h	PODR02.7	PODR02.6	PODR02.5	PODR02.4	PODR02.3	PODR02.2	PODR02.1	PODR02.0	0000 0000
PODR03	0E03h	PODR03.7	PODR03.6	PODR03.5	PODR03.4	PODR03.3	PODR03.2	PODR03.1	PODR03.0	0000 0000
PODR04	0E04h	PODR04.7	PODR04.6	PODR04.5	PODR04.4	PODR04.3	PODR04.2	PODR04.1	PODR04.0	0000 0000
PODR05	0E05h	PODR05.7	PODR05.6	PODR05.5	PODR05.4	PODR05.3	PODR05.2	PODR05.1	PODR05.0	0000 0000
PODR06	0E06h	PODR06.7	PODR06.6	PODR06.5	PODR06.4	PODR06.3	PODR06.2	PODR06.1	PODR06.0	0000 0000
PODR07	0E07h	PODR07.7	PODR07.6	PODR07.5	PODR07.4	PODR07.3	PODR07.2	PODR07.1	PODR07.0	0000 0000
PODR08	0E08h	PODR08.7	PODR08.6	PODR08.5	PODR08.4	PODR08.3	PODR08.2	PODR08.1	PODR08.0	0000 0000
PODR09	0E09h	PODR09.7	PODR09.6	PODR09.5	PODR09.4	PODR09.3	PODR09.2	PODR09.1	PODR09.0	0000 0000
PODR10	0E0Ah	PODR10.7	PODR10.6	PODR10.5	PODR10.4	PODR10.3	PODR10.2	PODR10.1	PODR10.0	0000 0000
PODR11	0E0Bh	PODR11.7	PODR11.6	PODR11.5	PODR11.4	PODR11.3	PODR11.2	PODR11.1	PODR11.0	0000 0000
PODR12	0E0Ch	PODR12.7	PODR12.6	PODR12.5	PODR12.4	PODR12.3	PODR12.2	PODR12.1	PODR12.0	0000 0000
PODR13	0E0Dh	PODR13.7	PODR13.6	PODR13.5	PODR13.4	PODR13.3	PODR13.2	PODR13.1	PODR13.0	0000 0000
PODR14	0E0Eh	PODR14.7	PODR14.6	PODR14.5	PODR14.4	PODR14.3	PODR14.2	PODR14.1	PODR14.0	0000 0000
PODR15	0E0Fh	PODR15.7	PODR15.6	PODR15.5	PODR15.4	PODR15.3	PODR15.2	PODR15.1	PODR15.0	0000 0000
PODR16	0E10h	PODR16.7	PODR16.6	PODR16.5	PODR16.4	PODR16.3	PODR16.2	PODR16.1	PODR16.0	0000 0000
PODR17	0E11h	PODR17.7	PODR17.6	PODR17.5	PODR17.4	PODR17.3	PODR17.2	PODR17.1	PODR17.0	0000 0000
PODR18	0E12h	PODR18.7	PODR18.6	PODR18.5	PODR18.4	PODR18.3	PODR18.2	PODR18.1	PODR18.0	0000 0000
PODR19	0E13h	PODR19.7	PODR19.6	PODR19.5	PODR19.4	PODR19.3	PODR19.2	PODR19.1	PODR19.0	0000 0000
PODR20	0E14h	PODR20.7	PODR20.6	PODR20.5	PODR20.4	PODR20.3	PODR20.2	PODR20.1	PODR20.0	0000 0000
PODR21	0E15h	PODR21.7	PODR21.6	PODR21.5	PODR21.4	PODR21.3	PODR21.2	PODR21.1	PODR21.0	0000 0000
PODR22	0E16h	PODR22.7	PODR22.6	PODR22.5	PODR22.4	PODR22.3	PODR22.2	PODR22.1	PODR22.0	0000 0000
PODR23	0E17h	PODR23.7	PODR23.6	PODR23.5	PODR23.4	PODR23.3	PODR23.2	PODR23.1	PODR23.0	0000 0000
PODR24	0E18h	PODR24.7	PODR24.6	PODR24.5	PODR24.4	PODR24.3	PODR24.2	PODR24.1	PODR24.0	0000 0000
PODR25	0E19h	PODR25.7	PODR25.6	PODR25.5	PODR25.4	PODR25.3	PODR25.2	PODR25.1	PODR25.0	0000 0000
PODR26	0E1Ah	PODR26.7	PODR26.6	PODR26.5	PODR26.4	PODR26.3	PODR26.2	PODR26.1	PODR26.0	0000 0000
PODR27	0E1Bh	PODR27.7	PODR27.6	PODR27.5	PODR27.4	PODR27.3	PODR27.2	PODR27.1	PODR27.0	0000 0000
PODR28	0E1Ch	PODR28.7	PODR28.6	PODR28.5	PODR28.4	PODR28.3	PODR28.2	PODR28.1	PODR28.0	0000 0000
PODR29	0E1Dh	PODR29.7	PODR29.6	PODR29.5	PODR29.4	PODR29.3	PODR29.2	PODR29.1	PODR29.0	0000 0000
PODR30	0E1Eh	PODR30.7	PODR30.6	PODR30.5	PODR30.4	PODR30.3	PODR30.2	PODR30.1	PODR30.0	0000 0000
PODR31	0E1Fh	PODR31.7	PODR31.6	PODR31.5	PODR31.4	PODR31.3	PODR31.2	PODR31.1	PODR31.0	0000 0000
PODR32	0E20h	PODR32.7	PODR32.6	PODR32.5	PODR32.4	PODR32.3	PODR32.2	PODR32.1	PODR32.0	0000 0000
PODR33	0E21h	PODR33.7	PODR33.6	PODR33.5	PODR33.4	PODR33.3	PODR33.2	PODR33.1	PODR33.0	0000 0000

PODR34	0E22h	PODR34.7	PODR34.6	PODR34.5	PODR34.4	PODR34.3	PODR34.2	PODR34.1	PODR34.0	0000 0000
PODR35	0E23h	PODR35.7	PODR35.6	PODR35.5	PODR35.4	PODR35.3	PODR35.2	PODR35.1	PODR35.0	0000 0000
		R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	

Chapter 31 External Bus

The external bus interface controller controls the interfaces with the internal bus for chips and with external memory and I/O devices.

This chapter explains each function of the external bus interface and its operation.

1. Overview of the External Bus Interface

1.1 Features

● The external bus interface has the following features:

- Addresses of up to 32 bits (4 GB space) can be output.
- Various kinds of external memory (8-bit/16-bit/32-bit modules) can be directly connected and multiple access timings can be mixed and controlled.
- Asynchronous SRAM and asynchronous ROM/FLASH memory (multiple write strobe method or byte enable method)
- Page mode ROM/FLASH memory (Page sizes 2, 4, and 8 can be used)
- Burst mode ROM/FLASH memory (such as MBM29BL160D/161D/162D)
- Address/data multiplex bus (8-bit/16-bit width only)
- SDRAM (FCRAM modules are also supported, including two - and four - bank types with CAS latency 1 to 8)
- Synchronous memory (such as ASIC built-in memory) (Synchronous SRAM cannot be directly connected)
- Eight independent banks (chip select areas) can be set, and chip select corresponding to each bank can be output.
- The size of each area can be set in multiples of 64 KB (64 KB to 2 GB for each chip select area).
- An area can be set at any location in the logical address space (Boundaries may be limited depending on the size of the area.)

● In each chip select area, the following functions can be set independently:

- Enabling and disabling of the chip select area (Disabled areas cannot be accessed)
- Setting of the access timing type to support various kinds of memory
- Detailed access timing setting (individual setting of the access type such as the wait cycle)
- Setting of the data bus width (8-bit/16-bit)
- Setting of the order of bytes (big or little endian) (Only big endian can be set for the CS0 area)
- Setting of write disable (read-only area)
- Enabling and disabling of fetches from the built-in cache
- Enabling and disabling of the prefetch function
- Maximum burst length setting (1, 2, 4, 8)

● A different detailed timing can be set for each access timing type.

- For the same type of access timing, a different setting can be made in each chip select area.
- Auto-wait can be set to up to 15 cycles (asynchronous SRAM, ROM, Flash, and I/O area).
- The bus cycle can be extended by external RDY input (asynchronous SRAM, ROM, Flash, and I/O area).
- The first access wait and page wait can be set (burst, page mode, and ROM/FLASH area).
- Various kinds of idle/recovery cycles and setting delays can be inserted.

- Capable of setting timing values such as the CAS latency and RAS - CAS delay (SDRAM area)
- Capable of controlling the distributed/centralized auto - refresh, self - refresh, and other refresh timings (SDRAM area)

● Fly-by transfer by DMA can be performed.

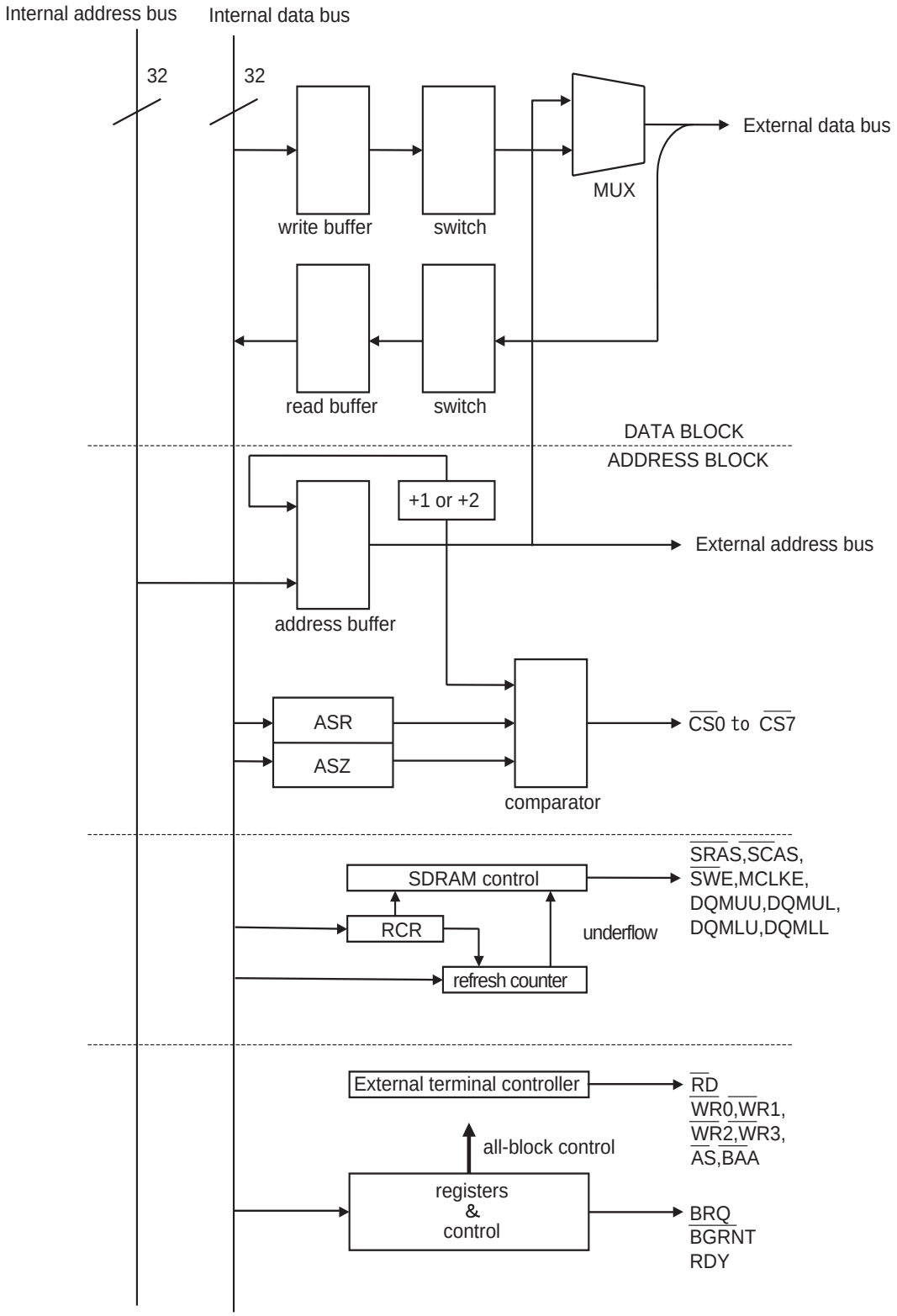
- Transfer between memory and I/O can be performed in a single access operation.
- The memory wait cycle can be synchronized with the I/O wait cycle in fly-by transfer.
- The hold time can be secured by only extending transfer source access.
- Idle/recovery cycles specific to fly-by transfer can be set.

● External bus arbitration using BRQ and $\overline{\text{BGRNT}}$ can be performed.

● Pins that are not used by the external interface can be used as general-purpose I/O ports through settings.

1.2 Block Diagram

Figure 1-1 Block Diagram of the External Bus Interface



1.3 I/O Pins

The I/O pins are external bus interface pins (Some pins have other uses).

The following lists the I/O pins for each interface:

● Ordinary bus interface

- A31 to A00, D31 to D00 (AD15 to AD00)
- $\overline{CS0}$, $\overline{CS1}$, $\overline{CS2}$, $\overline{CS3}$, $\overline{CS4}$, $\overline{CS5}$, $\overline{CS6}$, $\overline{CS7}$
- $\overline{AS}$, SYSCLK, MCLK
- $\overline{RD}$
- $\overline{WR}$, $\overline{WR0}$($\overline{UUB}$), $\overline{WR1}$($\overline{ULB}$), $\overline{WR2}$($\overline{ULB}$), $\overline{WR3}$($\overline{ULB}$)
- RDY, BRQ, $\overline{BGRNT}$

● Memory interface

- MCLK, MCLKE
- MCLKI (for SDRAM)
- $\overline{LBA}$(=$\overline{AS}$), $\overline{BAA}$ (for burst ROM/FLASH)
- $\overline{SRAS}$, $\overline{SCAS}$, $\overline{SWE}$ (=$\overline{WR}$) (for SDRAM)
- DQMUU,DQMUL,DQMLU,DQMLL (for SDRAM (=$\overline{WR0}$, $\overline{WR1}$, $\overline{WR2}$, $\overline{WR3}$))

● DMA interface

- $\overline{IOWR}$, $\overline{IORD}$
- DACK0, DACK1
- DREQ0, DREQ1
- DEOP0, DEOP1

1.4 Register List

Figure 1-2 "List of External Bus Interface Registers" shows the registers used by the external bus interface:

Figure 1-2 List of External Bus Interface Registers

Address	31 24	23 16	15 08	07 00
00000640_H	ASR0		ACR0	
00000644_H	ASR1		ACR1	
00000648_H	ASR2		ASR2	
$0000064c_H$	ASR3		ACR3	
00000650_H	ASR4		ACR4	
00000654_H	ASR5		ACR5	
00000658_H	ASR6		ACR6	
$0000065c_H$	ASR7		ACR7	
00000660_H	AWR0		AWR1	
00000664_H	AWR2		AWR3	
00000668_H	AWR4		AWR5	
$0000066c_H$	AWR6		AWR7	
00000670_H	MCRA	MCRB	Reserved	Reserved
00000674_H	Reserved	Reserved	Reserved	Reserved
00000678_H	IOWR0	IOWR1	IOWR2	IOWR3
$0000067c_H$	Reserved	Reserved	Reserved	Reserved
00000680_H	CSER	CHER	Reserved	TCR
00000684_H	Reserved	Reserved	Reserved	Reserved
00000688_H	Reserved	Reserved	Reserved	Reserved
$0000068c_H$	Reserved	Reserved	Reserved	Reserved
	Reserved	Reserved	Reserved	Reserved
$000007f8_H$	Reserved	Reserved	Reserved	Reserved
$000007fc_H$	Reserved	(MODR)	Reserved	Reserved

*1: Reserved indicates a reserved register. Be sure to set "0".
*2: MODR cannot be accessed from user programs.

2. External Bus Interface Registers

This section explains the registers used in the external bus interface.

■ Register Types

The following registers are used by the external bus interface:

- Area select registers (ASR0-7)
- Area configuration registers (ACR0-7)
- Area wait registers (AWR0-7)
- Memory configuration register (MCRA for SDRAM/FCRAM auto - precharge OFF mode)
- Memory configuration register (MCRB for FCRAM auto - precharge ON mode)
- I/O wait registers for DMAC (IOWR0-3)
- Chip select enable register (CSER)
- Cache enable register (CHER)
- Pin/timing control register (TCR)
- Mode register (MODR)

2.1 Area Select Registers 0-7(ASR0-7)

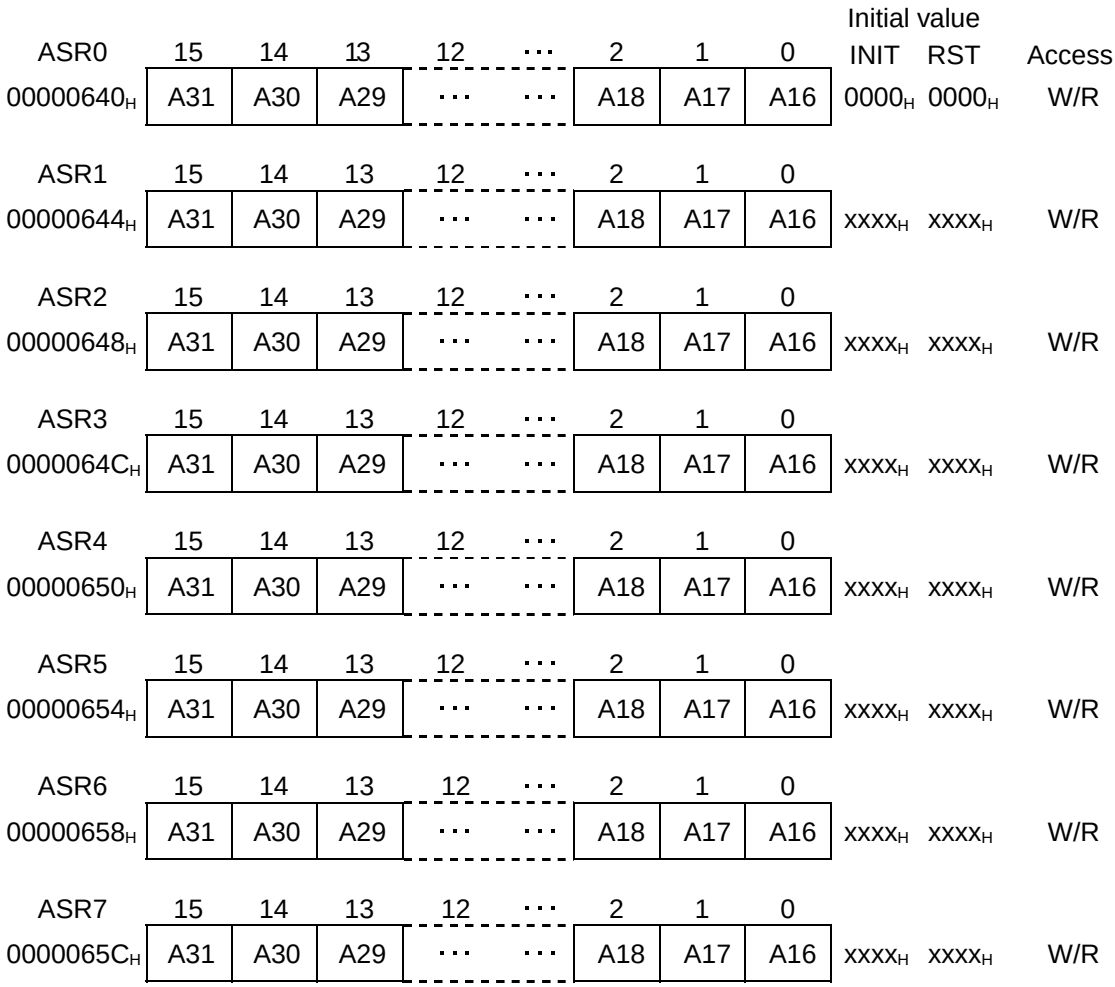
This section explains the configuration and functions of area select registers 0-7 (ASR0-7).

■ Configuration of area select registers 0-7 (ASR0-7)

The area select registers (ASR0-7: Area Select Registers 0-7) specify the start address of each chip select area of $\overline{CS0}$-$\overline{CS7}$.

Figure 2-1 "Configuration of the Area Select Registers (ASR0-7)" shows the configuration of area select registers 0-7 (ASR0-7: Area Select Register).

Figure 2-1 Configuration of the Area Select Registers (ASR0-7)



■ Functions of Bits in the Area Select Registers (ASR0-7)

The start address can be set in the high-order 16 bits (bits A31-A16). Each chip select area starts with the address set in this register and covers the range set by the four bits ASZ3-0 of the ASR0-7 registers.

The boundary of each chip select area obeys the setting of the four bits ASZ3-0 of the ACR0-7 registers. For example, if an area of 1 MB is set by the four bits ASZ3-0, the low-order four bits of the ASR0-7 registers are ignored and only bits A31-20 are valid.

The ASR0 register is initialized to 0000_H by INIT and RST. ASR1-7 are not initialized by INIT and RST, and are therefore undefined. After starting chip operation, be sure to set the corresponding ASR register before enabling each chip select area with the CSER register.

2.2 Area Configuration Registers 0-7 (ACR0-7)

This section explains the configuration and functions of area configuration registers 0-7 (ACR0-7).

■ Configuration of Area Configuration Registers 0-7 (ACR0-7)

The area configuration registers 0-7 (ACR0-7: FArea Configuration Register 0-7) set the function of each chip

select area.

Figure 2-2 "Configuration of Area Configuration Registers 0-7 (ACR0-7)" shows the configuration of area configuration registers 0-7 (ACR0-7).

Figure 2-2 Configuration of Area Configuration Registers 0-7 (ACR0-7) (Continued on next page)

									Initial value		
ACR0H	15	14	13	12	11	10	9	8	INIT	RST	Access
00000642_H	ASZ3	ASZ2	ASZ1	ASZ0	DBW1	DBW0	BST1	BST0	$1111**00_B$	$1111**00_B$	W/R
ACR0L	7	6	5	4	3	2	1	0			
00000643_H	SREN	PFEN	WREN	0	TYP3	TYP2	TYP1	TYP0	00000000_B	00000000_B	W/R
ACR1H	15	14	13	12	11	10	9	8			
00000646_H	ASZ3	ASZ2	ASZ1	ASZ0	DBW1	DBW0	BST1	BST0	$Xxxxxxxx_B$	$xxxxxxxx_B$	W/R
ACR1L	7	6	5	4	3	2	1	0			
00000647_H	SREN	PFEN	WREN	LEND	TYP3	TYP2	TYP1	TYP0	$xxxxxxxx_B$	$Xxxxxxxx_B$	W/R
ACR2H	15	14	13	12	11	10	9	8			
$0000064A_H$	ASZ3	ASZ2	ASZ1	ASZ0	DBW1	DBW0	BST1	BST0	$xxxxxxxx_B$	$xxxxxxxx_B$	W/R
ACR2L	7	6	5	4	3	2	1	0			
$0000064B_H$	SREN	PFEN	WREN	LEND	TYP3	TYP2	TYP1	TYP0	$xxxxxxxx_B$	$xxxxxxxx_B$	W/R
ACR3H	15	14	13	12	11	10	9	8			
$0000064E_H$	ASZ3	ASZ2	ASZ1	ASZ0	DBW1	DBW0	BST1	BST0	$xxxxxxxx_B$	$xxxxxxxx_B$	W/R
ACR3L	7	6	5	4	3	2	1	0			
$0000064F_H$	SREN	PFEN	WREN	LEND	TYP3	TYP2	TYP1	TYP0	$Xxxxxxxx_B$	$xxxxxxxx_B$	W/R

									Initial value		
ACR4H	15	14	13	12	11	10	9	8	INIT	RST	Access
00000652_H	ASZ3	ASZ2	ASZ1	ASZ0	DBW1	DBW0	BST1	BST0	$xxxxxxxx_B$	$xxxxxxxx_B$	W/R
ACR4L	7	6	5	4	3	2	1	0			
00000653_H	SREN	PFEN	WREN	LEND	TYP3	TYP2	TYP1	TYP0	$xxxxxxxx_B$	$xxxxxxxx_B$	W/R
ACR5H	15	14	13	12	11	10	9	8			
00000656_H	ASZ3	ASZ2	ASZ1	ASZ0	DBW1	DBW0	BST1	BST0	$xxxxxxxx_B$	$xxxxxxxx_B$	W/R
ACR5L	7	6	5	4	3	2	1	0			
00000657_H	SREN	PFEN	WREN	LEND	TYP3	TYP2	TYP1	TYP0	$xxxxxxxx_B$	$xxxxxxxx_B$	W/R
ACR6H	15	14	13	12	11	10	9	8			
$0000065A_H$	ASZ3	ASZ2	ASZ1	ASZ0	DBW1	DBW0	BST1	BST0	$xxxxxxxx_B$	$xxxxxxxx_B$	W/R
ACR6L	7	6	5	4	3	2	1	0			
$0000065B_H$	SREN	PFEN	WREN	LEND	TYP3	TYP2	TYP1	TYP0	$xxxxxxxx_B$	$xxxxxxxx_B$	W/R
ACR7H	15	14	13	12	11	10	9	8			
$0000065E_H$	ASZ3	ASZ2	ASZ1	ASZ0	DBW1	DBW0	BST1	BST0	$xxxxxxxx_B$	$xxxxxxxx_B$	W/R
ACR7L	7	6	5	4	3	2	1	0			
$0000065F_H$	SREN	PFEN	WREN	LEND	TYP3	TYP2	TYP1	TYP0	$xxxxxxxx_B$	$xxxxxxxx_B$	W/R

The following explains the function of each bit:

[Bits 15-12] ASZ3-0 (Area Size Bits 3-0)

These bits set the area size. Table 2-1 "Area Size Settings" shows their settings.

Table 2-1 Area Size Settings

ASZ3	ASZ2	ASZ1	ASZ0	Size of each chip select area
0	0	0	0	64 KB (00010000_H byte, ASR A[31:16] bits are valid)
0	0	0	1	128 KB (00020000_H byte, ASR A[31:17] bits are valid)
0	0	1	0	256 KB (00040000_H byte, ASR A[31:18] bits are valid)
0	0	1	1	512 KB (00080000_H byte, ASR A[31:19] bits are valid)
0	1	0	0	1 MB (00100000_H byte, ASR A[31:20] bits are valid)
0	1	0	1	2 MB (00200000_H byte, ASR A[31:21] bits are valid)
0	1	1	0	4 MB (00400000_H byte, ASR A[31:22] bits are valid)

Table 2-1 Area Size Settings

ASZ3	ASZ2	ASZ1	ASZ0	Size of each chip select area
0	1	1	1	8 MB (00800000_H byte, ASR A[31:23] bits are valid)
1	0	0	0	16 MB (01000000_H byte, ASR A[31:24] bits are valid)
1	0	0	1	32 MB (02000000_H byte, ASR A[31:25] bits are valid)
1	0	1	0	64 MB (04000000_H byte, ASR A[31:26] bits are valid)
1	0	1	1	128 MB (08000000_H byte, ASR A[31:27] bits are valid)
1	1	0	0	256 MB (10000000_H byte, ASR A[31:28] bits are valid)
1	1	0	1	512 MB (20000000_H byte, ASR A[31:29] bits are valid)
1	1	1	0	1024 MB (40000000_H byte, ASR A[31:30] bits are valid)
1	1	1	1	2048 MB (80000000_H byte, ASR A[31] bit is valid)

ASZ3-0 are used to set the size of each area by modifying the number of bits for address comparison to a value different from ASR. Thus, an ASR contains bits that are not compared. Bits ASZ3-0 of ACR0 are initialized to 1111_B ($0F_H$) by RST. Despite this setting, however, the CS0 area just after RST is executed is specially set from 00000000_H to $FFFFFFFF_H$ (setting of entire area). The entire-area setting is reset after the first write to ACR0 and an appropriate size is set as indicated in Table 2-1 "Area Size Settings".

[Bits 11-10] DBW1-0 (Data Bus Width 1-0)

These bits set the data bus width of each chip select area as indicated in Table 2-2 "Setting of the Data Bus Width of Each Chip Select Area":

Table 2-2 Setting of the Data Bus Width of Each Chip Select Area

DBW1	DBW0	Data bus width
0	0	8 bits (byte access)
0	1	16 bits (halfword access)
1	0	32 bits (word access)
1	1	Reserved Setting disabled

The same values as those of the WTH bits of the mode vector are written automatically to bits DBW1-0 of ACR0 during the reset sequence.

[Bits 9-8] BST1-0 (Burst Size 1-0)

These bits set the maximum burst length of each chip select area as indicated in Table 2-3 "Setting of the Maximum Burst Length of Each Chip Select".

Table 2-3 Setting of the Maximum Burst Length of Each Chip Select

BST1	BST0	Maximum burst length
0	0	1 (single access)
0	1	2 bursts (address boundary: 1 bit)

Table 2-3 Setting of the Maximum Burst Length of Each Chip Select

BST1	BST0	Maximum burst length
1	0	4 bursts (address boundary: 2 bits)
1	1	8 bursts (address boundary: 3 bits)

In areas for which a burst length other than the single access is set, continuous burst access is performed within the address boundary determined by the burst length only when prefetch access is performed or data having a size exceeding the bus width is read.

Setting of 2 bursts or less as the maximum burst length in the bus width 16-bit area is recommended.

RDY input is ignored in areas for which any burst length other than the single access is set.

[Bit 7] SREN (ShaRed Enable)

This bit sets enabling or disabling of sharing of each chip select area by BRQ/$\overline{\text{BGRNT}}$ as indicated in the following table.

SREN	Sharing enable/disable
0	Disable sharing by BRQ/$\overline{\text{BGRNT}}$ ($\overline{\text{CSn}}$ cannot be high impedance)
1	Enable sharing by BRQ/$\overline{\text{BGRNT}}$ ($\overline{\text{CSn}}$ can be high impedance)

In areas where sharing is enabled, chip select output ($\overline{\text{CSn}}$) is set to high impedance while the bus is open (during $\overline{\text{BGRNT}}$=Low output). In areas where sharing is disabled, chip select output ($\overline{\text{CSn}}$) is not set to high impedance even though the bus is open (during $\overline{\text{BGRNT}}$=Low output).

Access strobe output ($\overline{\text{AS}}$, $\overline{\text{BAA}}$, $\overline{\text{RD}}$, $\overline{\text{WR0}}$, $\overline{\text{WR1}}$, $\overline{\text{WR2}}$, $\overline{\text{WR3}}$, $\overline{\text{WR}}$, MCLK, MCLKE) is set to high impedance only if sharing of all areas enabled by CSER is enabled.

[Bit 6] PFEN (PreFetch Enable)

This bit sets enabling and disabling of prefetching of each chip select area as indicated in the following table.

PFEN	Prefetch enable/disable
0	Disable prefetch
1	Enable prefetch

When reading from an area for which prefetching is enabled, the subsequent address is read in advance and stored in the built-in prefetch buffer. When the stored address is accessed from the internal bus, the lookahead data in the prefetch buffer is returned without performing external access.

For more information, see Section 8. "Prefetch Operation".

[Bit 5] WREN(WRite Enable)

This bit sets enabling and disabling of writing to each chip select area.

WREN	Write enable/disable
0	Disable write

WREN	Write enable/disable
1	Enable write

If an area for which write operations are disabled is accessed for a write operation from the internal bus, the access is ignored and no external access at all is performed. Set the WREN bit of areas for which write operations are not required, such as data areas, to 0.

[Bit 4] LEND (Little ENDian select)

This bit sets the order of bytes of each chip select area as indicated in the following table.

LEND	Order of bytes
0	Big endian
1	Little endian

Be sure to set the LEND bit of ACR0 to 0. CS0 supports only the big endian method.

[Bits 3-0] TYP3-0 (TYPe select)

These bits set the access type of each chip select area as indicated in Table 2-4 "Access Type Settings for Each Chip Select Area".

Table 2-4 Access Type Settings for Each Chip Select Area

TYP3	TYP2	TYP1	TYP0	Access type
0	0	x	x	Normal access (asynchronous SRAM, I/O, and single/page/burst-ROM/FLASH)
0	1	x	x	Address data multiplex access (8/16-bit bus width only)
0	x	x	0	Disable WAIT insertion by the RDY pin.
0	x	x	1	Enable WAIT insertion by the RDY pin (disabled during bursts).
0	x	0	x	Use the $\overline{WR}$0-$\overline{WR3}$ pins as write strobes ($\overline{WEn}$ is always H).
0	x	1	x	Use the $\overline{WEn}$ pin as the write strobe. *1
1	0	0	0	Memory type A: SDRAM/FCRAM *2
1	0	0	1	Memory type B: FCRAM *2
1	0	1	0	Setting disabled
1	0	1	1	Setting disabled
1	1	0	0	Setting disabled
1	1	0	1	Setting disabled
1	1	1	0	Setting disabled

Table 2-4 Access Type Settings for Each Chip Select Area

TYP3	TYP2	TYP1	TYP0	Access type
1	1	1	1	Mask area setting (The access type is the same as that of the overlapping area) *3

*1: If this setting is made, $\overline{WR0}$-$\overline{WR3}$ can be used as the enable of each bit.
*2: Only the ACR6 and ACR7 registers are valid. The ACR0, ACR1, ACR2, ACR3, ACR4, and ACR5 registers are disabled.
*3: See the CS area mask setting function (next bullet).

Set the access type as the combination of all bits.

For details of the operations of each access type, see the explanation of operation of each type.

● CS area mask setting function

If you want to set an area some of whose operation settings are changed for a certain CS area (referred to as the base setting area), you can set TYPE3-0 of ARC in another CS area to 1111 so that the area can function as a mask setting area.

If you do not use the mask setting function, disable any overlapping area settings for multiple CS areas.

Access operations to the mask setting area are as follows:

- $\overline{CS}$ corresponding to a mask setting area is not asserted.
- $\overline{CS}$ corresponding to a base setting area is not asserted.
- For the following ACR settings, the settings on the mask setting area side are valid:
  - Bits 11-10 (DBW1-0): Bus width setting
  - Bits 9-8 (BST1-0): Burst length setting
  - Bit 7 (SREN): Sharing-enable setting
  - Bit 6 (PFEN): Prefetch-enable setting
  - Bit 5 (WREN): Write-enable setting (For this setting only, only a setting that is the same as that of the base setting area is allowed)
  - Bit 4 (LEND): Little endian setting
- For the following ACR setting, the setting on the base setting area side is valid:
  - Bits 3-0 (TYPE3-0): Access type setting
- For the AWR settings, the settings on the mask setting area side are valid.
- For the CHER settings, the settings on the mask setting area side are valid.

A mask setting area can be set for only part of another CS area (base setting area). You cannot set a mask setting area for an area without a base setting area. Use care when setting ASR and bits ASZ3-0 of ACR.

The following restrictions apply when using these bits:

- A write-enable setting cannot be implemented by a mask.
- Write-enable settings in the base CS area and the mask setting area must be identical.
- If write operations to a mask setting area are disabled, the area is not masked and operates as a base CS area.
- If write operations to the base CS area are disabled but are enabled to the mask setting area, the area has no base, resulting in malfunctions.

2.3 Area Wait Register (AWR0-7)

This section explains the configuration and functions of the area wait registers (AWR0-7).

■ Configuration of the Area Wait Registers (AWR0-7)

The area wait registers (AWR0-7: Area Wait Register 0-7) specify various kinds of waits for each chip select area.

Figure 2-3 "Configuration of the Area Wait Registers (AWR0-7)" shows the configuration of the area wait registers (AWR0-7).

Figure 2-3 Configuration of the Area Wait Registers (AWR0-7) (Continued on next page)

									Initial value		
AWR0H	31	30	29	28	27	26	25	24	INIT	RST	Access
00000660_H	W15	W14	W13	W12	W11	W10	W09	W08	01111111_b	01111111_b	W/R
AWR0L	23	22	21	20	19	18	17	16			
00000661_H	W07	W06	W05	W04	W03	W02	W01	W00	11111011_B	11111011_B	W/R
AWR1H	15	14	13	12	11	10	9	8			
00000662_H	W15	W14	W13	W12	W11	W10	W09	W08	$xxxxxxxx_b$	$xxxxxxxx_b$	W/R
AWR1L	7	6	5	4	3	2	1	0			
00000663_H	W07	W06	W05	W04	W03	W02	W01	W00	$xxxxxxxx_b$	$xxxxxxxx_b$	W/R
AWR2H	31	30	29	28	27	26	25	24			
00000664_H	W15	W14	W13	W12	W11	W10	W09	W08	$xxxxxxxx_b$	$xxxxxxxx_b$	W/R
AWR2L	23	22	21	20	19	18	17	16			
00000665_H	W07	W06	W05	W04	W03	W02	W01	W00	$xxxxxxxx_b$	$xxxxxxxx_b$	W/R
AWR3H	15	14	13	12	11	10	9	8			
00000666_H	W15	W14	W13	W12	W11	W10	W09	W08	$xxxxxxxx_b$	$xxxxxxxx_b$	W/R
AWR3L	7	6	5	4	3	2	1	0			
00000667_H	W07	W06	W05	W04	W03	W02	W01	W00	$xxxxxxxx_b$	$xxxxxxxx_b$	W/R
AWR4H	31	30	29	28	27	26	25	24			
00000668_H	W15	W14	W13	W12	W11	W10	W09	W08	$xxxxxxxx_b$	$xxxxxxxx_b$	W/R
AWR4L	23	22	21	20	19	18	17	16			
00000669_H	W07	W06	W05	W04	W03	W02	W01	W00	$xxxxxxxx_b$	$xxxxxxxx_b$	W/R

									Initial value		
AWR5H	15	14	13	12	11	10	9	8	INIT	RST	Access
$0000066A_H$	W15	W14	W13	W12	W11	W10	W09	W08	$xxxxxxxx_b$	$xxxxxxxx_b$	W/R
AWR5L	7	6	5	4	3	2	1	0			
$0000066B_H$	W07	W06	W05	W04	W03	W02	W01	W00	$xxxxxxxx_b$	$xxxxxxxx_b$	W/R
AWR6H	31	30	29	28	27	26	25	24			
$0000066C_H$	W15	W14	W13	W12	W11	W10	W09	W08	$xxxxxxxx_b$	$xxxxxxxx_b$	W/R
AWR6L	23	22	21	20	19	18	17	16			
$0000066D_H$	W07	W06	W05	W04	W03	W02	W01	W00	$xxxxxxxx_b$	$xxxxxxxx_b$	W/R
AWR7H	15	14	13	12	11	10	9	8			
$0000066E_H$	W15	W14	W13	W12	W11	W10	W09	W08	$xxxxxxxx_b$	$xxxxxxxx_b$	W/R
AWR7L	7	6	5	4	3	2	1	0			
$0000066F_H$	W07	W06	W05	W04	W03	W02	W01	W00	$xxxxxxxx_b$	$xxxxxxxx_b$	W/R

The function of each bit changes according to the access type (TYP(3-0) bits) setting of the ACR0-7 registers,. A chip select area determined by either of the following settings becomes the area for normal access or a address/ data multiplex access operation.

TYP3	TYP2	TYP1	TYP0	Access type
0	0	x	x	Normal access (asynchronous SRAM, I/O, and single/page/burst-ROM/FLASH)
0	1	x	x	Address data multiplex access (8/16-bit bus width only)

The following lists the functions of each AWR0-7 bit for a normal access or address/data multiplex access area. Since the initial values of registers other than AWR0 are undefined, set them to their initial values before enabling each area with the CSER register.

The following explains the functions of the bits in the area wait registers (AWR0-7).

[Bits 15-12] W15-12 (First Wait Cycle)

These bits set the number of auto-wait cycles to be inserted into the first access cycle of each cycle. Except for the burst access cycles, only this wait setting is used.

Table 2-5 "Settings for the Number of Auto-Wait Cycles (During First Access)" lists the settings for the number of auto-wait cycles during first access.

Table 2-5 Settings for the Number of Auto-Wait Cycles (During First Access)

W15	W14	W13	W12	First access wait cycle
0	0	0	0	Auto-wait cycle 0
0	0	0	1	Auto-wait cycle 1
...				...
1	1	1	1	Auto-wait cycle 15

[Bits 11-8] W11-08 (Inpage Access Wait Cycle)

These bits set the number of auto-wait cycles to be inserted into the inpage access cycle during burst access. They are valid only for burst cycles.

Table 2-6 "Settings for the Number of Auto-Wait Cycles (During Burst Access)" lists the settings for the number of auto-wait cycles during burst access.

Table 2-6 Settings for the Number of Auto-Wait Cycles (During Burst Access)

W11	W10	W09	W08	Inpage access wait cycle
0	0	0	0	Auto-wait cycle 0
0	0	0	1	Auto-wait cycle 1
...				...
1	1	1	1	Auto-wait cycle 15

If the same value is set for the first access wait cycle and inpage access wait cycle, the access time for the address in each access cycle is not the same. This is because the inpage access cycle contains an address output delay.

[Bits 7,6] W07-06 (Read -> Write Idle Cycle)

The read -> write idle cycle is set to prevent collision of read data and write data on the data bus when a write cycle follows a read cycle. During an idle cycle, all chip select signals are negated and the data terminals maintain the high impedance state. If a write cycle follows a read cycle or an access operation to another chip select area occurs after a read cycle, the specified idle cycle is inserted. Table 2-7 "Settings of the Idle Cycle" lists the settings for idle cycles.

Table 2-7 Settings of the Idle Cycle

W07	W06	Read -> write idle cycles
0	0	0 cycle
0	1	1 cycle
1	0	2 cycles
1	1	3 cycles

[Bits 5, 4] W05, W04 (Write Recovery Cycle)

The write recovery cycle is set if a device that limits the access period after write access is to be controlled. During a write recovery cycle, all chip select signals are negated and the data pins maintain the high impedance state. If the write recovery cycle is set to 1 or more, a write recovery cycle is always inserted after write access.

Table 2-8 "Settings for the Number of Write Recovery Cycles" lists the settings for the number of write recovery cycles.

Table 2-8 Settings for the Number of Write Recovery Cycles

W05	W04	Write recovery cycles
0	0	0 cycle
0	1	1 cycle
1	0	2 cycles
1	1	3 cycles

[Bit 3] W03 ($\overline{WR0}$-$\overline{WR3}$, $\overline{WRn}$ Output Timing Selection)

The $\overline{WR0}$-$\overline{WR3}$, $\overline{WRn}$ output timing setting selects whether to use write strobe output as an asynchronous strobe or synchronous write enable. The asynchronous strobe setting corresponds to normal memory/IO. The synchronous enable setting corresponds to clock-synchronized memory/IO (such as the memory in an ASIC).

W03	$\overline{WR0}$-$\overline{WR3}$, $\overline{WRn}$ output timing selection
0	MCLK synchronous write enable output (valid from $\overline{AS}$=L)
1	Asynchronous write strobe output (normal operation)

If synchronous write enable (W03 bit of AWR is 1) is used, operations are as follows:

- The timing of synchronous write enable output assumes that the output is captured by the rising edge of MCLK output of an external memory access clock. This timing is different from the asynchronous strobe output timing.
- The $\overline{WR0}$-$\overline{WR3}$ and $\overline{WRn}$ terminal output asserts synchronous write enable output at the timing at which $\overline{AS}$ pin output is asserted. For a write to an external bus, the synchronous write enable output is L. For a read from an external bus, the synchronous write enable output is H.
- Write data is output from the external data output pin in the clock cycle following the cycle in which synchronous write enable output is asserted. If write data cannot be output because the internal bus is temporarily unavailable, assertion of synchronous write enable output may be extended until write data can be output.
- Read strobe output ($\overline{RD}$) functions as an asynchronous read strobe regardless of the setting of the $\overline{WR0}$-$\overline{WR3}$ and $\overline{WRn}$ output timing. Use it as is for controlling the data I/O direction.

If synchronous write enable output is used, the following restrictions apply:

- Do not make the following additional wait settings:
  - $\overline{CSn}$ -> $\overline{RD}$/$\overline{WRn}$ setup (Always set 0 for the W01 bit of AWR)
  - First wait cycle setting (Always set 0000_B for the W15-W12 bits of AWR)
- Do not make the following access type settings (TYPE3-0 bits in the ACR register (bits 3-0))
  - Address/data multiplex bus setting (Always set 0 for the TYPE2 bit of ACR)
  - Setting to use $\overline{WR0}$-$\overline{WR3}$ as a strobe (Always set 0 for the TYPE1 bit of ACR)
  - RDY input enable setting (Always set 0 for the TYPE0 bit of ACR)
- For synchronous write enable output, always set 1(00_B for bits BST1-0 bits of ACR) as the burst length.

[Bit 2] W02 (Address -> $\overline{\text{CSn}}$ Delay)

The address -> $\overline{\text{CSn}}$ delay setting is made when a certain type of setup is required for the address when $\overline{\text{CSn}}$ falls or $\overline{\text{CSn}}$ edges are needed for successive accesses to the same chip select area.

Set the address and set the delay from $\overline{\text{AS}}$ output to $\overline{\text{CS0}}$-$\overline{\text{CS7}}$ output.

W02	Address -> $\overline{\text{CSn}}$ delay
0	No delay
1	Delay

If no delay is selected by setting 0, assertion of $\overline{\text{CS0}}$-$\overline{\text{CS7}}$ starts at the same timing that $\overline{\text{AS}}$ is asserted. If, at this point, successive accesses are made to the same chip select area, assertion of $\overline{\text{CS0}}$-$\overline{\text{CS7}}$ without change between two access operations may continue.

If delay is specified by selecting 1, assertion of $\overline{\text{CS0}}$-$\overline{\text{CS7}}$ starts when the external clock memory MCLK output rises. If, at this point, successive accesses are made to the same chip select area, $\overline{\text{CS0}}$-$\overline{\text{CS7}}$ are negated at a timing between two access operations. If $\overline{\text{CS}}$ delay is selected, one setup cycle is inserted before asserting the read/write strobe after assertion of the delayed $\overline{\text{CSn}}$ (operation is the same as the $\overline{\text{CSn}}$ ->$\overline{\text{RD}}$/$\overline{\text{WE}}$ setup setting of W01).

The address -> $\overline{\text{CSn}}$ delay setting works for $\overline{\text{DACK}}$ signal (basic mode) output to the same area in the same way. DACK output in basic mode has the same waveforms as those of $\overline{\text{CS}}$ output to the same area.

[Bit 1] W01 ($\overline{\text{CSn}}$ -> $\overline{\text{RD}}$/$\overline{\text{WRn}}$ Setup Extension Cycle)

The $\overline{\text{CSn}}$ -> $\overline{\text{RD}}$/$\overline{\text{WRn}}$ setup extension cycle is set to extend the period before the read/write strobe is asserted after $\overline{\text{CSn}}$ is asserted. At least one setup extension cycle is inserted before the read/write strobe is asserted after $\overline{\text{CS}}$ is asserted.

W01	$\overline{\text{CSn}}$ -> $\overline{\text{RD}}$/$\overline{\text{WRn}}$ setup delay cycle
0	0 cycle
1	1 cycle

If 0 cycle is selected by setting 0, $\overline{\text{RD}}$/$\overline{\text{WR0}}$-$\overline{\text{WR3}}$/$\overline{\text{WRn}}$ are output at the earliest when external clock MCLK output rises just after $\overline{\text{CS}}$ is asserted. $\overline{\text{WR0}}$-$\overline{\text{WR3}}$/$\overline{\text{WRn}}$ may be delayed one cycle or more depending on the internal bus state.

If 1 cycle is selected by setting 1, $\overline{\text{RD}}$/$\overline{\text{WR0}}$-$\overline{\text{WR3}}$/$\overline{\text{WRn}}$ are always output 1 cycle or more later.

When successive accesses are made within the same chip select area without negating $\overline{\text{CSn}}$, a setup extension cycle is not inserted. If a setup extension cycle for determining the address is required, set the W02 bit and insert the address -> $\overline{\text{CSn}}$ delay. Since $\overline{\text{CSn}}$ is negated for each access operation, the setup extension cycle is enabled.

If the $\overline{\text{CSn}}$ delay set by W02 is inserted, this setup cycle is always enabled regardless of the setting of the W01 bit.

[Bit 0] W00 (RD/WRn -> CSn Hold Extension Cycle)

The RD/WRn -> CSn hold extension cycle is set to extend the period before negating CSn after the read/write strobe is negated. One hold extension cycle is inserted before CSn is negated after the read/write strobe is negated.

W00	RD/WRn -> CSn hold extension cycle
0	0 cycle
1	1 cycle

If 0 cycle is selected by setting 0, CS0-CS7 are negated after the hold delay after it starts on the rising edge of external memory clock MCLK output after RD/WR0-WR3/WRn are negated.

If 1 cycle is selected by setting 1, CS0-CS7 are negated one cycle later.

When making successive accesses within the same chip select area without negating CSn, the hold extension cycle is not inserted. If a hold extension cycle for determining the address is required, set the W02 bit and insert the address -> CSn delay. Since CSn is negated for each access operation, this hold extension cycle is enabled.

- Memory type A(SDRAM/FCRAM) and Memory type B(FCRAM)

The chip select areas for which the access type (TYP3 to TYP0 bits) in the ACR6 and ACR7 registers has been set as in Table 1.2 - 18 serve for SDRAM/FCRAM access.

Table 1.2 - 18 lists the access type settings (TYP3 to TYP0 bits).

Table 2-9 Access Type Settings (TYP3 - TYP0 Bits)

TYP3	TYP2	TYP1	TYP0	Access type
1	0	0	0	Memory type A: SDRAM/FCRAM (Auto - precharge is not used.)

The following explains those functions of individual bits in AWR6 and AWR7 which apply to SDRAM access areas. As the initial value is undefined, set the access type before each area is enabled by the chip select area enable register (CSER).

For all the areas connected to SDRAM/FCRAM, use the same settings for this type of registers.

The following summarizes the functions of individual bits in the area wait registers (AWR6 and AWR7).

[Bit 15] W15: Reserved bit

Be sure to set this bit to 0.

[Bits 14 - 12] W14 to W12 (RAS - CAS delay Cycle): RAS - CAS delay cycles

Set these bits to the number of cycles from RAS output to CAS output.

Table 4.2 - 19 lists the settings for the number of cycles from RAS output to CAS output.

Table 2-10 Setting the Number of Cycles from RAS Output to CAS Output

W14	W13	W12	RAS-CAS delay cycle
0	0	0	1 cycle
0	0	0	2 cycles
...			...

Table 2-10 Setting the Number of Cycles from RAS Output to CAS Output

W14	W13	W12	RAS-CAS delay cycle
1	1	1	8 cycles

For all the areas connected to SDRAM/FCRAM, set these bits to the same RAS - CAS delay cycle.

[Bit 11] W11: Reserved bit

Be sure to set this bit to 0.

[Bits 10 - 8] W10 to W08 (CAS latency Cycle): CAS latency

Set these bits to the CAS latency.

Table 4.2 - 20 lists the settings for the CAS latency.

Table 2-11 CAS Latency Setting

W10	W09	W08	CAS latency
0	0	0	1 cycle
0	0	0	2 cycles
...			...
1	1	1	8 cycles

For all the areas connected to SDRAM/FCRAM, set these bits to the same CAS latency.

[Bits 7 - 6] W07 and W06 (Read - >Write Cycle): Read - to - write cycle

Set these bits to the minimum number of cycles from the last read data input cycle to the write command issuance. Set the minimum number of cycles taken until issuance.

Table 4.2 - 21 lists the settings for the read - to - write cycle.

Table 2-12 Read - to - write cycle

W07	W06	Read - to - write cycle
0	0	1 cycle
0	1	2 cycles
1	0	3 cycles
1	1	4 cycles

For all the areas connected to SDRAM/FCRAM, set these bits to the same read - to - write cycle.

The number of read - to - write idle cycles is one smaller than the number of cycles set by this bit.

[Bits 5 - 4] W05 and W04 (Write Recovery Cycle): Write recovery cycle

Set these bits to the minimum number of cycles from the last write data output to the next read command issuance.

Table 4.2 - 22 lists the settings for the write recovery cycle.

Table 2-13 Write recovery cycle

W05	W04	Write recovery cycle
0	0	Prohibited
0	1	2 cycles
1	0	3 cycles
1	1	4 cycles

For all the areas connected to SDRAM/FCRAM, set these bits to the same write recovery cycle.

[Bits 3 - 2] W03 and W02 (RAS Active time): RAS active time

Set these bits to the minimum number of cycles for RAS active time.

Table 4.2 - 23 lists the settings for RAS active time.

Table 2-14 RAS active time

W03	W02	RAS active time
0	0	1 cycle
0	1	2 cycles
1	0	5 cycles
1	1	6 cycles

For all the areas connected to SDRAM/FCRAM, set these bits to the same RAS active time.

[Bits 1 - 0] W01 and W00 (RAS precharge cycle): RAS precharge cycles

Set these bits to the number of RAS precharge cycles.

Table 4.2 - 24 lists the settings for the RAS precharge cycle.

Table 2-15 RAS precharge cycle

W03	W02	RAS precharge cycle
0	0	1 cycle
0	1	2 cycles
1	0	3 cycles
1	1	4 cycles

For all the areas connected to SDRAM/FCRAM, set these bits to the same RAS precharge cycle.

2.4 Memory setting register (MCRA for SDRAM/FCRAM auto - precharge OFF mode)

This section describes the configuration and the function of memory setting register (MCRA for SDRAM/FCRAM auto - precharge OFF mode).

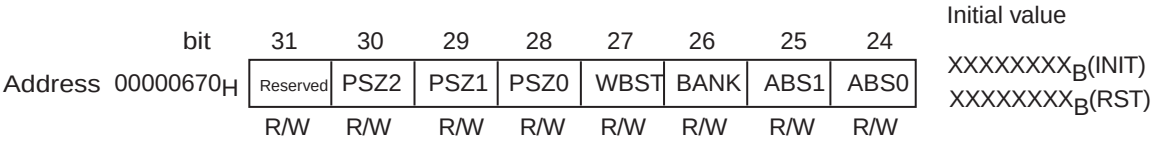
■ Structure of the Memory Setting Register (MCRA for SDRAM/FCRAM auto - precharge OFF mode)

Memory setting register (MCRA for SDRAM/FCRAM auto - precharge OFF mode)

The memory setting register (MCRA: Memory Setting Register for extend type - A for SDRAM/FCRAM auto - precharge OFF mode) is used to make various settings for SDRAM/FCRAM connected to the chip select area.

Figure 2-4 shows the bit configuration of the memory setting register (MCRA for SDRAM/FCRAM auto - precharge OFF mode).

Figure 2-4 Bit configuration of the memory setting register (MCRA for SDRAM/FCRAM auto - precharge OFF mode)



The register serves as the area for making various settings for SDRAM/FCRAM connected to the chip select area for which the access type (TYP3 to TYP0 bits) in the ACR6 and ACR7 registers has been set as in Table 2-25 .

Table 2-25 lists the access type settings (TYP3 to TYP0 bits).

Table 2-16 Access type settings (TYP3 to TYP0 bits)

TYP3	TYP2	TYP1	TYP0	Access type
1	0	0	0	Memory type A:SDRAM/FCRAM(not used auto precharge)

MCRB shares register hardware with MCRA. Updating the MCRA therefore updates the MCRB accordingly.

The following summarizes the functions of individual bits in the memory setting register (MCRA for SDRAM/FCRAM auto - precharge OFF mode).

[Bit 31] Reserved bit

Be sure to set this bit to 0.

[Bits 30 - 28] PSZ2, PSZ1, PSZ0 (Page SiZe): Page size

Set these bits to the page size of SDRAM to be connected.

Table 2-26 lists the settings for the page size of SDRAM connected.

Table 2-17 Settings for the page size of SDRAM

PSZ2	PSZ1	PSZ0	Page size of SDRAM
0	0	0	8-bit column address:A0 to A7(256 memory words)
0	0	1	9-bit column address:A0 to A8(512 memory words)
0	1	0	10-bit column address:A0 to A9(1024 memory words)
0	1	1	11-bit column address:A0 to A9, A11(2048memory words)
1	X	X	Prohibited

[Bit 27] WBST (Write BurST enable): Write burst setting

Set this bit to select whether to burst - write for write access.

Table 2-27 lists the settings for burst write.

Table 2-18 Settings for burst write

WBST	Settings for burst write
0	Single write
1	Burst write

For connecting FCRAM, be sure to set the bit to 1.

FCRAM supports neither burst read nor single write mode.

[BIt 26] BANK (BANK type select): Bank number setting

Set this bit to the number of banks of SDRAM to be connected.

Table 2-28 lists the settings for bank number.

Table 2-19 settings for bank number

BANK	Settings for bank number
0	2 banks
1	4 banks

[Bits 25 - 24] ABS1, ABS0 (Active Bank Select): Setting of active bank number

Set these bits to the maximum number of banks to be made active simultaneously.

Table 2-29 lists the settings for the number of active banks.

Table 2-20 Settings for the number of active banks

ABS1	ABS0	Number of active banks
0	0	1 bank
0	1	2 banks
1	0	3 banks
1	1	4 banks

2.5 Memory setting register (MCRB for FCRAM auto - precharge ON mode)

This section describes the memory setting register (MCRB for FCRAM auto - precharge ON mode).

■ Structure of the Memory Setting Register (MCRB for FCRAM auto - precharge ON mode)

Settings for Memory configuration register (MCRB: Memory Configuration Register for extend type - B for FCRAM auto - precharge ON mode) is used to make various settings for FCRAM connected to the chip select area.

Figure 2-5 shows the bit configuration of the memory setting register (MCRB for FCRAM auto - precharge ON mode).

Figure 2-5 Structure of the Memory Setting Register (MCRB for FCRAM auto - precharge ON mode)

	bit	23	22	21	20	19	18	17	16	Initial value
Address	00000671H	Reserved	PSZ2	PSZ1	PSZ0	WBST	BANK	ABS1	ABS0	XXXXXXXXB(INIT) XXXXXXXXB(RST)
		R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	

The register serves as the area for making various settings for FCRAM connected to the chip select area for which the access type (TYP3 to TYP0 bits) in the ACR6 and ACR7 registers has been set as in Table 2-30 .

Table 2-30 lists the access type settings (TYP3 to TYP0 bits).

Table 2-21 Access type settings (TYP3 to TYP0 bits)

TYP3	TYP2	TYP1	TYP0	Access type
1	0	0	1	Memory type B: FCRAM (used auto precharge)

MCRB shares register hardware with MCRA. Updating the MCRB therefore updates the MCRA accordingly.

The functions are the same as MCRA. Note, however, that the function of the WBST bit is not available to this TYPE setting.

(FCRAM supports neither burst read nor single write mode.)

2.6 I/O Wait Registers for DMAC (IOWR0-3)

This section explains the configuration and functions of the I/O wait registers for DMAC (IOWR0-3).

■ Configuration of the I/O Wait Registers for DMAC (IOWR0-3)

The I/O wait registers for DMAC (IOWR0-3: I/O Wait Register for DMAC 0-3) set various kinds of waits during DMA fly-by access.

Figure 2-6 "Configuration of the I/O wait registers for DMAC (IOWR0, 1, 2, 3)" shows the configuration of the I/O wait registers for DMAC (IOWR0-3).

Figure 2-6 Configuration of the I/O Wait Registers for DMAC (IOWR0-3)

									Initial value		
IOWR0	31	30	29	28	27	26	25	24	INIT	RST	Access
0000 0678H	RYE0	HLD0	WR01	WR00	IW03	IW02	IW01	IW00	xxxxxxxxb	xxxxxxxxb	W/R
IOWR1	23	22	21	20	19	18	17	16			
0000 0679H	RYE1	HLD1	WR11	WR10	IW13	IW12	IW11	IW10	xxxxxxxxb	xxxxxxxxb	W/R
IOWR2	15	14	13	12	11	10	9	8			
0000 067aH	RYE2	HLD2	WR21	WR20	IW23	IW22	IW21	IW20	xxxxxxxxb	xxxxxxxxb	W/R
IOWR3	7	6	5	4	3	2	1	0			
0000 067bH	RYE3	HLD3	WR31	WR30	IW33	IW32	IW31	IW30	xxxxxxxxb	xxxxxxxxb	W/R

■ Functions of Bits in the I/O Wait Registers for DMAC (IOWR0-3)

The following explains the functions of the bits in the I/O wait registers for DMAC.

[Bits 31, 23] RYE0,1 (RDY enable 0,1)

These bits set the wait control, using RDY, of channels 0-3 during DMAC fly-by access.

RYEn	RDY function setting
0	Disable RDY input for I/O access.
1	Enable RDY input for I/O access.

When 1 is set, wait insertion by the RDY pin can be performed during fly-by transfer on the relevant channel. $\overline{IOWR}$ and $\overline{IORD}$ are extended until the RDY pin is enabled. Also, $\overline{RD}$/$\overline{WR0}$-$\overline{WR3}$/$\overline{WR}$ on the memory side are extended synchronously. If the chip select area of the fly-by transfer destination is set to RDY-enabled in the ACR register, wait insertion by the RDY pin can be performed regardless of the RYEn bit of IOWR. When the chip select area of the fly-by transfer destination is set to RDY-disabled in the ACR register, wait insertion by the RDY pin can only be performed during fly-by access if the area is set to RDY-enabled by the RYEn bit on the IOWR side.

[Bits 30,22] HLD0,1 (Hold Wait Control)

These bits control the hold cycle of the read strobe signal on the transfer source access side during DMA fly-by access.

HLDn	Hold wait setting
0	Do not insert a hold extension cycle.
1	Insert a hold extension cycle to extend the read cycle by one cycle.

If 0 is set, the read strobe signal ($\overline{RD}$ for memory -> I/O and $\overline{IORD}$ for I/O -> memory) and the write strobe signal ($\overline{IOWR}$ for memory -> I/O and $\overline{WR0}$-$\overline{WR3}$ and $\overline{WR}$ for I/O -> memory) on the transfer source access side are output at the same timing.

If 1 is set, the read strobe signal is output one cycle longer than the write strobe signal to secure a hold time for data at the transfer source access side when sending it to the transfer destination.

[Bits 29, 28, 21, 20] WR01/00, WR11/00 (I/O Idle Wait)

These bits set the number of idle cycles for continuous access during DMA fly-by access. Table 2-22 "Settings for the Number of I/O Idle Cycles" lists the settings for the number of I/O idle cycles.

Table 2-22 Settings for the Number of I/O Idle Cycles

WRn1	WRn0	Setting of the number of I/O idle cycles
0	0	0 cycle
0	1	1 cycle
1	0	2 cycles
1	1	3 cycles

If one or more cycles is set as the number of idle cycles, cycles equal to the number specified are inserted after I/O access during DMA fly-by access. During the idle cycles, all $\overline{CS}$ and strobe output is negated and the data pin

is set to the high impedance state.

[Bits 27-24, 19-16, 11-8] IW03-00,IW13-10 (I/O Access Wait)

These bits set the number of auto-wait cycles for I/O access during DMA fly-by access.

Table 2-23 "Settings for the Number of I/O Wait Cycles" lists the settings for the number of I/O wait cycles.

Table 2-23 Settings for the Number of I/O Wait Cycles

IWn3	IWn2	IWn1	IWn0	Number of I/O wait cycles
0	0	0	0	0 cycle
0	0	0	1	1 cycle
...				...
1	1	1	1	15 cycle

Because data is synchronized between the transfer source and transfer destination, the I/O side setting of the IWnn bits and the wait setting for the fly-by transfer destination (such as memory), whichever is larger, is used as the number of wait cycles to be inserted. Consequently, more wait cycles than specified by the IWnn bits may be inserted.

2.7 Chip Select Enable Register (CSER)

Because data is synchronized between the transfer source and transfer destination, the I/O side setting of the IWnn bits and the wait setting for the fly-by transfer destination (such as memory), whichever is larger, is used as the number of wait cycles to be inserted. Consequently, more wait cycles than specified by the IWnn bits may be inserted.

■ Configuration of the Chip Select Enable Register (CSER)

The chip select enable register (CSER: Chip Select Enable register) enables and disables each chip select area.

Figure 2-7 "Configuration of the Chip Select Enable Register (CSER)" shows the configuration of the chip select enable register (CSER).

Figure 2-7 Configuration of the Chip Select Enable Register (CSER)

									Initial value		
	31	30	29	28	27	26	25	24	INIT	RST	Access
00000680_H	CSE7	CSE6	CSE5	CSE4	CSE3	CSE2	CSE1	CSE0	00000001_B	00000001_B	R/W

■ Functions of Bits in the Chip Select Enable Register (CSER)

The following explains the functions of the bits in the chip select enable register (CSER).

[Bits 31-24] CSE7-0 (Chip Select Enable 0-7)

These bits are the chip select enable bits for $\overline{CS0}$-$\overline{CS7}$.

The initial value is 00000001_B, which enables only the CS0 area.

When 1 is written, a chip select area operates according to the settings of ASR0-7, ACR0-7, and AWR0-7.

Before setting this register, be sure to make all settings required for the corresponding chip select areas.

CSE7-0	Area control
0	Disable
1	Enable

Table 2-24 " $\overline{CSn}$ Corresponding to the Chip Select Enable Bits" lists the corresponding $\overline{CSn}$ for the chip select enable bits.

Table 2-24 $\overline{CSn}$ Corresponding to the Chip Select Enable Bits

CSE bit	Corresponding $\overline{CSn}$
Bit 24: CSE0	$\overline{CS0}$
Bit 25: CSE1	$\overline{CS1}$
Bit 26: CSE2	$\overline{CS2}$
Bit 27: CSE3	$\overline{CS3}$
Bit 28: CSE4	$\overline{CS4}$
Bit 29: CSE5	$\overline{CS5}$
Bit 30: CSE6	$\overline{CS6}$
Bit 31: CSE7	$\overline{CS7}$

2.8 Cache Enable Register (CHER)

This section explains the configuration and functions of the cache enable register (CHER).

■ Configuration of the Cache Enable Register (CHER)

The cache enable register (CHER: CacHe Enable Register) controls the transfer of data read from each chip select area.

Figure 2-8 "Configuration of the Cache Enable Register (CHER)" shows the configuration of the cache enable register (CHER).

Figure 2-8 Configuration of the Cache Enable Register (CHER)

									Initial value		
	23	22	21	20	19	18	17	16	INIT	RST	Access
00000681_H	CHE7	CHE6	CHE5	CHE4	CHE3	CHE2	CHE1	CHE0	11111111_B	11111111_B	R/W

■ Functions of Bits in the Cache Enable Register (CHER)

The following explains the functions of the bits in the cache enable register (CHER).

[Bits 23-16] CHE7-0 (Cache Enable 7-0)

These bits enable and disable each chip select area for transfers to the built-in cache.

CHEn	Cache area setting
0	Not a cache area (data read from the applicable area is not saved in the cache)
1	Cache area (data read from the applicable area is saved in the cache)

2.9 Pin/Timing Control Register (TCR)

This section explains the configuration and functions of the pin/timing control register and its function.

■ Configuration of the Pin/Timing Control Register (TCR)

The pin/timing control register (TCR: Terminal and Limiting Control Register) controls the functions related to the general external bus interface controller, such as the setting of common pin functions and timing control.

Figure 2-9 "Configuration of the Pin/Timing Control Register (TCR)" shows the configuration of the pin/timing control register (TCR).

Figure 2-9 Configuration of the Pin/Timing Control Register (TCR)

									Initial value		
	7	6	5	4	3	2	1	0	INIT	RST	Access
00000683_H	BREN	PSUS	PCLR	Reserved	Reserved	Reserved	RDW1	RDW0	00000000_B	$0000xxxx_B$	R/W

■ Functions of Bits in the Pin/Timing Control Register (TCR)

The following explains the functions of the bits in the pin/timing control register (TCR).

[Bit 7] BREN (BRQ Enable)

This bit enables BRQ pin input and external bus sharing.

BREN	BRQ input enable setting
0	No bus sharing by BRQ/$\overline{\text{BGRNT}}$. BRQ input is disabled.
1	Bus sharing by BRQ/$\overline{\text{BGRNT}}$. BRQ input is enabled.

In the initial state (0), BRQ input is ignored. When 1 is set, the bus is made open (control with high impedance) and $\overline{\text{BGRNT}}$ is activated (L level is output) when the bus is ready to be made open after the BRQ input becomes H level.

[Bit 6] PSUS (Prefetch suspend)

This bit controls temporary stopping of prefetch to all areas.

PSUS	Prefetch control
0	Enable prefetch
1	Suspend prefetch

If 1 is set, no new prefetch operation is performed before 0 is written. Since during this time the contents of the prefetch buffer are not deleted unless a prefetch buffer occurs, clear the prefetch buffer using the PCLR bit function (bit 5) before restarting prefetch.

[Bit 5] PCLR (Prefetch buffer clear)

This bit completely clears the prefetch buffer.

PCLR	Prefetch buffer control
0	Normal state
1	Clear the prefetch buffer.

If 1 is written, the prefetch buffer is cleared completely. When clearing is completed, the bit value automatically returns to 0. Interrupt (set to 1) the prefetch by the PSUS bit (bit 6) and then clear the buffer (It is also possible to write 11_B to both the PSUS and PCLR bits).

[Bit 4-2] Reserved

This bit is reserved. Be sure to set it to 0.

[Bits 1,0] RDW1,0 (Reduce Wait cycle)

These bits instruct all chip select areas and fly-by I/O channels to reduce only the number of auto-wait cycles in the auto-access cycle wait settings uniformly while the AWR register settings are retained unchanged. The settings for idle cycles, recovery cycles, setup, and hold cycles are not affected. Table 2-25 "Settings for Wait Cycle Reduction" lists the settings for the wait cycle reduction for combinations of these bits.

Table 2-25 Settings for Wait Cycle Reduction

RDW1	RDW0	Wait cycle reduction
0	0	Normal wait (AWR0-7 settings)
0	1	1/2 (1-bit shift to the right) of the AWR0-7 settings
1	0	1/4 (2-bit shift to the right) of the AWR0-7 settings
1	1	1/8 (3-bit shift to the right) of the AWR0-7 settings

The purpose of this function is to prevent an excessive access cycle wait during operation on a low-speed clock (for example, when the base clock is switched to low speed or the frequency division ratio setting of the external bus clock is large).

To reset the wait cycle in these cases, each of the AWRs must usually be rewritten one at a time. However, when the RDW1/0 bit function is used, the access cycle wait is reduced for all of the AWRs in a single operation while all of the other high-speed clock settings in each register are retained.

Before returning the clock to high speed, be sure to reset the RDW1/0 bits to 00_B.

2.10 Refresh Control Register (RCR)

This section describes the bit configuration and functions of the refresh control register (RCR).

■ Structure of the Refresh Control Register (RCR)

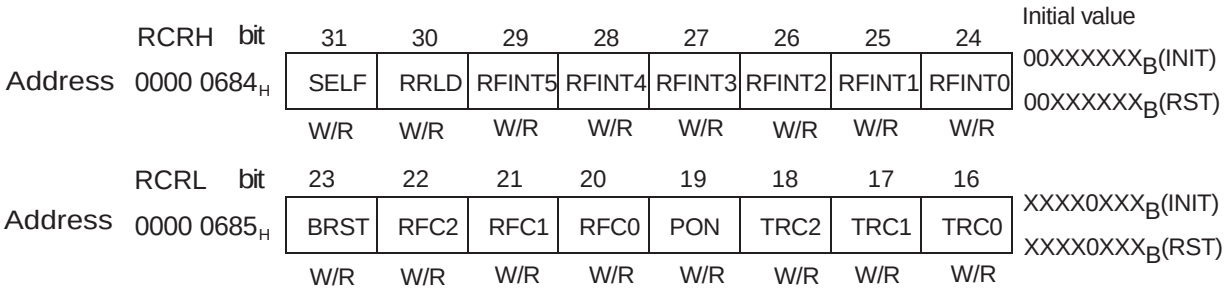
The refresh control register (RCR) is used to make various refresh control settings for SDRAM.

The setting of this register is meaningless as long as SDRAM control is not set for any area, in that case the register value must not be updated from the initial state.

When read by a Read - modify - Write instruction, the SELF, RRLD, and PON bits always return to 0.

Figure 2-10 shows the bit configuration of the refresh control register (RCR).

Figure 2-10 Structure of the Refresh Control Register (RCR)



■ Bit Functions of the Refresh Control Register (RCR)

The following summarizes the functions of individual bits in the refresh control register (RCR).

[Bit 31] SELF (SELF refresh assert): Self - refresh control

This bit is used to control the self - refresh mode for memory that supports the self - refresh mode.

Table 4.2-42 lists the settings for self - refresh control.

Table 2-26 Settings for self - refresh control

SELF	Self - refresh control
0	Auto - refresh or power - down
1	Transition to self-refresh mode

Setting the bit to 1 performs a self - refresh after issuing the SELF command. Writing 0 terminates the self - refresh mode.

To hold the contents of SDRAM when putting the LSI into stop mode, use this bit to enter the self - refresh mode before entering the stop mode. At this time, centralized refreshing is performed before transition to the self - refresh mode. External access requests generated before it is completed are put on hold. The mode transits to the stop mode.

The device is released from the self - refresh mode either when 0 is written to this bit or access to SDRAM occurs. At this time, centralized refreshing is performed immediately after the release. If external access such as SDRAM access is attempted, therefore, the external access request is kept on hold and the CPU stops operation for a while. An attempt to put the LSI into the stop mode when it cannot enter the self - refresh mode causes it to directly enter the power save mode, resulting in corruption of data in SDRAM.

When read by a Read - modify - Write instruction, the SELF, RRLD, and PON bits always return to 0.

[Bit 30] RRLD (Refresh counter ReLoaD): Refresh counter start control

This bit is used to start and reload the fresh counter.

Table 4.2-43 shows the function of refresh counter startup control.

Table 2-27 Function of refresh counter startup control

RRLD	Refresh counter startup control
0	Disable (no operation)
1	Execute auto - refreshing once and reload the RFINT value.

The refresh counter is inactive in the initial state.

If this bit is set to 1 in this state, all the SDRAM areas currently enabled in the CSER are auto - refreshed either once in distributed refresh mode or the RFC - specified number of times in centralized refresh mode. After that, the values in the RFINT5 to RFINT0 bits are reloaded.

From then on, the refresh counter starts being decremented. Whenever the counter causes an underflow from 000000_B, repeatedly, the values in the RFINT5 to RFINT0 bits are reloaded while at the same time auto - refreshing is performed once.

The bit returns to 0 upon completion of reloading.

To stop auto - refreshing, write 000000_B to the RFINT5 to RFINT0 bits.

When read by a Read - modify - Write instruction, the bit always returns a zero.

[Bits 29 - 24] RFINT5 to RFINT0 (ReFresh INTerval): Auto - refresh interval

Set these bits to the interval for automatic refreshing.

The auto - refresh interval can be obtained for distributed refresh mode {(REFINT5 - REFINT0 value) x 32 x (external bus clock cycle)} or for centralized refresh mode {(REFINT5 - REFINT0 value) x 32 x (RFC specified number of times) x (external bus clock cycle)}

Calculate the design value in consideration of the maximum RAS active time.

The refresh counter keeps on being decremented even while the auto - refresh command is being issued.

[Bit 23] BRST (BuRST refresh select): Burst refresh control

This bit is used to control the operation mode for auto - refreshing.

Table 4.2-44 shows the function of burst refresh control.

Table 2-28 Function of burst refresh control

BRST	Burst refresh control
0	Distributed refresh (Auto - refresh is activated at intervals.)
1	Burst refresh (Auto - refresh is activated repeatedly at one time.)

When distributed refreshing is set, the auto - refresh command is issued once at every refresh interval.

When burst refreshing is set, the auto - refresh command is issued continuously for the number of times set in the refresh counter at every refresh interval.

[Bits 22 - 20] RFC2, RFC1, RFC0 (ReFresh Count): Refresh count

Set these bits to the number of times a refresh must be performed to refresh all SDRAM.

Table 4.2-45 shows the number of times to refresh.

Table 2-29 Number of times to refresh

RFC2	RFC1	RFC0	Number of times to refresh
0	0	0	256
0	0	1	512
0	1	0	1024
0	1	1	2048
1	0	0	4096
1	0	1	8192
1	1	0	Setting prohibited
1	1	1	Refresh prohibited

The number of times to refresh specified here is the number of times centralized refreshing is performed before and after transition to the self - refresh mode. When burst refreshing has been selected with the BRST bit, the number of times to refresh is also the number of times the refresh command is issued at every refresh interval.

[Bit 19] PON (Power ON): Power - on control

This bit is used to control the SDRAM (FCRAM) power - on sequence.

Table 4.2-46 shows the function of power - on control.

Table 2-30 Function of power - on control

PON	Power-on control
0	Disabled (no-operation)
1	Start power-on sequence

Writing 1 to the PON bit starts the SDRAM power - on sequence.

Before starting the power - on sequence, be sure to set the relevant registers such as AWR, MCRA(B), and CSER.

This bit returns to 0 as soon as the power - on sequence is started.

When enabling the PON bit, set RFINT and enable RRLD to activate the refresh counter.

Refreshing is not performed only with the PON bit.

Do not enable this bit along with the SELF bit.

When read by a Read - modify - Write instruction, the bit always returns 0.

[Bits 18 - 16] TRC2, TRC1, TRC0 (Time of Refresh Cycle): Refresh cycle (tRC)

These bits set the refresh cycle (tRC).

Table 4.2-47 lists the settings for the refresh cycle (tRC).

Table 2-31 Settings for the refresh cycle (tRC)

TRC2	TRC1	TRC0	Refresh cycle (tRC)
0	0	0	4
0	0	1	5
0	1	0	6
0	1	1	7
1	0	0	8
1	0	1	9
1	1	0	10
1	1	1	11

3. Setting Example of the Chip Select Area

In the external bus interface, a total of eight chip select areas can be set.

This section presents an example of setting the chip select area.

■ Example of Setting the Chip Select Area

The address space of each area can be placed, in units of a minimum of 64 KB, anywhere in the 4 GB space using ASR0-7 (Area Select Registers) and ACR0-7 (Area Configuration Registers). When bus access is made to an area specified by these registers, the corresponding chip select signals ($\overline{CS0}$-$\overline{CS7}$) are activated (L output) during the access cycle.

● Example of setting ASRs and ASZ3-0

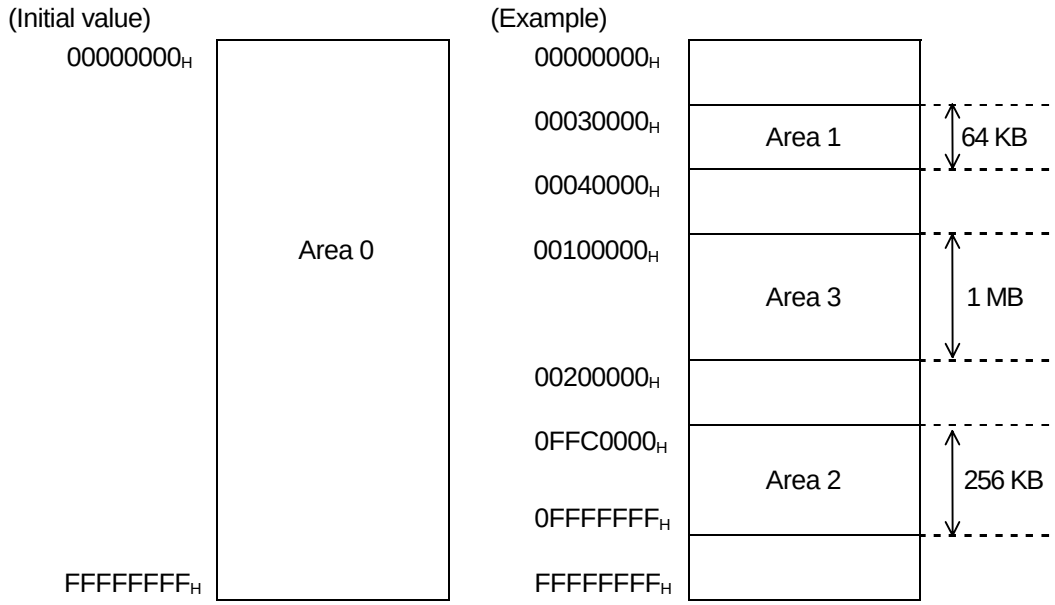
- ASR1=0003_H ACR1 ASZ3-0=0000_B: Chip select area 1 is assigned to 00030000_H to $0003FFFF_H$.
- ASR2=$0FFC_H$ ACR2 ASZ3-0=0010_B: Chip select area 2 is assigned to $0FFC0000_H$ to 10000000_H.
- ASR3=0011_H ACR3 SZ3-0=0100_B: Chip select area 3 is assigned to 00100000_H to 00200000_H.

Since at this point 1 MB is set for bits ASZ3-0 of the ACR, the unit for boundaries 1 MB and bits 19-16 of ASR3 are ignored. Before there is any writing to ACR0 after a reset, 00000000_H-$FFFFFFFF_H$ is assigned to chip select area 0.

Set the chip select areas so that there is no overlap.

Figure 3-1 "Example of Setting the Chip Select Area" shows an example of setting the chip select area.

Figure 3-1 Example of Setting the Chip Select Area



4. Endian and Bus Access

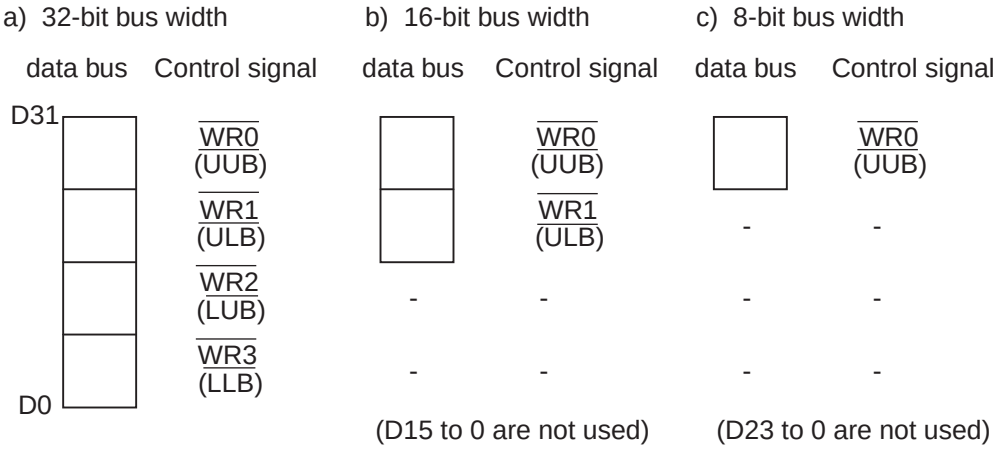
There is a one-to-one correspondence between the $\overline{WR0}$-$\overline{WR3}$ control signal and the byte location regardless of the endian method (big or little) and the data bus width. The following summarizes the location of bytes on the data bus of the MB91460 series used according to the specified data bus width and the corresponding control signal for each bus mode.

■ Relationship between Data Bus Width and Control Signal

This section summarizes the location of bytes on the data bus used according to the specified data bus width and the corresponding control signal for each bus mode.

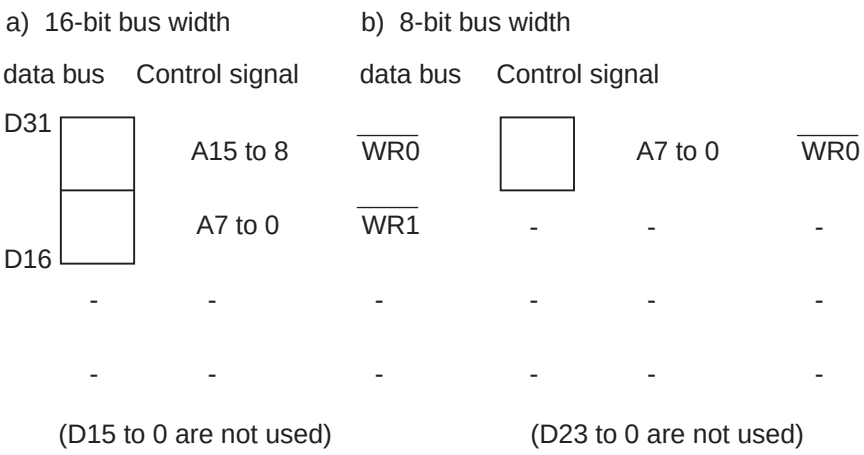
- Ordinary bus interface

Figure 4-1 Data Bus Width and Control Signal on the Ordinary Bus Interface



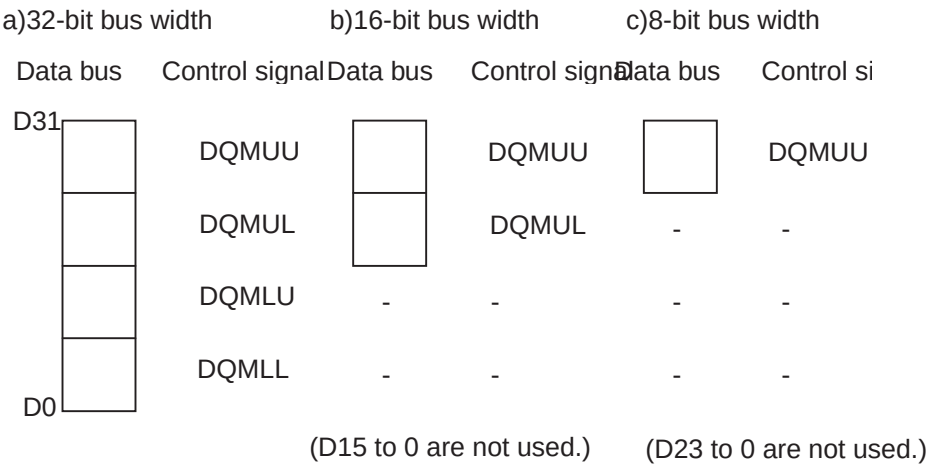
- Time division I/O interface

Figure 4-2 Data Bus Width and Control Signal in the Time Division I/O Interface



- SDRAM Interface

Figure 4-3 Data bus width of the SDRAM (FCRAM) interface and its control signals



4.1 Big Endian Bus Access

With the exception of the CS0 area of the MB91460 series, either the big endian method or the little endian method can be selected for each chip select area. If 0 is set for the LEND bit of the ACR register, the area is treated as big endian. The MB91460 series is normally big endian and performs external bus access.

■ Data Format

The relationship between the internal register and the external data bus is as follows:

- Word access (when LD/ST instruction executed)

Figure 4-4 Relationship between Internal Register and External Data Bus for Word Access

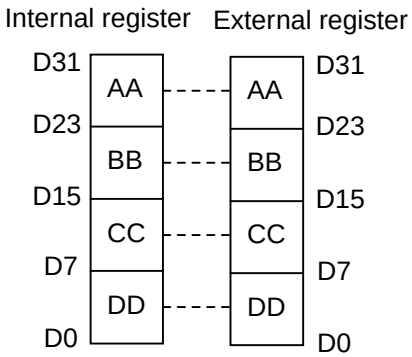


Figure 4-5 Relationship between the Internal Register and External Data Bus for Halfword Access

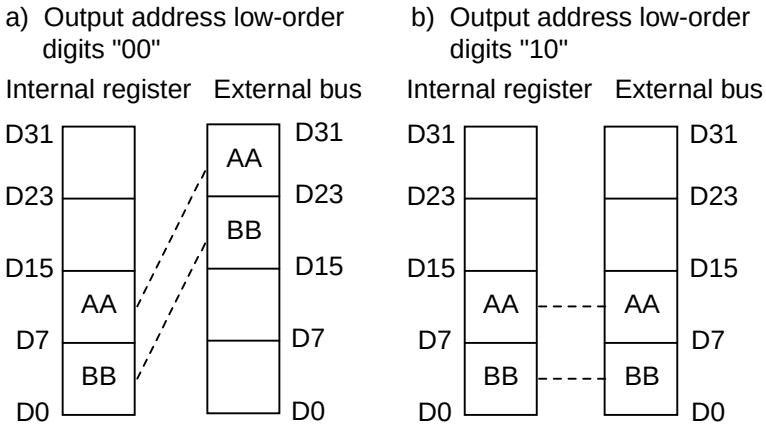
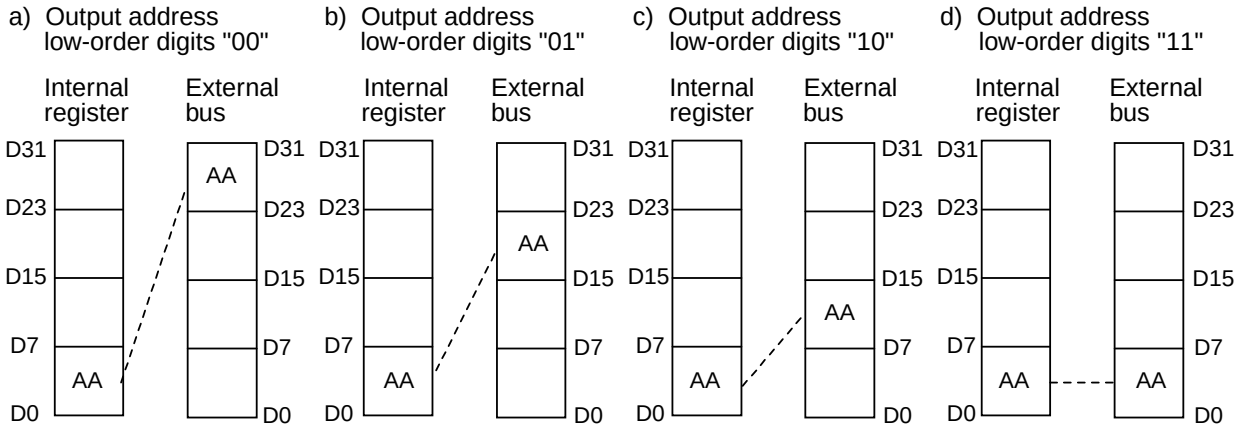


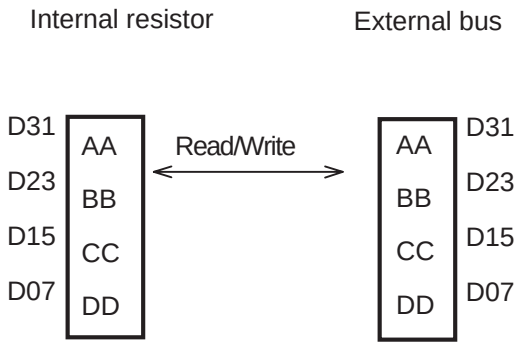
Figure 4-6 Relationship between Internal Register and External Data Bus for Byte Access



■ Data Bus Width

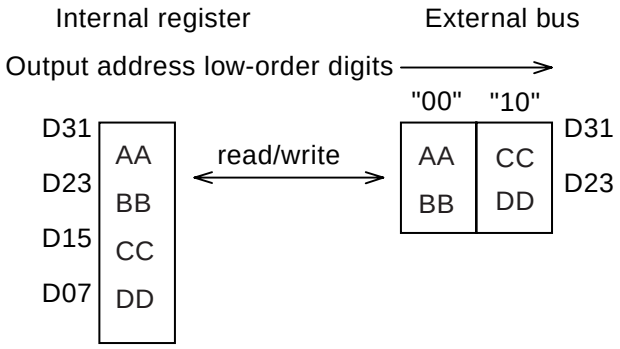
- 32-bit bus width

Figure 4-7 Relationship between Internal Register and External Bus Having 32-Bit Bus Width



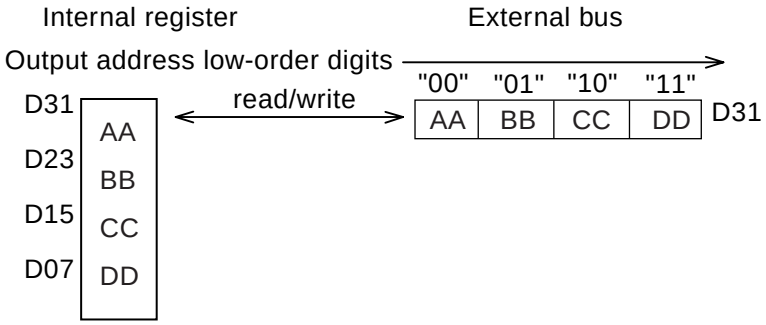
- 16-bit bus width

Figure 4-8 Relationship between Internal Register and External Bus Having 16-Bit Bus Width



- 8-bit bus width

Figure 4-9 Relationship between Internal Register and External Bus having 8-Bit bus Width



■ External Bus Access

Figure 4-11 "External bus Access for 16-Bit Bus Width" and Figure 4-12 "External Bus Access for 8-Bit Bus Width" show external bus access (16-bit/8-bit bus width) separately for word, halfword, and byte access. The following items are included in Figure 4-11 "External bus Access for 16-Bit Bus Width" and Figure 4-12 "External Bus Access for 8-Bit Bus Width":

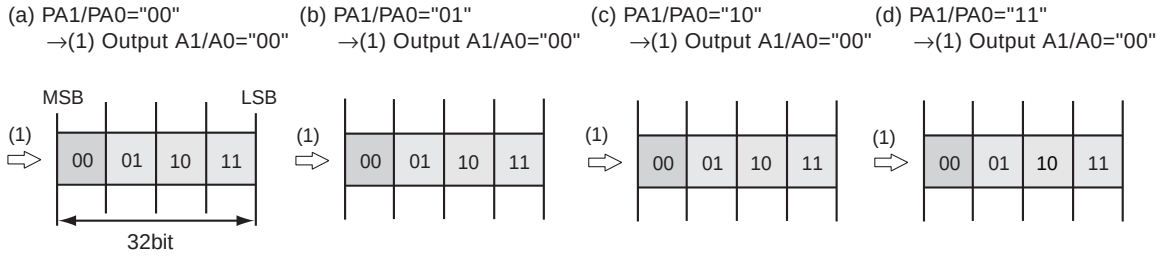
- Access byte location
- Program address and output address
- Bus access count

The MB91460 series does not detect misalignment errors.

Therefore, for word access, the lower two bits of the output address are always 00 regardless of whether 00, 01, 10, or 11 is specified as the lower two bits by the program. For halfword access, the lower two bits of the output address are 00 if the lower two bits specified by the program are 00 or 01, and are 10 if 10 or 11.

● 32-bit bus width

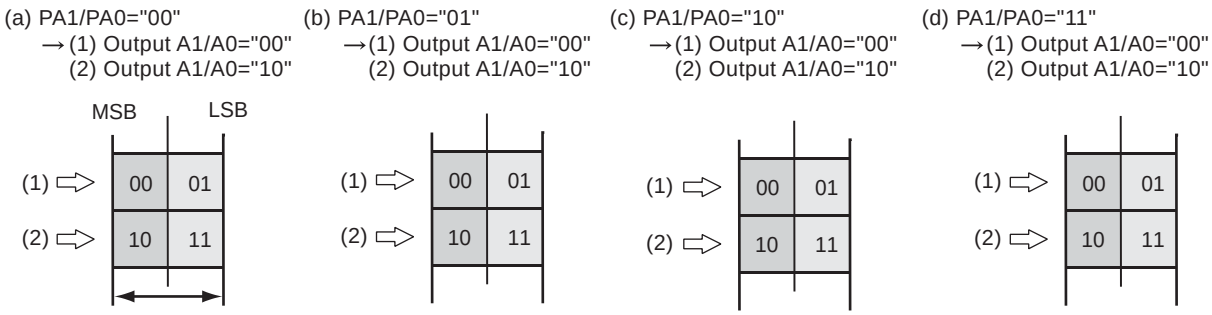
Figure 4-10 External bus Access for 32-Bit Bus Width



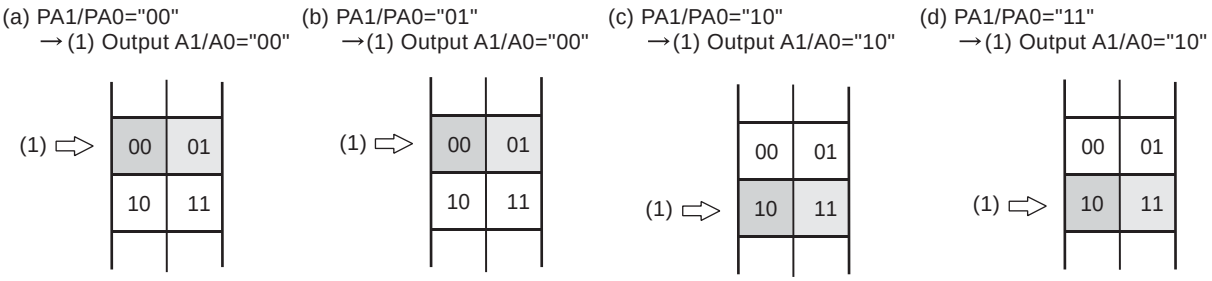
● 16-bit bus width

Figure 4-11 External bus Access for 16-Bit Bus Width

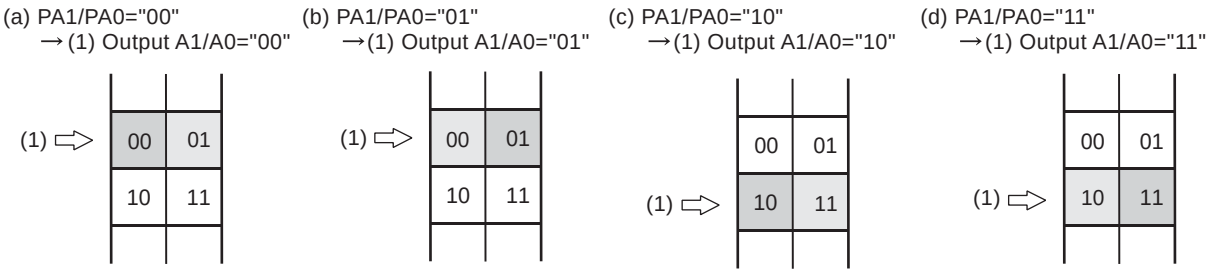
(A) Word access



(B) Halfword access

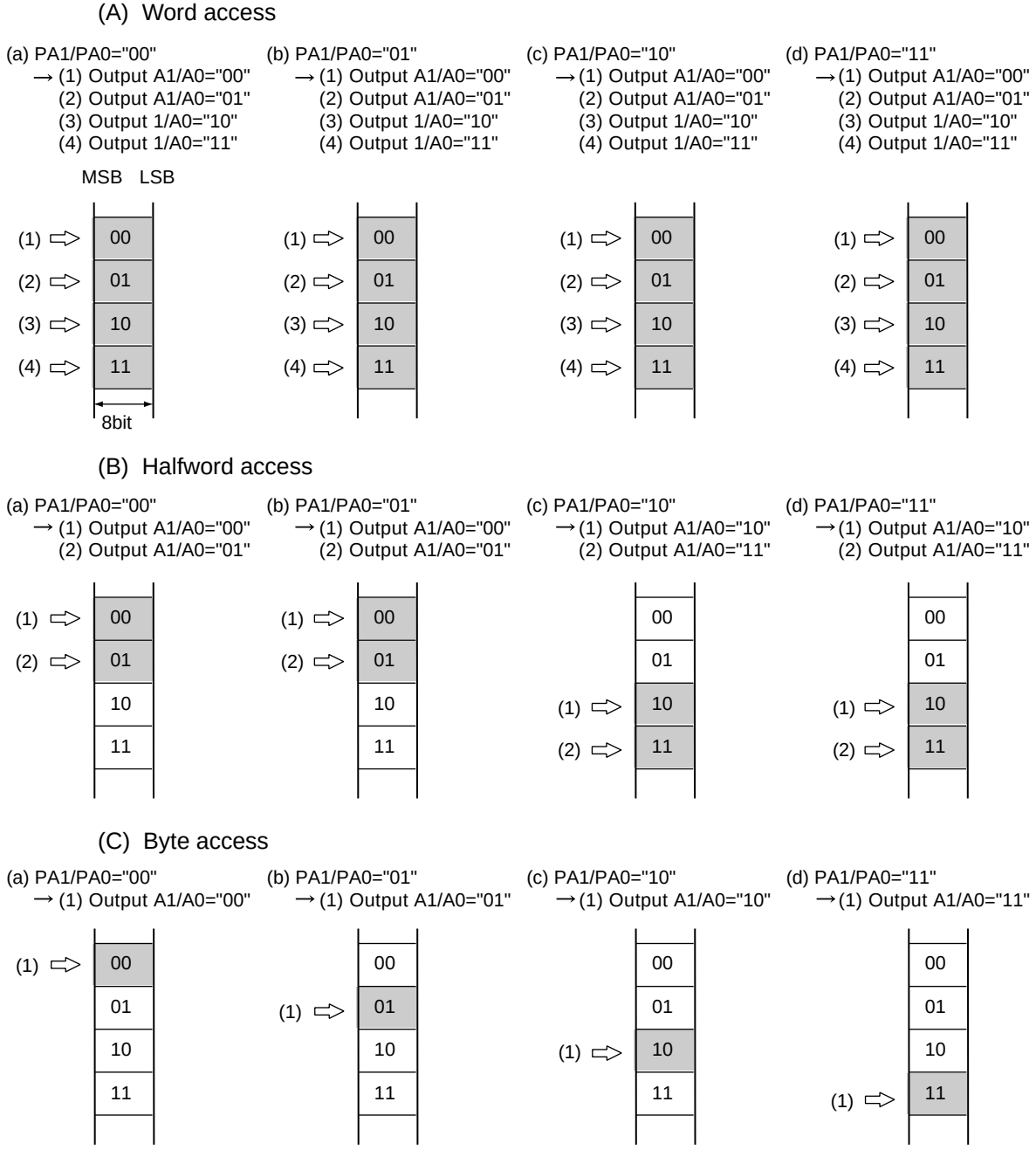


(C) Byte access



- 8-bit bus width

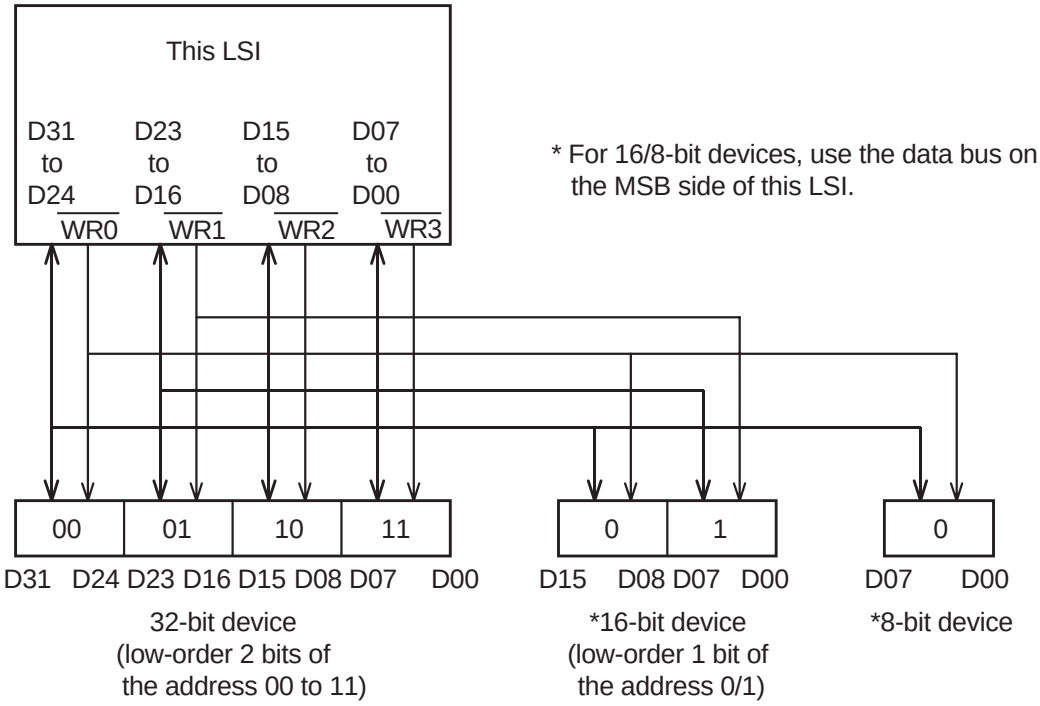
Figure 4-12 External Bus Access for 8-Bit Bus Width



■ Example of Connection with External Devices

Figure 4-13 "Example of Connecting the MB91460 Series to External Devices" shows an example of connection the MB91460 series to external devices.

Figure 4-13 Example of Connecting the MB91460 Series to External Devices



4.2 Little Endian Bus Access

Little endian (LER) external bus access is performed for an area for which the little endian method is set.

Little endian bus access on the MB91460 series is implemented by using the bus access operation used for the big endian method. Basically, the order of output addresses and control signal output are the same as for the big endian method and the byte locations on the data bus are swapped in accordance with the bus width.

Note that, when a connection is made, the big endian area and the little endian area must be kept physically separate.

■ Differences between Little Endian and Big Endian

The following explains the differences between little endian and big endian.

The order of addresses that are output is the same for little endian and big endian.

The data bus control signal used for 32/16/8-bit bus width is the same for little endian and big endian.

- Word access

The byte data on the MSB side for big endian address 00 becomes byte data on the LSB side when the little endian method is used.

For a word address, the locations of all four bytes in the word are reversed:

00 -> 11, 01 -> 10, 10 -> 01, 11 -> 00

- Halfword access

The byte data on the MSB side for the big endian address 0 becomes byte data on the LSB side when the little endian method is used.

For halfword access, the byte locations of two bytes are reversed.

0 -> 1, 1 -> 0

- Byte access

There is no difference between little endian and big endian.

■ Restrictions on the Little Endian Area

- If prefetch is enabled for a little endian area, always use word access to access the area. If data written to the prefetch buffer is accessed with any length other than word length, the correct endian conversion is not performed and the wrong data will be read. The reason is hardware restrictions related to the endian conversion mechanism.
- Do not place any instruction code in a little endian area.

■ Data Format

The relationship between the internal register and external data bus is as follows:

Figure 4-14 Relationship between the Internal Register and External Data Bus for Word Access

(1) Word access (when executing the LD/ST instructions)

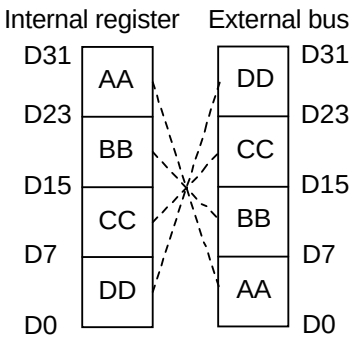
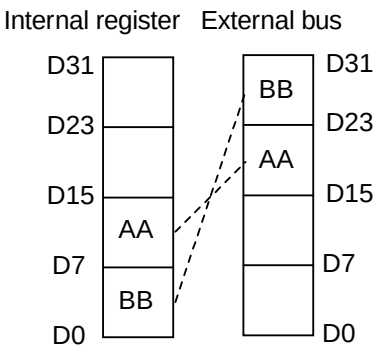


Figure 4-15 Relationship between Internal Register and External Data Bus for Halfword Access

(2) Halfword access (when executing the LDUH/STH instructions)

a) Output address low-order digits "00"



b) Output address low-order digits "10"

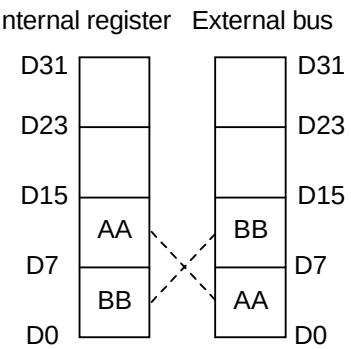
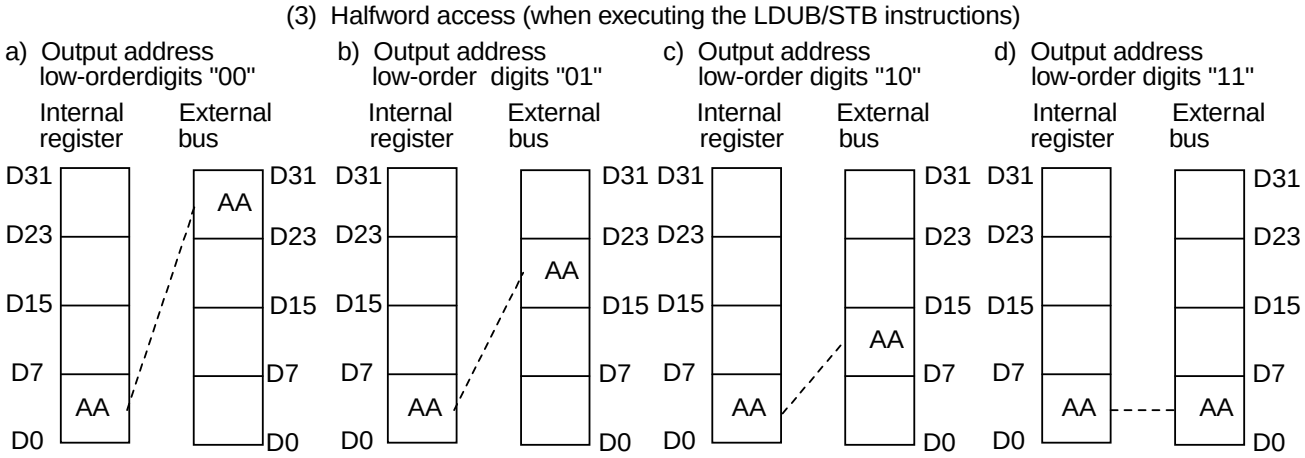


Figure 4-16 Relationship between Internal Register and External Data Bus for Byte Access

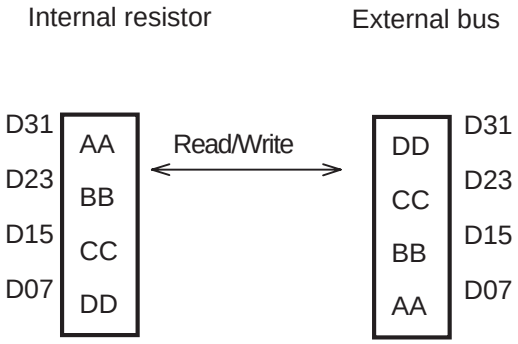


■ Data Bus Width

The following shows the relationships between the internal register and external data bus for each data bus width.

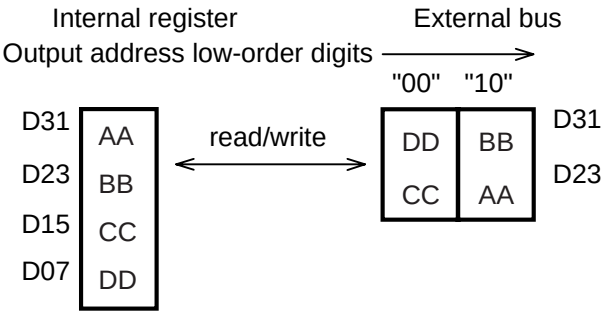
- 32-bit bus width

Figure 4-17 Relationship between Internal Register and External Bus Data for 32-bit Bus Width



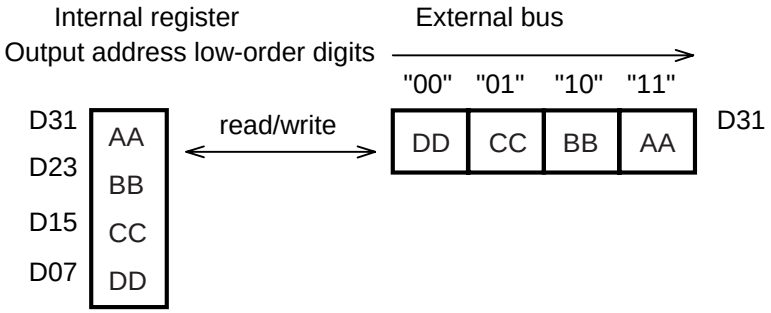
- 16-bit bus width

Figure 4-18 Relationship between Internal Register and External Bus Data for 16-bit Bus Width



● 8-bit bus width

Figure 4-19 Relationship between the Internal Register and External Data Bus in the 8-bit Bus Width

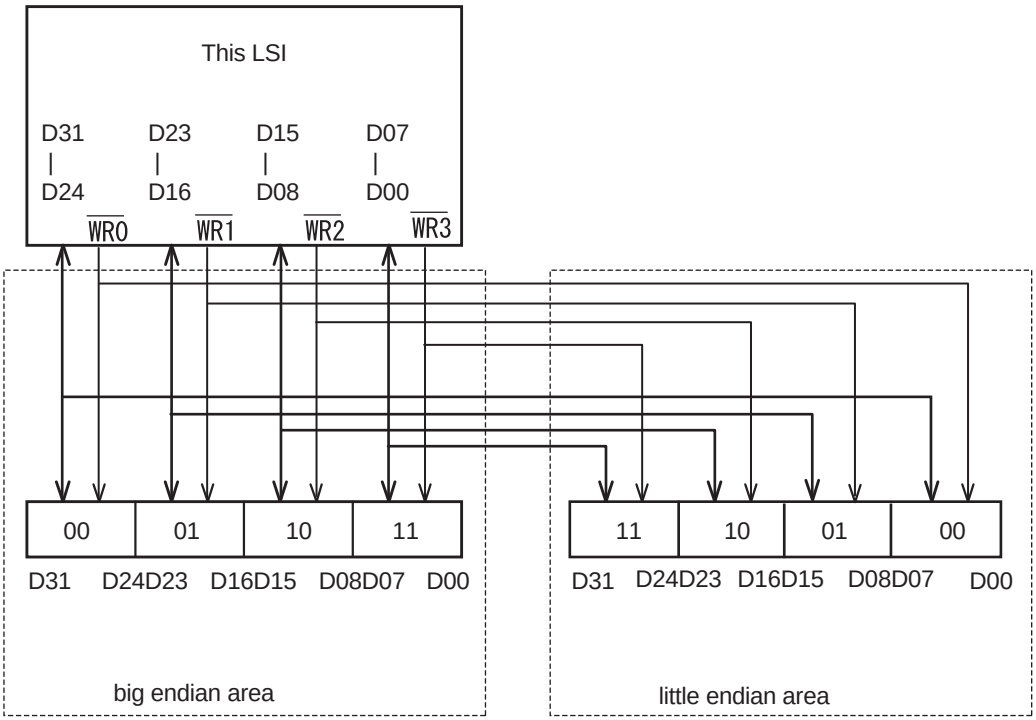


■ Examples of Connection with External Devices

The following shows examples of connecting the MB91460 series to external devices for each bus width.

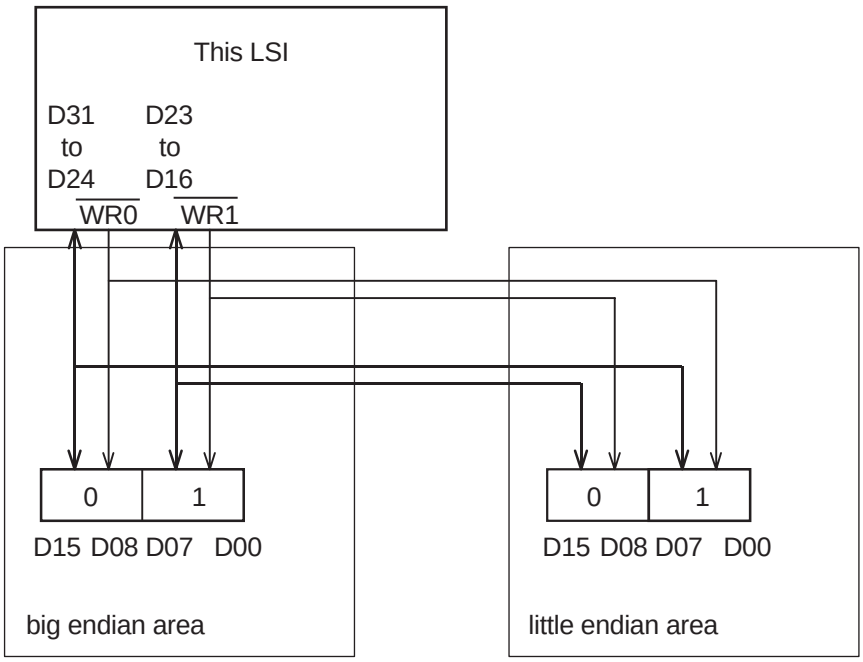
● 32-bit bus width

Figure 4-20 Example of Connecting the MB91460 Series to External Devices (32-Bit Bus Width)



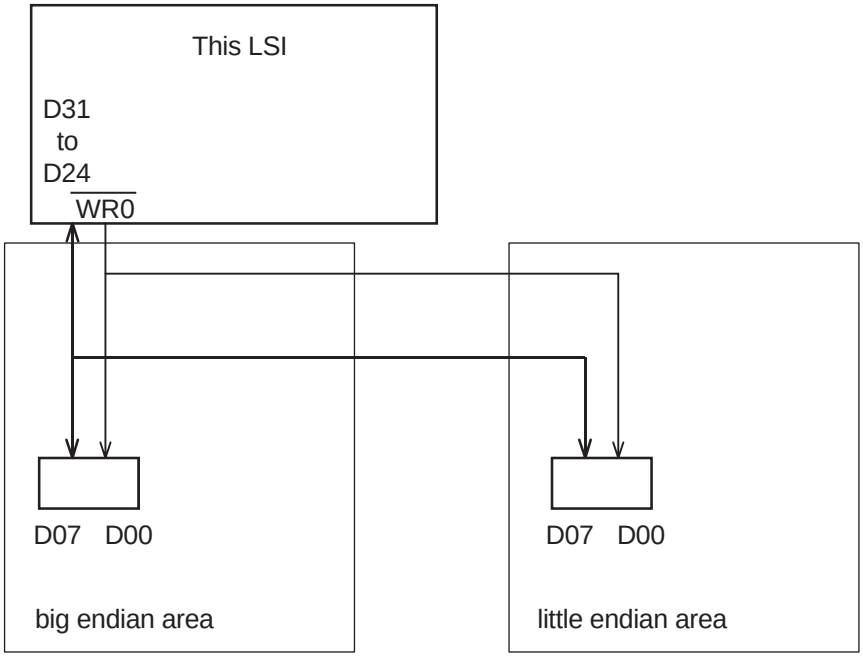
● 16-bit bus width

Figure 4-21 Example of Connecting the MB91460 Series to External Devices (16-Bit Bus Width)



● 8-bit bus width

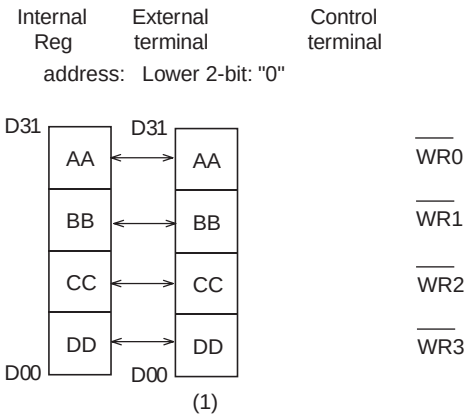
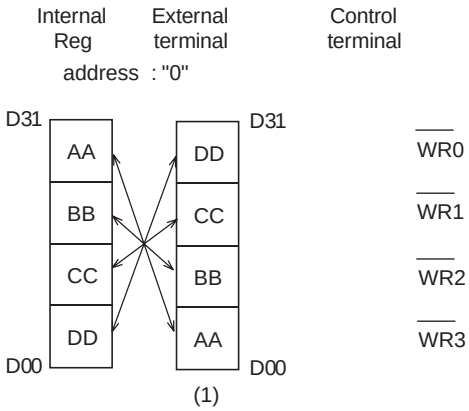
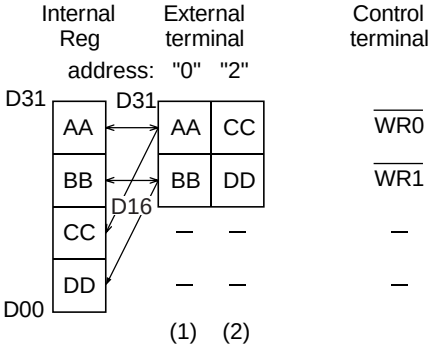
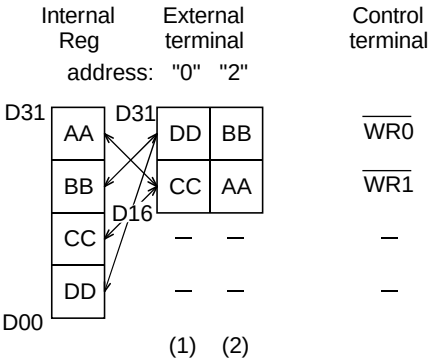
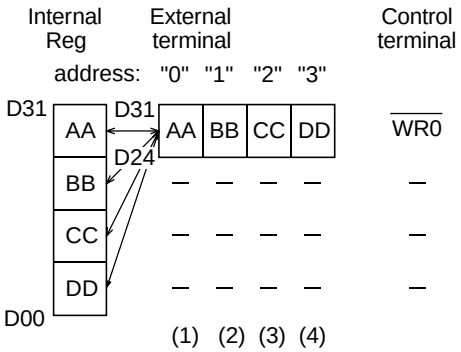
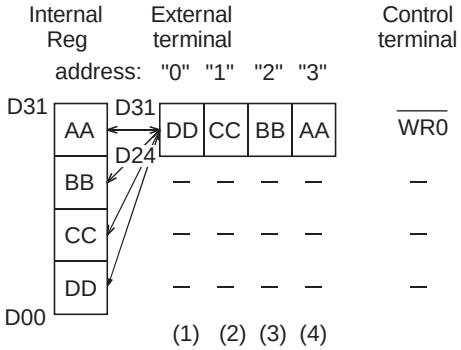
Figure 4-22 Example of Connecting the MB91460 Series to External Devices (8-Bit Bus Width)



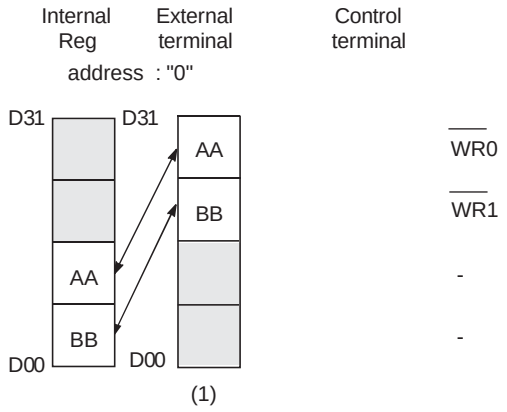
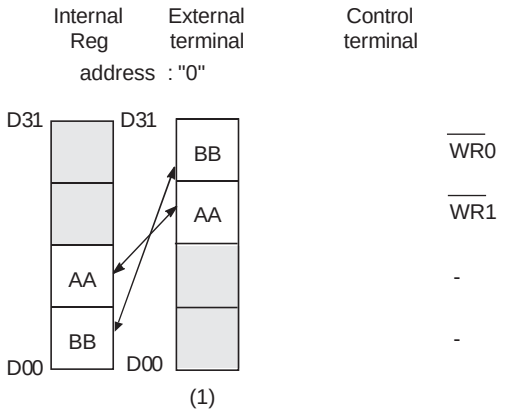
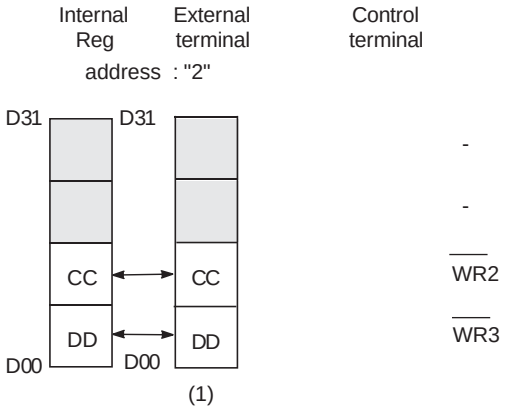
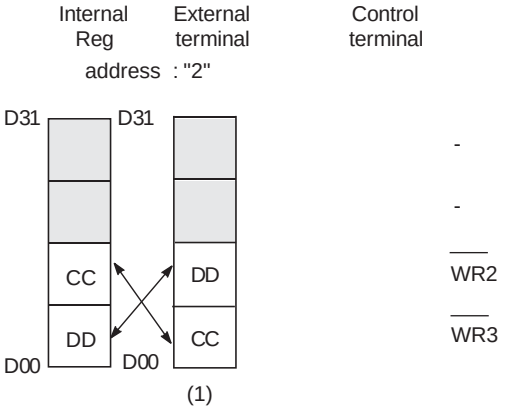
4.3 Comparison of Big Endian and Little Endian External Access

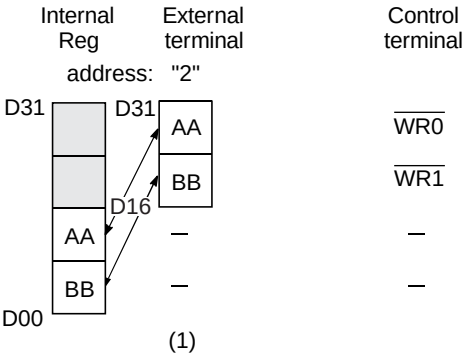
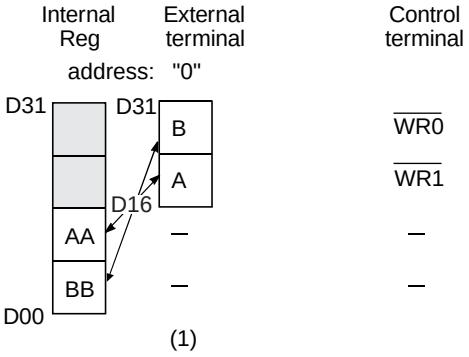
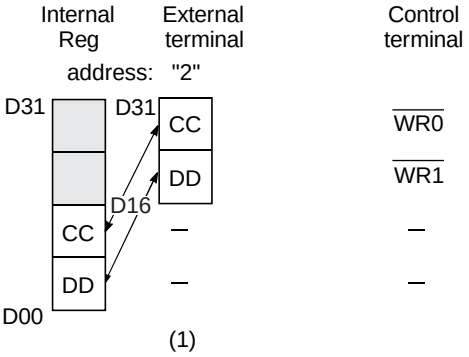
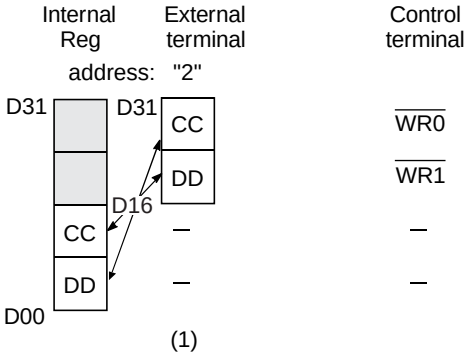
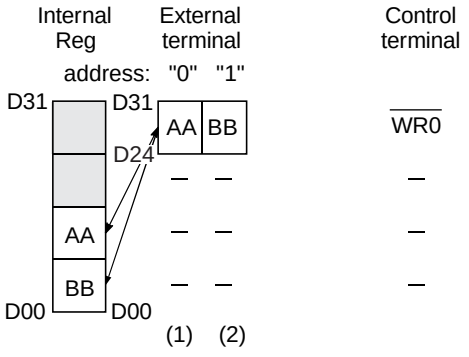
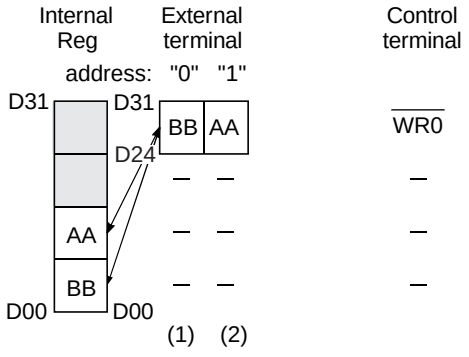
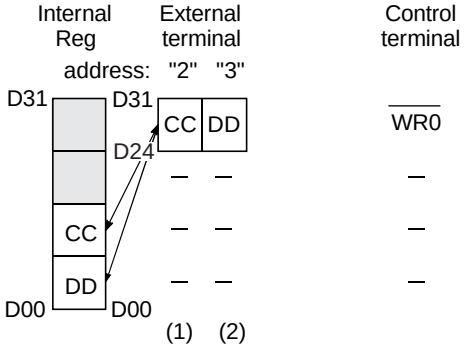
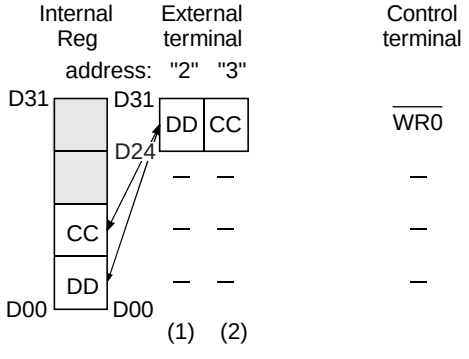
This section shows a comparison of big endian and little endian external access in word access, halfword access, and byte access for each bus width.

■ Word Access

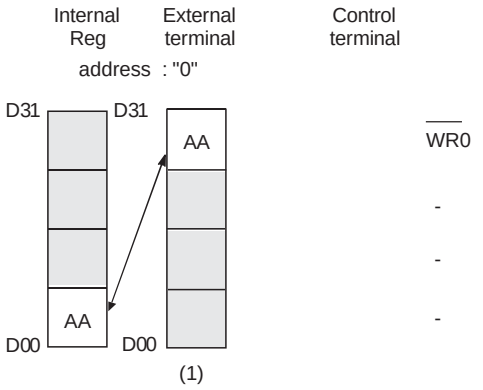
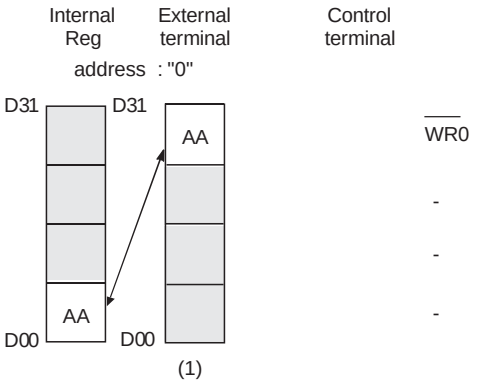
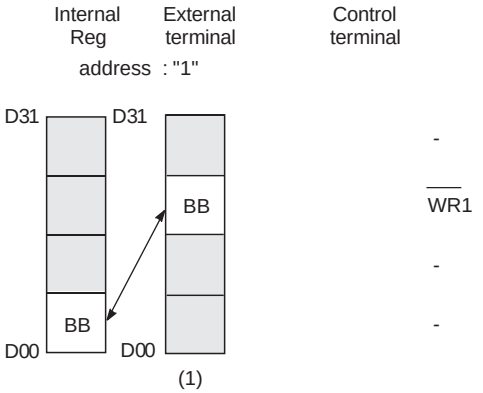
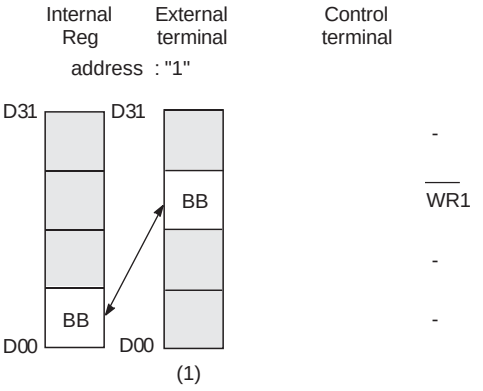
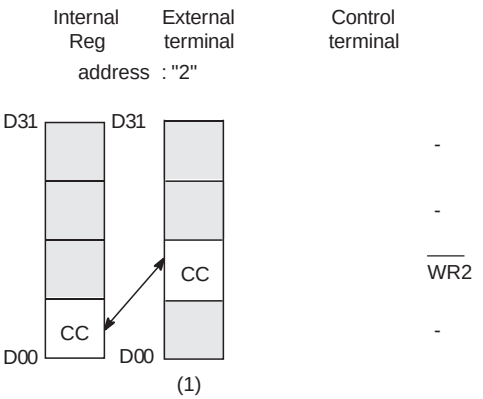
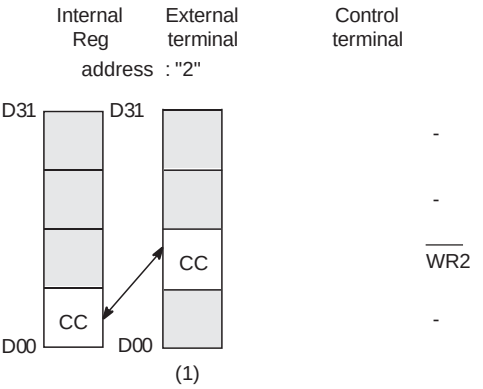
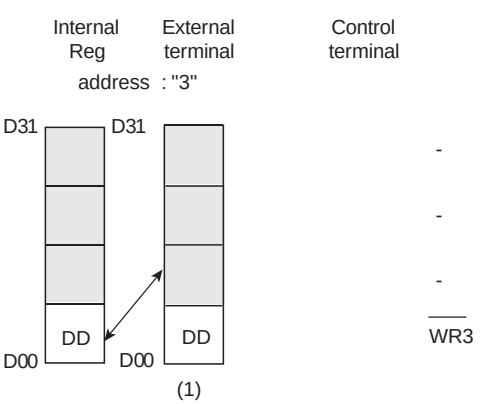
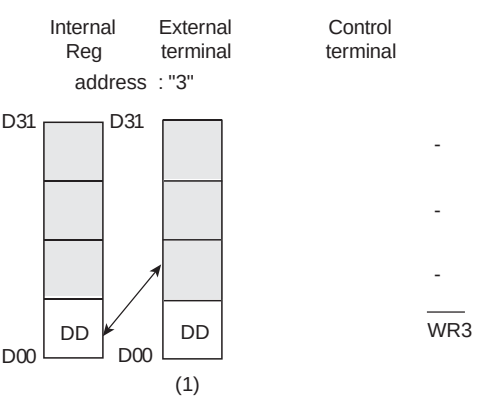
	Big endian mode	Little endian mode
32-bit bus width		
16-bit bus width		
8-bit bus width		

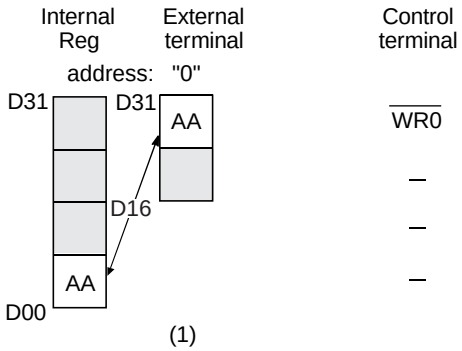
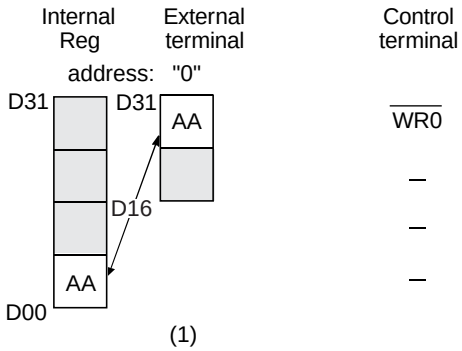
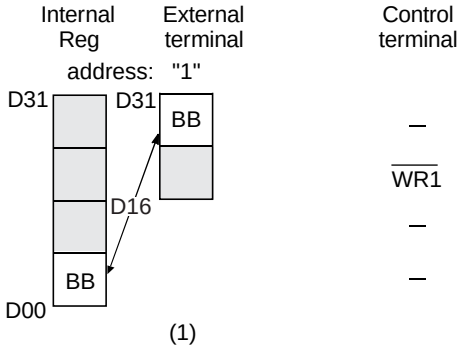
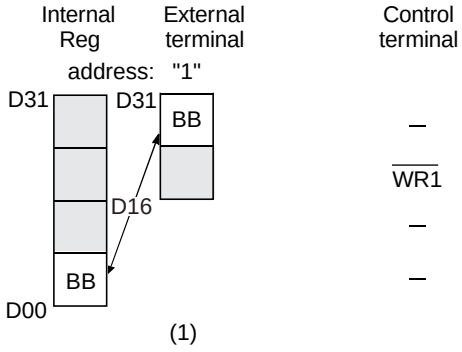
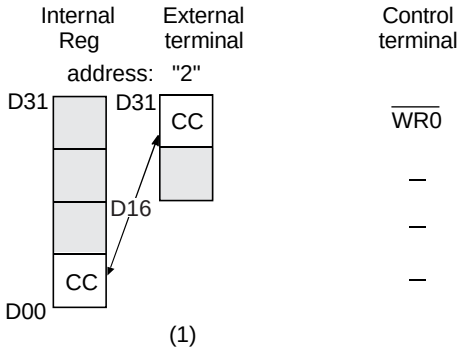
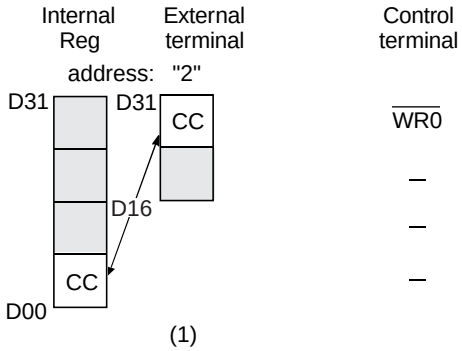
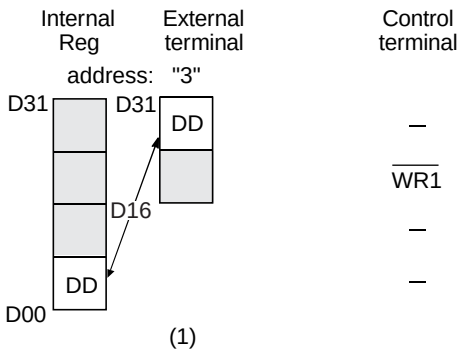
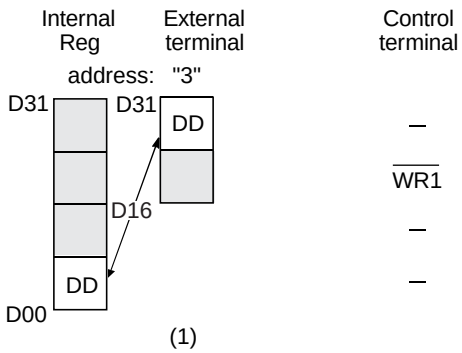
■ Halfword Access

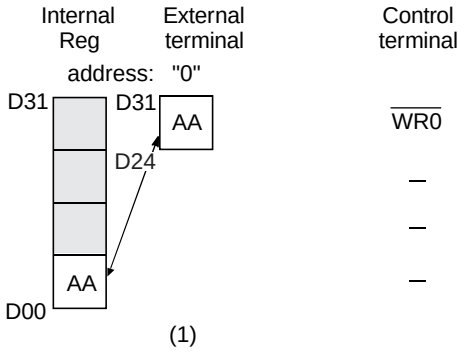
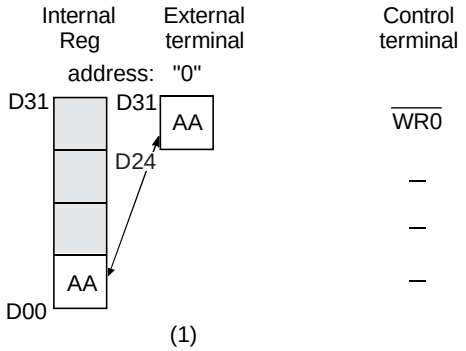
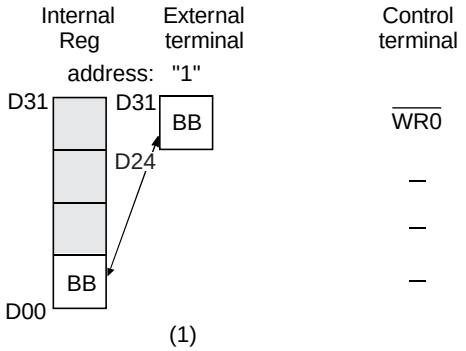
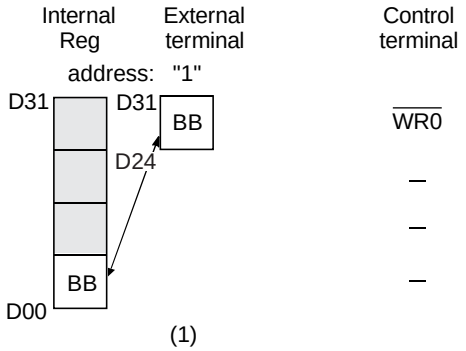
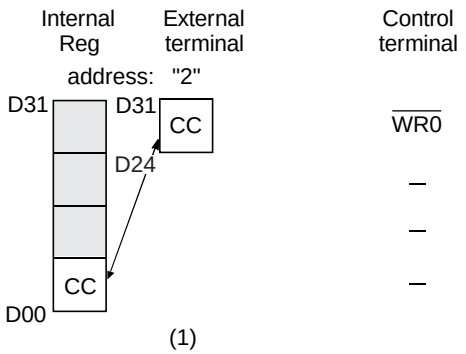
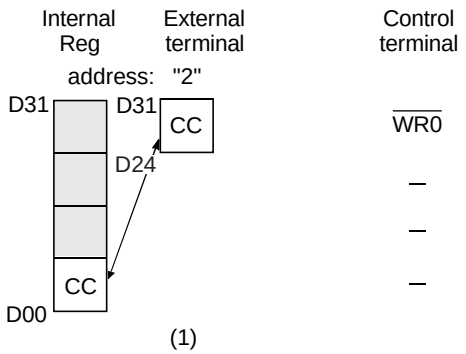
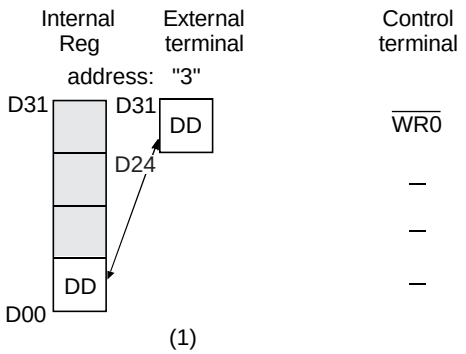
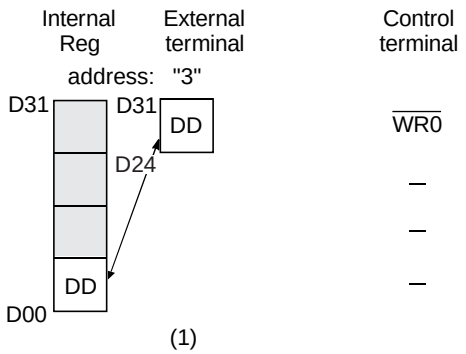
<table>
<tr><th></th><th>Big endian mode</th><th>Little endian mode</th></tr>
<tr><td rowspan="2">32-bit bus width</td><td> 
</td><td> 
</td></tr>
<tr><td> 
</td><td> 
</td></tr>
</table>

	Big endian mode	Little endian mode
16-bit bus width		
		
8-bit bus width		
		

■ Byte Access

	Big endian mode	Little endian mode
32-bit bus width		
		
		
		

	Big endian mode	Little endian mode
16-bit bus width		
		
		
		

	Big endian mode	Little endian mode
8-bit bus width		
		
		
		

5. Operation of the Ordinary bus interface

This section explains operation of the ordinary bus interface.

■ Ordinary Bus Interface

For the ordinary bus interface, two clock cycles are the basic bus cycles for both read access and write access.

The following operational phases of the ordinary bus interface are explained below with the use of a timing chart.

- Basic timing (for successive accesses)
- $\overline{\text{WRn}}$ + byte control type
- Read -> write
- Write -> write
- Auto-wait cycle
- External wait cycle
- Synchronous write enable output
- $\overline{\text{CSn}}$ delay setting
- $\overline{\text{CSn}}$ -> $\overline{\text{RD}}$/$\overline{\text{WRn}}$ setup, $\overline{\text{RD/WE}}$ -> $\overline{\text{CSn}}$ hold setting
- DMA fly-by transfer (I/O -> memory)
- DMA fly-by transfer (memory -> I/O)

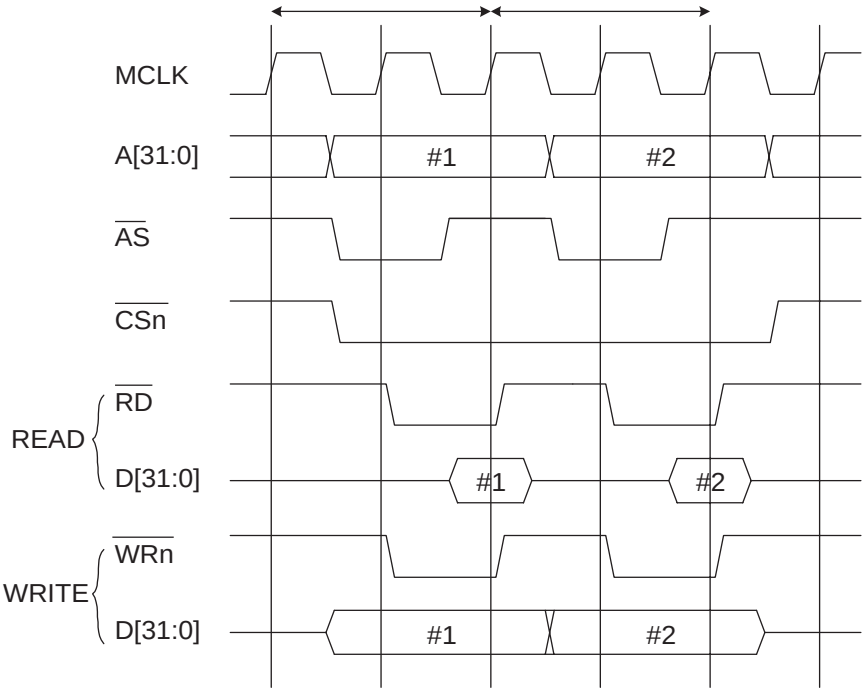
5.1 Basic Timing

This section shows the basic timing for successive accesses.

■ Basic Timing (For Successive Accesses)

Figure 5-1 "Basic Timing (For Successive Accesses)" shows the operation timing for (TYP3-0 = 0000_B, AWR = 0008_H)

Figure 5-1 Basic Timing (For Successive Accesses)



- $\overline{AS}$ is asserted for one cycle in the bus access start cycle.
- A31-0 continues to output the address of the location of the start byte in word/halfword/byte access from the bus access start cycle to the bus access end cycle.
- If the W02 bit of the AWR0-7 registers is 0, $\overline{CS0}$-$\overline{CS7}$ are asserted at the same timing as $\overline{AS}$. For successive accesses, $\overline{CS0}$-$\overline{CS7}$ are not negated. If the W00 bit of the AWR register is 0, $\overline{CS0}$-$\overline{CS7}$ are negated after the bus cycle ends. If the W00 bit is 1, $\overline{CS0}$-$\overline{CS7}$ are negated one cycle after bus access ends.
- $\overline{RD}$ and $\overline{WR0}$-$\overline{WR3}$ are asserted from the 2nd cycle of the bus access. Negation occurs after the wait cycle of bits W15-W12 of the AWR register is inserted. The timing of asserting $\overline{RD}$ and $\overline{WR0}$-$\overline{WR3}$ can be delayed by one cycle by setting the W01 bit of the AWR register to 1. However, depending on the internal state, the assertion of $\overline{WR0}$-$\overline{WR3}$ may not start in the 2nd cycle and may even be delayed if the W01 bit is set to 0.
- If a setting is made so that $\overline{WR0}$-$\overline{WR3}$ is used like TYP3-0=0x0xB, $\overline{WRn}$ is always H.
- For read access, D31-0 is read when MCLK rises in the cycle in which the wait cycle ended after $\overline{RD}$ was asserted.
- For write access, data output to D31-0 starts at the timing at which $\overline{WR0}$-$\overline{WR3}$ are asserted.

