User's Manual

μPD780228 Subseries

8-bit Single-chip Microcontroller

μPD780226
μPD780228
μPD78F0228

Document No. U12012EJ2V0UM00 (2nd edition)
Date Published June 1997 N
Printed in Japan

[MEMO]

NOTES FOR CMOS DEVICES

① PRECAUTION AGAINST ESD FOR SEMICONDUCTORS

Note:

Strong electric field, when exposed to a MOS device, can cause destruction of the gate oxide and ultimately degrade the device operation. Steps must be taken to stop generation of static electricity as much as possible, and quickly dissipate it once, when it has occurred. Environmental control must be adequate. When it is dry, humidifier should be used. It is recommended to avoid using insulators that easily build static electricity. Semiconductor devices must be stored and transported in an anti-static container, static shielding bag or conductive material. All test and measurement tools including work bench and floor should be grounded. The operator should be grounded using wrist strap. Semiconductor devices must not be touched with bare hands. Similar precautions need to be taken for PW boards with semiconductor devices on it.

② HANDLING OF UNUSED INPUT PINS FOR CMOS

Note:

No connection for CMOS device inputs can be cause of malfunction. If no connection is provided to the input pins, it is possible that an internal input level may be generated due to noise, etc., hence causing malfunction. CMOS devices behave differently than Bipolar or NMOS devices. Input levels of CMOS devices must be fixed high or low by using a pull-up or pull-down circuitry. Each unused pin should be connected to V_{DD} or GND with a resistor, if it is considered to have a possibility of being an output pin. All handling related to the unused pins must be judged device by device and related specifications governing the devices.

③ STATUS BEFORE INITIALIZATION OF MOS DEVICES

Note:

Power-on does not necessarily define initial status of MOS device. Production process of MOS does not define the initial operation status of the device. Immediately after the power source is turned ON, the devices with reset function have not yet been initialized. Hence, power-on does not guarantee out-pin levels, I/O settings or contents of registers. Device is not initialized until the reset signal is received. Reset operation must be executed immediately after power-on for devices having reset function.

The export of these products from Japan is regulated by the Japanese government. The export of some or all of these products may be prohibited without governmental license. To export or re-export some or all of these products from a country other than Japan may also be prohibited without a license from that country. Please call an NEC sales representative.

License not needed: μPD78F0228GF-3BA

The customer must judge the need for license: μPD780226GF-×××-3BA, 780228GF-×××-3BA

Major Revisions in This Edition

Page	Description
Throughout	Change of μPD780226 and 780228 from “under development” to “developed”
p.16	**2.3 I/O Circuits of Pins and Connections of Unused Pins** Change of I/O circuit types of ports 7, 8, 9, and 10 of mask ROM model as follows: Ports 7, 8, and 9: type 15-D to type 15-F Port 10 : type 14-D to type 14-F
p.33, 173, 175, 177	Addition of internal expansion RAM size select register (IXS)
p.59-62	Correction of block diagrams of ports 7, 8, 9, and 10
p.71	**5.4.1 Main system clock oscillator** Change of notes on inputting external clock
p.78	Change of count clock value by setting of TCL2 and TCL1 in **Figure 6-2 Timer Mode Control Register 1 Format**
p.103, 164	Change of oscillation stabilization time by setting of OSTS2-OSTS0 in Format of Oscillation Stabilization Time Select Register
p.130	**Figure 11-2 Display Mode Register 0 Format** Change of setting of FOUT5-FOUT0, and addition of notes on FIP output pins
p.139	Addition of **11.7 Calculation of Total Power Dissipation**
p.173, 175, 202	Change of product name of dedicated flash writer as follows: Flashpro to Falshpro II (part No.: FL-PR2)
p.197	Addition of **APPENDIX A DIFFERENCES BETWEEN μPD78044H, 780228, AND 780208 SUBSERIES**
p.203	Change of hardware for debugging tools as follows: Change of IE-780000-SL to IE-78001-R-A Deletion of IE-78K0-SL-EM (CPU core board) Change of EP-100GF-SL from “under development” to “developed”
Previous edition	Deletion of **“Upgrading Your In-Circuit Emulator to Emulator for 78K/0 Series”**
p.217	Addition of **APPENDIX E REVISION HISTORY**

The mark ★ shows major revised points.

Regional Information

Some information contained in this document may vary from country to country. Before using any NEC product in your application, please contact the NEC office in your country to obtain a list of authorized representatives and distributors. They will verify:

- Device availability
- Ordering information
- Product release schedule
- Availability of related technical literature
- Development environment specifications (for example, specifications for third-party tools and components, host computers, power plugs, AC supply voltages, and so forth)
- Network requirements

In addition, trademarks, registered trademarks, export restrictions, and other legal issues may also vary from country to country.

NEC Electronics Inc. (U.S.)
Santa Clara, California
Tel: 800-366-9782
Fax: 800-729-9288

NEC Electronics (Germany) GmbH
Duesseldorf, Germany
Tel: 0211-65 03 02
Fax: 0211-65 03 490

NEC Electronics (UK) Ltd.
Milton Keynes, UK
Tel: 01908-691-133
Fax: 01908-670-290

NEC Electronics Italiana s.r.1.
Milano, Italy
Tel: 02-66 75 41
Fax: 02-66 75 42 99

NEC Electronics (Germany) GmbH
Benelux Office
Eindhoven, The Netherlands
Tel: 040-2445845
Fax: 040-2444580

NEC Electronics (France) S.A.
Velizy-Villacoublay, France
Tel: 01-30-67 58 00
Fax: 01-30-67 58 99

NEC Electronics (France) S.A.
Spain Office
Madrid, Spain
Tel: 01-504-2787
Fax: 01-504-2860

NEC Electronics (Germany) GmbH
Scandinavia Office
Taeby, Sweden
Tel: 08-63 80 820
Fax: 08-63 80 388

NEC Electronics Hong Kong Ltd.
Hong Kong
Tel: 2886-9318
Fax: 2886-9022/9044

NEC Electronics Hong Kong Ltd.
Seoul Branch
Seoul, Korea
Tel: 02-528-0303
Fax: 02-528-4411

NEC Electronics Singapore Pte. Ltd.
United Square, Singapore 1130
Tel: 253-8311
Fax: 250-3583

NEC Electronics Taiwan Ltd.
Taipei, Taiwan
Tel: 02-719-2377
Fax: 02-719-5951

NEC do Brasil S.A.
Sao Paulo-SP, Brasil
Tel: 011-889-1680
Fax: 011-889-1689

PREFACE

Readers This manual has been prepared for user engineers who understand the functions of the μPD780228 subseries and design and develop its application systems and programs.

Purpose This manual is intended for the users to understand the functions described in the Organization section below.

Organization The μPD780228 subseries manual is separated into two parts: this manual and Instructions (common to the 78K/0 series)

μPD780228 Subseries User's Manual (This manual)	78K/0 Series User's Manual Instructions
• Pin functions • Internal block functions • Interrupts • Miscellaneous on-chip peripheral functions	• CPU functions • Instruction set • Explanation of each instruction

How to Read This Manual

Before reading this manual, you must have general knowledge of electric and logic circuits and microcomputer.

- ☐ When you want to understand the functions in general:
  → Read this manual in the order of the contents.
- ☐ How to interpret the register format:
  → For the circled bit number, the bit name is defined as a reserved word in RA78K/0, and in CC78K/0, already defined in the header file named sfrbit.h.
- ☐ When confirming the details of the register whose register name is known:
  → Refer to **APPENDIX D REGISTER INDEX**.
- ★ ☐ When you want to know the differences between the μPD78044H subseries and the μPD780208 subseries:
  → Refer to **APPENDIX A DIFFERENCES BETWEEN μPD78044H, 780228, AND 780208 SUBSERIES**.
- ☐ When you want to know the details of the μPD780228 subseries instruction funcion:
  → Refer to **78K/0 Series User's Manual: Instruction. (U12326E)**.
- ★ ☐ When you want to know the electrical characteristics of the μPD780288 subseries:
  → Refer to **μPD780226**, **780228 Data Sheet (U11797E)** separately available.
- ★ ☐ When you want to know the application examples of each function of the μPD780228 subseries
  → Refer to **78K/0 Series Application Note - Fundamental (II) (U10121E)** separately available.

Legend	Data representation weight	:	High digits on the left and low digits on the right
	Active low representations	:	$\overline{xxx}$ (top bar over pin or signal name)
	Note	:	Description of **Note** in the text.
	Caution	:	Information requiring particular attention
	Remark	:	Additional explanatory material
	Numeral representations	:	Binary ..... xxxx or xxxxB
			Decimal xxxx
			Hexadecimal ... xxxxH

Related Documents

The related documents indicated in this publication may include preliminary versions. However, preliminary versions are not marked as such.

● **Documents related to devices**

Document Name	Document Number	
	English	Japanese
μPD780228 Subseries User's Manual	This manual	U12012J
μPD780226, 780228 Data Sheet	U11797E	U11797J
μPD78F0228 Preliminary Product Information	U11971E	U11971J
78K/0 Series Instruction Table	—	U10903J
78K/0 Series Instruction Set	—	U10904J
78K/0 Series User's Manual: Instructions	U12326E	U12326J
★ 78K/0 Series Application Note: Fundamental (II)	U10121E	U10121J

Caution The contents of the above documents are subject to change without notice. Be sure to use the latest edition for designing.

● **Documents related to development tools (User's Manual)**

	Document Name		Document Number	
			English	Japanese
	RA78K Series Assembler Package	Operation	EEU-1399	EEU-809
		Language	EEU-1404	EEU-815
	RA78K Series Structured Assembler Preprocessor		EEU-1402	EEU-817
★	RA78K0 Assembler Package	Operation	U11802E	U11802J
★		Assembly Language	U11801E	U11801J
★		Structured Assembly Language	U11789E	U11789J
	CC78K Series C Compiler	Operation	EEU-1280	EEU-656
		Language	EEU-1284	EEU-655
	CC78K0 C Compiler	Operation	U11517E	U11517J
		Language	U11518E	U11518J
	CC78K Series Library Source File		—	U12322J
	CC78K Compiler Application Note	Programming Know-how	EEA-1208	EEA-618
	IE-78001-R-A		Planned	Planned
	IE-78K0-SL-P01		Planned	Planned
	IE-780228-SL-EM4		Planned	Planned
	EP-100GF-SL		Planned	Planned
	SM78K0 System Simulator Windows™ Base	Reference	U10181E	U10181J
	SM78K Series System Simulator	External Components User Open Interface Specifications	U10092E	U10092J
	ID78K0 Integrated Debugger EWS base	Reference	—	U11151J
	ID78K0 Integrated Debugger PC base	Reference	U11539E	U11539J
	ID78K0 Integrated Debugger Windows base	Guide	U11649E	U11649J

Caution The contents of the above documents are subject to change without notice. Be sure to use the latest edition for designing.

● **Documents related to embedded software (User's Manual)**

Document Name		Document Number	
		English	Japanese
78K/0 Series Real-time OS	Fundamental	—	U11537J
	Installation	—	U11536J
78K/0 Series OS MX78K0	Fundamental	—	U12257J
Fuzzy Knowledge Data Creation Tool		EEU-1438	EEU-829
78K/0, 78K/II, 87AD Series Fuzzy Inference Development Support System Translator		EEU-1444	EEU-862
78K/0 Series Fuzzy Inference Development Support System Fuzzy Inference Module		EEU-1441	EEU-858
78K/0 Series Fuzzy Inference Development Support System Fuzzy Inference Debugger		EEU-1458	EEU-921

● **Other related documents**

Document Name	Document Number	
	English	Japanese
IC Package Manual	C10943X	
Semiconductor Device Mounting Technology Manual	C10535E	C10535J
Quality Grades on NEC Semiconductor Devices	C11531E	C11531J
NEC Semiconductor Device Reliability/Quality Control System	C10983E	C10983J
Electrostatic Discharge (ESD) Test	—	MEM-539
Guide to Quality Assurance for Semiconductor Devices	MEI-1202	C11893J
Microcomputer Product Guide –By third party	—	U11416J

Caution The contents of the above documents are subject to change without notice. Be sure to use the latest edition for designing.

TABLE OF CONTENTS

LIST OF FIGURES (1/3)

LIST OF FIGURES (2/3)

LIST OF FIGURES (3/3)

LIST OF TABLES (1/2)

LIST OF TABLES (2/2)

CHAPTER 1 GENERAL

1.1 Features

- High-capacity ROM and RAM

Item / Part Number	Program Memory		Data Memory		
	Mask ROM	Flash memory	Internal high-speed RAM	Internal expansion RAM	FIP™ display RAM
μPD780226	48K bytes	—	1024 bytes	512 bytes	96 bytes
μPD780228	60K bytes	—			
μPD78F0228	—	60K bytes[Note]			

Note 48K or 60K bytes can be selected by the memory size select register (IMS).

- Minimum instruction execution time changeable from high speed (0.4 μs) to low speed (6.4 μs)
- I/O port : 72 pins
- FIP controller/driver : Total display output pins: 48 (universal grid compatible)
  - Display current 10 mA : 16 pins
  - Display current 3 mA : 32 pins
- 8-bit resolution A/D converter : 8 channels
  - Supply voltage (AV_{DD} = 4.5 to 5.5 V)
- Serial interface : 1 channel
  - 3-wire serial I/O mode : 1 channel
- Timer: 4 channels
  - 8-bit remote control timer : 1 channel
  - 8-bit PWM timer : 2 channels
  - Watchdog timer : 1 channel
- Vectored interrupt source : 12
- Supply voltage : V_{DD} = 4.5 to 5.5 V

1.2 Application Fields

Combined mini-component audio systems, separate mini-component audio systems, tuners, cassette decks, CD players, and audio amplifiers

1.3 Ordering Information

Part Number	Package	Internal ROM
★ μPD780226GF-×××-3BA	100-pin plastic QFP (14 × 20 mm)	Mask ROM
★ μPD780228GF-×××-3BA	100-pin plastic QFP (14 × 20 mm)	Mask ROM
μPD78F0228GF-3BA[Note]	100-pin plastic QFP (14 × 20 mm)	Flash memory

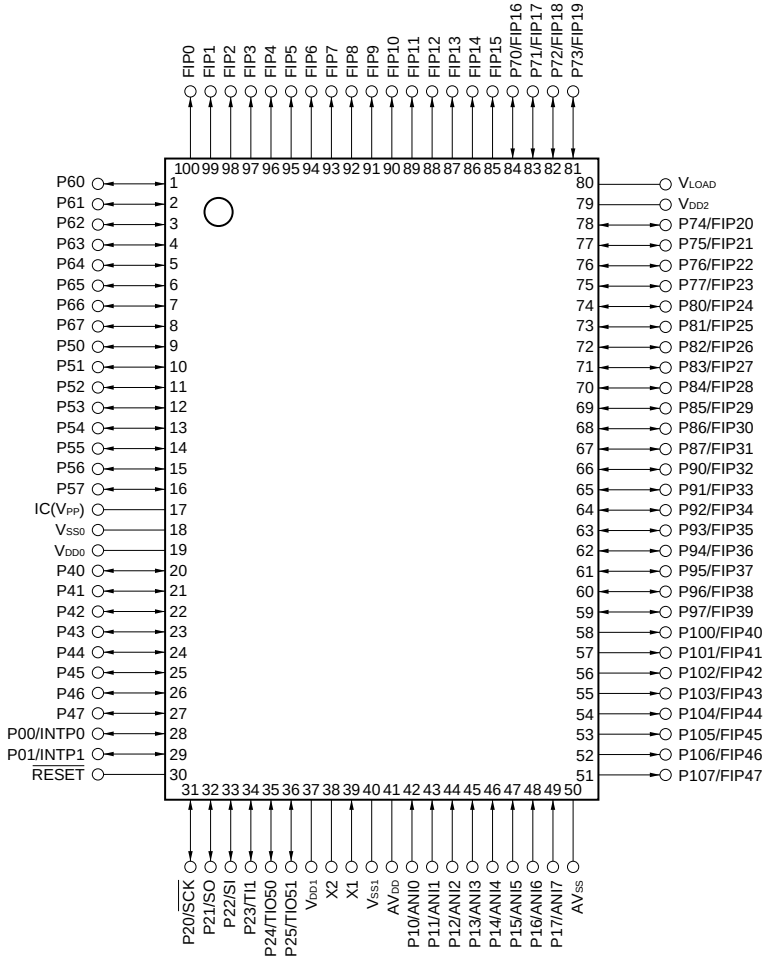
Note Under development

Remark ××× indicates a ROM code number.

1.4 Pin Configuration (Top View)

- **100-pin plastic QFP (14 × 20 mm)**

★ **μPD780226GF-xxx-3BA, 780228GF-xxx-3BA, 78F0228GF-3BA Note**



Note Under development

Cautions 1. Directly connect the IC (Internally Connected) pin to V_{SS1} in the normal operation mode.

2. Connect the AV_{DD} pin to V_{DD1}.

3. Connect the AV_{SS} pin to V_{SS1}.

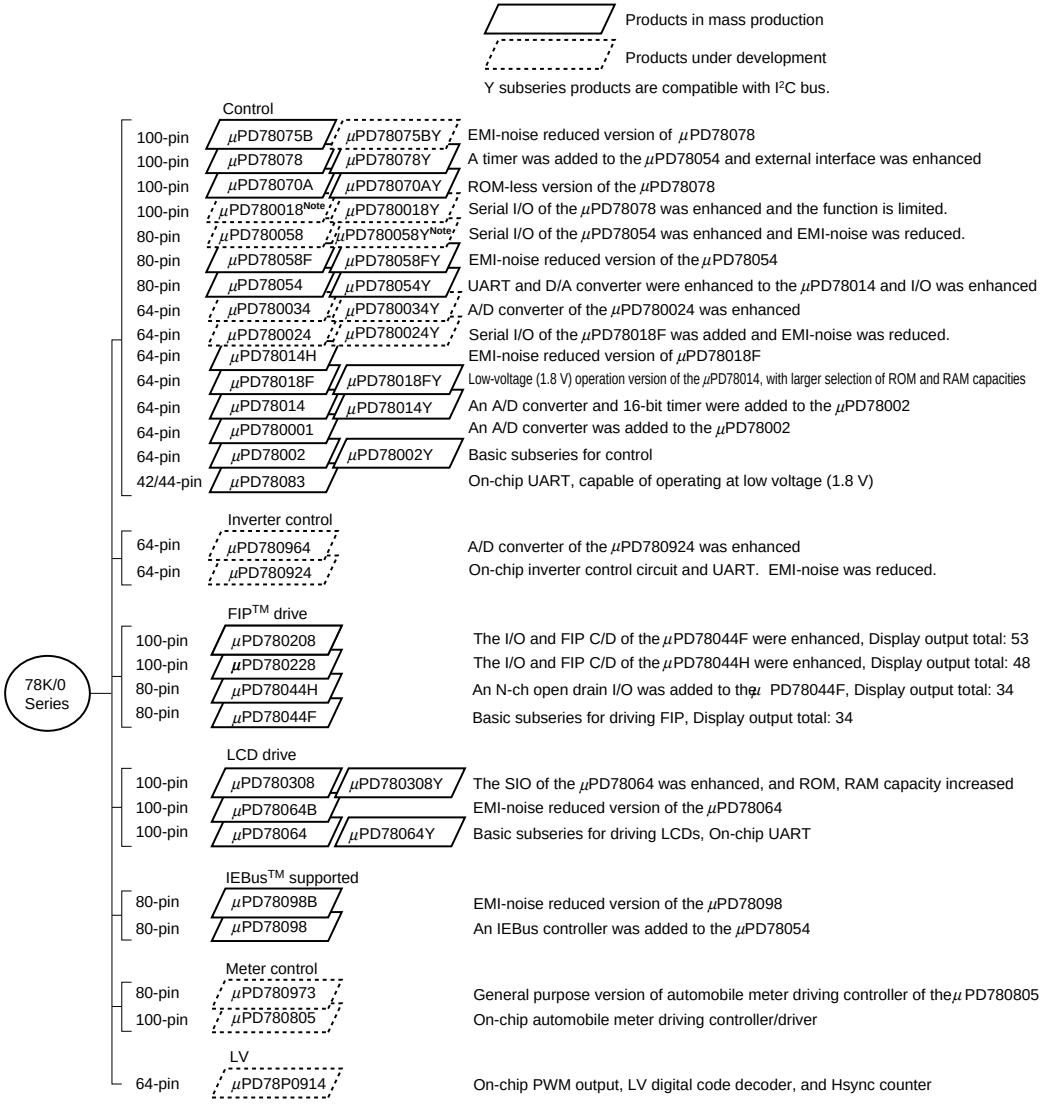
Remarks 1. When using the microcontroller in an application where the noise generated from the microcontroller must be suppressed, it is recommended that power be supplied to V_{DD0} and V_{DD1} from separate sources, and that V_{SS0} and V_{SS1} be connected to separate ground lines, to improve the noise immunity.

2. ( ): μPD78F0228

ANI0-ANI7	:	Analog Input
AV_{DD}	:	Analog Power Supply
AV_{SS}	:	Analog Ground
FIP0-FIP47	:	Fluorescent Indicator Panel
IC	:	Internally Connected
INTP0, INTP1	:	Interrupt from Peripherals
P00, P01	:	Port0
P10-P17	:	Port1
P20-P25	:	Port2
P40-P47	:	Port4
P50-P57	:	Port5
P60-P67	:	Port6
P70-P77	:	Port7
P80-P87	:	Port8
P90-P97	:	Port9
P100-P107	:	Port10
$\overline{RESET}$	:	Reset
$\overline{SCK}$	:	Serial Clock
SI	:	Serial Input
SO	:	Serial Output
TI1	:	Timer Input
TIO50, TIO51	:	Timer Input/Output
V_{DD0}-V_{DD2}	:	Power Supply
V_{LOAD}	:	Negative Power Supply
V_{PP}	:	Programming Power Supply
V_{SS0}, V_{SS1}	:	Ground
X1, X2	:	Crystal

★ 1.5 78K/0 Series Expansion

The following shows the 78K/0 Series products development. Subseries name are shown inside frames.



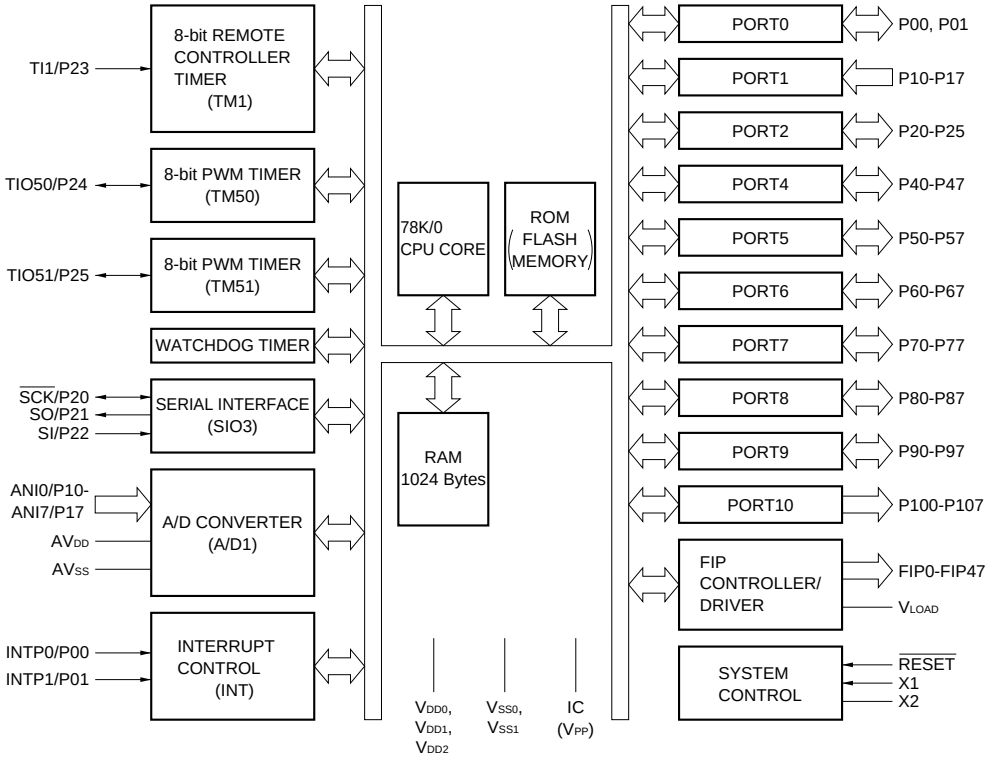
Note Under planning

The following lists the main functional differences between subseries products.

Subseries Name \ Function		ROM Capacity	Timer				8-bit A/D	10-bit A/D	8-bit D/A	Serial Interface	I/O	V_{DD} MIN. Value	External Expansion
			8-bit	16-bit	Watch	WDT							
Control	μPD78075B	32K-40K	4ch	1ch	1ch	1ch	8ch	–	2ch	3ch (UART: 1ch)	88	1.8 V	○
	μPD78078	48K-60K											
	μPD78070A	–									61	2.7 V	
	μPD780018	48K-60K							–	2ch (time division 3-wire: 1ch)	88		
	μPD780058	24K-60K	2ch						2ch	3ch (time division UART: 1ch)	68	1.8 V	
	μPD78058F	48K-60K								3ch (UART: 1ch)	69	2.7 V	
	μPD78054	16K-60K										2.0 V	
	μPD780034	8K-32K					–	8ch	–	3ch (UART: 1ch, time division 3-wire: 1ch)	51	1.8 V	
	μPD780024						8ch	–					
	μPD78014H									2ch	53	1.8 V	
	μPD78018F	8K-60K											
	μPD78014	8K-32K										2.7 V	
	μPD780001	8K		–	–					1ch	39		–
	μPD78002	8K-16K			1ch		–				53		○
	μPD78083				–		8ch			1ch (UART: 1ch)	33	1.8 V	–
Inverter control	μPD780964	8K-32K	3ch	**Note**	–	1ch	–	8ch	–	2ch (UART: 2ch)	47	2.7 V	○
	μPD780924						8ch	–					
FIP drive	μPD780208	32K-60K	2ch	1ch	1ch	1ch	8ch	–	–	2ch	74	2.7 V	–
	μPD780228	48K-60K	3ch	–	–					1ch	72	4.5 V	
	μPD78044H	32K-48K	2ch	1ch	1ch						68	2.7 V	
	μPD78044F	16K-40K								2ch			
LCD drive	μPD780308B	48K-60K	2ch	1ch	1ch	1ch	8ch	–	–	3ch (time division UART: 1ch)	57	2.0 V	–
	μPD78064	32K								2ch (UART: 1ch)			
	μPD78064	16K-32K											
IEBus supported	μPD78098B	40K-60K	2ch	1ch	1ch	1ch	8ch	–	2ch	3ch (UART: 1ch)	69	2.7 V	○
	μPD78098	32K-60K											
Meter control	μPD780973	24K-32K	3ch	1ch	1ch	1ch	5ch	–	–	2ch (UART: 1ch)	56	4.5 V	–
	μPD780805	40K-60K	2ch				8ch				39	2.7 V	
LV	μPD78P0914	32K	6ch	–	–	1ch	8ch	–	–	2ch	54	4.5 V	○

Note 10-bit timer: 1 channel

1.6 Block Diagram



Remarks 1. The internal ROM capacity differs depending on the model.

2. ( ): μPD78F0228

1.7 Functional Outline

Item \ Part Number		μPD780226	μPD780228	μPD78F0228
Internal memory	ROM	Mask ROM		Flash memory
		48K bytes	60K bytes	60K bytes[Note]
	High-speed RAM	1024 bytes		
	Expansion RAM	512 bytes		
	FIP display RAM	96 bytes		
General-purpose register		8 bits × 8 × 4 banks		
Minimum instruction execution time		0.4/0.8/1.6/3.2/6.4 μs (main system clock: 5.0 MHz)		
Instruction set		• 16-bit operation • Multiplication/division (8 bits × 8 bits, 16 bits ÷ 8 bits) • Bit manipulation (set, reset, test, Boolean operation) • BCD adjustment, etc.		
I/O port (including FIP-multiplexed pins)		• Total : 72 pins • CMOS input : 8 pin • CMOS I/O : 16 pins • N-ch open-drain I/O : 16 pins • P-ch open-drain I/O : 24 pins • P-ch open-drain output : 8 pins		
FIP controller/driver		• Total display output : 48 pins • Display current 10 mA : 16 pins • Display current: 3 mA : 32 pins		
A/D converter		• 8-bit resolution × 8 channels • Supply voltage: AV_{DD} = 4.5 to 5.5 V		
Serial interface		3-wire serial I/O mode: 1 channel		
Timer		• 8-bit remote control timer : 1 channel • 8-bit PWM timer : 2 channels • Watchdog timer : 1 channel		
Timer output		2 pins (8-bit PWM output)		
Vectored interrupt source	Maskable	Internal: 6, external: 4		
	Non-maskable	Internal: 1		
	Software	1		
Supply voltage		V_{DD} = 4.5 to 5.5 V		
Package		100-pin plastic QFP (14 × 20 mm)		

Note 48K or 60K bytes can be selected by the memory size select register (IMS)

1.8 Mask Option

The mask ROM models (μPD780226 and 780228) have mask options. By specifying the mask options when placing an order for these models, the pull-up and pull-down resistors shown in Table 1-1 can be connected. If these mask options are used when pull-up and pull-down resistors are necessary, the number of components can be decreased and the mounting area can be reduced.

Table 1-1 shows the mask options available for the μPD780228 subseries.

Table 1-1. Mask Options of Mask ROM Models

Pin Name	Mask Option
P50-P57, P60-P67	Pull-up resistors can be connected in 1-bit units.
P70/FIP16-P77/FIP23, P80/FIP24-P87/FIP31, P90/FIP32-P97/FIP39, P100/FIP40-P107/FIP47	Pull-down resistors can be connected in 1-bit units.

CHAPTER 2 PIN FUNCTIONS

2.1 Pin Function List

(1) Port pins (1/2)

Pin Name	I/O	Function	At Reset	Shared with:
P00	I/O	Port 0. 2-bit I/O port. Can be set in input or output mode in 1-bit units. Internal pull-up resistor can be used via software when this port is used as input port.	Input	INTP0
P01				INTP1
P10-P17	Input	Port 1. 8-bit input port.	Input	ANI0-ANI7
P20	I/O	Port 2. 6-bit I/O port. Can be set in input or output mode in 1-bit units. Internal pull-up resistor can be used via software when this port is used as input port.	Input	$\overline{SCK}$
P21				SO
P22				SI
P23				TI1
P24				TIO50
P25				TIO51
P40-P47	I/O	Port 4. 8-bit I/O port. Can be set in input or output mode in 1-bit units. Can directly drive LED. Internal pull-up resistor can be used via software when this port is used as input port.	Input	—
P50-P57	I/O	Port 5. N-ch open-drain 8-bit medium-voltage I/O port. Can be set in input or output mode in 1-bit units. Can directly drive LED. Internal pull-up resistor can be used by mask option in 1-bit units (mask ROM models only). μPD78F0228 does not have pull-up resistors, however.	Input	—
P60-P67	I/O	Port 6. N-ch open-drain 8-bit medium-voltage I/O port. Can be set in input or output mode in 1-bit units. Can directly drive LED. Internal pull-up resistor can be used by mask option in 1-bit units (mask ROM models only). μPD78F0228 does not have pull-up resistors, however.	Input	—
P70-P77	I/O	Port 7. P-ch open-drain 8-bit high-voltage I/O port. Can be set in input or output mode in 1-bit units. Internal pull-down resistor can be used by mask option in 1-bit units (mask ROM models only). μPD78F0228 does not have pull-down resistors, however.	Input	FIP16-FIP23

(1) Port pins (2/2)

Pin Name	I/O	Function	At Reset	Shared with:
P80-P87	I/O	Port 8. P-ch open-drain 8-bit high-voltage I/O port. Can be set in input or output mode in 1-bit units. Internal pull-down resistor can be used by mask option in 1-bit units (mask ROM models only). μPD78F0228 does not have pull-down resistors, however.	Input	FIP24-FIP31
P90-P97	I/O	Port 9. P-ch open-drain 8-bit high-voltage I/O port. Can be set in input or output mode in 1-bit units. Internal pull-down resistor can be used by mask option in 1-bit units (mask ROM models only). μPD78F0228 does not have pull-down resistors, however.	Input	FIP32-FIP39
P100-P107	Output	Port 10. P-ch open-drain 8-bit high-voltage output port. Internal pull-down resistor can be used by mask option in 1-bit units (mask ROM models only). μPD78F0228 does not have pull-down resistors, however.	Output	FIP40-FIP47

(2) Pins other than port pins

Pin Name	I/O	Function	At Reset	Shared with:
INTP0	Input	Valid edge (rising, falling, or both rising and falling) can be specified.	Input	P00
INTP1		External interrupt request input.		P01
$\overline{SCK}$	I/O	Serial clock input/output of serial interface.	Input	P20
SO	Output	Serial data output of serial interface.	Input	P21
SI	Input	Serial data input of serial interface.	Input	P22
TI1	Input	Timer input of 8-bit remote control timer (TM1).	Input	P23
TIO50	I/O	Capture trigger input/timer output of 8-bit PWM timer (TM50).	Input	P24
TIO51	I/O	Capture trigger input/timer output of 8-bit PWM timer (TM51).	Input	P25
FIP0-FIP15	Output	High-voltage high-current output of FIP controller/driver.	Output	—
FIP16-FIP23			Input	P70-P77
FIP24-FIP31				P80-P87
FIP32-FIP39				P90-P97
FIP40-FIP47				P100-P107
V_{LOAD}	—	Pull-down resistor connection of FIP controller/driver.	—	—
$\overline{RESET}$	Input	System reset input.	—	—
X1	Input	Crystal connection for main system clock oscillation.	—	—
X2	—		—	—
ANI0-ANI7	Input	Analog input for A/D converter.	Input	P10-P17
AV_{DD}	—	Analog power to A/D converter. Same potential as V_{DD1}.	—	—
AV_{SS}	—	Ground potential for A/D converter. Same potential as V_{SS1}.	—	—
V_{DD0}	—	Positive power supply to ports.	—	—
V_{DD1}	—	Positive power supply (except ports, analog block, and FIP controller/driver)	—	—
V_{DD2}	—	Positive power supply to FIP controller/driver.	—	—
V_{SS0}	—	Ground potential for ports.	—	—
V_{SS1}	—	Ground potential (except ports and analog block).	—	—
V_{PP}**Note**	—	High voltage is applied to this pin when program is written/verified. In the normal operation mode, directly connect this pin to V_{SS1}.	—	—
IC	—	Internally connected. Directly connect this pin to V_{SS1}.	—	—

Note V_{PP} is provided to the μPD78F0228 only.

2.2 Pin Functions

2.2.1 P00 and P01 (Port 0)

P00 and P01 are used as a 2-bit I/O port. These pins also have external interrupt request input functions in addition to the I/O port function.

Port 0 can be set in the following operation modes in 1-bit units.

(1) Port mode

P00 and P01 function as a 2-bit I/O port in this mode.

This 2-bit port can be set in the input or output mode in 1-bit units by the port mode register 0 (PM0). When used as an input port, the internal pull-up resistor can be connected by using the pull-up resistor option register 0 (PU0).

(2) Control mode

P00 and P01 functions as external interrupt request input pins (INTP0 and INTP1).

INTP0 and INTP1 input external interrupt requests whose valid edge can be specified (to be the rising edge, falling edge, or both the rising and falling edges).

2.2.2 P10 through P17 (Port 1)

P10 through P17 constitute an 8-bit input port, port 1. This port is also used to input analog signals to the internal A/D converter.

The following operation modes can be specified in 1-bit units.

(1) Port mode

Port 1 functions as an 8-bit input port in this mode.

(2) Control mode

P10 through P17 function as analog input pins (ANI0 through ANI7) of the A/D converter.

2.2.3 P20 through P25 (Port 2)

P20 through P25 constitute a 6-bit I/O port, port 2. These pins also have functions to input/output data of the serial interface, clock, and timer signals.

The following operation modes can be specified in 1-bit units.

(1) Port mode

In this mode, P20 through P25 function as a 6-bit I/O port. This port can be set in the input or output mode in 1-bit units by using the port mode register 2 (PM2). When the port is used as an input port, the internal pull-up resistor can be used if so specified by the pull-up resistor option register 2 (PU2).

(2) Control mode

In this mode, P20 through P25 are used to input/output serial interface data, clock, and timer signal, and input timers' capture trigger signals.

(a) SI, SO

These are I/O pins of the serial data of the serial interface.

(b) $\overline{\text{SCK}}$

This is an I/O pin of the serial clock of the serial interface.

(c) TI1

Timer input pin of the 8-bit remote control timer.

(d) TIO50 and TIO51

Capture trigger input pin of the 8-bit PWM timers and timer output pin.

2.2.4 P40 through P47 (Port 4)

P40 through 47 constitute an 8-bit I/O port. This port can be set in the input or output mode in 1-bit units by using the port mode register 4 (PM4). When it is used as an input port, the internal pull-up resistor can be connected by using the pull-up resistor option register 4 (PU4).

This port can directly drive an LED.

2.2.5 P50 through P57 (Port 5)

P50 through 57 constitute an 8-bit I/O port. This port can be set in the input or output mode in 1-bit units by using the port mode register 5 (PM5).

P50 through P57 are N-ch open-drain pins. Pull-up resistors can be connected to these pins of the mask ROM models by mask option. The μPD78F0228 does not have pull-up resistors.

This port can directly drive an LED.

2.2.6 P60 through P67 (Port 6)

P60 through 67 constitute an 8-bit I/O port. This port can be set in the input or output mode in 1-bit units by using the port mode register 6 (PM6).

P60 through P67 are N-ch open-drain pins. Pull-up resistors can be connected to these pins of the mask ROM models by mask option. The μPD78F0228 does not have pull-up resistors.

This port can directly drive an LED.

2.2.7 P70 through P77 (Port 7)

P70 through P77 constitute an 8-bit I/O port. These pins are also used as FIP controller/driver output pins.

The following operation modes can be specified in 1-bit units.

(1) Port mode

P70 through P77 function as an 8-bit I/O port in this mode.

These pins are P-ch open-drain pins. Pull-down resistors can be connected to these pins of the mask ROM models by mask option. The μPD78F0228 does not have pull-down resistors.

(2) Control mode

In this mode, P70 through P77 function as the output pins of the FIP controller/driver (FIP16 through FIP23).

2.2.8 P80 through P87 (Port 8)

P80 through P87 constitute an 8-bit I/O port. These pins are also used as FIP controller/driver output pins.

The following operation modes can be specified in 1-bit units.

(1) Port mode

P80 through P87 function as an 8-bit I/O port in this mode.

These pins are P-ch open-drain pins. Pull-down resistors can be connected to these pins of the mask ROM models by mask option. The μPD78F0228 does not have pull-down resistors.

(2) Control mode

In this mode, P80 through P87 function as the output pins of the FIP controller/driver (FIP24 through FIP31).

2.2.9 P90 through P97 (Port 9)

P90 through P97 constitute an 8-bit I/O port. These pins are also used as FIP controller/driver output pins. The following operation modes can be specified in 1-bit units.

(1) Port mode

P90 through P97 function as an 8-bit I/O port in this mode.

These pins are P-ch open-drain pins. Pull-down resistors can be connected to these pins of the mask ROM models by mask option. The μPD78F0228 does not have pull-down resistors.

(2) Control mode

In this mode, P90 through P97 function as the output pins of the FIP controller/driver (FIP32 through FIP39).

2.2.10 P100 through P107 (Port 10)

P100 through P107 constitute an 8-bit output port. These pins are also used as FIP controller/driver output pins. The following operation modes can be specified in 1-bit units.

(1) Port mode

P100 through P107 function as an 8-bit output port in this mode.

These pins are P-ch open-drain pins. Pull-down resistors can be connected to these pins of the mask ROM models by mask option. The μPD78F0228 does not have pull-down resistors.

(2) Control mode

In this mode, P100 through P107 function as the output pins of the FIP controller/driver (FIP40 through FIP47).

2.2.11 FIP0 through FIP15

These are the output pins of the FIP controller/driver.

2.2.12 V_{LOAD}

This pin connects a pull-down resistor to the FIP controller/driver.

2.2.13 AV_{DD}

This pin supply an analog voltage to the A/D converter.

Always keep this pin at the same potential as the V_{DD1} pin even when the A/D converter is not used.

2.2.14 AV_{SS}

This is the ground pin of the A/D converter.

Always keep this pin at the same potential as the V_{SS1} pin even when the A/D converter is not used.

2.2.15 $\overline{\text{RESET}}$

This pin inputs an active-low system reset signal.

2.2.16 X1 and X2

These pins connect a crystal resonator for main system clock oscillation.

To supply an external clock, input it to X1, and input a signal reverse to that input to X1, to X2.

2.2.17 V_{DD0} through V_{DD2}

V_{DD0} supplies a positive voltage to the ports.

V_{DD1} supplies a positive voltage to the internal function blocks other than the ports, analog block, and FIP controller/driver.

V_{DD2} supplies a positive voltage to the FIP controller/driver.

2.2.18 V_{SS0} and V_{SS1}

V_{SS0} is the ground pin for the ports.

V_{SS1} is the ground pin for the internal function blocks other than the ports and analog block.

2.2.19 V_{PP} (μPD78F0228 only)

A high voltage is applied to this pin when the flash memory programming mode is used and when a program is written or verified.

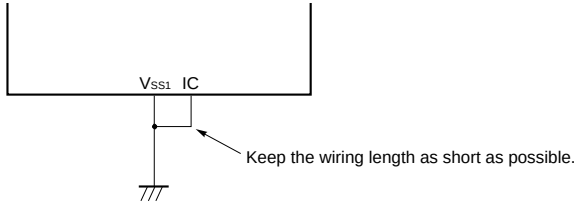
Directly connect this pin to V_{SS1} in the normal operation mode.

2.2.20 IC (Mask ROM product only)

The IC (Internally Connected) pin sets a test mode in which the μPD780226 and 780228 are tested before shipment. Usually, connect the IC pin directly to V_{SS1} with as short a wiring length as possible.

If there is a potential difference between the IC and V_{SS1} pins because the wiring length between the IC and V_{SS1} pin is too long, or external noise is superimposed on the IC pin, your program may not run correctly.

○ Directly connect the IC pin to the V_{SS1}.



2.3 I/O Circuits of Pins and Connections of Unused Pins

Table 2-1 shows the I/O circuit types of the respective pins, and the recommended connections of each pin when it is not used.

For the configuration of each type of the I/O circuit, refer to Figure 2-1.