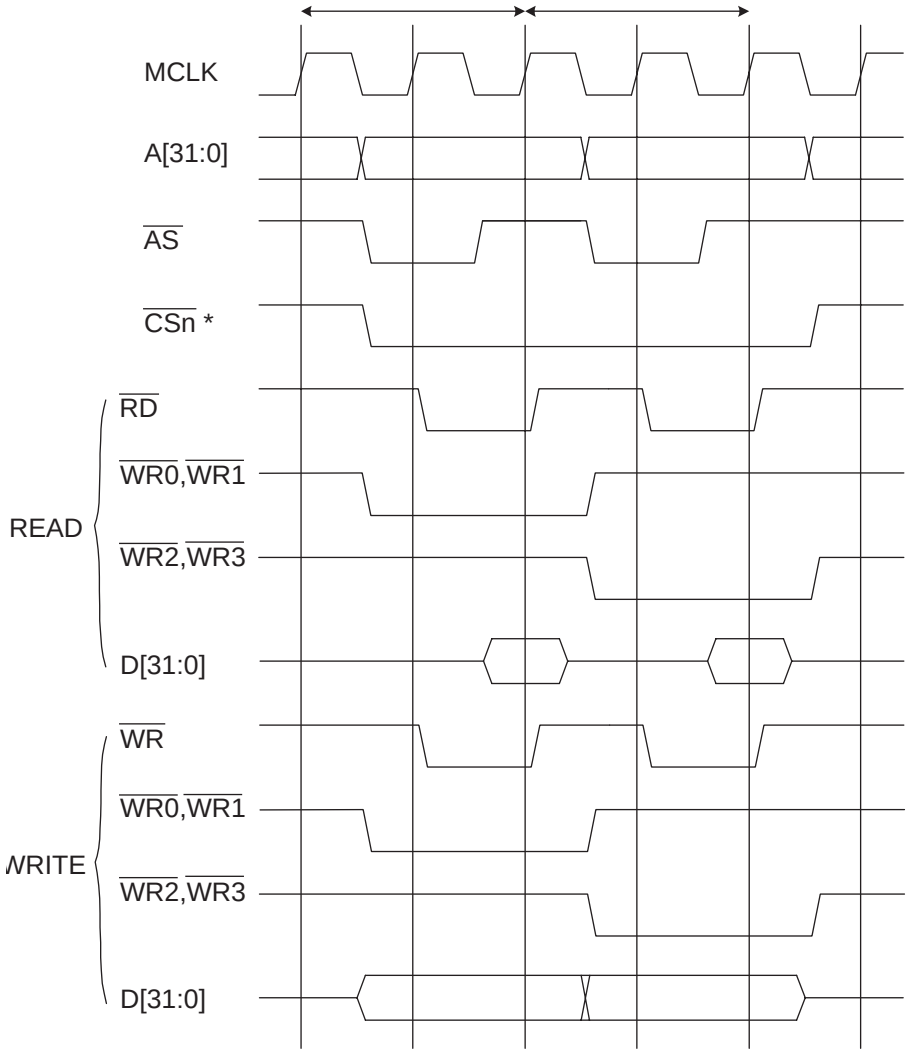
5.2 Operation of $\overline{WRn}$ + Byte Control Type

This section shows the operation timing for the $\overline{WRn}$ + byte control type.

■ Operation Timing of the $\overline{WRn}$ + Byte Control Type

Figure 5-2 "Timing Chart for the $\overline{WRn}$ + Byte Control Type" shows the operation timing for (TYP3-0 = 0010_B, AWR = 0008_H).

Figure 5-2 Timing Chart for the $\overline{WRn}$ + Byte Control Type



- Operation of $\overline{AS}$, $\overline{CSn}$, $\overline{RD}$, A31-0, and D31-16 is the same as that described in 5.1 "Basic Timing". $\overline{WRn}$ is asserted from the 2nd cycle of the bus access. Negation occurs after the wait cycle of bits W15-W12 of the AWR register is inserted. The timing of asserting $\overline{RD}$ and $\overline{WR0}$-$\overline{WR3}$ can be delayed by one cycle by setting the W01 bit of the AWR register to 1. However, depending on the internal state, assertion of $\overline{WR0}$-$\overline{WR3}$ may not start in the 2nd cycle and may even be delayed if the W01 bit is set to 0. (Operation is the same as that for $\overline{WR0}$-$\overline{WR3}$ described in 5.1 "Basic Timing" .)
- $\overline{WR0}$-$\overline{WR3}$ indicate the byte location expressed with negative logic when they are used for access as the byte enable signal. Assertion continues from the bus access start cycle to the bus access end cycle and changes at the same timing as the address timing. The byte location for access is indicated for both read access and write access.
- For write access, data output to D31-16 starts at the timing at which $\overline{WRn}$ is asserted. If the areas defined by TYP3-0=0x0x_B ($\overline{WR0}$-$\overline{WR3}$ used) and TYP3-0=0x1x_B ($\overline{WRn}$ + byte control) are mixed, be sure to make the following setting for all areas that will be used. (For details, see the notes).
  - Set at least one read -> write idle cycle.
  - Set at least one write recovery cycle.

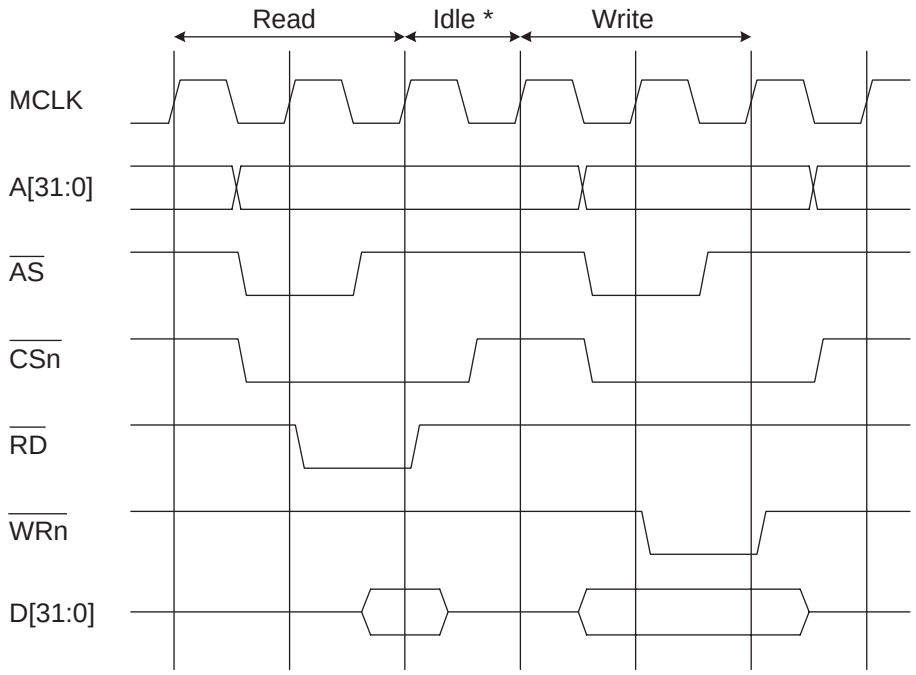
5.3 Read -> Write Operation

This section shows the operating timing for read -> write.

■ Operation Timing of Read -> Write

Figure 5-3 "Timing Chart for Read -> Write" shows the operation timing for (TYP3-0=0000_B, AWR=0048_H).

Figure 5-3 Timing Chart for Read -> Write



- Setting of the W07/W06 bits of the AWR register enables 0-3 idle cycles to be inserted.
- Settings in the CS area on the read side are enabled.
- This idle cycle is inserted if the next access after a read access is write access or access to another area.

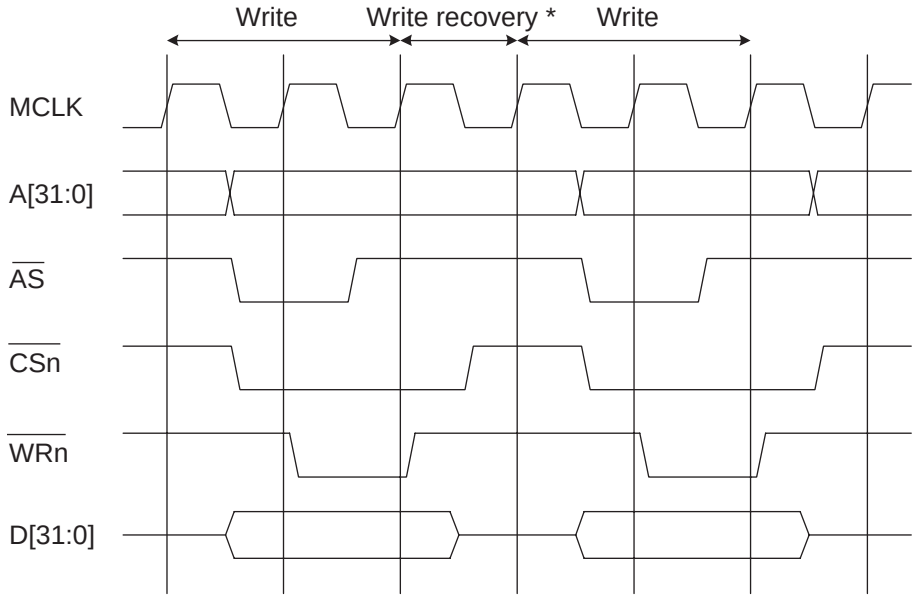
5.4 Write -> Write Operation

This section shows the operation timing for write -> write.

■ Write -> Write Operation

Figure 5-4 "Timing Chart for the Write -> Write Operation" shows the operation timing for (TYP3-0=0000_B, WR=0018_H).

Figure 5-4 Timing Chart for the Write -> Write Operation



- Setting of the W05/W04 bits of the AWR register enables 0-3 write cycles to be inserted.
- After all of the write cycles, recovery cycles are generated.
- Write recovery cycles are also generated if write access is divided into phases for access with a bus width wider than that specified.

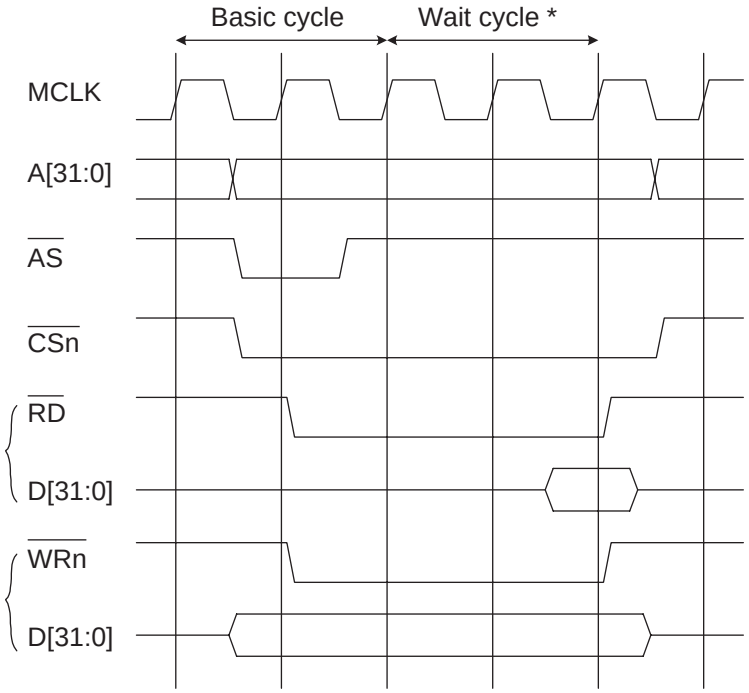
5.5 Auto-Wait Cycle

This section shows the operation timing for the auto-wait cycle.

■ Auto-Wait Cycle Timing

Figure 5-5 "Timing Chart for the Auto-Wait Cycle" shows the operation timing for (TYP3-0=0000_B, AWR=2008_H).

Figure 5-5 Timing Chart for the Auto-Wait Cycle



Setting of the W15-12 bits (first wait cycles) of the AWR register enables 0-15 auto-wait cycles to be set.

In Figure 5-5 "Timing Chart for the Auto-Wait Cycle", two auto-wait cycles are inserted, making a total of four cycles for access. If auto-wait is set, the minimum number of bus cycles is 2 cycles + (first wait cycles). For a write operation, the minimum number of bus cycles may be still longer depending on the internal state.

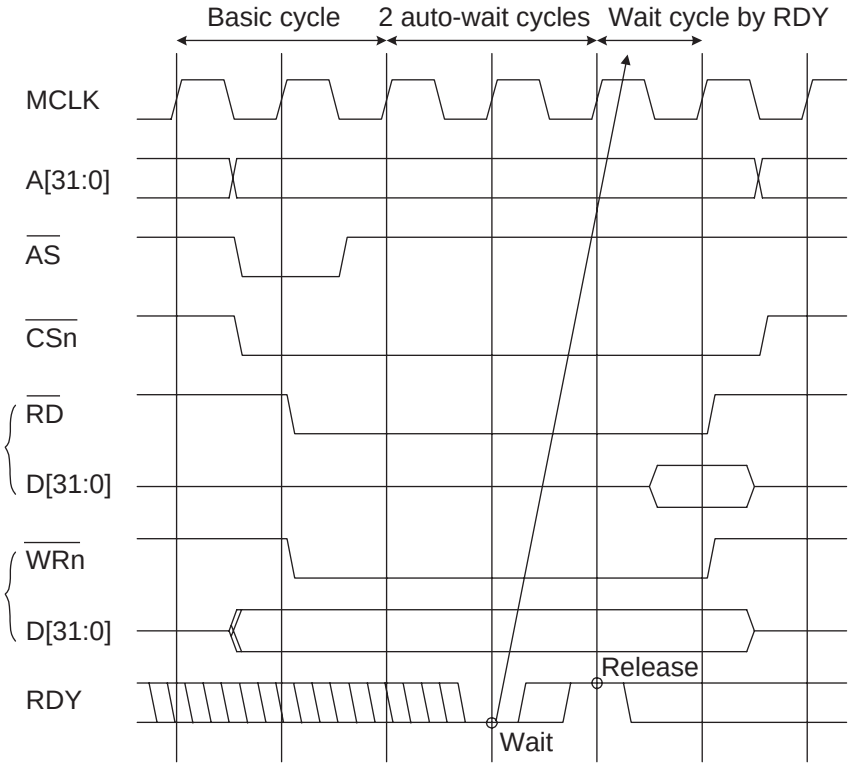
5.6 External Wait Cycle

This section shows the operation timing for the external wait cycle.

■ External Wait Cycle Timing

Figure 5-6 "Timing Chart for the External Wait Cycle" shows the operation timing for (TYP3-0=0001_B, AWR=2008_H).

Figure 5-6 Timing Chart for the External Wait Cycle



Setting 1 for the TYP0 bit of the ACR register and enabling the external RDY input pin enables external wait cycles to be inserted.

In Figure 4.5 - 6, the oblique - lined portion of the RDY pin is invalid because the wait based on the automatic wait cycle remains in effect.

The value at the RDY input pin is evaluated from the last automatic wait cycle on.

Once a wait cycle is completed, the value at the PDY input pin remains invalid until the next access cycle is started.

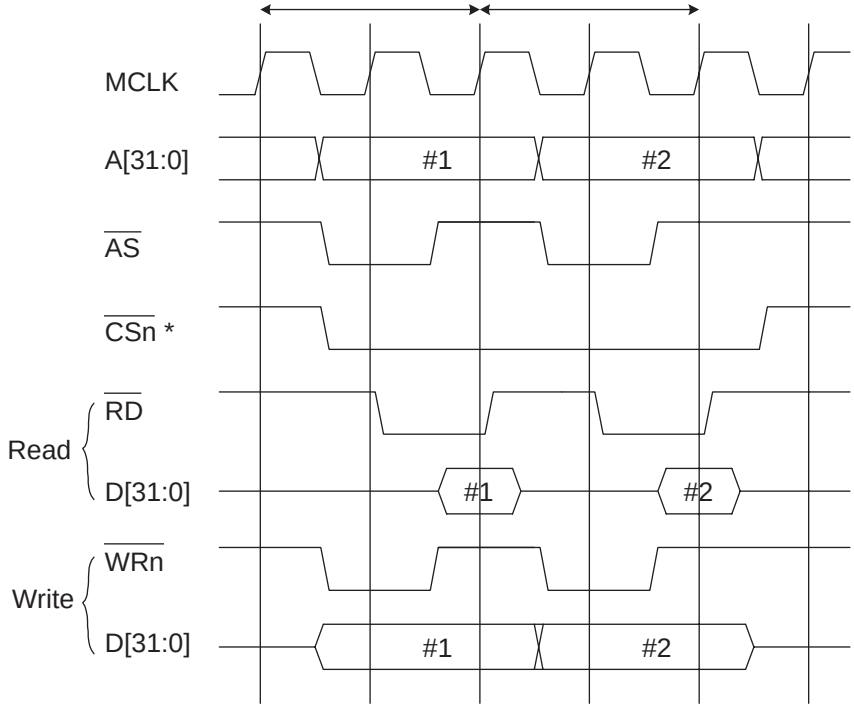
5.7 Synchronous Write Enable Output

This section shows the operation timing for synchronous write enable output.

■ Operation Timing for Synchronous Write Enable Output

Figure 5-7 "Timing Chart for Synchronous Write Enable Output" shows the operation timing for (TYP3-0=0000_B, AWR=0000_H).

Figure 5-7 Timing Chart for Synchronous Write Enable Output



- If synchronous write enable output is enabled (If the W03 bit of the AWR is 1), operation is as follows.
- WR0-WR3 and WRn pin output asserts synchronous write enable output at the timing at which AS pin output is asserted. For a write to an external bus, the synchronous write enable output is L. For a read from an external bus, the synchronous write enable output is H.
- Write data is output from the external data output pin in the clock cycle following the cycle in which synchronous write enable output is asserted. If write data cannot be output because the internal bus is temporarily unavailable, assertion of synchronous write enable output may be extended until write data can be output.
- Read strobe output (RD) functions as an asynchronous read strobe regardless of the setting of WR0-WR3 and WRn output timing. Use it as is for controlling the data I/O.

- If synchronous write enable output is used, the following restrictions apply:

  Do not set the following additional wait because the timing for synchronous write enable output becomes meaningless:

  - $\overline{CS}$ -> $\overline{RD}$/$\overline{WRn}$ setup (Always write 0 to the W01 bit of AWR)
  - First wait cycle setting (Always write 0000 to bits W15-W12 of AWR)

  Do not set the following access types (TYPE3-0 bits (Bits 3-0) in the ACR register) because the timing for synchronous write enable output becomes meaningless:

  - Multiplex bus setting (Always write 0 to the TYPE2 bit of ACR)
  - RDY input enable setting (Always write 0 to the TYPE0 bit of ACR)

  Always set the burst length to "1" (BST1 to 0 bit = 0) for the synchronous write enable output

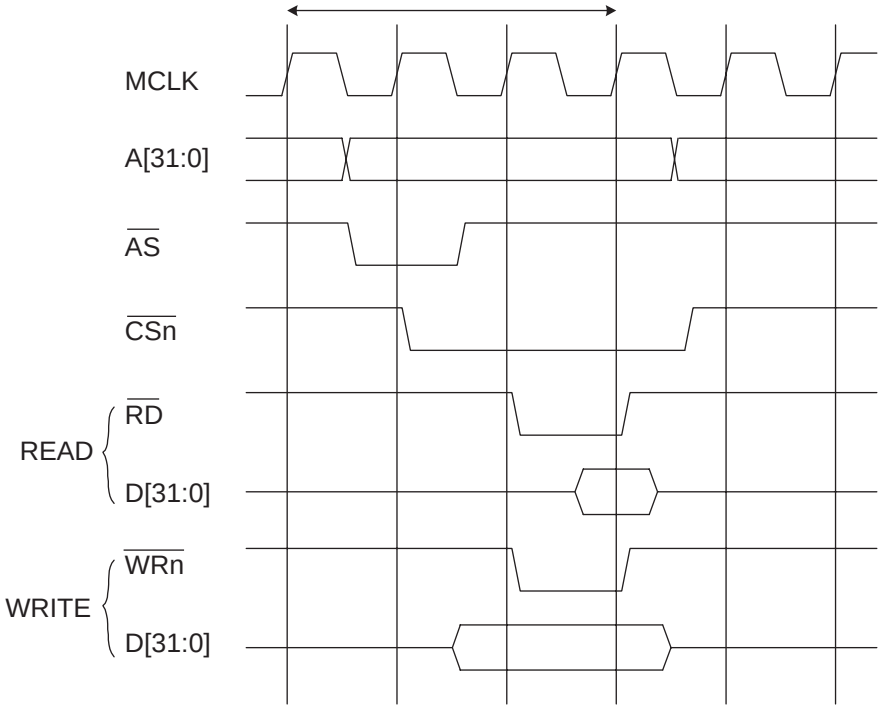
5.8 $\overline{CSn}$ Delay Setting

This section shows the operation timing for the $\overline{CSn}$ delay setting.

■ Operation Timing for the $\overline{CS}$ Delay Setting

Figure 5-8 "Operation Timing Chart for the $\overline{CS}$ Delay Setting" shows the operation timing for (TYP3-0=0000_B, AWR=$000C_H$).

Figure 5-8 Operation Timing Chart for the $\overline{CS}$ Delay Setting



If the W02 bit is 1, assertion starts in the cycle following the cycle in which $\overline{AS}$ is asserted. For successive accesses, a negation period is inserted.

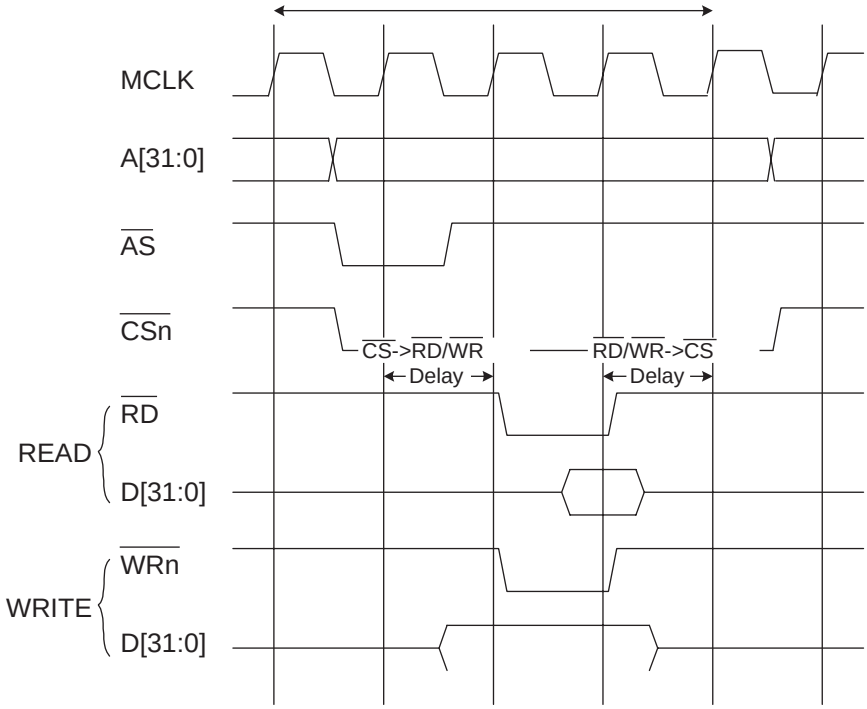
5.9 $\overline{CSn}$ -> $\overline{RD}$/$\overline{WRn}$ Setup and $\overline{RD}$/$\overline{WRn}$ -> $\overline{CSn}$ Hold Setting

This section shows the operation timing for the $\overline{CSn}$ -> $\overline{RD}$/$\overline{WRn}$ setup and $\overline{RD}$/$\overline{WRn}$ -> $\overline{CSn}$ hold settings.

■ Operation Timing for the $\overline{CSn}$ -> $\overline{RD}$/$\overline{WRn}$ Setup and $\overline{RD}$/$\overline{WRn}$ -> $\overline{CSn}$ Hold Settings

Figure 5-9 "Timing Chart for the $\overline{CSn}$ -> $\overline{RD}$/$\overline{WRn}$ Setup and $\overline{RD}$/$\overline{WRn}$ -> $\overline{CSn}$ Hold Settings" shows the operation timing for (TYP3-0=0000_B AWR=$000B_H$).

Figure 5-9 Timing Chart for the $\overline{CSn}$ -> $\overline{RD}$/$\overline{WRn}$ Setup and $\overline{RD}$/$\overline{WRn}$ -> $\overline{CSn}$ Hold Settings



- Setting 1 for the W01 bit of the AWR register enables the $\overline{CSn}$ -> $\overline{RD}$/$\overline{WRn}$ setup delay to be set. Set this bit to extend the period between chip select assertion and read/write strobe.
- Setting 1 for the W00 bit of the AWR register enables the $\overline{RD}$/$\overline{WRn}$ -> $\overline{CSn}$ hold delay to be set. Set this bit to extend the period between read/write strobe negation and chip select negation.
- The $\overline{CSn}$ -> $\overline{RD}$/$\overline{WRn}$ setup delay (W01 bit) and $\overline{RD}$/$\overline{WRn}$ -> $\overline{CSn}$ hold delay (W00 bit) can be set independently.
- When making successive accesses within the same chip select area without negating the chip select, neither a $\overline{CSn}$ -> $\overline{RD}$/$\overline{WRn}$ setup delay nor an $\overline{RD}$/$\overline{WRn}$ -> $\overline{CSn}$ hold delay is inserted.
- If a setup cycle for determining the address or a hold cycle for determining the address is needed, set 1 for the address -> $\overline{CSn}$ delay setting (W02 bit of the AWR register).

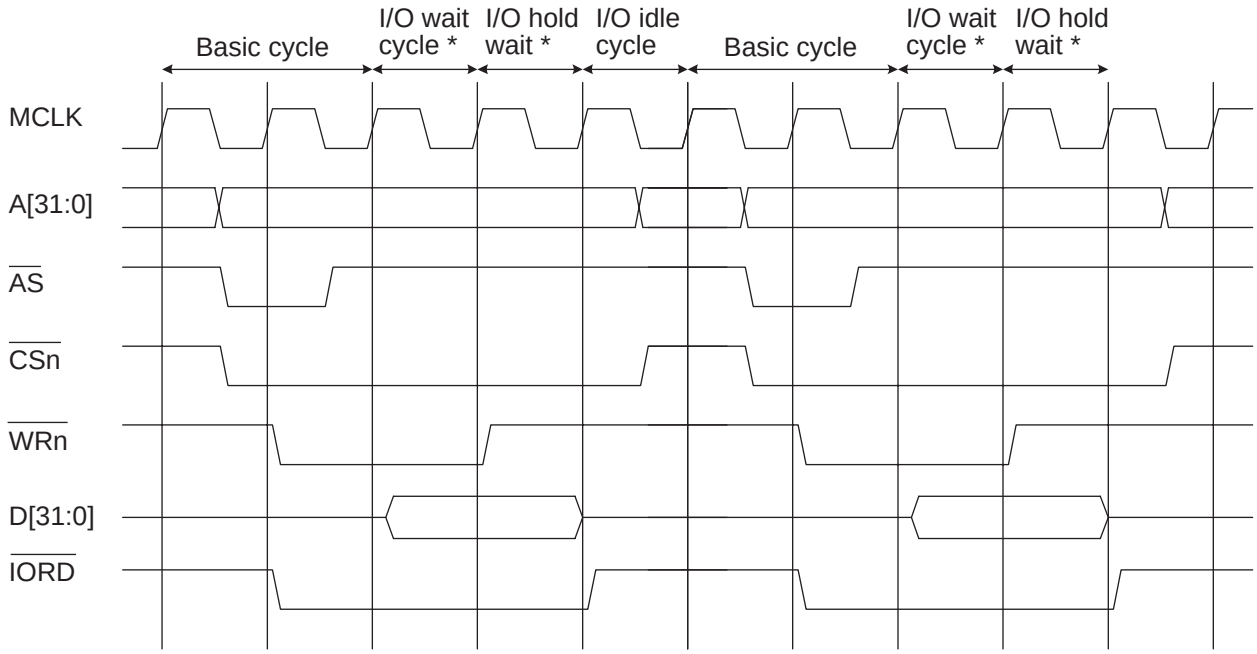
5.10 DMA Fly-By Transfer (I/O -> Memory)

This section shows the operation timing for DMA fly-by transfer (I/O -> memory).

■ Operation Timing for DMA Fly-By Transfer (I/O -> Memory)

Figure 5-10 "Timing Chart for DMA Fly-By Transfer (I/O -> Memory)" shows the operation timing for (TYP3-0=0000_B, AWR=0008_H, IOWR=51_H). This timing chart shows a case in which a wait is not set on the memory side.

Figure 5-10 Timing Chart for DMA Fly-By Transfer (I/O -> Memory)



- Setting 1 for the HLD bit of the IOWR0-3 registers enables the I/O read cycle to be extended by one cycle.
- Setting bits IW3-0 of the IOWR0-3 registers enables 0-15 wait cycles to be inserted.
- If wait is also set on the memory side (AWR15-12 is not 0), the larger value is used as the wait cycle after comparison with the I/O wait (IW3-0 bits).

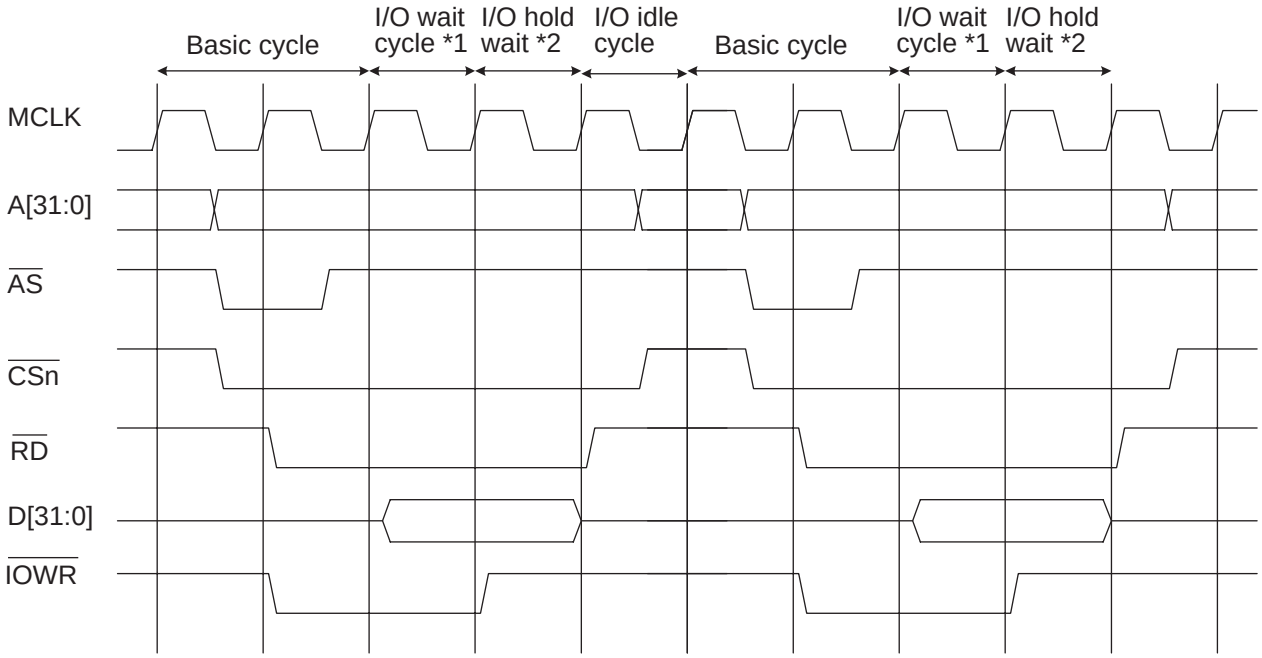
5.11 DMA Fly-By Transfer (Memory -> I/O)

This section shows the operation timing for DMA fly-by transfer (memory -> I/O).

■ Operation Timing for DMA Fly-By Transfer (Memory -> I/O)

Figure 5-11 "Timing Chart for DMA Fly-By Transfer (Memory -> I/O)" shows the operation timing chart for (TYP3-0=0000_B, AWR=0008_H, IOWR=51_H). This timing chart shows a case in which a wait is not set on the memory side.

Figure 5-11 Timing Chart for DMA Fly-By Transfer (Memory -> I/O)



- Setting 1 for the HLD bit of the IOWR0-3 registers enables the I/O read cycle to be extended by one cycle.
- Setting the WR1,0 bits of the IOWR0-3 registers enables 0-3 write recovery cycles to be inserted.
- If the write recovery cycle is set to 1 or more, a write recovery cycle is always inserted after write access.
- Setting bits IW3-0 of the IOWR0-3 registers enables 0-15 wait cycles to be inserted.
- If wait is also set on the memory side (AWR15-12 is not 0), the larger value is used as the wait cycle after comparison with the I/O wait (IW3-0 bits).

6. Burst Access Operation

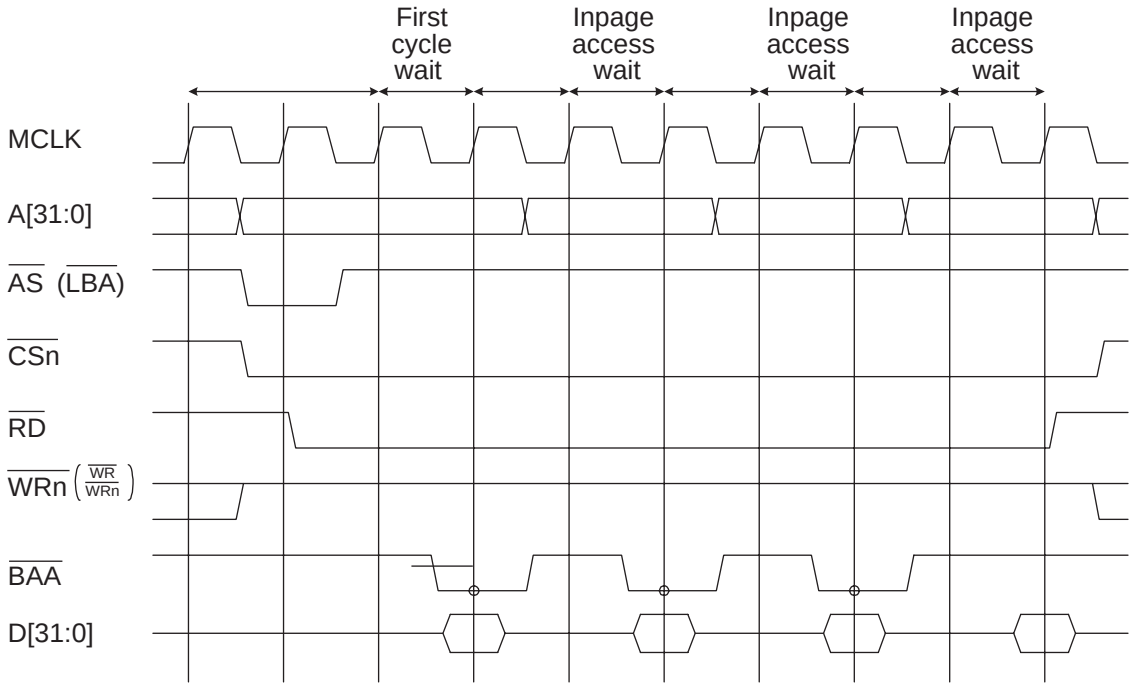
In the external bus interface, the operation that transfers successive data items in one access sequence is called burst access. The normal access cycle (that is, not burst access) is called single access. One access sequence starts with an assertion of $\overline{AS}$ and $\overline{CSn}$ and ends with negation of $\overline{CSn}$. Multiple data items two or more units of data of the unit set for the area.

This section explains burst access operation.

■ Burst Access Operation

Figure 6-1 "Timing chart for burst access" shows the operation timing chart for (first wait cycle=1, inpage access wait cycle=1, TYP3-0=0000_B, AWR=1108_H).

Figure 6-1 Timing Chart for Burst Access



- In addition to more efficient use of access cycles when a sizable amount of data of asynchronous memory such as page mode ROM and burst flash memory is read, burst cycles can also be used for reading from normal asynchronous memory.
- The access sequence when burst cycles are used can be divided into the following two types:

  - First access cycle

  The first access cycle is the start cycle for the burst access and operates in the same way as the normal single access cycle.

  - Page access cycle

  The page access cycle is a cycle following the first access cycle in which both $\overline{CSn}$ and $\overline{RD}$ (read strobe) are asserted. Wait cycles that are different from those set for a single cycle can be set. The page access cycle is repeated while access remains in the address boundary determined by the burst length setting. When access within the address boundary ends, burst access terminates and $\overline{CSn}$ is negated.

- Setting of the W15-W12 bits of the AWR register enable the first 0-15 wait cycles to be inserted. At this point,

the minimum number of the first access cycles is the wait cycles + 2 cycles (three cycles in the timing chart shown in Figure 6-1 "Timing Chart for Burst Access").

- Setting of the W11-W08 bits of the AWR register enables 0-15 page wait cycles to be inserted. At this point, the page access cycles can be obtained from the page wait cycles + 1 cycle (Two cycles in the timing chart shown in Figure 6-1 "Timing Chart for Burst Access")
- Setting of the BST bits of the ACR register enables the burst length to be set as 1, 2, 4, or 8. If the burst length is set to 1, single access mode is set and only the first cycle is repeated. However, if the data bus width is set to 32 bits (the BST bits of the ACR register are 10_B), set the burst length to 4 or less (A malfunction occurs if the burst length is set to 8).
- If burst access is enabled, burst access is used when prefetch access or transfer with a larger size than the specified data bus width is performed. For example, if word access to an area whose data bus width is set to 8 bits and burst length to 4 is performed, access of 4 bursts is performed once instead of repeating byte access four times.
- Since RDY input is ignored in areas for which burst access is set, do not set TYP3-0=0xx1_B.
- The $\overline{\text{LBA}}$ and $\overline{\text{BAA}}$ signals are designed for burst FLASH memory. $\overline{\text{LBA}}$ indicates the start of access and $\overline{\text{BAA}}$ indicates the address increment.
- A31-0 is updated after the wait cycles that were set during burst access.

7. Address/data Multiplex Interface

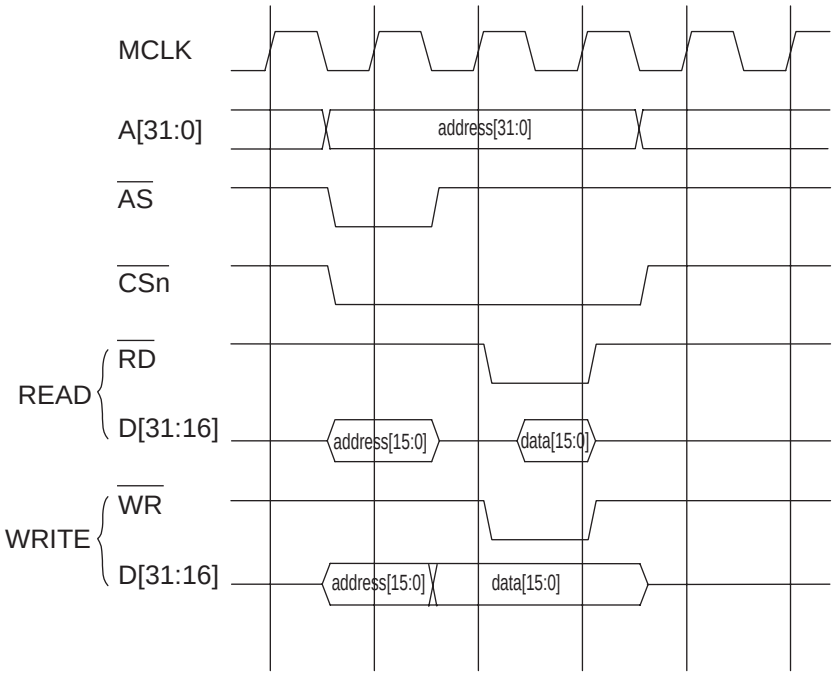
This section explains the following three cases of operation of the address/data multiplex interface:

- Without external wait
- With external wait
- $\overline{\text{CSn}}$ -> $\overline{\text{RD}}$/$\overline{\text{WRn}}$ setup

■ Without External Wait

Figure 7-1 "Timing Chart for the Address/Data Multiplex Interface (without External Wait)" shows the operation timing chart for (TYP3-0=0100_B, AWR=0008_H).

Figure 7-1 Timing Chart for the Address/Data Multiplex Interface (without External Wait)



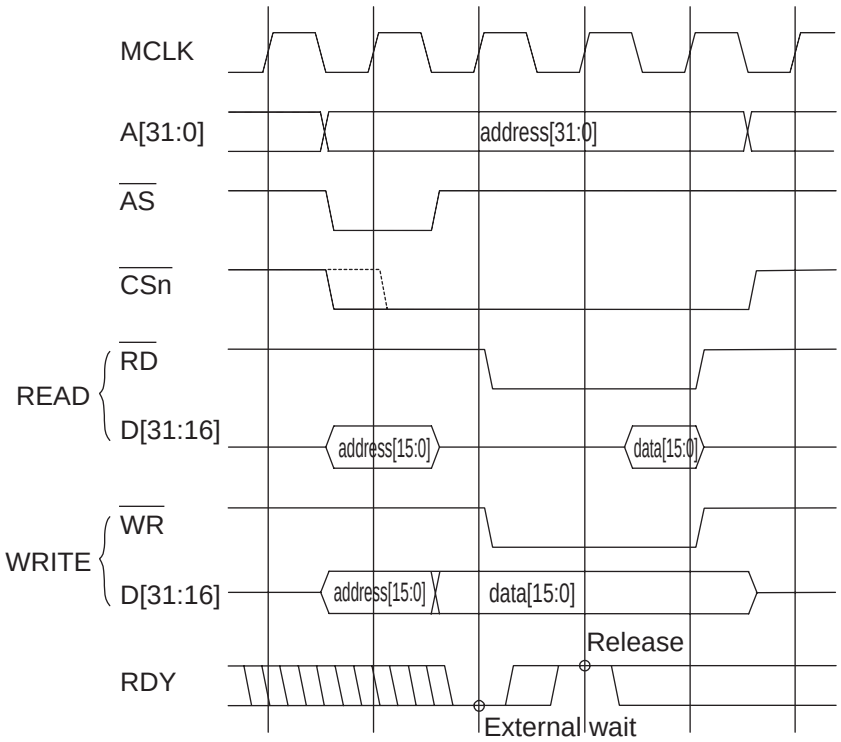
- Making a setting such as TYP3-0=$01xx_B$ in the ACR register enables the address/data multiplex interface to be set.
- If the address/data multiplex interface is set, set 8 bits or 16 bits for the data bus width (DBW1-0 bits).
- In the address/data multiplex interface, the total of 3 cycles of 2 address output cycles + 1 data cycle becomes the basic number of access cycles.
- In the address output cycles, $\overline{\text{AS}}$ is asserted as the output address latch signal.
- As with a normal interface, the address indicating the start of access is output to A31-0 during the time division bus cycle. Use this address if you want to use an address more than 8/16 bits in the address/data multiplex interface.

- As with the normal interface, auto-wait (AWR15-12), read -> write idle cycle (AWR7-6), write recovery (AWR5-4), address -> $\overline{CSn}$ delay (AWR2), $\overline{CSn}$ -> $\overline{RD}/\overline{WRn}$ setup delay (AWR1), and $\overline{RD}/\overline{WRn}$ -> $\overline{CSn}$ hold delay (AWR0) can be set.
- In areas for which the address/data multiplex interface is set, set 1(DBW1-0=00_B) as the burst length.

■ With External Wait

Figure 7-2 "Timing Chart for the Address/Data Multiplex Interface (with External Wait)" shows the operation timing chart for (TYP3-0=0101_B, AWR=1008_H).

Figure 7-2 Timing Chart for the Address/Data Multiplex Interface (with External Wait)

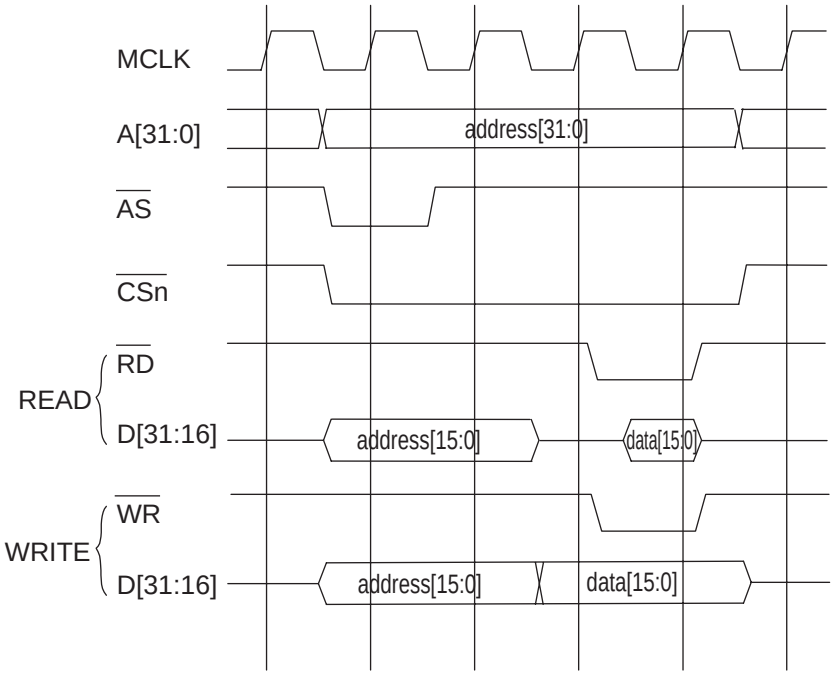


Making a setting such as TYP3-0=01x1_B in the ACR register enables RDY input in the address/data multiplex interface.

■ $\overline{CSn}$ -> $\overline{RD}/\overline{WRn}$ Setup

Figure 7-3 "Timing Chart for the Address/Data Multiplex Interface ($\overline{CSn}$ -> $\overline{RD}/\overline{WRn}$ Setup)" shows the operation timing chart for (TYP3-0=0101_B, AWR=$100B_H$).

Figure 7-3 Timing Chart for the Address/Data Multiplex Interface ($\overline{CSn}$ -> $\overline{RD}$/$\overline{WRn}$ Setup)



Setting 1 for the $\overline{CSn}$ -> $\overline{RD}$/$\overline{WRn}$ setup delay (AWR1) enables the multiplex address output cycle to be extended by one cycle as shown in Figure 7-3 "Timing Chart for the Address/Data Multiplex Interface ($\overline{CSn}$ -> $\overline{RD}$/$\overline{WRn}$ Setup)", allowing the address to be latched directly to the rising edge of $\overline{AS}$. Use this setting if you want to use $\overline{AS}$ as an ALE (Address Latch Enable) strobe without using MCLK.

8. Prefetch Operation

This section explains the prefetch operation.

■ Prefetch Operation

The external bus interface controller contains a prefetch buffer consisting of 16 x 8 bits.

If the PSUS bit of the TCR register is 0 and read access to an area to which the PFEN bit of the ACR register is set to 1 occurs, the subsequent address is prefetched and then stored in the prefetch buffer.

If the stored address is accessed from the internal bus, the lookahead data in the prefetch buffer is returned without external access being performed. This can reduce the wait time for successive accesses to the external bus areas.

● Basic conditions for starting external access using prefetch

External bus access using prefetch occurs when the following conditions are met:

- The PSUS bit of the TCR register is 0.
- Neither sleep mode nor stop mode is set.
- Read access by the external bus to a chip select area for which prefetch is enabled has been performed. DMA access and read access by a read modified write system instruction, however, are excluded.
- No external bus access request (external bus area access to an area for which prefetch is not enabled or DMA transfer with an external bus area) other than the prefetch access has occurred.
- The part of the prefetch buffer for the next operation of capturing the prefetch access is completely empty.

While the above conditions are met, the prefetch access will continue. If external bus area access to an area for which prefetch is not enabled occurs after prefetch access, prefetch access to the area for which prefetch is enabled will continue as long as the prefetch buffer clear conditions are not met.

For an access that mixes multiple prefetch-enabled areas and multiple prefetch-disabled areas, the prefetch buffer always holds data of the prefetch-enabled area accessed last. Since, in this case, access to prefetch-disabled areas does not affect the prefetch buffer state at all, data in the prefetch buffer is not wasted even if prefetch-disabled data access and prefetch-enabled instruction fetch are mixed.

● Optional clear for temporary stopping of a prefetch access

Setting 1 for the PSUS bit of the TCR register temporarily stops a prefetch. The prefetch can be restarted by setting the PSUS bit to 0. At this point, the contents of the buffer are retained if no error occurs or a buffer clear such as occurs when the PCLR bit is set does not occur.

Setting 1 for the PCLR bit of the TCR register completely clears the prefetch buffer. Clear the buffer by setting the PSUS bit when prefetch is interrupted.

Prefetch is temporarily stopped for the minimum unit (64 KB) of the boundary=chip select area where the high-order 16 bits of an address change. If the boundary is crossed, first a buffer read error occurs and then prefetch starts in a new area.

● Unit for one prefetch access operation

The unit for one prefetch access operation is determined by the DBW bits (bus width) and BST bits (burst length).

Prefetch access always occurs with the full size of the bus width specified by the DBW bits and access for the count of the burst length set by the BST bits in one access operation is performed. That is, if any value other than 00_B is set for the BST bits, the prefetch always occurs in page mode/burst mode. Keep in mind whether ROM/RAM is conformable and enough access time is applicable. (Set an appropriate value bits W15-08 bits of the AWR register).

During burst access, successive accesses occur only within the address boundary that that is determined by the burst length. Thus, if the boundary is crossed, for example, 4 bytes of free space are available in the buffer, these 4 bytes cannot be accessed in one operation (If the prefetch buffer starts at xxxxxx0E$_H$, 4 bytes of free space are available in the buffer, and two bursts are set even though the bus width is 16 bits, only 2 bytes, xxxxxx0E$_H$ and xxxxxx0F$_H$, can be captured in the next prefetch access).

The following provides two examples:

- Area whose bus width is set to 16 bits and whose burst length is set to 2

  The amount of data read into the buffer in one prefetch operation is 4 bytes. In this case, prefetch access is delayed until 4 bytes of free space are available in the prefetch buffer.

- Area whose bus width is set to 8 bits and whose burst length is set to 8

  The amount of data read into the buffer in one prefetch operation is 8 bytes. In this case, prefetch access is delayed until 8 bytes of free space are available in the prefetch buffer.

● Burst length setting and prefetch efficiency

If requests for external bus access, other than prefetch access, to or errors in the prefetch buffer occur during one operation of prefetch access as explained in the previous bullet, "Unit of one prefetch access operation," these access requests must wait until access to the prefetch buffer that is being executed is completed.

Thus, if the burst length is too long, the efficiency and reaction of bus access other than prefetch may be degraded. If, on the other hand, the burst length is set to 1, many read cycles may be wasted even if burst/page access memory is connected because single access is always performed.

If settings are made so that the amount of data read in one prefetch access operation is large, prefetch access can be started only after free space in the prefetch buffer for this amount is available. Thus, access to the prefetch buffer is infrequent, and the external bus tends to be idle. For example, if the bus width is set to 16 bits and the burst length is set to 8, the amount of data read into the buffer in one prefetch operation is 16 bytes. Thus, a new prefetch access can be started only after the prefetch buffer is completely empty.

Adjust the optimum burst length to suit use and the environment after taking the above into consideration. Generally, when connecting asynchronous memory to which burst/page access cannot be applied, it is best to set the burst length to 1 (single access). Conversely, when memory whose burst/page access cycle is short is connected, it is better to set the burst length to any value other than 1 (single access). In this case, it is best to make the setting so that 8 bytes (half of the buffer) are read in one read operation according to the bus width. However, the optimum condition varies with the frequency of external access and varies with the frequency divide-by rate setting of the external access clock.

● Reading from the prefetch buffer

Data stored in the prefetch buffer is read in response to access from the internal bus if an address matches, and no external access is performed. In reading from the buffer, addresses can be hit (up to 16 bytes) if they are in the forward direction but not continuous, so that a second read from the external bus is avoided, if possible, even for a short forward branch.

If the address currently being accessed for prefetch matches during access from the internal bus, a wait signal is returned internally before data is captured after prefetch access is completed. In this case, no buffer error occurs.

If an address in the prefetch buffer matches when a read is performed for DMA transfer, data in the prefetch buffer is not used, and instead, external data is read by the external bus. In this case, a buffer error occurs. The prefetch is not continued and no prefetch access is performed until a new external access operation to a prefetch-enabled area occurs.

● Clearing/updating the prefetch buffer

If either of the following conditions is met, the prefetch buffer is completely cleared:

- If 1 is written to the PCLR bit of the TCR register

- If a buffer read error occurs. A buffer read error is if any of the following events occurs:
  - When no address is found in the buffer that matches in an to read from a prefetch-enabled area. In this case, the external bus is accessed again. Data read in this case is not stored in the buffer, but the prefetch access is started from the subsequent address to store addresses in the buffer.
  - In an access to read from a prefetch-enabled area with a read modified system instruction. In this case, the external bus is accessed again. Data read in this case is not stored in the buffer. Also, no prefetch access is performed (This is because data is written to the next address).
  - In an access to read from a prefetch-enabled area for DMA transfer. In this case, the external bus is accessed again. Data read in this case is not stored in the buffer. Also, no prefetch access is performed.
- If a buffer write hit occurs. A buffer write hit is as follows:
  - When the address of just one byte that matches is found in the buffer in an access to write to a prefetch-enabled area. In this case, the external bus is accessed again, but no prefetch access is performed before a new read access occurs.

Only part of the prefetch buffer is cleared when the following condition is met:

- If a buffer read hit occurs

In this case, only the part of the buffer before the hit address is cleared.

● Restrictions on prefetch-enabled areas

If prefetch to a little endian area is enabled, be sure to access the area using word access. If data read into the prefetch buffer is accessed with any length other than word length, the correct endian conversion is not performed and thus the wrong data will be read. This is due to hardware restrictions related to the endian conversion mechanism.

9. SDRAM/FCRAM Interface Operation

This section describes the operations of the SDRAM/FCRAM interface.

■ SDRAM/FCRAM interface

The chip select areas can be used as SDRAM/FCRAM interface by setting the TYP3 to TYP0 bits in the area configuration register (ACR) to $100X_B$.

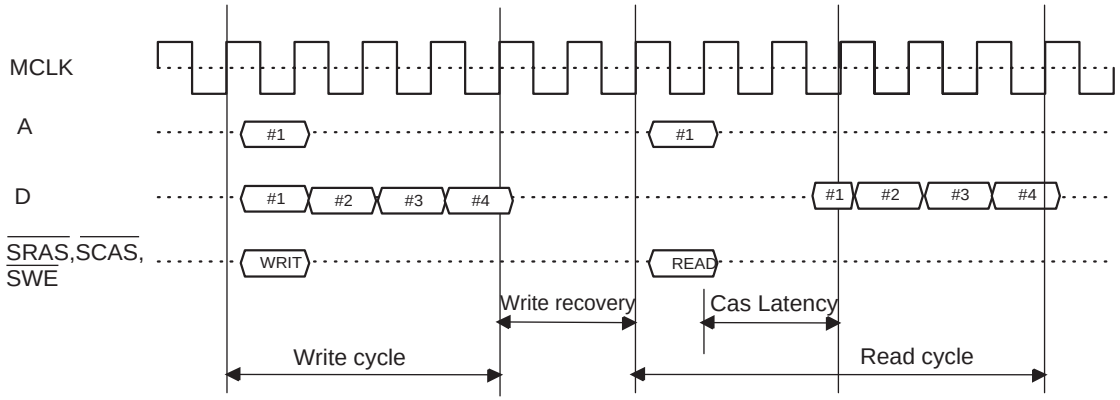
This section provides timing charts to describe the following operations of the SDRAM/FCRAM interface.

- Burst read/write (Settings: Page hit, CAS latency 2)
- Single read/write (Settings: Page hit, CAS latency 3, auto - precharge OFF)
- Single read (Settings: Page miss, CAS latency 3, auto - precharge OFF)
- Single read/write (Settings: CAS latency 1, TYP 1001_B, auto - precharge ON)
- Auto - refresh

■ Burst Read/Write Operation Timing

Figure 4.9 - 1 shows the operation timings assuming that page hits and CAS latency 2 are set.

Figure 9-1 Burst Read/Write Timing Chart

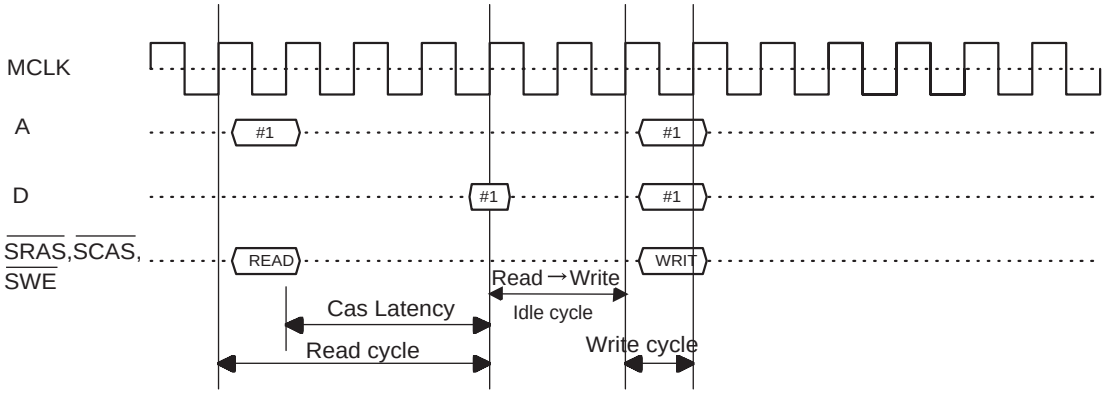


- All of the A13 to A0 pins may not be used depending on the SDRAM capacity. See Section " Memory Connection Examples " .
- The MCLK is a clock signal input to SDRAM. Signals such as addresses, data, and commands are input to SDRAM at the rise of the MCLK.
- Set the W05 and W04 bits in the area wait register (AWR) to the write recovery cycle according to the SDRAM/FCRAM standards.
- Set the W10 to W08 bits in the area wait register (AWR) to the CAS latency according to the SDRAM/FCRAM standards.
- Set the burst length using the BST bit in the area configuration register (ACR).

■ Single Read/Write Operation Timing

Figure 9-2 shows the operation timings assuming that page hits, CAS latency 3, and no auto - precharge are set.

Figure 9-2 Single Read/Write Timing Chart

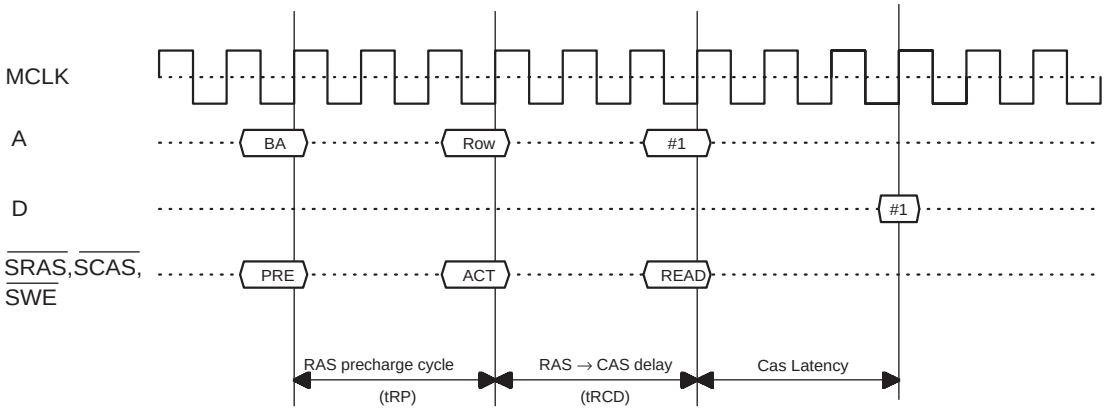


Set the W07 and W06 bits in the area wait register (AWR) to the read - to - write idle cycle according to the SDRAM/FCRAM standards.

■ Single Read Operation Timing

Figure 9-3 shows the operation timings assuming that page misses, CAS latency 3, and no auto - precharge are set.

Figure 9-3 Single Read Timing Chart

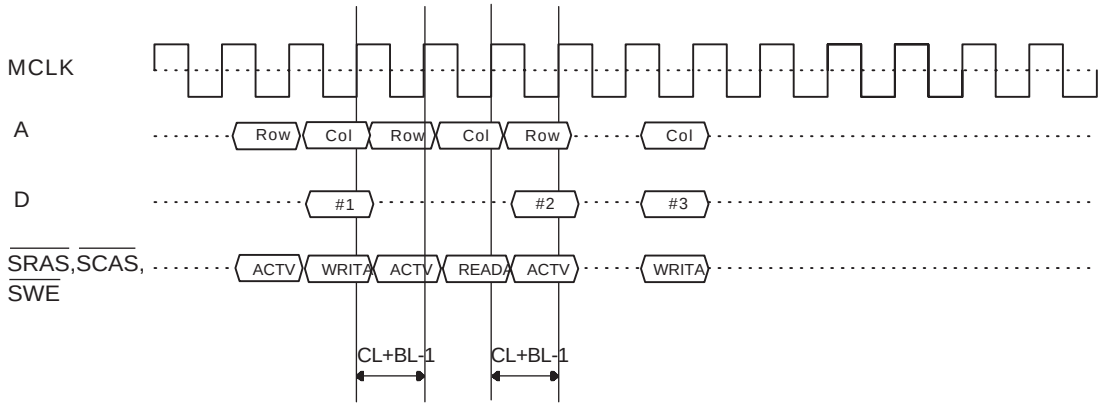


- When a page miss occurs, a read operation is performed after the PRE charge and ACTV commands are issued.
- Set the W01 and W00 bits in the area wait register (AWR) to the RAS precharge cycle (tRP) according to the SDRAM/FCRAM standards.
- Set the W14 to W12 bits in the area wait register (AWR) to the RAS - to - CAS delay (tRCD) according to the SDRAM/FCRAM standards.

■ Single Read/Write Operation Timing

Figure 9-4 shows the operation timings assuming that CAS latency 1, TYP = 1001_B, and auto - precharge are set.

Figure 9-4 Single Read/Write Timing Chart

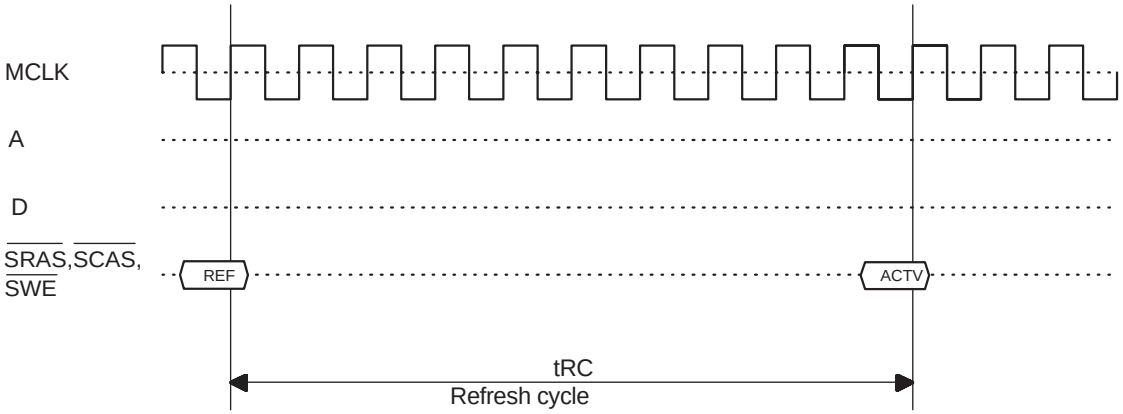


- Setting TYP to 1001_B causes a read/write command with auto - precharge to be issued. Since the cycle from READA/WRITA issuance to ACTV issuance is fixed at CL + BL - 1, however, TYP can be set to 1001_B only when FCRAM is connected.
- This timing is effective, for example, for recurring page misses as it eliminates the cycle for issuing the PRE command.

■ Auto - refresh Operation Timing

Figure 9-5 shows auto - refresh operation timings.

Figure 9-5 Auto - refresh Timing Chart



- The refresh command is issued every " refresh control register's (RCR's) RFINT5 - RFINT0 value x 32 " cycles and access is restarted upon completion of each refresh.
- Set the TRC bit in the refresh control register (RCR) according to the SDRAM/FCRAM standards.
- Satisfy the maximum RAS active time as well.

9.1 Self Refresh

This section describes self - refreshing.

■ Self Refresh

Writing 1 to the SELF bit in the refresh control register (RCR) causes the SDRAM/FCRAM interface to initiate the self - refresh transition sequence.

After executing auto - refreshing the number of times set in the RFC2 to RFC0 bits, the SDRAM/FCRAM interface issues the SELF command to SDRAM/FCRAM to enter the self - refresh mode.

The device is released from the self - refresh mode either when 0 is written to the SELF bit or read/write access to SDRAM/FCRAM occurs.

The SDRAM/FCRAM interface issues the SELFX command to execute auto - refreshing the number of times set in the RFC2 to RFC0 bits upon detection of writing 0 to the SELF bit or access to SDRAM/FCRAM in the self - refresh mode.

Even when access to SDRAM/FCRAM by DMA transfer occurs after setting the self - refresh mode and putting the chip into sleep mode, the self - refresh mode is canceled.

- Self - refresh mode transition procedure

1. Set SELF bit to "1".
2. Issue the REF command the number of times set in the RFC2 to RFC0 bits.
3. Issue SELF command

- Self - refresh mode reset procedure

1. Set the SELF bit to 0 or access to SDRAM/FCRAM.
2. Issue SELFX command
3. Issue the REF command the number of times set in the RFC2 to RFC0 bits.
4. Transition to the normal access state

9.2 Power-on Sequence

This section describes the power - on sequence.

■ Power-on Sequence

Setting the PON bit in the refresh control register (RCR) to 1 initiates the power - on sequence.

Take the following steps to set the PON bit to 1 for transition to the power - on sequence.

1. Reserve the clock stabilization wait time specified in the SDRAM/FCRAM manual.
2. Set ACR, AWR, MCRA(B).
3. Set the CSER to enable the area to which SDRAM/FCRAM has been connected.
4. Set the PON bit to 1 while setting the RCR value.

Taking the above steps causes the SDRAM/FCRAM interface to execute the following power - on sequence.

5. Execute the PALL command.
6. Execute the REF command eight times.
7. The mode register is set according to the BST bit in the ACR, CL (CAS Latency) bit in the AWR, and the WBST bit in the MCRA.
8. Transition to the normal access state

9.3 Connecting SDRAM/FCRAM to Many Areas

This section shows the connecting SDRAM/FCRAM to many areas.

■ Connecting SDRAM/FCRAM to Many Areas

SDRAM/FCRAM can basically be set for all chip select areas. When connecting SDRAM/FCRAM to several areas, connect the same type of modules. Also it needs considerations about bus load when connecting SDRAM/FCRAM to several areas.

More precisely, connect the modules common in the following register settings.

- Area configuration register (ACR): Set all of the DBW1 - DBW0, BST1 - BST0, and TYP3 - TYP0 bits to the same.
- Area wait register (AWR): Set all the bits to the same.
- Memory setting register (MCR): All the settings are the same as the registers are common.)
- Refresh control register (RCR): All the settings are the same as the registers are common.)

To enable the two areas at a time, execute the power - on sequence, auto - refresh, and self - refresh at the same time.

9.4 Address Multiplexing Format

This section describes the address multiplexing format.

■ Address Multiplexing Format

SDRAM/FCRAM access addresses correspond to row, bank, and column addresses differently depending on the settings of the ASZ3 to ASZ0, DBW1 and DBW0, PSZ2 to PSZ0, and BANK bits.

Addresses are arranged in the order of Column, BANK, and Row addresses, starting from the least significant bit.

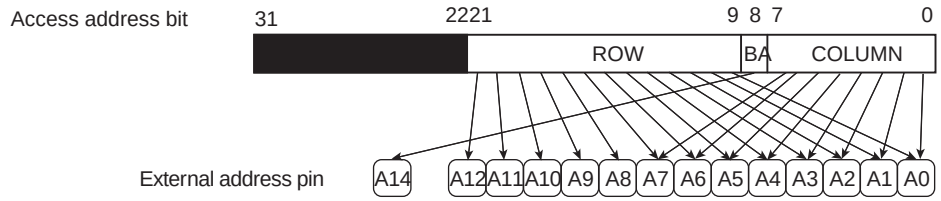
Set each bit as shown below.

- ASZ3 to ASZ0 bits: Set these bits to the total amount of SDRAM/FCRAM connected to the corresponding area. For using two modules in parallel, set the total amount. Affects the number of row addresses.
- DBW1 and DBW0 bits: Set these bits to the data bus width. (Set the bits to " 16 bits " for connecting a pair of eight - bit modules in parallel.) Column addresses are shifted according to the data bus width setting. 8 bits: Do not shift. 16 bits: Shift one bit. 32 bits: Shift two bits.
- PSZ2 to PSZ0 bits: Set these bits to the number of column addresses used for SDRAM/FCRAM.
- BANK bit: Set this bit to the number of SDRAM/FCRAM bank addresses.

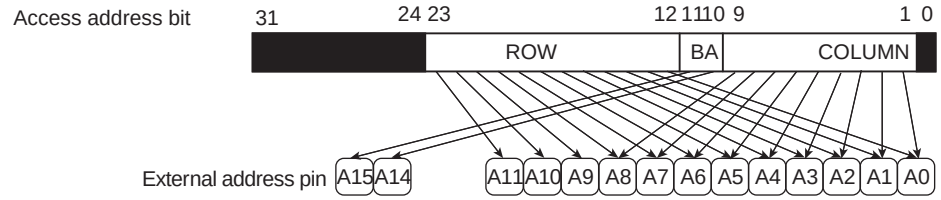
Figure 4.9 - 6 shows examples of combinations of access addresses and Row/BANK/Column addresses.

Figure 9-6 Examples of combinations of access addresses and Row/BANK/Column addresses

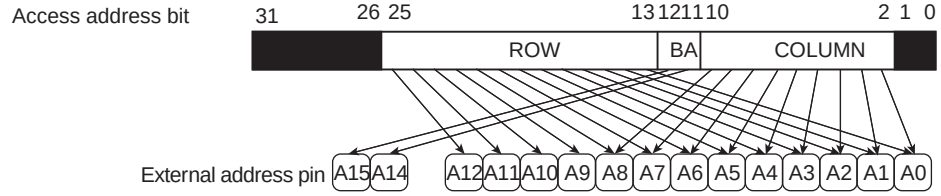
- 4M bytes (set ASZ to 0110B), 8-bit bus width (set DBW to 00B)
  256 column (set PSZ to 000B), 2 banks (set BANK to 0B)



- 16M bytes (set ASZ to 1000B), 16-bit bus width (set DBW to 01B)
  512 column (set PSZ to 001B), 4 banks (set BANK to 1B)



- 64M bytes (set ASZ to 1010B), 32-bit bus width (set DBW to 10B)
  512 column (set PSZ to 001B), 4 banks (set BANK to 1B)



9.5 Memory Connection Example

This section shows the memory connection example.

Memory Connection Example

The SDRAM/FCRAM interface is connected to SDRAM/FCRAM as shown in Table 4.9 - 1 in principle.

Table 9-1 SDRAM/FCRAM Interface to SDRAM/FCRAM Connection Table

SDRAM/ FCRAM interface pin	SDRAM/ FCRAM pin	Remarks
MCLKO	CLK	
MCLKE	CKE	
$\overline{\text{SRAS}}$ ($\overline{\text{AS}}$)	$\overline{\text{RAS}}$	
$\overline{\text{SCAS}}$ ($\overline{\text{BAA}}$)	$\overline{\text{CAS}}$	

Table 9-1 SDRAM/FCRAM Interface to SDRAM/FCRAM Connection Table

SDRAM/ FCRAM interface pin	SDRAM/ FCRAM pin	Remarks
$\overline{SWE}$ ($\overline{WR}$)	$\overline{WE}$	
$\overline{CS0}$ to $\overline{CS7}$	$\overline{CS}$	All chip select areas can be set as SDRAM/FCRAM space.
A0 to A9	A0 to A9	Addresses do not have to be shifted depending on the bus width.
A10/AP	A10/AP	A10 for row address output; otherwise AP
A11 to A13	A11 to A13	Connected to the address used for SDRAM/FCRAM.
A14	BA0	BA for 2 bank product
A15	BA1	The pin is not used for a two - bank module.
D31 to D0	DQ	The connection changes depending on the endian method and data bus width. For detailed connection, see Section " 4. Endian and Bus Access " .
DQMUU, DQMUL, DQMLU, DQMLL	DQM	The connection changes depending on the endian method and data bus width. For detailed connection, see Section " 4. Endian and Bus Access " .

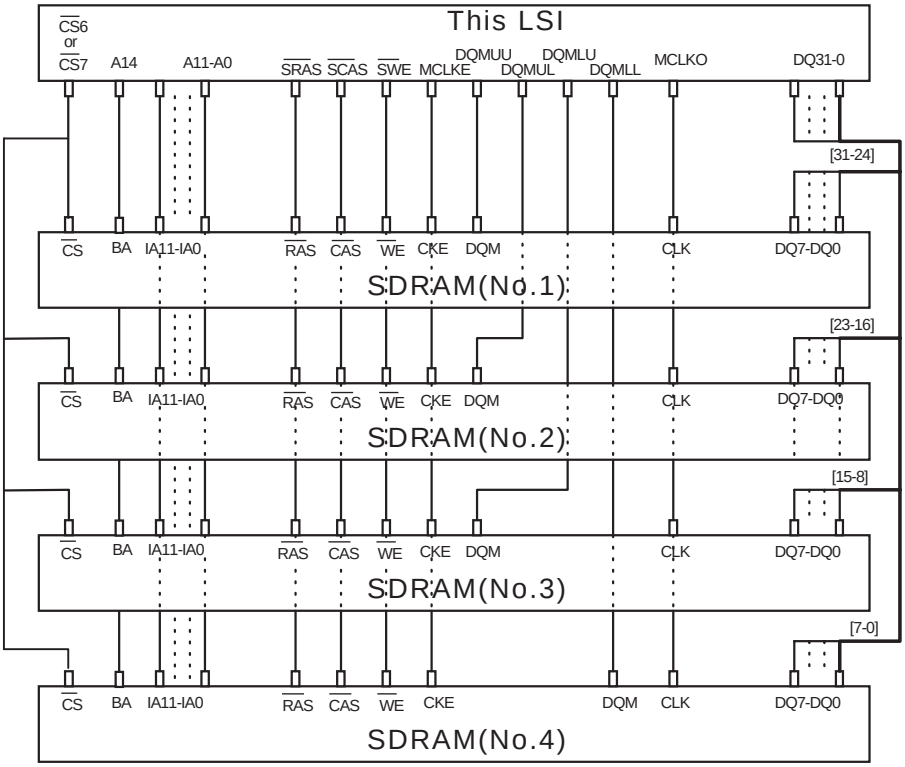
- Using 8 - bit SDRAM/FCRAM (Big endian)

Total data bus width of 32 bits: Use four SDRAM/FCRAM modules.

Total data bus width of 16 bits: Use two SDRAM/FCRAM modules.

Figure 9-7 shows how to use 64 - Mbit SDRAM (one bank address and 12 row addresses).

Figure 9-7 Using 64 - Mbit SDRAM



When SDRAM modules are used with a total data width of 16 bits, SDRAMs No. 3 and No. 4 are not required and DQ15 to DQ0 must be left open.

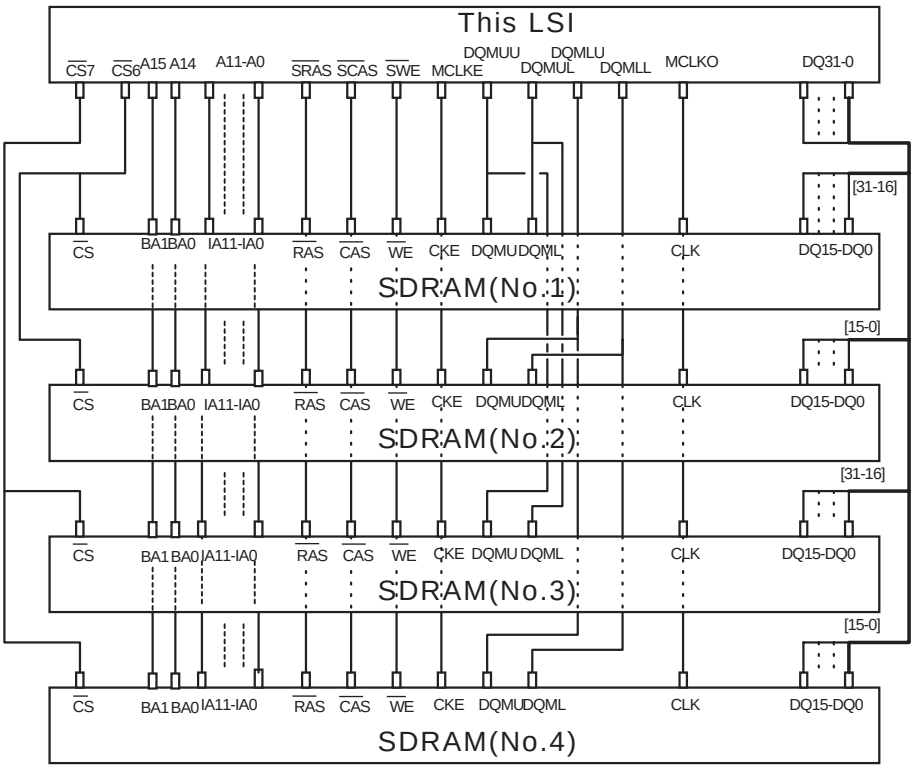
- Using 16 - bit SDRAM/FCRAM

Total data width of 32 bits: Use two or four SDRAM modules.

Total data width of 16 bits: Use one or two SDRAM modules.

Figure 9-8 shows how to use 64-Mbit SDRAM (two bank addresses and 12 row addresses).

Figure 9-8 Using 64 - Mbit SDRAM



When using one SDRAM module with a data width of 16 bits, SDRAMs No. 2, No. 3, and No. 4 are not required and DQ15 to DQ0 must be left open.

When two SDRAM modules are used with a data width of 16 bits, SDRAMs No. 2 and No. 4 are not required.

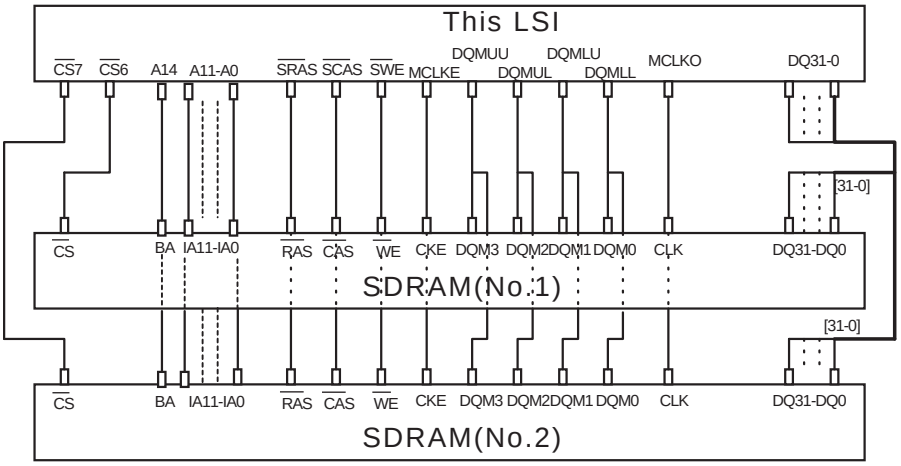
When two SDRAM modules are used with a data width of 32 bits, SDRAMs No. 3 and No. 4 are not required.

- Using 32 - bit SDRAM

When the data width is 32 bits: Use one or two SDRAM modules.

Figure 9-9 shows 64-Mbit SDRAM (one bank address and 12 row addresses).

Figure 9-9 Using 64 - Mbit



SDRAM No. 2 is not required when the device is used with only one SDRAM module.

10. DMA Access Operation

This section explains DMA access operation.

■ DMA Access Operation

This section explains the following five DMA operations:

- DMA fly-by transfer (I/O -> memory)
- DMA fly-by transfer (memory -> I/O)
- 2-cycle transfer (internal RAM -> I/O, RAM)
- 2-cycle transfer (external -> I/O)
- 2-cycle transfer (I/O -> external)

10.1 DMA Fly-By Transfer (I/O -> Memory)

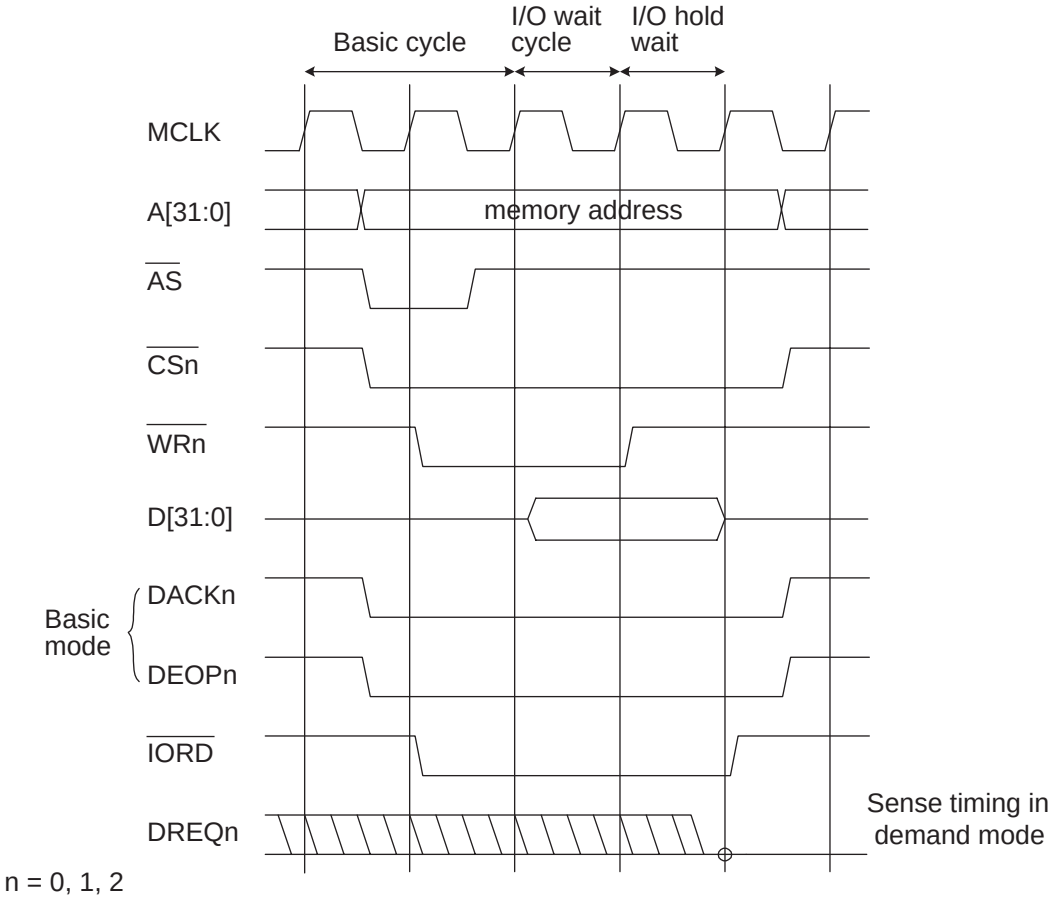
This section explains DMA fly-by transfer (I/O -> memory).

■ DMA Fly-By Transfer (I/O -> Memory)

Figure 10-1 "Timing Chart for DMA Fly-By Transfer (I/O -> Memory)" shows the operation timing chart for (TYP3-0=0000_B, AWR=0008_H, IOWR=41_H).

Figure 10-1 "Timing Chart for DMA Fly-By Transfer (I/O -> Memory)" shows when case when a wait is not set on the memory side.

Figure 10-1 Timing Chart for DMA Fly-By Transfer (I/O -> Memory)



- Setting 1 for the W01 bit of the AWR register enables the $\overline{CSn}$ -> $\overline{RD}/\overline{WRn}$ setup delay to be set. Set this bit to extend the period between assertion of chip select and the read/write strobe.
- Setting 1 for the W00 bit of the AWR register enables the $\overline{RD}/\overline{WRn}$ -> $\overline{CSn}$ hold delay to be set. Set this bit to extend the period between negation of the read/write strobe and negation of chip select.
- nThe $\overline{CSn}$ -> $\overline{RD}/\overline{WRn}$ setup delay (W01 bit) and $\overline{RD}/\overline{WRn}$ -> $\overline{CS}$ hold delay (W00 bit) can be set independently.
- When successive accesses are made within the same chip select area without negating the chip select, neither $\overline{CSn}$ -> $\overline{RD}/\overline{WRn}$ setup delay nor $\overline{RD}/\overline{WRn}$ -> $\overline{CSn}$ hold delay is inserted.
- If a setup cycle for determining the address or a hold cycle for determining the address is needed, set 1 for the address -> $\overline{CSn}$ delay setting (W02 bit of the AWR register).

For I/O on the data output side, a read strobe of three bus cycles extended by the I/O wait cycle and I/O hold wait cycle is generated. For memory on the receiving side, a write strobe of two bus cycles extended by the I/O wait cycle is generated. The I/O hold wait cycle does not affect the write strobe. However, the address and CS signal are retained until the fly-by bus access cycles end.

10.2 DMA Fly-By Transfer (Memory -> I/O)

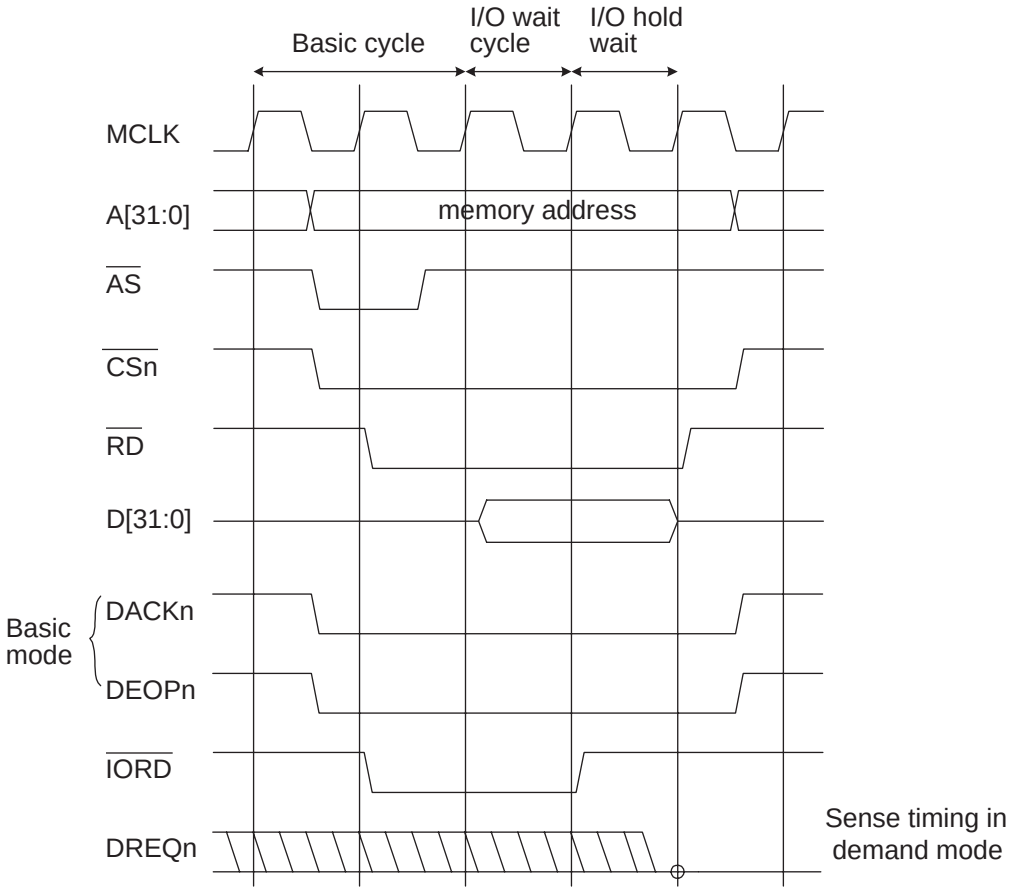
This section explains DMA fly-by transfer (memory -> I/O).

■ DMA Fly-By Transfer (Memory -> I/O)

Figure 10-2 "Timing chart for DMA Fly-By Transfer (Memory -> I/O)" shows the operation timing chart for (TYP3-0=0000_B, AWR=0008_H, IOWR=41_H).

Figure 10-2 "Timing chart for DMA Fly-By Transfer (Memory -> I/O)" shows a case in which a wait is not set on the memory side.

Figure 10-2 Timing chart for DMA Fly-By Transfer (Memory -> I/O)



- Setting 1 for the HLD bit of the IOWR0-3 registers extends the I/O read cycle by one cycle.
- Setting bits WR1-0 bits of the IOWR0-3 registers enables 0-3 write recovery cycles to be inserted.
- If the write recovery cycle is set to 1 or more, a write recovery cycle is always inserted after write access.
- Setting bits IW3-0 of the IOWR0-3 registers enables 0-15 wait cycles to be inserted.
- If wait is also set on the memory side (AWR15-12 is not 0), the larger value is used as the wait cycle after comparison with the I/O wait (IW3-0 bits).

Reference:

For memory on the data output side, a read strobe of three bus cycles extended by the I/O wait cycle and I/O hold wait cycle is generated. For I/O on the receiving side, a write strobe of two bus cycles extended by the I/O wait cycle is generated. The I/O hold wait cycle does not affect the write strobe. However, the address and CS signal are retained until the fly-by bus access cycles end.

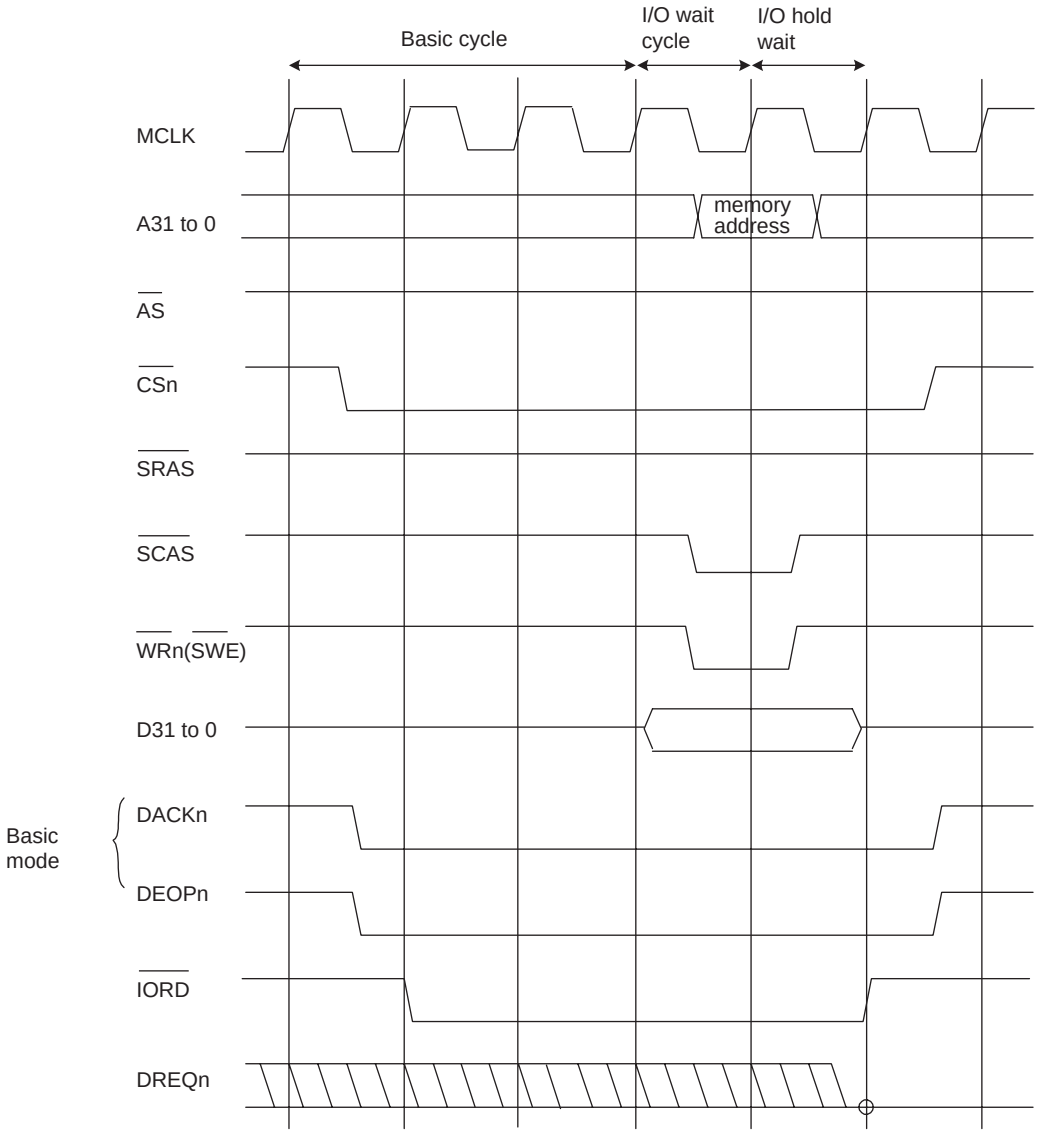
10.3 DMA Fly-By Transfer (I/O -> SDRAM/FCRAM)

This section describes the operation of DMA fly - by transfer (I/O device to SDRAM/FCRAM).

■ DMA Fly-By Transfer (I/O -> SDRAM/FCRAM)

Figure 4.10 - 3 shows an operation timing chart assuming TYP3 to TYP0 set to 1000B, AWR set to 0051H, and IOWR set to 41H.

Figure 10-3 Timing Chart for DMA Fly - by Transfer (I/O to SDRAM/FCRAM)



- For the I/O device on the data output side, a read strobe of three bus cycles extended by the I/O wait cycle and I/O hold wait cycle is generated.
- For SDRAM/FCRAM on the receiving side, a WRIT command is issued at the timing that allows writing after the I/O wait cycle. The I/O wait cycle may be longer depending on the SDRAM/FCRAM bank active state and SDRAM/FCRAM wait setting.
- The I/O hold wait cycle does not affect the write strobe. Note, however, that the CS signal is retained until the fly - by bus access cycles end.
- For fly - by transfer from an I/O device to SDRAM/FCRAM, be sure to set the HLD bit in the DMAC I/O wait register (IOWR) to 1 to enable the I/O hold wait cycle.
- Fly - by transfer must always be performed between data buses having the same bus width.

10.4 DMA Fly-By Transfer (SDRAM/FCRAM -> I/O)

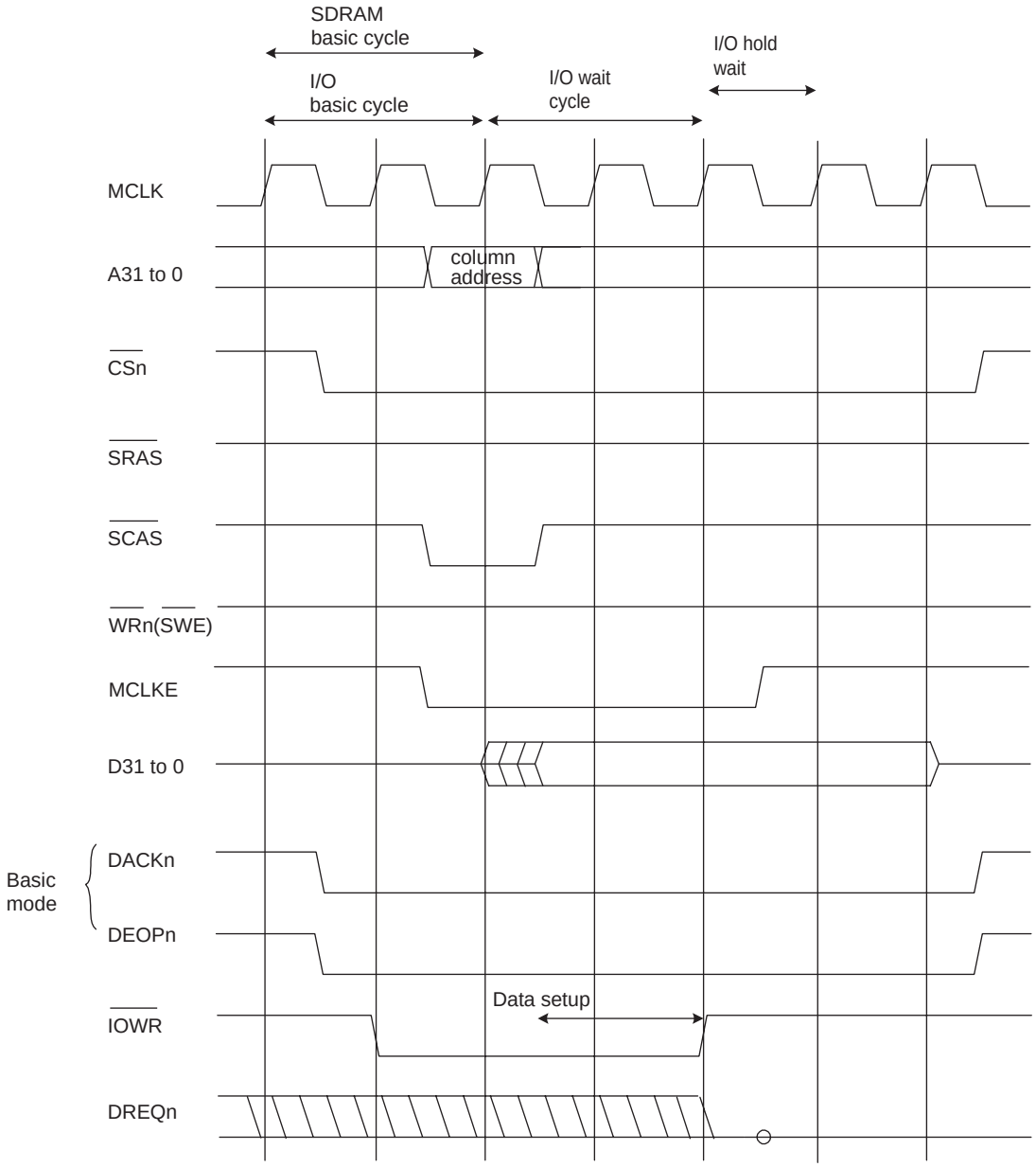
This section describes the operation of DMA fly - by transfer (SDRAM/FCRAM device to I/O).

■ DMA Fly-By Transfer (SDRAM/FCRAM -> I/O)

Figure 1.10 - 4 shows an operation timing chart assuming TYP3 to TYP0 set to 1000B, AWR set to 0051H, and IOWR set to 42H.

● At SDRAM page hit (Shortest)

Figure 10-4 Timing Chart for DMA Fly - by Transfer (SDRAM/FCRAM to I/O) with Page Hits (Shortest)

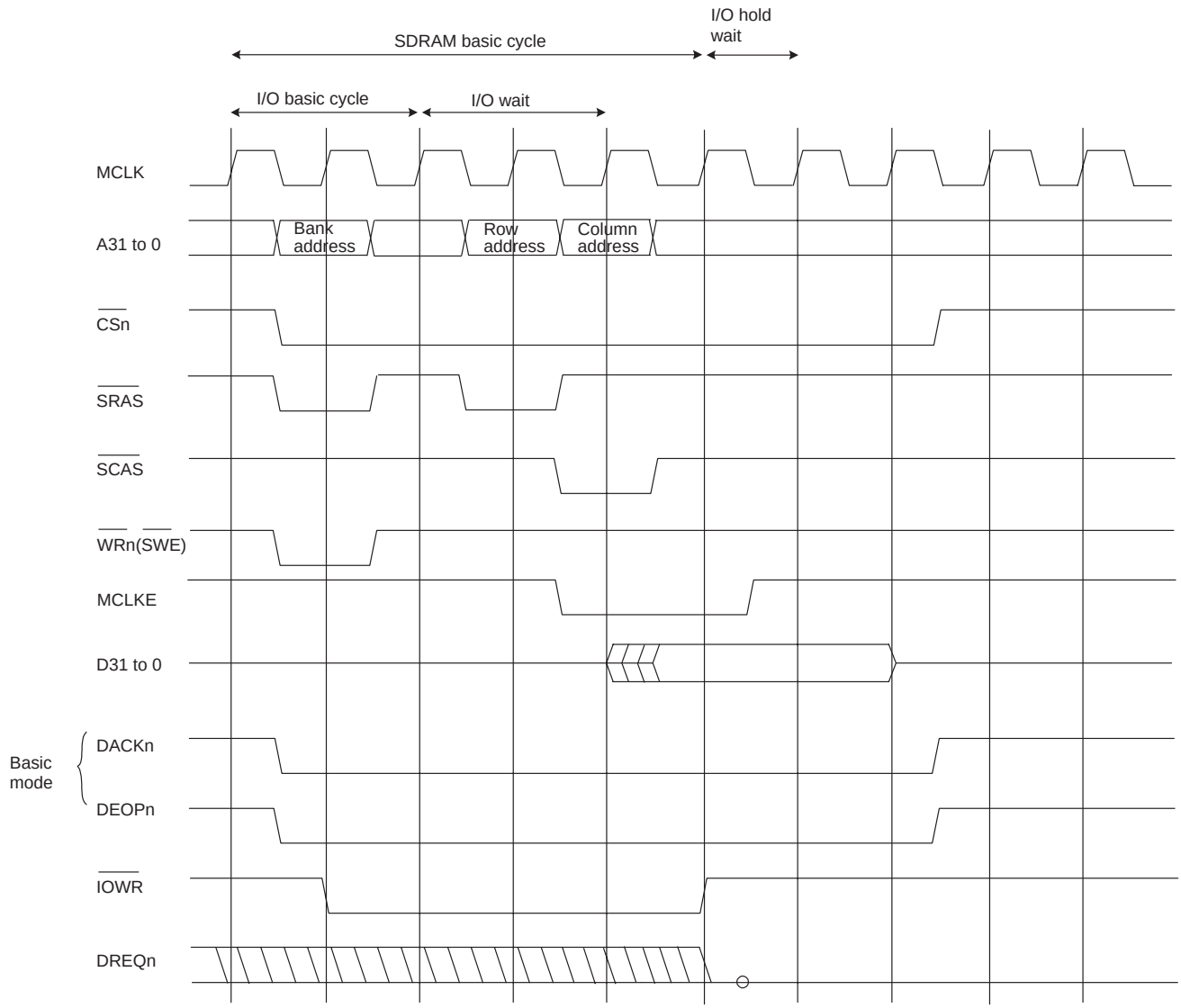


If SDRAM access is shorter than I/O access, the SDRAM access is extended by the I/O access (base access plus I/O wait).

Figure 4.10 - 5 shows an operation timing chart assuming TYP3 to TYP0 set to 1000B, AWR set to 0051H, and IOWR set to 42H.

- At SDRAM page misses

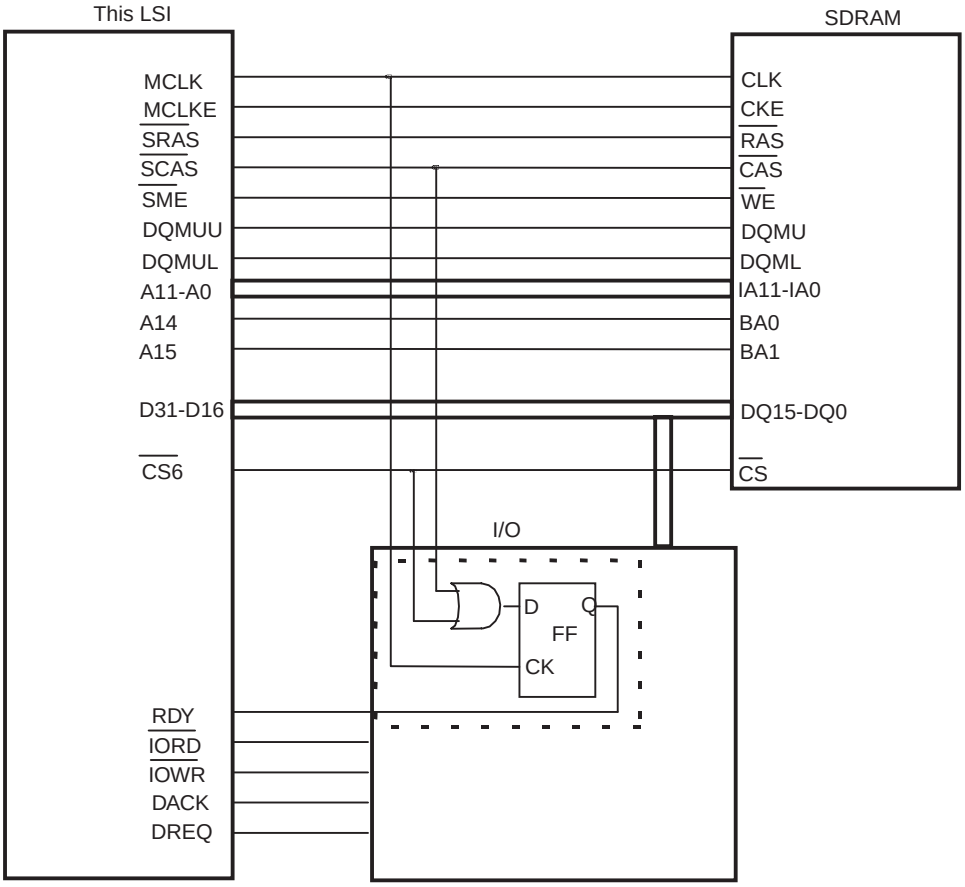
Figure 10-5 Timing Chart for DMA Fly - by Transfer (SDRAM/FCRAM to I/O) with Page Misses



- If SDRAM access is extended, for example, by precharging when a page miss occurs in reference to SDRAM, the SDRAM access exceeds the set I/O access, so that the I/O access is extended to be longer than the SDRAM access. When the I/O device requires data setup, therefore, the I/O wait cycle must be set such that I/O access is longer than the maximum SDRAM access cycle. For the above settings, set the number of I/O wait cycles to at least 4.
- For SDRAM/FCRAM on the data output side, a READ command is issued at the timing that satisfies the I/O wait cycle. If the I/O hold cycle has been set, then, a DESL command is issued to insert the I/O hold cycle in the cycle immediately followed by the

- For the I/O device on the receiving side, a write strobe of two bus cycles extended by the I/O wait cycle is generated.
- The I/O hold wait cycle does not affect the write strobe.
- Fly - by transfer must always be performed between data buses having the same bus width.
- When the I/O wait cycle is used to reserve data setup time, the I/O wait value must be set according to the page miss condition. A page hit therefore generates a penalty. If this penalty generated at a page hit causes a problem, prepare an external circuit as illustrated in Figure 4.10 - 6c to use an external wait cycle based on the CAS signal, thereby extending I/O access to reserve data setup time.

Figure 10-6 Sample Circuit Solving a Fly - by Penalty Using External Wait Cycles Based on the CAS Signal (CL = 2)

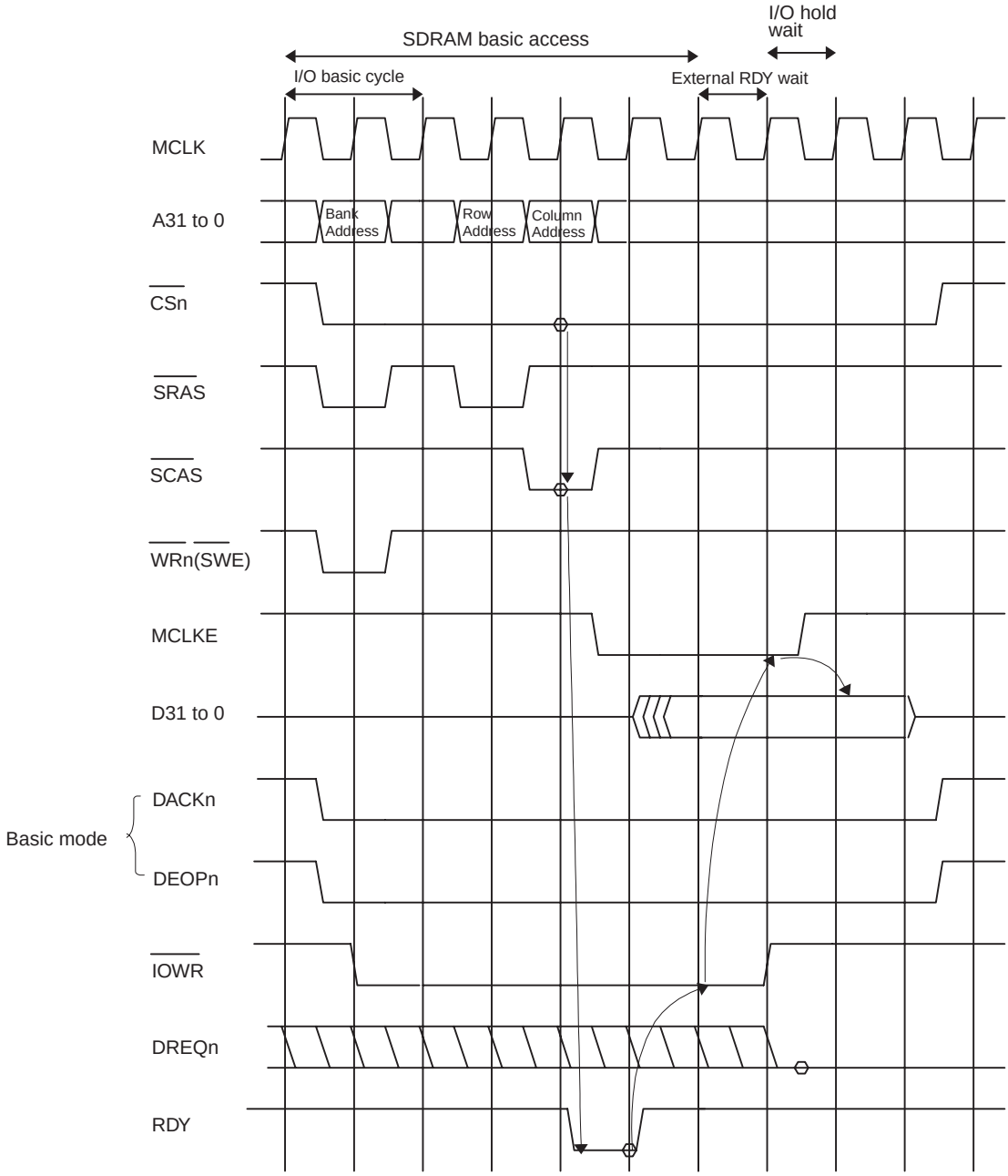


Note:

- For CL = 3, provide two stages of MCLK - based FF to cause a delay of another cycle.
- If any device requires an external wait cycle, add a logic gate to the RDY signal as required.

Figure 10-7 Timing Chart for Fly - by Penalty Solution Using External Wait Cycles Based on the CAS

Signal (CL = 2)



The rise of the IOWR signal can be delayed one cycle by extending SDRAM read access one cycle when the signal resulting from OR (negative - logic AND) operation of the CAS signal and the chip select signal for the SDRAM area subject to transfer is input t

As the external wait signal is generated based on the CAS signal rise timing in this case, the data setup time from the SDRAM data output to the I/O device can be reserved for one cycle, regardless of a page hit or miss in SDRAM.

Set the external wait using the RYE0 and RYE1 bits in the DMAC I/O wait register such that the RDY function of the DMA fly - by access channel to be used is enabled.

When the CAS latency is 3, SDRAM data output is delayed one cycle. Add one stage of FF by the MCLK to input the signal delayed one cycle from the above diagram to the RDY pin.

10.5 2-Cycle Transfer (Internal RAM -> External I/O, RAM)

This section explains 2-cycle transfer (internal RAM -> external I/O, RAM) operation.

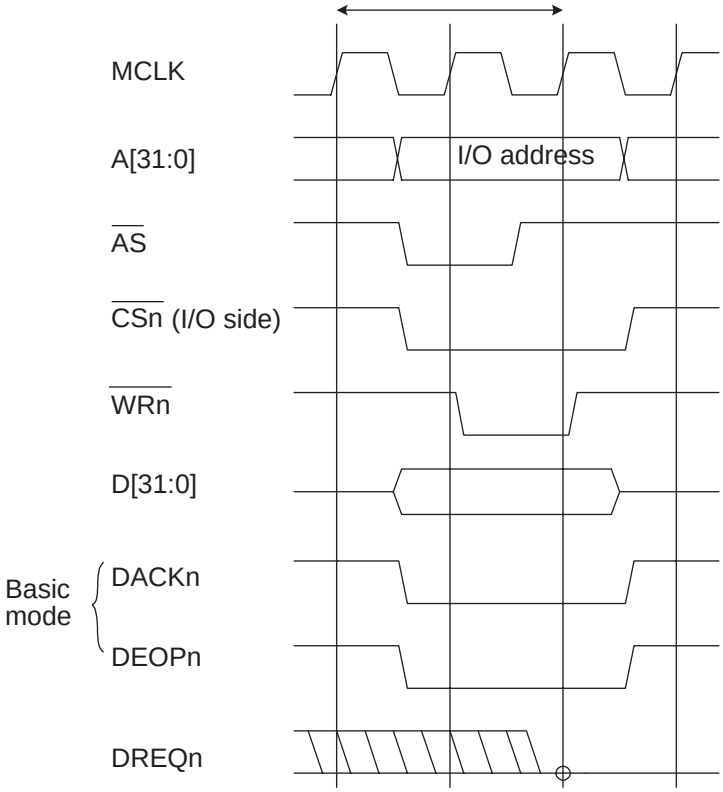
The timing is the same as for external I/O, RAM -> internal RAM.

■ 2-Cycle Transfer (Internal RAM -> External I/O, RAM)

Figure 10-8 "Timing Chart for 2-cycle Transfer (Internal RAM -> External I/O, RAM)" shows the operation timing chart for (TYP3-0=0000_B, AWR=0008_H, IOWR=00_H).

Figure 10-8 "Timing Chart for 2-cycle Transfer (Internal RAM -> External I/O, RAM)" shows a case in which a wait is not set on the I/O side.

Figure 10-8 Timing Chart for 2-cycle Transfer (Internal RAM -> External I/O, RAM)



- The bus is accessed in the same way as an interface when DMAC transfer is not performed.
- DACKn/DEOPn is not output in the internal RAM access cycles.

10.6 2-Cycle Transfer (External -> I/O)

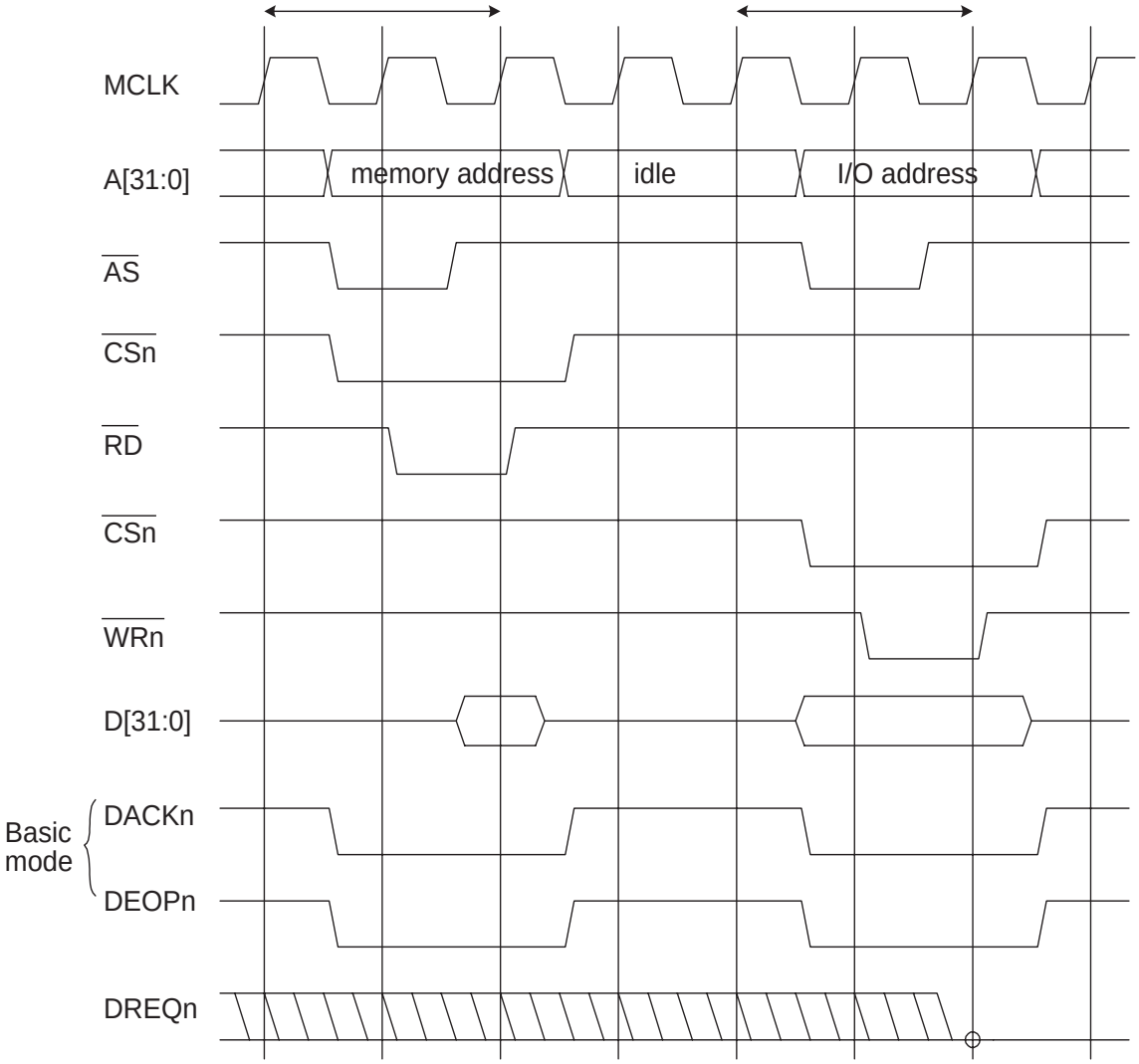
This section explains 2-cycle transfer (external -> I/O) operation.

■ 2-Cycle Transfer (External -> I/O)

Figure 10-9 "Timing Chart for 2-Cycle Transfer (External -> I/O" shows the operation timing chart for (TYP3-0=0000_B, AWR=0008_H, IOWR=00_H).

Figure 10-9 "Timing Chart for 2-Cycle Transfer (External -> I/O" shows a case in which a wait is not set for memory and I/O.

Figure 10-9 Timing Chart for 2-Cycle Transfer (External -> I/O)



- The bus is accessed in the same way as an interface when the DMAC transfer is not performed.
- In basic mode, DACKn/DEOPn is output in both transfer source bus access and transfer destination bus access.

10.7 2-Cycle Transfer (I/O -> External)

This section explains 2-cycle transfer (I/O -> external) operation.

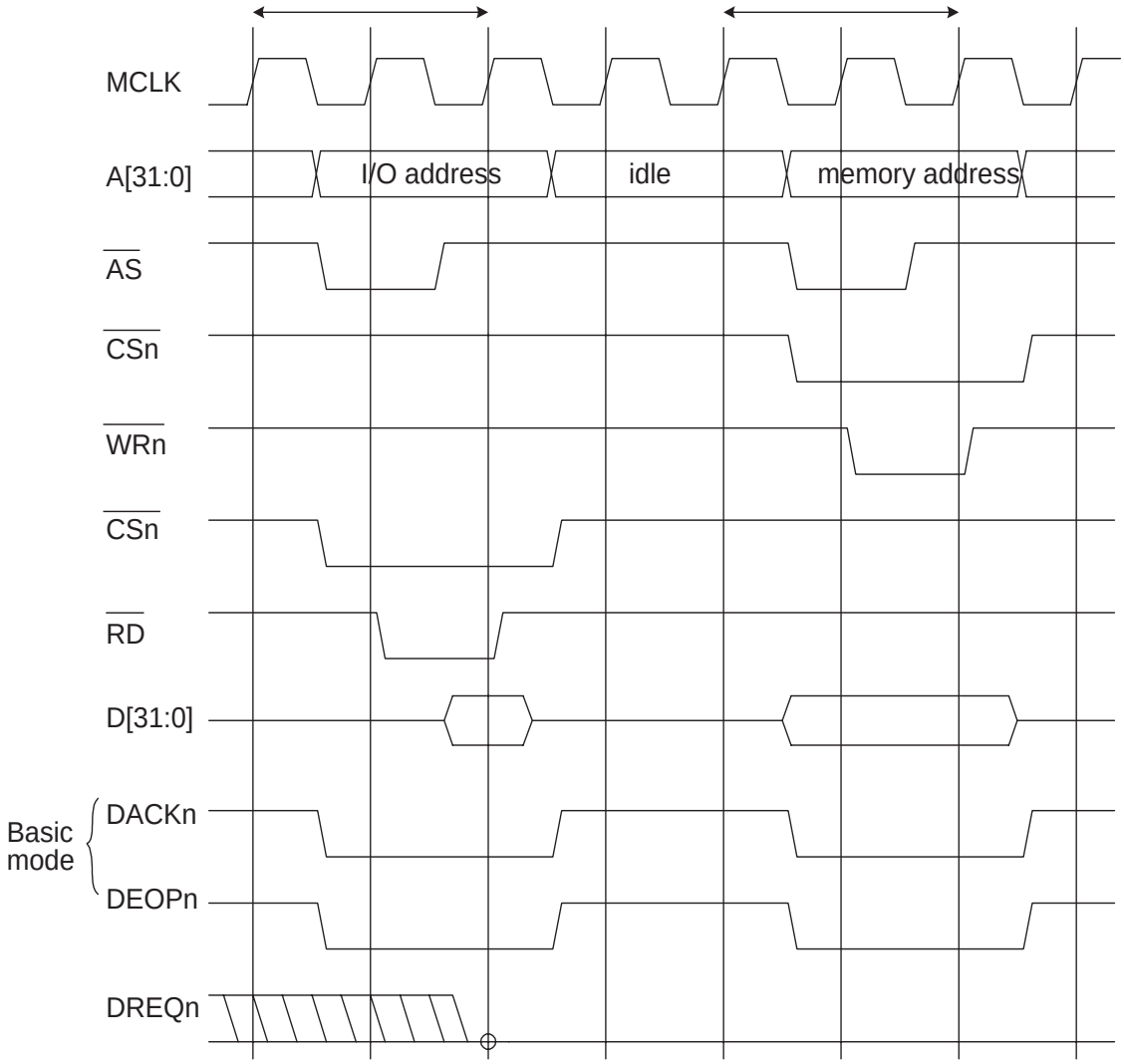
■ 2-Cycle Transfer (I/O -> External)

Figure 10-10 "Timing Chart for 2-Cycle Transfer (I/O -> External)" shows the operation timing chart for (TYP3-0=0000_B, AWR=0008_H, IOWR=00_H).

Figure 10-10 "Timing Chart for 2-Cycle Transfer (I/O -> External)" shows a case in which a wait is not set for

memory and I/O.

Figure 10-10 Timing Chart for 2-Cycle Transfer (I/O -> External)



- The bus is accessed in the same way as an interface when the DMAC transfer is not performed.
- In basic mode, DACKn/DEOPn is output both in the transfer source bus access and transfer destination bus access.

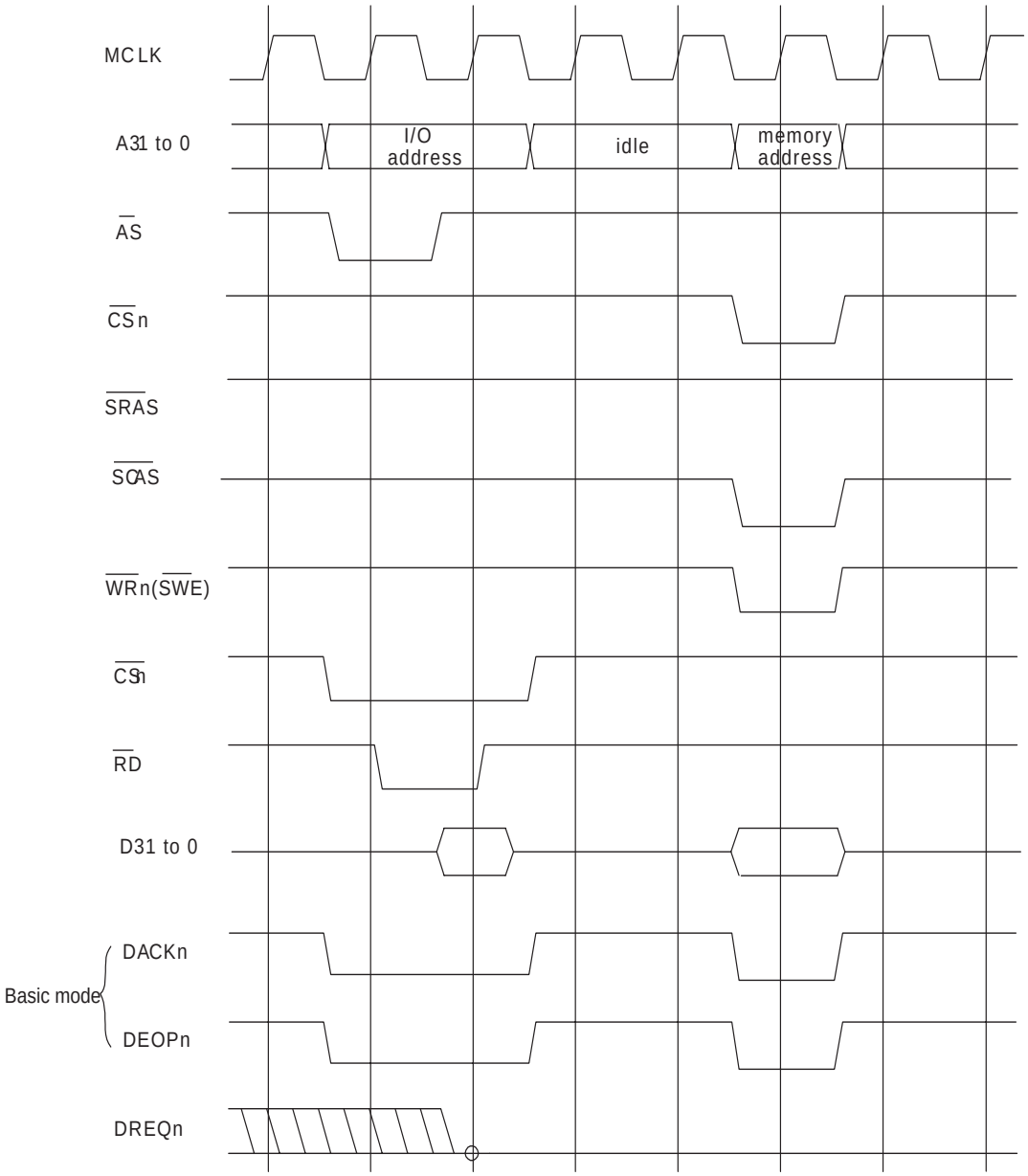
10.8 2-Cycle Transfer (I/O -> SDRAM/FCRAM)

This section describes the operation of two - cycle transfer (I/O device to SDRAM/FCRAM).

■ 2-Cycle Transfer (I/O -> SDRAM/FCRAM)

Figure 4.10 - 11 shows an operation timing chart assuming TYP3 to TYP0 set to 1000B, AWR set to 0051H, and IOWR set to 00H.

Figure 10-11 Timing Chart for Two - cycle Transfer (I/O to SDRAM/FCRAM)



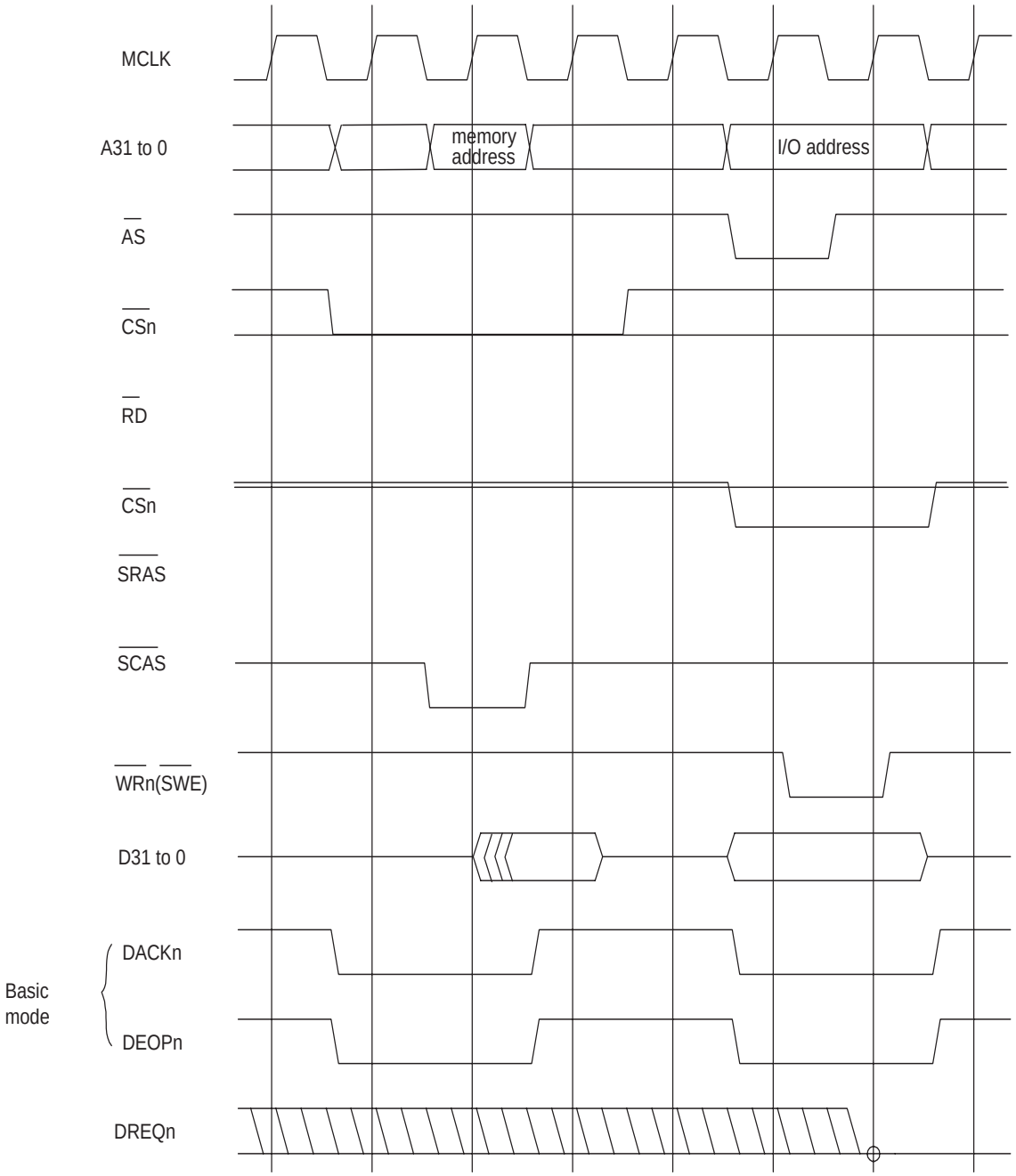
10.9 2-Cycle Transfer (SDRAM/FCRAM -> I/O)

This section describes the operation of two - cycle transfer (SDRAM/FCRAM to I/O device).

■ 2-Cycle Transfer (SDRAM/FCRAM -> I/O)

Figure 1.10 - 12 shows a timing chart for two - cycle transfer (SDRAM/FCRAM to I/O)

Figure 10-12 Timing Chart for Two - cycle Transfer (SDRAM/FCRAM to I/O)



- Bus access is the same as that of the interface for non - DMA transfer.
- In base mode, DACKn/DEOPn is output at both of transfer source bus access and transfer destination bus access.

11. Bus Arbitration

This section shows timing charts for releasing the bus right and for acquiring the bus right.

■ Releasing the Bus Right

Figure 11-1 "Timing Chart for Releasing the Bus Right" shows the timing chart for releasing the bus right. Figure 11-2 "Timing Chart for Releasing the Bus Right" shows the timing chart for acquiring the bus right.

Figure 11-1 Timing Chart for Releasing the Bus Right

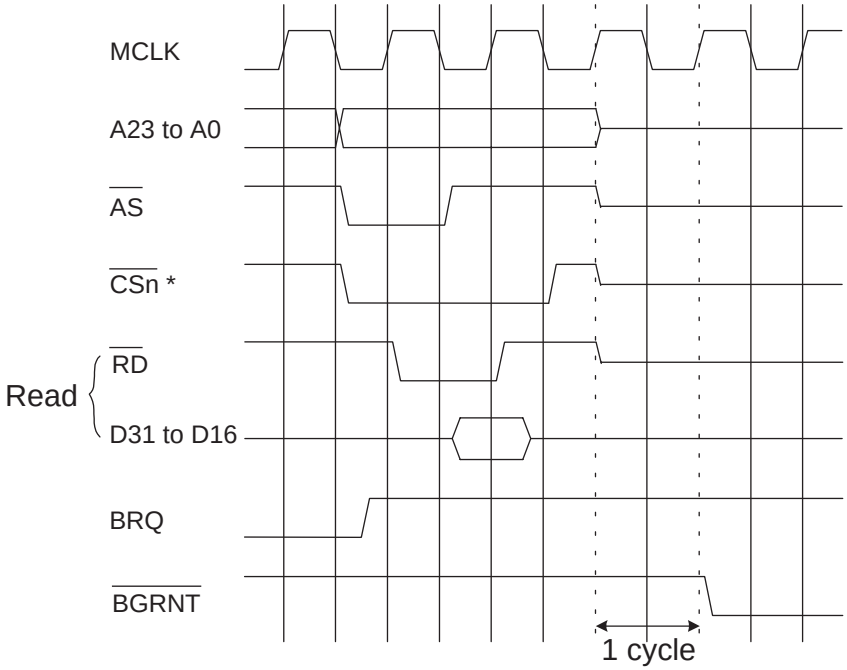
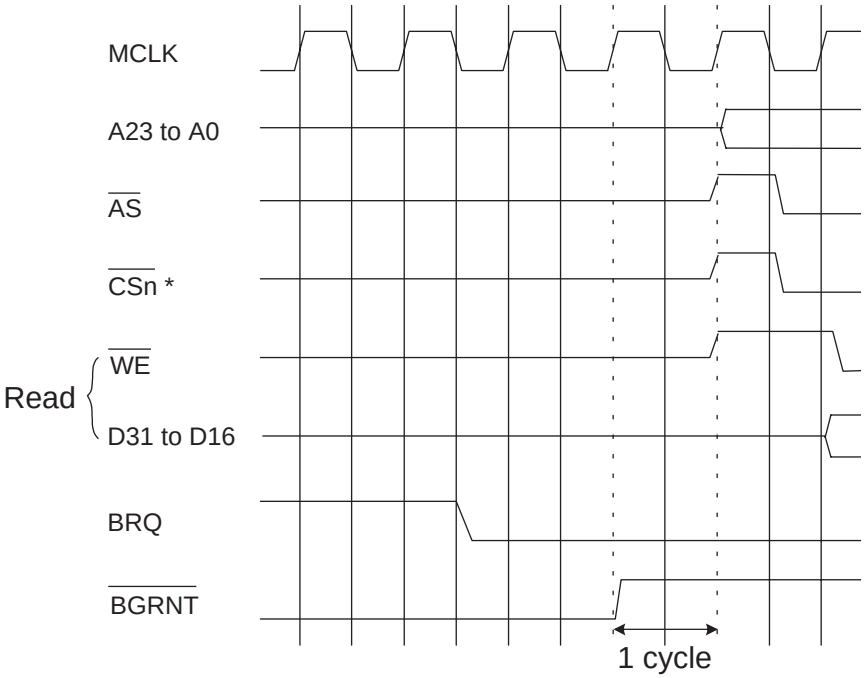


Figure 11-2 Timing Chart for Acquiring the Bus Right



- Setting 1 for the BREN bit of the TRC register enables bus arbitration by BRQ/BGRNT to be performed.
- When the bus right is released, the pin is set to high impedance and then $\overline{\text{BGRNT}}$ is asserted one cycle later.
- When the bus right is acquired, $\overline{\text{BGRNT}}$ is negated and then each pin is activated one cycle later.
- $\overline{\text{CSn}}$ is set to high impedance only if the SREN bit in the ACR0-7 registers is set.
- If all areas enabled by the CSER register are shared (the SREN bit of the ACR register is 1), $\overline{\text{AS}}$, $\overline{\text{BAA}}$, $\overline{\text{RD}}$, $\overline{\text{WE}}$, and $\overline{\text{WR0}}$-$\overline{\text{WR3}}$ are set to high impedance.

12. Procedure for Setting a Register

This section explains the procedure for setting a register.

■ Procedure for Setting a Register

Using the following procedures to make external bus interface settings:

1. Before rewriting the contents of a register, be sure to set the CSER register so that the corresponding area is not used (0). If you change the settings while 1 is set, access before and after the change cannot be guaranteed.
2. Use the following procedure to change a register:
   - Set 0 for the CSER bit corresponding to the applicable area.
   - Set both ASR and ACR at the same time using word access.
   - Set AWR.
   - Set the CHER bit corresponding to the applicable area.
   - Set the CSER bit corresponding to the applicable area.
3. The $\overline{CS0}$ area is enabled after a reset is released. If the area is used as a program area, the register contents need to be rewritten while the CSER bit is 1. In this case, make the settings described in 2) to 4) above in the initial state with a low-speed internal clock. Then, switch the clock to a high-speed clock.
4. Use the following procedure to change the register value in an area for which prefetch:
   - Set 0 for the bit of CSER corresponding to the applicable area.
   - Set 1 for both the PSUS bit and PCLR bit of the TCR register.
   - Set both ASR and ACR at the same time using word access.
   - Set AWR.
   - Set the CHER bit corresponding to the applicable area.
   - Set 0 for both the PSUS bit and PCLR bit of the TCR register.
   - Set 1 for the bit of CSER corresponding to the applicable area.

13. Notes on Using the External Bus Interface

This section explains some notes when using the external bus interface.

■ Notes for Use

If settings are made so that the area (TYP3-0=0x0x_B) where $\overline{WR0}$-$\overline{WR3}$ are used as a write strobe and the area (TYP3-0=0x1x_B) where $\overline{WR}$ is used as a write strobe are mixed, be sure to make the following setting in all areas that will be used:

- Set at least one read -> write idle cycle (other than AWR W07-W06=00_B).
- Set at least one write recovery cycle (other than AWR W05-W04=00_B).

However, if $\overline{WR0}$-$\overline{WR3}$ are disabled (ROM only is connected) in the area (TYP3-0=0x0x_B) where $\overline{WR0}$-$\overline{WR3}$ are used as a write strobe, the above restriction does not apply. Also, the above restriction does not apply if both the address -> $\overline{RD}$/$\overline{WRn}$ setup cycle (W01=1) and $\overline{RD}$/$\overline{WRn}$ -> address hold cycle (W00=1) are set in the area (TYP3-0=0x1x_B) where $\overline{WE}$ is used as a write strobe.

When PFR10_5 is set to ’1’ the memory clock MCLKO (GP10_4) must be fed back externally to the MCLKI pin (GP10_5). When PFR10_5 is set to ’0’ the memory clock MCLKO is fed back internally. In this case the MCLKI pin (GP10_5) can be used a general purpose IO. Setting PFR10_5 to ’0’ (internal feed back) is the recommended operation mode.

Chapter 32 USART (LIN / FIFO)

1. Overview

This chapter explains the function and operation of the USART. The USART with LIN (Local Interconnect Network) - Function is a general-purpose serial data communication interface for performing synchronous or asynchronous communication with external devices. 16 bytes transmission and reception FIFOs are available for selected channels.

The USART provides bidirectional communication function (normal mode), master-slave communication function (multiprocessor mode in master/slave systems), and special features for LIN-bus systems (working both as master or as slave device).

(Note) This chapter only lists the registers and addresses of USART04. Please refer to the IO-Map for the addresses of the other USARTs.

■ USART Functions

USART is a general-purpose serial data communication interface for transmitting serial data to and receiving data from another CPU or peripheral devices. It has the functions listed in table 1-1**.**

Table 1-1 USART functions

Item	Function
Data buffer	Full-duplex
Serial Input	5 times oversampling in asynchronous mode
Transfer mode	- Clock synchronous (start-stop synchronization and start-stop-bit-option) - Clock asynchronous (using start-, stop-bits)
Baud rate	- A dedicated baud rate generator is provided, which consists of a 15-bit-reload counter - An external clock can be input and also be adjusted by the reload counter
Data length	- 7 bits (not in synchronous or LIN mode) - 8 bits
Signal mode	Non-return to zero (NRZ) and return to zero (RZ)
Start bit timing	Clock synchronization to the falling edge of the start bit in asynchronous mode
Reception error detection	- Framing error - Overrun error - Parity error
Interrupt request	- Reception interrupt (reception complete, reception error detect, Bus-Idle, LIN-Synch-break detect) - Transmission interrupt (transmission complete)

Table 1-1 USART functions (continued)

Master-slave communication function (multiprocessor mode)	One-to-n communication (one master to n slaves) (This function is supported both for master and slave system).
Synchronous mode	Function as Master- or Slave-USART
Transceiving pins	Direct access possible
LIN bus options	- Operation as master device - Operation as slave device - Generation of LIN-Sync-break - Detection of LIN-Sync-break - Detection of start/stop edges in LIN-Sync-field connected to ICU 0 and 2
Synchronous serial clock	The synchronous serial clock can be output continuously on the SCK pin for synchronous communication with start & stop bits
Clock delay option	Special synchronous Clock Mode for delaying clock (useful for SPI-compliance)
16 word deep FIFO	FIFO can be activated with receive progammable trigger level

■ USART operation modes

The USART operates in four different modes, which are determined by the MD0- and the MD1-bit of the Serial mode register (SMR04). Mode 0 and 2 are used for bidirectional serial communication, mode 1 for master/ slave communication and mode 3 for LIN master/slave communication.

Table 1-2 USART operation modes

Operation mode		Data length		Synchroniza-tion of mode	Length of stop bit	data bit directio n*
		parity disabled	parity enabled			
0	normal mode	7 or 8		asynchronous	1 or 2	L/M
1	multiprocessor	7 or 8 + 1**	-	asynchronous	1 or 2	L/M
2	normal mode	8		synchronous	0, 1 or 2	L/M
3	LIN mode	8	-	asynchronous	1	L

* means the data bit transfer format: LSB or MSB first.

** "+1" means the indicator bit of the address/data selection in the multiprocessor mode, instead of parity.

Mode 1 operation is supported both for master or slave operation of USART in a master-slave connection system. In Mode 3 the USART function is locked to 8N1-Format, LSB first.

If the mode is changed, USART cuts off all possible transmission or reception and then awaits new action.

The MD1 and MD0 bits of the Serial Mode Register (SMR04) determine the operation mode of USART04 as shown in the following table:

Table 1-3 Mode Bit Setting

MD1	MD0	Mode	Description
0	0	0	Asynchronous (normal mode)
0	1	1	Asynchronous (multiprocessor mode)
1	0	2	Synchronous (normal mode)
1	1	3	Asynchronous (LIN mode)

■ USART Interrupts

Table 1-4 USART interrupts

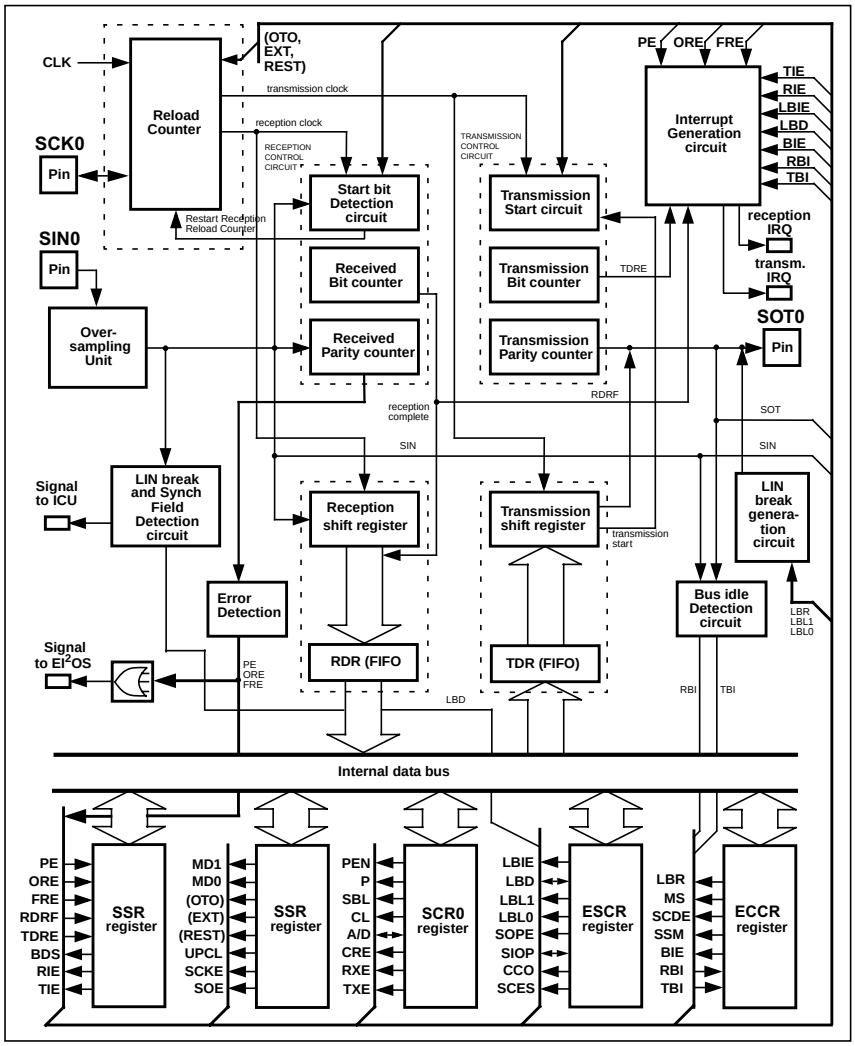
Interrupt cause	Interrupt number	Interrupt control register		Interrupt Vector	
		Register name	Address	Offset	Default address
USART04 reception interrupt	#66 (42_H)	ICR25	0459_H	$2F4_H$	$000FFEF4_H$
USART04 transmission interrupt	#67 (43_H)	ICR25	0459_H	$2F0_H$	$000FFEF0_H$

2. USART Configuration

■ USART consists of the following blocks:

- Reload Counter
- Reception Control Circuit
- Reception Shift Register
- Reception Data Register
- Transmission Control Circuit
- Transmission Shift Register
- Transmission Data Register
- Error Detection Circuit
- Oversampling Unit
- Interrupt Generation Circuit
- LIN Break Generation
- LIN Break and Synch Field Detection
- Bus Idle Detection Circuit
- Serial Mode Register (SMR04)
- Serial Control Register (SCR04)
- Serial Status Register (SSR04)
- Extended Com. Contr. Reg. (ECCR04)
- Extended Status/Contr. Reg. (ESCR04)
- FIFO Control Register (FCR04)
- FIFO Status Register (FSR04)

Figure 2-1 USART Block Diagram



■ Explanation of the different blocks

- Reload Counter

The reload counter functions as the dedicated baud rate generator. It can select external input clock or internal clock for the transmitting and receiving clocks. The reload counter has a 15 bit register for the reload value. The actual count of the transmission reload counter can be read via the BGR0/1 registers.

- Reception Control Circuit

The reception control circuit consists of a received bit counter, start bit detection circuit, and received parity counter. The received bit counter counts reception data bits. When reception of one data item for the specified data length is complete, the received bit counter sets the Reception data register full flag. When the FIFO is enabeld, the flag is set if the triggerlevel is reached. The start bit detection circuit detects start bits from the serial input signal and sends a signal to the reload counter to synchronize it to the falling edge of these start bits. The reception parity counter calculates the parity of the reception data.

- Reception Shift Register

The reception shift register fetches reception data input from the SIN04 pin, shifting the data bit by bit. When reception is complete, the reception shift register transfers receive data to the RDR04 register.

- Reception Data Register

This register retains reception data. Serial input data is converted and stored in this register. If the FIFO is enabled up to 16 receptions can be saved, the trigger level is progammable.

- Transmission Control Circuit

The transmission control circuit consists of a transmission bit counter, transmission start circuit, and transmission parity counter.The transmission bit counter counts transmission data bits. When the transmission of one data item of the specified data length is complete, the transmission bit counter sets the Transmission data register full flag. The transmission start circuit starts transmission when data is written to TDR04. The transmission parity counter generates a parity bit for data to be transmitted if parity is enabled.