Table 2-1. I/O Circuit Type of Each Pin

Pin Name	I/O Circuit Type	I/O	Recommended Connections of Unused Pin
P00/INTP0	8-C	I/O	Individually connect to V_{SS0} via resistor.
P01/INTP1			
P10/ANI0-P17/ANI7	9	Input	
P20/$\overline{SCK}$	8-C	I/O	Individually connect to V_{DD0} or V_{SS0} via resistor.
P21/SO	5-H		
P22/SI	8-C		
P23/TI1			
P24/TIO50			
P25/TIO51			
P40-P47			
Mask ROM model			
P50-P57	13-J	I/O	Individually connect to V_{DD0} via resistor.
P60-P67			
★ P70/FIP16-P77/FIP23	15-F	I/O	Individually connect to V_{DD0} or V_{SS0} via resistor.
P80/FIP24-P87/FIP31			
P90/FIP32-P97/FIP39			
★ P100/FIP40-P107/FIP47	14-F	Output	
IC	—	—	Directly connect to V_{SS1}.
μPD78F0228			
P50-P57	13-K	I/O	Individually connect to V_{DD0} via resistor.
P60-P67			
P70/FIP16-P77/FIP23	15-E	I/O	Individually connect to V_{DD0} or V_{SS0} via resistor.
P80/FIP24-P87/FIP31			
P90/FIP32-P97/FIP39			
P100/FIP40-P107/FIP47	14-E	Output	
V_{PP}	—	—	Directly connect to V_{SS1}.
FIP0-FIP15	14-C	Output	Individually connect to V_{DD0} or V_{SS0} via resistor.
$\overline{RESET}$	2	Input	—
AV_{DD}	—	—	Connect to V_{DD1}.
AV_{SS}			Connect to V_{SS1}.
V_{LOAD}			

Figure 2-1. Pin Input/Output Circuit List (1/2)

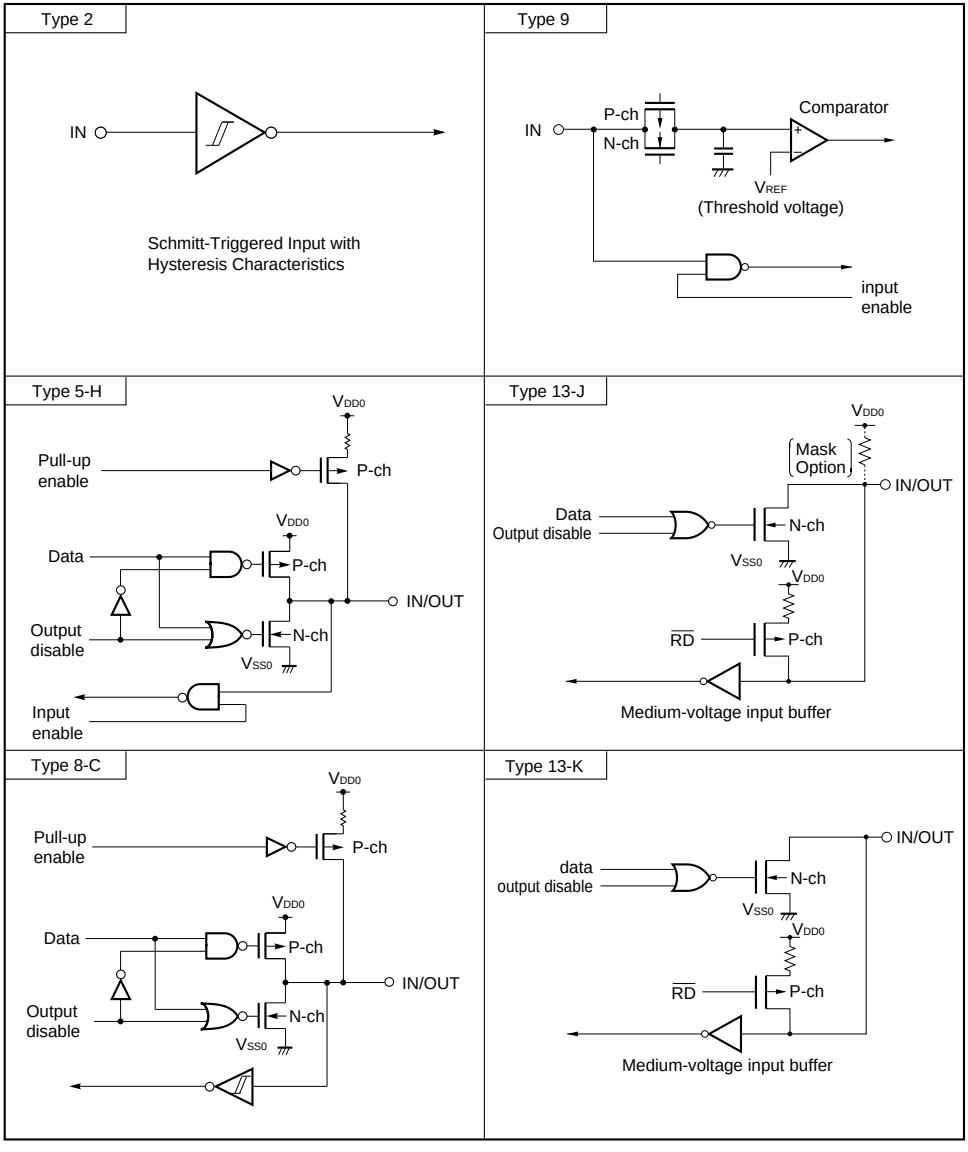
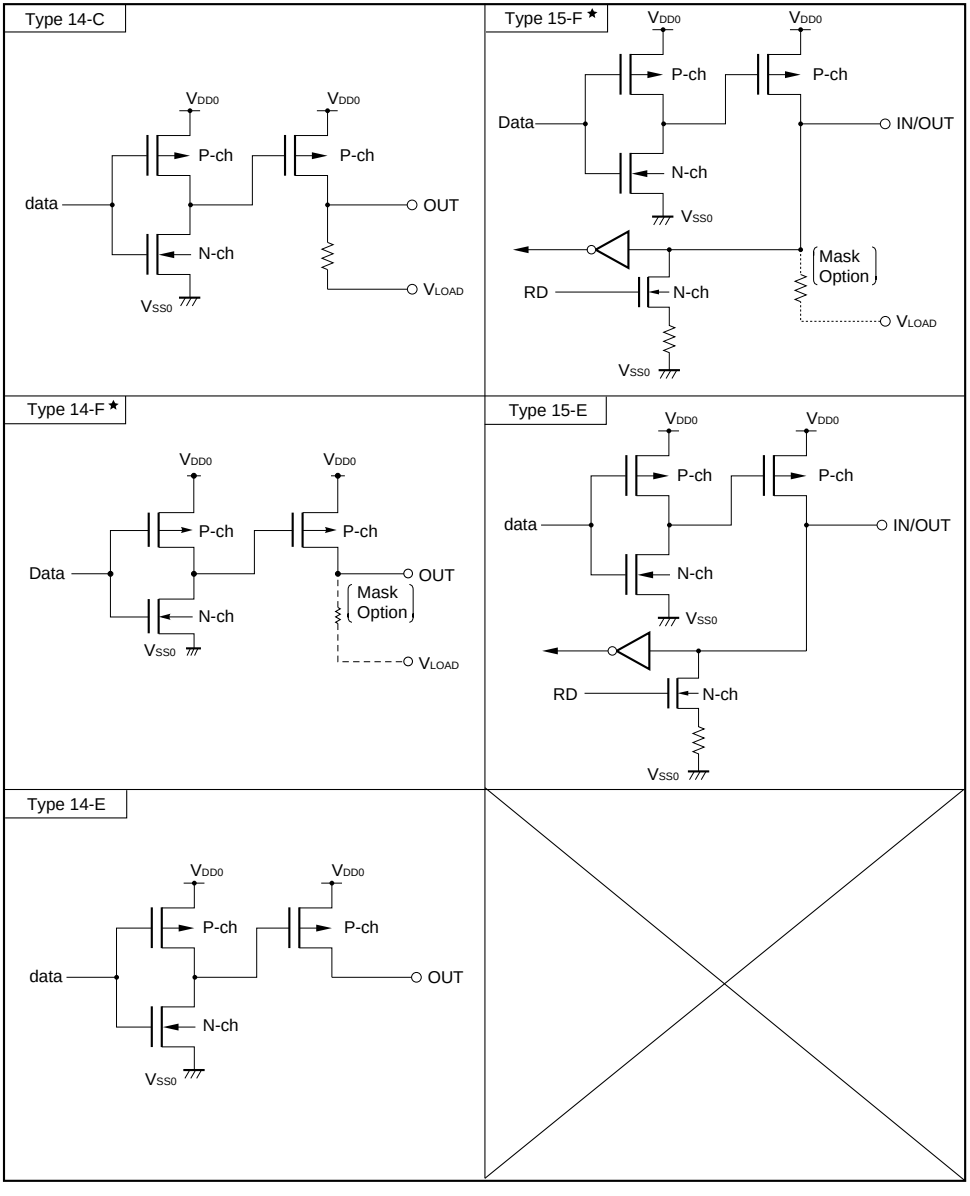


Figure 2-1. Pin Input/Output Circuit List (2/2)



CHAPTER 3 CPU ARCHITECTURE

3.1 Memory Space

Each model in the μPD780228 subseries can access a memory space of 64K bytes. Figures 3-1 through 3-3 show the memory map.

Figure 3-1. Memory Map (μPD780226)

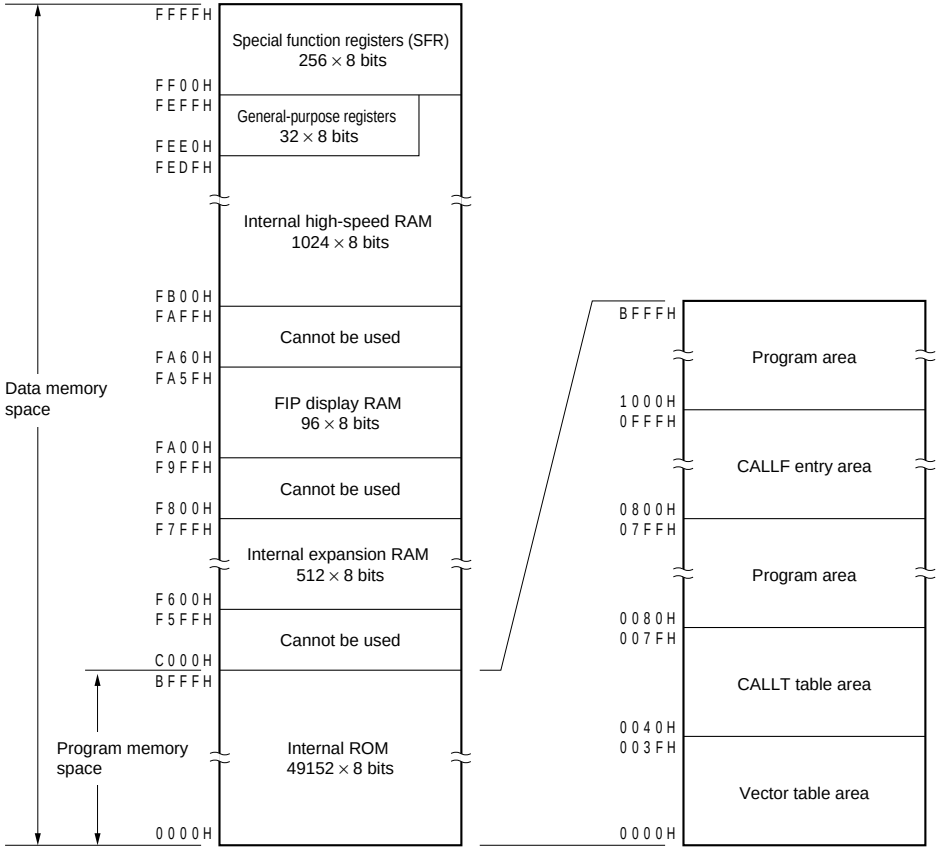


Figure 3-2. Memory Map (μPD780228)

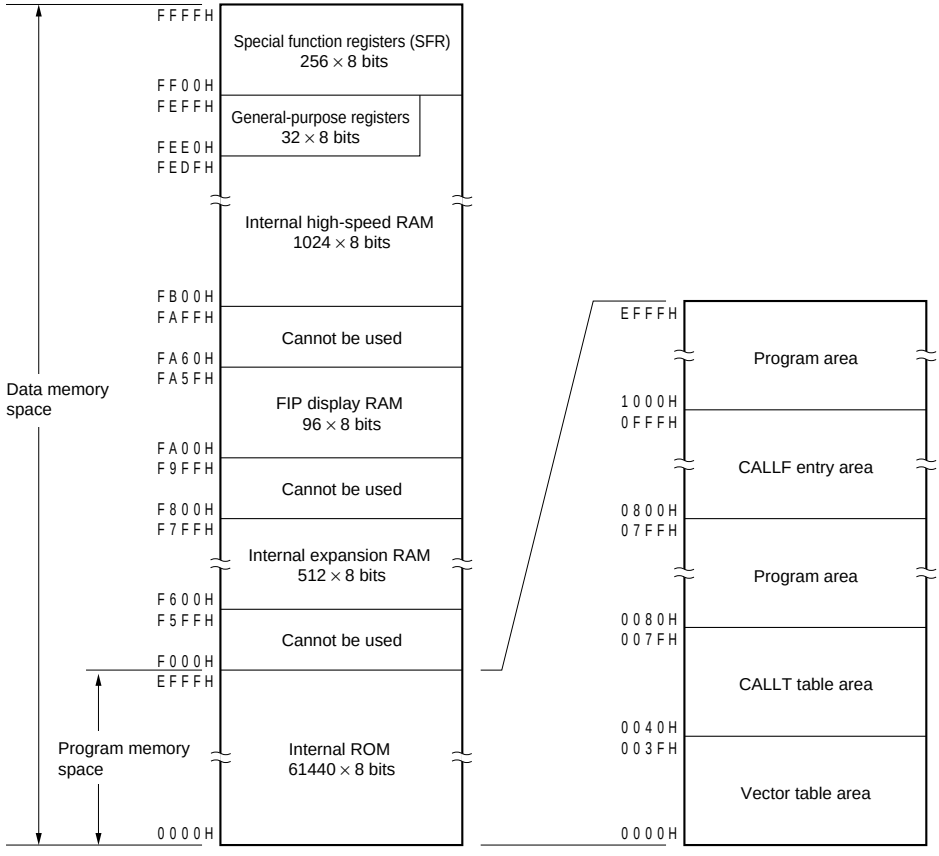
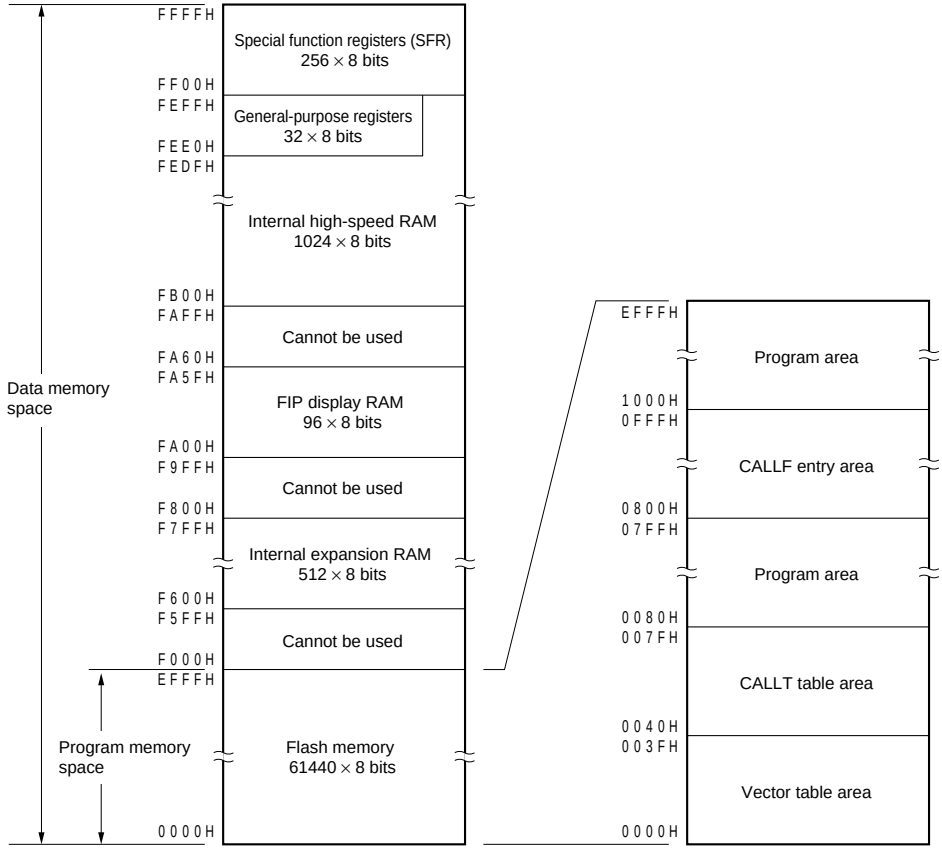


Figure 3-3. Memory Map (μPD78F0228)



3.1.1 Internal program memory space

The internal program memory space stores programs and table data. Usually, this space is accessed by program counter (PC).

Each model in the μPD780228 subseries has an internal ROM (or flash memory) of the following capacity.

Table 3-1. Internal ROM Capacity

Part Number	Internal ROM	
	Structure	Capacity
μPD780226	Mask ROM	49152 × 8 bits
μPD780228		61440 × 8 bits
μPD78F0228	Flash memory	

The following areas are allocated to the internal program memory space.

(1) Vector table area

A 64-byte area of addresses 0000H through 003FH is reserved as a vector table area. Program start addresses to which execution is to branch when the $\overline{\text{RESET}}$ signal is input or when an interrupt request is generated are stored in this area. Of a 16-bit address, the low-order 8 bits are stored to an even address, and the high-order 8 bits are stored to an odd address.

Table 3-2. Vector Table

Vector Table Address	Interrupt Request	Vector Table Address	Interrupt Request
0000H	$\overline{\text{RESET}}$ Input	000EH	INTKS
0004H	INTWDT	0010H	INTCSI3
0006H	INTP0	0012H	INTTM50
0008H	INTP1	0014H	INTTM51
000AH	INTTM10	0016H	INTAD
000CH	INTTM11	003EH	BRK

(2) CALLT instruction table area

The 64-byte area 0040H through 007FH can be used to store the subroutine entry addresses of the 1-byte call instruction (CALLT).

(3) CALLF instruction entry area

From an area of 0800H through 0FFFH, a subroutine can be directly called by using the 2-byte call instruction (CALLF).

3.1.2 Internal data memory space

The μPD780228 subseries has the following RAM.

(1) Internal high-speed RAM

This RAM consists of addresses FB00H through FEFFH, or 1024 × 8 bits.

Of these addresses, FEE0H through FEFFH constitute a 32-byte area to which four banks of general-purpose registers, with each bank consisting of eight 8-bit registers, are allocated.

The internal high-speed RAM can be also used as a stack memory.

(2) Internal expansion RAM

An internal expansion RAM is allocated to a 512-byte area of F600H through F7FFH.

(3) FIP display RAM

An FIP display RAM is allocated to a 96-byte area of FA00H through F5FH. This RAM can be also used as a normal RAM.

3.1.3 Special function register (SFR) area

Special function registers (SFR) are allocated to the area FF00H through FFFFH as on-chip peripheral hardware (refer to **Table 3-3 Special Function Registers** in **3.2.3 Special function register (SFR))**.

Caution Do not access an address to which no SFR is allocated.

3.1.4 Data memory addressing

Specifying the address of the instruction to be executed next, or specifying an address of the register or memory to be manipulated when an instruction is executed is called addressing.

The address of the instruction to be executed next is addressed by the program counter (PC) (for details, refer to **3.3 Addressing of Instruction Address**).

The μPD780288 subseries has many addressing modes to improve the operability when a memory area to be manipulated during instruction execution is addressed. The special function registers (SFR) and general-purpose registers can be addressed in accordance with their functions. All the 64K bytes of the data memory, 0000H through FFFFH, can be also addressed. Figures 3-4 through 3-6 illustrates how the data memory is addressed.

For details on each addressing mode, refer to **3.4 Addressing of Operand Address**.

Figure 3-4. Addressing of Data Memory (μPD780226)

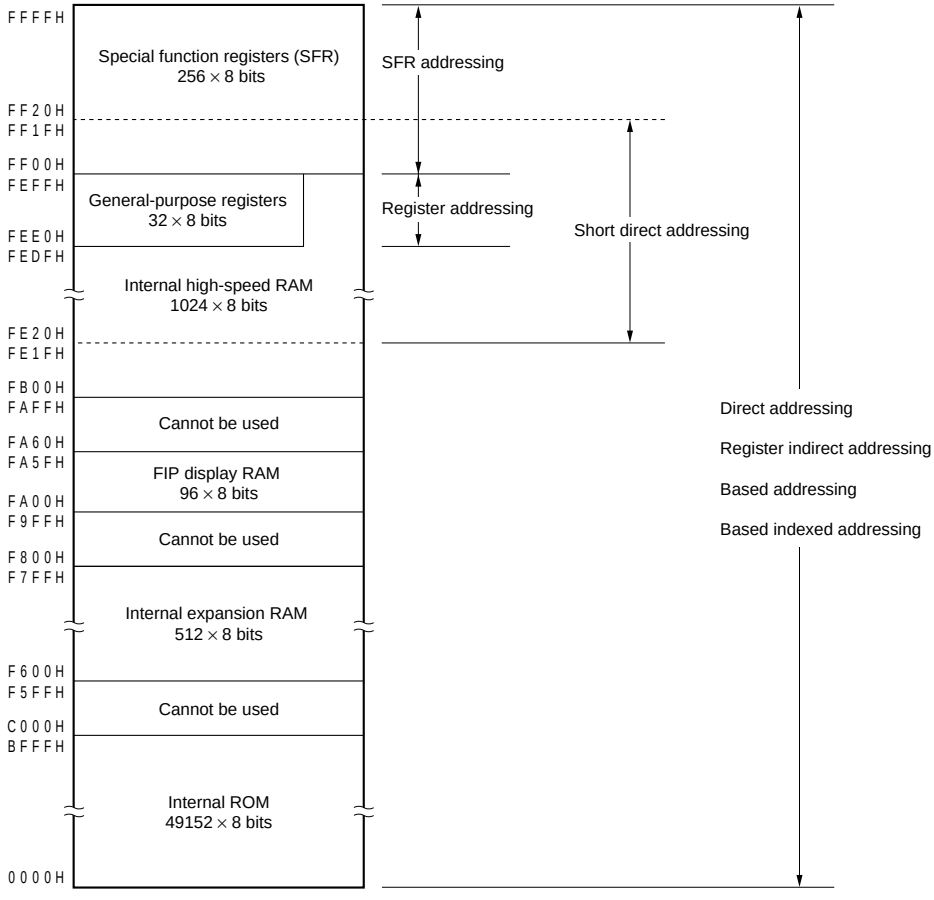


Figure 3-5. Addressing of Data Memory (μPD780228)

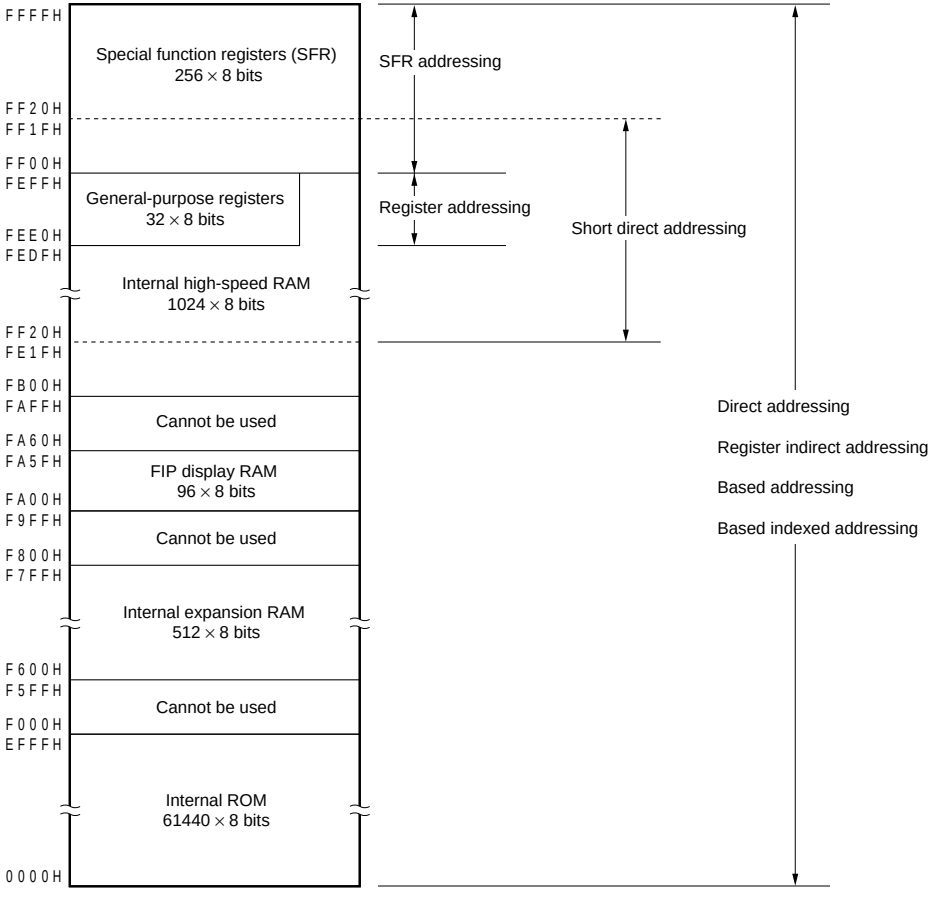
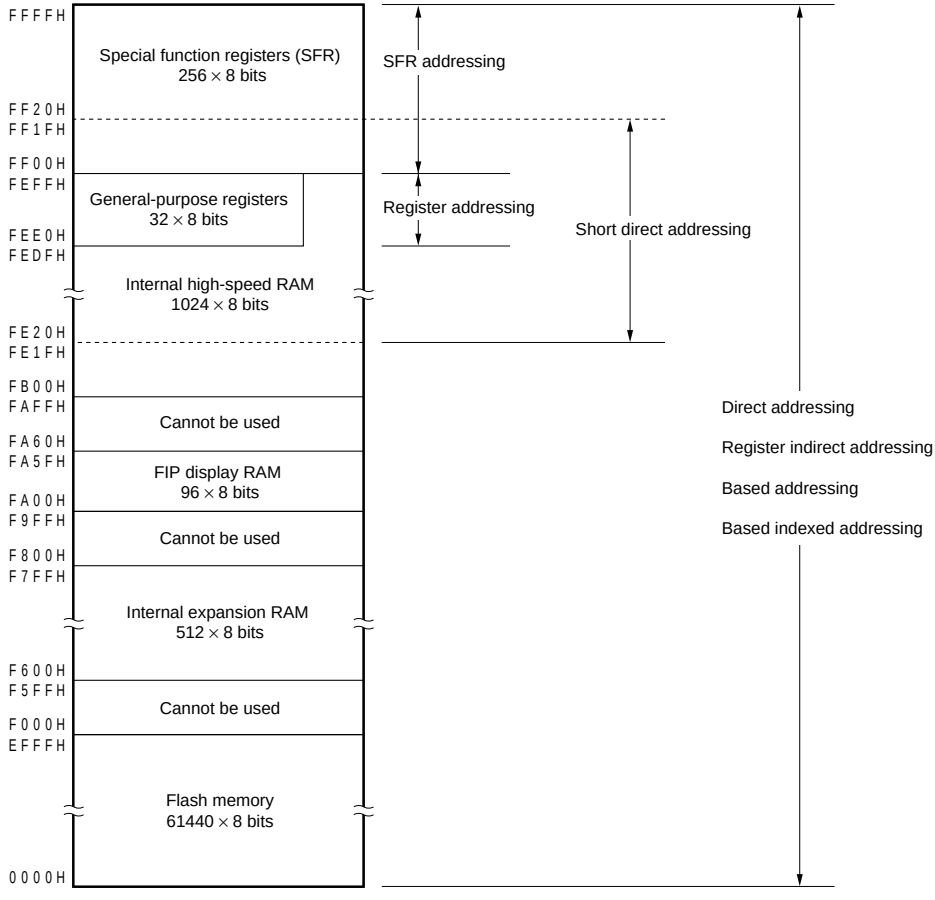


Figure 3-6. Addressing of Data Memory (μPD78F0228)



3.2 Processor Registers

The μPD780228 subseries units incorporate the following processor registers.

3.2.1 Control registers

The control registers control the program sequence, statuses and stack memory. A program counter (PC), a program status word (PSW) and a stack pointer (SP) are control registers.

(1) Program counter (PC)

The program counter is a 16-bit register which holds the address information of the next program to be executed.

In normal operation, the PC is automatically incremented according to the number of bytes of the instruction to be fetched. When a branch instruction is executed, immediate data and register contents are set.

$\overline{\text{RESET}}$ input sets the reset vector table values at addresses 0000H and 0001H to the program counter.

Figure 3-7. Program Counter Configuration

	15															0
PC	PC15	PC14	PC13	PC12	PC11	PC10	PC9	PC8	PC7	PC6	PC5	PC4	PC3	PC2	PC1	PC0

(2) Program status word (PSW)

The program status word is an 8-bit register consisting of various flags to be set/reset by instruction execution. Program status word contents are automatically stacked upon interrupt request generation or PUSH PSW instruction execution and are automatically reset upon execution of the RETB, RETI and POP PSW instructions.

$\overline{\text{RESET}}$ input sets the PSW to 02H.

Figure 3-8. Program Status Word Configuration

	7							0
PSW	IE	Z	RBS1	AC	RBS0	0	ISP	CY

(a) Interrupt enable flag (IE)

This flag controls interrupt request acknowledge operations of CPU.

When IE = 0, all interrupts, except the non-maskable interrupt, are disabled (DI status).

When IE = 1, the interrupts are enabled (EI status). At this time, acknowledging interrupts is controlled with an inservice priority flag (ISP) and an interrupt mask flag for various interrupt sources and a priority specify flag.

The interrupt enable flag is also reset to 0 when the DI instruction or an interrupt request has been acknowledged and is set to 1 when the EI instruction has been executed.

(b) Zero flag (Z)

When the operation result is zero, this flag is set (1). It is reset (0) in all other cases.

(c) Register bank select flags (RBS0 and RBS1)

These are 2-bit flags to select one of the four register banks.

In these flags, the 2-bit information which indicates the register bank selected by SEL RBn instruction execution is stored.

(d) Auxiliary carry flag (AC)

If the operation result has a carry from bit 3 or a borrow at bit 3, this flag is set (1). It is reset (0) in all other cases.

(e) In-service priority flag (ISP)

This flag manages the priority of acknowledgeable maskable vectored interrupts.

When ISP = 0, the vectored interrupt request specified by the priority specify flag registers (PR0L and PR0H) (refer to **12.3 (3) Priority specify flag registers (PR0L and PR0H)**) to have a low priority is disabled. Whether an interrupt request is actually acknowledged is controlled by the status of the interrupt enable flag (IE).

(f) Carry flag (CY)

This flag stores overflow and underflow upon add/subtract instruction execution. It stores the shift-out value upon rotate instruction execution and functions as a bit accumulator during bit manipulation instruction execution.

(3) Stack pointer (SP)

This is a 16-bit register to hold the start address of the memory stack area. Only the internal high-speed RAM area (FB00H to FEFFH) can be set as the stack area.

Figure 3-9. Stack Pointer Configuration

	15															0
SP	SP15	SP14	SP13	SP12	SP11	SP10	SP9	SP8	SP7	SP6	SP5	SP4	SP3	SP2	SP1	SP0

The SP is decremented ahead of write (save) to the stack memory and is incremented after read (restore) from the stack memory.

Each stack operation saves/restores data as shown in Figures 3-10 and 3-11.

Caution Since $\overline{\text{RESET}}$ input makes SP contents indeterminate, be sure to initialize the SP before instruction execution.

Figure 3-10. Data to be Saved to Stack Memory

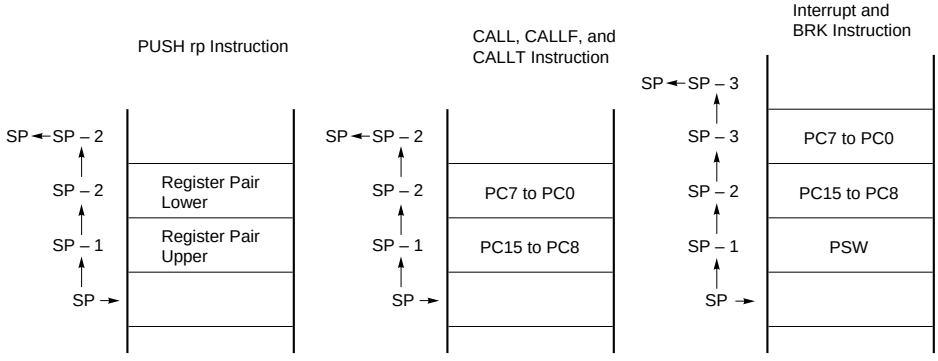
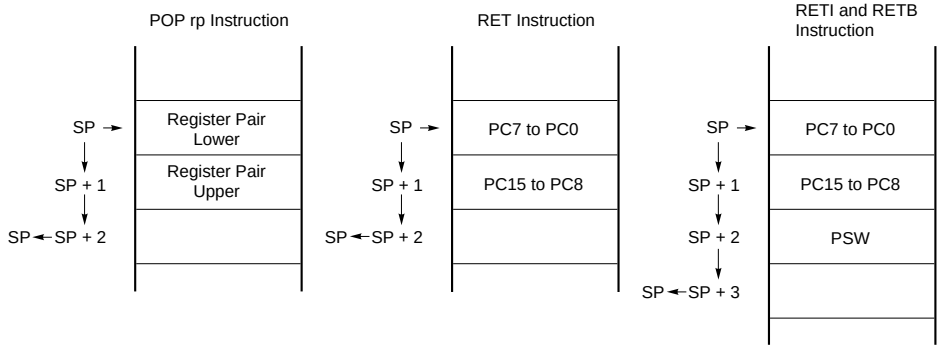


Figure 3-11. Data to be Restored from Stack Memory



3.2.2 General registers

A general register is mapped at particular addresses (FEE0H to FEFFH) of the data memory. It consists of 4 banks, each bank consisting of eight 8-bit registers (X, A, C, B, E, D, L, and H).

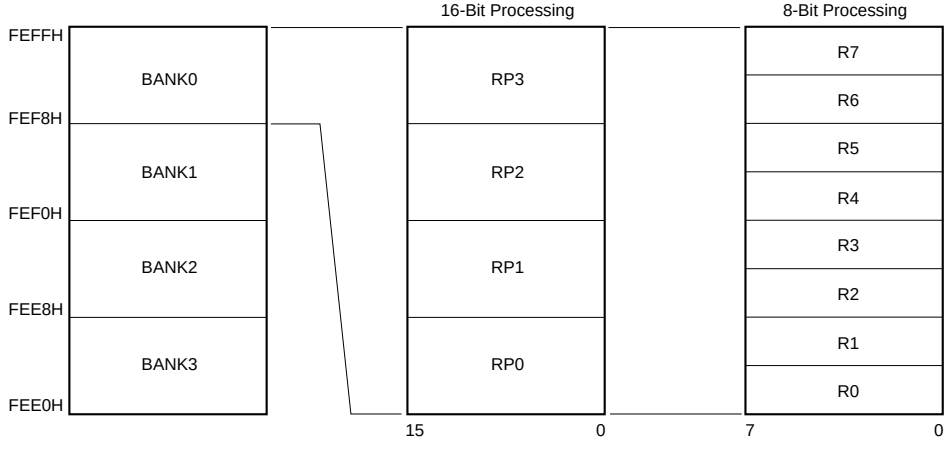
Each register can also be used as an 8-bit register. Two 8-bit registers can be used in pairs as a 16-bit register (AX, BC, DE, and HL).

They can be described in terms of function names (X, A, C, B, E, D, L, H, AX, BC, DE, and HL) and absolute names (R0 to R7 and RP0 to RP3).

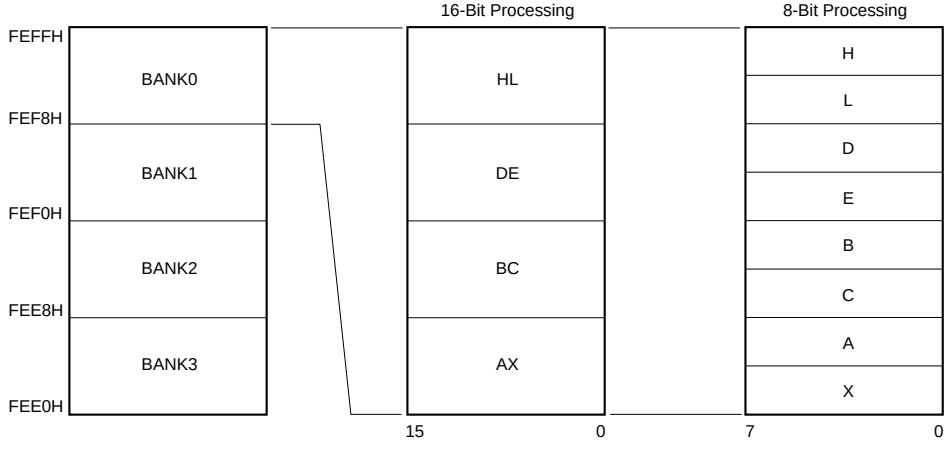
Register banks to be used for instruction execution are set with the CPU control instruction (SEL RBn). Because of the 4-register bank configuration, an efficient program can be created by switching between a register for normal processing and a register for interruption for each bank.

Figure 3-12. General Register Configuration

(a) Absolute Name



(b) Function Name



3.2.3 Special function registers (SFR: Special Function Register)

Unlike a general register, each special function register has special functions. It is allocated in the FF00H to FFFFH area.

The special function register can be manipulated, like the general register, with the operation, transfer and bit manipulation instructions. Manipulatable bit units, 1, 8, and 16, depend on the special function register type.

Each manipulation bit unit can be specified as follows.

- **1-bit manipulation**
  Describe the symbol reserved with assembler for the 1-bit manipulation instruction operand (sfr.bit).
  This manipulation can also be specified with an address.
- **8-bit manipulation**
  Describe the symbol reserved with assembler for the 8-bit manipulation instruction operand (sfr).
  This manipulation can also be specified with an address.
- **16-bit manipulation**
  Describe the symbol reserved with assembler for the 16-bit manipulation instruction operand (sfrp).
  When addressing an address, describe an even address.

Table 3-3 gives a list of special function registers. The meaning of items in the table is as follows.

- **Symbol**
  This is a symbol to indicate an address of the special function register.
  The symbols shown in this column are reserved words of the RA78K/0, and have already been defined in the header file called “sfrbit.h” of the CC78K/0. These are describable as instruction operands if the RA78K/0, ID78K0, or SD78K0 is used.
- **R/W**
  Indicates whether the corresponding special function register can be read or written.
  R/W : Read/write enable
  R : Read only
  W : Write only
- **Manipulatable bit units**
  ○ indicates the bit unit (1, 8, or 16 bits) in which the register can be manipulated. – indicates that the register cannot be manipulated in the indicated bit unit.
- **At reset**
  Indicates each register status upon $\overline{\text{RESET}}$ input.

Table 3-3. Special Function Registers (1/2)

Address	Special Function Register (SFR) Name	Symbol		R/W	Manipulatable Bit Units			At Reset
					1 bit	8 bits	16 bits	
FF00H	Port 0	P0		R/W	○	○	—	00H
FF01H	Port 1	P1		R	○	○	—	
FF02H	Port 2	P2		R/W	○	○	—	
FF04H	Port 4	P4			○	○	—	
FF05H	Port 5	P5			○	○	—	
FF06H	Port 6	P6			○	○	—	
FF07H	Port 7	P7			○	○	—	
FF08H	Port 8	P8			○	○	—	
FF09H	Port 9	P9			○	○	—	
FF0AH	Port 10	P10			○	○	—	
FF0BH	Port read 7	PLR7		R	○	○	—	Undefined
FF0CH	Port read 8	PLR8			○	○	—	
FF0DH	Port read 9	PLR9			○	○	—	
FF0EH	8-bit capture register 10	CP10			—	○	—	00H
FF0FH	8-bit capture register 11	CP11			—	○	—	
FF10H	8-bit compare register 50	CR50		R/W	—	○	—	Undefined
FF11H	8-bit compare register 51	CR51			—	○	—	
FF12H	8-bit counter 50	TM5	TM50	R	—	○	○	00H
FF13H	8-bit counter 51		TM51		—	○		
FF14H	A/D conversion result register	ADCRH0			—	○	—	Undefined
FF18H	Serial I/O shift register 3	SIO3		R/W	—	○	—	
FF20H	Port mode register 0	PM0			○	○	—	FFH
FF22H	Port mode register 2	PM2			○	○	—	
FF24H	Port mode register 4	PM4			○	○	—	
FF25H	Port mode register 5	PM5			○	○	—	
FF26H	Port mode register 6	PM6			○	○	—	
FF30H	Pull-up resistor option register 0	PU0			○	○	—	00H
FF32H	Pull-up resistor option register 2	PU2			○	○	—	
FF34H	Pull-up resistor option register 4	PU4			○	○	—	
FF42H	Watchdog timer clock select register	WDCS			○	○	—	
FF48H	External interrupt rising edge enable register	EGP			○	○	—	
FF49H	External interrupt falling edge enable register	EGN			○	○	—	
FF60H	Timer mode control register 1	TMC1			○	○	—	
FF70H	8-bit timer mode control register 50	TMC50			○	○	—	04H
FF71H	Timer clock select register 50	TCL50			○	○	—	00H
FF78H	8-bit timer mode control register 51	TMC51			○	○	—	04H
FF79H	Timer clock select register 51	TCL51			○	○	—	00H
FF80H	A/D converter mode register	ADM0			○	○	—	
FF81H	Analog input channel specification register	ADS0			○	○	—	
FF86H	Serial operation mode register 3	CSIM3			○	○	—	

Table 3-3. Special Function Registers (2/2)

Address	Special Function Register (SFR) Name	Symbol		R/W	Manipulatable Bit Units			At Reset
					1 bit	8 bits	16 bits	
FF90H	Display mode register 0	DSPM0		R/W	○	○	—	10H
FF91H	Display mode register 1	DSPM1			○	○	—	01H
FF92H	Display mode register 2	DSPM2			○	○	—	00H
FFE0H	Interrupt request flag register 0L	IF0	IF0L		○	○	○	00H
FFE1H	Interrupt request flag register 0H		IF0H		○	○		
FFE4H	Interrupt mask flag register 0L	MK0	MK0L		○	○	○	FFH
FFE5H	Interrupt mask flag register 0H		MK0H		○	○		
FFE8H	Priority specification flag register 0L	PR0	PR0L		○	○	○	
FFE9H	Priority specification flag register 0H		PR0H		○	○		
FFF0H	Memory size select register	IMS			—	○	—	CCH[Note 1]
★ FFF4H	Internal expansion RAM size select register	IXS			—	○	—	0CH[Note 2]
FFF9H	Watchdog timer mode register	WDTM			○	○	—	00H
FFFAH	Oscillation stabilization time select register	OSTS			○	○	—	04H
FFFBH	Processor clock control register	PCC			○	○	—	

Notes 1. Be sure to set the value of this register to CFH when the μPD780228 is used.

2. After reset, be sure to set this bit to 0BH.

3.3 Addressing of Instruction Address

An instruction address is determined by program counter (PC) contents. Program counter (PC) contents are normally incremented (+1 for each byte) automatically according to the number of bytes of an instruction to be fetched each time another instruction is executed. When a branch instruction is executed, the branch destination information is set to the PC and branched by the following addressing (For details of instructions, refer to **78K/0 Series User's Manual: Instructions (U12326E)**.

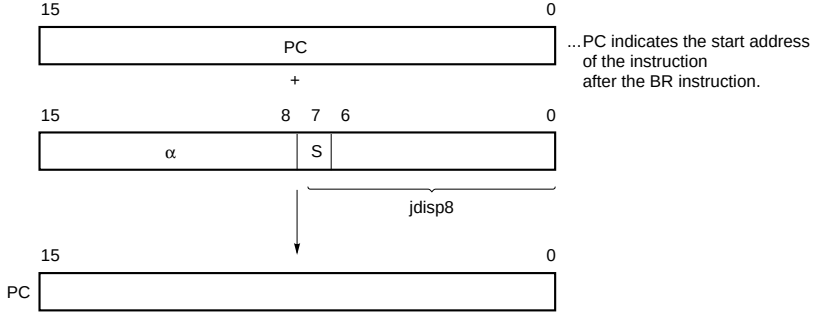
3.3.1 Relative addressing

[Function]

The value obtained by adding 8-bit immediate data (displacement value: jdisp8) of an instruction code to the start address of the following instruction is transferred to the program counter (PC) and branched. The displacement value is treated as signed two's complement data (–128 to +127) and bit 7 becomes a sign bit. In the relative addressing modes, execution branches in a relative range of –128 to +127 from the first address of the next instruction.

This function is carried out when the BR $addr16 instruction or a conditional branch instruction is executed.

[Illustration]



When S = 0, all bits of α are 0.
When S = 1, all bits of α are 1.

3.3.2 Immediate addressing

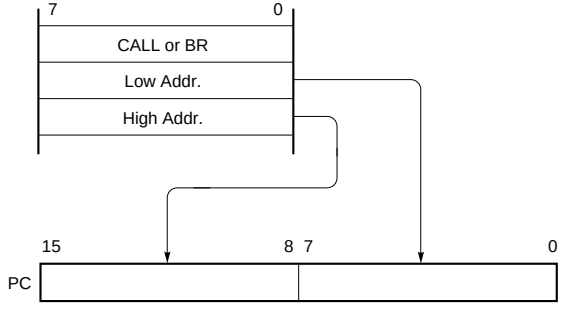
[Function]

Immediate data in the instruction word is transferred to the program counter (PC) and branched.

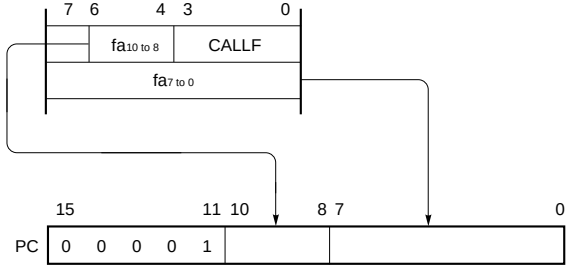
This function is carried out when the CALL !addr16 or BR !addr16 or CALLF !addr11 instruction is executed. The CALL !addr16 and BR !addr16 instruction can branch in the entire memory space. The CALLF !addr11 instruction branches to an area of addresses 0800H to 0FFFH.

[Illustration]

In the case of CALL !addr16 and BR !addr16 instructions



In the case of CALLF !addr11 instruction



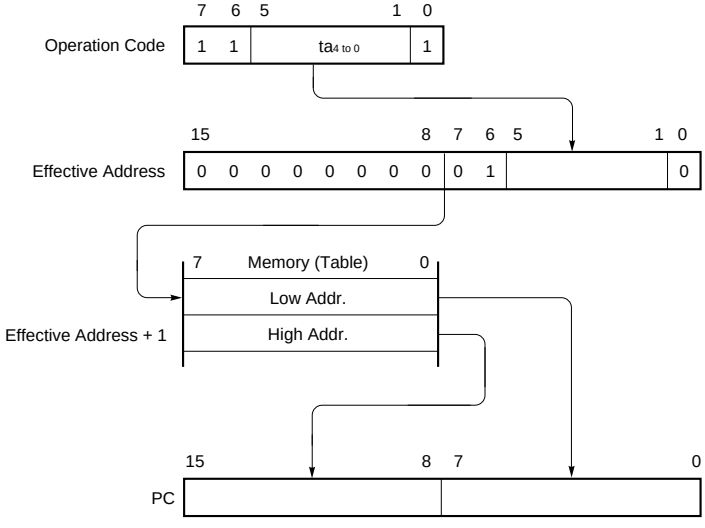
3.3.3 Table indirect addressing

[Function]

Table contents (branch destination address) of the particular location to be addressed by bits 1 to 5 of the immediate data of an operation code are transferred to the program counter (PC) and branched.