- Transmission Shift Register

The transmission shift register transfers data written to the TDR04 register to itself and outputs the data to the SOT04 pin, shifting the data bit by bit.

- Transmission Data Register

This register sets transmission data. Data written to this register is converted to serial data and output.

If the FIFO is enabled up to 16 transmissions can be saved and continuously transmitted

- Error Detection Circuit

The error detection circuit checks if there was any error during the last reception. If an error has occurred it sets the corresponding error flags.

- Oversampling Unit

The oversampling unit oversamples the incoming data at the SIN04 pin for five times. It is switched off in synchronous operation mode.

- Interrupt Generation Circuit

The interrupt generation circuit administers all cases of generating a reception or transmission interrupt. If a corresponding enable flag is set and an interrupt case occurs the interrupt will be generated immediately.

- LIN Break and Synchronization Field Detection Circuit

The LIN break and LIN synchronization field detection circuit detects a LIN break, if a LIN master node is sending a message header. If a LIN break is detected a special flag bit is generated. The first and the fifth falling edge of the synchronization field is recognized by this circuit by generating an internal signal for the Input Capture Unit to measure the actual serial clock time of the transmitting master node.

- LIN Break Generation Circuit

The LIN break generation circuit generates a LIN break of a determined length.

- Bus Idle Detection circuit

The bus idle detection circuit recognizes if neither reception nor transmission is going on. In this case the circuit generates a special flag bit.

- Serial Mode Register

This register performs the following operations:

  - Selecting the USART operation mode
  - Selecting a clock input source
  - Selecting if an external clock is connected “one-to-one” or connected to the reload counter
  - Resetting the USART (preserving the settings of the registers)
  - Specifying whether to enable serial data output to the corresponding pin
  - Specifying whether to enable clock output to the corresponding pin

- Serial Control Register

This register performs the following operations:

  - Specifying whether to provide parity bits
  - Selecting parity bits
  - Specifying a stop bit length

- Specifying a data length
- Selecting a frame data format in mode 1
- Clearing the error flags
- Specifying whether to enable transmission
- Specifying whether to enable reception

• Serial Status Register

This register checks the transmission and reception status and error status, and enables and disables transmission and reception interrupt requests.

• Extended Status/Control Register

This register provides several LIN functions, direct access to the SIN04 and SOT04 pin and setting for the USART synchronous clock mode.

• Extended Communication Control Register

The extended communication control register provides bus idle recognition interrupt settings, synchronous clock settings, and the LIN break generation.

• FIFO Control Register

With the FCR4 register the TX/RX FIFOs can be enabled, the RX interrupt triggerlevel can be set and the FIFO status register mode can be set.

• FIFO Status Register

With the FSR4 register shows the number of valid RX/TX data in the FIFO buffers.

3. USART Pins

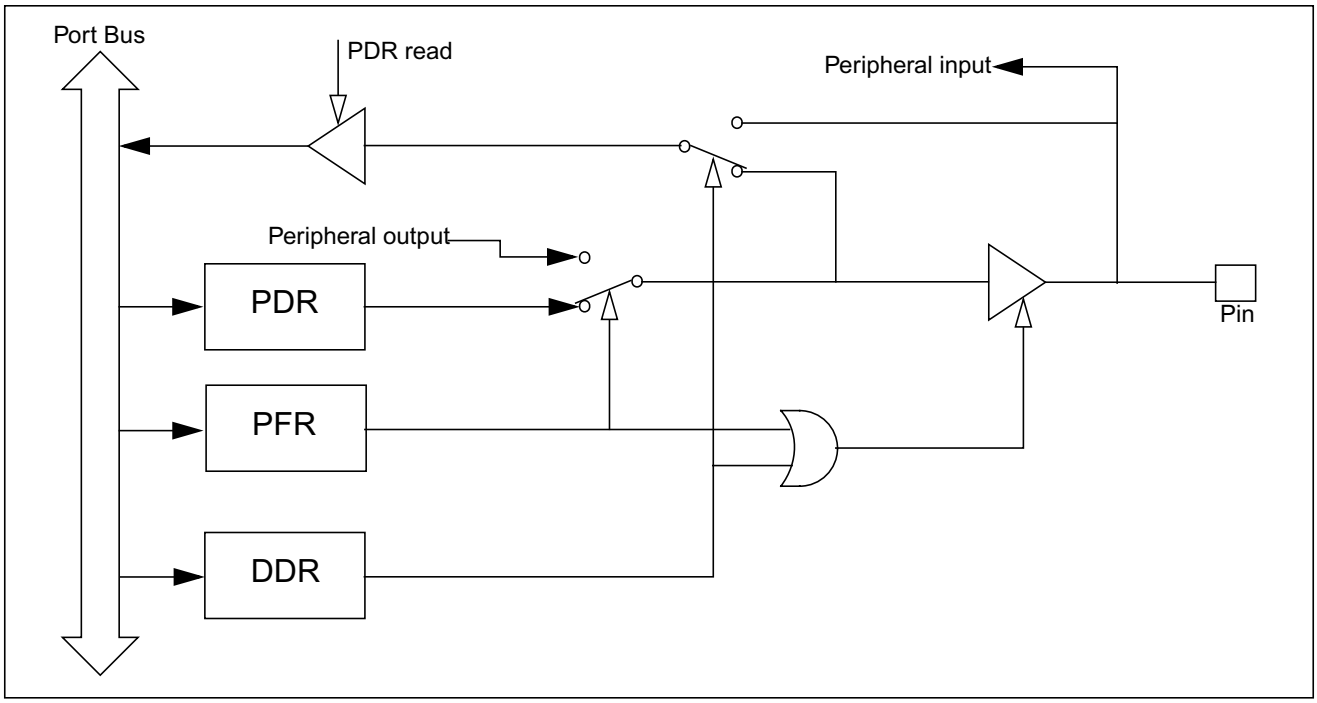
■ USART Pins

The USART pins also serve as general ports. Table 3-1 lists the pin functions, I/O formats, and settings required to use USART.

Table 3-1 USART04 Pins

Pin name	Pin function	I/O format	Pull-up Pull-down	Standby control	Setting required to use pin
SIN04	Port I/O or serial data input	CMOS output and CMOS hysteresis, CMOS Automotive hysteresis, TTL input	Programmable	Provided	Set port function mode: PFR: bit0 = 1, EPFR: bit0 = 0
SOT04	Port I/O or serial data output				Set to output enable mode: SMR04: SOE = 1, Set port function mode: PFR: bit1 = 1 EPFR: bit1 = 0
SCK04	Port I/O or serial clock input/output				Set port function mode: PFR: bit2 = 1 EPFR: bit2 = 0 Set to output enable mode when a clock is output SMR04: SCKE = 1

Figure 3-1 Block Diagram of USART pins



4. USART Registers

The following table defines the USART04 registers:

Table 4-1 USART04 Registers

Address	bit 15 bit 8	bit 7 bit 0
060_H, 061_H	SCR04 (Serial Control Register)	SMR04 (Serial Mode Register)
062_H, 063_H	SSR04 (Serial Status Register)	RDR04/TDR04 (Rx, Tx Data Register)
064_H, 065_H	ESCR04 (Extended Status/Control Reg.)	ECCR04 (Extended Comm. Contr. Reg.)
066_H, 067_H	FSR04 (FIFO status register)	FCR04 (FIFO control register)
088_H, 089_H	BGR104 (Baud Rate Generator Reg. 1)	BGR004 (Baud Rate Generator Reg. 0)

(Note) FSR (FIFO status register) and FCR (FIFO control register) register are only available on USARTs with FIFO option, i.e. USART ch. 4-7.

4.1 Serial Control Register 04 (SCR04)

This register specifies parity bits, selects the stop bit and data lengths, selects a frame data format in mode 1, clears the reception error flag, and specifies whether to enable transmission and reception.

Figure 4-1 Serial Control Register 04 (SCR04)

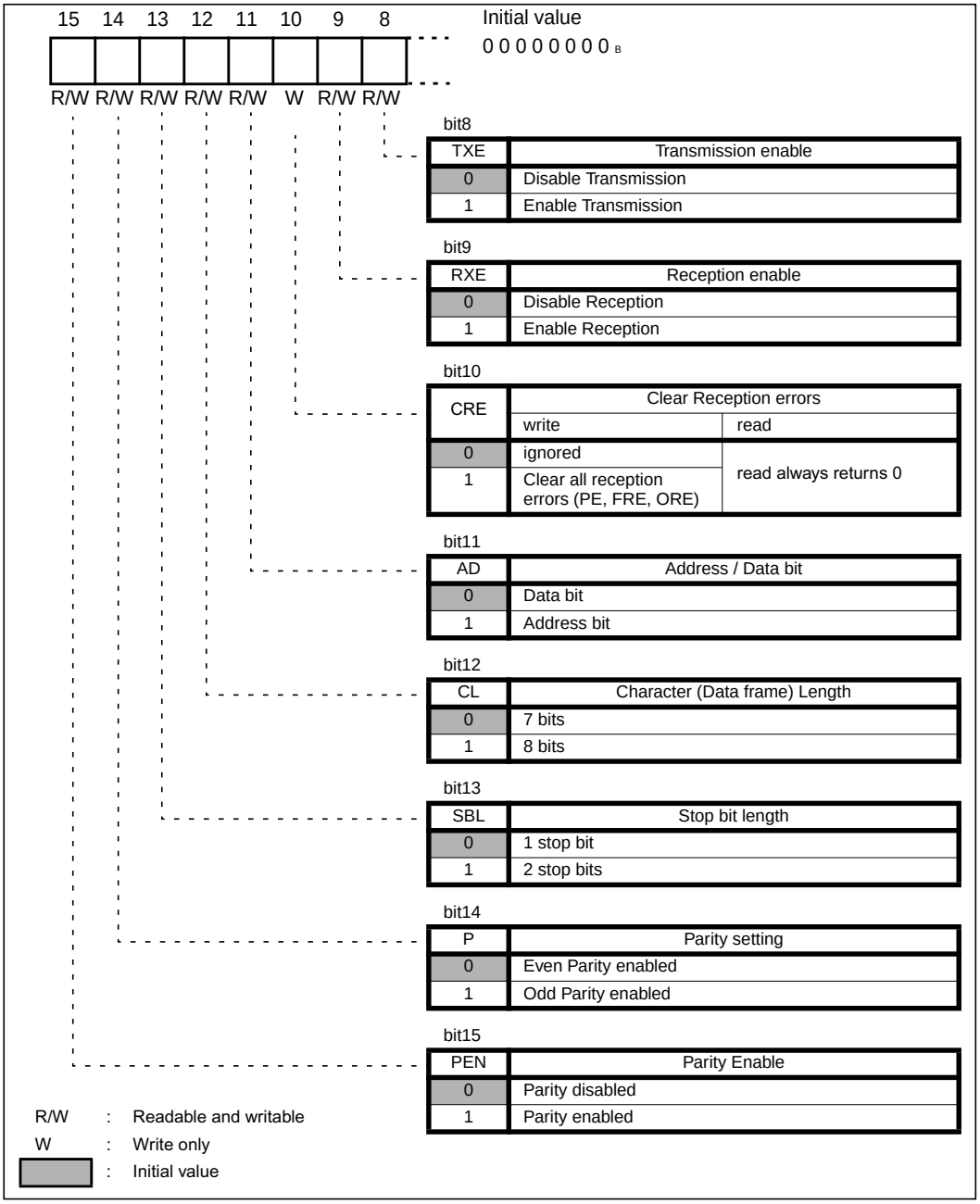


Table 4-2 Functions of each bit of control register 04 (SCR04)

Bit name		Function
bit15	PEN: Parity enable bit	This bit selects whether to add a parity bit during transmission in serial asynchronous mode or detect it during reception. Parity is only provided in mode 0 and in mode 2 if SSM of the ECCR04 is selected. This bit is fixed to 0 (no parity) in mode 3 (LIN).
bit14	P: Parity selection bit	When parity is provided and enabled this bit selects even (0) or odd (1) parity
bit13	SBL: Stop bit length selection bit	This bit selects the length of the stop bit of an asynchronous data frame or a synchronous frame if SSM of the ECCR04 is selected. This bit is fixed to 0 (1 stop bit) in mode 3 (LIN).
bit12	CL: Data length selection bit	This bit specifies the length of transmission or reception data. This bit is fixed to 1 (8 bits) in mode 2 and 3.
bit11	AD: Address/Data selection bit *	This bit specifies the data format in multiprocessor mode 1. Writing to this bit determines an address or data frame to be sent next, reading from it returns the last received kind of frame. “1” indicates an address frame, “0” indicates a usual data frame. **Note:** During a RMW-Read cycle the AD bit returns the value to be sent instead of the last received AD bit. see table below*
bit10	CRE: Clear reception error flags bit	This bit clears the FRE, ORE, and PE flag of the Serial Status Register (SSR04). This bit also clears a possible reception interrupt caused by errors. Writing a 1 to it clears the error flag. Writing a 0 has no effect. Reading from it always returns 0.
bit9	RXE: Reception enable bit	This bit enables USART reception. If this bit is set to 0, USART disables the reception of data frames. The LIN break detection in mode 0 or 3 remains unaffected.
bit8	TXE: Transmission enable bit	This bit enables USART transmission. If this bit is set to 0, USART disables the transmission of data frames.

* see table 4-3 for R/W options

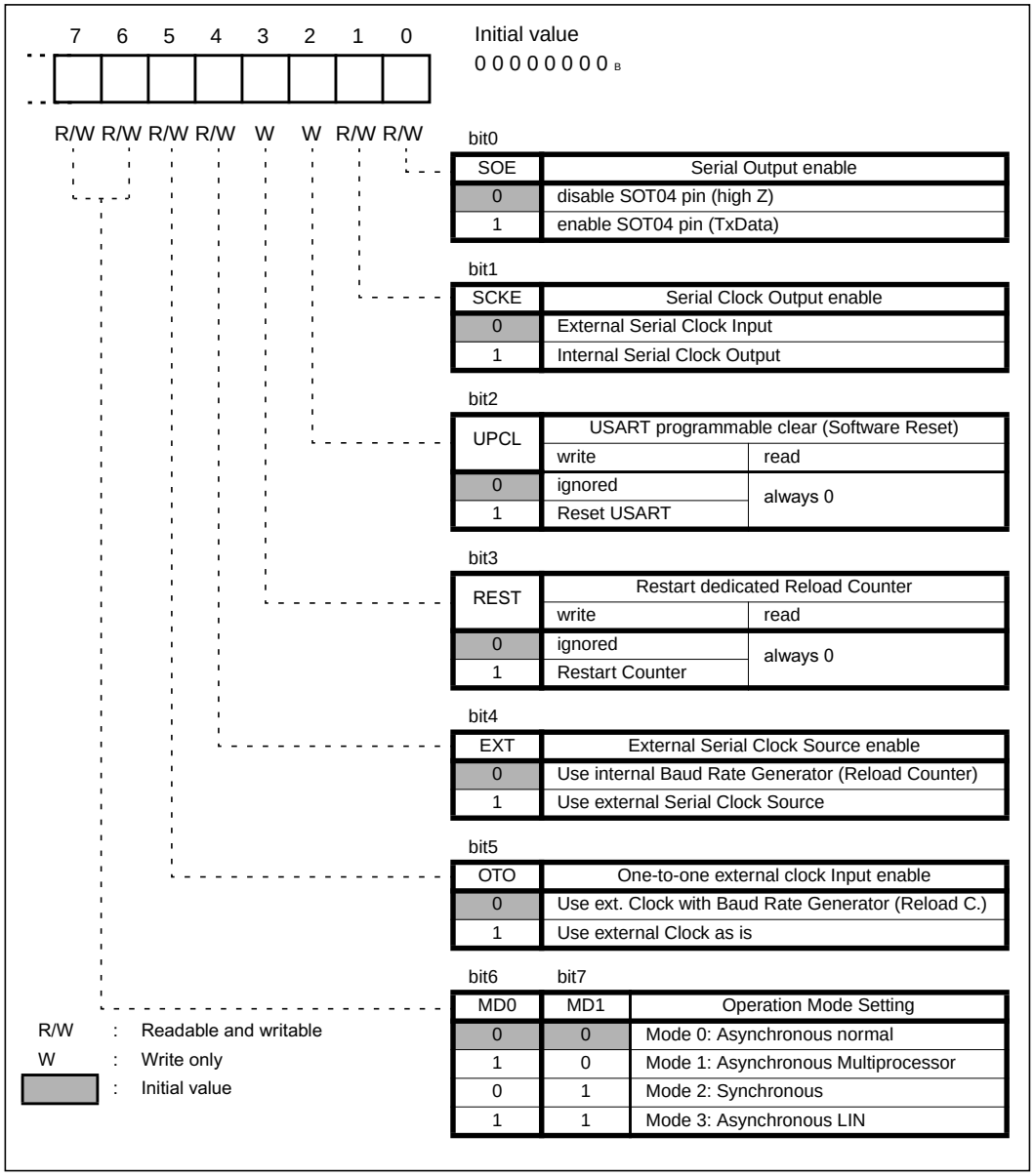
Table 4-3 * Read/Write options of AD-Bit

Cycle	Action
Write	Write data to be sent to AD-Bit
Normal Read	Read received AD-Bit
RMW-Read	Read data to be sent from AD-Bit

4.2 Serial Mode Register 04 (SMR04)

This register selects an operation mode and baud rate clock and specifies whether to enable output of serial data and clocks to the corresponding pin.

Figure 4-2 Configuration of the Serial Mode register 04 (SMR04)



bit0

SOE	Serial Output enable
0	disable SOT04 pin (high Z)
1	enable SOT04 pin (TxData)

bit1

SCKE	Serial Clock Output enable
0	External Serial Clock Input
1	Internal Serial Clock Output

bit2

UPCL	USART programmable clear (Software Reset)	
	write	read
0	ignored	always 0
1	Reset USART	

bit3

REST	Restart dedicated Reload Counter	
	write	read
0	ignored	always 0
1	Restart Counter	

bit4

EXT	External Serial Clock Source enable
0	Use internal Baud Rate Generator (Reload Counter)
1	Use external Serial Clock Source

bit5

OTO	One-to-one external clock Input enable
0	Use ext. Clock with Baud Rate Generator (Reload C.)
1	Use external Clock as is

bit6 bit7

MD0	MD1	Operation Mode Setting
0	0	Mode 0: Asynchronous normal
1	0	Mode 1: Asynchronous Multiprocessor
0	1	Mode 2: Synchronous
1	1	Mode 3: Asynchronous LIN

Table 4-4 Bit function of the Serial Mode register 04 (SMR04)

Bit name		Function
bit7 bit6	MD1 and MD0: Operation mode selection bits	These two bits sets the USART operation mode.
bit5	OTO: One-to-one external clock selection bit	This bit sets an external clock directly to the USART's serial clock. This function is used for synchronous slave mode operation
bit4	EXT: External clock selection bit	This bit executes internal or external clock source for the reload counter
bit3	REST: Restart of transmission reload counter bit	If a 1 is written to this bit the reload counter is restarted. Writing 0 to it has no effect. Reading from this bit always returns 0.
bit2	UPCL: USART programmable clear bit (Software reset)	Writing a 1 to this bit resets USART immediately. The register settings are preserved. Possible reception or transmission will cut off. All error flags are cleared and the Reception Data Register (RDR04) contains 00h. Writing 0 to this bit has no effect. Reading from it always returns 0.
bit1	SCKE: Serial clock output enable	• This bit controls the serial clock input-output ports. • When this bit is 0, the SCK04 pin operates as serial clock input pin. When this bit is 1, the SCK04 pin operates as serial clock output pin. <Caution> • When using the SCK04 pin as serial clock input (SCKE=0) pin, set the pin as input port. Also, select external clock (EXT = 1) using the external clock selection bit.
bit0	SOE: Serial data output enable bit	• This bit enables or disables the output of serial data. • When this bit is 0, the SOT04 pin outputs the default mark level. When this bit is 1, the SOT04 pin outputs the transmission data.

4.3 Serial Status Register 04 (SSR04)

This register checks the transmission statuts, reception status and error status, and enables and disables the transmission and reception interrupts.

Figure 4-3 Configuration of the Serial Status register 04 (SSR04)

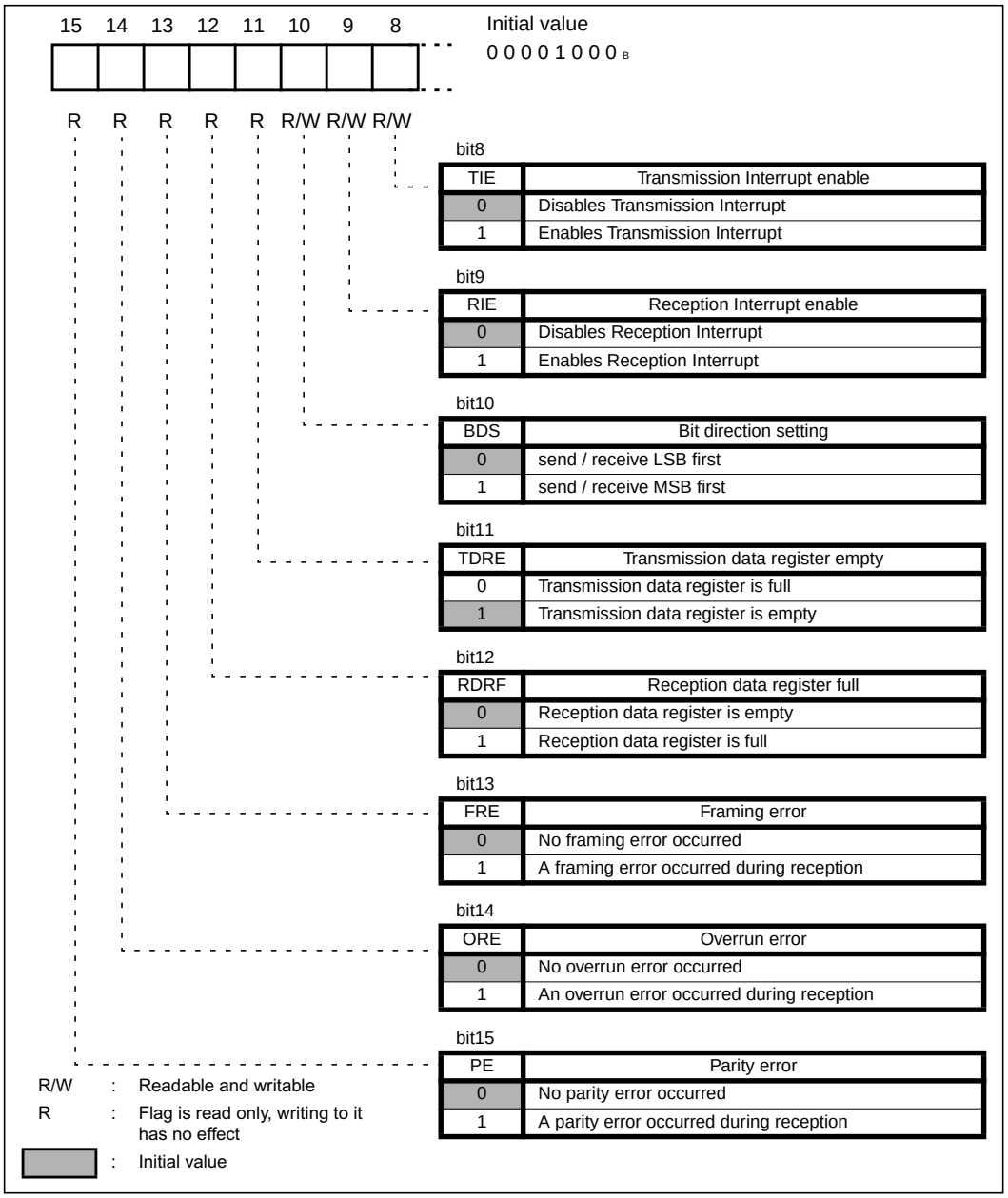


Table 4-5 Functions of each bit of status register 04 (SSR04)

Bit name		Function
bit15	PE: Parity error flag bit	• This bit is set to 1 when a parity error occurs during reception and is cleared when 0 is written to the CRE bit of the serial control register (SCR04). • A reception interrupt request is output when this bit and the RIE bit are 1. • Data in the reception data register (RDR04) is invalid when this flag is set.
bit14	ORE: Overrun error flag bit	• This bit is set to 1 when an overrun error occurs during reception and is cleared when 0 is written to the CRE bit of the serial control register (SCR04). • A reception interrupt request is output when this bit and the RIE bit are 1. • Data in the reception data register (RDR04) is invalid when this flag is set.
bit13	FRE: Framing error flag bit	• This bit is set to 1 when a framing error occurs during reception and is cleared when 0 is written to the CRE bit of the serial control register 1 (SCR04). • A reception interrupt request is output when this bit and the RIE bit are 1. • Data in the reception data register (RDR04) is invalid when this flag is set.
bit12	RDRF: Receive data full flag bit	• This flag indicates the status of the reception data register (RDR04). • This bit is set to 1 when reception data is loaded into RDR04 and can only be cleared to 0 when the reception data register (RDR04) is read. • A reception interrupt request is output when this bit and the RIE bit are 1.
bit11	TDRE: Transmission data empty flag bit	• This flag indicates the status of the transmission data register (TDR04). • This bit is cleared to 0 when transmission data is written to TDR04 and is set to 1 when data is loaded into the transmission shift register and transmission starts. • A transmission interrupt request is generated if this bit and the RIE bit are 1. <Caution> This bit is set to 1 (TDR04 empty) as its initial value.
bit10	BDS: Transfer direction selection bit	• This bit selects whether to transfer serial data from the least significant bit (LSB first, BDS=0) or the most significant bit (MSB first, BDS=1). <Caution> The high-order and low-order sides of serial data are interchanged with each other during reading from or writing to the serial data register. If this bit is set to another value after the data is written to the RDR04 register, the data becomes invalid. This bit is fixed to 0 in mode 3 (LIN)
bit9	RIE: Reception interrupt request enable bit	• This bit enables or disables input of a request for transmission interrupt to the CPU. • A reception interrupt request is output when this bit and the reception data flag bit (RDRF) are 1 or this bit and one or more error flag bits (PE, ORE, and FRE) are 1.

Table 4-5 Functions of each bit of status register 04 (SSR04)

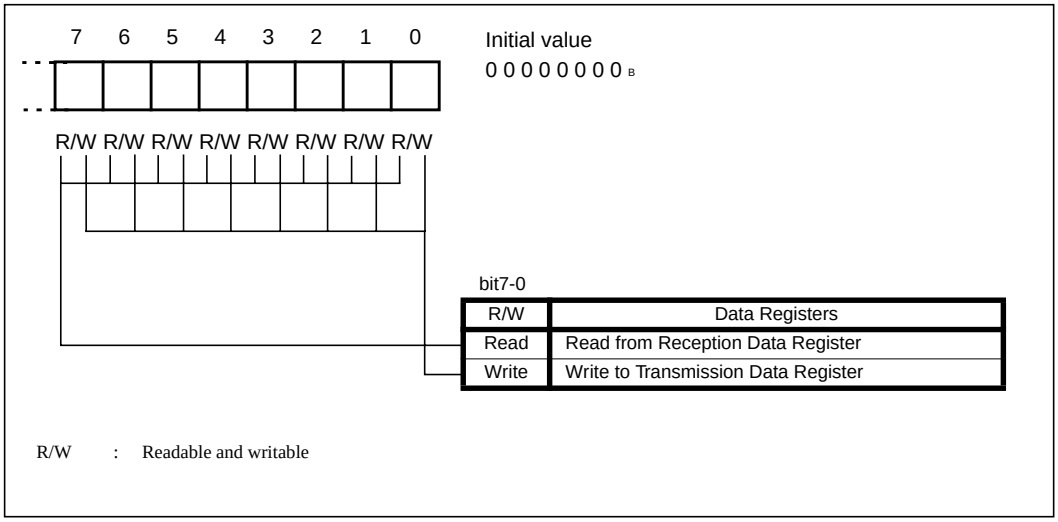
Bit name		Function
bit8	TIE: Transmission interrupt request enable bit	• This bit enables or disables output of a request for transmission interrupt to the CPU. • A transmission interrupt request is output when this bit and the TDRE bit are 1.

4.4 Reception and Transmission Data Register (RDR04 / TDR04)

The reception data register (RDR04) holds the received data. The transmission data register (TDR04) holds the transmission data. Both RDR04 and TDR04 registers are located at the same address.

(Note) TDR04 is a write-only register and RDR04 is a read-only register. These registers are located in the same address, so the read value is different from the write value. Therefore, instructions that perform a read-modify-write (RMW) operation, such as the INC/DEC instruction, cannot be used.

Figure 4-4 Transmission and Reception Data registers 04 (RDR04 / TDR04)



■ Reception:

RDR04 is the register that contains reception data. The serial data signal transmitted to the SIN04 pin is converted in the shift register and stored there. When the data length is 7 bits, the uppermost bit (D7) contains 0. When reception is complete the data is stored in this register and the reception data full flag bit (SSR04: RDRF) is set to 1. If a reception interrupt request is enabled at this point, a reception interrupt occurs.

Read RDR04 when the RDRF bit of the status register (SSR04) is 1. The RDRF bit is cleared automatically to 0 when RDR04 is read. Also the reception interrupt is cleared if it is enabled and no error has occurred.

Data in RDR04 is invalid when a reception error occurs (SSR04: PE, ORE, or FRE = 1).

■ Transmission:

When data to be transmitted is written to the transmission data register in transmission enable state, it is transferred to the transmission shift register, then converted to serial data, and transmitted from the serial data output terminal (SOT04 pin). If the data length is 7 bits, the uppermost bit (D7) is not sent.

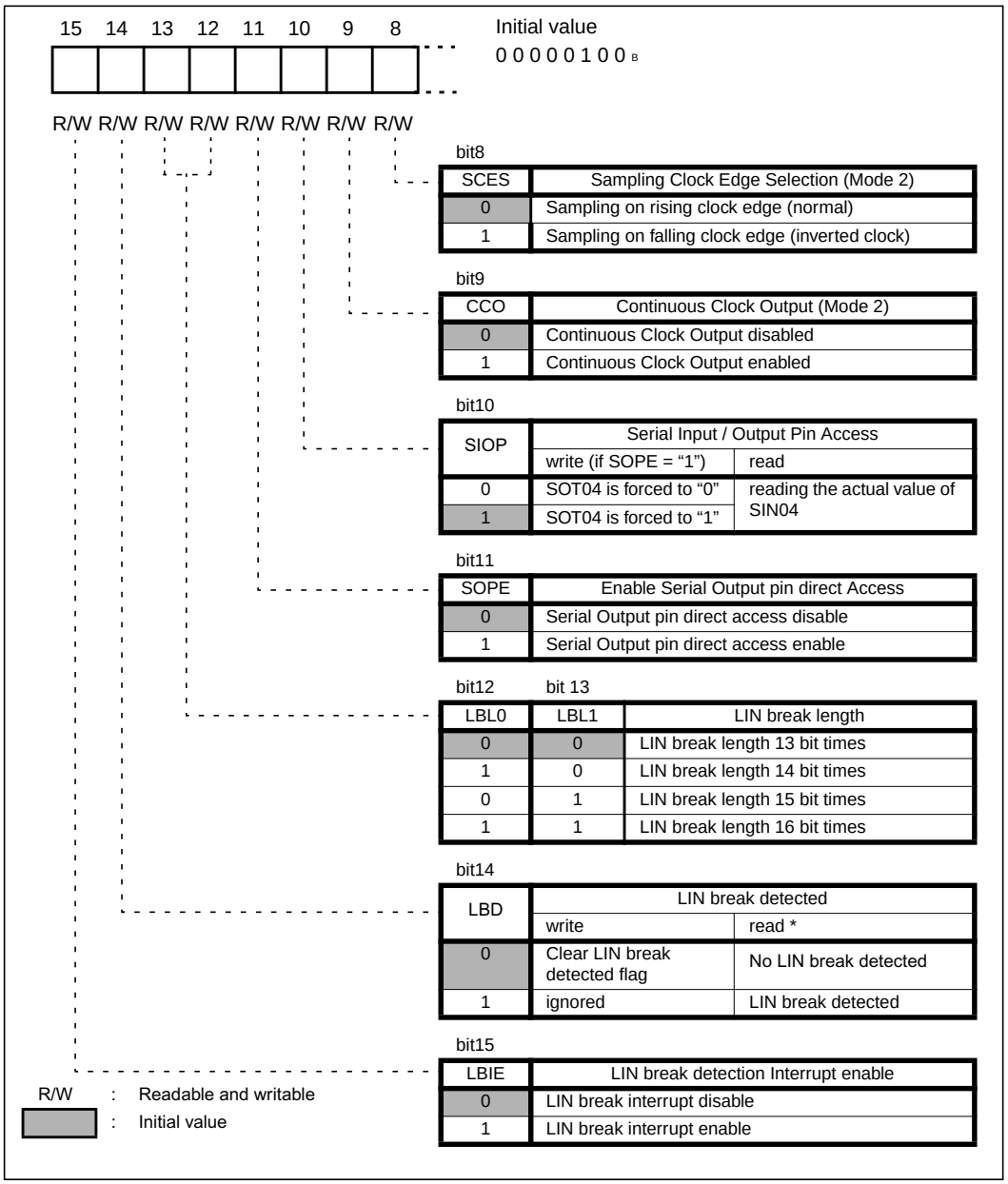
When transmission data is written to this register, the transmission data empty flag bit (SSR04: TDRE) is cleared to 0. When transfer to the transmission shift register is complete, the bit is set to 1. When the TDRE bit is 1, the next part of transmission data can be written. If output transmission interrupt requests have been

enabled, a transmission interrupt is generated. Write the next part of transmission data when a transmission interrupt is generated or the TDRE bit is 1.

4.5 Extended Status/Control Register (ESCR04)

This register provides several LIN functions, direct access to the SIN04 and SOT04 pin and setting for USART synchronous clock mode.

Figure 4-5 Configuration of the Extended Status/Control Register (ESCR04)



bit8

SCES	Sampling Clock Edge Selection (Mode 2)
0	Sampling on rising clock edge (normal)
1	Sampling on falling clock edge (inverted clock)

bit9

CCO	Continuous Clock Output (Mode 2)
0	Continuous Clock Output disabled
1	Continuous Clock Output enabled

bit10

SIOP	Serial Input / Output Pin Access	
	write (if SOPE = “1”)	read
0	SOT04 is forced to “0”	reading the actual value of SIN04
1	SOT04 is forced to “1”	

bit11

SOPE	Enable Serial Output pin direct Access
0	Serial Output pin direct access disable
1	Serial Output pin direct access enable

bit12 bit 13

LBL0	LBL1	LIN break length
0	0	LIN break length 13 bit times
1	0	LIN break length 14 bit times
0	1	LIN break length 15 bit times
1	1	LIN break length 16 bit times

bit14

LBD	LIN break detected	
	write	read *
0	Clear LIN break detected flag	No LIN break detected
1	ignored	LIN break detected

bit15

LBIE	LIN break detection Interrupt enable
0	LIN break interrupt disable
1	LIN break interrupt enable

* see table 4-6 for RMW access!

Table 4-6 Function of each bit of the Extended Status/Control Register (ESCR4)

Bit name		Function
bit15	LBIE: LIN break detection interrupt enable bit	This bit enables a reception interrupt, if a LIN break was detected.
bit14	LBD: LIN break detected flag	This bit goes 1 if a LIN break was detected. Writing a 0 to it clears this bit and the corresponding interrupt, if it is enabled. **Note:** RMW instructions always return "1". In this case, the value "1" does not indicate a LIN-Break.
bit13 bit12	LBL1/0: LIN break length selection	These two bits determine how many serial bit times the LIN break is generated by USART. Receiving a LIN break is always fixed to 13 bit times.
bit11	SOPE: Serial Output pin direct access enable*	Setting this bit to 1 enables the direct write to the SOT04 pin, if SOE = 1 (SMR04).*
bit10	SIOP: Serial Input/ Output Pin direct access*	Normal read instructions always return the actual value of the SIN04 pin. Writing to it sets the bit value to the SOT04 pin, if SOPE = 1. During a Read-Modify-Write instruction the bit returns the SOT04 value in the read cycle.*
bit9	CCO: Continuos Clock Output enable bit	This bit enables a continuos serial clock at the SCK04 pin if USART operates in master mode 2 (synchronous) and the SCK04 pin is configured as a clock output.
bit8	SCES: Serial clock edge selection bit	This bit inverts the internal serial clock in mode 2 (synchronous) and the output clock signal, if USART operates in master mode 2 (synchronous) and the SCK04 pin is configured as a clock output. In slave mode 2 the sampling time turns from rising edge to falling edge.

* see table 4-7 for SOPE and SIOP interaction

Table 4-7 * Description of the interaction of SOPE and SIOP:

SOPE	SIOP	Writing to SIOP	Reading from SIOP
0	R/W	has no effect on the SOT4 pin but holds the written value.	returns current value of SIN04
1	R/W	write "0" or "1" to SOT04	returns current value of SIN04
1	RMW		returns current value of SOT04 and writes it back

4.6 Extended Communication Control Register (ECCR04)

The extended communication control register provides bus idle recognition, interrupt settings, synchronous clock settings, and the LIN break generation.

Figure 4-6 Configuration of the Extended Communication Control Register (ECCR04)

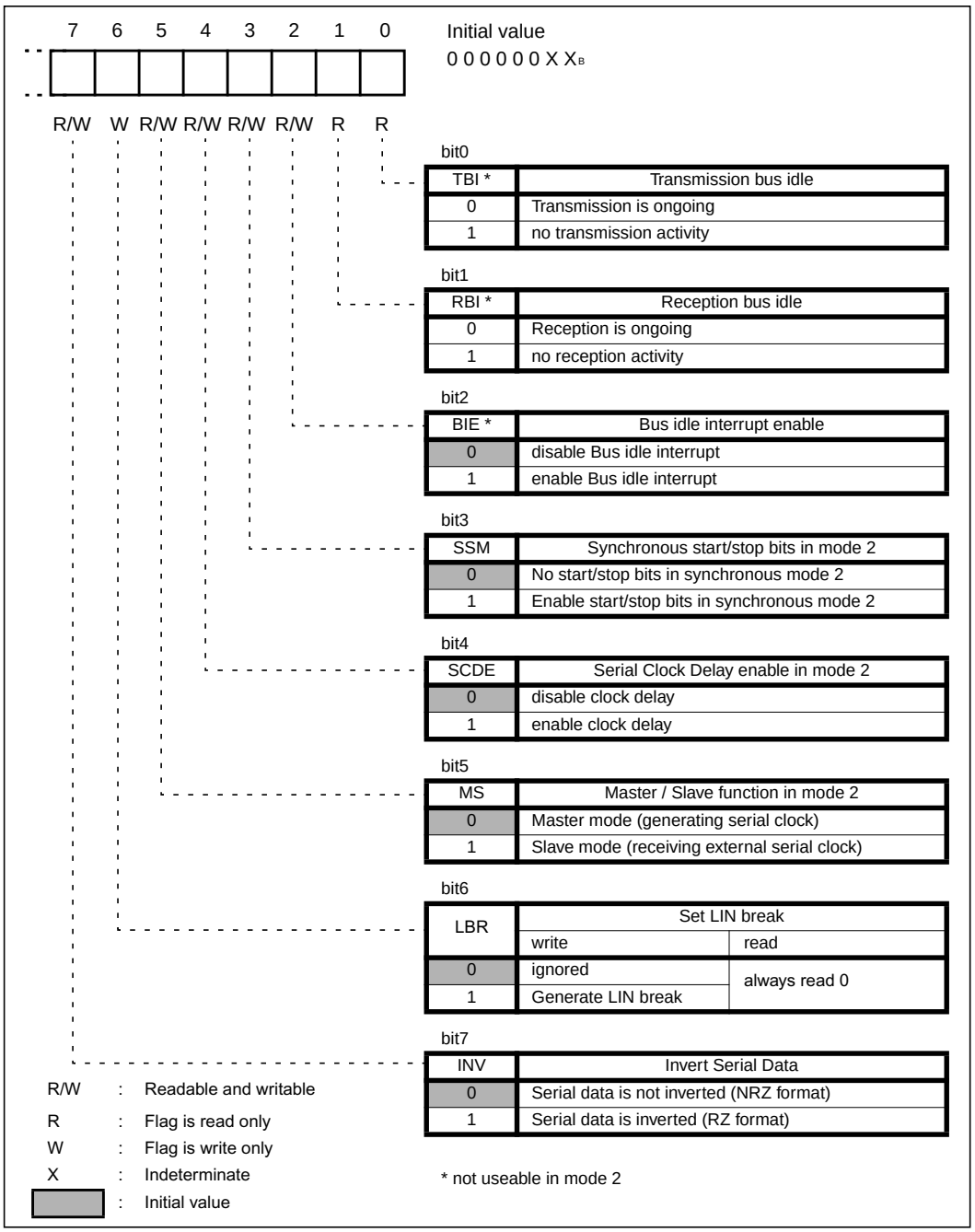


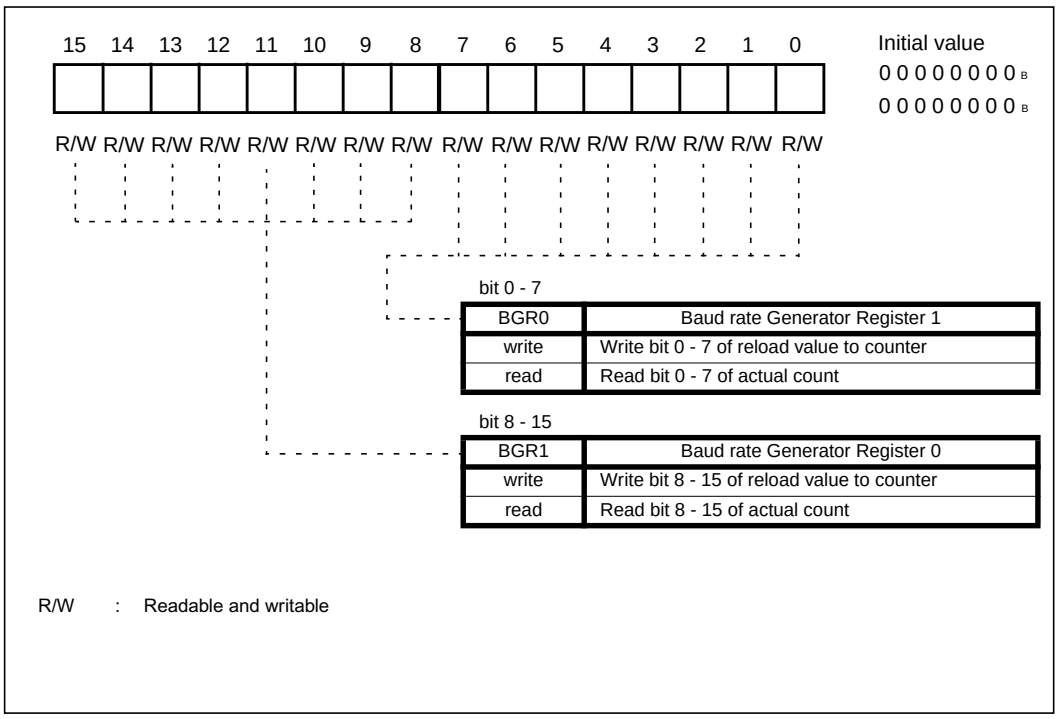
Table 4-8 Function of each bit of the Extended Communication Control Register (ECCR04)

Bit name		Function
bit7	INV: Invert serial data	This bit inverts the serial data at SIN04 and SOT04 pin. SCK04 is not affected (see ESCR04: SCES). Writing "0": The serial data format is NRZ (default) Writing "1": The serial data is inverted (RZ format) RMW instructions do not affect this bit.
bit6	LBR: Set LIN break bit	Writing a 1 to this bit generates a LIN break of the length selected by the LBL0/1 bits of the ESCR04, if operation mode 0 or 3 is selected.
bit5	MS: Master/Slave mode selection bit	This bit selects master or slave mode of USART in synchronous mode 2. If master is selected USART generates the synchronous clock by itself. If slave mode is selected USART receives external serial clock. <Caution> If slave mode is selected, the clock source must be external and set to “One-to-One” (SMR04: SCKE = 0, EXT = 1, OTO = 1).
bit4	SCDE: Serial clock delay enable bit	If this bit is set, the serial output clock is delayed by 1 CLKP cycle (or half of its period in SPI-compliance). This only applies, if USART operates in master mode 2.
bit3	SSM: Start/Stop bit mode enable	This bit adds start and stop bits to the synchronous data format in operation mode 2. It is ignored in mode 0, 1, and 3.
bit2	BIE: Bus idle interrupt enable	This bit enables a reception interrupt, if there is neither reception nor transmission ongoing (RBI = 1, TBI = 1). **Note:** Do not use BIE in mode 2.
bit1	RBI: Reception bus idle flag bit	This bit is “1” if there is no reception activity on the SIN04 pin. **Note:** Do not use this flag in mode 2.
bit0	TBI: Transmission bus idle flag bit	This bit is “1” if there is no transmission activity on the SOT04 pin. **Note:** Do not use this flag in mode 2.

4.7 Baud Rate / Reload Counter Register 0 and 1 (BGR04 / 14)

The baud rate / reload counter registers set the division ratio for the serial clock. Also the actual count of the transmission reload counter can be read.

Figure 4-7 Baudrate Reload Counter Register 0 and 1 (BGR04 / 14)



The Baud Rate / Reload Counter Registers determine the division ratio for the serial clock.
Both registers can be read or written via byte or word access.

4.8 FIFO Control Register (FCR04)

Figure 4-8 Configuration of FIFO control register

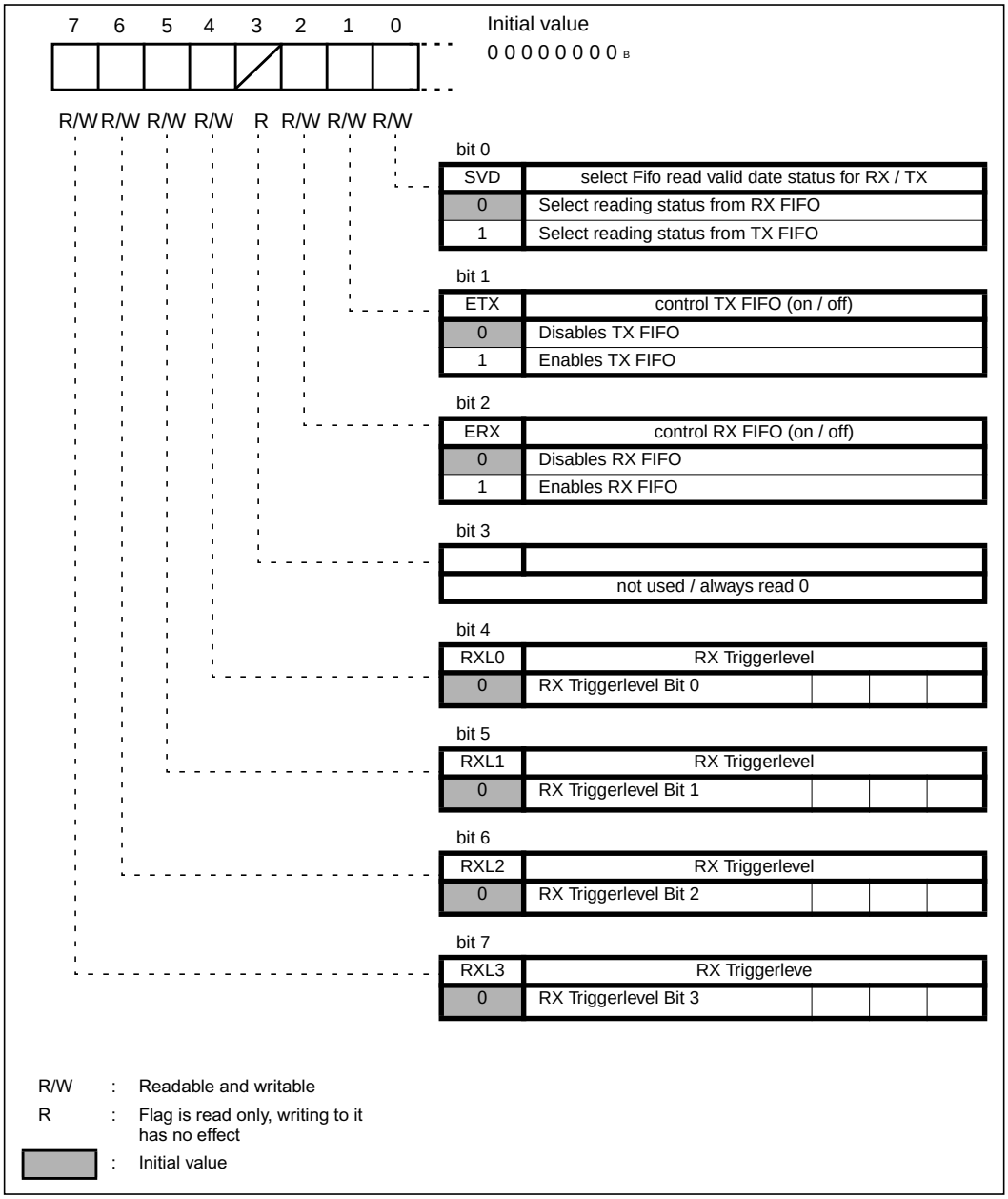


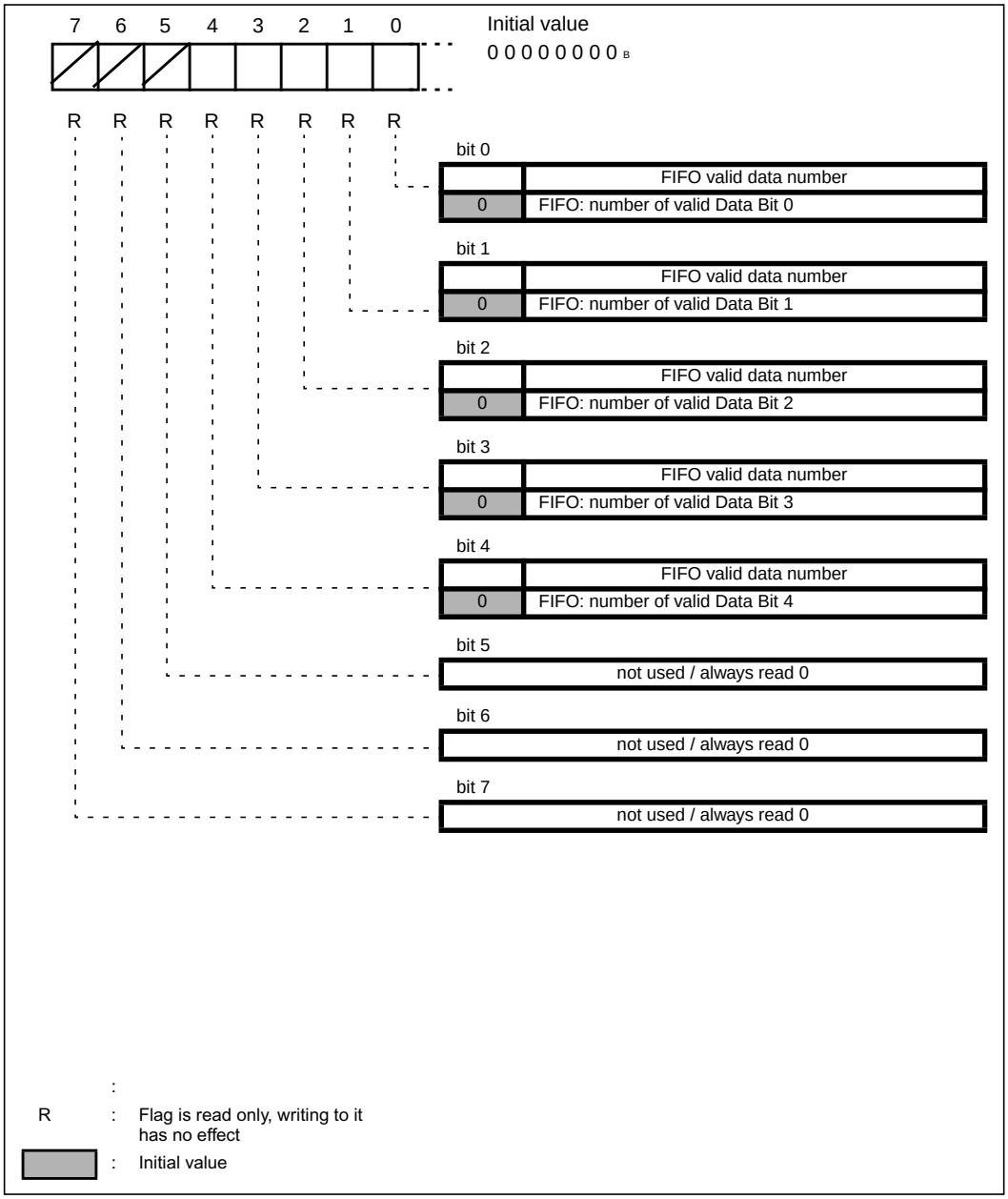
Table 4-9 Functions of each bit of fifo control register (FCR4)

Bit name		Function
bit 0	SVD: select Valid Data Fifo read	• If this bit is set to 0 the fifo status register shows the number of valid data from the RX fifo • If this bit is set to 1 the fifo status register shows the number of valid data from the TX fifo
bit 1	ETX: enable TX fifo	• If this bit is set to 0 the TX fifo is disabled / fifo data is cleared • If this bit is set to 1 the TX fifo is enabled.
bit 2	ERX: enable RX fifo	• If this bit is set to 0 the RX fifo is disabled / fifo data is cleared • If this bit is set to 1 the RX fifo is enabled.
bit 3		reserved
bit 4	RXL0: RX Triggerlevel bit 0	• Set the triggerlevel of RX Interrupt
bit 5	RXL1: RX Triggerlevel bit 1	• Set the triggerlevel of RX Interrupt
bit 6	RXL2: RX Triggerlevel bit 2	• Set the triggerlevel of RX Interrupt
bit 7	RXL3: RX Triggerlevel bit 3	• Set the triggerlevel of RX Interrupt

(Note) The RX triggerlevel sets the reception FIFO level where the reception interrupt is activated. E.g. if the triggerlevel is at its default value of RXL[3:0]=0000, the interrupt is activated if one reception is stored in the FIFO. If the the triggerlevel is set to RXL[3:0]=1111, the interrupt is activated if 16 receptions are stored in the FIFO. In general: a reception interrupt is triggered if FSR[4:0] > FCR[7:4].

4.9 FIFO Status Register (FSR04)

Figure 4-9 Configuration of FIFO status register



(Note) The FSR04[4:0] FIFO valid data bits indicates the number of stored receptions (SVD=0) or pending transmissions (SVD=1) in the FIFO buffer.

Table 4-10 Functions of each bit of FIFO status Register

Bit name		Function
bit 0	FIFO: number of valid Data	• shows the number of valid FIFO - Data for RX and TX Fifo, depending on selection bit.
bit 1	FIFO: number of valid Data	• shows the number of valid FIFO - Data for RX and TX Fifo, depending on selection bit.
bit 2	FIFO: number of valid Data	• shows the number of valid FIFO - Data for RX and TX Fifo, depending on selection bit..
bit 3	FIFO: number of valid Data	• shows the number of valid FIFO - Data for RX and TX Fifo, depending on selection bit..
bit 4	FIFO: number of valid Data	• shows the number of valid FIFO - Data for RX and TX Fifo, depending on selection bit.

5. USART Interrupts

The USART uses both reception and transmission interrupts. An interrupt request can be generated for either of the following causes:

- Receive data is set in the Reception Data Register (RDR04), or a reception error occurs.
- Transmission data is transferred from the Transmission Data Register (TDR04) to the transmission shift register.
- A LIN break is detected
- No bus activity (neither reception nor transmission)

■ USART Interrupts

Table 5-1 Interrupt control bits and interrupt causes of USART

Reception/ transmission/ ICU	Interrupt request flag bit	Flag Register	Operation mode				Interrupt cause	Interrupt cause enable bit	How to clear the Interrupt Request
			0	1	2	3			
Reception	RDRF	SSR04	x	x	x	x	receive data is written to RDR (FIFO level reached)	SSR04 : RIE	Receive data is read
	ORE	SSR04	x	x	x	x	Overrun error		"1" is written to clear rec. error bit (SCR04: CRE)
	FRE	SSR04	x	x	*	x	Framing error		
	PE	SSR04	x		*		Parity error		
	LBD	ESCR04	x			x	LIN synch break detected	ESCR04 : LBIE	"0" is written to ESCR04 : LBD
	TBI & RBI	ESCR04	x	x		x	no bus activity	ECCR04 : BIE	Receive data / Send data
Transmission	TDRE	SSR04	x	x	x	x	Empty transmission register	SSR04 : TIE	Transfer data is written
Input Capture Unit	ICP4	IPCP4	x			x	1st falling edge of LIN synch field	IPCP4 : ICE4	disable ICE4 temporary
	ICP4	IPCP4	x			x	5th falling edge of LIN synch field	IPCP4 : ICE4	disable ICE4

x : Used
* : Only available if ECCR04/SSM = 1

■ Reception Interrupt

If one of the following events occur in reception mode, the corresponding flag bit of the Serial Status Register (SSR04) is set to "1":

- - Data reception is complete, i. e. the received data was transferred from the serial input shift register to the Reception Data Register (RDR04) and data can be read: **RDRF** (if FIFO is enabled, trigger level is reached)
- - Overrun error, i. e. RDRF = 1 and RDR04 was not read by the CPU: **ORE**

- - Framing error, i. e. a stop bit was expected, but a "0"-bit was received: **FRE**
- - Parity error, i. e. a wrong parity bit was detected: **PE**

If at least one of these flag bits above go "1" and the reception interrupt is enabled (SSR04: RIE = 1), a reception interrupt request is generated.

If the Reception Data Register (RDR04) is read, the RDRF flag is automatically cleared to "0". Note that this is the *only* way to reset the RDRF flag. The error flags are cleared to "0", if a "1" is written to the Clear Reception Error (CRE) flag bit of the Serial Control Register (SCR04). The RDR04 contains only valid data if the RDRF flag is "1" and no error bits are set.

Note, that the CRE flag is "write only" and by writing a "1" to it, it is internally held to "1" for one CPU clock cycle.

■ Transmission Interrupt

If transmission data is transferred from the Transmission Data Register (TDR04) to the transfer shift register (this happens, if the shift register (or FIFO) is empty and transmission data exists), the Transmission Data Register Empty flag bit (TDRE) of the Serial Status Register (SSR04) is set to "1". In this case an interrupt request is generated, if the Transmission Interrupt Enable (TIE) bit of the SSR04 was set to "1" before.

Note, that the initial value of TDRE (after hardware or software reset) is "1". So an interrupt is generated immediately then, if the TIE flag is set to "1". Also note, that the *only* way to reset the TDRE flag is writing data to the Transmission Data Register (TDR04).

■ LIN Synchronization Break Interrupt

This paragraph is only relevant, if USART operates in mode 0 or 3 as a LIN slave.

If the bus (serial input) goes "0" (dominant) for more than 11 bit times, the LIN Break Detected (LBD) flag bit of the Extended Status/Control Register (ESCR04) is set to "1". Note, that in this case after 9 bit times the reception error flags are set to "1", therefore the RIE flag has to be set to "0" or the RXE flag has to be set to "0", if only a LIN synch break detect is desired. In the other case a reception error interrupt would be generated first, and the interrupt handler routine has then to wait for LBD = 1.

The interrupt and the LBD flag are cleared after writing a "1" to the LBD flag. This makes sure, that the CPU has detected the LIN synch break, because of the following procedure of adjusting the serial clock to the LIN master.

■ LIN Synchronization Field Edge Detection Interrupts

This paragraph is only relevant, if USART operates in mode 0 or 3 as a LIN slave. After a LIN break detection the next falling edge of the reception bus is indicated by USART. Simultaneously an internal signal connected to the ICU is set to "1". This signal is reset to "0" after the fifth falling edge of the LIN Synchronization Field. In both cases the ICU4 generates an interrupt, if "both edge detection" and the ICU1/5 interrupt are enabled. The difference of the ICU4 counter values is the serial clock multiplied by 8. Dividing it by 8 results in the new detected and calculated baud rate for the dedicated reload counter. This value - 1 has then to be written to the Baud Rate Generator Registers (BGR1/0).There is no need to restart the reload counter, because it is automatically reset if a falling edge of a start bit is detected.

■ Bus Idle Interrupt

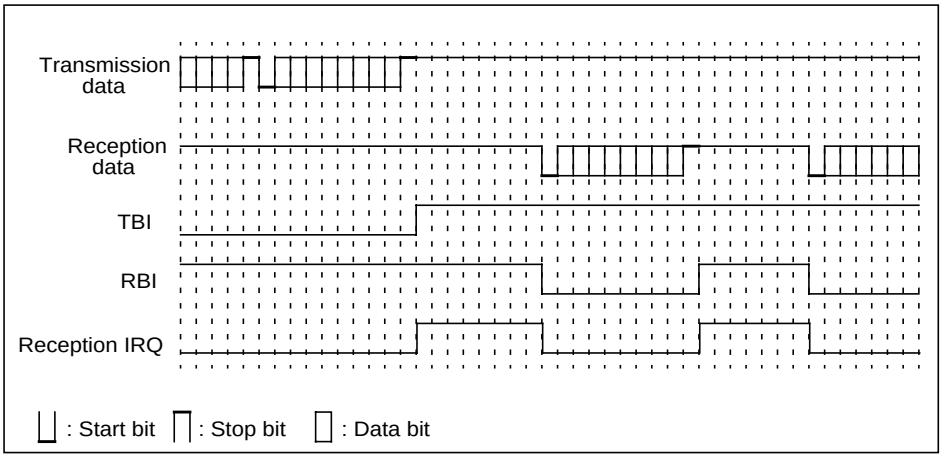
If there is no reception activity on the SIN04 pin, the RBI flag bit of the ECCR04 goes "1". The TBI flag bit respectively goes "1", when no data is transmitted. If the Bus Idle Interrupt Enable bit (BIE) of the ECCR04 is set and **both** bus idle flag bits (TBI **and** RBI) are "1", an interrupt is generated.

(Note) The TBI flag goes also "0" if there is no bus activity, but a "0" is written to the SIOP bit, if SOPE is "1".

(Note) TBI nd RBI cannot be used in mode 2 (synchronous communication).

Figure 5-1 illustrates how the bus idle interrupt is generated

Figure 5-1 Bus idle interrupt generation



5.1 Reception Interrupt Generation and Flag Set Timing

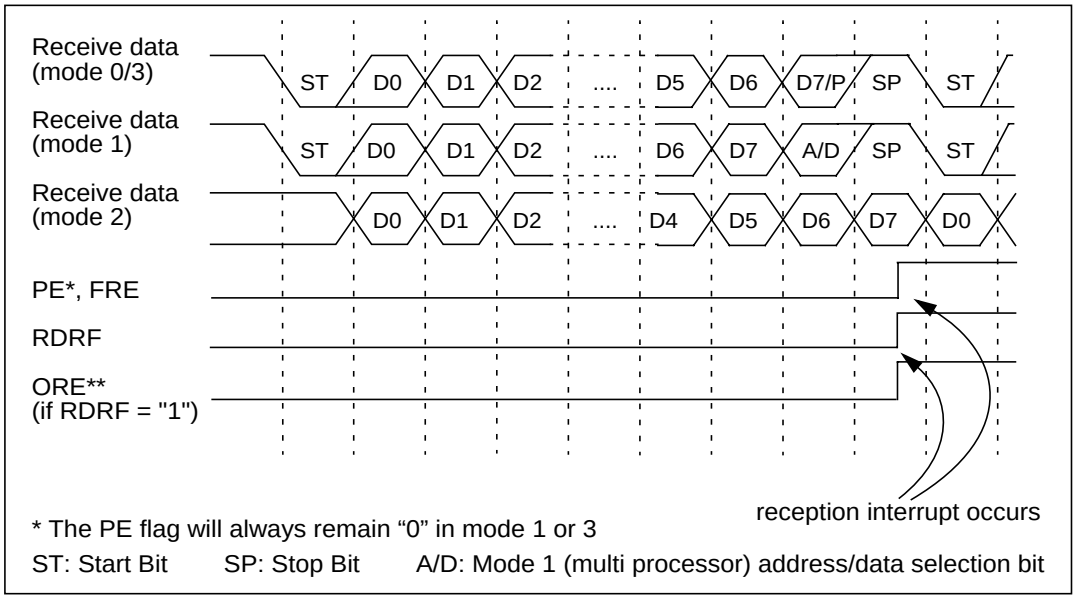
The following are the reception interrupt causes: Completion of reception (SSR04: RDRF) and occurrence of a reception error (SSR04: PE, ORE, or FRE).

■ Reception Interrupt Generation and Flag Set Timing

Generally a reception interrupt is generated, if the received data is complete (RDRF = 1) and the Reception Interrupt Enable (RIE) flag bit of the Serial Status Register (SSR04) was set to "1". This interrupt is generated if the first stop bit is detected in mode 0, 1, 2 (if SSM = 1), 3, or the last data bit was read in mode 2 (if SSM = 0).

(Note) If a reception error has occurred, the Reception Data Register (RDR04) contains invalid data in each mode.

Figure 5-2 Reception operation and flag set timing

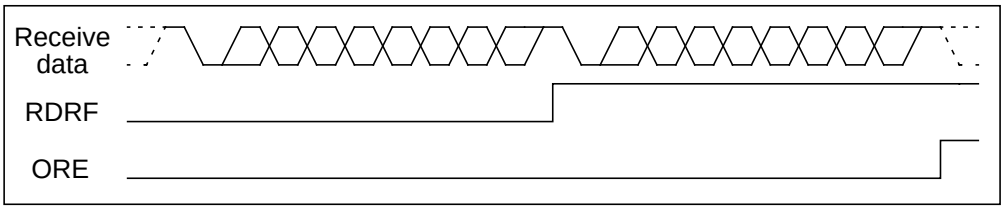


(Note) The example in figure 5-2 does not show all possible reception options for mode 0 and 3. Here it is:

"7p1" and "8N1" (p = "E" [even] or "O" [odd]), all in NRZ data format (ECCR04: INV = 0).

(Note) **ORE only occurs, if the reception data is not read by the CPU (RDRF = 1) and another data frame is read.

Figure 5-3 ORE set timing



5.2 Transmission Interrupt Generation and Flag Set Timing

A transmission interrupt is generated when the next data to be sent is ready to be written to the output data register (TDR04).

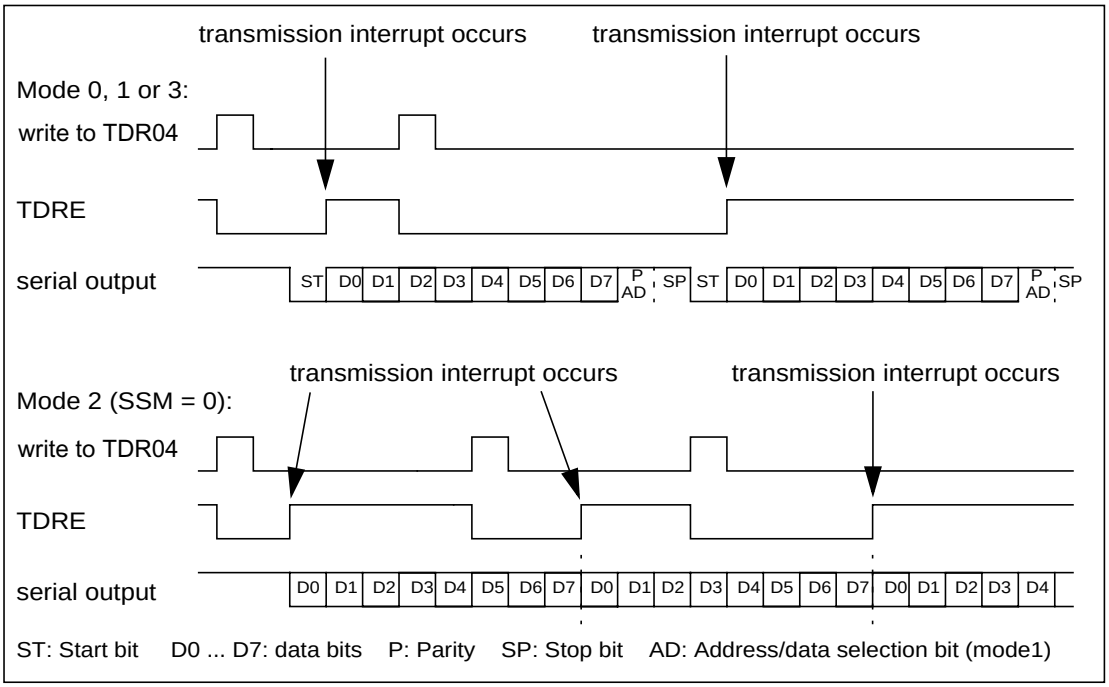
■ Transmission Interrupt Generation and Flag Set Timing

A transmission interrupt is generated, when the next data to be send is ready to be written to the Transmission Data Register (TDR04), i. e. the TDR04 is empty, and the transmission interrupt is enabled by setting the Transmission Interrupt Enable (TIE) bit of the Serial Status Register (SSR04) to "1".

The Transmission Data Register Empty (TDRE) flag bit of the SSR04 indicates an empty TDR04. Because the TDRE bit is "read only", it only can be cleared by writing data into TDR04.

The following figure demonstrates the transmission operation and flag set timing for the four modes of USART.

Figure 5-4 Transmission operation and flag set timing



(Note) The example in figure 5-4 does not show all possible transmission options for mode 0. Here it is: "8p1" (p = "E" [even] or "O" [odd]), ECCR04: INV = 0. Parity is not provided in mode 3 or 2, if SSM = 0.

■ Transmission Interrupt Request Generation Timing

If the TDRE flag is set to 1 when a transmission interrupt is enabled (SSR04: TIE=1) a transmission interrupt request is generated.

A transmission completion interrupt is generated immediately after the transmission interrupt is enabled (TIE=1) because the TDRE bit is set to 1 as its initial value. TDRE is a read-only bit that can be cleared only by writing new data to the output data register (TDR04). Carefully specify the transmission interrupt enable timing.

6. USART Baud Rates

One of the following can be selected for the USART serial clock source:

- Dedicated baud rate generator (Reload Counter)
- External clock as it is (clock input to the SCK04 pin)
- External clock connected to the baud rate generator (Reload Counter)

■ USART Baud Rate Selection

The baud rate selection circuit is designed as shown below. One of the following three types of baud rates can be selected:

- Baud Rates Determined Using the Dedicated Baud Rate Generator (Reload Counter)

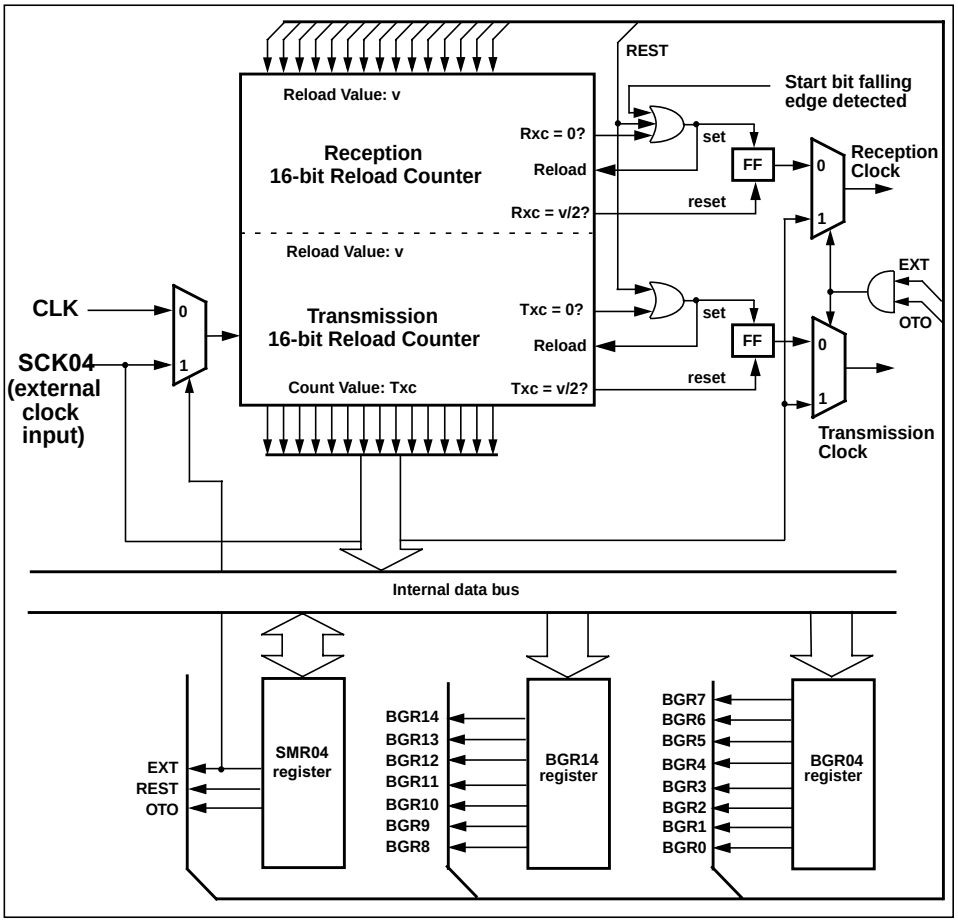
  USART has two independent internal reload counters for transmission and reception serial clock. The baud rate can be selected via the 15-bit reload value determined by the Baud Rate Generator Register 0 and 1 (BGR0/1).

  The reload counter divides the peripheral clock by the value set in the Baud Rate Generator Register 0 and 1.
- Baud Rates determined using external clock (one-to-one mode)

  The clock input from USART clock pulse input pins (SCK04) is used as it is (synchronous). Any baud rate less than the peripheral clock divided by 4 and is divisible can be set externally
- Baud Rates determined using the dedicated baud rate generator with external clock

  An external clock source can also be connected internally to the reload counter. In this mode it is used instead of the internal peripheral clock. This was designed to use quartz oscillators with special frequencies and having the possibility to divide them.

Figure 6-1 Baud rate selection circuit (reload counter)



6.1 Setting the Baud Rate

This section describes how the baud rates are set and the resulting serial clock frequency is calculated.

■ Calculating the baud rate

The both 15-bit Reload Counters are programmed by the Baud Rate Generator Registers 1 and 0 (BGR14, 04). The following calculation formula should be used to set the wanted baud rate:

Reload Value:

$$v = [F / b] - 1 \quad ,$$

where F is the resource clock (CLKP), *b* the baud rate and [ ] gaussian brackets (mathematical rounding function).

■ Example of Calculation

If the CPU clock is 16 MHz and the desired baud rate is 19200 baud then the reload value *v* is:

$v = [16 * 10^6 / 19200] - 1 =$ **832**

The exact baud rate can then be recalculated: $b_{exact} = F / (v + 1)$, here it is: $16 * 10^6 / 833 = 19207.6831$

(Note) Setting the reload value to 0 stops the reload counter.

(Note) The minimum recommended division ratio is 4 (i.e. reload value is 3) due to RX oversampling filter in asynchronous communication modes (mode 0,1 and 3).

■ Suggested Division Ratios for different machine speeds and baud rates

The following settings are suggested for different MCU clock speeds and baud rates:

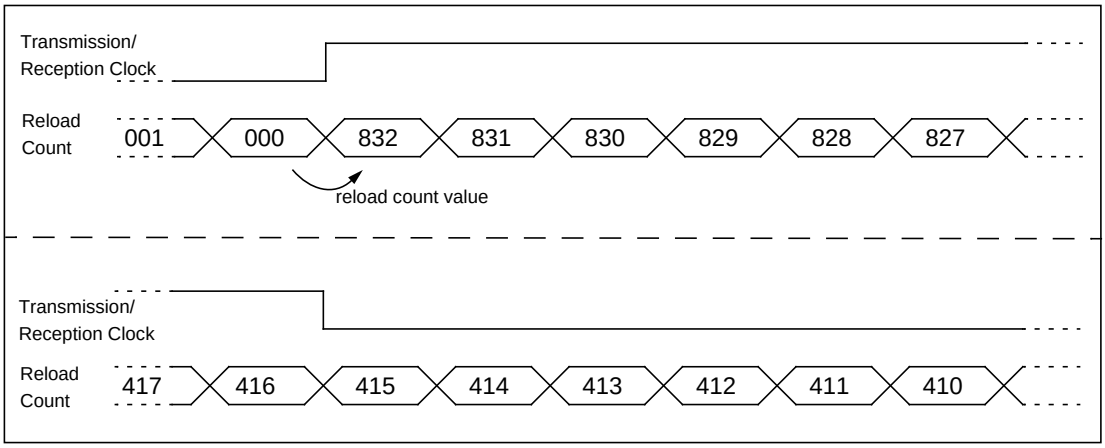
Table 6-1 Suggested Baud Rates and reload values at different machine speeds.

Baud rate	8 MHz		10 MHz		16 MHz		20 MHz		24 MHz		32 MHz	
	value	% dev.	value	% dev.	value	% dev.	value	% dev.	value	% dev.	value	% dev.
2M	3	0	4	0	7	0	9	0	11	0	15	0
1M	7	0	9	0	15	0	19	0	23	0	31	0
500000	15	0	19	0	31	0	39	0	47	0	63	0
460800	-	-	-	-	-	-	-	-	51	-0.16	-	-
250000	31	0	39	0	63	0	79	0	95	0	127	0
230400	-	-	-	-	-	-	86	0.22	103	-0.16	138	-0.08
153600	51	-0.16	64	-0.16	103	-0.16	129	-0.16	155	-0.16	207	0.16
125000	63	0	79	0	127	0	159	0	191	0	255	0
115200	-	-	86	0.22	138	0.08	173	0.22	207	-0.16	278	-0.08
76800	103	-0.16	129	-0.16	207	-0.16	259	-0.16	311	-0.16	416	0.08
57600	138	0.08	173	0.22	277	0.08	346	-0.06	416	0.08	555	0.08
38400	207	-0.16	259	-0.16	416	0.08	520	0.03	624	0	832	-0.04
28800	277	0.08	346	<0.01	554	-0.01	693	-0.06	832	-0.03	1110	-0.01
19200	416	0.08	520	0.03	832	-0.03	1041	0.03	1249	0	1666	0.02
10417	767	<0.01	959	<0.01	1535	<0.01	1919	<0.01	2303	<0.01	3071	<0.01
9600	832	0.04	1041	0.03	1666	0.02	2083	0.03	2499	0	3332	0.01
7200	1110	<0.01	1388	<0.01	2221	<0.01	2777	<0.01	3332	<0.01	4443	-0.01
4800	1666	0.02	2082	-0.02	3332	<0.01	4166	<0.01	4999	0	6666	<0.01
2400	3332	<0.01	4166	<0.01	6666	<0.01	8332	<0.01	9999	0	13332	<0.01
1200	6666	<0.01	8334	0.02	13332	<0.01	16666	<0.01	19999	0	26666	<0.01
600	13332	<0.01	16666	<0.01	26666	<0.01	-	-	-	-		
300	26666	<0.01	-	-	-	-	-	-	-	-		

■ Counting Example

Assume the reload value is 832. The figure 6-2 demonstrates the behavior of the both Reload Counters:

Figure 6-2 Counting example of the reload counters



(Note) The falling edge of the Serial Clock Signal always occurs after | (v + 1) / 2 |.

6.2 Restarting the Reload Counter

The Reload Counter can be restarted because of the following reasons:

Transmission and Reception Reload Counter:

- Global MCU Reset
- USART programmable clear (SMR04:UPCL bit)
- User programmable restart (SMR04: REST bit)

Reception Reload Counter:

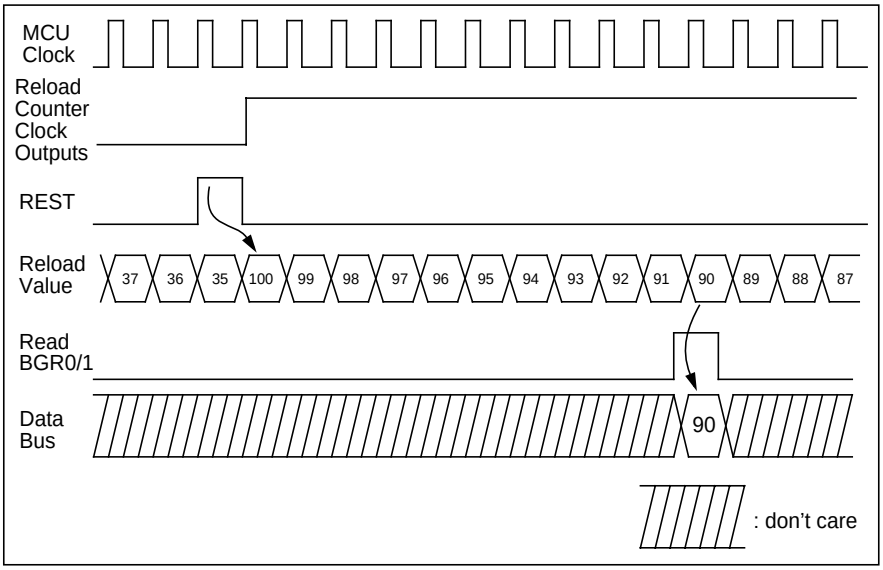
- Start bit falling edge detection in asynchronous mode

■ Programmable Restart

If the REST bit of the Serial Mode Register (SMR04) is set by the user, both Reload Counters are restarted at the next clock cycle. This feature is intended to use the Transmission Reload Counter as a small timer.

The following figure illustrates a possible usage of this feature (assume that the reload value is 100.)

Figure 6-3 Reload Counter Restart example



In this example the number of MCU clock cycles (cyc) after REST is then:

***cyc* = *v* - *c* + 1 = 100 - 90 + 1 = 11**,

where *v* is the reload value and *c* is the read counter value.

(Note) If USART is reset by setting SMR04:UPCL, the Reload Counters will restart too.

■ Automatic Restart

In asynchronous UART mode if a falling edge of a start bit is detected the Reception Reload Counter is restarted. This is intended to synchronize the serial input shifter to the incoming serial data stream.

7. USART Operation

USART operates in operation mode 0 for normal bidirectional serial communication, in mode 2 and 3 in bidirectional communication as master or slave, and in mode 1 as master or slave in multiprocessor communication.

■ Operation of USART

• Operation modes

There are four USART operation modes: modes 0 to 3. As listed in table 7-1, an operation mode can be selected according to the inter-CPU connection method and data transfer mode.

Table 7-1 USART operation mode

Operation mode		Data length		Synchroniza-tion of mode	Length of stop bit	data bit direction*
		parity disabled	parity enabled			
0	normal mode	7 or 8		asynchronous	1 or 2	L/M
1	multiprocessor	7 or 8 + 1**	-	asynchronous	1 or 2	L/M
2	normal mode	8		synchronous	0, 1 or 2	L/M
3	LIN mode	8	-	asynchronous	1	L

* means the data bit transfer format: LSB or MSB first

** "+1" means the indicator bit of the address/data selection in the multiprocessor mode, instead of parity.

(Note) Mode 1 operation is supported both for master or slave operation of USART in a master-slave connection system. In Mode 3 the USART function is locked to 8N1-Format, LSB first.

If the mode is changed, USART cuts off all possible transmission or reception and awaits then new action.

■ Inter-CPU Connection Method

External Clock One-to-one connection (normal mode) and master-slave connection (multiprocessor mode) can be selected. For either connection method, the data length, whether to enable parity, and the synchronization method must be common to all CPUs. Select an operation mode as follows:

- In the one-to-one connection method, operation mode 0 or 2 must be used in the two CPUs. Select operation mode 0 for asynchronous transfer mode and operation mode 2 for synchronous transfer mode.
  Note, that one CPU has to set to the master and one to the slave in synchronous mode 2.
- Select operation mode 1 for the master-slave connection method and use it either for the master or slave system.

■ Synchronization Methods

In asynchronous operation USART reception clock is automatically synchronized to the falling edge of a received start bit.

In synchronous mode the synchronization is performed either by the clock signal of the master device or by USART itself if operating as master.

■ Signal Mode

USART can treat data in non-return to zero (NRZ) and return to zero (RZ) format. For this option the ECCR: INV bit is provided.

■ Operation Enable Bit

USART controls both transmission and reception using the operation enable bit for transmission (SCR04:

TXE) and reception (SCR04: RXE). If each of the operations is disabled, stop it as follows:

- If reception operation is disabled during reception (data is input to the reception shift register), finish frame reception and read the received data of the reception data register (RDR04). Then stop the reception operation.
- If the transmission operation is disabled during transmission (data is output from the transmission shift register), wait until there is no data in the transmission data register (TDR04) before stopping the transmission operation.

7.1 Operation in Asynchronous Mode (Op. Modes 0 and 1)

When USART is used in operation mode 0 (normal mode) or operation mode 1 (multiprocessor mode), the asynchronous transfer mode is selected.

■ Transfer data format

Generally each data transfer in the asynchronous mode operation begins with the start bit (low-level on bus) and ends with at least one stop bit (high-level). The direction of the bit stream (LSB first or MSB first) is determined by the BDS-Bit of the Serial Status Register (SSR04). The parity bit (if enabled) is always placed between the last data bit and the (first) stop bit.

In operation mode 0 the length of the data frame can be 7 or 8 bits, with or without parity, and 1 or 2 stop bits.

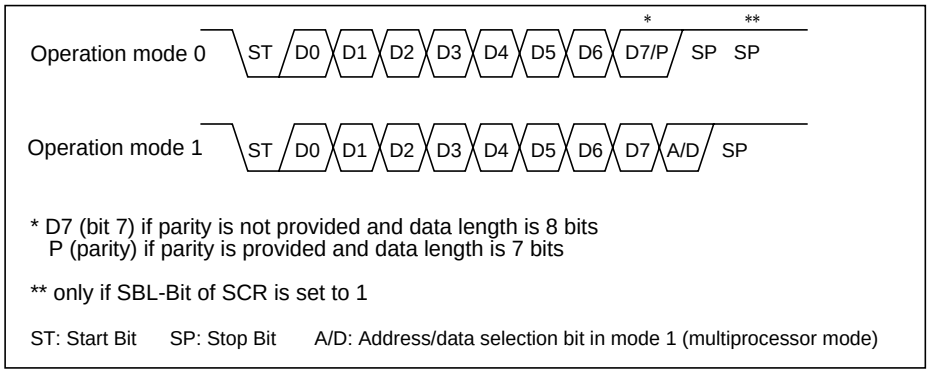
In operation mode 1 the length of the data frame can be 7 or 8 bits with a following address-/data-selection bit instead of a parity bit. 1 or 2 stop bits can be selected.

The calculation formula for the bit length of a transfer frame is:

Length = 1 + d + p + s

(d = number of data bits [7 or 8], p = parity [0 or 1], s = number of stop bits [1 or 2]

Figure 7-1 Transfer data format (operation modes 0 and 1)



(Note) If BDS-Bit of the Serial Status Register (SSR04) is set to "1" (MSB first), the bit stream processes as: D7, D6, ... , D1, D0, (P).

During Reception both stop bits are detected, if selected. But the Reception data register full (RDRF) flag will go "1" at the first stop bit. The bus idle flag (RBI of ECCR04) goes "1" after the second stop bit if no further start bit is detected. (The second stop bit belongs to "bus activity", although it is just mark level.)

■ Transmission Operation

If the Transmission Data Register Empty (TDRE) flag bit of the Serial Status Register (SSR04) is "1", transmission data is allowed to be written to the Transmission Data Register (TDR04). When data is written, the TDRE flag goes "0". If the transmission operation is enabled by the TXE-Bit ("1") of the Serial Control Register (SCR04), the data is written next to the transmission shift register and the transmission starts at the next clock cycle of the serial clock, beginning with the start bit. Thereby the TDRE flag goes "1", so that new data can be written to the TDR04.

If transmission interrupt is enabled (TIE = 1), the interrupt is generated by the TDRE flag. Note, that the initial value of the TDRE flag is "1", so that in this case if TIE is set to "1" an interrupt will occur immediately.

■ Reception Operation

Reception operation is performed every time it is enabled by the Reception Enable (RXE) flag bit of the SCR04. If a start bit is detected, a data frame is received according to the format specified by the SCR04. By occurring errors, the corresponding error flags are set (PE, ORE, FRE). However after the reception of the data frame the data is transferred from the serial shift register to the Reception Data Register (RDR04) and the Receive Data Register Full (RDRF) flag bit of the SSR5 is set. The data then has to be read by the CPU. By doing so, the RDRF flag is cleared. If reception interrupt is enabled (RIE = 1), the interrupt is simply generated by the RDRF.

Note: Only when the RDRF flag bit is set and no errors have occurred the Reception Data Register (RDR04) contains valid data.

■ Stop Bit, Error Detection, and Parity

For transmission, 1 or 2 stop bits can be selected. During reception, if selected, both stop bits are checked, to set the reception bus idle (RBI) flag of ECCR04 correctly after the second stop bit.

In mode 0 parity, overrun, and framing errors can be detected.

In mode 1, overrun and framing errors can be detected. Parity is not provided.

By setting the Parity Enable (PEN) bit of the Serial Control Register (SCR04) the USART provides parity calculation (during transmission) and parity detection and check (during reception) in mode 0 (and mode 2 if the SSM bit of ECCR04 is set).

Even parity is set, if the P bit of SCR04 is cleared, odd parity if the flag bit is set. In mode 1, overrun and framing errors can be detected. Parity is not provided.

■ Signal mode NRZ and RZ

To set USART to the NRZ data format set the ECCR04:INV bit to 0 (initial value).
RZ data format is set, if the ECCR04:INV bit was set to 1.

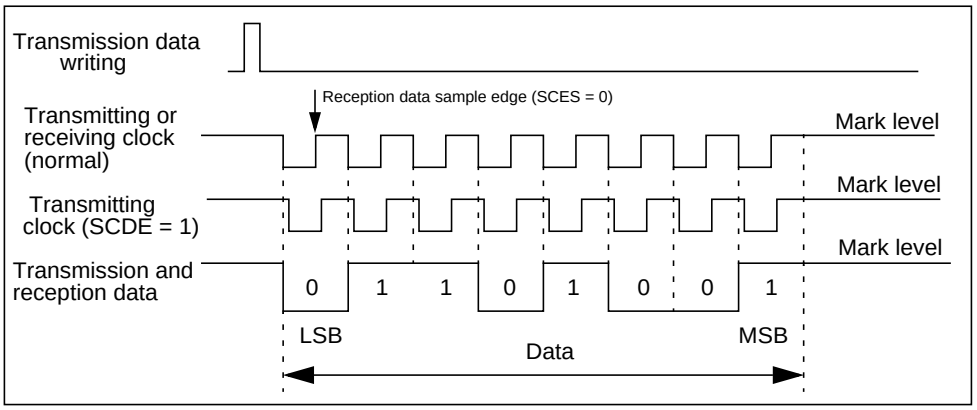
7.2 Operation in Synchronous Mode (Operation Mode 2)

The clock synchronous transfer method is used for USART operation mode 2 (normal mode).

■ Transfer data format (standard synchronous)

In the synchronous mode, 8-bit data is transferred with no start or stop bits if the SSM bit of the Extended Communication Control Register (ECCR04) is 0. A special clock signal belongs to the data format in mode 2. The figure below illustrates the data format during a transmission in the synchronous operation mode

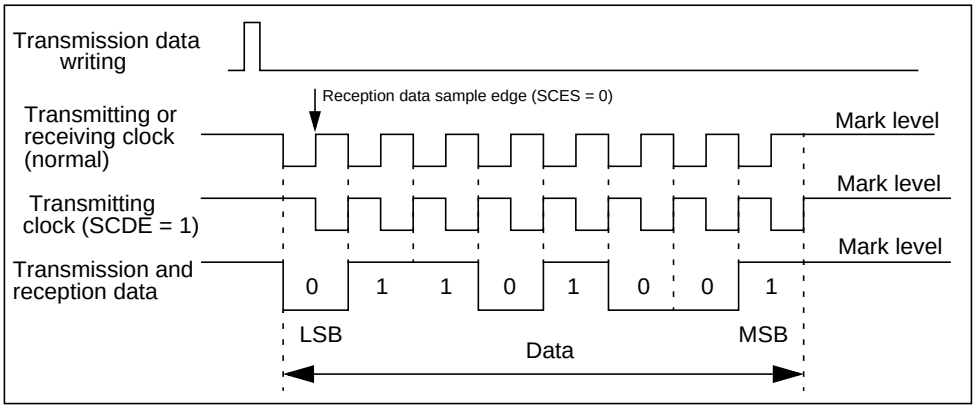
Figure 7-2 Transfer data format (operation mode 2).



■ Transfer data format

In the synchronous mode, 8-bit data is transferred with no start or stop bits if the SSM bit of the Extended Communication Control Register (ECCR04) is 0. A special clock signal belongs to the data format in mode 2. The figure below illustrates the data format during a transmission in the synchronous operation mode.

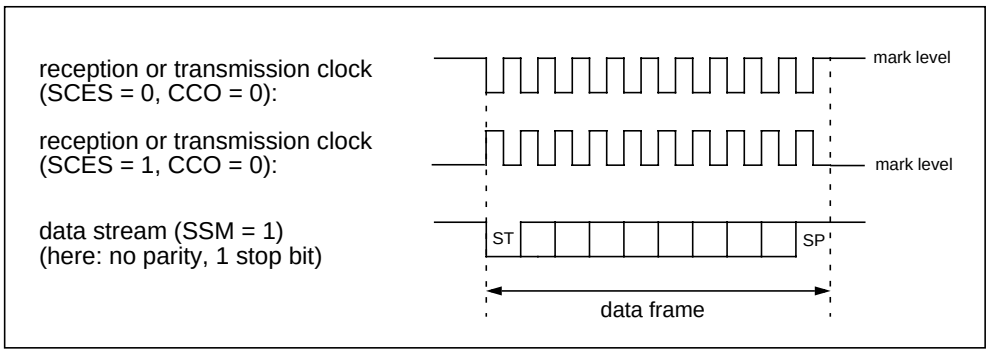
Figure 7-3 SPI Transfer data format (operation mode 2)



■ Clock inversion and start/stop bits in mode 2

If the SCES bit of the Extended Status/Control Register (ESCR04) is set the serial clock is inverted. Therefore in slave mode USART samples the data bits at the falling edge of the received serial clock. Note, that in master mode if SCES is set the clock signal's mark level is "0". If the SSM04 bit of the Extended Communication Control Register (ECCR04) is set the data format gets additional start and stop bits like in asynchronous mode

Figure 7-4 Transfer data format with clock inversion



■ Clock Supply

In clock synchronous (normal) mode (I/O extended serial), the number of the transmission and reception bits has to be equal to the number of clock cycles. Note, that if start/stop bits communication is enabled, the number of clock cycles has to match with the quantity for the additional start and stop bit(s).

If the internal clock (dedicated reload counter) is selected, the data receiving synchronous clock is generated automatically if data is transmitted.

If external clock is selected, be sure, that the transmission side of the Transmission Data Register contains data and then clock cycles for each bit to sent have to be generated and supplied from outside. The mark level ("H") must be retained before transmission starts and after it is complete if SCES is "0".

Setting the SCDE bit of ECCR delays the transmitting clock signal by 1 CLKP cycle (or half a clock period in

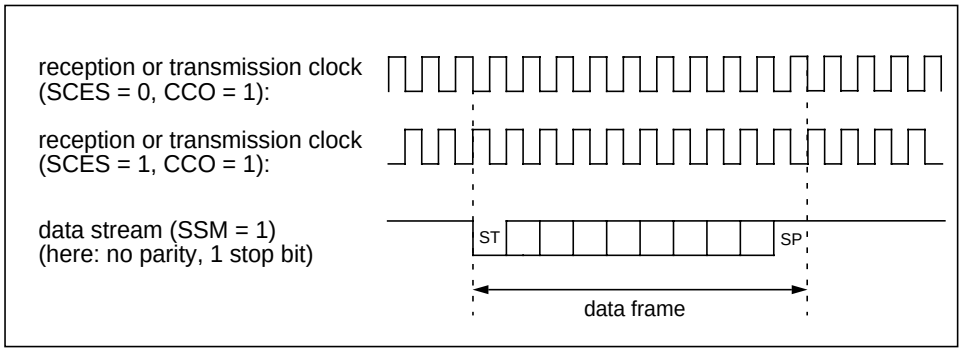
SPI). This will make sure, that the transmission data is valid and stable at any falling clock edge. (Necessary, if the receiving device samples the data at falling clock edge). This function is disabled when CCO is enabled.

If the Serial Clock Edge Select (SCES) bit of the ESCR is set, the USARTs clock is inverted and thus samples the reception data at the falling clock edge. In this case, the sending device must make sure that the serial data is valid at the falling serial clock edge.

When both the SCES and the SCDE bit are set, data is stable at the rising clock edge, as in the case of SCES = SCDE = 0. However, the marker value for idle state is inverted (low).

If the CCO bit of the Extended Status/Control Register (ESCR5) is set, the serial clock on the SCK5 pin in master mode is continuously clocked out. It is strongly recommended to use start and stop bits in this mode to signalize the receiver, when a data frame begins and when it stops. Figure 7-5 illustrates this.

Figure 7-5 Continuous clock output in mode 2



■ Data signal mode

NRZ data format is selected, if ECCR04: INV = 0, otherwise the signal mode for the serial data input and output pin is RZ.

■ Error Detection

If no Start/Stop bits are selected (ECCR04: SSM = 0) only overrun errors are detected.

■ Communication

For initialization of the synchronous slave mode, the following settings have to be done:

- Baud Rate Generator Registers (BGR0/1):
  Set the desired reload value for the dedicated Baud Rate Reload Counter
- Serial Control Register (SCR04):
  - RXE, TXE: set both of these flags to “0“
  - PEN: no parity provided - Value: don’t care
  - P, SBL, A/D: no parity, no stop bit(s), no Address/Data selection - Value: don’t care
  - CL: automatically fixed to 8-bit data - Value: don’t care
  - CRE: "1" (the error flag is cleared for initialization, possible transmission or reception will cut off)
- Serial Mode Control Register (SMR04):
  - MD1, MD0: "10b" (Mode 2)
  - SCKE: “1” for dedicated Baud Rate Reload Counter
    “0” for external clock input
  - SOE: “1” for transmission and reception
    “0” for reception only\
- Serial Status Register (SSR04):
  - BDS: "0" for LSB first, “1" for MSB first

- RIE: “1" if interrupts are used; "0" if not
- TIE: "1" if interrupts are used; "0" if not

- Extended Communication Control Register (ECCR04):
  - SSM: "0" if no start/stop bits are desired (normal)
    "1" for adding start/stop bits (special)
  - MS: "0" for master mode (USART generates the serial clock);
    "1" for slave mode (USART receives serial clock from the master device)
- Serial Control Register (SCR04):
  - RXE, TXE: set one or both of these control bits to "1" to begin communication.

To start the communication, write data into the Transmission Data Register (TDR04).

To receive data, disable the Serial Output Enable (SOE) bit of the SMR04 and write dummy data to TDR04.

(Note) Setting continuous clock and start-/stop-bit mode, duplex transfer is possible like in asynchronous modes.

7.3 Operation with LIN Function (Operation Mode 3)

USART can be used either for LIN-Master devices or LIN-Slave devices. For this LIN function a special mode (3) is provided. Setting the USART to mode 3, configure the data format to
8N1-LSB-first format.

■ USART as LIN master

In LIN master mode, the master determines the baud rate of the whole sub bus. Therefore, slave devices have to synchronize to the master and the desired baud rate remains fixed in master operation after initialization.

Writing a "1" into the LBR bit of the Extended Status/Communication Register (ECCR04) generates a 13 - 16 bit times low-level on the SOT04 pin, which is the LIN synchronization break and the start of a LIN message. Thereby the TDRE flag of the Serial Status Register (SSR04) goes "0" and is reset to "1" after the break, and generates a transmission interrupt for the CPU (if TIE of SSR04 is "1").

The length of the Synchronization break to be sent can be determined by the LBL1/0 bits of the ESCR04 as follows:

Table 7-2 LIN break length

LBL1	LBL0	Length of Break
0	0	13 Bit times
0	1	14 Bit times
1	0	15 Bit times
1	1	16 Bit times

The Synch Field can be sent as a simple 0x55-Byte after the LIN break. To prevent a transmission interrupt, the 0x55 can be written to the TDR04 just after writing the "1" to the LBR bit, although the TDRE flag is "0". The internal transmission shifter waits until the LIN break has finished and shifts the TDR04 value out afterwards. In this case no interrupt is generated after the LIN break and before the start bit.

■ USART as LIN slave

In LIN slave mode USART has to synchronize to the master’s baud rate. If Reception is disabled (RXE = 0) but LIN break Interrupt is enabled (LBIE = 1) USART will generate a reception interrupt, if a synchronization break of

the LIN master is detected, and indicates it with the LBD flag of the ESCR04. Writing a "0" to this bit clears the interrupt. The next step is to analyze the baud rate of the LIN master. The first falling edge of the Synch Field is detected by USART. The USART signals it then to the Input Capture Unit (ICU1/5) via a rising edge of an internal connection. The fifth falling edge resets the ICU signal. Therefore the ICU has to be configured for the LIN input capture (PFR14.4=1, EPFR14.4=1) and its interrupts have to be enabled (ICS4). The values of th ICU counter register after the first Interrupt (a) and after the second interrupt (b) yield the BGR value:

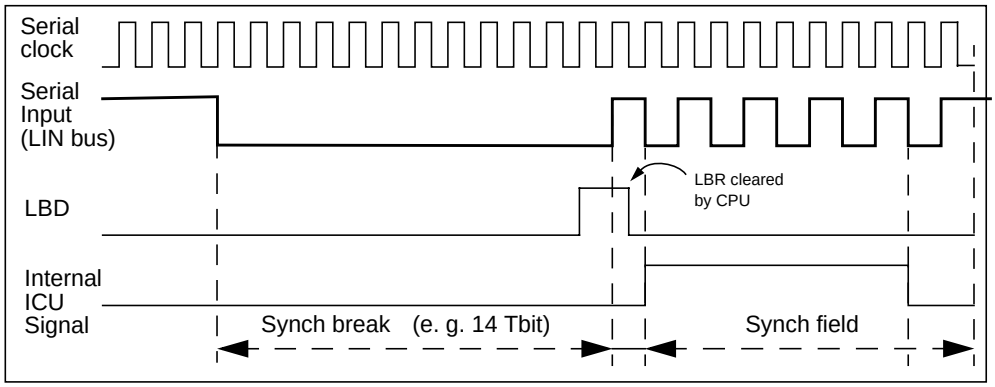
without timer overflow:BGR value = (b - a) / 8 ,

with timer overflow: BGR value = (max - b + a) / 8 ,

where max is the timer maximum value at which the overflow occurs.

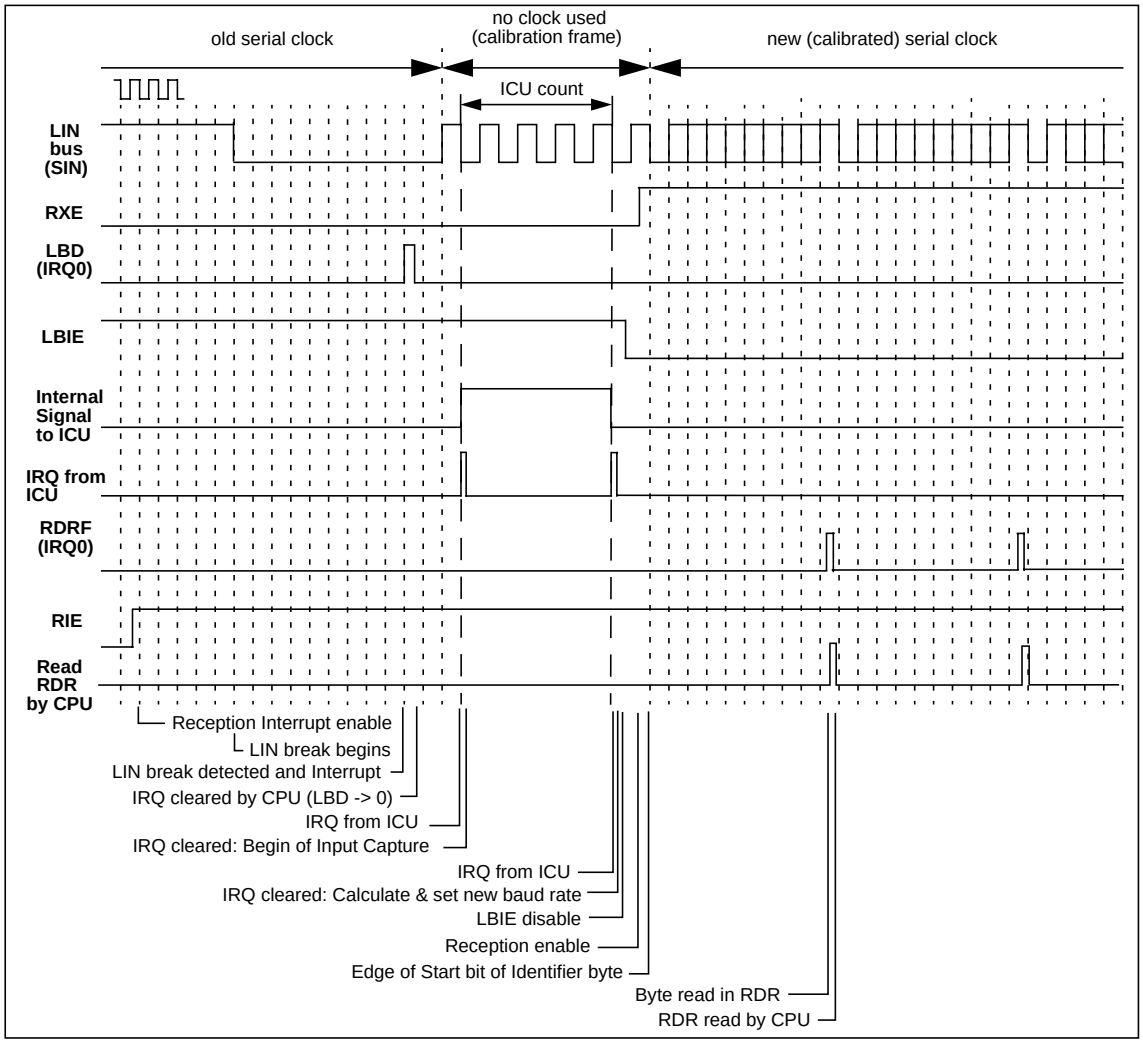
The figure 7-6 shows a typical start of a LIN message frame and the behavior of the USART.

Figure 7-6 USART behavior as slave in LIN mode



■ LIN bus timing

Figure 7-7 LIN bus timing and USART signals



7.4 Direct Access to Serial Pins

USART allows the user to access directly the transmission pin (SOT04) or the reception pin (SIN04).

■ USART Direct Pin Access

The USART provides the ability for the user to directly access the serial input or output pin. The software can always monitor the incoming serial data by reading the SIOP bit of the ESCR04. If setting the Serial Output Pin direct access Enable (SOPE) bit of the ESCR04 the software can force the SOT04 pin to a desired value. Note, that this access is only possible, if the transmission shift register is empty (i. e. no transmission activity).

In LIN mode this function can be used for reading back the own transmission and is used for error handling if something is physically wrong with the single-wire LIN-bus.

(Note) Write the desired value to the SIOP pin **before** enabling the output pin access, to prevent undesired peaks. The peaks can occur because SIOP holds the last written value.

During a Read-Modify-Write operation the SIOP bit returns the actual value of the SOT04 pin in the read cycle instead the value of SIN04 during a normal read instruction.

7.5 Bidirectional Communication Function (Normal Mode)

In operation mode 0 or 2, normal serial bidirectional communication is available. Select operation mode 0 for asynchronous communication and operation mode 2 for synchronous communication.

■ Bidirectional Communication Function

The settings shown in figure 7-8 are required to operate USART in normal mode (operation mode 0 or 2).

Figure 7-8 Settings for USART operation mode 0 and 2

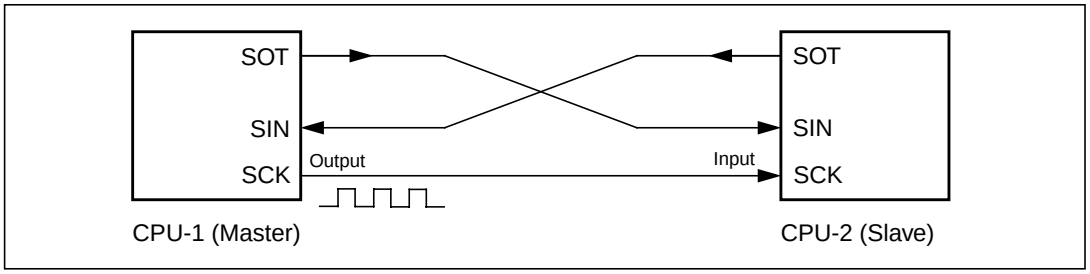
bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SCR04,SMR04	PEN	P	SBL	CL	AD	CRE	RXE	TXE	MD1	MD0	OTO	EXT	REST	UPCL	SCKE	SOE
Mode 0	◎	◎	◎	◎	x	0	◎	◎	0	0	x	0	0	0	◎	◎
Mode 2	□	□	□	x	x	0	◎	◎	1	0	◎	◎	0	0	◎	◎
SSR04, TDR04/RDR04	PE	ORE	FRE	RDRF	TDRE	BDS	RIE	TIE	Set transmission data (during writing) Retain reception data (during reading)							
Mode 0	◎	◎	◎	◎	◎	◎	◎	◎								
Mode 2	□	◎	□	◎	◎	◎	◎	◎								
ESCR04,ECCR04	LBIE	LBD	LBL1	LBL0	SOPE	SIOP	CCO	SECS	-	LBR	MS	SCDE	SSM	BIE	RBI	TBI
Mode 0	x	x	x	x	x	x	x	x		x	x	x	x	x	x	x
Mode 2	x	x	x	x	x	x	◎	◎		x	◎	◎	◎	x	x	x

◎ Bit is used
x Bit is not used
0 / 1 Set bit to 0 / 1
□ Bit is used if SSM = 1 (Synchronous start-/stop-bit mode)

■ Inter-CPU Connection

As shown in figure 7-9, interconnect two CPUs in USART mode 2

Figure 7-9 Connection example of USART mode 2 bidirectional communication



7.6 Master-Slave Communication Function (Multiprocessor Mode)

USART communication with multiple CPUs connected in master-slave mode is available for both master or slave systems.

■ Master-slave Communication Function

The settings shown in figure 7-10 are required to operate USART in multiprocessor mode (operation mode 1).

Figure 7-10 Settings for USART operation mode 1

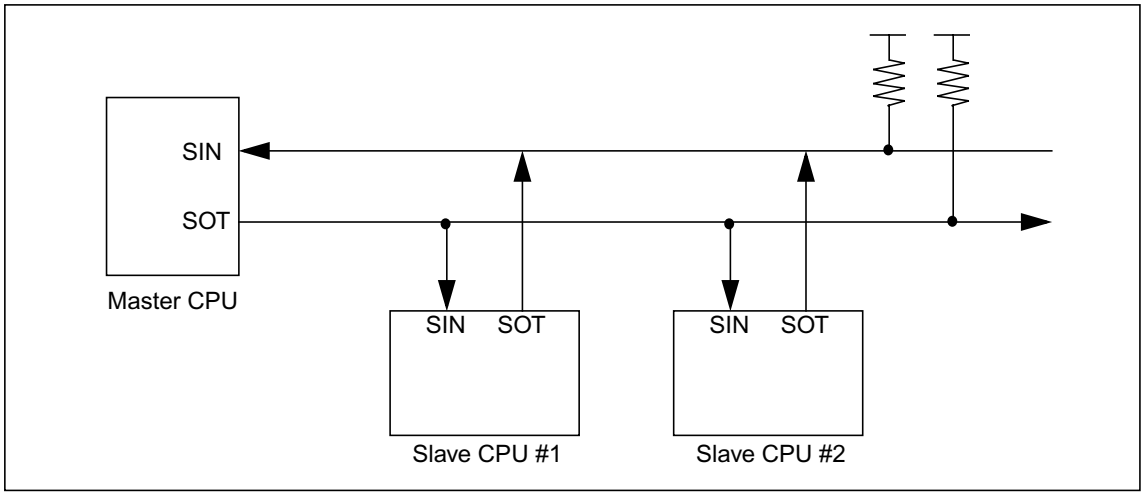
bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SCR04,SMR04	PEN	P	SBL	CL	AD	CRE	RXE	TXE	MD1	MD0	OTO	EXT	REST	UPCL	SCKE	SOE
Mode 1	x	x	◎	◎	◎	0	◎	◎	0	1	x	0	0	0	1	◎
SSR04, TDR04/RDR04	PE	ORE	FRE	RDRF	TDRE	BDS	RIE	TIE	Set transmission data (during writing) Retain reception data (during reading)							
Mode 1	x	◎	◎	◎	◎	◎	◎	◎								

◎ Bit is used
x Bit is not used
0 / 1 Set bit to 0 / 1

■ Inter-CPU Connection

As shown in figure 7-11, a communication system consists of one master CPU and multiple slave CPUs connected to two communication lines. USART can be used for the master or slave CPU.

Figure 7-11 Connection example of USART master-slave communication



■ Function Selection

Select the operation mode and data transfer mode for master-slave communication as shown in table 7-3.

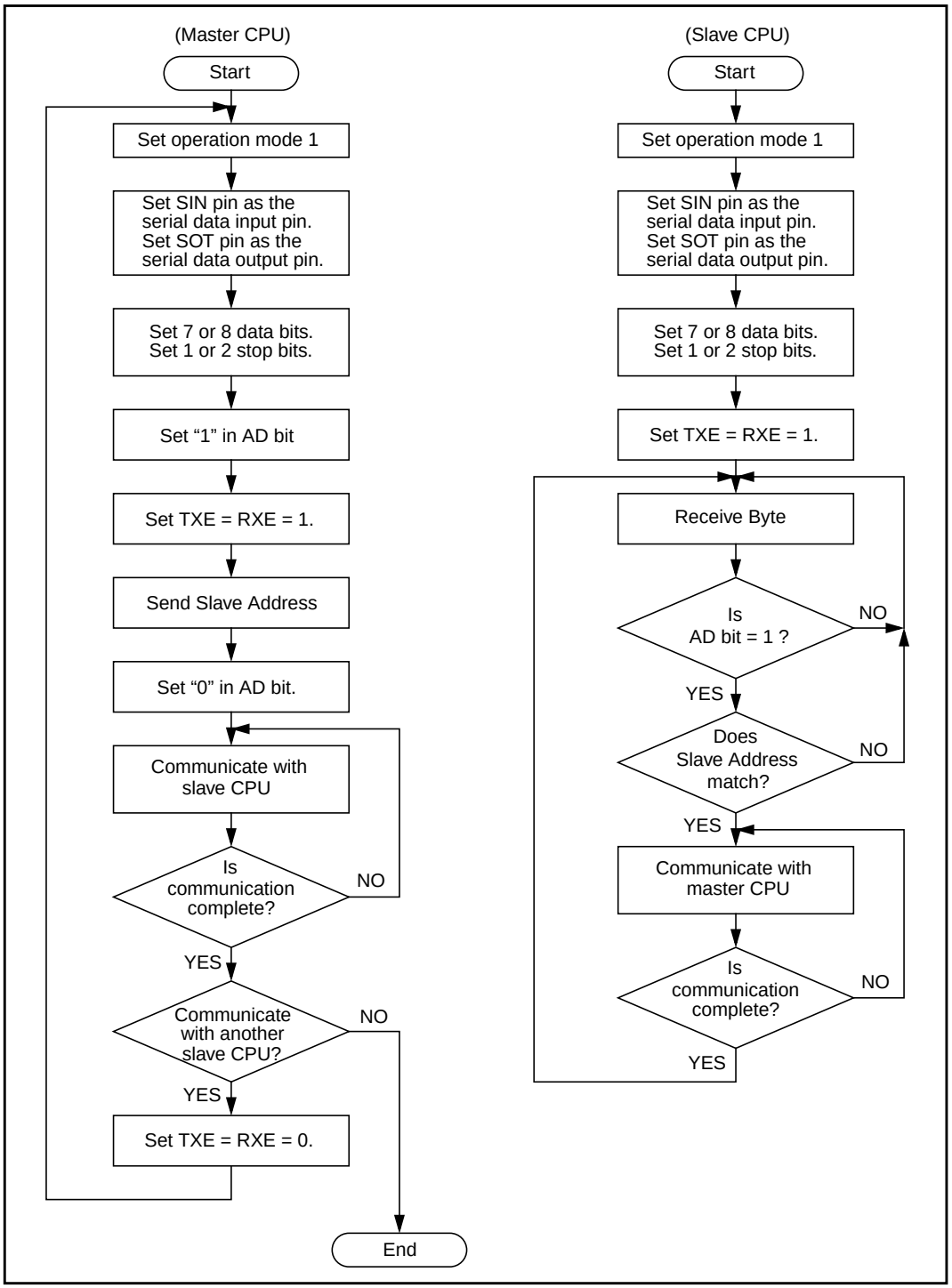
Table 7-3 Selection of the master-slave communication function

	Operation mode		Data	Parity	Synchro-nization method	Stop bit	Bit direction
	Master CPU	Slave CPU					
Address transmis-sion and reception	Mode 1 (send AD-bit)	Mode 1 (receive AD-bit)	AD="1" + 7- or 8-bit address	None	Asyn-chronous	1 or 2 bits	LSB or MSB first
Data trans-mission and recep-tion			AD="0" + 7- or 8-bit data				

■ Communication Procedure

When the master CPU transmits address data, communication starts. The A/D bit in the address data is set to 1, and the communication destination slave CPU is selected. Each slave CPU checks the address data using a program. When the address data indicates the address assigned to a slave CPU, the slave CPU communicates with the master CPU (ordinary data). Figure 7-12 shows a flowchart of master-slave communication (multiprocessor mode)

Figure 7-12 Master-slave communication flowchart



7.7 LIN Communication Function

USART communication with LIN devices is available for both LIN master or LIN slave systems.

■ LIN-Master-Slave Communication Function

The settings shown in the figure below are required to operate USART in LIN communication mode (operation mode 3).

Figure 7-13 Settings for USART

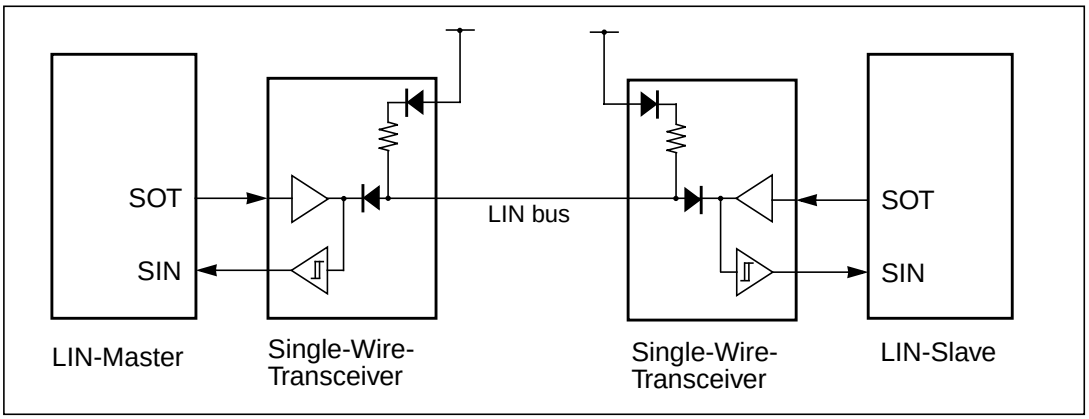
bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SCR04,SMR04	PEN	P	SBL	CL	AD	CRE	RXE	TXE	MD1	MD0	OTO	EXT	REST	UPCL	SCKE	SOE
Mode 3	x	x	+	+	x	0	◎	◎	1	1	x	0	0	0	1	◎
SSR04, TDR04/RDR04	PE	ORE	FRE	RDRF	TDRE	BDS	RIE	TIE	Set transmission data (during writing) Retain reception data (during reading)							
Mode 3	x	◎	◎	◎	◎	+	◎	◎								
ESCR04,ECCR 04	LBIE	LBD	LBL1	LBL0	SOPE	SIOP	CCO	SECS	-	LBR	MS	SCDE	SSM	BIE	RBI	TBI
Mode 3	◎	◎	◎	◎	x	x	x	x		◎	x	x	x	◎	x	x

◎ Bit is used
x Bit is not used
0 / 1 Set bit to 0 / 1
\+ Bit is automatically set to the correct value

■ LIN device connection

As shown in the Figure below, a communication system of one LIN-Master device and a LIN-Slave device. USART can operate both as LIN-Master or LIN-Slave.

Figure 7-14 Connection example of a small LIN-Bus system

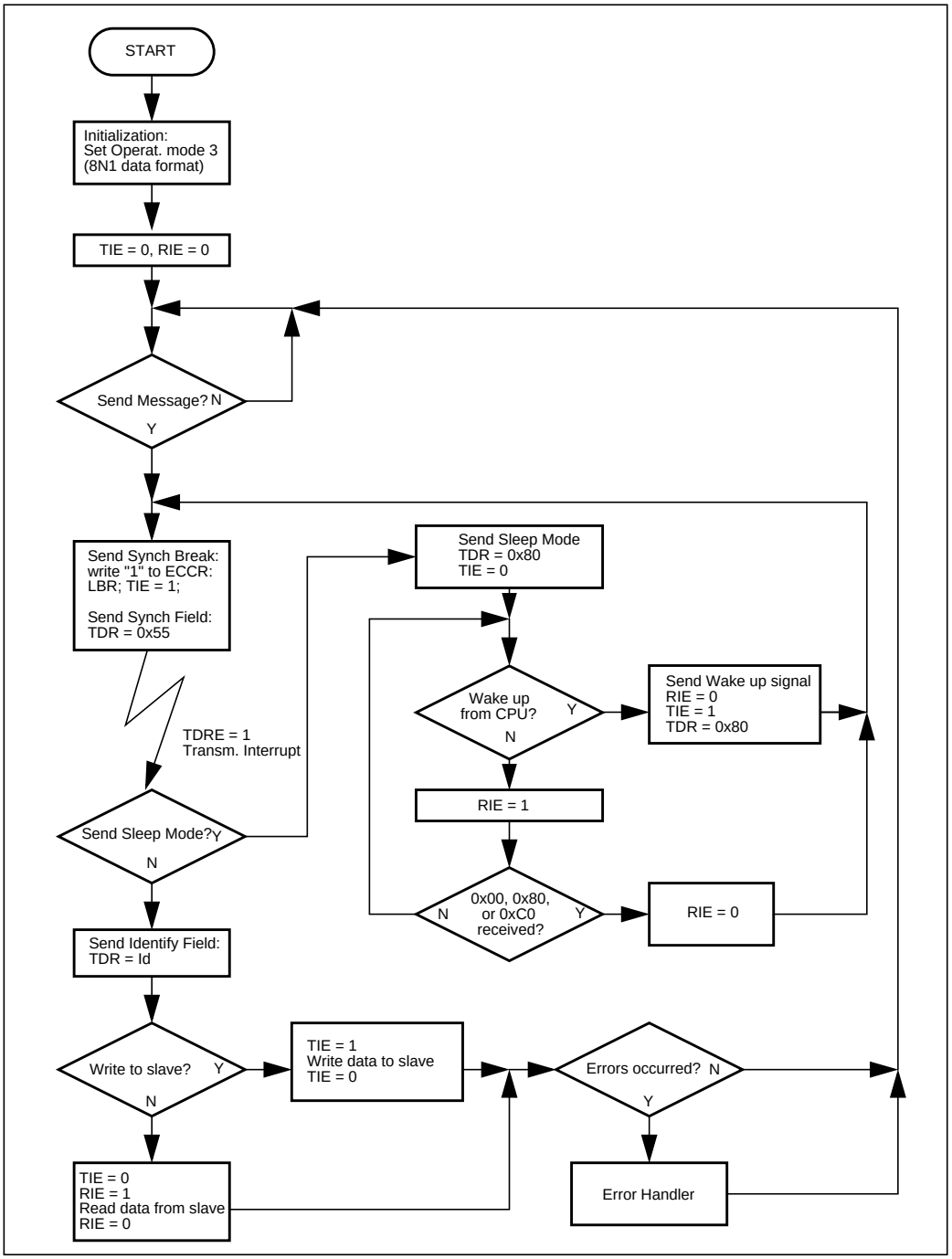


7.8 Sample Flowcharts for USART in LIN Communication (Operation Mode 3)

This section contains sample flowcharts for USART in LIN communication.

■ USART as master device

Figure 7-15 USART LIN master flow chart



■ USART as slave device

Figure 7-16 USART LIN slave flow chart (part1)

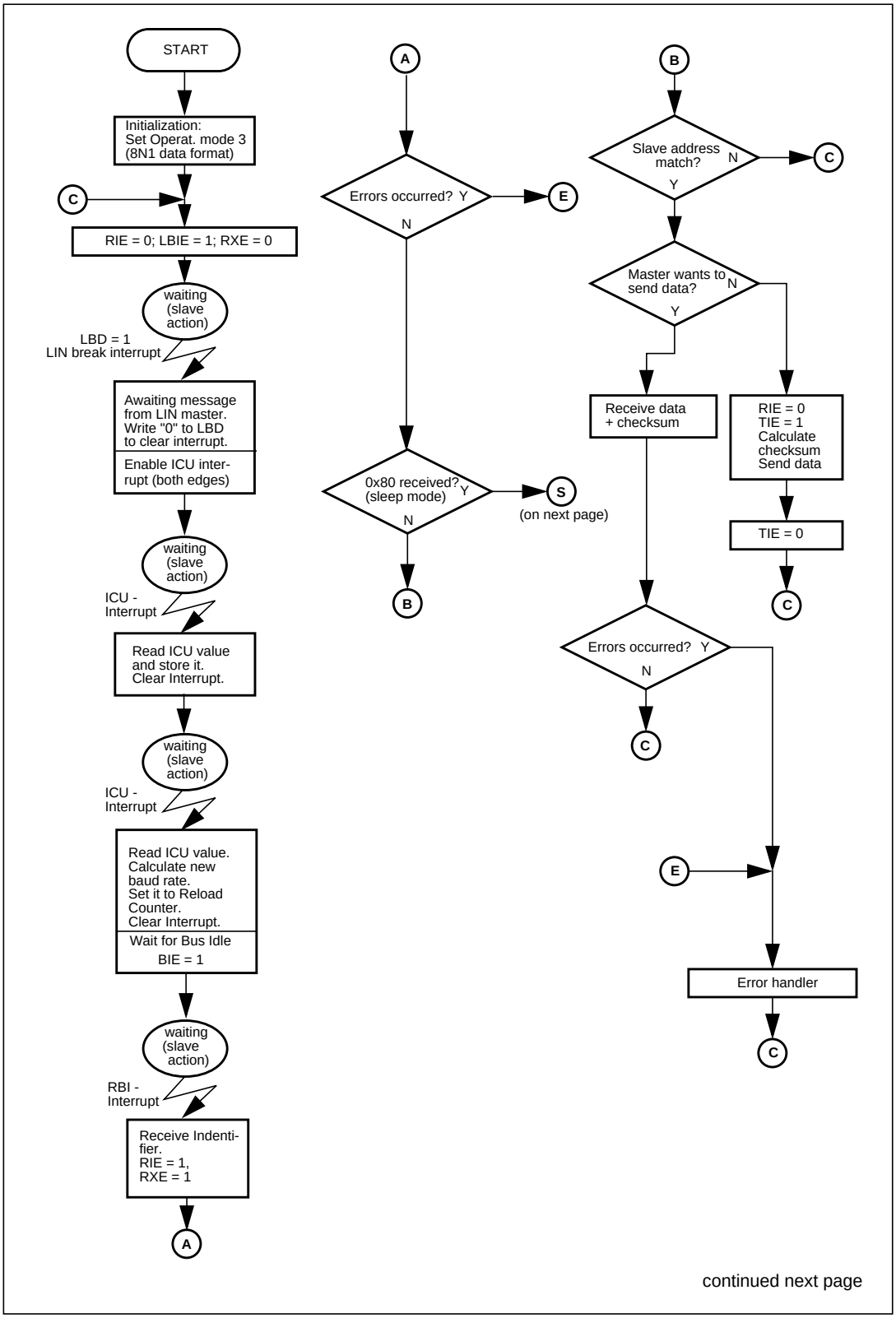
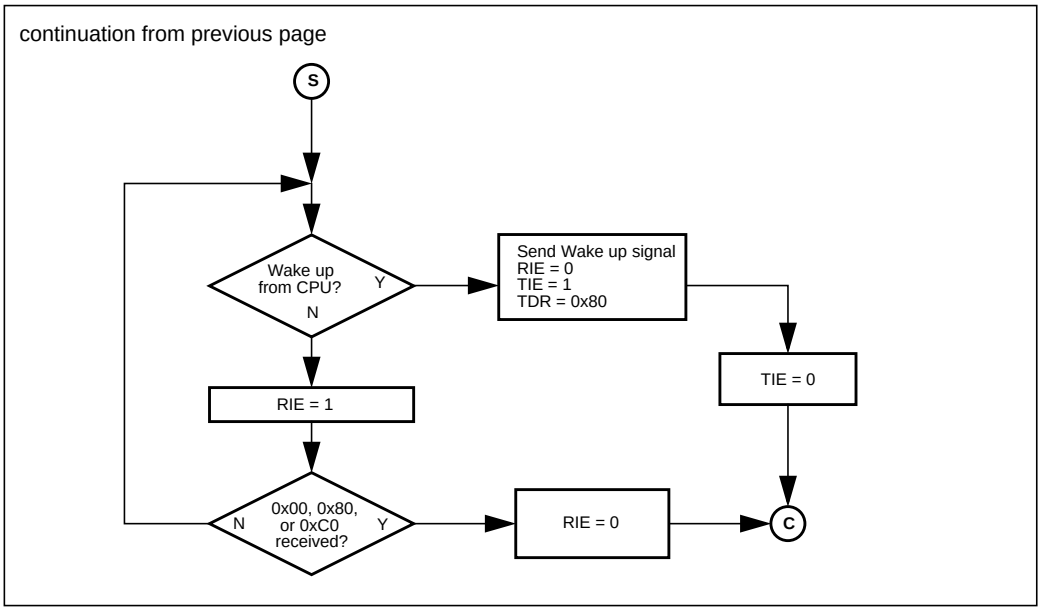


Figure 7-17 USART LIN slave flow chart (part 2)



8. Notes on using USART

Notes on using USART are given below.

■ Enabling Operations

In USART, the control register (SCR04) has TXE (transmission) and RXE (reception) operation enable bits. Both, transmission and reception operations, must be enabled before the transfer starts because they have been disabled as the default value (initial value).

In single wire bus systems like ISO 9141 (LIN bus system) because of the mono directional communication it is highly recommended to enable only one of these two bits at the same time. Because of the automatic reception the sent data by USART would be received by USART too.

■ Cancelling Transfers

Transfers can be cancelled by clearing their operation enable bits TXE / RXE. If RXE is cleared during an ongoing reception, then the USART state machines must be reset (set UPCL=1). If a transmission is ongoing at the same time, it will be cancelled too! In this case, wait until TDRE=1 and TBI=1 before setting UPCL.

■ Software reset of UART

Perform the software reset (SMR: UPCL=1), when the TXE bit of the SCR register is "0".

■ Clearing reception errors

Please set SCR:CRE in synchronous slave mode only, if SCR:RXE = 0.

■ Communication Mode Setting

Set the communication mode while the system is not operating. If the mode is changed during transmission or reception, the transmission or reception is stopped and possible data will be get lost.

■ Transmission Interrupt Enabling Timing

The default (initial value) of the transmission data empty flag bit (SSR04: TDRE) is “1” (no transmission data and transmission data write enable state). A transmission interrupt request is generated as soon as the transmission interrupt request is enabled (SSR04: TIE=1). Be sure to set the TIE flag to “1” after setting the transmission data to avoid an immediate interrupt.

■ Using LIN operation mode 3

The LIN features are also available in mode 0 (transmitting, receiving break), but using mode 3 sets the USART data format automatically to LIN format (8N1, LSB first). So, break features are appliable for bus protocols other than LIN in mode 0. Note, that the transmission time of the break is variable, but the detection is specified to a minimum of 11 serial bit times.

■ Changing Operation Settings

It is recommended to disable the communication (RXE = 0, TXE = 0), if the UART setting or mode is changed or UART is initialized.

It is strongly recommended to reset USART **after changing operation settings.**
Particularly in synchronous mode 2 if (for example) start-/stop-bits are added to or removed from the data format.

<Caution>

If settings in the Serial Mode Register (SMR04) are desired, it is not useful to set the UPCL bit at the same time to reset USART. The correct operation settings are *not* guaranteed in this case. Thus it is recommended to set the bits of the SMR04 and *then* to set them again plus the UPCL bit.

■ LIN slave settings

To initiate USART for LIN slave make sure to set the baudrate before receiving the first LIN synchronization break. This is needed to detect safely the minimum of 11 bit times of a LIN synch break.

■ Software compatibility

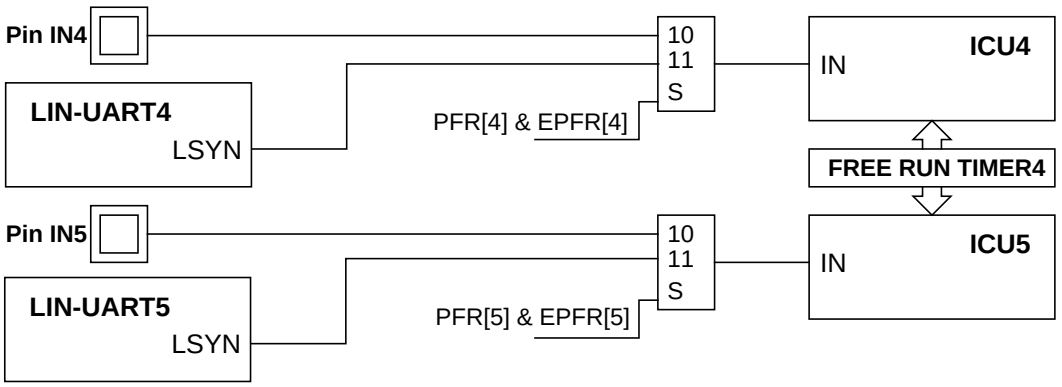
Although USART is similar to older Fujitsu-UARTs it is **not** software compatible to them. The programming models may be the same, but the structure of the registers differ. Furthermore the setting of the baud rate is now determined by a reload value instead of selecting a preset value.

■ Bus Idle Function

The Bus Idle Function cannot be used in synchronous mode 2.

■ Baud Rate Detection Using the Input Capture Units

The USARTs provide the signal LSYN that can be connected to the ICU so that LSYN's pulse length can be measured to derive the baud rate. The connection of the LSYN signals to the ICUs is controlled by the Port 14 function register PFR and EPFR :



If the PFR bit equals '1' and the EPFR bits equals '0', the ICU is connected to its corresponding input pin IN.
If the PFR bit equals '1' and the EPFR bits equals '1', the USARTs are connected to the ICU.

The user has to take into account that:

- ICU4 and ICU5 share one free running timer (prescaler).

■ Effects of reception errors and CRE bit

CRE resets reception state machine and next falling edge at SINn starts reception of new byte. Therefore either set CRE bit immediately (within half bit time) after receiving errors to prevent data stream desynchronization or wait an application dependent time after receiving errors and set CRE, when SINn is idle.

Figure 8-1 Timing of the CRE bit

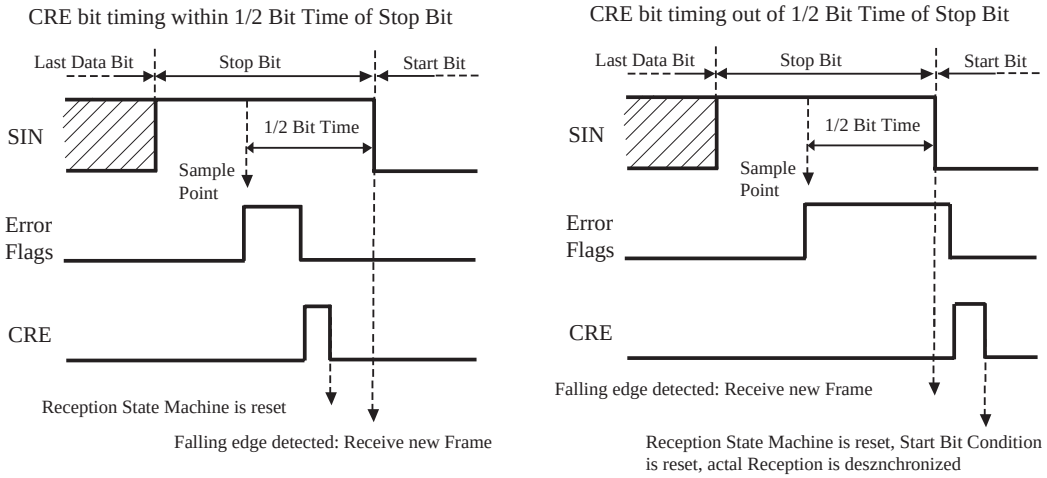
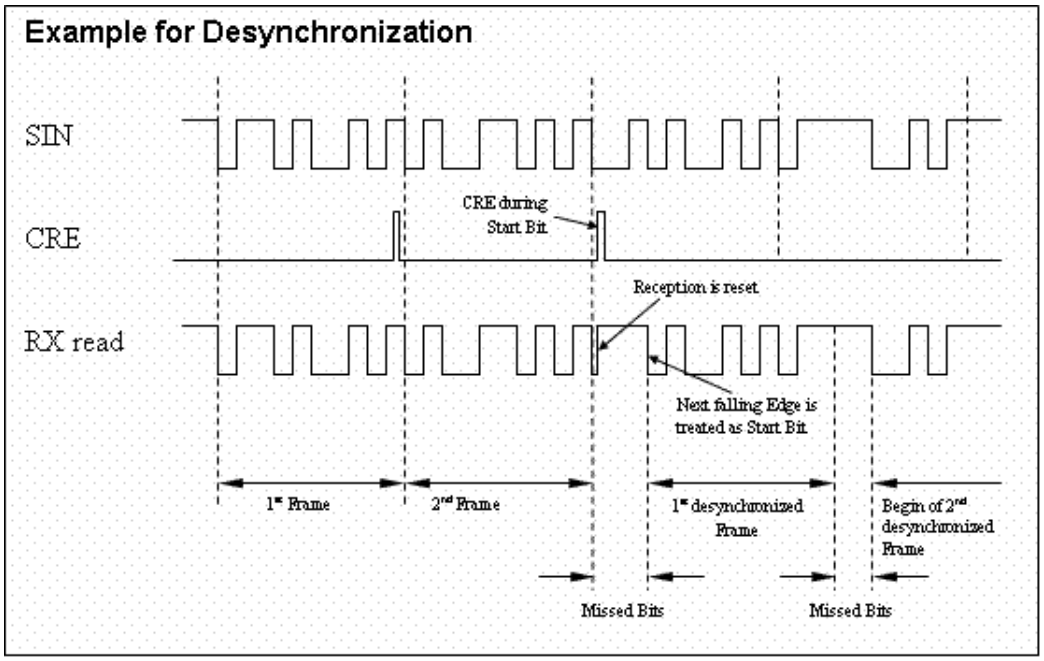
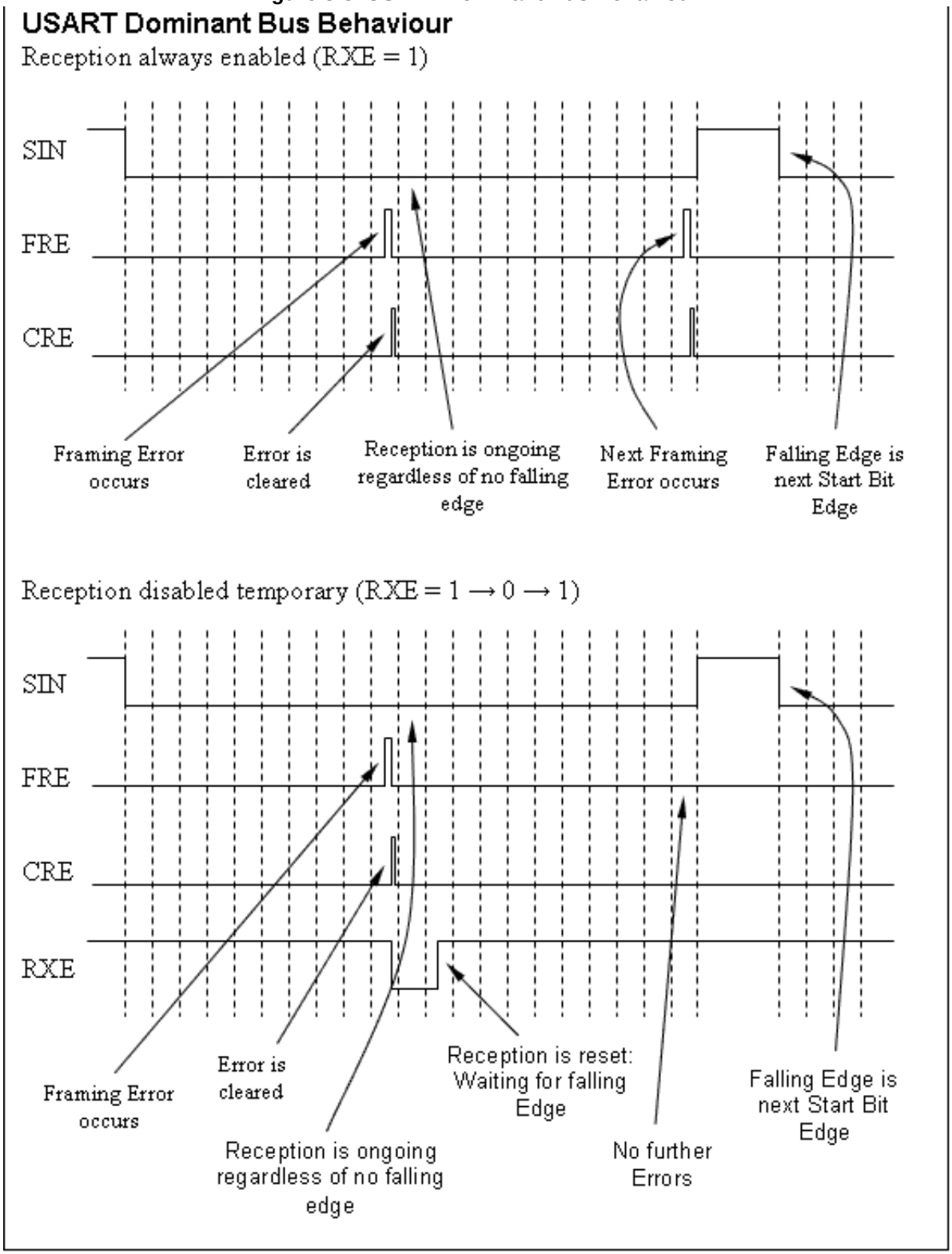


Figure 8-2 Data Stream Synchronisation



Please note, that in case a framing error occurred (stop bit: SINn = "0") and next start bit (SINn = "0") follows immediately, this start bit is recognized regardless of no falling edge before. This is used to remain UART synchronized to the data stream and to determine bus always dominant errors (“Fig. 8-3 on P. 666” upper figure) by producing next framing errors, if a recessive stop bit is expected. If this behaviour is not wanted, please disable the reception temporarily (RXE = 1 -> 0 -> 1) after framing error. In this case, reception goes on at next falling edge on SINn. (“Fig. 8-3 on P. 666“ lower figure).

Figure 8-3 USART Dominant Bus Behaviour



Chapter 33 I^2C Controller

1. Overview

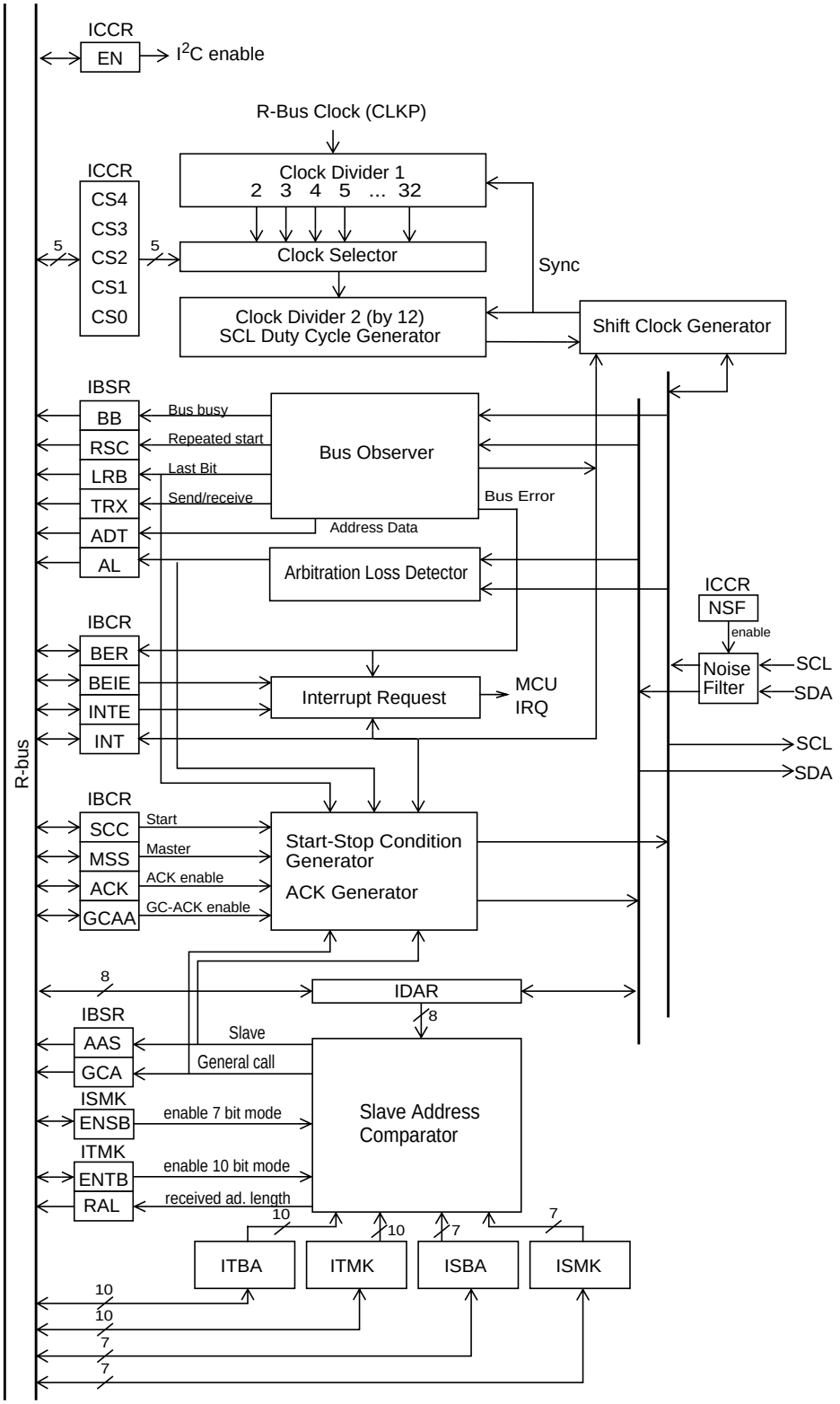
The I^2C interface is a serial I/O port supporting the Inter IC bus, operating as a master/slave device on the I^2C bus.

■ Features

- Master/slave transmitting and receiving functions
- Arbitration function
- Clock synchronization function
- General call addressing support
- Transfer direction detection function
- Repeated start condition generation and detection function
- Bus error detection function
- 7 bit addressing as master and slave
- 10 bit addressing as master and slave
- Possibility to give the interface a seven *and* a ten bit slave address
- Acknowledging upon slave address reception can be disabled (Master-only operation)
- Address masking to give interface several slave addresses (in 7 and 10 bit mode)
- Up to 400 kBit transfer rate
- Possibility to use built-in noise filters for SDA and SCL
- Can receive data at 400 kBit if R-Bus-Clock is higher than 6 MHz regardless of prescaler setting
- Can generate MCU interrupts on transmission and bus error events
- Supports being slowed down by a slave on bit and byte level

The I^2C interface does not support SCL clock stretching on bit level since it can receive the full 400 kBit datarate if the R-Bus-Clock (CLKP) is higher than 6 MHz regardless of the prescaler setting. However, clock stretching on byte level is performed since SCL is pulled low during an interrupt (INT='1' in IBCR2 register).