Before the CALLT [addr5] instruction is executed, table indirect addressing is performed. This instruction references an address stored in the memory table at addresses 40H to 7FH, and can branch in the entire memory space.

[Illustration]



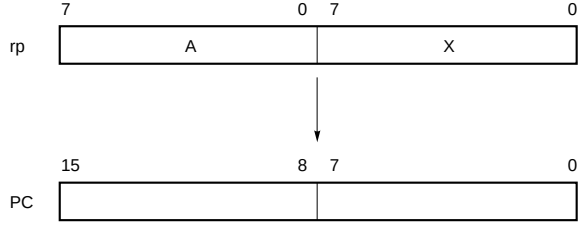
3.3.4 Register addressing

[Function]

Register pair (AX) contents to be specified with an instruction word are transferred to the program counter (PC) and branched.

This function is carried out when the BR AX instruction is executed.

[Illustration]



3.4 Addressing of Operand Address

The following various methods are available to specify the register and memory (addressing) which undergo manipulation during instruction execution.

3.4.1 Implied addressing

[Function]

The register which functions as an accumulator (A and AX) in the general register is automatically (tacitly) addressed.

Of the μPD780228 subseries instruction words, the following instructions employs implied addressing.

Instruction	Register to be Specified by Implied Addressing
MULU	A register for multiplicand and AX register for product storage
DIVUW	AX register for dividend and quotient storage
ADJBA/ADJBS	A register for storage of numeric values subject to decimal adjustment
ROR4/ROL4	A register for storage of digit data which undergoes digit rotation

[Operand format]

Because implied addressing can be automatically employed with an instruction, no particular operand format is necessary.

[Description example]

In the case of MULU X

With an 8-bit x 8-bit multiply instruction, the product of A register and X register is stored in AX. In this example, the A and AX registers are specified by implied addressing.

3.4.2 Register addressing

[Function]

This addressing accesses a general register as an operand. The general register accessed is specified by the register bank select flags (RBS0 and RBS1) and register specify code (Rn or RPn) in an instruction code. Register addressing is carried out when an instruction with the following operand format is executed. When an 8-bit register is specified, one of the eight registers is specified with 3 bits in the operation code.

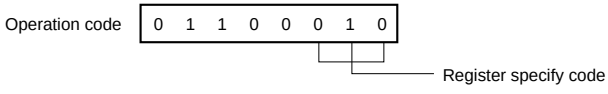
[Operand format]

Identifier	Description
r	X, A, C, B, E, D, L, H
rp	AX, BC, DE, HL

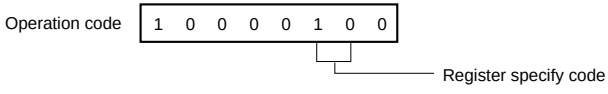
'r' and 'rp' can be described with function names (X, A, C, B, E, D, L, H, AX, BC, DE, and HL) as well as absolute names (R0 to R7 and RP0 to RP3).

[Description example]

MOV A, C; when selecting C register as r



INCW DE; when selecting DE register pair as rp



3.4.3 Direct addressing

[Function]

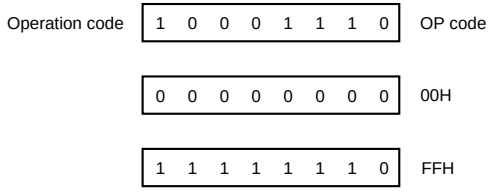
This addressing directly addresses the memory indicated by the immediate data in an instruction word.

[Operand format]

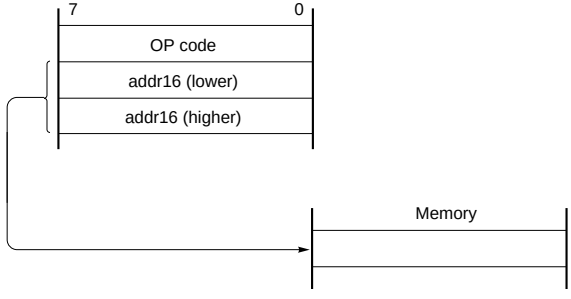
Identifier	Description
addr16	Label or 16-bit immediate data

[Description example]

MOV A, !0FE00H; when setting !addr16 to FE00H



[Illustration]



3.4.4 Short direct addressing

[Function]

The memory to be manipulated in the fixed space is directly addressed with 8-bit data in an instruction word. The fixed space to which this addressing is applied is the 256-byte space of addresses FE20H to FF1FH. Addresses FE20H to FEFFH constitute a part of the SFR area, and the internal high-speed RAM is mapped to this area. The special function registers (SFRs) are mapped to the area from addresses FF00H to FF1FH. If the SFR area (FF00H to FF1FH) where short direct addressing is applied, ports which are frequently accessed in a program and a compare register of the timer/event counter and a capture register of the timer/event counter are mapped and these SFRs can be manipulated with a small number of bytes and clocks.

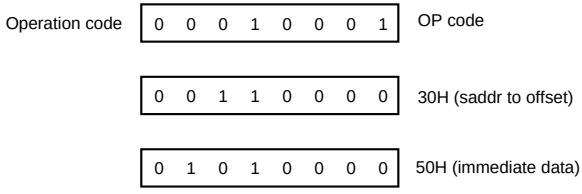
When 8-bit immediate data is at 20H to FFH, bit 8 of an effective address is set to 0. When it is at 00H to 1FH, bit 8 is set to 1. Refer to **[Illustration]** on the next page.

[Operand format]

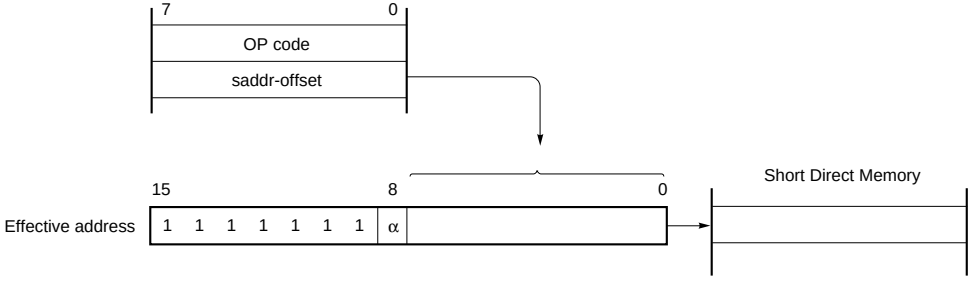
Identifier	Description
saddr	Label or FE20H to FF1FH immediate data
saddrp	Label of FE20H to FF1FH immediate data (even address only)

[Description example]

MOV 0FE30H, #50H; when setting saddr to FE30H and immediate data to 50H



[Illustration]



When 8-bit immediate data is 20H to FFH, $\alpha = 0$

When 8-bit immediate data is 00H to 1FH, $\alpha = 1$

3.4.5 Special function register (SFR) addressing

[Function]

The memory-mapped special function register (SFR) is addressed with 8-bit immediate data in an instruction word.

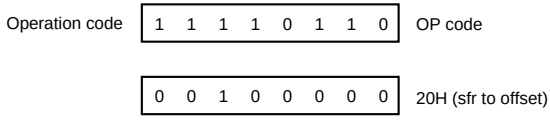
This addressing is applied to the 240-byte spaces FF00H to FFCFH and FFE0H to FFFFH. However, the SFR mapped at FF00H to FF1FH can be accessed with short direct addressing.

[Operand format]

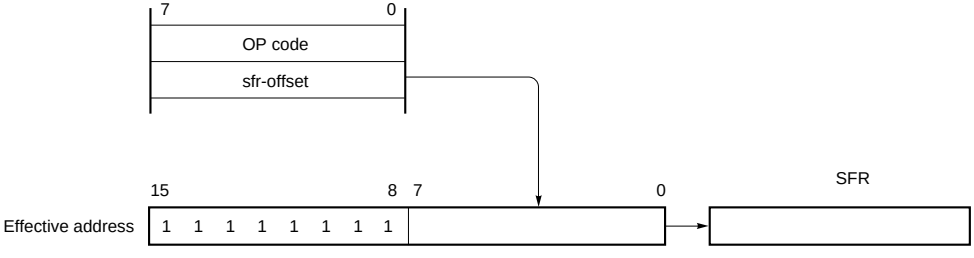
Identifier	Description
sfr	Special function register name
sfrp	16-bit manipulatable special function register name (even address only)

[Description example]

MOV PM0, A; when selecting PM0 (FF20H) as sfr



[Illustration]



3.4.6 Register indirect addressing

[Function]

The addressing addresses the memory with the contents of a register pair specified as an operand. The register pair to be accessed is specified by the register bank select flags (RBS0 and RBS1) and register pair specify code in an instruction code. This addressing can be carried out for all the memory spaces.

[Operand format]

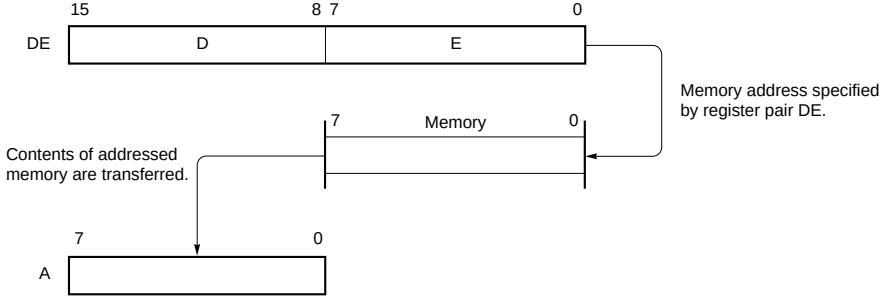
Identifier	Description
–	[DE], [HL]

[Description example]

MOV A, [DE]; when selecting [DE] as register pair

Operation code	1	0	0	0	0	1	0	1

[Illustration]



3.4.7 Based addressing

[Function]

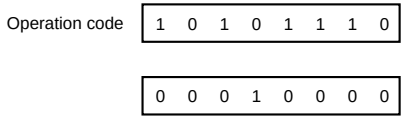
This addressing addresses the memory by adding 8-bit immediate data to the contents of the HL register pair which is used as a base register and by using the result of the addition. The HL register pair to be accessed is in the register bank specified by the register bank select flags (RBS0 and RBS1). Addition is performed by expanding the offset data as a positive number to 16 bits. A carry from the 16th bit is ignored. This addressing can be carried out for all the memory spaces.

[Operand format]

Identifier	Description
—	[HL+byte]

[Description example]

MOV A, [HL+10H]; When setting byte to 10H



3.4.8 Based indexed addressing

[Function]

This addressing addresses the memory by adding the contents of the HL register pair, which is used as a base register, to the contents of the B or C register specified in the instruction word, and by using the result of the addition. The HL, B, and C registers to be accessed are registers in the register bank specified by the register bank select flags (RBS0 and RBS1). The addition is performed by extending the contents of the B or C register to 16 bits as a positive number. A carry from the 16th bit is ignored. This addressing can be carried out for all the memory spaces.

[Operand format]

Identifier	Description
—	[HL+B], [HL+C]

[Description example]

In the case of MOV A, [HL+B]

Operation code	1	0	1	0	1	0	1	1

3.4.9 Stack addressing

[Function]

The stack area is indirectly addressed with the stack pointer (SP) contents.

This addressing method is automatically employed when the PUSH, POP, subroutine call and RETURN instructions are executed or the register is saved/reset upon generation of an interrupt request.

Stack addressing enables to address the internal high-speed RAM area only.

[Description example]

In the case of PUSH DE

Operation code	1	0	1	1	0	1	0	1

[MEMO]

CHAPTER 4 PORT FUNCTIONS

4.1 Port Functions

The μPD780228 subseries incorporates eight input ports, eight output ports and 56 input/output ports. Figure 4-1 shows the port configuration. Every port is capable of 1-bit and 8-bit manipulations and can carry out considerably varied control operations. Besides port functions, the ports can also serve as built-in hardware input/output pins.

Figure 4-1. Port Types

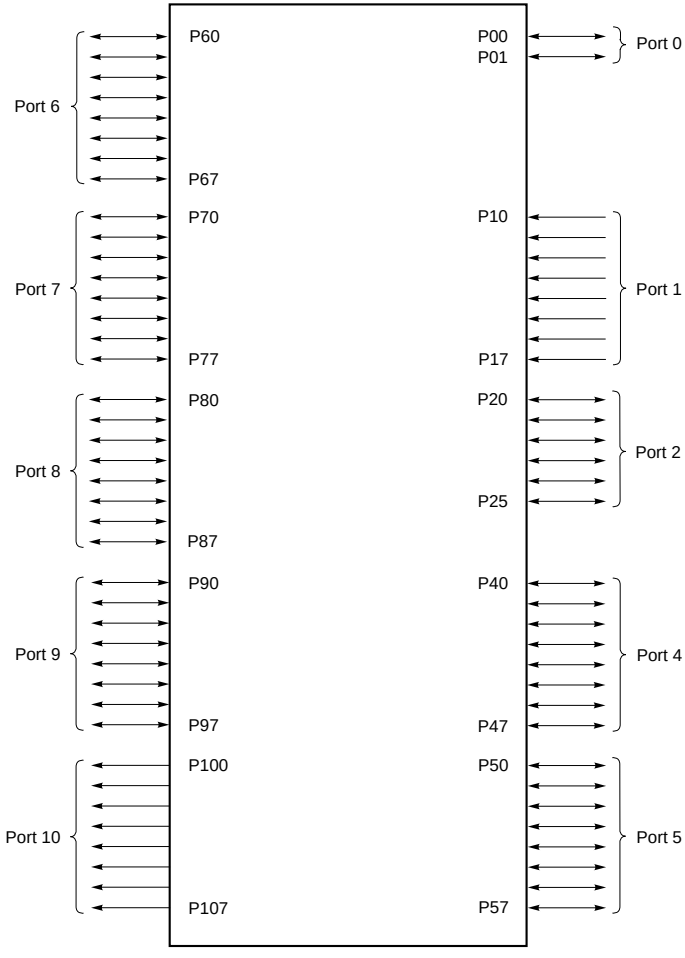


Table 4-1. Port Function

Pin Name	Function	Shared with:
P00	Port 0. 2-bit I/O port. Can be set in input or output mode in 1-bit units. Internal pull-up resistor can be used via software when this port is used as input port.	INTP0
P01		INTP1
P10-P17	Port 1. 8-bit input port.	ANI0-ANI7
P20	Port 2. 6-bit I/O port. Can be set in input or output mode in 1-bit units. Internal pull-up resistor can be used via software when this port is used as input port.	$\overline{SCK}$
P21		SO
P22		SI
P23		TI1
P24		TIO50
P25		TIO51
P40-P47	Port 4. 8-bit I/O port. Can be set in input or output mode in 1-bit units. Can directly drive LED. Internal pull-up resistor can be used via software when this port is used as input port.	—
P50-P57	Port 5. N-ch open-drain 8-bit medium-voltage I/O port. Can be set in input or output mode in 1-bit units. Can directly drive LED. Internal pull-up resistor can be used by mask option in 1-bit units (mask ROM models only). μPD78F0228 does not have pull-up resistors, however.	—
P60-P67	Port 6. N-ch open-drain 8-bit medium-voltage I/O port. Can be set in input or output mode in 1-bit units. Can directly drive LED. Internal pull-up resistor can be used by mask option in 1-bit units (mask ROM models only). μPD78F0228 does not have pull-up resistors, however.	—
P70-P77	Port 7. P-ch open-drain 8-bit high-voltage I/O port. Can be set in input or output mode in 1-bit units. Internal pull-down resistor can be used by mask option in 1-bit units (mask ROM models only). μPD78F0228 does not have pull-down resistors, however.	FIP16-FIP23
P80-P87	Port 8. P-ch open-drain 8-bit high-voltage I/O port. Can be set in input or output mode in 1-bit units. Internal pull-down resistor can be used by mask option in 1-bit units (mask ROM models only). μPD78F0228 does not have pull-down resistors, however.	FIP24-FIP31
P90-P97	Port 9. P-ch open-drain 8-bit high-voltage I/O port. Can be set in input or output mode in 1-bit units. Internal pull-down resistor can be used by mask option in 1-bit units (mask ROM models only). μPD78F0228 does not have pull-down resistors, however.	FIP32-FIP39
P100-P107	Port 10. P-ch open-drain 8-bit high-voltage output port. Internal pull-down resistor can be used by mask option in 1-bit units (mask ROM models only). μPD78F0228 does not have pull-down resistors, however.	FIP40-FIP47

4.2 Port Configuration

A port consists of the following hardware.

Table 4-2. Port Configuration

Item	Configuration
Control register	Port mode register (PMm: m = 0, 2, 4-6) Pull-up resistor option register (PUn: n = 0, 2, 4)
Port	Total: 72 (8 inputs, 8 outputs, 56 inputs/outputs)
Pull-up resistor	• Mask ROM model Total: 32 (software control: 16, mask option control: 16) • μPD78F0228 Total: 16
Pull-down resistor	• Mask ROM product Total: 32 (mask option control: 32) • μPD78F0228 None

4.2.1 Port 0

Port 0 is a 2-bit input/output port with output latch. P00 and P01 pins can specify the input mode/output mode in 1-bit units with the port mode register 0 (PM0). When P00 and P01 pins are used as input ports, an on-chip pull-up resistor can be connected to them in 1-bit units with a pull-up resistor option register 0 (PU0).

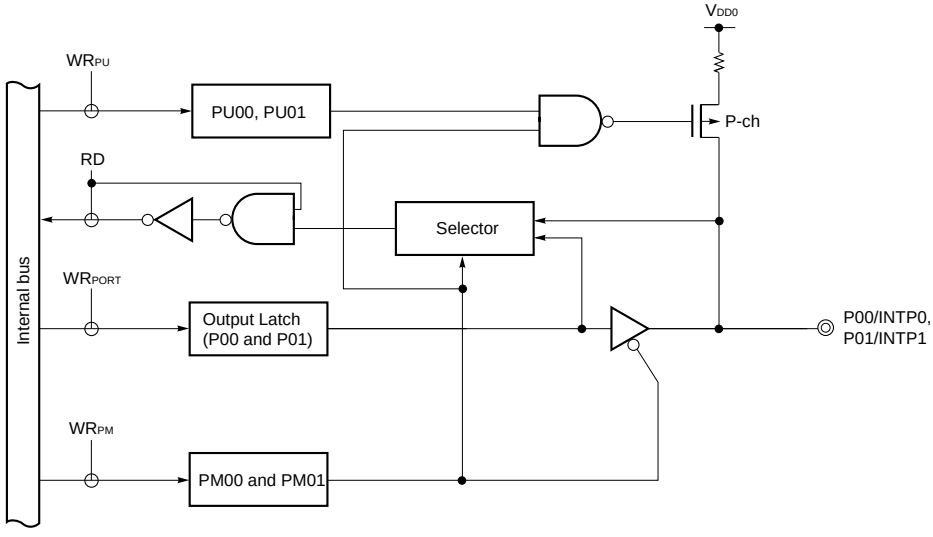
Alternate functions include external interrupt request input.

$\overline{\text{RESET}}$ input sets port 0 to input mode.

Figure 4-2 shows a block diagram of port 0.

Caution Because port 0 also serves for external interrupt request input, when the port function output mode is specified and the output level is changed, the interrupt request flag is set. Thus, when the output mode is used, set the interrupt mask flag to 1.

Figure 4-2. P00 and P01 Block Diagram



PU : Pull-up resistor option register
PM : Port mode register
RD : Port 0 read signal
WR : Port 0 write signal

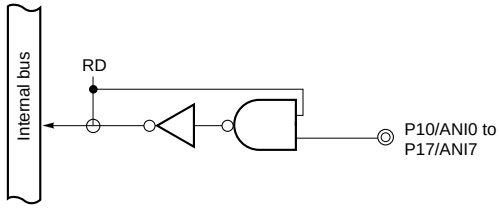
4.2.2 Port 1

Port 1 is an 8-bit input only port.

A/D converter analog input is provided as an alternate function.

Figure 4-3 shows a block diagram of port 1.

Figure 4-3. P10 to P17 Block Diagram



RD : Port 1 read signal

4.2.3 Port 2

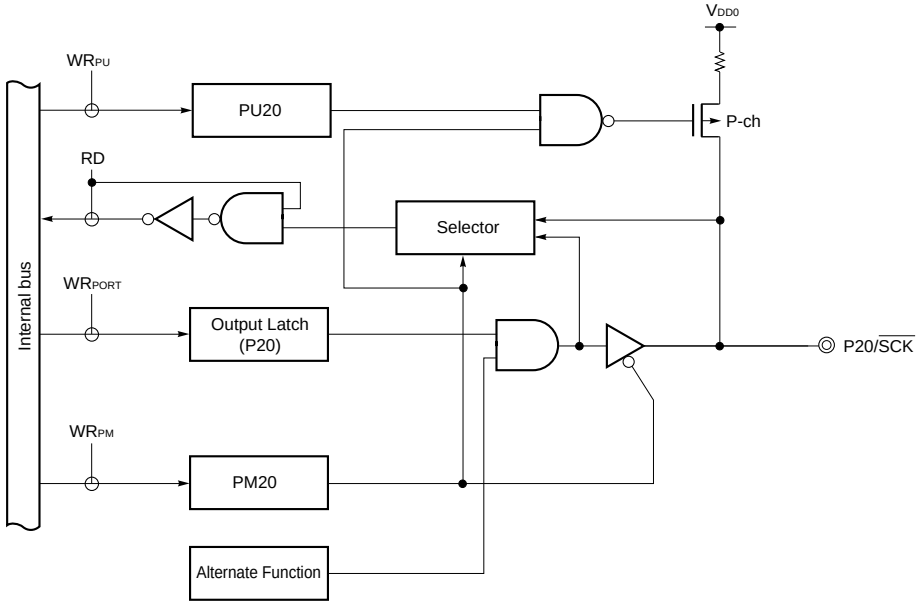
Port 2 is a 6-bit input/output port with output latch. P20 to P25 pins can specify the input mode/output mode in 1-bit units with the port mode register 2 (PM2). When P20 to P25 pins are used as input ports, an on-chip pull-up resistor can be connected to them in 1-bit units with a pull-up resistor option register 2 (PU2).

Alternate functions include serial interface data input/output, clock input/output, and timer input/output.

$\overline{\text{RESET}}$ input sets port 2 to input mode.

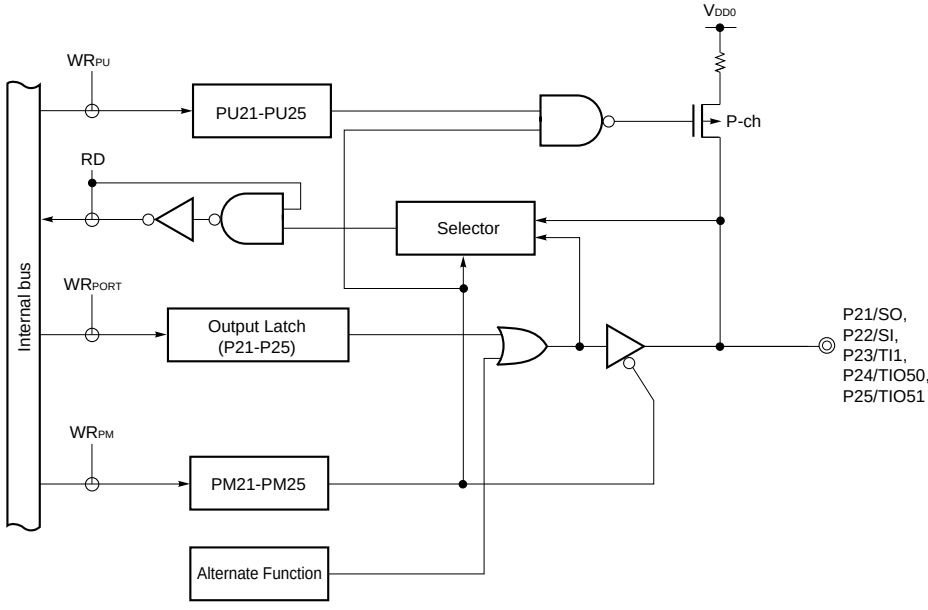
Figures 4-4 and 4-5 show block diagrams of port 2.

Figure 4-4. P20 Block Diagram



PU : Pull-up resistor option register
PM : Port mode register
RD : Port 2 read signal
WR : Port 2 write signal

Figure 4-5. P21 to P25 Block Diagram



PU : Pull-up resistor option register
PM : Port mode register
RD : Port 2 read signal
WR : Port 2 write signal

4.2.4 Port 4

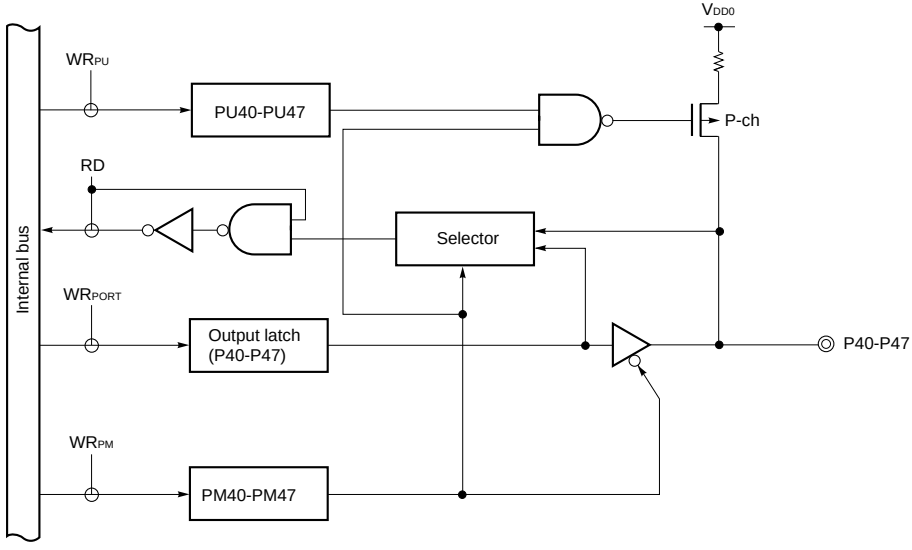
Port 4 is an 8-bit input/output port with output latch. P40 to P47 pins can specify the input mode/output mode in 1-bit units with the port mode register 4 (PM4). When P40 to P47 pins are used as input ports, an on-chip pull-up resistor can be connected to them in 1-bit units with a pull-up resistor option register 4 (PU4).

Port 4 can drive LEDs directly.

$\overline{\text{RESET}}$ input sets port 4 to input mode.

Figure 4-6 shows a block diagram of port 4.

Figure 4-6. P40 to P47 Block Diagram



PU : Pull-up resistor option register
PM : Port mode register
RD : Port 4 read signal
WR : Port 4 write signal

4.2.5 Port 5

Port 5 is an 8-bit input/output port with output latch. Pins from 50 to 57 can specify I/O mode in 1-bit units with the port mode register 5 (PM5). On-chip pull-up resistors can be connected in 1-bit units with the mask option in case of mask ROM model. The μPD78F0228 has no pull-up resistor.

Port 5 can drive LEDs directly.

$\overline{\text{RESET}}$ input sets port 5 to input mode.

Figure 4-7 shows a block diagram of port 5.

Caution Low-level input leak current in P50 to P57 pins differs depending on the following conditions:

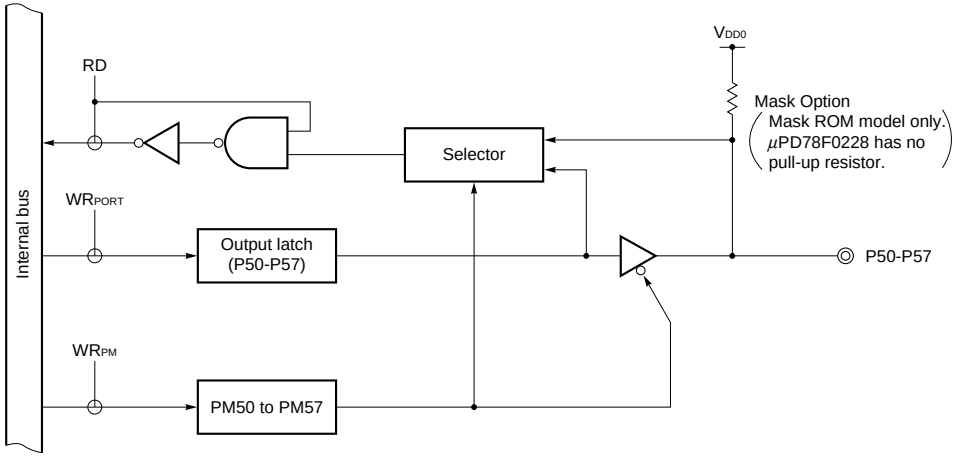
[Mask ROM model]

- **When pull-up resistors are contained : always –3 μA (Max.)**
- **When pull-up resistors are not contained :**
  - **1 clock interval when read instruction is executed to port 5 (P5) and port mode register 5 (PM5) : –200 μA (Max.)**
  - **Other than above : –3 μA (Max.)**

[Flash memory model]

- **1 clock interval when read instruction is executed to port 5 (P5) and port mode register 5 (PM5) : –200 μA (Max.)**
- **Other than above : –3 μA (Max.)**

Figure 4-7. P50 to P57 Block Diagram



PM : Port mode register

RD : Port 5 read signal

WR : Port 5 write signal

4.2.6 Port 6

Port 6 is an 8-bit input/output port with output latch. Pins from 60 to 67 can specify I/O mode in 1-bit units with the port mode register 6 (PM6). On-chip pull-up resistors can be connected in 1-bit units with the mask option in case of mask ROM model. The μPD78F0228 has no pull-up resistor.

Port 6 can drive LEDs directly.

$\overline{\text{RESET}}$ input sets port 6 to input mode.

Figure 4-8 shows a block diagram of port 6.

Caution Low-level input leak current in P60 to P67 pins differs depending on the following conditions:

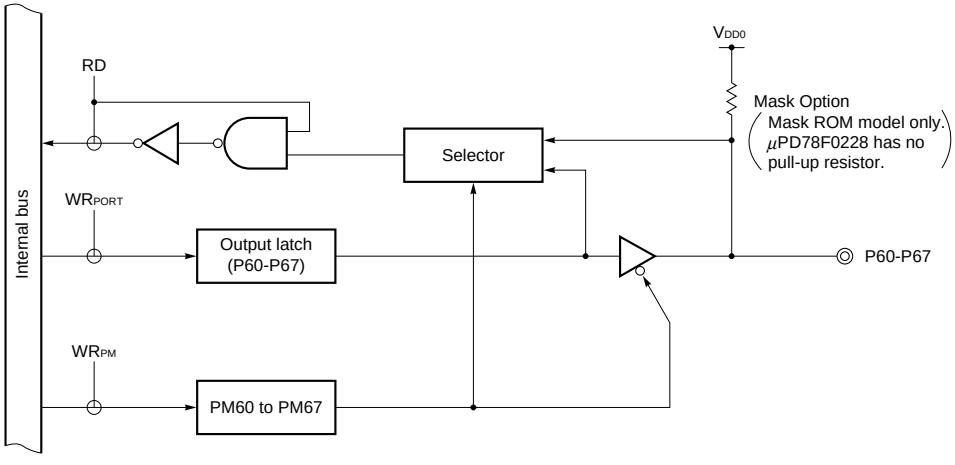
[Mask ROM model]

- **When pull-up resistors are contained : always –3 μA (Max.)**
- **When pull-up resistors are not contained :**
  - **1 clock interval when read instruction is executed to port 6 (P6) and port mode register 6 (PM6) : –200 μA (Max.)**
  - **Other than above : –3 μA (Max.)**

[Flash memory model]

- **1 clock interval when read instruction is executed to port 6 (P6) and port mode register 6 (PM6) : –200 μA (Max.)**
- **Other than above : –3 μA (Max.)**

Figure 4-8. P60 to P67 Block Diagram



PM : Port mode register

RD : Port 6 read signal

WR : Port 6 write signal

4.2.7 Port 7

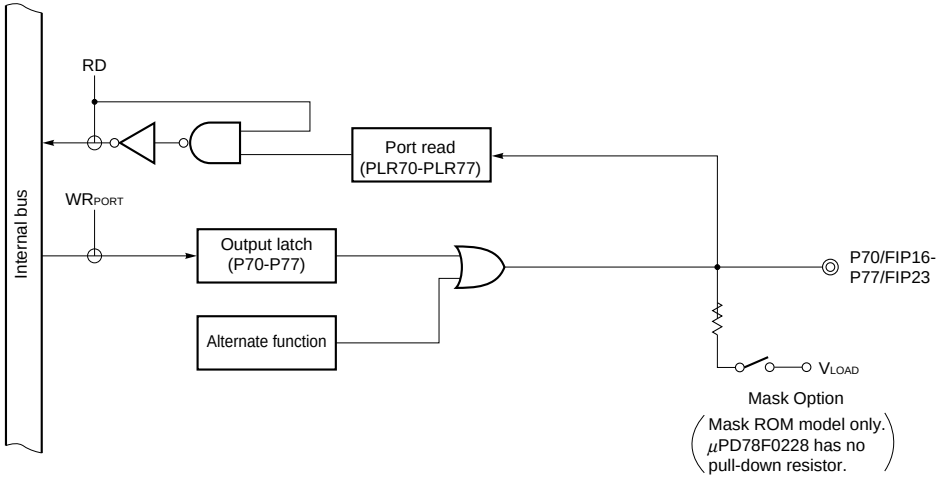
Port 7 is an 8-bit input/output port with output latch. When using this port as an output port, the value assigned to the output latch (P70 through P77) is output. When it is used as an input port, set the output latch (P70 through P77) to "0", and read the port read (PLR70 through PLR77). On-chip pull-down resistors can be connected in 1-bit units with the mask option. The μPD78F0228 has no pull-down resistor.

In addition, FIP controller/driver output is provided as an alternate function.

$\overline{\text{RESET}}$ input sets port 7 to input mode.

Figure 4-9 shows a block diagram of port 7.

★ **Figure 4-9. P70 to P77 Block Diagram**



RD : Port 7 read signal
WR : Port 7 write signal

4.2.8 Port 8

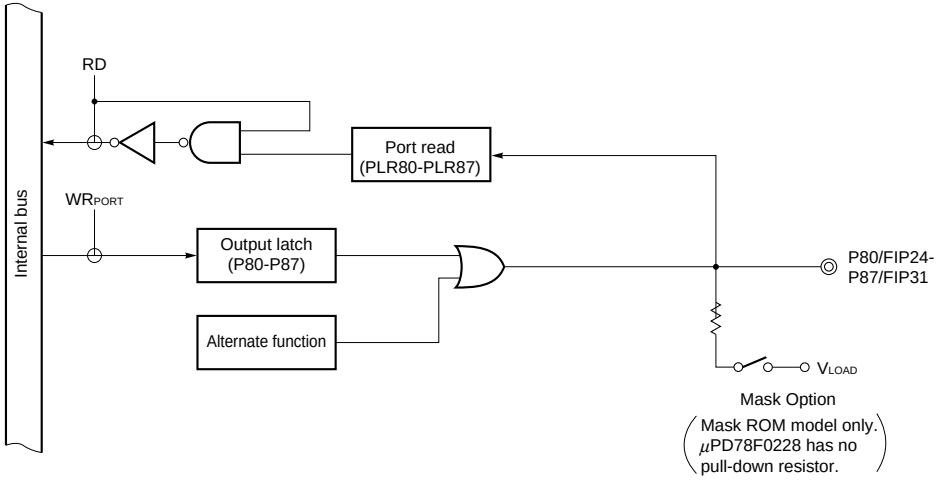
Port 8 is an 8-bit input/output port with output latch. When using this port as an output port, the value assigned to the output latch (P80 through P87) is output. When it is used as an input port, set the output latch (P80 through P87) to "0", and read the port read (PLR80 through PLR87). On-chip pull-down resistors can be connected in 1-bit units with the mask option. The μPD78F0228 has no pull-down resistor.

In addition, FIP controller/driver output is provided as an alternate function.

$\overline{\text{RESET}}$ input sets port 8 to input mode.

Figure 4-10 shows a block diagram of port 8.

★ **Figure 4-10. P80 to P87 Block Diagram**



RD : Port 8 read signal
WR : Port 8 write signal

4.2.9 Port 9

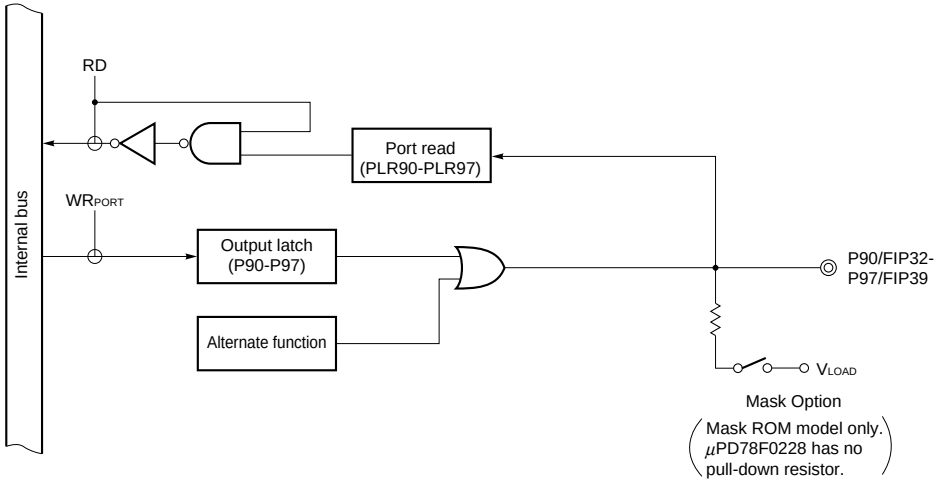
Port 9 is an 8-bit input/output port with output latch. When using this port as an output port, the value assigned to the output latch (P90 through P97) is output. When it is used as an input port, set the output latch (P90 through P97) to "0", and read the port read (PLR90 through PLR97). On-chip pull-down resistors can be connected in 1-bit units with the mask option. The μPD78F0228 has no pull-down resistor.

In addition, FIP controller/driver output is provided as an alternate function.

$\overline{\text{RESET}}$ input sets port 9 to input mode.

Figure 4-11 shows a block diagram of port 9.

★ **Figure 4-11. P90 to P97 Block Diagram**



RD : Port 9 read signal
WR : Port 9 write signal

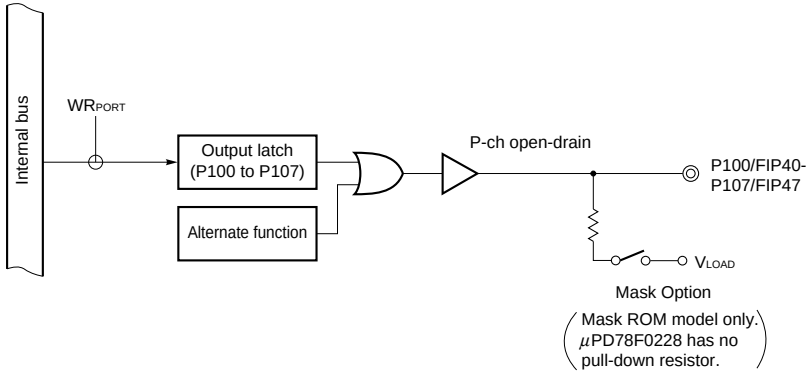
4.2.10 Port 10

Port 10 is an 8-bit output only port. On-chip pull-down resistors can be connected in 1-bit units with the mask option in case of mask ROM model. The μPD78F0228 has no pull-down resistor.

In addition, FIP controller/driver segment/digit output is provided as an alternate function.

Figure 4-12 shows a block diagram of port 10.

★ **Figure 4-12. P100 to P107 Block Diagram**



WR: Port 10 write signal

4.3 Port Function Control Registers

The following two types of registers control the ports.

- Port mode registers (PM0, PM2, PM4 to PM6)
- Pull-up resistor option register (PU0, PU2, PU4)

(1) Port mode registers (PM0, PM2, PM4 to PM6)

These registers are used to set port input/output in 1-bit units.

PM0, PM2 and PM4 to PM6 are independently set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets these registers to FFH.

When a port pin is used as its alternate function pin, set the port mode register and the output latch according to Table 4-3.

Cautions 1. Pins P10 to P17 are input-only pins.

2. Pins P100 to P107 are output-only pins.

3. As port 0 has an alternate function as external interrupt request input, when the port function output mode is specified and the output level is changed, the interrupt request flag is set. When the output mode is used, therefore, the interrupt mask flag should be set to 1 beforehand.

Table 4-3. Port Mode Register and Output Latch Setting when Alternate Function is Used

	Pin Name	Alternate Function		PMxx	Pxx	Pin Name	Alternate Function		PMxx	Pxx
		Function Name	Input/output				Function Name	Input/output		
	P00	INTP0	input	1	×	P23	TI1	input	1	×
	P01	INTP1	input	1	×	P24	TIO50	input	1	×
★	P20	$\overline{\text{SCK}}$	I/O	**Note**	1			output	0	0
★	P21	SO	Output	0	0	P25	TIO51	input	1	×
★	P22	SI	Input	1	0			output	0	0

Note The setting of PMxx varies depending on the clock selected by bits 1 and 0 (SCL31 and SCL30) of the serial operation mode register (CSIM3).

Internal clock (SCL31, SCL30 • 00): 0

External clock (SCL31, SCL30 = 00): 1

Remark × : Don't care

PMxx: Port mode register

Pxx : Port output latch

Figure 4-13. Port Mode Register Format

Symbol	7	6	5	4	3	2	1	0	Address	At Reset	R/W
PM0	1	1	1	1	1	1	PM01	PM00	FF20H	FFH	R/W
PM2	1	1	PM25	PM24	PM23	PM22	PM21	PM20	FF22H	FFH	R/W
PM4	PM47	PM46	PM45	PM44	PM43	PM42	PM41	PM40	FF24H	FFH	R/W
PM5	PM57	PM56	PM55	PM54	PM53	PM52	PM51	PM50	FF25H	FFH	R/W
PM6	PM67	PM66	PM65	PM64	PM63	PM62	PM61	PM60	FF26H	FFH	R/W

PMmn	Pmn Pin Input/Output Mode Selection (m = 0: n = 0, 1) (m = 2: n = 0-5) (m = 4-6: n = 0-7)
0	Output mode (output buffer ON)
1	Input mode (output buffer OFF)

(2) Pull-up resistor option registers (PU0, PU2, PU4)

This register is used to set whether or not to use an internal pull-up resistor of pins at ports 0, 2, 4 in 1-bit units. A pull-up resistor is internally used at bits which are set to the input mode at a bit where on-chip pull-up resistor use has been specified with PU0, PU2, and PU4. No on-chip pull-up resistors can be used to the bits set to the output mode irrespective of PU0, PU2, PU4 setting.

PU0, PU2, and PU4 are set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets this register to 00H.

Figure 4-14. Pull-Up Resistor Option Register Format

Symbol	7	6	5	4	3	2	1	0	Address	At Reset	R/W
PU0	0	0	0	0	0	0	PU01	PU00	F F 3 0 H	0 0 H	R/W
PU2	0	0	PU25	PU24	PU23	PU22	PU21	PU20	F F 3 2 H	0 0 H	R/W
PU4	PU47	PU46	PU45	PU44	PU43	PU42	PU41	PU40	F F 3 4 H	0 0 H	R/W

PUmn	Pmn Internal Pull-up Resistor Selection (m = 0 : n = 0, 1) (m = 2 : n = 0-5) (m = 4 : n = 0-7)
0	Internal pull-up resistor not used
1	Internal pull-up resistor used

4.4 Port Function Operations

Port operations differ depending on whether the input or output mode is set, as shown below.

4.4.1 Writing to input/output port

(1) Output mode

A value is written to the output latch by a transfer instruction, and the output latch contents are output from the pin.

Once data is written to the output latch, it is retained until data is written to the output latch again.

(2) Input mode

A value is written to the output latch by a transfer instruction, but since the output buffer is OFF, the pin status does not change.

Once data is written to the output latch, it is retained until data is written to the output latch again.

Caution In the case of 1-bit memory manipulation instruction, although a single bit is manipulated the port is accessed as an 8-bit unit. Therefore, on a port with a mixture of input and output pins, the output latch contents for pins specified as input are undefined except for the manipulated bit.

4.4.2 Reading from input/output port

(1) Output mode

The output latch contents are read by a transfer instruction. The output latch contents do not change.

(2) Input mode

The pin status is read by a transfer instruction. The output latch contents do not change.

4.4.3 Operations on input/output port

(1) Output mode

An operation is performed on the output latch contents, and the result is written to the output latch. The output latch contents are output from the pins.

Once data is written to the output latch, it is retained until data is written to the output latch again.

(2) Input mode

The output latch contents are undefined, but since the output buffer is OFF, the pin status does not change.

Caution In the case of 1-bit memory manipulation instruction, although a single bit is manipulated the port is accessed as an 8-bit unit. Therefore, on a port with a mixture of input and output pins, the output latch contents for pins specified as input are undefined except for the manipulated bit.

4.5 Selecting Mask Option

The mask ROM models have the following mask options. The μPD78F0228 does not have mask options.