■ Block Diagram



2. I²C Interface Registers

This section describes the function of the I²C interface registers in detail.

■ Bus Control Register (IBCR0)

Bus control register

	15	14	13	12	11	10	9	8	⇐ Bit no.
Address : 0000D0H	BER	BEIE	SCC	MSS	ACK	GCAA	INTE	INT	IBCR0
Read/write ⇒	(R/W)	(R/W)	(W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	
Default value⇒	(0)	(0)	(0)	(0)	(0)	(0)	(0)	(0)	

■ Bus Status Register (IBSR0)

Bus status register

	7	6	5	4	3	2	1	0	⇐ Bit no.
Address : 0000D1H	BB	RSC	AL	LRB	TRX	AAS	GCA	ADT	IBSR0
Read/write ⇒	(R)	(R)	(R)	(R)	(R)	(R)	(R)	(R)	
Default value⇒	(0)	(0)	(0)	(0)	(0)	(0)	(0)	(0)	

■ Ten Bit slave Address register (ITBA0)

Ten Bit Address high byte

	15	14	13	12	11	10	9	8	⇐ Bit no.
Address : 0000D2H	---	---	---	---	---	---	TA9	TA8	ITBAH0
Read/write ⇒	(-)	(-)	(-)	(-)	(-)	(-)	(R/W)	(R/W)	
Default value⇒	(0)	(0)	(0)	(0)	(0)	(0)	(0)	(0)	

Ten Bit Address low byte

	7	6	5	4	3	2	1	0	⇐ Bit no.
Address : 0000D3H	TA7	TA6	TA5	TA4	TA3	TA2	TA1	TA0	ITBAL0
Read/write ⇒	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	
Default value⇒	(0)	(0)	(0)	(0)	(0)	(0)	(0)	(0)	

■ Ten bit slave address MasK register (ITMK0)

Ten Bit Address Mask high byte

	15	14	13	12	11	10	9	8	⇐ Bit no.
Address : 0000D4H	ENTB	RAL	---	---	---	---	TM9	TM8	ITMKH0
Read/write ⇒	(R/W)	(R)	(-)	(-)	(-)	(-)	(R/W)	(R/W)	
Default value⇒	(0)	(0)	(1)	(1)	(1)	(1)	(1)	(1)	

Ten Bit Address Mask low byte

	7	6	5	4	3	2	1	0	⇐ Bit no.
Address : 0000D5H	TM7	TM6	TM5	TM4	TM3	TM2	TM1	TM0	ITMKL0
Read/write ⇒	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	
Default value⇒	(1)	(1)	(1)	(1)	(1)	(1)	(1)	(1)	

■ Seven bit slave address MasK register (ISMK0)

Seven Bit Address Mask register	15	14	13	12	11	10	9	8	⇐ Bit no.
Address : 0000D6H	ENSB	SM6	SM5	SM4	SM3	SM2	SM1	SM0	ISMK0
Read/write ⇒	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	
Default value⇒	(0)	(1)	(1)	(1)	(1)	(1)	(1)	(1)	

■ Seven Bit slave Address register (ISBA0)

Seven Bit Address register	7	6	5	4	3	2	1	0	⇐ Bit no.
Address : 0000D7H	---	SA6	SA5	SA4	SA3	SA2	SA1	SA0	ISBA0
Read/write ⇒	(-)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	
Default value⇒	(0)	(0)	(0)	(0)	(0)	(0)	(0)	(0)	

■ Data Register (IDAR0)

Data register	7	6	5	4	3	2	1	0	⇐ Bit no.
Address : 0000D9H	D7	D6	D5	D4	D3	D2	D1	D0	IDAR0
Read/write ⇒	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	
Default value⇒	(0)	(0)	(0)	(0)	(0)	(0)	(0)	(0)	

■ Clock control register (ICCR0)

Clock Control register	15	14	13	12	11	10	9	8	⇐ Bit no.
Address : 0000DAH	---	NSF	EN	CS4	CS3	CS2	CS1	CS0	ICCR0
Read/write ⇒	(-)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	
Default value⇒	(0)	(0)	(0)	(1)	(1)	(1)	(1)	(1)	

2.1 Bus Control Register (IBCR0)

The bus control register (IBCR0) has the following functions:

- Interrupt enabling flags
- Interrupt generation flag
- Bus error detection flag
- Repeated start condition generation
- Master / slave mode selection
- General call acknowledge generation enabling
- Data byte acknowledge generation enabling

Write access to this register should only occur while the INT='1' or if a transfer is to be started. The user should not write to this register during an ongoing transfer since changes to the ACK or GCAA bits could result in bus errors. All bits in this register except the BER and the BEIE bit are cleared if the interface is not enabled (EN='0' in ICCR0).

Bus control register

	15	14	13	12	11	10	9	8	⇐ Bit no.
Address : 0000D0H	BER	BEIE	SCC	MSS	ACK	GCAA	INTE	INT	IBCR0
Read/write ⇒	(R/W)	(R/W)	(W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	
Default value⇒	(0)	(0)	(0)	(0)	(0)	(0)	(0)	(0)	

[bit 15] BER (Bus ERror)

This bit is the bus error interrupt flag. It is set by the hardware and cleared by the user. It always reads '1' in a Read-Modify-Write access.

(Write access)

0	Clear bus error interrupt flag.
1	No effect.

(Read access)

0	No bus error detected.
1	One of the error conditions described below detected.

When this bit is set, the EN bit in the ICCR0 register is cleared, the I^2C interface goes to pause status, data transfer is interrupted and all bits in the IBSR0 and the IBCR0 registers except BER, BEIE and INT are cleared. *The BER bit must be cleared before the interface may be reenabled.*

This bit is set to '1' if:

- start or stop conditions are detected at wrong places: during an address data transfer or during the transfer of the bits two to nine (acknowledge bit)
- a ten bit address header with read access is received before a ten bit write access
- a stop condition is detected while the interface is in master mode

The detection of the first two of the above conditions is enabled after the reception of the first stop condition to prevent false bus error reports if the interface is being enabled during an ongoing transfer.

[bit 14] BEIE (Bus Error Interrupt Enable)

This bit enables the bus error interrupt. It can only be changed by the user.

0	Bus error interrupt disabled.
1	Bus error interrupt enabled.

Setting this bit to ‘1’ enables MCU interrupt generation when the BER bit is set to ‘1’.

[bit 13] SCC (Start Condition Continue)

This bit is used to generate a repeated start condition. It is write only - it always reads ‘0’.

0	No effect.
1	Generate repeated start condition during master transfer.

A repeated start condition is generated if a ‘1’ is written to this bit while an interrupt in master mode (MSS=‘1’ and INT=‘1’) and the INT bit is cleared automatically.

[bit 12] MSS (Master Slave Select)

This is the master/slave mode selection bit. It can only be set by the user, but it can be cleared by the user and the hardware.

0	Go to slave mode.
1	Go to master mode, generate start condition and send address data byte in IDAR0 register.

It is cleared if an arbitration loss event occurs during master sending.

If a ‘0’ is written to it during a master interrupt (MSS=‘1’ and INT=‘1’), the INT bit is cleared automatically, a stop condition will be generated and the data transfer ends. Note that the MSS bit is reset immediately, the generation of the stop condition can be checked by polling the BB bit in the IBSR0 register.

If a ‘1’ is written to it while the bus is idle (MSS=‘0’ and BB=‘0’), a start condition is generated and the contents of the IDAR0 register (which should be address data) is sent.

If a ‘1’ is written to the MSS bit while the bus is in use (BB=‘1’ and TRX=‘0’ in IBSR2; MSS=‘0’ in IBCR0), the interface waits until the bus is free and then starts sending.

If the interface is addressed as slave with write access (data *reception*) in the meantime, it will start sending after the transfer ended and the bus is free again. If the interface is *sending* data as slave in the meantime (AAS=‘1’ and TRX=‘1’ in IBSR0), it will not start sending data if the bus is free again. It is important to check whether the interface was addressed as slave (AAS=‘1’ in IBSR0), sent the data byte successfully (MSS=‘1’ in IBCR0) or failed to send the data byte (AL=‘1’ in IBSR0) at the next interrupt!

[bit 11] ACK (ACKnowledge)

This is the acknowledge generation on *data* byte reception enable bit. It can only be changed by the user.

0	The interface will not acknowledge on data byte reception.
1	The interface will acknowledge on data byte reception.

This bit is not valid when receiving *address* bytes in slave mode - if the interface detects its 7 or 10 bit slave address, it will acknowledge if the corresponding enable bit (ENTB in ITMK0 or ENSB in ISMK0) is set.

Write access to this bit should occur during an interrupt (INT='1') or if the bus is idle (BB='0' in the IBSR0 register) write access to this bit is only possible if the interface is enabled (EN='1' in ICCR0) and if there is no bus error (BER='0' in IBCR0).

[bit 10] GCAA (General Call Address Acknowledge)

This bit enables acknowledge generation when a general call address is received. It can only be changed by the user.

0	The interface will not acknowledge on general call address byte reception.
1	The interface will acknowledge on general call address byte reception.

Write access to this bit should occur during an interrupt (INT='1') or if the bus is idle (BB='0' in IBSR0 register) write access to this bit is only possible if the interface is enabled (EN='1' in ICCR0) and if there is no bus error (BER='0' in IBCR0).

[bit 9] INTE (INTerrupt Enable)

This bit enables the MCU interrupt generation. It can only be changed by the user.

0	Interrupt disabled.
1	Interrupt enabled.

Setting this bit to '1' enables MCU interrupt generation when the INT bit is set to '1' (by the hardware).

[bit 8]: INT (INTerrupt)

This bit is the transfer end interrupt request flag. It is changed by the hardware and can be cleared by the user. It always reads '1' in a Read-Modify-Write access.

(Write access)

0	Clear transfer end interrupt request flag.
1	No effect.

(Read access)

0	Transfer not ended or not involved in current transfer or bus is idle.
1	Set *at the end of a 1-byte* data *transfer or reception* including the acknowledge bit under the following conditions: • Device is bus master. • Device is addressed as slave. • General call address received. • Arbitration loss occurred. Set *at the end of an address data reception* (after first byte if seven bit address received, after second byte if ten bit address received) including the acknowledge bit if the device is addressed as slave.

While this bit is ‘1’ the SCL line will hold an ‘L’ level signal. Writing ‘0’ to this bit clears the setting, releases the SCL line, and executes transfer of the next byte or a repeated start or stop condition is generated. Additionally, this bit is cleared if a ‘1’ is written to the SCC bit or the MSS bit is being cleared.

SCC, MSS And INT Bit Competition

Simultaneously writing to the SCC, MSS and INT bits causes a competition to transfer the next byte, to generate a repeated start condition or to generate a stop condition. In these cases the order of priority is as follows:

Next byte transfer and stop condition generation.
When ‘0’ is written to the INT bit and ‘0’ is written to the MSS bit, the MSS bit takes priority and a stop condition is generated.

Next byte transfer and start condition generation.
When ‘0’ is written to the INT bit and ‘1’ is written to the SCC bit, the SCC bit takes priority. A repeated start condition is generated and the content of the IDAR0 register is sent.

Repeated start condition generation and stop condition generation.
When ‘1’ is written to the SCC bit and ‘0’ to the MSS bit, the MSS bit clearing takes priority. A stop condition is generated and the interface enters slave mode.

2.2 Bus Status Register (IBSR0)

The bus status register (IBSR0) has the following functions:

- Bus busy detection
- Repeated start condition detection
- Arbitration loss detection
- Acknowledge detection
- Data transfer direction indication
- Addressing as slave detection
- General call address detection
- Address data transfer detection

This register is read-only, all bits are controlled by the hardware. All bits are cleared if the interface is not enabled (EN='0' in ICCR0).

Bus status register	7	6	5	4	3	2	1	0	⇐ Bit no.
Address : 0000D1H	BB	RSC	AL	LRB	TRX	AAS	GCA	ADT	IBSR0
Read/write ⇒	(R)	(R)	(R)	(R)	(R)	(R)	(R)	(R)	
Default value⇒	(0)	(0)	(0)	(0)	(0)	(0)	(0)	(0)	

[bit 7] BB (Bus Busy)

This bit indicates the status of the I^2C bus.

0	Stop condition detected (bus idle).
1	Start condition detected (bus in use).

This bit is set to '1' if a start condition is detected. It is reset upon a stop condition.

[bit 6] RSC (Repeated Start Condition)

This bit indicates detection of a repeated start condition.

0	Repeated start condition not detected.
1	Bus in use, repeated start condition detected.

This bit is cleared at the end of an address data transfer (ADT='0') or detection of a stop condition.

[bit 5] AL (Arbitration Loss)

This bit indicates an arbitration loss.

0	No arbitration loss detected.
1	Arbitration loss occurred during master sending.

This bit is cleared by writing '0' to the INT bit or by writing '1' to the MSS bit in the IBCR0 register.
An arbitration loss occurs if:

- the data sent does not match the data read on the SDA line at the rising SCL edge

- a repeated start condition is generated by another master in the first bit of a data byte
- the interface could not generate a start or stop condition because another slave pulled the SCL line low before

[bit 4] LRB (Last Received Bit)

This bit is used to store the acknowledge message from the receiving side at the transmitter side.

0	Receiver acknowledged.
1	Receiver did not acknowledge.

It is changed by the hardware upon reception of bit 9 (acknowledge bit) and is also cleared by a start or stop condition.

[bit 3] TRX (Transferring data)

This bit indicates data sending operation during data transfer.

0	Not transmitting data.
1	Transmitting data.

It is set to ‘1’:
- if a start condition was generated in master mode at the end of a first byte transfer and read access as slave or sending data as master

It is set to ‘0’ if:
- the bus is idle (BB=‘0’ in IBCR0)
- an arbitration loss occured
- a ‘1’ is written to the SCC bit during master interrupt (MSS=‘1’ and INT=‘1’)
- the MSS bit is cleared during master interrupt (MSS=‘1’ and INT=‘1’)
- the interface is in slave mode and the last transferred byte was not acknowledged
- the interface is in slave mode and it is receiving data
- the interface is in master mode and is reading data from a slave

[bit 2] AAS (Addressed As Slave)

This bit indicates detection of a slave addressing.

0	Not addressed as slave.
1	Addressed as slave.

This bit is cleared by a (repeated-) start or stop condition. It is set if the interface detects its seven and/or ten bit slave address.

[bit 1] GCA (General Call Address)

This bit indicates detection of a general call address (0x00).

0	General call address not received as slave.

1	General call address received as slave.

This bit is cleared by a (repeated-) start or stop condition.

[bit 0] ADT (Address Data Transfer)

This bit indicates the detection of an address data transfer.

0	Incoming data is not address data (or bus is not in use).
1	Incoming data is address data.

This bit is set to ‘1’ by a start condition. It is cleared after the second byte if a ten bit slave address header with write access is detected, else it is cleared after the first byte.

"After" the first/second byte means:

- a ‘0’ is written to the MSS bit during a master interrupt (MSS=‘1’ and INT=‘1’ in IBCR0)
- a ‘1’ is written to the SCC bit during a master interrupt (MSS=‘1’ and INT=‘1’ in IBCR0)
- the INT bit is being cleared
- the beginning of every byte transfer if the interface is not involved in the current transfer as master or slave

2.3 Ten Bit Slave Address Register (ITBA0)

This register (ITBAH0 / ITBAL0) designates the ten bit slave address.

Write access to this register is only possible if the interface is disabled (EN='0' in ICCR0).

Ten Bit Address high byte

	15	14	13	12	11	10	9	8	⇐ Bit no.
Address : 0000D2H	---	---	---	---	---	---	TA9	TA8	ITBAH0
Read/write ⇒	(-)	(-)	(-)	(-)	(-)	(-)	(R/W)	(R/W)	
Default value⇒	(0)	(0)	(0)	(0)	(0)	(0)	(0)	(0)	

Ten Bit Address low byte

	7	6	5	4	3	2	1	0	⇐ Bit no.
Address : 0000D3H	TA7	TA6	TA5	TA4	TA3	TA2	TA1	TA0	ITBAL0
Read/write ⇒	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	
Default value⇒	(0)	(0)	(0)	(0)	(0)	(0)	(0)	(0)	

[bit 15] - [bit 10] Not used.

These bits always read '0'.

[bit 9] - [bit 0] TBA - Ten Bit slave Address (TA9-TA0)

When address data is received in slave mode, it is compared to the ITBA0 register if the ten bit address is enabled (ENTB='1' in the ITMK0 register). An acknowledge is sent to the master after reception of a ten bit address header with write access[1]. Then, the second incoming byte is compared to the ITBA0 register. If a match is detected, an acknowledge signal is sent to the master device and the AAS bit is set.

Additionally, the interface acknowledges upon the the reception of a ten bit header with read access[2] after a *repeated* start conditon.

All bits of the slave address may be masked using the ITMK0 register. The received ten bit slave address is written back to the ITBA0 register, it is only valid while the AAS bit in the IBSR0 register is '1'.

1. Note: a ten bit header (write access) consists of the following bit sequence: 11110, TA9, TA8, 0.
2. Note: a ten bit header (read access) consists of the following bit sequence: 11110, TA9, TA8, 1.

2.4 Ten Bit Address Mask Register (ITMK0)

This register contains the ten bit slave address mask and the ten bit slave address enable bit.

Ten Bit Address Mask high byte	15	14	13	12	11	10	9	8	⇐ Bit no.
Address : 0000D4H	ENTB	RAL	---	---	---	---	TM9	TM8	ITMKH0
Read/write ⇒	(R/W)	(R)	(-)	(-)	(-)	(-)	(R/W)	(R/W)	
Default value⇒	(0)	(0)	(1)	(1)	(1)	(1)	(1)	(1)	

Ten Bit Address Mask low byte	7	6	5	4	3	2	1	0	⇐ Bit no.
Address : 0000D5H	TM7	TM6	TM5	TM4	TM3	TM2	TM1	TM0	ITMKL0
Read/write ⇒	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	
Default value⇒	(1)	(1)	(1)	(1)	(1)	(1)	(1)	(1)	

[bit 15] ENTB - EnaBle Ten Bit slave address

This bit enables the ten bit slave address (and the acknowleding upon its reception). Write access to this bit is only possible if the interface is disabled (EN='0' in ICCR0).

0	Ten bit slave address disabled.
1	Ten bit slave address enabled.

[bit 14] RAL - Received slave Address Length

This bit indicates whether the interface was addressed as a seven or ten bit slave. It is read-only.

0	Addressed as seven bit slave.
1	Addressed as ten bit slave.

This bit can be used to determine whether the interface was addressed as a seven or ten bit slave if both slave addresses are enabled (ENTB='1' and ENSB='1'). Its contents is only valid if the AAS bit in the IBSR0 register is '1'. This bit is also reset if the interface is disabled (EN='0' in ICCR0).

[bit 13] - [bit 10] Not used.

These bits always read '1'.

[bit 9] - [bit 0] TMK - Ten bit slave address MasK (TM9-TM0).

This register is used to mask the ten bit slave address of the interface. Write access to these bits is only possible if the interface is disabled (EN='0' in ICCR0).

0	Bit is not used in slave address comparision.
1	Bit is used in slave address comparision.

This can be used to make the interface acknowledge on multiple ten bit slave addresses. Only the bits set to '1' in this register are used in the ten bit slave address comparision. The received slave address is written back to the ITBA0 register and thus may be determined by reading the ITBA0 register if the AAS bit in the

IBSR0 register is ‘1’.

Note: If the address mask is changed after the interface had been enabled, the slave address should also be set again since it could have been overwritten by a previously received slave address.

2.5 Seven Bit Slave Address Register (ISBA0)

This register designates the seven bit slave address.

Write access to this register is only possible if the interface is disabled (EN='0' in ICCR0).

Seven Bit Address register	7	6	5	4	3	2	1	0	⇐ Bit no.
Address : 0000D7H	---	SA6	SA5	SA4	SA3	SA2	SA1	SA0	ISBA0
Read/write ⇒	(-)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	
Default value⇒	(0)	(0)	(0)	(0)	(0)	(0)	(0)	(0)	

[bit 7] Not used.

This bit always reads '0'.

[bit 6] - [bit 0] Seven Bit slave Address (SA6-SA0)

When address data is received in slave mode, it is compared to the ISBA0 register if the seven bit address is enabled (ENSB='1' in the ISMK0 register). If a match is detected, an acknowledge signal is sent to the master device and the AAS bit is set.

All bits of the slave address may be masked using the ISMK0 register. The received seven bit slave address is written back to the ISBA0 register, it is only valid while the AAS bit in the IBSR0 register is '1'.

The interface does not compare the contents of this register to the incoming data if a ten bit header or a general call is received.

2.6 Seven Bit Slave Address Mask Register (ISMK0)

This register contains the seven bit slave address mask and the seven bit mode enable bit. Write access to this register is only possible if the interface is disabled (EN='0' in ICCR0).

Seven Bit Address Mask register	15	14	13	12	11	10	9	8	⇐ Bit no.
Address : 0000D6H	ENSB	SM6	SM5	SM4	SM3	SM2	SM1	SM0	ISMK0
Read/write ⇒	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	
Default value⇒	(0)	(1)	(1)	(1)	(1)	(1)	(1)	(1)	

[bit 15] ENSB - EnaBle Seven Bit slave address

This bit enables the seven bit slave address (and the acknowleding upon its reception).

0	Seven bit slave address disabled.
1	Seven bit slave address enabled.

[bit 14] - [bit 8] SMK - Seven bit slave address MasK (SM6-SM0)

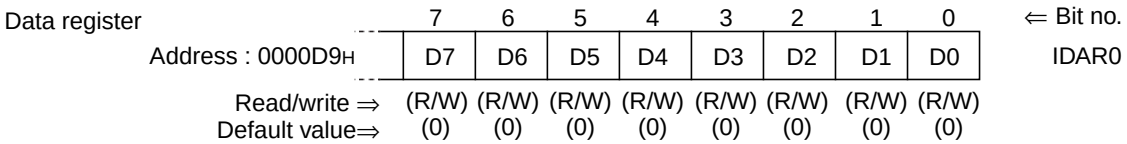
This register is used to mask the seven bit slave address of the interface.

0	Bit is not used in slave address comparision.
1	Bit is used in slave address comparision.

This can be used to make the interface acknowledge on multiple seven bit slave addresses. Only the bits set to '1' in this register are used in the seven bit slave address comparision. The received slave address is written back to the ISBA0 register and thus may be determined by reading the ISBA0 register if the AAS bit in the IBSR0 register is '1'.

Note: If the address mask is changed after the interface had been enabled, the slave address should also be set again since it could have been overwritten by a previously received slave address.

2.7 Data Register (IDAR0)



[bit 15] - [bit 8] Not used.

These bits always read ‘0’.

[bit 7] - [bit 0] Data bits (D7-D0)

The data register is used in serial data transfer, and transfers data MSB-first. This register is double buffered on the write side, so that when the bus is in use (BB=‘1’), write data can be loaded to the register for serial transfer. The data byte is loaded into the internal transfer register if the INT bit in the IBCR0 register is being cleared or the bus is idle (BB=‘0’ in IBSR0). In a read access, the internal register is read directly, therefore *received data values in this register are only valid if INT=‘1’ in the IBCR2 register*.

2.8 Clock Control Register (ICCR0)

The clock control register (ICCR0) has the following functions:

- Enable IO pad noise filters
- Enable I^2C interface operation
- Setting the serial clock frequency

Clock Control register	15	14	13	12	11	10	9	8	⇐ Bit no.
Address : 0000DAH	---	NSF	EN	CS4	CS3	CS2	CS1	CS0	ICCR0
Read/write ⇒	(-)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	
Default value⇒	(0)	(0)	(0)	(1)	(1)	(1)	(1)	(1)	

[bit 15] Not used.

This bit always reads ‘0’.

[bit 14] IO pad NoiSe Filter enable.

This bit enables the noise filters built into the SDA and SCL IO pads.
The noise filter will suppress single spikes with a pulse width of 0 ns (minimum) and between 1 and 1.5 cycles of R-bus (maximum). The maximum depends on the phase relationship between I^2C signals (SDA, SCL) and R-bus clock.
It should be set to ‘1’ if the interface is transmitting or receiving at datarates above 100 kBit.

[bit 13] EN (ENable)

This bit enables the I^2C interface operation. It can only be set by the user but may be cleared by the user and the hardware.

0	Interface disabled.
1	Interface enabled.

When this bit is set to ‘0’ all bits in the IBSR0 register and IBCR0 register (except the BER and BEIE bits) are cleared, the module is disabled and the I^2C lines are left open. It is cleared by the hardware if a bus error occurs (BER=‘1’ in IBCR0).
Warning: The interface *immediately* stops transmitting or receiving if is it is being disabled. This might leave the I^2C bus in an undesired state!

[bit 12] - [bit 8] CS4-0 (Clock preScaler)

These bits select the serial bitrate. They can only be changed if the interface is disabled (EN=‘0’) or the EN bit is being cleared in the same write access.
It is determined by the following formula:

$$\text{Bitrate} = \frac{\phi}{n*12 + 18}$$

Noise filter disabled
n>0; ϕ : R-Bus clock CLKP (set by DIVR0 register)

$$\text{Bitrate} = \frac{\phi}{n*12 + 19\ (+1)}$$

Noise filter enabled
n>0; ϕ : R-Bus clock CLKP (set by DIVR0 register)
(+1): Unaccurancy caused by noise filter operation

(Note) Because of the noise filter (depending on relationship between external signal and internal clock it will cause different delays) the divider in the second formula can vary between (12n + 19) and (12n + 20).

■ Prescaler settings:

Table 2-1 I2C Prescaler Settings

n	CS4	CS3	CS2	CS1	CS0
1	0	0	0	0	1
2	0	0	0	1	0
3	0	0	0	1	1
...					
31	1	1	1	1	1

Do not use n=0 prescaler setting, it violates SDA/SCL timings!

The table below shows SCL frequency measurement results for the most common R-bus clock settings and the recommended related pre-scaler settings for 100 Kbit and 400 Kbit operation.

R-Bus Clock (CLKP) [MHz]	100 kBit (Noise filter disabled) n	Bitrate [kBit]	400 kBit (Noise filter enabled) n	Bitrate [kBit]
32			5	387.5
24	19	97.5	4	352.5
16	12	98	2	372
8	6	89	1	266.5

It should be noted that the measured values have been determined by examining the last 8 cycles of a transfer. This was done because the first cycle of all address or data transfers is longer than the other cycles. To be more precise: In case of an address transfer this first cycle is 3 prescaler periods longer than the other cycles, in case of a data transfer it is 4 prescaler periods longer (see figure below).

■ SCL Waveforms

Figure 2-1 SCL Waveforms

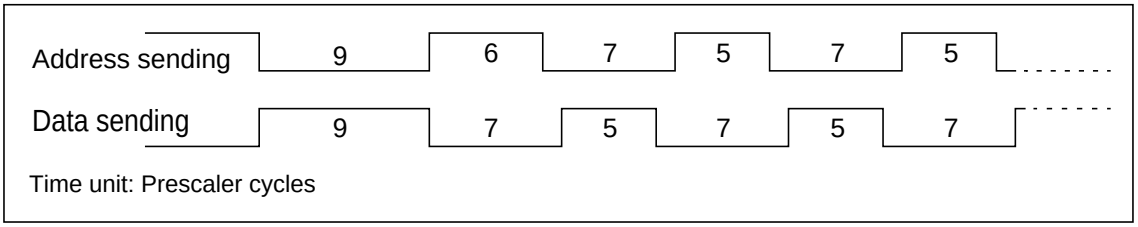


Figure 2-1 shows the SCL waveform for sending of address and data bits. The timings given in the figure are prescaler periods (e.g. ‘9’ means 9 times the prescaler count based on the R-Bus clock). The timings in the figure are only valid if no other device on the I2C bus influences the SCL timing.

3. I²C Interface Operation

The I²C bus executes communication using two bi-directional bus lines, the serial data line (SDA) and serial clock line (SCL). The I²C interface has two open-drain I/O pins (SDA/SCL) corresponding to these lines, enabling wired logic applications.

■ Start Conditions

When the bus is free (BB='0' in IBSR0, MSS='0' in IBCR2), writing '1' to the MSS bit places the I²C interface in master mode and generates a start condition.
If a '1' is written to it while the bus is idle (MSS='0' and BB='0'), a start condition is generated and the contents of the IDAR0 register (which should be address data) is sent.
Repeated start conditions can be generated by writing '1' to the SCC bit when in bus master mode and interrupt status (MSS='1' and INT='1' in IBCR0).

If a '1' is written to the MSS bit while the bus is in use (BB='1' and TRX='0' in IBSR0; MSS='0'and INT='0'in IBCR0), the interface waits until the bus is free and then starts sending.
If the interface is addressed as slave with write access (data reception) in the meantime, it will start sending after the transfer ended and the bus is free again. If the interface is sending data as slave in the meantime, it will not start sending data if the bus of free again. It is important to check whether the interface was addressed as slave (MSS='0' in IBCR0 and AAS='1' in IBSR0), sent the data byte successfully (MSS='1' in IBCR0) or failed to send the data byte (AL='1' in IBSR0) at the next interrupt!

Writing '1' to the MSS bit or SCC bit in any other situation has no significance.

■ Stop Conditions

Writing '0' to the MSS bit in master mode (MSS='1' and INT='1' in IBCR0) generates a stop condition and places the device in slave mode. Writing '0' to the MSS bit in any other situation has no significance.
After clearing the MSS bit, the interface tries to generate a stop condition which might fail if another master pulls the SCL line low before the stop condition has been generated. *This will generate an interrupt after the next byte has been transferred!*

■ Slave Address Detection

In slave mode, after a start condition is generated the BB is set to '1' and data sent from the master device is received into the IDAR0 register.
After the reception of eight bits, the contents of the IDAR2 register is compared to the ISBA register using the bit mask stored in ISMK0 if the ENSB bit in the ISMK0 register is '1'. If a match results, the AAS bit is set to '1' and an acknowledge signal is sent to the master. Then bit 0 of the received data (bit 0 of the IDAR0 register) is inverted and stored in the TRX bit.
If the ENTB bit in the ITMK0 register is '1' and a ten bit address header (11110, TA1, TA0, write access) is detected, the interface sends an acknowledge signal to the master and stores the inverted last data bit in the TRX register. No interrupt is generated. Then, the next transferred byte is compared (using the bit mask stored in ITMK0) to the lower byte of the ITBA0 register. If a match is found, an acknowledge signal is sent to the master, the AAS bit is set and an interrupt is generated.
If the interface was addressed as slave and detects a repeated start condition, the AAS bit is set after reception of the ten bit address header (11110, TA1, TA0, read access) and an interrupt is generated.
Since there are seperate registers for the ten and seven bit address and their bitmasks, it is possible to make the interface acknowledge on both addresses by setting the ENSB (in ISMK0) and ENTB (in ITMK0) bits. The received slave address length (seven or ten bit) may be determined by reading the RAL bit in the ITMK0 register (this bit is valid if the AAS bit is set only).
It is also possible to give the interface no slave address by setting both bits to '0' if it is only used as a master.
All slave address bits may be masked with their corresponding mask register (ITMK0 or ISMK0).

■ Slave Address Masking

Only the bits set to ‘1’ in the mask registers (ITMK0 / ISMK0) are used for address comparision, all other bits are ignored. The received slave address can be read from the ITBA0 (if ten bit address received, RAL=‘1’) or ISBA0 (if seven bit address received, RAL=‘0’) register if the AAS bit in the IBSR0 register is ‘1’.
If the bitmasks are cleared, the interface can be used as a bus monitor since it will always be addressed as slave. Note that this is not a real bus monitor because it *acknowledges* upon any slave address reception, even if there is no other slave listening.

■ Addressing Slaves

In master mode, after a start condition is generated the BB and TRX bits are set to ‘1’ and the contents of the IDAR0 register is sent in MSB first order. After address data is sent and an acknowledge signal was received from the slave device, bit 0 of the sent data (bit 0 of the IDAR0 register after sending) is inverted and stored in the TRX bit. Acknowledgement by the slave may be checked using the LRB bit in the IBSR0 register. This procedure also applies to a repeated start condition.
In order to address a ten bit slave for write access, two bytes have to be sent. The first one is the ten bit address header which consists of the bitsequence ‘1 1 1 1 0 A9 A8 0’, it is followed by the second byte containing the lower eight bits of the ten bit slave address (A7 - A0).
A ten bit slave is accessed for reading by sending the above byte sequence and generating a repeated start condition (SCC bit in IBCR0) followed by a ten bit address header with read access (1 1 1 1 0 A9 A8 1).
Summary of the address data bytes:

7 bit slave, write access: Start condition - A6 A5 A4 A3 A2 A1 A0 0.

7 bit slave, read access: Start condition - A6 A5 A4 A3 A2 A1 A0 1.

10 bit slave, write access: Start condition - 1 1 1 1 0 A9 A8 0 - A7 A6 A5 A4 A3 A2 A1 A0.

10 bit slave, read access: Start condition - 1 1 1 1 0 A9 A8 1 - A7 A6 A5 A4 A3 A2 A1 A0 - repeated start - 1 1 1 1 0 A9 A8 1.

■ Arbitration

During sending in master mode, if another master device is sending data at the same time, arbitration is performed. If a device is sending the data value ‘1’ and the data on the SDA line has an ‘L’ level value, the device is considered to have lost arbitration, and the AL bit is set to ‘1.’ Also, the AL bit is set to ‘1’ if a start conditon is detected at the first bit of a data byte but the interface did not want to generate one or the generation of a start or stop condition failed by some reason.
Arbitration loss detection clears both the MSS and TRX bit and immediately places the device in slave mode so it is able to acknowledge if its own slave address is being sent.

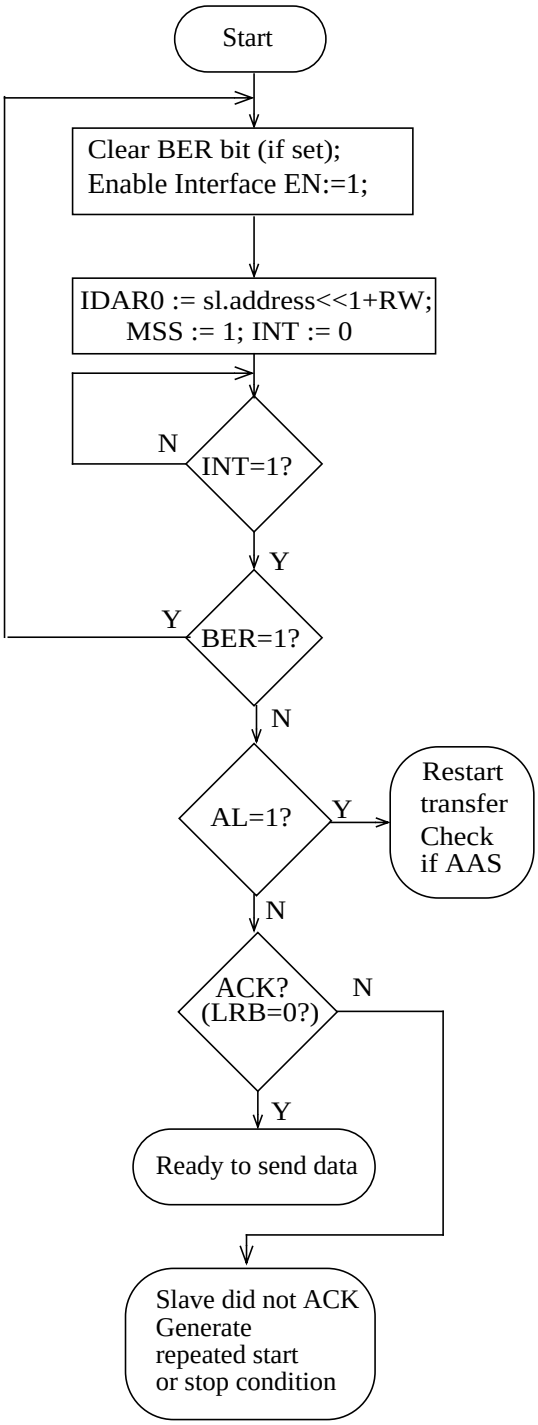
■ Acknowledgement

Acknowledge bits are sent from the receiver to the transmitter. The ACK bit in the IBCR0 register can be used to select whether to send an acknowledgment when data bytes are received.
When data is send in slave mode (read access from another master), if no acknowledgement is received from the master, the TRX bit is set to ‘0’ and the device goes to receiving mode. This enables the master to generate a stop condition as soon as the slave has released the SCL line.
In master mode, acknowledgement by the slave may be checked by reading the LRB bit in the IBSR0 register.

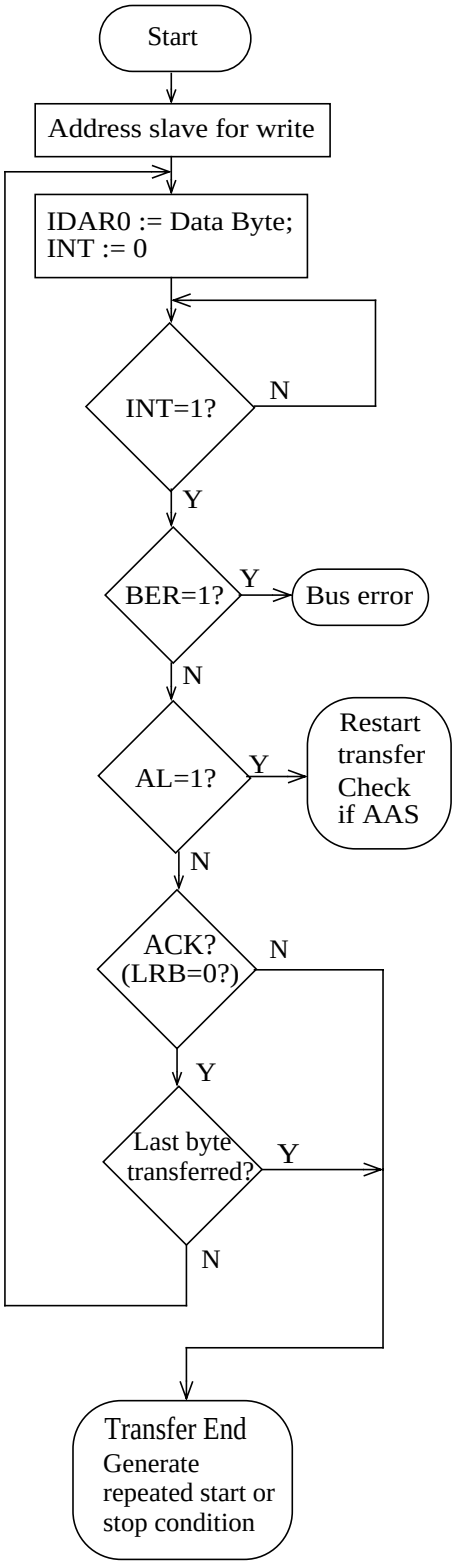
4. Programming Flow Charts

■ Example Of Slave Addressing And Sending Data

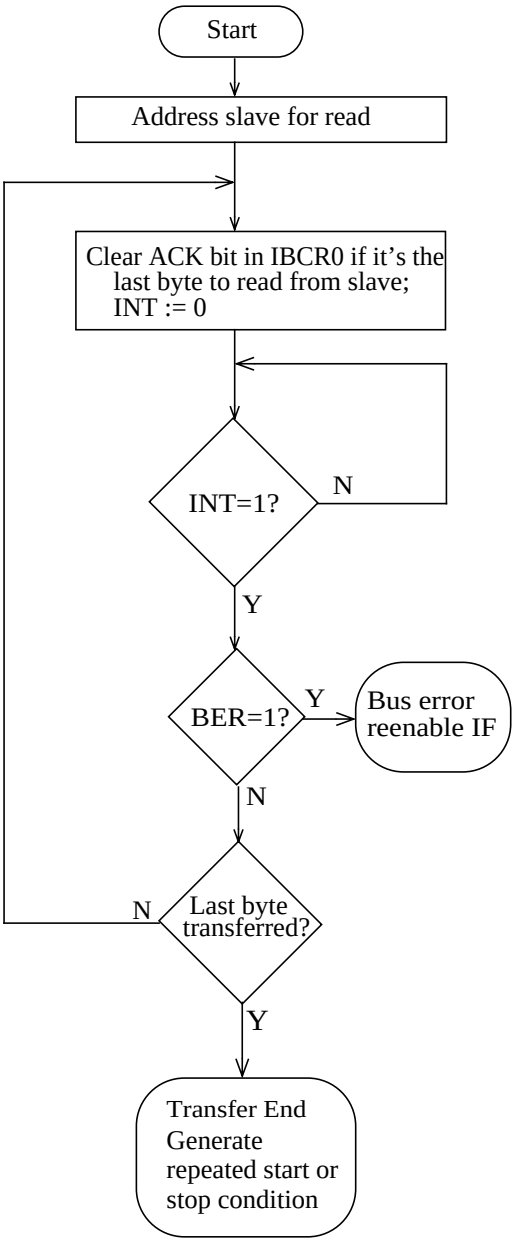
Addressing a 7 bit slave



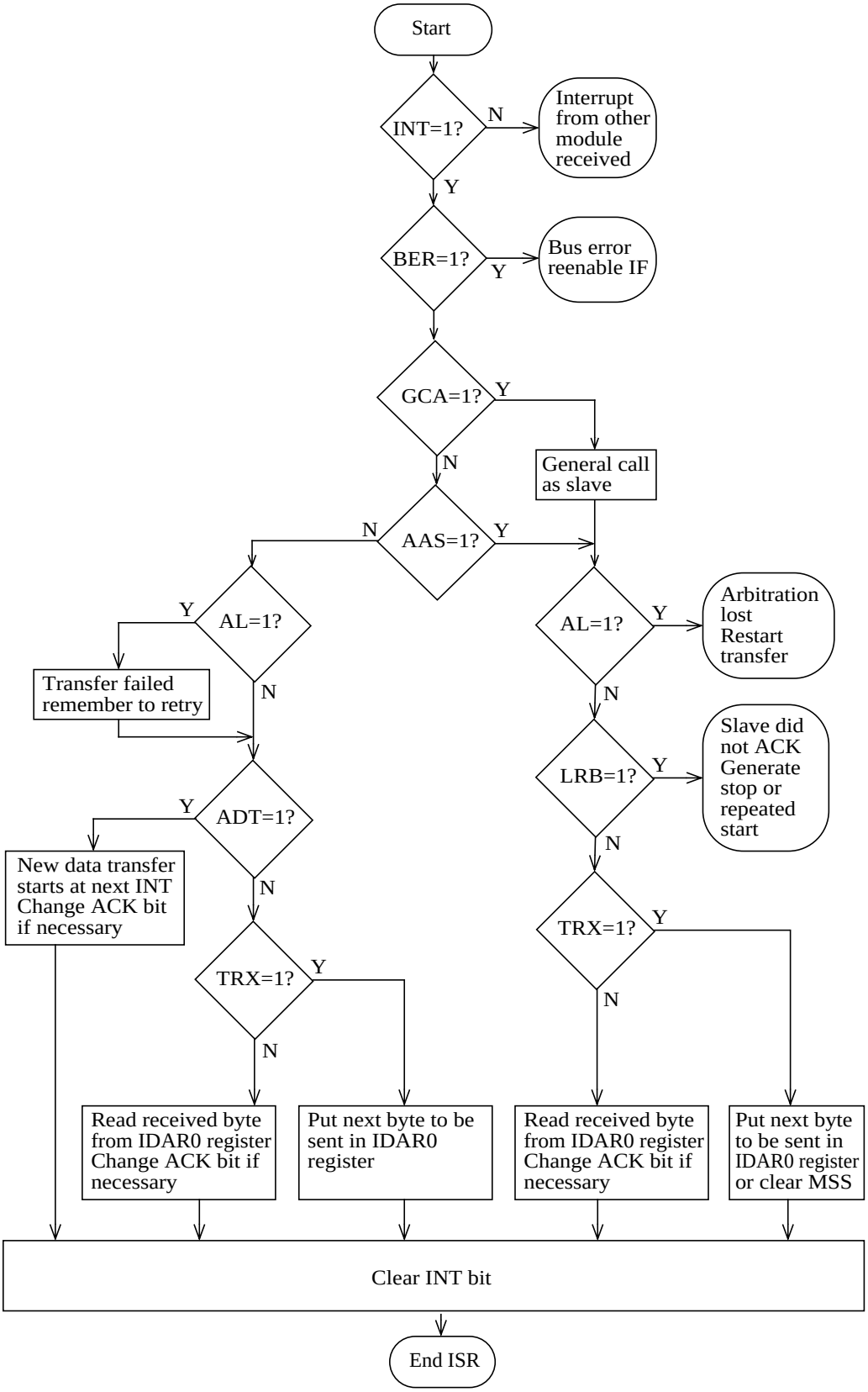
Sending data



■ Example Of Receiving Data



■ Example Of An Interrupt Handler



4.Programming Flow Charts

Chapter 34 CAN Controller

1. Overview

The CAN performs communication according to the CAN protocol version 2.0 part A and B. The bit rate can be programmed to values up to 1MBit/s. For the connection to the physical layer additional transceiver hardware is required.

■ The CAN implements the following features:

- Supports CAN protocol version 2.0 part A and B
- Bit rates up to 1 MBit/s
- 32 (up to 128) Message Objects
- Each Message Object has its own identifier mask
- Programmable FIFO mode (concatenation of Message Objects)
- Maskable interrupt
- **D**isabled **A**utomatic **R**etransmission mode for Time Triggered CAN applications
- Programmable loop-back mode for self-test operation

■ This chapter uses the following terms and abbreviations.

Term	Meaning
CAN	Controller Area Network
BSP	Bit Stream Processor
BTL	Bit Timing Logic
CRC	Cyclic Redundancy Check
DLC	Data Length Code
EML	Error Management Logic
FSM	Finite State Machine
TTCAN	Time Triggered CAN

2. Register Description

This section lists the CAN registers and describes the function of each register in detail.

2.1 Programmer's Model

The CAN module allocates an address space of 256 bytes (64 words). The CAN registers can be accessed from the CPU in byte, halfword and word.

The two sets of interface registers (IF1 and IF2) control the CPU access to the Message RAM. They buffer the data to be transferred to and from the RAM, avoiding conflicts between CPU accesses and message reception/transmission.

The data registers (IF1 Data and IF2 Data) are doubled in the address map, ordered both in little endian byte and big endian byte.

If several CAN modules are present on a device then they are located linear in the address space with a constant offset of 256 bytes (64 words). The base address of each CAN module is given by the following table:

- Base-address of CAN0 :0x00C000
- Base-address of CAN1 :0x00C100
- Base-address of CAN2 :0x00C200
- Base-address of CAN3 :0x00C300
- Base-address of CAN4 :0x00C400
- Base-address of CAN5 :0x00C500

Address	Register				Note
	+0	+1	+2	+3	
Base-addr + 0x00	**Control Register**		**Status Register**		
	bit[15:8]	bit[7:0]	bit[15:8]	bit[7:0]	
	reserved	see descr. CTRLR	reserved	see descr. STATR	
	Reset: 0x00	Reset: 0x01	Reset: 0x00	Reset: 0x00	
Base-addr + 0x04	**Error Counter**		**Bit Timing Register**		Error Counter is read only. Bit Timing Register is write enabled by **CCE**
	bit[15:8]	bit[7:0]	bit[15:8]	bit[7:0]	
	RP,REC[6:0]	TEC[7:0]	TSeg2[2:0],TSeg1[3:0]	SJW[1:0],BRP[5:0]	
	Reset: 0x00	Reset: 0x00	Reset: 0x23	Reset: 0x01	
Base-addr + 0x08	**Interrupt Register**		**Test Register**		Interrupt Register is read only. Test Register is write enabled by **Test**. **r** signifies the actual value of the CAN_RX pin.
	bit[15:8]	bit[7:0]	bit[15:8]	bit[7:0]	
	Int-Id[15:8]	Int-Id[7:0]	reserved	see descr. TESTR	
	Reset: 0x00	Reset: 0x00	Reset: 0x00	Reset: 0x00 & 0b**r**0000000	
Base-addr + 0x0C	**BRP Extension Register**		**Reserved**		BRP Extension Register is write enabled by **CCE**.
	bit[15:8]	bit[7:0]	bit[15:8]	bit[7:0]	
	reserved	BRP[3:0]	reserved	reserved	
	Reset: 0x00	Reset: 0x00	Reset: 0x00	Reset: 0x00	

Address	Register				Note
	+0	+1	+2	+3	
Base-addr + 0x10	**IF1 Command Request**		**IF1 Command Mask**		
	bit[15:8]	bit[7:0]	bit[15:8]	bit[7:0]	
	Busy	Mess. No. [5:0]	reserved	see descr. IF1CMSK	
	Reset: 0x00	Reset: 0x01	Reset: 0x00	Reset: 0x00	
Base-addr + 0x14	**IF1 Mask 2**		**IF1 Mask 1**		
	bit[15:8]	bit[7:0]	bit[15:8]	bit[7:0]	
	MXtd,MDir,Msk[28:24]	Msk[23:16]	Msk[15:8]	Msk[7:0]	
	Reset: 0xFF	Reset: 0xFF	Reset: 0xFF	Reset: 0xFF	
Base-addr + 0x18	**IF1 Arbitration 2**		**IF1 Arbitration 1**		
	bit[15:8]	bit[7:0]	bit[15:8]	bit[7:0]	
	MsgVal,Xtd,Dir,ID[28:24]	ID[23:16]	ID[15:8]	ID[7:0]	
	Reset: 0x00	Reset: 0x00	Reset: 0x00	Reset: 0x00	
Base-addr + 0x1C	**IF1 Message Control**		**Reserved**		
	bit[15:8]	bit[7:0]	bit[15:8]	bit[7:0]	
	see descr. IF1MCTR	see descr. IF1MCTR	reserved	reserved	
	Reset: 0x00	Reset: 0x00	Reset: 0x00	Reset: 0x00	
Base-addr + 0x20	**IF1 Data A1**		**IF1 Data A2**		Big Endian byte ordering.
	bit[7:0]	bit[15:8]	bit[7:0]	bit[15:8]	
	Data[0]	Data[1]	Data[2]	Data[3]	
	Reset: 0x00	Reset: 0x00	Reset: 0x00	Reset: 0x00	
Base-addr + 0x24	**IF1 Data B1**		**IF1 Data B2**		Big Endian byte ordering.
	bit[7:0]	bit[15:8]	bit[7:0]	bit[15:8]	
	Data[4]	Data[5]	Data[6]	Data[7]	
	Reset: 0x00	Reset: 0x00	Reset: 0x00	Reset: 0x00	
Base-addr + 0x30	**IF1 Data A2**		**IF1 Data A1**		Little Endian byte ordering.
	bit[15:8]	bit[7:0]	bit[15:8]	bit[7:0]	
	Data[3]	Data[2]	Data[1]	Data[0]	
	Reset: 0x00	Reset: 0x00	Reset: 0x00	Reset: 0x00	
Base-addr + 0x34	**IF1 Data B2**		**IF1 Data B1**		Little Endian byte ordering.
	bit[15:8]	bit[7:0]	bit[15:8]	bit[7:0]	
	Data[7]	Data[6]	Data[5]	Data[4]	
	Reset: 0x00	Reset: 0x00	Reset: 0x00	Reset: 0x00	

Address	Register				Note
	+0	+1	+2	+3	
Base-addr + 0x40	**IF2 Command Request**		**IF2 Command Mask**		
	bit[15:8]	bit[7:0]	bit[15:8]	bit[7:0]	
	Busy	Mess. No. [5:0]	reserved	see descr. IF2CMSK	
	Reset: 0x00	Reset: 0x01	Reset: 0x00	Reset: 0x00	
Base-addr + 0x44	**IF2 Mask 2**		**IF2 Mask 1**		
	bit[15:8]	bit[7:0]	bit[15:8]	bit[7:0]	
	MXtd,MDir,Msk[28:24]	Msk[23:16]	Msk[15:8]	Msk[7:0]	
	Reset: 0xFF	Reset: 0xFF	Reset: 0xFF	Reset: 0xFF	
Base-addr + 0x48	**IF2 Arbitration 2**		**IF2 Arbitration 1**		
	bit[15:8]	bit[7:0]	bit[15:8]	bit[7:0]	
	MsgVal,Xtd,Dir,ID[28:24]	ID[23:16]	ID[15:8]	ID[7:0]	
	Reset: 0x00	Reset: 0x00	Reset: 0x00	Reset: 0x00	
Base-addr + 0x4C	**IF2 Message Control**		**Reserved**		
	bit[15:8]	bit[7:0]	bit[7:0]	bit[15:8]	
	see descr. IF2MCTR	see descr. IF2MCTR	reserved	reserved	
	Reset: 0x00	Reset: 0x00	Reset: 0x00	Reset: 0x00	
Base-addr + 0x50	**IF2 Data A1**		**IF2 Data A2**		Big Endian byte ordering.
	bit[7:0]	bit[15:8]	bit[7:0]	bit[15:8]	
	Data[0]	Data[1]	Data[2]	Data[3]	
	Reset: 0x00	Reset: 0x00	Reset: 0x00	Reset: 0x00	
Base-addr + 0x54	**IF2 Data B1**		**IF2 Data B2**		Big Endian byte ordering.
	bit[7:0]	bit[15:8]	bit[7:0]	bit[15:8]	
	Data[4]	Data[5]	Data[6]	Data[7]	
	Reset: 0x00	Reset: 0x00	Reset: 0x00	Reset: 0x00	
Base-addr + 0x60	**IF2 Data A2**		**IF2 Data A1**		Little Endian byte ordering.
	bit[15:8]	bit[7:0]	bit[15:8]	bit[7:0]	
	Data[3]	Data[2]	Data[1]	Data[0]	
	Reset: 0x00	Reset: 0x00	Reset: 0x00	Reset: 0x00	
Base-addr + 0x64	**IF2 Data B2**		**IF2 Data B1**		Little Endian byte ordering.
	bit[15:8]	bit[7:0]	bit[15:8]	bit[7:0]	
	Data[7]	Data[6]	Data[5]	Data[4]	
	Reset: 0x00	Reset: 0x00	Reset: 0x00	Reset: 0x00	

Address	Register				Note
	+0	+1	+2	+3	
Base-addr + 0x80	**Transmission Request Register 2**		**Transmission Request Register 1**		Transmission Request Register is read only.
	bit[15:8]	bit[7:0]	bit[15:8]	bit[7:0]	
	TxRqst[32-25]	TxRqst[24-17]	TxRqst[16-9]	TxRqst[8-1]	
	Reset: 0x00	Reset: 0x00	Reset: 0x00	Reset: 0x00	
Base-addr + 0x84	**Reserved (>32..128 Message buffer)**				
Base-addr + 0x90	**New Data 2**		**New Data 1**		New Data is read only.
	bit[15:8]	bit[7:0]	bit[15:8]	bit[7:0]	
	NewDat[32-25]	NewDat[24-17]	NewDat[16-9]	NewDat[8-1]	
	Reset: 0x00	Reset: 0x00	Reset: 0x00	Reset: 0x00	
Base-addr + 0x94	**Reserved (>32..128 Message buffer)**				
Base-addr + 0xA0	**Interrupt Pending 2**		**Interrupt Pending 1**		Interrupt Pending is read only.
	bit[15:8]	bit[7:0]	bit[15:8]	bit[7:0]	
	IntPnd[32-25]	IntPnd[24-17]	IntPnd[16-9]	IntPnd[8-1]	
	Reset: 0x00	Reset: 0x00	Reset: 0x00	Reset: 0x00	
Base-addr + 0xA4	**Reserved (>32..128 Message buffer)**				
Base-addr + 0xB0	**Message Valid 2**		**Message Valid 1**		Message Valid is read only.
	bit[15:8]	bit[7:0]	bit[15:8]	bit[7:0]	
	MsgVal[32-25]	MsgVal[24-17]	MsgVal[16-9]	MsgVal[8-1]	
	Reset: 0x00	Reset: 0x00	Reset: 0x00	Reset: 0x00	
Base-addr + 0xB4	**Reserved (>32..128 Message buffer)**				

Figure 2-1 CAN Register Summary

Address	Register				Note
	+0	+1	+2	+3	
0x04C0	**CANPRE**	**CANCKD**	**-**	**-**	CAN Prescaler
	bit[3:0]	bit[5:0]	-	-	
	CANPRE[3:0]	CANCKD[5:0]	-	-	
	Reset: 0x00	Reset: 0x00	-	-	

Figure 2-2 CAN Prescaler Register Summary

2.2 Hardware Reset Description

After hardware reset, the registers of the CAN hold the values described in Figure 2-1.

Additionally the busoff state is reset and the output **CAN_TX** is set to recessive(HIGH). The value 0x0001 (**Init** = '1') in the CAN Control Register enables the software initialisation. The CAN does not influence the CAN bus until the CPU resets **Init** to '0'.

The data stored in the Message RAM is not affected by a hardware reset. After power-on, the contents of the Message RAM is undefined.

2.3 CAN Protocol Related Registers

These registers are related to the CAN protocol controller in the CAN Core. They control the operating modes and the configuration of the CAN bit timing and provide status information.

■ CAN Control Register (CTRLR)

CAN Control Register high byte

	15	14	13	12	11	10	9	8	⇐ Bit no.
Address : Base + 0x00H	res	res	res	res	res	res	res	res	CTRLRH
Read/write ⇒	(R)	(R)	(R)	(R)	(R)	(R)	(R)	(R)	
Default value⇒	(0)	(0)	(0)	(0)	(0)	(0)	(0)	(0)	

CAN Control Register low byte

	7	6	5	4	3	2	1	0	⇐ Bit no.
Address : Base + 0x01H	Test	CCE	DAR	res	EIE	SIE	IE	Init	CTRLRL
Read/write ⇒	(R/W)	(R/W)	(R/W)	(R)	(R/W)	(R/W)	(R/W)	(R/W)	
Default value⇒	(0)	(0)	(0)	(0)	(0)	(0)	(0)	(1)	

■ Function of the CAN Control Register (CTRLR)

[bit15 - bit8] Reserved Bits
Always write "0". Read value is not defined. Read-Modify-Write is not affected.

[bit7] **Test** Test Mode Enable

0	Normal Operation.
1	Test Mode.

[bit6] **CCE** Configuration Change Enable

0	The CPU has no write access to the Bit Timing Register.
1	The CPU has write access to the Bit Timing Register (while Init = 1)

[bit5] **DAR** Disable Automatic Retransmission

0	Automatic Retransmission of disturbed messages enabled.
1	Automatic Retransmission disabled.

[bit4] **res** reserved bit

[bit3] **EIE** Error Interrupt Enable

0	Disabled - No Error Status Interrupt will be generated.
1	Enabled - A change in the bits BOff or EWarn in the Status Register will generate an interrupt.

[bit2] **SIE** Status Change Interrupt Enable

0	Disabled - No Status Change Interrupt will be generated.
1	Enabled - An interrupt will be generated when a message transfer is successfully completed or a CAN bus error is detected.

[bit1] **IE** Module Interrupt Enable

0	Disabled - Module Interrupt is always inactive.
1	Enabled - Interrupts will set the internal request. The request remains active until all pending interrupts are processed.

[bit0] **Init** Initialization

0	Normal Operation
1	Initialization is started.

(Note) The busoff recovery sequence (see CAN Specification Rev. 2.0) cannot be shortened by setting or resetting **Init**. If the device goes busoff, it will set **Init** of its own accord, stopping all bus activities. Once **Init** has been cleared by the CPU, the device will then wait for 129 occurrences of Bus Idle (129 * 11 consecutive recessive bits) before resuming normal operations. At the end of the busoff

recovery sequence, the Error Management Counters will be reset.

(Note) During the waiting time after the resetting of **Init**, each time a sequence of 11 recessive bits has been monitored, a **Bit0Error** code is written to the Status Register, enabling the CPU to readily check up whether the CAN bus is stuck at dominant or continuously disturbed and to monitor the proceeding of the busoff recovery sequence.

■ Status Register (STATR)

Status Register high byte

	15	14	13	12	11	10	9	8	⇐ Bit no.
Address : Base + 0x02H	res	res	res	res	res	res	res	res	STATRH
Read/write ⇒	(R)	(R)	(R)	(R)	(R)	(R)	(R)	(R)	
Default value⇒	(0)	(0)	(0)	(0)	(0)	(0)	(0)	(0)	

Status Register low byte

	7	6	5	4	3	2	1	0	⇐ Bit no.
Address : Base + 0x03H	BOff	EWarn	EPass	RxOK	TxOK	LEC			STATRL
Read/write ⇒	(R)	(R)	(R)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	
Default value⇒	(0)	(0)	(0)	(0)	(0)	(0)	(0)	(0)	

■ Function of the Status Register (STATR)

[bit15 - bit8] Reserved Bits

[bit7] **BOff** Busoff Status

0	The CAN module is not busoff.
1	The CAN module is in busoff state.

[bit6] **EWarn** Warning Status

0	oth error counters are below the error warning limit of 96.
1	At least one of the error counters in the EML has reached the error warning limit of 96.

[bit5] **EPass** Error Passive

0	The CAN Core is error active.
1	The CAN Core is in the error passive state as defined in the CANSpecification.

[bit4] **RxOk** Received a Message Successfully

0	Since this bit was last reset by the CPU, no message has been successfully received. This bit is never reset by the CAN Core.
1	Since this bit was last reset (to zero) by the CPU, a message has been successfully received (independent of the result of acceptance)

[bit3] **TxOk** Transmitted a Message Successfully

0	Since this bit was reset by the CPU, no message has been successfully transmitted. This bit is never reset by the CAN Core.
1	Since this bit was last reset by the CPU, a message has been successfully (error free and acknowledged by at least one other node) transmitted.

[bit2 - bit0] **LEC** Last Error Code (Type of the last error to occur on the CAN bus)

0	No Error	
1	Stuff Error	More than 5 equal bits in a sequence have occurred in a part of a received message where this is not allowed.
2	Form Error	A fixed format part of a received frame has the wrong format.
3	AckError	The message this CAN Core transmitted was not acknowledged by another node.
4	Bit1Error	During the transmission of a message (with the exception of the arbitration field), the device wanted to send a recessive level (bit of logical value ‘1’), but the monitored bus value was dominant.

5	Bit0Error	During the transmission of a message (or acknowledge bit or active error flag, or overload flag), the device wanted to send a dominant level (data or identifier bit logical value ‘0’), but the monitored Bus value was recessive. During busoff recovery this status is set each time a sequence of 11 recessive bits has been monitored. This enables the CPU to monitor the proceeding of the busoff recovery sequence (indicating the bus is not stuck at dominant or continuously disturbed).
6	CRCError	The CRC check sum was incorrect in the message received, the CRC received for an incoming message does not match with the calculated CRC for the received data.
7	unused	When the LEC shows the value ‘7’, no CAN bus event was detected since the CPU wrote this value to the LEC.

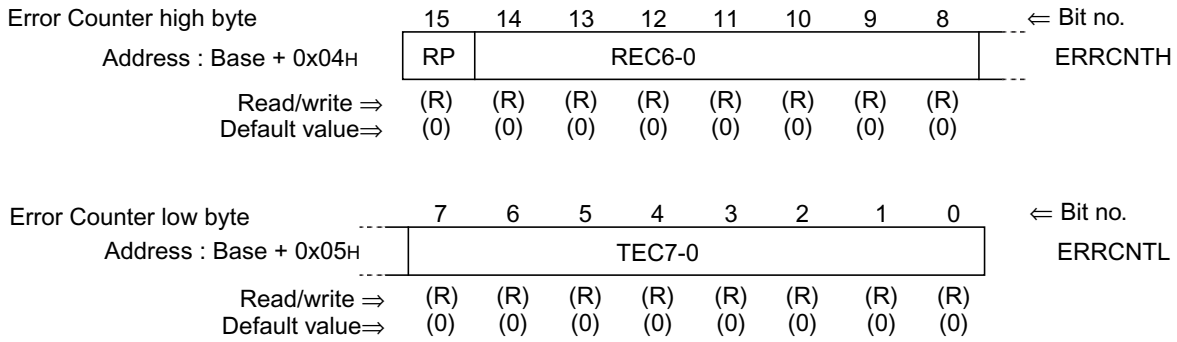
The **LEC** field holds a code which indicates the type of the last error to occur on the CAN bus. This field will be cleared to ‘0’ when a message has been transferred (reception or transmission) without error. The unused code ‘7’ may be written by the CPU to check for updates.

■ Status Interrupts

A Status Interrupt is generated by bits **BOff** and **EWarn** (Error Interrupt) or by **RxOk**, **TxOk**, and **LEC** (Status Change Interrupt) assumed that the corresponding enable bits in the CAN Control Register are set. A change of bit **EPass** or a write to **RxOk**, **TxOk**,or **LEC** will never generate a Status Interrupt.

Reading the Status Register will clear the Status Interrupt value (8000h) in the Interrupt Register, if it is pending.

■ Error Counter (ERRCNT)



■ Function of the Error Counter (ERRCNT)

[bit15] **RP** Receive Error Passive

0 The Receive Error Counter is below the error passive level.
1 The Receive Error Counter has reached the error passive level as defined in the CAN Specification.

[bit14 - bit8] **REC6-0** Receive Error Counter

Actual state of the Receive Error Counter. Values between 0 and 127.

[bit7 - bit0] **TEC7-0** Transmit Error Counter

Actual state of the Transmit Error Counter. Values between 0 and 255.

■ Bit Timing Register (BTR)

Bit Timimg Register high byte
Address : Base + 0x06H

15	14	13	12	11	10	9	8	⇐ Bit no.
res	TSeg2			TSeg1				BTRH
Read/write ⇒	(R)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	
Default value⇒	(0)	(0)	(1)	(0)	(0)	(0)	(1)	(1)	

Bit Timing Register low byte
Address : Base + 0x07H

7	6	5	4	3	2	1	0	⇐ Bit no.
SJW		BRP						BTRL
Read/write ⇒	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	
Default value⇒	(0)	(0)	(0)	(0)	(0)	(0)	(0)	(1)	

■ Function of the Bit Timing Register (BTR)

[bit15]	**res**	Reserved bit
[bit14 - bit12]	**TSeg2**	The time segment after the sample point
	0x0-0x7	Valid values for TSeg2 are [0 … 7]. The actual interpretation by the hardware of this value is such that one more than the value programmed here is used.
[bit11 - bit8]	**TSeg1**	The time segment before the sample point
	0x01-0x0F	Valid values for TSeg1 are [1 … 15]. The actual interpretation by the hardware of this value is such that one more than the value programmed here is used
[bit7 - 6]	**SJW**	(Re)Synchronisation Jump Width
	0x0-0x3	Valid programmed values are 0-3. The actual interpretation by the hardware of this value is such that one more than the value programmed here is used.
[bit5 - bit0]	**BRP**	Baud Rate Prescaler
	0x00-0x3F	The value by which the oscillator frequency is divided for generating the bit time quanta. The bit time is built up from a multiple of this quanta. Valid values for the Baud Rate Prescaler are[0 … 63]. The actual interpretation by the hardware of this value is such that one more than the value programmed here is used.

(Note) With a module clock **CAN_CLK** of 8 MHz, the reset value of 0x2301 configures the CAN for a bit rate of 500 kBit/s. The registers are only writable if bits **CCE** and **Init** in the CAN Control Register are set.

■ Test Register (TESTR)

Test Register high byte

	15	14	13	12	11	10	9	8	⇐ Bit no.
Address : Base + 0x0AH	res	res	res	res	res	res	res	res	TESTRH
Read/write ⇒	(R)	(R)	(R)	(R)	(R)	(R)	(R)	(R)	
Default value⇒	(0)	(0)	(0)	(0)	(0)	(0)	(0)	(0)	

Test Register low byte

	7	6	5	4	3	2	1	0	⇐ Bit no.
Address : Base + 0x0BH	Rx	Tx1	Tx0	LBack	Silent	Basic	res	res	TESTRL
Read/write ⇒	(R)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R)	(R)	
Default value⇒	(0)	(0)	(0)	(0)	(0)	(0)	(0)	(0)	

■ Function of the Test Register (TESTR)

[bit15-bit8]	**res**	Reserved bits
[bit7]	**Rx**	Monitors the actual value of the CAN_RX Pin
	0	The CAN bus is dominant (CAN_RX = ‘0’).
	1	The CAN bus is recessive (CAN_RX = ‘1’).
[bit6-bit5]	**Tx1-0**	Control of CAN_TX pin
	00	Reset value, CAN_TX is controlled by the CAN Core.
	01	Sample Point can be monitored at CAN_TX pin.
	10	CAN_TX pin drives a dominant (‘0’) value.
	11	CAN_TX pin drives a recessive (‘1’) value.
[bit4]	**LBack**	Loop Back Mode
	0	Loop Back Mode is disabled.
	1	Loop Back Mode is enabled.
[bit3]	**Silent**	Silent Mode
	0	Normal operation.
	1	The module is in Silent Mode.
[bit2]	**Basic**	Basic Mode
	0	Basic Mode disabled.
	1	IF1 Registers used as Tx Buffer, IF2 Registers used as Rx Buffer.
[bit1-bit0]	**res**	Reserved Bits

Write access to the Test Register is enabled by setting bit **Test** in the CAN Control Register. The different test functions may be combined, but **Tx1-0** /= “00” disturbs message transfer.

■ BRP Extension Register (BRPER)

BRP Extension Register high byte	15	14	13	12	11	10	9	8	⇐ Bit no.
Address : Base + 0x0CH	res	res	res	res	res	res	res	res	BRPERH
Read/write ⇒	(R)	(R)	(R)	(R)	(R)	(R)	(R)	(R)	
Default value⇒	(0)	(0)	(0)	(0)	(0)	(0)	(0)	(0)	

BRP Extension Register low byte	7	6	5	4	3	2	1	0	⇐ Bit no.
Address : Base + 0x0DH	res	res	res	res	BRPE				BRPERL
Read/write ⇒	(R)	(R)	(R)	(R)	(R/W)	(R/W)	(R/W)	(R/W)	
Default value⇒	(0)	(0)	(0)	(0)	(0)	(0)	(0)	(0)	

■ Function of the BRP Extension Register (BRPER)

[bit15-bit4] **res** Reserved Bits

[bit3-bit0] **BRPE** Baud Rate Prescaler Extension

0x00-0x0F By programming BRPE the Baud Rate Prescaler can be extended to values up to 1023. The actual interpretation by the hardware is that one more than the value programmed by BRPE (MSBs) and BRP (LSBs) is used.

2.4 Message Interface Register Sets

There are two sets of Interface Registers which are used to control the CPU access to the Message RAM. The Interface Registers avoid conflicts between CPU access to the Message RAM and CAN message reception and transmission by buffering the data to be transferred. A complete Message Object (see chapter 2.5 “Message Object in the Message Memory” on page 713.) or parts of the Message Object may be transferred between the Message RAM and the IFx Message Buffer registers in one single transfer.

The function of the two interface register sets is identical (except for test mode **Basic**). They can be used the way that one set of registers is used for data transfer **to** the Message RAM while the other set of registers is used for the data transfer **from** the Message RAM, allowing both processes to be interrupted by each other. Figure 2-3 gives an overview of the two Interface Register sets.

Each set of Interface Registers consists of Message Buffer Registers controlled by their own Command Registers. The Command Mask Register specifies the direction of the data transfer and which parts of a Message Object will be transferred. The Command Request Register is used to select a Message Object in the Message RAM as target or source for the transfer and to start the action specified in the Command Mask Register.

Address	IF1 Register Set	Address	IF2 Register Set
CAN Base + 0x10	IF1 Command Request	CAN Base + 0x40	IF2 Command Request
CAN Base + 0x12	IF1 Command Mask	CAN Base + 0x42	IF2 Command Mask
CAN Base + 0x14	IF1 Mask 2	CAN Base + 0x44	IF2 Mask 2
CAN Base + 0x16	IF1 Mask 1	CAN Base + 0x46	IF2 Mask 1
CAN Base + 0x18	IF1 Arbitration 2	CAN Base + 0x48	IF2 Arbitration 2
CAN Base + 0x1A	IF1 Arbitration 1	CAN Base + 0x4A	IF2 Arbitration 1
CAN Base + 0x1C	IF1 Message Control	CAN Base + 0x4C	IF2 Message Control
CAN Base + 0x20 CAN Base + 0x32	IF1 Data A1	CAN Base + 0x50 CAN Base + 0x62	IF2 Data A1
CAN Base + 0x22 CAN Base + 0x30	IF1 Data A2	CAN Base + 0x52 CAN Base + 0x60	IF2 Data A2
CAN Base + 0x24 CAN Base + 0x36	IF1 Data B1	CAN Base + 0x54 CAN Base + 0x66	IF2 Data B1
CAN Base + 0x26 CAN Base + 0x34	IF1 Data B2	CAN Base + 0x56 CAN Base + 0x64	IF2 Data B2

Figure 2-3 IF1 and IF2 Message Interface Register Sets

■ IFx Command Request Registers (IFxCREQ)

A message transfer is started as soon as the CPU has written the message number to the Command Request Register. With this write operation the CPU is notified that a transfer is in progress. If a CPU access to the CAN happens while the transfer is in progress then this access is delayed until the transfer has finished. After 3 to 6 **CAN_CLK** periods, the transfer between the Interface Register and the Message RAM has completed and the upcoming CPU access is executed.

IFx Command Request Register high byte

	15	14	13	12	11	10	9	8	⇐ Bit no.
Address : Base + 0x10H & Base + 0x40H	BUSY	res	res	res	res	res	res	res	IFxCREQH
Read/write ⇒	(R/(W))	(R)	(R)	(R)	(R)	(R)	(R)	(R)	
Default value⇒	(0)	(0)	(0)	(0)	(0)	(0)	(0)	(0)	

IFx Command Request Register low byte

	7	6	5	4	3	2	1	0	⇐ Bit no.
Address : Base + 0x11H & Base + 0x41H	Message Number								IFxCREQL
Read/write ⇒	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	
Default value⇒	(0)	(0)	(0)	(0)	(0)	(0)	(0)	(1)	

■ Function of the IFx Command Request Registers (IFxCREQ)

[bit15]	**BUSY**	Busy Flag
	0	Reset to zero when read/write action has finished
	1	Set to one when writing to the IFx Command Request Register
[bit14-bit8]	**res**	Reserved Bits
[bit5-bit0]		Message Number (for 32 message buffer CANs)
	0x00	Not a valid Message Number, interpreted as 0x20.
	0x01-0x20	Valid Message Number, the Message Object in the Message RAM is selected for data transfer.
	0x21-0x3F	Not a valid Message Number, interpreted as 0x01-0x1F.
[bit7-bit0]		Message Number (for 128 message buffer CANs)
	0x00	Not a valid Message Number, interpreted as 0x80.
	0x01-0x80	Valid Message Number, the Message Object in the Message RAM is selected for data transfer.
	0x81-0xFF	Not a valid Message Number, interpreted as 0x01-0x7F.

(Note) Note: The Busy Flag can only be used in BASIC mode (see chapter 3.8). When using the Message RAM (BASIC=0) the hardware interface controls the access for read and write and this flag is always read as ‘0’.

(Note) When a **Message Number** that is not valid is written into the Command Request Register, the **Message Number** will be transformed into a valid value and that Message Object will be transferred.

■ IFx Command Mask Register (IFxCMSK)

The control bits of the IFx Command Mask Register specify the transfer direction and select which of the IFx Message Buffer Registers are source or target of the data transfer.

IFx Command Mask Register high byte

	15	14	13	12	11	10	9	8	
Address : Base + 0x12H & Base + 0x42H	res	res	res	res	res	res	res	res	IFxCMSKH
Read/write ⇒	(R)	(R)	(R)	(R)	(R)	(R)	(R)	(R)	
Default value⇒	(0)	(0)	(0)	(0)	(0)	(0)	(0)	(0)	

⇐ Bit no.

IFx Command Mask Register low byte

	7	6	5	4	3	2	1	0	
Address : Base + 0x13H & Base + 0x43H	WR/RD	Mask	Arb	Control	CIP	TxReq/NewDat	Data A	Data B	IFxCMSKL
Read/write ⇒	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	
Default value⇒	(0)	(0)	(0)	(0)	(0)	(0)	(0)	(0)	

⇐ Bit no.

■ Function of the IFx Command Mask Register (IFxCMSK)

[bit15-bit8] **res** Reserved Bits

[bit7] **WR/RD** Write / Read

0 Read: Transfer data from the Message Object addressed by the Command Request Register into the selected Message Buffer Registers.
1 Write: Transfer data from the selected Message Buffer Registers to the Message Object addressed by the Command Request Register.

The other bits of IFx Command Mask Register have different functions depending on the transfer direction :

- **Direction = Write**

[bit6] **Mask** Access Mask Bits

0 Mask bits unchanged.
1 Transfer Identifier Mask + MDir + MXtd to Message Object.

[bit5] **Arb** Access Arbitration Bits

0 Arbitration bits unchanged.
1 Transfer Identifier + Dir + Xtd + MsgVal to Message Object.

[bit4] **Control** Access Control Bits

0 Control Bits unchanged.
1 Transfer Control Bits to Message Object.

[bit3] **CIP** Clear Interrupt Pending Bit

When writing to a Message Object, this bit is ignored.

[bit2] **TxReq/ NewDat** Access Transmission Request Bit

0 TxRqst bit unchanged
1 set TxRqst bit

[bit1] **Data A** Access Data Bytes 0-3

0 Data Bytes 0-3 unchanged.
1 Transfer Data Bytes 0-3 to Message Object.

[bit0] **Data B** Access Data Bytes 4-7

0 Data Bytes 4-7 unchanged.
1 Transfer Data Bytes 4-7 to Message Object.

(Note) If a transmission is requested by programming bit **TxRqst/NewDat** in the IFx Command Mask Register, bit **TxRqst** in the IFx Message Control Register will be ignored.

- **Direction = Read**

[bit6] **Mask** Access Mask Bits

0 Mask bits unchanged.
1 Transfer Identifier Mask + MDir + MXtd to IFx Message Buffer Register.

[bit5] **Arb** Access Arbitration Bits

0 Arbitration bits unchanged.
1 Transfer Identifier + Dir + Xtd + MsgVal to IFx Message Buffer Register.

[bit4] **Control** Access Control Bits

0 Control Bits unchanged.
1 Transfer Control Bits to IFx Message Buffer Register.

[bit3] **CIP** Clear Interrupt Pending Bit

0 IntPnd bit remains unchanged.
1 Clear IntPnd bit in the Message Object.

[bit2] **TxReq/ NewDat** Access New Data Bit

0 NewDat bit remains unchanged.
1 Clear NewDat bit in the Message Object.

[bit1] **Data A** Access Data Bytes 0-3

0 Data Bytes 0-3 unchanged.
1 Transfer Data Bytes 0-3 to IFx Message Buffer Register.

[bit0] **Data B** Access Data Bytes 4-7

0 Data Bytes 4-7 unchanged.
1 Transfer Data Bytes 4-7 to IFx Message Buffer Register.

(Note) A read access to a Message Object can be combined with the reset of the control bits **IntPnd** and **NewDat**. The values of these bits transferred to the IFx Message Control Register always reflect the status before resetting these bits.

■ IFx Mask Registers (IFxMSK)

The bits of the Message Buffer registers mirror the Message Objects in the Message RAM.

IFx Mask 2 Register high byte

15	14	13	12	11	10	9	8	⇐ Bit no.
MXtd	MDir	res	Msk28-24					IFxMSK2H
(R/W)	(R/W)	(R)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	Read/write ⇒
(1)	(1)	(1)	(1)	(1)	(1)	(1)	(1)	Default value⇒

Address : Base + 0x14H & Base + 0x44H

IFx Mask 2 Register low byte

7	6	5	4	3	2	1	0	⇐ Bit no.
Msk23-16								IFxMSK2L
(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	Read/write ⇒
(1)	(1)	(1)	(1)	(1)	(1)	(1)	(1)	Default value⇒

Address : Base + 0x15H & Base + 0x45H

IFx Mask 1 Register high byte

15	14	13	12	11	10	9	8	⇐ Bit no.
Msk15-8								IFxMSK1H
(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	Read/write ⇒
(1)	(1)	(1)	(1)	(1)	(1)	(1)	(1)	Default value⇒

Address : Base + 0x16H & Base + 0x46H

IFx Mask 1 Register low byte

7	6	5	4	3	2	1	0	⇐ Bit no.
Msk7-0								IFxMSK1L
(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	Read/write ⇒
(1)	(1)	(1)	(1)	(1)	(1)	(1)	(1)	Default value⇒

Address : Base + 0x17H & Base + 0x47H

■ IFx Arbitration Registers (IFxARB)

The bits of the Message Buffer registers mirror the Message Objects in the Message RAM.

IFx Arbitration 2 Register high byte

15	14	13	12	11	10	9	8	⇐ Bit no.
MsgVal	Xtd	Dir	ID28-24					IFxARB2H
(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	Read/write ⇒
(0)	(0)	(0)	(0)	(0)	(0)	(0)	(0)	Default value⇒

Address : Base + 0x18H & Base + 0x48H

IFx Arbitration 2 Register low byte

7	6	5	4	3	2	1	0	⇐ Bit no.
ID23-16								IFxARB2L
(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	Read/write ⇒
(0)	(0)	(0)	(0)	(0)	(0)	(0)	(0)	Default value⇒

Address : Base + 0x19H & Base + 0x49H

IFx Arbitration 1 Register high byte

15	14	13	12	11	10	9	8	⇐ Bit no.
ID15-8								IFxARB1H
(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	Read/write ⇒
(0)	(0)	(0)	(0)	(0)	(0)	(0)	(0)	Default value⇒

Address : Base + 0x1AH & Base + 0x4AH

IFx Arbitration 1 Register low byte

7	6	5	4	3	2	1	0	⇐ Bit no.
ID7-0								IFxARB1L
(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	Read/write ⇒
(0)	(0)	(0)	(0)	(0)	(0)	(0)	(0)	Default value⇒

Address : Base + 0x1BH & Base + 0x4BH

■ IFx Message Control Register (IFxMCTR)

IFx Message Control Register high byte

15	14	13	12	11	10	9	8	⇐ Bit no.
NewDat	MsgLst	IntPnd	UMask	TxIE	RxIE	RmtEn	TxRqst	IFxMCTRH
(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	Read/write ⇒
(0)	(0)	(0)	(0)	(0)	(0)	(0)	(0)	Default value⇒

Address : Base + 0x1CH & Base + 0x4CH

IFx Message Control Register low byte

7	6	5	4	3	2	1	0	⇐ Bit no.
EoB	res	res	res	DLC3-0				IFxMCTRL
(R/W)	(R)	(R)	(R)	(R/W)	(R/W)	(R/W)	(R/W)	Read/write ⇒
(0)	(0)	(0)	(0)	(0)	(0)	(0)	(0)	Default value⇒

Address : Base + 0x1DH & Base + 0x4DH

■ IFx Data A and Data B Registers (IFxDTA, IFxDTB)

The data bytes of CAN messages are stored in the IFx Message Buffer Registers in the following order:

	addr+0	addr+1	addr+2	addr+3
IFxMessage Data A1 (addresses 0x20 & 0x50)	Data(0)	Data(1)		
IFx Message Data A2 (addresses 0x22 & 0x52)			Data(2)	Data(3)
IFx Message Data B1 (addresses 0x24 & 0x54)	Data(4)	Data(5)		
IFx Message Data B2 (addresses 0x26 & 0x56)			Data(6)	Data(7)
IFx Message Data A2 (addresses 0x30 & 0x60)	Data(3)	Data(2)		
IFx Message Data A1 (addresses 0x32 & 0x62)			Data(1)	Data(0)
IFx Message Data B2 (addresses 0x34 & 0x64)	Data(7)	Data(6)		
IFx Message Data B1 (addresses 0x36 & 0x66)			Data(5)	Data(4)

In a CAN Data Frame, Data(0) is the first, Data(7) is the last byte to be transmitted or received. In CAN's serial bit stream, the MSB of each byte will be transmitted first.

2.5 Message Object in the Message Memory

There are 32 Message Objects (up to 128 depending on the implementation) in the Message RAM. To avoid conflicts between CPU access to the Message RAM and CAN message reception and transmission, the CPU cannot directly access the Message Objects, these accesses are handled via the IFx Interface Registers. Figure 2-4 gives an overview of the two structures of a Message Object.

Message Object												
UMask	Msk28-0	MXtd	MDir	EoB	NewDat		MsgLst	RxIE	TxIE	IntPnd	RmtEn	TxRqst
MsgVal	ID28-0	Xtd	Dir	DLC3-0	Data0	Data1	Data2	Data3	Data4	Data5	Data6	Data7

Figure 2-4 Structure of a Message Object in the Message Memory

MsgVal Message Valid

0 The Message Object is ignored by the Message Handler.
1 The Message Object is configured and should be considered by the Message Handler.

(Note) The CPU must reset the **MsgVal** bit of all unused Message Objects during the initialization before it resets bit **Init** in the CAN Control Register. This bit must also be reset before the identifier **ID28-0**, the control bits **Xtd**, **Dir**, or the Data Length Code **DLC3-0** are modified, or if the Message Object is no longer required.

UMask Use Acceptance Mask

0 Mask ignored.
1 Use Mask (Msk28-0, MXtd, and MDir) for acceptance filtering.

(Note) If the **UMask** bit is set to one, the Message Object's mask bits have to be programmed during initialization of the Message Object before **MsgVal** is set to one.

ID28-0 Message Identifier

ID28 - ID0 29-bit Identifier ("Extended Frame").
ID28 - ID18 11-bit Identifier ("Standard Frame").

Msk28-0 Identifier Mask

0 The corresponding bit in the identifier of the message object cannot inhibit the match in the acceptance filtering.
1 The corresponding identifier bit is used for acceptance filtering.

Xtd Extended Identifier

0 The 11-bit ("standard") Identifier will be used for this Message Object.

1 The 29-bit (“extended”) Identifier will be used for this Message Object.

MXtd Mask Extended Identifier

0 The extended identifier bit (IDE) has no effect on the acceptance filtering
1 The extended identifier bit (IDE) is used for acceptance filtering.

(Note) When 11-bit (“standard”) Identifiers are used for a Message Object, the identifiers of received Data Frames are written into bits **ID28** to **ID18**. For acceptance filtering, only these bits together with mask bits **Msk28** to **Msk18** are considered.

Dir Message Direction

0 Direction = receive: On TxRqst, a Remote Frame with the identifier of this Message Object is transmitted. On reception of a Data Frame with matching identifier, that message is stored in this Message Object.
1 Direction = transmit: On TxRqst, the respective Message Object is transmitted as a Data Frame. On reception of a Remote Frame with matching identifier, the TxRqst bit of this Message Object is set (if RmtEn = 1).

MDir Mask Message Direction

0 The message direction bit (Dir) has no effect on the acceptance filtering.
1 The message direction bit (Dir) is used for acceptance filtering.

The Arbitration Registers **ID28-0**, **Xtd**, and **Dir** are used to define the identifier and type of outgoing messages and are used (together with the mask registers **Msk28-0**, **MXtd**, and **MDir**) for acceptance filtering of incoming messages. A received message is stored into the valid Message Object with matching identifier and Direction= receive (Data Frame) or Direction= transmit(Remote Frame). Extended frames can be stored only in Message Objects with **Xtd** = 1, standard frames in Message Objects with **Xtd** = 0. If a received message (Data Frame or Remote Frame) matches with more than one valid Message Object, it is stored into that with the lowest message number. For details see 4.5 “Acceptance Filtering of Received Messages” on page 729.

EoB End of Buffer

0 Message Object belongs to a FIFO Buffer and is not the last Message Object of that FIFO Buffer.
1 Single Message Object or last Message Object of a FIFO Buffer.

(Note) This bit is used to concatenate two ore more Message Objects (up to 32) to build a FIFO Buffer. **For single Message Objects (not belonging to a FIFO Buffer) this bit must always be set to 1**. For details on the concatenation of Message Objects see chapter 4.13 “Configuration of a FIFO Buffer” on page 731.

NewDat New Data

0 No new data has been written into the data portion of this Message Object by the Message Handler since last time this flag was cleared by the CPU.
1 The Message Handler or the CPU has written new data into the data portion of this Message Object.

MsgLst Message Lost (only valid for Message Objects with direction = receive)

0 No message lost since last time this bit was reset by the CPU.
1 The Message Handler stored a new message into this object when NewDat was still set, the CPU has lost a message.

RxIE Receive Interrupt Enable

0 IntPnd will be left unchanged after a successful reception of a frame.
1 IntPnd will be set after a successful reception of a frame.

TxIE Transmit Interrupt Enable

0 IntPnd will be left unchanged after the successful transmission of a frame.
1 IntPnd will be set after a successful transmission of a frame.

IntPnd Interrupt Pending

0 This message object is not the source of an interrupt.
1 This message object is the source of an interrupt. The Interrupt Identifier in the Interrupt Register will point to this message object if there is no other interrupt source with higher priority.

RmtEn Remote Enable

0 At the reception of a Remote Frame, TxRqst is left unchanged.
1 At the reception of a Remote Frame, TxRqst is set.

TxRqst Transmit Request

0 This Message Object is not waiting for transmission.
1 The transmission of this Message Object is requested and is not yet done.

(Note) When the lowest priority message buffer is used for transmission, setting **TxRqst** = "0" may cause a delay of transmission, when **TxRqst** is set to "1" again.

Depending on the exact time when **TxRqst** was set to 0, the message may not be transmitted immediately after setting **TxRqst** = "1", but after any of the following events:

1. there is activity ongoing on the CANbus
2. a transmission request is issued on another message object
3. the CANbus is initialized by the INIT bit

In general, there is no need to cancel an ongoing transmission by setting TxRqst = 0. If the content of a message object needs to be changed while TxRqst = 1, it is sufficient to update the message object via the CPU interface registers (Identifier, **DLC**, **Data**, with **TxRqst** and **NewDat**, optionally **TxIE**). The updated content will be transmitted at the next opportunity.

DLC3-0 Data Length Code

0-8 Data Frame has 0-8 data bytes.
9-15 Data Frame has 8 data bytes.

(Note) The Data Length Code of a Message Object must be defined the same as in all the corresponding

objects with the same identifier at other nodes. When the Message Handler stores a data frame, it will write the DLC to the value given by the received message.

Data 0 1st data byte of a CAN Data Frame
Data 1 2nd data byte of a CAN Data Frame
Data 2 3rd data byte of a CAN Data Frame
Data 3 4th data byte of a CAN Data Frame
Data 4 5th data byte of a CAN Data Frame
Data 5 6th data byte of a CAN Data Frame
Data 6 7th data byte of a CAN Data Frame
Data 7 8th data byte of a CAN Data Frame

(Note) Byte **Data 0** is the first data byte shifted into the shift register of the CAN Core during a reception, byte **Data 7** is the last. When the Message Handler stores a Data Frame, it will write all the eight data bytes into a Message Object. If the Data Length Code is less than 8, the remaining bytes of the Message Object will be overwritten by **non specified values**.

2.6 Message Handler Registers

All Message Handler registers are read-only. Their contents (**TxRqst**, **NewDat**, **IntPnd**, and **MsgVal** bits of each Message Object and the Interrupt Identifier) is status information provided by the Message Handler FSM.

■ Interrupt Register (INTR)

Interrupt Register high byte — Address : Base + 0x08H

	15	14	13	12	11	10	9	8	
	IntId15-8								INTRH
Read/write ⇒	(R)	(R)	(R)	(R)	(R)	(R)	(R)	(R)	
Default value⇒	(0)	(0)	(0)	(0)	(0)	(0)	(0)	(0)	

⇐ Bit no.

Interrupt Register low byte — Address : Base + 0x09H

	7	6	5	4	3	2	1	0	
	IntId7-0								INTRL
Read/write ⇒	(R)	(R)	(R)	(R)	(R)	(R)	(R)	(R)	
Default value⇒	(0)	(0)	(0)	(0)	(0)	(0)	(0)	(0)	

⇐ Bit no.

■ Function of the Interrupt Register (INTR)

- (For 32 message buffer CANs)

IntId15-0 Interrupt Identifier (the number here indicates the source of the interrupt)

0x0000	No interrupt is pending.
0x0001-0x0020	Number of Message Object which caused the interrupt.
0x0021-0x7FFF	unused.
0x8000	Status Interrupt.

0x8001-0xFFFF	unused.

- (For 128 message buffer CANs)

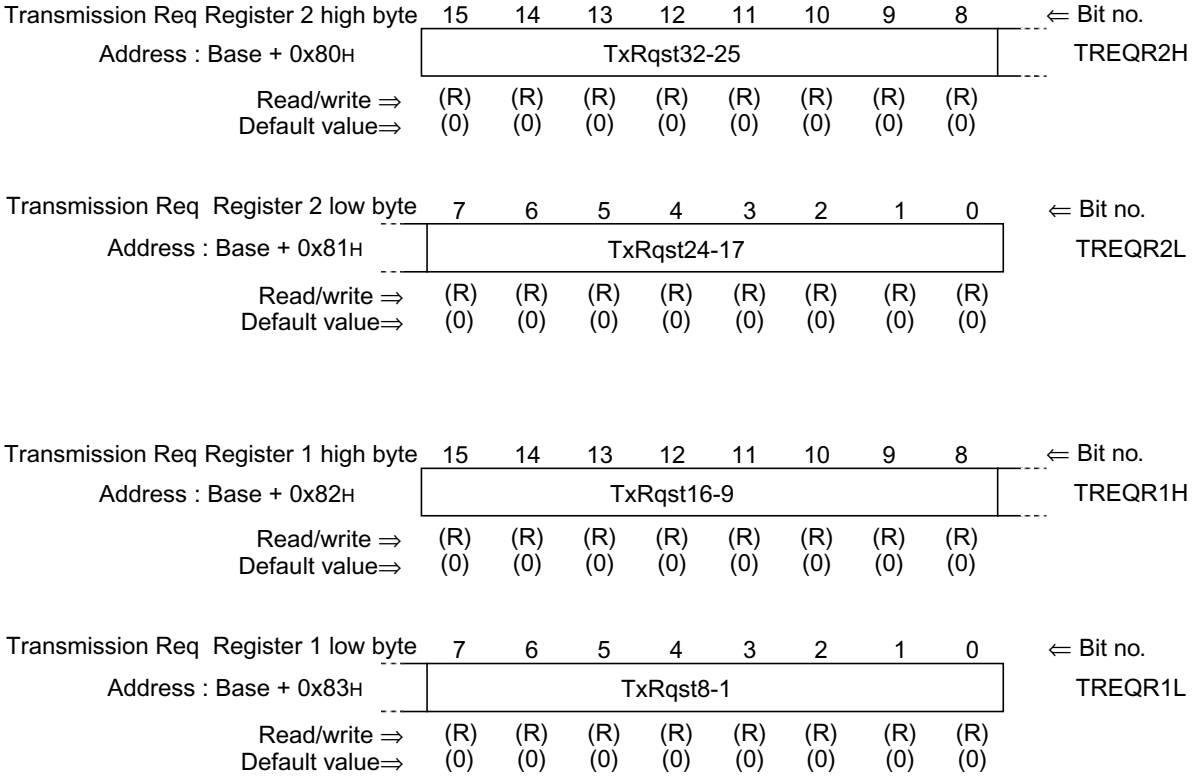
IntId15-0	Interrupt Identifier (the number here indicates the source of the interrupt)
0x0000	No interrupt is pending.
0x0001-0x0080	Number of Message Object which caused the interrupt.
0x0081-0x7FFF	unused.
0x8000	Status Interrupt.
0x8001-0xFFFF	unused.

If several interrupts are pending, the CAN Interrupt Register will point to the pending interrupt with the highest priority, disregarding their chronological order. An interrupt remains pending until the CPU has cleared it. If **IntId** is different from 0x0000 and **IE** is set, the interrupt line to the CPU is active. The interrupt line remains active until **IntId** is back to value 0x0000 (the cause of the interrupt is reset) or until **IE** is reset.

The Status Interrupt has the highest priority. Among the message interrupts, the Message Object' s interrupt priority decreases with increasing message number.

A message interrupt is cleared by clearing the Message Object's **IntPnd** bit. The Status Interrupt is cleared by reading the Status Register.

■ Transmission Request Registers (TREQR)



TxRqst32-1 Transmission Request Bits (of all Message Objects)

0 This Message Object is not waiting for transmission.
1 The transmission of this Message Object is requested and is not yet done.

These registers hold the **TxRqst** bits of the 32 Message Objects. By reading out the **TxRqst** bits, the CPU can check for which Message Object a Transmission Request is pending. The **TxRqst** bit of a specific Message Object can be set/reset by the CPU via the IFx Message Interface Registers or by the Message Handler after reception of a Remote Frame or after a successful transmission.

(Note) When the lowest priority message buffer is used for transmission, setting **TxRqst** = "0" may cause a delay of transmission, when **TxRqst** is set to "1" again.

Depending on the exact time when **TxRqst** was set to 0, the message may not be transmitted immediately after setting **TxRqst** = "1", but after any of the following events:

1. there is activity ongoing on the CANbus
2. a transmission request is issued on another message object
3. the CANbus is initialized by the INIT bit

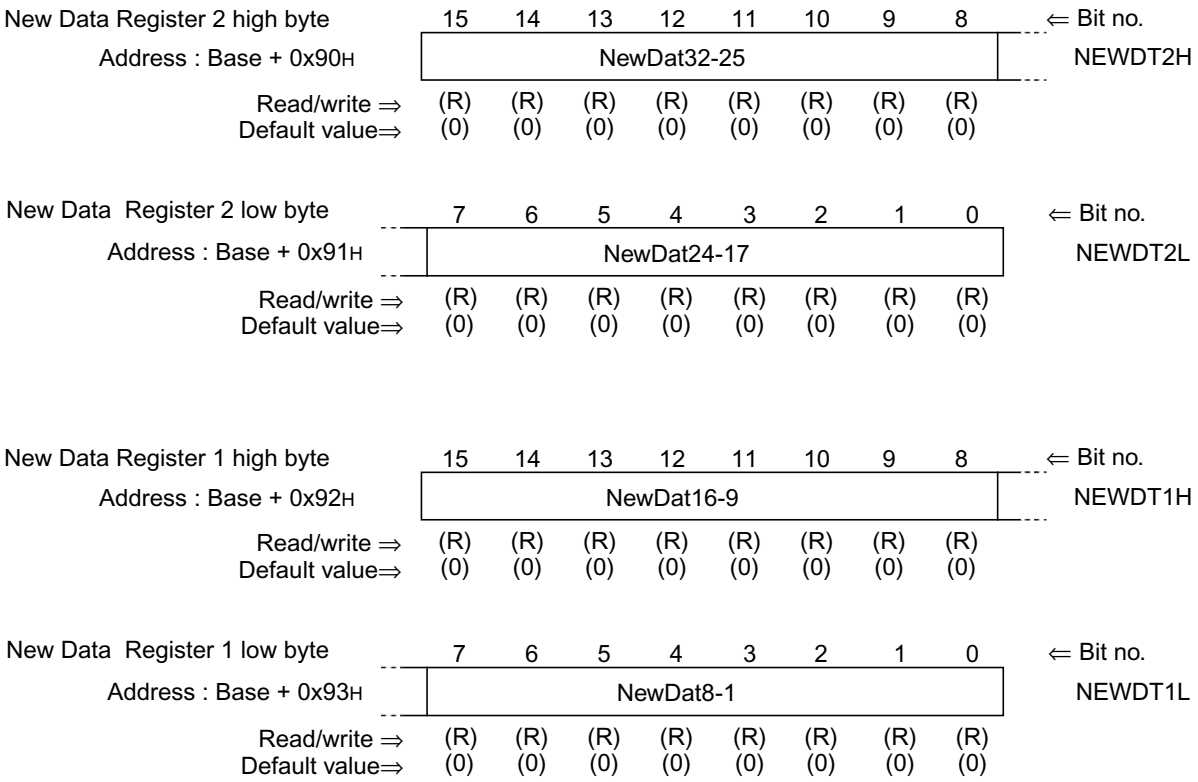
In general, there is no need to cancel an ongoing transmission by setting **TxRqst** = 0. If the content of a message object needs to be changed while **TxRqst** = 1, it is sufficient to update the message object via the CPU interface registers (Identifier, **DLC**, **Data**, with **TxRqst** and **NewDat**, optionally **TxIE**). The updated content will be transmitted at the next opportunity.

If more than 32 message buffers are implemented, the following table gives an overview about the additional flags:

Table 2-1 Additional flags when more than 32 message buffers exist

		addr+0	addr+1	addr+2	addr+3
TREQR 4 & 3	TxRqst 64-33 (address 0x84)	TxRqst64-57	TxRqst56-49	TxRqst48-41	TxRqst40-33
TREQR 6 & 5	TxRqst 96-65 (address 0x88)	TxRqst96-89	TxRqst88-81	TxRqst80-73	TxRqst72-65
TREQR 8 & 7	TxRqst 128-97 (address 0x8C)	TxRqst128-121	TxRqst120-113	TxRqst112-105	TxRqst104-97

■ New Data Registers (NEWDT)



NewDat32-1 New Data Bits (of all Message Objects)

0	No new data has been written into the data portion of this Message Object by the Message Handler since last time this flag was cleared by the CPU.
1	The Message Handler or the CPU has written new data into the data portion of this Message Object.

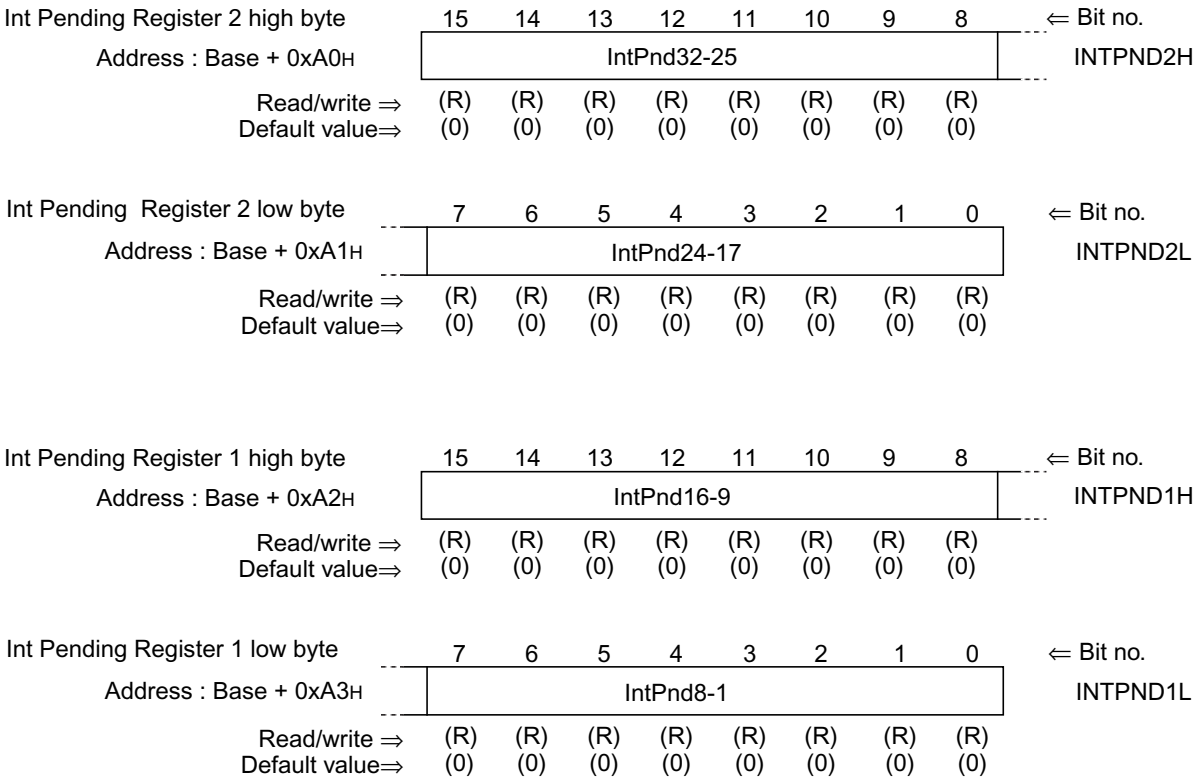
These registers hold the **NewDat** bits of the 32 Message Objects. By reading out the **NewDat** bits, the CPU can check for which Message Object the data portion was updated. The **NewDat** bit of a specific Message Object can be set/reset by the CPU via the IFx Message Interface Registers or by the Message Handler after reception of a Data Frame or after a successful transmission.

If more than 32 message buffers are implemented, the following table gives an overview about the additional flags:

Table 2-2 Additional flags when more than 32 message buffers exist

		addr+0	addr+1	addr+2	addr+3
NEWDT 4 & 3	NewDat 64-33 (address 0x94)	NewDat64-57	NewDat56-49	NewDat48-41	NewDat40-33
NEWDT 6 & 5	NewDat 96-65 (address 0x98)	NewDat96-89	NewDat88-81	NewDat80-73	NewDat72-65
NEWDT 8 & 7	NewDat 128-97 (address 0x9C)	NewDat128-121	NewDat120-113	NewDat112-105	NewDat104-97

■ Interrupt Pending Registers (INTPND)



IntPnd32-1 Interrupt Pending Bits (of all Message Objects)

0 This message object is not the source of an interrupt.
1 This message object is the source of an interrupt.

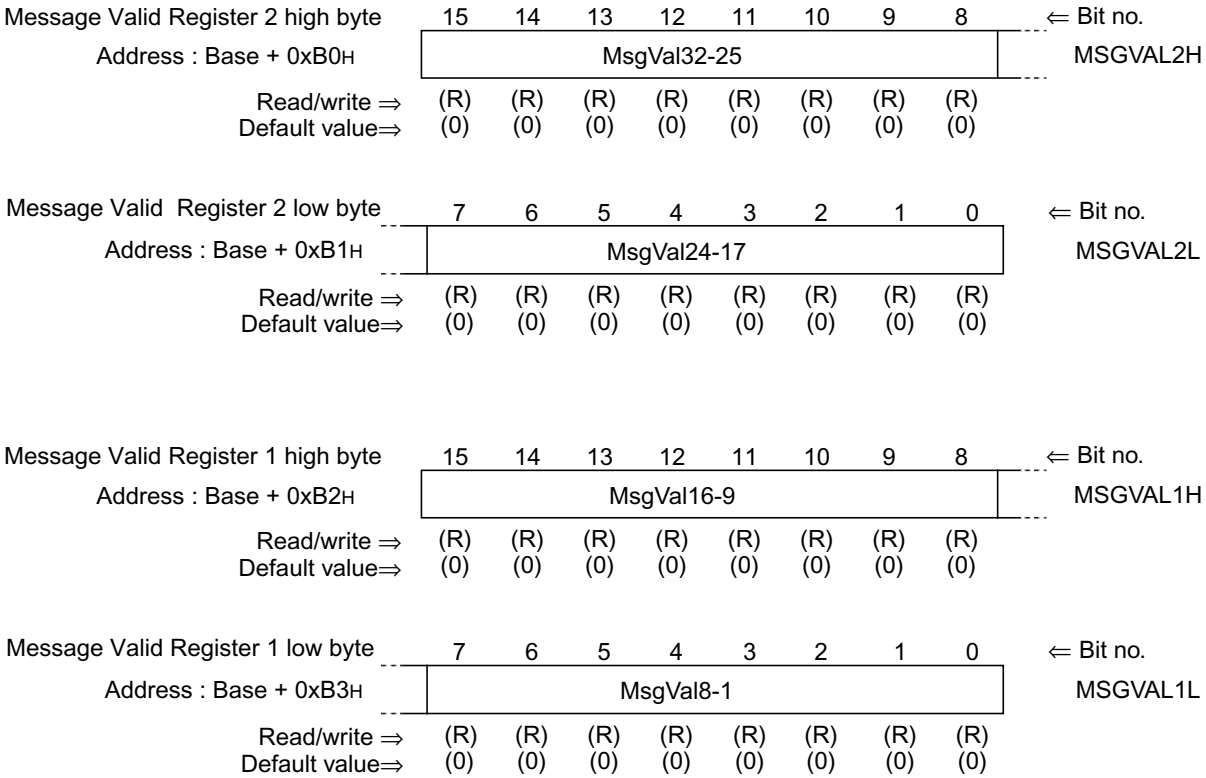
These registers hold the **IntPnd** bits of the 32 Message Objects. By reading out the **IntPnd** bits, the CPU can check for which Message Object an interrupt is pending. The **IntPnd** bit of a specific Message Object can be set/reset by the CPU via the IFx Message Interface Registers or by the Message Handler after reception or after a successful transmission of a frame. This will also affect the value of **IntId** in the Interrupt Register.

If more than 32 message buffers are implemented, the following table gives an overview about the additional flags:

Table 2-3 Additional flags when more than 32 message buffers exist

		addr+0	addr+1	addr+2	addr+3
INTPND 4 & 3	IntPnd 64-33 (address 0xA4)	IntPnd64-57	IntPnd56-49	IntPnd48-41	IntPnd40-33
INTPND 6 & 5	IntPnd 96-65 (address 0xA8)	IntPnd96-89	IntPnd88-81	IntPnd80-73	IntPnd72-65
INTPND 8 & 7	IntPd 128-97 (address 0xAC)	IntPnd128-121	IntPnd120-113	IntPnd112-105	IntPnd104-97

■ Message Valid Registers (MSGVAL)



MsgVal32-1 Message Valid Bits (of all Message Objects)

0 This Message Object is ignored by the Message Handler.
1 This Message Object is configured and should be considered by the Message Handler.

These registers hold the **MsgVal** bits of the 32 Message Objects. By reading out the **MsgVal** bits, the CPU can check which Message Object is valid. The **MsgVal** bit of a specific Message Object can be set/reset by the CPU via the IFx Message Interface Registers.

If more than 32 message buffers are implemented then the following table gives an overview about the

additional flags:

Table 2-4 Additional flags when more than 32 message buffers exist

		addr+0	addr+1	addr+2	addr+3
MSGVAL 4 & 3	MsgVal 64-33 (address 0xB4)	MsgVal64-57	MsgVal56-49	MsgVal48-41	MsgVal40-33
MSGVAL 6 & 5	MsgVal 96-65 (address 0xB8)	MsgVal96-89	MsgVal88-81	MsgVal80-73	MsgVal72-65
MSGVAL 8 & 7	MsgVal 128-97 (address 0xBC)	MsgVal128-121	MsgVal120-113	MsgVal112-105	MsgVal104-97

3. Functional Description

This chapter provides an overview of the CAN module's operating modes and how to use them.

3.1 Software Initialisation

The software initialization is started by setting the bit **Init** in the CAN Control Register, either by software or by a hardware reset, or by going Bus_Off.

While **Init** is set, all message transfers from and to the CAN bus are stopped, the status of the CAN bus output **CAN_TX** is recessive (HIGH). The counters of the EML are unchanged. Setting **Init** does not change any configuration register.

To initialize the CAN Controller, the CPU has to set up the Bit Timing Register and each Message Object. If a Message Object is not needed, it is sufficient to set its **MsgVal** bit to not valid. Otherwise, the whole Message Object has to be initialized.

Access to the Bit Timing Register and to the BRP Extension Register for the configuration of the bit timing is enabled when both bits **Init** and **CCE** in the CAN Control Register are set.

Resetting **Init** (by CPU only) finishes the software initialisation. Afterwards the Bit Stream Processor BSP synchronizes itself to the data transfer on the CAN bus by waiting for the occurrence of a sequence of 11 consecutive recessive bits (= Bus Idle) before it can take part in bus activities and starts the message transfer.

The initialization of the Message Objects is independent of **Init** and can be done on the fly, but the Message Objects should all be configured to particular identifiers or set to not valid before the BSP starts the message transfer.

To change the configuration of a Message Object during normal operation, the CPU has to start by setting **MsgVal** to not valid. When the configuration is completed, **MsgVal** is set to valid again.

3.2 CAN Message Transfer

Once the CAN is initialized and **Init** is reset to zero, the CAN's CAN Core synchronizes itself to the CAN bus and starts the message transfer.

Received messages are stored in their appropriate Message Objects if they pass the Message Handler's acceptance filtering. The whole message including all arbitration bits, DLC and eight data bytes is stored in the Message Object. If the Identifier Mask is used, the arbitration bits which are masked to "don't care" may be overwritten in the Message Object.

The CPU may read or write each message any time via the Interface Registers, the Message Handler guarantees data consistency in case of concurrent accesses.

Messages to be transmitted are updated by the CPU. If a permanent Message Object (arbitration and control bits set up during configuration) exists for the message, only the data bytes are updated and then **TxRqst** bit with **NewDat** bit are set to start the transmission. If several transmit messages are assigned to the same Message Object (when the number of Message Objects is not sufficient), the whole Message Object has to be configured before the transmission of this message is requested.

The transmission of any number of Message Objects may be requested at the same time, they are transmitted subsequently according to their internal priority. Messages may be updated or set to not valid any time, even when their requested transmission is still pending. The old data will be discarded when a message is updated before its pending transmission has started. Depending on the configuration of the Message Object, the transmission of a message may be requested autonomously by the reception of a remote frame with a matching identifier.

3.3 Disabled Automatic Retransmission

According to the CAN Specification (see ISO11898, 6.3.3 Recovery Management), the CAN provides means for automatic retransmission of frames that have lost arbitration or that have been disturbed by errors during transmission. The frame transmission service will not be confirmed to the user before the transmission is successfully completed. By default, this means for automatic retransmission is enabled. It can be disabled to enable the CAN to work within a Time Triggered CAN (TTCAN, see ISO11898-1) environment. The Disabled

Automatic Retransmission mode is enabled by setting the bit **DAR** in the CAN Control Register to one. In this operation mode the programmer has to consider the different behaviour of bits **TxRqst** and **NewDat** in the Control Registers of the Message Buffers:

When a transmission starts bit **TxRqst** of the respective Message Buffer is reset, while bit **NewDat** remains set.

When the transmission completed successfully bit **NewDat** is reset.

When a transmission failed (lost arbitration or error) bit **NewDat** remains set. To restart the transmission the CPU has to set **TxRqst** back to one.

When the host requests the transmission of several messages at the same time, only two of these messages will be transmitted. For all other requested transmit messages, the **TxRqst** bits will be reset, but no transmission will be started, **NewDat** and **IntPnd** will be left unchanged. For the two messages that are transmitted, the **TxRqst** and **NewDat** bits will be reset and, if enabled by **TxIE**, **IntPnd** will be set.

3.4 Test Mode

The Test Mode is entered by setting bit **Test** in the CAN Control Register to one. In Test Mode the bits **Tx1**, **Tx0**, **LBack**, **Silent** and **Basic** in the Test Register are writable. Bit **Rx** monitor the state of pin **CAN_RX** and therefore is only readable. All Test Register functions are disabled when bit Test is reset to zero.

3.5 Silent Mode

The CAN Core can be set in Silent Mode by programming the Test Register bit **Silent** to one.

In Silent Mode, the CAN is able to receive valid data frames and valid remote frames, but it sends only recessive bits on the CAN bus and it cannot start a transmission. If the CAN Core is required to send a dominant bit (ACK bit, overload flag, active error flag), the bit is rerouted internally so that the CAN Core monitors this dominant bit, although the CAN bus may remain in recessive state. The Silent Mode can be used to analyse the traffic on a CAN bus without affecting it by the transmission of dominant bits (Acknowledge Bits, Error Frames). Figure 3-1 shows the connection of signals **CAN_TX** and **CAN_RX** to the CAN Core in Silent Mode.

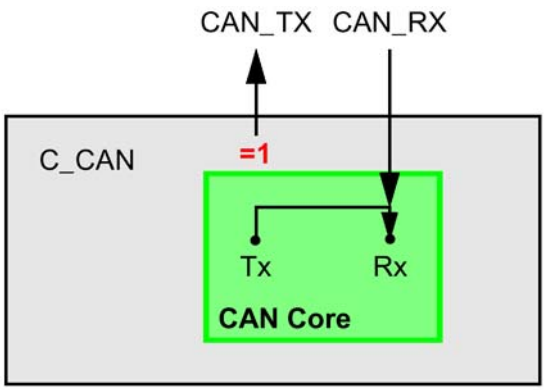


Figure 3-1 CAN Core in Silent Mode

In ISO 11898-1, the Silent Mode is called the Bus Monitoring Mode.

3.6 Loop Back Mode

The CAN Core can be set in Loop Back Mode by programming the Test Register bit **LBack** to one. In Loop Back Mode, the CAN Core treats its own transmitted messages as received messages and stores them (if they pass acceptance filtering) into a Receive Buffer. Figure 3-2 shows the connection of signals **CAN_TX** and

CAN_RX to the CAN Core in Loop Back Mode.

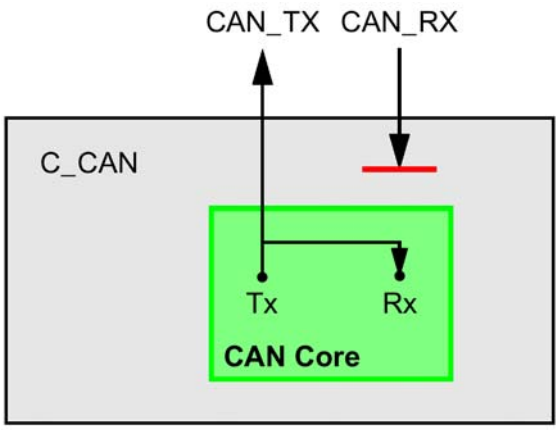


Figure 3-2 CAN Core in Loop Back Mode

This mode is provided for self-test functions. To be independent from external stimulation, the CAN Core ignores acknowledge errors (recessive bit sampled in the acknowledge slot of a data/remote frame) in Loop Back Mode. In this mode the CAN Core performs an internal feedback from its Tx output to its Rx input. The actual value of the **CAN_RX** input pin is disregarded by the CAN Core. The transmitted messages can be monitored at the **CAN_TX** pin.

3.7 Loop Back combined with Silent Mode

It is also possible to combine Loop Back Mode and Silent Mode by programming bits **LBack** and **Silent** to one at the same time. This mode can be used for a “Hot Selftest”, meaning the CAN can be tested without affecting a running CAN system connected to the pins **CAN_TX** and **CAN_RX**. In this mode the **CAN_RX** pin is disconnected from the CAN Core and the **CAN_TX** pin is held recessive. Figure 3-3shows the connection of signals **CAN_TX** and **CAN_RX** to the CAN Core in case of the combination of Loop Back Mode with Silent Mode.

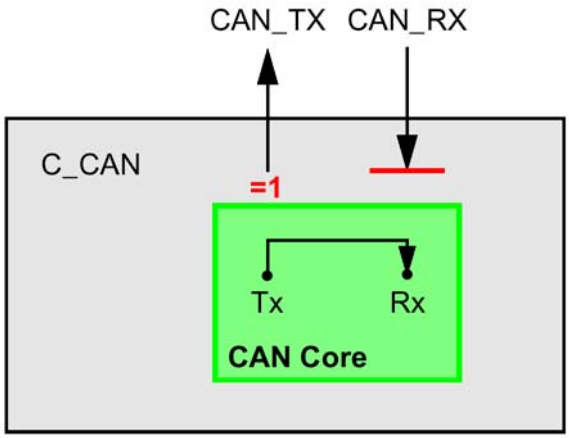


Figure 3-3 CAN Core in Loop Back combined with Silent Mode

3.8 Basic Mode

The CAN Core can be set in Basic Mode by programming the Test Register bit **Basic** to one. In this mode the

CAN module runs without the Message RAM.

The IF1 Registers are used as Transmit Buffer. The transmission of the contents of the IF1 Registers is requested by writing the **Busy** bit of the IF1 Command Request Register to ‘1’. The IF1 Registers are locked while the **Busy** bit is set. The **Busy** bit indicates that the transmission is pending.

As soon as the CAN bus is idle, the IF1 Registers are loaded into the shift register of the CAN Core and the transmission is started. When the transmission has completed, the **Busy** bit is reset and the locked IF1 Registers are released.

A pending transmission can be aborted at any time by resetting the **Busy** bit in the IF1 Command Request Register while the IF1 Registers are locked. If the CPU has reset the Busy bit, a possible retransmission in case of lost arbitration or in case of an error is disabled.

The IF2 Registers are used as Receive Buffer. After the reception of a message the content of the shift register is stored into the IF2 Registers, without any acceptance filtering.

Additionally, the actual contents of the shift register can be monitored during the message transfer. Each time a read Message Object is initiated by writing the **Busy** bit of the IF2 Command Request Register to ‘1’, the content of the shift register is stored in the IF2 Registers.

In Basic Mode the evaluation of all Message Object related control and status bits and of the control bits of the IFx Command Mask Registers is turned off. The message number of the Command request registers is not evaluated. The **NewDat** and **MsgLst** bits of the IF2 Message Control Register retain their function, **DLC3-0** will show the received **DLC**, the other control bits will be read as ‘0’.

3.9 Software control of Pin CAN_TX

Four output functions are available for the CAN transmit pin **CAN_TX**. Additionally to its default function – the serial data output – it can drive the CAN Sample Point signal to monitor CAN_Core’s bit timing and it can drive constant dominant or recessive values. The last two functions, combined with the readable CAN receive pin **CAN_RX**, can be used to check the CAN bus’ physical layer.

The output mode of pin **CAN_TX** is selected by programming the Test Register bits **Tx1** and **Tx0** as described in section 2.3 “CAN Protocol Related Registers” on page 698.

The three test functions for pin **CAN_TX** interfere with all CAN protocol functions. **CAN_TX** must be left in its default function when CAN message transfer or any of the test modes Loop Back Mode, Silent Mode, or Basic Mode are selected.

4. CAN Application

This section describes how to use the CAN module in the application

4.1 Management of Message Objects

The configuration of the Message Objects in the Message RAM will (with the exception of the bits **MsgVal**, **NewDat**, **IntPnd**, and **TxRqst**) not be affected by resetting the chip. All the Message Objects must be initialized by the CPU or they must be not valid (**MsgVal** = '0') and the bit timing must be configured before the CPU clears the **Init** bit in the CAN Control Register.

The configuration of a Message Object is done by programming Mask, Arbitration, Control and Data field of one of the two interface register sets to the desired values. By writing to the corresponding IFx Command Request Register, the IFx Message Buffer Registers are loaded into the addressed Message Object in the Message RAM.

When the **Init** bit in the CAN Control Register is cleared, the CAN Protocol Controller state machine of the CAN_Core and the Message Handler State Machine control the CAN's internal data flow. Received messages that pass the acceptance filtering are stored into the Message RAM, messages with pending transmission request are loaded into the CAN_Core's Shift Register and are transmitted via the CAN bus.

The CPU reads received messages and updates messages to be transmitted via the IFx Interface Registers. Depending on the configuration, the CPU is interrupted on certain CAN message and CAN error events.

4.2 Message Handler State Machine

The Message Handler controls the data transfer between the Rx/Tx Shift Register of the CAN Core, the Message RAM and the IFx Registers.

- The Message Handler FSM controls the following functions:
- Data Transfer from IFx Registers to the Message RAM
- Data Transfer from Message RAM to the IFx Registers
- Data Transfer from Shift Register to the Message RAM
- Data Transfer from Message RAM to Shift Register
- Data Transfer from Shift Register to the Acceptance Filtering unit
- Scanning of Message RAM for a matching Message Object
- Handling of**TxRqst** flags.
- Handling of interrupts.

4.3 Data Transfer from/to Message RAM

When the CPU initiates a data transfer between the IFx Registers and Message RAM, the Message Handler sets an internal busy signal which delays a consecutive access. After the transfer has completed, the busy signal is set back and the consecutive access is executed.

The respective Command Mask Register specifies whether a complete Message Object or only parts of it will be transferred. Due to the structure of the Message RAM it is not possible to write single bits/bytes of one Message Object, it is always necessary to write a complete Message Object into the Message RAM. Therefore the data transfer from the IFx Registers to the Message RAM requires a read-modify-write cycle. First parts of the Message Object that are not to be changes are read from the Message RAM and then the complete contents of the Message Buffer Registers are written into the Message Object.

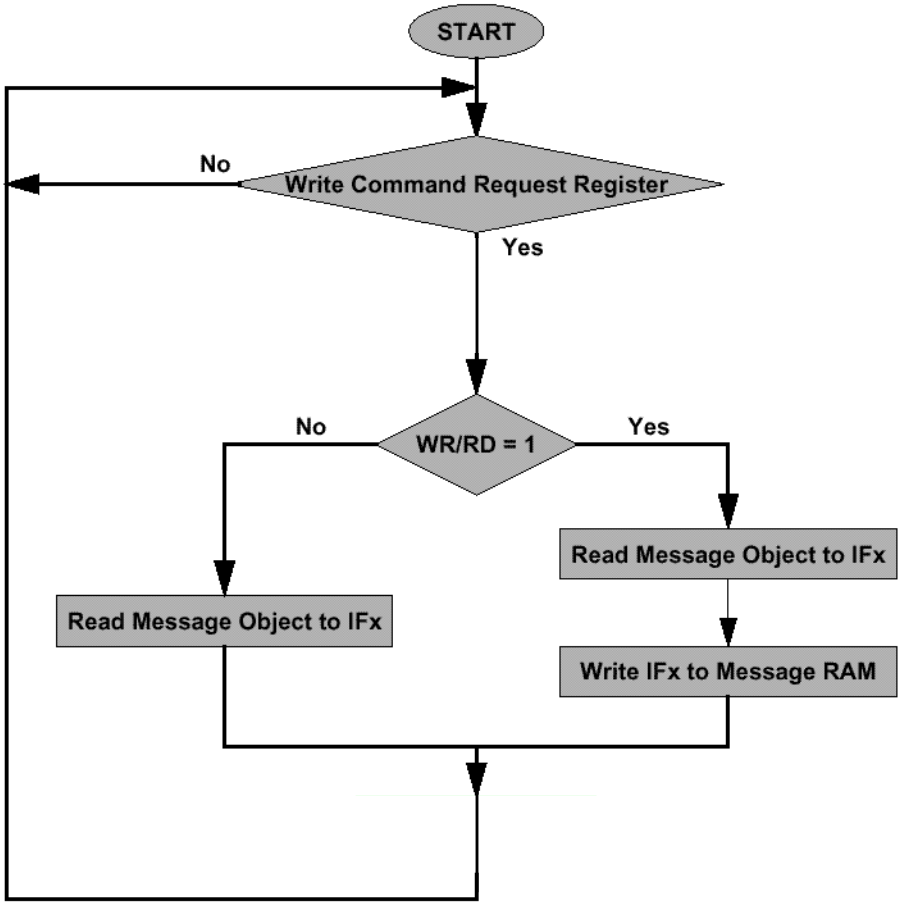


Figure 4-1 Data Transfer between IFx Registers and Message RAM

After the partial write of a Message Object, that Message Buffer Registers that are not selected in the Command Mask Register will set to the actual contents of the selected Message Object.

After the partial read of a Message Object, that Message Buffer Registers that are not selected in the Command Mask Register will be left unchanged.

4.4 Transmission of Messages

If the shift register of the CAN Core cell is ready for loading and if there is no data transfer between the IFx Registers and Message RAM, the **MsgVal** bits in the Message Valid Register and the **TxRqst** bits in the Transmission Request Register are evaluated. The valid Message Object with the highest priority pending transmission request is loaded into the shift register by the Message Handler and the transmission is started. The Message Object's **NewDat** bit is reset.

After a successful transmission and if no new data was written to the Message Object (**NewDat** = '0') since the start of the transmission, the **TxRqst** bit will be reset. If **TxIE** is set, **IntPnd** will be set after a successful transmission. If the CAN has lost the arbitration or if an error occurred during the transmission, the message will be retransmitted as soon as the CAN bus is free again. If meanwhile the transmission of a message with higher priority has been requested, the messages will be transmitted in the order of their priority.

4.5 Acceptance Filtering of Received Messages

When the arbitration and control field (Identifier + IDE + RTR + DLC) of an incoming message is completely shifted into the Rx/Tx Shift Register of the CAN Core, the Message Handler FSM starts the scanning of the Message RAM for a matching valid Message Object.

To scan the Message RAM for a matching Message Object, the Acceptance Filtering unit is loaded with the arbitration bits from the CAN Core shift register. Then the arbitration and mask fields (including **MsgVal**, **UMask**, **NewDat**, and **EoB**) of Message Object 1 are loaded into the Acceptance Filtering unit and compared with the arbitration field from the shift register. This is repeated with each following Message Object until a matching Message Object is found or until the end of the Message RAM is reached.

If a match occurs, the scanning is stopped and the Message Handler FSM proceeds depending on the type of frame (Data Frame or Remote Frame) received.

4.6 Reception of Data Frame

The Message Handler FSM stores the message from the CAN Core shift register into the respective Message Object in the Message RAM. Not only the data bytes, but all arbitration bits and the Data Length Code are stored into the corresponding Message Object. This is implemented to keep the data bytes connected with the identifier even if arbitration mask registers are used.

The **NewDat** bit is set to indicate that new data (not yet seen by the CPU) has been received. The CPU should reset **NewDat** when it reads the Message Object. If at the time of the reception the **NewDat** bit was already set, **MsgLst** is set to indicate that the previous data (supposedly not seen by the CPU) is lost. If the **RxIE** bit is set, the **IntPnd** bit is set, causing the Interrupt Register to point to this Message Object.

The **TxRqst** bit of this Message Object is reset to prevent the transmission of a Remote Frame, while the requested Data Frame has just been received.

4.7 Reception of Remote Frame

When a Remote Frame is received, three different configurations of the matching Message Object have to be considered:

1) **Dir** = ‘1’ (direction = transmit), **RmtEn** = ‘1’, **UMask** = ‘1’ or ’0’

At the reception of a matching Remote Frame, the **TxRqst** bit of this Message Object is set. The rest of the Message Object remains unchanged.

2) **Dir** = ‘1’ (direction = transmit), **RmtEn** = ‘0’, **UMask** = ’0’

At the reception of a matching Remote Frame, the **TxRqst** bit of this Message Object remains unchanged; the Remote Frame is ignored.

3) **Dir** = ‘1’ (direction = transmit), **RmtEn** = ‘0’, **UMask** = ’1’

At the reception of a matching Remote Frame, the **TxRqst** bit of this Message Object is reset. The arbitration and control field (Identifier + IDE + RTR + DLC) from the shift register is stored into the Message Object in the Message RAM and the **NewDat** bit of this Message Object is set. The data field of the Message Object remains unchanged; the Remote Frame is treated similar to a received Data Frame.

4.8 Receive / Transmit Priority

The receive/transmit priority for the Message Objects is attached to the message number. Message Object 1 has the highest priority, while Message Object 32 (the highest implemented message object number) has the lowest priority. If more than one transmission request is pending, they are serviced due to the priority of the corresponding Message Object.

4.9 Configuration of a Transmit Object

Figure 4-2 shows how a Transmit Object should be initialised.

MsgVal	Arb	Data	Mask	EoB	Dir	NewDat	MsgLst	RxIE	TxIE	IntPnd	RmtEn	TxRqst
1	appl.	appl.	appl.	1	1	0	0	0	appl.	0	appl.	0

Figure 4-2 Initialisation of a Transmit Object

The Arbitration Registers (**ID28-0** and **Xtd** bit) are given by the application. They define the identifier and type of the outgoing message. If an 11-bit Identifier (“Standard Frame”) is used, it is programmed to **ID28** - **ID18**, **ID17** - **ID0** can then be disregarded.

If the **TxIE** bit is set, the **IntPnd** bit will be set after a successful transmission of the Message Object.

If the **RmtEn** bit is set, a matching received Remote Frame will cause the **TxRqst** bit to be set; the Remote Frame will autonomously be answered by a Data Frame.

The Data Registers (**DLC3-0**, **Data0-7**) are given by the application, **TxRqst** and **RmtEn** may not be set before the data is valid.

The Mask Registers (**Msk28-0**, **UMask**, **MXtd**, and **MDir** bits) may be used (**UMask**=’1’) to allow groups of Remote Frames with similar identifiers to set the **TxRqst** bit. For details see section 4.4 “Transmission of Messages” on page 728., handle with care. The **Dir** bit should not be masked.

4.10 Updating a Transmit Object

The CPU may update the data bytes of a Transmit Object any time via the IFx Interface registers, neither **MsgVal** nor **TxRqst** have to be reset before the update.

(Note) When the lowest priority message buffer is used for transmission, setting **TxRqst** = "0" may cause a delay of transmission, when **TxRqst** is set to "1" again. Please refer to the note at the description of **TxRqst** in section 22.8.2 "Transmission Request Registers (**TREQR1n**, **TREQR2n**)"

Even if only a part of the data bytes are to be updated, all four bytes of the corresponding IFx Data A Register or IFx Data B Register have to be valid before the content of that register is transferred to the Message Object. Either the CPU has to write all four bytes into the IFx Data Register or the Message Object is transferred to the IFx Data Register before the CPU writes the new data bytes.

When only the (eight) data bytes are updated, first 0x0087 is written to the Command Mask Register and then the number of the Message Object is written to the Command Request Register, concurrently updating the data bytes and setting **TxRqst**.

To prevent the reset of **TxRqst** at the end of a transmission that may already be in progress while the data is updated, **NewDat** has to be set together with **TxRqst**. For details see section section 4.4 “Transmission of Messages” on page 728.

When **NewDat** is set together with **TxRqst**, **NewDat** will be reset as soon as the new transmission has started.

4.11 Configuration of a Receive Object

Figure 4-3 shows how a Transmit Object should be initialised.

MsgVal	Arb	Data	Mask	EoB	Dir	NewDat	MsgLst	RxIE	TxIE	IntPnd	RmtEn	TxRqst

1	appl.	appl.	appl.	1	0	0	0	appl.	0	0	0	0

Figure 4-3 Initialisation of a Receive Object

The Arbitration Registers (**ID28-0** and **Xtd** bit) are given by the application. They define the identifier and type of accepted received messages. If an 11-bit Identifier (“Standard Frame”) is used, it is programmed to **ID28** - **ID18**, **ID17** - **ID0** can then be disregarded. When a Data Frame with an 11-bit Identifier is received, **ID17** - **ID0** will be set to ‘0’.

If the **RxIE** bit is set, the **IntPnd** bit will be set when a received Data Frame is accepted and stored in the Message Object.

The Data Length Code (**DLC3-0**) is given by the application. When the Message Handler stores a Data Frame in the Message Object, it will store the received Data Length Code and eight data bytes. If the Data Length Code is less than 8, the remaining bytes of the Message Object will be overwritten by **non specified values**.

The Mask Registers (**Msk28-0**, **UMask**, **MXtd**, and **MDir** bits) may be used (**UMask**=’1’) to allow groups of Data Frames with similar identifiers to be accepted. For details see section 4.6 “Reception of Data Frame” on page 729.. The **Dir** bit should not be masked in typical applications.

4.12 Handling of Received Messages

The CPU may read a received message any time via the IFx Interface registers, the data consistency is guaranteed by the Message Handler state machine.

Typically the CPU will write first 0x007F to the Command Mask Register and then the number of the Message Object to the Command Request Register. That combination will transfer the whole received message from the Message RAM into the Message Buffer Register. Additionally, the bits **NewDat** and **IntPnd** are cleared in the Message RAM (not in the Message Buffer).

If the Message Object uses masks for acceptance filtering, the arbitration bits show which of the matching messages have been received.

The actual value of **NewDat** shows whether a new message has been received since last time this Message Object was read. The actual value of **MsgLst** shows whether more than one message has been received since last time this Message Object was read. **MsgLst** will not be automatically reset.

By means of a Remote Frame, the CPU may request another CAN node to provide new data for a receive object. Setting the **TxRqst** bit of a receive object will cause the transmission of a Remote Frame with the receive object’s identifier. This Remote Frame triggers the other CAN node to start the transmission of the matching Data Frame. If the matching Data Frame is received before the Remote Frame could be transmitted, the **TxRqst** bit is automatically reset.

4.13 Configuration of a FIFO Buffer

With the exception of the **EoB** bit, the configuration of Receive Objects belonging to a FIFO Buffer is the same as the configuration of a (single) Receive Object, see section 4.11 “Configuration of a Receive Object” on page 730.

To concatenate two or more Message Objects into a FIFO Buffer, the identifiers and masks (if used) of these Message Objects have to be programmed to matching values. Due to the implicit priority of the Message Objects, the Message Object with the lowest number will be the first Message Object of the FIFO Buffer. The **EoB** bit of all Message Objects of a FIFO Buffer except the last have to be programmed to zero. The **EoB** bits of the last Message Object of a FIFO Buffer is set to one, configuring it as the **E**nd **o**f the **B**lock.

4.14 Reception of Messages with FIFO Buffers

Received messages with identifiers matching to a FIFO Buffer are stored into a Message Object of this FIFO Buffer starting with the Message Object with the lowest message number.

When a message is stored into a Message Object of a FIFO Buffer the **NewDat** bit of this Message Object is set. By setting **NewDat** while **EoB** is 0 the Message Object is locked for further write accesses by the

Message Handler until the CPU has written the **NewDat** bit back to 0.

Messages are stored into a FIFO Buffer until the last Message Object of this FIFO Buffer is reached. If none of the preceding Message Objects is released by writing **NewDat** to 0, all further messages for this FIFO Buffer will be written into the last Message Object of the FIFO Buffer and therefore overwrite previous messages.

4.15 Reading from a FIFO Buffer

When the CPU transfers the contents of Message Object to the IFx Message Buffer registers by writing its number to the IFx Command Request Register, the corresponding Command Mask Register should be programmed the way that bits **NewDat** and **IntPnd** are reset to zero (**TxRqst/NewDat** = '1' and **ClrIntPnd** = '1'). The values of these bits in the Message Control Register always reflect the status before resetting the bits.

To assure the correct function of a FIFO Buffer, the CPU should read out the Message Objects starting at the FIFO Object with the lowest message number.

Figure 4-4 shows how a set of Message Objects which are concatenated to a FIFO Buffer can be handled by the CPU.

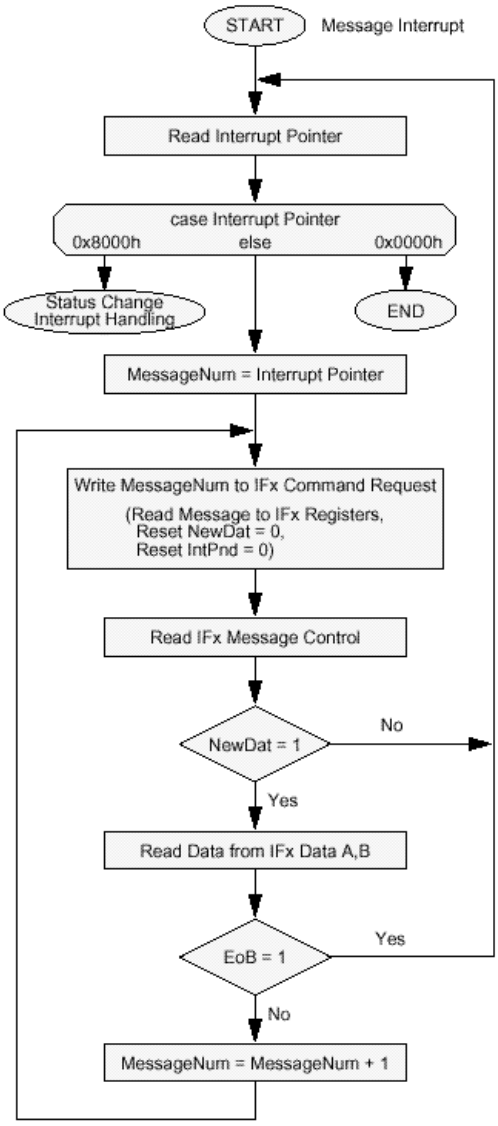


Figure 4-4 CPU Handling of a FIFO Buffer

4.16 Handling of Interrupts

If several interrupts are pending, the CAN Interrupt Register will point to the pending interrupt with the highest priority, disregarding their chronological order. An interrupt remains pending until the CPU has cleared it.

The Status Interrupt has the highest priority. Among the message interrupts, the Message Object' s interrupt priority decreases with increasing message number.

A message interrupt is cleared by clearing the Message Object's IntPnd bit. The Status Interrupt is cleared by reading the Status Register.

The interrupt identifier **IntId** in the Interrupt Register indicates the cause of the interrupt. When no interrupt is pending, the register will hold the value zero. If the value of the Interrupt Register is different from zero, then there is an interrupt pending and, if **IE** is set, the interrupt line to the CPU is active. The interrupt line remains active until the Interrupt Register is back to value zero (the cause of the interrupt is reset) or until **IE** is reset.

The value 0x8000 indicates that an interrupt is pending because the CAN Core has updated (not necessarily changed) the Status Register (Error Interrupt or Status Interrupt). This interrupt has the highest priority. The CPU can update (reset) the status bits **RxOk**, **TxOk** and **LEC**, but a write access of the CPU to the Status Register can never generate or reset an interrupt.

All other values indicate that the source of the interrupt is one of the Message Objects, **IntId** points to the pending message interrupt with the highest interrupt priority.

The CPU controls whether a change of the Status Register may cause an interrupt (bits **EIE** and **SIE** in the CAN Control Register) and whether the interrupt line becomes active when the Interrupt Register is different from zero (bit **IE** in the CAN Control Register). The Interrupt Register will be updated even when **IE** is reset.

The CPU has two possibilities to follow the source of a message interrupt. First it can follow the **IntId** in the Interrupt Register and second it can poll the Interrupt Pending Register (see section 2.6 Message Handler Registers (Page No.716).

An interrupt service routine reading the message that is the source of the interrupt may read the message and reset the Message Object's **IntPnd** at the same time (bit **ClrIntPnd** in the Command Mask Register). When **IntPnd** is cleared, the Interrupt Register will point to the next Message Object with a pending interrupt.

4.17 Bit Time and Bit Rate

The timing parameter of the bit time (i.e. the reciprocal of the bit rate) can be configured individually for each CAN node, creating a common bit rate even though the CAN nodes' oscillator periods (f_{osc}) may be different.

The frequencies of these oscillators are not absolutely stable, small variations are caused by changes in temperature or voltage and by deteriorating components. As long as the variations remain inside a specific oscillator tolerance range (df), the CAN nodes are able to compensate for the different bit rates by resynchronising to the bit stream.

According to the CAN specification, the bit time is divided into four segments (see Figure 4-5). The Synchronisation Segment, the Propagation Time Segment, the Phase Buffer Segment 1, and the Phase Buffer Segment 2. Each segment consists of a specific, programmable number of time quanta (see Table 1). The length of the time quantum (t_q) , which is the basic time unit of the bit time, is defined by the CAN controller's system clock f_{sys} and the Baud Rate Prescaler (BRP) : t_q = BRP / f_{sys}. The CAN's system clock f_{sys} is the frequency of its **CAN_CLK** input.

The Synchronisation Segment Sync_Seg is that part of the bit time where edges of the CAN bus level are expected to occur; the distance between an edge that occurs outside of Sync_Seg and the Sync_Seg is called the phase error of that edge. The Propagation Time Segment Prop_Seg is intended to compensate for the physical delay times within the CAN network. The Phase Buffer Segments Phase_Seg1 and Phase_Seg2 surround the Sample Point. The (Re-)Synchronisation Jump Width (SJW) defines how far a resynchronisation may move the Sample Point inside the limits defined by the Phase Buffer Segments to compensate for edge phase errors.

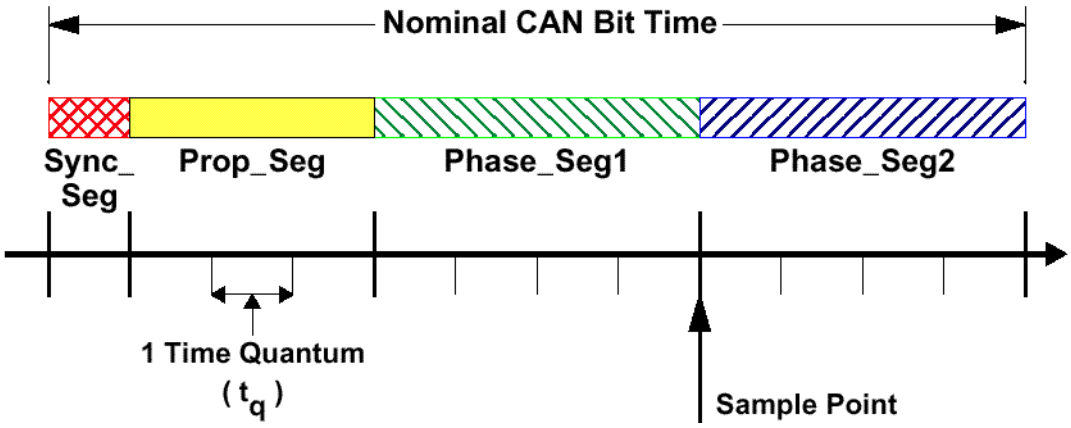


Figure 4-5 Bit Timing

Parameter	Range	Remark
BRP	[1..32]	defines the legth of the time quantum tq
Sync_Seg	1 tq	fixed length, synchronisation of us into system clock
Prop_Seg	[1..8] tq	compensates for the physical delay times
Phase_Seg1	[1..8] tq	may be lengthened temporarily by synchronisation
Phase_Seg2	[1..8] tq	may be shortened temporarily by synchronisation
SJW	[1..4] tq	may not be longer than either Phase Buffer Segment

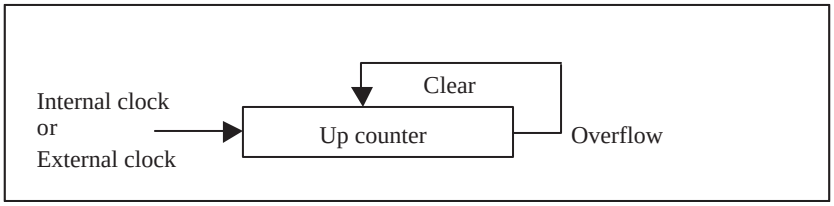
Table 4-1 Parameters of the CAN Bit Time

(Note) This table describes the minimum programmable ranges required by the CAN protocol.

Chapter 35 Free-Run Timer

1. Overview

The free-run timer consists of a 16-bit timer (up counter) and control circuits.
The free-run timer can be used with the input capture and the output compare.

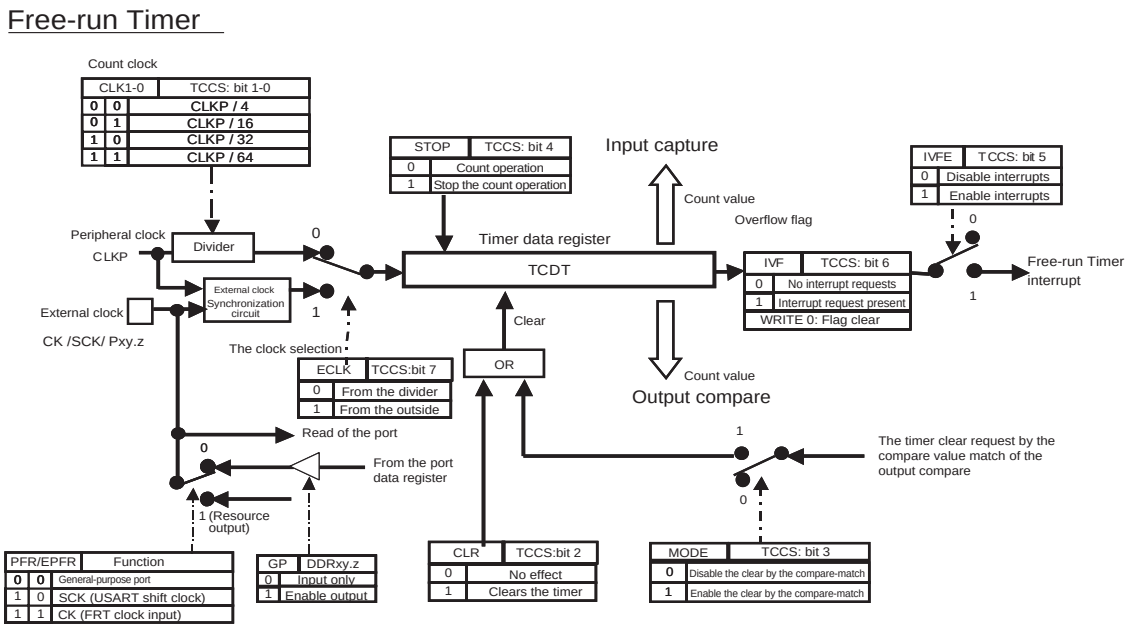


2. Features

- Format: 16-bit up counter
- Quantity: 8 (free-run timer 0 up to free-run timer 7)
- Clock source: 4 internal clocks (1/4, 1/16, 1/32, and 1/64 of CLKP)
- External clock (CK)
- Clear factor of the count:
  - Software
  - Reset
  - Compare-match (match of the compare-register value and the count value of the free-run timer)
- Operation start/stop: operations can be started/stopped with software.
- Interrupt:
  - Overflow interrupt
  - An interrupt generated when the compare clear register value and the count value of the free-run timer match.
- Count value: Readable/writable (write is only possible when the counting stops)
- Others: Operates from immediately after reset.
- Free Run Timer to ICU/OCU mapping:
  - Free-run timer 0 and input capture 0/1 co-operate
  - Free-run timer 1 and input capture 2/3 co-operate
  - Free-run timer 2 and output compare 0/1 co-operate
  - Free-run timer 3 and output compare 2/3 co-operate
  - Free-run timer 4 and input capture 4/5 co-operate
  - Free-run timer 5 and input capture 6/7 co-operate
  - Free-run timer 6 and output compare 4/5 co-operate
  - Free-run timer 7 and output compare 6/7 co-operate

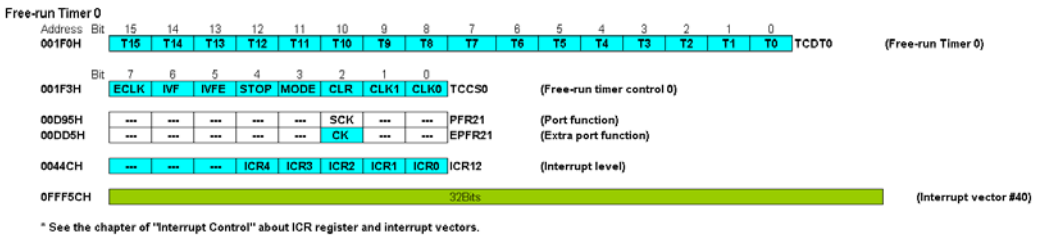
3. Configuration Diagram

Figure 3-1 Configuration Diagram



Notes: When using the input/output (SCK), the external clock (CK) cannot be used because the port is shared.

Figure 3-2 Register List



Note: See “Chapter 24 Interrupt Control (Page No.315)” about ICR register and interrupt vectors.

4. Registers

4.1 TCCS: Timer Control Register

A register for controlling the operation of the free-run timer.

- **TCCS0 (free-run timer 0): Address 01F3h (access: Byte, Half-word, Word)**
- **TCCS1 (free-run timer 1): Address 01F7h (access: Byte, Half-word, Word)**
- **TCCS2 (free-run timer 2): Address 01FBh (access: Byte, Half-word, Word)**
- **TCCS3 (free-run timer 3): Address 01FFh (access: Byte, Half-word, Word)**
- **TCCS4 (free-run timer 4): Address 02F3h (access: Byte, Half-word, Word)**
- **TCCS5 (free-run timer 5): Address 02F7h (access: Byte, Half-word, Word)**
- **TCCS6 (free-run timer 6): Address 02FBh (access: Byte, Half-word, Word)**
- **TCCS7 (free-run timer 7): Address 02FFh (access: Byte, Half-word, Word)**

7	6	5	4	3	2	1	0	bit
ECLK	IVF	IVFE	STOP	MODE	CLR	CLK1	CLK0	
0	0	0	0	0	0	0	0	Initial value
R/W	R (RM1), W	R/W	R/W	R/W	R/W	R/W	R/W	Attribute

(About attributes, see “Meaning of Bit Attribute Symbols (Page No.10)”.)

- bit7: Select the count clock

ECLK	Select the count clock
0	Internal clock (the peripheral clock divided by n)
1	External clock (CK pin)

  - When you change the setting of the count clock selection bit, do so when other peripheral modules (the output compare, input capture, etc.) using the output of the free-run timer are stopped.
  - When using the external clock, the period of the external clock must be more than double of the peripheral clock (CLKP). When using the output compare, in order to allow the compare-match output and interrupt generation, the external clock input of at least 1 clock is required after the compare-match.
- bit6: Interrupt request flag

IVF	Status	
	Read	Write
0	No interrupt request present	Clear the flag (IVF).
1	Interrupt request present (Overflow or compare-match)	No effect on operation

  - When the count value of the free-run timer overflows, or the clear mode bit (MODE) is “1”, the interrupt request flag is set to “1” if the count values of the free-run timer and the compare register (OCCP) match and the counter is cleared.
  - To enable the interrupt request, the interrupt enabling bit must be set to do so (IVFE=“1”).
  - When the interrupt request flag is set to “1”, and “0” is written at the same time, the interrupt request flag is set to “1”. (Setting the flag has priority.)
- bit5: Enable interrupt requests

IVFE	Operation
0	Disable interrupts
1	Enable interrupts

  - When the interrupt request enabling bit is set to “1”, the interrupt request (IVF) is enabled.

- bit4: Stop counting

STOP	Operation
0	Enable counting
1	Disable count (stop)

  - When the count stop bit is set to “1”, the free-run timer stops.
  - When the output compare is being used, if the free-run timer stops, the output compare also stops.

- bit3: Clear mode

MODE	Clear mode
0	Clear the free-run timer by the reset and the clear bit (CLR).
1	Clear the free-run timer by the match with the reset, the clear bit (CLR), and the compare register value of the output compare (OCCP).

  - Set the clear mode of the free-run timer.
  - If the clear mode bit is set to “1”, when the count value of the free-run timer and the compare-register value (OCCP) match, the count value of the free-run timer is cleared to “0000h”.
  - The reset and writing “1” to the clear bit (CLR) cause to clear the count value of the free-run timer to “0000h”, regardless of the setting of the clear mode bit.
  - The count value of the free-run timer is only cleared when the free-run timer is running. When the free-run timer is stopped, clear it by writing “0000h” to the timer data register (TCDT).
- bit2: Clear

CLR	Operation
0	No effect on operation
1	Clear the free-run timer.

  - When the clear bit is set to “1”, the count value of the free-run timer is cleared to “0000h”. The clear bit is read as “1” until the free-run timer is completely cleared.
    When the free-run timer is completely cleared, the clear bit is also cleared to “0”.
  - When the clear operation of the free-run timer and writing “1” to the clear bit occurs at the same time, the clear bit keeps “1”, and after the next time the free-run timer is cleared, it is cleared.
- bit1-bit0: Count clock division ratio selection (when the internal clock is selected)

CLK1	CLK0	The division ratio of the count clock
0	0	Peripheral clock (CLKP) divided by 4
0	1	Peripheral clock (CLKP) divided by 16
1	0	Peripheral clock (CLKP) divided by 32
1	1	Peripheral clock (CLKP) divided by 64

  - Select the division ratio of the count clock of the free-run timer.
  - Change the division ratio when the setting of the count clock division ratio selection bit is changed. When the internal clock is selected as the count clock of the free-run timer (count clock selection bit ECLK=“0”), change the setting when other peripheral modules (output compare, input capture, etc.) using the output of the free-run timer are stopped.

4.2 TCDT: Timer Data Register

This register can read 16-bit free-run timer count values.

- **TCDT0 (free-run timer 0): Address 01F0h (access: Half-word, Word)**
- **TCDT1 (free-run timer 1): Address 01F4h (access: Half-word, Word)**
- **TCDT2 (free-run timer 2): Address 01F8h (access: Half-word, Word)**
- **TCDT3 (free-run timer 3): Address 01FCh (access: Half-word, Word)**
- **TCDT4 (free-run timer 4): Address 02F0h (access: Half-word, Word)**
- **TCDT5 (free-run timer 5): Address 02F4h (access: Half-word, Word)**
- **TCDT6 (free-run timer 6): Address 02F8h (access: Half-word, Word)**
- **TCDT7 (free-run timer 7): Address 02FCh (access: Half-word, Word)**

15	14	13	12	11	10	9	8	bit
T15	T14	T13	T12	T11	T10	T9	T8	
0	0	0	0	0	0	0	0	Initial value
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Attribute

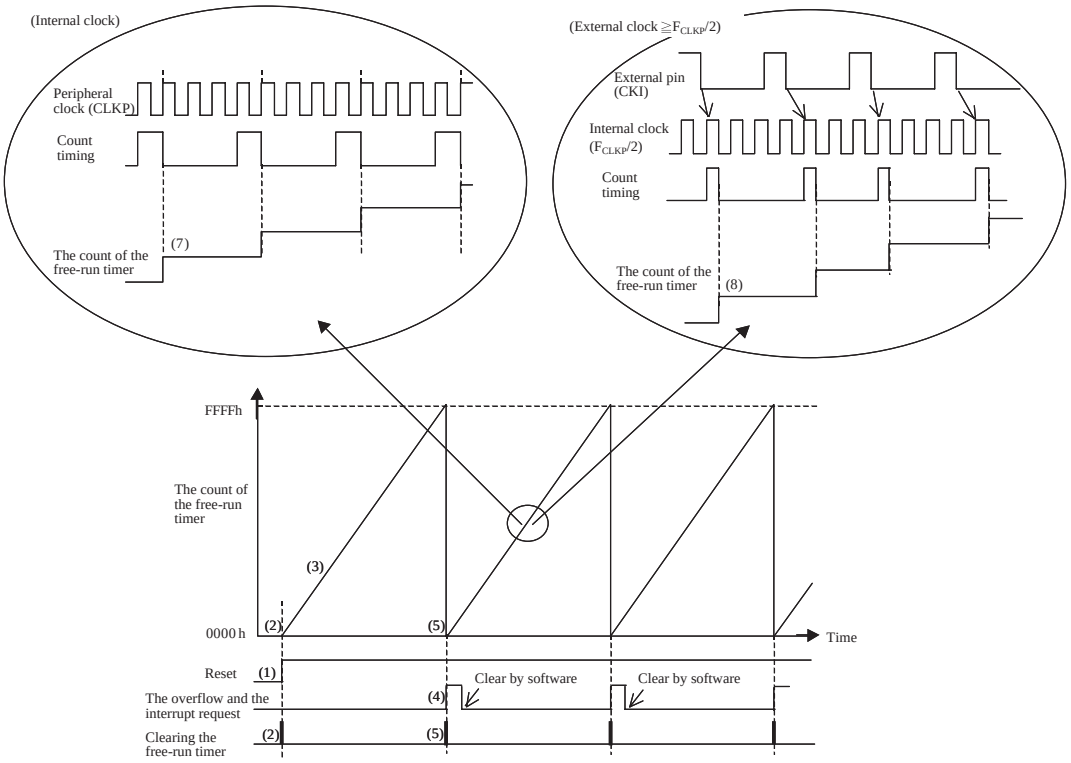
7	6	5	4	3	2	1	0	bit
T7	T6	T5	T4	T3	T2	T1	T0	
0	0	0	0	0	0	0	0	Initial value
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Attribute

(About attributes, see “Meaning of Bit Attribute Symbols (Page No.10)”.)

- When the timer data register is read, the count value of the free-run timer is also obtained.
- By writing to the timer data register, the timer value can be written in the free-run timer. When it is written, make sure that the free-run timer is in the idle state (the count stop bit (TCCS.STOP= 1”)).

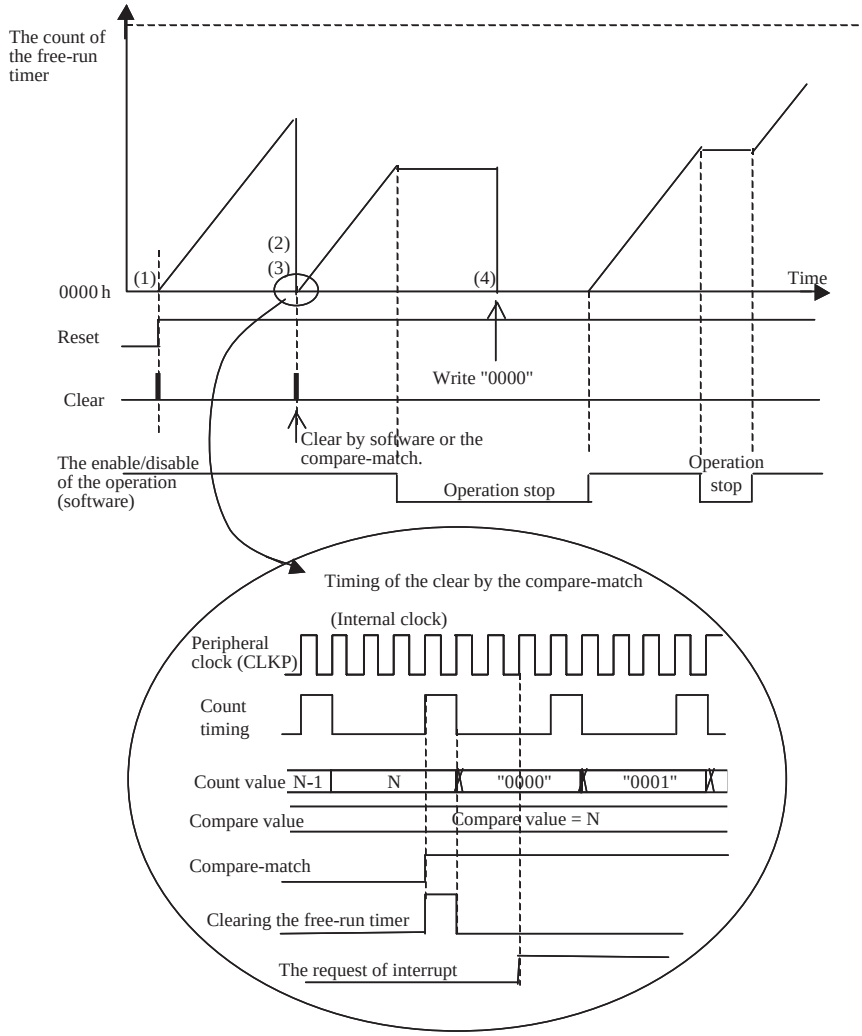
5. Operation

5.1 Count Operation of the Free-run Timer



(1) Reset
(2) Clearing of the free-run timer by reset. (Count value “0000”)
(3) Count-up of the free-run timer
(4) Overflow and interrupt of the free-run timer.
(5) Clearing of the free-run timer by overflow. (Count value “0000”)
(6) Repeat (3) to (5)
(7) The free-run timer counts up at the count clock (the internal clock divided by n).
(8) The free-run timer counts up at the count clock (the external clock synchronized with the internal clock).

5.2 Various Clear Operations of the Free-run Timer



Clear operations of the free-run timer (4 types)

(1) Reset

(2) Clear by software

(3) Clear by the compare-match

(4) Writing “0000”

6. Setting

Table 6-1 Setting Required in Order to Use the Free-run Timer

Setting	Setting Registers	Setting Procedures *
Setting of the initialization conditions of the timer	Timer control register (TCCS0-TCCS7)	See 7.4
Setting of the count clock Selection of the internal clock		See 7.1
Selection of the external clock		See 7.2
Start the count operation		See 7.3
In the case of the external clock Set the clock input pin (CK) as the input.	Port function register (PFRxy.z) Extra port function register (EPFRxy.z)	See 7.2

*: For the setting procedure, refer to the section indicated by the number.

Table 6-2 Setting Required to Enable the Free-run Timer Interrupt

Setting	Setting Registers	Setting Procedures *
Setting of the free-run timer interrupt vector, and the free-run timer interrupt level	See "Chapter 24 Interrupt Control (Page No.315)".	See 7.5
Setting of the free-run timer interrupt Clearing interrupt requests Enabling interrupt requests	Timer control register (TCCS0-TCCS7)	See 7.7

*: For the setting procedure, refer to the section indicated by the number.

Table 6-3 Setting Required to Stop the Free-run Timer

Setting	Setting Registers	Setting Procedures *
Setting of the free-run timer stop bit	Timer control register (TCCS0-TCCS7)	See 7.8

*: For the setting procedure, refer to the section indicated by the number.

7. Q & A

7.1 What are the types of the internal clock, and how do I select?

There are 4 types of internal clocks, and these are set by the clock selection bit (TCCS.ECLK) and the count clock bit (TCCS.CLK [1:0]).

Internal clock	Setting		Count period	
	Clock selection Bit (ECLK)	Count clock bit (CLK [1:0])	F_{CLKP} = 32MHz	F_{CLKP} = 16MHz
To select $F_{CLKP}/4$	Set to “0”	Sets to “00”	125 ns	250 ns
To select $F_{CLKP}/16$	Set to “0”	Set to “01”	0.5 µs	1 µs
To select $F_{CLKP}/32$	Set to “0”	Set to “10”	1 µs	2 µs
To select $F_{CLKP}/64$	Set to “0”	Set to “11”	2 µs	4 µs

7.2 How do I select the external clock?

Set with clock selection bits (TCCS.ECLK), data direction bits, and (extra) port function bits.

To use the external clock input	Setting			Pins	Count Cycle
Free-run Timer 0-7	The clock selection bit (ECLK) to “1”	The port function bit (PFRxy.z) to “1”	The extra port function bit (EPFRxy.z} to “1”	CK0-CK7	Over $2/F_{CLKP}$

7.3 How do I enable / disable the count operation of the free-run timer?

Set with count operation bits (TCCS.STOP).

Operation	Count operation bit (STOP)
To enable the free-run timer	Set to “0”
To stop the free-run timer	Set to “1”

7.4 How do I clear the free-run timer?

You can clear the free-run timer by performing the following operations:

- Set with clear bit (TCCS.CLR).

Operation	Clear bit (CLR)
To clear the free-run timer	Write “1”

- How to clear the free-run timer when the free-run timer value and the compare-register value match

  Set with the timer initialization condition bit (TCCS.MODE).

Operation	Timer initialization condition bit (MODE)
To clear the free-run timer at the compare-match	Set to “1”

  The setting of the output compare is also required. (See “Chapter 37 Output Compare (Page No.761)”.)

- Reset.

  When you reset (the INIT pin input, the watchdog reset, the software reset), the free-run timer is cleared.

- Write “0000_H” while the free-run timer is stopped.

  The count value will be set to “0000_H”, when “0000_H” is written while the free-run timer is stopped.

- With the overflow of the free-run timer, the count value returns to “0000_H”.

7.5 What interrupt registers are used?

Setting of the free-run timer interrupt vector and the free-run timer interrupt level

The relationship among the free-run timer number, interrupt levels and vectors is shown in the table below.

See “Chapter 24 Interrupt Control (Page No.315)” about the details of interrupt levels and interrupt vectors.

Number	Interrupt Vectors (default)	Interrupt level setting bits (ICR[4:0])
Free-run Timer 0	#40 Address: 0FFF5Ch	Interrupt level register (ICR12) Address: 0044Ch
Free-run Timer 1	#41 Address: 0FFF58h	
Free-run Timer 2	#42 Address: 0FFF54h	Interrupt level register (ICR13) Address: 0044Dh
Free-run Timer 3	#43 Address: 0FFF50h	
Free-run Timer 4	#44 Address: 0FFF4Ch	Interrupt level register (ICR14) Address: 0044Eh
Free-run Timer 5	#45 Address: 0FFF48h	
Free-run Timer 6	#46 Address: 0FFF44h	Interrupt level register (ICR15) Address: 0044Fh
Free-run Timer 7	#47 Address: 0FFF40h	

Since the interrupt request flag (TCCS.IVF) is not cleared automatically, make sure to clear it with software before returning from the interrupt process. (Write “0” in the IVF bit)

7.6 Interrupt Types

There is only one type of interrupt, and it is generated at the overflow of the free-run timer. (Selection is not required)

7.7 How do I enable interrupts?

Enable interrupt requests, interrupt request flag

Use interrupt request enable bit (TCCS.IVFE) to enable interrupts.

	Interrupt request permission bit (IVFE)
Disable interrupts	Set to “0”
Enable interrupts	Set to “1”

Use interrupt request bit (TCCS.IVF) to clear interrupt requests.

	Interrupt request bit (IVF)
Clear interrupt requests	Write “0”

7.8 How do I stop the free-run timer?

Set with count operation bit (TCCS.STOP).

See “7.3 How do I enable / disable the count operation of the free-run timer? (Page No.744)”.

7.9 How are the free-run timer assigned to ICU and OCU?

- The value of Free-run timer 0 can be used as capture data by ICU0 and ICU1
- The value of Free-run timer 1 can be used as capture data by ICU2 and ICU3
- The value of Free-run timer 2 can be used as compare data by OCU0 and OCU1
- The value of Free-run timer 3 can be used as compare data by OCU2 and OCU3
- The value of Free-run timer 4 can be used as capture data by ICU4 and ICU5
- The value of Free-run timer 5 can be used as capture data by ICU6 and ICU7
- The value of Free-run timer 6 can be used as compare data by OCU4 and OCU5
- The value of Free-run timer 7 can be used as compare data by OCU6 and OCU7

8. Caution

- Clearing the free-run timer
  - When you reset (the INIT pin input, the watchdog reset, the software reset), the counter is initialized to “0000” and the counting is running.
  - When the free-run timer is cleared by software, the counter is cleared and the clear request is generated almost at the same time. If the counter is cleared by the compare-match, it is cleared when it is counted up.
  - After writing “1” in the clear bit (CLR), this request (CLR=“1”) is cleared at the same clear timing of the free-run timer. When the clear operation of this CLR and writing “1” to the clear bit occurs at the same time, the clear bit (CLR) keeps “1”, and after the next time the timer is cleared, it is cleared. (As a result, the free-run timer is cleared twice.)
  - The counter clear operation (the software, the overflow, and the compare-match) of the free-run timer is enabled while the free-run timer is counting. To clear while the free-run timer is stopped, write 0000_H in the timer count data register.

- Write to the timer data register
  When writing the value in the free-run timer, make sure to do so while the free-run timer is stopped (STOP=“0”), and with word access.

- External clock operation
  - The pulse width required for the external clock is $2/F_{CLKP}$ minimum.
  - When using an external clock, the timing of the compare-match output and the interrupt occurrence is the same as the next count clock timing after the compare-match. Therefore, to allow the compare-match output and interrupt generation, an external clock input of at least 1 clock is required after the compare-match.

- Read/modify/write
  The interrupt request flag (IVF) is always read “1” in read/modify/write.

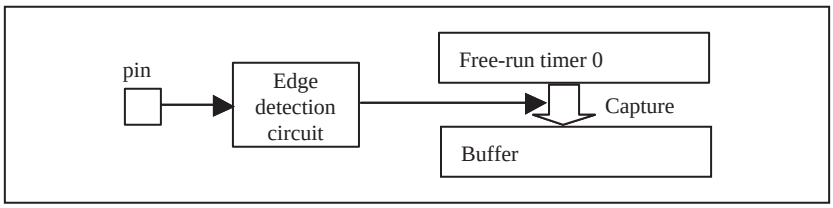
- Interrupt request flag
  If the interrupt request flag set timing and clear timing are simultaneous, the flag setting operation overrides the flag clearing operation.

8.Caution

Chapter 36 Input Capture

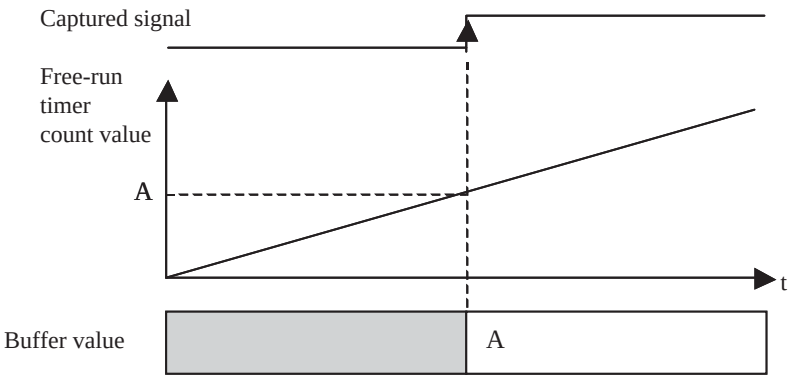
1. Overview

Input Capture records the free-run timer count value using timing detected from an external signal. It is then possible to calculate the time between signals using the record of the repeated count.



2. Features

- Format: Edge detection circuit + 16 bit buffer (capture register)
- Quantity: 4 groups = 8 channels (input capture channels 0/1, 2/3, 4/5, 6/7)
- Compatible timers: Input capture channels 0/1 use free-run timer 0
  Input capture channels 2/3 use free-run timer 1
  Input capture channels 4/5 use free-run timer 4
  Input capture channels 6/7 use free-run timer 5
- Edge Detection: Rising/falling/both edges
- Interrupt: Edge detection
- Capture value: Timer count value (0000_H-$FFFF_H$)
- Timer: Uses free-run timer 0
- Precision: $4/F_{CLKP}$, $16/F_{CLKP}$, $32/F_{CLKP}$, $64/F_{CLKP}$ (Free-run timer count clock)



3. Configuration

Figure 3-1 Configuration Diagram

Input capture 0-1

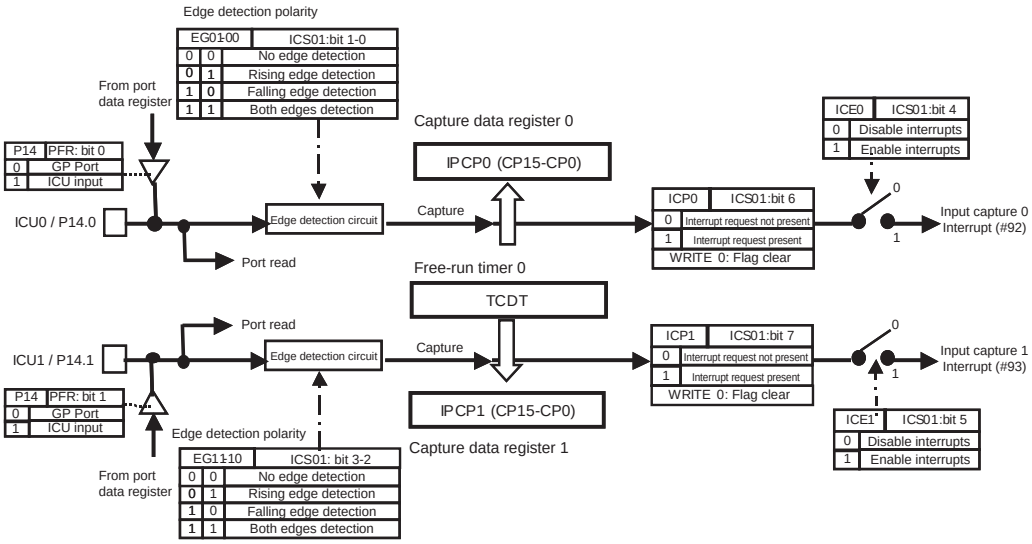
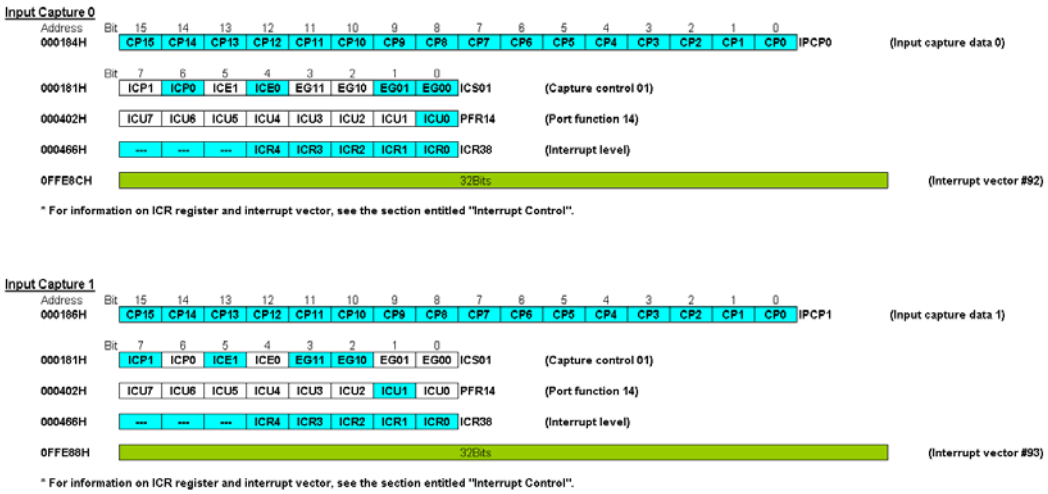


Figure 3-2 Register List



Note: For information about ICR registers and interrupt vectors, see “Chapter 24 Interrupt Control (Page No.315)”.

4. Register

4.1 IPCP: Input Capture Data Register

A register that, using changes in an external signal as a trigger, stores the free-run timer count and can read it out later.

- **IPCP0 (Input capture 0): Address 0184h (Access: Half-word, Word)**
- **IPCP1 (Input capture 1): Address 0186h (Access: Half-word, Word)**
- **IPCP2 (Input capture 2): Address 0188h (Access: Half-word, Word)**
- **IPCP3 (Input capture 3): Address 018Ah (Access: Half-word, Word)**
- **IPCP4 (Input capture 4): Address 02D4h (Access: Half-word, Word)**
- **IPCP5 (Input capture 5): Address 02D6h (Access: Half-word, Word)**
- **IPCP6 (Input capture 6): Address 02D8h (Access: Half-word, Word)**
- **IPCP7 (Input capture 7): Address 02DAh (Access: Half-word, Word)**

15	14	13	12	11	10	9	8	bit
CP15	CP14	CP13	CP12	CP11	CP10	CP9	CP8	
X	X	X	X	X	X	X	X	Initial value
R/WX	R/WX	R/WX	R/WX	R/WX	R/WX	R/WX	R/WX	Attribute

7	6	5	4	3	2	1	0	bit
CP7	CP6	CP5	CP4	CP3	CP2	CP1	CP0	
X	X	X	X	X	X	X	X	Initial value
R/WX	R/WX	R/WX	R/WX	R/WX	R/WX	R/WX	R/WX	Attribute

(For information on attributes, see “Meaning of Bit Attribute Symbols (Page No.10)”.)

- Stores the free-run timer 0 count value for the input signal from external pins (ICU0, ICU1) using the signal change (edge) selected by active edge selection bits (ICS01.EG[01:00]), (ICS01.EG[11:10]).
- Input capture 0 and input capture 1 store the count value of the free-run timer.

4.2 ICS: Input Capture Control Register

A register for controlling input capture

- **ICS01 (Input capture 0-1): Address 0181h (Access: Byte)**
- **ICS23 (Input capture 2-3): Address 0183h (Access: Byte)**
- **ICS45 (Input capture 4-5): Address 02D1h (Access: Byte)**
- **ICS67 (Input capture 6-7): Address 02D3h (Access: Byte)**

7	6	5	4	3	2	1	0	bit
ICP1	ICP0	ICE1	ICE0	EG11	EG10	EG01	EG00	
0	0	0	0	0	0	0	0	Initial value
R(RM1),W	R(RM1),W	R/W	R/W	R/W	R/W	R/W	R/W	Attribute

(For information on attributes, see "Meaning of Bit Attribute Symbols (Page No.10)".)

- bit7: Input capture 1 interrupt request flag

ICP1	Status	
	Read	Write
0	No interrupt request	Clear flag
1	Interrupt request present (edge detection present)	No effect on operation

  - When the signal change (edge) selected by the active capture edge selection bit (EG[11:10]) is detected on the input from an external pin, the flag becomes "1".
  - To activate the interrupt request it is necessary to set the interrupt enable bit (ICE1="1").
  - If the timing of the interrupt request flag becoming "1" and the writing of "0" occur simultaneously, the interrupt request flag will become "1".
- bit6: Input capture 0 interrupt request flag

ICP0	Status	
	Read	Write
0	No interrupt request	Clear flag
1	Interrupt request present	No effect on operation

  - When the signal change selected by the active capture edge selection bit (EG[01:00]) is detected on the input from an external pin (CS0), the flag becomes "1".
  - To activate the interrupt request, the interrupt request permission setting (ICE1="1") is necessary.
  - If the timing of the interrupt request flag becoming "1" and the writing of "0"occur simultaneously, the interrupt request flag will become "1".
- bit5: Input capture 1 interrupt request permission

ICE1	Operation
0	Interrupt disabled
1	Interrupt enabled

  - If input capture 1 interrupt request permission bit is set to "1", input capture 1 interrupt request ICP1 will be enabled.
- bit4: Input capture 0 interrupt request permission

ICE0	Operation
0	Interrupt disabled
1	Interrupt enabled

  - If input capture 0 interrupt request permission bit is set to "1", input capture 0 interrupt request ICP0 will be enabled.

- bit3-bit2: Input capture 1 active edge selection

EG11	EG10	Edge selection
0	0	Stop input capture
0	1	Rising edge
1	0	Falling edge
1	1	Both edges (rising edge and falling edge)

  - Select the active capture edge for the input capture signal from external pin (ICU1)
  - If the active edge selection bit is “00”, input capture 1 is stopped.
- bit1-bit0: Input capture 0 active edge selection

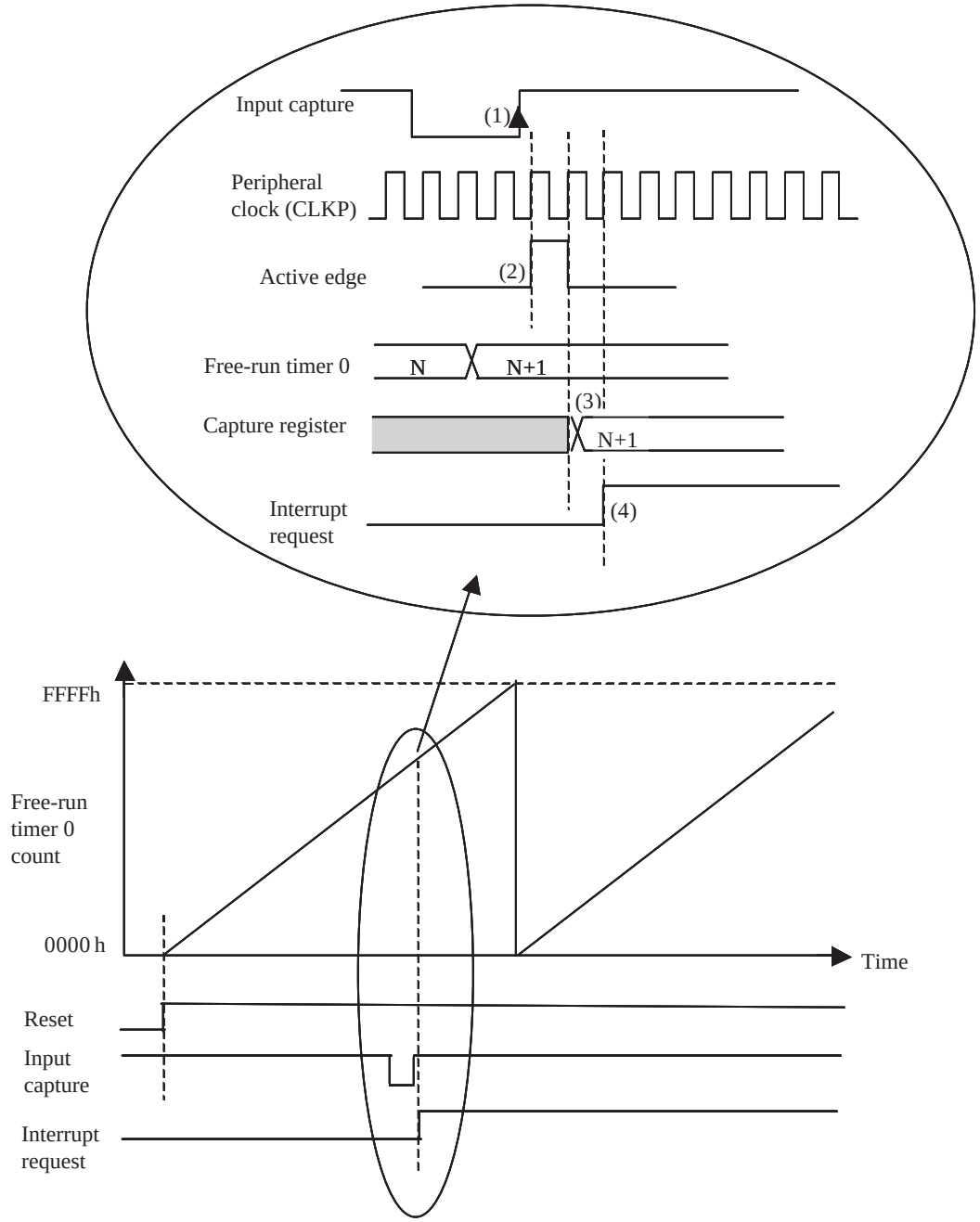
EG01	EG00	Edge selection
0	0	Stop input capture
0	1	Rising edge
1	0	Falling edge
1	1	Both edges (rising edge and falling edge)

  - Select the active capture edge for the input capture signal for external pin (ICU0).
  - When the active edge selection bit is “00”, input capture 0 is stopped.

5. Operation

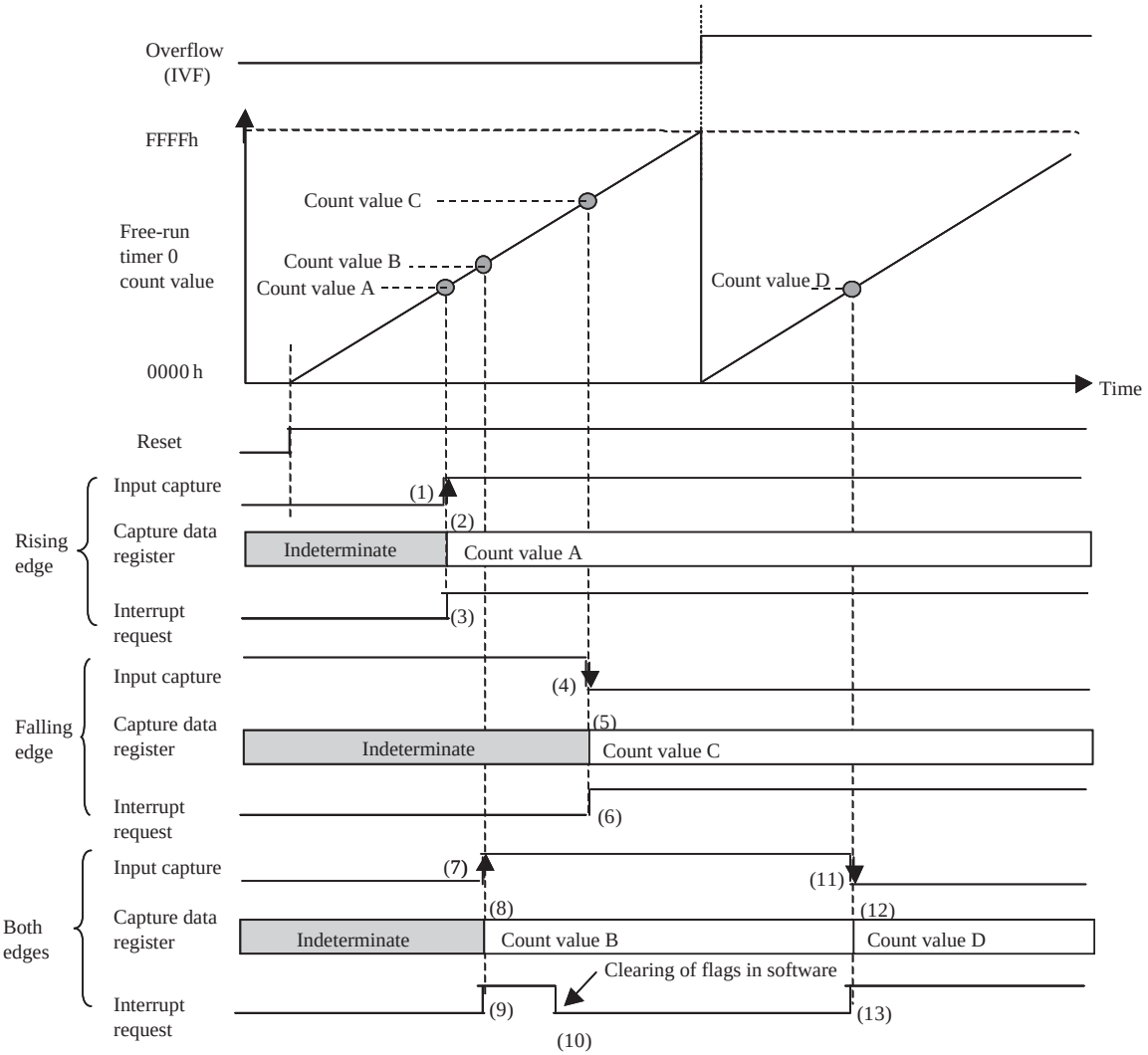
The input capture operation is described below.

5.1 Capture Timing, Interrupt Timing



(1) Rising edge of input signal
(2) Internal signal generated by edge detection (synchronous with peripheral clock)
(3) Store free-run timer value in capture register (capture)
(4) Input capture interrupt generation (ICU0-ICU1="1")

5.2 Input Capture Edge Specification and Operation



- When specifying rising edge
  - (1) Detection of rising edge of input signal
  - (2) Storage of free-run timer value in capture register (capture)
  - (3) Input capture interrupt generation
- When specifying falling edge
  - (4) Detection of input signal falling edge
  - (5) Storage of free-run timer value in capture register (capture)
  - (6) Input capture interrupt generation
- Both edges
  - (7) Detection of input signal rising edge
  - (8) Storage of free-run timer value in capture register (capture)
  - (9) Input capture interrupt generation
  - (10) Clear interrupt request flag (ICS01.ICP0), (ICS01.ICP1) in software
  - (11) Detection of input signal falling edge
  - (12) Storage of free-run timer value in capture register (capture)
  - (13) Input capture interrupt generation

6. Settings

Table 6-1 Settings Necessary for Using Input Capture

Settings	Setting register	Setting procedure*
Free-run timer settings	See “Chapter 35 Free-Run Timer (Page No.735)”	–
Free-run timer activation		
Input pin ICU0-ICU7 settings	Port function register (PFR14.0 - PFR14.7) Extra port function register (EPFR14.0 - EPFR14.7)	7.2
Active edge polarity selection for external input	Input capture control register (ICS01, ICS23, ICS45, ICS67)	7.1

*: For the setting procedure, refer to the section indicated by the number.

Table 6-2 Required Settings for ICU Interrupt

Settings	Settings register	Setting procedure*
Input Capture interrupt vector, Input capture interrupt level settings	See “Chapter 24 Interrupt Control (Page No.315)”	7.3
Input capture interrupt settings Interrupt request clear Interrupt request permission	Input capture control register (ICS01, ICS23, ICS45, ICS67)	7.5

*: For the setting procedure, refer to the section indicated by the number.

7. Q&A

7.1 What are the varieties of active edge polarity for external input, and how do I select them?

The active edge polarity varieties consist of rising, falling, and both, for a total of 3, and are set using the external input active edge selection bit (ICS01.EG[01:00]) and (ICS01:EG[11:10]), (ICS23.EG[01:00]) and (ICS23:EG[11:10]), (ICS45.EG[01:00]) and (ICS45:EG[11:10]), (ICS67.EG[01:00]) and (ICS67:EG[11:10]).

Operation	External input active edge polarity bit (EG[01:00]), (EG[11:10])
To select rising edge	Select “00”
To select falling edge	Select “10”
To select both edges	Select “11”

7.2 What about setting the external input pins (ICU0-7)?

Use the port function register and extra port function bits (PFR14.x/EPFR14.x).

Operation	Port function (PFR14.x)	Extra Port function (EPFR14.x)
To set it to the external input pins (ICU0)	Set PFR14.0 to “1”	Set EPFR14.0 to “0”
To set it to the external input pins (ICU1)	Set PFR14.1 to “1”	Set EPFR14.1 to “0”
To set it to the external input pins (ICU2)	Set PFR14.2 to “1”	Set EPFR14.2 to “0”
To set it to the external input pins (ICU3)	Set PFR14.3 to “1”	Set EPFR14.3 to “0”
To set it to the external input pins (ICU4)	Set PFR14.4 to “1”	Set EPFR14.4 to “0”
To set it to the external input pins (ICU5)	Set PFR14.5 to “1”	Set EPFR14.5 to “0”
To set it to the external input pins (ICU6)	Set PFR14.6 to “1”	Set EPFR14.6 to “0”
To set it to the external input pins (ICU7)	Set PFR14.7 to “1”	Set EPFR14.7 to “0”

Remark: When setting the Extra port function register EPFR14.x to “1” the corresponding input capture macro is internally connected to the corresponding LIN-USART macro for LIN Sync Field measurement. Hence, with this setting the corresponding ICU channel is not available as external input.

7.3 What about interrupt-related registers?

Input capture interrupt vector and input capture interrupt level settings

The relationship between input capture number, interrupt level, and vector is explained in the following table.

For more information on interrupt level and interrupt vectors, see “Chapter 24 Interrupt Control (Page No.315)”.

Number	Interrupt vector (Default)	Interrupt level setting bit (ICR[4:0])
Input Capture 0	#92 Address: 0FFE8Ch	Interrupt level register (ICR38) Address: 0466h
Input Capture 1	#93 Address: 0FFE88h	
Input Capture 2	#94 Address: 0FFE84h	Interrupt level register (ICR39) Address: 0467h
Input Capture 3	#95 Address: 0FFE80h	
Input Capture 4	#96 Address: 0FFE7Ch	Interrupt level register (ICR40) Address: 0468h
Input Capture 5	#97 Address: 0FFE78h	

Input Capture 6	#98 Address: 0FFE74h	Interrupt level register (ICR41) Address: 0469h
Input Capture 7	#99 Address: 0FFE70h	

Interrupt request flags (ICS01.ICP0), (ICS01.ICP1), (ICS23.ICP0), (ICS23.ICP1), (ICS45.ICP0), (ICS45.ICP1), (ICS67.ICP0), (ICS67.ICP1) are not automatically cleared, so please set the input capture interrupt request flag (ICP1, ICP0) to “0” to clear them before returning from interrupt processing.

7.4 What are the types of interrupts?

There is only one kind of interrupt, and it is generated by input signal edge detection.

7.5 How do I enable interrupts?

Interrupt request permission, interrupt request flag

Interrupts are enabled via interrupt request permission bit (ICS01.ICE0), (ICS01.ICE1), (ICS23.ICE0), (ICS23.ICE1), (ICS45.ICE0), (ICS45.ICE1), (ICS67.ICE0), (ICS67.ICE1).

	Interrupt request permission bit (ICE0), (CE1)
Disable interrupts	Set to “0”
Enable interrupts	Set to “1”

Clearing of interrupt requests is done using interrupt request bit (ICS01.ICP0), (ICS01.ICP1), (ICS23.ICP0), (ICS23.ICP1), (ICS45.ICP0), (ICS45.ICP1), (ICS67.ICP0), (ICS67.ICP1).

	Interrupt request bit (ICP0), (ICP1)
Interrupt request clear	Write “0”

7.6 How do I measure the pulse width of the input signal?

- "H" Width measurement:

  Specify both edges for edge detection.

  First detect the rising edge, then detect the falling edge.

  Pulse width = {value recorded during falling (input capture register value)
  + “10000h” x Overflow frequency
  – value recorded during rising (input capture register value)}
  x Count clock width of free-run timer

  Example: value recorded during falling = 2320h, Value recorded during rising = A635h,
  Overflow frequency = 1, count clock = 125ns
  ==> pulse width = (2320h+10000h-A635h) x 125ns = 3997.375us

- Cycle measurement:

  Specify rising (or falling) for edge detection.

  Detect edge 2 times.

  Cycle = {Second recorded value (input capture register value)
  + “10000h” x Overflow frequency
  – First recorded value (input capture register value)}
  x Count clock width of free-run timer

8. Caution

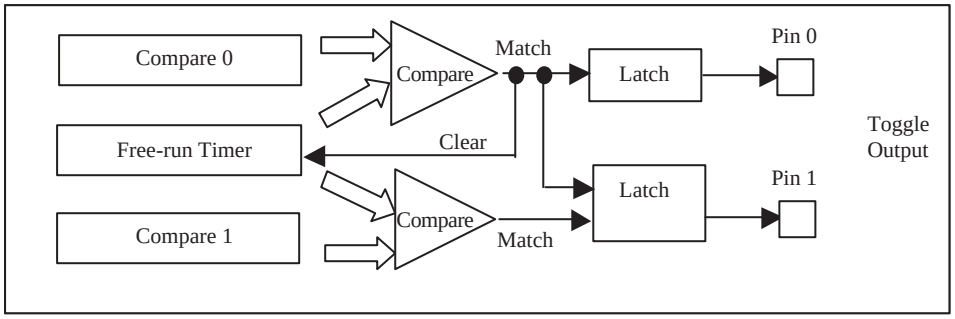
- Input capture register
  The value of the input capture register during reset is indeterminate.
  Read out of the input capture register must always be done using 16 or 32 bit access.

- Read modify write
  Input capture interrupt request bit (ICP0), (ICP1) will be read as “1” when read with read modify write.

Chapter 37 Output Compare

1. Overview

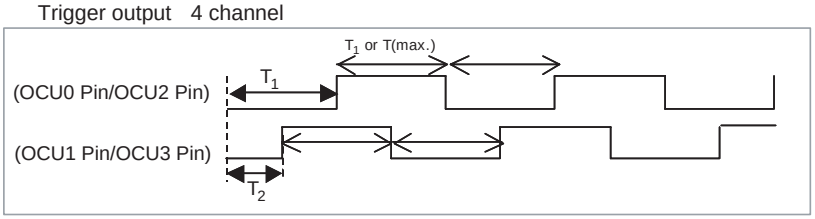
Output compare is a feature that compares the value set to the compare register with the count value of the free-run timer, and reverses the level of the pins when they are equal.



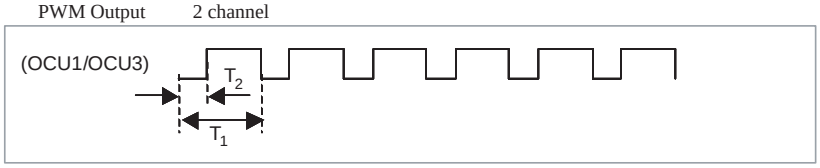
2. Features

• Output wave form: Toggle output 4 channel

Trigger output 4 channel



PWM Output 2 channel



- Format: 16 bit compare register + Compare circuit
- Quantity: 4 groups = 8 channels (output compare channels 0/1, 2/3, 4/5, 6/7)
- Compatible timers: Output compare channels 0/1 use free-run timer 2
  Output compare channels 2/3 use free-run timer 3
  Output compare channels 4/5 use free-run timer 6
  Output compare channels 6/7 use free-run timer 7
- Operation on compare match:
  - Reversal of pin output value (toggle output)
  - Free-run timer clear
  - Interrupt generation
- Count precision: $4/F_{CLKP}$, $16/F_{CLKP}$, $32/F_{CLKP}$, $64/F_{CLKP}$ (dependent on free-run timer)
- Toggle change width (T): 1 x count precision - 10000_H x count precision
- Interrupt: Compare-match interrupt
- Others: Setting of initial output level value is possible (“H”/“L”)
  Pins not used for OCU output can be used as general-use ports

3. Configuration Diagram

Figure 3-1 Configuration Diagram

Output Compare 0-1

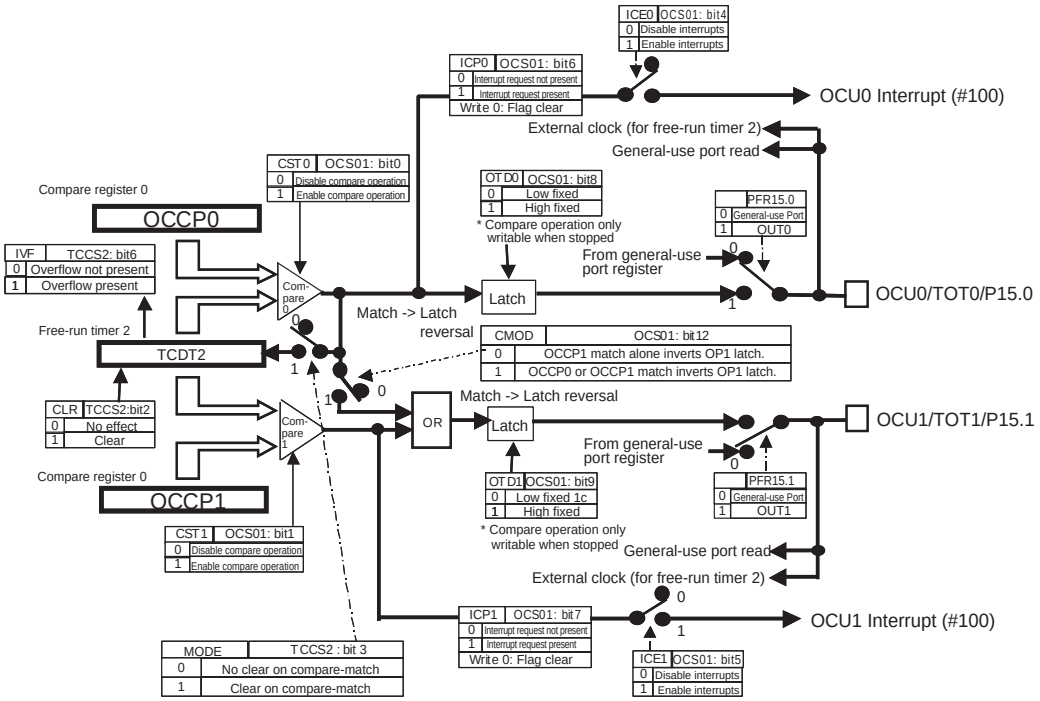
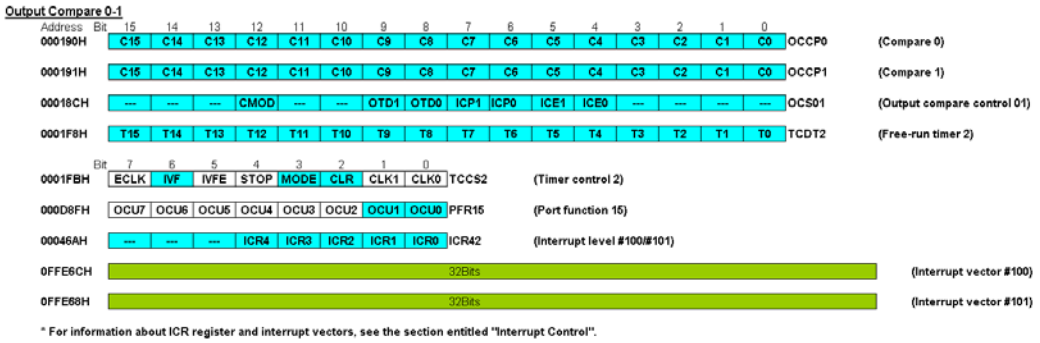


Figure 3-2 Register List



Note: For information about ICR registers and interrupt vectors, see “Chapter 24 Interrupt Control (Page No.315)”.

4. Registers

4.1 OCS: Output Control Register

A register for controlling the operation of output compare.

- **OCS01 (Output compare 0-1): Address 018Ch (Access: Byte, Half-word, Word)**
- **OCS23 (Output compare 2-3): Address 018Eh (Access: Byte, Half-word, Word)**
- **OCS45 (Output compare 4-5): Address 02DCh (Access: Byte, Half-word, Word)**
- **OCS67 (Output compare 6-7): Address 02DEh (Access: Byte, Half-word, Word)**

15	14	13	12	11	10	9	8	bit
–	–	–	CMOD	–	–	OTD1	OTD0	
1	1	1	0	1	1	0	0	Initial Value
R1/W1	R1/W1	R1/W1	R/W	R1/W1	R1/W1	R/W	R/W	Attribute

7	6	5	4	3	2	1	0	bit
ICP1	ICP0	ICE1	ICE0	–	–	CST1	CST0	
0	0	0	0	1	1	0	0	Initial Value
R(RM1),W	R(RM1),W	R/W	R/W	R1/W1	R1/W1	R/W	R/W	Attribute

(For information on attributes, see "Meaning of Bit Attribute Symbols (Page No.10)".)

- bit15-bit13: undefined Writing does not affect the operation. The read out value is "1".
- bit12: Reverse Mode

CMOD	Operation Mode
0	Independent operation (the output level reversal operation of pins OCU0-OCU1 is independent)
1	Combined operation (OCU1 output pin level is inverted when output compare 0 or output compare 1 is matched in the compare operation.)

- Specifies the output level reversal operation of pin OCU1 when free-run timer count value TCDT2 matches compare registers OCCP0, OCCP1.
- When the reverse mode bit is set to "1", the operation is as follows.
  OCU0 pin: output reverses when free-run timer TCDT2 matches compare register 0 (OCCP0)
  OCU1 pin: output reverses when free-run timer TCDT2 matches compare register 1 (OCCP1)
- When the reverse mode bit is set to "0", the operation is as follows.
  OCU0 pin: output reverses when free-run timer TCDT2 matches compare register 0(OCCP0)
  OCU1 pin: output reverses when free-run timer TCDT2 matches compare register 0 (OCCP0) or compare register 1 (OCCP1)

Note: Reversal mode does not allow interrupts, even with cooperative operation (CMOD="1").

- For output from pins OCU0, OCU1, registers PFR15.0, PFR15.1 must be set.

- bit11-bit10: Undefined Writing does not affect the operation. The read value is "1".
- bit9: Pin-level settings (output compare 1)

OTD1	Operation
0	Set the output level of pin OCU1 to "L"
1	Set the output level of pin OCU1 to "H"

To perform output on pin OCU1, general-purpose port settings must be performed.

- bit8: Pin-level settings (output compare 0)

OTD0	Operation
0	Set the output level of pin OCU0 to "L"
1	Set the output level of pin OCU0 to "H"

- To perform output on pin OCU0, general-purpose port settings must be performed.

- bit7: Interrupt request flag (output compare 1)

ICP1	Status	
	Read	Write
0	Interrupt request not present	Clear flag (ICP1)
1	Interrupt request present	No effect on operation

  - If free-run timer count value TCDT2 matches the output compare register OCCP1, ICP1 becomes “1”.
  - Interrupt request is enabled when the interrupt permission bit (ICP1) is set to “1”.
  - If the interrupt request flag becoming “1” and writing of “0” to it happen at the same time, the interrupt request flag will become “1” (flag setting is given priority).
  - When using an external clock as the free-run timer operation clock, at least one external clock input is necessary after compare match for output compare-match output and interrupt generation.
- bit6: Interrupt request flag (output compare 0)

ICP0	Status	
	Read	Write
0	Interrupt request not present	Clear flag (ICP0)
1	Interrupt request present	No effect on operation

  - If free-run timer count value TCDT2 matches output compare register OCCP0, ICP0 becomes “1”.
  - Interrupt request is enabled when the interrupt permission bit (ICP0) is “1”.
  - If the interrupt request flag becoming “1” and writing of “0” to it happen at the same time, the interrupt request flag will become “1” (flag setting is given priority).
  - When using an external clock as the free-run timer operation clock, at least one external clock input is necessary after compare match for output compare-match output and interrupt generation.
- bit5: Interrupt request enable (output compare 1)

ICE1	Status
0	Disable output compare 1 interrupt requests
1	Enable output compare 1 interrupt requests

- bit4: Interrupt request enable (output compare 0)

ICE0	Status
0	Disable output compare 0 interrupt requests
1	Enable output compare 0 interrupt requests

- bit3-bit2: Undefined Writing does not affect the operation. The read value is always “1”.
- bit1: Enable operation requests (output compare 1)

CST1	Operation
0	Stop operation of output compare 1
1	Enable operation of output compare 1

  - A bit that enables a comparison operation between the free-run timer count value and the output compare register (TCDT2 and OCCP1).
  - Before enabling the operation, always set a value to compare register OCCP1.
  - If you stop the free-run timer, output compare also stops.

- bit0: Enable operation requests (output compare 0)

CST0	Operation
0	Disable output compare 0 operation
1	Enable output compare 0 operation

  - A bit that enables a comparison operation between the free-run timer count value and the output compare register (TCDT2 and OCCP0).
  - Before enabling the operation, always set a value to compare register OCCP0.
  - If you stop the free-run timer, output compare also stops.

4.2 OCCP: Compare Register

Register that sets the value to be compared to the 16 bit free-run timer count value.

- **OCCP0 (Compare 0): Address 0190h (Access: Half-word, Word)**
- **OCCP1 (Compare 1): Address 0192h (Access: Half-word, Word)**
- **OCCP2 (Compare 2): Address 0194h (Access: Half-word, Word)**
- **OCCP3 (Compare 3): Address 0196h (Access: Half-word, Word)**
- **OCCP4 (Compare 4): Address 02E0h (Access: Half-word, Word)**
- **OCCP5 (Compare 5): Address 02E2h (Access: Half-word, Word)**
- **OCCP6 (Compare 6): Address 02E4h (Access: Half-word, Word)**
- **OCCP7 (Compare 7): Address 02E6h (Access: Half-word, Word)**

15	14	13	12	11	10	9	8	bit
C15	C14	C13	C12	C11	C10	C9	C8	
X	X	X	X	X	X	X	X	Initial Value
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Attribute

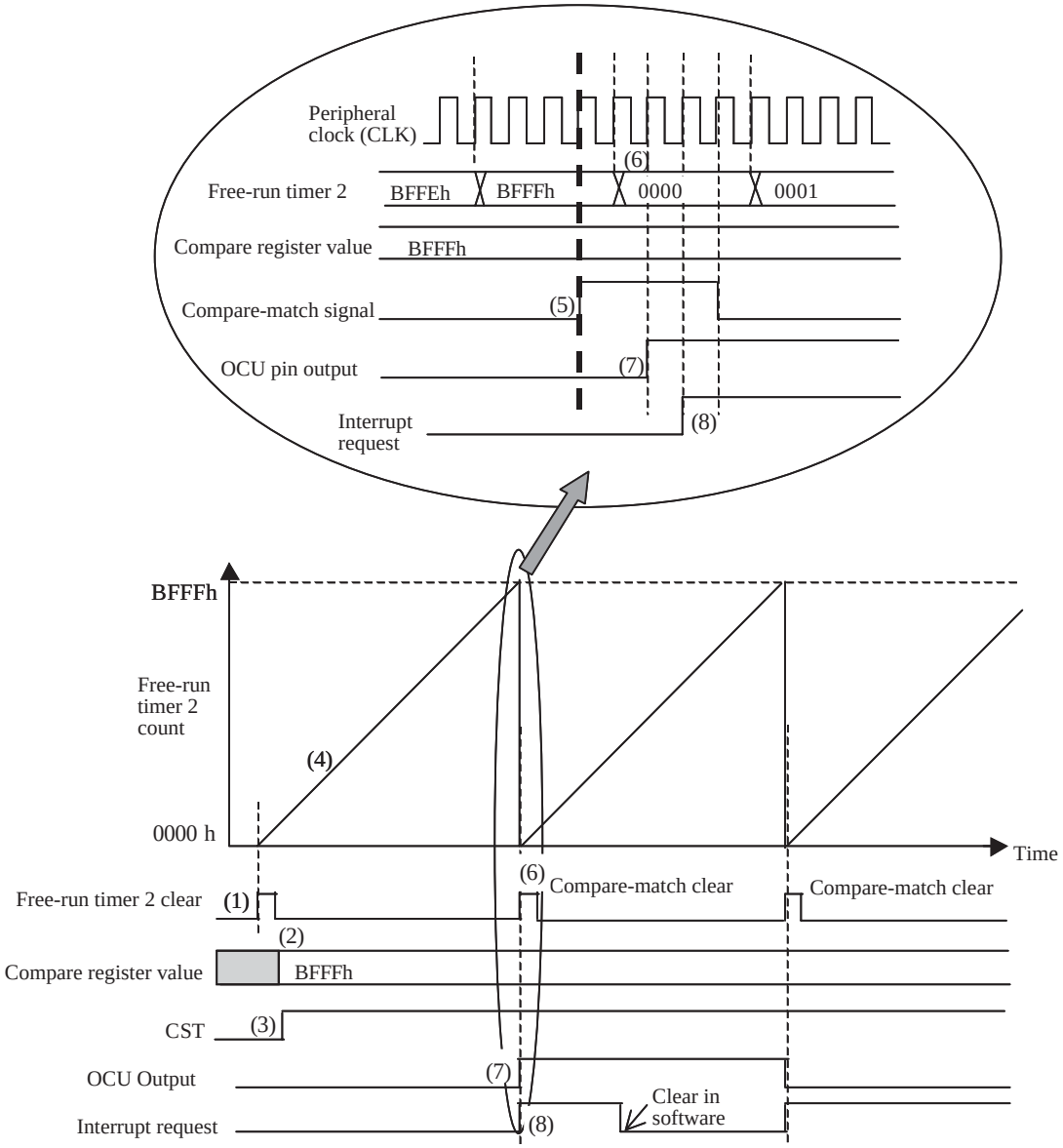
7	6	5	4	3	2	1	0	bit
C7	C6	C5	C4	C3	C2	C1	C0	
X	X	X	X	X	X	X	X	Initial Value
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Attribute

(For information on attributes, see “Meaning of Bit Attribute Symbols (Page No.10)”.)

- Compares the compare registers OCCP0, OCCP1 to free-run timer 2 count value TCD2.
- Compares the compare registers OCCP2, OCCP3 to free-run timer 3 count value TCD3.
- Compares the compare registers OCCP4, OCCP5 to free-run timer 6 count value TCD6.
- Compares the compare registers OCCP6, OCCP7 to free-run timer 7 count value TCD7.

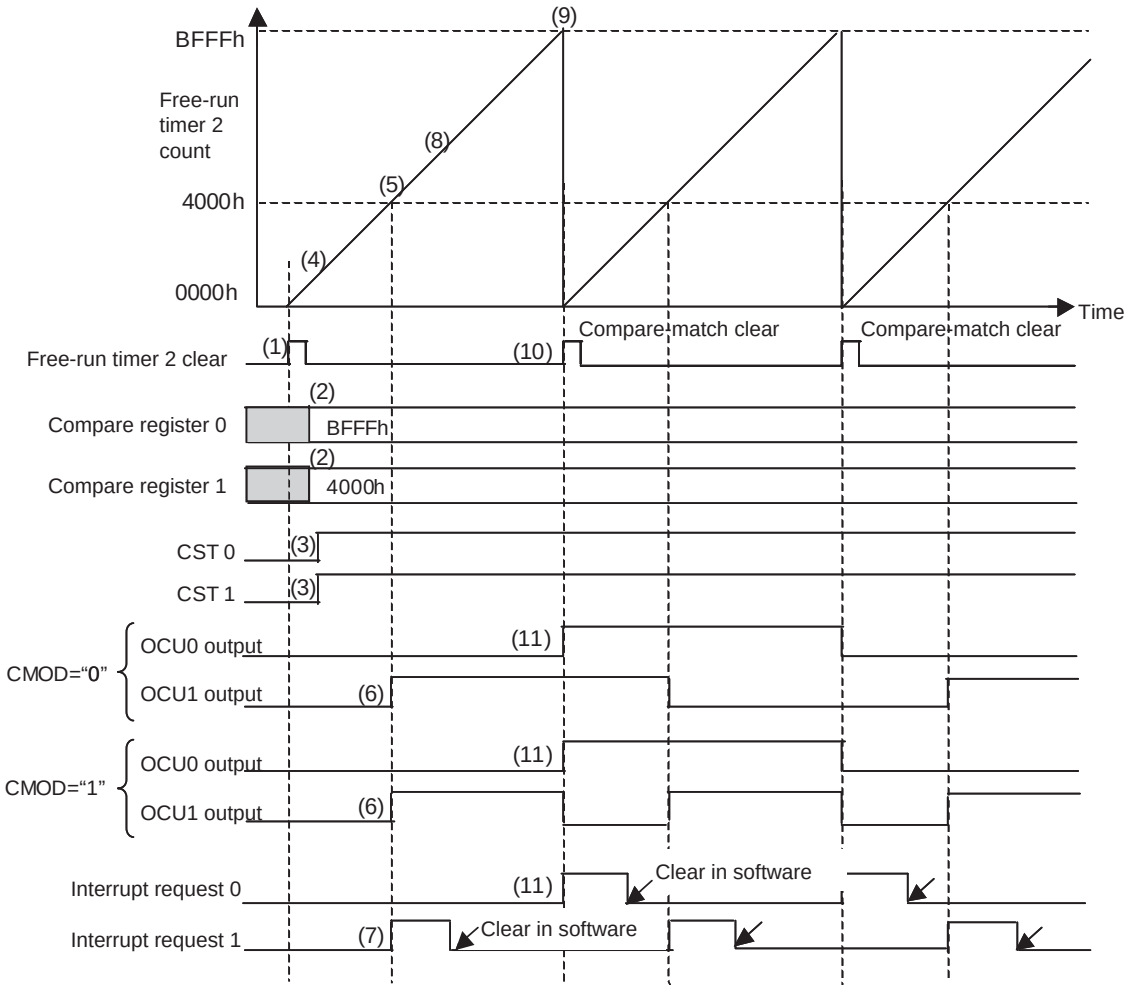
5. Operation

5.1 Output Compare Output (Independent Reversal) CMODE="0"



(1) Free-run timer clear/reset
(2) Compare value setting
(3) Enable compare operation (CST="1")
(4) Free-run timer count up (example of 1 clock in 4)
(5) Compare free-run timer value and compare value and match (compare match).
(6) Free-run timer clear from compare match (free run timer 2)
(7) OCU output level reversal
(8) Compare match interrupt request generation

5.2 Output Compare Output (Cooperative Reversal) CMODE="1"



(1) Free-run timers clear/reset
(2) Compare 0 and compare 1 value settings
(3) Enable compare operation
(4) Free-run timer count up
(5) Compare 1 match
(6) OCU1 output level reversal
(7) Compare 1 match interrupt
(8) Free-run timer count up
(9) Compare 0 match
(10) OCU0 output level reversal
When CMOD="1", OCU1 output level also reverses
(11) Compare 0 match interrupt

6. Settings

Table 6-1 Settings Necessary for Using Output Compare

Settings	Setting Register	Setting Procedure*
Free-run timer setting	See “Chapter 35 Free-Run Timer (Page No.735)”	---
Compare value setting	Compare register (OCCP0 - OCCP7)	See 7.1
Compare mode setting	Output control register (OCS01, OCS23, OCS45, OCS67)	See 7.2
Stop compare operation		See 7.3
Set initial level of compare pin output		See 7.4
Set OCU0-OCU7 pins to output	Port function register (PFR15.0 - PFR15.7) Extra port function register (EPFR15.0 - EPFR15.7)	See 7.5
Clear free-run timer	Timer control register (TCCS2, TCCS3, TCCS6, TCCS7) See “Chapter 35 Free-Run Timer (Page No.735)”	See 7.6
Enable compare operation (activate)	Output control register (OCS01, OCS23, OCS45, OCS67)	See 7.7

*: For the setting procedure, refer to the section indicated by the number.

Table 6-2 Item Necessary to Clear the Free-run Timer upon Compare-match.

Setting	Setting Register	Setting Procedure*
Select free-run timer clear mode	Timer control register (TCCS2, TCCS3, TCCS6, TCCS7) See “Chapter 35 Free-Run Timer (Page No.735)”	See 7.8

*: For the setting procedure, refer to the section indicated by the number.

Table 6-3 Item Necessary for Performing Interrupts

Setting	Setting register	Setting Procedure*
Output compare interrupt vector, output compare interrupt level setting	See “Chapter 24 Interrupt Control (Page No.315)”	See 7.9
Output compare interrupt setting Clear interrupt request Enable interrupt request	Output control register (OCS01, OCS23, OCS45, OCS67)	See 7.11

*: For the setting procedure, refer to the section indicated by the number.

7. Q & A

7.1 How do I set the compare value?

Write the compare value to compare registers OCCP0 - OCCP7.

7.2 How do I set the compare mode? (for OCU1, OCU3, OCU5, OCU7 output)

This is done using compare mode bits (OCS01.CMOD), (OCS23.CMOD), (OCS45.CMOD), (OCS67.CMOD).

Operation	Compare mode bit
To reverse OCU1 output using a compare-match from only free-run timer 2 and compare register 1	Set (OCS01.CMOD) bit to “0”
To reverse OCU3 output using a compare-match from only free-run timer 3 and compare register 3	Set (OCS23.CMOD) bit to “0”
To reverse OCU5 output using a compare-match from only free-run timer 6 and compare register 5	Set (OCS45.CMOD) bit to “0”
To reverse OCU7 output using a compare-match from only free-run timer 7 and compare register 7	Set (OCS67.CMOD) bit to “0”
To reverse OCU1 output using a compare-match from free-run timer 2 and compare register 0, as well as free-run timer 0 and compare register 1	Set (OCS01.CMOD) bit to “1”
To reverse OCU3 output using a compare-match from free-run timer 3 and compare register 2, as well as free-run timer 1 and compare register 3	Set (OCS23.CMOD) bit to “1”
To reverse OCU5 output using a compare-match from free-run timer 6 and compare register 4, as well as free-run timer 4 and compare register 1	Set (OCS45.CMOD) bit to “1”
To reverse OCU7 output using a compare-match from free-run timer 7 and compare register 6, as well as free-run timer 5 and compare register 3	Set (OCS67.CMOD) bit to “1”

With no relation to CMOD bit,

OCU0 output is reversed by a compare-match between free-run timer 2 and compare register 0 only.

OCU2 output is reversed by a compare-match between free-run timer 3 and compare register 2 only.

OCU4 output is reversed by a compare-match between free-run timer 6 and compare register 4 only.

OCU6 output is reversed by a compare-match between free-run timer 7 and compare register 6 only.

7.3 How do I enable/disable the compare operation?

Set it via the compare operation permission bit (OCS01.CST[1:0]), (OCS23.CST[1:0]), (OCS45.CST[1:0]), (OCS67.CST[1:0]).

Operation	Compare	Compare operation permission bit
To stop (disable) the compare operation	Compare 0	Set (OCS01.CST[0]) to “0”
	Compare 1	Set (OCS01.CST[1]) to “0”
	Compare 2	Set (OCS23.CST[0]) to “0”
	Compare 3	Set (OCS23.CST[1]) to “0”
	Compare 4	Set (OCS45.CST[0]) to “0”
	Compare 5	Set (OCS45.CST[1]) to “0”
	Compare 6	Set (OCS67.CST[0]) to “0”
	Compare 7	Set (OCS67.CST[1]) to “0”

To enable compare operation	Compare 0	Set (OCS01.CST[0]) to “1”
	Compare 1	Set (OCS01.CST[1]) to “1”
	Compare 2	Set (OCS23.CST[0]) to “1”
	Compare 3	Set (OCS23.CST[1]) to “1”
	Compare 4	Set (OCS45.CST[0]) to “1”
	Compare 5	Set (OCS45.CST[1]) to “1”
	Compare 6	Set (OCS67.CST[0]) to “1”
	Compare 7	Set (OCS67.CST[1]) to “1”

7.4 How do I set the initial level of the compare pin output?

Set it with compare pin output specification bit (OCS01.OTD[1:0]), (OCS23.OTD[1:0]), (OCS45.OTD[1:0]), (OCS67.OTD[1:0]).

Operation	Compare pin output specification bit
To set compare 0 pin to “L”	Set (OCS01.OTD0) to “0”
To set compare 0 pin to “H”	Set (OCS01.OTD0) to “1”
To set compare 1 pin to “L”	Set (OCS01.OTD1) to “0”
To set compare 1 pin to “H”	Set (OCS01.OTD1) to “1”
To set compare 2 pin to “L”	Set (OCS23.OTD0) to “0”
To set compare 2 pin to “H”	Set (OCS23.OTD0) to “1”
To set compare 3 pin to “L”	Set (OCS23.OTD1) to “0”
To set compare 3 pin to “H”	Set (OCS23.OTD1) to “1”
To set compare 4 pin to “L”	Set (OCS45.OTD0) to “0”
To set compare 4 pin to “H”	Set (OCS45.OTD0) to “1”
To set compare 5 pin to “L”	Set (OCS45.OTD1) to “0”
To set compare 5 pin to “H”	Set (OCS45.OTD1) to “1”
To set compare 6 pin to “L”	Set (OCS67.OTD0) to “0”
To set compare 6 pin to “H”	Set (OCS67.OTD0) to “1”
To set compare 7 pin to “L”	Set (OCS7.OTD1) to “0”
To set compare 7 pin to “H”	Set (OCS67.OTD1) to “1”

7.5 How do I set the output for compare pins OCU0-OCU7?

Set it with port function register (PFR15[7:0]).

Operation	Port function bit	Extra port function bit
To set compare 0 pin (OCU0) to output	Set PFR15.0 bit to “1”	Set EPFR15.0 bit to “0”
To set compare 1 pin (OCU1) to output	Set PFR15.1 bit to “1”	Set EPFR15.1 bit to “0”
To set compare 2 pin (OCU2) to output	Set PFR15.2 bit to “1”	Set EPFR15.2 bit to “0”
To set compare 3 pin (OCU3) to output	Set PFR15.3 bit to “1”	Set EPFR15.3 bit to “0”
To set compare 4 pin (OCU4) to output	Set PFR15.4 bit to “1”	Set EPFR15.4 bit to “0”
To set compare 5 pin (OCU5) to output	Set PFR15.5 bit to “1”	Set EPFR15.5 bit to “0”
To set compare 6 pin (OCU6) to output	Set PFR15.6 bit to “1”	Set EPFR15.6 bit to “0”
To set compare 7 pin (OCU7) to output	Set PFR15.7 bit to “1”	Set EPFR15.7 bit to “0”

7.6 How do I clear the free-run timer?

Set it with clear bits (TCCS2.CLR), (TCCS3.CLR), (TCCS6.CLR), (TCCS7.CLR).

Operation	Clear Bit (CLR)
To clear the free-run timer	Write “1”

For other methods, see “Chapter 35 Free-Run Timer (Page No.735)”.

7.7 How do I enable the compare operation?

Enable it with compare operation permission bit (OCS01.CST[1:0]), (OCS23.CST[1:0]), (OCS45.CST[1:0]), (OCS67.CST[1:0]).

See “7.4 How do I set the initial level of the compare pin output? (Page No.771)”.

7.8 How do I compare the free-run timer value with the compare register value and clear the free-run timer when they match?

Do this with timer initialization condition bit (TCCS2.MODE), (TCCS3.MODE), (TCCS6.MODE), (TCCS7.MODE).

Operation	Timer initialization condition bit (MODE)
To clear free-run timer upon compare 0 match	Set (TCCS2.MODE) to “1”
To clear free-run timer upon compare 2 match	Set (TCCS3.MODE) to “1”
To clear free-run timer upon compare 4 match	Set (TCCS6.MODE) to “1”
To clear free-run timer upon compare 6 match	Set (TCCS7.MODE) to “1”

7.9 What are the interrupt-related registers?

Set the output compare interrupt vector and output compare interrupt level.

The relationship between output compare number, interrupt level, and vector is shown in the following table.

For detailed information on interrupt levels and interrupt vectors, see “Chapter 24 Interrupt Control (Page No.315)”.

Number	Interrupt vector (default)	Interrupt level setting bit (ICR[4:0])
Output Compare 0	#100 Address: 0FFE6Ch	Interrupt level register (ICR42) Address: 046Ah
Output Compare 1	#101 Address: 0FFE68h	

Output Compare 2	#102 Address: 0FFE64h	Interrupt level register (ICR43) Address: 046Bh
Output Compare 3	#103 Address: 0FFE60h	
Output Compare 4	#104 Address: 0FFE5Ch	Interrupt level register (ICR44) Address: 046Ch
Output Compare 5	#105 Address: 0FFE58h	
Output Compare 6	#106 Address: 0FFE54h	Interrupt level register (ICR45) Address: 046Dh
Output Compare 7	#107 Address: 0FFE50h	

Interrupt request flags (OCS01. ICP[1:0]), (OCS23. ICP[1:0]), (OCS45. ICP[1:0]), (OCS67. ICP[1:0]), are not automatically cleared, so write “0” to the ICP[7:0] bit before returning from interrupt processing to clear them.

7.10 What are the types of interrupts?

There is only one type of interrupt, generated upon a compare-match.

7.11 How do I enable interrupts?

Enabling of interrupts is done with interrupt request permission bit (OCS01. ICE[1:0]), (OCS23. ICE[1:0]), (OCS45. ICE[1:0]), (OCS67. ICE[1:0]).

	Interrupt request permission bit (ICE0, ICE1)
Interrupt disabled	Set to “0”
Enable interrupts	Set to “1”

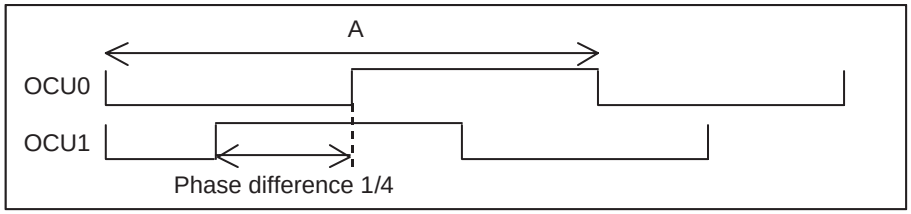
Interrupt requests are cleared with interrupt request bits (OCS01. ICP[1:0]), (OCS23. ICP[1:0]), (OCS45. ICP[1:0]), (OCS67. ICP[1:0]).

	Interrupt request bit (ICP0, ICP1)
Interrupt request clear	Write “0”

7.12 Compare value calculation procedure

- Toggle output pulse

(Example) To output a period: A, phase difference 1/4 phase pulse

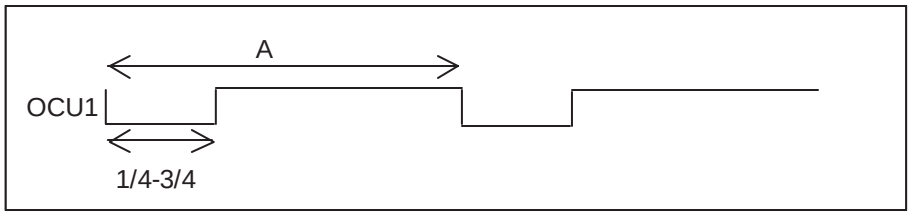


Formula: Compare 0 value = (A/2) / count clock
Compare 1 value = (A/4) / count clock
(Count clock: time set with free-run timer)

Note: To clear free-run timer 2 on compare 0 match setting (TCCS2.MODE=“1”) and CMOD=“0” setting are necessary.
Calculation example: A=1024us, count clock =125ns
Compare 0 value = (1024000 / 2) / 125 - 1 = 4095 = FFFh
Compare 1 value = (1024000 / 4) / 125 - 1 = 1023 = 7FFh

- PWM output

(Example) To output a period: A, duty 1/4 - 3/4 (“L”) PWM,

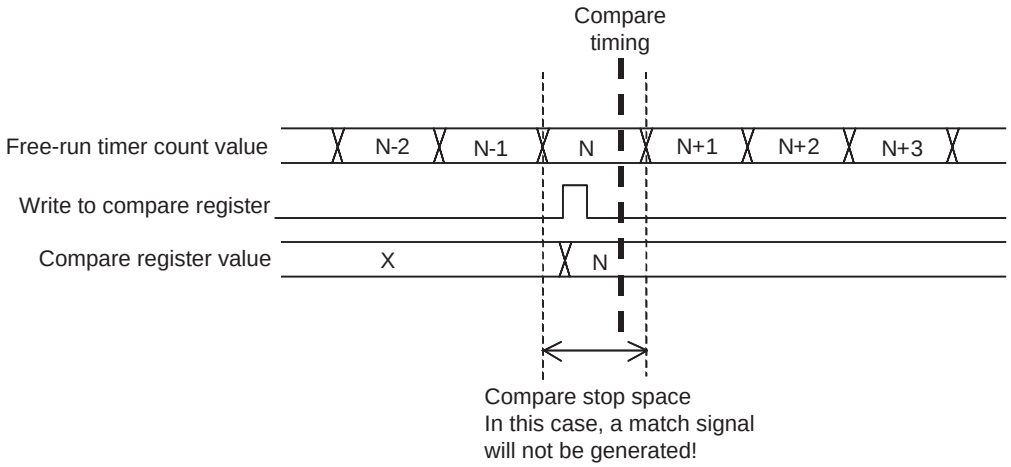


Formula: Compare 0 value = A / count clock
Compare 1 value = (A/4) / count clock (when duty 1/4)
(A x 3/4) / count clock (when duty 3/4)
(count clock: time set with free-run timer)

Note: To clear free-run timer 0 on compare 0 match setting (TCCS0.MODE=“1”) and CMOD=“1” setting are necessary.
Calculation example: A=1024us, count clock =125ns
Compare 0 value = 1024000 / 125 - 1 = 8191 = 1FFFh
Compare 1 value = (1024000 / 4) / 125 - 1 = 1023 = 7FFh (when duty 1/4)
(1024000 x 3 / 4) / 125 - 1 = 1023 = BFFh (when duty 3/4)

8. Caution

- Compare stop space during compare operation
  As shown below, for one count directly after the compare value is written to the compare register, the compare operation cannot be used.



- When CMOD="1" and OCCP0=OCCP1 setting, if a compare match is generated, the port will only reverse once.

- Compare registers (OCCP0 - OCCP7) are set to the initial values. Always set a value before activating them.

- When specifying the output level of compare pins (OCU0 - OCU7), first stop the compare operation.

- Output compare is synchronous with the free-run timer, so if you stop the free-run timer the compare operation also stops.

- Even when reversal mode specification (CMOD) is set to "1" and the compare operation is in cooperative mode, interrupts are generated independently.

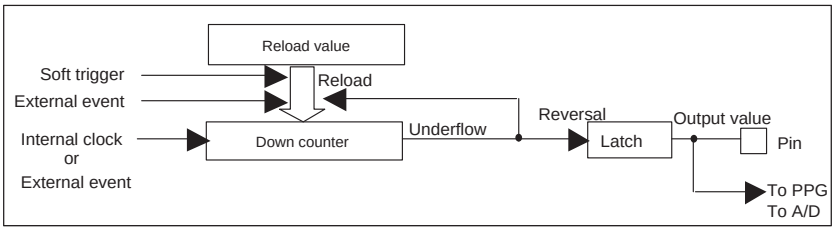
- When using an external clock as the free-run timer, compare-matches and interrupts are generated with the following clock. To generate compare match output and interrupts, at least "1 clock division" must be input to the external clock free-run timer after the compare-match.

8.Caution

Chapter 38 Reload Timer

1. Overview

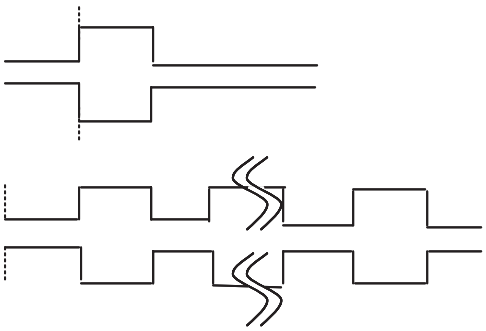
The reload timer uses a 16 bit down counter to detect the input signal trigger and perform a count down. The count length is 16 bits.



2. Features

Operation: 2 kinds of operation are possible

- One-shot Operation
  Initial output level
  Reversed output level
- Reload Operation
  Initial output level
  Reversed output level



Format: 16 bit down counter with reload register
Quantity: 8 (Output: 8 channels TOT[0-7])
Clock mode: Select from two modes

- Internal clock mode
  Count clock: CLKP/2, CLKP/8, CLKP/32, CLKP/64, CLKP/128
  Activation triggers (4 types)
- External event clock mode
  Count clock : External event (TIN[7:0] pins)
  Count active edge: Rising/falling/both edges of external event
  Activation trigger: Software trigger
  Cycle : Cycle = count clock x (reload value + 1)
  (Example) When count clock = 16MHz, reload value = 15999
  Cycle = 62.5ns x (15999+1) = 1.0ms

Count active edge: When in external event mode, choose from 3 types.

- External trigger (rising /falling/both edges)
  Interrupt: Request generated by underflow
  Other 1: Counter stop in software/can be reopened
  Other 2: Control of other peripheral functions possible
- PPG activation trigger source:
  Reload timer 0 : PPG0, PPG1
  Reload timer 1 : PPG2, PPG3

Reload timer 2 : PPG4, PPG5
Reload timer 3 : PPG6, PPG7
Reload timer 4 : PPG8, PPG9
Reload timer 5 : PPG10, PPG11
Reload timer 6 : PPG12, PPG13
Reload timer 7 : PPG14, PPG15

- A/D converter activation trigger source (Reload timer 7 : A/D)

3. Configuration

Figure 3-1 Configuration Diagram

Reload Timer 0 (Internal clock count)

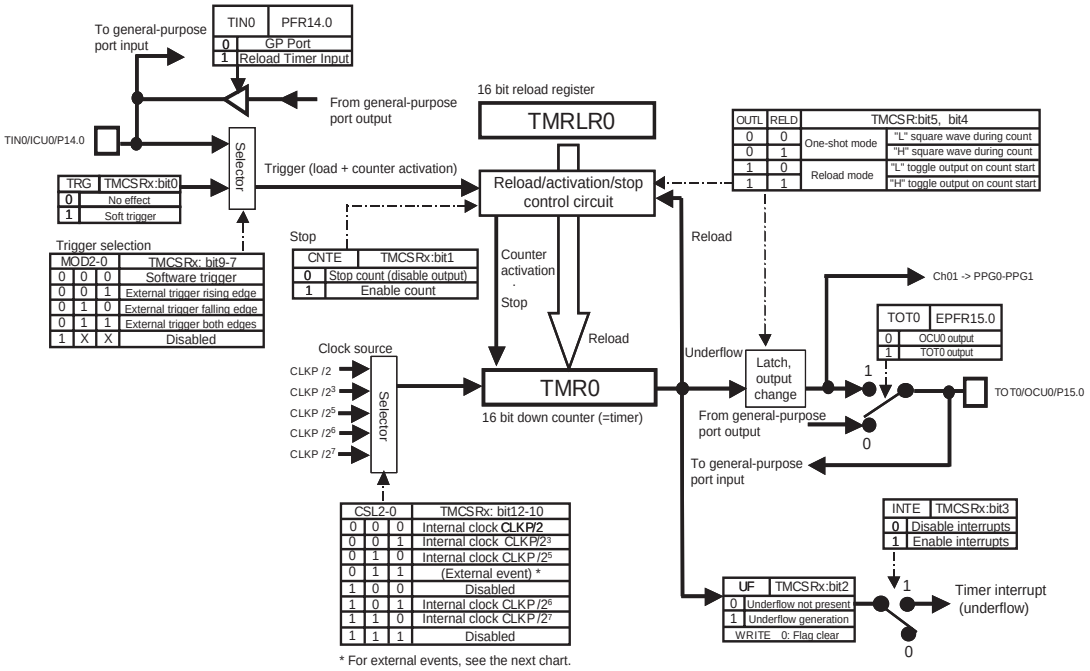


Figure 3-2 Configuration Diagram

Reload timer 0 (External event count)

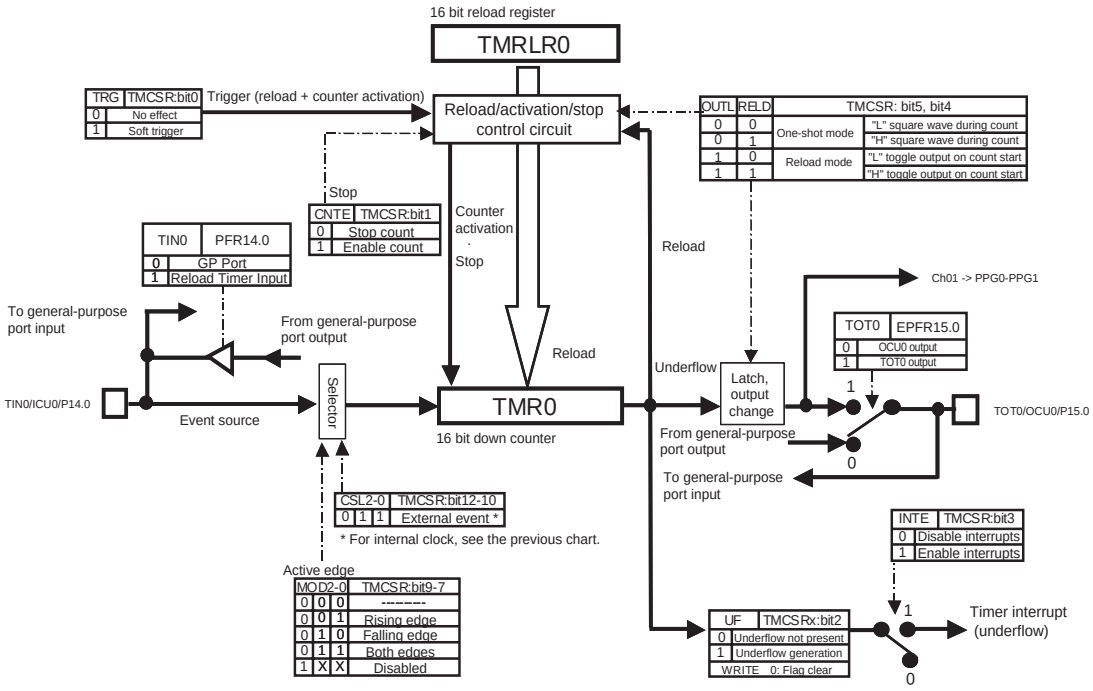
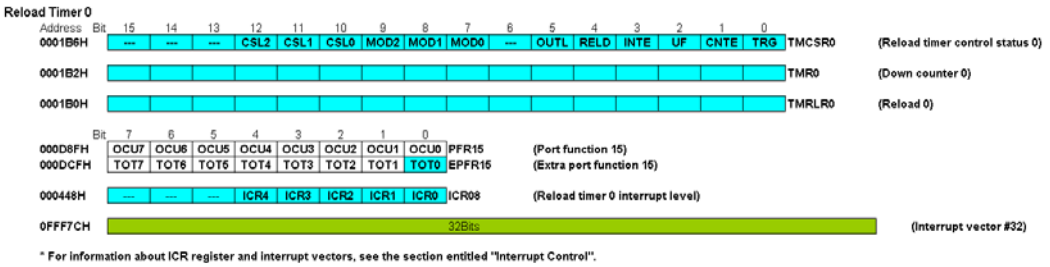


Figure 3-3 Register List



Note: For information about ICR registers and interrupt vectors, see “Chapter 24 Interrupt Control (Page No.315)”.

4. Registers

4.1 TMCSR: Reload Timer Control Status Register

The control status register controls the operation mode of the reload timer and interrupts.

- **TMCSR0 (Reload timer 0): Address: 001B6H (Access: Byte, Half-word)**
- **TMCSR1 (Reload timer 1): Address: 001BEH (Access: Byte, Half-word)**
- **TMCSR2 (Reload timer 2): Address: 001C6H (Access: Byte, Half-word)**
- **TMCSR3 (Reload timer 3): Address: 001CEH (Access: Byte, Half-word)**
- **TMCSR4 (Reload timer 4): Address: 001D6H (Access: Byte, Half-word)**
- **TMCSR5 (Reload timer 5): Address: 001DEH (Access: Byte, Half-word)**
- **TMCSR6 (Reload timer 6): Address: 001E6H (Access: Byte, Half-word)**
- **TMCSR7 (Reload timer 7): Address: 001EEH (Access: Byte, Half-word)**

15	14	13	12	11	10	9	8	bit
–	–	–	CSL2	CSL1	CSL0	MOD2	MOD1	
-	-	-	0	0	0	0	0	Initial Value
RX/WX	RX/WX	RX/WX	R0/WX	R/W	R/W	R/W0	R/W	Attribute
×	×	×	×	×	×	×	×	Rewrite during operation

7	6	5	4	3	2	1	0	bit
MOD0	–	OULT	RELD	INTE	UF	CNTE	TRG	
0	-	0	0	0	0	0	0	Initial Value
R/W	RX/WX	R/W	R/W	R/W	R(RM1),W	R/W	R0/W	Attribute
×	–	×	×	×	O	O	O	Rewrite during operation

(O: can be rewritten, x: cannot be rewritten)

(For information on attributes, see “Meaning of Bit Attribute Symbols (Page No.10)”.)

- bit15-14: Undefined

  Writing has no effect on the operation. The read value is “0”.

- bit13: Undefined (reload timer 0 - reload timer 2)

  Always write “0”. The read value is “0”.

- bit12-10: Count clock selection CLKP: peripheral clock

CSL2	CSL1	CSL0	Count clock	Remarks
0	0	0	Internal clock CLKP/2	
0	0	1	Internal clock CLKP/8	
0	1	0	Internal clock CLKP/32	
0	1	1	External event (external clock)	
1	0	1	Internal clock CLKP/64	
1	1	0	Internal clock CLKP/128	

Notes: Depending on whether an internal clock or an external event is selected, the meaning of the operation mode selection bit (MOD[2:0]) changes.

- bit9-7: Operation mode selection

  Reload trigger when internal clock is selected

MOD2	MOD1	MOD0	Reload trigger
0	0	0	Software trigger
0	0	1	External trigger (rising edge)
0	1	0	External trigger (falling edge)
0	1	1	External trigger (both edges)

  When the selected reload trigger is input, the value of reload register TMRLR is loaded to the down counter and the count operation is started.

  Count trigger when external event is selected

MOD2	MOD1	MOD0	Count trigger
0	0	0	--------
0	0	1	External trigger (rising edge)
0	1	0	External trigger (falling edge)
0	1	1	External trigger (both edges)

  Counts an external event using the selected count trigger.

  Always set MOD2 to “0”. The read value is the written value.

- bit6: Undefined

  Writing has no effect on the operation. The read value is “0”.

- bit5: Output level setting

OUTL	One-shot mode (RELD=“0”)	Reload mode (RELD=“1”)
0	During count “H” square wave	During count start “L” toggle output
1	During count “L” square wave	During count start “H” toggle output

  - During one-shot mode, a pulse is output during the count, and during reload mode a toggle is output.
  - For output level setting bit “0” and “1” the output level is reversed.

- bit4: Enable reload

RELD	Enable reload
0	One-shot mode (reload disabled)
1	Reload mode (reload enabled)

  - In reload mode, down counter underflow (0000H -> FFFFH) causes the value set to reload register (TMRLR) to be loaded to the down counter, and the count operation continues.
  - In one-shot mode, down counter underflow (0000H -> FFFFH) causes the count operation to stop.

- bit3: Enable timer interrupt requests

INTE	Enable timer interrupt requests
0	Disable interrupt requests
1	Enable interrupt requests

  When timer interrupt requests are enabled, the timer interrupt request flag (UF) becomes “1” and interrupt requests are generated.

- bit2: Timer interrupt request flag

UF	Timer interrupt request flag	
	When read	When write
0	Underflow not present	Clear interrupt requests
1	Underflow present	No effect

  Upon down counter underflow (0000H -> FFFFH) generation, the timer interrupt request flag becomes “1”. If the

interrupt request is enabled (INTE="1") an interrupt request is generated.

- bit1: Enable timer count

CNTE	Enable timer count
0	Stop count operation
1	Enable count operation (waiting for activation trigger)

If timer count is enabled, it waits for an activation trigger, and when an activation trigger is generated, the count operation starts. The activation trigger can be a software trigger or an external trigger.

- bit0: Software trigger

TRG	Software trigger
0	No effect. (The read value is “0”.)
1	Start count operation after data load.

If the count operation is enabled (CNTE=“1”) and the software trigger bit is set to “1”, the value of the reload register (TMRLR) is loaded to the down counter and the count operation starts.
If the count operation is not enabled (CNTE=“0”), the software trigger has no effect.

4.2 TMR: Timer Register

- **TMR0 (Reload timer 0): Address: 01B2H (Access: Half-word)**
- **TMR1 (Reload timer 1): Address: 01BAH (Access: Half-word)**
- **TMR2 (Reload timer 2): Address: 01C2H (Access: Half-word)**
- **TMR3 (Reload timer 3): Address: 01CAH (Access: Half-word)**
- **TMR4 (Reload timer 4): Address: 01D2H (Access: Half-word)**
- **TMR5 (Reload timer 5): Address: 01DAH (Access: Half-word)**
- **TMR6 (Reload timer 6): Address: 01E2H (Access: Half-word)**
- **TMR7 (Reload timer 7): Address: 01EAH (Access: Half-word)**

15	14	13	12	11	10	9	8	bit
D15	D14	D13	D12	D11	D10	D9	D8	
X	X	X	X	X	X	X	X	Initial Value
R/WX	R/WX	R/WX	R/WX	R/WX	R/WX	R/WX	R/WX	Attribute

7	6	5	4	3	2	1	0	bit
D7	D6	D5	D4	D3	D2	D1	D0	
X	X	X	X	X	X	X	X	Initial Value
R/WX	R/WX	R/WX	R/WX	R/WX	R/WX	R/WX	R/WX	Attribute

(For information on attributes, see “Meaning of Bit Attribute Symbols (Page No.10)”.)
The reload timer count value can be read out through the timer register TMR.
Please perform the read out using half-word access.

4.3 TMRLR: Reload register

- **TMRLR0 (Reload timer 0): Address: 01B0H (Access: Half-word)**
- **TMRLR1 (Reload timer 1): Address: 01B8H (Access: Half-word)**
- **TMRLR2 (Reload timer 2): Address: 01C0H (Access: Half-word)**
- **TMRLR3 (Reload timer 3): Address: 01C8H (Access: Half-word)**
- **TMRLR4 (Reload timer 4): Address: 01D0H (Access: Half-word)**
- **TMRLR5 (Reload timer 5): Address: 01D8H (Access: Half-word)**
- **TMRLR6 (Reload timer 6): Address: 01E0H (Access: Half-word)**
- **TMRLR7 (Reload timer 7): Address: 01E8H (Access: Half-word)**

15	14	13	12	11	10	9	8	bit
D15	D14	D13	D12	D11	D10	D9	D8	
X	X	X	X	X	X	X	X	Initial Value
RX/W	RX/W	RX/W	RX/W	RX/W	RX/W	RX/W	RX/W	Attribute

7	6	5	4	3	2	1	0	bit
D7	D6	D5	D4	D3	D2	D1	D0	

X	X	X	X	X	X	X	X	Initial Value
RX/W	RX/W	RX/W	RX/W	RX/W	RX/W	RX/W	RX/W	Attribute

(For information on attributes, see “Meaning of Bit Attribute Symbols (Page No.10)”.)

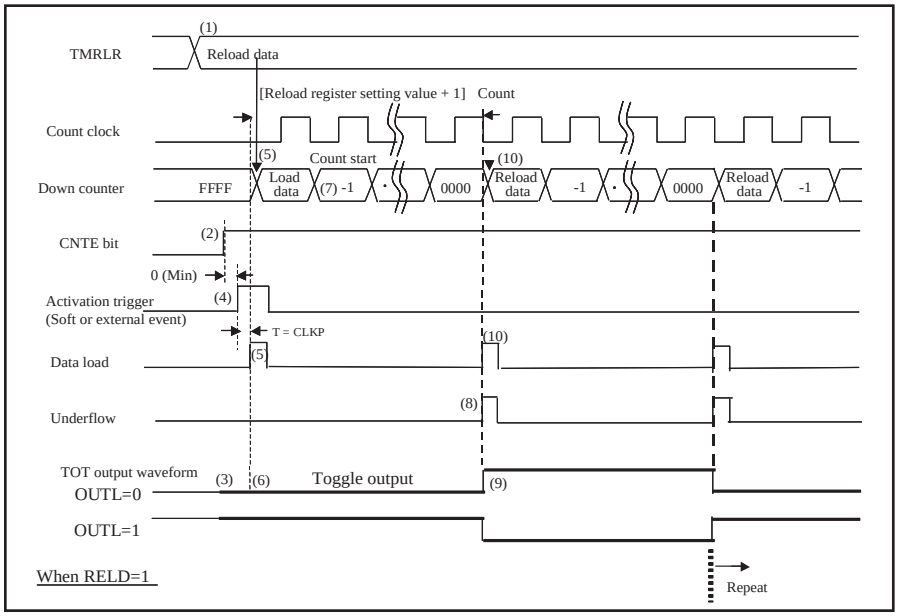
The reload value for the down counter is stored in reload register TMRLR.

Please write using half-word access.

5. Operation

5.1 Internal Clock/Reload Mode

In reload mode, a pulse with a 50% duty ratio is output.

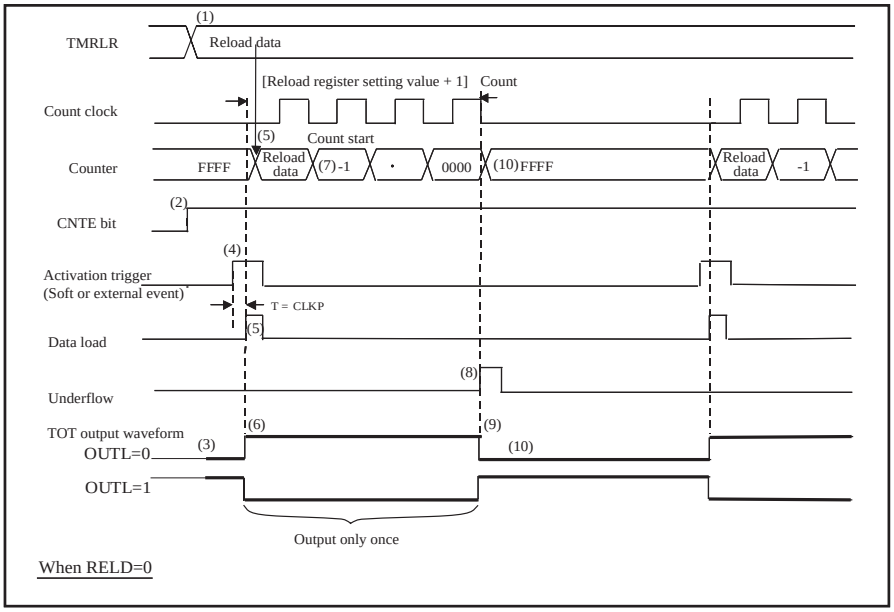


(1) Set reload value to reload register
(2) Enable reload timer count operation
(3) TOT pin output
(4) Generate reload trigger (activation): soft trigger or external event trigger
(5) Load reload value
(6) TOT toggle output start
(7) Counter count down (internal clock synchronous)
(8) Generate counter underflow
(9) TOT pin output level reversal (toggle output)
(10) Reload reload value
(11) Repeat steps (7) to (10)

(See “8. Caution (Page No.796)”.)

5.2 Internal Clock/One-shot Mode

In one-shot mode, a one-shot pulse is output.

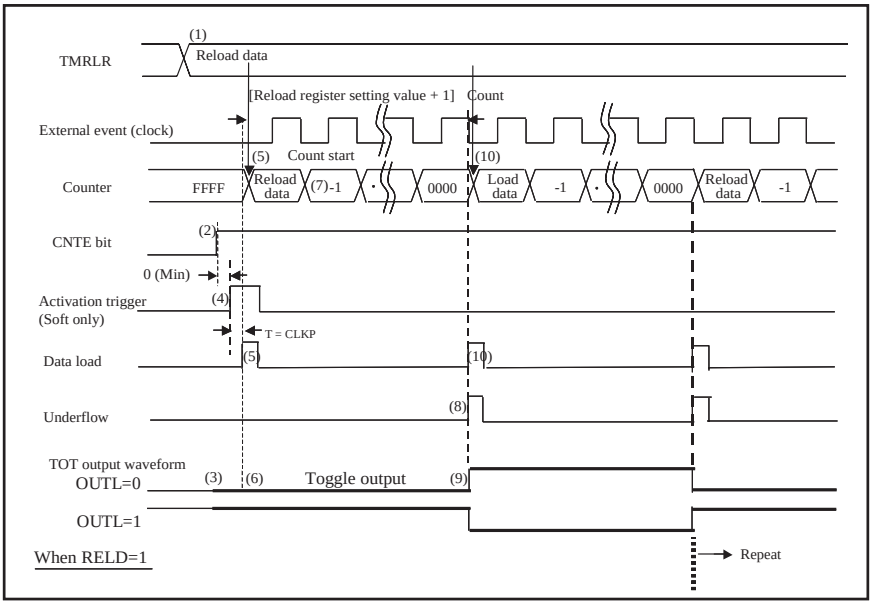


(1) Set reload value to reload register
(2) Enable reload timer count operation
(3) TOT pin output
(4) Generate reload trigger (activation): soft trigger or external event trigger
(5) Load reload value
(6) Square wave output (during count, “H” output/OUTL=“0”)
(7) Counter count down (internal clock synchronous)
(8) Generate counter underflow
(9) Return TOT pin output level
(10) Count stop, wait for next activation trigger

(See “8. Caution (Page No.796)”.)

5.3 External Event Clock Reload Mode

External event reload mode counts external events and outputs a pulse with a 50% duty ratio.

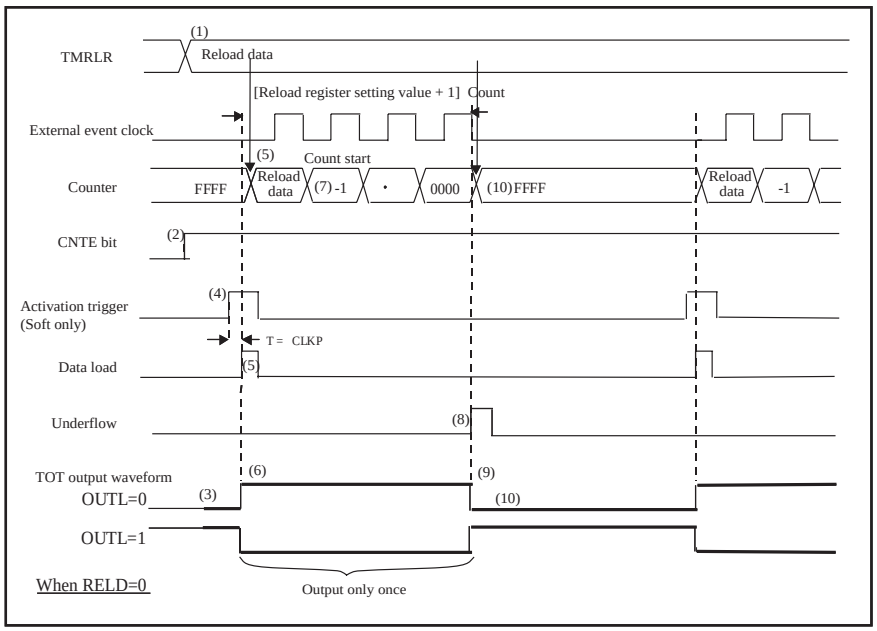


(1) Set reload value to reload register
(2) Enable reload timer count operation
(3) TOT pin output
(4) Generate reload trigger (activation): software trigger only
(5) Load reload value
(6) TOT pin output (initial value)
(7) Counter count down (external event synchronous)
(9) TOT pin output level reversal
(10) Reload reload value
(11) Repeat steps (6) to (9)

(See “8. Caution (Page No.796)”.)

5.4 External Event Clock/One-shot Mode

In external event one-shot mode, external events are counted and a one-shot pulse is output.



(1) Set reload value to reload register
(2) Enable reload timer count operation
(3) TOT pin output
(4) Generate reload trigger (activation): soft trigger only
(5) Load reload value
(6) TOT pin output (during count, output “H”/OUTL=“0”)
(7) Counter count down (via external events)
(8) Generate counter underflow
(9) TOT pin output reversal
(10) Stop counter, wait for next activation trigger

Note: The first reload will be delayed by a maximum of 1 T (T: count clock).

5.5 Operation during Reset

A reset (reset on INITX signal, watchdog reset, software reset) will cause the registers in the reload timer to be initialized. The initial value of reload registers is indeterminate.

For detailed information on initial values, see the explanation of registers.

5.6 Operation during Sleep Mode

Even after making the transition to sleep mode, the operation of the reload timer will continue.

5.7 Operation during Stop Mode

When the transition is made to stop mode, the operation of the reload timer stops.

Afterwards, when returning from stop mode, it will return to the state it was in before the transition to stop mode.

5.8 Operation when Returning from Stop Mode

When returning due to an external interrupt, the reload timer will continue operation from its stopped state.

When returning from a reset (INITX), it will return to the initial state (down counter stopped, no TOT pin output).

5.9 Status Transition

The status of the counter is decided by the CNTE bit of the reload timer control register and the internal WAIT signal.

Settable statuses are:

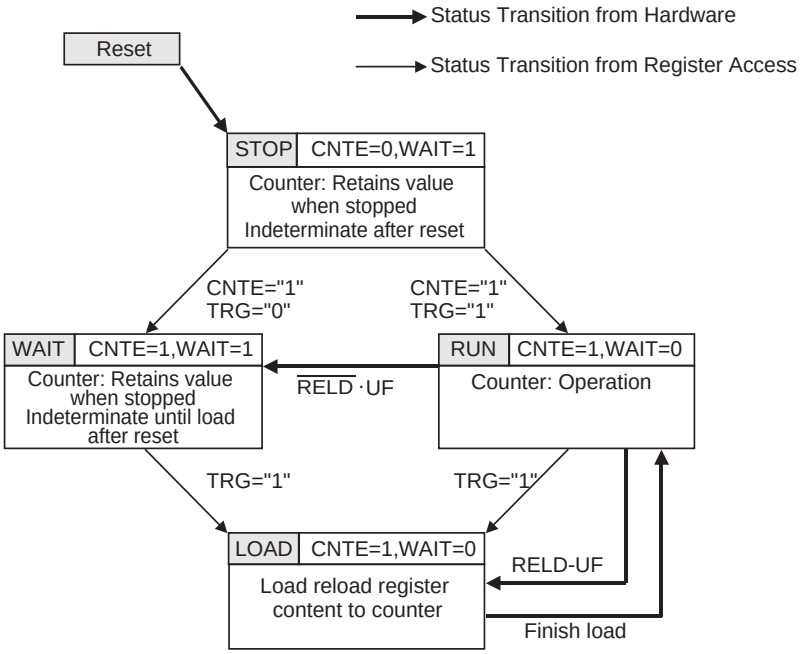
STOP status: Stopped (CNTE="0", WAIT="1")

WAIT status: Waiting for activation trigger (CNTE="1", WAIT="1")

RUN status : Count operation running (CNTE="1", WAIT="0")

LOAD status: Loading value to counter (from RUN/WAIT, TRG="1" or underflow: CNTE="1", WAIT="0")

Figure 5-1 Status Transition Diagram



6. Setting

Table 6-1 Settings Necessary for Moving the Reload Timer (Internal Clock Operation)

Setting	Setting Registers	Setting Procedure*
Reload value settings	Reload (TMRLR0-TMRLR7)	See7.1
Count clock selection (internal clock selection)	Reload timer control status (TMCSR0-TMCSR7)	See 7.2
Enable reload timer count operation		See 7.3
Mode selection (reload /one-shot)		See 7.4
Output reversal specification		See 7.5
Reload trigger selection (activation selection) Soft trigger External trigger (Rising edge/falling edge/both edges)		See 7.6
TOT0 - TOT7 pin output	(Extra) port function register (PFR15.0 - PFR15.7 and EPFR15.0 - EPFR15.7)	See 7.8
Generate activation trigger	–	See 7.10
Soft trigger -> Software trigger bit setting	Reload timer control status (TMCSR0-TMCSR7)	
External trigger -> Input trigger to TIN pin	External input	

*: For the setting procedure, refer to the section indicated by the number.

Table 6-2 Settings Necessary for Moving the Reload Timer (External Event Operation)

Setting	Setting Registers	Setting Procedure*
Reload value setting	Reload (TMRLR0-TMRLR7)	See 7.1
Count clock selection (external event clock selection)	Reload timer control status (TMCSR0-TMCSR7)	See 7.2
Enable reload timer count operation		See 7.3
Mode selection (reload /one-shot)		See 7.4
Output reversal specification		See 7.5
External event clock active edge selection (Rising edge/falling edge/both edges)		See 7.7
TOT0 - TOT7 pin output	(Extra) port function register (PFR15.0 - PFR15.7 and EPFR15.0 - EPFR15.7)	See 7.8
TIN0 - TIN7 pin external event input	Port function register (PFR14.0 - PFR14.7)	See 7.9
Generate activation trigger Soft trigger -> Software trigger bit setting	Reload timer control status (TMCSR0-TMCSR7)	See 7.10

*: For the setting procedure, refer to the section indicated by the number.

Table 6-3 Items Necessary for Performing Reload Timer Interrupts

Setting	Setting Registers	Setting Procedure*
Reload timer interrupt vector Reload timer interrupt level setting	See "Chapter 24 Interrupt Control (Page No.315)"	See 7.11
Reload timer interrupt settings Interrupt request clear Enable interrupt requests	Reload timer control status (TMCSR0-TMCSR7)	See 7.12

*: For the setting procedure, refer to the section indicated by the number.

Table 6-4 Settings Necessary for Stopping the Reload Timer

Setting	Setting Registers	Setting Procedure*
Reload timer stop bit setting	Reload timer control status (TMCSR0-TMCSR7)	See 7.13

*: For the setting procedure, refer to the section indicated by the number.

7. Q & A

7.1 What is the reload value setting (rewriting) procedure?

The reload value is set by the 16 bit reload registers TMRLR0-TMRLR7.

The equation for the values to be set is as follows.

- Formula

  TMRLR register value = {reload interval/count clock}-1

- Allowed Range

  TMRLR register value = 0~FFFh (65535)

7.2 What are the kinds of count clocks and how are they selected?

The count clock is chosen from the 4 types in the table below.

Selection is done via the count clock selection bit.

Table 7-1 TMCSR.CSL[2:0]

Count Clock	Counter clock selection bit			Count clock example		
	CSL2	CSL1	CSL0	When CLKP= 32MHz	When CLKP= 16MHz	When CLKP= 8MHz
CLKP/2	0	0	0	62.5ns	125ns	250ns
CLKP/8	0	0	1	250ns	500ns	1.0µs
CLKP/32	0	1	0	1.0µs	2.0µs	4.0µs
External event	0	1	1	Pulse width: 2/CLKP min		
CLKP/64	1	0	1	2.0µs	4.0µs	8.0µs
CLKP/128	1	1	0	4.0µs	8.0µs	16.0µs
Disabled *	1	0	0	-----		
	1	1	1	-----		

(*: See “8. Caution (Page No.796)”.)

7.3 How to I enable/disable the reload timer count operation?

Use the timer count enable bit (TMCSR.CNTE).

Control Details	RLT operation permission bit (CNTE)
To stop the reload timer	Set to “0”
To enable the reload timer’s count operation	Set to “1”

Cannot be reopened from the stopped state. Enable before activation or simultaneous with activation.

7.4 How do I set the reload timer mode (reload/one-shot)?

Use mode selection bit (TMCSR.RELD).

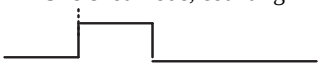
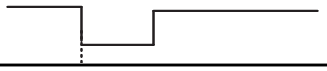
Operation Mode	Mode selection bit (RELD)
To set to one-shot mode	Set to “0”
To set to reload	Set to “1”

7.5 How do I reverse the output level?

The settings for the output level are detailed in the following table.

The setting is done via timer output level bit (TMCSR.OUTL).

Output level	Timer output level bit (OUTL)

Reload mode, Initial value “L” level output 	Set to “0”
Reload mode, initial value “H” level output (reversed) 	Set to “1”
One-shot mode, counting “H” level output 	Set to “0”
One-shot mode, counting “L” level output (reversed) 	Set to “1”

7.6 What are the kinds of triggers, and how do I select them?

- Selection is done via the trigger selection bit (TMCSR.MOD[2:0]).

  There are 4 types of reload triggers when an internal clock is selected.

Trigger	Trigger specification bit (MOD[2:0])
Software trigger (TRG bit set)	Set to “000”
External trigger from TINx pin (rising edge)	Set to “001”
External trigger from TINx pin (falling edge)	Set to “010”
External trigger from TINx pin (both edges)	Set to “011”
————————	“100”, “101”, “110”, “111” are disabled *

  Reload is repeated on down counter underflow.

  (*: See “8. Caution (Page No.796)”.)

- The reload trigger (activation) when an external event is selected is a software trigger.

  Reload is repeated on down counter underflow.

7.7 What are the types of external event clock active edges and how do I select them?

The setting is done via the trigger selection bit (TMCSR.MOD[1:0]).

There are three types of active edges.

Active edge	Trigger selection bit (MOD1-MOD0)
Rising edge	Set to “01”
Falling edge	Set to “10”
Both edges	Set to “11”

MOD2 settings have no meaning, no matter if they are set to “0” or “1”.

7.8 How do I make a pin a TOT output pin?

Write “1” to the TOT output selection bits (PFR15/EPFR15) to change the port to a TOT pin output.

Pin	Control bit	
TOT0 pin	PFR15.0 = ‘1’	EPFR15.0 = ‘1’
TOT1 pin	PFR15.1 = ‘1’	EPFR15.1 = ‘1’
TOT2 pin	PFR15.2 = ‘1’	EPFR15.2 = ‘1’
TOT3 pin	PFR15.3 = ‘1’	EPFR15.3 = ‘1’
TOT4 pin	PFR15.4 = ‘1’	EPFR15.4 = ‘1’
TOT5 pin	PFR15.5 = ‘1’	EPFR15.5 = ‘1’
TOT6 pin	PFR15.6 = ‘1’	EPFR15.6 = ‘1’
TOT7 pin	PFR15.7 = ‘1’	EPFR15.7 = ‘1’

7.9 How do I make the TIN pin into an external event input pin, or an external trigger input pin?

Write “1” to the TIN input selection bits (PFR14) to change the port to a TIN pin input.

Pin	Control bit	
TIN0 pin	PFR14.0 = ‘1’	-
TIN1 pin	PFR14.1 = ‘1’	-
TIN2 pin	PFR14.2 = ‘1’	-
TIN3 pin	PFR14.3 = ‘1’	-
TIN4 pin	PFR14.4 = ‘1’	-
TIN5 pin	PFR14.5 = ‘1’	-

TIN6 pin	PFR14.6 = ‘1’	-
TIN7 pin	PFR14.7 = ‘1’	-

7.10 How do I generate an activation trigger?

- Generating a soft trigger

  The setting is done via the software trigger bit (TMCSR.TRG).

  When the software trigger bit (TGR) is set to“1”, a trigger is generated.

  To enable operation and activate at the same time, set the count permission bit (TMCSR.CNTE) and the soft trigger bit (TMCSR.TRG) simultaneously.

- Generating an external trigger

  By inputting the edge specified by the trigger selection bit to the trigger pin corresponding to each reload timer, a trigger is generated.

Timer	Trigger pin
Reload timer 0	TIN0
Reload timer 1	TIN1
Reload timer 2	TIN2
Reload timer 3	TIN3
Reload timer 4	TIN4
Reload timer 5	TIN5
Reload timer 6	TIN6
Reload timer 7	TIN7

7.11 What are the interrupt-related registers?

The relationship between reload timer numbers, interrupt level, vector, control register, etc is outlined in the following table.

For details on interrupt level and interrupt vectors, see “Chapter 24 Interrupt Control (Page No.315)”.

	Interrupt vector (default)	Interrupt level setting bit (ICR[4:0])
Reload timer 0	#32 Address: 0FFF7Ch	Interrupt level register (ICR08) Address: 0448h
Reload timer 1	#33 Address: 0FFF78h	
Reload timer 2	#34 Address: 0FFF74h	Interrupt level register (ICR09) Address: 0449h
Reload timer 3	#35 Address: 0FFF70h	
Reload timer 4	#36 Address: 0FFF6Ch	Interrupt level register (ICR10) Address: 044Ah
Reload timer 5	#37 Address: 0FFF68h	
Reload timer 6	#38 Address: 0FFF64h	Interrupt level register (ICR11) Address: 044Bh
Reload timer 7	#39 Address: 0FFF60h	

Interrupt request flag (TMCSR0.UF) ~ (TMCSR7.UF) is not automatically cleared, so before returning from interrupt processing, set the UF bit to “0” to reset it.

7.12 How do I enable interrupts?

Enabling interrupts, interrupt request flag

Enabling of interrupts is done via the interrupt request permission bit (TMCSR0.INTE) ~ (TMCSR7.INTE).

	Interrupt request permission bit (INTE)
To disable interrupt requests	Set to “0”
To enable interrupt requests	Set to “1”

Clearing of interrupt requests is done via the interrupt request bit (TMCSR0.UF) ~ (TMCSR7.UF).

	Interrupt request bit (UF)
To disable interrupt requests	Set to “0”

7.13 How do I stop the reload timer?

This setting is done via the reload timer stop bit.

See “7.3 How to I enable/disable the reload timer count operation? (Page No.791)”.

8. Caution

- Count source select bit (TMCSR.CSL[2:0]) settings not in the table: “100”, “111” are disabled.
  If they are set, disable the reload timer operation before resetting the count source select bit.

- Operation mode bit (TMCSR.MOD2) must be set to “0”. If it is set to “1”, disable the reload timer count operation before resetting it. Also the value written during read/modify/write access may be read.

- Control bits (Count source select, operation mode, reload permission) must not be rewritten during operation.
  If they are set during operation, disable the reload timer count operation before resetting them.

- From activation timing, it takes T cycle for the reload value to be loaded to the down counter. (Cycle = 1/CLKP, CLKP = peripheral clock)

- About output signal internal connections
  - Reload timer TOT0-TOT7 outputs are connected to the PPG0-PPG15 internal trigger inputs.
  - Reload timer TOT7 output is connected to the A/D converter 0 trigger input.

- Rewriting of the count clock selection bit (CSL[2:0]), operation mode selection bit (MOD[2:0]), output level setting bit (OUTL), reload permission bit (RELD), and timer interrupt request permission bit (INTE) should be done when the reload timer is stopped (TMCSR.CNTE=“0”).

- The internal prescaler should be already set when the timer count permission bit (TMCSR.CNTE) is set to “1”.

- If interrupt request flag set timing and clear timing overlap, the flag setting will be given priority and the clear operation will be made invalid.

- When writing to the reload register and the reload timing overlap, the old data will be loaded to the counter. The new data will be loaded during the next reload timing.

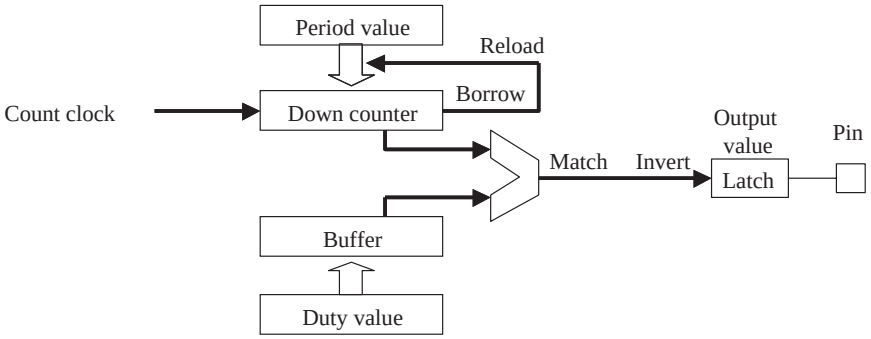
- If the loading and counting of the timer register overlap, the load (reload) operation is given priority.

- If you want to enable the count at the same time as you start the count operation, set both the timer count permission bit (TMCSR.CNTE) and the software trigger bit (TMCSR.TRG) to “1”.

Chapter 39 Programmable Pulse Generator

1. Overview

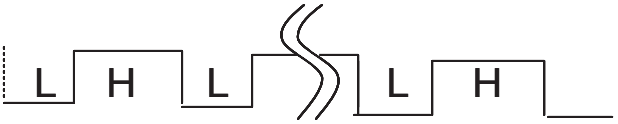
Programmable Pulse Generators (PPGs) are used to gain one-shot (rectangular wave) output or pulse width modulation (PWM) output. With their software-programmable cycle and duty capability, the PPGs comfortably fit into broad applications.



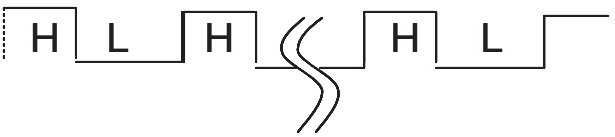
2. Features

- Output waveforms: The PPGs can generate the following six kinds of waveforms:
  - PWM waveform

    Normal polarity:

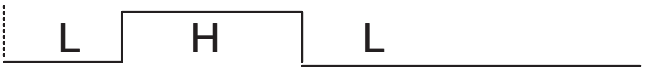
    

    Inverted polarity:

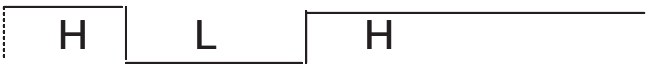
    

- One-shot waveform (Rectangular wave)

  Normal polarity:

  Inverted polarity:

- Clamped output

  Normal polarity: “L” Clamped output

  Inverted polarity: “H” Clamped output
- Quantity: 4 groups (Output: 16 channels PPG0 - PPG15)
- Count clock: Choose from four choices.

  1, 1/4, 1/16, 1/64 of the peripheral clock (CLKP)
- Period: Setting range = Duty value ~ 65535 (specified with a 16-bit register)

  Period = Count clock (PCSR register value + 1)

  (Example) Count clock = 32MHz(31.25ns), PCSR value = 63999

  Period = 31.25ns (63999+1) = 2ms
- Duty: Setting range = 0 ~ Period value (specified with a 16-bit register)

  Duty = Count clock (PDUT register value + 1)

- Interrupt: Choose from four choices:
  - Software trigger
  - Counter borrow (cycle match)
  - Duty match
  - Counter borrow (cycle match) or duty match
- Activation trigger:
  - Software trigger
  - Internal triggers
    Reload timer output 0 (TOT0) available as trigger for PPG0-PPG3
    Reload timer output 1 (TOT1) available as trigger for PPG0-PPG3
    Reload timer output 2 (TOT2) available as trigger for PPG4-PPG7
    Reload timer output 3 (TOT3) available as trigger for PPG4-PPG7
    Reload timer output 4 (TOT4) available as trigger for PPG8-PPG11
    Reload timer output 5 (TOT5) available as trigger for PPG8-PPG11
    Reload timer output 6 (TOT6) available as trigger for PPG12-PPG15
    Reload timer output 7 (TOT7) available as trigger for PPG12-PPG15
  - External triggers
    Port GP14_0 (ICU0, RLT0 ext-trig) available as trigger for PPG0-PPG3/PPG8-PP11
    Port GP14_1 (ICU1, RLT1 ext-trig) available as trigger for PPG0-PPG3/PPG8-PP11
    Port GP14_2 (ICU2, RLT2 ext-trig) available as trigger for PPG0-PPG3/PPG8-PP11
    Port GP14_3 (ICU3, RLT3 ext-trig) available as trigger for PPG0-PPG3/PPG8-PP11
    Port GP14_4 (ICU4, RLT4 ext-trig) available as trigger for PPG4-PPG7/PPG12-PP15
    Port GP14_5 (ICU5, RLT5 ext-trig) available as trigger for PPG4-PPG7/PPG12-PP15
    Port GP14_6 (ICU6, RLT6 ext-trig) available as trigger for PPG4-PPG7/PPG12-PP15
    Port GP14_7 (ICU7, RLT7 ext-trig) available as trigger for PPG4-PPG7/PPG12-PP15

3. Configuration

Figure 3-1 Configuration Diagram

PPG (0-3)

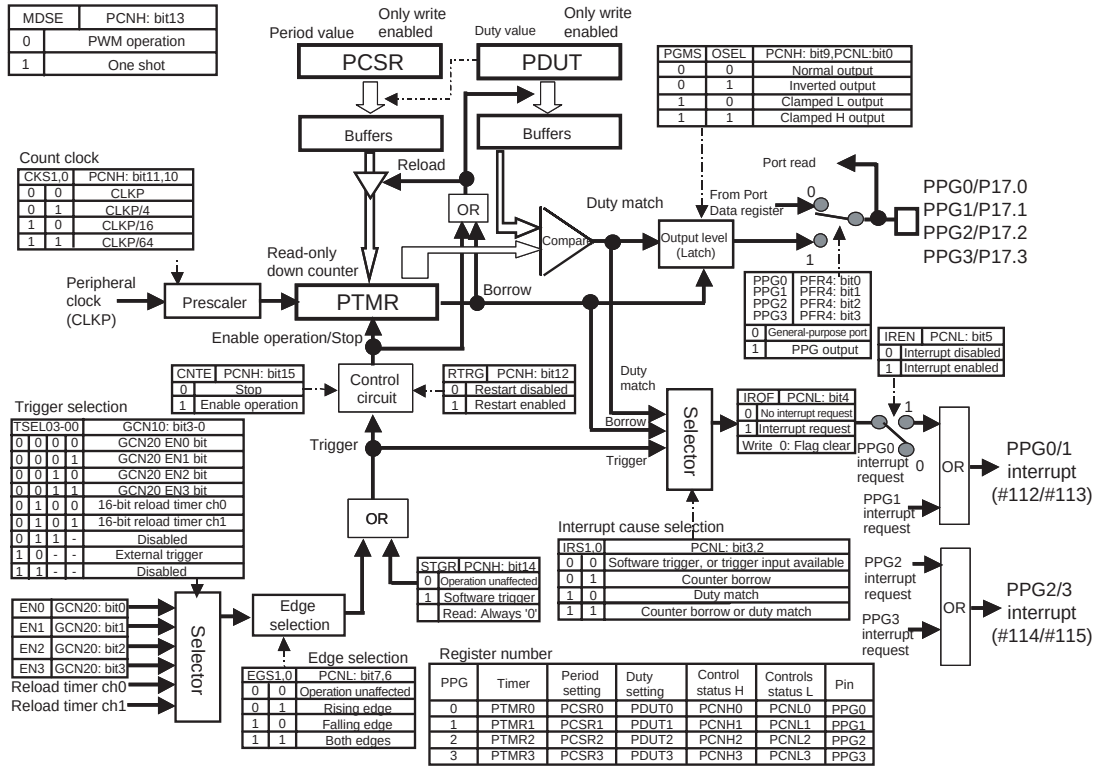
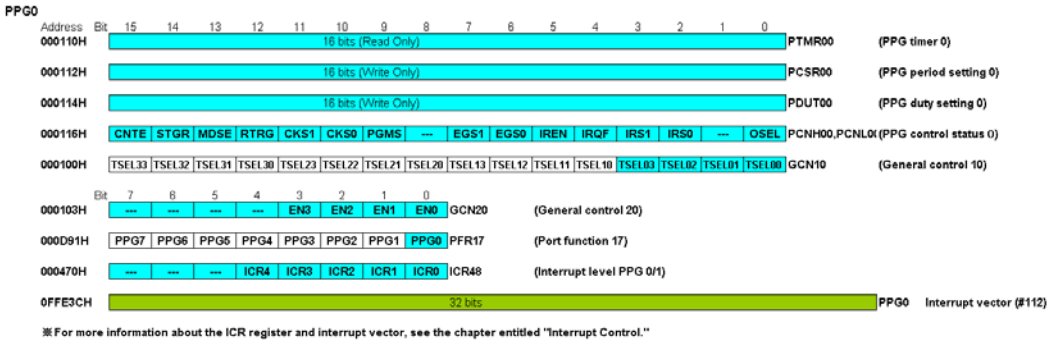
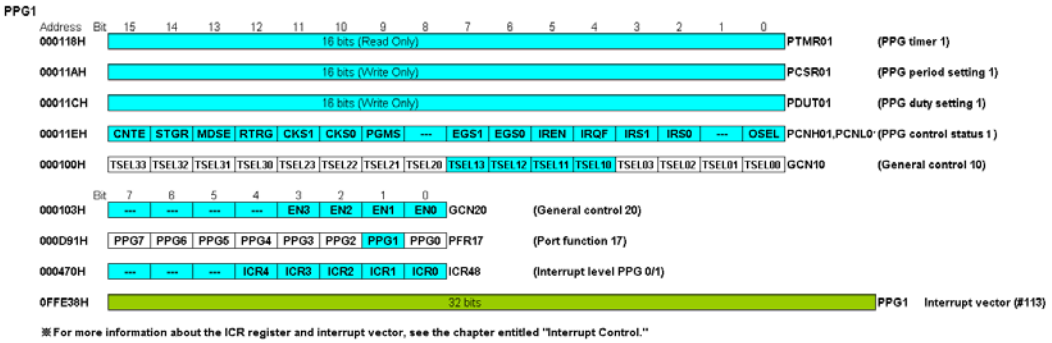
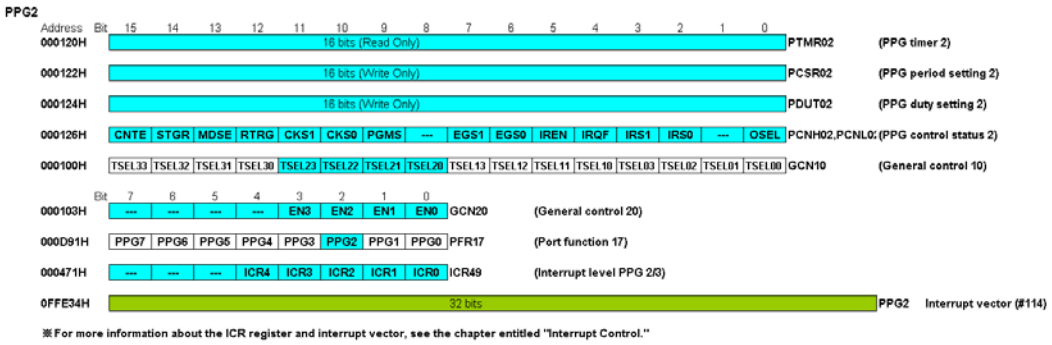
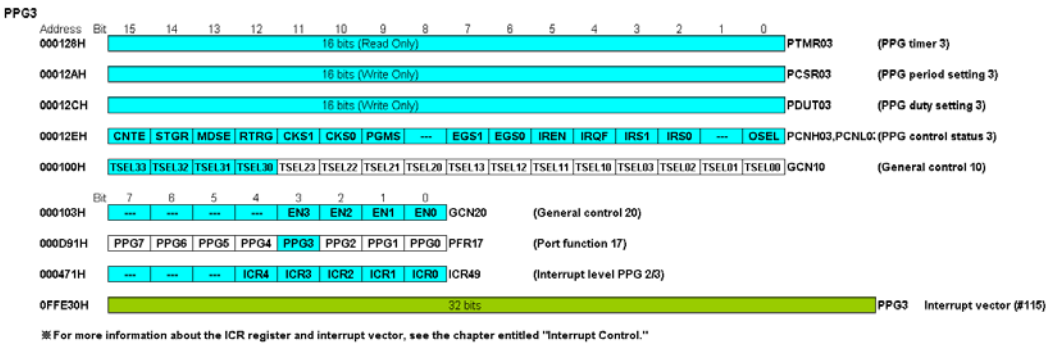


Figure 3-2 Register List









Note: For more information about the ICR register and interrupt vector, see “Chapter 24 Interrupt Control (Page No.315)”.

4. Registers

4.1 PCSR: PPG Cycle Setting Register

Controls the cycle of the PPG.

- **PCSR00 (PPG0): Address 0112h (Access: Half-word)**
- **PCSR01 (PPG1): Address 011Ah (Access: Half-word)**
- **PCSR02 (PPG2): Address 0122h (Access: Half-word)**
- **PCSR03 (PPG3): Address 012Ah (Access: Half-word)**
- **PCSR04 (PPG4): Address 0132h (Access: Half-word)**
- **PCSR05 (PPG5): Address 013Ah (Access: Half-word)**
- **PCSR06 (PPG6): Address 0142h (Access: Half-word)**
- **PCSR07 (PPG7): Address 014Ah (Access: Half-word)**
- **PCSR08 (PPG8): Address 0152h (Access: Half-word)**
- **PCSR09 (PPG9): Address 015Ah (Access: Half-word)**
- **PCSR10 (PPG10): Address 0162h (Access: Half-word)**
- **PCSR11 (PPG11): Address 016Ah (Access: Half-word)**
- **PCSR12 (PPG12): Address 0332h (Access: Half-word)**
- **PCSR13 (PPG13): Address 033Ah (Access: Half-word)**
- **PCSR14 (PPG14): Address 0342h (Access: Half-word)**
- **PCSR15 (PPG15): Address 034Ah (Access: Half-word)**

15	14	13	12	11	10	9	8	Bit
D15	D14	D13	D12	D11	D10	D9	D8	
X	X	X	X	X	X	X	X	Initial value
RX, W	RX, W	RX, W	RX, W	RX, W	RX, W	RX, W	RX, W	Attribute

7	6	5	4	3	2	1	0	Bit
D7	D6	D5	D4	D3	D2	D1	D0	
X	X	X	X	X	X	X	X	Initial value
RX, W	RX, W	RX, W	RX, W	RX, W	RX, W	RX, W	RX, W	Attribute

(See “Meaning of Bit Attribute Symbols (Page No.10)” for details of the attributes.)

- The PPG Period Setting registers are buffered. Transfers from the buffers to the counter take place automatically at counter overflow or underflow.
- After the PPG Period Setting registers have been written, be sure to set PPG Duty Setting registers PDUT.
- Always access the PPG Period Setting registers in a half-word (16-bit) format.

(See “8. Caution (Page No.823)”.)

4.2 PDUT: PPG Duty Setting Register

Sets the duty of the PPG output waveform.

- **PDUT00 (PPG0): Address 0114h (Access: Half-word)**
- **PDUT01 (PPG1): Address 011Ch (Access: Half-word)**
- **PDUT02 (PPG2): Address 0124h (Access: Half-word)**
- **PDUT03 (PPG3): Address 012Ch (Access: Half-word)**
- **PDUT04 (PPG4): Address 0134h (Access: Half-word)**
- **PDUT05 (PPG5): Address 013Ch (Access: Half-word)**
- **PDUT06 (PPG6): Address 0144h (Access: Half-word)**
- **PDUT07 (PPG7): Address 014Ch (Access: Half-word)**
- **PDUT08 (PPG8): Address 0154h (Access: Half-word)**
- **PDUT09 (PPG9): Address 015Ch (Access: Half-word)**
- **PDUT10 (PPG10): Address 0164h (Access: Half-word)**
- **PDUT11 (PPG11): Address 016Ch (Access: Half-word)**
- **PDUT12 (PPG12): Address 0334h (Access: Half-word)**
- **PDUT13 (PPG13): Address 033Ch (Access: Half-word)**
- **PDUT14 (PPG14): Address 0344h (Access: Half-word)**
- **PDUT15 (PPG15): Address 034Ch (Access: Half-word)**

15	14	13	12	11	10	9	8	Bit
D15	D14	D13	D12	D11	D10	D9	D8	
X	X	X	X	X	X	X	X	Initial value
RX, W	RX, W	RX, W	RX, W	RX, W	RX, W	RX, W	RX, W	Attribute

7	6	5	4	3	2	1	0	Bit
D7	D6	D5	D4	D3	D2	D1	D0	
X	X	X	X	X	X	X	X	Initial value
RX, W	RX, W	RX, W	RX, W	RX, W	RX, W	RX, W	RX, W	Attribute

(See “Meaning of Bit Attribute Symbols (Page No.10)” for details of the attributes.)

- The PPG Duty Setting registers registers are buffered. Transfers from the buffers to the counter take place automatically at counter overflow or underflow.
- Set a value smaller than the setting of PPG Period Setting register PCSR in a PPG Duty Setting register. (See “8. Caution (Page No.823)”.)
- If the same value as set in PPG Period Setting register PCSR is set in a PPG Duty Setting register,
  - “H” is always output to (OSEL=”0”) at normal polarity time.
  - “L” is always output to (OSEL=”1”) at inverted polarity time.

    (The OSEL bit is an output polarity specification bit of the PPG control register PCN.)
- Always access the PPG Duty Setting registers in a half-word (16-bit) format.
  (See “8. Caution (Page No.823)”.)

4.3 PCN: PPG Control Status register

Controls the operations and status of PPGs.

- **PCN00 (PPG0): Address 0116h (Access: Byte, Half-word)**
- **PCN01 (PPG1): Address 011Eh (Access: Byte, Half-word)**
- **PCN02 (PPG2): Address 0126h (Access: Byte, Half-word)**
- **PCN03 (PPG3): Address 012Eh (Access: Byte, Half-word)**
- **PCN04 (PPG4): Address 0136h (Access: Byte, Half-word)**
- **PCN05 (PPG5): Address 013Eh (Access: Byte, Half-word)**
- **PCN06 (PPG6): Address 0146h (Access: Byte, Half-word)**
- **PCN07 (PPG7): Address 014Eh (Access: Byte, Half-word)**
- **PCN08 (PPG8): Address 0156h (Access: Byte, Half-word)**
- **PCN09 (PPG9): Address 015Eh (Access: Byte, Half-word)**
- **PCN10 (PPG10): Address 0166h (Access: Byte, Half-word)**
- **PCN11 (PPG11): Address 016Eh (Access: Byte, Half-word)**
- **PCN12 (PPG12): Address 0336h (Access: Byte, Half-word)**
- **PCN13 (PPG13): Address 033Eh (Access: Byte, Half-word)**
- **PCN14 (PPG14): Address 0346h (Access: Byte, Half-word)**
- **PCN15 (PPG15): Address 034Eh (Access: Byte, Half-word)**

15	14	13	12	11	10	9	8	Bit
CNTE	STGR	MDSE	RTRG	CKS1	CKS0	PGMS	–	
0	0	0	0	0	0	0	X	Initial value
R/W	R0/W	R/W	R/W	R/W	R/W	R/W	RX/WX	Attribute
O	O	×	×	×	×	×	–	Rewrite during operation

7	6	5	4	3	2	1	0	Bit
EGS1	EGS0	IREN	IRQF	IRS1	IRS0	–	OSEL	
0	0	0	0	0	0	X	0	Initial value
R/W	R/W	R/W	R(RM1), W	R/W	R/W	RX/WX	R/W	Attribute
×	×	O	O	×	×	–	×	Rewrite during operation

(See “Meaning of Bit Attribute Symbols (Page No.10)” for details of the attributes.)

O: Rewritable, x: Not writable (See “8. Caution (Page No.823)”.)

- Bit 15: Timer enable operation

CNTE	Operation
0	Stop
1	Operation

This bit enables the operation of the PPG.

- Bit 14: Software trigger

STGR	Operation
0	The operation is unaffected by writing (The read value always equals “0”).
1	Software trigger activation

When the Software Trigger bit is set to “1”, a software trigger is generated to activate the PPG, separately from the generation of an internal trigger (EN bit, reload timer output).

- Bit 13: Mode selection

MDSE	Mode
0	PWM operation
1	One-shot operation

  - When the Mode Selection bit is set to “0”, a PWM operation is enabled to generate pulses in sequence.
  - When the Mode Selection bit is set to “1”, pulse output takes place only once.

- Bit 12: Restart enable

RTRG	Operation
0	Disable restart.
1	Enable restart.

When the Enable Restart bit is set to “1“, a trigger (software/internal) leads to a restart of the PPG, (even if the PPG is enabled (CNTE = “1“)). If the Enable Restart bit is set to “0” the trigger has no effect on the fuction of the PPG.

- Bits 11-10: Counter clock selection

CKS1	CKS0	Down Counter Count Clock Selection
0	0	Peripheral clock (CLKP)
0	1	Peripheral clock divided by 4
1	0	Peripheral clock divided by 16
1	1	Peripheral clock divided by 64

- Bit 9: PPG output mask selection

PGMS	Operation
0	No output mask
1	Output mask (Output “L” level latched:OSEL=“0”)

  - When the PPG Output Mask Selection bit is set to “1”, the PPG output can be clamped at “L” or “H” regardless of the mode, cycle, and duty settings.
  - The output level can be specified using the Output Polarity Specification bit (PCN.OSEL).

- Bit 8: Undefined.The operation is unaffected by writing. The read value is indeterminate.
- Bits 7-6: Trigger input edge selection

EGS1	EGS0	Selected Edge
0	0	The operation is unaffected by writing.
0	1	Rising edge
1	0	Falling edge
1	1	Both edges (rising edge, or, falling edge)

Select an edge to trigger the activation of the trigger input selected with the Trigger Specification bits
(GCN10[15:12]), (GCN10[11:8]), (GCN10[7:4]), and (GCN10[3:0]) of PPG3 to PPG0,
(GCN11[15:12]), (GCN11[11:8]), (GCN11[7:4]), and (GCN11[3:0]) of PPG7 to PPG4,
(GCN12[15:12]), (GCN12[11:8]), (GCN12[7:4]), and (GCN12[3:0]) of PPG11 to PPG8,
(GCN13[15:12]), (GCN13[11:8]), (GCN13[7:4]), and (GCN13[3:0]) of PPG15 to PPG12,
using the Trigger Input Edge Selection bit (EGS[1:0]).