Table 4-4. Comparison between Mask Options of Mask ROM Models and μPD78F0228

Pin Name	Mask Option of Mask ROM Model	μPD78F0228
P50-P57, P60-P67	Pull-up resistor can be connected in 1-bit units.	Pull-up resistor is not provided.
P70/FIP16-P77/FIP23, P80/FIP24-P87/FIP31, P90/FIP32-P97/FIP39, P100/FIP40-P107/FIP47	Pull-down resistor can be connected in 1-bit units.	Pull-down resistor is not provided.

[MEMO]

CHAPTER 5 CLOCK GENERATOR

5.1 Clock Generator Functions

The clock generator generates clock to be supplied to the CPU and peripheral hardware. The following type of system clock oscillators is available.

- **Main system clock oscillator**
  This circuit oscillates at frequencies of 5.0 MHz. Oscillation can be stopped by executing the STOP instruction.

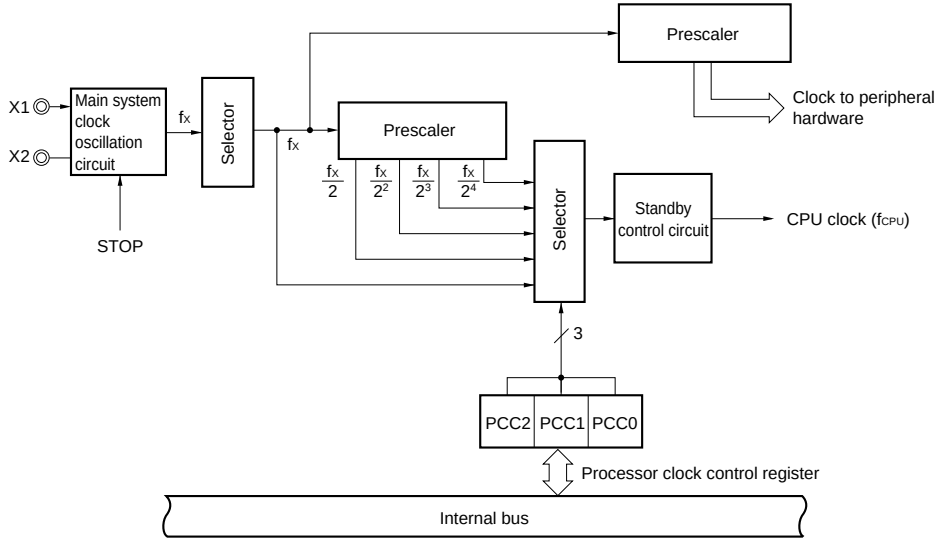
5.2 Clock Generator Configuration

The clock generator consists of the following hardware.

Table 5-1. Clock Generator Configuration

Item	Configuration
Control register	Processor clock control register (PCC)
Oscillator	Main system clock oscillator

Figure 5-1. Clock Generation Circuit Block Diagram



5.3 Register Controlling Clock Generation Circuit

The clock generation circuit is controlled by the processor clock control register (PCC).

- **Processor clock control register (PCC)**
  This register selects a CPU clock and selects a division ratio.
  PCC is set by using a 1-bit or 8-bit memory manipulation instruction.
  Its value is set to 04H at $\overline{\text{RESET}}$.

Figure 5-2. Processor Clock Control Register Format

Symbol	7	6	5	4	3	2	1	0	Address	At Reset	R/W
PCC	0	0	0	0	0	PCC2	PCC1	PCC0	FFFBH	04H	R/W

PCC2	PCC1	PCC0	Selects CPU clock (f_{CPU})
0	0	0	f_X
0	0	1	$f_X/2$
0	1	0	$f_X/2^2$
0	1	1	$f_X/2^3$
1	0	0	$f_X/2^4$
Others			Setting prohibited

Caution Be sure to clear bits 3 through 7 to “0”.

Remark f_X: Oscillation frequency of main system clock

★ The shortest instruction of the μPD780228 subseries is executed in two CPU clocks. Therefore, the relation between the CPU clock (f_{CPU}) and minimum instruction execution time is as shown in Table 5-2.

Table 5-2. Relation between CPU Clock and Minimum Instruction Execution Time

CPU Clock (f_{CPU})	Minimum Instruction Execution Time: $2/f_{CPU}$
f_X	0.4 μs
$f_X/2$	0.8 μs
$f_X/2^2$	1.6 μs
$f_X/2^3$	3.2 μs
$f_X/2^4$	6.4 μs

f_X = 5.0 MHz

f_X: Main system clock oscillation frequency

5.4 System Clock Oscillator

5.4.1 Main system clock oscillator

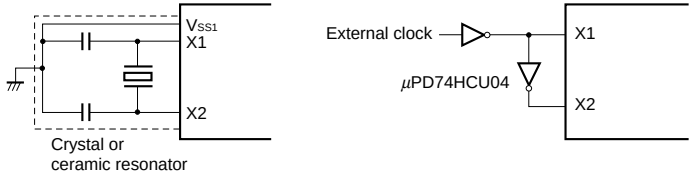
The main system clock oscillator oscillates with a crystal resonator or a ceramic resonator (5.0 MHz TYP.) connected to the X1 and X2 pins.

External clocks can be input to the main system clock oscillator. In this case, input a clock signal to the X1 pin and an antiphase clock signal to the X2 pin.

Figure 5-3 shows an external circuit of the main system clock oscillator.

Figure 5-3. External Circuit of Main System Clock Oscillator

(a) Crystal or ceramic oscillation **(b) External clock**



★ **Cautions 1. The STOP mode cannot be set when the external clock is input. This is because the X2 pin is pulled up by V_{DD1}.**

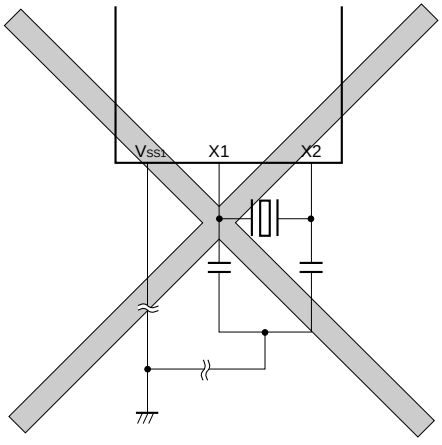
2. When using a main system clock oscillator, carry out wiring in the broken line area in Figure 5-3 as follows to prevent any effects from wiring capacities.

- **Minimize the wiring length.**
- **Do not allow wiring to intersect with other signal conductors. Do not allow wiring to come near abruptly changing high current.**
- **Set the potential of the grounding position of the oscillator capacitor to that of V_{SS1}. Do not ground to any ground pattern where high current is present.**
- **Do not fetch signals from the oscillator.**

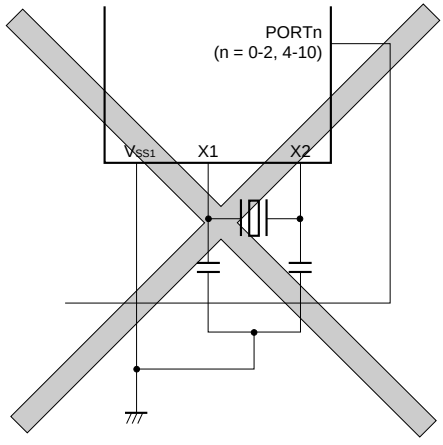
Figure 5-4 shows examples of resonator having bad connection.

Figure 5-4. Examples of Resonator with Bad Connection (1/2)

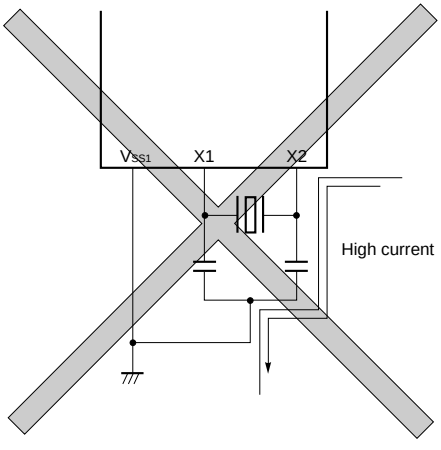
(a) Too long wiring of connected circuit



(b) Crossed signal lines



(c) High alternating current close to signal lines



(d) Current flowing through ground line of oscillator circuit (potentials at points A, B, and C change)

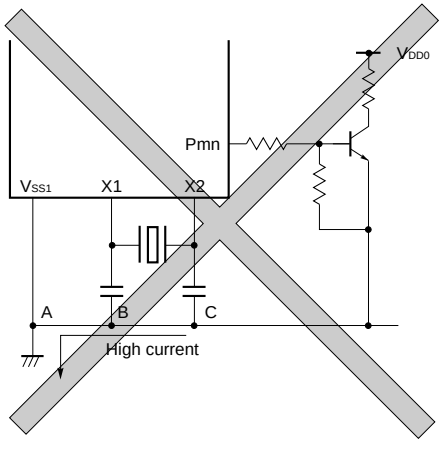
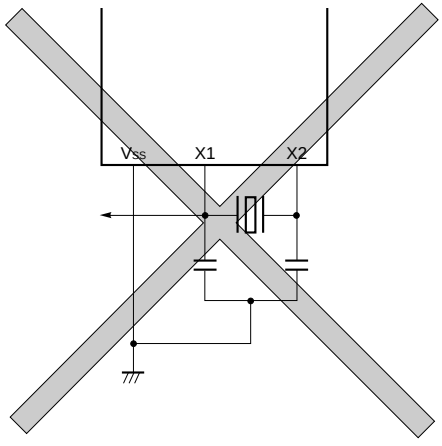


Figure 5-4. Examples of Resonator with Bad Connection (2/2)

(e) Signal extracted



5.4.2 Divider

The divider divides the main system clock oscillator output (f_X) and generates various clocks.

5.5 Clock Generator Operations

The clock generator generates the following various types of clocks and controls the CPU operating mode including the standby mode.

- Main system clock f_X
- CPU clock f_{CPU}
- Clock to peripheral hardware

The function and operation of the clock generator circuit are determined by the processor clock control register (PCC) as follows:

(a) Upon generation of $\overline{\text{RESET}}$ signal, the lowest speed mode of the main system clock (6.4 μs when operated at 5.0 MHz) is selected (PCC = 04H). Main system clock oscillation stops while low level is applied to $\overline{\text{RESET}}$ pin.

(b) One of the five (0.4 μs, 0.8 μs, 1.6 μs, 3.2 μs, and 6.4 μs: when operated at 5.0 MHz) CPU clock stages can be selected by setting the PCC.

(c) Two standby modes, the STOP and HALT modes, are available.

(d) The main system clock is divided and supplied to the peripheral hardware. Therefore, the peripheral hardware also stops if the main system clock is stopped. (Except external input clock operation)

5.6 Changing CPU Clock

5.6.1 Time required to change CPU clock

The CPU clock can be changed by using bits 0 through 2 (PCC0 through PCC2) of the processor clock control register (PCC).

Actually, the clock is not changed immediately after PCC has been rewritten, and the CPU operates with the old clock until the specified number of instructions (refer to **Table 5-3**) has been executed after PCC was changed.

Table 5-3. Maximum Time Required for Changing CPU Clock

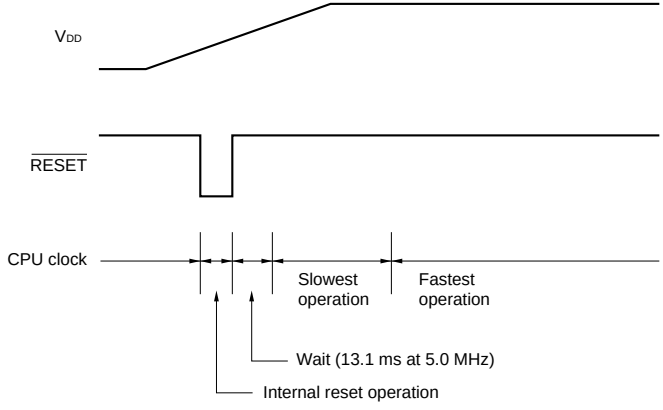
Set Value before Change			Set Value after Change														
PCC2	PCC1	PCC0	PCC2	PCC1	PCC0	PCC2	PCC1	PCC0	PCC2	PCC1	PCC0	PCC2	PCC1	PCC0	PCC2	PCC1	PCC0
			0	0	0	0	0	1	0	1	0	0	1	1	1	0	0
0	0	0				16 instructions			16 instructions			16 instructions			16 instructions		
0	0	1	8 instructions						8 instructions			8 instructions			8 instructions		
0	1	0	4 instructions			4 instructions						4 instructions			4 instructions		
0	1	1	2 instructions			2 instructions			2 instructions						2 instructions		
1	0	0	1 instruction			1 instruction			1 instruction			1 instruction					

Remark The time required to execute one instruction is equal to the minimum instruction execution time with the CPU clock before change.

5.6.2 CPU clock changing procedure

The CPU clock is changed in the following procedure.

Figure 5-5. Changing CPU Clock



(1) The CPU is reset if the $\overline{\text{RESET}}$ pin is made low after power application. The reset is cleared and the main system clock starts oscillating if the $\overline{\text{RESET}}$ pin is later made high. At this time, it is automatically ensured ★ that oscillation stabilization time ($2^{16}/f_X$) elapses.
After that, the CPU starts executing instructions at the slowest speed of the main system clock (6.4 μs at 5.0 MHz).

(2) After sufficient time has elapsed during which the V_{DD} voltage rises to the level at which the CPU can operate at the highest speed, the contents of the processor clock control register (PCC) are rewritten, and the CPU operates at the highest speed.

CHAPTER 6 8-BIT REMOTE CONTROL TIMER

6.1 Function of 8-Bit Remote Control Timer

The 8-bit remote control timer has a pulse width measurement function with a resolution of 8 bits.

Pulse width is measured from a difference in count value when the valid edge has been detected while the timer operates in the free-running mode.

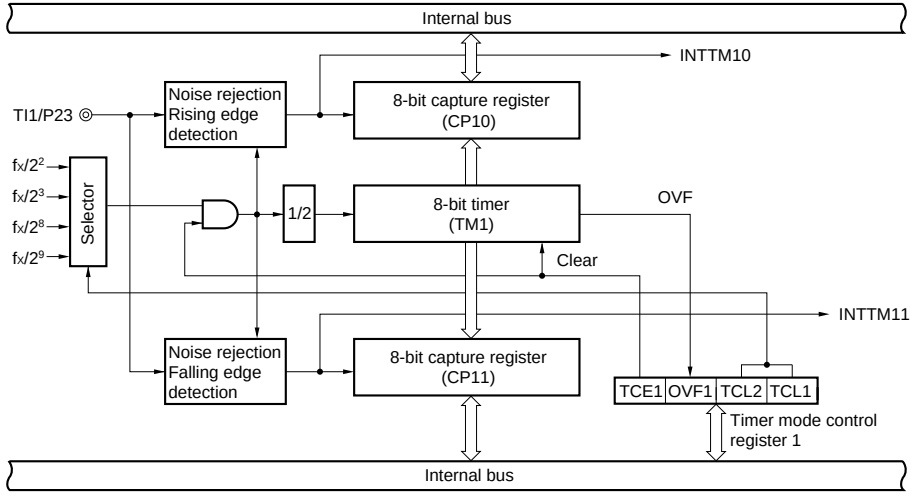
6.2 Configuration of 8-Bit Remote Control Timer

The 8-bit remote control timer consists of the following hardware:

Table 6-1. 8-Bit Remote Control Timer Configuration

Item	Configuration
Timer register	8-bit timer (TM1)
Register	8-bit capture register: × 2 (CP10 and CP11)
Control register	Timer mode control register 1 (TMC1)

Figure 6-1. Block Diagram of 8-Bit Remote Control Timer



6.3 Registers Controlling 8-Bit Remote Control Timer

The following three types of registers control the 8-bit remote control timer.

- Timer mode control register 1 (TMC1)
- 8-bit capture registers (CP10 and CP11)
- 8-bit timer register (TM1)

(1) Timer mode control register 1 (TMC1)

This register enables or disables the operation of the 8-bit timer (TM1), sets the count clock, and detects overflow.

TMC1 is set by using a 1-bit or 8-bit memory manipulation instruction.

This register is initialized to 00H by $\overline{\text{RESET}}$ input.

Figure 6-2. Timer Mode Control Register 1 Format

Symbol	< 7 >	6	5	4	3	< 2 >	1	0	Address	At Reset	R/W
TMC1	0	0	0	0	0	OVF1	TCL2	TCL1	FF60H	00H	R/W

TCE1	Controls count operation of TM1
0	Clears count to 0 and stops operation
1	Starts count operation

OVF1	Detects overflow of TM1
0	No overflow
1	Overflow

★

TCL2	TCL1	Selects count clock
0	0	$f_X/2^{10}$ (4.9 kHz)
0	1	$f_X/2^9$ (9.8 kHz)
1	0	$f_X/2^4$ (313 kHz)
1	1	$f_X/2^3$ (625 kHz)

Caution Be sure to clear bits 3 through 6 to “0”.

Remarks 1. f_X: Main system clock oscillation frequency

2. ( ): f_X = 5.0 MHz

(2) 8-bit capture registers (CP10 and CP11)

These 8-bit registers capture the contents of the 8-bit timer (TM1).

The capture operation is performed in synchronization with the valid edge input to the TI1 pin (capture trigger).

The contents of CP10 are retained until the next rising edge of the TI1 pin is detected. The contents of CP11 are retained until the next falling edge of the TI1 pin is detected.

CP10 and CP11 can be read by using an 8-bit memory manipulation instruction.

The values of these registers are initialized to 00H by $\overline{\text{RESET}}$ input.

(3) 8-bit timer register (TM1)

This 8-bit register counts the count pulse.

It can be read by using an 8-bit memory manipulation instruction.

The value of this register is initialized to 00H by $\overline{\text{RESET}}$ input or by clearing the TCE1 bit.

6.4 Operation of 8-Bit Remote Control Timer

The 8-bit remote control timer operates as a pulse width measuring circuit.

The width of a high-level or low-level external pulse input to the TI1 pin is measured by operating the 8-bit timer (TM1) in the free-running mode.

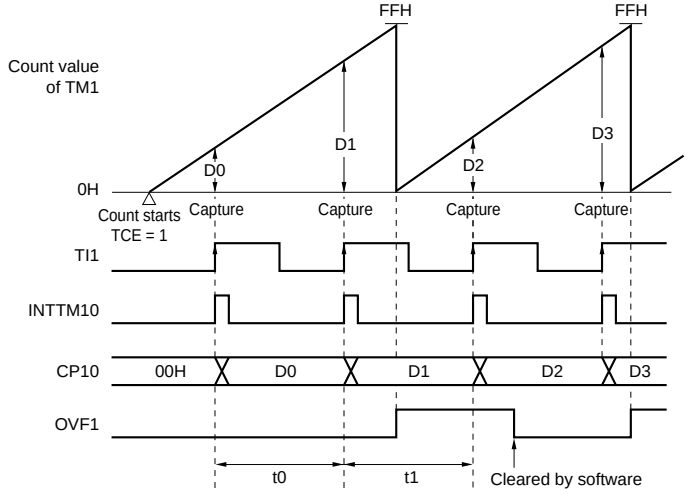
Detection of the valid edge is sampled every 2 cycles of the count clock selected by TCL1 and TCL2, and the capture operation is not performed until the valid level has been detected two times. Therefore, the pulse width input to the TI1 pin must be 5 or more of the count clock set by TCL1 and TCL2, regardless of whether the level is high or low. If the pulse width is less than 5 clocks, it cannot be detected, and the capture operation is not performed.

The value of timer register 1 (TM1) is loaded to and retained in the capture registers (CP10 and CP11) in synchronization with the valid edge of the pulse input to the TI1 pin, as shown in Figure 6-3.

Figure 6-3 shows the timing of pulse width measurement.

Figure 6-3. Timing of Pulse Width Measurement (1/2)

(1) To measure pulse width in synchronization with rising edge



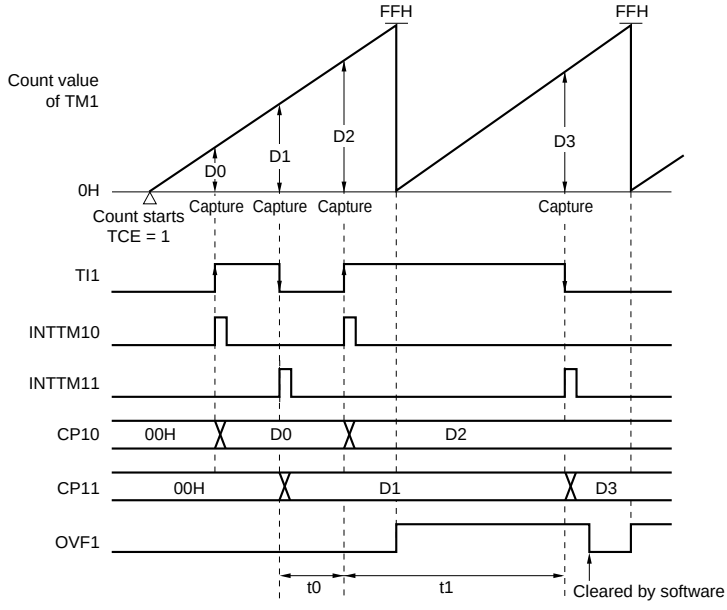
Remark t0 = (D1 – D0) × 1/f_{COUNT}

t1 = (100H – D1 + D2) × 1/f_{COUNT}

f_{COUNT}: Count clock frequency set by TCL1 and TCL2

Figure 6-3. Timing of Pulse Width Measurement (2/2)

(2) Measure pulse width in synchronization with both rising and falling edges



Remark t0 = (D1 – D0) × 1/f_{COUNT}

t1 = (100H – D2 + D3) × 1/f_{COUNT}

f_{COUNT}: Count clock frequency set by TCL1 and TCL2

[MEMO]

CHAPTER 7 8-BIT PWM TIMERS

7.1 Functions of 8-Bit PWM Timers

The 8-bit PWM timers have the following two operation modes:

- Mode in which only an 8-bit timer (TM5n: n = 0 or 1) is used (single mode)
- Mode in which the two 8-bit PWM timers are cascaded (16-bit resolution: cascade mode)

These two modes are explained next.

(1) Mode in which only a TM5n (n = 0 or 1) is used (single mode)

In this mode, the 8-bit PWM timer operates as an 8-bit timer/event counter. In this mode, the following functions can be used.

- Interval timer
- External event counter
- Square wave output
- PWM output

(2) Mode in which two timers are cascaded (16-bit resolution: cascade mode)

When the two PWM timers are cascaded, they operate as a 16-bit timer/event counter. In this mode, the following functions can be used.

- Interval timer with 16-bit resolution
- External event counter with 16-bit resolution
- Square output with 16-bit resolution

7.2 Configuration of 8-Bit PWM Timers

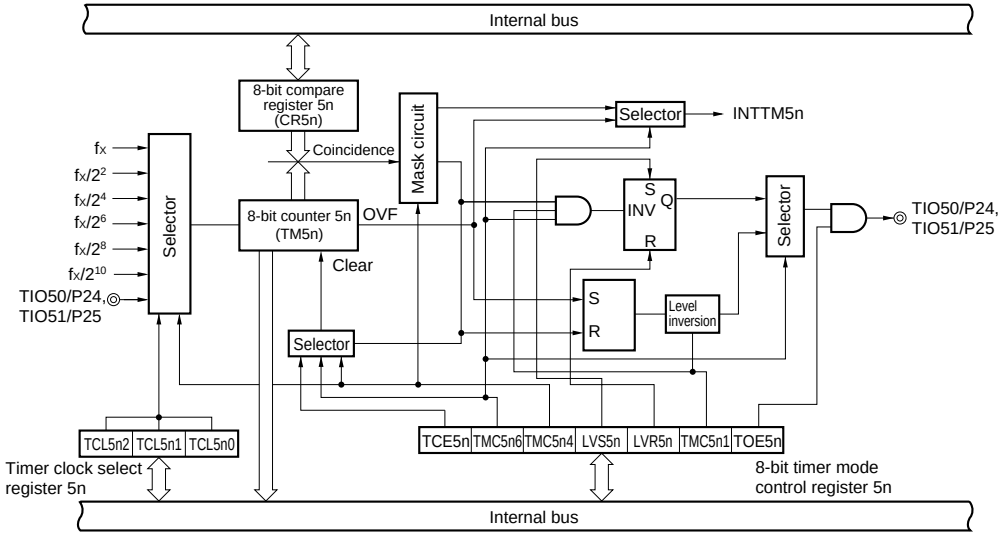
The 8-bit PWM timers consist of the following hardware:

Table 7-1. Configuration of 8-Bit PWM Timers

Item	Configuration
Timer register	8-bit counter 5n (TM5n)
Register	8-bit compare register 5n (CR5n)
Timer output	TIO5n
Control registers	Timer clock select register 5n (TCL5n) 8-bit timer mode control register 5n (TMC5n)

n = 0, 1

Figure 7-1. 8-Bit PWM Timers Block Diagram



n = 0 or 1

(1) 8-bit counter 5n (TM5n: n = 0 or 1)

TM5n is an 8-bit read-only register that counts the count pulse.

The value of this counter is incremented in synchronization with the rising edge of the count clock. When the count value is read during operation, input of the count clock is temporarily stopped, and the count value at that point is read. The count value is cleared to 00H in the following cases.

<1> $\overline{\text{RESET}}$ input
<2> Clearing TCE5n
<3> Coincidence between TM5n and CR5n in clear & start mode

Caution In the cascade mode, TCE5n of the low-order timer is 00H even if it is cleared.

Remark n = 0 or 1

(2) 8-bit compare register 5n (CR5n: n = 0 or 1)

The value set in this register is constantly compared with the count value of the 8-bit counter 5n (TM5n). When the two values coincide, an interrupt request (INTTM5n) is generated (in the modes other than PWM mode). The value of CR5n can be set in a range of 00H to FFH, and can be rewritten during count operation.

Caution When setting data to this register in the cascade mode, be sure to stop the timer operation.

Remark n = 0 or 1

7.3 Registers Controlling 8-Bit PWM Timers

The following two types of registers control the 8-bit PWM timers.

- Timer clock select register 5n (TCL5n: n = 0 or 1)
- 8-bit timer mode control register 5n (TMC5n: n = 0 or 1)

(1) Timer clock select register 5n (TCL5n: n = 0 or 1)

This register sets the count clock of the 8-bit counter 5n (TM5n: n = 0 or 1).

TCL5n is set by using an 8-bit memory manipulation instruction.

The value of this register is initialized to 00H by $\overline{\text{RESET}}$ input.

Figure 7-2. Format of Timer Clock Select Register

Symbol	7	6	5	4	3	2	1	0	Address	At Reset	R/W
TCL5n	0	0	0	0	0	TCL5n2	TCL5n1	TCL5n0	FF71H(TCL50), FF79H(TCL51)	00H	R/W

TCL5n2	TCL5n1	TCL5n0	Selects count clock
0	0	0	Falling edge of TIO5n
0	0	1	Rising edge of TIO5n
0	1	0	f_X (5.0 MHz)
0	1	1	$f_X/2^2$ (1.25 MHz)
1	0	0	$f_X/2^4$ (313 kHz)
1	0	1	$f_X/2^6$ (78.1 kHz)
1	1	0	$f_X/2^8$ (19.5 kHz)
1	1	1	$f_X/2^{10}$ (4.9 kHz)

Cautions 1. When rewriting the data of TCL5n, once stop the timer operation.

2. Be sure to clear bits 3 through 7 to "0".

Remarks 1. The setting by TCL5n2 through TCL5n0, except that for the low-order timer, is invalid in the cascade mode.

2. n = 0 or 1

3. f_X: Main system clock oscillation frequency

4. ( ): f_X = 5.0 MHz

(2) 8-bit timer mode control register 5n (TMC5n: n = 0 or 1)

TMC5n sets has the following six functions:

<1> Controls count operation of 8-bit counter 5n (TM5n: n = 0 or 1)
<2> Selects operation mode of 8-bit counter 5n (TM5n: n = 0 or 1)
<3> Selects single or cascade mode
<4> Sets status of timer output F/F (flip-flop)
<5> Controls timer F/F or selects active level in PWM (free-running) mode
<6> Controls timer output

TMC5n is set by using a 1-bit or 8-bit memory manipulation instruction.
This register is set to 04H by $\overline{\text{RESET}}$ input.

Figure 7-3. Format of 8-Bit Timer Control Register 5n

Symbol	< 7 >	6	5	4	< 3 >	< 2 >	1	0	Address	At Reset	R/W
TMC5n	TCE5n	TMC5n6	0	TMC5n4	LVS5n	LVR5n	TMC5n1	TOE5n	FF70H(TMC50), FF78H(TMC51)	04H	R/W

TCE5n	Controls counting by TM5n
0	Clears counter to 0 and disables counting (prescaler disabled)
1	Starts counting

TMC5n6	Selects operation mode of TM5n
0	Clear & start on coincidence between TM5n and CR5n
1	PWM (free-running) mode

TMC5n4	Selects single or cascade mode
0	Single mode (used as low-order timer)
1	Cascade mode (connected to low-order timer)

LVS5n	LVR5n	Sets status of timer output F/F
0	0	Does not affect
0	1	Resets timer output F/F to 0
1	0	Sets timer output F/F to 1
1	1	Setting prohibited

TMC5n1	Other than PWM mode (TMC5n6 = 0)	PWM mode (TMC5n6 = 1)
	Controls timer F/F	Selects active level
0	Disables inversion	High active
1	Enables inversion	Low active

TOE5n	Controls timer output
0	Disables output (port mode)
1	Enables output

Remarks 1. In the PWM mode, the PWM output is at the inactive level if TCE5n = 0.

2. When LVS5n and LVR5n are read after data has been set, they are 0.

3. n = 0 or 1

7.4 Operations of 8-Bit PWM Timers

7.4.1 Operation as interval timer (8-bit operation)

An 8-bit PWM timer operates as an interval timer that generates an interrupt request at intervals specified by the count value set to the 8-bit compare register 5n (CR5n) in advance.

When the count value of the 8-bit counter 5n (TM5n) coincides with the set value of CR5n, the value of TM5n is cleared to 0 and TM5n continues counting. At the same time, an interrupt request signal (INTTM5n) is generated.

The count clock of TM5n can be selected by using the bits 0 through 2 (TCL5n0 through TCL5n2) of the timer clock select register 5n (TCL5n).

Remark n = 0 or 1

[Setting]

(1) Set the registers.
- TCL5n : Selects count clock.
- CR5n : Compare value
- TMC5n : Selects clear & start mode in which TM5n is cleared and started when its value coincides with CR5n.
  (TMC5n = 0000xxx0B × = don't care)

(2) The count operation is started when TCE5n = 1.

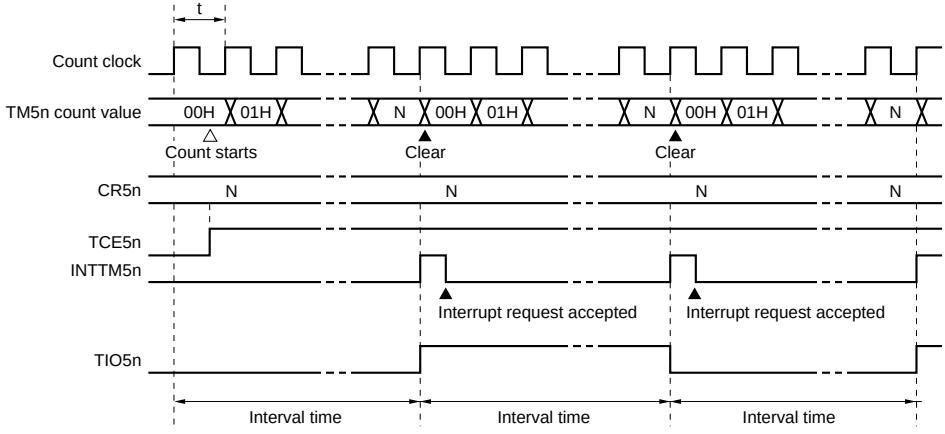
(3) When the values of TM5n and CR5n coincide, INTTM5n is generated (TM5n is cleared to 00H).

(4) After that, INTTM5n is generated at fixed intervals. To stop the count operation, clear TCE5n = 0.

Remark n = 0 or 1

Figure 7-4. Timing of Interval Timer Operation (1/3)

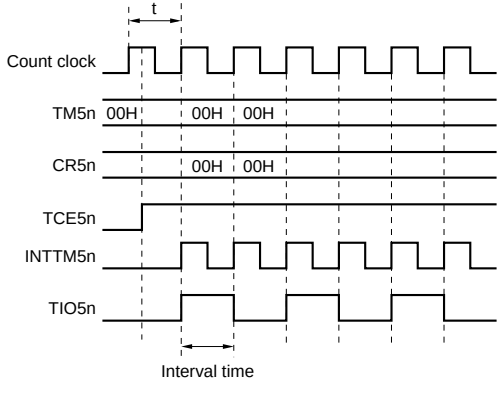
(a) Basic operation



Remarks 1. Interval time = (n + 1) × t: N = 00H to FFH

2. n = 0 or 1

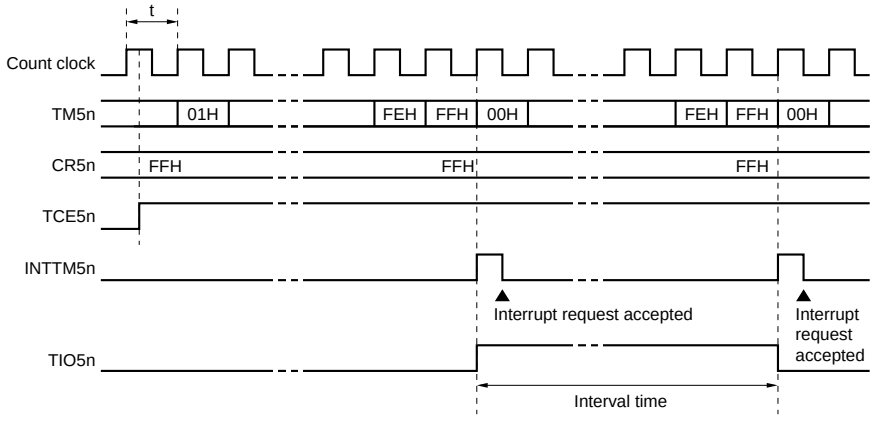
(b) When CR5n = 00H



n = 0 or 1

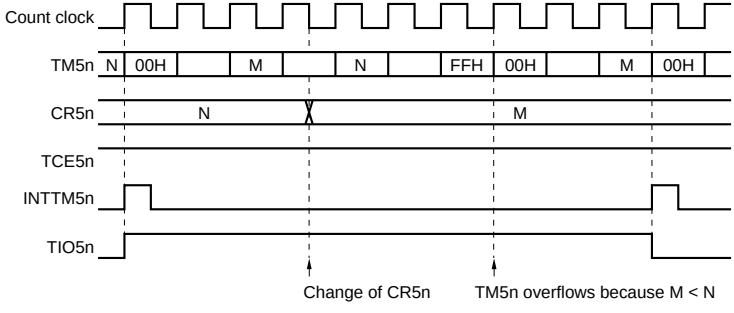
Figure 7-4. Timing of Interval Timer Operation (2/3)

(c) When CR5n = FFH



n = 0 or 1

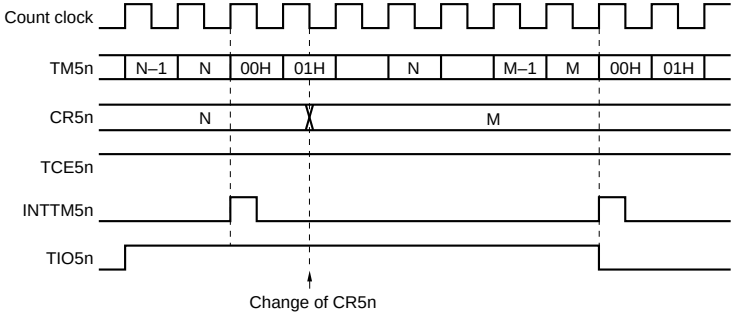
(d) Operation when CR5n is changed (M < N)



n = 0 or 1

Figure 7-4. Timing of Interval Timer Operation (3/3)

(e) Operation when CR5n is changed (M > N)



n = 0 or 1

7.4.2 Operation as external event counter

The external event counter counts the number of count clock pulses input to TIO5n from an external source.

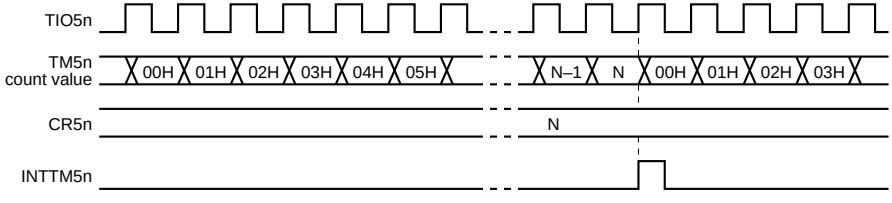
Each time the valid edge specified by the timer clock select register 5n (TCL5n) has been input to TIO5n, the value of TM5n is incremented. The edge can be selected from rising or falling.

If the measured value of TM5n coincides with the value of the 8-bit compare register 5n (CR5n), TM5n is cleared to 0 and an interrupt request signal (INTTM5n) is generated.

After that, INTTM5n is generated each time the value of TM5n coincides with the value of CR5n.

Remark n = 0 or 1

Figure 7-5. Timing of External Event Counter Operation (with rising edge specified)



n = 0 or 1

7.4.3 Square wave (8-bit resolution) output operation

A square wave of any frequency can be output at intervals specified by the value set in advance to the 8-bit compare register 5n (CR5n).

If the bit 0 (TOE5n) of the 8-bit timer mode control register 5n (TMC5n) is set to 1, the output status of TIO5n is inverted at intervals specified by the count value set in advance to CR5n. In this way, a square wave of any frequency (duty = 50%) can be output.

Remark n = 0 or 1

[Setting]

(1) Set each register.

- Set “0” to port latch and port mode register.
- TCL5n : Selects count clock.
- CR5n : Compare value
- TMC5n : Clear & start mode in which TM5n is cleared and started when its value coincides with that of CR5n.

LVS5n	LVR5n	Status setting of timer output F/F
1	0	High-level output
0	1	Low-level output

Inversion of the timer output F/F is enabled.

Timer output enable → TOE5n = 1

(2) The count operation is started when TCE5n = 1.

(3) The timer output F/F is inverted when the values of TM5n and CR5n coincide. Moreover, INTTM5n is generated, and TM5n is cleared to 00H.

(4) After that, the timer output F/F is inverted at fixed intervals, and TIO5n outputs a square wave.

Remark n = 0 or 1

7.4.4 8-bit PWM output operation

The PWM timer performs 8-bit PWM output operation when bit 6 of the 8-bit timer mode control register 5n (TMC5n) is set to “1”, and outputs a pulse with a duty factor determined by the value set to the 8-bit compare register 5n (CR5n) from the TIO5n pin.

Set the width of the active level of the PWM pulse to CR5n. The active level can be selected by bit 1 (TMC5n1) of TMC5n.

The count clock can be selected by bits 0 through 2 (TCL5n0 through TCL5n2) of the timer clock select register 5n (TCL5n).

PWM output can be enabled or disabled by bit 0 (TOE5n) of TMC5n.

Caution CR5n can be rewritten only once in one cycle in the PWM mode.

Remark n = 0 or 1

(1) Basic PWM output operation

[Setting]

(1) Set “0” to the port latch and port mode register n.
(2) Set the active level width by using 8-bit compare register 5n (CR5n).
(3) Select the count clock by using timer clock select register 5n (TCL5n).
(4) Select the active level by using bit 1 (TMC5n1) of TMC5n.
(5) The timer starts counting when bit 7 (TCE5n) of TMC5n is set to “1”.
To stop the counting, set “0” to TCE5n.

Remark n = 0 or 1

[PWM output operation]

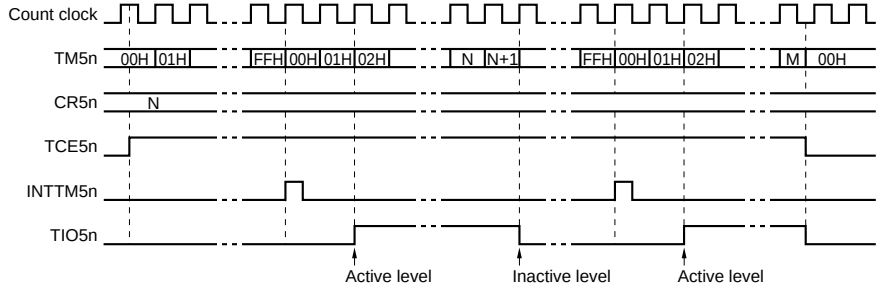
(1) When the timer starts counting, an inactive level is output from TIO5n as PWM output, until the timer overflows.
(2) When the overflow occurs, the active level set in (1) in [Setting] above is output. The active level is continuously output until the CR5n and the count value of the 8-bit counter 5n (TM5n) coincide.
(3) The inactive level is output after CR5n and the count value have coincided, until an overflow occurs again.
(4) After that, (2) and (3) are repeated, until the counting operation is stopped.
(5) PWM output is deasserted inactive when the counting operation is stopped by clearing TCE5n to 0.

Remark n = 0 or 1

(a) Basic PWM output operation

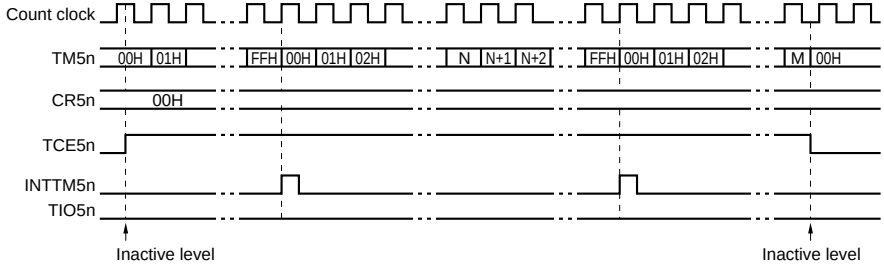
Figure 7-6. PWM Output Operation Timing

(i) Basic operation (when active level = H)



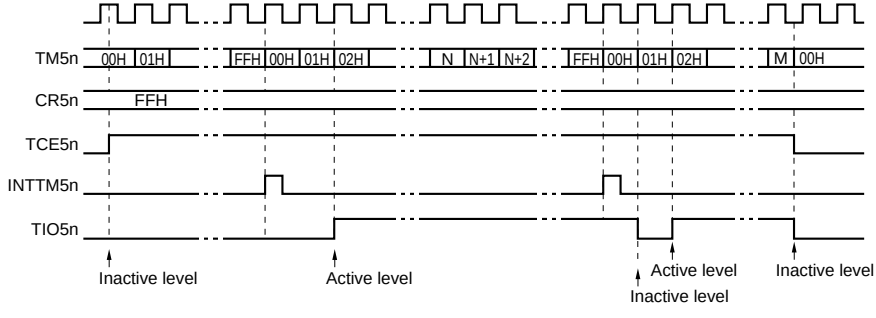
n = 0 or 1

(ii) When CR5n = 0



n = 0 or 1

(iii) When CR5n = FFH

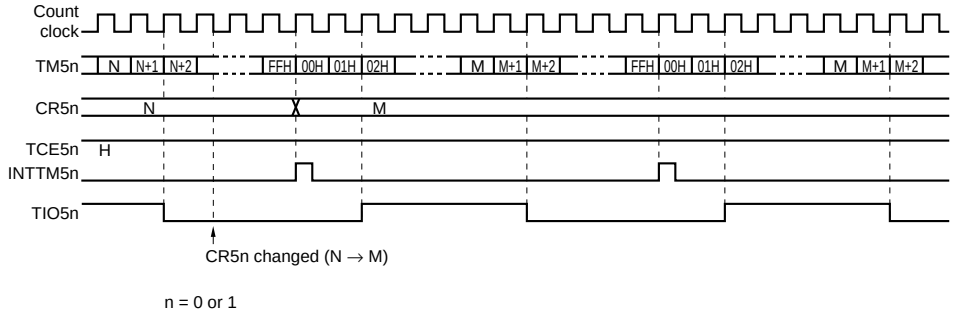


n = 0 or 1

(b) Operation when CR5n is changed

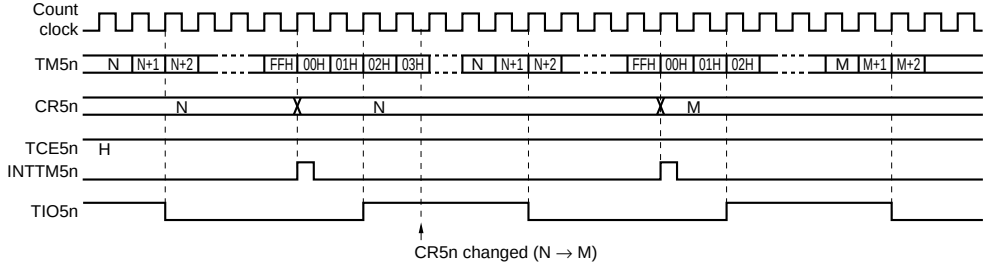
Figure 7-7. Operation Timing When CR5n Is Changed

(i) If CR5n value is changed from N to M before overflow of TM5n



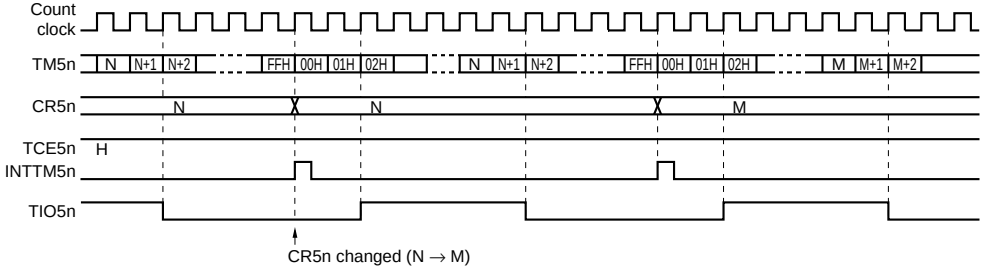
n = 0 or 1

(ii) If CR5n value is changed from N to M after overflow of TM5n



n = 0 or 1

(iii) If CR5n value is changed from N to M for duration of 2 clocks (00H and 01H) immediately after overflow of TM5n



n = 0 or 1

(2) Cascade (16-bit timer) mode

- Operation as interval timer (with 16-bit resolution)
  The two PWM timers can be used as a 16-bit timer/counter by setting bit 4 (TMC5n4) of the 8-bit timer mode control register 5n (TMC5n) to “1”.
  In this case, the 16-bit timer/counter operates as an interval timer that repeatedly generates an interrupt request at intervals specified by the count value set in advance to the 8-bit compare register 5n (CR5n).