- Bit 5: Interrupt request enable

IREN	Operation
0	Interrupt request disable
1	Enable interrupt requests.

- Bit 4: interrupt request flag

IRQF	Read Operation	Write Operation

0	No interrupt request	Clear the Interrupt Request flag.
1	Interrupt request	The operation is unaffected by writing.

If the Interrupt Request flag (IRQF) equals “1” and writing “0” to the flag take place at the same time, the setting of the Interrupt Request flag (IRQF=“1”) overrides.

- Bit 3-2: Interrupt cause selection

IRS1	IRS0	Selection
0	0	Software trigger, or, trigger input
0	1	Counter borrow
1	0	The counter matches the duty value.
1	1	Counter borrow, or the counter equals the duty value.

  - Select the operation in which to generate an interrupt request.
- Bit 1: Undefined.The operation is unaffected by writing. The read value is indeterminate.
- Bit 0: PPG output polarity specification

OSEL	Operation
0	Normal polarity
1	Inverted polarity

When the PPG Output Mask Selection bit (PCN.PGMS) has been set to “1”, if the Output Polarity Specification bit (OSEL) is set to “0”, the output is clamped at “L”; if the Output Polarity Specification bit is set to “1”, the output is clamped at “H”.

4.4 GCN1: General Control register 1

Selects a trigger input to PPG0-PPG3, PPG4-PPG7, PPG8-PPG11 and PPG12-PPG15.

- **GCN10 (PPG0-PPG3): Address 0100h (Access: Half-word)**
- **GCN11 (PPG4-PPG7): Address 0104h (Access: Half-word)**
- **GCN12 (PPG8-PPG11): Address 0108h (Access: Half-word)**
- **GCN13 (PPG12-PPG15): Address 0320h (Access: Half-word)**

15	14	13	12	11	10	9	8	Bit
TSEL33	TSEL32	TSEL31	TSEL30	TSEL23	TSEL22	TSEL21	TSEL20	
0	0	1	1	0	0	1	0	Initial value
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Attribute

7	6	5	4	3	2	1	0	Bit
TSEL13	TSEL12	TSEL11	TSEL10	TSEL03	TSEL02	TSEL01	TSEL00	
0	0	0	1	0	0	0	0	Initial value
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Attribute

(See “Meaning of Bit Attribute Symbols (Page No.10)” for details of the attributes.)

PPG12-PPG15:
- •GCN13: Bits 15-12 (TSEL3[3:0]) PPG15 trigger specification
- •GCN13: Bits 11-8 (TSEL2[3:0]) PPG14 trigger specification
- •GCN13: Bits 7-4 (TSEL1[3:0]) PPG13 trigger specification
- •GCN13: Bits 3-0 (TSEL0[3:0]) PPG12 trigger specification

PPG8-PPG11:
- •GCN12: Bits 15-12 (TSEL3[3:0]) PPG11 trigger specification
- •GCN12: Bits 11-8 (TSEL2[3:0]) PPG10 trigger specification
- •GCN12: Bits 7-4 (TSEL1[3:0]) PPG9 trigger specification
- •GCN12: Bits 3-0 (TSEL0[3:0]) PPG8 trigger specification

PPG4-PPG7:
- •GCN11: Bits 15-12 (TSEL3[3:0]) PPG7 trigger specification
- •GCN11: Bits 11-8 (TSEL2[3:0]) PPG6 trigger specification
- •GCN11: Bits 7-4 (TSEL1[3:0]) PPG5 trigger specification
- •GCN11: Bits 3-0 (TSEL0[3:0]) PPG4 trigger specification

PPG0-PPG3:
- •GCN10: Bits 15-12 (TSEL3[3:0]) PPG3 trigger specification
- •GCN10: Bits 11-8 (TSEL2[3:0]) PPG2 trigger specification
- •GCN10: Bits 7-4 (TSEL1[3:0]) PPG1 trigger specification
- •GCN10: Bits 3-0 (TSEL0[3:0]) PPG0 trigger specification

TSEL				Activation trigger specification
0	0	0	0	EN0 bit (GCN2 register)
0	0	0	1	EN1 bit (GCN2 register)
0	0	1	0	EN2 bit (GCN2 register)
0	0	1	1	EN3 bit (GCN2 register)
0	1	0	0	16-bit reload timer 0/2/4/6
0	1	0	1	16-bit reload timer 1/3/5/7
1	0	0	0	External trigger 0/4
1	0	0	1	External trigger 1/5
1	0	1	0	External trigger 2/6
1	0	1	1	External trigger 3/7
None of the above				Disabled (See “8. Caution (Page No.823)”.)

- PPG0 to PPG15 as selected are activated when the edge specified by the Trigger Input Edge Selection bits (PCN.EGS[1:0]) are detected during the specified activation trigger.
- For detailed setting of each channel see chapter 7.7 What activation triggers are available and how are they selected? (Page No.817)

4.5 GCN2: General Control Register 2

Generates PPG0-PPG3, PPG4-PPG7, PPG8-PPG11 and PPG12-PPG15 internal trigger levels using software.

- **GCN20 (PPG0-PPG3): Address 0103h (Access: Byte)**
- **GCN21 (PPG4-PPG7): Address 0107h (Access: Byte)**
- **GCN22 (PPG8-PPG11): Address 010Bh (Access: Byte)**
- **GCN23 (PPG12-PPG15): Address 0323h (Access: Byte)**

7	6	5	4	3	2	1	0	bit
–	–	–	–	EN3	EN2	EN1	EN0	
0	0	0	0	0	0	0	0	Initial value
R/W0	R/W0	R/W0	R/W0	R/W	R/W	R/W	R/W	Attribute

(See “Meaning of Bit Attribute Symbols (Page No.10)” for details of the attributes.)

- Bits 7-4: Undefined.Always write “0”. The read value is the value as written. (See “8. Caution (Page No.823)”.)
- Bit 3: EN3 trigger input
- Bit 2: EN2 trigger input
- Bit 1: EN1 trigger input
- Bit 0: EN0 trigger input

EN0, EN1, EN2, and EN3	Internal Triggers EN0, EN1, EN2, and EN3
0	Set the level to “L”.
1	Set the level to “H”.

- Set the levels of internal triggers EN0, EN1, EN2, and EN3.
- If any of the EN trigger inputs (EN0, EN1, EN2, EN3) is selected with the trigger specification bits (GCN10.TSEL0[3:0], GCN10.TSEL1[3:0], GCN10.TSEL2[3:0], and GCN10.TSEL3[3:0]) of PPG0, PPG1, PPG2, PPG3, then the selected EN serves as a PPG trigger input bit.
- If any of the EN trigger inputs (EN0, EN1, EN2, EN3) is selected with the trigger specification bits (GCN11.TSEL0[3:0], GCN11.TSEL1[3:0], GCN11.TSEL2[3:0], and GCN11.TSEL3[3:0]) of PPG4, PPG5, PPG6, PPG7, then the selected EN serves as a PPG trigger input bit.
- If any of the EN trigger inputs (EN0, EN1, EN2, EN3) is selected with the trigger specification bits (GCN12.TSEL0[3:0], GCN12.TSEL1[3:0], GCN12.TSEL2[3:0], and GCN12.TSEL3[3:0]) of PPG8, PPG9, PPG10, PPG11, then the selected EN serves as a PPG trigger input bit.
- If any of the EN trigger inputs (EN0, EN1, EN2, EN3) is selected with the trigger specification bits (GCN13.TSEL0[3:0], GCN13.TSEL1[3:0], GCN13.TSEL2[3:0], and GCN13.TSEL3[3:0]) of PPG12, PPG13, PPG14, of PPG15, then the selected EN serves as a PPG trigger input bit.
- If the state selected with the trigger input edge selection bit (EGS[1:0]) is generated by software using the trigger input bit (selected EN0, EN1, EN2, or EN3), the choice serves as an activation trigger to activate the PPG.

4.6 PTMR: PPG Timer Register

Reads the counts of PPG0-PPG3, PPG4-PPG7, PPG8-PPG11 and PPG12-PPG15.

- **PTMR00 (PPG0): Address 0110h (Access: Half-word)**
- **PTMR01 (PPG1): Address 0118h (Access: Half-word)**
- **PTMR02 (PPG2): Address 0120h (Access: Half-word)**
- **PTMR03 (PPG3): Address 0128h (Access: Half-word)**
- **PTMR04 (PPG4): Address 0130h (Access: Half-word)**
- **PTMR05 (PPG5): Address 0138h (Access: Half-word)**
- **PTMR06 (PPG6): Address 0140h (Access: Half-word)**
- **PTMR07 (PPG7): Address 0148h (Access: Half-word)**
- **PTMR08 (PPG8): Address 0150h (Access: Half-word)**
- **PTMR09 (PPG9): Address 0158h (Access: Half-word)**
- **PTMR10 (PPG10): Address 0160h (Access: Half-word)**
- **PTMR11 (PPG11): Address 0168h (Access: Half-word)**
- **PTMR12 (PPG12): Address 0330h (Access: Half-word)**
- **PTMR13 (PPG13): Address 0338h (Access: Half-word)**
- **PTMR14 (PPG14): Address 0340h (Access: Half-word)**
- **PTMR15 (PPG15): Address 0348h (Access: Half-word)**

15	14	13	12	11	10	9	8	Bit
D15	D14	D13	D12	D11	D10	D9	D8	
1	1	1	1	1	1	1	1	Initial value
R/WX	R/WX	R/WX	R/WX	R/WX	R/WX	R/WX	R/WX	Attribute

7	6	5	4	3	2	1	0	Bit
D7	D6	D5	D4	D3	D2	D1	D0	
1	1	1	1	1	1	1	1	Initial value
R/WX	R/WX	R/WX	R/WX	R/WX	R/WX	R/WX	R/WX	Attribute

(See “Meaning of Bit Attribute Symbols (Page No.10)” for details of the attributes.)

- The count of the 16-bit down counter can be read.
- Be sure to access the PPG Timer register PTMR in half words (16 bits).
- The register would not be read correctly if it is byte-accessed.

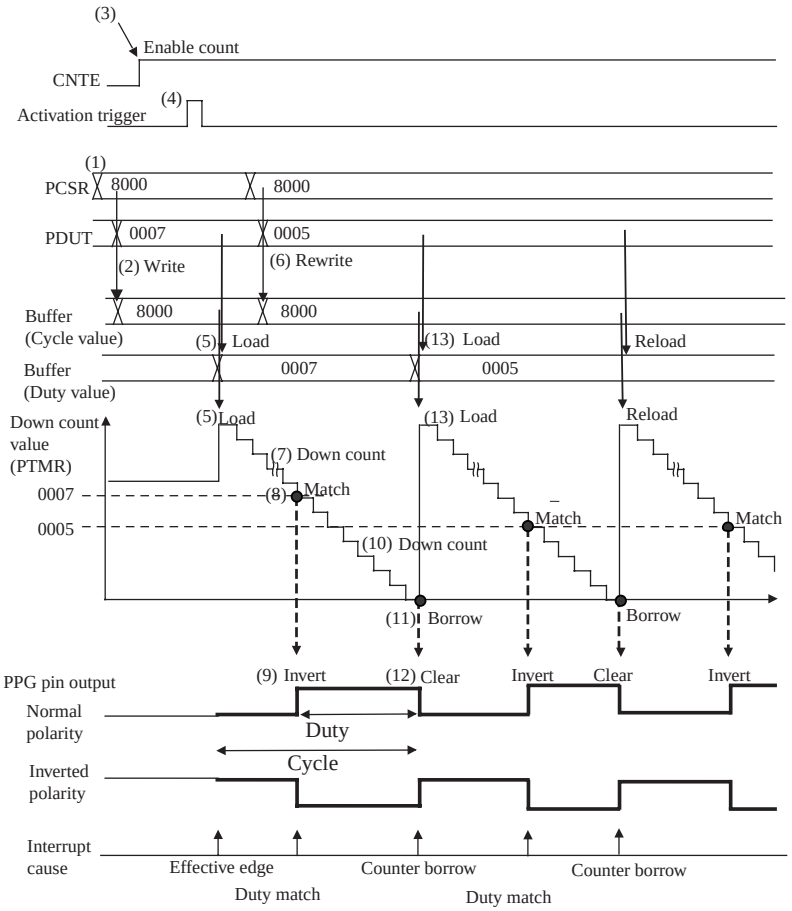
5. Operation

The MB91460 series features a maximum of 16 programmable pulse generators (PPGs), which provide programmable pulse output independently or jointly.

The individual modes of operation are described below.

5.1 PWM Operation

In PWM operation, variable-duty pulses are generated from the PPG pin.



(1) Write a cycle value.
(2) Write a duty value and transfer the cycle value to buffers.
(3) Enable PPG operation.
(4) Generate an activation trigger.
(5) Load the cycle and duty values.
(6) Rewrite the duty value and transfer the cycle value to buffers.
(6) Counter down count
(7) The down counter equals the duty value.
(8) Inverses the PPG pin output level.
(9) Counter down count
(10) Counter borrow
(11) Clear the PPG pin output level (return to normal).
(12) Reload the cycle value.
(13) Reload the duty value.
(14) Steps from (6) to (13) are iterated.

(See “8. Caution (Page No.823)”.)

- Equation

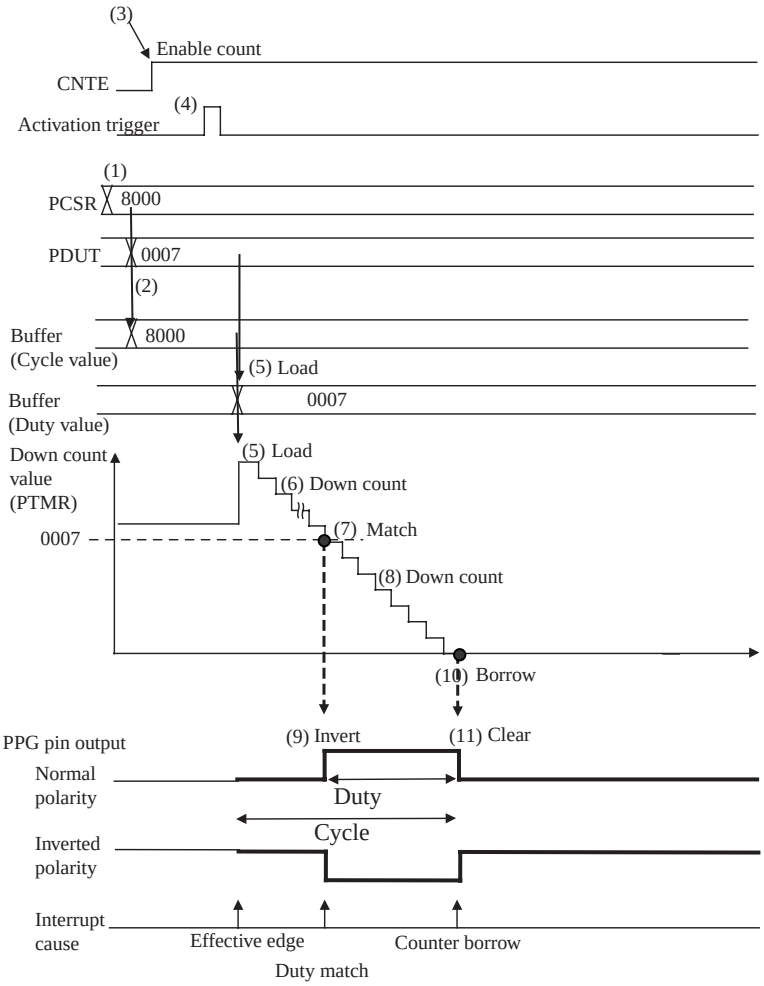
  Period = {Period value (PCSR) + 1} x Count clock

  Duty = {Duty value (PDUT) + 1} x Count clock

  Width up to pulse output = {Period value (PCSR) – Duty value (PDUT)} x Count clock

5.2 One-Shot Operation

In one-shot operation, one-shot pulses are generated from the PPG pin.



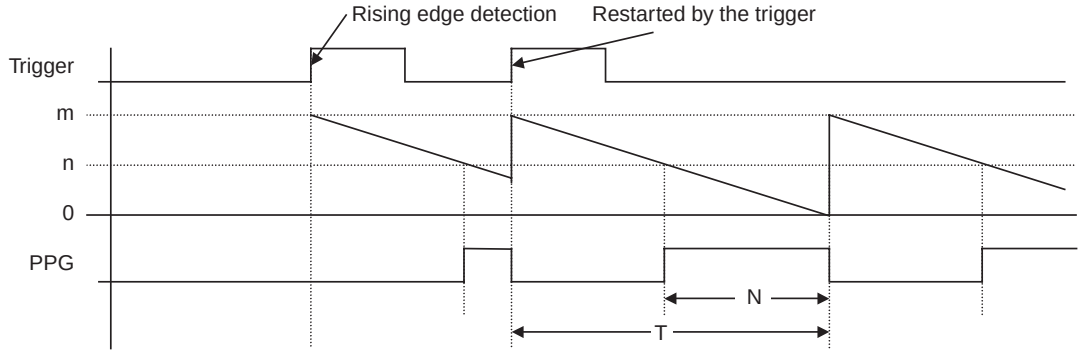
(1) Write a cycle value.
(2) Write a duty value and transfer the cycle value to buffers.
(3) Enable PPG operation.
(4) Generate an activation trigger.
(5) Load the cycle and duty values.
(6) Counter down count
(7) Down counter value and duty value
(8) Inverse the PPG pin output level.
(9) Counter down count
(10) Counter borrow
(11) Clear the PPG pin output level (return to normal).
(12) The operating sequence is now completed.

(See “8. Caution (Page No.823)”.)

5.3 Restart Operation

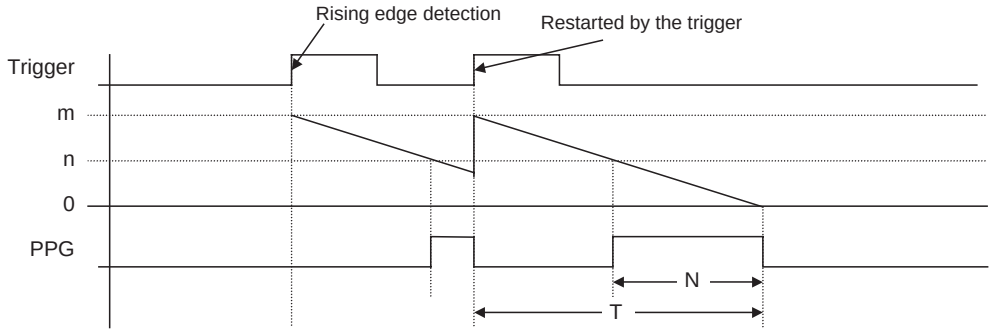
The restart operation is described below.

- Restart available in PWM operation:



N = duty, T = cycle

- Restart available in one-shot operation:



If a restart is not available, the second and subsequent triggers have no effect in both PWM and one-shot operations.

(The second and subsequent triggers following a shutdown of the down counter are functional.)

6. Setting

Table 6-1 Settings Needed to Start the PPG

Setting	Setting Registers	Setting Procedure*
Period and duty value settings	PPG cycle settings (PCSR00-PCSR15) PPG duty settings (PDUT00-PDUT15)	7.1
Enable PPG operation.	PPG control status (PCN00-PCN15)	7.2
Operation mode selection (PWM/one-shot)		7.3
Enable restart.		7.4
Count clock selection		7.5
PPG output mask selection		7.6
Trigger selection Software Internal trigger External trigger	General Control 1 (GCN10, GCN11, GCN12, GCN13)	7.7
Output polarity specification		7.8
PPG pin output setting	Port functions (PFR16, PFR17)	7.9
Trigger generation (software trigger) (Reload timer) (GCN2.EN bit)	PPG Control Status (PCN00-PCN15)	7.10
	See "Chapter 38 Reload Timer (Page No.777)".	
	General Control 2 (GCN20, GCN21, GCN22, GCN23)	

* For refer to the section indicated by the number.

Table 6-2 Settings Needed to Stop the PPG

Setting	Setting Registers	Setting Procedure*
PPG stop bit setting	PPG control status (PCN00-PCN15)	7.11

*For the setting procedure, refer to the section indicated by the number.

Table 6-3 Settings Needed to Clamp the Output Level

Setting	Setting Registers	Setting Procedure*
Output polarity specification	PPG control status (PCN00-PCN15)	7.8
PPG output mask selection		7.6
Period value = Duty value setting	PPG duty settings (PDUT00-PDUT15)	7.6

*For the setting procedure, refer to the section indicated by the number.

Table 6-4 Settings Needed to Implement PPG Interrupts

Setting	Setting Registers	Setting Procedure*
PPG interrupt vector, PPG interrupt level setting	See "Chapter 24 Interrupt Control (Page No.315)".	7.12

Table 6-4 Settings Needed to Implement PPG Interrupts

PPG interrupt cause selection (Generate an activation trigger, borrow, and duty match)	PPG control status (PCN00-PCN15)	7.13
PPG interrupt setting Clear interrupt requests. Enable interrupt requests.		7.14

*:For the setting procedure, refer to the section indicated by the number.

7. Q & A

7.1 How do I set (rewrite) a cycle and a duty?

Period and duty value settings

- Set each cycle value in PPG Period Setting Register PCSR.
- Set each duty value in PPG Duty Setting Register PDUT.
- The PPG Period Setting and the PPG Duty Setting registers each have a buffer to allow the user to ignore the write timing.
  - Equation

    PCSR register value = {Cycle/Count clock} –1

    PDUT register value = {“H” width (duty)*/Count clock} –1

*: Normal polarity (OSEL= 0)

  - Allowed range

    PCSR register value = PCSR register value - FFFFh (65535)

    PDUT register value = 0 - PCSR register value

Note: Be sure to set a cycle following the setting of a cycle. (See “8. Caution (Page No.823)”.)

7.2 How do I enable or disable PPG operations?

Enabling the PPG operation

Use the PPG operation enable bit (PCN.CNTE).

Control	PPG Operation Enable Bit (CNTE)
To stop a PPG operation	Set “0”.
To enable a PPG operation	Set “1”.

Enable PPG operation before starting the PPG.

(See “8. Caution (Page No.823)”.)

7.3 How do I set the PPG operation mode (PWM operation/one-shot operation)?

Operation mode selection

Use the mode selection bit (PCN.MDSE).

Operation Mode	Mode Selection Bit (MDSE)
To implement a PWM operation	Set “0”.
To implement a one-shot operation	Set “1”.

(See “8. Caution (Page No.823)”.)

7.4 How do I get it restarted?

Enable restart.

A restart of a PPG can be enabled while the PPG is in operation.

Use the Enable Restart bit (PCN.RTRG) to set.

(See “8. Caution (Page No.823)”.)

7.5 What count clocks are available and how are they selected?

Count clock selection

The count clock is selectable out of the four choices listed below.

Use the count clock selection bit (PCN.CKS[1:0]).

Count Clock	Count Clock Selection Bit		(Example) CLKP = 32 MHz	
	CKS1	CKS0	Count Clock	Period (1 - FFFFh)
CLKP	0	0	32MHz	62.5ns - 2.048µs
CLKP/4	0	1	8MHz	250ns - 8.192µs
CLKP/16	1	0	2MHz	1µs - 32.76ms
CLKP/64	1	1	500kHz	4µs - 131.0ms

(See “8. Caution (Page No.823)”.)

7.6 How do I clamp the PPG pin output level?

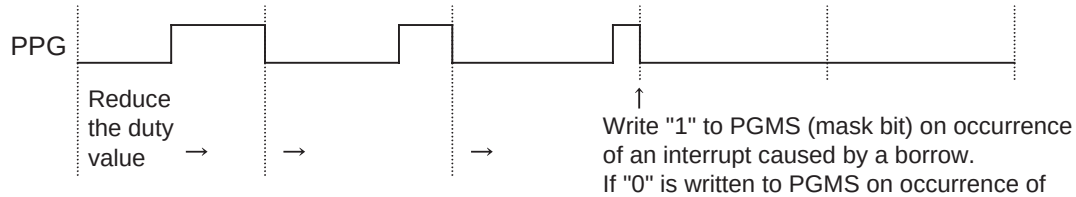
PPG output mask selection

The level of PPG pin output can be clamped.

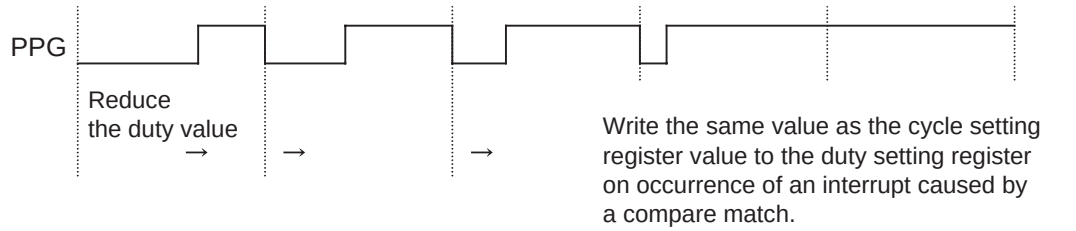
Use the PPG Output Mask Selection bit (PCN.PGMS) and the duty value (PDUT) to set.

PPG Pin Output	PPG Output Polarity Specification Bit (OSEL)	Setting Procedure
To clamp the “L” level under normal polarity	When “0”	Set the PPG Output Mask Selection bit (PGMS) to “1”.
To clamp the “H” level under normal polarity	When “0”	Period value (PCSR) = Set a duty value (PDUT).
To clamp the “H” level under inverted polarity	When “1”	Set the PPG Output Mask Selection bit (PGMS) to “1”.
To clamp the “L” level under inverted polarity	When “1”	Period value (PCSR) = Set a duty value (PDUT).

PPG pin output can be set to all “L”. (when OSEL=“0”)



PPG pin output can be set to all “H”. (when OSEL=“0”)



PPG output will also equal all “H” if “0” is set in both the PPG Period Setting Register (PCSR) and PPG Duty Setting Register (PDUT). (when OSEL=“0”)

7.7 What activation triggers are available and how are they selected?

- Trigger selection
  - Activation triggers are broadly grouped into software triggers, internal triggers and external triggers.
  - Software triggers work at all times.
  - Internal and external trigger availability depends on each device specification.

An trigger is set using the trigger specification bits (GCN1.TSEL0[3:0]), (GCN1.TSEL1[3:0]), (GCN1.TSEL2[3:0]), and (GCN1.TSEL3[3:0]).

Triggers are selectable for PPG0, PPG1, PPG2, and PPG3 independently.

Internal Trigger	PPG0	PPG1	PPG2	PPG3
	Internal Trigger Specification Bit			
	TSEL0[3:0]	TSEL1[3:0]	TSEL2[3:0]	TSEL3[3:0]
To select the EN0 bit of the GCN20 register	Set “0000”	Set “0000”	Set “0000”	Set “0000”
To select the EN1 bit of the GCN20 register	Set “0001”	Set “0001”	Set “0001”	Set “0001”
To select the EN2 bit of the GCN20 register	Set “0010”	Set “0010”	Set “0010”	Set “0010”
To select the EN3 bit of the GCN20 register	Set “0011”	Set “0011”	Set “0011”	Set “0011”
To select reload timer 0	Set “0100”	Set “0100”	Set “0100”	Set “0100”
To select reload timer 1	Set “0101”	Set “0101”	Set “0101”	Set “0101”

External Trigger	PPG0	PPG1	PPG2	PPG3
	External Trigger Specification Bit			
	TSEL0[3:0]	TSEL1[3:0]	TSEL2[3:0]	TSEL3[3:0]
To select the external trigger on GP14_0	Set “1000”	Set “1000”	Set “1000”	Set “1000”
To select the external trigger on GP14_1	Set “1001”	Set “1001”	Set “1001”	Set “1001”
To select the external trigger on GP14_2	Set “1010”	Set “1010”	Set “1010”	Set “1010”
To select the external trigger on GP14_3	Set “1011”	Set “1011”	Set “1011”	Set “1011”

Triggers are selectable for PPG4, PPG5, PPG6, and PPG7 independently.

Internal Trigger	PPG4	PPG5	PPG6	PPG7
	Internal Trigger Specification Bit			
	TSEL0[3:0]	TSEL1[3:0]	TSEL2[3:0]	TSEL3[3:0]
To select the EN0 bit of the GCN21 register	Set “0000”	Set “0000”	Set “0000”	Set “0000”
To select the EN1 bit of the GCN21 register	Set “0001”	Set “0001”	Set “0001”	Set “0001”
To select the EN2 bit of the GCN21 register	Set “0010”	Set “0010”	Set “0010”	Set “0010”
To select the EN3 bit of the GCN21 register	Set “0011”	Set “0011”	Set “0011”	Set “0011”
To select reload timer 2	Set “0100”	Set “0100”	Set “0100”	Set “0100”
To select reload timer 3	Set “0101”	Set “0101”	Set “0101”	Set “0101”

External Trigger	PPG4	PPG5	PPG6	PPG7
	External Trigger Specification Bit			
	TSEL0[3:0]	TSEL1[3:0]	TSEL2[3:0]	TSEL3[3:0]
To select the external trigger on GP14_4	Set “1000”	Set “1000”	Set “1000”	Set “1000”
To select the external trigger on GP14_5	Set “1001”	Set “1001”	Set “1001”	Set “1001”
To select the external trigger on GP14_6	Set “1010”	Set “1010”	Set “1010”	Set “1010”
To select the external trigger on GP14_7	Set “1011”	Set “1011”	Set “1011”	Set “1011”

Triggers are selectable for PPG8, PPG9, PPG10, and PPG11 independently.

Internal Trigger	PPG8	PPG9	PPG10	PPG11
	Internal Trigger Specification Bit			
	TSEL0[3:0]	TSEL1[3:0]	TSEL2[3:0]	TSEL3[3:0]
To select the EN0 bit of the GCN22 register	Set “0000”	Set “0000”	Set “0000”	Set “0000”
To select the EN1 bit of the GCN22 register	Set “0001”	Set “0001”	Set “0001”	Set “0001”
To select the EN2 bit of the GCN22 register	Set “0010”	Set “0010”	Set “0010”	Set “0010”
To select the EN3 bit of the GCN22 register	Set “0011”	Set “0011”	Set “0011”	Set “0011”
To select reload timer 4	Set “0100”	Set “0100”	Set “0100”	Set “0100”
To select reload timer 5	Set “0101”	Set “0101”	Set “0101”	Set “0101”

External Trigger	PPG9	PPG9	PPG10	PPG11
	External Trigger Specification Bit			
	TSEL0[3:0]	TSEL1[3:0]	TSEL2[3:0]	TSEL3[3:0]
To select the external trigger on GP14_0	Set “1000”	Set “1000”	Set “1000”	Set “1000”
To select the external trigger on GP14_1	Set “1001”	Set “1001”	Set “1001”	Set “1001”
To select the external trigger on GP14_2	Set “1010”	Set “1010”	Set “1010”	Set “1010”
To select the external trigger on GP14_3	Set “1011”	Set “1011”	Set “1011”	Set “1011”

Triggers are selectable for PPG12, PPG13, PPG14, and PPG15 independently.

Internal Trigger	PPG12	PPG13	PPG14	PPG15
	Internal Trigger Specification Bit			
	TSEL0[3:0]	TSEL1[3:0]	TSEL2[3:0]	TSEL3[3:0]
To select the EN0 bit of the GCN23 register	Set “0000”	Set “0000”	Set “0000”	Set “0000”
To select the EN1 bit of the GCN23 register	Set “0001”	Set “0001”	Set “0001”	Set “0001”
To select the EN2 bit of the GCN23 register	Set “0010”	Set “0010”	Set “0010”	Set “0010”
To select the EN3 bit of the GCN23 register	Set “0011”	Set “0011”	Set “0011”	Set “0011”
To select reload timer 6	Set “0100”	Set “0100”	Set “0100”	Set “0100”
To select reload timer 7	Set “0101”	Set “0101”	Set “0101”	Set “0101”

External Trigger	PPG12	PPG13	PPG14	PPG15
	External Trigger Specification Bit			
	TSEL0[3:0]	TSEL1[3:0]	TSEL2[3:0]	TSEL3[3:0]
To select the external trigger on GP14_4	Set “1000”	Set “1000”	Set “1000”	Set “1000”
To select the external trigger on GP14_5	Set “1001”	Set “1001”	Set “1001”	Set “1001”
To select the external trigger on GP14_6	Set “1010”	Set “1010”	Set “1010”	Set “1010”
To select the external trigger on GP14_7	Set “1011”	Set “1011”	Set “1011”	Set “1011”

The same trigger can be specified for a group of PPGs to activate all these PPGs simultaneously. (See “8. Caution (Page No.823)”.)

- Trigger edge selection

Trigger edges are set using trigger input edge selection bits (PCN.EG[1:0]).

Trigger Edge Selection	Trigger Input Edge Selection Bits (EG1-EG0)
When not detected (software trigger only)	Set “00”.
“L” -> “H” Trigger generated on the rising edge	Set “01”.
“H” -> “L” Trigger generated on the falling edge	Set “10”.
Trigger generated on both edges	Set “11”.

(See “8. Caution (Page No.823)”.)

7.8 How do I invert the output polarity?

Output polarity specification

The polarity in the normal state can be specified as follows:

Use the PPG Output Polarity Specification bit (PCN.OSEL) to set.

(“Normal state” means the state in which pulse output is not executed.)

Output Level in Normal State	PPG Output Polarity Specification Bit (OSEL)
To enable “L” level output (normal polarity)	Set “0”.
To enable “H” level output (inverted polarity)	Set “1”.

(See “8. Caution (Page No.823)”.)

7.9 How do I program a pin as a PPG output pin?

-> PPG pin output setting

Software programming allows ports to be switched to PPG pin output.

Pin	Control Bit Location	
PPG0 pin	Port Function register PFR17.0 = ‘1’	PPG0 Output specification bit (PPG0)
PPG1 pin	Port Function register PFR17.1 = ‘1’	PPG1 Output specification bit (PPG1)
PPG2 pin	Port Function register PFR17.2 = ‘1’	PPG2 Output specification bit (PPG2)
PPG3 pin	Port Function register PFR17.3 = ‘1’	PPG3 Output specification bit (PPG3)
PPG4 pin	Port Function register PFR17.4 = ‘1’	PPG4 Output specification bit (PPG4)
PPG5 pin	Port Function register PFR17.5 = ‘1’	PPG5 Output specification bit (PPG5)
PPG6 pin	Port Function register PFR17.6 = ‘1’	PPG6 Output specification bit (PPG6)
PPG7 pin	Port Function register PFR17.7 = ‘1’	PPG7 Output specification bit (PPG7)
PPG8 pin	Port Function register PFR16.0 = ‘1’	PPG8 Output specification bit (PPG8)
PPG9 pin	Port Function register PFR16.1 = ‘1’	PPG9 Output specification bit (PPG9)
PPG10 pin	Port Function register PFR16.2 = ‘1’	PPG10 Output specification bit (PPG10)
PPG11 pin	Port Function register PFR16.3 = ‘1’	PPG11 Output specification bit (PPG11)
PPG12 pin	Port Function register PFR16.4 = ‘1’	PPG12 Output specification bit (PPG12)
PPG13 pin	Port Function register PFR16.5 = ‘1’	PPG13 Output specification bit (PPG13)
PPG14 pin	Port Function register PFR16.6 = ‘1’	PPG14 Output specification bit (PPG14)
PPG15 pin	Port Function register PFR16.7 = ‘1’	PPG15 Output specification bit (PPG15)

7.10 How do I generate an activation trigger?

Generating a trigger

Methods of generating an activation trigger are described below.

- Activating a software trigger

  Use the Software Trigger bit (PCN.STGR) to set.

  Write “1” to the Software Trigger bit (STGR) to generate an activation trigger.

  Always functional, regardless of the internal trigger.

- Activating PPGs with reload timers

  The reload timers need to be set up and activated. For more information, see “Chapter 38 Reload Timer (Page No.777)”.

  An activation trigger is generated when the edge specified by the reload timer output signal is generated with the reload timer underflow.

- Activating PPGs with external triggers

  An activation trigger is generated when the edge specified on the specified pin appears. There is no need for configuration of the port registers; the trigger is always connected to the PPGs.

- Activating a PPG with the EN trigger input bits (GCN2.EN0) - (GCN2.EN3)

  An activation trigger can be generated by rewriting the level of the EN trigger input bits (GCN2.EN0) - (GCN2.EN3).

Edge	Software-Based Setting Procedure (EN0, EN1, EN2, EN3)
Rising edge	First, set the EN bit to “0”, then the EN bit to “1”.
Falling edge	First, set the EN bit to “1”, then to “0”.

- Activating multiple PPGs concurrently

  The same trigger (trigger input bit) can be specified with the PPG trigger specification bits to activate all the PPGs simultaneously when the trigger is generated.

- Even if an activation trigger is generated before the operation of a PPG is enabled, that PPG would not be activated. Be sure to enable the operation of a PPG before generating a trigger to activate it. (See “7.2 How do I enable or disable PPG operations? (Page No.815)”.)

7.11 How do I stop a PPG operation?

PPG stop bit setting (See “7.2 How do I enable or disable PPG operations? (Page No.815)”.)

7.12 What interrupt registers are used?

PPG interrupt vector, PPG interrupt level setting

The table below summarizes the relationships among the PPG number, interrupt level and interrupt vector.

For more information about the interrupt levels and interrupt vectors, see “Chapter 24 Interrupt Control (Page No.315)”.

	Interrupt Vector (Default)	Interrupt Level Setting Bit (ICR[4:0])
PPG0	#112 Address: 0FFE3Ch	Interrupt Level register (ICR48) Address: 0470h
PPG1	#113 Address: 0FFE38h	
PPG2	#114 Address: 0FFE34h	Interrupt Level register (ICR49) Address: 0471h
PPG3	#115 Address: 0FFE30h	
PPG4	#116 Address: 0FFE2Ch	Interrupt Level register (ICR50) Address: 0472h
PPG5	#117 Address: 0FFE28h	

PPG6	#118 Address: 0FFE24h	Interrupt Level register (ICR51) Address: 0473h
PPG7	#119 Address: 0FFE20h	
PPG8	#120 Address: 0FFE1Ch	Interrupt Level register (ICR52) Address: 0474h
PPG9	#121 Address: 0FFE18h	
PPG10	#122 Address: 0FFE14h	Interrupt Level register (ICR53) Address: 0475h
PPG11	#123 Address: 0FFE10h	
PPG12	#124 Address: 0FFE0Ch	Interrupt Level register (ICR54) Address: 0476h
PPG13	#125 Address: 0FFE08h	
PPG14	#126 Address: 0FFE04h	Interrupt Level register (ICR55) Address: 0477h
PPG15	#127 Address: 0FFE00h	

The Interrupt Request flag (PCN.IRQF) does not clear itself automatically. Use software to clear it before returning from the interrupt handler. (Write “0” to the IRQF bit.)

7.13 What interrupts are available and how are they selected?

Interrupt cause selection

Four kinds of interrupts are selectable as follows:

Use the Interrupt Cause Setting bit (PCN.IRS[1:0]) to set.

Interrupt Cause	Interrupt Cause Setting Bit (IRS[1:0])
Software trigger or Internal trigger generation (PPG0-PPG15)	Set “00”.
Down counter borrow (cycle match)	Set “01”.
Duty match	Set “10”.
Down counter borrow (cycle match) or Duty match	Set “11”.

7.14 How do I enable, disable and clear interrupts?

Interrupt Request Enable flag, Interrupt Request flag

Use the interrupt request enable bit (PCN.IREN) to enable interrupts.

	Interrupt Request Enable Bit (IREN)
To disable interrupt requests	Set “0”.
To enable interrupt requests	Set “1”.

Use the interrupt request bit (PCN.IRQF) to clear interrupt requests.

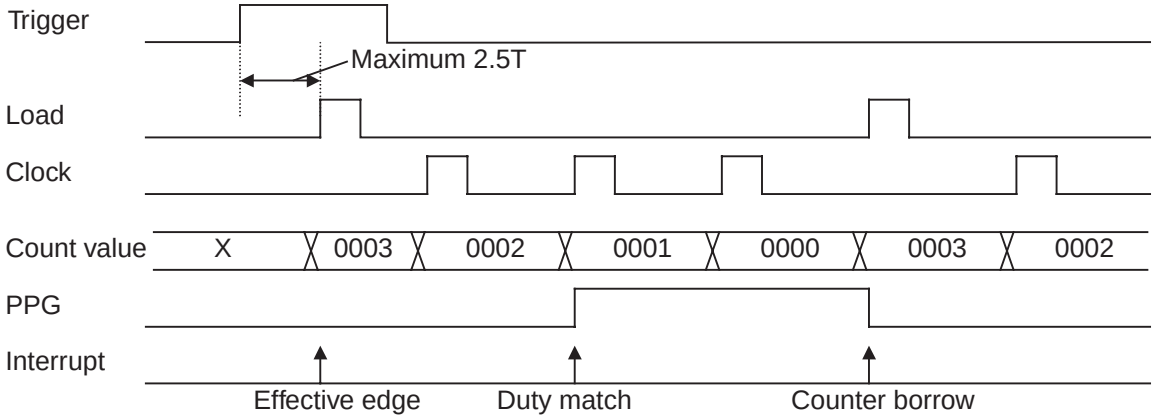
	Interrupt Request Bit (IRQF)
To select interrupt request	Write “0”.

(See “8. Caution (Page No.823)”.)

8. Caution

- If the Interrupt Request flag (PCN.IRQF) equals “1” and the Interrupt Request flag is set to “0” at the same time, the setting of the Interrupt Request flag to “1” overrides the flag clear request.

- The first load comes with a maximum delay of 2.5T after the activation trigger. (T: Count clock)
  If the down counter is loaded and counts at the same time, the load operation overrides.



- Be sure to write duty value PDUT after cycle PCSR has been initialized and rewritten.
  (Always write in the order of (1)PCSR and (2)PDUT.)
  Only the PDUT can be written for rewriting the duty.

- Set the duty value PDUT smaller than the cycle value PCSR. If any larger value has been set, disable the operation of the PPG before replacing the duty with a smaller value.

- Always access PPG Period Setting registers PCSR and PPG Duty Setting registers in a half-word (16-bit) format. If these registers are byte-accessed, no values would be written to their upper and lower bit positions.

- To activate a PPG, it is necessary to set the Timer Operation Enable bits (PCN.CNTE) to “1” before or concurrently with the activation to enable the PPG operation.

- The values of mode (MDSE), restart enable (RTRG), count clock (CKS[1:0]), trigger input edge (EGS[1:0]), interrupt cause (IRS), internal trigger (TSEL) and output polarity specification (OSEL) may not be changed while the PPG is operating.
  If any of these values has been changed while the PPG was operating, disable the operation of the PPG before reloading the register.

- Whenever writing a value to GCN2, be sure to write “0” to any undefined part of the upper 4 bits.
  If “1” is written, disable the operation of the PPG before reloading the register.

- If any value outside the specified range (0110, 0111, 1100 - 1111) is set in Activation Trigger Specification bits (TSEL0[3:0]), (TSEL1[3:0]), (TSEL2[3:0]), (TSEL3[3:0]) has been set, disable the operation of the PPG and then write the specified value to let the register return to normal.

- If the Timer Operation Enable bit (PCN.CNTE) is set to “0” to disable PPGn while it is operating, the PPG stops, with its status (count and output level) being latched.
  If the Timer Operation Enable bit is subsequently set to (PCN.CNTE) “1” to enable the PPG, it restarts from the point of interruption.

8.Caution

Chapter 40 Pulse Frequency Modulator

This chapter provides an overview of the 16-bit pulse frequency modulator, describes the register structure/functions, and describes the operation of the 16-bit pulse frequency modulator.

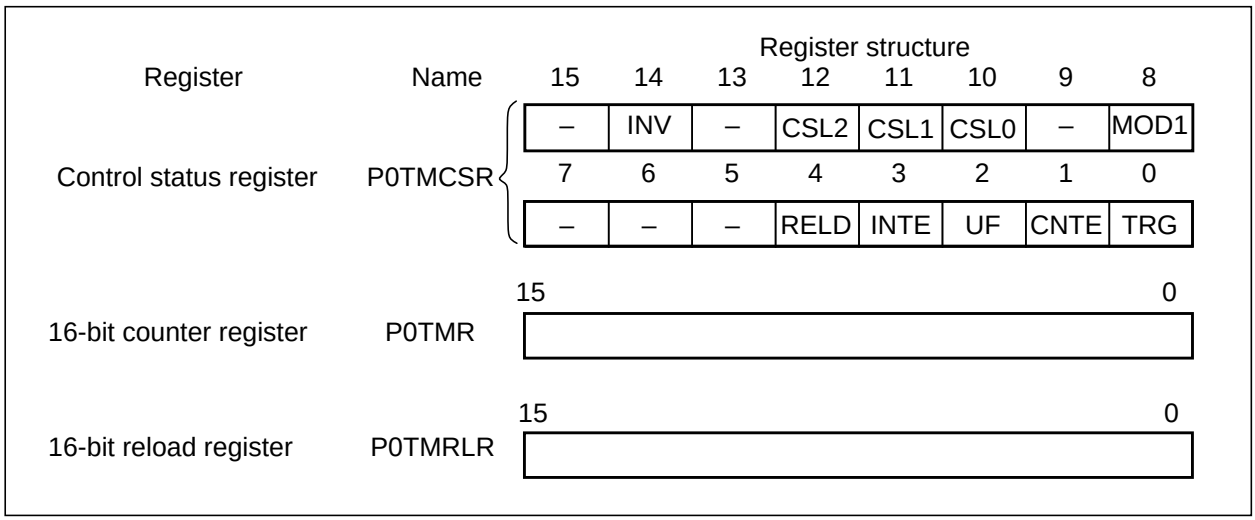
1. PFM Overview

The 16-bit pulse frequency modulator consists of two 16-bit down-counters, two 16-bit reload registers, prescalers for generating the internal count clocks and control registers.

■ Functions

- Two independent programmable 16-bit down-counters generating low and high pulses.
- The input clock can be selected from prescaled internal clocks (the peripheral clock (CLKP) divided by 2/8/32/64/128) separately for each counter.
- The mark level and output waveform can be inverted.

■ 16-bit Reload Counter 0 Register Configuration



■ 16-bit Reload Counter 1 Register Configuration

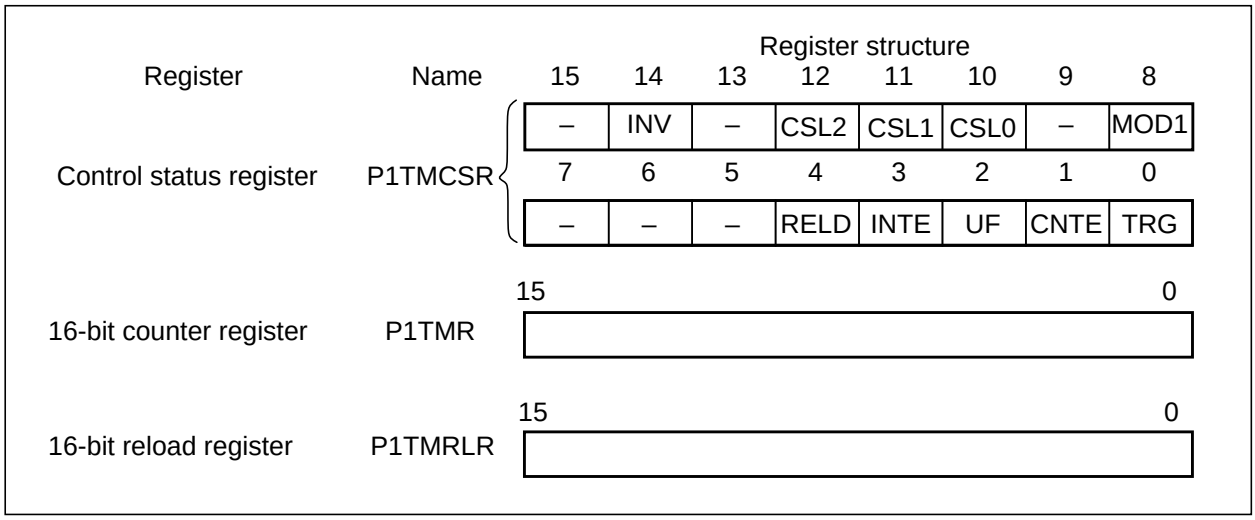


Figure 1-1 16-bit Reload Counter Register Configuration

■ Block Diagram of the 16-Bit Pulse Frequency Modulator

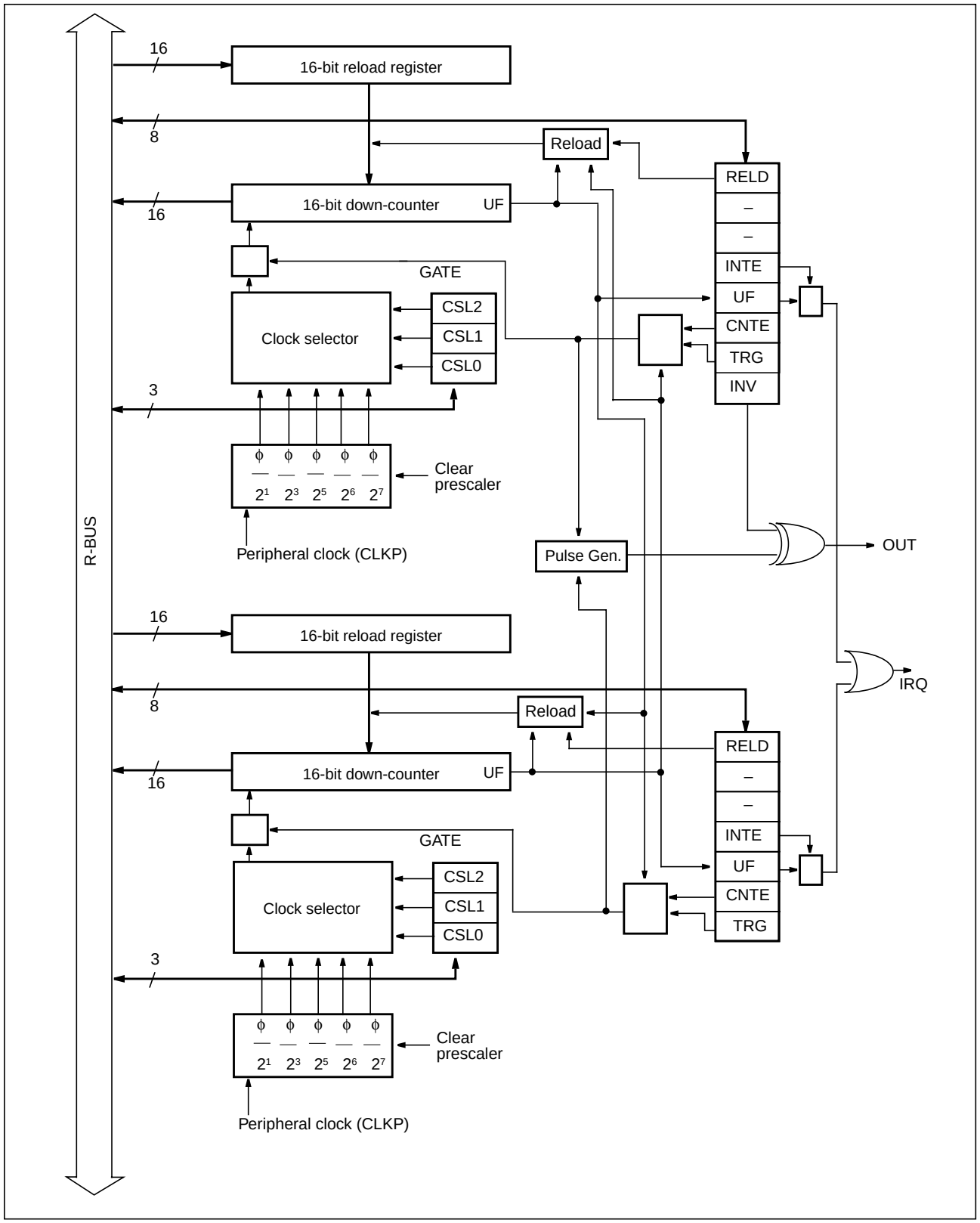


Figure 1-2 Block Diagram of the 16-bit Pulse Frequency Modulator

2. Reload Counter Registers

This section describes the 16-bit pulse frequency modulator registers listed below.

Control status register (P0TMCSR, P1TMCSR)

16-bit counter register (P0TMR, P1TMR)

16-bit reload register (P0TMRLR, P1TMRLR)

■ Control Status Register (P0TMCSR, P1TMCSR)

Controls the operation mode and interrupts for the 16-bit reload counter.
Only change the value of bits other than UF and TRG when CNTE = "0".
The bits can be written simultaneously.

● P0TMCSR, P1TMCSR structure

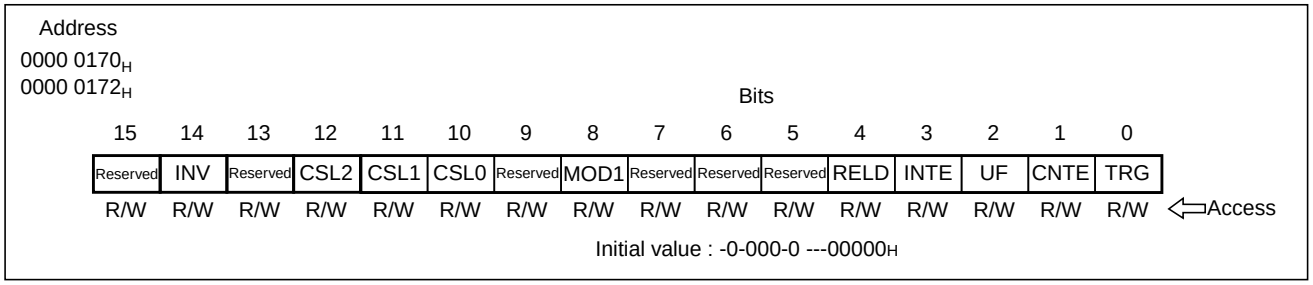


Figure 2-1 Structure of the Control Status Register

● Functions of the P0TMCSR, P1TMCSR bits

[Bit 15] Reserved

Always set to "0".

[Bit 14] INV (INVersion)

The output signal inversion bit.

"0" is default level (counter 0 high level, counter 1 low level).

"1" inverts the output signal (counter 0 low level, counter 1 high level).

Remark: Writing INV of P0TMCSR or INV of P1TMCSR or both has the same effect.

[Bit 13] Reserved

Always set to "0".

[Bits 12, 10] CSL2, CSL1, CSL0 (Count clock SeLect)

The count clock select bits.

Table 2.1 lists the clock source selections.

Table 2-1 CSL Bit Clock Source Settings

CSL2	CSL1	CSL0	Clock source (ϕ : Peripheral clock (CLKP))
0	0	0	$\phi / 2^1$
0	0	1	$\phi / 2^3$
0	1	0	$\phi / 2^5$
0	1	1	External clock
1	0	0	Clock disabled
1	0	1	$\phi / 2^6$
1	1	0	$\phi / 2^7$
1	1	1	Clock disabled

[Bits 9] Reserved

Always set to "0".

[Bits 8] MOD1

Sets the Trigger level to Falling edge (MOD1='1' is necessary for PFM operation)

[Bits 7 to 5] Reserved

Always set to "010".

[Bit 4] RELD

This bit enables reload operations.

When RELD is "1", the counter operates in reload mode. In this mode, the counter loads the reload register contents into the counter and continues counting whenever an underflow occurs (when the counter value changes from 0000_H to $FFFF_H$).

When RELD is "0", the count operation stops when an underflow occurs due to the counter value changing from 0000_H to $FFFF_H$.

[Bit 3] INTE

The interrupt request enable bit.

When INTE is "1", an interrupt request is generated when the UF bit changes to "1".

When INTE is "0", no interrupt requests are generated.

[Bit 2] UF

The counter interrupt request flag.

UF is set to "1" when an underflow occurs (when the counter value changes from 0000_H to $FFFF_H$).

Writing "0" clears the bit. Writing "1" has no meaning. Read as "1" by read-modify-write instructions.

[Bit 1] CNTE

The counter count enable bit.

Writing "1" sets the counter to wait for a trigger.

Writing "0" stops count operation.

[Bit 0] TRG

Software trigger bit.

Writing "1" to TRG applies a software trigger, causing the counter to load the reload register contents to the counter and start counting.

Writing "0" has no meaning. Reading always returns "0".

Applying a trigger using this register is only valid when CNTE = "1". Writing "1" has no effect if CNTE = "0".

■ 16-bit Counter Register (P0TMR, P1TMR)

Reading this register reads the count value of the 16-bit counter.

The initial value is indeterminate.

Always read this register using 16-bit data transfer instructions.

● P0TMR, P1TMR structure

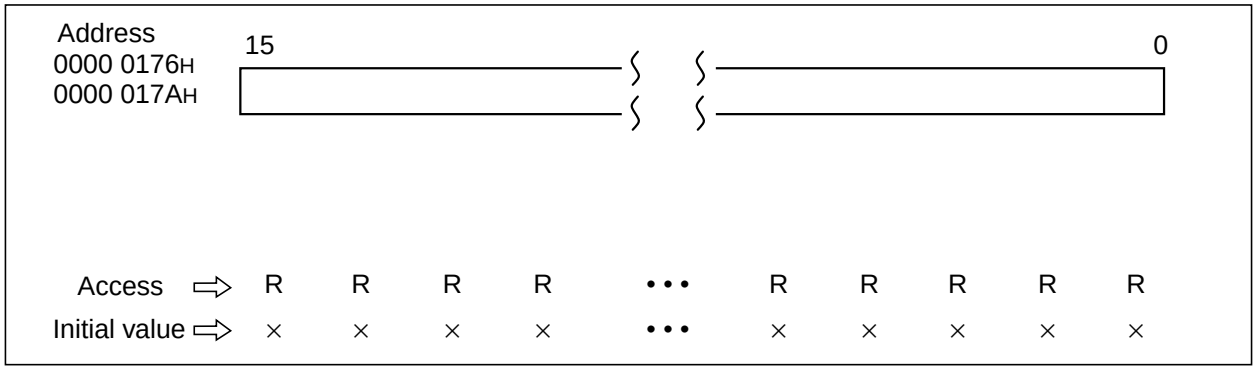


Figure 2-2 Structure of the 16-bit Counter Register

■ 16-bit Reload Register (P0TMRLR, P1TMRLR)

The 16-bit reload register stores the initial count value.

The initial value is indeterminate.

Always write to this register using 16-bit data transfer instructions.

● P0TMRLR, P1TMRLR structure

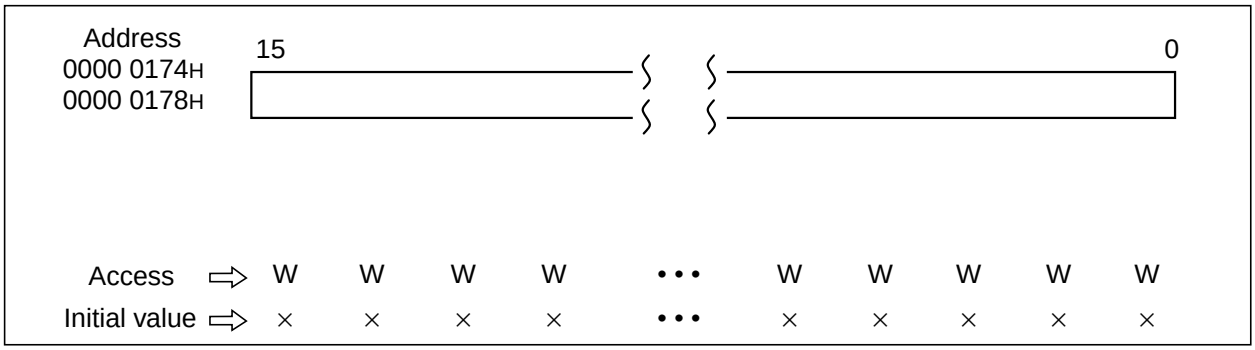


Figure 2-3 Structure of the 16-bit Reload Register

3. Reload Counter Operation

This section describes the operations of the 16-bit reload counter: Internal clock operation and Underflow operation

■ Internal Clock Operation

The machine clock divided by 2, 8, 32, 64 or 128 can be selected as the clock source when operating the counter from an internal clock.

Writing "1" to both the CNTE and TRG bits in the control status register enables and starts counting simultaneously.

Using the TRG bit as a trigger input is always available when the counter is enabled (CNTE = "1"), regardless of the operation mode.

Figure 3-1 shows counter activation and counter operation.

A time f (f: peripheral clock machine cycle) is required from the counter start trigger being input until the reload register data is loaded into counter.

● Counter activation and operation timing

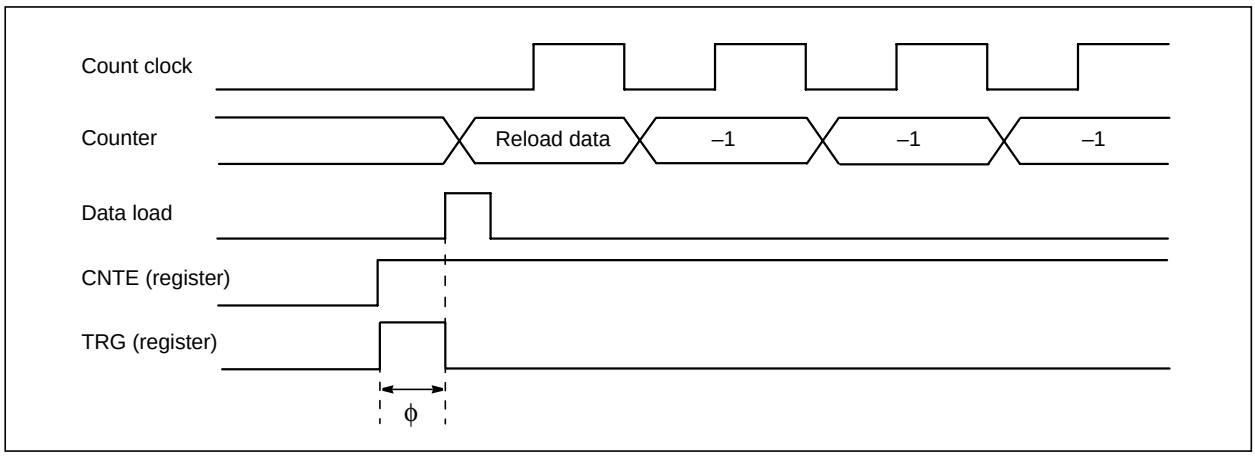


Figure 3-1 Counter Activation and Operation Timing

■ Underflow Operation

An underflow occurs when the counter value changes from 0000_H to $FFFF_H$. Therefore, an underflow occurs after "reload register setting + 1" counts.

If the RELD bit in the control register is "1" when the underflow occurs, the contents of the reload register are loaded into the counter and counting continues. When RELD is "0", counting stops with the counter at $FFFF_H$.

The UF bit in the control register is set when the underflow occurs. If the INTE bit is "1" at this time, an interrupt request is generated.

Figure 3-2 shows the operation when an underflow occurs.

● Underflow operation timing

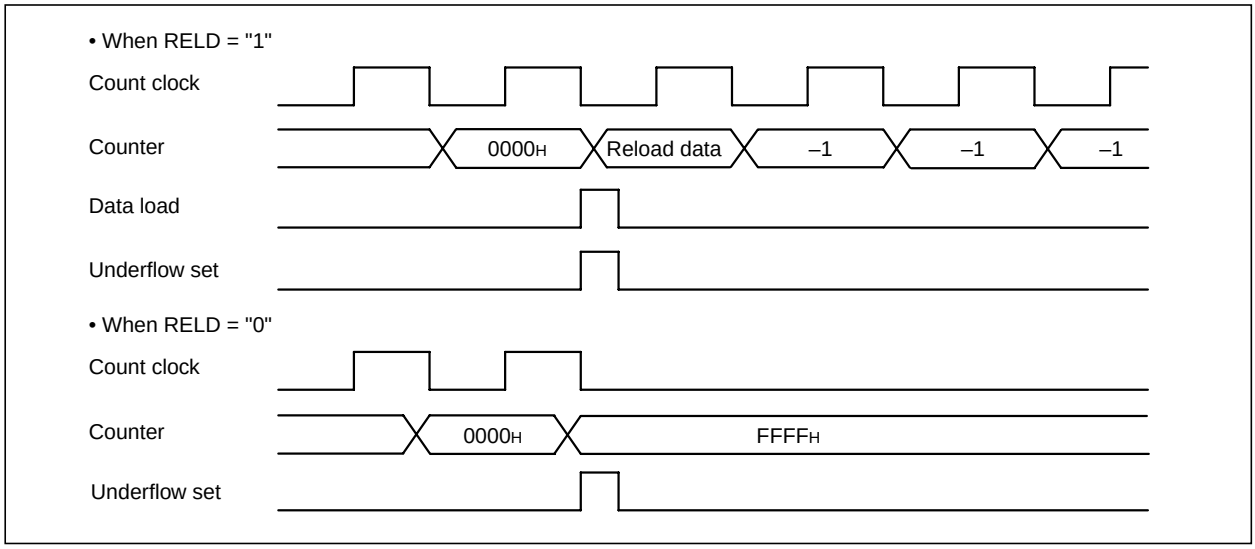


Figure 3-2 Underflow Operation Timing

■ Counter Operation States

The counter state is determined by the CNTE bit in the control register and the internal WAIT signal. The available states are CNTE = "0" and WAIT = "1" (STOP state: operation halted), CNTE = "1" and WAIT = "1" (WAIT state: waiting for a trigger), and CNTE = "1" and WAIT = "0" (RUN state: operating).

Figure 3-3 shows the transitions between each state.

● Counter state transitions

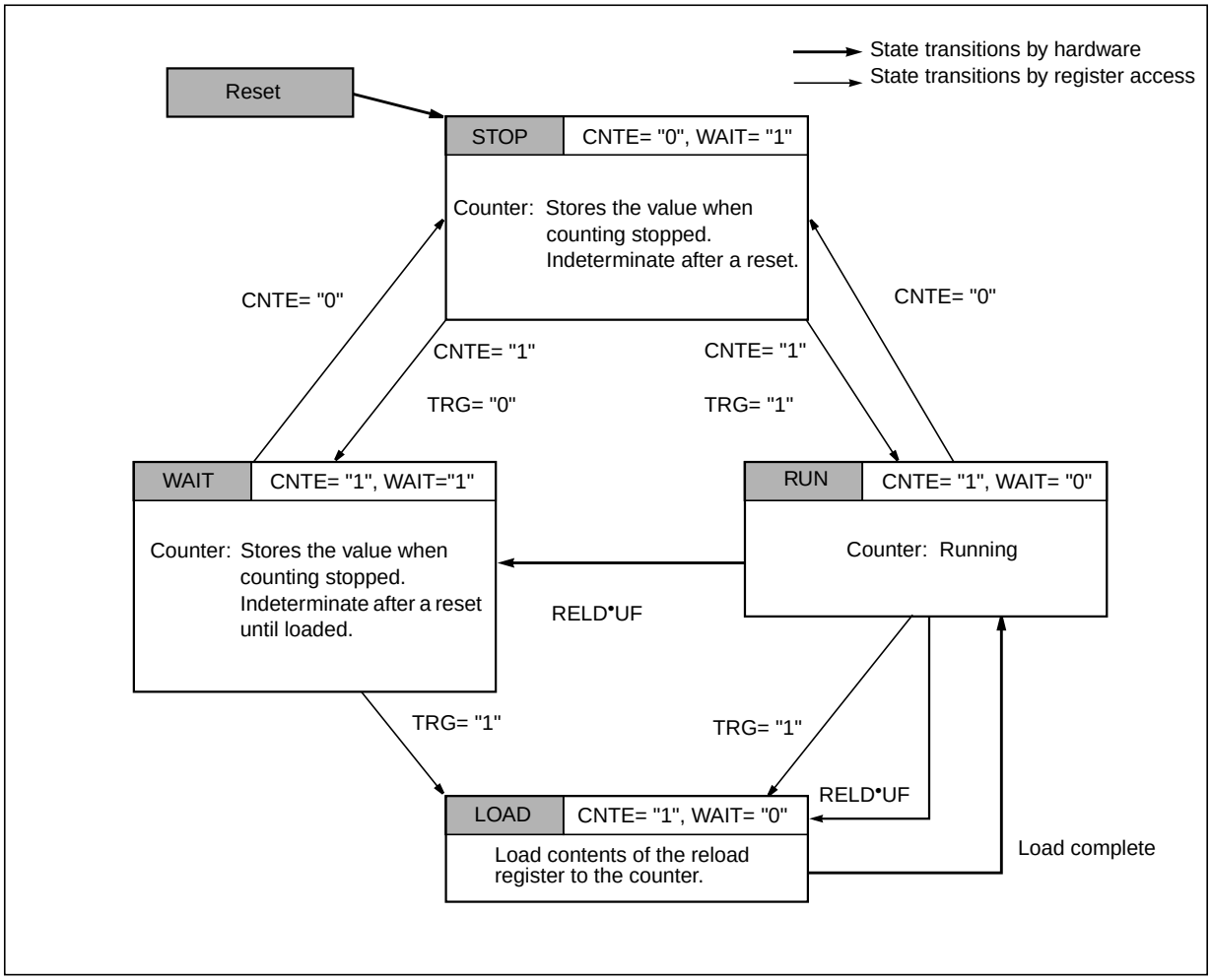


Figure 3-3 Counter State Transitions

4. PFM Operation and Setting

This section describes the following operations of the 16-bit pulse frequency mod (combining the functionality of both reload counters).

1. The underflow output of reload counter channel 0 is connected internally to the trigger input reload counter channel 1. The underflow output of reload counter channel 1 is connected internally to the trigger input of reload counter channel 0.

2. Both counters must be set up with RELD = “0”. Counter 0 should then be started by software trigger TRG = “1”. By starting counter 0 a high level is generated at the output. At the underflow condition of counter 0, counter 1 is automatically reloaded and started by the internal trigger (falling edge, set MOD1 = “1” for both counters) and a low level is generated at the output. At the underflow condition of counter 1, counter 0 is automatically reloaded and started by the internal trigger and a high level is generated at the output.

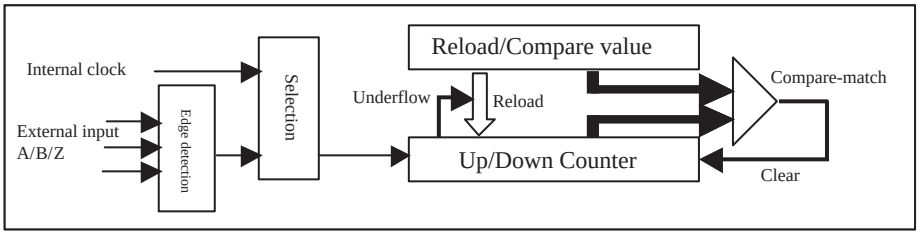
3. Interrupts can be set up on underflow condition of counter 0, counter 1 or both. The interrupts of counter 0 and counter 1 are OR’ed together,

4. The default output is low level if both CNTE = “0”, and both INV = “0”.

Chapter 41 Up/Down Counter

1. Overview

Triggered by an input signal, 16-bit Up/Down Counter counts up or down within the range of 0 to 65535. Specifically, Up/Down Counter running in the phase difference count mode is suitable for counting the encoder pulse of motors and other equipment. When encoder's output signals of phase A, phase B and phase Z are applied, the counter can achieve precise counting of rotation angles or number of revolutions.



2. Feature

- Format: 16 bit length or 8 bit x 2
- Quantity: 2 for 16 bit (Inputs: AIN0/BIN0/ZIN0, AIN2/BIN2/ZIN2)
  4 for 8 bit (Inputs: AIN0/BIN0/ZIN0, AIN1/BIN1/ZIN1, AIN2/BIN2/ZIN2, AIN3/BIN3/ZIN3)
- Count mode: Four types
  - Timer mode
    Count down the internal clock.
  - Up/down count mode
    Counting up is triggered by an AIN pin signal.
    Counting down is triggered by a BIN pin signal.
  - Phase difference count mode (Multiply by 2)
    Counting is triggered by the rising edge of a BIN pin signal. Up/Down Counter counts up or down, depending on the AIN pin signal level.
  - Phase difference count mode (Multiply by 4)
    Counting is triggered by the rising edge of AIN and BIN pin signals. Up/Down Counter counts up or down, depending on the ZIN pin signal level.
- Count Source
  Internal clock (Timer mode): Peripheral clock (CLKP) divided by 2 or 8
  External trigger (Up/down count mode): Edge detection (Rising/falling/both edges/no detection)
- Counting range: Any value between 0 and 65535 can be set.
- Interrupt: Select from the following four types:
  (1) Compare-match interrupt
  (2) Underflow interrupt
  (3) Overflow interrupt
  (4) Count direction change interrupt
- Others:
  Whether counting is performed or not can be controlled based on the pin input level.
  The software can activate or deactivate the counter.
  The ZIN pin has two functions: Counter clear and gate.
  The count direction flag allows identification of the previous count direction.

3. Configuration

Figure 3-1 Configuration Diagram

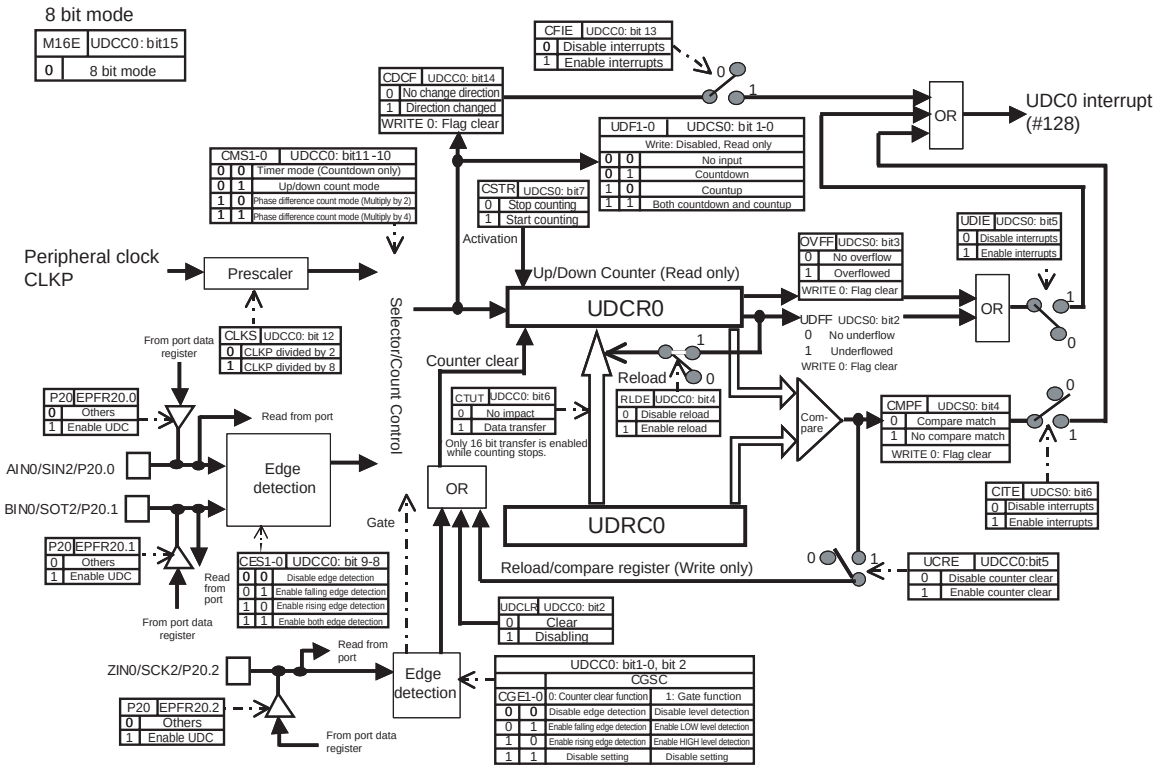


Figure 3-2 Configuration Diagram

Up/Down Counter 1 (8 Bit Mode)

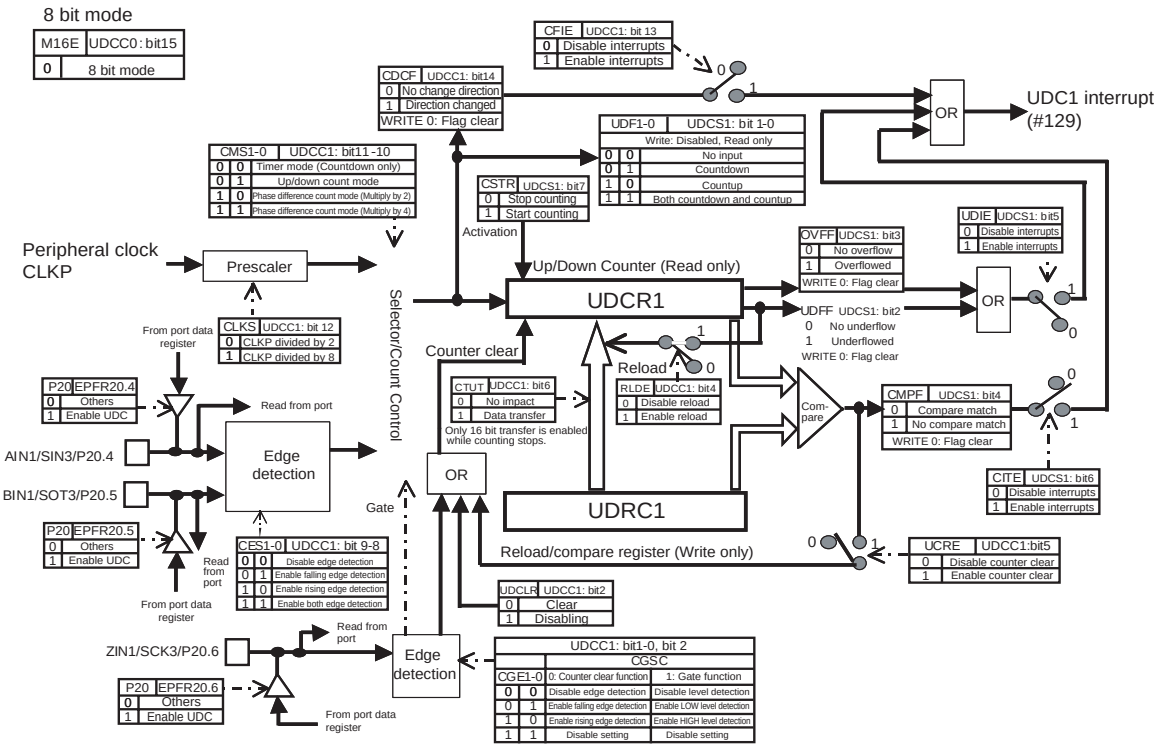


Figure 3-3 Configuration Diagram

Up/Down Counter (16 Bit Mode)

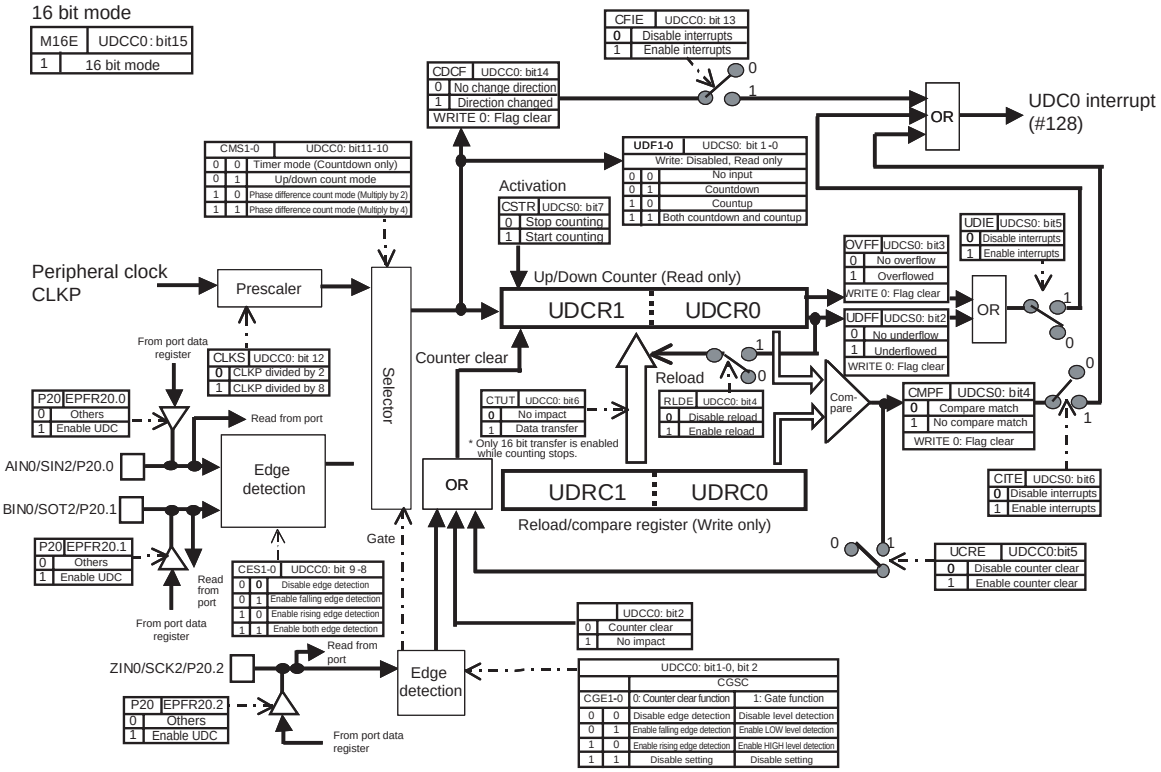
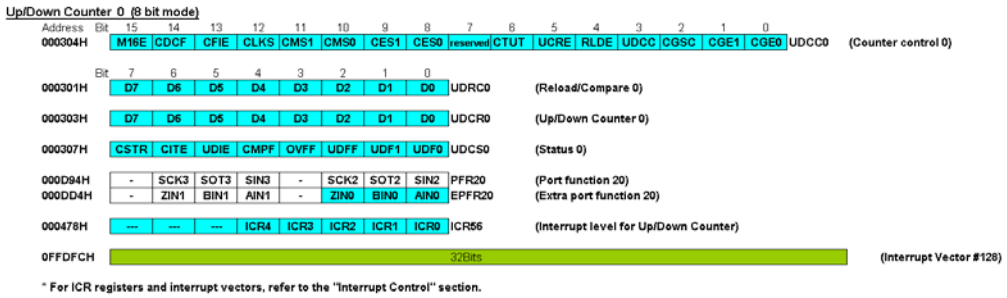
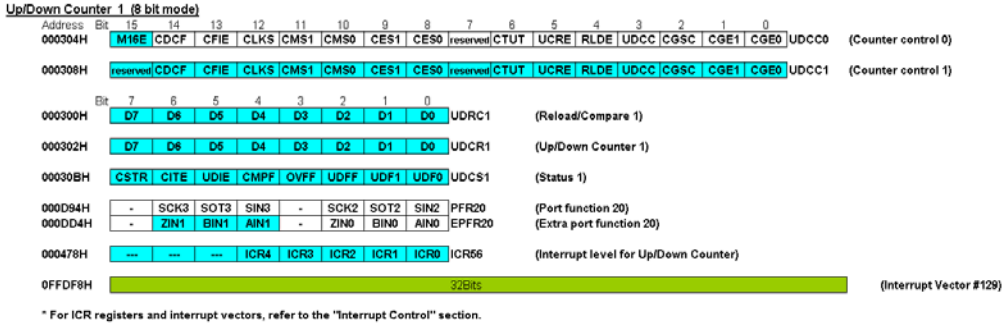


Figure 3-4 Register List



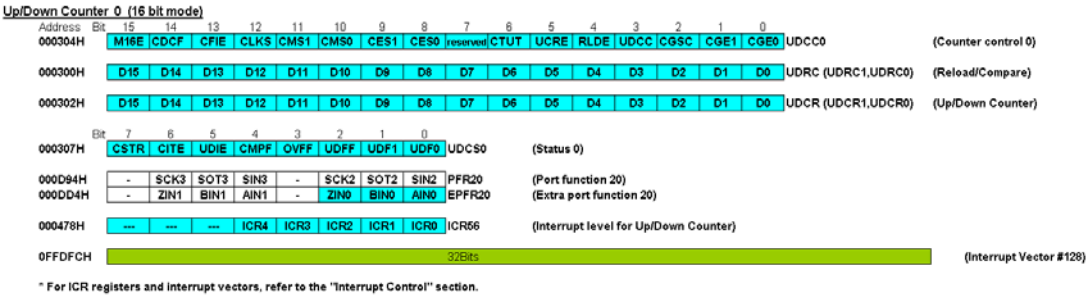
Note: For ICR registers and interrupt vectors, refer to “Chapter 24 Interrupt Control (Page No.315)”.

Figure 3-5 Register List



Note: For ICR registers and interrupt vectors, refer to “Chapter 24 Interrupt Control (Page No.315)”.

Figure 3-6 Register List



Note: For ICR registers and interrupt vectors, refer to “Chapter 24 Interrupt Control (Page No.315)”.

4. Register

4.1 UDCC: Counter Control Register

This register is used to control behaviors of Up/Down Counter.

- **UDCC0 (Up/Down Counter 0): Address 0304_H (Access: Byte, Half-word)**
- **UDCC1 (Up/Down Counter 1): Address 0308_H (Access: Byte, Half-word)**
- **UDCC2 (Up/Down Counter 2): Address 0314_H (Access: Byte, Half-word)**
- **UDCC3 (Up/Down Counter 3): Address 0318_H (Access: Byte, Half-word)**

15	14	13	12	11	10	9	8	bit
M16E/ Reserved	CDCF	CFIE	CLKS	CMS1	CMS0	CES1	CES0	UDCCH
0	0	0	0	0	0	0	0	Initial value
R/W *	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Attribute

7	6	5	4	3	2	1	0	bit
Reserved	CTUT	UCRE	RLDE	UDCLR	CGSC	CGE1	CGE0	UDCCL
0	0	0	0	1	0	0	0	Initial value
R/W0	R/W	R/W	R/W	R1,W	R/W	R/W	R/W	Attribute

(For attributes, refer to "Meaning of Bit Attribute Symbols (Page No.10)".)

- bit15: Enable 16 bit mode (Up/Down Counter 0 only)

M16E	Enable 16 bit mode
0	8 bit × 2 channel operation mode (8 bit mode)
1	16 bit × 1 channel operation mode (16 bit mode)

* Reserved bit (Up/Down Counter 1 and 3). Be sure to write 0. The read value is the value written.

- bit14: Count direction change flag (Interrupt request flag)

CDCF	Direction change detection	
	When read:	When written:
0	A direction change has not made.	Clear the flag.
1	Direction change has been made once or more.	Writing does not affect the operation.

  - When the count direction has been changed during count operation, the count direction change flag (CDCF) is set to "1".
  - Since the count direction is set to countdown immediately after a reset, the count direction change flag (CDCF) is set to "1" on counting up following the reset.
  - To enable interrupt requests, the interrupt request permission bit must be set (CFIE="1").
- bit13: Enable count direction change interrupt request

CFIE	Direction change interrupt request
0	Disable direction change interrupt requests.
1	Enable direction change interrupt requests.

When the interrupt request permission bit is set to "1", the interrupt request flag (CDCF) is enabled.

- bit12: Select internal prescaler

CLKS	Internal clock frequency
0	$F_{CLKP}/2$
1	$F_{CLKP}/8$

F_{CLKP}: Frequency of Peripheral clock (CLKP)

This setting is enabled only in the timer mode, in which only countdown is performed.

- bit11,10: Select count mode

CMS1	CMS0	Count mode
0	0	Timer mode (Countdown)
0	1	Up/down count mode
1	0	Phase difference count mode (Multiply by 2)
1	1	Phase difference count mode (Multiply by 4)

- bit9,8: Select count clock edge

CES1	CES0	Edge selection
0	0	Disable edge detection.
0	1	Detect a falling edge.
1	0	Detect a rising edge.
1	1	Detect both rising and falling edges.

This bit is used in the up/down count mode (CMS1,CMS0= “01”) to select the edge, to be detected, of an AIN and BIN pin signal. This setting is disabled in modes other than the up/down count.

- bit7: Reserved.

Be sure to write “0”. The read value is the value written.

- bit6: Counter write

CTUT	Data transfer
0	No impact on operation
1	Transfer data from the RCR register to UDCR.

During count operation (CSR.CSTR=“1”), the counter write bit must not be set to “1”.

- bit5: Enable compare-match clear

UCRE	Compare-match counter clear
0	Disable counter clear due to compare-match.
1	Enable counter clear due to compare-match.

This setting does not affect clear operations other than compare-match, such as ZIN pin clear.

- bit4: Enable reload

RLDE	Reload function
0	Disable reload function.
1	Enable reload function.

If the reload enable bit is set to “1”, the reload/compare value (RCR) is transferred to Up/Down Counter (UDCR) when Up/Down Counter is underflowed.

- bit3: Clear UDCR

UDCLR	Counter clear
0	Set (Clear) Up/Down Counter (UDCR) to “0000H”.
1	No impact on operation

- bit2: Select counter clear/gate

CGSC	ZIN pin function
0	Counter clear function
1	Gate function

- bit1,0: Select counter clear/gate edge

CGE1	CGE0	Edge detection/level selection	
		When the counter clear function is selected (CGSC="0")	When the gate function is selected (CGSC="1")
0	0	Disable edge detection.	Disable level detection. (Disable count.)
0	1	Detect a falling edge.	Detect a "L" level.
1	0	Detect a rising edge.	Detect a "H" level.
1	1	Disable setting.	Disable setting.

4.2 UDCS: Count Status Register

This register is used to control Up/Down Counter and to indicate the status of the counter.

- **UDCS0 (Up/Down Counter 0): Address 0307_H (Access: Byte, Half-Word)**
- **UDCS1 (Up/Down Counter 1): Address $030B_H$ (Access: Byte, Half-Word)**
- **UDCS2 (Up/Down Counter 2): Address 0317_H (Access: Byte, Half-Word)**
- **UDCS3 (Up/Down Counter 3): Address $031B_H$ (Access: Byte, Half-Word)**

7	6	5	4	3	2	1	0	bit
CSTR	CITE	UDIE	CMPF	OVFF	UDFF	UDF1	UDF0	
0	0	0	0	0	0	0	0	Initial value
R/W	R/W	R/W	R/W0	R/W0	R/W0	R/WX	R/WX	Attribute

(For attributes, refer to "Meaning of Bit Attribute Symbols (Page No.10)".)

- bit7: Enable count operation

CSTR	Count operation
0	Disable count operation.
1	Enable count operation. (Activate counter.)

- bit6: Enable compare interrupt requests

CITE	Compare interrupt request
0	Disable compare interrupt requests.
1	Enable compare interrupt requests.

Setting the interrupt request permission bit to "1" enables the interrupt request flag (CMPF).

- bit5: Enable overflow/underflow interrupt requests

UDIE	Overflow/underflow interrupt request
0	Disable overflow/underflow interrupt requests.
1	Enable overflow/underflow interrupt requests.

Setting the interrupt request permission bit to "1" enables the interrupt request flag (OVFF or UDFF).

- bit4: Compare detection flag

CMPF	Compare detection	
	When read:	When written:
0	Comparison results do not agree.	Clear the flag.
1	Comparison results agree.	Disable setting.

To enable interrupt requests, the interrupt request permission bit must be set (CITE= "1").

- bit3: Overflow detection flag

OVFF	Overflow detection	
	When read:	When written:
0	No overflow	Clear the flag.
1	An overflow has occurred.	Disable setting.

To enable interrupt requests, the interrupt request permission bit must be set (UDIE= "1").

- bit2: Underflow detection flag

UDFF	Underflow detection	
	When read:	When written:
0	No Underflow	Clear the flag.
1	An underflow has occurred.	Disable setting.

To enable interrupt requests, the interrupt request permission bit must be set (UDIE= “1”).

- bit1,0: Up/down flag

UDF1	UDF0	Previous count operation
0	0	No input
0	1	Count down
1	0	Count up
1	1	Both of count up and count down

4.3 UDCR: Up/Down Counter Register

This register is used to read the count value of Up/Down Counter.

- **UDCR10 (Up/Down Counter 0/1): Address 0302_H (Access: Byte, Half-Word)**
- **UDCR32 (Up/Down Counter 2/3): Address 0312_H (Access: Byte, Half-Word)**

Depending on the setting of the 16-bit mode enable bit (CCR.M16E), this register behaves differently.

■ 16 Bit Mode (M16E= “1”)

In the 16 bit mode, this register functions as 16-bit up/down counter register.

- UDCR10 (Up/Down Counter): Address 0302_H (Access: Half-word)
- UDCR32 (Up/Down Counter): Address 0312_H (Access: Half-word)

15	14	13	12	11	10	9	8	bit
D15	D14	D13	D12	D11	D10	D09	D08	
0	0	0	0	0	0	0	0	Initial value
R/WX	R/WX	R/WX	R/WX	R/WX	R/WX	R/WX	R/WX	Attribute

7	6	5	4	3	2	1	0	bit
D07	D06	D05	D04	D03	D02	D01	D00	
0	0	0	0	0	0	0	0	Initial value
R/WX	R/WX	R/WX	R/WX	R/WX	R/WX	R/WX	R/WX	Attribute

(For attributes, refer to “Meaning of Bit Attribute Symbols (Page No.10)”.)

For the 16 bit mode, be sure to read by half-word access.

■ 8 Bit Mode (M16E=“0”)

In the 8 bit mode, this register functions as a 8-bit up/down counter register 0 and a 8-bit up/down counter register 1.

- UDCR1 (Up/Down Counter 1): Address 00302_H (Access: Byte, Half-word)
- UDCR0 (Up/Down Counter 0): Address 00303_H (Access: Byte, Half-word)
- UDCR3 (Up/Down Counter 3): Address 00312_H (Access: Byte, Half-word)
- UDCR2 (Up/Down Counter 2): Address 00313_H (Access: Byte, Half-word)

7	6	5	4	3	2	1	0	bit
D07	D06	D05	D04	D03	D02	D01	D00	
0	0	0	0	0	0	0	0	Initial value
R/WX	R/WX	R/WX	R/WX	R/WX	R/WX	R/WX	R/WX	Attribute

(For attributes, refer to “Meaning of Bit Attribute Symbols (Page No.10)”.)

4.4 UDRC: Up/Down Reload/Compare Register

This register is used to reload a value to Up/Down Counter and for comparison.

This register is also used to write to Up/Down Counter.

- **UDRC10 (Up/Down Counter 0/1): Address 0300_H (Access: Byte, Half-Word)**
- **UDRC32 (Up/Down Counter 2/3): Address 0310_H (Access: Byte, Half-Word)**

Depending on the setting of the 16 bit mode enable bit (CCR.M16E), this register behaves differently.

■ 16 Bit Mode (M16E="1")

In the 16 bit mode, this register functions as 16-bit reload/compare register.

- UDCR10 (Reload compare): Address 00300_H (Access: Byte, Half-word)
- UDCR32 (Reload compare): Address 00310_H (Access: Byte, Half-word)

15	14	13	12	11	10	9	8	bit
D15	D14	D13	D12	D11	D10	D09	D08	
0	0	0	0	0	0	0	0	Initial value
RX, W	RX, W	RX, W	RX, W	RX, W	RX, W	RX, W	RX, W	Attribute

7	6	5	4	3	2	1	0	bit
D07	D06	D05	D04	D03	D02	D01	D00	
0	0	0	0	0	0	0	0	Initial value
RX, W	RX, W	RX, W	RX, W	RX, W	RX, W	RX, W	RX, W	Attribute

(For attributes, refer to "Meaning of Bit Attribute Symbols (Page No.10)".)

- The reload and compare values are same.

  When Up/Down Counter counts up, the value in RCR is used as a compare value.

  When Up/Down Counter counts down, underflow is generated and the value in RCR is used as a reload value for reloading.

  (Up/Down Counter counts between 0000_H and the reload/compare value.)

- In the 16 bit mode, be sure to write by half-word access.

■ 8 Bit Mode (M16E="0")

In the 8 bit mode, this register functions as 8-bit reload/compare register 0 and 8-bit reload/compare register 1.

- UDCR1 (reload compare 1): Address 00300_H (Access: Byte, Half-word)
- UDCR0 (reload compare 0): Address 00301_H (Access: Byte, Half-word)
- UDCR3 (reload compare 3): Address 00310_H (Access: Byte, Half-word)
- UDCR2 (reload compare 2): Address 00311_H (Access: Byte, Half-word)

7	6	5	4	3	2	1	0	bit
D07	D06	D05	D04	D03	D02	D01	D00	
0	0	0	0	0	0	0	0	Initial value
RX, W	RX, W	RX, W	RX, W	RX, W	RX, W	RX, W	RX, W	Attribute

(For attributes, refer to "Meaning of Bit Attribute Symbols (Page No.10)".)

- The reload and compare values are same.

  When Up/Down Counter counts up, the value in RCR is used as a compare value.

  When Up/Down Counter counts down, underflow is generated and the value in RCR is used as a reload value for reloading.

  (Up/Down Counter counts between 0000_H and the reload/compare value.)

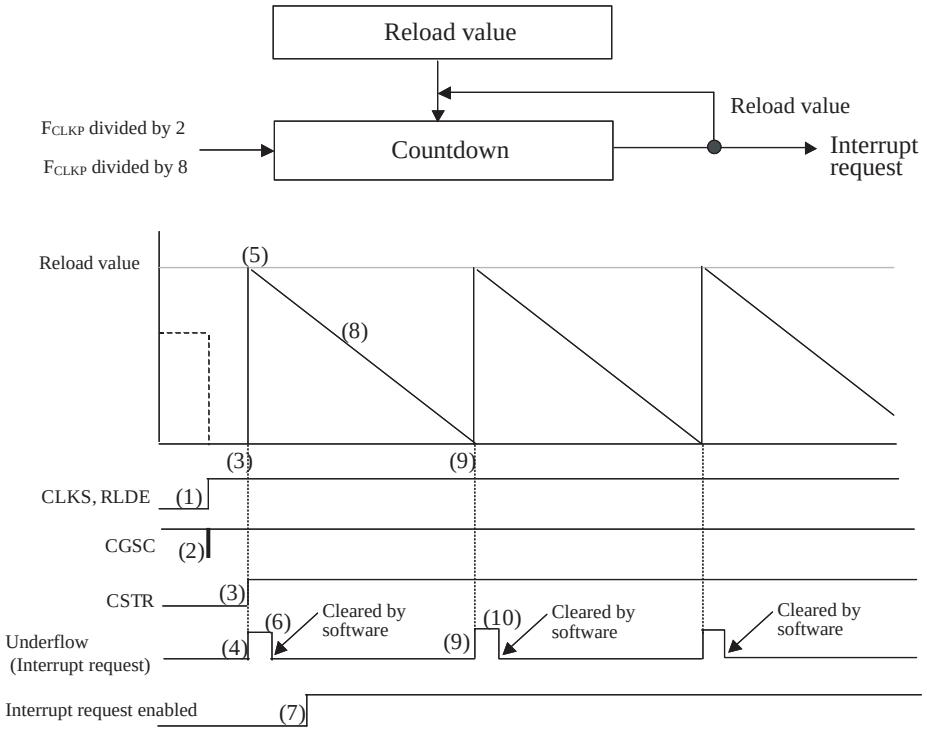
- Perform the following procedure to write to Up/Down Counter.

(1) Stop counting.
(2) Write a value to the reload/compare register.
(3) Write “1”to the counter write bit (CCR.CTUT).

5. Operation

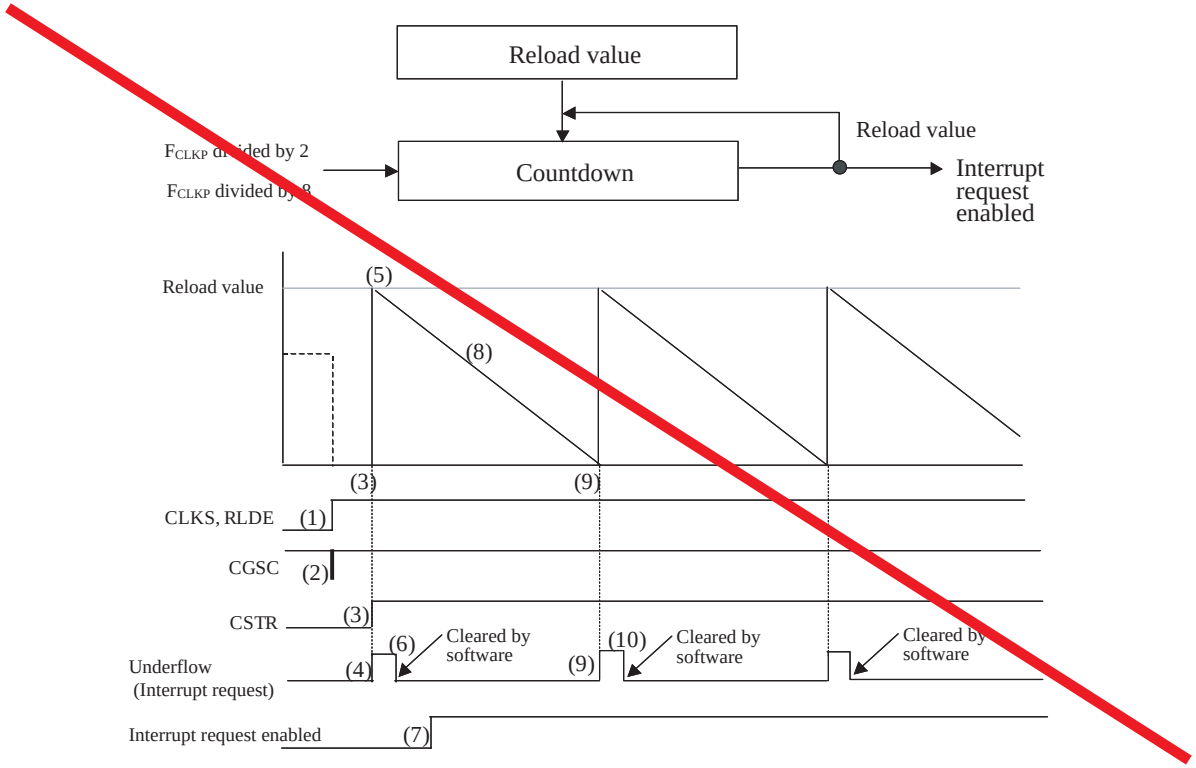
This section describes each operation mode for Up/Down Counter.

5.1 Timer Mode CMS[1:0]="00"



(1) An appropriate bit (Reload enable RLDE) is set.
(2) Up/Down Counter is cleared ("0" is written to CGSC).
(3) The software activates Up/Down Counter.
(4) An underflow occurs.
(5) The reload value is reloaded to Up/Down Counter.
(6) The software clears the underflow flag.
(7) The software enables interrupts.
(8) Up/Down Counter counts down.
(9) An underflow occurs. (An interrupt request has been made.)
(10) The software clears the underflow flag.
(11) Repeat (8) to (10).

5.2 Up/Down Count Mode CMS[1:0]="01"

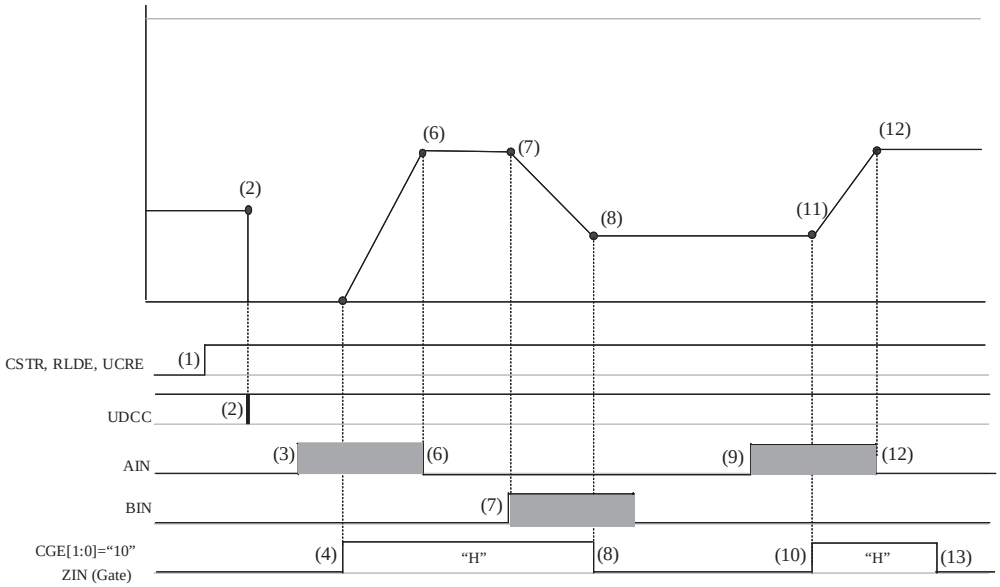


Up/Down Counter clear control using the ZIN pin

(1) Appropriate bits (Counting enable CSTR, Reload enable RLDE, Clear enable UCRE) are set.
(2) When pulse input to the AIN pin is detected, Up/Down Counter counts up.
(3) The count direction change flag is set to "1".
(4) When an edge is applied to the ZIN pin, Up/Down Counter is cleared.
(5) Continuous pulse input to the AIN pin causes Up/Down Counter to count up.
(6) The Up/Down Counter's count value agrees with the compare value (compare-match) and the compare-match flag is set to "1".
(7) Compare-match clears Up/Down Counter.
(8) Continuous pulse input to the AIN pin causes Up/Down Counter to count up.
(9) When pulse input to the AIN pin stops, Up/Down Counter stops counting.
(10) When pulse input to the BIN pin is detected, Up/Down Counter counts down.
(11) The count direction change flag is set to "1".
(12) Continuous pulse input to the BIN pin causes Up/Down Counter to count down.
(13) Up/Down Counter is underflowed and the underflow flag is set to "1".
(14) The underflow causes the reload value to be reloaded to Up/Down Counter.
(15) Next time when Up/Down Counter counts down, the compare-match flag is set to "1".

5.3 Up/Down Count Mode CMS[1:0]="01"

ZIN=Gate control



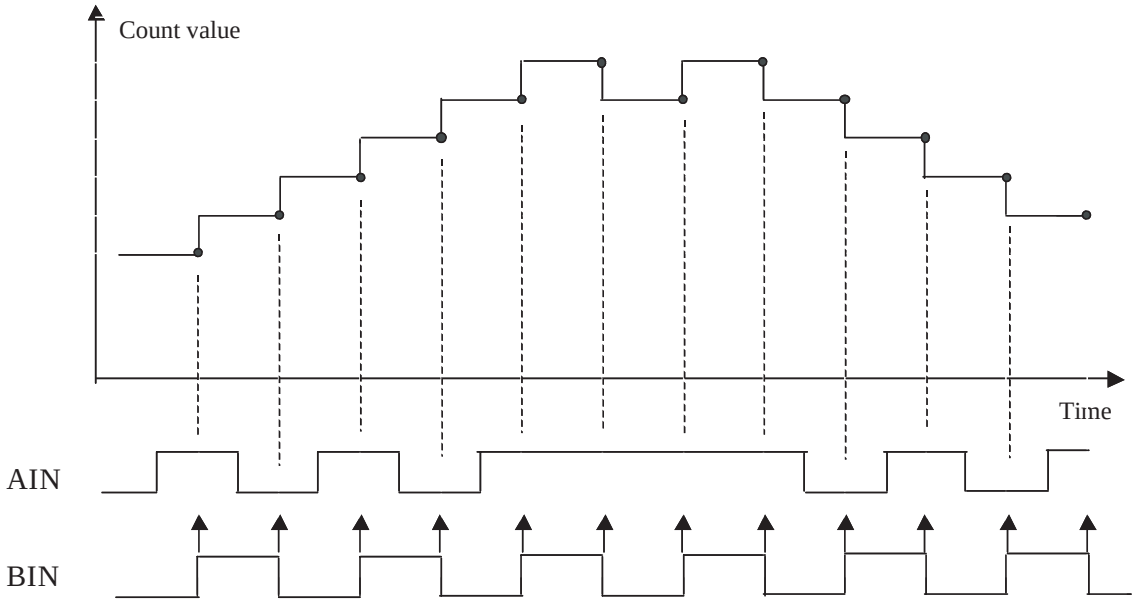
Countgate at the ZIN pin

(1) Appropriate bits (Counting enable CSTR, Reload enable RLDE and Clear enable UCRE) are set.
(2) Up/Down Counter is cleared. ("0" is written to CGSC).
(3) Neither pulse input to the AIN pin nor counting at the ZIN pin being enabled, Up/Down Counter neither counts up nor down.
(4) Counting is enabled at the ZIN pin.
(5) Up/Down Counter counts up.
(6) When pulse input to the AIN pin stops, Up/Down Counter stops counting.
(7) When a pulse input to the BIN pin is detected, Up/Down Counter counts down.
(8) When counting is disabled at the ZIN pin, Up/Down Counter stops counting.
(9) Neither pulse input to the AIN pin nor counting at the ZIN pin being enabled, Up/Down Counter neither counts up nor down.
(10) Counting is enabled at the ZIN pin.
(11) Up/Down Counter counts up.
(12) When pulse input to the AIN pin stops, Up/Down Counter stops counting.
(13) Counting is disabled at the ZIN pin.

5.4 Phase Difference Count Mode (Multiply by 2) CMS[1:0]="10"

Frequency multiplied by 2 in phase difference count mode:

On the rising and falling edges at the BIN count pin, Up/Down Counter counts up or down, depending on the voltage level at the AIN pin.



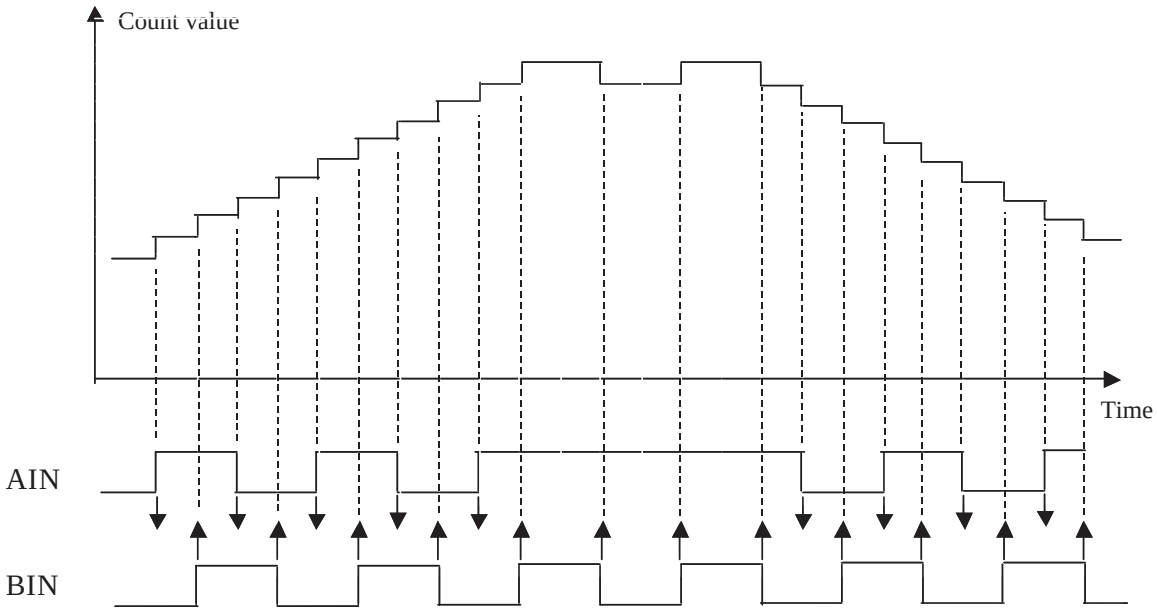
- Count up Conditions:
  - When the voltage level at the AIN pin detected on the rising edge at the BIN pin is "H"
  - When the voltage level at the AIN pin detected on the falling edge at the BIN pin is "L"
- Count down Conditions:
  - When the voltage level at the AIN pin detected on the rising edge at the BIN pin is "L"
  - When the voltage level at the AIN pin detected on the falling edge at the BIN pin is "H"

When this count mode is selected, selection of the edge to be detected using CES1 or CES0 is disabled.

5.5 Phase Difference Count Mode (Multiply by 4) CMS[1:0]="11"

Frequency multiplied by 4 in phase difference count mode:

On the rising and falling edges at the BIN pin, Up/Down Counter counts up or down, depending on the voltage level at the AIN pin, and on the rising and falling edges at the AIN pin, Up/Down Counter counts up or down, depending on the voltage level at the BIN pin.



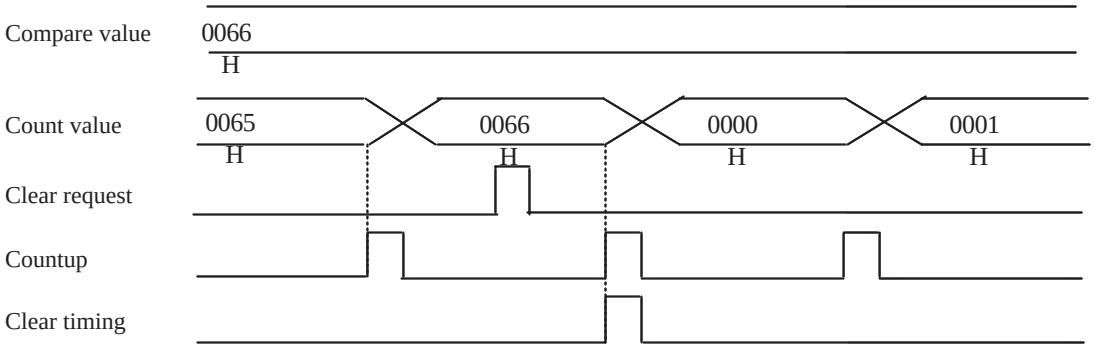
- Count up Conditions:
  - When the voltage level at the AIN pin detected on the rising edge at the BIN pin is "H"
  - When the voltage level at the AIN pin detected on the falling edge at the BIN pin is "L"
  - When the voltage level at the BIN pin detected on the rising edge at the AIN pin is "L"
  - When the voltage level at the BIN pin detected on the falling edge at the AIN pin is "H"
- Count down Conditions:
  - When the voltage level at the AIN pin detected on the rising edge at the BIN pin is "L"
  - When the voltage level at the AIN pin detected on the falling edge at the BIN pin is "H"
  - When the voltage level at the BIN pin detected on the rising edge at the AIN pin is "H"
  - When the voltage level at the BIN pin detected on the falling edge at the AIN pin is "L"

When Up/Down Counter is used to count encoder output, high precise counting of rotation angles and number of revolutions, as well as detecting of rotation directions, can be achieved by applying encoder output signals of phase A, phase B and phase Z to the AIN, BIN and ZIN pins, respectively.

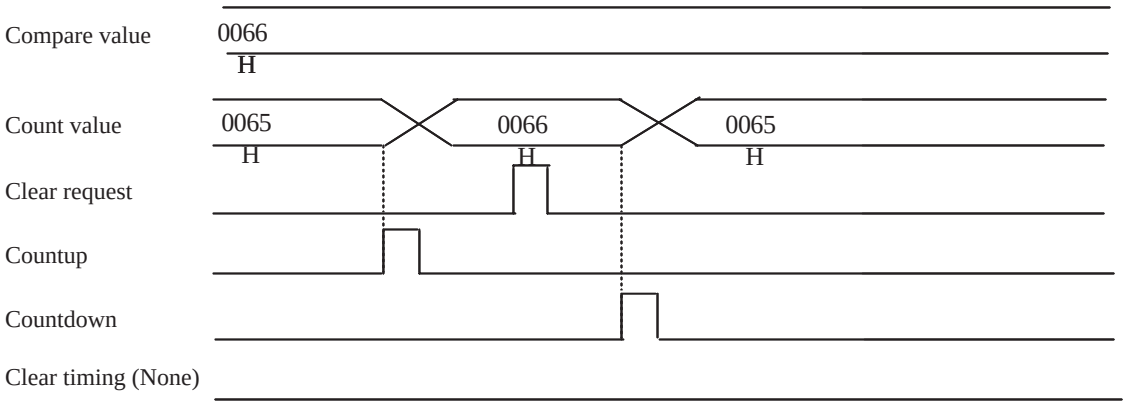
Note that when this count mode is selected, selection of the edge to be detected using CES1 or CES0 is disabled.

5.6 Clear Timing

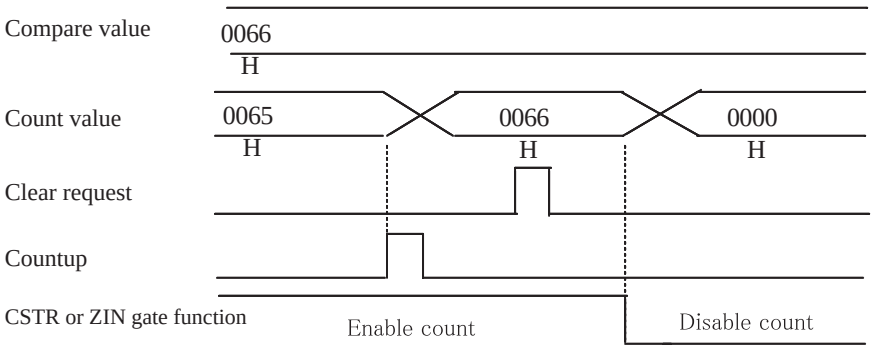
(1) When a clear request (Compare-match, ZIN edge detection and writing “0” to the clear bit UDCLR) is made, clear is performed next time when Up/Down Counter counts up.



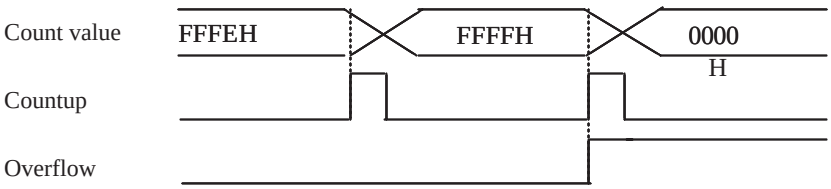
(2) Even if a clear request (Compare-match, ZIN edge detection and writing “0” to the clear bit DCC) is made, clear is not performed when UP/Down Counter counts neither up nor down.



(3) If Up/Down Counter does not count up after a clear request (Compare-match, ZIN edge detection and writing “0” to the clear bit DCC) is made, the counter is cleared when counting is disabled (CSTR=“0”).

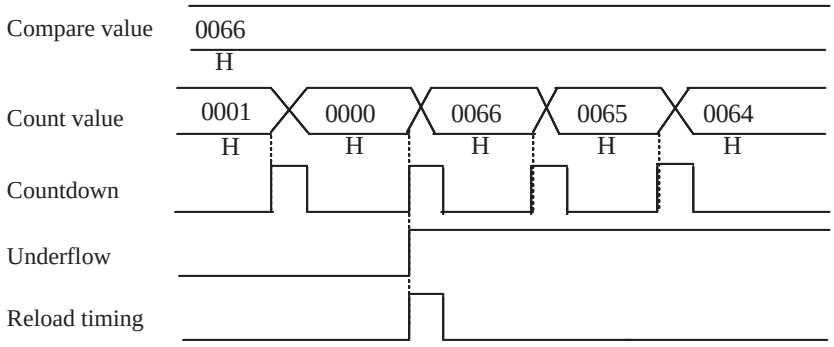


(4) When Up/Down Counter exceeds the maximum count, the overflow flag is set to “1” and the counter value is returned to “0000”.



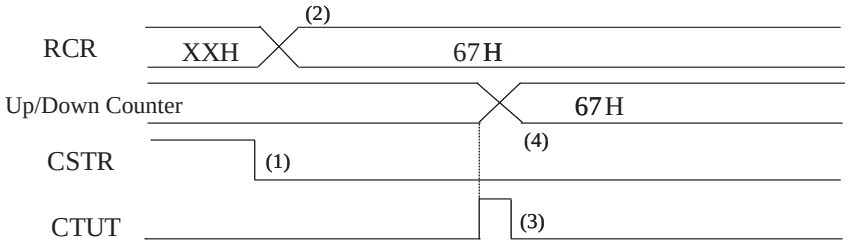
5.7 Reload Timing

The next time when Up/Down Counter counts down below “0000”, an underflow occurs (an interrupt request is made) and then reloading is performed.



Note: If clear and reload operations occur at the same time, clear takes precedence.

5.8 Writing a Value to Counter



(1) Counting of Up/Down Counter is disabled.
(2) A value is written to PCR.
(3) “1” is written to the count write bit CTUT.
(4) A value is transferred from the reload/compare register RCR to Up/Down Counter.

6. Setting

Table 6-1 Required Settings to Run Up/Down Counter in Timer Mode

Setting	Setting registers	Setting procedure*
Set the reload value.	Reload/compare register (UDRC)	See 7.16.
(Optional) Set a value to Up/Down Counter or Clear the count value of Up/Down Counter.	Reload/compare register (UDRC)	See 7.5.
	Count control register (UDCC)	See 7.8.
Set a bit length.	Count control register (UDCC)	See 7.1.
Set the count mode to timer mode.		See 7.2.
Select a count source.		See 7.3.
Enable reloading at the time of underflow.		See 7.7.
Enable count control (clear/gate) using the ZIN pin.		See 7.9 and 7.10
Activate Up/Down Counter.	Count status register (UDCS)	See 7.11.

*: For the setting procedure, refer to the section indicated by the number.

Table 6-2 Required Settings to Run Up/Down Counter in Up/Down Count Mode

Setting	Setting registers	Setting procedure*
Set the reload value/compare value.	Reload/compare register (UDRC)	See 7.16.
(Optional) Set a value to Up/Down Counter Or Clear the count value of Up/Down Counter.	Reload/compare register (UDRC)	See 7.5.
	Count control register (UDCC)	See 7.8.
Set a bit length.	Count control register (UDCC)	See 7.1.
Set the count mode to up/down count mode.		See 7.2.
Select the edge, to be detected, of a signal (AIN or BIN), for which counting is performed.		See 7.4.
Enable clearing of Up/Down Counter at the time of the counting following a compare-match.		See 7.6.
Enable reloading at the time of underflow.		See 7.7.
Enable count control (clear/gate) using the ZIN pin.		See 7.9 and 7.10
Activate Up/Down Counter.	Count status register (UDCS)	See 7.11.

*: For the setting procedure, refer to the section indicated by the number.

Table 6-3 Required Settings to Run Up/Down Counter in Phase Difference Count Mode (Multiply by 2 or 4)

Setting	Setting registers	Setting procedure*
Set the reload value/compare value.	Reload/compare register (UDRC).	See 7.16
(Optional) Set a value to Up/Down Counter or Clear the count value of Up/Down Counter.	Reload/compare register (UDRC)	See 7.5
	Count control register (UDCC)	See 7.8
Set a bit length.	Count control register (UDCC)	See 7.1
Set the count mode to phase difference count mode (Multiply by 2 or 4).		See 7.2
Enable clearing of Up/Down Counter at the time of the counting following a compare-match.		See 7.6
Enable reloading at the time of underflow.		See 7.7
Enable count control (clear/gate) using the ZIN pin.		See 7.9 and 7.10
Activate Up/Down Counter.	Count status register (UDCS)	See 7.11

*: For the setting procedure, refer to the section indicated by the number.

Table 6-4 Required Settings for Up/Down Counter Interrupt

Setting	Setting registers	Setting procedure*
Set Up/Down Counter interrupt vectors and Up/Down Counter interrupt levels.	Refer to “Chapter 24 Interrupt Control (Page No.315)”.	See 7.17
Set Up/Down Counter interrupts. Clear interrupt requests. Enable interrupt requests.	Count control register (UDCC) Count status register (UDCS)	See 7.19

*: For the setting procedure, refer to the section indicated by the number.

Table 6-5 Required Settings to Deactivate Up/Down Counter

Setting	Setting registers	Setting procedure*
Deactivate Up/Down Counter (Controlled through the ZIN pin)	Count control register (UDCC)	See 7.10
Deactivate Up/Down Counter.	Count status register (UDCS)	See 7.11

*: For the setting procedure, refer to the section indicated by the number.

7. Q&A

7.1 How do I select a bit length (8 or 16) of Up/Down Counter?

Use the 16 bit mode enable bit (UDCC.M16E).

Up/Down Counter's bit length	16 bit mode enable bit (M16E)
To set the bit length to 8	Set the bit to “0”.
To set the bit length to 16 bit	Set the bit to “1”.

7.2 What types of count modes are available and how are they set?

There are four types of count modes:

Timer, Up/down count, Phase difference count (Multiply by 2 or 4)

Use the count mode selection bits (UDCC.CMS[1:0]) to set a count mode.

Count mode	Count mode selection bit (CMS[1:0])
To set the count mode to timer	Set the bit to “00”.
To set the count mode to up/down count	Set the bit to “01”.
To set the count mode to phase difference count (Multiply by 2)	Set the bit to “10”.
To set the count mode to phase difference count mode (Multiply by 4)	Set the bit to “11”.

7.3 How do I select a count source for Up/Down Counter running in the timer mode?

Use the internal prescaler select bit (UDCC.CLKS).

Count source for timer mode	Internal prescaler select bit (CLKS)
To obtain the F_{CLKP} divided by 2	Set the bit to “0”.
To obtain the F_{CLKP} divided by 8	Set the bit to “1”.

7.4 How do I select the edge with which Up/Down Counter running in the Up/down count mode detects an input signal (AIN or BIN)?

Use count clock edge select bits (UDCC.CES[1:0]).

Edge to be detected by counter	Count clock edge select bit (CES[1:0])
To disable detection	Set the bit to “00”.
To enable detection of a falling edge	Set the bit to “01”.
To enable detection of a rising edge	Set the bit to “10”.
To enable detection of both edges	Set the bit to “11”.

7.5 How do I set a value to Up/Down Counter?

A value can be set to Up/Down Counter by writing the value to the reload/compare register (RCR) and then writing “1” to the counter write bit (UDCC.CTUT).

7.6 When the Up/Down Counter's count-up value agrees with the compare value (RCR[0:1]), how do I enable clearing of Up/Down Counter the next time when the counter counts up?

Use the up/down counter clear enable bit (UDCC.UCRE).

When the count-up value agrees with the compare value and then Up/Down Counter counts up:	Up/down counter clear enable bit (UCRE)
To disable clearing of Up/Down Counter	Set the bit to “0”.
To enable clearing of Up/Down Counter	Set the bit to “1”.

7.7 How do I enable reloading of the reload value (RCR[1:0]) to Up/Down Counter when Up/Down Counter is underflowed?

Use the reload enable bit (UDCC.RLDE).

When the count-up value agrees with the compare value:	Reload enable bit (RLDE)
To disable reloading of the reload value (RCR) to Up/ Down Counter	Set the bit to “0”.
To enable reloading of the reload value (RCR) to Up/Down Counter	Set the bit to “1”.

7.8 How do I clear Up/Down Counter?

Up/Down Counter can be cleared in any of the following ways:

- Writing “0” to the up/down counter clear bit (UDCC.UDCLR).
- Applying an edge to the ZIN pin (For details, refer to 7.9.)
- When the compare value agrees with the Up/Down Counter's count-up value.
- When Up/Down Counter tries to count up after reaching the maximum count.
- Reset input (INIT pin input, watchdog reset, software reset)

7.9 How do I clear Up/Down Counter using the ZIN pin?

Use counter clear gate bit (UDCC.CGSC) and counter clear gate edge select bits (UDCC.CGE[1:0]). (These bits are enabled in the up/down count mode.)

ZIN pin input	Counter clear gate bit (CGSC)	Counter clear gate edge select bit (CGE[1:0])
To disable edge detection (clear)	Set the bit to “0”.	Set the bit to “00”.
To clear Up/Down Counter on the falling edge	Set the bit to “0”.	Set the bit to “01”.
To clear Up/Down Counter on the rising edge	Set the bit to “0”.	Set the bit to “10”.

GCE[1:0]=“11” indicates that setting is disabled.

7.10 How do I control Up/Down Counter's count operation using the ZIN pin?

Use counter clear gate bit (UDCC.CGSC) and counter clear gate edge select bits (UDCC.CGE[1:0]). (These settings are enabled for all the count modes.)

ZIN pin input	Counter clear gate bit (CGSC)	Counter clear gate edge select bit (CGE[1:0])
To disable level detection (counting)	Set the bit to “1”.	Set the bit to “00”.
To start counting up or down at the “L” level To stop counting up or down at the “H” level	Set the bit to “1”.	Set the bit to “01”.
To stop counting up or down at the “L” level To start counting up or down at the “H” level	Set the bit to “1”.	Set the bit to “10”.

GCE[1:0]= “11” indicates that setting is disabled.

7.11 How do I enable/disable Up/Down Counter's count operation?

Use the count activate bit (UDCS.CSTR).

When the count-up value agrees with the compare value:	Count activate bits (UDCS.CSTR)
To disable Up/Down Counter's count operation	Set the bit to “0”.
To enable Up/Down Counter's count operation (To activate count operation)	Set the bit to “1”.

- How do I start counting?

  Timer mode — Counting starts using the internal clock (See 7.3.)

  Up/down count mode — Counting starts when the edge of an AIN or BIN pin input signal is detected. (See 7.4.)

  Phase difference count mode — Counting starts when a phase difference between AIN and BIN pins is detected.

  Note that the count operation enable level must be detected, before the ZIN pin's gate function can be selected.

7.12 How do I know the previous count direction (the current rotation direction)?

Use the up/down flags (UDCS.UDF[1:0]).

Up/down flag (UDF[1:0])
“00” indicates that no counting is performed after resetting.
“01” indicates that counting down is performed.
“10” indicates that counting up is performed.
“11” indicates that both counting up and down are performed, resulting in no change in the count value.

This flag has nothing to do with interrupts. So, use the count direction change flag (UDCC.CDCF) for interrupt processing.

7.13 How do I know count direction changes?

Use the count direction change flag (UDCC.CDCF).

Count direction change flag (CDCF)
“0” indicates that no direction change has been made after clearing the flag.
“1” indicates that a direction change has been made once or more after clearing the flag.

7.14 How do I know that a compare-match has occurred?

Use the compare detection flag (UDCS.CMPF).

Compare detection flag (CMPF)
“0” indicates that the Up/Down Counter's count value does not agree with the compare value.
“1” indicates that the Up/Down Counter's count value agrees with the compare value.

Regardless of counter operations (counting up/down, or a value being set or reloaded), the compare detection flags are set to “1” when the count value agrees with the compare value.

7.15 How do I know that an overflow or underflow has occurred?

Use the overflow detection flag (UDCS.OVFF) and the underflow detection flag (UDCS.UDFF).

OVFF =“1” indicates that Up/Down Counter has been overflowed.
UDFF =“1” indicates that Up/Down Counter has been underflowed.

7.16 How do I set the reload/compare value?

Set a value to the reload/compare registers (UDRC). (This value is used as a compare or reload value.)

7.17 What are interrupt-related registers?

Configure the up/down counter interrupt vectors and up/down counter interrupt level settings.

The following table shows the relationship among the up/down counter number, interrupt levels and vectors:

For details on interrupt levels and interrupt vectors, refer to “Chapter 24 Interrupt Control (Page No.315)”.

	Interrupt vector (Default)	Interrupt level set bit (ICR[4:0])
Up/Down Counter 0/1 (16 bit)	#128 Address: 0FFDFCh	Interrupt level register (ICR56) Address: 0478h
Up/Down Counter 0 (8 bit)		
Up/Down Counter 1 (8 bit)	#129 Address: 0FFDF8h	
Up/Down Counter 2/3 (16 bit)	#130 Address: 0FFDF4h	Interrupt level register (ICR57) Address: 0479h
Up/Down Counter 2 (8 bit)		
Up/Down Counter 3 (8 bit)	#131 Address: 0FFDF0h	

The following interrupt request flags are not automatically cleared:

- Count direction change: UDCC.CDCF
- Compare detection: UDCS.CMPF
- Overflow: UDCS.OVFF
- Underflow: UDCS.UDFF

So, the software must write “0” to the interrupt request flag before control is returned from interrupt processing.

7.18 What interrupts are available and how are they selected?

There are three interrupt causes:

- Count direction change
- compare-match
- overflow/underflow

An interrupt request is made by ORing these three interrupt causes; each interrupt cause cannot be isolated.

Use the interrupt request permission bit to enable a desired interrupt.

7.19 How do I enable (select), disable or clear interrupts?

Interrupt request enable and interrupt request flags

To enable (select) interrupts, use the following interrupt request permission bits:

- Count direction change interrupt request permission bits: UDCC.CFIE
- Compare interrupt request permission bits : UDCS.CITE
- Overflow/underflow interrupt request permission bits : UDCS.UDIE

	Interrupt request permission bits (CFIE, CITE and UDIE)
To disable interrupt requests	Set the bit to “0”.
To enable interrupt requests	Set the bit to “1”.

To clear interrupt requests, use the following interrupt request bits:

- For count direction changes : UDCC.CDCF
- For compare detection : UDCS.CMPF
- For overflow : UDCS.OVFF
- For underflow : UDCS.UDFF

	Interrupt request bits (CDCF, CMPF, OVFF and UDFF)
To clear interrupt requests	Write “0”.

8. Caution

- The count direction is set to “countdown” immediately after resetting the counter. So, when the counter counts up immediately after resetting, the count direction change bit (UDCC.CDCF) is set to “1” to indicate a direction change has been made.

- When the up/down counter register UDCR has reached the maximum count, the overflow flag is set to 1 and counting continues. This time UDCR is cleared.

- The minimum pulse width for AIN, BIN and ZIN signals is 2xT (T=1/CLKP: Period of a peripheral clock)

- If you determine whether a change has been made to the count direction change interrupt and count direction flags, you must take it into consideration that when several direction changes have been made continuously in a short period of time, the count direction flag may be returned to the original value, which looks as if no change has been made.

- The compare detection flag (UDCS.CMPF) is set to “1” when Up/Down Counter's count value agrees with the compare value during both counting up and down. These flags are also set to 1 when:
  - The reload value is reloaded to Up/Down Counter; or
  - The Up/Down Counter's count value agrees with the compare value when Up/Down Counter is activated.

- The Up/Down Counter's count value is cleared by a clear request which is generated:
  - On the edge of a signal input from the ZIN pin;
  - By writing “0” to the up/down counter clear bit (UDCC.UDCLR); or
  - When the compare value agrees with the count value.

  In addition,
  - On reset input (INITX, RST and watchdog reset); or
  - When the counter counts up from the maximum account,

  the count value is also set to “0000_H”.

- When Up/Down Counter clear and reload requests are made at the same time, the former takes precedence over the latter.

- When Up/Down Counter is counting up, writing to the counter is disabled.
  {Writing “1” to the counter write bit (UDCC.CTUT) after writing to the UDRC register is disabled.}
  Should the software perform a reload operation during counting, the reload takes precedence and the event that should have taken place no longer occurs.

- The software cannot clear the up/down flags (UDCS.UDF[1:0]) to “0”. Only reset (initialization) can clear the flag to “0”.

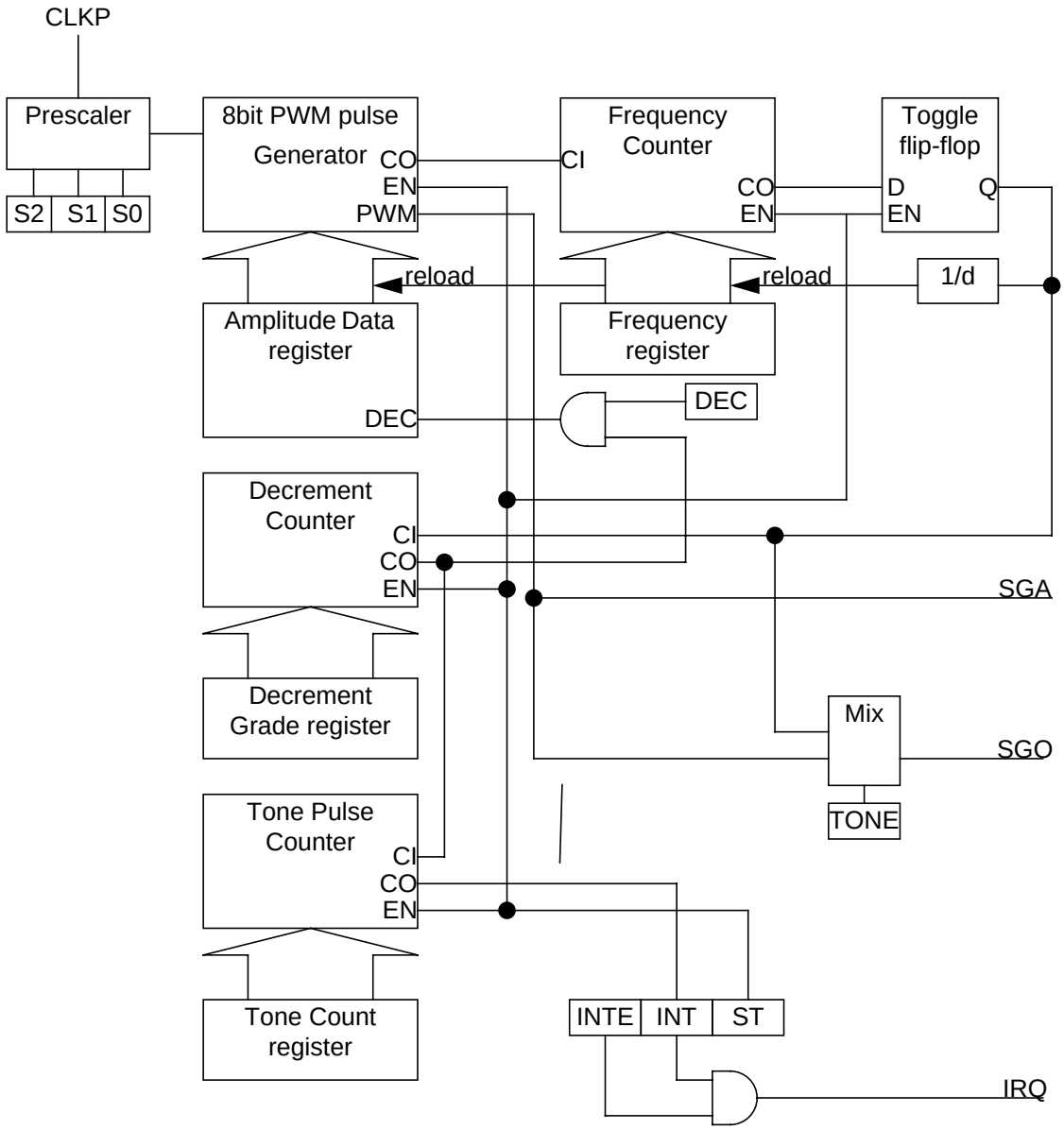
Chapter 42 Sound Generator

1. Overview

This Chapter provides an overview of the Sound Generator, describes the register structure and functions, and describe the operation of the Sound Generator.

The Sound Generator consists of the Sound Control register, Frequency Data register, Amplitude Data register, Decrement Grade register, Tone Count register, Sound Disable register, PWM pulse generator, Frequency counter, Decrement counter and Tone Pulse counter.

2. Block Diagram



3. Registers

Sound Control register

Address: 000198H

Bit number	15	14	13	12	11	10	9	8
SGCRH	TST	S2	S1	S0	—	—	BUSY	DEC
Read/write	(R/W)	(R/W)	(R/W)	(R/W)	—	—	(R)	(R/W)
Initial value	(0)	(0)	(0)	(0)	—	—	(0)	(0)

Address: 000199H

Bit number	7	6	5	4	3	2	1	0
SGCRL	—	—	TONE	—	—	INTE	INT	ST
Read/write	—	—	(R/W)	—	—	(R/W)	(R/W)	(R/W)
Initial value	—	—	(0)	—	—	(0)	(0)	(0)

Frequency Data register

Address: 00019AH

Bit number	15	14	13		3	2	1	0
SGFR	D15	D14	D13		D3	D2	D1	D0
Read/write	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)
Initial value	(X)	(X)	(X)	(X)	(X)	(X)	(X)	(X)

Amplitude Data register

Address: 00019CH

Bit number	15	14	13	12	11	10	9	8
SGAR	D7	D6	D5	D4	D3	D2	D1	D0
Read/write	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)
Initial value	(0)	(0)	(0)	(0)	(0)	(0)	(0)	(0)

Tone Count register

Address: 00019EH

Bit number	15	14	13	12	11	10	9	8
SGTR	D7	D6	D5	D4	D3	D2	D1	D0
Read/write	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)
Initial value	(X)	(X)	(X)	(X)	(X)	(X)	(X)	(X)

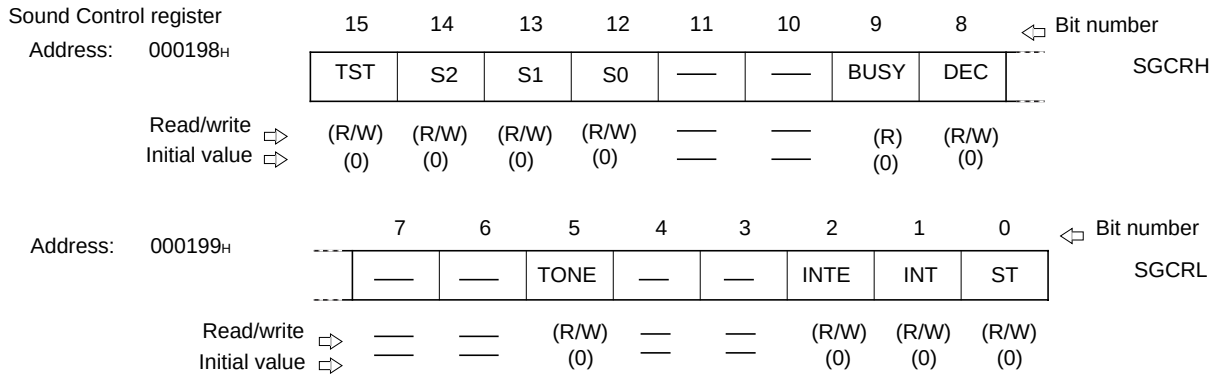
Decrement Grade register

Address: 00019FH

Bit number	7	6	5	4	3	2	1	0
SGDR	D7	D6	D5	D4	D3	D2	D1	D0
Read/write	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)
Initial value	(X)	(X)	(X)	(X)	(X)	(X)	(X)	(X)

3.1 Register Details

■ Sound Control Register (SGCR)



[bit 15] TST : Test bit

This bit is prepared for the device test. In any user application it should be set to "0".

[bits14 to12] S2 to S0 : Operation clock select bits

These bits specify the clock input signal for the Sound Generator.

S2	S1	S0	Clock input
0	0	0	CLK
0	0	1	1/2 CLK
0	1	0	1/4 CLK
0	1	1	1/8 CLK
1	0	0	1/16 CLK

[bit 9] BUSY : Busy bit

This bit indicates whether the Sound Generator is in operation. This bit is set to "1" upon the ST bit is set to "1". It is reset to "0" when the ST bit is reset to "0" and the operation is completed at the end of one tone cycle. Any write instruction performed on this bit has no effect.

[bit 8] DEC : Auto-decrement enable bit

The DEC bit is prepared for an automatic decrement of the sound in conjunction with the Decrement Grade register.

If this bit is set to "1", the stored value in the Amplitude Data register is decremented by 1(one), every time when the Decrement counter counts the number of tone pulses from the toggle flip-flop specified by the Decrement Grade register.

[bit 5] TONE : Tone output bit

When this bit is set to "1", the SGO signal becomes a simple square-waveform (tone pulses) from the toggle flip-flop. Otherwise it is the mixed (AND logic) signal of the tone and PWM pulses.

[bit 2] INTE : Interrupt enable bit

This bit enables the interrupt signal of the Sound Generator. When this bit is "1" and the INT bit is set to "1", the Sound Generator signals an interrupt.

[bit 1] INT : Interrupt bit

This bit is set to "1" when the Tone Pulse counter counts the number of the tone pulses specified by the Tone Count register and Decrement Grade register.

This bit is reset to "0" by writing "0". Writing "1" has no effect and Read-Modify-Write instructions always result in reading "1".

[bit 0] ST : Start bit

This bit is for starting the operation of the Sound Generator. While this bit is "1", the Sound Generator perform its operation.

When this bit is reset to "0", the Sound Generator stops its operation at the end of the current tone cycle. The BUSY bit indicates whether the Sound Generator is fully stopped.

■ Frequency Data Register (SGFR)

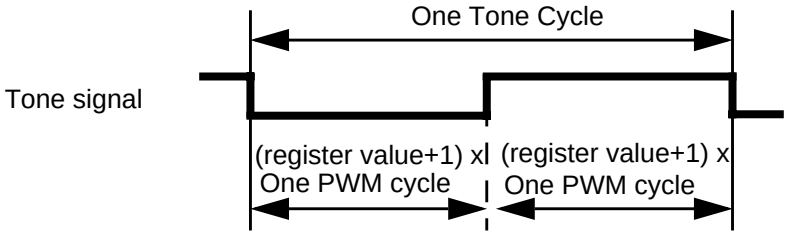
Frequency Data register

Address: 00019A$_H$

	15	14	13		3	2	1	0	⇦ Bit number
	D15	D14	D13		D3	D2	D1	D0	SGFR
Read/write ⇨	(R/W)	(R/W)	(R/W)		(R/W)	(R/W)	(R/W)	(R/W)	
Initial value ⇨	(X)	(X)	(X)		(X)	(X)	(X)	(X)	

The Frequency Data register stores the reload value for the Frequency counter. The stored value represents the frequency of the sound (or the tone signal from the toggle flip-flop). The register value is reloaded into the counter at every transition of the toggle signal.

The following figure shows the relationship between the tone signal and the register value.



It should be noted that modifications of the register value while operation may alter the duty cycle of 50% depending on the timing of the modification.

■ Amplitude Data Register (SAGR)

Amplitude Data register

Address: 00019C$_H$

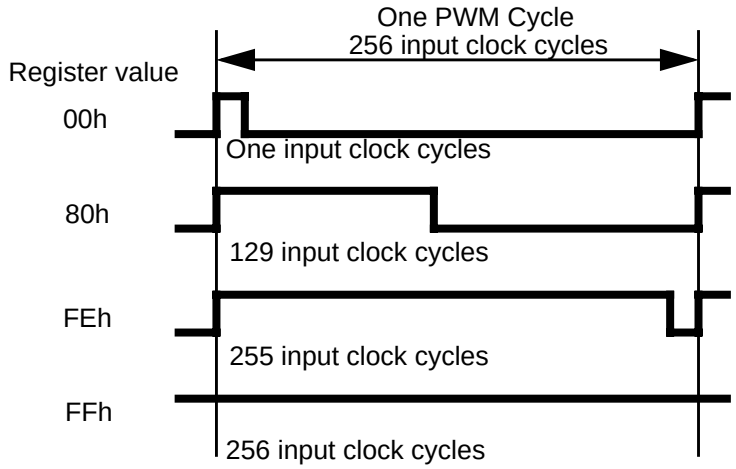
	15	14	13	12	11	10	9	8	⇦ Bit number
	D7	D6	D5	D4	D3	D2	D1	D0	SGAR
Read/write ⇨	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	
Initial value ⇨	(0)	(0)	(0)	(0)	(0)	(0)	(0)	(0)	

The Amplitude Data register stores the reload value for the PWM pulse generator. The register

value represents the amplitude of the sound. The register value is reloaded into the PWM pulse generator at the end of every tone cycle.

When the DEC bit is "1" and the Decrement counter reaches its reload value, this register value is decremented by 1(one). And when the register value reaches "00", further decrements are not performed. However the sound generator continues its operation until the ST bit is cleared.

The following figure shows the relationship between the register value and the PWM pulse.



When the register value is set to "FF", the PWM signal is always "1".

■ Decrement Grade Register (SGDR)

Decrement Grade register

Address: 00019F$_H$

	7	6	5	4	3	2	1	0	⇦ Bit number
	D7	D6	D5	D4	D3	D2	D1	D0	SGDR
Read/write ⇨	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	
Initial value ⇨	(X)	(X)	(X)	(X)	(X)	(X)	(X)	(X)	

The Decrement Grade register stores the reload value for the Decrement counter. They are prepared to automatically decrement the stored value in the Amplitude Data register.

When the DEC bit is "1" and the Decrement counter counts the number of tone pulses up to the reload value, the stored value in the Amplitude Data register is decremented by 1 (one) at the end of the tone cycle.

This operation realizes automatic decrement of the sound with fewer number of CPU interventions.

It should be noted that the number of the tone pulses specified by this register equals to "register value +1". When the Decrement Grade register is set to "00", the decrement operation is performed every tone cycle.

■ Tone Count Register (SGTR)

Tone Count register

Address: 00019E$_H$

	15	14	13	12	11	10	9	8	⇦ Bit number
	D7	D6	D5	D4	D3	D2	D1	D0	SGTR
Read/write ⇨	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	
Initial value ⇨	(X)	(X)	(X)	(X)	(X)	(X)	(X)	(X)	

The Tone Count register stores the reload value for the Tone Pulse counter. The Tone Pulse counter accumulate the number of tone pulses (or number of decrement operations) and when it

reaches the reload value it sets the INT bit. They are intended to reduce the frequency of interrupts.

The count input of the Tone Pulse counter is connected to the carry-out signal from the Decrement counter. And when the Tone count register is set to "00", the Tone Pulse counter sets the INT bit every carry-out from the Decrement counter. Thus the number of accumulated tone pulses is:
((Decrement Grade register) +1) x ((Tone Count register) +1)

i.e. When the both registers are set to "00", the INT bit is set every tone cycle.

Chapter 43 Stepper Motor Controller

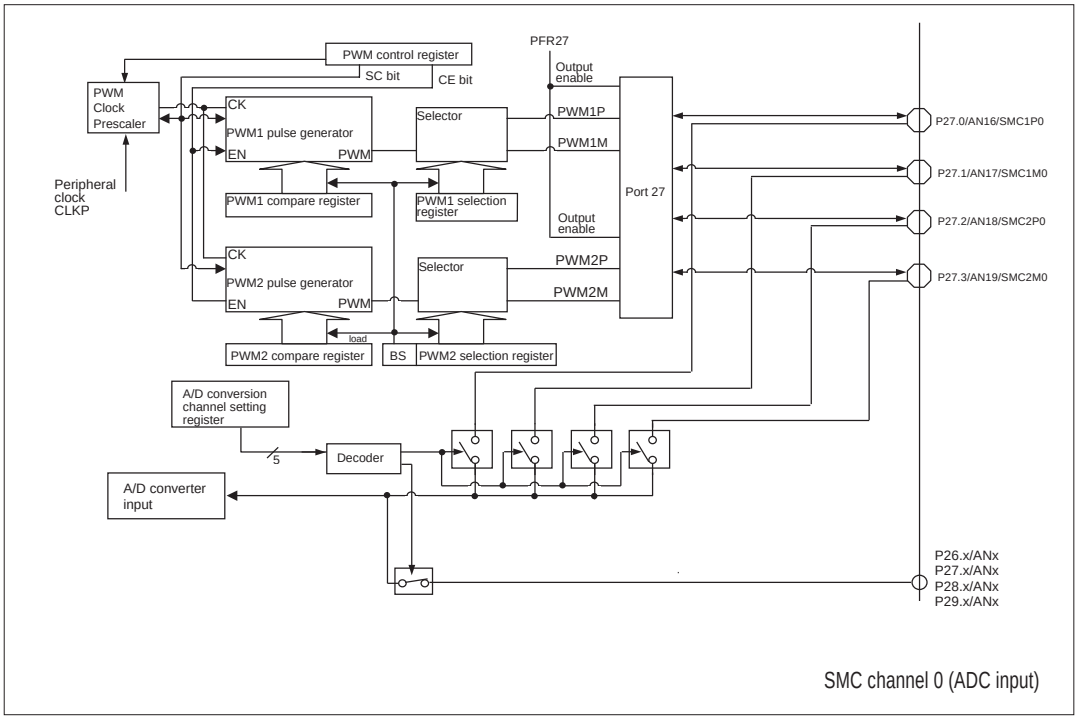
1. Overview

The stepping motor controller consists of PWM pulse generators, motor drivers, selector logic circuits and A/D converter inputs.

The four motor drivers have a high-output driving capability and two motor coils can be connected directly to four pins. The motor rotation is designed to be controlled by a combination of the PWM pulse generators and selector logic circuits. The synchronization mechanism enables synchronous operation of two PWM pulse generators.

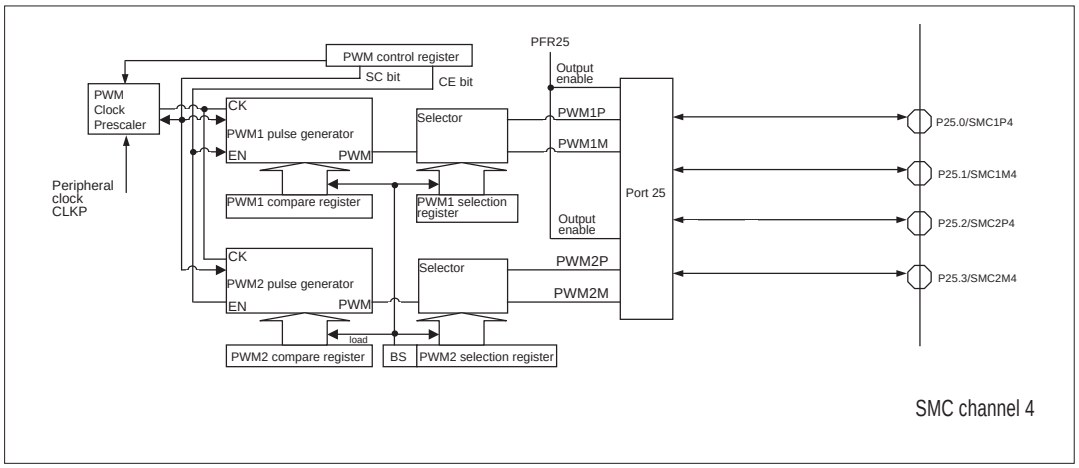
■ Block Diagram of Stepping Motor Controller

Figure 1-1 Stepping Motor Controller with A/D Converter Function



Remark: The SMC channels 0, 1, 2 and 3 are shared with ADC inputs.

Figure 1-2 Stepping Motor Controller



Remark: The SMC channels 4 and 5 are not shared with ADC inputs.

2. Registers

There are seven types of registers for the stepping motor controller:
PWM Control register
PWM1 Compare register
PWM2 Compare register
PWM1 Selection register
PWM2 Selection register

2.1 Registers for Stepping Motor Controller

PWM Control register (PWC0, PWC1, PWC2, PWC3, PWC4, PWC5)

Address	7	6	5	4	3	2	1	0
0x0C1, 0x0C3	–	P2	P1	P0	CE	SC	–	–
0x0C5, 0x0C7	–	R/W	R/W	R/W	R/W	R/W	–	–
0x0C9, 0x0CB	–	0	0	0	0	0	–	–

PWM1 Compare register (PWC10, PWC11, PWC12, PWC13, PWC14, PWC15)

Address	15	14	13	12	11	10	9	8
0x092, 0x09A	–	–	–	–	–	–	D9	D8
0x0A2, 0x0AA	–	–	–	–	–	–	R/W	R/W
0x0B2, 0x0BA	–	–	–	–	–	–	X	X

Address	7	6	5	4	3	2	1	0
0x093, 0x09B	D7	D6	D5	D4	D3	D2	D1	D0
0x0A3, 0x0AB	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
0x0B3, 0x0BB	X	X	X	X	X	X	X	X

PWM2 Compare register (PWC20, PWC21, PWC22, PWC23, PWC24, PWC25)

Address	15	14	13	12	11	10	9	8
0x090, 0x098	–	–	–	–	–	–	D9	D8
0x0A0, 0x0A8	–	–	–	–	–	–	R/W	R/W
0x0B0, 0x0B8	–	–	–	–	–	–	X	X

Address	7	6	5	4	3	2	1	0
0x091, 0x099	D7	D6	D5	D4	D3	D2	D1	D0
0x0A1, 0x0A9	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
0x0B1, 0x0B9	X	X	X	X	X	X	X	X

PWM2 Selection register (PWS20, PWS21, PWS22, PWS23, PWS24, PWS25)

Address	15	14	13	12	11	10	9	8
0x096, 0x09E	–	BS	P2	P1	P0	M2	M1	M0
0x0A6, 0x0AE	–	R/W	R/W	R/W	R/W	R/W	R/W	R/W
0x0B6, 0x0BE	–	0	0	0	0	0	0	0

PWM1 Selection register (PWS10, PWS11, PWS12, PWS13, PWS14, PWS15)

Address	7	6	5	4	3	2	1	0
0x097, 0x09F	–	–	P2	P1	P0	M2	M1	M0
0x0A7, 0x0AF	–	–	R/W	R/W	R/W	R/W	R/W	R/W
0x0B7, 0x0BF	–	–	0	0	0	0	0	0

2.2 PWM Control Register

The PWM control register starts/stops the stepping motor controller, performs interrupt control and performs setting of external output pins, etc., for the stepping motor controller.

■ PWM Control Register

PWM Control register (PWC0, PWC1, PWC2, PWC3, PWC4, PWC5)

Address	7	6	5	4	3	2	1	0
0x0C1, 0x0C3	–	P2	P1	P0	CE	SC	–	–
0x0C5, 0x0C7	–	R/W	R/W	R/W	R/W	R/W	–	–
0x0C9, 0x0CB	–	0	0	0	0	0	–	–

[bit 7] Reserved bit

Always set the reserved bit to "0".

[bit 6 to 4] P2, P1, P0: Operating clock select bits (bits to select operating clock)

The P2, P1 and P0 bits specify the clock input signal for the PWM pulse generator.

P2	P1	P0	Clock input	PWM cycle (at F_{cp} = 32 MHz)		PWM cycle (at F_{cp} = 40 MHz)	
				SC=0	SC=1	SC=0	SC=1
0	0	0	F_{CP}	8.0 us	32.0 us	6.4 us	25.6 us
0	0	1	$F_{CP}/4$	32.0 us	128.0 us	25.6 us	102.4 us
0	1	0	$F_{CP}/5$	40.0 us	160.0 us	32.0 us	128.0 us
0	1	1	$F_{CP}/6$	48.0 us	192.0 us	38.4 us	153.6 us
1	0	0	$F_{CP}/8$	64.0 us	256.0 us	51.2 us	204.8 us
1	0	1	$F_{CP}/10$	80.0 us	320.0 us	64.0 us	256.0 us
1	1	0	$F_{CP}/12$	96.0 us	384.0 us	76.8 us	307.2 us
1	1	1	$F_{CP}/16$	128.0 us	512.0 us	102.4 us	409.6 us

F_{CP}: Peripheral clock CLKP

Note:

After operation clock selection, set 1 in CE.

[bit 3] CE: Count enable bit

The CE bit enables operation of the PWM pulse generator. When "1" is set to the CE bit, the PWM pulse generator starts its operating. The PWM2 pulse generator starts after the PWM1 pulse generator starts in order to reduce the switching noise generated by the output driver.

When the CE bit is cleared to 0 during operation of the PWM pulse generator, the generator is initialized to stop operating.

[bit 2] SC: 8/10 bits switching bit

When "1" is set to the SC bit, the PWM pulse generator operates at 10 bit. When "0" is set to the SC bit, the

PWM pulse generator operates at 8 bit.

[bit 1 to 0] Reserved bits

Always set the reserved bits to "00".

2.3 PWM1&2 Compare Registers

The value of the two 8(10) bits compare register of PWM1&2 determine the width of the PWM pulse. The stored "00_H (000_H)" value indicates that the PWM duty is 0%, and the stored "FF_H" ("$3FF_H$") value indicates that the PWM duty is 99.6% (99.9%).

■ PWM1&2 Compare Registers

The PWM1&2 compare registers can be accessed at any time, but the changed value is reflected in the pulse width at the end of the current PWM cycle after "1" is set to the BS bit of the PWM2 selection register.

When "0" is set to SC bit of the PWM control register, and PWM performs 8-bit operation, the D9 and D8 bits are undefined value.

Be sure to perform half-word access to the PWM1&2 compare registers

PWM1 Compare register (PWC10, PWC11, PWC12, PWC13, PWC14, PWC15)

Address	15	14	13	12	11	10	9	8
0x092, 0x09A	–	–	–	–	–	–	D9	D8
0x0A2, 0x0AA	–	–	–	–	–	–	R/W	R/W
0x0B2, 0x0BA	–	–	–	–	–	–	X	X
Address	7	6	5	4	3	2	1	0
0x093, 0x09B	D7	D6	D5	D4	D3	D2	D1	D0
0x0A3, 0x0AB	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
0x0B3, 0x0BB	X	X	X	X	X	X	X	X

PWM2 Compare register (PWC20, PWC21, PWC22, PWC23, PWC24, PWC25)

Address	15	14	13	12	11	10	9	8
0x090, 0x098	–	–	–	–	–	–	D9	D8
0x0A0, 0x0A8	–	–	–	–	–	–	R/W	R/W
0x0B0, 0x0B8	–	–	–	–	–	–	X	X
Address	7	6	5	4	3	2	1	0
0x091, 0x099	D7	D6	D5	D4	D3	D2	D1	D0
0x0A1, 0x0A9	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
0x0B1, 0x0B9	X	X	X	X	X	X	X	X

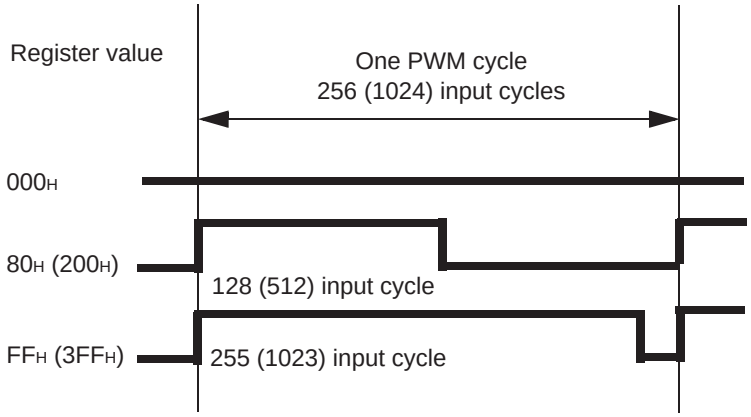
[bit 15 to 10] Reserved bit

Always set the reserved bit to "0".

[bits 9 to 0] D9 to D0: Compare data

These bits are used to set the PWM pulse width.

Figure 2-1 Relationship between the Compare Register Setting Value and PWM Pulse Width



2.4 PWM1&2 Selection Registers

The PWM1&2 selection registers determine whether to set the output of the external pin of the stepping motor controller to "0", "1", PWM pulse or high impedance.

■ PWM1&2 Selection Registers

PWM2 Selection register (PWS20, PWS21, PWS22, PWS23, PWS24, PWS25)

Address	15	14	13	12	11	10	9	8
0x096, 0x09E	–	BS	P2	P1	P0	M2	M1	M0
0x0A6, 0x0AE	–	R/W	R/W	R/W	R/W	R/W	R/W	R/W
0x0B6, 0x0BE	–	0	0	0	0	0	0	0

PWM1 Selection register (PWS10, PWS11, PWS12, PWS13, PWS14, PWS15)

Address	7	6	5	4	3	2	1	0
0x097, 0x09F	–	–	P2	P1	P0	M2	M1	M0
0x0A7, 0x0AF	–	–	R/W	R/W	R/W	R/W	R/W	R/W
0x0B7, 0x0BF	–	–	0	0	0	0	0	0

[bit 15] Reserved bit

Always set the reserved bit to "0".

[bit 14] BS: Rewrite bit

The BS bit makes the setting for the PWM output match another setting. Until the BS bit is set, changes made to the two compare registers and two selection registers are not reflected in the output signal.

When "1" is set to the BS bit, the PWM pulse generator and the selector load the values of the registers at the end of the current PWM cycle. The BS bit is cleared to “0” automatically at the beginning of the next PWM cycle.

When "1" is set to the BS bit and is being cleared automatically by software simultaneously, the BS bit is set to “1” (no change is made to the BS bit) and the automatic clearing is cancelled.

When "0" is set to the BS bit and is being cleared automatically by software simultaneously, the BS bit is cleared to “0”, however the PWM pulse generator and the selector are not load the values of the registers at the end of the current PWM cycle.

Note:

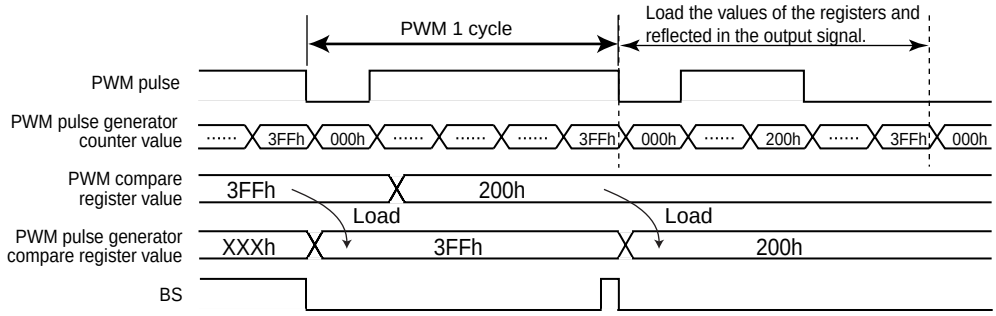
When the BS bit equals “1” when executing a read-modify-write type instruction the BS bit is "1" read as “1” and wrote as "1" in the BS bit again.

When the BS bit was cleared automatically by the starting of a PWM cycle in between read and write, "1" is set after BS bit has cleared again.

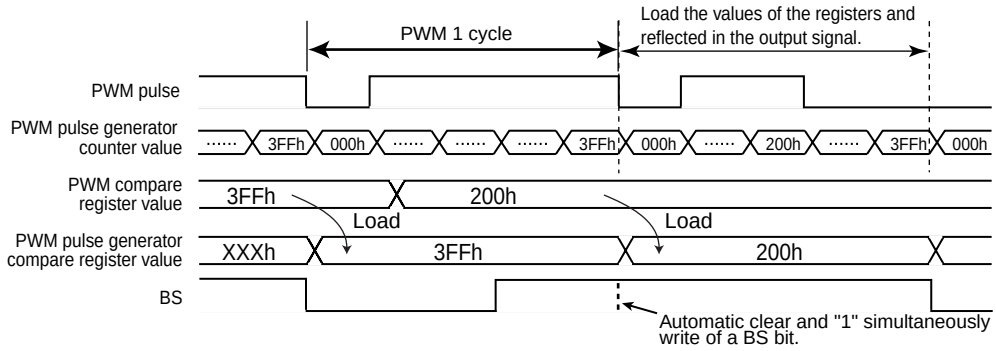
Therefore, values of the registers are loaded in the PWM pulse generator and selector when the BS bit is not set to "1" by the end of the next PWM cycle.

Figure 2-2 load timing of PWM compare register value

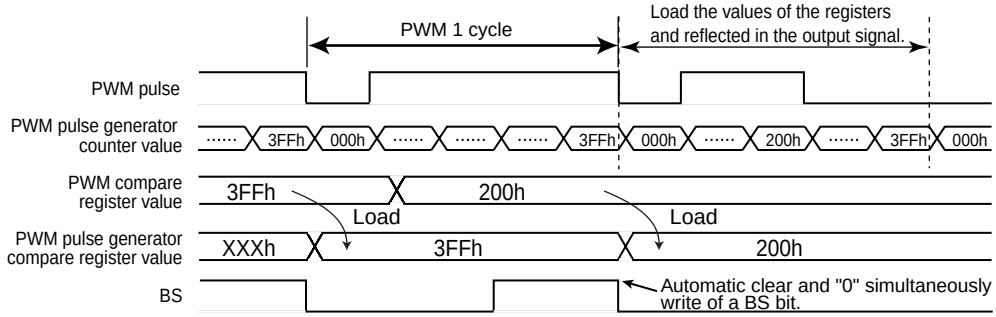
[Automatic clear of BS bit]



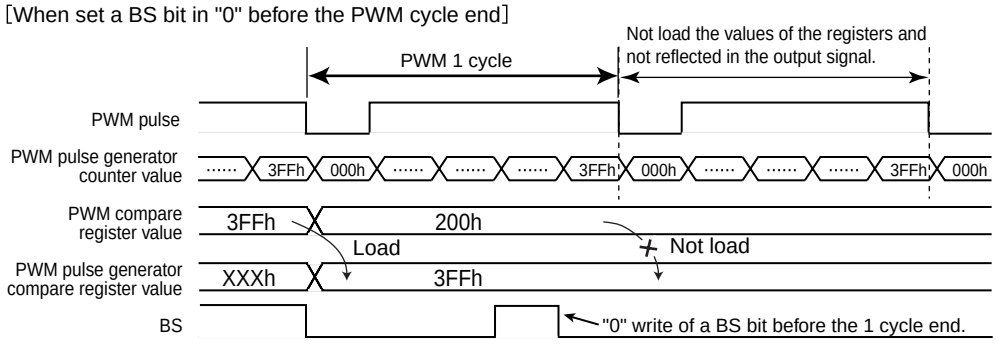
[When set a BS bit in "1" with automatic clear simultaneously]



[When set a BS bit in "0" with automatic clear simultaneously]



[When set a BS bit in "0" before the PWM cycle end]



[bit 13 to 11] P2 to P0: Output select bits

These bits are used to select the output signal for SMC2P.

[bit 10 to 8] M2 to M0: Output select bits

These bits are used to select the output signal for SMC2M.

[bit 7 to 6] Reserved bit

Always set the reserved bit to "0".

[bit 5 to 3] P2 to P0: Output select bits

These bits are used to select the output signal for SMC1P.

[bit 2 to 0] M2 to M0: Output select bits

These bits are used to select the output signal for SMC1M.

The table below shows the relationship between the output levels and the select bits.

P2	P1	P0	PWMmPn	M2	M1	M0	PWMmMn
0	0	0	L	0	0	0	L
0	0	1	H	0	0	1	H
0	1	X	PWM Pulse	0	1	X	PWM Pulse
1	X	X	High impedance	1	X	X	High impedance

m = 1 to 2 (motor coils)

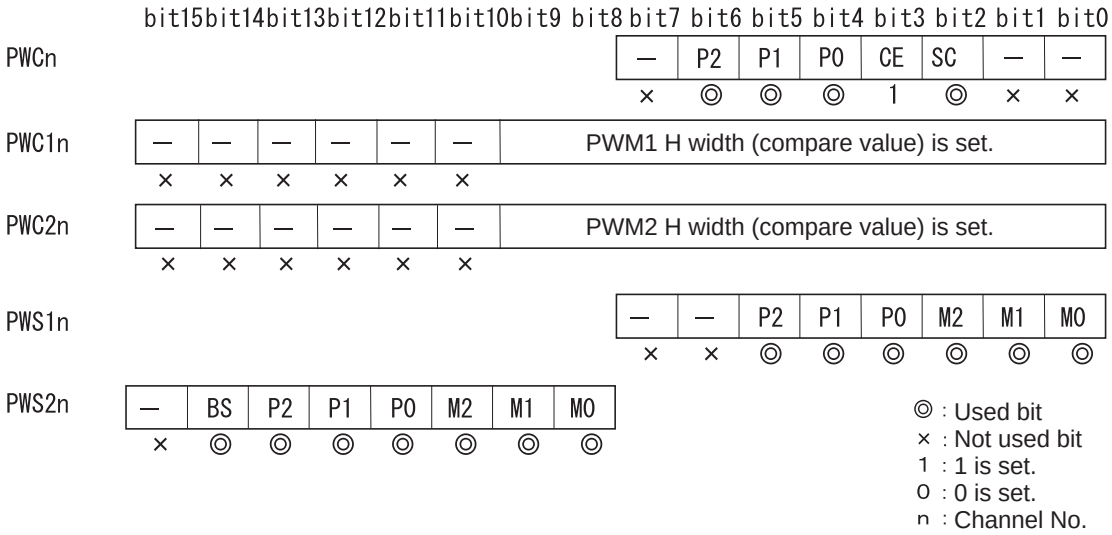
n = 0 to 5 (stepper motor channels)

3. Operation

The operation of the stepping motor controller is explained.

■ Setting Operation of Stepping Motor Controller

Figure 3-1 Setting of Stepping Motor Controller



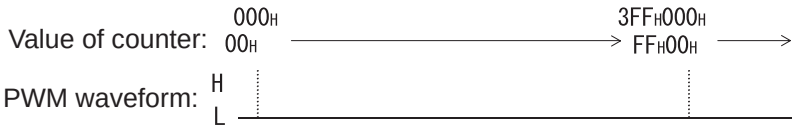
■ Operation of PWM-pulse generator

When the counter is started (PWC: CE = 1), the counter starts incrementing from 00H on the selected count clock rising. The PWM output pulse wave remains "H" until the value of the counter matches the value set to PWM compare register, and then changes to and remains "L" until the value of the counter overflows (FF_H --> 00_H).

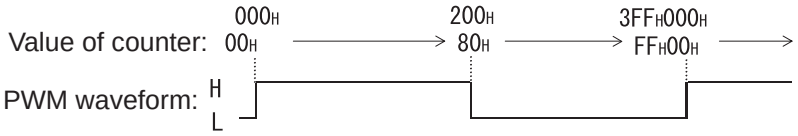
Figure 3-2 "Examples of PWM1&2 Waveform Output" shows the PWM waveform generated by the PWM generator

Figure 3-2 Examples of PWM1&2 Waveform Output

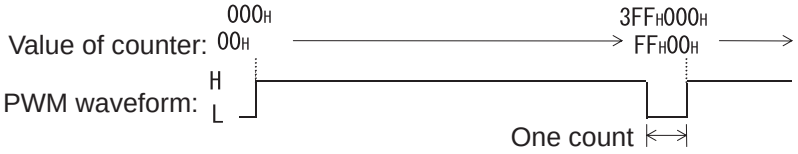
When the value of compare register is "00_H"/"000_H"(duty ratio is 0%):



When the value of compare register is "80_H"/"200_H"(duty ratio is 50%):



When the value of compare register is "FF_H"/"$3FF_H$"(duty ratio is 99.6%/99.9%):



■ Selection of motor drive signals

Motor drive signals that are output to each pin related to the stepping motor controller can be selected among four types of signals for each pin by setting the PWM selection register.

Table 3-1 "Selection of Motor Drive Signals and Setting of PWM Selection Registers 1&2" lists the selection of the motor drive signals and the settings of PWM selection registers 1&2.

When these registers are set and "1" is written to the BS bit of the PWM selection register 2, the setting of these registers is enabled at the end of the current PWM cycle. The BS bit is to be cleared automatically to 0 at the beginning of the next PWM cycle. When "1" is written to the BS bit, and the BS bit is cleared to 0 simultaneously at the beginning of the next PWM cycle, "1" is written to the BS bit and clearing of the BS bit is cancelled.

Table 3-1 Selection of Motor Drive Signals and Setting of PWM Selection Registers 1&2

P2, P1, P0 Bits	PWM2P Output PWM1P Output	M2, M1, M0 Bits	PWM1M Output PWM2M Output
000_B	L	000_B	L
001_B	H	001_B	H
$01X_B$	PWM Pulse	$01X_B$	PWM Pulse
$1XX_B$	High impedance	$1XX_B$	High impedance

4. Caution

The caution when using the stepping motor controller are described below.

■ Caution when Changing PWM Setting

The PWM compare registers 1&2 (PWC1, PWC2) and the PWM selection registers 1&2 (PWS1, PWS2) can be accessed at any time. However, to change setting of the "H" width of PWM or to change the PWM output, "1" must be written to the BS bit of the PWM2 selection register after (or and at the same time) a setting is written to those registers (the PWM compare register 1&2 and the PWM selection register 1&2).

When "1" is set to the BS bit, the new setting is enabled at the end of the current PWM cycle and the BS bit is cleared automatically.

Also, when "1" is written to the BS bit and the BS bit is reset at the end of the PWM cycle simultaneously, "1" is written to the BS and resetting of the BS bit is cancelled.

4.Caution

Chapter 44 A/D Converter

This chapter provides an overview of the A/D converter, describes the register structure and functions, and describes the operation of the A/D converter.

1. Overview of A/D Converter

The A/D converter converts analog input voltage into digital values and provides the following features.

Features of A/D converter:

Conversion time: minimum 3 us per channel.
RC type successive approximation conversion with sample & hold circuit
10-bit or 8-bit resolution
Program section analog input from 32 channels
Single conversion mode: conversion of one selected channel
Scan conversion mode: continuous conversion of multiple channels, programmable for up to 32 channels

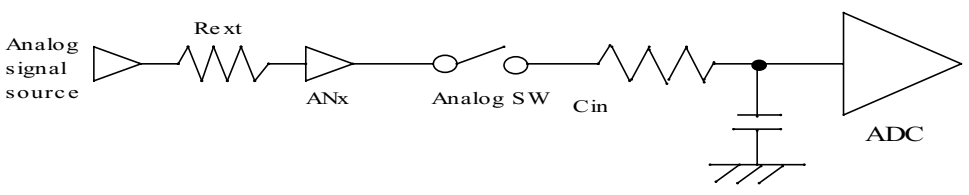
Single conversion mode:	Convert the specified channel only once.
Continuous mode:	Repeatedly convert the specified channels.
Stop mode:	Convert one channel then temporarily halt until the next activation. (Enables synchronization of the conversion start timing.)

A/D conversion can be followed by an A/D conversion interrupt request to CPU. This interrupt, an option that is ideal for continuous processing can be used to start a DMA transfer of the results of A/D conversion to memory.

Startup may be by software, external trigger (falling edge) or timer (rising edge).

■ Input impedance

Sampling circuit of A/D converter is expressed in Figure 1-1 Input Impedance



Don•ft set Rext over maximum sampling time (Tsamp).
Rext = Tsamp/ (7*Cin) - Rin

Figure 1-1 Input Impedance

2. Block Diagram of A/D Converter

Following figure shows block diagram of A/D converter.

Block diagram of A/D converter

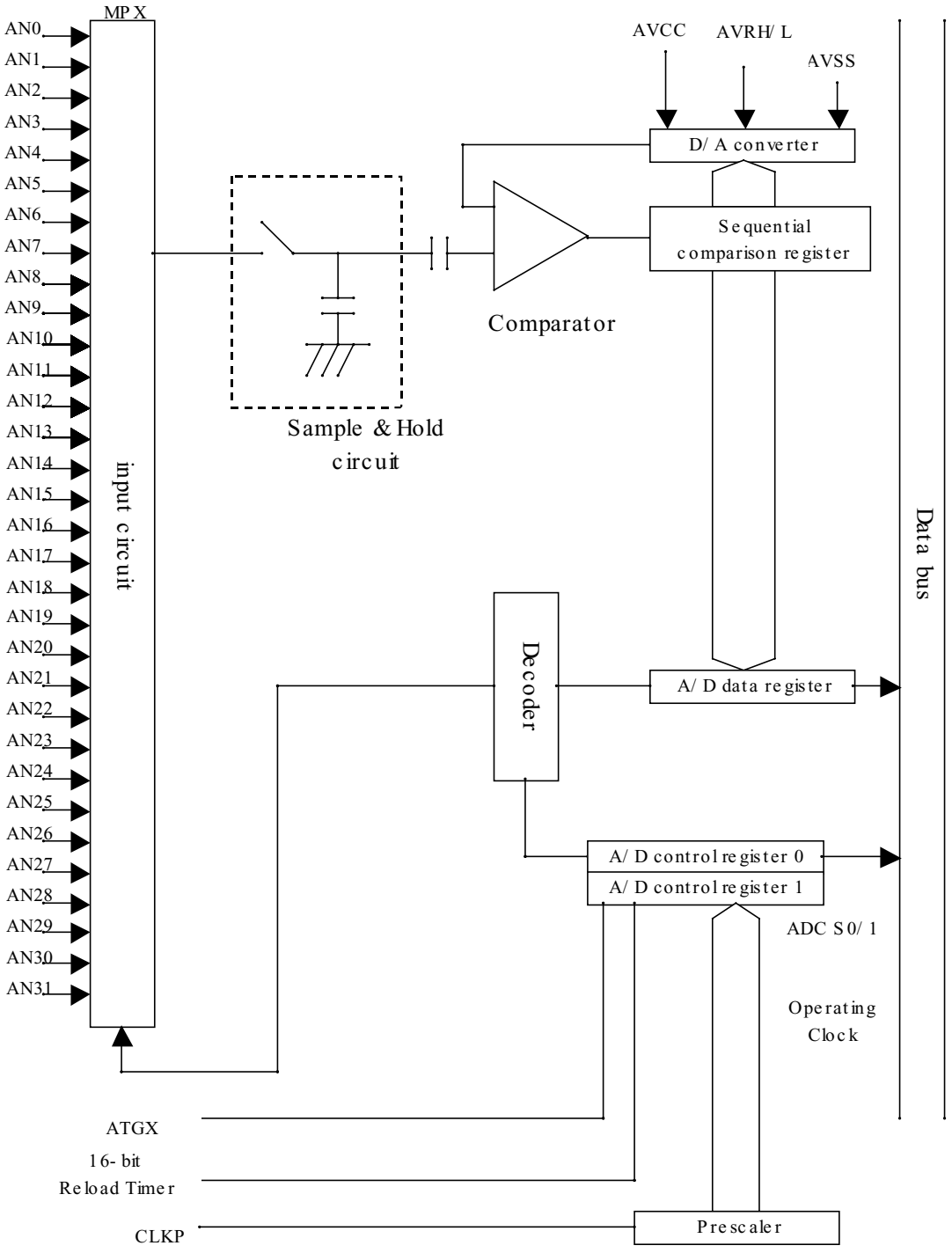


Figure 2-1 Figure 20.2-1 "Block Diagram of A/D Converter"

3. Registers of A/D Converter

The A/D converter has the following registers.

- • A/D enable register (ADER)
- • A/D control status register (ADCS)
- • Data register (ADCR)
- • Sampling timer setting register (ADCT)
- • Start channel setting register (ADSC)
- • End channel setting register (ADEC)

■ Register list

- **ADERH (ADC0): Address 01A0h (Access: Word, Half-word, Byte)**

15	14	13	12	11	10	9	8	Bit
ADE31	ADE30	ADE29	ADE28	ADE27	ADE26	ADE25	ADE24	
0	0	0	0	0	0	0	0	Initial value
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Attribute

7	6	5	4	3	2	1	0	Bit
ADE23	ADE22	ADE21	ADE20	ADE19	ADE18	ADE17	ADE16	
0	0	0	0	0	0	0	0	Initial value
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Attribute

(See “Meaning of Bit Attribute Symbols (Page No.10)” for details of the attributes.)

- **ADERL (ADC0): Address 01A2h (Access: Word, Half-word, Byte)**

15	14	13	12	11	10	9	8	Bit
ADE15	ADE14	ADE13	ADE12	ADE11	ADE10	ADE9	ADE8	
0	0	0	0	0	0	0	0	Initial value
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Attribute

7	6	5	4	3	2	1	0	Bit
ADE7	ADE6	ADE5	ADE4	ADE3	ADE2	ADE1	ADE0	
0	0	0	0	0	0	0	0	Initial value
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Attribute

(See “Meaning of Bit Attribute Symbols (Page No.10)” for details of the attributes.)

- **ADCS1 (ADC0): Address 01A4h (Access: Half-word, Byte)**

15	14	13	12	11	10	9	8	Bit
BUSY	INT	INTE	PAUS	STS1	STS0	STRT	reserved	
0	0	0	0	0	0	0	0	Initial value
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Attribute

(See “Meaning of Bit Attribute Symbols (Page No.10)” for details of the attributes.)

- **ADCS0 (ADC0): Address 01A5h (Access: Half-word, Byte)**

7	6	5	4	3	2	1	0	Bit
MD1	MD0	S10	ACH4	ACH3	ACH2	ACH1	ACH0	
0	0	0	0	0	0	0	0	Initial value
R/W	R/W	R/W	R	R	R	R	R	Attribute

(See “Meaning of Bit Attribute Symbols (Page No.10)” for details of the attributes.)

- **ADCR1 (ADC0): Address 01A6h (Access: Word, Half-word, Byte)**

15	14	13	12	11	10	9	8	Bit
-	-	-	-	-	-	D9	D8	
0	0	0	0	0	0	X	X	Initial value
R0	R0	R0	R0	R0	R0	R	R	Attribute

(See “Meaning of Bit Attribute Symbols (Page No.10)” for details of the attributes.)

- **ADCR0 (ADC0): Address 01A7h (Access: Word, Half-word, Byte)**

7	6	5	4	3	2	1	0	Bit
D7	D6	D5	D4	D3	D2	D1	D0	
X	X	X	X	X	X	X	X	Initial value
R	R	R	R	R	R	R	R	Attribute

(See “Meaning of Bit Attribute Symbols (Page No.10)” for details of the attributes.)

- **ADCT1 (ADC0): Address 01A8h (Access: Word, Half-word, Byte)**

15	14	13	12	11	10	9	8	Bit
CT5	CT4	CT3	CT2	CT1	CT0	ST9	ST8	
0	0	0	1	0	0	0	0	Initial value
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Attribute

(See “Meaning of Bit Attribute Symbols (Page No.10)” for details of the attributes.)

- **ADCT0 (ADC0): Address 01A9h (Access: Word, Half-word, Byte)**

7	6	5	4	3	2	1	0	Bit
ST7	ST6	ST5	ST4	ST3	ST2	ST1	ST0	
0	0	1	0	1	1	0	0	Initial value
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Attribute

(See “Meaning of Bit Attribute Symbols (Page No.10)” for details of the attributes.)

- **ADSCH (ADC0): Address 01AAh (Access: Word, Half-word, Byte)**

15	14	13	12	11	10	9	8	Bit
-	-	-	ANS4	ANS3	ANS2	ANS1	ANS0	
-	-	-	0	0	0	0	0	Initial value
RX, W0	RX, W0	RX, W0	R/W	R/W	R/W	R/W	R/W	Attribute

(See “Meaning of Bit Attribute Symbols (Page No.10)” for details of the attributes.)

- **ADECH (ADC0): Address 01ABh (Access: Word, Half-word, Byte)**

7	6	5	4	3	2	1	0	Bit
-	-	-	ANE4	ANE3	ANE2	ANE1	ANE0	
-	-	-	0	0	0	0	0	Initial value
RX, W0	RX, W0	RX, W0	R/W	R/W	R/W	R/W	R/W	Attribute

(See “Meaning of Bit Attribute Symbols (Page No.10)” for details of the attributes.)

3.1 A/D Enable Register (ADER)

While a pin is used as analog input, corresponding bit in ADER register have to be set to 1.

■ A/D enable register (ADER)

- **ADERH (ADC0): Address 01A0h (Access: Word, Half-word, Byte)**

15	14	13	12	11	10	9	8	Bit
ADE31	ADE30	ADE29	ADE28	ADE27	ADE26	ADE25	ADE24	
0	0	0	0	0	0	0	0	Initial value
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Attribute

7	6	5	4	3	2	1	0	Bit
ADE23	ADE22	ADE21	ADE20	ADE19	ADE18	ADE17	ADE16	
0	0	0	0	0	0	0	0	Initial value
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Attribute

(See “Meaning of Bit Attribute Symbols (Page No.10)” for details of the attributes.)

- **ADERL (ADC0): Address 01A2h (Access: Word, Half-word, Byte)**

15	14	13	12	11	10	9	8	Bit
ADE15	ADE14	ADE13	ADE12	ADE11	ADE10	ADE9	ADE8	
0	0	0	0	0	0	0	0	Initial value
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Attribute

7	6	5	4	3	2	1	0	Bit
ADE7	ADE6	ADE5	ADE4	ADE3	ADE2	ADE1	ADE0	
0	0	0	0	0	0	0	0	Initial value
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Attribute

(See “Meaning of Bit Attribute Symbols (Page No.10)” for details of the attributes.)

[ADE31-0]: A/D Input Enable

ADE	Function
0	General purpose port [Initial value]
1	Analog input

RST clears them to 0.
Be sure to set start channel and end channel to 1.

3.2 A/D Control Status Register (ADCS)

A/D control status register controls and shows the status of A/D converter. Do not overwrite ADCS0 register during A/D converting.

■ A/D control status register 1 (ADCS1)

- **ADCS1 (ADC0): Address 01A4h (Access: Half-word, Byte)**

15	14	13	12	11	10	9	8	Bit
BUSY	INT	INTE	PAUS	STS1	STS0	STRT	reserved	
0	0	0	0	0	0	0	0	Initial value
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Attribute

(See “Meaning of Bit Attribute Symbols (Page No.10)” for details of the attributes.)

[bit 7] BUSY (busy flag and stop)

BUSY	Function
Reading	A/D converter operation indication bit. Set on activation of A/D conversion and cleared on completion.
Writing	Writing "0" to this bit during A/D conversion forcibly terminates conversion. Use to forcibly terminate in continuous and stop modes.

RMW instructions read the bit as "1".
Cleared on the completion of A/D conversion in single conversion mode.
In continuous and stop mode, the flag is not cleared until conversion is terminated by writing "0".
Initialized to "0" by a reset.
Do not specify forcible termination and software activation (BUSY="0" and STRT="1") at the same time.

[bit 6] INT (interrupt)

This bit is set when conversion data is stored in ADCR.
If bit 5 (INTE) is "1" when this bit is set, an interrupt request is generated or, if activation of DMA is enabled, DMA is activated.
Only clear this bit by writing "0" when A/D conversion is halted.
Initialized to "0" by a reset.
If DMA is used, this bit is cleared at the end of DMA transfer.

[bit 5] INTE (Interrupt enable)

This bit is enables or disables the conversion completion interrupt.

INTE	Function
0	Disable interrupt [Initial value]
1	Enable interrupt

Cleared by a reset.

[bit 4] PAUS (A/D converter pause)

This bit is set when A/D conversion temporarily halts.
The A/D converter has only one register to store the conversion result. Therefore, the previous conversion result is lost if it is not transferred by DMA when performing continuous conversion.
To avoid this problem, the next conversion data is not stored in the data register until the previous value has been transferred by DMA. A/D conversion halts during this time. A/D conversion restarts when using DMA.
This bit is only meaningful when using DMA.

Cleared by writing "0" or by a reset. (Not cleared at the end of DMA transfer.) However when waiting condition of DMA transfer, this bit cannot be cleared.
Regarding protect function of converted data, see Section “4. Operation of A/D Converter".

[bit 3, 2] STS1, STS0 (Start source select)

These bits initialized "00" by reset.
These bits select the A/D activation source.

STS1	STS0	Function
0	0	Software activation [Initial value]
0	1	External trigger pin activation and software activation
1	0	Timer activation and software activation
1	1	External trigger pin activation, timer activation and software activation

In multiple-activation modes, the first activation to occur starts A/D conversion.
The activation source changes immediately on writing to the register. Therefore care is required when switching activation mode during A/D operation.
The A/D converter detects falling edges on the external trigger pin. When external trigger level is "L" and if these bits are changed to external trigger activation mode, A/D converting may starts.
Selecting the timer selects the 16-bit reload timer 7.

[bit 1] STRT (Start)

Writing "1" to this bit starts A/D conversion (software activation).
Write "1" again to restart conversion.
Initialized to "0" by a reset.
In continuous and stop mode, restarting is not occurred. Check BUSY bit before writing "1". (Activate conversion after clearing.)
Do not specify forcible termination and software activation (BUSY="0" and STRT="1") at the same time.

[bit 0] reserved bit

Always write "0" to this bit.

■ A/D control status register 0 (ADCS0)

- **ADCS0 (ADC0): Address 01A5h (Access: Half-word, Byte)**

7	6	5	4	3	2	1	0	Bit
MD1	MD0	S10	ACH4	ACH3	ACH2	ACH1	ACH0	
0	0	0	0	0	0	0	0	Initial value
R/W	R/W	R/W	R	R	R	R	R	Attribute

(See “Meaning of Bit Attribute Symbols (Page No.10)” for details of the attributes.)

[bit 7, 6] MD1, MD0 (A/D converter mode set)

These bits the operation mode.

MD1	MD0	Operating mode
0	0	Single mode ; all restarts conversion during operation enabled
0	1	Single mode ; restarts conversion during operation disabled
1	0	Continuous mode ; restarts conversion during operation disabled
1	1	Stop mode ; restarts conversion during operation disabled

Single mode: Continuous A/D conversion from selected channel(s) ANS4 to ANS0 to selected channel(s) ANE4 to ANE0 with a pause after every conversion cycle.

Continuous mode: Repeated A/D conversion cycles from selected channels ANS4 to ANS0 to selected channels ANE4 to ANE0.

Stop mode: A/D conversion for each channel from selected ANS4 to ANS0 to selected channels ANE4 to ANE0, followed by a pause. Restart is determined by the occurrence of a start source.

When A/D conversion is started in continuous mode or stop mode, conversion operation continued until stopped by the BUSY bit.
Conversion is stopped by writing "0" to the BUSY bit.
On activation after forcibly stopping, conversion starts from channel selected ANS4 to ANS0.
All restarts are disabled for any of the timer, external trigger and software start sources in single, continuous and stop modes.

[bit 5] S10

This bit defines resolution of A/D conversion. If this bit set "0", the resolution is 10-bit. In the other case, resolution is 8-bit and the conversion result is stored to ADCR0.
Initialized to "0" by a reset.

[bit 4 to 0] ACH4-0 (Analog convert select channel)

These bits show current converted channel.

ACH4	ACH3	ACH2	ACH1	ACH0	Converted channel
0	0	0	0	0	AN0
0	0	0	0	1	AN1
0	0	0	1	0	AN2
0	0	0	1	1	AN3
0	0	1	0	0	AN4
0	0	1	0	1	AN5
0	0	1	1	0	AN6
0	0	1	1	1	AN7
0	1	0	0	0	AN8
0	1	0	0	1	AN9
0	1	0	1	0	AN10
0	1	0	1	1	AN11
0	1	1	0	0	AN12
0	1	1	0	1	AN13
0	1	1	1	0	AN14
0	1	1	1	1	AN15
1	0	0	0	0	AN16
1	0	0	0	1	AN17
1	0	0	1	0	AN18
1	0	0	1	1	AN19
1	0	1	0	0	AN20
1	0	1	0	1	AN21
1	0	1	1	0	AN22
1	0	1	1	1	AN23
1	1	0	0	0	AN24
1	1	0	0	1	AN25
1	1	0	1	0	AN26
1	1	0	1	1	AN27
1	1	1	0	0	AN28
1	1	1	0	1	AN29
1	1	1	1	0	AN30
1	1	1	1	1	AN31

ACH	Function
Reading	Current converted channel is shown during A/D converting (BUSY="1"). If conversion is halt by forcibly stopping, they shows the stopped channel.
Writing	No effect to these bits.

Initialized to "0000" by reset.

3.3 Data Register (ADCR1, ADCR0)

These registers store the conversion results of the A/D converter. ADCR0 stores lower 8-bit. ADCR1 stores upper

2-bit. The register values are updated at the completion of each conversion. The registers normally store the results of the previous conversion.

■ Data register (ADCR1, ADCR0)

- **ADCR1 (ADC0): Address 01A6h (Access: Word, Half-word, Byte)**

15	14	13	12	11	10	9	8	Bit
-	-	-	-	-	-	D9	D8	
-	-	-	-	-	-	X	X	Initial value
R0, W0	R0, W0	R0, W0	R0, W0	R0, W0	R0, W0	R	R	Attribute

(See “Meaning of Bit Attribute Symbols (Page No.10)” for details of the attributes.)

- **ADCR0 (ADC0): Address 01A7h (Access: Word, Half-word, Byte)**

7	6	5	4	3	2	1	0	Bit
D7	D6	D5	D4	D3	D2	D1	D0	
X	X	X	X	X	X	X	X	Initial value
R	R	R	R	R	R	R	R	Attribute

(See “Meaning of Bit Attribute Symbols (Page No.10)” for details of the attributes.)

Bit 15 to 10 of ADCR1 are read as "0".
The A/D converter has a conversion data protection function. See the "Operation" section for further information.

3.4 Sampling Timer Setting Register (ADCT)

ADCT register controls the sampling time and comparison time of analog input. This register sets A/D conversion time. Do not update value of this register during A/D conversion operation.

■ Sampling timer setting register (ADCT)

- **ADCT1 (ADC0): Address 01A8h (Access: Word, Half-word, Byte)**

15	14	13	12	11	10	9	8	Bit
CT5	CT4	CT3	CT2	CT1	CT0	ST9	ST8	
0	0	0	1	0	0	0	0	Initial value
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Attribute

(See “Meaning of Bit Attribute Symbols (Page No.10)” for details of the attributes.)

- **ADCT0 (ADC0): Address 01A9h (Access: Word, Half-word, Byte)**

7	6	5	4	3	2	1	0	Bit
ST7	ST6	ST5	ST4	ST3	ST2	ST1	ST0	
0	0	1	0	1	1	0	0	Initial value
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Attribute

(See “Meaning of Bit Attribute Symbols (Page No.10)” for details of the attributes.)

[bit 15 to 10] CT5-0 (A/D comparison time set)

These bits specify clock division of comparison time.
Setting "000001" means one division (=CLKP).
Do not set these bits "000000".
Initialized these bits to "000100" by reset.

Comparison time = CT value * CLKP cycle * 10 + (4 * CLKP)

Remarks : Do not set comparison time over 500 us.

[bit 9 to 0] ST9-0 (Analog input sampling time set)

These bits specify sampling time of analog input.
Initialized these bits to "0000101100" by reset.

Sampling time = ST value * CLKP cycle

Remarks : Do not set sampling time below 1.2 us when AVCC is below 4.5 V.

Necessary sampling time and ST value are calculated by following.
Necessary sampling time (Tsamp) = (Rext + Rin) * Cin * 7
ST9 to ST0 = Tsamp / CLKP cycle

ST has to be set that sampling time is over Tsamp.
ex. CLKP = 32MHz, AVCC >= 4.5V, Rext = 200K
Tsamp = (200 * 103 + 2.52 * 103) * 10.7 * 10-12 * 7 = 15.17 [us]
ST = 15.17-6 / 31.25-9 = 485.44
ST has to be set over 486_D (111100110_B).

Tsamp is decided by Rext. Thus conversion time should be considered together with Rext.

3.5 A/D Channel Setting Register (ADSCH, ADECH)

These registers specify the channels for the A/D converter to convert. Do not update these registers while the A/D converting is operating.

■ A/D channel setting register (ADSCH, ADECH)

- **ADSCH (ADC0): Address 01AAh (Access: Word, Half-word, Byte)**

15	14	13	12	11	10	9	8	Bit
-	-	-	ANS4	ANS3	ANS2	ANS1	ANS0	
-	-	-	0	0	0	0	0	Initial value
RX, W0	RX, W0	RX, W0	R/W	R/W	R/W	R/W	R/W	Attribute

(See “Meaning of Bit Attribute Symbols (Page No.10)” for details of the attributes.)

- **ADECH (ADC0): Address 01ABh (Access: Word, Half-word, Byte)**

7	6	5	4	3	2	1	0	Bit
-	-	-	ANE4	ANE3	ANE2	ANE1	ANE0	
-	-	-	0	0	0	0	0	Initial value
RX, W0	RX, W0	RX, W0	R/W	R/W	R/W	R/W	R/W	Attribute

(See "Meaning of Bit Attribute Symbols (Page No.10)" for details of the attributes.)

These bits set the start and end channel for A/D converter.
Setting of ANE4 to ANE0 the same channel as in ANS4 to ANS0 specifies conversion for that channel only. (Single conversion)
In continuous or stop mode, conversion is performed up to the channel specified by ANE4 to ANE0. Conversion then starts again from the start channel specified by ANS4 to ANS0.
If ANS > ANE, conversion starts with the channel specified by ANS, continuous up to channel 31, starts again from channel 0, and ends with the channel specified by ANE.
Initialized to ANS="00000", ANE="00000" by a reset.

ex. Channel Setting ANS=30ch, ANE=3ch, single conversion mode
Operation : Conversion channel 30ch -> 31ch -> 0ch -> 1ch -> 2ch -> 3ch end

[bit 12 to 8] ANS4-0 (Analog start channel set)

[bit 4 to 0] ANE4-0 (Analog end channel set)

ANS4 ANE4	ANS3 ANE3	ANS2 ANE2	ANS1 ANE1	ANS0 ANE0	Start / End Channel
0	0	0	0	0	AN0
0	0	0	0	1	AN1
0	0	0	1	0	AN2
0	0	0	1	1	AN3
0	0	1	0	0	AN4
0	0	1	0	1	AN5
0	0	1	1	0	AN6
0	0	1	1	1	AN7
0	1	0	0	0	AN8
0	1	0	0	1	AN9
0	1	0	1	0	AN10
0	1	0	1	1	AN11
0	1	1	0	0	AN12
0	1	1	0	1	AN13
0	1	1	1	0	AN14
0	1	1	1	1	AN15
1	0	0	0	0	AN16
1	0	0	0	1	AN17
1	0	0	1	0	AN18
1	0	0	1	1	AN19
1	0	1	0	0	AN20
1	0	1	0	1	AN21
1	0	1	1	0	AN22
1	0	1	1	1	AN23
1	1	0	0	0	AN24
1	1	0	0	1	AN25
1	1	0	1	0	AN26
1	1	0	1	1	AN27
1	1	1	0	0	AN28
1	1	1	0	1	AN29
1	1	1	1	0	AN30
1	1	1	1	1	AN31

4. Operation of A/D Converter

The A/D converter operates using the successive approximation method with 10-bit or 8-bit resolution. As only one 16-bit register is provided to store conversion results, the conversion data register (ADCR0 and ADCR1) is updated each time conversion completes. Therefore, as the A/D converter on its own is not suitable for performing continuous conversion, it is recommended that you use the DMA service. The following describes the operation modes.

■ Single Mode

In single conversion mode, the analog input signals selected by the ANS bits and ANE bits are converted in order until the completion of conversion on the end channel determined by the ANE bits. A/D conversion then ends. If the start channel and end channel are the same (ANS=ANE), only a single channel conversion is performed.

ex.

ANS=00000b, ANE=00011b
Start -> AN0 -> AN1 -> AN2 -> AN3 -> End

ANS=00010b, ANE=00010b
Start -> AN2 -> End

■ Continuous Mode

In continuous mode the analog input signals selected by the ANS bits and ANE bits are converted in order until the completion of conversion on the end channel determined by the ANE bits, then the converter returns to the ANS channel for analog input and repeats the process continuously. When the start and end channels are the same (ANS=ANE), conversion is performed continuously for that channel.

ex.

ANS=00000b, ANE=00011b
Start -> AN0 -> AN1 -> AN2 -> AN3 -> AN0 ... -> repeat

ANS=00010b, ANE=00010b
Start -> AN2 -> AN2 -> AN2 ... -> repeat

In continuous mode, conversion is repeated until '0' is written to the BUSY bit. (Writing '0' to the BUSY bit forcibly stops the conversion operation.) Note that forcibly terminating operation halts the current conversion during mid-conversion. (If operation is forcibly terminated, the value in the conversion register is the result of the most recently completed conversion.)

■ Stop Mode

In stop mode the analog input signal selected by the ANS bits and ANE bits are converted in order, but conversion operation pauses for each channel. The pause is released by applying another start signal.

At the completion of conversion on the end channel determined by the ANE bits, the converter returns to the ANS channel for analog input signal and repeats the conversion process continuously. When the start and end channel are the same (ANS=ANE), only a signal channel conversion is performed.

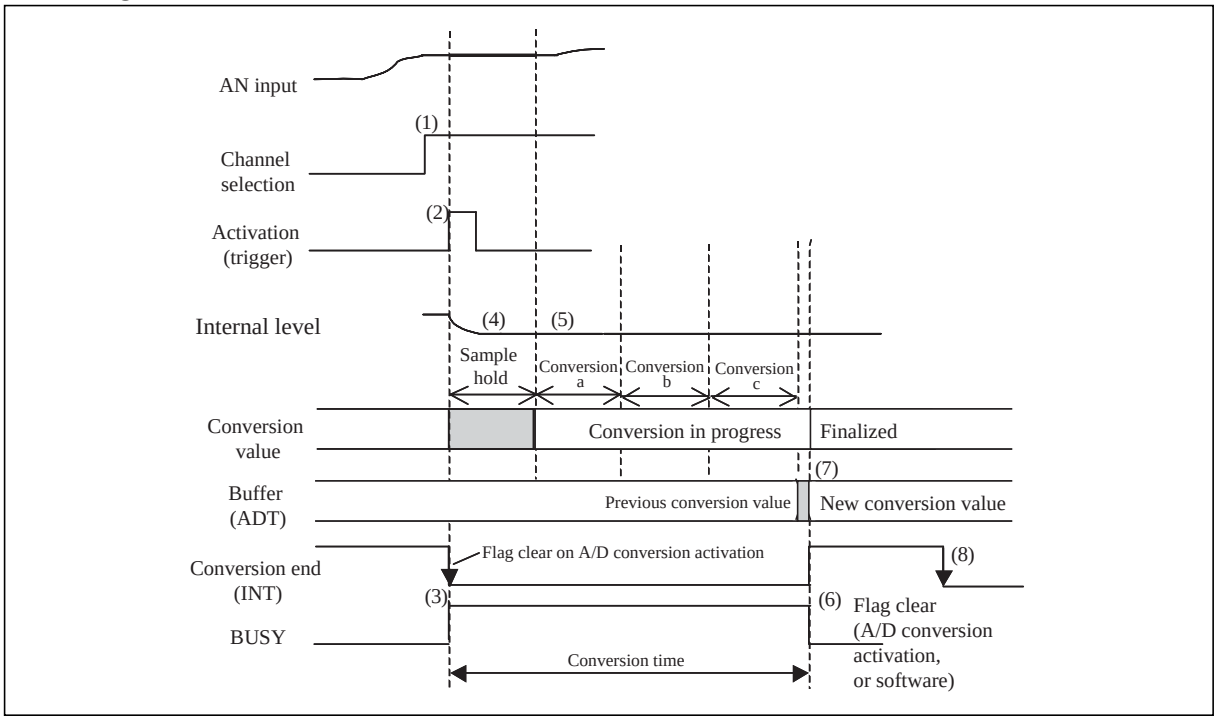
ex.

ANS=00000b, ANE=00011b
Start -> AN0 -> stop -> start -> AN1 -> stop -> start -> AN2 -> stop -> start -> AN3 -> stop -> start -> AN0 ... -> repeat

ANS=00010b, ANE=00010b
Start -> AN2 -> stop -> start -> AN2 -> stop -> start -> AN2 ... -> repeat

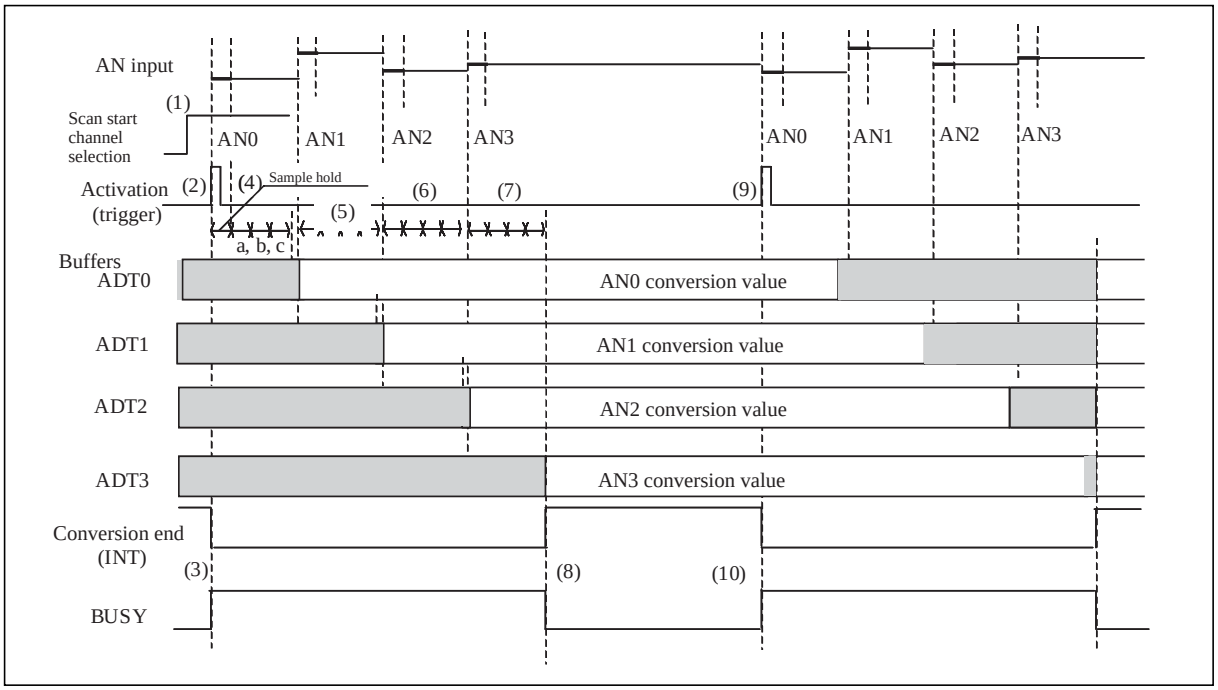
In stop mode the startup source is only the source determined by the STS1, STS0 bits. This mode enables synchronization of the conversion start signal.

4.1 Single-shot conversion mode



(1) Channel selection
(2) A/D conversion activation (Trigger input: Software trigger/Reload timer/External trigger)
(3) INT flag clear, BUSY flag set
(4) Sample hold
(5) Conversion (Conversion a + Conversion b + Conversion c)
(6) Conversion end, INT flag set, BUSY flag clear
(7) Buffers the conversion value. Buffered data storage
(8) Software-based INT flag clear

4.2 Scan conversion mode



(1) Activation channel selection
(2) A/D activation (Trigger: Software trigger/Reload timer/External trigger)
(3) INT flag clear, BUSY flag set
(4) AN0 conversion
a. Sample hold, conversion (conversion a + conversion b + conversion c)
b. Conversion end
c. Buffers the conversion value.
(5) AN1 conversion
(6) AN2 conversion
(7) AN3 conversion
(8) INT flag set, BUSY flag clear
(9) Next A/D activation
(10) INT flag clear, BUSY flag set

5. Setting

Table 5-1 Settings needed to use A/D - Single-Shot Conversion Mode

Setting	Setting Registers	Setting Procedure*
Mode selection (Single-shot conversion)	A/D control (ADCS)	See 7.1
Bit length selection		See 7.2
Channel selection		See 7.3
Conversion time setting	Conversion time setting (ADCT)	See 7.4
To program the AN pin as an input	Port Function (PFR26-PFR29) and Extra Port Function (EPFR26-EPFR27)	See 7.5
A/D activation trigger selection	A/D control (ADCS)	See 7.6
A/D activation trigger generation Software trigger -> Software trigger bit setting		See 7.7
Reload timer -> Reload timer rising output	See “Chapter 38 Reload Timer (Page No.777)”.	
External trigger -> Inputs a trigger to the ADTG(0, 1) pin.	External input	
Conversion end flag check	A/D control (ADCS)	See 7.8
Conversion value read	Data buffers (ADCR)	See 7.9

*: For the setting procedure, refer to the section indicated by the number.

Table 5-2 Settings needed to use A/D - Scan Conversion Mode

Setting	Setting Registers	Setting Procedure*
Mode selection (Scan conversion)	A/D control (ADCS)	See 7.1
Bit length selection		See 7.2
Starting channel selection		See 7.3
Conversion time setting	Conversion time setting (ADCT)	See 7.4
Program the AN pin as an input.	Port Function (PFR26-PFR29) and Extra Port Function (EPFR26-EPFR27)	See 7.5
A/D activation trigger selection	A/D control (ADCS)	See 7.6
A/D activation trigger generation Software trigger -> Software trigger bit setting		See 7.7
Reload timer -> Reload timer falling output	See “Chapter 38 Reload Timer (Page No.777)”.	
External trigger -> Inputs a trigger to the ATGX pin.	External input	
Conversion end flag check	A/D control (ADCS)	See 7.8
Conversion value read	Data buffers (ADCR)	See 7.9

*: For the setting procedure, refer to the section indicated by the number.

Table 5-3 Forcing A/D operations to Stop

Setting	Setting Registers	Setting Procedure*
Forced stop	A/D control (ADCS)	See 7.10

*: For the setting procedure, refer to the section indicated by the number.

Table 5-4 Items needed to enable A/D Interrupts

Setting	Setting Registers	Setting Procedure*
A/D interrupt vector and A/D interrupt level settings	See “Chapter 24 Interrupt Control (Page No.315)”.	See 7.11.
A/D interrupt cause selection (A/D conversion end)	A/D control registers (ADCS)	See 7.12
A/D interrupt setting Clear interrupt requests. Enable interrupt requests.		See 7.13.

*: For the setting procedure, refer to the section indicated by the number.

6. Q & A

6.1 What conversion modes are available and how are they selected?

Two modes of conversion are available:

- Single-shot conversion mode, in which the conversion takes place only once.
- Scan conversion mode, in which a specified sequence of channels are converted.

Mode selection is made using the conversion mode selection bits (ADCS.MD[1:0]).

MD1	MD0	Operating mode
0	0	Single mode ; all restarts conversion during operation enabled
0	1	Single mode ; restarts conversion during operation disabled
1	0	Continuous mode ; restarts conversion during operation disabled
1	1	Stop mode ; restarts conversion during operation disabled

6.2 How do I specify a bit length?

Configure the conversion result storage bit length setting (ADCS.S10).

Operation Mode	Conversion Result Storage Bit Length (S10)
To store conversion results in the ADCR register in 10 bits	Set “0”.
To store conversion results in the ADCR register in 8 bits	Set “1”.

6.3 How do I set a conversion time?

Use Conversion Time Setting registers ADCT to set.

[bit 15 to 10] CT5-0 (A/D comparison time set)

These bits specify clock division of comparison time.
Setting "000001" means one division (=CLKP).
Do not set these bits "000000".
Initialized these bits to "000100" by reset.
Comparison time = CT value * CLKP cycle * 10 + (4 * CLKP)

Remarks : Do not set comparison time over 500 us.

[bit 9 to 0] ST9-0 (Analog input sampling time set)

These bits specify sampling time of analog input.
Initialized these bits to "0000101100" by reset.

Sampling time = ST value * CLKP cycle

Remarks : Do not set sampling time below 1.2 us when AVCC is below 4.5 V.

Necessary sampling time and ST value are calculated by following.
Necessary sampling time (Tsamp) = (Rext + Rin) * Cin * 7
ST9 to ST0 = Tsamp / CLKP cycle

ST has to be set that sampling time is over Tsamp.
ex. CLKP = 32MHz, AVCC >= 4.5V, Rext = 200K
Tsamp = (200 * 103 + 2.52 * 103) * 10.7 * 10-12 * 7 = 15.17 [us]
ST = 15.17-6 / 31.25-9 = 485.44
ST has to be set over 486_D (111100110_B).

Tsamp is decided by Rext. Thus conversion time should be considered together with Rext.

6.4 How do I enable analog pin input?

Use Port Function register PFR and Extra Port Function register EPFR.

Operation	PFR setting	EPFR setting
To program the AN0 pin as an input	PFR29.0 = ‘1’	EPFR29.0 = ‘0’
To program the AN1 pin as an input	PFR29.1 = ‘1’	EPFR29.1 = ‘0’
To program the AN2 pin as an input	PFR29.2 = ‘1’	EPFR29.2 = ‘0’
To program the AN3 pin as an input	PFR29.3 = ‘1’	EPFR29.3 = ‘0’
To program the AN4 pin as an input	PFR29.4 = ‘1’	EPFR29.4 = ‘0’
To program the AN5 pin as an input	PFR29.5 = ‘1’	EPFR29.5 = ‘0’
To program the AN6 pin as an input	PFR29.6 = ‘1’	EPFR29.6 = ‘0’
To program the AN7 pin as an input	PFR29.7 = ‘1’	EPFR29.7 = ‘0’
To program the AN8 pin as an input	PFR28.0 = ‘1’	EPFR28.0 = ‘0’
To program the AN9 pin as an input	PFR28.1 = ‘1’	EPFR28.1 = ‘0’
To program the AN10 pin as an input	PFR28.2 = ‘1’	EPFR28.2 = ‘0’
To program the AN11 pin as an input	PFR28.3 = ‘1’	EPFR28.3 = ‘0’
To program the AN12 pin as an input	PFR28.4 = ‘1’	EPFR28.4 = ‘0’
To program the AN13 pin as an input	PFR28.5 = ‘1’	EPFR28.5 = ‘0’
To program the AN14 pin as an input (*)	PFR28.6 = ‘1’	EPFR28.6 = ‘0’
To program the AN15 pin as an input (*)	PFR28.7 = ‘1’	EPFR28.7 = ‘0’
To program the AN16 pin as an input	PFR27.0 = ‘1’	EPFR27.0 = ‘1’
To program the AN17 pin as an input	PFR27.1 = ‘1’	EPFR27.1 = ‘1’

To program the AN18 pin as an input	PFR27.2 = ‘1’	EPFR27.2 = ‘1’
To program the AN19 pin as an input	PFR27.3 = ‘1’	EPFR27.3 = ‘1’
To program the AN20 pin as an input	PFR27.4 = ‘1’	EPFR27.4 = ‘1’
To program the AN21 pin as an input	PFR27.5 = ‘1’	EPFR27.5 = ‘1’
To program the AN22 pin as an input	PFR27.6 = ‘1’	EPFR27.6 = ‘1’
To program the AN23 pin as an input	PFR27.7 = ‘1’	EPFR27.7 = ‘1’
To program the AN24 pin as an input	PFR26.0 = ‘1’	EPFR26.0 = ‘1’
To program the AN25 pin as an input	PFR26.1 = ‘1’	EPFR26.1 = ‘1’
To program the AN26 pin as an input	PFR26.2 = ‘1’	EPFR26.2 = ‘1’
To program the AN27 pin as an input	PFR26.3 = ‘1’	EPFR26.3 = ‘1’
To program the AN28 pin as an input	PFR26.4 = ‘1’	EPFR26.4 = ‘1’
To program the AN29 pin as an input	PFR26.5 = ‘1’	EPFR26.5 = ‘1’
To program the AN30 pin as an input	PFR26.6 = ‘1’	EPFR26.6 = ‘1’
To program the AN31 pin as an input	PFR26.7 = ‘1’	EPFR26.7 = ‘1’

* Analogue input channels AN14 and AN15 can be also used as D/A converter outputs. In case of exclusive A/D converter usage do not enable the D/A converter output with DACR.DAE[1:0] since this has priority over A/D converter input. See chapter “D/A Converter (Page No.911)” for further information about D/A converter.

6.5 How do I enable analog pin input for Stepper Motor Controller?

Use Port Function register PFR and Extra Port Function register EPFR.

Operation	PFR setting	EPFR setting
To program the AN16 pin as an input for SMC1P0	PFR27.0 = ‘1’	EPFR27.0 = ‘0’
To program the AN17 pin as an input for SMC1M0	PFR27.1 = ‘1’	EPFR27.1 = ‘0’
To program the AN18 pin as an input for SMC2P0	PFR27.2 = ‘1’	EPFR27.2 = ‘0’
To program the AN19 pin as an input for SMC2M0	PFR27.3 = ‘1’	EPFR27.3 = ‘0’
To program the AN20 pin as an input for SMC1P1	PFR27.4 = ‘1’	EPFR27.4 = ‘0’
To program the AN21 pin as an input for SMC1M1	PFR27.5 = ‘1’	EPFR27.5 = ‘0’
To program the AN22 pin as an input for SMC2P1	PFR27.6 = ‘1’	EPFR27.6 = ‘0’
To program the AN23 pin as an input for SMC2M1	PFR27.7 = ‘1’	EPFR27.7 = ‘0’
To program the AN24 pin as an input for SMC1P2	PFR26.0 = ‘1’	EPFR26.0 = ‘0’
To program the AN25 pin as an input for SMC1M2	PFR26.1 = ‘1’	EPFR26.1 = ‘0’
To program the AN26 pin as an input for SMC2P2	PFR26.2 = ‘1’	EPFR26.2 = ‘0’
To program the AN27 pin as an input for SMC2M2	PFR26.3 = ‘1’	EPFR26.3 = ‘0’
To program the AN28 pin as an input for SMC1P3	PFR26.4 = ‘1’	EPFR26.4 = ‘0’
To program the AN29 pin as an input for SMC1M3	PFR26.5 = ‘1’	EPFR26.5 = ‘0’
To program the AN30 pin as an input for SMC2P3	PFR26.6 = ‘1’	EPFR26.6 = ‘0’
To program the AN31 pin as an input for SMC2M3	PFR26.7 = ‘1’	EPFR26.7 = ‘0’

See chapter “Stepper Motor Controller (Page No.873)” about using the A/D converter for SMC operation.

6.6 To select how to activate the A/D converter

There are three types of activation triggers:

- Software trigger
- Reload timer rising signal
- External trigger input falling signal

To set an activation trigger, use Activation Trigger Selection bits (ADCS.STS[1: 0]).

A/D Activation Trigger	Activation Trigger Selection bit (STS[1: 0])

To specify a software trigger	Set “00”.
To specify an external trigger/software trigger	Set “01”.
To specify a reload timer/software trigger	Set “10”.
To specify an external trigger/reload timer/software trigger	Set “11”.

The converter A/D is activated on the first instance of any one of these causes selected.

6.7 To activate the A/D converter

- **Generating a software trigger**

A software trigger is generated using A/D Conversion Software Trigger bits (ADCS.STRT).

Operation	A/D Conversion Software Trigger Bit (STRT)
To generate a software trigger	Write “1”.

- **Activating A/D converter with reload timer 7**

  The reload timers must be setup and activated. For more information, see “Chapter 38 Reload Timer (Page No.777)” When an underflow of a reload timer causes the reload timer output signal to rise, an activation trigger is generated.

- **Activating A/D converter with an external trigger**

  Use external trigger input pin ATGX to generate an external trigger.

  The external trigger input pin is set using Port Function bits (PFR16.7) and Extra Port Function Register (EPFR16.7).

Operation	Setting
To program the ATGX pin as a trigger input	Set PFR16.7 = ‘0’ and DDR16.7 = ‘0’ (port input mode)
	Set PFR16.7 = ‘1’ and EPFR16.7 = ‘1’ (function mode)

6.8 To verify the end of a conversion

There are two ways to verify the end of a conversion, as follows:

- **Checking the A/D Conversion End Interrupt Request bits (ADCS.INT)**

(INT)	Description
If the read value is “0”	No A/D conversion end interrupt request
If the read value is “1”	A/D conversion end interrupt request

- **Checking the Operation Verification bits (ADCS.BUSY)**

(BUSY)	Setting
If the read value is “0”	A/D conversion end (stop)
If the read value is “1”	A/D conversion in progress

6.9 How do I read a conversion value?

The conversion value can be read from Data Buffer register ADCR.

6.10 How do I force an A/D conversion operation to a stop?

Use the Forced Stop bits (ADCS.BUSY)

Operation	Forced Stop Bit (BUSY)
To force an A/D conversion operation to a stop	Write “0”.

The operation of the A/D is unaffected by writing “1” to the Forced Stop bit (BUSY).

6.11 What interrupt registers are used?

A/D interrupt vector, A/D interrupt level setting

The table below summarizes the relationships among the machine cycle, A/D number, interrupt level, and interrupt vector.

For more information about the interrupt level and interrupt vector, see “Chapter 24 Interrupt Control (Page No.315).”

	Interrupt Vector (Default)	Interrupt Level Setting Bit (ICR[4:0])
AD0	#134 Address: 0FFDE4h	Interrupt Level register (ICR59) Address: 047Bh

6.12 What interrupts are available?

A/D Conversion End interrupt only. No interrupt cause selection bit is available.

6.13 How do I enable, disable, clear interrupts?

Interrupt Request Enable flag, Interrupt Request flag

Interrupts are enabled using the Interrupt Request Enable bits (ADCS.INTE).

	Interrupt Request Enable Bit (INTE)
To disable interrupt requests	Set “0”.
To enable interrupt requests	Set “1”.

Interrupt request are cleared using the Interrupt Request bits (ADCS.INT).

	Interrupt Request Bit (INT)
To clear interrupt requests	Write “0” or activate A/D. (See “6.7 To activate the A/D converter (Page No.906)”.)

(See “7. Caution (Page No.908)”.)

7. Caution

Tips on using the A/D converter are summarized as follows:

- Power-on sequence
  Be sure to turn on the MCU power (Vdd*) before turning on the power to the A/D converter (AVcc, AVRH) and applying a voltage to the analog input.

- Input impedance of the analog input pin
  The A/D converter has a built-in sample hold circuit to receive the voltage present on the analog input pin in the sample hold capacitor after the activation of an A/D conversion. Therefore, if the analog input external circuit has a high output impedance, it may happen that analog input voltage fails to get stabilized within the sampling cycle. For this reason, keep the output impedance of the external circuit sufficiently low.
  If the output impedance of the external circuit cannot be kept sufficiently low, lengthen the sampling time fully.

- As AVRH-AV_{SS} decreases the error grows in proportion.

■ Definitions of A/D Converter Terms

- **Resolution**

Analog change identifiable to an A/D converter.

- **Linearity error**

Deviation between the straight line connecting zero transition point (00 0000 0000 <- -> 00 0000 0001) and full-scale transition point (11 1111 1110 <- -> 11 1111 1111) from actual conversion characteristics

- **Differential linearity error**

Deviation of the input voltage required for changing the output by one LSB, from its ideal value

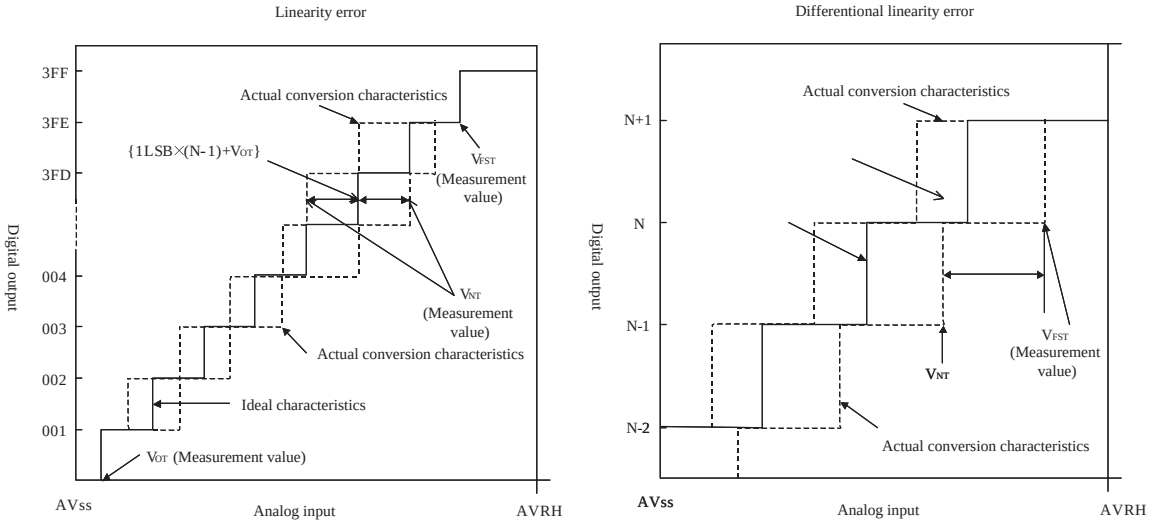
$$1LSB = \frac{V_{FST} - V_{OT}}{1022} \quad [V]$$

V_{OT} : Voltage at which digital output transit from $(000)_H$ to $(001)_H$
V_{FST} : Voltage at which digital output transit from $(3FE)_H$ to $(3FF)_H$

$$\text{Digital output N Linearity error} = \frac{V_{NT} - \{1LSB \times (N-1) + V_{OT}\}}{1LSB} \quad [LSB]$$

$$\text{Digital output N Differential linearity error} = \frac{V_{(N+1)T} - V_{NT}}{1LSB} - 1 \quad [LSB]$$

V_{NT}: Voltage at which digital output transit from (N+1) to N



- **Overall error**

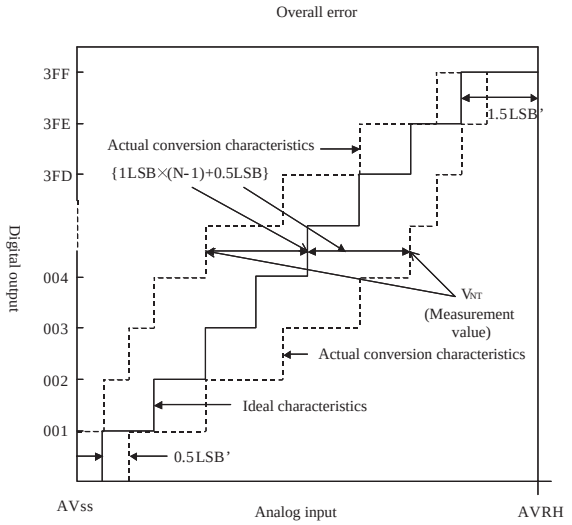
  Difference between an actual value and a theoretical value, containing a zero transition error/full transition error/linearity error

$$1LSB'(\text{Ideal value}) = \frac{AVRH - AV_{SS}}{1024} \quad [V]$$

$$V_{OT}'\ (\text{Ideal value}) = AV_{SS} + 0.5LSB' \quad [V]$$
$$V_{FST}'\ (\text{Ideal value}) = AVRH - 1.5LSB' \quad [V]$$

$$\text{Overall error of digital output N} = \frac{V_{NT} - \{1LSB' \times (N - 1) + 0.5LSB'\}}{1LSB'}$$

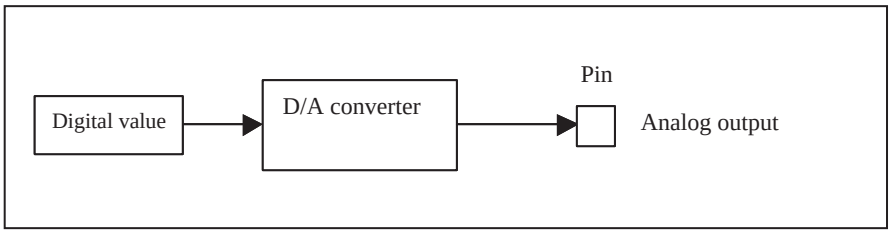
V_{NT}: Voltage at which digital output transit from (N+1) to N



Chapter 45 D/A Converter

1. Overview

The D/A converter converts digital values to analog output values on an R-2R type conversion basis.

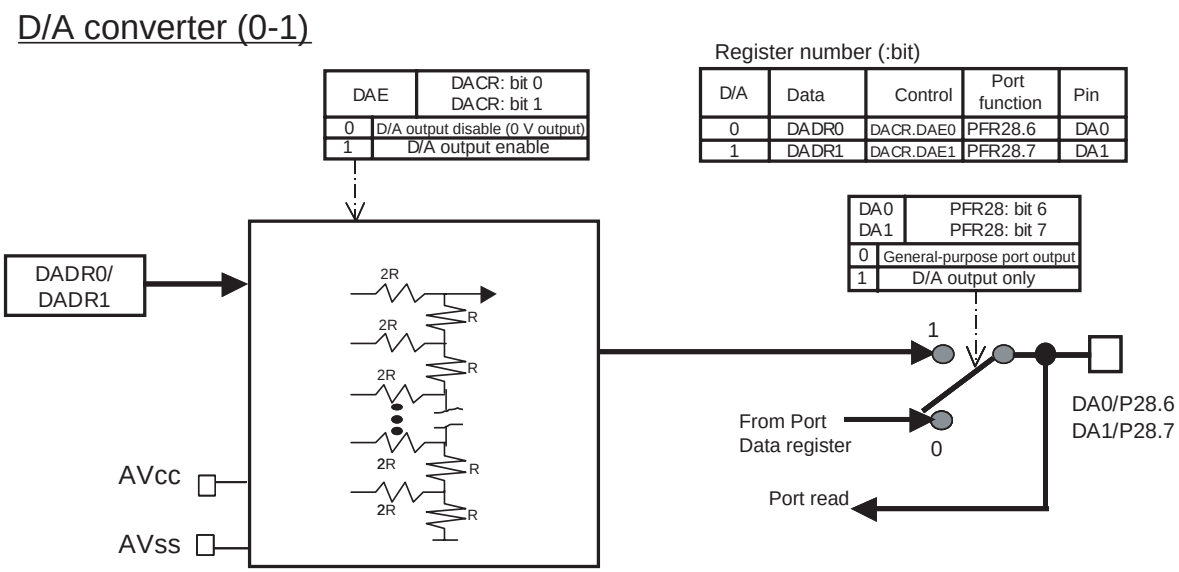


2. Features

Method	: R-2R type conversion
Quantity	: 2 (Output: DA0 pin and DA1 pin)
Conversion time	: 0.45us (Typ) (Load capacitance = 20pF) 2.0us (Typ) (Load capacitance = 100pF)
Resolution	: 10-bit resolution
Output range	: From AV_{SS} (0V) to 1023/1024 x AV_{CC}
Interrupt	: None
Others	: Power-down feature available (fixed 0 V output). Useful for saving current consumption when in sleep mode.

3. Configuration

Figure 3-1 Configuration Diagram



For a detailed description of the D/A pin circuit, see the chapter entitled “Basic Information”.

Figure 3-2 Register List

D/A Converter 0

Address	Bit 7	6	5	4	3	2	1	0	
000364H	---	---	---	---	---	---	DA9	DA8	DADR0 (D/A data 0)
000365H	DA7	DA6	DA5	DA4	DA3	DA2	DA1	DA0	DADR0 (D/A data 0)
000361H	---	---	---	---	---	MD08	DAE1	DAE0	DACR (D/A control)
000D9CH	DA1	DA0	---	---	---	---	---	---	PFR28 (Port function 28)

D/A Converter 1

Address	Bit 7	6	5	4	3	2	1	0	
000366H	---	---	---	---	---	---	DA9	DA8	DADR1 (D/A data 1)
000367H	DA7	DA6	DA5	DA4	DA3	DA2	DA1	DA0	DADR1 (D/A data 1)
000361H	---	---	---	---	---	MD08	DAE1	DAE0	DACR (D/A control)
000D9CH	DA1	DA0	---	---	---	---	---	---	PFR28 (Port function 28)

4. Registers

4.1 DADR: D/A Data Register

The D/A Data Register sets the output voltage of the D/A converter.

- **DADR0(ch0): Address 0364_H (Access: Byte, Half-word)**
- **DADR1(ch1): Address 0366_H (Access: Byte, Half-word)**

15	14	13	12	11	10	9	8	Bit
-	-	-	-	-	-	DA9	DA8	
-	-	-	-	-	-	X	X	Initial value
RX/W0	RX/W0	RX/W0	RX/W0	RX/W0	RX/W0	R/W	R/W	Attributes
7	6	5	4	3	2	1	0	Bit
DA7	DA6	DA5	DA4	DA3	DA2	DA1	DA0	
X	X	X	X	X	X	X	X	Initial value
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Attributes

(For the attributes, refer to the "Meaning of Bit Attribute Symbols (Page No.10)".)

- The D/A Data Register is not initialized on a reset.
- The setting is "000_H" - "$3FF_H$".

4.2 DACR: D/A Control Register

The D/A Control Register controls whether D/A converter output is enabled or disabled.

- **DACR(ch0/ch1): Address 0361_H (Access: Byte)**

7	6	5	4	3	2	1	0	bit
–	–	–	–	–	MD08	DAE1	DAE0	
–	–	–	–	–	0	0	0	Initial value
RX, W0	RX, W0	RX, W0	RX, W0	RX, W0	R/W	R/W	R/W	Attributes

(For the attributes, refer to the "Meaning of Bit Attribute Symbols (Page No.10)".)

- bit7-3: Undefined

  At write, always write "0". At read, the read value is indeterminate.
- bit0: D/A output control

DAE0	Operation
0	D/A output disabled
1	D/A output enabled

  - Enables a converted analog level to be output from the DA pin.
    (To place the DA0 pin in the output state, it is necessary to set PFR28.6="1".
  - The D/A output equals 0.0 V when the D/A output control bit is "0".
- bit1: D/A output control

DAE1	Operation
0	D/A output disabled
1	D/A output enabled

  - Enables a converted analog level to be output from the DA pin.
    (To place the DA1 pin in the output state, it is necessary to set PFR28.7="1".
  - The D/A output equals 0.0 V when the D/A output control bit is "0".

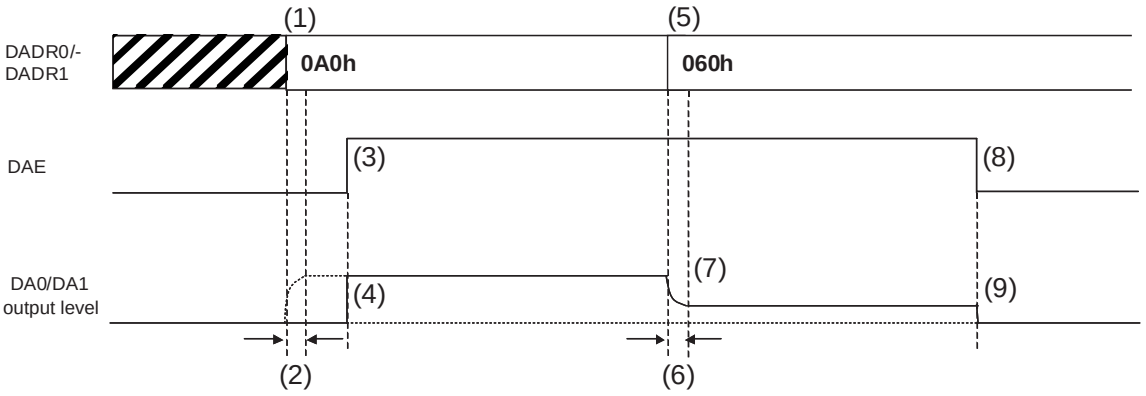
- bit2: D/A 8-/10-bit mode control

MD08	Operation
0	D/A resolution is 10 bits
1	D/A resolution is 8 bits

- In case MD08='1' the 8-bit value of DA7-DA0 (DADR[7:0]) is output.

5. Operation

The operations of the D/A converter are described below.



(1) Digital value setting (software-programmable)
(2) D/A conversion in progress
(3) Output enabled (software-programmable)
(4) Analog value output
(5) Digital value rewrite (software-programmable)
(6) D/A conversion in progress
(7) Output level finalized
(8) Output disabled (software-programmable)
(9) Fixed 0 V output

6. Setting

Table 6-1 Settings Needed to Use D/A

Setting	Setting Registers	Setting Procedure*
Digital value settings	D/A Data Registers (DADR)	See 7.1.
Pin settings	Port Function Register (PFR28.7, PFR28.6)	See 7.2.
Output enabled	D/A Control Registers (DACR)	See 7.3.

*:For the setting procedure, refer to the section indicated by the number.

Table 6-2 Settings Needed to Stop D/A Output

Setting	Setting Registers	Setting Procedure*
Output halted	D/A Control Registers (DACR)	See 7.3.

*:For the setting procedure, refer to the section indicated by the number.

7. Q & A

7.1 Where should I set digital values?

Write digital values to the D/A Data Registers (DADR[7:0] for 8-bit mode, DADR[9:0] for 10-bit mode).
Access in a byte or halfword format.
D/A conversion begins immediately on writing.

7.2 How do I program the D/A pins for D/A output?

DA Pin output setting
Setting is accomplished by writing “1” to the output specification bits (PFR28.7 for DA1), (PFR28.6 for DA0).
(Switch the port to DA pin output by software programming.)

Pins	Control bit location	
DA0 pin	PFR28.6 = ‘1’	DA0 Output specification bit (DA0)
DA1 pin	PFR28.7 = ‘1’	DA1 Output specification bit (DA1)

7.3 How do I enable or disable D/A output?

Use the D/A output control bits (DACR.DAE0), (DACR.DAE1).

Operations	D/A output control bit (DAE)
To disable output	Set “0”.
To enable output	Set “1”.

O V (= AVss) is output when disabled. This is functional even while in a stopped state.

7.4 How do I activate a D/A conversion?

A conversion begins on writing a digital value. See 7.1.

7.5 What is the formula used to work out the value necessary to produce an expected voltage?

Equation

Value = [{V (Expected analog value) x 1024} / (AV_{CC})]

To output 2.8 V from the pin with AV_{CC} = 5.0V, for example.

(2.8V x 1024) / 5.0V = 573.44 => Value = 573

8. Caution

- The table below lists the output voltages of the D/A converter (in 10-bit resolution mode).

DADR Settings	D/A Converter Output Voltage Value
000_H	0V (AVss=0.0V)
001_H	1/1024 x AV_{CC} V
002_H	2/1024 x AV_{CC} V
~	~
$3FD_H$	1021/1024 x AV_{CC} V
$3FE_H$	1022/1024 x AV_{CC} V
$3FF_H$	1023/1024 x AV_{CC} V
When stopped	0V (Avss=0.0V)

- The table below lists the output voltages of the D/A converte (in 8-bit resolution mode)r.

DADR Settings	D/A Converter Output Voltage Value
00_H	0V (AVss=0.0V)
01_H	1/256 x AV_{CC} V
02_H	2/256 x AV_{CC} V
~	~
FD_H	253/256 x AV_{CC} V
FE_H	254/256 x AV_{CC} V
FF_H	255/256 x AV_{CC} V
When stopped	0V (Avss=0.0V)

- The conversion speed depends on the line load capacitance.
  Indicates the conversion speed of the D/A converter.

Load Capacitance	Conversion Speed (TYP)
20pF	0.45µs
100pF	2.0µs

- Power supply
  - The power supply of the analog circuit in the D /A converter is AV_{CC}.
  - Use of a power supply within the limits of “Recommended Power Supply Operation Conditions” in the chapter entitled “2. Instruction for Users (Page No.3)” is recommended.

Chapter 46 Alarm Comparator

1. Overview

This chapter provides an overview of the Alarm Comparator (also called Under/Overvoltage Detection), describes the register structure and functions, and describes the operation of the Alarm Comparator.

2. Block Diagram

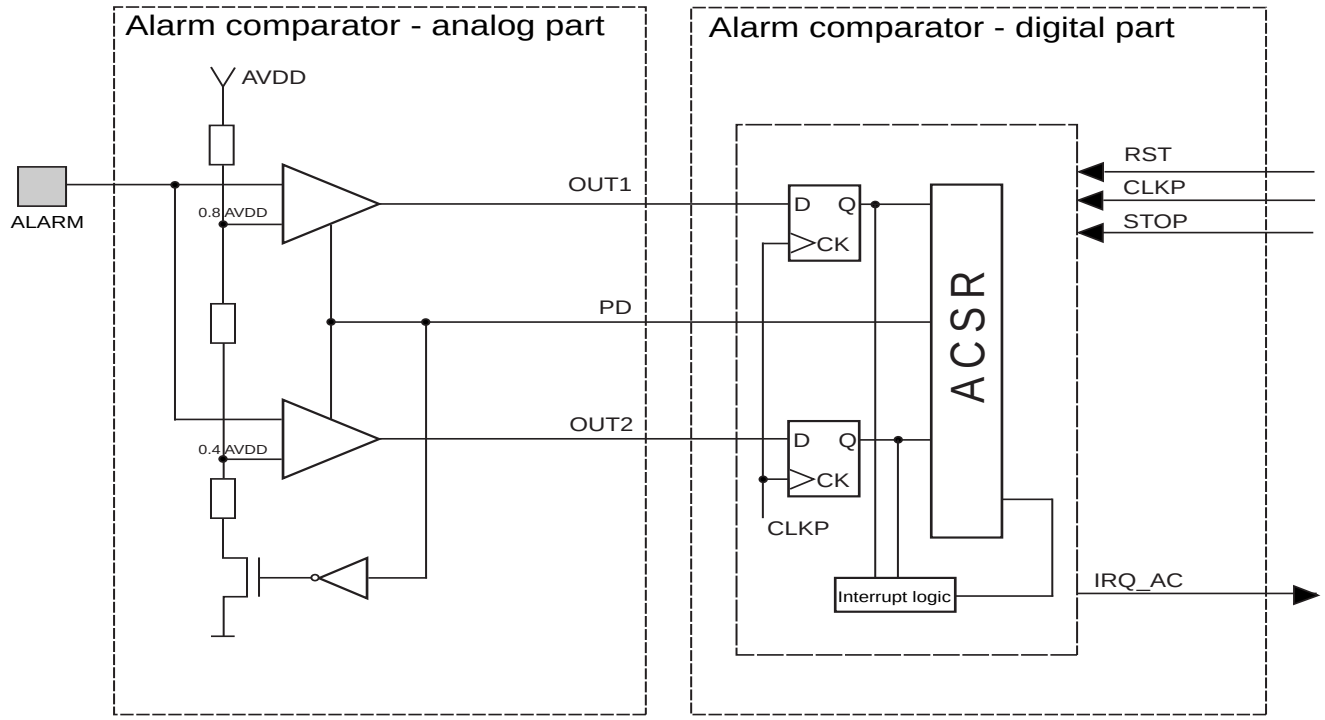


Figure 2-1 Alarmcomparator (simplified circuit)

3. Alarm Comparator Control/Status Register (ACSR)

- **ACSR0 (ch0): Address $01AD_H$ (Access: Byte)**
- **ACSR1 (ch1): Address $01AF_H$ (Access: Byte)**

	Address	Bits 7	6	5	4	3	2	1	0	Initial value
ACSR0	0000 01AD$_H$	MD	OV_EN	UV_EN	OUT2	OUT1	IRQ	IEN	PD	011XXX00$_B$
ACSR1	0000 01AF$_H$	R/W	R/W	R/W	R	R	R/W	R/W	R/W	⇦ Access

Figure 3-1 Structure of Alarm comparator control/status register

Bit 7: MD Mode select

1	Fast mode enabled
0	Slow mode enabled [Initial value]

Bit 6: OV_EN Overvoltage Enable

1	Interrupt enabled in case of overvoltage. [Initial value]
0	No interrupt in case of overvoltage

Bit 5: UV_EN Undervoltage Enable

1	Interrupt enabled in case of undervoltage [Initial value]
0	No interrupt in case of undervoltage.

Bit 4: OUT2 synchronized output of Alarm comparator UV output.

0	analog input voltage < 2/5 AVDD
1	analog input voltage > 2/5 AVDD

Bit 3: OUT1 synchronized output of Alarm comparator OV output.

1	analog input voltage > 4/5 AVDD
0	analog input voltage < 4/5 AVDD

Bit 2: IRQ Interrupt bit.

1	Under- or overvoltage condition detected
0	Normal operation

Bit 1: IEN Interrupt enable bit.

1	Interrupt assertion enabled
0	Interrupt assertion disabled [Initial value]

Bit 0: PD Power down bit.

1	Power down (analog part)
0	Runmode (analog part) [Initial value]

4. Operation Modes

The alarm comparator circuit can operate in interrupt or polling mode. The internal interrupt logic will detect each interrupt event independent from setting of the IEN bit.

4.1 Interrupt Mode (IEN=1)

The following truth table describes the valid interrupt events

Table 4-1 Valid interrupt events

OUT2	OUT1	IRQ	analog input voltage range
1	1	1	Vin > 0.8 AVDD (overvoltage)
1	0	0	0.4 AVDD < Vin < 0.8 AVDD (normal operation)
0	0	1	Vin < 0.4 AVDD (undervoltage)

The interrupt Bit IRQ will be set with the next positive transition of CLKP after detecting an interrupt event. If IEN=1 this will create an interrupt request to the CPU. In order to determine the reason for the asserted interrupt - if both interrupts are enabled - it is necessary to read the ACSR register immediately inside the interrupt service routine. OUT2 and OUT1 always contain the actual status of the comparator outputs, i.e. the interrupt trigger event will not be stored.

4.2 Polling Mode (IEN=0)

The IRQ register bit will be set by an active interrupt event and can be reset by writing to the ACSR register.The ACSR can be polled continuously in order to monitor the input voltage which is feed to the AC comparator inputs.

4.3 Setting and Resetting of IRQ-Flagbit

The IRQ bit of the ACSR register can be reset to zero by writing a “0” to it. Writing an “1” to the IRQ bit of ACSR register has no effect. IRQ can only be set to “1” by hardware, i.e. by the outputs of the comparator circuits. IRQ will remain active as long as an active interrupt status is detected, even if a “0” is written to it.

A bitset command performed on the ACSR register will result in a RMW access on the R-Bus. Every read access during performing a RMW command will return a “1” for the IRQ flag to the CPU. That avoids any loss in detecting interrupt events due to software setting of IRQ-Flag Bit.

4.4 Power Down Modes of the Alarm Comparator

The alarm comparator circuit has the following power down modes:

Table 4-2 Alarm Comparator power down modes

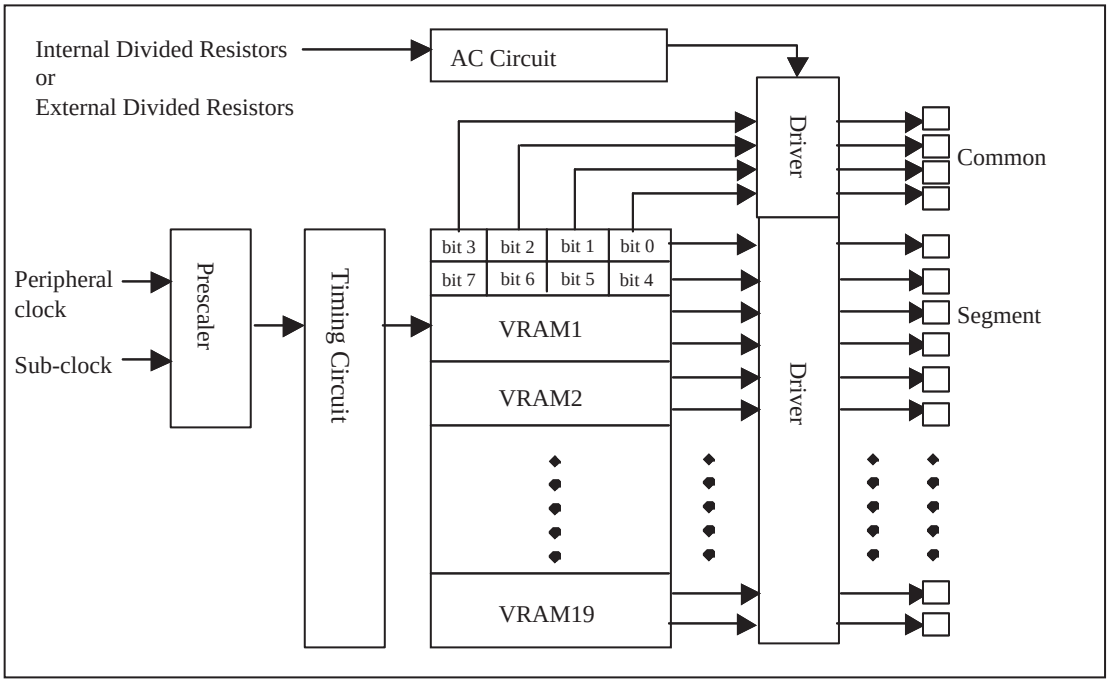
STOP	PD	analog part	digital part
0	0	run mode	run mode
0	1	power down mode	run mode
1	0	power down mode	power down mode
1	1	power down mode	power down mode

Precaution: The outputs of the alarm comparator (analog parts) will remain undefined for at least 3 us after power on and also after reentering the runmode. You have to make sure whether the IRQ is correct set before enabling the alarm comparator interrupt source.

Chapter 47 LCD Controller

1. Overview

LCD allows display of up to 160 cells and selection of a duty cycle from 1/2, 1/3 and 1/4.
LCD has many applications.

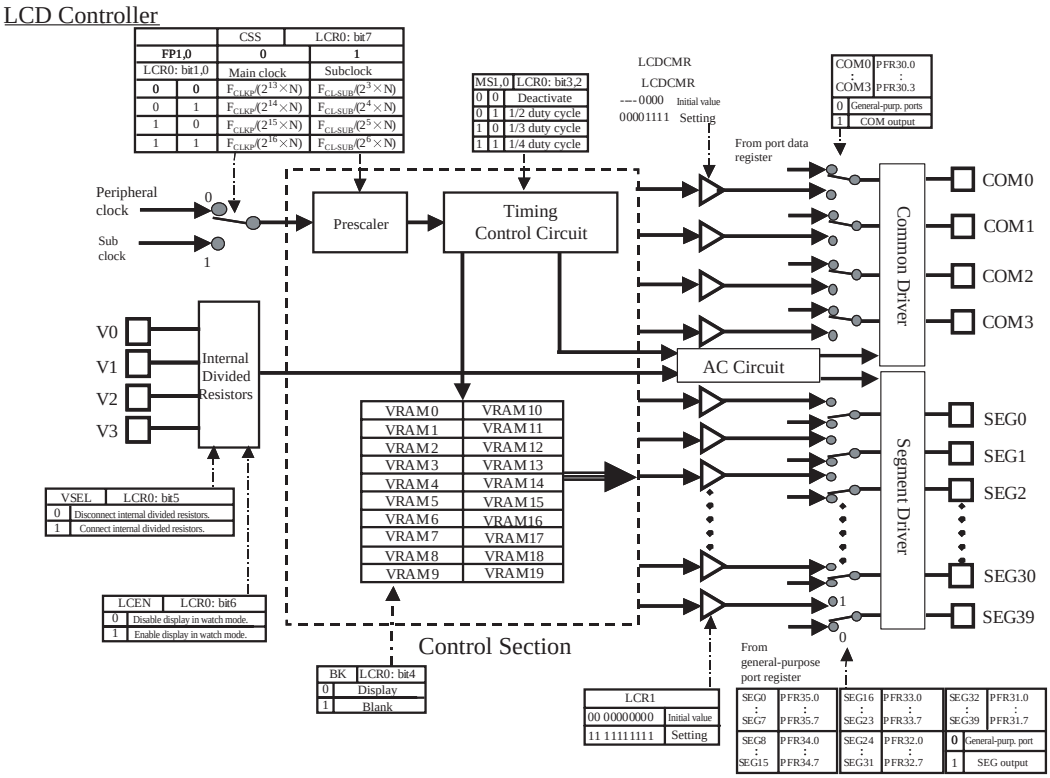


2. Features

- Quantity: 1 (4 common x 40 segment)
- Display: Up to 160 cells (for 1/4 duty cycle)
- Duty cycle: Selectable from options: 1/2, 1/3 and 1/4.
- Bias: Fixed at 1/3
- Frame period: Selectable from four options. (For clock, peripheral clock or subclock is selectable.)
- Driver: Built-in (for internal divided resistors), or external divided resistors can be connected to the V0 - V3 pins.
- Data memory: Built-in 20-byte data memory for display
- Stop mode: Enable LCD display in the sub-stop mode.
- Blank display: Selectable.
- Pin: The SEG0-39 of COM0-4 pin usage can be switched between general and specialized purposes.
- Other: External divided resistors can be also used to shut off the current when LCD is deactivated.

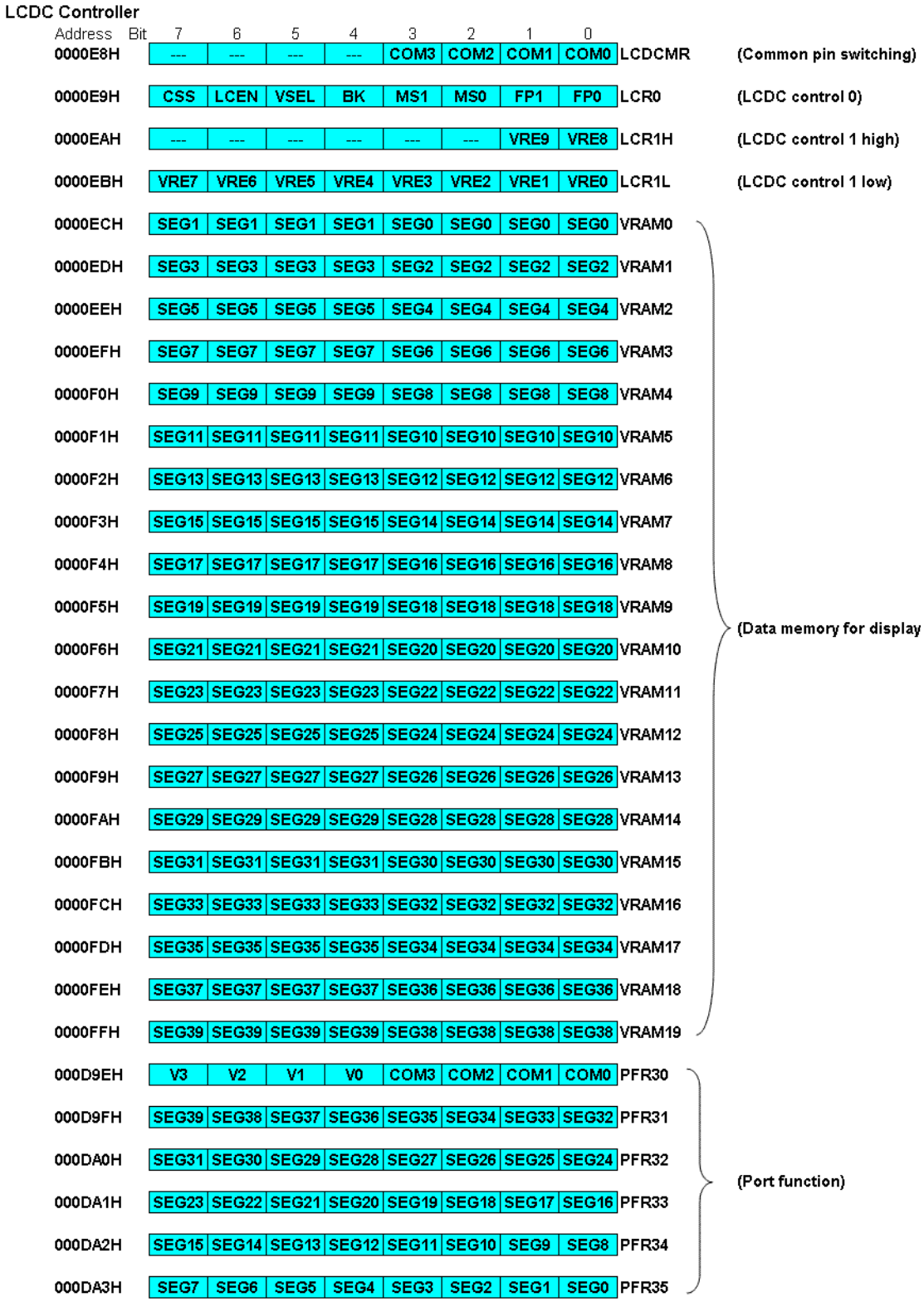
3. Configuration

Figure 3-1 Configuration Diagram



Note: For details on ports, refer to “Chapter 30 I/O Ports (Page No.433)” and “Chapter 3 MB91460 Series Basic Information (Page No.23)”.

Figure 3-2 Register List



4. Registers

4.1 LCR0: LCDC Control Register 0

This register is used to select a frame period and its clock and the display mode, to enable/disable LCD display and the operation in the watch mode, and to control the drive power source.

- **LCR0: Address 0E9$_H$ (Access: Byte)**

7	6	5	4	3	2	1	0	bit
CSS	LCEN	VSEL	BK	MS1	MS0	FP1	FP0	
0	0	0	1	0	0	0	0	Initial value
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Attribute

(For attributes, refer to “Meaning of Bit Attribute Symbols (Page No.10)” chapter.)

- bit7: Select the frame period generation clock

CSS	Operation
0	Peripheral clock (CLKP)
1	Subclock

When the peripheral clock is selected, LCD does not operate if the main clock stops (that is, LCD is in the subclock mode in which the main clock stops, or in the sub-stop mode).

- bit6: Enable operation in sub-stop

LCEN	Operation
0	Disable LCD display in the sub-stop mode.
1	Enable LCD display in the sub-stop mode.

To enable LCD display, the frame period generation clock select bit (CCS) must be also set to “1”.

- bit5: Control LCD drive power supply

VSEL	Operation
0	Disconnect internal divided resistors.
1	Connect internal divided resistors.

To connect external divided resistors, the LCD drive power supply control bit (VSEL) must be set to “0”.

- bit4: Select blanking

BK	Operation
0	Enable LCD display.
1	Disable (blank) LCD display.

- bit3-2: Select a display mode

MS1	MS0	Display mode
0	0	Deactivate LCD.
0	1	1/2 duty cycle output mode (Time division number: N=2, COM0-COM1)
1	0	1/3 duty cycle output mode (Time division number: N=3, COM0-COM2)
1	1	1/4 duty cycle output mode (Time division number: N=4, COM0-COM3)

If the display mode select bit (MS[1:0]) is set to “00”, LCD Controller ceases to operate.
A “L” level is output through common/segment pins.

- bit1-0: Frame period

FP1	FP0	Frame period	
		When peripheral clock is selected:	When subclock is selected:
0	0	$F_{CLKP}/(2^{13} \times N)$	$F_{CL\text{-}SUB}/(2^{3} \times N)$
0	1	$F_{CLKP}/(2^{14} \times N)$	$F_{CL\text{-}SUB}/(2^{4} \times N)$
1	0	$F_{CLKP}/(2^{15} \times N)$	$F_{CL\text{-}SUB}/(2^{5} \times N)$
1	1	$F_{CLKP}/(2^{16} \times N)$	$F_{CL\text{-}SUB}/(2^{6} \times N)$

F_{CLKP} Peripheral clock (CLKP) frequency

$F_{CL\text{-}SUB}$ Subclock frequency

N Time division number (Selected with the display mode select bits, MS1 and MS0.)

Select an appropriate value in accordance with the frame frequency of your LCD panel.

4.2 VRAM: Data Memory for Display

Memory area (VRAM) for setting display data

- **VRAM0 (SEG0, SEG1): Address $0EC_H$ (Access: Byte)**
- **VRAM1 (SEG2, SEG3): Address $0ED_H$ (Access: Byte)**
- **VRAM2 (SEG4, SEG5): Address $0EE_H$ (Access: Byte)**
- **VRAM3 (SEG6, SEG7): Address $0EF_H$ (Access: Byte)**
- **VRAM4 (SEG8, SEG9): Address $0F0_H$ (Access: Byte)**
- **VRAM5 (SEG10, SEG11): Address $0F1_H$ (Access: Byte)**
- **VRAM6 (SEG12, SEG13): Address $0F2_H$ (Access: Byte)**
- **VRAM7 (SEG14, SEG15): Address $0F3_H$ (Access: Byte)**
- **VRAM8 (SEG16, SEG17): Address $0F4_H$ (Access: Byte)**
- **VRAM9 (SEG18, SEG19): Address $0F5_H$ (Access: Byte)**
- **VRAM10 (SEG20, SEG21): Address $0F6_H$ (Access: Byte)**
- **VRAM11 (SEG22, SEG23): Address $0F7_H$ (Access: Byte)**
- **VRAM12 (SEG24, SEG25): Address $0F8_H$ (Access: Byte)**
- **VRAM13 (SEG26, SEG27): Address $0F9_H$ (Access: Byte)**
- **VRAM14 (SEG28, SEG29): Address $0FA_H$ (Access: Byte)**
- **VRAM15 (SEG30, SEG31): Address $0FB_H$ (Access: Byte)**
- **VRAM16 (SEG32, SEG33): Address $0FC_H$ (Access: Byte)**
- **VRAM17 (SEG34, SEG35): Address $0FD_H$ (Access: Byte)**
- **VRAM18 (SEG36, SEG37): Address $0FE_H$ (Access: Byte)**
- **VRAM19 (SEG38, SEG39): Address $0FF_H$ (Access: Byte)**

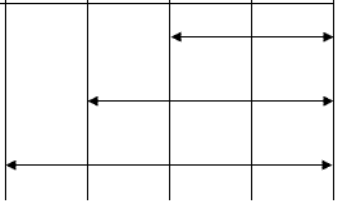
7	6	5	4	3	2	1	0	bit
D07	D06	D05	D04	D03	D02	D01	D00	
X	X	X	X	X	X	X	X	Initial value
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Attribute

(For attributes, refer to “Meaning of Bit Attribute Symbols (Page No.10)”.)

Regardless of the operation of LCD Controller/V driver, RAM can be read and written any time.

- Correspondence between VRAM and Common/Segment Pins

Address						
0x0EC	VRAM0	bit 3	bit 2	bit 1	bit 0	SEG0
		bit 7	bit 6	bit 5	bit 4	SEG1
0x0ED	VRAM1	bit 3	bit 2	bit 1	bit 0	SEG2
		bit 7	bit 6	bit 5	bit 4	SEG3
0x0EE	VRAM2	bit 3	bit 2	bit 1	bit 0	SEG4
		bit 7	bit 6	bit 5	bit 4	SEG5
0x0EF	VRAM3	bit 3	bit 2	bit 1	bit 0	SEG6
		bit 7	bit 6	bit 5	bit 4	SEG7
0x0F0	VRAM4	bit 3	bit 2	bit 1	bit 0	SEG8
		bit 7	bit 6	bit 5	bit 4	SEG9
0x0F1	VRAM5	bit 3	bit 2	bit 1	bit 0	SEG10
		bit 7	bit 6	bit 5	bit 4	SEG11
0x0F2	VRAM6	bit 3	bit 2	bit 1	bit 0	SEG12
		bit 7	bit 6	bit 5	bit 4	SEG13
0x0F3	VRAM7	bit 3	bit 2	bit 1	bit 0	SEG14
		bit 7	bit 6	bit 5	bit 4	SEG15
0x0F4	VRAM8	bit 3	bit 2	bit 1	bit 0	SEG16
		bit 7	bit 6	bit 5	bit 4	SEG17
0x0F5	VRAM9	bit 3	bit 2	bit 1	bit 0	SEG18
		bit 7	bit 6	bit 5	bit 4	SEG19
0x0F6	VRAM10	bit 3	bit 2	bit 1	bit 0	SEG20
		bit 7	bit 6	bit 5	bit 4	SEG21
0x0F7	VRAM11	bit 3	bit 2	bit 1	bit 0	SEG22
		bit 7	bit 6	bit 5	bit 4	SEG23
0x0F8	VRAM12	bit 3	bit 2	bit 1	bit 0	SEG24
		bit 7	bit 6	bit 5	bit 4	SEG25
0x0F9	VRAM13	bit 3	bit 2	bit 1	bit 0	SEG26
		bit 7	bit 6	bit 5	bit 4	SEG27
0x0FA	VRAM14	bit 3	bit 2	bit 1	bit 0	SEG28
		bit 7	bit 6	bit 5	bit 4	SEG29
0x0FB	VRAM15	bit 3	bit 2	bit 1	bit 0	SEG30
		bit 7	bit 6	bit 5	bit 4	SEG31
0x0FC	VRAM16	bit 3	bit 2	bit 1	bit 0	SEG32
		bit 7	bit 6	bit 5	bit 4	SEG33
0x0FD	VRAM17	bit 3	bit 2	bit 1	bit 0	SEG34
		bit 7	bit 6	bit 5	bit 4	SEG35
0x0FE	VRAM18	bit 3	bit 2	bit 1	bit 0	SEG36
		bit 7	bit 6	bit 5	bit 4	SEG37
0x0FF	VRAM19	bit 3	bit 2	bit 1	bit 0	SEG38
		bit 7	bit 6	bit 5	bit 4	SEG39



RAM area and common pins used in 1/2 duty cycle output mode

RAM area and common pins used in 1/3 duty cycle output mode

RAM area and common pins used in 1/4 duty cycle output mode

4.3 LCR1: LCDC Control Register 1

- **LCR1H: Address 0EA$_H$ (Access: Byte, Half-word, Word)**

15	14	13	12	11	10	9	8	bit
-	-	-	-	-	-	SEGEN9	SEGEN8	
-	-	-	-	-	-	0	0	Initial value
-	-	-	-	-	-	R/W	R/W	Attribute

(For attributes, refer to “Meaning of Bit Attribute Symbols (Page No.10)”.)

- bit15-10: Undefined (Read: Indeterminate, Write: “0” is always written.)
- bit9-8: Segment driver enable.
  Always set to “11$_B$” when LCD is used.

- **LCR1L: Address 0EB$_H$ (Access: Byte, Half-word, Word)**

7	6	5	4	3	2	1	0	bit
SEGEN7	SEGEN6	SEGEN5	SEGEN4	SEGEN3	SEGEN2	SEGEN1	SEGEN0	
0	0	0	0	0	0	0	0	Initial value
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Attribute

(For attributes, refer to “Meaning of Bit Attribute Symbols (Page No.10)”.)

- bit7-0: Segment driver enable.
  When LCD is used, always set this register to “11111111$_B$”.

4.4 LCDCMR: Common Pin Switching Register

- **LCDCMR: Address 0E8$_H$ (Access: Byte, Half-word, Word)**

7	6	5	4	3	2	1	0	bit
DTCH	–	–	–	COMEN3	COMEN2	COMEN1	COMEN0	
0	–	–	–	0	0	0	0	Initial value
R/W	RX/WX	RX/WX	RX/WX	R/W	R/W	R/W	R/W	Attribute

(For attributes, refer to “Meaning of Bit Attribute Symbols (Page No.10)”.)

- bit7: Analogue macro control.
  Always set to “0$_B$” when LCD is used.
- bit6-4: Undefined (Read: Indeterminate, Write: “0” is always written.)
- bit3-0: Common driver enable.
  Always set to “1111$_B$” when LCD is used.

5. Operation

This section describes operation.

5.1 LCD Controller/Driver (LCDC) Operation

(1) Set values to the display data memory (VRAM) in advance.

(2) Make necessary settings to each register.

(3) When the frame period generation clock oscillates, LCD drive waveform is output through common/ segment output pins (COM0 - COM3, SEG0 - SEG39).

(4) More detailedly,

VRAM contents are automatically read in synchronization with common signals to be output through segment output pins.

(If the bit is set to “1”, the selected waveform is output through the segment output pins.
If the bit is set to “0”, non-selected waveform is output through the segment output pins.)

If the display mode is set to 1/2 duty cycle, non-selected waveform is output through the COM2 and COM3 pins. For the 1/3 duty cycle, the COM3 pin is used to output non-selected waveform.

(5) This output waveform is a 2-frame AC waveform in accordance with the duty cycle setting, and drives LCD.

(6) When MS[1:0] = “00” is used to deactivate LCD, a “L” level is output through both common and segment pins.

(7) If LCD operation is enabled in the sub-stop mode (LCEN=“1”), LCD display is displayed.

Note that frame period generation clock signals must be supplied at this time.

(8) LCD display can be blanked by selecting “blank” (BK=“1”) in blanking selection.

Note that non-selected waveform continues to be output.

(9) When LCD deactivation (MS[1:0]=“00”) is selected with the display mode, LCD ceases to operate.

5.2 1/2 Duty Cycle Output Waveform

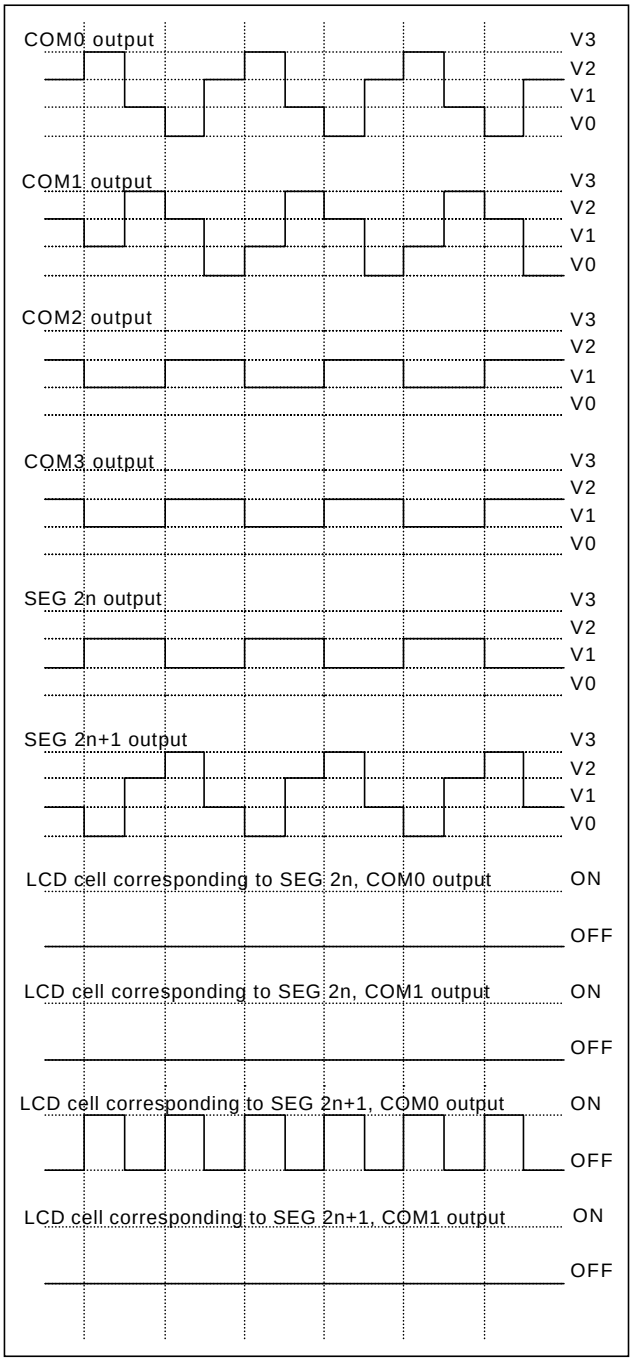
Only COM0 and COM1 outputs are used for LCD display. Neither COM2 nor COM3 output is used.

- Example of 1/3 Bias Output Waveform

LCD cells with the maximum voltage difference between common and segment outputs are lit.

Table 5-1 Example of Data Memory Contents for display

Segment	Contents of data memory for display			
	COM3 output	COM2 output	COM1 output	COM0 output
SEG 2n output	-	-	0	0
SEG2n+1 output	-	-	0	1


COM0 output
V3
V2
V1
V0
COM1 output
V3
V2
V1
V0
COM2 output
V3
V2
V1
V0
COM3 output
V3
V2
V1
V0
SEG 2n output
V3
V2
V1
V0
SEG 2n+1 output
V3
V2
V1
V0
LCD cell corresponding to SEG 2n, COM0 output
ON
OFF
LCD cell corresponding to SEG 2n, COM1 output
ON
OFF
LCD cell corresponding to SEG 2n+1, COM0 output
ON
OFF
LCD cell corresponding to SEG 2n+1, COM1 output
ON
OFF

5.3 1/3 Duty Cycle Output Waveform

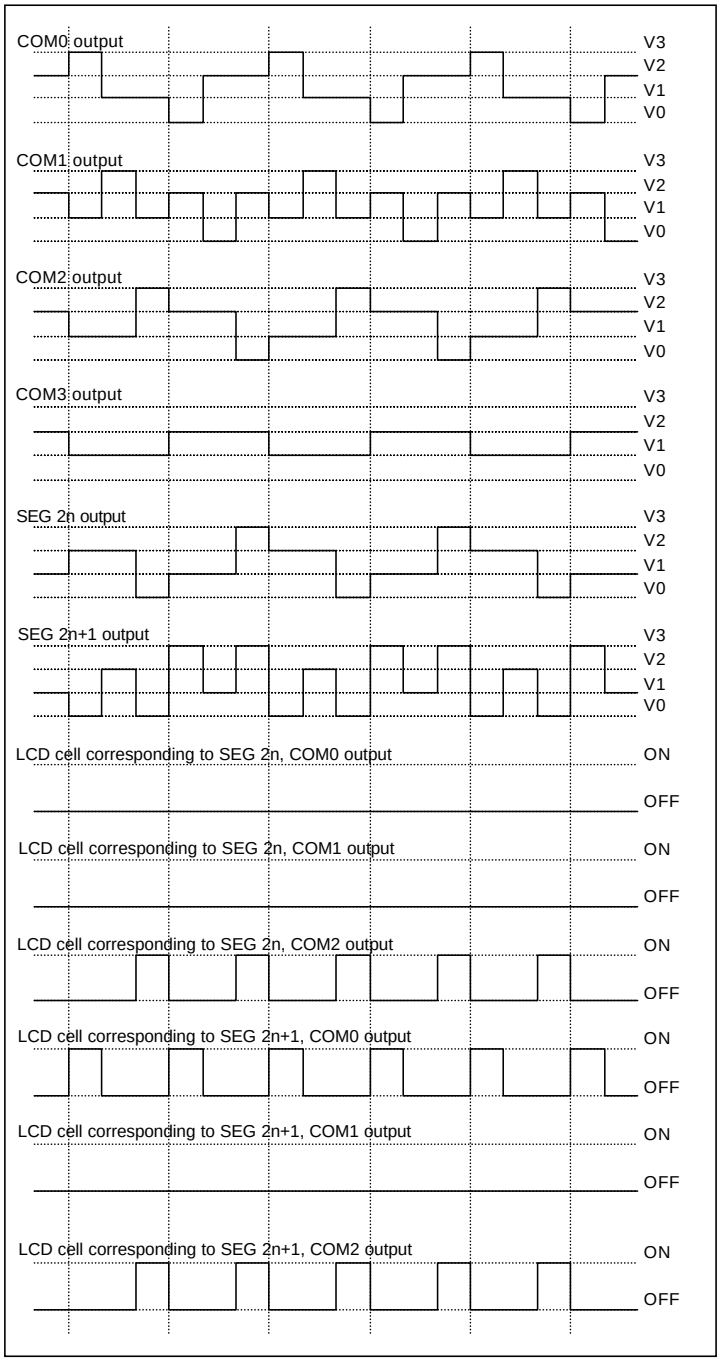
In the 1/3 duty cycle output mode, COM0, COM1 and COM2 outputs are used for LCD display. COM3 output is not used.

- Example of 1/3 Bias Output Waveform

LCD cells with the maximum voltage difference between common and segment outputs are lit.

Table 5-2 Example of Data Memory Contents for Display

Segment	Contents of data memory for display			
	COM3 output	COM2 output	COM1 output	COM0 output
SEG 2n output	-	1	0	0
SEG2n+1 output	-	1	0	1


COM0 output
V3
V2
V1
V0
COM1 output
V3
V2
V1
V0
COM2 output
V3
V2
V1
V0
COM3 output
V3
V2
V1
V0
SEG 2n output
V3
V2
V1
V0
SEG 2n+1 output
V3
V2
V1
V0
LCD cell corresponding to SEG 2n, COM0 output
ON
OFF
LCD cell corresponding to SEG 2n, COM1 output
ON
OFF
LCD cell corresponding to SEG 2n, COM2 output
ON
OFF
LCD cell corresponding to SEG 2n+1, COM0 output
ON
OFF
LCD cell corresponding to SEG 2n+1, COM1 output
ON
OFF
LCD cell corresponding to SEG 2n+1, COM2 output
ON
OFF

5.4 1/4 Duty Cycle Output Waveform

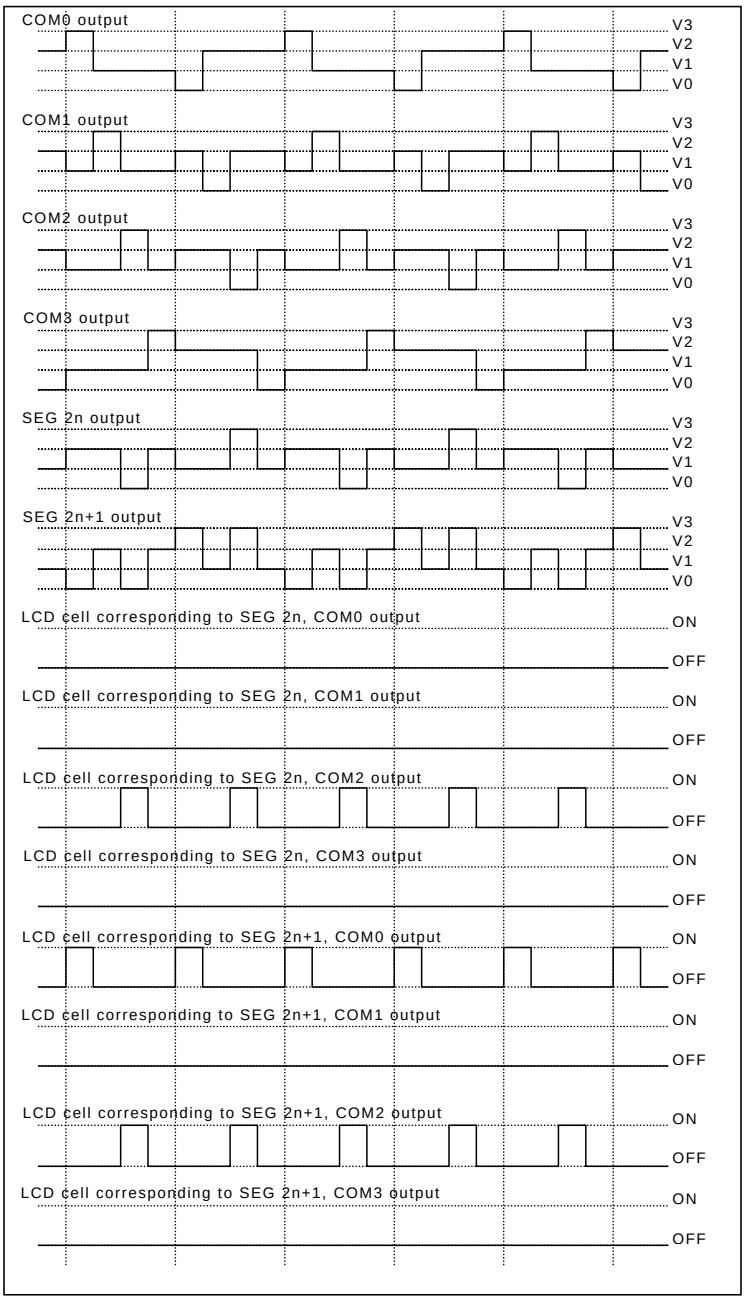
In the 1/4 duty cycle output mode, COM0, COM1, COM2, and COM3 outputs are all used for LCD display.

- Example of 1/3 Bias Output Waveform

LCD cells with the maximum voltage difference between common and segment output are lit.

Table 5-3 Example of Data Memory Contents for Display

Segment	Contents of data memory for display			
	COM3 output	COM2 output	COM1 output	COM0 output
SEG 2n output	0	1	0	0
SEG2n+1 output	0	1	0	1



6. Setting

Table 6-1 Required Setting to Use LCD

Setting	Setting register *	Setting procedure
Presetting	Common pin switching register (LCDCMR) LCD control register 1 (LCR1)	–
Set divided resistors.	LCD control register 0 (LCR0)	See 7.8 and 7.9
Set ports	Port function register (PFR)	See 7.1
Set display data.	Display data memory (VRAM)	See 7.2
Select the frame period generation clock. Set a frame period.	LCD control register 0 (LCR0)	See 7.3
Select a duty cycle. (Activation)		See 7.4
Enable LCD display.		See 7.6

* :For the setting procedure, refer to the section indicated by the number.

Table 6-2 Required Setting to Disable LCD display

Setting	Setting register *	Setting procedure
Disable (blank) LCD display.	LCD control register 0 (LCR0)	See 7.6

* :For the setting procedure, refer to the section indicated by the number.

Table 6-3 Required Setting to Deactivate LCD

Setting	Setting register *	Setting procedure
Deactivate LCD.	LCD control register 0 (LCR0)	See 7.5

*:For the setting procedure, refer to the section indicated by the number.

Table 6-4 Required Setting to Enable LCD Display in Sub-Stop Mode

Setting	Setting register *	Setting procedure
Enable LCD display in the sub-stop mode.	LCD control register 0 (LCR0)	See 7.7
Select the frame period generation clock.		See 7.3
Switch to subclock operation.	See “Chapter 13 Clock Control (Page No.189)”.	–
Change to the stop mode.	See “Chapter 10 Standby (Page No.155)”.	–

* :For the setting procedure, refer to the section indicated by the number.

7. Q&A

7.1 How do I specify pins as COM or SEG output pins?

Use COM and SEG output settings.

Software can switch ports to COM or SEG output ports.

To do so, write “1” to the output designation bit (COM[3:0], SEG[39:0]).

Pin	Register	Output designation bit
COM0	Port function register PFR30[3:0]	(COM0)
COM1		(COM1)
COM2		(COM2)
COM3		(COM3)
SEG0	Port function register PFR35[7:0]	(SEG0)
SEG1		(SEG1)
SEG2		(SEG2)
SEG3		(SEG3)
SEG4		(SEG4)
SEG5		(SEG5)
SEG6		(SEG6)
SEG7		(SEG7)
SEG8	Port function register PFR34[7:0]	(SEG8)
SEG9		(SEG9)
SEG10		(SEG10)
SEG11		(SEG11)
SEG12		(SEG12)
SEG13		(SEG13)
SEG14		(SEG14)
SEG15		(SEG15)
SEG16	Port function register PFR33[7:0]	(SEG16)
SEG17		(SEG17)
SEG18		(SEG18)
SEG19		(SEG19)
SEG20		(SEG20)
SEG21		(SEG21)
SEG22		(SEG22)
SEG23		(SEG23)
SEG24	Port function register PFR32[7:0]	(SEG24)
SEG25		(SEG25)
SEG26		(SEG26)
SEG27		(SEG27)
SEG28		(SEG28)
SEG29		(SEG29)
SEG30		(SEG30)
SEG31		(SEG31)

SEG32	Port function register PFR31[7:0]	(SEG32)
SEG33		(SEG33)
SEG34		(SEG34)
SEG35		(SEG35)
SEG36		(SEG36)
SEG37		(SEG37)
SEG38		(SEG38)
SEG39		(SEG39)

7.2 How do I set VRM?

The following tables show the relationship between pins and the bit positions of VRAM(n). (n=0 to 19)

Table 7-1 1/2 duty cycle

Pin	COM1	COM0
SEG 2n	bit 1	bit 0
SEG 2n+1	Bit 5	Bit 4

Table 7-2 1/3 duty cycle

pin	COM2	COM1	COM0
SEG 2n	bit 2	bit 1	bit 0
SEG 2n+1	Bit 6	Bit 5	Bit 4

Table 7-3 1/4 duty cycle

pin	COM3	COM2	COM1	COM0
SEG 2n	bit 3	bit 2	bit 1	bit 0
SEG 2n+1	bit 7	Bit 6	Bit 5	Bit 4

(Non-selected waveform is output through the pins other than the above.)

Example: 1/4 duty cycle

When “1” is set to the bit6 of VRAMn, selected waveform is output through the SEG2n+1 of COM2.

If a bit is set to “0”, non-selected waveform is output through the corresponding pin.

7.3 How do I set a frame period?

Use the frame period generation clock select bit (LCR0.CSS) and the frame period bit (LCR0.FP[1:0]). The following settings are available:

Frame period	Selected value	
	Frame period generation clock select bit (CSS)	Frame period bit (FP[1:0])
Peripheral clock (F_{CLKP})/($2^{13} \times N$)	Set to “0”.	Set to “00”.
Peripheral clock (F_{CLKP})/($2^{14} \times N$)	Set to “0”.	Set to “01”.
Peripheral clock (F_{CLKP})/($2^{15} \times N$)	Set to “0”.	Set to “10”.
Peripheral clock (F_{CLKP})/($2^{16} \times N$)	Set to “0”.	Set to “11”.
Subclock (F_{CL-SUB})/($2^{3} \times N$)	Set to “1”.	Set to “00”.
Subclock (F_{CL-SUB})/($2^{4} \times N$)	Set to “1”.	Set to “01”.
Subclock (F_{CL-SUB})/($2^{5} \times N$)	Set to “1”.	Set to “10”.
Subclock (F_{CL-SUB})/($2^{6} \times N$)	Set to “1”.	Set to “11”.

N (Time division number) = MS[1:0] value + “1”

Set an appropriate frame period that corresponds to the frame frequency of your LCD panel.

7.4 How do I set a duty cycle?

Use the display mode select bit (LCR0.MS[1:0]).

Controlled operation	Display mode select bit (MS[1:0])	N (Time division number)
To deactivate LCD (Pin output: “L”)	Set to “00”.	N/A
To set the 1/2 duty cycle output mode	Set to “01”.	2
To set the 1/3 duty cycle output mode	Set to “10”.	3
To set the 1/4 duty cycle output mode	Set to “11”.	4

The display mode select bit also serves as an operation start/stop control bit.

7.5 How do I control starting and stopping of LCD?

Use the display mode select bit (LCR0. MS[1:0]) to control start and stop of operation.
See (4).

7.6 How do I enable or disable LCD display?

Use either of the following methods:

- Use the blanking select bit (LCR0. BK).

Controlled operation	Blanking select bit (BK)
To enable LCD display	Set to “0”.
To disable (blank) LCD display (Non-selected waveform is output through segment pins.)	Set to “1”.

- The LCD display can be blanked by using the display mode select bit (LCR0. MS[1:0]) to select LCD deactivation.

Controlled operation	Display mode select bit (MS[1:0])
To deactivate LCD (A “L” level is output through common and segment pins.)	Set to “00”.

7.7 How do I enable LCD display even in the sub-stop mode?

Use the sub-stop operation enable bit (LCR0. LCEN).

Controlled operation	Frame period generation clock select bit (CSS)	Sub-stop operation enable bit (LCEN)
To disable LCD display during sub-stop	–	Set to “0”.
To enable LCD display during sub-stop	Set to “1”.	Set to “1”.
To enable LCD display when the main clock stops and the subclock operates	Set to “1”.	–

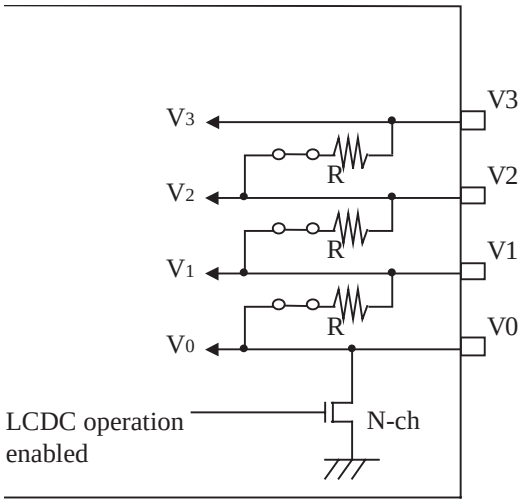
7.8 How do I select internal or external divided resistors?

Use the LCD drive power supply control bit (LCR0. VSEL).

Controlled operation	LCD drive power supply control bit (VSEL).
To use external divided resistors (Internal divided resistors disconnected)	Set to “0”.
To use internal divided resistors (Internal divided resistors connected)	Set to “1”.

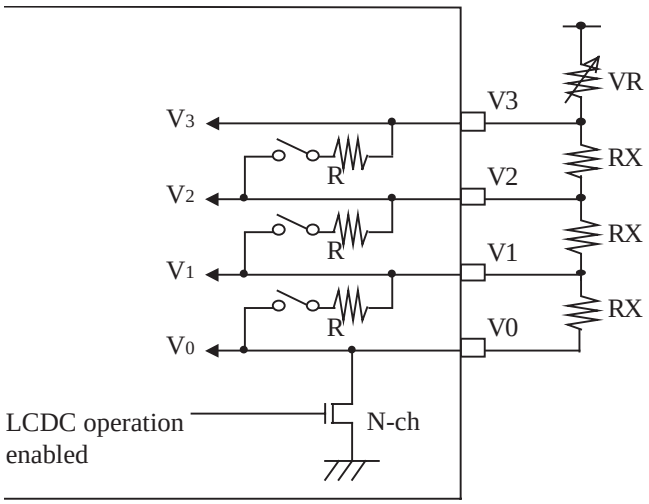
7.9 How do I select internal or external divided resistors?

- When using internal divided resistors:



- When using external divided resistors:

  The LCD driving voltage can be generated by connecting external divided resistors to the LCD drive power supply pins (V0 to V3).



To avoid the effect of internal divided resistors, the resistors must be disconnected by setting “0” to the LCD drive power supply control bit (LCR0.VSEL).

7.10 How do I use external divided resistors to shut off the current when LCD is deactivated?

The V0 pin is internally connected to Vss (GND) via a transistor. For this reason, the current generated on deactivating LCD controller can be shut off by connecting external divided resistors to the V0 pin on the Vss side. To shut off the current, use the display mode select bit (MS[1:0]= “00”).

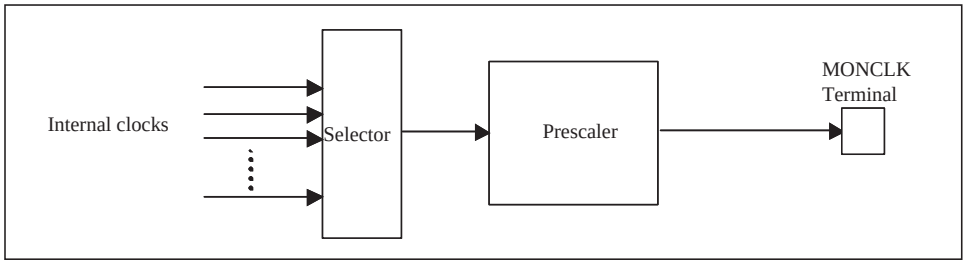
8. Caution

- To access VRAM, be sure to use byte-by-byte access.
- Switching the frame period generation clocks:
  Frame period generation clocks (LCR0:CSS) can be switched even during LCD display. However, switching may cause some screen flicker. To avoid such flicker, be sure to set the blanking select bit (LCR0:BK) to “1” (blank display) before switching.
- Depending on your LCD, different external divided resistors are used. Use appropriate resistor values.
- When the display mode is set to 1/2 duty cycle, non-selected waveform is output through the COM2 and COM3 pins. For 1/3 duty cycle, the COM3 pin is used to output non-selected waveform.
- Inappropriate selection or setting of frame period generation clock (CSS), LCD drive power supply control (VSEL), duty cycle (MS[1:0]) and frame period (FP[1:0]) results in inappropriate LCD display.
- LCD display is disabled in the main stop mode. To enable LCD display in the stop mode, use the sub-stop mode. (See 6 “Setting procedure”.)

Chapter 48 Clock Monitor

1. Overview

The Clock Monitor is a macro that outputs internal clock signals to a terminal to externally monitor them. The Clock Monitor provides a function to divide the frequency of a clock signal before it outputs to the terminal, thus allowing the clock signal to be used as an event at which external circuits act in synchronization with a MCU function.