  Remark n = 0 or 1

  [Setting]

  (1) Set each register.
  - TCL5n : The low-order timer selects the count clock.
    The setting of the high-order cascaded timer is not necessary.
  - CR5n : Compare value (Each compare value can be set in a range of 00H to FFH.)
  - TMC5n : Selects the clear & start mode in which the timers are cleared and started on coincidence between TM5n and CR5n.

    (Low-order timer → TMC5n = 0000×××0B ×: don’t care
    High-order timer → TMC5n = 0001×××0B ×: don’t care)

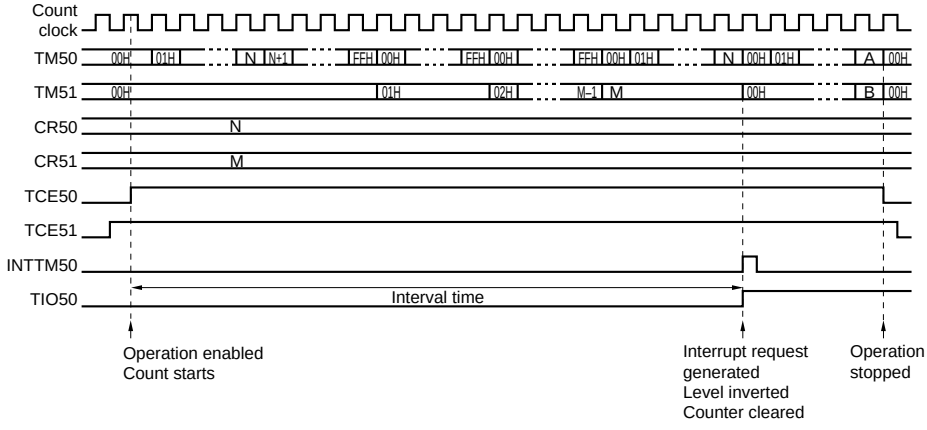
  (2) The counting is started when TCE5n of the high-order timer is set to 1 followed by setting of TCE5n of the low-order timer to 1.
  (3) When the values of TM5n and CR5n of the cascaded timers cascade coincide, INTTM5n is generated by the low-order timer (all the TM5n’s are cleared to 00H).
  (4) After that, INTTM5n is repeatedly generated at the same interval.

  Cautions 1. Before setting the 8-bit compare register 5n (CR5n), be sure to stop the timer operation.
  2. Even when the timers are cascaded, if the count value of the high-order timer coincides with the value of CR5n, INTT5n of the high-order timer is generated, unless masked. Be sure to mask and disable the interrupt of the high-order timer.
  3. Set TCE5n of the high-order timer first, and then that of the low-order timer.
  4. The counting can be restarted or stopped by setting 1 or 0 to TCE5n of only the low-order timer.

  Remark n = 0 or 1

Figure 7-8 shows an example of the timing in the 16-bit resolution cascade mode.

Figure 7-8. 16-Bit Resolution Cascade Mode

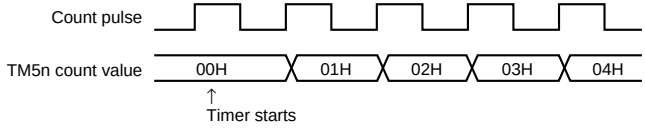


7.5 Notes on 8-Bit PWM Timers

(1) Error on starting timer

The time until the coincidence signal is generated after the timer has been started includes an error of up to 1 clock, because the 8-bit counter n (TM5n: n = 0 or 1) is started in asynchronization with the count pulse.

Figure 7-9. Start Timing of 8-Bit Counter 5n (TM5n)



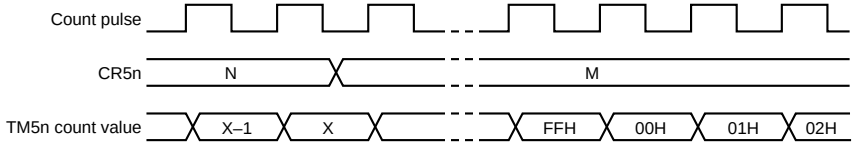
n = 0 or 1

(2) Operation after changing compare register during timer count operation

If the value to which the current value of the 8-bit compare register 5n (CR5n) is changed is less than the value of the 8-bit timer register 5n, the timer continues counting, overflows, and restarts counting from 0. If the new value of CR5n (M) is less than the old value (N), the timer must be restarted after CR5n has been changed.

Remark n = 0 or 1

Figure 7-10. Timing after Changing Compare Register Value during Timer Count Operation



N > X > M
n = 0 or 1

Caution Except when TIO5n input is selected, be sure to clear TCE5n to 0 to set the stop status (n = 0 or 1).

(3) Reading TM5n during timer operation

Because the selected clock is temporarily stopped when TM5n (n = 0 or 1) is read during operation, select a clock with a long high/low level.

[MEMO]

CHAPTER 8 WATCHDOG TIMER

8.1 Function of Watchdog Timer

The watchdog timer has the following functions:

- Watchdog timer
- Interval timer
- Selection of oscillation stabilization time

Caution Select whether the watchdog timer is used in the watchdog timer mode or interval timer mode, by using the watchdog timer mode register (WDTM).

(1) Watchdog timer mode

In this mode, the watchdog timer detects a program hang-up. On detection of hang-up, the non-maskable interrupt or $\overline{\text{RESET}}$ signal can be generated.

Table 8-1. Hang-up Detection Time of Watchdog Timer

Hang-up Detection Time	f_X = 5.0 MHz	Hang-up Detection Time	f_X = 5.0 MHz
$2^{12} \times 1/f_X$	819 μs	$2^{16} \times 1/f_X$	13.1 ms
$2^{13} \times 1/f_X$	1.64 ms	$2^{17} \times 1/f_X$	26.2 ms
$2^{14} \times 1/f_X$	3.28 ms	$2^{18} \times 1/f_X$	52.4 ms
$2^{15} \times 1/f_X$	6.55 ms	$2^{20} \times 1/f_X$	210 ms

f_X: Main system clock oscillation frequency

(2) Interval timer mode

In this mode, the watchdog timer generates an interrupt request at fixed time intervals.

Table 8-2. Interval Time

Interval Time	f_X = 5.0 MHz	Interval Time	f_X = 5.0 MHz
$2^{12} \times 1/f_X$	819 μs	$2^{16} \times 1/f_X$	13.1 ms
$2^{13} \times 1/f_X$	1.64 ms	$2^{17} \times 1/f_X$	26.2 ms
$2^{14} \times 1/f_X$	3.28 ms	$2^{18} \times 1/f_X$	52.4 ms
$2^{15} \times 1/f_X$	6.55 ms	$2^{20} \times 1/f_X$	210 ms

f_X: Main system clock oscillation frequency

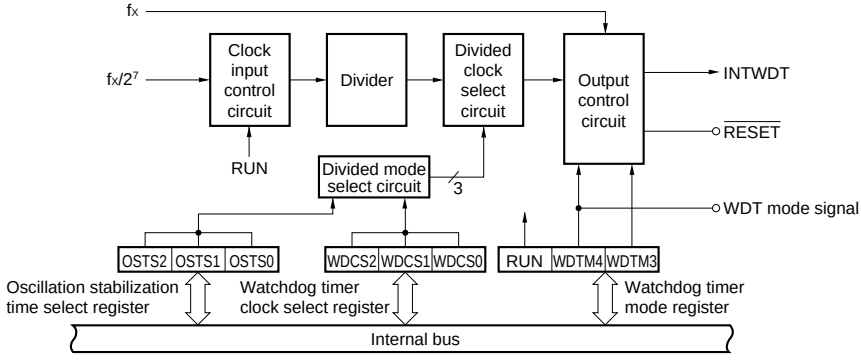
8.2 Configuration of Watchdog Timer

The watchdog timer consists of the following hardware.

Table 8-3. Configuration of Watchdog Timer

Item	Configuration
Control register	Oscillation stabilization time select register (OSTS) Watchdog timer clock select register (WDCS) Watchdog timer mode register (WDTM)

Figure 8-1. Watchdog Timer Block Diagram



8.3 Registers Controlling Watchdog Timer

The following three types of registers control the watchdog timer.

- Oscillation stabilization time select register (OSTS)
- Watchdog timer clock select register (WDCS)
- Watchdog timer mode register (WDTM)

(1) Oscillation stabilization time select register (OSTS)

This register selects the oscillation stabilization time during which oscillation is stabilized after the $\overline{\text{RESET}}$ signal has been deasserted or the STOP mode has been released.

This register is set by using a 1-bit or 8-bit memory manipulation instruction.

Its value is set to 04H by $\overline{\text{RESET}}$ input.

Figure 8-2. Oscillation Stabilization Time Select Register Format

Symbol	7	6	5	4	3	2	1	0	Address	At Reset	R/W
OSTS	0	0	0	0	0	OSTS2	OSTS1	OSTS0	FFFAH	04H	R/W

★

OSTS2	OSTS1	OSTS0	Selects oscillation stabilization time at STOP mode release
0	0	0	$2^{11}/f_X$ (410 μs)
0	0	1	$2^{13}/f_X$ (1.64 ms)
0	1	0	$2^{14}/f_X$ (3.28 ms)
0	1	1	$2^{15}/f_X$ (6.55 ms)
1	0	0	$2^{16}/f_X$ (13.1 ms)
Others			Setting prohibited

Remarks 1. f_X: Main system clock oscillation frequency

2. ( ): f_X = 5.0 MHz

(2) Watchdog timer clock select register (WDCS)

This register selects the overflow time of the watchdog timer or interval timer.

It is set by using an 8-bit manipulation instruction.

The value of this register is initialized to 00H by $\overline{\text{RESET}}$ input.

Figure 8-3. Watchdog Timer Clock Select Register Format

Symbol	7	6	5	4	3	2	1	0	Address	At Reset	R/W
WDCS	0	0	0	0	0	WDCS2	WDCS1	WDCS0	FF42H	00H	R/W

WDCS2	WDCS1	WDCS0	Overflow time of watchdog timer/interval timer
0	0	0	$2^{12}/f_X$ (819 μs)
0	0	1	$2^{13}/f_X$ (1.64 ms)
0	1	0	$2^{14}/f_X$ (3.28 ms)
0	1	1	$2^{15}/f_X$ (6.55 ms)
1	0	0	$2^{16}/f_X$ (13.1 ms)
1	0	1	$2^{17}/f_X$ (26.2 ms)
1	1	0	$2^{18}/f_X$ (52.4 ms)
1	1	1	$2^{20}/f_X$ (210 ms)

Remarks 1. f_X: Main system clock oscillation frequency

2. ( ): f_X = 5.0 MHz

(3) Watchdog timer mode register (WDTM)

This register selects the operation mode of the watchdog timer, and enables or disables the counting operation.

It is set by using a 1-bit or 8-bit memory manipulation instruction.

The value of this register is initialized to 00H by $\overline{\text{RESET}}$ input.

Figure 8-4. Watchdog Timer Mode Register Format

Symbol	< 7 >	6	5	4	3	2	1	0	Address	At Reset	R/W
WDTM	RUN	0	0	WDTM4	WDTM3	0	0	0	FFF9H	00H	R/W

RUN	Selects operation of watchdog timer[Note 1]
0	Stops counting
1	Clears counter and starts counting

WDTM4	WDTM3	Selects operation mode of watchdog timer[Note 2]
0	×	Interval timer mode (Maskable interrupt request occurs when overflow occurs.)
1	0	Watchdog timer mode 1 (Non-maskable interrupt request occurs when overflow occurs.)
1	1	Watchdog timer mode 2 (Reset operation is started when overflow occurs.)

Notes 1. The RUN bit cannot be cleared to 0 by software once it has been set. Therefore, counting cannot be stopped, after it has been started, by any means other than $\overline{\text{RESET}}$ input.

2. The WDTM3 and WDTM4 bits cannot be cleared to 0 by software once they have been set.

Caution When the RUN bit is set to 1 and the watchdog timer is cleared, the actual overflow time is up to 0.5% shorter than the set time.

Remark ×: don't care

8.4 Operation of Watchdog Timer

8.4.1 Operation as watchdog timer

The watchdog timer operates to detect a program hang-up when bit 4 (WDTM4) of the watchdog timer mode register (WDTM) is set to 1.

The count clock of the watchdog timer (hang-up detection time interval) can be selected by bits 0 through 2 (WDCS0 through WDCS2) of the watchdog timer clock select register (WDCS). By setting bit 7 (RUN) of WDTM, the watchdog timer starts the count operation. If RUN is set to 1 again within the specified hang-up detection time interval after the counting operation has been started, the watchdog timer is cleared, and starts the count operation again.

If RUN is not set to 1 and the hang-up detection time is exceeded, the system is reset or the non-maskable interrupt request is generated, depending on the value of bit 3 (WDTM3) of WDTM.

The watchdog timer continues its operation in the HALT mode, but stops in the STOP mode. Therefore, set RUN to 1 and clear the watchdog timer before executing the STOP instruction to set the STOP mode.

Caution The actual hang-up detection time may be up to 0.5% shorter than the set time.

Table 8-4. Hang-up Detection Time of Watchdog Timer

WDCS2	WDCS1	WDCS0	Hang-up Detection Time of Watchdog Timer
0	0	0	$2^{12}/f_X$ (819 μs)
0	0	1	$2^{13}/f_X$ (1.64 ms)
0	1	0	$2^{14}/f_X$ (3.28 ms)
0	1	1	$2^{15}/f_X$ (6.55 ms)
1	0	0	$2^{16}/f_X$ (13.1 ms)
1	0	1	$2^{17}/f_X$ (26.2 ms)
1	1	0	$2^{18}/f_X$ (52.4 ms)
1	1	1	$2^{20}/f_X$ (210 ms)

Remarks 1. f_X: Main system clock oscillation frequency

2. ( ): f_X = 5.0 MHz

8.4.2 Operation as interval timer

The watchdog timer operates as an interval timer that repeatedly generates an interrupt request at intervals specified by the count value set in advance if bit 4 (WDTM4) of the watchdog timer mode register (WDTM) is cleared to 0.

The count clock (interval time) can be selected by bits 0 through 2 (WDCS0 through WDCS2) of the watchdog timer clock select register (WDCS). By setting bit 7 (RUN) of WDTM to 1, the watchdog timer starts interval timer operation.

While the watchdog timer is operating as an interval timer, the interrupt mask flag (TMMK4) and priority specification flag (TMPR4) are valid, and a maskable interrupt (INTWDT) can be generated. The default priority of INTWDT is the highest of all the maskable interrupts.

The interval timer continues operating in the HALT mode, but stops in the STOP mode. Therefore, set RUN and clear the interval timer before executing the STOP instruction to set the STOP mode.

Cautions 1. If bit 4 (WDTM4) of WDTM has been set to 1 once (to select the watchdog timer mode), the interval timer mode cannot be set unless the $\overline{\text{RESET}}$ signal is input.

2. The interval time immediately after WDTM has been set may be up to 0.5% shorter than the set time.

Table 8-5. Interval Time of Interval Timer

WDCS2	WDCS1	WDCS0	Interval Time
0	0	0	$2^{12}/f_X$ (819 μs)
0	0	1	$2^{13}/f_X$ (1.64 ms)
0	1	0	$2^{14}/f_X$ (3.28 ms)
0	1	1	$2^{15}/f_X$ (6.55 ms)
1	0	0	$2^{16}/f_X$ (13.1 ms)
1	0	1	$2^{17}/f_X$ (26.2 ms)
1	1	0	$2^{18}/f_X$ (52.4 ms)
1	1	1	$2^{20}/f_X$ (210 ms)

Remarks 1. f_X: Main system clock oscillation frequency

2. ( ): f_X = 5.0 MHz

[MEMO]

CHAPTER 9 A/D CONVERTER

9.1 Function of A/D Converter

The A/D converter converts analog input signals into digital values with a resolution of 8 bits. Eight analog input channels (ANI0 through ANI7) can be controlled.

The A/D conversion operation can be started only by software.

One of the analog input channels, ANI0 through ANI7, is selected for A/D conversion. The A/D conversion operation is repeatedly performed, and each time it has been completed once, an interrupt request (INTAD) is generated.

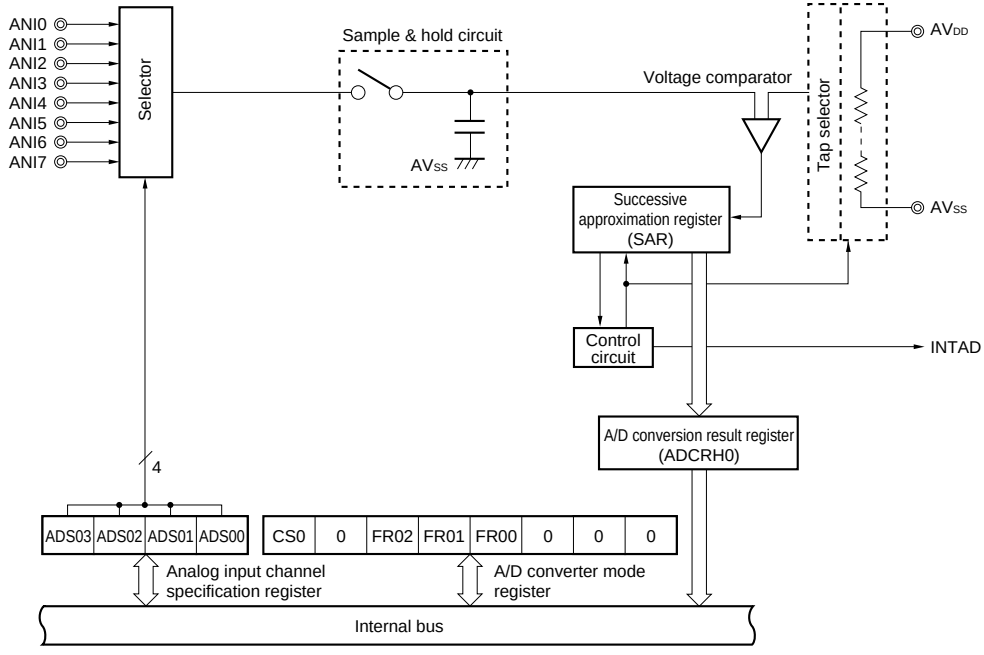
9.2 Configuration of A/D Converter

The A/D converter consists of the following hardware:

Table 9-1. A/D Converter Configuration

Item	Configuration
Analog input	8 channels (ANI0 through ANI7)
Register	Successive approximation register (SAR) A/D conversion result register (ADCRH0)
Control register	A/D converter mode register (ADM0) Analog input channel specification register (ADS0)

Figure 9-1. A/D Converter Block Diagram



(1) Successive approximation register (SAR)

This register compares the voltage value of the input analog signal with the value of the voltage tap (compare voltage) from the series resistor string, and holds the result of the comparison, starting from the most significant bit (MSB).

When the comparison result has been retained to this register up to the least significant bit (LSB) (i.e., when the A/D conversion has been completed), the contents of this register are transferred to the A/D conversion result register (ADCRH0).

(2) A/D conversion result register (ADCRH0)

This register holds the result of the A/D conversion. Each time the A/D conversion has been completed, the conversion result is loaded to this register from the successive approximation register (SAR).

ADCRH0 is read by using an 8-bit memory manipulation instruction.

The value of this register is initialized to 00H by $\overline{\text{RESET}}$ input.

(3) Sample & hold circuit

The sample & hold circuit samples the analog input signals sent from the input circuit one by one, and sends them to the voltage comparator. It also holds the voltage value of the sampled analog input signal during A/D conversion.

(4) Voltage comparator

The voltage comparator compares the analog input signal with the output voltage of the series resistor string.

(5) Series resistor string

The series resistor string is connected between the AV_{DD} and AV_{SS} pins, and generates a voltage to be compared with the input analog signal.

(6) ANI0 through ANI7 pins

These are 8 channels of analog input pins of the A/D converter, and input analog signals to be converted.

Caution Make sure that the input voltages of ANI0 through ANI7 are within the rated range. If a voltage greater than AV_{DD} or less than AV_{SS} is input a channel (even if it is within the absolute maximum rating range), the converted value of the channel is undefined, and, in the worst case, the converted values of the other channels are affected.

(7) AV_{SS} pin

This is the ground potential pin of the A/D converter. Make sure this pin is always at the same potential as the V_{SS1} pin even when the A/D converter is not used.

(8) AV_{DD} pin

This is the analog power supply pin of the A/D converter. Make sure that this pin is always at the same potential as the V_{DD1} pin even when the A/D converter is not used.

In the standby mode, the current flowing to the series resistor string can be lowered by stopping the conversion operation (by clearing bit 7 (CS0) of the A/D converter mode register (ADM0)).

9.3 Registers Controlling A/D Converter

The following two types of registers control the A/D converter.

- A/D converter mode register (ADM0)
- Analog input channel specification register (ADS0)

(1) A/D converter mode register (ADM0)

This register specifies the conversion time of the input analog signal to be converted, and starts or stops the conversion operation.

It is set by using a 1-bit or 8-bit memory manipulation instruction.

The value of this register is initialized to 00H by $\overline{\text{RESET}}$ input.

Figure 9-2. A/D Converter Mode Register Format

Symbol	< 7 >	6	5	4	3	2	1	0	Address	At Reset	R/W
ADM0	CS0	0	FR02	FR01	FR00	0	0	0	FF80H	00H	R/W

CS0	Control A/D conversion operation
0	Stops conversion
1	Enables conversion

FR02	FR01	FR00	Selects A/D conversion time[Note 1]	
			f_X = 5.0 MHz	f_X = 4.19 MHz
0	0	0	144/f_X (28.8 μs)	144/f_X (34.4 μs)
0	0	1	120/f_X (24 μs)	120/f_X (28.6 μs)
0	1	0	96/f_X (19.2 μs)	96/f_X (22.9 μs)
1	0	0	72/f_X (14.4 μs)	72/f_X (17.2 μs)
1	0	1	60/f_X (setting prohibited[Note 2])	60/f_X (14.3 μs)
1	1	0	48/f_X (setting prohibited[Note 2])	48/f_X (setting prohibited[Note 2])
Others			Setting prohibited	

Notes 1. Make sure that the A/D conversion time is 14 μs or longer.

2. These settings are prohibited because the A/D conversion time is less than 14 μs.

Caution The conversion result is undefined immediately after bit 7 (CS0) has been set.

Remark f_X: Main system clock oscillation frequency

(2) Analog input channel specification register (ADS0)

This register specifies a port that inputs the analog voltage to be converted.

It is set by using a 1-bit or 8-bit memory manipulation instruction.

The value of this register is initialized to 00H by $\overline{\text{RESET}}$ input.

Figure 9-3. Analog Input Channel Specification Register Format

Symbol	7	6	5	4	3	2	1	0	Address	At Reset	R/W
ADS0	0	0	0	0	ADS03	ADS02	ADS01	ADS00	FF81H	00H	R/W

ADS03	ADS02	ADS01	ADS00	Specifies analog input channel
0	0	0	0	ANI0
0	0	0	1	ANI1
0	0	1	0	ANI2
0	0	1	1	ANI3
0	1	0	0	ANI4
0	1	0	1	ANI5
0	1	1	0	ANI6
0	1	1	1	ANI7
Others				Setting prohibited

9.4 Operation of A/D Converter

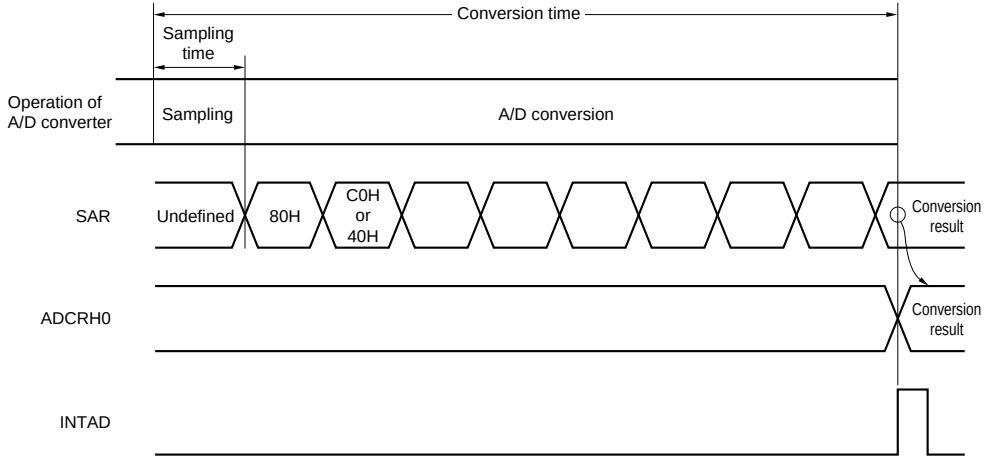
9.4.1 Basic operation of A/D converter

(1) Select one channel for A/D conversion by using the analog input channel specification register (ADS0).
(2) The sample & hold circuit samples the voltage input to the selected analog input channel.
(3) The sample & hold circuit enters the hold status after it has performed sampling for fixed time, and holds the input analog voltage until the A/D conversion is completed.
(4) Bit 7 of the successive approximation register (SAR) is set. The tap selector sets the voltage tap of the series resistor string to (1/2) AV_{DD}.
(5) The voltage comparator compares the voltage difference between the voltage of the series resistor string and voltage tap. If the input analog voltage is greater than (1/2) AV_{DD}, the MSB of SAR remains set. If it is less than (1/2) AV_{DD}, MSB is reset.
(6) Bit 6 of SAR is automatically set, and the next comparison is performed. The voltage tap of the series resistor string is selected as follows, depending on the value of bit 7 to which the result has been already set.
   - Bit 7 = 1: (3/4) AV_{DD}
   - Bit 7 = 0: (1/4) AV_{DD}

   This voltage tap is compared with the input analog voltage. Depending on this result, bit 6 of SAR is manipulated as follows:
   - If input analog voltage • voltage tap: Bit 6 = 1
   - If input analog voltage - voltage tap: Bit 6 = 0

(7) Comparison continues like this up to bit 0 of SAR.
(8) When comparison of 8 bits has been completed, the valid digital result remains in SAR, and its value is transferred and latched to the A/D conversion result register (ADCRH0).
   At the same time, an A/D conversion end interrupt request (INTAD) is generated.

Figure 9-4. Basic Operation of A/D Converter



The A/D conversion operation is performed successively until bit 7 (CS0) of the A/D converter mode register (ADM0) is reset to 0 by software.

If an attempt is made to write data to ADM0 or analog input channel specification register (ADS0) during A/D conversion operation, the conversion operation is initialized, and conversion is started from the beginning if CS0 is set to 1.

The value of the A/D conversion result register (ADCRH0) is undefined when the $\overline{\text{RESET}}$ signal is input.

9.4.2 Input voltage and conversion result

The analog voltage input to an analog input pin (ANI0 to ANI7) and the result of A/D conversion (A/D conversion result register (ADCRH0)) have the following relation:

$$ADCRH0 = INT\left(\frac{V_{IN}}{AV_{DD}} \times 256 + 0.5\right)$$

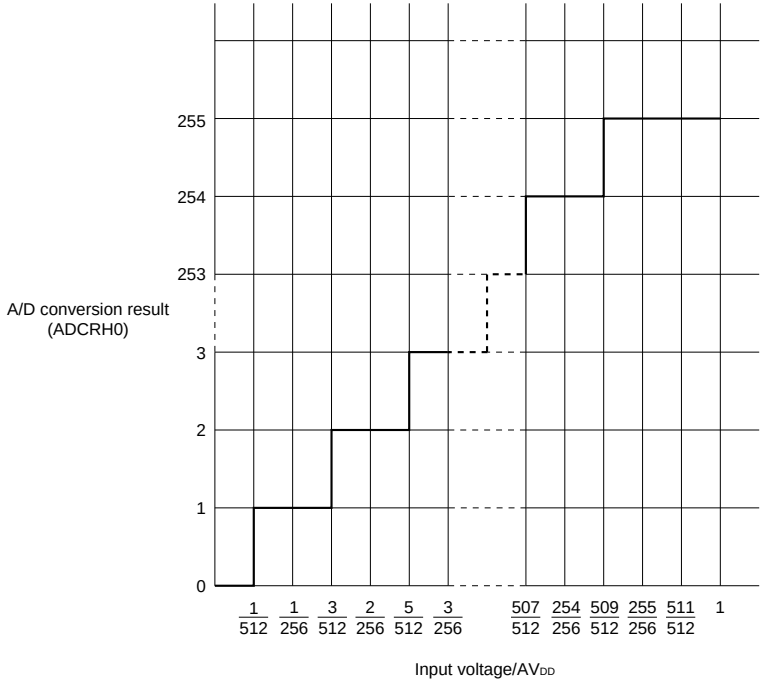
or,

$$(ADCRH0 - 0.5) \times \frac{AV_{DD}}{256} - V_{IN} < (ADCRH0 + 0.5) \times \frac{AV_{DD}}{256}$$

INT( ) : Function that returns integer of value in ( )
V_{IN} : Analog input voltage
AV_{DD} : Supply voltage to A/D converter
ADCRH0 : Value of A/D conversion result register (ADCRH0)

Figure 9-5 shows the relation between the input analog voltage and A/D conversion result.

Figure 9-5. Relation between Input Analog Voltage and A/D Conversion Result



9.4.3 Operation mode of A/D converter

Select one analog input channel from ANI0 through ANI7 by the analog input channel specification register (ADS0) to start A/D conversion.

The A/D conversion operation can be started only by software (by setting the A/D converter mode register (ADM0)).

The A/D conversion result is stored in the A/D conversion result register (ADCRH0), and an interrupt request signal (INTAD) is generated.

- A/D conversion by software start

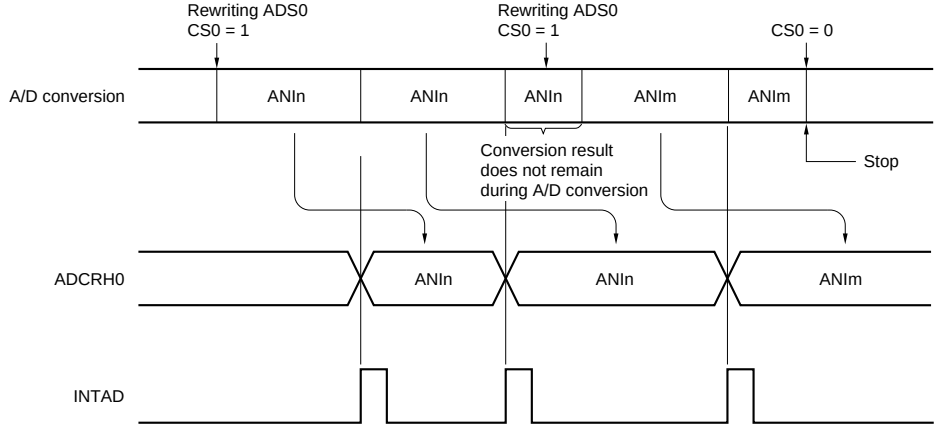
  Converting the voltage applied to the analog input pin specified by the analog input channel specification register (ADS0) is started when bit 7 (CS0) of the A/D converter mode register (ADM0) is set to 1.

  When the A/D conversion has been completed, the result of the conversion is stored in the A/D conversion result register (ADCRH0), and an interrupt request (INTAD) is generated. When the A/D conversion has been started and completed once, the next conversion operation is immediately started. This is repeated until new data is written to ADS0.

★ If ADS0 is rewritten during A/D conversion, the conversion under execution is stopped, and conversion of the selected analog input channel is started.

  If data with CS0 being 0 is written to ADM0 during A/D conversion, the conversion is immediately stopped.

Figure 9-6. A/D Conversion by Software Start



Remark n = 0, 1, ... 7
m = 0, 1, ... 7

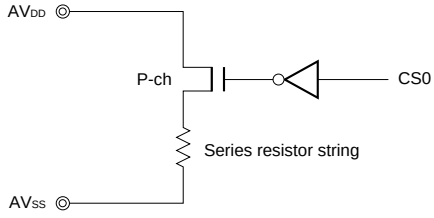
9.5 Notes on A/D Converter

(1) Current consumption in standby mode

The A/D converter is stopped in the standby mode. At this time, the current consumption can be reduced by stopping the conversion (by clearing bit 7 (CS0) of the A/D converter mode register (ADM0) to 0).

Figure 9-7 shows how the current consumption can be reduced in the standby mode.

Figure 9-7. Example of Reducing Current Consumption in Standby Mode



(2) Input range of ANI0 to ANI7

Make sure that the input voltages of ANI0 through ANI7 are within the rated range. If a voltage greater than AV_{DD} or less than AV_{SS} is input to a channel (even if it is within the absolute maximum rating range), the converted value of the channel is undefined, and, in the worst case, the converted values of the other channels are affected.

(3) Conflicting operation

<1> Conflict between writing and reading A/D conversion result register (ADCRH0) on completion of conversion

Reading ADRCH0 takes precedence. After ADCRH0 has been read, a new conversion result is written to ADCRH0.

<2> Conflict between writing ADCRH0 and input of external trigger signal on completion of conversion

An external trigger signal is not accepted during A/D conversion. Therefore, the external trigger signal is not accepted while ADCRH0 is written.

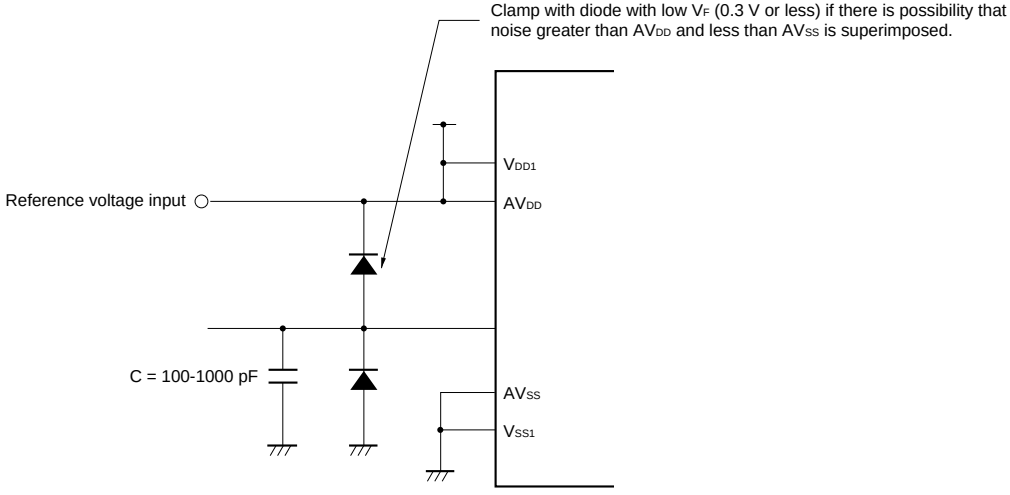
<3> Conflict between writing ADCRH0 and writing A/D converter mode register (ADM0) or writing analog input channel specification register (ADS0) on completion of conversion

Writing ADM0 or ADS0 takes precedence. ADCRH0 is not written. Nor is the conversion end interrupt request signal (INTAD) generated.

(4) Noise measures

To maintain the 8-bit resolution, care must be exercised that no noise is superimposed on the AV_{DD} and ANI0 through ANI7 pins. The higher the output impedance of the analog input source, the heavier the influence of noise. To suppress noise, connecting external C as shown in Figure 9-8 is recommended.

Figure 9-8. Processing of Analog Input Pin



(5) ANI0 through ANI7

The analog input pins (ANI0 through ANI7) are multiplexed with port pins (P10 through P17).

When one of ANI0 through ANI7 is selected for A/D conversion, do not execute an instruction that inputs data to the port during the conversion. If such an instruction is executed, the conversion resolution may drop.

When a digital pulse is applied to a pin adjacent to the analog input pins during A/D conversion, the expected A/D conversion value may not be obtained because of coupling noise. Therefore, do not apply a pulse to the pins adjacent to the analog input pins during A/D conversion.

(6) Input impedance of AV_{DD} pin

A series resistor string with a resistance of about 24.7 kΩ is connected between the AV_{DD} and AV_{SS} pins.

If the output impedance of the reference voltage source is high, therefore, the impedance is virtually connected in parallel with the resistor string between the AV_{DD} and AV_{SS} pins, increasing the error of the reference voltage.

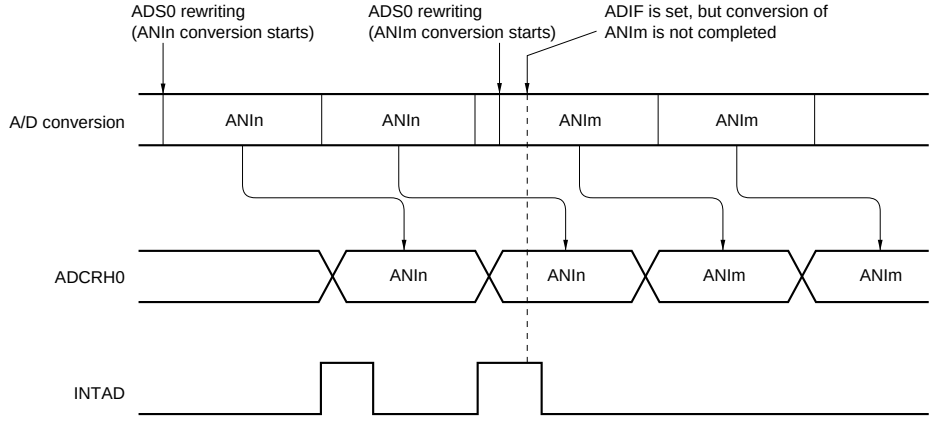
(7) Interrupt request flag (ADIF)

★ The interrupt request flag (ADIF) is not cleared even if the contents of the analog input channel specification register (ADS0) are changed.

If the analog input pin is changed during A/D conversion, therefore, the A/D conversion result of the old analog input may be written to ADS0 immediately before ADS0 is rewritten, and consequently, the conversion end interrupt flag may be set. If ADIF is read immediately after ADS0 has been rewritten, ADIF may be set despite that the A/D conversion of the new analog input has not been completed.

Before resuming A/D conversion that has been stopped, clear ADIF.

Figure 9-9. A/D Conversion End Interrupt Request Generation Timing

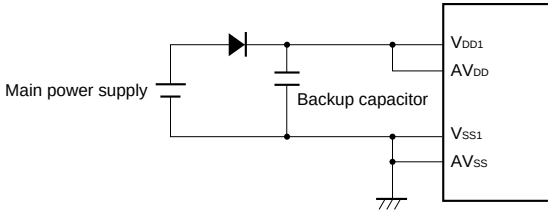


Remark n = 0, 1, ... 7
m = 0, 1, ... 7

(8) AV_{DD} pin

The AV_{DD} pin supplies power to the analog circuit. It also supplies power to the input circuit of ANI0 through ANI7. Therefore, apply the same potential as that of the V_{DD1} pin to this pin, as shown in Figure 9-10, in an application where a backup power supply is used.

Figure 9-10. Processing of AV_{DD} Pin



CHAPTER 10 SERIAL INTERFACE

10.1 Function of Serial Interface

The serial interface has the following two modes.

- Operation stop mode
- Three-wire serial I/O mode

(1) Operation stop mode

This mode is used when serial transfer is not performed.

(2) Three-wire serial I/O mode (with MSB first)

In this mode, 8-bit data is transferred by using three lines: serial clock ($\overline{SCK}$), serial output (SO), and serial input (SI).

Because simultaneous transmission/reception operation can be performed in the three-wire serial I/O mode, the processing time of data transfer can be shortened.

The first bit of the 8-bit data to be transferred is fixed to the MSB.

The three-wire serial I/O mode is useful when connecting peripheral I/Os or display controller having a clocked serial interface.

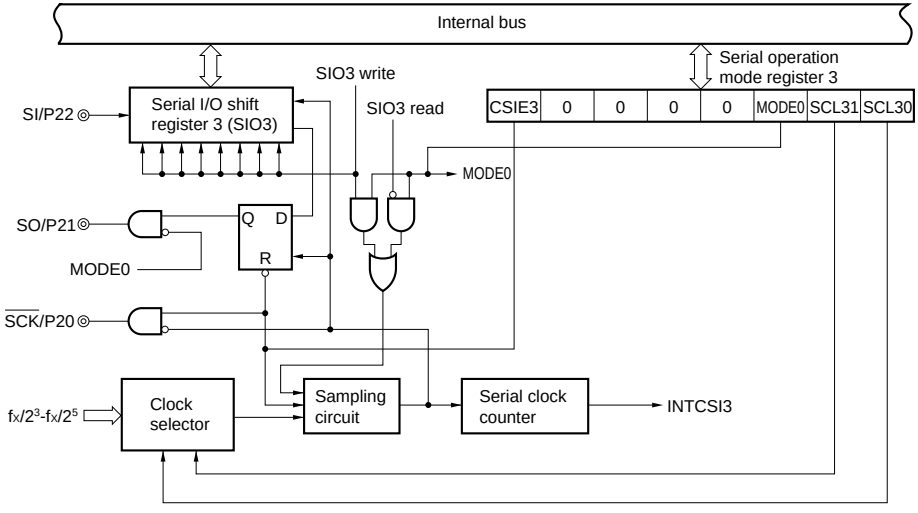
10.2 Configuration of Serial Interface

The serial interface consists of the following hardware.

Table 10-1. Serial Interface Configuration

Item	Configuration
Register	Serial I/O shift register 3 (SIO3)
Control register	Serial operation mode register 3 (CSIM3)

Figure 10-1. Serial Interface Block Diagram



(1) Serial I/O shift register 3 (SIO3)

This 8-bit register converts parallel data to serial data to perform serial transmission/reception (shift operation) in synchronization with the serial clock.

SIO3 is set by using an 8-bit memory manipulation instruction.

The serial operation is started by writing data to or reading data from SIO3 when bit 7 (CSIE3) of the serial operation mode register 3 (CSIM3) is 1.

During transmission, the data written to SIO3 is output to the serial output line (SO).

During reception, the data is read to SIO3 from the serial input line (SI).

The contents of this register are undefined when the $\overline{\text{RESET}}$ signal is input.

Caution Do not access SIO3 during transmission, except when triggering the transmission (reading SIO3 is prohibited when bit 2 (MODE0) of CSIM3 = 0, and writing is prohibited when MODE0 = 1).

(2) Serial clock counter

This counter counts the serial clock output or input during transmission/reception to check that 8-bit data has been transmitted/received.

10.3 Registers Controlling Serial Interface

The serial interface is controlled by the serial operation mode register 3 (CSIM3).

- **Serial operation mode register 3 (CSIM3)**
  This register selects the serial clock and operation mode of the serial interface, and enables or disables the operation.
  It is set by using a 1-bit or 8-bit memory manipulation instruction.
  The value of this register is initialized to 00H by $\overline{\text{RESET}}$ input.

Figure 10-2. Serial Operation Mode Register 3 Format

Symbol	< 7 >	6	5	4	3	2	1	0	Address	At Reset	R/W
CSIM3	CSIE3	0	0	0	0	MODE0	SCL31	SCL30	FF86H	00H	R/W

CSIE3	Enables or disables operation of SIO3		
	Shift register operation	Serial counter	Port
0	Disabled	Cleared	Port function[Note]
1	Enabled	Count operation enabled	Serial function + port function

MODE0	Transfer operation mode flag		
	Operation mode	Transfer start trigger	SO output
0	Transmission or transmission/ reception mode	SIO3 write	Normal output
1	Reception mode	SIO3 read	Fixed to low level

SCL31	SCL30	Selects clock
0	0	External clock input to $\overline{\text{SCK}}$ pin
0	1	$f_X/2^3$ (625 kHz)
1	0	$f_X/2^4$ (313 kHz)
1	1	$f_X/2^5$ (156 kHz)

Note The pins connected to SI, SO, and $\overline{\text{SCK}}$ can be used as port pins when CSIE3 = 0 (when the SIO3 operation is stopped).