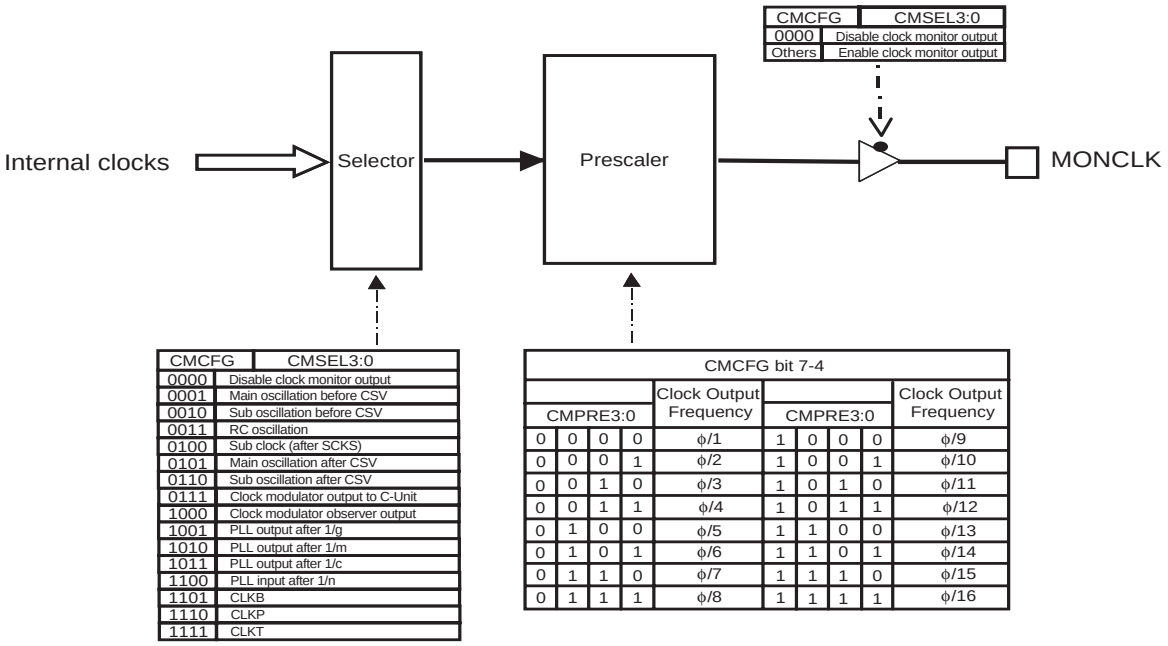
2. Features

- Format: Divide an internal clock signal to output it to a terminal (MONCLK).
- Channel: 1
- Division ratios: CLK/1, CLK/2, CLK/3, ..., CLK/16
- Glitch free output enable
- Programmable mark level (output “L” or “H” before enabling the clock output)
- Interrupt: None

3. Configuration

Figure 3-1 Configuration Diagram

Clock Monitor



CMCFG	CMSEL3:0
0000	Disable clock monitor output
Others	Enable clock monitor output

CMCFG	CMSEL3:0
0000	Disable clock monitor output
0001	Main oscillation before CSV
0010	Sub oscillation before CSV
0011	RC oscillation
0100	Sub clock (after SCKS)
0101	Main oscillation after CSV
0110	Sub oscillation after CSV
0111	Clock modulator output to C-Unit
1000	Clock modulator observer output
1001	PLL output after 1/g
1010	PLL output after 1/m
1011	PLL output after 1/c
1100	PLL input after 1/n
1101	CLKB
1110	CLKP
1111	CLKT

CMCFG bit 7-4									
CMPRE3:0				Clock Output Frequency	CMPRE3:0				Clock Output Frequency
0	0	0	0	ϕ/1	1	0	0	0	ϕ/9
0	0	0	1	ϕ/2	1	0	0	1	ϕ/10
0	0	1	0	ϕ/3	1	0	1	0	ϕ/11
0	0	1	1	ϕ/4	1	0	1	1	ϕ/12
0	1	0	0	ϕ/5	1	1	0	0	ϕ/13
0	1	0	1	ϕ/6	1	1	0	1	ϕ/14
0	1	1	0	ϕ/7	1	1	1	0	ϕ/15
0	1	1	1	ϕ/8	1	1	1	1	ϕ/16

Figure 3-2 Register List

Clock Monitor

Address	Bit 7	6	5	4	3	2	1	0	
0004AEH	EDSUEN	---	---	MONCKI	CSC3	CSC2	CSC1	CSC0	CSCFG (Clock source config)
0004AFH	CMPRE3	CMPRE2	CMPRE1	CMPRE0	CMSEL3	CMSEL2	CMSEL1	CMSEL0	CMCFG (Clock monitor config)

4. Register

4.1 Clock Monitor Configuration Register

A register for output settings of an internal clock signal

- **CMCFG: Address 04AF$_H$ (Access: Byte)**

7	6	5	4	3	2	1	0	bit
CMPRE3	CMPRE2	CMPRE1	CMPRE0	CMSEL3	CMSEL2	CMSEL1	CMSEL0	
0	0	0	0	0	0	0	0	Initial value
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Attributes

(For attributes, refer to "Meaning of Bit Attribute Symbols (Page No.10)".)

- bit7-4: Select an output frequency prescaler

CMPRE3	CMPRE2	CMPRE1	CMPRE0	Clock Frequency Output to the MONCLK pin
0	0	0	0	Source clock (selected by CMSEL) divided by 1 (Initial)
0	0	0	1	Source clock (selected by CMSEL) divided by 2
0	0	1	0	Source clock (selected by CMSEL) divided by 3
0	0	1	1	Source clock (selected by CMSEL) divided by 4
0	1	0	0	Source clock (selected by CMSEL) divided by 5
0	1	0	1	Source clock (selected by CMSEL) divided by 6
0	1	1	0	Source clock (selected by CMSEL) divided by 7
0	1	1	1	Source clock (selected by CMSEL) divided by 8
1	0	0	0	Source clock (selected by CMSEL) divided by 9
1	0	0	1	Source clock (selected by CMSEL) divided by 10
1	0	1	0	Source clock (selected by CMSEL) divided by 11
1	0	1	1	Source clock (selected by CMSEL) divided by 12
1	1	0	0	Source clock (selected by CMSEL) divided by 13
1	1	0	1	Source clock (selected by CMSEL) divided by 14
1	1	1	0	Source clock (selected by CMSEL) divided by 15
1	1	1	1	Source clock (selected by CMSEL) divided by 16

Specifies the frequency of a clock signal output to a clock monitor terminal.

- bit3-0: Select an output frequency prescaler

CMSEL	CMSEL2	CMSEL1	CMSEL0	Clock Source Output to the MONCLK pin
0	**0**	**0**	**0**	**MONCLK output disabled (high impedance) (Initial)**
0	0	0	1	Main oscillation before clock supervisor
0	0	1	0	Sub oscillation before clock supervisor
0	0	1	1	RC oscillation
0	1	0	0	Sub clock (after sub clock selector SCKS)
0	1	0	1	Main oscillation after clock supervisor (MCLK_OUT)
0	1	1	0	Sub oscillation after clock supervisor (SCLK_OUT)
0	1	1	1	Clock modulator output to Clock Control
1	0	0	0	Clock modulator observer output
1	0	0	1	PLL output after 1/g divider (Auto Gear)
1	0	1	0	PLL output after 1/m divider (PLL output)
1	0	1	1	PLL output after 1/c divider (CAN clock)
1	1	0	0	PLL input after 1/n divider (PLL feedback)
1	1	0	1	CLKB
1	1	1	0	CLKP
1	1	1	1	CLKT

- **CSCFG: Address 04AEh (Access: Byte)**

7	6	5	4	3	2	1	0	bit
EDSUEN	PLLLOCK	RCSEL	**MONCKI**	CSC3	CSC2	CSC1	CSC0	
0	X	0	**0**	0	0	0	0	Initial value (INIT pin input, watchdog reset)
X	X	X	**X**	X	X	X	X	Initial value (software reset)
R/W	R	R/W	**R/W**	R/W	R/W	R/W	R/W	Attribute

(See “Meaning of Bit Attribute Symbols (Page No.10)” for details of the attributes.)

- bit4: Clock Monitor MONCLK inverter

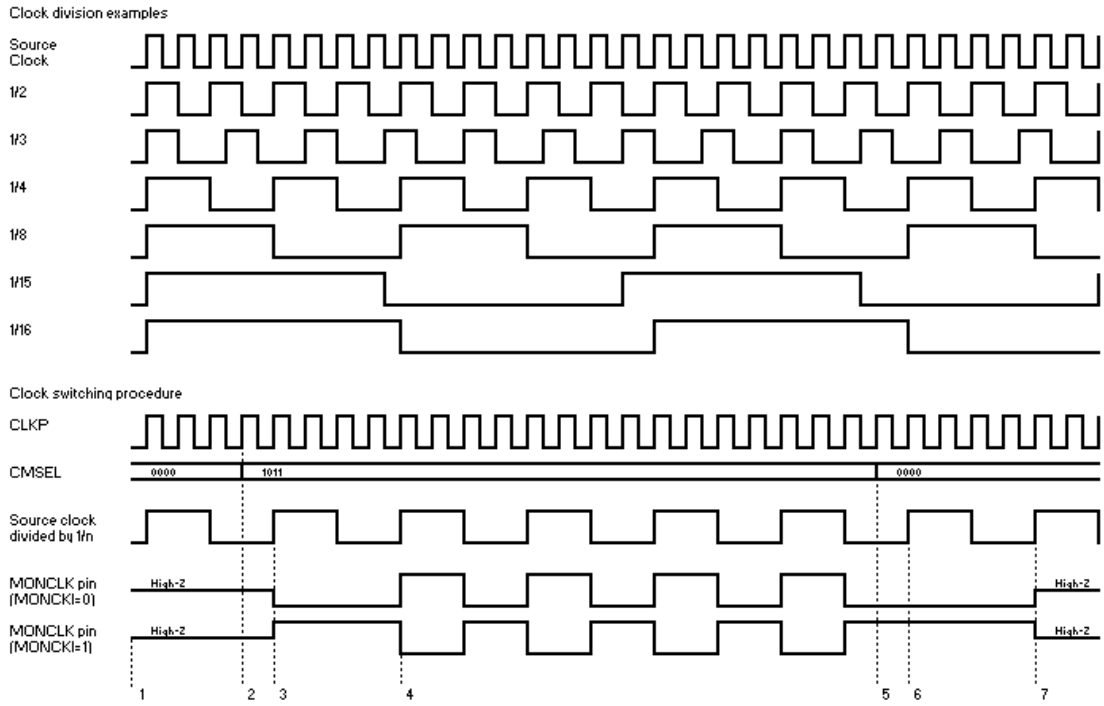
MONCKI	Function
0	MONCLK mark level is low [Initial value]
1	MONCLK mark level is high

- Other bits7-5, bits3-0

  Not belonging to clock monitor operation. See chapter “CSCFG: Clock Source Configuration Register (Page No.196)” about information on the additional functions of this register.

5. Operation

The following diagram shows the output waveforms of the Clock Monitor.



(1) The MONCLK pin is in high impedance state.

(2) CMSEL is set from “0000” (no clock selected) to the selected (and prescaled) clock.

(3) The MONCLK pin changes to output “L” status (output “H” if MONCKI is set to ‘1’) for one period of the internal (prescaled) clock.

(4) After one period of the selected (and prescaled) internal clock MONCLK outputs the selected (and prescaled) internal clock.

(5) CMSEL is set from the selected clock to “0000” (no clock selected).

(6) The MONCLK pin changes to output “L” status (output “H” if MONCKI is set to ‘1’) for one period of the internal (prescaled) clock.

(7) The MONCLK pin switches to high impedance state.

6. Settings

Table 6-1 Settings for Using Clock Monitor

Settings	Setting Registers	Setting Procedure*
Set a prescaler value	Clock Monitor Prescaler (CMCFG.CMPRE[3:0])	See 7.1
Set a source clock	Clock Monitor Selection (CMCFG.CMSEL[3:0])	See 7.1
Change the mark level	Clock Monitor Inverter (CSCFG.MONCKI)	See 7.1
Enable clock monitor output.(MONCLK)	Clock Monitor Selection (CMCFG.CMSEL[3:0])	See 7.2 See 7.3

*:For each setting procedure, refer to an appropriate section.

7. Q&A

7.1 How do I set an output terminal (MONCLK)?

Use the Clock Monitor Selection bits (CMCFG.CMSEL[3:0])

Operation	CMCFG.CMSEL[3:0]
To set an output terminal (MONCLK)	Set to the appropriate clock (!= “0000”)

7.2 How do I select an output frequency?

Use the Output Frequency Select bit (CMCFG.CMPRE[3:0]).

Clock Division Ratio	Output Frequency (Example)		Frequency Prescaler (CMCFG.CMPRE[3:0])
	CLKP=32MHz	CLKP=40MHz	
1/2	16.0 MHz	20.0 MHz	Set to “0001”.
1/3	10.7 MHz	13.3 MHz	Set to “0010”.
1/4	8.0 MHz	10.0 MHz	Set to “0011”.
1/8	4.0 MHz	5.0 MHz	Set to “0111”.
1/15	2.1 MHz	2.7 MHz	Set to “1110”.
1/16	2.0 MHz	2.5 MHz	Set to “1111”.

7.3 How do I enable/disable clock monitor output?

Use the Output Enable bit (CMCFG.CMSEL[3:0]).

Operation	Output Enable Bit (CMCFG.CMSEL[3:0])
To disable clock monitor output (To set the terminal to the Hi-z status)	Set to “0000”.
Enable clock monitor output.	Set to “0001”-”1111”.

8. Caution

Due to the glitch free switching mechanism it is necessary to follow these rules when switching the clock source (CMCFG3:0) or the prescaler ratio (CMPRE3:0):

- The CMPRE3:0 registers can only be written if the CMSEL3:0 registers are currently 0x0.
- The CMPRE3:0 registers can only be written if the CMSEL3:0 registers are written to 0x0 within the same write access.
- Between 2 write accesses to CMPRE/CMCFG there must be at least 2 cycles of the divided monitor clock.
- For changing the selector value, the MONCLK must be disabled.

Example access:

1. access:
   CMCFG_CMSEL = 0
   CMCFG_CMPRE = prescaler
2. access:
   CMCFG_CMSEL = clock

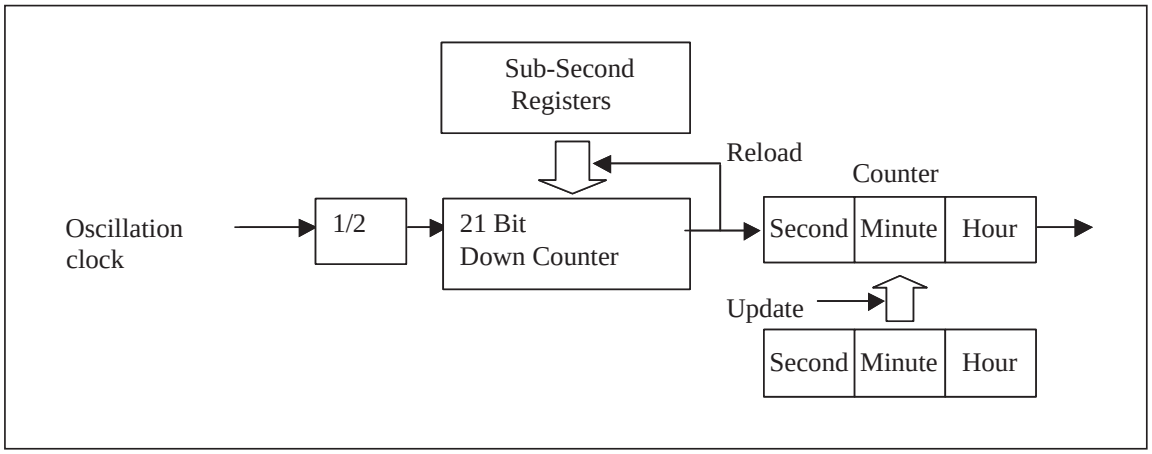
8.Caution

Chapter 49 Real-Time Clock

1. Overview

Real-time Clock (RTC) continues to count elapsed time even in the STOP mode to provide the current real time (HH/MM/SS) based on main oscillation (4MHz), sub oscillation (32kHz) or RC oscillation (~100kHz).

This allows precise time counting without a return from an interrupt during stand by periods.



2. Features

- Information: Time count (HH/MM/SS). (This clock continues to operate even in the STOP mode.)
- Operational on main clock (4MHz), sub clock (32kHz) or RC clock (~100kHz)
- Quantity: 1
- Time unit: Clock divided by 2
- Operation clock:
- For register access: CLKP
- For time count: Mainclock, Subclock, RC clock
- Time: Initial setting and adjustment are possible.
- Interrupt: Interrupts can be generated at any of the five intervals: 1 half-second, 1 second, 1 minute, 1 hour and 1 day.
- Others: By changing the value of the sub-second register, interrupts can be generated at any interval (from short to long).

3. Configuration

Figure 3-1 Configuration Diagram

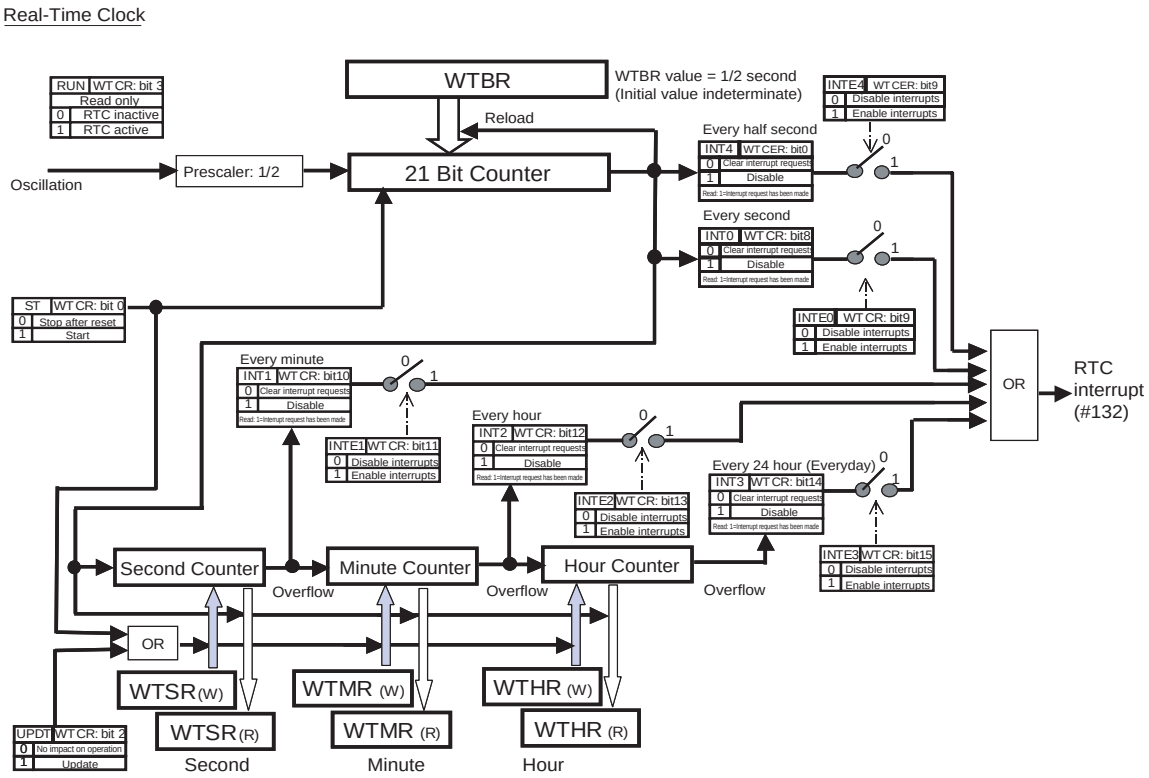
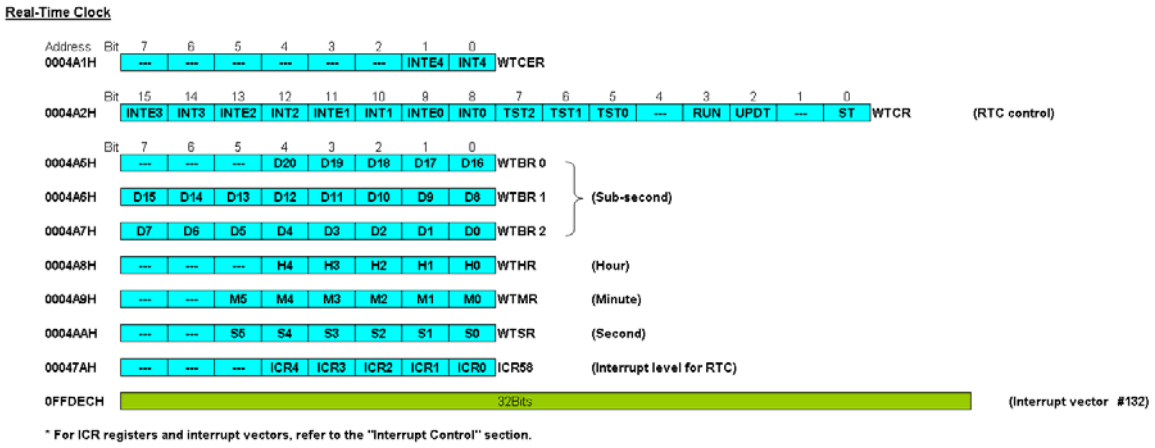


Figure 3-2 Register List



Note: For ICR registers and interrupt vectors, refer to “Chapter 24 Interrupt Control (Page No.315)”.

4. Registers

4.1 WTCR: RTC Control Register

This register is used to control behavior of the Real-time Clock module.

- **WTCR: Address $04A2_H$ (Access: Byte, Half-word)**

15	14	13	12	11	10	9	8	bit
INTE3	INT3	INTE2	INT2	INTE1	INT1	INTE0	INT0	
0	0	0	0	0	0	0	0	Initial value
0	0	0	0	0	0	0	0	When reset
R/W	R(R1),W	R/W	R(R1),W	R/W	R(R1),W	R/W	R(R1),W	Attribute

7	6	5	4	3	2	1	0	bit
Reserved	Reserved	Reserved	–	RUN	UPDT	–	ST	
0	0	0	–	0	0	–	X	Initial value
0	0	0	–	0	0	–	X	When reset
R/W0	R/W0	R/W0	RX/WX	R/WX	R(R0)/W	RX/WX	R/W	Attribute

(For attributes, refer to “Meaning of Bit Attribute Symbols (Page No.10)”.)

- bit15: Enable interrupt requests at 1-day intervals

INTE3	Operation
0	No interrupt requests
1	Generate interrupt requests at 1-day (24 hour) intervals.

When the hour counter overflows, this flag is set to “1”.

- bit14: 1-day interrupt request flag

INT3	Status	
	Read	Write
0	No interrupt requests	Clear the flag.
1	Generate interrupt requests at 1-day (24 hour) intervals.	Writing does not affect the operation.

- bit13: Enable interrupt requests at 1-hour intervals

INTE2	Operation
0	No interrupt requests
1	Generate interrupt requests at 1-hour intervals.

When the minute counter overflows, this flag is set to “1”.

- bit12: 1-hour interrupt request flag

INT2	Status	
	Read	Write
0	No interrupt requests	Clear the flag.
1	Generate interrupt requests at 1-hour intervals.	Writing does not affect the operation.

- bit11: Enable interrupt requests at 1-minute intervals

INTE1	Operation
0	No interrupt requests
1	Generate interrupt requests at 1-minute intervals.

When the minute counter overflows, this flag is set to “1”.

- bit10: 1-minute interrupt request flag

INT1	Operation	
	Read	Write
0	No interrupt requests	Clear the flag.
1	Generate interrupt requests at 1-minute intervals.	Writing does not affect the operation.

- bit9: Enable interrupt requests at 1-second intervals

INTE0	Operation
0	No interrupt requests
1	Generate interrupt requests at 1-second intervals.

When the 21 bit down counter is set to “0”, this flag is set to “1”.

- bit8: 1-second interrupt request flag

INT0	Status	
	Read	Write
0	No interrupt requests	Clear the flag.
1	Generate interrupt requests at 1-second intervals.	Writing does not affect the operation.

- bit7-5: Reserved

  Be sure to write “0”. The read value is the value written.
- bit4: Undefined

  Writing does not affect the operation. The read value is indeterminate.
- bit3: Operation status

RUN	Status
0	The Real-time Clock module is inactive.
1	The Real-time Clock module is active.

- bit2: Update

UPDT	Status/Operation
0	The update has been completed. (Writing “0” does not affect the operation.)
1	Update the hour/minute/second counters with the values of the hour/minute/second registers, respectively.

Before writing “1” to the update bit (UPDT), the hour/minute/second registers must be set to the values with which to update the hour/minute/second counters. The hour/minute/second registers are updated on reloading to the 21 bit down counter.

- bit1: Undefined

  Writing does not affect the operation. The read value is indeterminate.
- bit0: Start

ST	Operation
0	The Real-time Clock module stops to operate, and the 21 bit down counter and the hour/minute/second counters are cleared.
1	The settings of the hour/minute/second registers are loaded to the hour/minute/second counters, and the Real-time Clock module starts to operate.

Application Note: The Sub-second register of the RTC module stores the reload value for the 21bit prescaler. This value is reloaded after the reload counter reaches "0". When modifying all three bytes, make sure the reload operation will not be performed in between the write instructions. Otherwise the 21-bit prescaler loads the incorrect value of the combination of new data and old data bytes. It is generally

recommended that the Sub-Second register is updated while the ST bit is "0".

However, if this update is done immediately after an RTC second interrupt there should be enough time to securely modify the registers until the next reload operation (next second interrupt) even if ST is not set to "0" and the module is in operation.

When updating the registers by using the ST bit the following must be taken into account:

The new value is written into the registers with the rising edge of the RUN bit. This RUN bit is clocked by the RTC clock (32 kHz, 100kHz or 4 MHz depending on device and mode). To make sure that the update is done properly, write the new values into the registers, set ST to 0, wait for the RUN bit to go low and then start the circuit again by setting ST to 1. RUN will go low at the second rising edge of the RTC clock after ST has been set to 0. It will rise again at the half second rising edge of RTC clock after ST has been set to 1. If this operation is to be done several times directly after each other, wait for RUN to go to high before setting ST to low again.

- **WTCER: Address $04A1_H$ (Access: Byte)**

7	6	5	4	3	2	1	0	bit
–	–	–	–	–	–	INTE4	INT4	
–	–	–	–	–	–	0	0	Initial value
–	–	–	–	–	–	0	0	When reset
RX/WX	RX/WX	RX/WX	RX/WX	RX/WX	RX/WX	R/W	R(R1),W	Attribute

(For attributes, refer to “Meaning of Bit Attribute Symbols (Page No.10)”.)

- bit7-2: Undefined

  Writing does not affect the operation. The read value is indeterminate.
- bit1: Enable interrupt requests at half-second (500ms) intervals

INTE4	Operation
0	No interrupt requests
1	Generate interrupt requests at half-second (500 ms) intervals.

When the hour counter overflows, this flag is set to “1”.

- bit0: half-second (500ms) interrupt request flag

INT4	Status	
	Read	Write
0	No interrupt requests	Clear the flag.
1	Generate interrupt requests at half-second (500 ms) intervals.	Writing does not affect the operation.

4.2 WTBR: Sub-Second Registers

These registers are used to hold values to be reloaded to the 21 bit down counter.

- **WTBR0: Address $04A5_H$ (Access: Byte, Half-word, Word)**
- **WTBR1: Address $04A6_H$ (Access: Byte, Half-word, Word)**
- **WTBR2: Address $04A7_H$ (Access: Byte, Half-word, Word)**

WTBR0

7	6	5	4	3	2	1	0	bit
–	–	–	D20	D19	D18	D17	D16	
–	–	–	X	X	X	X	X	Initial value
–	–	–	Unchanged	Unchanged	Unchanged	Unchanged	Unchanged	When reset
RX/WX	RX/WX	RX/WX	R/W	R/W	R/W	R/W	R/W	Attribute

WTBR1

7	6	5	4	3	2	1	0	bit
D15	D14	D13	D12	D11	D10	D9	D8	
X	X	X	X	X	X	X	X	Initial value
Unchanged	Unchanged	Unchanged	Unchanged	Unchanged	Unchanged	Unchanged	Unchanged	When reset
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Attribute

WTBR2

7	6	5	4	3	2	1	0	bit
D7	D6	D5	D4	D3	D2	D1	D0	
X	X	X	X	X	X	X	X	Initial value
Unchanged	Unchanged	Unchanged	Unchanged	Unchanged	Unchanged	Unchanged	Unchanged	When reset
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Attribute

(For attributes, refer to “Meaning of Bit Attribute Symbols (Page No.10)”.)

The sub-second registers, WTBR, holds values to be reloaded to the 21 bit down counter. When the 21 bit down counter value becomes “0”, the settings of WTBR are reloaded to the 21 bit down counter.

(See “8. Caution (Page No.963)”.)

Remark: The reload value to be set in the sub-second registers corresponds to the time for half a second. One second is reached after counting twice the reload value set in WTBR.

(See “7.1 How do I set the count period of 1 second? (Page No.961)”.)

4.3 WTHR/WTMR/WTSR: Hour/Minute/Second Registers

These registers hold time information (HH/MM/SS) for Real-time Clock.

- **WTHR (Hour register): Address $04A8_H$ (Access: Byte, Half-word)**
- **WTMR (Minute register): Address $04A9_H$ (Access: Byte, Half-word)**
- **WTSR (Second register): Address $04AA_H$ (Access: Byte, Half-word)**

WTHR

7	6	5	4	3	2	1	0	bit
–	–	–	H4	H3	H2	H1	H0	
–	–	–	X	X	X	X	X	Initial value
–	–	–	Unchanged	Unchanged	Unchanged	Unchanged	Unchanged	When reset
RX/WX	RX/WX	RX/WX	R,W	R,W	R,W	R,W	R,W	Attribute

WTMR

7	6	5	4	3	2	1	0	bit
–	–	M5	M4	M3	M2	M1	M0	
–	–	X	X	X	X	X	X	Initial value
–	–	Unchanged	Unchanged	Unchanged	Unchanged	Unchanged	Unchanged	When reset
RX/WX	RX/WX	R,W	R,W	R,W	R,W	R,W	R,W	Attribute

WTSR

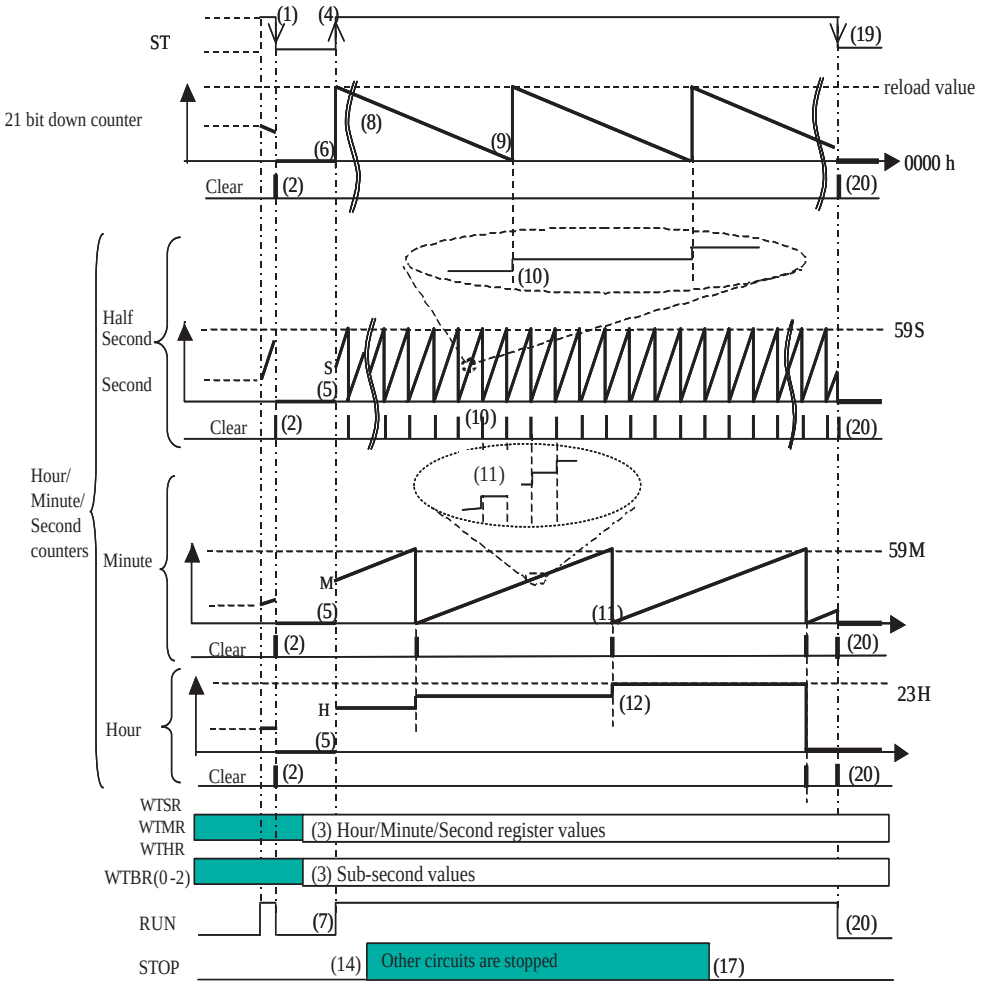
7	6	5	4	3	2	1	0	bit
–	–	S5	S4	S3	S2	S1	S0	
–	–	X	X	X	X	X	X	Initial value
–	–	Unchanged	Unchanged	Unchanged	Unchanged	Unchanged	Unchanged	When reset
RX/WX	RX/WX	R,W	R,W	R,W	R,W	R,W	R,W	Attribute

- The values written to the hour/minute/second registers are the initial values to be loaded to the hour/minute/second counters.
  By setting “1” to the update bit (WTCR.UPDT) or the start bit (WTCR.ST), the hour/minute/second register values are written to the hour/minute/second counters.
- The hour/minute/second counter values are saved to the hour/minute/second registers every time when the second counter overflows, that is, at intervals of one minute. When the hour/minute/second counters are read, the saved count values, not written ones, are read.
- The hour/minute/second registers consist of two separate sets of registers: one for reading and the other for writing.

(See “8. Caution (Page No.963)”.)

5. Operation

This section describes Real-time Clock operation.



(1) The start bit (ST) is set to “1” and then “0”. (Register initialization operation)

(2) This (ST=“0”) resets to 0 and stops the 21 bit down counter and the hour/minute/second timers.

(3)
- The software writes hour, minute, and second values to the hour/minute/second registers, WTHR/WTMR/WTSR.
- The software writes the appropriate values to the sub-second registers, WTBR0/WTBR1/WTBR2.
- The interrupt request bits (INT0, INT1, INT2, INT3 and INT4) are initialized, and the interrupt request enable bits (INTE0, INTE1, INTE2, INT3 and INTE4) are set to “interrupts enabled”.

(4) The start bit (ST) is set to “1”.

(5) This (ST=“1”) causes the values of the hour/minute/second registers, WTHR/WTMR/WTSR, to be loaded to the hour/minute/second timers.

(6) The values of the sub-second registers, WTBR0/WTBR1/WTBR2, are loaded to the 21 bit down counter.

(7) The run flag (RUN) is set to “1”.

(8) The 21 bit down counter begins counting at the mainclock divided by 2 (4/2 MHz), subclock divided by 2 (32.768/2 KHz) or RC clock divided by 2 (100/2 kHz).

(9) When the 21 bit down counter reaches “000000_H”, the value of the sub-second registers is loaded to the 21 bit down counter, generating a half-second interrupt request. Each second half-second interrupt a second interrupt will be generated.

(10) When the second counter counts up to “59”, the counter is cleared next time when the counter counts up, at which the minute counter counts up, generating a 1-minute interrupt request.

(11) When the minute counter counts up to “59”, the counter is cleared next time when the counter counts up, at which the hour counter counts up, generating a 1-hour interrupt request.

(12) When the hour counter counts up to “23”, the counter is cleared next time when the counter counts up, at which a 1-day interrupt request is generated.

(14) The software changes the status of Real-time Clock to STOP. (Set the stop bit (STCR.STOP) to “1”.) Real-time Clock continues to operate in the STOP state.

(17) The device recovers from STOP mode (by interrupt request).

(20) This (ST=“0”) resets and stops the 21 bit down counter and the hour/minute/second counters.

6. Setting

Table 6-1 Required Settings to Run Real-time Clock

Setting	Setting Registers	Setting Procedure *
Set a reload value to the sub-second registers.	Sub-second registers (WTBR0,WTBR1 and WTBR2)	See 7.1.
Initialize Real-time Clock.	RTC control register (WTCR)	See 7.2.
Set time (hour/minute/second).	Hour/minute/second registers (WTHR/WTMR/WTSR)	See 7.3.
Activate Real-time Clock.	RTC control register (WTCR)	See 7.4.

*: For the setting procedure, refer to the section indicated by the number.

Table 6-2 Required Settings to Know Time

Setting	Setting Registers	Setting Procedure *
Read time.	Hour/minute/second registers (WTHR/WTMR/WTSR)	See 7.6.

*: For the setting procedure, refer to the section indicated by the number.

Table 6-3 Required Settings to Stop Real-Time Clock

Setting	Setting Registers	Setting Procedure *
Stop Real-time Clock.	RTC control register (WTCR)	See 7.7.

*: For the setting procedure, refer to the section indicated by the number.

Table 6-4 Required Settings to Generate Interrupts to Real-time Clock

Setting	Setting Registers	Setting Procedure *
Set a RTC interrupt vector and RTC interrupt level.	See "Chapter 24 Interrupt Control (Page No.315)".	See 7.10.
Enable RTC interrupts. Clear interrupt requests. Enable interrupt requests.	RTC control register (WTCR)	See 7.10.

*: For the setting procedure, refer to the section indicated by the number.

Table 6-5 Required Settings to select clock source to Real-time Clock

Setting	Setting Registers	Setting Procedure *
Set a RTC clock source.	See "4.4 CSCFG: Clock Source Configuration Register (Page No.196)".	

*: For the setting procedure, refer to the section indicated by the number.

7. Q&A

7.1 How do I set the count period of 1 second?

Stop Real-time Clock and then set the sub-second register WTBR. The reload value corresponds to the time needed for half a second, i.e.

- At 32 kHz RTC operation set WTBR to “$001FFF_H$”.
- At 100 kHz RTC operation set WTBR to “$0061A7_H$”.
- At 4 MHz RTC operation set WTBR to “$0F423F_H$”.

7.2 How do I initialize Real-time Clock?

Use the start bit (WTCR.ST).

Changing the start bit from “1” to “0” resets the hour/minute/second counters and the 21 bit down counter to “0” (initialization), and stops count operation.

7.3 How do I set or update time (hour/minute/second)?

Write values to the hour/minute/second registers, WTHR/WTMR/WTSR, and set the update bit (UPDT).

Operation	Update bit (UPDT)
To update the hour/minute/second counters	Set the bit to “1”.

7.4 How do I start or stop Real-time Clock's counting?

Use the start bit (WTCR.ST).

Operation	Start bit (ST)
To stop Real-time Clock's counting	Set the bit to “0”.
To start Real-time Clock's counting	Set the bit to “1”.

7.5 How do I confirm that Real-time Clock is active?

Use the run flag (WTCR.RUN).

Operation	Run flag (RUN)
Real-time Clock is inactive.	The flag is set to “0”.
Real-time Clock is active.	The flag is set to “1”.

7.6 How do I know time?

Read the hour/minute/second registers, WTHR/WTMR/WTSR.

Note that only byte-access is allowed to these register. So, when these registers are read at the very timing of changing over the hour or minute boundary as shown below, there is a possibility of misjudging the time. So, read several times to get a logically consistent value.

Example: Read begins at the second register: 02:59:59 (SS) => 03:59:59 (SS) => 03:00:00

Read begins at the hour register: 02:59:59 => 02:00:00 => 03:00:00

7.7 How do I stop Real-time Clock?

See 7.4.

7.8 What are interrupt-related registers?

RTC interrupt vector and level settings.

The following table shows the relationship between interrupt levels and vectors.

For details on interrupt levels and vectors, refer to “Chapter 24 Interrupt Control (Page No.315)”.

Interrupt vectors (Default)	Interrupt level set bit (ICR[4.0])
#132 (0FFDECh)	Interrupt level register ICR58 (047Ah)

The interrupt request flags (INT0,INT1,INT2,INT3 and INT4) are not automatically cleared, so the software must clear them by writing “0” to these flags before control is returned from interrupt processing.

7.9 What interrupts are available and how are they selected?

There are four interrupt causes:

Interrupt cause	Interrupt request bit	Interrupt request enable bits
On counting seconds	INT0	INTE0
On counting minutes	INT1	INTE1
On counting hours	INT2	INTE2
On counting days	INT3	INTE3
On counting half-seconds	INT4	INTE4

An interrupt request is made by ORing these four interrupt causes. Each cause can be selected with the corresponding interrupt request enable bit.

7.10 How do I enable interrupts?

Use the interrupt request enable bits (WTCR.INTE0, WTCR.INTE1, WTCR.INTE2, WTCR.INTE3 and WTCER.INTE4).

	Setting Procedure
	Interrupt request enable bits (INTE0, INTE1, INTE2, INTE3 and INTE4)
To disable interrupts	Set the bit to “0”.
To enable interrupts	Set the bit to “1”.

To clear interrupt requests,

Use the interrupt request bits (WTCR.INT0, WTCR.INT1, WTCR.INT2, WTCR.INT3 and WTCER.INT4).

	Setting Procedure
	Interrupt request bits (INT0, INT1, INT2, INT3 and INT4)
To clear interrupt requests	Write “0”.

8. Caution

- Setting the interrupt request flags (WTCR.INT0, WTCR.INT1, WTCR.INT2, WTCR.INT4 and WTCER.INT4) to “1” due to overflow, and writing “0” to that bit have occurred at the same time, the flag is set to “1”. (Flag setting takes precedence.)

- Writing “1” to the update bit (ETCR.UPDT) and update completion have occurred at the same time, the update bit (UPDT) is set to “0”.

- When the second counter holds the value of 59, even if “1” is written to the update bit (WTCR.UPDT), the hour/minute/second counters are not updated, leaving the update bit to remain “0”.
  In order to update the hour/minute/second counters, it is recommended that “0” should be written to the start bit (WTCR.ST), the hour/minute/second counters be cleared to “0”, and then “1” be written to the start bit (ST).

- If you stop the peripheral clock (CLKP) after updating the hour/minute/second counters using the update bit (WTCR.UPDT), read the hour/minute/second registers to confirm that they have been updated before stopping the peripheral clock.

- When you start to use the Real-time Clock module, change the start bit (ST) from “1” to “0”, and clear the hour/minute/second counters and the 21 bit down counter to “0”.

- The Sub-second register of the RTC module stores the reload value for the 21bit prescaler. This value is reloaded after the reload counter reaches "0". When modifying all three bytes, make sure the reload operation will not be performed in between the write instructions. Otherwise the 21-bit prescaler loads the incorrect value of the combination of new data and old data bytes. It is generally recommended that the Sub-Second register is updated while the ST bit is "0".
  However, if this update is done immediately after an RTC second interrupt there should be enough time to securely modify the registers until the next reload operation (next second interrupt) even if ST is not set to "0" and the module is in operation.
  When updating the registers by using the ST bit the following must be taken into account:
  The new value is written into the registers with the rising edge of the RUN bit. This RUN bit is clocked by the RTC clock (32 kHz, 100kHz or 4 MHz depending on device and mode). To make sure that the update is done properly, write the new values into the registers, set ST to 0, wait for the RUN bit to go low and then start the circuit again by setting ST to 1. RUN will go low at the second rising edge of the RTC clock after ST has been set to 0. It will rise again at the half second rising edge of RTC clock after ST has been set to 1. If this operation is to be done several times directly after each other, wait for RUN to go to high before setting ST to low again.

- If a reload has occurred during updating the sub-second registers, WTBR0 to WTBR2, an unexpected value may be reloaded to the 21 bit down counter. Therefore, it is recommended that the sub-second register, WTBR, should be updated with the start bit (WTCR.ST) set to “0”.

- If all the sub-second registers, WTBR0 to WTBR2, are set to “0”, the 21 bit down counter does not operate, resulting in the Real-time Clock module to be inoperational.

- If a carry has occurred during reading from the hour/minute/second registers, WTHR/WTMR/WTSR, inappropriate values may be read. To avoid this, it is recommended that interrupts (INT0-4) should be used to read time (HH/MM/SS).

- In order for the Real-time Clock module to function properly, the frequency of the subclock must be much

lower than that of the peripheral clock (CLKP). If not, correct values cannot be read from WTHR/WTMR/WTSR.

- Note that only byte-access is allowed to these register. So, when these registers are read at the very timing of changing over the hour or minute boundary as shown below, there is a possibility of misjudging the time. So, read several times to get a logically consistent value.

  Example: Read begins at the second register: 02:59:59=> 03:59:59 => 03:00:00

  Read begins at the hour register: 02:59:59 => 02:00:00 => 03:00:00

  In this case, the current time should be interpreted as 3 o'clock.

Chapter 50 Subclock Calibration Unit

1. Overview

The Clock Calibration Module provides possibilities to calibrate the 32kHz oscillation clock or 100kHz RC oscillation clock with respect to the 4MHz oscillation clock. This chapter gives an overview of the calibration unit, describes the registers and provides some application notes.

1.1 Description

This hardware allows the software to measure time generated by the 32kHz clock (or 100kHz RC clock) with the 4MHz clock.

By utilizing this hardware in conjunction with software processing, the accuracy of the 32kHz clock (or 100kHz RC clock) can come closer to that of the 4MHz clock. The measurement result from the Clock Calibration Module can be processed by the software and the setting required for the Real Time Clock Module can be obtained.

This module consists of two timers, one operating with the 32kHz clock (or 100kHz RC clock) and the other operating with the 4MHz clock. The 32kHz (100kHz) timer triggers the 4MHz timer and resulting 4MHz timer value is stored in a register. The value stored in this register can be used for the subsequent software processing to calculate the desired Real Time Clock module’s setting.

2. Block Diagram

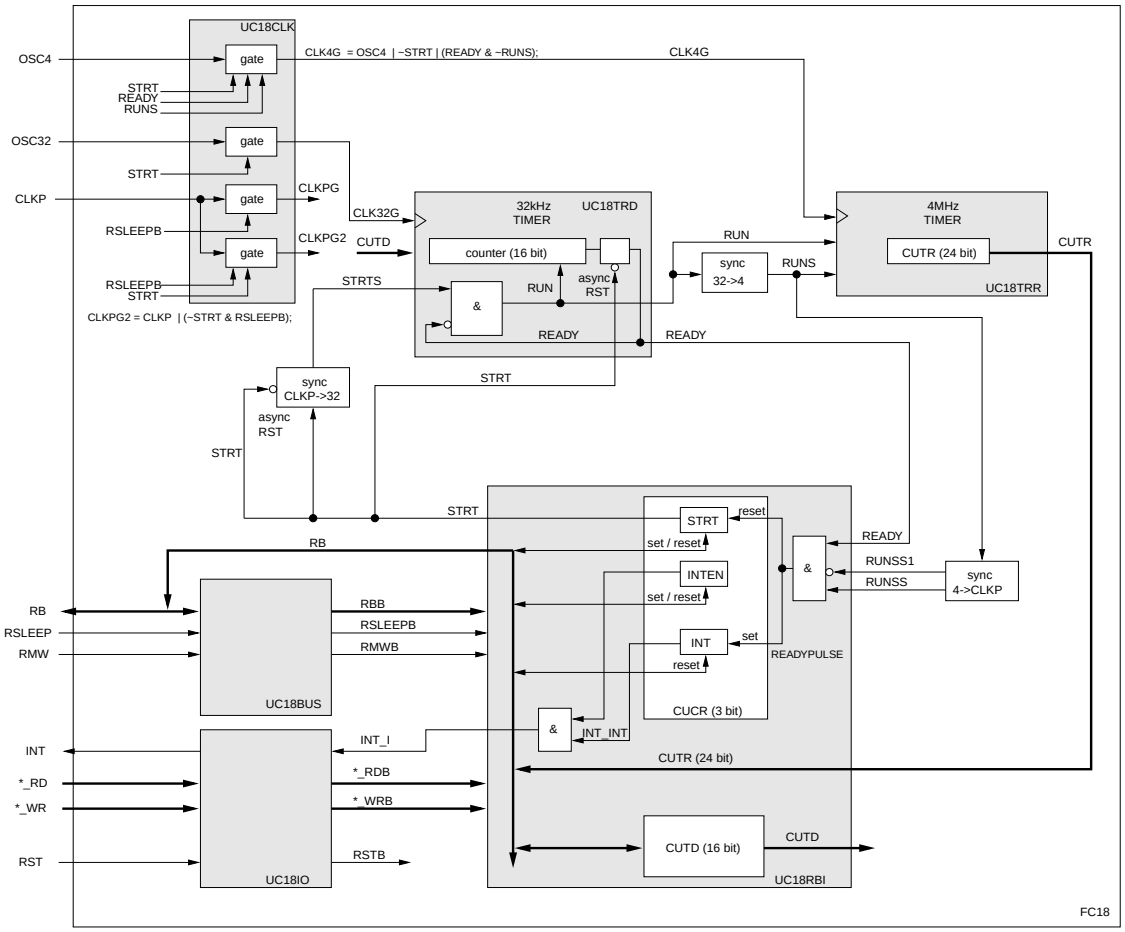


Figure 2-1 Block Diagram of the calibration unit

3. Timing

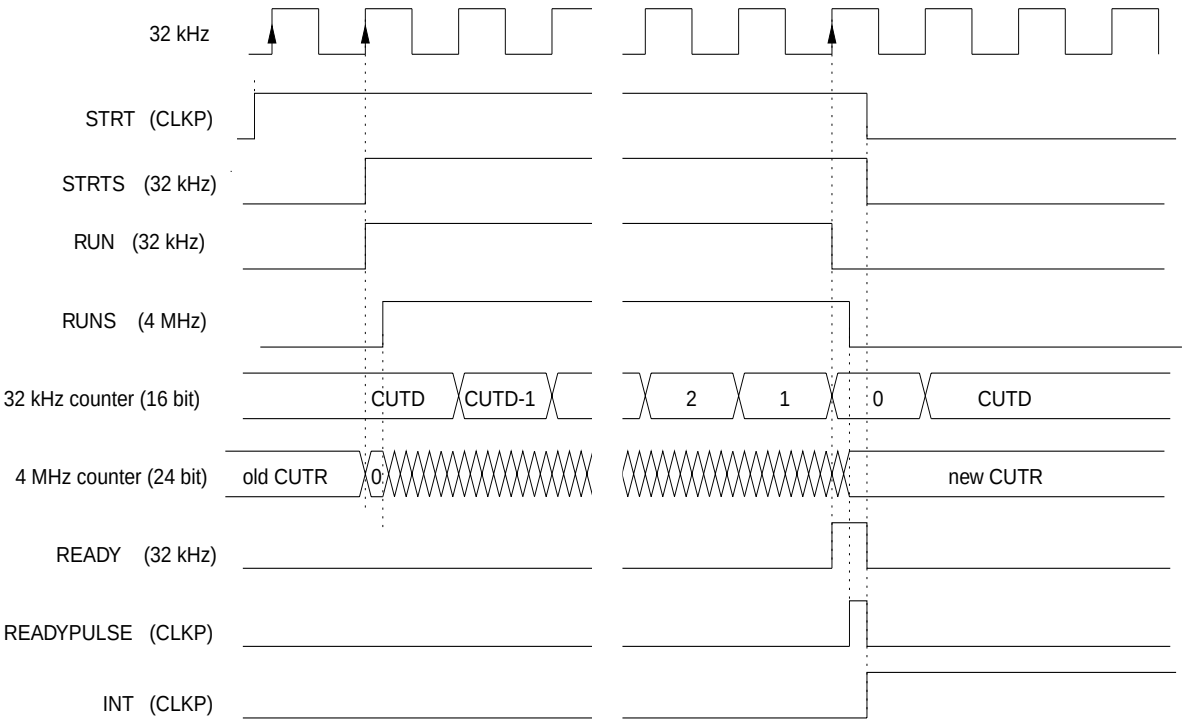


Figure 3-1 Timing of the measurement process

4. Clocks

The module operates with 3 different clocks: The 4 MHz clock OSC4, the 32 kHz clock OSC32 (ot the 100kHz clock OSC100) and the peripheral clock CLKP. Synchronization circuits adapt the different domains.

The clock frequencies have to fulfill the following requirements:

- Clock ratio

$$T_{OSC32/OSC100} > 2 \times T_{OSC4} + 3 \times T_{CLKP}$$

$$T_{OSC4} < 1/2 \times T_{OSC32/OSC100} - 3/2 \times T_{CLKP}$$

$$T_{CLKP} < 1/3 \times T_{OSC32/OSC100} - 2/3 \times T_{OSC4}$$

- The input frequencies must not exceed the values given in Table 4-1 .

Table 4-1 Maximum operation frequencies

	OSC32/OSC100		OSC4		CLKP	
maximum	2 MHz	500 ns	10 MHz	100 ns	50 MHz	20 ns

Table 4-2 Example of valid clock ratios which fulfill requirements 1 and 2

	OSC32		OSC100		OSC4		CLKP	
maximum operation speed	2 MHz	500 ns	2 MHz	500 ns	10 MHz	100 ns	50 MHz	20 ns
normal operation	32 kHz	31.25 us	100 kHz	10 us	4 MHz	250 ns	>2 MHz	500 ns

5. Register Description

This section lists the registers of the calibration unit and describes the function of each register in detail.

■ Calibration Unit Control Register (CUCR)

Control Register low byte	7	6	5	4	3	2	1	0	⇐ Bit no.
Address : 0004B0H	-	-	-	STRT	-	-	INT	INTEN	CUCRL
Read/write ⇒	(R)	(R)	(R)	(R/W)	(R)	(R/W)	(R/W)	(R/W)	
Default value⇒	(0)	(0)	(0)	(0)	(0)	(0)	(0)	(0)	

■ 32kHz/100kHz Timer Data Register (CUTD)

32/100kHz Timer Register high byte	15	14	13	12	11	10	9	8	⇐ Bit no.
Address : 0004B2H	TDD15	TDD14	TDD13	TDD12	TDD11	TDD10	TDD9	TDD8	CUTDH
Read/write ⇒	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	
Default value⇒	(1)	(0)	(0)	(0)	(0)	(0)	(0)	(0)	

32/100kHz Timer Register low byte	7	6	5	4	3	2	1	0	⇐ Bit no.
Address : 0004B3H	TDD7	TDD6	TDD5	TDD4	TDD3	TDD2	TDD1	TDD0	CUTDL
Read/write ⇒	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	
Default value⇒	(0)	(0)	(0)	(0)	(0)	(0)	(0)	(0)	

See also 4.4 CSCFG: Clock Source Configuration Register (Page No.196) for the configuration of the Calibration Unit clock sources.

■ 4MHz Timer Data Register (CUTR1/CUTR2)

4MHz Timer Register1 high byte

	15	14	13	12	11	10	9	8	⇐ Bit no.
Address : 0004B4H	-	-	-	-	-	-	-	-	CUTR1H
Read/write ⇒	(R)	(R)	(R)	(R)	(R)	(R)	(R)	(R)	
Default value⇒	(0)	(0)	(0)	(0)	(0)	(0)	(0)	(0)	

4MHz Timer Register1 low byte

	7	6	5	4	3	2	1	0	⇐ Bit no.
Address : 0004B5H	TDR23	TDR22	TDR21	TDR20	TDR19	TDR18	TDR17	TDR16	CUTR1L
Read/write ⇒	(R)	(R)	(R)	(R)	(R)	(R)	(R)	(R)	
Default value⇒	(0)	(0)	(0)	(0)	(0)	(0)	(0)	(0)	

4MHz Timer Register2 high byte

	15	14	13	12	11	10	9	8	⇐ Bit no.
Address : 0004B6H	TDR15	TDR14	TDR13	TDR12	TDR11	TDR10	TDR9	TDR8	CUTR2H
Read/write ⇒	(R)	(R)	(R)	(R)	(R)	(R)	(R)	(R)	
Default value⇒	(0)	(0)	(0)	(0)	(0)	(0)	(0)	(0)	

4MHz Timer Register2 low byte

	7	6	5	4	3	2	1	0	⇐ Bit no.
Address : 0004B7H	TDR7	TDR6	TDR5	TDR4	TDR3	TDR2	TDR1	TDR0	CUTR2L
Read/write ⇒	(R)	(R)	(R)	(R)	(R)	(R)	(R)	(R)	
Default value⇒	(0)	(0)	(0)	(0)	(0)	(0)	(0)	(0)	

Address	Register				Block
	+0	+1	+2	+3	
$0004B0_H$	CUCR [R/W] -------- ---0--00		CUTD [R/W] 10000000 00000000		Calibration Unit of 32kHz and 100kHz oscillator
$0004B4_H$	CUTR1 [R] -------- 00000000		CUTR2 [R] 00000000 00000000		

5.1 Calibration Unit Control Register (CUCR)

The Control Register (CUCR) has the following functions:
start / stop calibration measurement
enable / disable interrupt
indicates the end of the calibration measurement

Control Register low byte

	7	6	5	4	3	2	1	0	⇐ Bit no.
Address : 0004B1H	-	-	-	STRT	-	-	INT	INTEN	CUCRL
Read/write ⇒	(R)	(R)	(R)	(R/W)	(R)	(R/W)	(R/W)	(R/W)	
Default value⇒	(0)	(0)	(0)	(0)	(0)	(0)	(0)	(0)	

BIT[0]: INTEN - Interrupt enable

0	interrupt disabled (default)
1	interrupt enabled

This is the interrupt enable bit corresponding to the INT bit. When this bit is set to 1 and the INT bit is set by the hardware, the calibration module signals an interrupt to the CPU. The INT-bit itself is not affected by the INTEN bit and is set by hardware even if interrupts are disabled (INTEN=0).

BIT[1]: INT - Interrupt

0	calibration ongoing / module inactive (default)
1	calibration completed

This bit indicates the end of the calibration. When the 32KHz/100kHz timer reaches zero after the start of calibration, the 4MHz Timer Data Register stores the last 4MHz timer value and the INT bit is set to 1.

The read-modify-write operation to this bit results in reading 1 and writing 0 to this bit clears this flag(INT=0). Writing 1 to this bit has no effect.

The interrupt flag INT is not reset by hardware. Therefor it must be reset by software before starting a new calibration. Otherwise the end of the calibration process is only signalized by the STRT bit (the INT flag stays 1 also during calibration).

BIT[4]: STRT - Calibration Start

0	calibration stopped, module switched off (default)
1	start calibration

When the STRT bit is set to 1 by the software, the calibration starts. The 32kHz/100kHz Timer starts counting down from the value stored in the 32KHz/100kHz Timer Data Register and the 4MHz Timer starts counting up from zero.

When the 32KHz/100kHz Timer reaches zero, this bit is reset to 0 by the hardware.

If 0 is written into this bit by the software during the calibration process, the calibration is immediately stopped. If writing 0 by the software and reset to 0 by the hardware happens at the same time, the hardware operation supersede the software operation. This means the calibration is properly completed and the INT bit is set to “1”. Writing 1 to this bit during the calibration has no effect.

5.2 32 kHz / 100 kHz Timer Data Register (16 bit) (CUTD)

The 32kHz/100kHz Timer Data Register (CUTD) holds the value which determines the duration of calibration (32kHz/100kHz reload value)

32/100kHz Timer Register high byte	15	14	13	12	11	10	9	8	⇐ Bit no.
Address : 0004B2H	TDD15	TDD14	TDD13	TDD12	TDD11	TDD10	TDD9	TDD8	CUTDH
Read/write ⇒	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	
Default value⇒	(1)	(0)	(0)	(0)	(0)	(0)	(0)	(0)	

32/100kHz Timer Register low byte	7	6	5	4	3	2	1	0	⇐ Bit no.
Address : 0004B3H	TDD7	TDD6	TDD5	TDD4	TDD3	TDD2	TDD1	TDD0	CUTDL
Read/write ⇒	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	
Default value⇒	(0)	(0)	(0)	(0)	(0)	(0)	(0)	(0)	

Default value is 0x8000 which corresponds to a measurement duration of 1 second, if a 32.768 kHz crystal is used.

This register should be written only when the calibration is inactive (STRT=0).

The 32kHz/100kHz Timer Register stores the value to specify the duration of the calibration. When the calibration is started, the stored value is loaded into the 32kHz/100kHz Timer and the timer starts counting down until it reaches zero.

If CUTD is initialized with 0000 an underflow will occur and the measurement will take (FFFF hex + 1) * $T_{osc32/osc100}$

The 32kHz/100kHz timer operates with the 32kHz or with the 100kHz oscillation clock (see chapter “4.4 CSCFG: Clock Source Configuration Register (Page No.196)” how to select between those clocks.

In order to achieve a measurement duration of 1 second at a 32kHz oscillation, the CUTD register has to be load with 0x8000 = 32768 dec. This number results from the exact oscillation frequency of the crystal, which is F_{osc}=32768 Hz. The ideal values of the measurement results (if a 4.00 MHz crystal is used) are shown in the following table.

In order to achieve a measurement duration of half a second at a 100kHz oscillation, the CUTD register has to be load with 0xC350 = 50000 dec. This number results from the exact oscillation frequency of the crystal, which is F_{osc}=100000 Hz. The ideal values of the measurement results (if a 4.00 MHz crystal is used) are shown in the following table.

Table 5-1 32kHz : Ideal measurement results depending on measurement duration

duration of calibration	CUTD value	CUTR value
2 sec	0x0000	0x7A1200
1.75 sec	0xE000	0x6ACFC0
1.5 sec	0xC000	0x5B8D80
1.25 sec	0xA000	0x4C4B40
1 sec	0x8000	0x3D0900
0.75 sec	0x6000	0x2DC6C0
0.5 sec	0x4000	0x1E8480
0.25 sec	0x2000	0x0F4240

The duration of the whole process from writing a 1 into the STRT bit until STRT is reset by hardware is longer than the actual calibration measurement time, due to synchronization between the different clock domains. Process Duration < (CUTD + 3)*T_{osc32} .

The calibration measurement time is exact CUTD * T_{osc32}.

Table 5-2 100kHz : Ideal measurement results depending on measurement duration

duration of calibration	CUTD value	CUTR value
0.5 sec	0xC350	0x1E8480
0.25 sec	0x61A8	0x0F4240
0.125 sec	0x30D4	0x07A120
0.1 sec	0x2710	0x061A80

The duration of the whole process from writing a 1 into the STRT bit until STRT is reset by hardware is longer than the actual calibration measurement time, due to synchronization between the different clock domains. Process Duration < (CUTD + 3)*T_{osc100} .

The calibration measurement time is exact CUTD * T_{osc100}.

5.3 4 MHz Timer Data Register (24 bits) (CUTR)

The Timer Data Register (CUTR) holds the value of the calibration result (4MHz counter)

Precaution: Reading this register during calibration, results in random values.
The end of calibration is indicated by the INT-bit and the STRT-bit in the CUCR-register.
After INT has changed from 0 to 1 / STRT has changed from 1 to 0, the value of CUTR is valid.

4MHz Timer Register1 high byte — Address : 0004B4H — CUTR1H

⇐ Bit no.	15	14	13	12	11	10	9	8
	-	-	-	-	-	-	-	-
Read/write ⇒	(R)	(R)	(R)	(R)	(R)	(R)	(R)	(R)
Default value⇒	(0)	(0)	(0)	(0)	(0)	(0)	(0)	(0)

4MHz Timer Register1 low byte — Address : 0004B5H — CUTR1L

⇐ Bit no.	7	6	5	4	3	2	1	0
	TDR23	TDR22	TDR21	TDR20	TDR19	TDR18	TDR17	TDR16
Read/write ⇒	(R)	(R)	(R)	(R)	(R)	(R)	(R)	(R)
Default value⇒	(0)	(0)	(0)	(0)	(0)	(0)	(0)	(0)

4MHz Timer Register2 high byte — Address : 0004B6H — CUTR2H

⇐ Bit no.	15	14	13	12	11	10	9	8
	TDR15	TDR14	TDR13	TDR12	TDR11	TDR10	TDR9	TDR8
Read/write ⇒	(R)	(R)	(R)	(R)	(R)	(R)	(R)	(R)
Default value⇒	(0)	(0)	(0)	(0)	(0)	(0)	(0)	(0)

4MHz Timer Register2 low byte — Address : 0004B7H — CUTR2L

⇐ Bit no.	7	6	5	4	3	2	1	0
	TDR7	TDR6	TDR5	TDR4	TDR3	TDR2	TDR1	TDR0
Read/write ⇒	(R)	(R)	(R)	(R)	(R)	(R)	(R)	(R)
Default value⇒	(0)	(0)	(0)	(0)	(0)	(0)	(0)	(0)

The 4MHz Timer Data Register stores the result of the calibration. When the calibration is started, the 4MHz Timer starts counting up from zero. When the 32kHz/100kHz Timer reaches zero, the 4MHz Timer stops counting and the register holds the calibration result until the next calibration is triggered by software.

Reading this register during calibration, results in random values.

The end of calibration is indicated by the INT-bit and the STRT-bit in CUCR-register. After these bits changed from 0 to 1, resp. 1 to 0, the value of CUTR is valid.

Writing into this register by software has no effect.

The 4MHz Timer operates with the 4MHz oscillation clock.

6. Application Note

This section lists application notes concerning accuracy of the calibration, power dissipation and measurement duration.

- 32kHz

  The setting of the 32KHz Timer Data Register can be calculated in the following way.

  If the duration of 1 second is desired for the calibration, 8000Hex = 32768Dec should be set in the 32kHz Timer Data Register and it represents 32,768 pulses of the 32.768kHz oscillation clock.

  This setting should result in the stored value of approximately 3D0900Hex in the 4MHz Time Data Register. This value represents 4,000,000 pulses of the 4MHz oscillator.

- 100kHz

  The setting of the 100KHz Timer Data Register can be calculated in the following way.

  If the duration of 0.5 second is desired for the calibration, C350Hex = 50000Dec should be set in the 100kHz Timer Data Register and it represents 50,000 pulses of the 100kHz oscillation clock.

  This setting should result in the stored value of approximately 1E8480Hex in the 4MHz Time Data Register. This value represents 2,000,000 pulses of the 4MHz oscillator.

Table 6-1 Ideal measurement results (CUTR) with 32.768kHz and 4.0MHz oscillators

duration of calibration	CUTD value	CUTR value
2 sec	0x0000	0x7A1200
1.75 sec	0xE000	0x6ACFC0
1.5 sec	0xC000	0x5B8D80
1.25 sec	0xA000	0x4C4B40
1 sec	0x8000	0x3D0900
0.75 sec	0x6000	0x2DC6C0
0.5 sec	0x4000	0x1E8480
0.25 sec	0x2000	0x0F4240

The key to the use of the calibration module is power dissipation as well as accuracy of the calibration.

Table 6-2 Ideal measurement results (CUTR) with 100kHz and 4.0MHz oscillators

duration of calibration	CUTD value	CUTR value
0.5 sec	0xC350	0x1E8480
0.25 sec	0x61A8	0x0F4240
0.125 sec	0x30D4	0x07A120
0.1 sec	0x2710	0x061A80

The key to the use of the calibration module is power dissipation as well as accuracy of the calibration.

■ Accuracy:

The accuracy of the calibration is dependent on the clock frequency used by the 4MHz Timer and duration of the calibration. The maximum error of the 4MHz timer is +/- 1 digit. If the clock frequency is 4MHz and duration of the calibration is 1 second, the achieved accuracy is calculated in the following way:

0.25us (Clock cycle time) / 1 second (duration)=0.25 ppm.

In general:

Accuracy = (Clock cycle time of 4MHz Timer) / (Duration of calibration)

■ Power dissipation:

Suppose the current consumption I_{RUN} in run mode is 20 times the consumption I_{RTC} in RTC mode ($I_{RUN} = 20 \times I_{RTC}$).

If the MCU is woken up from RTC mode by software to trigger the calibration measurement every minute and the duration of the calibration is set to 1 second, the increase in the power dissipation can be $20xI_{RTC}/60= 1/3 * I_{RTC}$.

Therefore the software has to make sure that the increase should not affect the hardware limitations coming from the system requirements. For example, the software has to be designed to trigger the calibration least frequently.

It is generally recommended that the theoretical increase in power dissipation is not more than 5% particularly in the RTC mode of the MCU.

■ Measurement limits:

The limit to the duration of the calibration is roughly 2 seconds if the 32kHz/100kHz Timer is operating with 32kHz clock (0.5 seconds if operating with 100kHz). On the other hand, the 4MHz Timer can measure time up to 4 seconds if it is working with a 4MHz clock.

Chapter 51 Low Voltage Reset/Interrupt

1. Overview

- Module for generating a low voltage reset or interrupt depending on the supply state of either the internal or external supply voltage.

2. Features

- Generates a low voltage reset or a low voltage interrupt
- Interrupt activation source can be selected between external supply (VDD5) detection and internal supply (VDD) detection
- Selectable trigger levels for external and internal supply levels
- Power down function

3. Registers

3.1 LV Detection Control Registers

Controls the low voltage detection function.

- **LVDET: Address 04C5h (Access: Byte, Halfword, Word)**

7	6	5	4	3	2	1	0	bit
-	LVSEL	LVEPD	LVIPD	LVREN	-	LVIEN	LVIRQ	
-	0	0	0	0	-	0	0	Initial value (INIT pin input, watchdog reset)
-	X	X	X	X	-	0	0	Initial value (Software reset)
R0/W0	R/W	R/W	R/W	R/W	R0/W0	R/W	R/W	Attribute

(See "Meaning of Bit Attribute Symbols (Page No.10)" for details of the attributes.)

- Bit7: Reserved bit. The read value is always '0'.
- Bit6: Low-Voltage Interrupt Source selection.

LVSEL	Function
0	Internal (VDD) LV detection used for LV-Int and LV-Reset [Initial value]
1	External (VDD5) LV detection used for LV-Int and LV-Reset

- Bit5: External Low-Voltage Detection Power-Down.

LVEPD	Function
0	External (VDD5) LV detection active [Initial value]
1	External (VDD5) LV detection power down

- Bit4: Internal Low-Voltage Detection Power-Down.

LVIPD	Function
0	Internal (VDD) LV detection active [Initial value]
1	Internal (VDD) LV detection power down

- Bit3: Low-Voltage Reset Enable.

LVREN	Function
0	Low Voltage Reset Disabled [Initial value]
1	Low Voltage Reset Enabled

- Bit2: Reserved bit.
- Bit1: Low-Voltage Interrupt Enable.

LVIEN	Function
0	Low Voltage Interrupt Disabled [Initial value]
1	Low Voltage Interrupt Enabled

- Bit0: Low-Voltage Interrupt Flag.

LVIRQ	Function
0	Low Voltage Interrupt Flag (no interrupt request) [Initial value]
1	Low Voltage Interrupt Flag (interrupt requested)

- **LVSEL: Address 04C4h (Access: Byte, Halfword, Word)**

7	6	5	4	3	2	1	0	bit
LVESEL3	LVESEL2	LVESEL1	LVESEL0	LVISEL3	LVISEL2	LVISEL1	LVISEL0	
0	0	0	0	0	1	1	1	Initial value (INIT pin input, watchdog reset)
X	X	X	X	X	X	X	X	Initial value (Software reset)
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Attribute

(See "Meaning of Bit Attribute Symbols (Page No.10)" for details of the attributes.)

- Bit7-4: External LV detection voltage level

LVESEL3-LVESEL0	Trigger level
1001	4.2V +/- 0.1V
1000	4.1V +/- 0.1V
0111	4.0V +/- 0.1V
0110	3.9V +/- 0.1V
0101	3.8V +/- 0.1V
0100	3.7V +/- 0.1V
0011	3.6V +/- 0.1V
0010	3.2V +/- 0.1V
0001	3.0V +/- 0.1V
0000	2.8V +/- 0.1V (initial)

- Bit3-0: Internal LV detection voltage level

LVISEL3-LVISEL0	Trigger level
0111	1.6V +/- 0.1V (initial)
0110	1.5V +/- 0.1V
0101	1.4V +/- 0.1V
0100	1.3V +/- 0.1V
0011	1.2V +/- 0.1V
0010	1.1V +/- 0.1V
0001	1.0V +/- 0.1V
0000	0.9V +/- 0.1V

Note: The set level of the Internal Low Voltage Detection is only effective in case of Main Regulator is switched off (for decreasing the trigger level when decreasing the sub regulator voltage). Otherwise (with main regulator on) the default level is applied internally by hardware to the low voltage detection module (the register setting is not changed in this case and will be applied next time the main regulator is switched off).

3.Registers

Chapter 52 Regulator Control

1. Overview

- Module for controlling the behaviour of the MAIN-Regulator and SUB-Regulator in the device modes.

2. Features

- Main Regulator enable and disable independently for Sub-run and STOP/RTC
- Main regulator standby flag output
- Selectable Sub regulator output voltage for Sub-run and STOP/RTC

3. Registers

3.1 Regulator Control Registers

Controls the regulator function.

- **REGCTR: Address 04CFh (Access: Byte, Halfword, Word)**

7	6	5	4	3	2	1	0	bit
-	-	-	MSTBO	-	-	MAINKPEN	MAINDSBL	
X	X	X	X	X	X	0	0	Initial value (INIT pin input, watchdog reset)
X	X	X	X	X	X	X	X	Initial value (Software reset)
R0/WX	R0/WX	R0/WX	R	R0/WX	R0/WX	R/W	R/W	Attribute

(See “Meaning of Bit Attribute Symbols (Page No.10)” for details of the attributes.)

- Bit7-5: Reserved bit. The read value is always “0”.
- Bit4: Main regulator Standby output flag.

MSTBO	Function
0	Main regulator is in RUN mode
1	Main regulator is in STANDBY mode

- Bit3-2: Reserved bit. The read value is always “0”.
- Bit1: Main Regulator enable in STOP/RTC mode.

MAINKPEN	Function
0	Main regulator disabled in STOP/RTC mode [Initial value]
1	Main regulator enabled in STOP/RTC mode

- Bit0: Main Regulator disable in Subrun Mode.

MAINDSBL	Function
0	Main regulator enabled in Subrun mode [Initial value]
1	Main regulator disabled in Subrun mode

- **REGSEL: Address 04CEh (Access: Byte, Halfword, Word)**

7	6	5	4	3	2	1	0	bit
-	-	FLASHSEL	MAINSEL	SUBSEL3	SUBSEL2	SUBSEL1	SUBSEL0	
0	0	0	0	0	1	1	0	Initial value (INIT pin input, watchdog reset)
X	X	X	X	X	X	X	X	Initial value (Software reset)
R0/WX	R0/WX	R/W	R/W	R/W	R/W	R/W	R/W	Attribute

(See “Meaning of Bit Attribute Symbols (Page No.10)” for details of the attributes.)

- Bit7-6: Reserved bit. The read value is always “0”.
- Bit5: Flash memory supply mode.

FLASHSEL	Function
0	Flash memory operation mode is 1.8V [Initial value]
1	Flash memory operation mode is 1.9V

- Bit4: Main Regulator supply mode.

MAINSEL	Function
0	Main regulator operation mode is 1.8V [Initial value]
1	Main regulator operation mode is 1.9V

- Bit3-0: Sub-regulator voltage level

SUBSEL3-SUBSEL0	Voltage level
0111	1.9V +/- 0.1V
0110	1.8V +/- 0.1V (initial)
0101	1.7V +/- 0.1V
0100	1.6V +/- 0.1V
0011	1.5V +/- 0.1V
0010	1.4V +/- 0.1V
0001	1.3V +/- 0.1V
0000	1.2V +/- 0.1V

Note: The set level of the Sub-regulator voltage is only be effective in case of Main Regulator is switched off. Otherwise (with main regulator on) the default level is applied internally by hardware to the Sub-Regulator (the register setting is not changed in this case and will be applied next time the main regulator is switched off).

Chapter 53 Fixed Mode-Reset Vector / BOOT-ROM

1. Overview

The Boot ROM is a fixed start-up routine, which is located at memory addresses 0xB000 to 0xBFFF. The entry point 0xBFF8 is determined by the Fixed Reset Vector if the device is configured with the mode pins set to MD[2:0]=”000” (internal ROM/vector mode). In this mode MB91460 series devices use the Fixed Mode Vector data (FMV, address 0x0F:FFF8, data 0x06000000) and Fixed Reset Vector data (FRV, address 0x0F:FFFC, data 0x0000BFF8). The data of both these vectors are independent from the flash content at these two addresses.
The purpose of the Boot ROM is to configure the device after a reset and to provide a simple serial bootloader for programming the embedded flash memories.
Therefore it is executed always after the Reset Cancellation Sequence (see chapter 5.6 Reset Cancellation Sequence (Page No.147) or 7.6 Reset Cancellation Sequence (Page No.149)) of every INIT or RST reset.

2. Check for Boot Conditions

The check for boot conditions is slightly different for the evaluation chip MB91V460 (A-version) and the flash derivates (e.g. MB91F467DA).

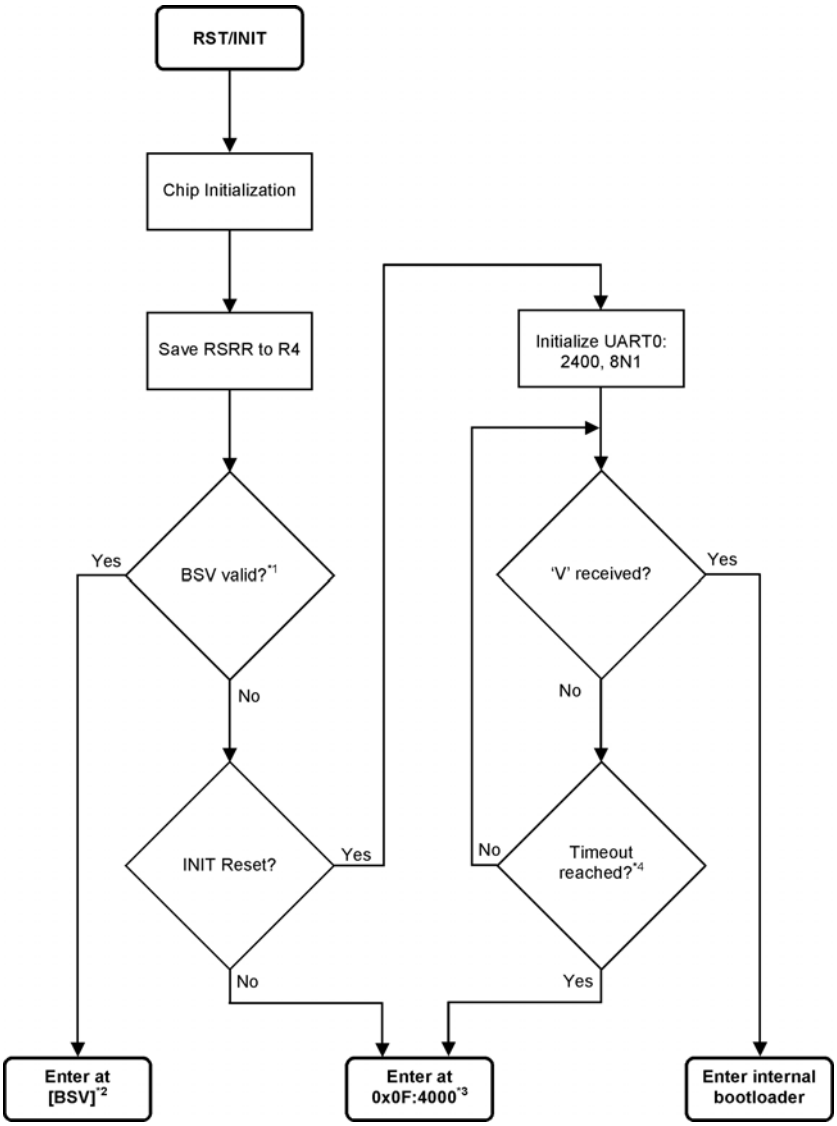
2.1 Evaluation Chip MB91V460

After the chip initialization and saving the RSRR (Reset Cause Register) to CPU register R4, the Boot Security Vector (BSV: Vector #144, 0x0F:FDBC) will be checked. This check is performed as follows: if the data of this vector represents a valid address in the specified address range, the Boot Security Vector itself becomes valid.

Device	Valid Boot Security Vector address range
MB91V460	0x04:0000 – 0x13:FFFF

If the Boot Security Vector is valid, the Boot ROM is left and the user application is started at the address given by this vector.
The purpose of this feature is to disable the execution of the internal bootloader due to security reasons or to minimize startup time of the application. Only if the user sets the Boot Security Vector to an address outside the given address range or leaves this vector at default content after erase (0xFFFF:FFFF) the internal bootloader can be entered.
If the Boot Security Vector is not valid, the reset cause will be checked as second boot condition. Only if the reset cause was an INIT reset (external INITX pin input, RSRR=0x80), the check for boot conditions will go on. Otherwise Boot ROM is left and application is started at default user program entry address 0x0F:4000.
If the reset cause was an INIT reset, UART0 is initialized: 2400 baud, 8 data bits, 1 stop bit, no parity. UART-reception is checked for about 100 ms. If during this time period the ASCII-character “V” (0x56) is received, the internal bootloader is entered. Otherwise Boot ROM is left and application is started at default user program entry address 0x0F:4000.

2.Check for Boot Conditions



*1) Boot-Security-Vector points to address in valid address range (0x04:000 – 0x13:FFFF)?
*2) Start user application at address given by Boot Security Vector
*3) Start user application at default user program entry address
*4) Timeout about 100 ms

Flow Chart of checking boot conditions on MB91V460

2.2 Flash devices of MB91460 series (MB91F46x)

After the chip initialization and saving the RSRR (Reset Cause Register) to CPU register R4, there is a check for boot conditions.
All Flash devices have two Boot Security Vectors (BSV1: 0x14:8004, BSV2: 0x14:800C). These vectors are located in parallel sector to the Flash Security Vectors (FSV1, FSV2):

	Sector SA4 (8kB)	Sector SA5 (8kB)
	…	…
0x14:8008	FSV2	**BSV2**
0x14:8000	FSV1	**BSV1**
	64bit width	

The Flash Security Vectors are used for configuring the protection mode of the flash memory sectors and do no influence startup of Boot ROM. Refer to Chapter 55 Flash Security (Page No.1011).

At first, BSV1 is checked: if the data of this vector represents a valid address in the specified address range (depending on Flash-ROM size), the Boot Security Vector itself becomes valid.

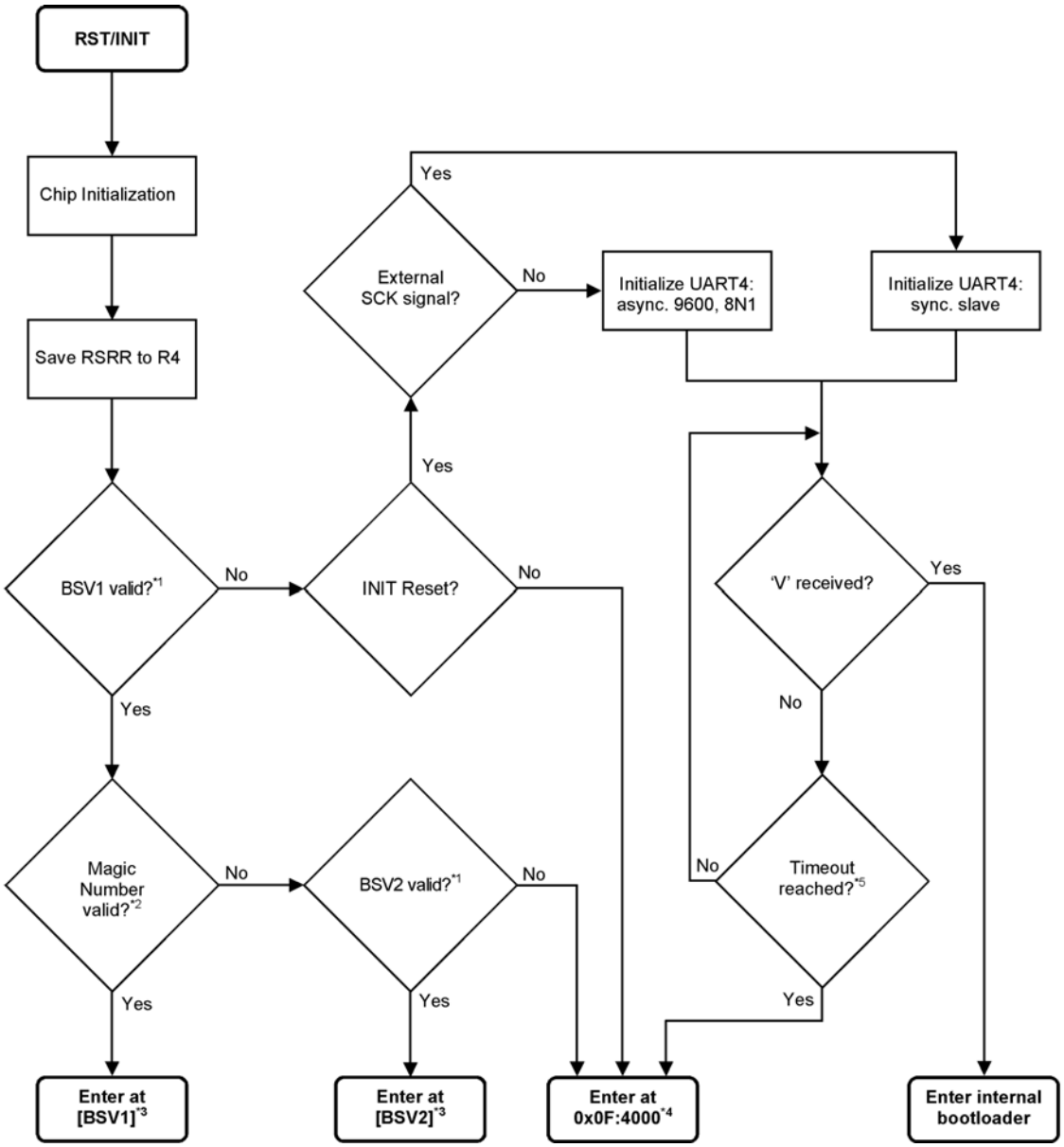
Device	Valid Boot Security Vector address range
MB91F467DA	0x04:0000 – 0x14:FFFF
MB91F469GA	0x04:0000 – 0x24:FFFF
MB91F464AA	0x0A:0000 – 0x0F:FFFF 0x14:8000 – 0x14:FFFF
MB91F465KA	0x08:0000 – 0x0F:FFFF 0x14:8000 – 0x14:FFFF

If BSV1 is valid, there will be an additional check before entering user program at the entry address given by BSV1 (1). Otherwise checks for entering the internal bootloader will be done (2).
The purpose of this feature is to disable the execution of the internal bootloader due to security reasons or to minimize startup time of application. If the user sets BSV1 to a valid address range, this bootloader cannot be entered any more.

(1) If the check for BSV1 is valid, the Magic Number, which should be located on the four bytes before the address BSV1 points to, is compared to **0x000A897A**. If the Magic Number matches this value, the user application is entered at the address given by BSV1.
The Magic Number is used as flag for a valid user application, or especially for a user bootloader. If you want to re-program this user bootloader, a second user bootloader, which handles the re-programming of the first user bootloader, has to be located at the address BSV2 points to. If BSV2 does not point to a valid address range, then application is started at default user program entry address 0x0F:4000. For more information on this bootloader update strategy refer to chapter 5. Bootloader Update Strategy (Page No.992).

(2) If the check for BSV1 is not valid, the reset cause will be checked as second boot condition. Only if the reset cause was an INIT reset (external INITX input, RSRR=0x80), the check for boot conditions will go on. Otherwise Boot ROM is left and application is started at default user program entry address 0x0F:4000.
If the reset cause was an INIT reset, serial clock pin SCK4 is checked for an external clock signal. Therefore logic level at this pin is monitored for about 1ms. If port level is constant, UART4 is initialized to asynchronous mode: 9600 baud, 8 data bits, 1 stop bit, no parity. If port level changes, UART4 is initialized to synchronous slave mode. UART-reception is checked for about 100 ms. If during this time period the ASCII-character “V” (0x56) is received,

the internal bootloader is entered. Otherwise Boot ROM is left and application is also started at default user program entry address 0x0F:4000.



*1) Boot Security Vector points to address in Flash-ROM
*2) Magic Number = 0x0A897A?
*3) Start user application at address given by Boot Security Vector 1/2
*4) Start user application at default user program entry address
*5) Timeout about 100 ms

Flow Chart of checking boot conditions on flash derivates of MB91460 series

2.3 Internal Bootloader Description

If a valid boot condition for entering the internal bootloader was met, the ASCII-character “F” (0x46) will be transmitted via UART0 (MB91V460) or UART4 (MB91460 series flash derivatives) to indicate that the bootloader is ready to accept commands.

There are five different commands supported by the internal bootloader. See the table below for serial protocol of these commands.

(Note) Software to transfer data using this protocol is available from Fujitsu.

Command	PC to MCU	MCU to PC	Remark
READ	1 (0x01) →	241 (0xF1)	
	2 (0x02) →	130 (0x82)	
	Address (4 bytes)		Lo, MidLo, MidHi, Hi
	Size (2 bytes)		Lo, Hi
	→	Binary Dump	Direct read and dump
		CheckSum (2 bytes)	Bootloader sends 16bit checksum
WRITE	1 (0x01) →	241 (0xF1)	
	3 (0x03) →	131 (0x83)	
	Address (4 bytes)		Lo, MidLo, MidHi, Hi
	Size (2 bytes)		Lo, Hi
	Binary Dump		Receive and store dump in RAM
	→	CheckSum (2 bytes)	Bootloader sends 16bit checksum
CALL	1 (0x01) →	241 (0xF1)	
	4 (0x04) →	132 (0x84)	
	Address (4 bytes)		Lo, MidLo, MidHi, Hi
		→	Calls specified Address and waits for a return
		Return Parameter	The returned parameter in R4 will be echoed to the PC
CHECKSUM	1 (0x01) →	241 (0xF1)	
	5 (0x05) →	133 (0x85)	
		CheckSum (2 bytes)	MCU re-dumps 16bit checksum (Lo, Hi) calculated at last write or read operation
BAUDRATE	1 (0x01) →	241 (0xF1)	
	6 (0x06) →	134 (0x86)	
	Baudrate (4 bytes)		Lo, MidLo, MidHi, Hi
			Initializes UART with new baudrate value

3. Registers modified by Boot ROM

The Boot ROM initializes the chip and changes the settings of some registers (see table below).

(Note) The RSRR register can only be read once. After reading the RSRR-register, the contents will be present in R4 (C-Compiler convention for parameters) after a branch to application start or the address specified by the Boot-Security-Vector.
The RSRR register is also saved in RAM. The RAM address depends on the device. The dependency of the RAM address and the device is listed below.

Device	RAM Address
MB91V460A	0x20500
MB91V467DA	0x28500
MB91V469GA	0x28500
MB91V464AA	0x2E500
MB91V465KA	0x2E500

3.1 Evaluation Chip MB91V460

Register	Value	Address	Description
TBCR	0x03	0x482	Time-Base Counter/Sync-RST Register
RSRR	0 (see note)	0x480	Reset Source Register (visible in R4)
TBR	0x0FFC00	-	Table Base Register
SP*	0x0203F8	-	Stack Pointer
PFR21*	0x07	0xD95	Port Function Register Port 21 (UART0)
SMR00*	0x05	0x041	UART0 Mode Register
SCR00*	0x17	0x040	UART0 Control Register
SSR00*		0x042	UART0 Serial Status Register
RDR00*		0x043	UART0 Reception Data Register
TRD00*		0x043	UART0 Transmission Data Register
BGR00*	0x0CF	0x080	UART0 Baud Rate/Reload Counter Register
TMCSR0*	0x0003	0x1B6	Reload Timer 0 Control Status Register
TMRLR0*	0x03E8	0x1B0	Reload Timer 0 Reload Register
TMR0*		0x1B2	Reload Timer 0 Timer Register

*(Note) * these registers will only be modified if a check for valid boot condition is performed*

3.2 Flash devices of MB91460 series

Register	Value	Address	Description
TBCR	0x03	0x482	Time-Base Counter/Sync-RST Register
RSRR	0 (see note)	0x480	Reset Source Register (visible in R4)
TBR	0x0FFC00	-	Table Base Register
SP*		-	Stack Pointer (depends on RAM size)

*(Note) * these registers will only be modified if a check for valid boot condition is performed*

4. Flash Access Mode Switching

On MB91460 series flash devices it is possible to switch between different flash access modes. These modes are depending on device type.

Device	16bit access	32bit access	64bit access
MB91F467DA	read/write	read/write	read
MB91F469GA	read/write	read/write	read
MB91F464AA	read/write	read	-
MB91F465KA	read/write	read	-

The 16bit mode is intended basically for flash programming, so program execution from flash is prohibited anyway. The 32bit and 64bit modes, however, are intended for program execution from flash. Since switching between these modes directly from an application located in the flash is prohibited it is necessary to locate the switching routine either in RAM or use the available routine in the Boot ROM:

This routine is located at address 0xBF60 for all MB91F46x devices. To call this function two parameters are expected in CPU registers R4 and R5:

R4: 2bits (LSB of register) for setting the flash access mode

b’00: 32bit read/write mode
b’01: 16bit read/write mode
b’1x: 64bit read only mode

R5: 16bits for setting wait time

The wait time should be in the range of some 100ns after switching the flash access mode for stabilization of the Flash-ROM. The wait function is a simple delay loop that needs about 3 CPU cycles for one run. The value in R5 represents the number of runs through this delay loop.

Example (time for 1 run through delay loop):

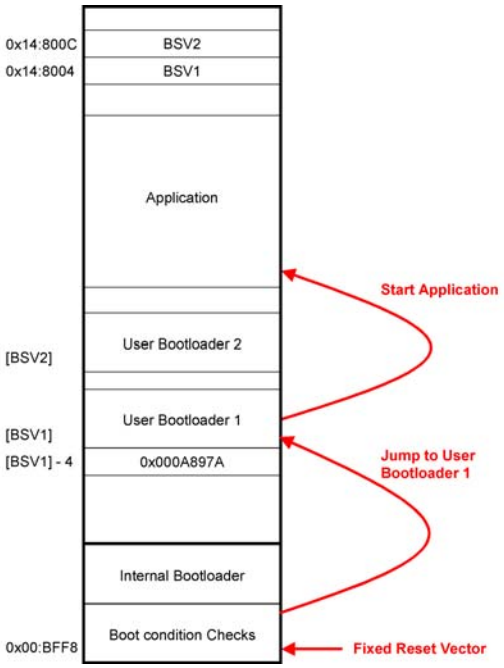
30 ns @ 100MHz CPU clock	min. 4-5 runs
47 ns @ 64MHz CPU clock	min. 3 runs
94 ns @ 32MHz CPU clock	min. 2 runs
500 ns @ 2MHz CPU clock	min. 1 runs

After leaving this delay loop, the switching function will return to the application.

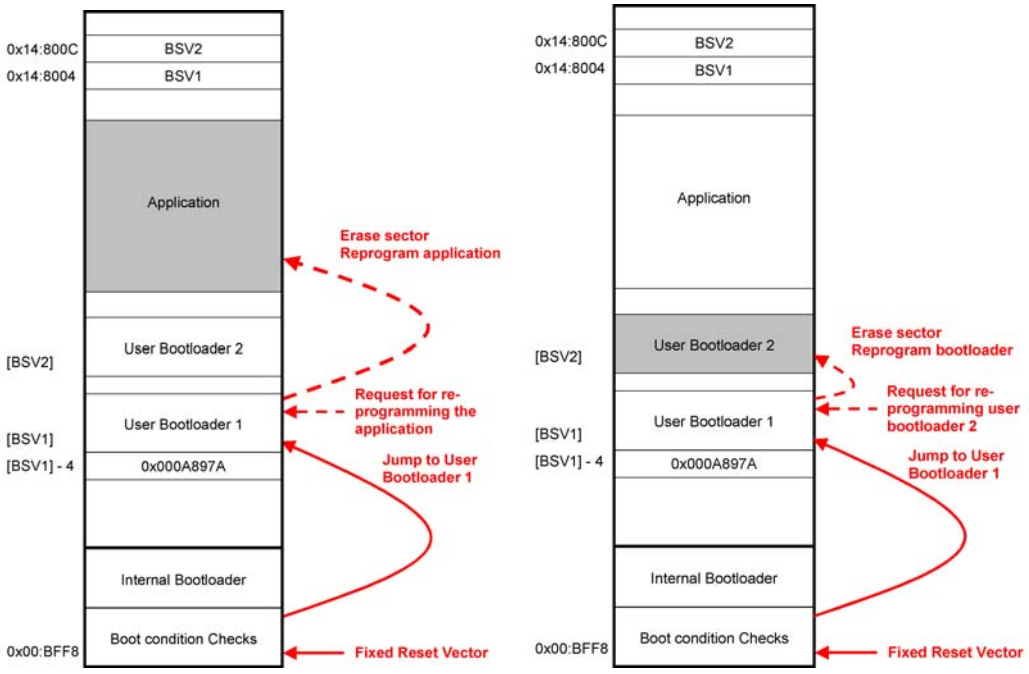
5. Bootloader Update Strategy

Some applications require the possibility of software updates in the final product without great effort. To be able to program an updated application to the microcontroller, all connected busses (I2C, CAN, LIN, K-Line etc.) can be used, so that the microcontroller can remain ‘embedded’ in a bigger system.
But therefore a user bootloader has to be integrated to the application software, which handles the startup of the application, the requests for new programming and the application update itself.
This user bootloader should be executed after every reset command, so the Boot Security Vector should point to the starting address of this bootloader. The request for re-programming the application has to be done by a defined communication protocol.
If the Boot Security Vector feature is used, there is also no possibility to enter the internal bootloader of the microcontroller. So the code is secured from read-out or manipulation.
If there is a request for updating an application, the user bootloader causes the erase of flash sectors where the application is located. Flash section, where the Boot Security Vector is located and the section of the bootloader itself may not be erased. After erasing, the bootloader has to handle the programming of the application.
If during this procedure problems like reset or power-down occur, the program-ming can be started again without problems because user bootloader and Boot Security Vector are not changed.
This system brings along some problems if the user bootloader has to be updated. Therefore also the user bootloader sector has to be erased and re-programmed. If there is a reset or power-down before ending the programming, there is no valid bootloader anymore and there will be no access to the microcontroller via the above mentioned busses.

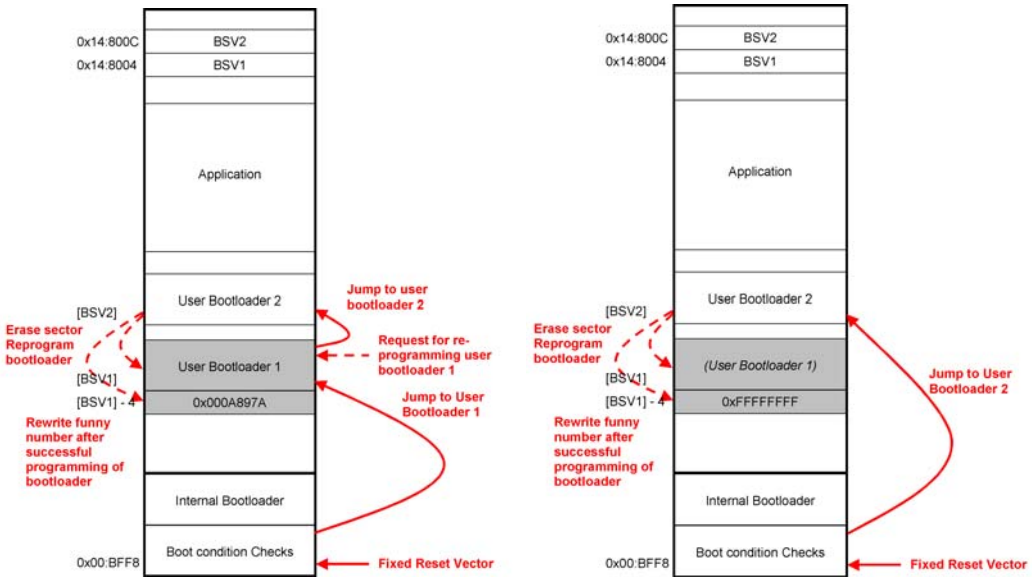
All MB91460 series flash devices therefore offer the possibility of a safe bootloader update by use of two Boot Security Vectors and a Magic Number.
If user bootloader 1 is valid, set the Magic Number, which has to be located on the four bytes before the address defined by BSV1, to 0x000A897A. So after every reset this bootloader is executed. If there is no request for a re-programming, the application is started normally.



By use of this bootloader the application as well as user bootloader 2 can be re-programmed.



If you want to re-program user bootloader 1, then functions in user bootloader 2 have to be called to erase the sector where user bootloader 1 and the Magic Number are located. After erasing, program the new user bootloader to the section. As last step, the Magic Number should be set to 0x000A897A.
This procedure guarantees that there is started always a valid bootloader. If after erasing or during programming user bootloader 1 a reset or a power-down occurs, user bootloader 2 is started and the programming can be resumed.



Chapter 54 Flash Memory

This chapter describes the use of the built-in flash memory.

1. Overview

The MB91F46x devices have built-in Flash memory with a variety of capacities, capable of batch-erasing all sectors or erasing on the sector level via single +3.0V-5.5V power supply, and writing by the FR-CPU at the half-word (16-bit) and word (32-bit) level.

2. Features

- Power: Single +3.0-5.5V supply
- Basic specification: Same as MBM29LV400TC (except size and part of sector configuration)
- Faster device operation by enabling commands/data reads at D-word (64-bit) level (not all devices)
- External writers: Interface for Parallel Flash Programmer available.
- Operation modes:
  (1) 64-bit CPU mode (not all devices)
  CPU reads and executes programs in word (32-bit) length units.
  Flash writing is not possible.
  Actual Flash Memory access is performed in d-word (64-bit) length units.
  (2) 32-bit CPU mode
  CPU reads, writes and executes programs in word (32-bit) length units.
  Actual Flash Memory access is performed in word (32-bit) length units.
  (3) 16-bit CPU mode:
  CPU reads and writes in half-word (16-bit) length units.
  Program exectuion from the Flash is not possible.
  Actual Flash Memory access is performed in word (16-bit) length units.
  (4) Flash memory mode (external access to Flash memory enabled)
- Features (Through combination of Flash memory macro and FR-CPU interface circuit):
  - Functions as CPU program/data storage memory.
  - Enables access to 32-bit bus width.
  - Enables read/write/erase by CPU (auto program algorithm*).
  - Functions equivalent to MBM29LV400TC stand-alone Flash-memory product.
  - Enables read/write/erase by parallel Flash programmer (auto program algorithm*).

*: Auto program algorithm = Embedded Algorithm TM

3. Configuration

Figure 3-1 Block Diagram (32bit flash)

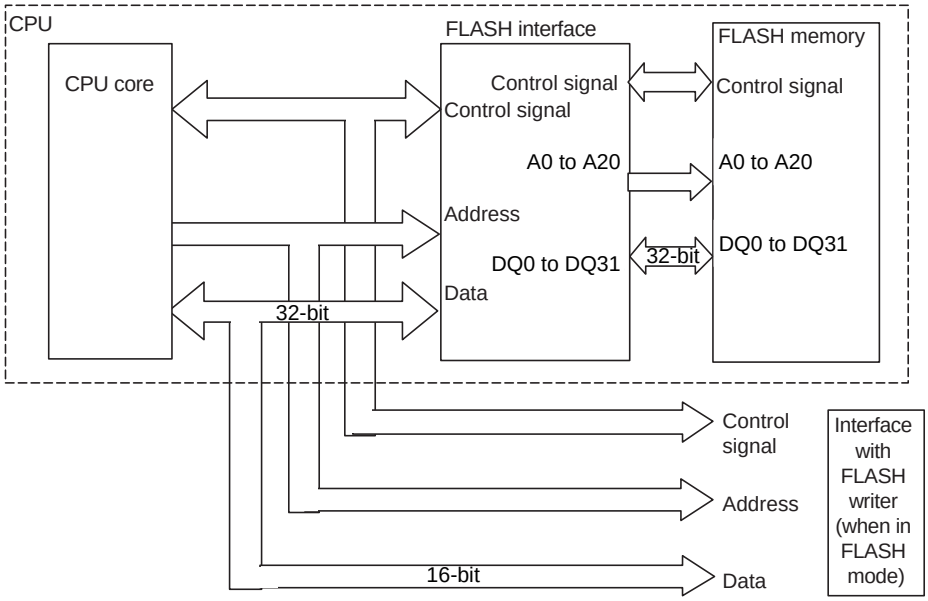
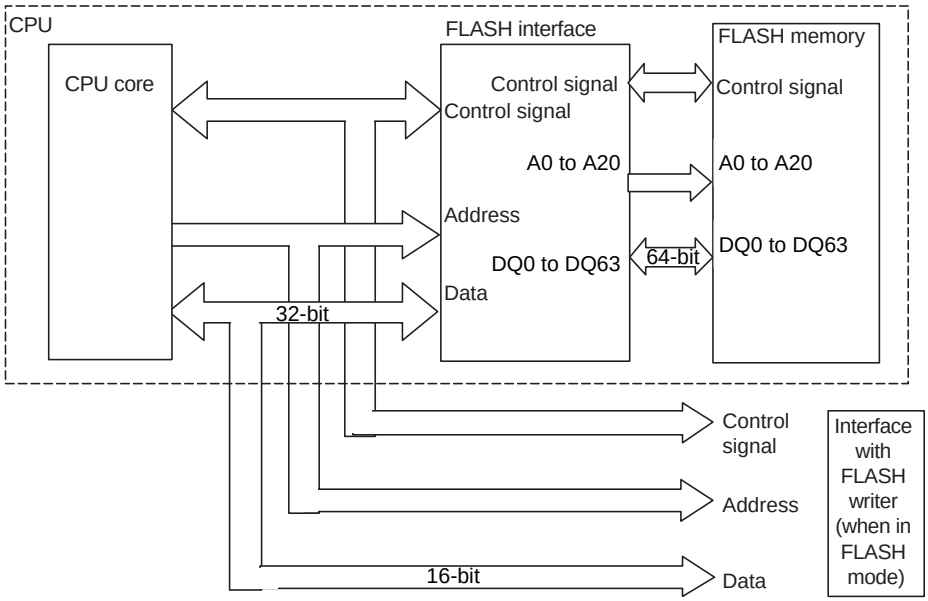


Figure 3-2 Block Diagram (64bit flash)



3.1 Address conversion from CPU Mode to Flash Programming Mode

Flash memory's address mapping is different depending on whether it is being accessed from the FR-CPU or parallel Flash programmer

The following equations are an example to calculate a Flash programming mode address (FA) from a CPU mode address (addr) for the 64bit flash type with a memory size of 1088kB (16x64kB + 8x8kB), as used on MB91F467DA:

SA0, SA2, SA4, SA6 (14:0000h ≤ addr ≤ 14:FFFFh; addr[2]=0):
FA = addr - addr%00:4000h + (addr%00:4000h)/2 - (addr/2)%4 + addr%4 - 05:0000h

SA1, SA3, SA5, SA7 (14:0000h ≤ addr ≤ 14:FFFFh; addr[2]=1):
FA = addr - addr%00:4000h + (addr%00:4000h)/2 - (addr/2)%4 + addr%4 - 04:E000h

SA8, SA10, SA12, SA14, SA16, SA18, SA20, SA22 (04:0000h ≤ addr ≤ 13:FFFFh; addr[2]=0):
FA = addr - addr%02:0000h + (addr%02:0000h)/2 - (addr/2)%4 + addr%4 + 0C:0000h

SA9, SA11, SA13, SA15, SA17, SA19, SA21, SA23 (04:0000h ≤ addr ≤ 13:FFFFh; addr[2]=1):
FA = addr - addr%02:0000h + (addr%02:0000h)/2 - (addr/2)%4 + addr%4 + 0D:0000h

For other flash memory sizes or types please see the related datasheets.

4. Registers

For the description of the Flash related registers, refer to Chapter 11 Memory Controller (Page No.167).

5. Access Modes

This section describes the Flash memory access modes.

5.1 Access from the FR-CPU

The following three types of access mode are available:

■ 64-bit CPU mode (read/execute), not for all devices

This mode does not allow data erase/write. Data can only be accessed by the CPU in lengths of words (32 bits) but is actually read from the Flash Memory in lengths of d-words (64 bits).

Programs can be executed in Flash memory while this mode is enabled.

- Specifying the mode

  Use the Flash setting procedure located in the Boot-ROM at address 0000:$BF60_H$ with R4 (2 bits) set to $1X_B$ to set this mode. Refer to Chapter 53 Fixed Mode-Reset Vector / BOOT-ROM (Page No.985).
- Description of operation

  When reading or executing code from the Flash memory area, data is read by the CPU in word (32-bit) length units but is actually read from memory in d-word (64-bit) length units.

  Running Auto Algorithms is not possible.

■ 32-bit CPU mode (read/write/execute), write not for all devices

This mode allows data erase/write. Data can only be accessed in lengths of words (32 bits).

Programs cannot be executed in Flash memory while the Flash is being written/erased.

- Specifying the mode

  Use the Flash setting procedure located in the Boot-ROM at address 0000:$BF60_H$ with R4 (2 bits) set to 00_B to set this mode. Refer to Chapter 53 Fixed Mode-Reset Vector / BOOT-ROM (Page No.985).

  Flash memory always goes to 32-bit mode after a reset is cleared, when the CPU is running.
- Description of operation

  When reading or executing code from the Flash memory area, data is read from memory in word (32-bit) length units.

  Auto Algorithms can be run by writing commands to Flash memory. It is possible to erase/write to Flash memory by running an Auto Algorithm. See “7. Auto Algorithms (Page No.1001)” for details about Auto Algorithms.

■ 16-bit CPU mode (read/write)

This mode allows data erase/write. Data can only be accessed in lengths of half-words (16 bits).

Programs cannot be executed in Flash memory while this mode is enabled.

- Specifying the mode

  Use the Flash setting procedure located in the Boot-ROM at address 0000:$BF60_H$ with R4 (2 bits) set to 01_B to set this mode. Refer to Chapter 53 Fixed Mode-Reset Vector / BOOT-ROM (Page No.985).
- Description of operation

  When reading from the Flash memory area, data is read from memory in half-word (16-bit) length units.

  Auto Algorithms can be run by writing commands to Flash memory. It is possible to erase/write the Flash memory by running an Auto Algorithm. See “7. Auto Algorithms (Page No.1001)” for details about Auto Algorithms.

6. Flash Access Mode Switching

On MB91460 series flash devices it is possible to switch between different flash access modes. These modes are depending on device type.

Device	16bit access	32bit access	64bit access
MB91F467DA	read/write	read/write	read
MB91F469GA	read/write	read/write	read
MB91F464AA	read/write	read	-
MB91F465KA	read/write	read	-

The 16bit mode is intended basically for flash programming, so program execution from flash is prohibited anyway. The 32bit and 64bit modes, however, are intended for program execution from flash. Since switching between these modes directly from an application located in the flash is prohibited it is necessary to locate the switching routine either in RAM or use the available routine in the Boot ROM:

This routine is located at address 0xBF60 for all MB91F46x devices. To call this function two parameters are expected in CPU registers R4 and R5:

R4: 2bits (LSB of register) for setting the flash access mode

b’00: 32bit read/write mode
b’01: 16bit read/write mode
b’1x: 64bit read only mode

R5: 16bits for setting wait time

The wait time should be in the range of some 100ns after switching the flash access mode for stabilization of the Flash-ROM. The wait function is a simple delay loop that needs about 3 CPU cycles for one run. The value in R5 represents the number of runs through this delay loop.

Example (time for 1 run through delay loop):

30 ns @ 100MHz CPU clock	min. 4-5 runs
47 ns @ 64MHz CPU clock	min. 3 runs
94 ns @ 32MHz CPU clock	min. 2 runs
500 ns @ 2MHz CPU clock	min. 1 runs

After leaving this delay loop, the switching function will return to the application.

6.1 Flash Memory Mode

Resetting after setting the MD2, MD1 and MD0 pins to “1”, “1” and “1” will halt the MCU functions. At this time, the Flash memory's interface circuit functions are emabled for direct control of the Flash memory unit from external pins, by directly linking some of the signals to the Flash memory unit's control signal. In this mode, the Flash memory appears to the external pins as a stand-alone unit. This mode is generally set when writing/ erasing using the parallel Flash programmer. In this mode, all operations of the Flash memory's Auto Algorithms are available.

For correspondence between the MCU pins and Flash memory’s interface circuit in the flash parallel programming mode please check the related datasheets of each product.

7. Auto Algorithms

Writes and erases to Flash memory are performed by launching the Flash memory's own Auto Algorithms.

7.1 Command Operation

Auto Algorithms are launched by writing one to six half words (16 bits) to the Flash memory in succession. This is called a “command.” Writing an illegal address or data, or writing them in the incorrect order, will reset the Flash memory to read mode.

Table 7-1 List of Commands in CPU mode

Command Sequence	Bus Write Cycle	1st Bus Write cycle		2nd Bus Write cycle		3rd Bus Write cycle		4th Bus Read/write cycle		5th Bus Write cycle		6th Bus Write cycle	
		Address	Data	Address	Data	Address	Data	Address	Data	Address	Data	Address	Data
Read/ reset	1	*XXXX$_H$	F0$_H$	---	---	---	---	---	---	---	---	---	---
Read/ Reset	4	*x557$_H$	AA$_H$	*yAAF$_H$	55$_H$	*x557$_H$	F0$_H$	RA	RD	---	---	---	---
Writing	4	*x557$_H$	AA$_H$	*yAAF$_H$	55$_H$	*x557$_H$	A0$_H$	PA	PD	---	---	---	---
Chip erase	6	*x557$_H$	AA$_H$	*yAAF$_H$	55$_H$	*x557$_H$	80$_H$	*x557$_H$	AA$_H$	*yAAF$_H$	55$_H$	*x557$_H$	10$_H$
Sector erase	6	*x557$_H$	AA$_H$	*yAAF$_H$	55$_H$	*x557$_H$	80$_H$	*x557$_H$	AA$_H$	*yAAF$_H$	55$_H$	SA	30$_H$
Sector erase suspend	Suspend erase during sector erase with input of Address= “*XXXX$_H$”, data = “B0$_H$”												
Sector erase resume	Resume erase after Sector erase suspend, with input of Address= “*XXXX$_H$”, data = “30$_H$”												
Continuous mode	3	*x557$_H$	AA$_H$	*yAAF$_H$	55$_H$	*x557$_H$	20$_H$	---	---	---	---	---	---
Continuous writing	2	*XXXX$_H$	A0$_H$	PA	PD	---	---	---	---	---	---	---	---
Continuous mode Reset	2	*XXXX$_H$	90$_H$	*XXXX$_H$	F0$_H$ or 00$_H$	---	---	---	---	---	---	---	---

Both word mode and half-word mode are the same for commands. Any data can be set in the non-specified bits.
RA: Read address
PA: Write address
SA: Sector address (Specify any address in the sector)
RD: Read data
PD: Write data
*: “0004”, “0005”, “0006”, “0007”, “0008”, “0009”, “000A”, “000B”, “000C”, “000D”, “000E”, “000F”, “0010”, “0011”, “0012”, “0013” or “0014”
x: Any odd hex digit: “1“, “3“, “5“, “7“, “9“, “B“, “D“ or “F“
y: Any even hex digit: “0“, “2“, “4“, “6“, “8“, “A“, “C“ or “E“

- Auto Algorithm Execution Status

  If an Auto Algorithm is started in CPU mode, it is possible to learn the operational state of the Auto Algorithm via the internal ready signal (RDY). The level of the ready signal can be read from the “RDY” bit of the FLASH Memory Control Status Register.

  While the “RDY” bit is set to “0”, data read is the hardware sequence flag indicating Flash memory status (see Hardware sequence flag in (3) and (4)).

7.2 Auto Algorithm Commands

■ Read/reset command

Issue a Read/reset command sequence to recover to read mode after a timing limit has been exceeded. Data is read from Flash memory via the read cycle. Flash memory stays in a read state until another command is input.

When powered up, Flash memory is automatically set to read/reset. In this case, commands are not required for data reading.

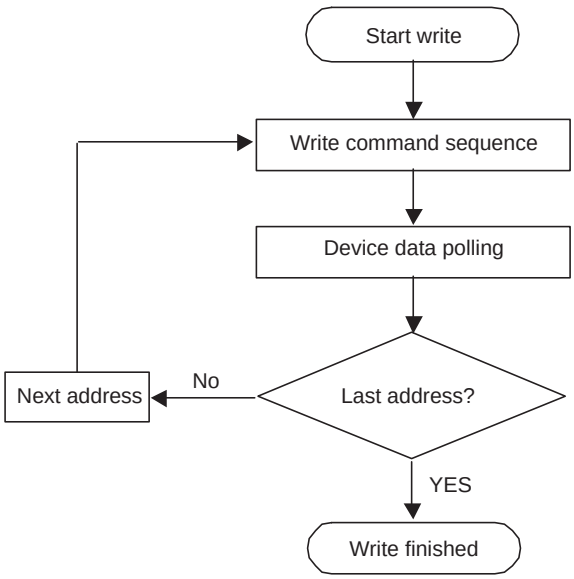
■ Program (write)

In CPU programming mode, writes are performed in basic units of half words. Writes are performed in 4 bus operations. The command sequence as two “unlock” cycles, followed by a write setup command and write data cycle. Then, in the final write cycle, writing to memory starts.

After the auto write algorithm command sequence is executed, Flash memory no longer requires external control. Flash memory generates appropriate write pulses that it has automatically created internally, and validates the margins of written cells. Auto write operation ends when the bit 7 data matches the data written to this bit via data polling (see (3) Hardware sequence flag). Flash memory then returns to read mode, and no longer accepts write addresses. As a result, at this time Flash memory requests the next valid address. Thus, data polling indicates that writing is ongoing.

During writing, all commands written to Flash memory are ignored. If a hardware reset is started during writing, the data in addresses that have been written is not guaranteed. Data can be written to addresses in any order, and may also cross sector boundaries. Writing cannot return data “0” to data “1”. If data “1” is written to data “0”, then either the data polling algorithm will determine that the device is bad, or it will appear that data “1” has been written, but in reset/read mode, when the data is read, it will still be read as “0”. Only erase operation can change “0” data to “1” data.

Figure 7-1 The Writing Sequence Using Write Commands



■ Chip erase

Chip erase (erase all sectors at once) is performed via six accesses. First, there are two “unlock” cycles, after which a setup command is written. This is then followed by two more “unlock” commands before the chip erase command.

The user does not have to write to Flash memory before a chip erase can be performed. While the auto erase algorithm is executing, Flash memory automatically validates each cell before erasing it by writing a “0” pattern (preprogram). During this operation, the Flash memory does not require external control.

Auto erase begins with a write during the command sequence, and ends when “1” is written to bit 7, at which point the Flash memory returns to read mode. The time for chip erase is equal to [sector erase time] x [number of sectors] + [chip write time (preprogram)].

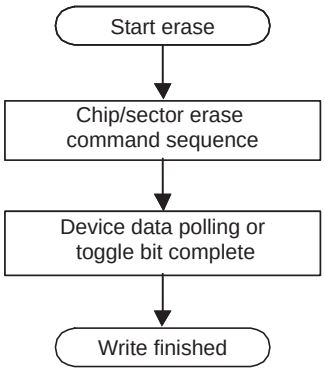
The figure below shows the chip erase sequence using the chip erase command.

■ Sector erase

Sector erase is performed via six accesses. There are two “unlock” cycles, after which a “setup” command is written, followed by another two “unlock” cycles. On the sixth cycle, a sector erase command is input, starting the sector erase. From the time that the last sector-erase command is written until the timeout of 50us, the next sector-erase command will be accepted.

It is possible to submit multiple sector erases simultaneously by writing the six bus cycles described above. This sequence is performed by writing the addresses of sectors to erase in succession after the sector-erase command (30_H). After a timeout of 50us since the last sector-erase command was written, sector erase begins. In other words, to erase multiple sectors simultaneously, each sector must be entered within 50us of the other, after which commands may no longer be accepted. It is possible to monitor whether successive sector-erase commands are valid via bit 3 (see (3) Hardware sequence flag). After finishing, Flash memory returns to read mode. Other commands are ignored. Data polling works on any address in an erased sector. The time for multiple-sector erase is equal to ([sector erase time] + [sector write time (preprogram)]) x [number of sectors erased].

Figure 7-2 Chip Erase Sequence Using the Chip erase Command



■ Erase suspend

The erase suspend command allows the user to pause the Flash memory’s Auto Algorithm during sector erase, and read data from/write data to sectors not being erased. This command is only valid during sector erase. It is ignored during chip erase and write operations. The erase suspend command ($B0_H$) is only valid during sector-erase operation, including the timeout period after a sector-erase command (30_H). Entering this command during the timeout period immediately ends the timeout, and interrupts the erase operation. When the erase resume command is written, the erase operation resumes. Any address can be used for erase suspend and erase resume command input.

If a erase suspend command is input during sector-erase operation, it will take up to 20us for the Flash memory to halt the erase operation. When the Flash memory goes into erase suspend mode, it outputs ready/ busy and bit 7 outputs “1”, and bit-6 toggling is halted. It is possible to confirm whether the erase operation

has halted, by entering the address of an erased sector, and monitoring the values read from bits 6 and 7. Additionally, writes of erase-suspend commands are ignored. When erasing is halted, the Flash memory goes into erase-suspend read mode. In this mode, data reads from sectors where erase has not been paused are enabled, but for other sectors, it is the same as standard reading. While in erase-suspend read mode, when data is read sequentially from an erase-suspended sector, bit 2 is toggled (see (3) Hardware sequence flag for details).

After entering erase-suspend read mode, the user can write to the Flash memory by writing a write command sequence. This mode is called “erase-suspend write mode”. In this mode, data writes to sectors where erase has not been paused are enabled, but for other sectors, it is the same as normal byte writing. While in erase-suspend write mode, when data is read sequentially from an erase-suspended sector, bit 2 is toggled. This mode can be detected via the erase-suspend bit (bit 6).

A word of caution is required for using this mode: Although bit 6 can be read from any address, bit 7 must be read from a write address. To resume sector erase, a resume command must be entered (30_H). At this point, further resume commands will be ignored. Conversely, it is possible to enter a erase-suspend command after the Flash memory resumes erasing.

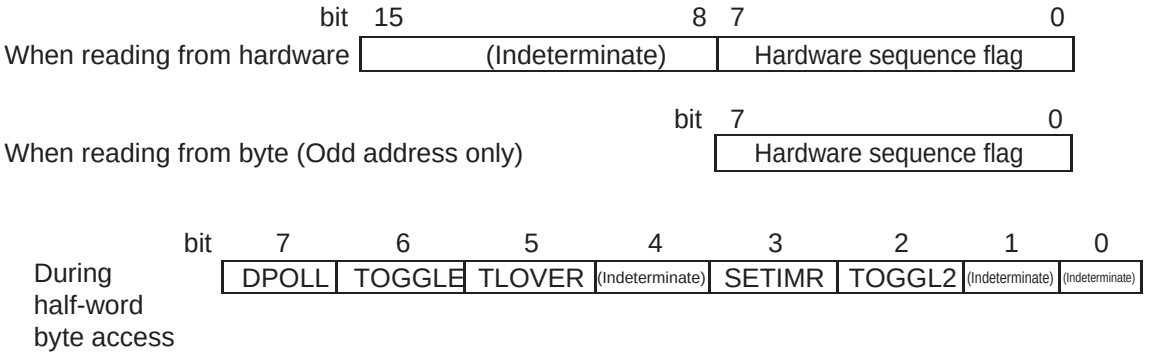
7.3 Hardware Sequence Flag

This Flash memory performs the write/erase sequence via Auto Algorithms. It thus has hardware for informing the outside world when it has finished internal operations.

Hardware sequence flag

The hardware sequence flag can be obtained as data by reading any address (an odd address during byte access) from the Flash memory while an Auto Algorithm is executing. Five of the retrieved bits of data are valid, each indicating the status of its corresponding Auto Algorithm.

Figure 7-3 Hardware Sequence Flag Format



Note that these flags are meaningless in FR-CPU ROM mode. Read this data as a half-word or byte, and only in FR-CPU programming mode.

Table 7-2 List of Hardware Sequence Flag States

Status			DPOLL (Bit 7)	TOGGLE (Bit 6)	TLOVER (Bit 5)	SETIMR (Bit 3)	TOGGL2 (Bit 2)
Executing	Auto write		Inverted data	Toggles	0	0	1
	Write/erase during auto-erase		0	Toggles	0	1	Toggles
	Erase suspended (paused)	Read (sectors being erased)	1	1	0	0	Toggles
		Read (sectors not being erased)	Data	Data	Data	Data	Data
		Write (sectors not being erased)	Inverted data	Toggles	0	0	1 (*1)
Time limit exceeded	Auto write		Inverted data	Toggles	1	0	1
	Write/erase during auto-erase		0	Toggles	1	1	(*2)

*1: During erase-suspend write mode, when an address that has been written to is read, bit 2 outputs logical "1".
When data is read sequentially from an erase-suspended sector, however, bit 2 is toggled.

*2: When bit 5 is set to "1" (time limit exceeded), sequential reads of sectors being written to/erased toggle bit 2, while reads from other sectors will not toggle bit 2.

- Ready/busy signal (RDY/BUSYX)

  In addition to the hardware sequence flag, the Flash memory has a ready/busy signal for indicating whether an internal Auto Algorithm is executing. This ready/busy signal can be connected to the Flash memory interface circuit, and read as the "RDY" bit of the FLASH Memory Control Status Register. Additionally, by starting up the ready/busy signal, it is possible to issue interrupt requests to the CPU
  (See "4. Registers (Page No.998)" for more information).

  Value "0" read from "RDY" bit: Flash memory is currently writing or erasing. At this time, write and erase commands are not accepted.

  Value "1" read from "RDY" bit: Flash memory is currently on standby for read/write or erase.

7.4 FLCR: Hardware Sequence Flag

• FLCR: Address. Any address in Flash memory. (Access: Byte or half-word)

7	6	5	4	3	2	1	0	bit
DPOLL	TOGGLE	TLOVER	-	SETIMR	TOGGL2	-	-	
-	-	-	-	-	-	-	-	Initial value
R	R	R	RX	R	R	RX	RX	Attribute

- bit 7: Data polling (DPOLL)
  - Auto write under way
    If data is read while the auto write algorithm is executing, the Flash memory outputs the data with the last value written to bit 7 inverted. If a read access is performed after the auto-write algorithm finishes, the Flash memory outputs the bit 7 of the address indicated by the address signal for the read data.
  - Auto erase under way
    If a read is performed while the auto-erase algorithm is executing, the Flash memory outputs “0”, regardless of the address indicated by the address signal. Similarly, upon termination it outputs “1”.
  - Sector erase suspend mode
    If a read is performed while in sector erase suspend mode, the Flash memory outputs “1” if the address indicated by the address signal belongs to a sector being erased. If it does not belong to the sector being erased, bit 7 of the value read from the address indicated by the address signal is output. Referring to this value while toggling bit 6 (described below) makes it possible to determine whether the current sector is in sector erase-suspend state, and which sectors are being erased.
  - Note that when the operation of an Auto Algorithm approaches termination, the value of bit 7 (data polling) changes asynchronously. This means that Flash memory sends the operating status to bit 7, then sends this data next. When the Flash memory terminates the Auto Algorithm, and also when it outputs data set in bit 7, the other bits are still undefined.
  - The defined data for other bits can be read by successfully executing reads.
- bit 6: Toggle bit (TOGGLE)
  - Auto write/erase under way
    If successive reads are performed while the auto-write or erase algorithm is executing, the Flash memory outputs the result of toggling between “1” and “0” to bit 6. After the auto write or erase algorithm terminates, bit 6 stops toggling in response to successive reads, and outputs valid data.
    The toggle bit becomes effective after the final write cycle of the command sequence in question.
    Note that when writing, if an attempt is made to write to an overwrite-protected sector, toggling is performed for about 2us, after which toggling ends; the data is not overwritten. When erasing, if all selected sectors are write-protected, the toggle bit toggles for about 100us, the returns to read mode; the data is not overwritten.
  - Sector erase suspend mode
    If a read is performed while in sector erase suspend mode, the Flash memory outputs “1” if the address indicated by the address signal belongs to a sector being erased. If it does not belong to the sector being erased, bit 6 of the value read from the address indicated by the address signal is output.
- bit 5: Timing limit exceeded (TLOVER)
  - Auto write/erase under way
    Bit 5 indicates that the Auto Algorithm has exceeded the length of time stipulated internally by the Flash memory (internal pulse count). In this state, bit 5 outputs “1”. In other words, if this flag outputs “1” while an Auto Algorithm is running, it means that the write or erase has failed.
    Attempts to write to bit 5, or an unerased non-flag area, will fail. When this happens, it will not be possible to read set data from bit 7 (data polling), and bit 6 (toggle bit) will remain toggled. If the time limit is exceeded while in this state, bit 5 outputs “1”. Note that this means that the Flash memory was not used properly; it does not mean that something is wrong with the Flash memory. If the Flash memory reaches this state, execute the reset command.

- bit 4: Undefined:
  The read value is indeterminate.
- bit 3: Sector erase timer (SETIMR)
  - During sector erase
    After executing the first sector-erase command sequence, Flash memory goes into standby for sector erase. During this time, bit 3 is “0”. After the sector-erase wait period ends, it outputs “1”. The data-polling and toggle bits become enabled after the first sector-erase command sequence is executed.
    If the data-polling and toggle-bit functions set this flag to “1” while the erase algorithm is executing, it indicates that internally controlled erasing has begun. Until the data-polling or toggle bit indicates that the erase is finished, subsequent command writes are ignored (only the erase-suspend code is accepted). If this flag is “0”, the Flash memory will accept additional sector erase-code writes. It is recommended that you check this flag via the software before writing subsequent sector-erase codes, in order to confirm this. If the flag is “1” on the second status check, it is possible that additional sector-erase codes will not be accepted. If a read is performed while in sector erase suspend mode, the Flash memory outputs “1” if the address indicated by the address signal belongs to an erased sector. If it does not belong to an erased sector, bit 3 of the value read from the address indicated by the address signal is output.
- bit 2: Toggle bit 2 (TOGGL2)
  - During sector erase
    This toggle bit is used in addition to the bit-6 toggle bit, in order to detect whether the Flash memory is performing auto erase, or erase is suspended. Bit 2 toggles if data is read repeatedly from an erased sector during auto erase. If Flash memory is in erase-suspend read mode, bit 2 will toggle if data is read repeatedly from a sector for which erase is suspended.
    If Flash memory is in erase-suspend write mode, “1” will be read from bit 2 if addresses are read repeatedly from a sector for which erase is not suspended. Unlike bit 2, bit 6 only toggles during normal writing and erasing, and erase-suspend write mode.
    For example, bits 2 and 6 are used together in order to detect erase-suspend read mode (bit 2 toggles, but bit 6 does not). Bit 2 is also used to detect erased sectors. When Flash memory is performing erase operation, this bit toggles if data is read from an erased sector.
- bit 1-0: Undefined
  The read value is indeterminate.

7.5 Sample Use of Hardware Sequence Flag

It is possible to determine the state of the Flash memory's internal Auto Algorithms using the hardware sequence flag mentioned above. As an example, the figures below show the write/erase determination sequence when the data-polling function is used, and when the toggle-bit function is used.

Figure 7-4 Write/erase Determination Sequence Using Data-polling Function

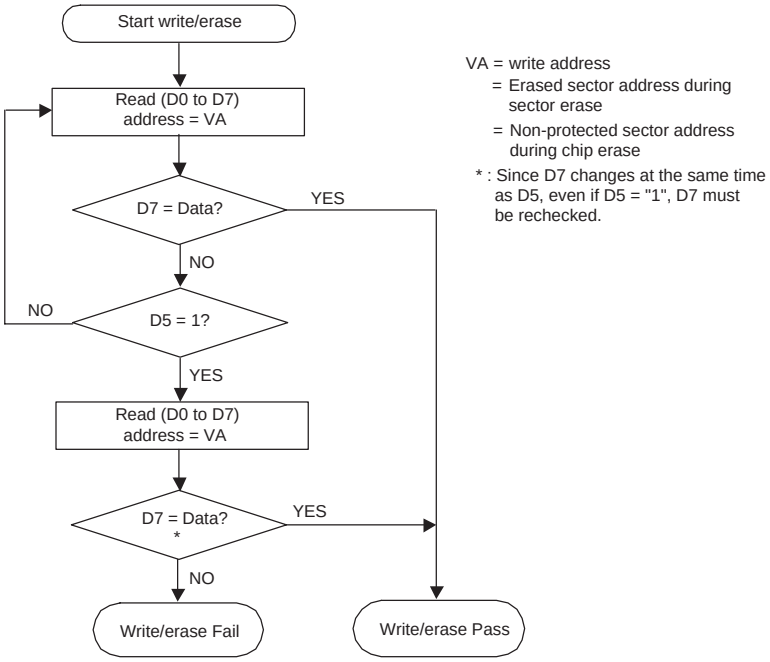
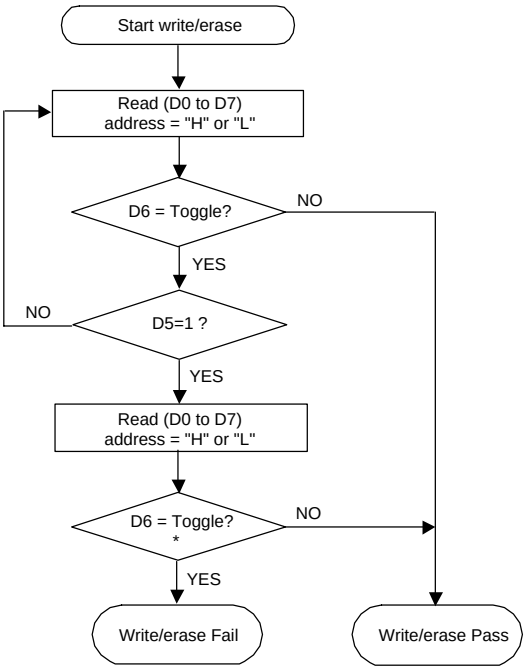


Figure 7-5 Write/erase Determination Sequence Using Toggle-bit Function



* When D5 changes to "1", D6 stops toggling, so even if D5 = "1", D6 must be rechecked.

8. Caution

- Please review the MBM29LV400TC data sheet in conjunction with this document.

- CPU mode

When in CPU mode, the address-allocation method is different than when writing via a parallel Flash programmer. Refer to “3.1 Address conversion from CPU Mode to Flash Programming Mode (Page No.997)”.

- Flash memory mode (writing via parallel Flash programmer)

  This Flash memory allows writing via an external device, by means of the parallel Flash programmer. In this state, pin functions equivalent to the stand-alone product MBM29LV400TC are assigned to the device's external pins, and operation of the CPU halts.
  In Flash memory mode, the address line connections are changed from CPU mode, to mapping within the memory area.

- See the section in your parallel Flash programmer manual concerning the MBM29LV400TC or MB91460 for details about using Flash memory from a parallel Flash programmer.

8.Caution

Chapter 55 Flash Security

1. Overview

- Module for controlling the read and write access protection to the embedded Flash memory

2. Features

- Common read protection for all flash sectors depending on device mode
- Individual write protection for each flash sector depending on device mode
- Write protection level depending on device mode
- Security vector re-fetch sequence (for security status update after chip erase)
- CRC checksum calculation (CRC32/AAL5 algorithm)
- CRC polygon is x^32+x^26+x^23+x^22+x^16+x^12+x^11+x^10+x^8+x^7+x^5+x^4+x^2+x+1

3. Flash Security Vectors

3.1 Vector addresses

Two Flash Security Vectors (FSV1, FSV2) are located parallel to the Boot Security Vectors (BSV1, BSV2) controlling the protection functions of the flash security module:

FSV1: 0x14:8000 BSV1: 0x14:8004
FSV2: 0x14:8008 BSV2: 0x14:800C

Remark: The addresses of both boot security vectors and flash security vectors depend on the size and the data width configuration of the embedded flash memory. Therefore always check the appropriate datasheet if the addresses shown here are valid for the product you are using.

3.2 Security Vector FSV1

The setting of the Flash Security Vector FSV1 is responsible for the read and write protection modes and the individual write protection of the 8 kByte sectors.

■ FSV1 (bits 31 to 16)

The setting of the Flash Security Vector FSV1 bits [31:16] is responsible for the read and write protection modes.

Explanation of the bits in the Flash Security Vector FSV1[31:16]:

FSV1[31:19]	FSV1[18] Write Protection Level	FSV1[17] Write Protection	FSV1[16] Read Protection	Flash Security Mode
set all to ‘0’	set to ‘0’	set to ‘0’	set to ‘1’	Read Protection (all device modes, except INTVEC mode MD[2:0]=”000”)
set all to ‘0’	set to ‘0’	set to ‘1’	set to ‘0’	Write Protection (all device modes, without exception)
set all to ‘0’	set to ‘0’	set to ‘1’	set to ‘1’	Read Protection (all device modes, except INTVEC mode MD[2:0]=”000”) and Write Protection (all device modes)
set all to ‘0’	set to ‘1’	set to ‘0’	set to ‘1’	Read Protection (all device modes, except INTVEC mode MD[2:0]=”000”)
set all to ‘0’	set to ‘1’	set to ‘1’	set to ‘0’	Write Protection (all device modes, except INTVEC mode MD[2:0]=”000”)
set all to ‘0’	set to ‘1’	set to ‘1’	set to ‘1’	Read Protection (all device modes, except INTVEC mode MD[2:0]=”000”) and Write Protection (all device modes except INTVEC mode MD[2:0]=”000”)

■ FSV1 (bits 15 to 0)

The setting of the Flash Security Vector FSV1 bits [15:0] is responsible for the individual write protection of the 8 kByte sectors. It is only evaluated if write protection bit FSV1[17] is set.

Explanation of the bits in the Flash Security Vector FSV1[15:0] (valid for MB91F467DA):

FSV1 bit	Sector	Enable Write Protection	Disable Write Protection	Comment
FSV1[0]	SA0	set to ‘0’	set to ‘1’	
FSV1[1]	SA1	set to ‘0’	set to ‘1’	
FSV1[2]	SA2	set to ‘0’	set to ‘1’	
FSV1[3]	SA3	set to ‘0’	set to ‘1’	
FSV1[4]	SA4	set to ‘0’	-	**Write protection is mandatory !**
FSV1[5]	SA5	set to ‘0’	set to ‘1’	
FSV1[6]	SA6	set to ‘0’	set to ‘1’	
FSV1[7]	SA7	set to ‘0’	set to ‘1’	
FSV1[8]	-	set to ‘0’	set to ‘1’	not available
FSV1[9]	-	set to ‘0’	set to ‘1’	not available
FSV1[10]	-	set to ‘0’	set to ‘1’	not available
FSV1[11]	-	set to ‘0’	set to ‘1’	not available
FSV1[12]	-	set to ‘0’	set to ‘1’	not available
FSV1[13]	-	set to ‘0’	set to ‘1’	not available
FSV1[14]	-	set to ‘0’	set to ‘1’	not available
FSV1[15]	-	set to ‘0’	set to ‘1’	not available

Remark: It is mandatory to always set the sector where the Flash Security Vectors FSV1 and FSV2 are located to write protected (here sector SA4). Otherwise it is possible to overwrite the Security Vector to a setting where it is possible to either read out the flash content or manipulate data by writing.

Remark: The content of the security vector FSV1[15:0] depends on the size and the data width configuration of the embedded flash memory. Therefore always check the appropriate datasheet if the data shown here are valid for the product you are using.

See Chapter 54 Flash Memory (Page No.995) for an overview about the sector organisation of the Flash Memory.

3.3 Security Vector FSV2

The setting of the Flash Security Vector FSV2 bits [31:0] is responsible for the individual write protection of the 64 kByte sectors. It is only evaluated if write protection bit FSV1[17] is set.

Explanation of the bits in the Flash Security Vector FSV2[31:0] (valid for MB91F467DA):

FSV1 bit	Sector	Enable Write Protection	Disable Write Protection	Comment
FSV2[0]	SA8	set to ‘0’	set to ‘1’	
FSV2[1]	SA9	set to ‘0’	set to ‘1’	
FSV2[2]	SA10	set to ‘0’	set to ‘1’	
FSV2[3]	SA11	set to ‘0’	set to ‘1’	
FSV2[4]	SA12	set to ‘0’	set to ‘1’	
FSV2[5]	SA13	set to ‘0’	set to ‘1’	
FSV2[6]	SA14	set to ‘0’	set to ‘1’	
FSV2[7]	SA15	set to ‘0’	set to ‘1’	
FSV2[8]	SA16	set to ‘0’	set to ‘1’	
FSV2[9]	SA17	set to ‘0’	set to ‘1’	
FSV2[10]	SA18	set to ‘0’	set to ‘1’	
FSV2[11]	SA19	set to ‘0’	set to ‘1’	
FSV2[12]	SA20	set to ‘0’	set to ‘1’	
FSV2[13]	SA21	set to ‘0’	set to ‘1’	
FSV2[14]	SA22	set to ‘0’	set to ‘1’	
FSV2[15]	SA23	set to ‘0’	set to ‘1’	
FSV2[16]	-	set to ‘0’	set to ‘1’	not available
FSV2[17]	-	set to ‘0’	set to ‘1’	not available
FSV2[18]	-	set to ‘0’	set to ‘1’	not available
FSV2[19]	-	set to ‘0’	set to ‘1’	not available
FSV2[20]	-	set to ‘0’	set to ‘1’	not available
FSV2[21]	-	set to ‘0’	set to ‘1’	not available
FSV2[22]	-	set to ‘0’	set to ‘1’	not available
FSV2[23]	-	set to ‘0’	set to ‘1’	not available
FSV2[24]	-	set to ‘0’	set to ‘1’	not available
FSV2[25]	-	set to ‘0’	set to ‘1’	not available
FSV2[26]	-	set to ‘0’	set to ‘1’	not available
FSV2[27]	-	set to ‘0’	set to ‘1’	not available
FSV2[28]	-	set to ‘0’	set to ‘1’	not available
FSV2[29]	-	set to ‘0’	set to ‘1’	not available
FSV2[30]	-	set to ‘0’	set to ‘1’	not available
FSV2[31]	-	set to ‘0’	set to ‘1’	not available

Remark: The content of the security vector FSV2[31:0] depends on the size and the data width configuration of the embedded flash memory. Therefore always check the appropriate datasheet if the data shown here are valid for the product you are using.

See Chapter 54 Flash Memory (Page No.995) for an overview about the sector organisation of the Flash Memory.

4. Register

4.1 Flash Security Control Register

■ FSCR0: Address 7100h (Access: Byte (write), Word (read))

31	30	29	28	27	26	25	24	bit
CRC31/ S7	CRC30/ S6	CRC29/ S5	CRC28/ S4	CRC27/ S3	CRC26/ S2	CRC25/ S1	CRC24/ S0	
1	1	1	1	1	1	1	1	Initial value (INIT pin input, watchdog reset)
X	X	X	X	X	X	X	X	Initial value (Software reset)
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Attribute

23	22	21	20	19	18	17	16	bit
CRC23	CRC22	CRC21	CRC20	CRC19	CRC18	CRC17	CRC16	
1	1	1	1	1	1	1	1	Initial value (INIT pin input, watchdog reset)
X	X	X	X	X	X	X	X	Initial value (Software reset)
R	R	R	R	R	R	R	R	Attribute

15	14	13	12	11	10	9	8	bit
CRC15	CRC14	CRC13	CRC12	CRC11	CRC10	CRC9	CRC8	
1	1	1	1	1	1	1	1	Initial value (INIT pin input, watchdog reset)
X	X	X	X	X	X	X	X	Initial value (Software reset)
R	R	R	R	R	R	R	R	Attribute

7	6	5	4	3	2	1	0	bit
CRC7	CRC6	CRC5	CRC4	CRC3	CRC2	CRC1	CRC0	
1	1	1	1	1	1	1	1	Initial value (INIT pin input, watchdog reset)
X	X	X	X	X	X	X	X	Initial value (Software reset)
R	R	R	R	R	R	R	R	Attribute

(See “Meaning of Bit Attribute Symbols (Page No.10)” for details of the attributes.)

- **Bit31-24**: Sequence activation

S7-S0	Function
0xA5 -> 0x5A	Start of a Flash Security Vector Re-Fetch Sequence (write only)
0xF0 -> 0X0F	Start of a Flash Memory CRC32 Checksum Sequence (write only)

Continuously writing “$A5_H$”, “$5A_H$” in the FSCR0[31:24] register will start a Flash Security Vector Re-fetch sequence immediately after writing “$5A_H$”. There is no time restrictions between “$A5_H$” and “$5A_H$”, but if “$A5_H$” is written followed by the one other than “$5A_H$”, you should write “$A5_H$” again. If not, the Re-Fetch sequence cannot be started even if “$5A_H$” is written.

Continuously writing “$F0_H$”, “$0F_H$” in the FSCR0[31:24] register will start a CRC32 checksum sequence immediately after writing “$0F_H$”. There is no time restrictions between “$F0_H$” and “$0F_H$”, but if “$F0_H$” is written followed by the one other than “$0F_H$”, you should write “$F0_H$” again. If not, the CRC checksum sequence cannot be started even if “$0F_H$” is written.

Remark: The Flash Security Vector Re-Fetch sequence is especially intented to be used after a chip erase command to update the security status without the need of applying an external INITX reset or after changing the status of the FSV1 security vector.

Remark: For both Security Vector Re-Fetch and CRC32 it is not recommended to start these sequences from programs located in the Flash Memory itself.

Remark: Before starting the CRC32 checksum it is recommended to set the RC oscillation in the 2MHz operation mode to avoid long runtimes. See the RCSEL bit in Chapter 4.1 Clock Monitor Configuration Register (Page No.945).

- **Bit31-0**: CRC32 checksum result

CRC31-CRC0	Function
	CRC32 Checksum (read only)

This register contains the CRC32 checksum result after completion of the CRC32 checksum sequence (the sequence completion is indicated by FSCR1.RDY). The CRC checksum is calculated in a standard CRC32/ AAL5 algorithm with the polygon x^32+x^26+x^23+x^22+x^16+x^12+x^11+x^10+x^8+x^7+x^5+x^4+x^2+x+1.

■ FSCR1: Address 7104h (Access: Byte (read), Word (write))

31	30	29	28	27	26	25	24	bit
-	-	-	-	-	-	-	RDY	
0	0	0	0	0	0	0	0	Initial value (INIT pin input, watchdog reset)
X	X	X	X	X	X	X	X	Initial value (Software reset)
RX/WX	RX/WX	RX/WX	RX/WX	R0/WX	R0/WX	R0/WX	R	Attribute
23	22	21	20	19	18	17	16	bit
-	-	-	-	CSZ3	CSZ2	CSZ1	CSZ0	
0	0	0	0	0	0	0	0	Initial value (INIT pin input, watchdog reset)
X	X	X	X	X	X	X	X	Initial value (Software reset)
R0/WX	R0/WX	R0/WX	R0/WX	R/W	R/W	R/W	R/W	Attribute
15	14	13	12	11	10	9	8	bit
CSA15	CSA14	CSA13	CSA12	CSA11	CSA10	CSA9	CSA8	
0	0	0	0	0	0	0	0	Initial value (INIT pin input, watchdog reset)
X	X	X	X	X	X	X	X	Initial value (Software reset)
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Attribute
7	6	5	4	3	2	1	0	bit
CSA7	CSA6	CSA5	CSA4	CSA3	CSA2	CSA1	CSA0	
0	0	0	0	0	0	0	0	Initial value (INIT pin input, watchdog reset)
X	X	X	X	X	X	X	X	Initial value (Software reset)
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Attribute

(See “Meaning of Bit Attribute Symbols (Page No.10)” for details of the attributes.)

- **Bit31-25**: Reserved bit. The read value is always “0”.
- **Bit24**: RDY: CRC32 Sequence Ready

RDY	Function
0	CRC32 sequence running or not yet started
1	CRC32 sequence ready (data in the FSCR0 register is valid)

- **Bit23-20**: Reserved bit. The read value is always “0”.
- **Bit19-16**: CSZ3-0: CRC32 Size Mask

CSZ3-0	Function
0000	CRC32 sequence size mask is 4 kByte
0001	CRC32 sequence size mask is 8 kByte
0010	CRC32 sequence size mask is 16 kByte
0011	CRC32 sequence size mask is 32 kByte
0100	CRC32 sequence size mask is 64 kByte
0101	CRC32 sequence size mask is 128 kByte
0110	CRC32 sequence size mask is 256 kByte
0111	CRC32 sequence size mask is 512 kByte
1000	CRC32 sequence size mask is 1024 kByte
1001	CRC32 sequence size mask is 2048 kByte
1010	CRC32 sequence size mask is 4096 kByte
1011-1111	Not supported

Remark: CSZ3-0 is used as an OR-mask for the address given by CSA15-0. See address calculation below.

- **Bit15-0**: CSA15-0: CRC32 Start Address

This register contains the CRC32 startaddress which is aligned to 4kByte addresses. It is only possible to calculate the CRC32 checksum over addresses located in the Flash Memory address space. Other addresses are invalid and might lead to wrong checksums.

Remark: The addresses to be written in this register are flash memory addresses like used in the flash parallel programming mode and not the mapped addresses which are used in CPU mode.

■ Calculation of the CRC32 start- and end-address:

The CSZ3-0 setting is first translated into a mask value:

CSZ3-0	MASK
0000	0000_0000_0000_0000
0001	0000_0000_0000_0001
0010	0000_0000_0000_0011
0011	0000_0000_0000_0111
0100	0000_0000_0000_1111
0101	0000_0000_0001_1111
0110	0000_0000_0011_1111
0111	0000_0000_0111_1111
1000	0000_0000_1111_1111
1001-1111	and so on...

CRC32 Startaddress = CSA[15:0] << 12 + 0x000

CRC32 Endaddress = (CSA[15:0] or MASK) << 12 + 0xFFF

4.Register

Chapter 56 Electrical Specification

See the appropriate data sheet for the electrical specification of each device.

MB91460 Series Hardware Manual

European Microcontroller Design Centre
Author : MBo

CM71-xxxxx-1E

FUJITSU SEMICONDUCTOR • CONTROLLER MANUAL

FR60

32-BIT MICROCONTROLLER

MB91460 Series

User's Manual

Jan 2005 the zero edition

Published **FUJITSU LIMITED** Electronic Devices

Edited European Microcontroller Design Centre