Remarks 1. f_X: Main system clock oscillation frequency
2. ( ): f_X = 5.0 MHz

10.4 Operation of Serial Interface

The serial interface operates in the following two modes:

- Operation stop mode
- Three-wire serial I/O mode

10.4.1 Operation stop mode

In the operation stop mode, the power consumption can be reduced because serial transfer is not executed.

Because the serial I/O shift register 3 (SIO3) does not perform the shift operation, this register can be used as a normal 8-bit register.

In this mode, the P20/$\overline{SCK}$, P21/SO, and P22/SI pins can be used as normal I/O port pins.

(1) Register setting

The operation stop mode is set by the serial operation mode register 3 (CSIM3).

CSIM3 is set by using a 1-bit or 8-bit memory manipulation instruction.

The value of this register is initialized to 00H by $\overline{RESET}$ input.

Symbol	< 7 >	6	5	4	3	2	1	0	Address	At Reset	R/W
CSIM3	CSIE3	0	0	0	0	MODE0	SCL31	SCL30	FF86H	00H	R/W

CSIE3	Enables or disables operation of SIO3		
	Shift register operation	Serial counter	Port
0	Disabled	Cleared	Port function[Note]
1	Enabled	Count operation enabled	Serial function + port function

Note The pins connected to SI, SO, and $\overline{SCK}$ can be used as port pins when CSIE3 = 0 (when the SIO3 operation is stopped).

10.4.2 Three-wire serial I/O mode

The three-wire serial I/O mode is useful for connecting a peripheral I/O or display controller having a clocked serial interface.

Communication is established by using three lines: serial clock ($\overline{SCK}$), serial output (SO), and serial input (SI).

(1) Register setting

The three-wire serial I/O mode is set by the serial operation mode register 3 (CSIM3).

It is set by using a 1-bit or 8-bit memory manipulation instruction.

The value of this register is initialized to 00H by $\overline{RESET}$ input.

Symbol	< 7 >	6	5	4	3	2	1	0	Address	At Reset	R/W
CSIM3	CSIE3	0	0	0	0	MODE0	SCL31	SCL30	FF86H	00H	R/W

CSIE3	Enables or disables operation of SIO3		
	Shift register operation	Serial counter	Port
0	Disabled	Cleared	Port function[Note]
1	Enabled	Count operation enabled	Serial function + port function

MODE0	Transfer operation mode flag		
	Operation mode	Transfer start trigger	SO output
0	Transmission or transmission/ reception mode	SIO3 write	Normal output
1	Reception mode	SIO3 read	Fixed to low level

SCL31	SCL30	Selects clock
0	0	External clock input to $\overline{SCK}$ pin
0	1	$f_X/2^3$ (625 kHz)
1	0	$f_X/2^4$ (313 kHz)
1	1	$f_X/2^5$ (156 kHz)

Note The pins connected to SI, SO, and $\overline{SCK}$ can be used as port pins when CSIE3 = 0 (when the SIO3 operation is stopped).

Remarks 1. f_X: Main system clock oscillation frequency

2. ( ): f_X = 5.0 MHz

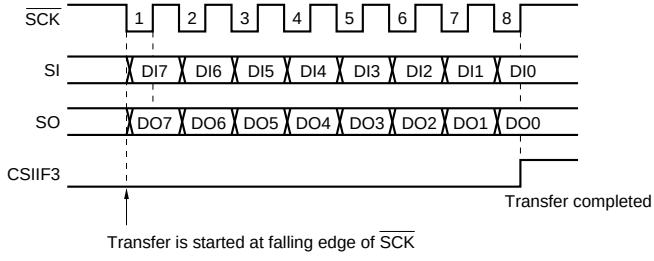
(2) Communication operation

In the three-wire serial I/O mode, data is transmitted or received in 8-bit units. Each bit of the data is transmitted or received in synchronization with the serial clock.

The shift operation of the serial I/O shift register 3 (SIO3) is performed in synchronization with the falling of the serial clock ($\overline{SCK}$). The transmit data is retained by the SO latch and output from the SO pin. At the rising edge of $\overline{SCK}$, the receive data input to the SI pin is latched to SIO3.

When transfer of 8-bit data has been completed, SIO3 automatically stops its operation, and an interrupt request flag (CSIIF3) is set.

Figure 10-3. Timing in Three-Wire Serial I/O Mode



(3) Transfer start

Serial transfer is started when data is assigned to (or read from) the serial I/O shift register 3 (SIO3) if the following two conditions are satisfied.

- Operation control bit of SIO3 (CSIE3) = 1
- If the internal serial clock is stopped or $\overline{SCK}$ is high after 8-bit serial transfer
- Transmission or transmission/reception mode
  Transfer is started if SIO3 is written when CSIE3 = 1 and MODE0 = 0.
- Reception mode
  Transfer is started if SIO3 is read when CSIE3 = 1 and MODE0 = 1.

Caution Transfer is not started even if CSIE3 is set to "1" after data has been written to SIO3.

Serial transfer is automatically stopped and an interrupt request flag (CSIIF3) is set when 8-bit transfer has been completed.

CHAPTER 11 FIP CONTROLLER/DRIVER

11.1 Function of FIP Controller/Driver

The FIP controller/driver of the μPD780228 subseries has the following functions.

(1) Can output display signals (DMA operation) by automatically reading display data.
(2) The pins not used for FIP display can be used as I/O port or output port pins (FIP16 through FIP47 pins only).
(3) Luminance can be adjusted in 8 steps by display mode register 1 (DSPM1).
(4) Hardware for key scan application
    - Generates an interrupt signal (INTKS) indicating key scan timing
    - Timing in which key scan data is output can be detected by key scan flag (KSF).
    - Whether key scan timing is inserted or not can be selected.
(5) High-voltage output buffer that can directly drive FIP
(6) FIP0 through FIP15 pins are connected to pull-down resistors. FIP16 through FIP47 pins can be connected to pull-down resistors by mask option (mask ROM model only). The μPD78F0228 does not have pull-down resistors).

Of the 48 FIP output pins of the μPD780228 subseries, FIP16 through FIP47 are multiplexed with port pins. FIP0 through FIP15 are dedicated output pins.

FIP16 through FIP47 can be used as port pins when FIP display is disabled by bit 7 (DSPEN) of the display mode register 0 (DSPM0). Even when FIP display is enabled, the FIP output pins not used for display signal output can be used as port pins.

Table 11-1. FIP Output Pins and Multiplexed Port Pins

FIP Pin Name	Multiplexed Port Name	I/O
FIP16-FIP23	P70-P77	I/O port
FIP24-FIP31	P80-P87	I/O port
FIP32-FIP39	P90-P97	I/O port
FIP40-FIP47	P100-P107	Output port

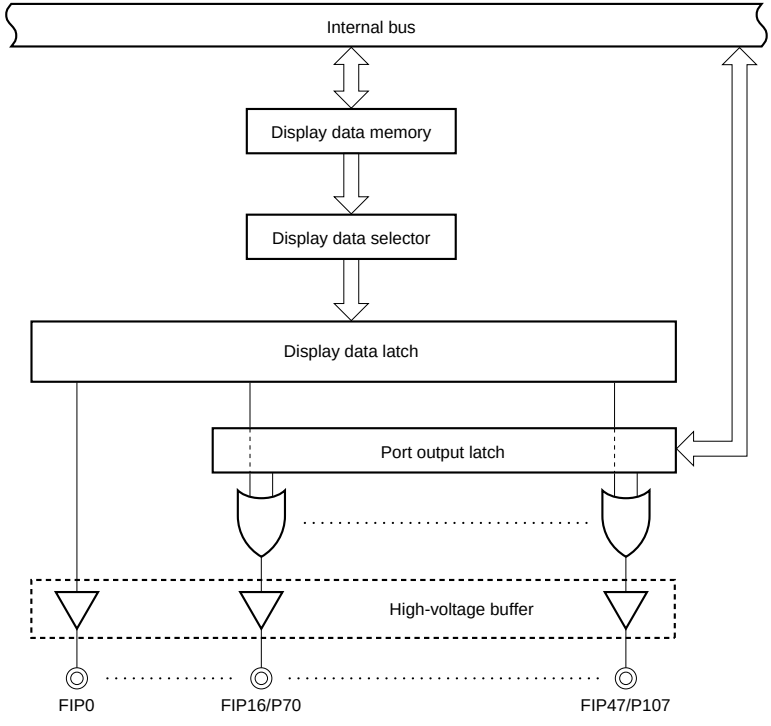
11.2 Configuration of FIP Controller/Driver

The FIP controller/driver consists of the following hardware.

Table 11-2. FIP Controller/Driver Configuration

Item	Configuration
Display	48
Control register	Display mode register 0 (DSPM0) Display mode register 1 (DSPM1) Display mode register 2 (DSPM2)

Figure 11-1. FIP Controller/Driver Block Diagram



11.3 Registers Controlling FIP Controller/Driver

11.3.1 Control registers

The following three types of registers control the FIP controller/driver.

- Display mode register 0 (DSPM0)
- Display mode register 1 (DSPM1)
- Display mode register 2 (DSPM2)

(1) Display mode register 0 (DSPM0)

DSPM0 performs the following setting.

- Enables or disables display
- Number of FIP output pins

DSPM0 Is set by using a 1-bit or 8-bit memory manipulation instruction.
The value of this register is set to 10H by $\overline{\text{RESET}}$ input.

Figure 11-2. Display Mode Register 0 Format

Symbol	< 7 >	6	5	4	3	2	1	0	Address	At Reset	R/W
DSPM0	DSPEN	0	FOUT5	FOUT4	FOUT3	FOUT2	FOUT1	FOUT0	FF90H	10H	R/W

DSPEN	Enables or disables FIP
0	Disables
1	Enables

★

FOUT5	FOUT4	FOUT3	FOUT2	FOUT1	FOUT0	Number of FIP output pins
0	1	0	1	1	1	17-24
0	1	1	1	1	1	25-32
1	0	0	1	1	1	33-40
1	0	1	1	1	1	41-48
Others						Setting prohibited

Cautions 1. Be sure to set bit 6 to “0”.

2. Do not write data to the bits other than DSPEN when bit 7 (DSPEN) = 1.

★ **3. Be sure to set the output latch of the multiplexed port of a pin used for FIP output to “0”.**

(2) Display mode register 1 (DSPM1)

DSPM1 performs the following setting:

- Blanking width of FIP output signal
- Number of display patterns

DSPM1 is set by using an 8-bit memory manipulation instruction.
The value of this register is set to 01H by $\overline{\text{RESET}}$ input.

Figure 11-3. Display Mode Register 1 Format

Symbol	7	6	5	4	3	2	1	0	Address	At Reset	R/W
DSPM1	FBLK2	FBLK1	FBLK0	FPAT4	FPAT3	FPAT2	FPAT1	FPAT0	FF91H	01H	R/W

FBLK2	FBLK1	FBLK0	Blanking width of FIP output signal
0	0	0	1/16
0	0	1	2/16
0	1	0	4/16
0	1	1	6/16
1	0	0	8/16
1	0	1	10/16
1	1	0	12/16
1	1	1	14/16

FPAT4	FPAT3	FPAT2	FPAT1	FPAT0	Number of display patterns
0	0	0	0	1	2
0	0	0	1	0	3
0	0	0	1	1	4
0	0	1	0	0	5
0	0	1	0	1	6
0	0	1	1	0	7
0	0	1	1	1	8
0	1	0	0	0	9
0	1	0	0	1	10
0	1	0	1	0	11
0	1	0	1	1	12
0	1	1	0	0	13
0	1	1	0	1	14
0	1	1	1	0	15
0	1	1	1	1	16
Others					Setting prohibited

Caution Do not write data to the display mode register 1 (DSPM1) when bit 7 (DSPEN) of the display mode register 0 (DSPM0) is 1.

(3) Display mode register 2 (DSPM2)

DSPM2 performs the following setting. It also indicates the status of the display timing/key scan.

- Insertion of key scan timing
- Display cycle (TDSP)

DSPM2 is set by using a 1-bit or 8-bit memory manipulation instruction. However, only bit 7 (KSF) can be read by a 1-bit memory manipulation instruction.

The value of this register is initialized to 00H by $\overline{\text{RESET}}$ input.

Figure 11-4. Display Mode Register Format

Symbol	< 7 >	6	5	4	3	2	1	0	Address	At Reset	R/W
DSPM2	KSF	KSM	0	0	0	0	FCYC1	FCYC0	FF92H	00H	R/W

KSF	Status of key scan cycle
0	Other than key scan cycle
1	Key scan cycle

KSM	Selects insertion of key scan cycle
0	Not inserted
1	Inserted

	FCYC1	FCYC0	Display cycle
★	0	0	$2^{12}/f_X$ (819.2 μs)
	0	1	$2^{11}/f_X$ (409.6 μs)
★	1	0	$2^{10}/f_X$ (204.8 μs)
	1	1	Setting prohibited

Cautions 1. Be sure to set bits 2 through 5 to "0".

2. Do not write data to the display mode register 2 (DSPM2) when bit 7 (DSPEN) of the display mode register 0 (DSPM0) is 1.

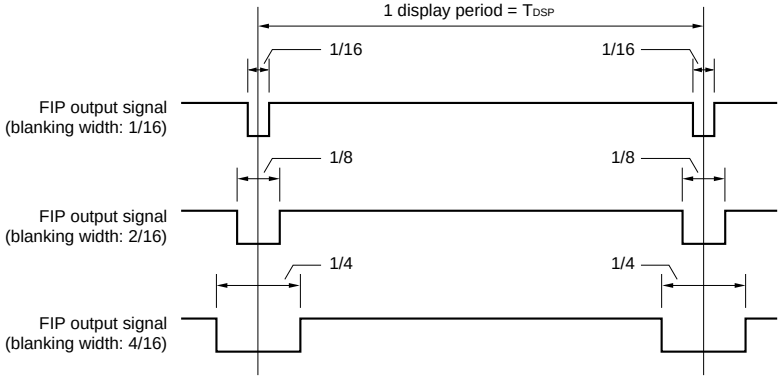
Remarks 1. f_X: Main system clock oscillation frequency

2. ( ): f_X = 5.0 MHz

11.3.2 One display period and blanking width

The FIP output signals are blanked equally at the beginning and end of the display period by the blanking width set by bits 0 through 2 (FBLK0 through FBLK2) of the display mode register 1 (DSPM1).

Figure 11-5. Blanking Width of FIP Output Signal



11.4 Display Data Memory

The display data memory is a 96-byte RAM area that stores data to be displayed, and is mapped to addresses FA00H through FA5FH.

The FIP controller reads the data stored in the display data memory independently of the CPU operation for FIP display (DMA operation).

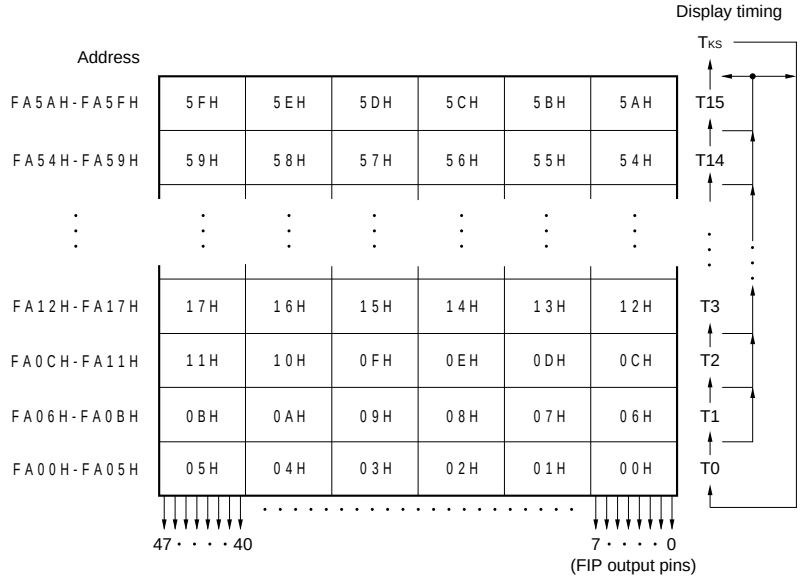
The area of the display data memory not used for display can be used as a normal RAM area.

At key scan timing, all the FIP output pins are cleared to “0”, and the data of the output latches of ports 7 through 10 are output to FIP16/P70 through FIP47/P107.

The address location of the display data memory is as follows:

- **With 48 FIP output pins and 16 patterns**
  The addresses of the display data memory corresponding to the data output at each display timing (T0 through T15) are as shown in Figure 11-6 (for example, T0 = FA00H through FA05H, and T1 = FA06H through FA0BH). When 48 FIP output pins (FIP0 through FIP47) are used, one block of display data consists of 6 bytes. FIP output pins 0 (FIP0) through 47 (FIP47) correspond to one block of display data sequentially, starting from the least significant bit toward the most significant bit.

Figure 11-6. Relation between Address Location of Display Data Memory and FIP Output (with 48 FIP output pins and 16 patterns)



11.5 Key Scan Flag and Key Scan Data

11.5.1 Key scan flag

The key scan flag (KSF) is set to 1 during key scan timing, and is automatically reset to 0 at display timing.

KSF is mapped to bit 7 of the display mode register 2 (DSPM2) and can be tested in 1-bit units. It cannot be written, however.

By testing KSF, it can be determined whether key scan timing is in progress, and whether key input data is correct can be checked.

Whether key scan timing is inserted or not can be selected by using the key scan timing insertion specification flag (KSM) (bit 6 of the display mode register 2 (DSPM2)).

11.5.2 Key scan data

Data stored to ports 7 through 10 are output from the FIP16 through FIP47 pins during key scan timing.

Caution If scanning is performed in such a manner that both a segment and a digit turn ON during key scan timing, the display may flicker.

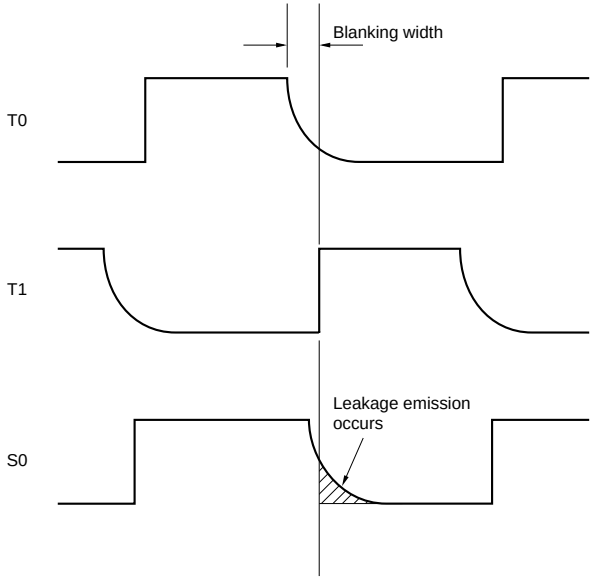
11.6 Leakage Emission of Fluorescent Indicator Panel

Leakage emission may take place when a fluorescent indicator panel is driven by the μPD780228 subseries. The possible causes of this leakage emission are as follows:

(1) Short blanking time

Figure 11-7 shows the signal waveforms of a 2-digit display where the first digit T0 lights and the second digit remains dark. If the blanking time is too short as shown in this figure, the T1 signal rises before the segment signal is deasserted, causing leakage emission. Generally, the blanking time must be about 20 μs. Determine the set value of the display mode register 1 (DSPM1), taking this into consideration.

Figure 11-7. Leakage Emission Because of Short Blanking Time



(2) Segment-grid capacitance of fluorescent indicator panel

Even if a sufficiently long blanking time is ensured as shown in Figure 11-9, leakage emission may still occur. This is because the fluorescent indicator panel has a capacitance between the grid and segment, as indicated by C_{SG} in the figure, and the timing signal pin is raised via C_{SG}. If the voltage of the timing signal rises beyond the cutoff voltage (E_K) as shown in Figure 11-9, leakage emission occurs.

This whisker-like voltage changes with the values of C_{SG} and internal pull-down resistor (R_L). The greater the value of C_{SG}, and the greater the value of R_L, the higher this voltage, increasing the possibility of the occurrence of leakage emission.

The value of C_{SG} differs depending on the display area of the fluorescent indicator panel. The larger the area, the higher the C_{SG}.

Therefore, the value of the pull-down resistor differs depending on the size of the fluorescent indicator panel, in order to prevent leakage emission.

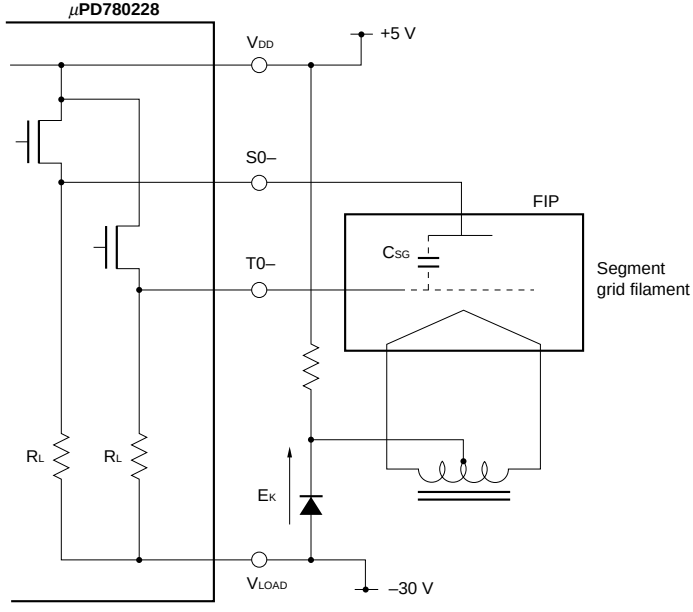
Because the value of the pull-down resistor that can be connected by mask option is relatively high, the leakage emission may not be suppressed by the internal pull-down resistor alone.

In case sufficient display quality cannot be obtained, deepen the back bias (increase E_K), attach a filter to the fluorescent indicator panel, or connect an external pull-down resistor of several 10 kΩ to the timing signal pin.

The likelihood of leakage emission caused by C_{SG} occuring changes depending on the duty cycle of the whisker voltage vis-a-vis the total display cycle. The fewer the number of display digits, the less likelihood of occurrence of leakage emission.

Lowering the display luminance also has an effect of suppressing the leakage emission.

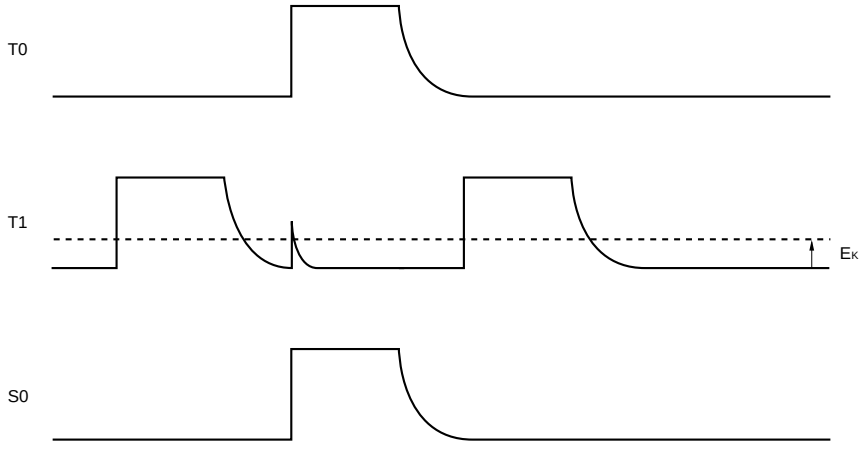
Figure 11-8. Leakage Emission Caused by C_{SG}



E_K : Cutoff voltage

R_L : Internal pull-down resistor

Figure 11-9. Leakage Emission Caused by C_{SG}

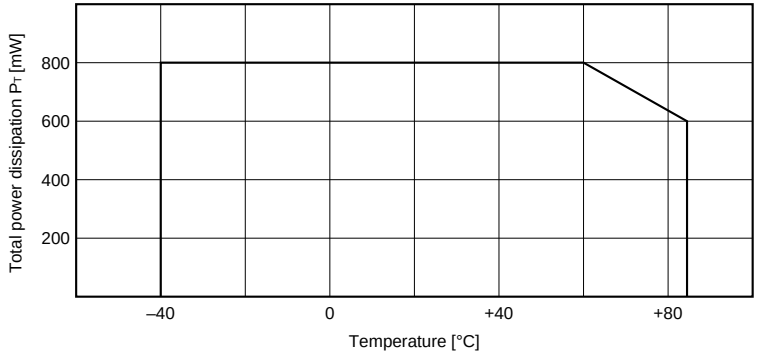


★ 11.7 Calculation of Total Power Dissipation

The following three power dissipation are available for the μPD780208 subseries. The sum of the three power dissipation should be less than the total power dissipation P_T (refer to **Figure 11-10**) (80 % or less of ratings is recommended).

<1> CPU power dissipation: Calculate V_{DD} (MAX.) × I_{DD} (MAX.).
<2> Output pin power dissipation: Power dissipation when maximum current flows into each FIP output pin.
<3> Pull-down resistor power dissipation: Power dissipation by pull-down resistor incorporated in FIP output pin.

Figure 11-10. Total Power Dissipation P_T (T_A = –40 to +85 °C)



The following is how to calculate total power dissipation for the example in Figure 11-11.

Example Assume the following conditions:
V_{DD} = 5.5 V, 5.0 MHz oscillation
Supply current (I_{DD}) = 21.0 mA
FIP output: 11 grids × 10 segments (Blanking width = 1/16: when FBLK0-FBLK2 = 000B)
Maximum current at the grid pin is 10 mA.
Maximum current at the segment pin is 3 mA.
At the key scan timing, FIP output pin is OFF.
FIP output voltage: grid $V_{OD} = V_{DD} – 2$ V (voltage drop of 2 V)
segments $V_{OD} = V_{DD} – 0.5$ V (voltage drop of 0.5 V)
Fluorescant display control voltage (V_{LOAD}) = –35 V
Mask option pull-down resistor = 25 kΩ

By placing the above conditions in calculation <1> to <3>, the total dissipation can be worked out.

<1> CPU power dissipation: 5.5 V × 21.0 mA = 115.5 mW

<2> Output pin power dissipation:

Grid $(V_{DD} - V_{OD}) \times \dfrac{\text{Total current value of each grid}}{\text{The no. of grids + 1}} \times (1 - \text{Blanking width}) =$

$$2\ V \times \frac{10\ mA \times 11\ Grids}{11\ Grids + 1} \times \left(1 - \frac{1}{16}\right) = 17.2\ mW$$

Segment $(V_{DD} - V_{OD}) \times \dfrac{\text{Total segment current value of illuminated dots}}{\text{The no. of grids + 1}} \times (1 - \text{Blanking width}) =$

$$0.5\ V \times \frac{3\ mA \times 31\ Dots}{11\ Grids + 1} \times \left(1 - \frac{1}{16}\right) = 3.6\ mW$$

<3> Pull-down resistor power dissipation:

Grid $\dfrac{(V_{DD} - V_{LOAD})^2}{\text{Pull-down resistor value}} \times \dfrac{\text{The no. of grids}}{\text{The no. grids + 1}} \times (1 - \text{Blanking width}) =$

$$\frac{(5.5\ V - 2\ V - (-35\ V))^2}{25\ k\Omega} \times \frac{11\ Grids}{11\ Grids + 1} \times \left(1 - \frac{1}{16}\right) = 50.9\ mW$$

Segment $\dfrac{(V_{OD} - V_{LOAD})^2}{\text{Pull-down resistor value}} \times \dfrac{\text{The no. of illuminated dots}}{\text{The no. of grids + 1}} \times (1 - \text{Blanking width}) =$

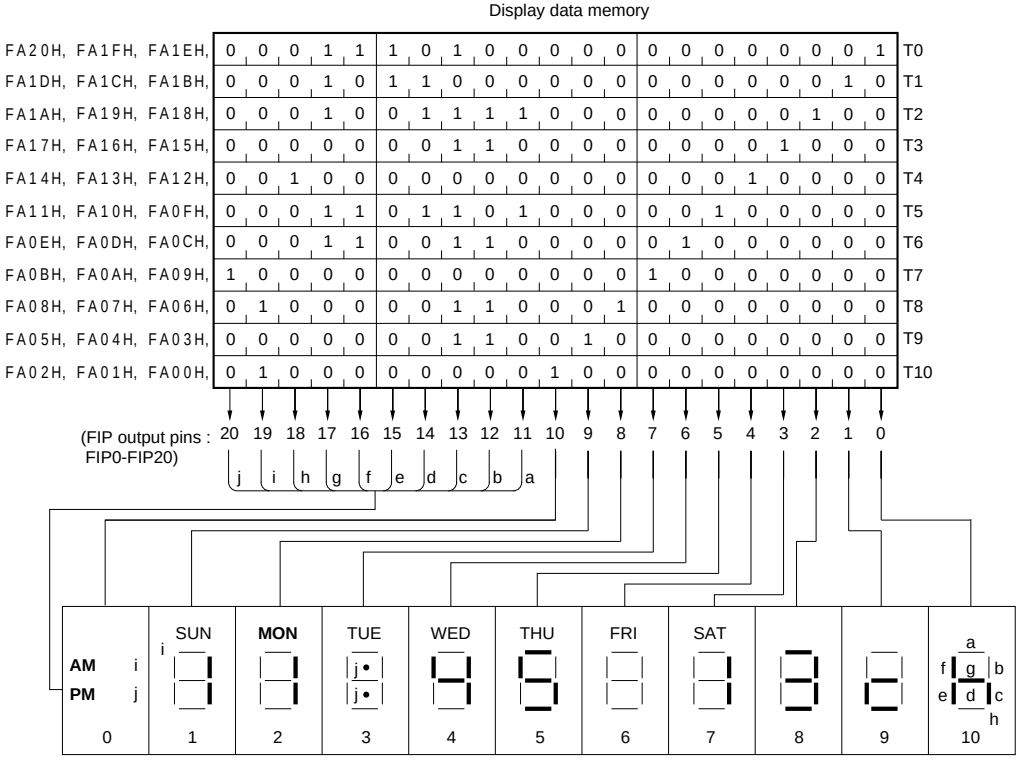
$$\frac{(5.5\ V - 0.5\ V - (-35\ V))^2}{25\ k\Omega} \times \frac{31\ dots}{11\ Grids + 1} \times \left(1 - \frac{1}{16}\right) = 155.0\ mW$$

Total power dissipation = <1> + <2> + <3> = 115.5 + 17.2 + 3.6 + 50.9 + 155.0 = 342.2 mW

In this example, the total power dissipation do not exceed the rating of the total power dissipation, so there is no problem in power dissipation.

However, when the total power dissipation exceeds the rating of the total power dissipation, it is necessary to lower the power dissipation. To reduce power dissipation, reduce the number of pull-down resistor.

Figure 11-11. Relationship between Display Data Memory and FIP Output with 10 Segments-11 Digits Displayed



[MEMO]

CHAPTER 12 INTERRUPT FUNCTIONS

12.1 Interrupt Function Types

The following three types of interrupt functions are used.

(1) Non-maskable interrupt

This interrupt is acknowledged unconditionally even in a disabled state. It does not undergo interrupt priority control and is given top priority over all other interrupt requests.
It generates a standby release signal.
One interrupt request source from the watchdog timer is incorporated as a non-maskable interrupt.

(2) Maskable interrupts

These interrupts undergo mask control. Maskable interrupts can be divided into a high interrupt priority group and a low interrupt priority group by setting the priority specify flag register (PR0L and PR0H).
Multiple high priority interrupts can be applied to low priority interrupts. If two or more interrupts with the same priority are simultaneously generated, each interrupt has a predetermined priority (see **Table 12-1**).
A standby release signal is generated.
Four external interrupt request sources and six internal interrupt request sources are incorporated as maskable interrupts.

(3) Software interrupt

This is a vectored interrupt to be generated by executing the BRK instruction. It is acknowledged even in a disabled state. The software interrupt does not undergo interrupt priority control.

12.2 Interrupt Sources and Configuration

A total of 12 non-maskable, maskable and software interrupts are incorporated in the interrupt sources (see **Table 12-1**).

Table 12-1. Interrupt Sources

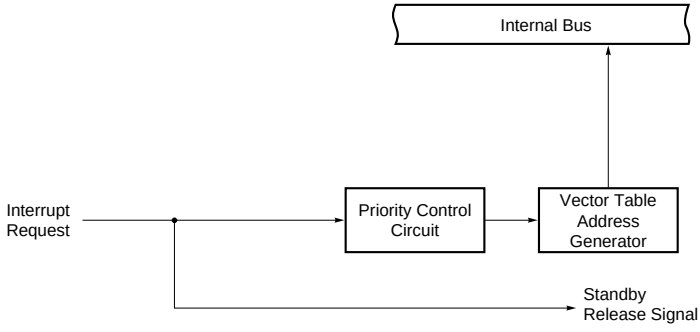
Interrupt Type	Default Priority[Note 1]	Interrupt Source		Internal/ External	Vector Table Address	Basic Configuration Type[Note 2]
		Name	Trigger			
Non-maskable	—	INTWDT	Overflow of watchdog timer (with watchdog timer mode 1 selected)	Internal	0004H	(A)
Maskable	0	INTWDT	Overflow of watchdog timer (with internal timer mode selected)			(B)
	1	INTP0	Detection of pin input edge	External	0006H	(C)
	2	INTP1			0008H	
	3	INTTM10	Detection of timer input edge		000AH	(D)
	4	INTTM11			000CH	
	5	INTKS	Key scan timing from FIP controller/driver	Internal	000EH	(B)
	6	INTCSI3	End of transfer of serial interface		0010H	
	7	INTTM50	Coincidence of 8-bit timer (TM50)		0012H	
	8	INTTM51	Coincidence of 8-bit timer (TM51)		0014H	
	9	INTAD	End of A/D conversion		0016H	
Software	—	BRK	Execution of BRK instruction	—	003EH	(E)

Notes 1. If two or more maskable interrupt requests are simultaneously generated, they are controlled according to the default priority. 0 indicates the highest priority and 9 indicates the lowest.

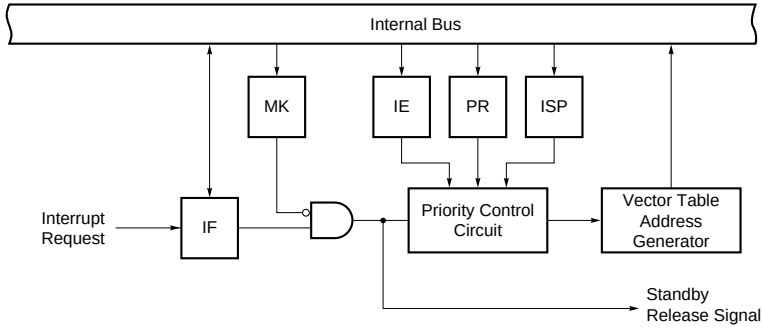
2. (A) through (E) under the heading Basic Configuration Type correspond to (A) through (E) in Figure 12-1.

Figure 12-1. Basic Configuration of Interrupt Function (1/2)

(A) Internal non-maskable interrupt



(B) Internal maskable interrupt



(C) External maskable interrupt (INTP0)

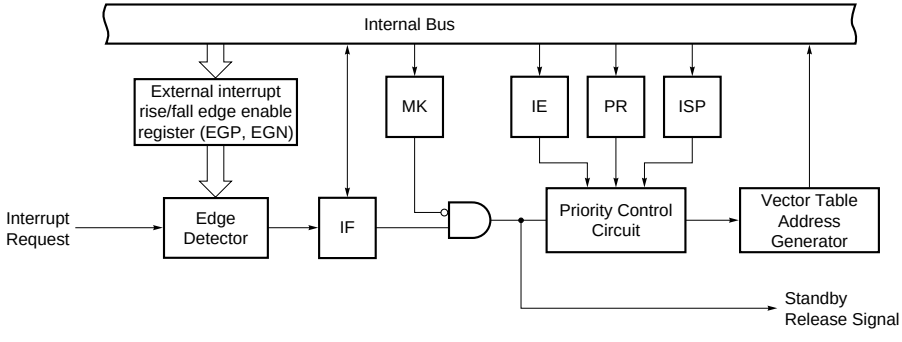
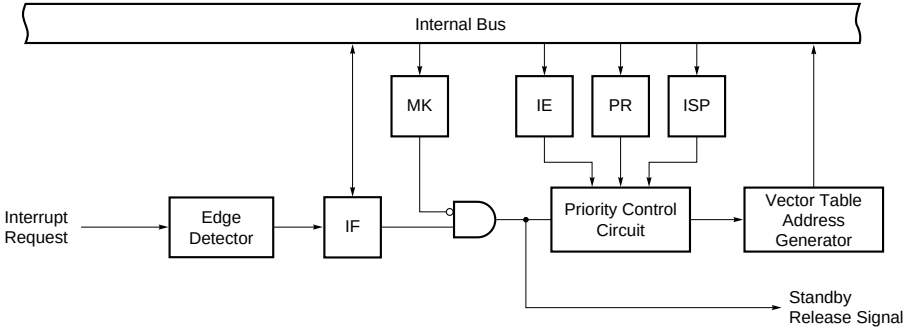
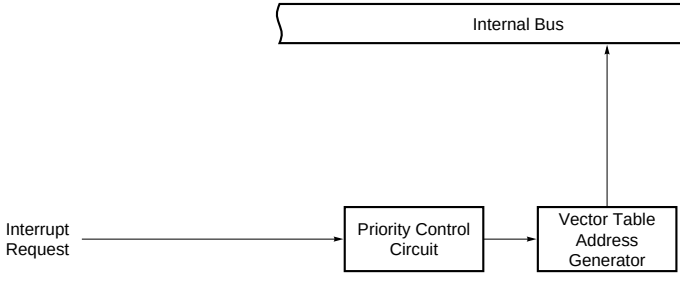


Figure 12-1. Basic Configuration of Interrupt Function (2/2)

(D) External maskable interrupt (INTTM10, INTTM11)



(E) Software interrupt



IF : Interrupt request flag
IE : Interrupt enabled flag
ISP: In-service priority flag
MK : Interrupt mask flag
PR : Priority specify flag

12.3 Interrupt Function Control Registers

The following six types of registers are used to control the interrupt functions.

- Interrupt request flag register (IF0L and IF0H)
- Interrupt mask flag register (MK0L and MK0H)
- Priority specify flag register (PR0L and PR0H)
- External interrupt rising edge enable register (EGP)
- External interrupt falling edge enable register (EGN)
- Program status word (PSW)

Table 12-2 gives a listing of interrupt request flags, interrupt mask flags and priority specify flag names corresponding to interrupt request sources.

Table 12-2. Various Flags Corresponding to Interrupt Request Sources

Interrupt Request Source	Interrupt Request Flag		Interrupt Mask Flag		Priority Specify Flag	
		Register		Register		Register
INTWDT	WDTIF[Note]	IF0L	WDTMK[Note]	MK0L	WDTPR[Note]	PR0L
INTP0	PIF0		PMK0		PPR0	
INTP1	PIF1		PMK1		PPR1	
INTTM10	TMIF10		TMMK10		TMPR10	
INTTM11	TMIF11		TMMK11		TMPR11	
INTKS	KSIF		KSMK		KSPR	
INTCSI3	CSIIF3		CSIMK3		CSIPR3	
INTTM50	TMIF50		TMMK50		TMPR50	
INTTM51	TMIF51	IF0H	TMMK51	MK0H	TMPR51	PR0H
INTAD	ADIF		ADMK		ADPR	

Note The WDTIF, WDTMK, and WDTPR flags are interrupt control flags used when the watchdog timer is used as an interval timer.

(1) Interrupt request flag registers (IF0L and IF0H)

The interrupt request flag is set to (1) when the corresponding interrupt request is generated or an instruction is executed. It is cleared to (0) when an instruction is executed upon acknowledgment of an interrupt request or upon application of $\overline{\text{RESET}}$ input.

IF0L and IF0H are set with a 1-bit or 8-bit memory manipulation instruction. When IF0L and IF0H are used in combination as a 16-bit register IF0, they are set with a 16-bit memory operation instruction.

$\overline{\text{RESET}}$ input sets these registers to 00H.

Figure 12-2. Interrupt Request Flag Register Format

Symbol	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>	Address	At Reset	R/W
IF0L	TMIF50	CSIIF3	KSIF	TMIF11	TMIF10	PIF1	PIF0	WDTIF	FFE0H	00H	R/W

	7	6	5	4	3	2	<1>	<0>			
IF0H	0	0	0	0	0	0	ADIF	TMIF51	FFE1H	00H	R/W

xxIF	Interrupt Request Flag
0	No interrupt request signal
1	Interrupt request signal is generated: Interrupt request state

Cautions 1. WDTIF flag is R/W enabled only when a watchdog timer is used as an interval timer. If a watchdog timer is used in watchdog timer mode 1, set WDTIF flag to 0.

2. Be sure to set bits 2 through 7 of IF0H to 0.

(2) Interrupt mask flag registers (MK0L and MK0H)

The interrupt mask flag is used to enable/disable the corresponding maskable interrupt service and to set standby clear enable/disable.

MK0L and MK0H are set with a 1-bit or 8-bit memory manipulation instruction. When MK0L and MK0H are used in combination as a 16-bit register MK0, they are set with a 16-bit memory operation instruction.

$\overline{\text{RESET}}$ input sets these registers to FFH.

Figure 12-3. Interrupt Mask Flag Register Format

Symbol	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>	Address	At Reset	R/W
MK0L	TMMK50	CSIMK3	KSMK	TMMK11	TMMK10	PMK1	PMK0	WDTMK	FFE4H	FFH	R/W

Symbol	7	6	5	4	3	2	<1>	<0>	Address	At Reset	R/W
MK0H	1	1	1	1	1	1	ADMK	TMMK51	FFE5H	FFH	R/W

xxMK	Interrupt Serving and Standby Mode Control
0	Interrupt serving enabled, standby mode release enabled
1	Interrupt serving disabled, standby mode release disabled

Cautions 1. If WDTMK flag is read when a watchdog timer is used in watchdog timer mode 1, MK0 value becomes undefined.

2. Because port 0 has a dual function as the external interrupt request input, when the output level is changed by specifying the output mode of the port function, an interrupt request flag is set.

Therefore, 1 should be set in the interrupt mask flag before using the output mode.

3. Be sure to set bits 2 through 7 of MK0H to 1.

(3) Priority specify flag registers (PR0L and PR0H)

The priority specify flag is used to set the corresponding maskable interrupt priority orders.

PR0L and PR0H are set with a 1-bit or 8-bit memory manipulation instruction. When PR0L and PR0H are used in combination as a 16-bit register PR0, they are set with a 16-bit memory operation instruction.

$\overline{\text{RESET}}$ input sets these registers to FFH.

Figure 12-4. Priority Specify Flag Register Format

Symbol	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>	Address	At Reset	R/W
PR0L	TMPR50	CSIPR3	KSPR	TMPR11	TMPR10	PPR1	PPR0	WDTPR	FFE8H	FFH	R/W

	7	6	5	4	3	2	<1>	<0>			
PR0H	1	1	1	1	1	1	ADPR	TMPR51	FFE9H	FFH	R/W

xxPR	Priority Level Selection
0	High priority level
1	Low priority level

Cautions 1. When a watchdog timer is used in watchdog timer mode 1, set 1 in WDTPR flag.

2. Be sure to set bits 2 through 7 of PR0H to 1.

(4) External interrupt rising edge enable register (EGP)

This register specifies whether the valid edges of INTP0 and INTP1 are specified to be the rising edge.

It is set by using a 1-bit or 8-bit memory manipulation instruction.

The value of this register is initialized to 00H by $\overline{\text{RESET}}$ input.

Figure 12-5. External Interrupt Rising Edge Enable Register Format

	7	6	5	4	3	2	1	0	Address	At Reset	R/W
EGP	0	0	0	0	0	0	EGP1	EGP0	FF48H	00H	R/W

EGPn	Enables or disables specification of rising edge of INTPn as valid edge (n = 0 or 1)
0	Disable
1	Enable

(5) External interrupt falling edge enable register (EGN)

This register specifies whether the valid edges of INTP0 and INTP1 are specified to be the falling edge.

It is set by using a 1-bit or 8-bit memory manipulation instruction.

The value of this register is initialized to 00H by $\overline{\text{RESET}}$ input.

Figure 12-6. External Interrupt Falling Edge Enable Register Format

	7	6	5	4	3	2	1	0	Address	At Reset	R/W
EGN	0	0	0	0	0	0	EGN1	EGN0	FF49H	00H	R/W

EGNn	Enables or disables specification of falling edge of INTPn as valid edge (n = 0 or 1)
0	Disable
1	Enable

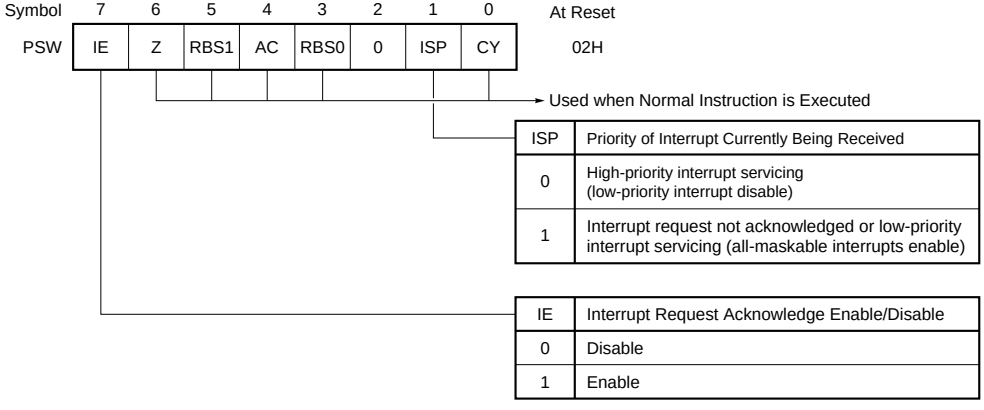
(6) Program status word (PSW)

The program status word is a register to hold the instruction execution result and the current status for interrupt request. The IE flag to set maskable interrupt enable/disable and the ISP flag to control multiple interrupt processing are mapped.

Besides 8-bit unit read/write, this register can carry out operations with a bit manipulation instruction and dedicated instructions (EI and DI). When a vectored interrupt request is acknowledged and the BRK instruction is executed, the contents of the PSW is automatically saved into a stack and the IE flag is reset to (0). If a maskable interrupt request is acknowledged, the contents of the priority specify flag of the acknowledged interrupt are transferred to the ISP flag. The contents of the PSW is also saved into the stack with the PUSH PSW instruction. It is reset from the stack with the RETI, RETB, and POP PSW instructions.

$\overline{\text{RESET}}$ input sets PSW to 02H.

Figure 12-7. Program Status Word Configuration



ISP	Priority of Interrupt Currently Being Received
0	High-priority interrupt servicing (low-priority interrupt disable)
1	Interrupt request not acknowledged or low-priority interrupt servicing (all-maskable interrupts enable)

IE	Interrupt Request Acknowledge Enable/Disable
0	Disable
1	Enable

12.4 Interrupt Servicing Operations

12.4.1 Non-maskable interrupt request acknowledge operation

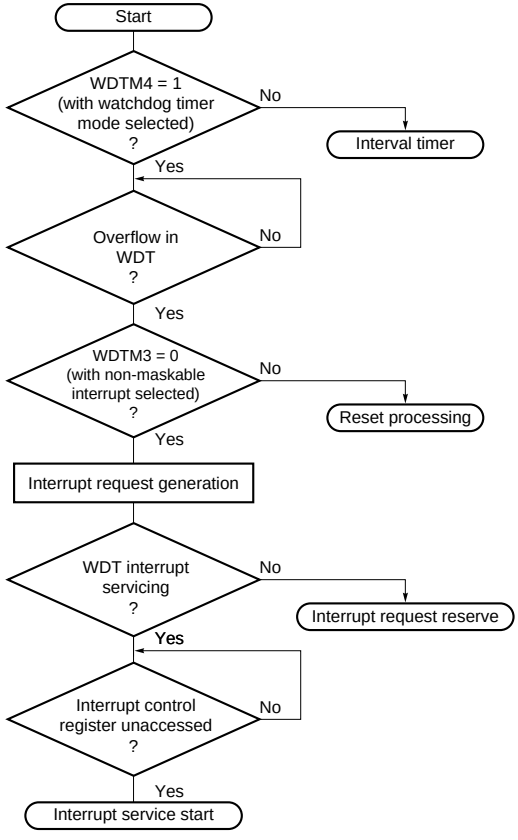
A non-maskable interrupt request is unconditionally acknowledged even if in an interrupt acknowledge disable state. It does not undergo interrupt priority control and has highest priority over all other interrupts.

If a non-maskable interrupt request is acknowledged, the contents are saved in the stacks, PSW and PC, in that order, the IE and ISP flags are reset to 0, and the vector table contents are loaded into PC and branched.

A new non-maskable interrupt request generated during execution of a non-maskable interrupt servicing program is acknowledged after the current execution of the non-maskable interrupt servicing program is terminated (following RETI instruction execution) and one main routine instruction is executed. If a new non-maskable interrupt request is generated twice or more during non-maskable interrupt service program execution, only one non-maskable interrupt request is acknowledged after termination of the non-maskable interrupt service program execution.

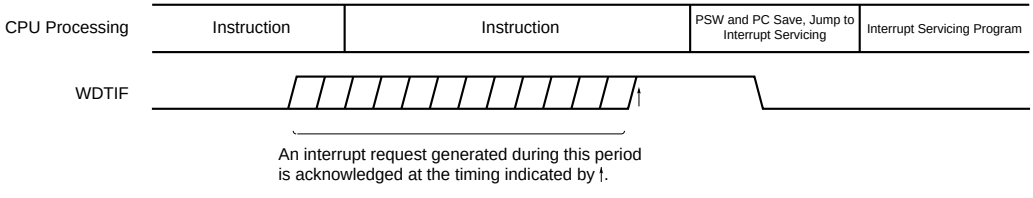
Figure 12-8 shows the flowchart illustrating how the non-maskable interrupt request occurs and is acknowledged. Figure 12-9 shows the acknowledge timing of the non-maskable interrupt. Figure 12-10 shows acknowledge operation of multiple non-maskable interrupts.

Figure 12-8. Flowchart of Occurrence and Acknowledge Non-Maskable Interrupt



WDTM : Watchdog timer mode register
WDT : Watchdog timer

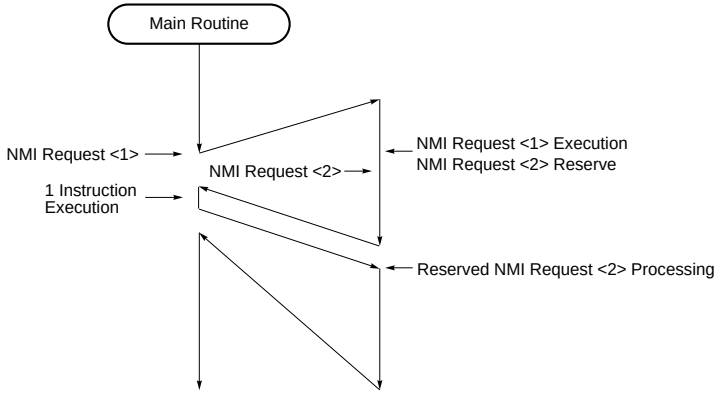
Figure 12-9. Non-Maskable Interrupt Request Acknowledge Timing



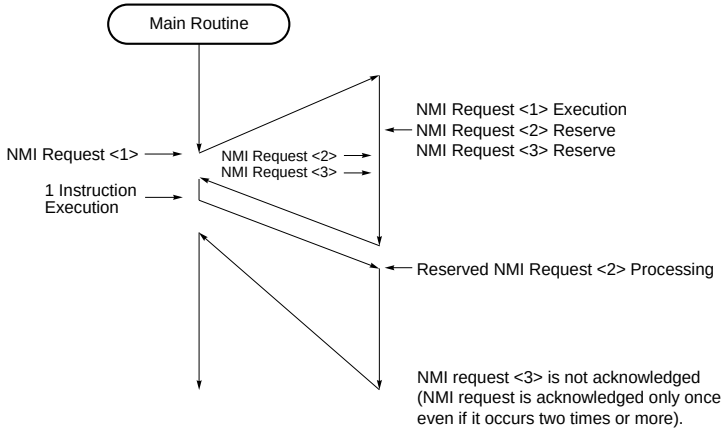
WDTIF : Watchdog timer interrupt request flag

Figure 12-10. Non-Maskable Interrupt Request Acknowledge Operation

(a) If a new non-maskable interrupt request is generated during non-maskable interrupt servicing program execution



(b) If two non-maskable interrupt requests are generated during non-maskable interrupt servicing program execution



12.4.2 Maskable interrupt request acknowledge operation

A maskable interrupt request becomes acknowledgeable when an interrupt request flag is set to 1 and the interrupt mask (MK) flag is cleared to 0. A vectored interrupt request is acknowledged in an interrupt enable state (with IE flag set to 1). However, a low-priority interrupt request is not acknowledged during high-priority interrupt service (with ISP flag reset to 0).

Wait times from maskable interrupt request generation to interrupt servicing are shown in Table 12-3.

For the timing to acknowledge an interrupt request, refer to Figures 12-12 and 12-13.

Table 12-3. Times from Maskable Interrupt Request Generation to Interrupt Service

	Minimum Time	Maximum Time[Note]
When xxPR = 0	7 clocks	32 clocks
When xxPR = 1	8 clocks	33 clocks

Note If an interrupt request is generated just before a divide instruction, the wait time is maximized.

Remark 1 clock : $\frac{1}{f_{CPU}}$ (f_{CPU}: CPU clock)

If two or more maskable interrupt requests are generated simultaneously, the request specified for higher priority with the priority specify flag is acknowledged first. If two or more requests are assigned the same priority by the interrupt priority specify flag, the one with the higher default priority is acknowledged first.

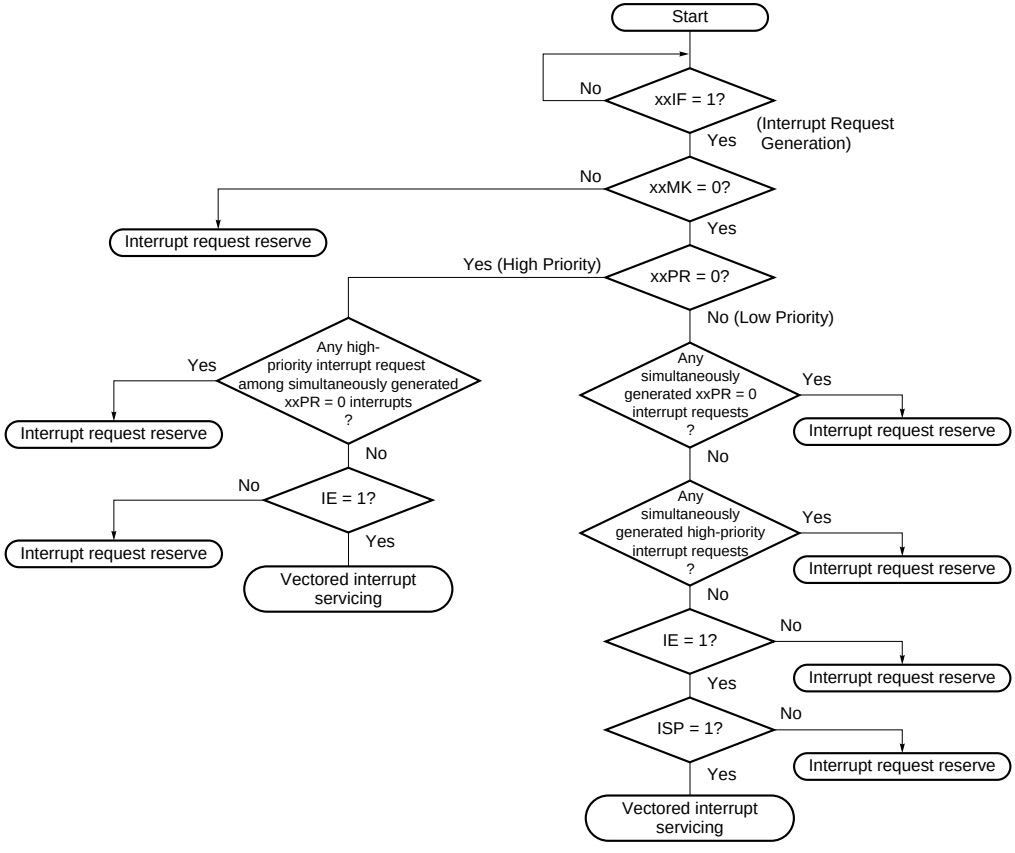
Any reserved interrupt requests are acknowledged when they become acknowledgeable.

Figure 12-11 shows interrupt request acknowledge algorithms.

If a maskable interrupt request is acknowledged, the contents are saved in the stacks, in the order of program status word (PSW), program counter (PC), the IE flag is reset to 0, and the acknowledged interrupt request priority specify flag contents are transferred to the ISP flag. Further, the vector table data determined for each interrupt request is loaded into PC and branched.

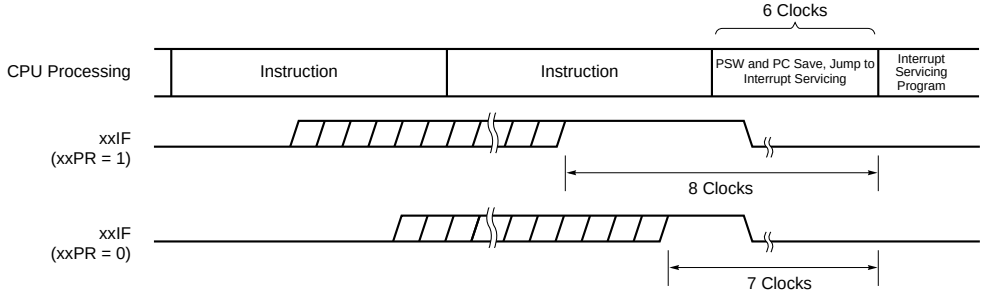
Restore from the interrupt is possible with the RETI instruction.

Figure 12-11. Interrupt Request Acknowledge Processing Algorithm



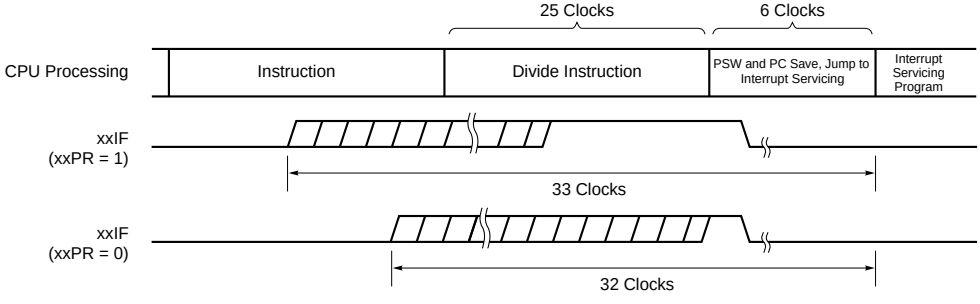
xxIF : Interrupt request flag
xxMK : Interrupt mask flag
xxPR : Priority specify flag
IE : Flag that controls maskable interrupt request acknowledge (1 = enable, 0 = disable)
ISP : Flag indicating priority of interrupt currently processed (0 = interrupt with high priority is processed. 1 = interrupt request is not acknowledged or interrupt with low priority processed).

Figure 12-12. Interrupt Request Acknowledge Timing (Minimum Time)



Remark 1 clock : $\frac{1}{f_{CPU}}$ (f_{CPU}: CPU clock)

Figure 12-13. Interrupt Request Acknowledge Timing (Maximum Time)



Remark 1 clock : $\frac{1}{f_{CPU}}$ (f_{CPU}: CPU clock)

12.4.3 Software interrupt request acknowledge operation

A software interrupt request is acknowledged by BRK instruction execution. Software interrupt cannot be disabled.

If a software interrupt request is acknowledged, it is saved in the stacks, program status word (PSW), program counter (PC), in that order, the IE flag is reset to 0 and the contents of the vector tables (003EH and 003FH) are loaded into PC and branched.

Return from the software interrupt is possible with the RETB instruction.

Caution Do not use the RETI instruction for returning from the software interrupt.

12.4.4 Multiple interrupt servicing

Acknowledging another interrupt while one interrupt is processed is called multiple interrupts.

A multiple interrupt is not generated unless acknowledge of the interrupt request is enabled (IE = 1) (except the non-maskable interrupt). When an interrupt request is acknowledged, the other interrupt requests are disabled (IE = 0). To enable a multiple interrupt, therefore, the IE flag must be set to 1 by executing the EI instruction during interrupt servicing and the interrupt must be enabled. Even in the EI status, a multiple interrupt may not be enabled. In such a case, it is controlled according to the priority of the interrupt. An interrupt has two types of priorities: default priority and programmable priority. The multiple interrupt is controlled by the programmable priority.

In the EI status, if an interrupt request having the same as or higher priority than that of the interrupt currently processed is generated, and it is acknowledged as the multiple interrupt. If an interrupt request with a priority lower than that of the interrupt currently processed is generated, the multiple interrupt is not acknowledged.

If an interrupt is disabled, or if a multiple interrupt is not acknowledged because it has a low priority, the interrupt is kept pending. After the processing of the current interrupt has been completed, and after one instruction of the main processing has been executed, the pending interrupt is acknowledged.

Multiple interrupts are not acknowledged while the non-maskable interrupt is processed.

Table 12-4 shows interrupt requests enabled for multiple interrupts. Figure 12-14 shows multiple interrupt examples.

Table 12-4. Interrupt Request Enabled for Multiple Interrupt during Interrupt Servicing

Multiple Interrupt Request / Interrupt being Serviced		Non-maskable Interrupt Request	Maskable Interrupt Request			
			xx PR = 0		xx PR = 1	
			IE = 1	IE = 0	IE = 1	IE = 0
Non-maskable interrupt		×	×	×	×	×
Maskable interrupt	ISP = 0	○	○	×	×	×
	ISP = 1	○	○	×	○	×
Software interrupt		○	○	×	○	×

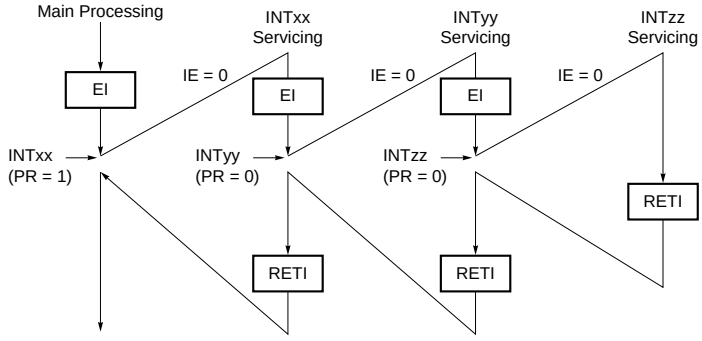
Remarks 1. ○ : Multiple interrupt enable
× : Multiple interrupt disable

2. ISP and IE are flags included in PSW.
ISP = 0 : High-priority interrupt servicing
ISP = 1 : Interrupt request is not acknowledged or low-priority interrupt servicing
IE = 0 : Interrupt request acknowledge disabled
IE = 1 : Interrupt request acknowledge enabled

3. xxPR is a flag included in PR0L and PR0H.
xxPR = 0 : High-priority flag
xxPR = 1 : Low-priority flag

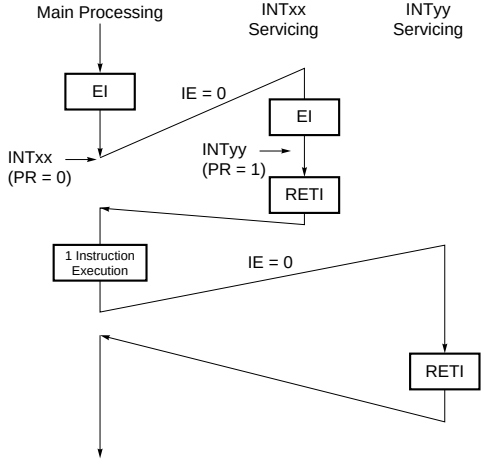
Figure 12-14. Multiple Interrupt Examples

Example 1. Two multiple interrupts are generated.



Two interrupt requests INTyy and INTzz are acknowledged and multiple interrupts are generated while interrupt INTxx request is processed. Before each interrupt request is acknowledged, the EI instruction is always issued and the interrupt request is enabled.

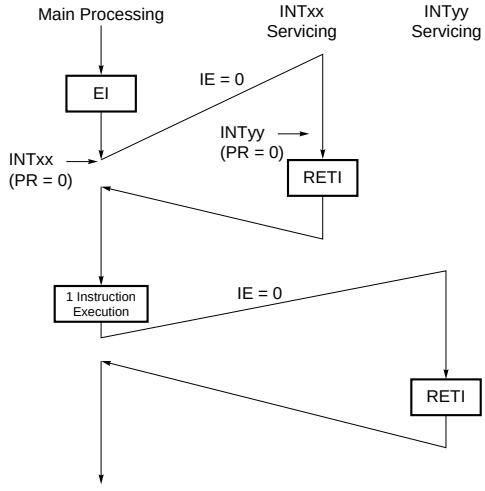
Example 2. Multiple interrupt is not generated because of its priority.



INTyy that occurs while INTxx is processed is not acknowledged and a multiple interrupt is not generated, because the priority of INTyy is lower than that of INTxx. INTyy is reserved and is acknowledged after one instruction of the main processing has been executed.

PR = 0 : High-priority interrupt
PR = 1 : Low-priority interrupt
IE = 0 : Interrupt acknowledge disabled

Example 3. Multiple interrupt is not generated because an interrupt is not processed.



Because interrupts are not enabled (the EI instruction is not issued) in interrupt processing INTxx, interrupt request INTyy is not accepted, and therefore, the interrupt is not nested. INTyy request is kept pending, and is accepted after main processing 1 instruction has been executed.

PR = 0 : High priority level
IE = 0 : Disables accepting interrupt request

12.4.5 Interrupt request reserve

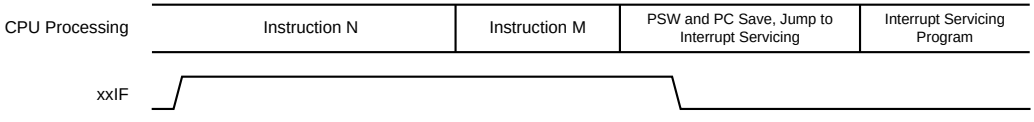
Some instructions keep the acceptance of an interrupt request, if one occurs, pending until execution of the next instruction is completed. These instructions (that keep interrupt requests pending) are listed below.

- MOV PSW, #byte
- MOV A, PSW
- MOV PSW, A
- MOV1 PSW.bit, CY
- MOV1 CY, PSW.bit
- AND1 CY, PSW.bit
- OR1 CY, PSW.bit
- XOR1 CY, PSW.bit
- SET1 PSW.bit
- CLR1 PSW.bit
- RETB
- RETI
- PUSH PSW
- POP PSW
- BT PSW.bit, $addr16
- BF PSW.bit, $addr16
- BTCLR PSW.bit, $addr16
- EI
- DI
- Manipulation instructions for IF0L, IF0H, MK0L, MK0H, PR0L, PR0H, and INTM0 registers

Caution The BRK instruction is not included in the above list of instructions. The software interrupt that is started by execution of the BRK instruction, however, clears the IE flag to 0. Therefore, even if a maskable interrupt request is generated while the BRK instruction is being executed, the interrupt request is not accepted. However, a non-maskable interrupt request is accepted.

Figure 12-15 shows the timing at which an interrupt request is reserved.

Figure 12-15. Interrupt Request Reserve



Remarks 1. Instruction N : Interrupt request reserve instruction

2. Instruction M : Instruction except interrupt request reserve instructions

3. Operation of xxIF (interrupt request) is not effected by xxPR (priority level) value.

CHAPTER 13 STANDBY FUNCTION

13.1 Standby Function and Configuration

13.1.1 Standby function

The standby function is intended to decrease the power consumption of the system. The following two modes are available.

(1) HALT mode

HALT instruction execution sets the HALT mode. The HALT mode is intended to stop the CPU operation clock. The system clock oscillator continues oscillating. In this mode, the current consumption cannot be decreased as in the STOP mode. The HALT mode is valid to restart immediately upon interrupt request and to carry out intermittent operations like clock operations.

(2) STOP mode

STOP instruction execution sets the STOP mode. In the STOP mode, the main system clock oscillator stops and the whole system stops. The CPU current consumption can be considerably decreased.

Data memory low-voltage hold (down to V_{DD} = 2 V) is possible. Thus, the STOP mode is effective to hold data memory contents with ultra-low current consumption. Because this mode can be cleared upon interrupt request, it enables intermittent operations to be carried out.

However, because a wait time is necessary to secure the oscillation stabilization time after the STOP mode is cleared, select the HALT mode if it is necessary to start processing immediately upon interrupt request.

In any mode, all the contents of the register, flag and data memory just before standby mode setting are held. The input/output port output latch and output buffer statuses are also held.

Cautions 1. When proceeding to the STOP mode, be sure to stop the peripheral hardware operation and execute the STOP instruction.

2. In order to decrease the power consumption in the A/D converter, clear bit 7 (CS0) in the A/D converter mode register (ADM0) to 0 and stop A/D conversion operation before executing a HALT or STOP instruction.

13.1.2 Standby function control register

The wait time after the STOP mode is cleared upon interrupt request until the oscillation stabilizes is controlled with the oscillation stabilization time select register (OSTS).

OSTS is set with a 1-bit/8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets OSTS to 04H. Therefore, when the STOP mode is cleared with $\overline{\text{RESET}}$ input, the time until it is cleared is 2^{16}/fx. ★

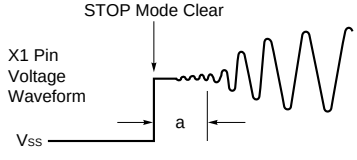
Figure 13-1. Oscillation Stabilization Time Select Register Format

Symbol	7	6	5	4	3	2	1	0	Address	At Reset	R/W
OSTS	0	0	0	0	0	OSTS2	OSTS1	OSTS0	FFFAH	04H	R/W

★

OSTS2	OSTS1	OSTS0	Selection of oscillation stabilization time at STOP mode clear
0	0	0	2^{11}/fx (410 μs)
0	0	1	2^{13}/fx (1.64 ms)
0	1	0	2^{14}/fx (3.28 ms)
0	1	1	2^{15}/fx (6.55 ms)
1	0	0	2^{16}/fx (13.1 ms)
Others			Setting prohibited

Caution The wait time after STOP mode clear does not include the time (see "a" below) from STOP mode clear to clock oscillation start, regardless of clearance by $\overline{\text{RESET}}$ input or by interrupt request generation.



Remarks 1. fx: Main system clock frequency

2. Values in parentheses apply to operation with fx = 5.0 MHz

13.2 Standby Function Operations

13.2.1 HALT mode

(1) HALT mode set and operating status

The HALT mode is set by executing the HALT instruction.

The operating status in the HALT mode is described below.

Table 13-1. HALT Mode Operating Status

Item	Operating status
Clock Generator	Oscillation enabled. Clock supply to the CPU stops.
CPU	Operation stop.
Port (output latch)	Status before HALT instruction execution is held.
8-bit remote control timer	Operation enabled.
8-bit PWM timer	
Watchdog timer	
A/D converter	
Serial interface	
FIP controller/driver	Operation disabled.
External interrupt request	Operation enabled.

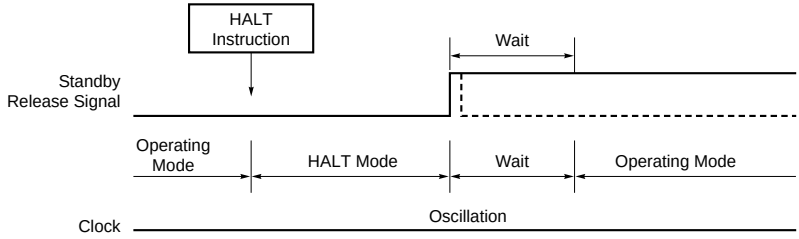
(2) HALT mode clear

The HALT mode can be cleared with the following three types of sources.

(a) Clear upon unmasked interrupt request

An unmasked interrupt request is used to clear the HALT mode. If interrupt request acknowledge is enabled, vectored interrupt service is carried out. If disabled, the next address instruction is executed.

Figure 13-2. HALT Mode Clear upon Interrupt Request Generation



Remarks 1. The broken line indicates the case when the interrupt request which has cleared the standby status is acknowledged.

2. Wait time will be as follows:

- When vectored interrupt service is carried out : 8 to 9 clocks
- When vectored interrupt service is not carried out : 2 to 3 clocks

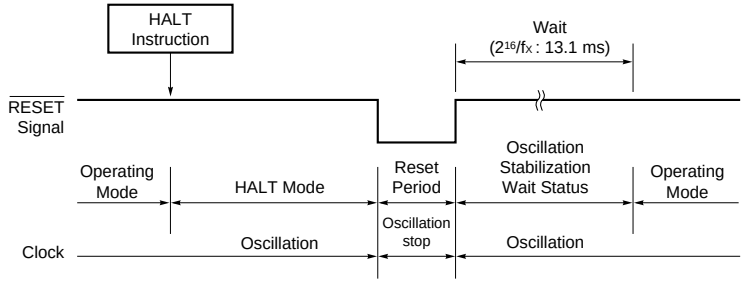
(b) Clear upon non-maskable interrupt request

The HALT mode is cleared and vectored interrupt service is carried out whether interrupt request acknowledge is enabled or disabled.

(c) Clear upon $\overline{\text{RESET}}$ input

As is the case with normal reset operation, a program is executed after branch to the reset vector address.

Figure 13-3. HALT Mode Release by $\overline{\text{RESET}}$ Input



Remarks 1. fx: Main system clock oscillation frequency

2. Figures in parentheses apply to operation with fx = 5.0 MHz.

Table 13-2. Operation after HALT Mode Release

Release Source	MKxx	PRxx	IE	ISP	Operation
Maskable interrupt request	0	0	0	×	Next address instruction execution
	0	0	1	×	Interrupt service execution
	0	1	0	1	Next address instruction execution
	0	1	×	0	
	0	1	1	1	Interrupt service execution
	1	×	×	×	HALT mode hold
Non-maskable interrupt request	—	—	×	×	Interrupt service execution
$\overline{\text{RESET}}$ input	—	—	×	×	Reset processing

×: Don't care

13.2.2 STOP mode

(1) STOP mode set and operating status

The STOP mode is set by executing the STOP instruction.

Cautions 1. When the STOP mode is set, X2 pin is internally pulled-up to V_{DD1} to suppress the leakage at the crystal oscillator.
Thus, do not use the STOP mode in a system where an external clock is used for the main system clock.

2. Because the interrupt request signal is used to clear the standby mode, if there is an interrupt source with the interrupt request flag set and the interrupt mask flag reset, the standby mode is immediately cleared if set. Thus, the STOP mode is reset to the HALT mode immediately after execution of the STOP instruction. After the wait set using the oscillation stabilization time select register (OSTS), the operating mode is set.

The operating status in the STOP mode is described below.

Table 13-3. STOP Mode Operating Status

Item	Operating status
Clock Generator	Oscillation stop.
CPU	Operation stop.
Output port (output latch)	Status before STOP mode setting is held.
8-bit remote control timer	Operation stop.
8-bit PWM timer	
Watchdog timer	
A/D converter	
Serial interface	Operation enabled only when external input clock is selected as serial clock.
FIP controller/driver	Operation disabled.
External interrupt request	Operation enabled.

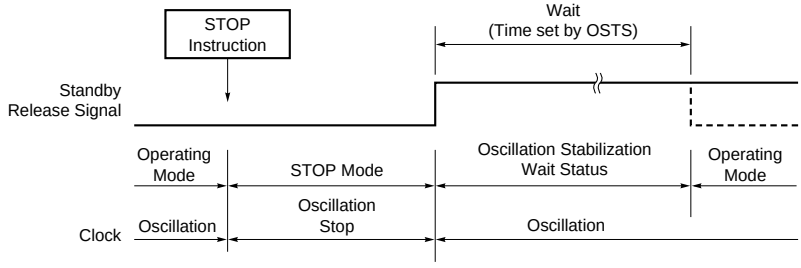
(2) STOP mode release

The STOP mode can be cleared with the following two types of sources.

(a) Release by unmasked interrupt request

An unmasked interrupt request is used to release the STOP mode. If interrupt request acknowledge is enabled after the lapse of oscillation stabilization time, vectored interrupt service is carried out. If interrupt request acknowledge is disabled, the next address instruction is executed.

Figure 13-4. STOP Mode Release by Interrupt Request Generation

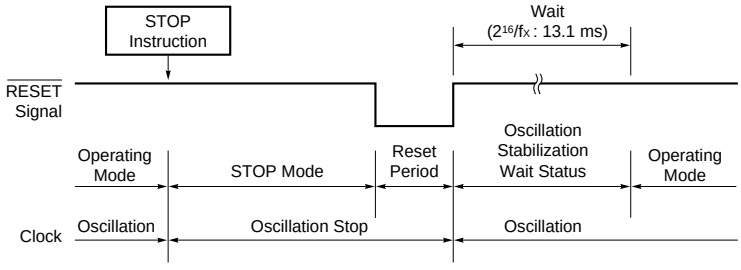


Remark The broken line indicates the case when the interrupt request which has cleared the standby status is acknowledged.

(b) Release by $\overline{\text{RESET}}$ input

The STOP mode is cleared and after the lapse of oscillation stabilization time, reset operation is carried out.

Figure 13-5. Release by STOP Mode $\overline{\text{RESET}}$ Input



Remarks 1. fx: Main system clock oscillation frequency

2. Figures in parentheses apply to operation with fx = 5.0 MHz.

Table 13-4. Operation after STOP Mode Release

Release Source	MKxx	PRxx	IE	ISP	Operation
Maskable interrupt request	0	0	0	×	Next address instruction execution
	0	0	1	×	Interrupt service execution
	0	1	0	1	Next address instruction execution
	0	1	×	0	
	0	1	1	1	Interrupt service execution
	1	×	×	×	STOP mode hold
$\overline{\text{RESET}}$ input	—	—	×	×	Reset processing

×: Don't care

CHAPTER 14 RESET FUNCTION

14.1 Reset Function

The following two operations are available to generate the reset signal.

(1) External reset input with $\overline{\text{RESET}}$ pin
(2) Internal reset by watchdog timer overrun time detection

External reset and internal reset have no functional differences. In both cases, program execution starts at the address at 0000H and 0001H by $\overline{\text{RESET}}$ input.

When a low level is input to the $\overline{\text{RESET}}$ pin or the watchdog timer overflows, a reset is applied and each hardware is set to the status as shown in Table 14-1. Each pin has high impedance during reset input or during oscillation stabilization time just after reset clear.

When a high level is input to the $\overline{\text{RESET}}$ pin, the reset is cleared and program execution starts after the lapse of ★ oscillation stabilization time ($2^{16}/f_X$). The reset applied by watchdog timer overflow is automatically cleared after a ★ reset and program execution starts after the lapse of oscillation stabilization time ($2^{16}/f_X$) (see **Figures 14-2** to **14-4**).

Cautions 1. For an external reset, input a low level for 10 μs or more to the $\overline{\text{RESET}}$ pin.
2. When the STOP mode is cleared by reset, the STOP mode contents are held during reset input. However, the port pin becomes high-impedance.

Figure 14-1. Block Diagram of Reset Function

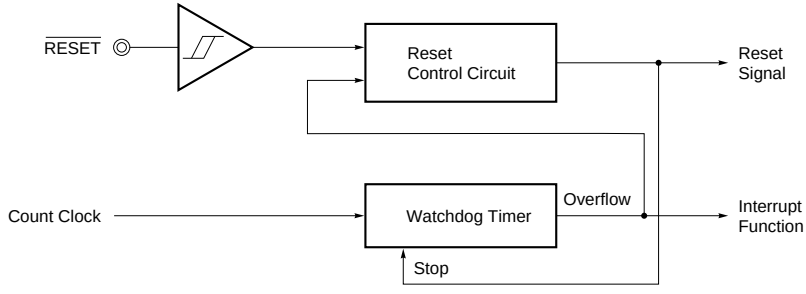


Figure 14-2. Timing of Reset Input by $\overline{\text{RESET}}$ Input

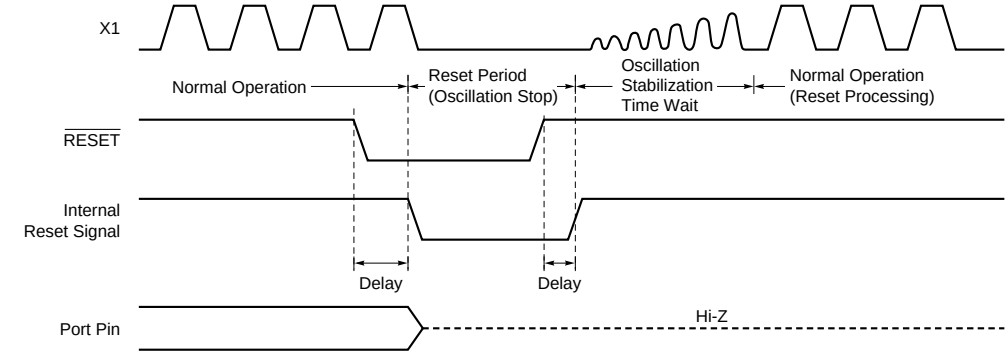


Figure 14-3. Timing of Reset due to Watchdog Timer Overflow

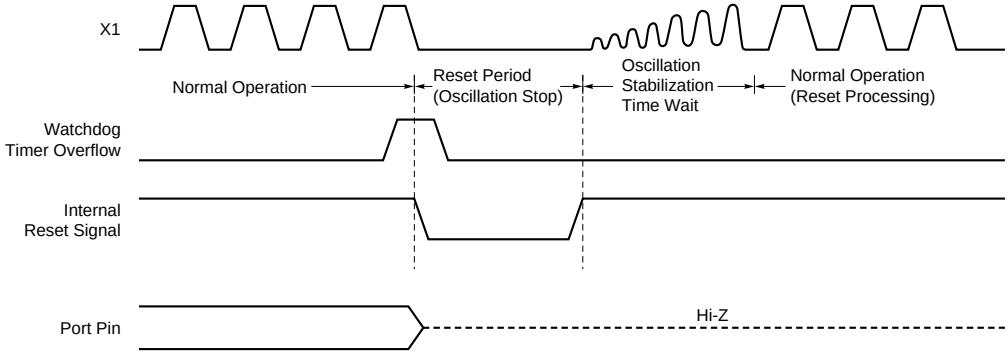


Figure 14-4. Timing of Reset in STOP Mode by $\overline{\text{RESET}}$ Input

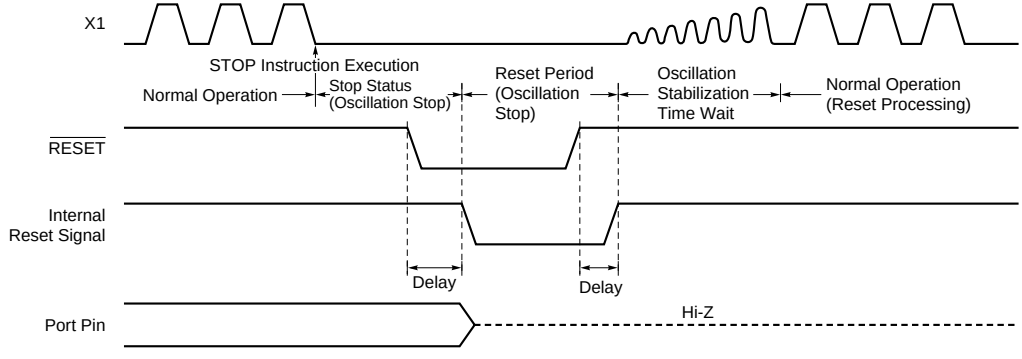


Table 14-1. Hardware Status after Reset

Hardware		Status after Reset
Program counter (PC)Note 1		Contents of reset vector table (0000H, 0001H) are set
Stack pointer (SP)		Undefined
Program status word (PSW)		02H
RAM	Data memory	UndefinedNote 2
	General-purpose register	UndefinedNote 2
Port (output latch)	Ports 0-2, ports 4-10 (P0-P2, P4-P10)	00H
Port read	(PLR7-PLR9)	Undefined
Port mode register	(PM0, PM2, PM4-PM6)	FFH
Pull-up resistor option register (PU0, PU2, PU4)		00H
Processor clock control register (PCC)		04H
Memory size select register (IMS)		CCHNote 3
★ Internal expansion RAM size select register (IXS)		0CHNote 4
Oscillation stabilization time select register (OSTS)		04H
8-bit remote control timer	Capture registers (CP10, CP11)	00H
	Mode control register (TMC1)	00H
8-bit PWM timer	Timer registers (TM50, TM51)	00H
	Compare registers (CR50, CR51)	Undefined
	Mode control registers (TMC50, TMC51)	04H
	Clock select registers (TCL50, TCL51)	00H
Watchdog timer	Clock select register (WDCS)	00H
	Mode register (WDTM)	00H
A/D converter	Conversion result register (ADCRH0)	Undefined
	Mode register (ADM0)	00H
	Analog input channel specification register (ADS0)	00H
Serial interface	Shift register (SIO3)	Undefined
	Mode register (CSIM3)	00H
FIP controller/driver	Display mode register 0 (DSPM0)	10H
	Display mode register 1 (DSPM1)	01H
	Display mode register 2 (DSPM2)	00H
Interrupt	External interrupt rising edge enable register (EGP)	00H
	External interrupt falling edge enable register (EGN)	00H
	Request flag registers (IF0L, IF0H)	00H
	Mask flag registers (MK0L, MK0H)	FFH
	Priority specification flag registers (PR0L, PR0H)	FFH

Notes 1. Of the hardware units, only the contents of the PC are undefined during reset input or oscillation stabilization time wait. The statuses of the other hardware units are the same as those after reset.

2. The status after reset is retained in the standby mode.

3. Be sure to set CFH when using the μPD780228.

4. After reset, be sure to set this bit to 0BH.

[MEMO]

CHAPTER 15 μPD78F0228

The μPD78F0228 has a flash memory to which a program can be written or whose contents can be erased with the device mounted on the PC board of the target system. Table 15-1 shows the differences between the flash memory model (μPD78F0228) and mask ROM models (μPD780226 and 780228).

Table 15-1. Differences between μPD78F0228 and Mask ROM Models

Item	μPD78F0228	Mask ROM Models
Internal ROM structure	Flash memory	Mask ROM
Internal ROM capacity	60K bytes	μPD780226: 48K bytes μPD780228: 60K bytes
Changing internal ROM capacity by memory size select register (IMS)	Possible[Note]	Impossible
★ Internal expansion RAM size select register (IXS)	Provided	Not provided
IC pin	Not provided	Provided
V_{PP} pin	Provided	Not provided
Mask option of connecting pull-up resistors to P50 through P57 and P60 through P67	Not provided	Provided
Mask option of connecting pull-down resistors to P70 through P77, P80 through P87, P90 through P97, and P100 through P107	Not provided	Provided

Note The flash memory capacity is set to 60K bytes by $\overline{\text{RESET}}$ input.

Caution The flash memory model and mask ROM model differ in terms of noise immunity. When replacing a flash memory model with a mask ROM model in the course of experimental production to mass production, make a thorough evaluation with a CS model (not ES model) of the mask ROM model.

15.1 Memory Size Select Register

The μPD78F0228 can select the internal memory capacities by using the memory size select register (IMS). By setting IMS, the memory mapping of the μPD78F0228 can be made the same as that of a mask ROM model with different internal memory capacities.

To make the memory map of the μPD78F0228 the same as that of a mask ROM model, set IMS as shown in Table 15-2.

IMS is set by using an 8-bit memory manipulation instruction.

The value of this register is set to CCH by $\overline{\text{RESET}}$ input.

Figure 15-1. Memory Size Select Register Format

Symbol	7	6	5	4	3	2	1	0	Address	At Reset	R/W
IMS	RAM2	RAM1	RAM0	0	ROM3	ROM2	ROM1	ROM0	FFF0H	CCH	R/W

RAM2	RAM1	RAM0	Selects internal high-speed RAM capacity
1	1	0	1024 bytes
Others			Setting prohibited

ROM3	ROM2	ROM1	ROM0	Selects internal ROM capacity
1	1	0	0	48K bytes
1	1	1	1	60K bytes
Others				Setting prohibited

Caution When using a mask ROM model, do not set a value other than those shown in Table 15-2 to IMS.

Table 15-2. Set Value of Memory Size Select Register

Part Number	Set Value of IMS
μPD780226	CCH
μPD780228	CFH

★ 15.2 Internal Expansion RAM Size Select Register

The internal expansion RAM size select register (IXS) specifies the internal expansion RAM capacity.

IXS is set by using an 8-bit memory manipulation instruction.

The value of this register is set to 0CH by $\overline{\text{RESET}}$ input.

Caution Be sure to set 0BH to IXS in the initial settings of the program. Because IXS is set to 0CH at RESET, be sure to set the register to 0BH again after reset.

Figure 15-2. Internal Expansion RAM Size Select Register Format

Symbol	7	6	5	4	3	2	1	0	Address	At Reset	R/W
IXS	0	0	0	IXRAM4	IXRAM3	IXRAM2	IXRAM1	IXRAM0	FFF4H	0CH	R/W

IXRAM4	IXRAM3	IXRAM2	IXRAM1	IXRAM0	Selects internal expansion RAM capacity
0	1	0	1	1	512 bytes
Others					Setting prohibited

IXS is not provided on the μPD780226 and 780228.

However, the operation is not affected even if an instruction that writes data to IXS is executed on the μPD780226 and 780228.

15.3 Flash Memory Programming

The flash memory can be written with the device mounted on the target system (on-board). To write the flash ★ memory, connect a dedicated flash writer (Flashpro II: part number FL-PR2) to the host machine and target system.

The flash memory can be also written on a flash memory writing adapter connected to the Flashpro II.

Remark Flashpro II is a product of Naito Densei Machida Mfg. Co., Ltd.

15.3.1 Selecting communication mode

The flash memory is written by using the Flashpro II and by means of serial communication. Select one of the communication modes listed in Table 15-3 to write the flash memory. To select a communication mode, use the format shown in Figure 15-3. Each communication mode is selected by the number of V_{P-P} pulses shown in Table 15-3.

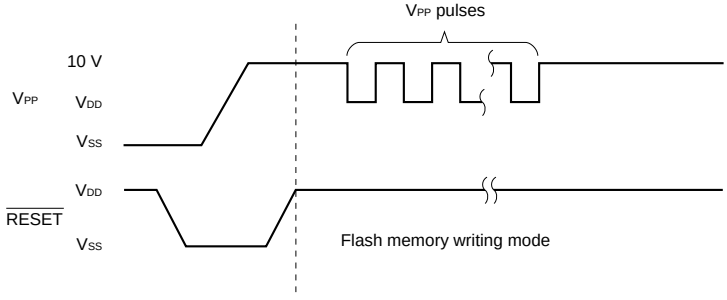
Table 15-3. Communication Modes

Communication Mode	Number of Channels	Pin Used	Number of V_{P-P} Pulses
3-wire serial I/O	1	$\overline{SCK}$/P20 SO/P21 SI/P22	0
Pseudo 3-wire serial I/O**Note**	1	P40 (serial clock I/O) P41 (serial data output) P42 (serial data input)	12

Note Execute serial transfer by controlling the ports by software.

Caution Be sure to select a communication mode with the number of V_{P-P} pulses shown in Table 15-3.

Figure 15-3. Communication Mode Select Format



15.3.2 Function of flash memory programming

By transmitting/receiving commands/data in the selected communication mode, operations of the flash memory such as writing are performed. Table 15-14 lists the major functions of the flash memory programming.

Table 15-4. Major Functions of Flash Memory Programming

Function	Description
Reset	Used to stop writing or detect communication synchronization
Batch verify	Compares all memory contents with input data
Batch erase	Erases all memory contents
Batch blank check	Checks erased status of entire memory
High-speed write	Writes flash memory based on write start address and number of written data (number of bytes)
Successive write	Successively writes based on information input in high-speed write mode
Status	Used to check current operation mode and end of operation
Oscillation frequency setting	Inputs frequency information on oscillator
Erase time setting	Inputs erase time of memory
Silicon signature read	Outputs device name, memory capacity, and block information of device

15.3.3 Connection of Flashpro II

Connection between the Flashpro II and μPD78F0228 differs depending on the selected communication mode. Figures 15-4 and 15-5 show the connections in the respective mode.

Figure 15-4. Connection of Flashpro II in 3-Wire Serial I/O Mode

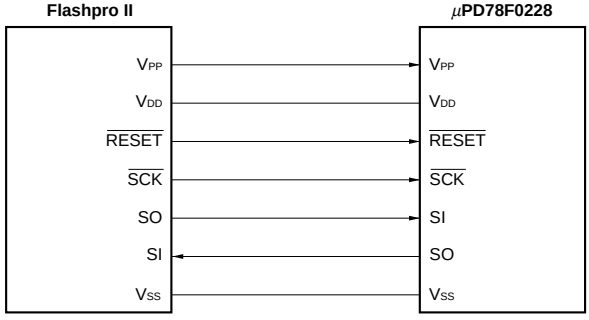
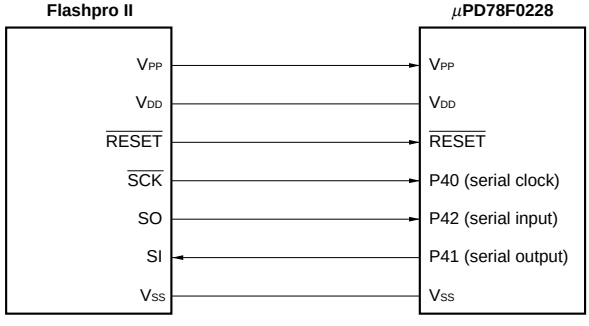


Figure 15-5. Connection of Flashpro II in Pseudo 3-Wire Serial I/O Mode



CHAPTER 16 INSTRUCTION SET

The instruction set for the μPD780228 subseries is described in the following pages. For the details of operations and mnemonics (instruction codes) of each instruction, refer to **78K/0 Series User's Manual: Instructions (U12326E)**.

16.1 Legend

16.1.1 Operand identifiers and description methods

Operands are described in the “Operand” column of each instruction in accordance with the description method of the instruction operand identifier (refer to the assembler specifications for detail). When there are two or more description methods, select one of them. Alphabetic letters in capitals and symbols, #, !, $ and [ ] are key words and are described as they are. Symbols have the following meaning.

- # : Immediate data specification
- ! : Absolute address specification
- $: Relative address specification
- [ ] : Indirect address specification

In the case of immediate data, describe an appropriate numeric value or a label. When using a label, be sure to describe the #, !, $ and [ ] symbols.

For operand register identifiers, r and rp, either function names (X, A, C, etc.) or absolute names (names in parentheses in the table below, R0, R1, R2, etc.) can be used for description.

Table 16-1. Operand Identifiers and Description Methods

Identifier	Description Method
r	X (R0), A (R1), C (R2), B (R3), E (R4), D (R5), L (R6), H (R7)
rp	AX (RP0), BC (RP1), DE (RP2), HL (RP3)
sfr	Special function register symbol[Note]
sfrp	Special function register symbols (16-bit manipulatable register even addresses only)[Note]
saddr	FE20H to FF1FH Immediate data or labels
saddrp	FE20H to FF1FH Immediate data or labels (even addresses only)
addr16	0000H to FFFFH Immediate data or labels (Only even addresses for 16-bit data transfer instructions)
addr11	0800H to 0FFFH Immediate data or labels
addr5	0040H to 007FH Immediate data or labels (even addresses only)
word	16-bit immediate data or label
byte	8-bit immediate data or label
bit	3-bit immediate data or label
RBn	RB0 to RB3

Note FFD0H to FFDFH are not addressable.

Remark For special-function register symbols, refer to **Table 3-3 Special Function Register List**.

16.1.2 Description of “operation” column

A	:	A register; 8-bit accumulator
X	:	X register
B	:	B register
C	:	C register
D	:	D register
E	:	E register
H	:	H register
L	:	L register
AX	:	AX register pair; 16-bit accumulator
BC	:	BC register pair
DE	:	DE register pair
HL	:	HL register pair
PC	:	Program counter
SP	:	Stack pointer
PSW	:	Program status word
CY	:	Carry flag
AC	:	Auxiliary carry flag
Z	:	Zero flag
RBS	:	Register bank select flag
IE	:	Interrupt request enable flag
NMIS	:	Non-maskable interrupt servicing flag
( )	:	Memory contents indicated by address or register contents in parentheses
X_H, X_L	:	Higher 8 bits and lower 8 bits of 16-bit register
∧	:	Logical product (AND)
∨	:	Logical sum (OR)
∀	:	Exclusive logical sum (exclusive OR)
——	:	Inverted data
addr16	:	16-bit immediate data or label
jdisp8	:	Signed 8-bit data (displacement value)

16.1.3 Description of “flag operation” column

(Blank)	:	Unchanged
0	:	Cleared to 0
1	:	Set to 1
x	:	Set/cleared according to the result
R	:	Previously saved value is restored

16.2 Operation List

Instruction Group	Mnemonic	Operands	Byte	Clock		Operation	Flag
				Note 1	Note 2		Z AC CY
8-bit data transfer	MOV	r, #byte	2	4	—	r ← byte	
		saddr, #byte	3	6	7	(saddr) ← byte	
		sfr, #byte	3	—	7	sfr ← byte	
		A, r Note 3	1	2	—	A ← r	
		r, A Note 3	1	2	—	r ← A	
		A, saddr	2	4	5	A ← (saddr)	
		saddr, A	2	4	5	(saddr) ← A	
		A, sfr	2	—	5	A ← sfr	
		sfr, A	2	—	5	sfr ← A	
		A, !addr16	3	8	9	A ← (addr16)	
		!addr16, A	3	8	9	(addr16) ← A	
		PSW, #byte	3	—	7	PSW ← byte	× × ×
		A, PSW	2	—	5	A ← PSW	
		PSW, A	2	—	5	PSW ← A	× × ×
		A, [DE]	1	4	5	A ← (DE)	
		[DE], A	1	4	5	(DE) ← A	
		A, [HL]	1	4	5	A ← (HL)	
		[HL], A	1	4	5	(HL) ← A	
		A, [HL+byte]	2	8	9	A ← (HL + byte)	
		[HL+byte], A	2	8	9	(HL + byte) ← A	
		A, [HL+B]	1	6	7	A ← (HL + B)	
		[HL+B], A	1	6	7	(HL + B) ← A	
		A, [HL+C]	1	6	7	A ← (HL + C)	
		[HL+C], A	1	6	7	(HL + C) ← A	
	XCH	A, r Note 3	1	2	—	A ↔ r	
		A, saddr	2	4	6	A ↔ (saddr)	
		A, sfr	2	—	6	A ↔ sfr	
		A, !addr16	3	8	10	A ↔ (addr16)	
		A, [DE]	1	4	6	A ↔ (DE)	
		A, [HL]	1	4	6	A ↔ (HL)	
		A, [HL+byte]	2	8	10	A ↔ (HL + byte)	
		A, [HL+B]	2	8	10	A ↔ (HL + B)	
		A, [HL+C]	2	8	10	A ↔ (HL + C)	

Notes 1. When the internal high-speed RAM area is accessed or instruction with no data access

2. When an area except the internal high-speed RAM area is accessed

3. Except r = A

Remark One instruction clock cycle is one cycle of the CPU clock (f_{CPU}) selected by the processor clock control register (PCC).

Instruction Group	Mnemonic	Operands	Byte	Clock		Operation	Flag
				Note 1	Note 2		Z AC CY
16-bit data transfer	**MOVW**	rp, #word	3	6	—	rp ← word	
		saddrp, #word	4	8	10	(saddrp) ← word	
		sfrp, #word	4	—	10	sfrp ← word	
		AX, saddrp	2	6	8	AX ← (saddrp)	
		saddrp, AX	2	6	8	(saddrp) ← AX	
		AX, sfrp	2	—	8	AX ← sfrp	
		sfrp, AX	2	—	8	sfrp ← AX	
		AX, rp Note 3	1	4	—	AX ← rp	
		rp, AX Note 3	1	4	—	rp ← AX	
		AX, !addr16	3	10	12	AX ← (addr16)	
		!addr16, AX	3	10	12	(addr16) ← AX	
	XCHW	AX, rp Note 3	1	4	—	AX ↔ rp	
8-bit operation	**ADD**	A, #byte	2	4	—	A, CY ← A + byte	× × ×
		saddr, #byte	3	6	8	(saddr), CY ← (saddr) + byte	× × ×
		A, r Note 4	2	4	—	A, CY ← A + r	× × ×
		r, A	2	4	—	r, CY ← r + A	× × ×
		A, saddr	2	4	5	A, CY ← A + (saddr)	× × ×
		A, !addr16	3	8	9	A, CY ← A + (addr16)	× × ×
		A, [HL]	1	4	5	A, CY ← A + (HL)	× × ×
		A, [HL+byte]	2	8	9	A, CY ← A + (HL + byte)	× × ×
		A, [HL+B]	2	8	9	A, CY ← A + (HL + B)	× × ×
		A, [HL+C]	2	8	9	A, CY ← A + (HL + C)	× × ×
	ADDC	A, #byte	2	4	—	A, CY ← A + byte + CY	× × ×
		saddr, #byte	3	6	8	(saddr), CY ← (saddr) + byte + CY	× × ×
		A, r Note 4	2	4	—	A, CY ← A + r + CY	× × ×
		r, A	2	4	—	r, CY ← r + A + CY	× × ×
		A, saddr	2	4	5	A, CY ← A + (saddr) + CY	× × ×
		A, !addr16	3	8	9	A, CY ← A + (addr16) + CY	× × ×
		A, [HL]	1	4	5	A, CY ← A + (HL) + CY	× × ×
		A, [HL+byte]	2	8	9	A, CY ← A + (HL + byte) + CY	× × ×
		A, [HL+B]	2	8	9	A, CY ← A + (HL + B) + CY	× × ×
		A, [HL+C]	2	8	9	A, CY ← A + (HL + C) + CY	× × ×

Notes 1. When the internal high-speed RAM area is accessed or instruction with no data access

2. When an area except the internal high-speed RAM area is accessed

3. Only when rp = BC, DE or HL

4. Except r = A

Remark One instruction clock cycle is one cycle of the CPU clock (f_{CPU}) selected by the processor clock control register (PCC).

Instruction Group	Mnemonic	Operands	Byte	Clock		Operation	Flag		
				Note 1	Note 2		Z	AC	CY
8-bit operation	SUB	A, #byte	2	4	—	A, CY ← A – byte	×	×	×
		saddr, #byte	3	6	8	(saddr), CY ← (saddr) – byte	×	×	×
		A, r Note 3	2	4	—	A, CY ← A – r	×	×	×
		r, A	2	4	—	r, CY ← r – A	×	×	×
		A, saddr	2	4	5	A, CY ← A – (saddr)	×	×	×
		A, !addr16	3	8	9	A, CY ← A – (addr16)	×	×	×
		A, [HL]	1	4	5	A, CY ← A – (HL)	×	×	×
		A, [HL+byte]	2	8	9	A, CY ← A – (HL + byte)	×	×	×
		A, [HL+B]	2	8	9	A, CY ← A – (HL + B)	×	×	×
		A, [HL+C]	2	8	9	A, CY ← A – (HL + C)	×	×	×
	SUBC	A, #byte	2	4	—	A, CY ← A – byte – CY	×	×	×
		saddr, #byte	3	6	8	(saddr), CY ← (saddr) – byte – CY	×	×	×
		A, r Note 3	2	4	—	A, CY ← A – r – CY	×	×	×
		r, A	2	4	—	r, CY ← r – A – CY	×	×	×
		A, saddr	2	4	5	A, CY ← A – (saddr) – CY	×	×	×
		A, !addr16	3	8	9	A, CY ← A – (addr16) – CY	×	×	×
		A, [HL]	1	4	5	A, CY ← A – (HL) – CY	×	×	×
		A, [HL+byte]	2	8	9	A, CY ← A – (HL + byte) – CY	×	×	×
		A, [HL+B]	2	8	9	A, CY ← A – (HL + B) – CY	×	×	×
		A, [HL+C]	2	8	9	A, CY ← A – (HL + C) – CY	×	×	×
	AND	A, #byte	2	4	—	A ← A ∧ byte	×		
		saddr, #byte	3	6	8	(saddr) ← (saddr) ∧ byte	×		
		A, r Note 3	2	4	—	A ← A ∧ r	×		
		r, A	2	4	—	r ← r ∧ A	×		
		A, saddr	2	4	5	A ← A ∧ (saddr)	×		
		A, !addr16	3	8	9	A ← A ∧ (addr16)	×		
		A, [HL]	1	4	5	A ← A ∧ (HL)	×		
		A, [HL+byte]	2	8	9	A ← A ∧ (HL + byte)	×		
		A, [HL+B]	2	8	9	A ← A ∧ (HL + B)	×		
		A, [HL+C]	2	8	9	A ← A ∧ (HL + C)	×		

Notes 1. When the internal high-speed RAM area is accessed or instruction with no data access

2. When an area except the internal high-speed RAM area is accessed

3. Except r = A

Remark One instruction clock cycle is one cycle of the CPU clock (f_{CPU}) selected by the processor clock control register (PCC).

Instruction Group	Mnemonic	Operands	Byte	Clock		Operation	Flag
				Note 1	Note 2		Z AC CY
8-bit operation	OR	A, #byte	2	4	—	A ← A ∨ byte	×
		saddr, #byte	3	6	8	(saddr) ← (saddr) ∨ byte	×
		A, r Note 3	2	4	—	A ← A ∨ r	×
		r, A	2	4	—	r ← r ∨ A	×
		A, saddr	2	4	5	A ← A ∨ (saddr)	×
		A, !addr16	3	8	9	A ← A ∨ (addr16)	×
		A, [HL]	1	4	5	A ← A ∨ (HL)	×
		A, [HL+byte]	2	8	9	A ← A ∨ (HL + byte)	×
		A, [HL+B]	2	8	9	A ← A ∨ (HL + B)	×
		A, [HL+C]	2	8	9	A ← A ∨ (HL + C)	×
	XOR	A, #byte	2	4	—	A ← A ⊻ byte	×
		saddr, #byte	3	6	8	(saddr) ← (saddr) ∨ byte	×
		A, r Note 3	2	4	—	A ← A ⊻ r	×
		r, A	2	4	—	r ← r ⊻ A	×
		A, saddr	2	4	5	A ← A ⊻ (saddr)	×
		A, !addr16	3	8	9	A ← A ⊻ (addr16)	×
		A, [HL]	1	4	5	A ← A ⊻ (HL)	×
		A, [HL+byte]	2	8	9	A ← A ⊻ (HL + byte)	×
		A, [HL+B]	2	8	9	A ← A ⊻ (HL + B)	×
		A, [HL+C]	2	8	9	A ← A ⊻ (HL + C)	×
	CMP	A, #byte	2	4	—	A – byte	× × ×
		saddr, #byte	3	6	8	(saddr) – byte	× × ×
		A, r Note 3	2	4	—	A – r	× × ×
		r, A	2	4	—	r – A	× × ×
		A, saddr	2	4	5	A – (saddr)	× × ×
		A, !addr16	3	8	9	A – (addr16)	× × ×
		A, [HL]	1	4	5	A – (HL)	× × ×
		A, [HL+byte]	2	8	9	A – (HL + byte)	× × ×
		A, [HL+B]	2	8	9	A – (HL + B)	× × ×
		A, [HL+C]	2	8	9	A – (HL + C)	× × ×

Notes 1. When the internal high-speed RAM area is accessed or instruction with no data access

2. When an area except the internal high-speed RAM area is accessed

3. Except r = A

Remark One instruction clock cycle is one cycle of the CPU clock (f_{CPU}) selected by the processor clock control register (PCC).

Instruction Group	Mnemonic	Operands	Byte	Clock		Operation	Flag
				Note 1	Note 2		Z AC CY
16-bit operation	**ADDW**	AX, #word	3	6	—	AX, CY ← AX + word	× × ×
	SUBW	AX, #word	3	6	—	AX, CY ← AX – word	× × ×
	CMPW	AX, #word	3	6	—	AX – word	× × ×
Multiply/ divide	**MULU**	X	2	16	—	AX ← A × X	
	DIVUW	C	2	25	—	AX (Quotient), C (Remainder) ← AX ÷ C	
Increase/ decrease	**INC**	r	1	2	—	r ← r + 1	× ×
		saddr	2	4	6	(saddr) ← (saddr) + 1	× ×
	DEC	r	1	2	—	r ← r – 1	× ×
		saddr	2	4	6	(saddr) ← (saddr) – 1	× ×
	INCW	rp	1	4	—	rp ← rp + 1	
	DECW	rp	1	4	—	rp ← rp – 1	
Rotation	**ROR**	A, 1	1	2	—	(CY, A_7 ← A_0, A_{m-1} ← A_m) × 1	×
	ROL	A, 1	1	2	—	(CY, A_0 ← A_7, A_{m+1} ← A_m) × 1	×
	RORC	A, 1	1	2	—	(CY ← A_0, A_7 ← CY, A_{m-1} ← A_m) × 1	×
	ROLC	A, 1	1	2	—	(CY ← A_7, A_0 ← CY, A_{m+1} ← A_m) × 1	×
	ROR4	[HL}	2	10	12	A_{3-0} ← $(HL)_{3-0}$, ← $(HL)_{7-4}$ ← A_{3-0}, $(HL)_{3-0}$ ← $(HL)_{7-4}$	
	ROL4	[HL]	2	10	12	A_{3-0} ← $(HL)_{7-4}$, ← $(HL)_{3-0}$ ← A_{3-0}, $(HL)_{7-4}$ ← $(HL)_{3-0}$	
BCD correction	**ADJBA**		2	4	—	Decimal Adjust Accumulator after Addition	× × ×
	ADJBS		2	4	—	Decimal Adjust Accumulator after Subtract	× × ×
Bit manipulation	**MOV1**	CY, saddr.bit	3	6	7	CY ← (saddr.bit)	×
		CY, sfr.bit	3	—	7	CY ← sfr.bit	×
		CY, A.bit	2	4	—	CY ← A.bit	×
		CY, PSW.bit	3	—	7	CY ← PSW.bit	×
		CY, [HL].bit	2	6	7	CY ← (HL).bit	×
		saddr.bit, CY	3	6	8	(saddr.bit) ← CY	
		sfr.bit, CY	3	—	8	sfr.bit ← CY	
		A.bit, CY	2	4	—	A.bit ← CY	
		PSW.bit, CY	3	—	8	PSW.bit ← CY	× ×
		[HL].bit, CY	2	6	8	(HL).bit ← CY	

Notes 1. When the internal high-speed RAM area is accessed or instruction with no data access

2. When an area except the internal high-speed RAM area is accessed

Remark One instruction clock cycle is one cycle of the CPU clock (f_{CPU}) selected by the processor clock control register (PCC).

Instruction Group	Mnemonic	Operands	Byte	Clock		Operation	Flag		
				Note 1	Note 2		Z	AC	CY
Bit manipulation	**AND1**	CY, saddr.bit	3	6	7	CY ← CY ∧ (saddr.bit)			×
		CY, sfr.bit	3	—	7	CY ← CY ∧ sfr.bit			×
		CY, A.bit	2	4	—	CY ← CY ∧ A.bit			×
		CY, PSW.bit	3	—	7	CY ← CY ∧ PSW.bit			×
		CY, [HL].bit	2	6	7	CY ← CY ∧ (HL).bit			×
	OR1	CY, saddr.bit	3	6	7	CY ← CY ∨ (saddr.bit)			×
		CY, sfr.bit	3	—	7	CY ← CY ∨ sfr.bit			×
		CY, A.bit	2	4	—	CY ← CY ∨ A.bit			×
		CY, PSW.bit	3	—	7	CY ← CY ∨ PSW.bit			×
		CY, [HL].bit	2	6	7	CY ← CY ∨ (HL).bit			×
	XOR1	CY, saddr.bit	3	6	7	CY ← CY ⊻ (saddr.bit)			×
		CY, sfr.bit	3	—	7	CY ← CY ⊻ sfr.bit			×
		CY, A.bit	2	4	—	CY ← CY ⊻ A.bit			×
		CY, PSW.bit	3	—	7	CY ← CY ⊻ PSW.bit			×
		CY, [HL].bit	2	6	7	CY ← CY ⊻ (HL).bit			×
	SET1	saddr.bit	2	4	6	(saddr.bit) ← 1			
		sfr.bit	3	—	8	sfr.bit ← 1			
		A.bit	2	4	—	A.bit ← 1			
		PSW.bit	2	—	6	PSW.bit ← 1	×	×	×
		[HL].bit	2	6	8	(HL).bit ← 1			
	CLR1	saddr.bit	2	4	6	(saddr.bit) ← 0			
		sfr.bit	3	—	8	sfr.bit ← 0			
		A.bit	2	4	—	A.bit ← 0			
		PSW.bit	2	—	6	PSW.bit ← 0	×	×	×
		[HL].bit	2	6	8	(HL).bit ← 0			
	SET1	CY	1	2	—	CY ← 1			1
	CLR1	CY	1	2	—	CY ← 0			0
	NOT1	CY	1	2	—	CY ← $\overline{CY}$			×

Notes 1. When the internal high-speed RAM area is accessed or instruction with no data access

2. When an area except the internal high-speed RAM area is accessed

Remark One instruction clock cycle is one cycle of the CPU clock (f_{CPU}) selected by the processor clock control register (PCC).

Instruction Group	Mnemonic	Operands	Byte	Clock		Operation	Flag
				Note 1	Note 2		Z AC CY
Call return	**CALL**	!addr16	3	7	—	(SP – 1) ← (PC + 3)H, (SP – 2) ← (PC + 3)L, PC ← addr16, SP ← SP – 2	
	CALLF	!addr11	2	5	—	(SP – 1) ← (PC + 2)H, (SP – 2) ← (PC + 2)L, PC15–11 ← 00001, PC10–0 ← addr11, SP ← SP – 2	
	CALLT	[addr5]	1	6	—	(SP – 1) ← (PC + 1)H, (SP – 2) ← (PC +1)L, PCH ← (00000000, addr5 + 1), PCL ← (00000000, addr5) SP ← SP – 2	
	BRK		1	6	—	(SP – 1) ← PSW, (SP – 2) ← (PC + 1)H, (SP – 3) ← (PC + 1)L, PCH ← (003FH), PCL ← (003EH), SP ← SP – 3, IE ← 0	
	RET		1	6	—	PCH ← (SP + 1), PCL ← (SP), SP ← SP + 2	
	RETI		1	6	—	PCH ← (SP + 1), PCL ← (SP), PSW ← (SP + 2), SP ← SP + 3, NMIS ← 0	R R R
Stack manipulation	**RETB**		1	6	—	PCH ← (SP + 1), PCL ← (SP), PSW ← (SP + 2), SP ← SP + 3	R R R
	PUSH	PSW	1	2	—	(SP – 1) ← PSW, SP ← SP – 1	
		rp	1	4	—	(SP – 1) ← rpH, (SP – 2) ← rpL, SP ← SP – 2	
	POP	PSW	1	2	—	PSW ← (SP), SP ← SP + 1	R R R
		rp	1	4	—	rpH ← (SP + 1), rpL ← (SP), SP ← SP + 2	
	MOVW	SP, #word	4	—	10	SP ← word	
		SP, AX	2	—	8	SP ← AX	
		AX, SP	2	—	8	AX ← SP	
Unconditional branch	**BR**	!addr16	3	6	—	PC ← addr16	
		$addr16	2	6	—	PC ← PC + 2 jdisp8	
		AX	2	8	—	PCH ← A, PCL ← X	
Conditional branch	**BC**	$addr16	2	6	—	PC ← PC + 2 jdisp8 if CY = 1	
	BNC	$addr16	2	6	—	PC ← PC + 2 jdisp8 if CY = 0	
	BZ	$addr16	2	6	—	PC ← PC + 2 jdisp8 if Z = 1	
	BNZ	$addr16	2	6	—	PC ← PC + 2 jdisp8 if Z = 0	

Notes 1. When the internal high-speed RAM area is accessed or instruction with no data access

2. When an area except the internal high-speed RAM area is accessed

Remark One instruction clock cycle is one cycle of the CPU clock (f_{CPU}) selected by the processor clock control register (PCC).

Instruction Group	Mnemonic	Operands	Byte	Clock		Operation	Flag
				Note 1	Note 2		Z AC CY
Conditional branch	BT	saddr.bit, $addr16	3	8	9	PC ← PC + 3 + jdisp8 if(saddr.bit) = 1	
		sfr.bit, $addr16	4	—	11	PC ← PC + 4 + jdisp8 if sfr.bit = 1	
		A.bit, $addr16	3	8	—	PC ← PC + 3 + jdisp8 if A.bit = 1	
		PSW.bit, $addr16	3	—	9	PC ← PC + 3 + jdisp8 if PSW.bit = 1	
		[HL].bit, $addr16	3	10	11	PC ← PC + 3 + jdisp8 if (HL).bit = 1	
	BF	saddr.bit, $addr16	4	10	11	PC ← PC + 4 + jdisp8 if (saddr.bit) = 0	
		sfr.bit, $addr16	4	—	11	PC ← PC + 4 + jdisp8 if sfr.bit = 0	
		A.bit, $addr16	3	8	—	PC ← PC + 3 + jdisp8 if A.bit = 0	
		PSW.bit, $addr16	4	—	11	PC ← PC + 4 + jdisp8 if PSW.bit = 0	
		[HL].bit, $addr16	3	10	11	PC ← PC + 3 + jdisp8 if (HL).bit = 0	
	BTCLR	saddr.bit, $addr16	4	10	12	PC ← PC + 4 + jdisp8 if(saddr.bit) = 1 then reset (saddr.bit)	
		sfr.bit, $addr16	4	—	12	PC ← PC + 4 + jdisp8 if sfr.bit = 1 then reset sfr.bit	
		A.bit, $addr16	3	8	—	PC ← PC + 3 + jdisp8 if A.bit = 1 then reset A.bit	
		PSW.bit, $addr16	4	—	12	PC ← PC + 4 + jdisp8 if PSW.bit = 1 then reset PSW.bit	× × ×
		[HL].bit.$addr16	3	10	12	PC ← PC + 3 + jdisp8 if (HL).bit = 1 then reset (HL).bit	
	DBNZ	B, $addr16	2	6	—	B ← B – 1, then PC ← PC + 2 + jdisp8 if B • 0	
		C, $addr16	2	6	—	C ← C – 1, then PC ← PC + 2 + jdisp8 if C • 0	
		saddr, $addr16	3	8	10	(saddr) ← (saddr) –1 , then PC ← PC + 3 + jdisp8 if (saddr) • 0	
CPU control	SEL	RBn	2	4	—	RBS1, 0 ← n	
	NOP		1	2	—	No Operation	
	EI		2	—	6	IE ← 1 (Enable Interrupt)	
	DI		2	—	6	IE ← 0 (Disable Interrupt)	
	HALT		2	6	—	Set HALT Mode	
	STOP		2	6	—	Set STOP Mode	

Notes 1. When the internal high-speed RAM area is accessed or instruction with no data access

2. When an area except the internal high-speed RAM area is accessed

Remark One instruction clock cycle is one cycle of the CPU clock (f_{CPU}) selected by the processor clock control register (PCC).

16.3 Instructions Listed by Addressing Type

(1) 8-bit instructions

MOV, XCH, ADD, ADDC, SUB, SUBC, AND, OR, XOR, CMP, MULU, DIVUW, INC, DEC, ROR, ROL, RORC, ROLC, ROR4, ROL4, PUSH, POP, DBNZ

First Operand \ Second Operand	#byte	A	r[Note]	sfr	saddr	!addr16	PSW	[DE]	[HL]	[HL+byte] [HL+B] [HL+C]	$addr16	1	None
A	ADD ADDC SUB SUBC AND OR XOR CMP		MOV XCH ADD ADDC SUB SUBC AND OR XOR CMP	MOV XCH	MOV XCH ADD ADDC SUB SUBC AND OR XOR CMP	MOV XCH ADD ADDC SUB SUBC AND OR XOR CMP	MOV	MOV XCH	MOV XCH ADD ADDC SUB SUBC AND OR XOR CMP	MOV XCH ADD ADDC SUB SUBC AND OR XOR CMP		ROR ROL RORC ROLC	
r	MOV	MOV ADD ADDC SUB SUBC AND OR XOR CMP											INC DEC
B, C											DBNZ		
sfr	MOV	MOV											
saddr	MOV ADD ADDC SUB SUBC AND OR XOR CMP	MOV									DBNZ		INC DEC
!addr16		MOV											
PSW	MOV	MOV											PUSH POP
[DE]		MOV											
[HL]		MOV											ROR4 ROL4
[HL+byte] [HL+B] [HL+C]		MOV											
X													MULU
C													DIVUW

Note Except r = A

(2) 16-bit instructions

MOVW, XCHW, ADDW, SUBW, CMPW, PUSH, POP, INCW, DECW

First Operand \ Second Operand	#word	AX	rp[Note]	sfrp	saddrp	!addr16	SP	None
AX	ADDW SUBW CMPW		MOVW XCHW	MOVW	MOVW	MOVW	MOVW	
rp	MOVW	MOVW[Note]						INCW DECW PUSH POP
sfrp	MOVW	MOVW						
saddrp	MOVW	MOVW						
!addr16		MOVW						
SP	MOVW	MOVW						

Note Only when rp = BC, DE, HL

(3) Bit manipulation instructions

MOV1, AND1, OR1, XOR1, SET1, CLR1, NOT1, BT, BF, BTCLR

First Operand \ Second Operand	A.bit	sfr.bit	saddr.bit	PSW.bit	[HL].bit	CY	$addr16	None
A.bit						MOV1	BT BF BTCLR	SET1 CLR1
sfr.bit						MOV1	BT BF BTCLR	SET1 CLR1
saddr.bit						MOV1	BT BF BTCLR	SET1 CLR1
PSW.bit						MOV1	BT BF BTCLR	SET1 CLR1
[HL].bit						MOV1	BT BF BTCLR	SET1 CLR1
CY	MOV1 AND1 OR1 XOR1	MOV1 AND1 OR1 XOR1	MOV1 AND1 OR1 XOR1	MOV1 AND1 OR1 XOR1	MOV1 AND1 OR1 XOR1			SET1 CLR1 NOT1

(4) Call instructions/branch instructions

CALL, CALLF, CALLT, BR, BC, BNC, BZ, BNZ, BT, BF, BTCLR, DBNZ

Second Operand / First Operand	AX	!addr16	!addr11	[addr5]	$addr16
Basic instruction	BR	CALL BR	CALLF	CALLT	BR BC BNC BZ BNZ
Compound instruciton					BT BF BTCLR DBNZ

(5) Other instructions

ADJBA, ADJBS, BRK, RET, RETI, RETB, SEL, NOP, EI, DI, HALT, STOP

[MEMO]

★ APPENDIX A DIFFERENCES BETWEEN μPD78044H, 780228, AND 780208 SUBSERIES

Table A-1 shows the major differences between the μPD78044H, 780228, and 780208 subseries.

Table A-1. Major Differences between μPD78044H, 780226, and 780208 Subseries

Item \ Part Number		μPD78044H Subseries	μPD780228 Subseries	μPD780208 Subseries
PROM or flash memory model		μPD78P048B (PROM)	μPD78F0228 (flash memory)	μPD78P0208 (PROM)
Supply voltage		V_{DD} = 2.7 to 5.5 V	V_{DD} = 4.5 to 5.5 V	V_{DD} = 2.7 to 5.5 V
Internal ROM size		μPD78044H: 32K bytes μPD78045H: 40K bytes μPD78046H: 48K bytes μPD780P048B: 60K bytes	μPD780226: 48K bytes μPD780228: 60K bytes μPD78F0228: 60K bytes	μPD780204: 32K bytes μPD780205: 40K bytes μPD780206: 48K bytes μPD780208: 60K bytes μPD78P0208: 60K bytes
Internal expansion RAM size		μPD78P048B only: 1024 bytes	512 bytes	μPD780206, 780208, and 78P0208 only: 1024 bytes
Internal buffer RAM size		μPD78P048B only: 64 bytes	None	64 bytes
FIP display RAM size		48 bytes	96 bytes	80 bytes
CPU clock		Main system clock or subsystem clock selectable	Main system clock only	Main system clock or subsystem clock selectable
I/O port		68 pins	72 pins	74 pins
Total of FIP display output pins		34 pins	48 pins	53 pins
Serial interface		1 channel		2 channels
Timer		16-bit timer/event counter : 1 channel 8-bit timer/event counter : 2 channels Watch timer : 1 channel Watchdog timer: 1 channel	8-bit remote control timer : 1 channel 8-bit PWM timer: 2 channels Watchdog timer : 1 channel	16-bit timer/event counter : 1 channel 8-bit timer/event counter : 2 channels Watch timer : 1 channel Watchdog timer: 1 channel
Clock output		Provided	None	Provided
Buzzer output		Provided	None	Provided
Vectored interrupt source	Internal	10	8	11
	External	4	4	4
Test input		Provided	None	Provided
Package		80-pin plastic QFP (14 × 20 mm) 80-pin ceramic WQFN (μPD78P048B only)	100-pin plastic QFP (14 × 20 mm)	100-pin plastic QFP (14 × 20 mm) 100-pin ceramic WQFN (μPD78P0208 only)
Electrical characteristics and recommended soldering conditions		Refer to individual Data Sheet.		

Remark In addition to the above items, the organization of the development tools also differ between the above subseries (especially between the PROM model and flash memory model). For details, refer to the User's Manual of each subseries.

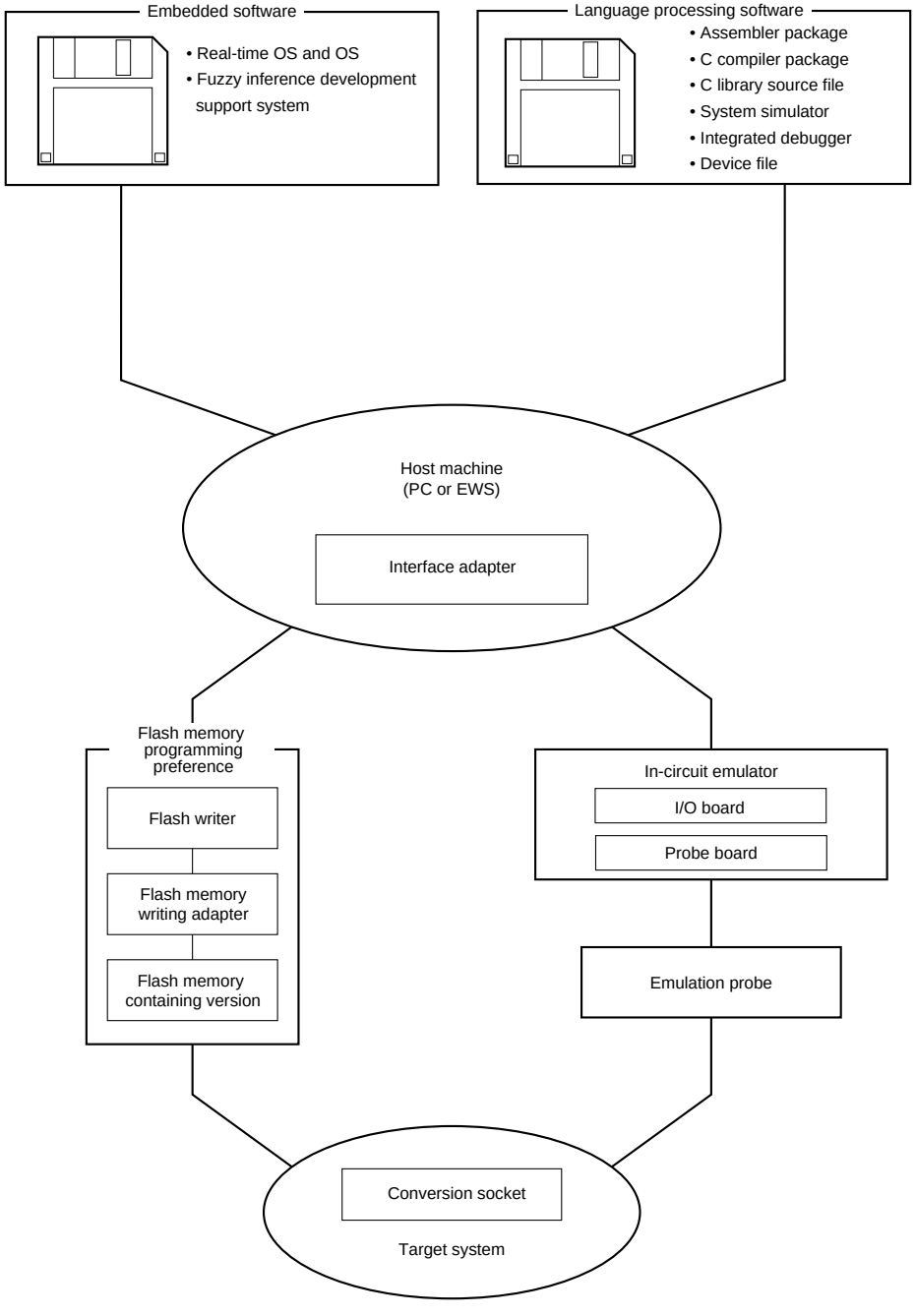
[MEMO]

APPENDIX B DEVELOPMENT TOOLS

The following development tools are available for the development of systems which employ the μPD780228 subseries.

Figure B-1 shows the development tools.

Figure B-1. Development Tools



B.1 Language Processing Software

RA78K/0 Assembler package	This is a program to convert a program written in mnemonics into an object code executable with a microcontroller. Further, this assembler is provided with functions capable of automatically creating symbol tables and branch instruction optimization. Used in combination with optional device file (DF780228).
	Part Number: μSxxxxRA78K0
CC78K/0 C Compiler package	This is a program to convert a program written in C language into an object code executable with a microcontroller. Used in combination with optional assembler package (RA78K/0) and device file (DF780228).
	Part Number: μSxxxxCC78K0
DF780228[Note 1, 2] Device file	This is a file containning the information inherent to the device. Used in combination with optional RA78K/0, CC78K/0, SM78K0, or ID78K0.
	Part Number: μSxxxxDF780228
CC78K/0-L C Library source file	A function source program configurating object library included in C compiler (CC78K/0). Necessary for changing object library included in CC78K/0 according to customer's specifications.
	Part Number: μSxxxxCC78K0-L

Notes 1. The DF780228 can be used for any of the RA78K/0, CC78K/0, SM78K0, and ID78K0 products.
2. Under development

Remark xxxx in the part number differs depending on the host machine and OS used.

μSxxxxRA78K0
μSxxxxCC78K0
μSxxxxDF780228
μSxxxxCC78K0-L

xxxx	Host Machine	OS	Distribution Medium
5A13	PC-9800 series	MS-DOS	3.5-inch 2HD
5A10		(Ver.3.30 to Ver.6.2[Note])	5-inch 2HD
7B13	IBM PC/AT and	Refer to **B.4**	3.5-inch 2HC
7B10	compatible machines		5-inch 2HC
3H15	HP9000 series 300™	HP-UX™ (rel.7.05B)	Cartridge tape (QIC-24)
3P16	HP9000 series 700™	HP-UX (rel.9.01)	Digital audio tape (DAT)
3K15	SPARCstation™	SunOS™ (rel.4.1.1)	Cartridge tape
3M15	EWS4800 series (RISC)	EWS-UX/V (rel.4.0)	(QIC-24)

Note Although MS-DOS Ver. 5.0 and above have a task swap function, this function cannot be used with this software.

B.2 Flash Memory Writing Tools

★

Flashpro II (part number: FL-PR2) Flash writer	Flash writer dedicated to microcontroller with flash memory. This is a product of Naito Densei Machida Mfg. Co., Ltd.
FA-100GF Flash memory writing adapter	Flash memory writing adapter for μPD780228 subseries and is connected to Flashpro II. This adapter is for a 100-pin plastic QFP (GF-3BA type). This is a product of Naito Densei Machida Mfg. Co., Ltd.

B.3 Debugging Tools

B.3.1 Hardware

★	IE78001-R-A-SL[Note] In-circuit emulator	This in-circuit emulator is used to debug the hardware and software when an application system using the 78K/0 series is developed. It supports the integrated debugger (ID78K0). This emulator is used with an emulation probe, and an interface adapter to connect the host machine.
	IE-70000-98-IF-B Interface adapter	Adapter necessary for using the PC-9800 series (excluding the notebook type) as the host machine of the IE-78001-R-A.
	IE-70000-98N-F Interface adapter	Adapter and cable necessary for using a PC-9800 series notebook type computer as the host machine of the IE-78001-R-A. This adapter cannot be connected unless the connector of the expansion bus of the notebook type computer is of 110-pin type.
	IE-70000-PC-IF-B Interface adapter.	Adapter necessary for using an IBM PC/AT or compatible machine as the host machine of the IE-78001-R-A.
	IE-78K0-SL-P01[Note] I/O board	This board is used to emulate device-specific hardware. It is used with an in-circuit emulator and probe board.
	IE-780228-SL-EM4[Note] Probe board	This board is used to set mask option and converts pin connector.
★	EP-100GF-SL Emulation probe	This probe connects an in-circuit emulator and the target system. It is for a 100-pin plastic QFP (GF-3BA type). A 100-pin conversion socket, NQPACK100RB, and a conversion adapter, YQPACK100RB, which facilitates development of the target system, are also supplied.
	NQPACK100RB conversion socket	This conversion socket is used to connect the board of the target system that is designed to mount a 100-pin plastic QFP (GF-3BA type), and the EP100GF-SL. To connect the EP-100GF-SL, this socket is used with YQPACK100RB. Instead of connecting EP-100GF-SL, a device can be mounted.
	YQPACK100RB conversion adapter	This is a conversion adapter used to connect the NQPACK100RB and EP-100GF-SL.
	HQPACK100RB	This is a lid used when a device is mounted to the NQPACK100RB.

Note Under development

Remarks 1. NQPACK100RB, YQPACK100RB, and HQPACK100RB are products of Tokyo Eletech Corp. (03-5295-1661). Consult NEC distributor when purchasing these products.

2. NQPACK100RB, YQPACK100RB, and HQPACK100RB are available in one unit.

B.3.2 Software (1/2)

SM78K0 System simulator	Debugs program at C source level or assembler level while simulating operation of target system on host machine. SM78K0 runs on Windows. By using the SM78K0, the logic and performance of an application can be verified independently of hardware development even when the in-circuit emulator is not used. This enhances development efficiency and improves software quality. Used in combination with optional device file (DF780228).
	Part Number : μSxxxxSM78K0

Remark xxxx in the part number differs depending on the host machine and OS used.

μSxxxxSM78K0

xxxx	Host Machine	OS	Distribution Medium
AA13	PC-9800 series	MS-DOS (Ver.3.30 to Ver.6.2[Note]) + Windows (Ver.3.0 to Ver.3.1)	3.5-inch 2HD
AB13	IBM PC/AT and compatible machines (Windows Japanese version)	Refer to **B.4**	3.5-inch 2HC
BB13	IBM PC/AT and compatible machines (Windows English version)		

Note Although MS-DOS Ver. 5.0 and above have a task swap function, this function cannot be used with this software.

B.3.2 Software (2/2)

ID78K0 Integrated Debugger	The ID78K0 is a control program to debug the 78K/0 series. The ID78K0 uses Windows on personal computers and OSF/Motif™ on EWS as graphical interface, and presents the appearance and operatability conforming to these platforms. The ID78K0 has enhanced debugging function supporting C language, and the trace result can be displayed by using the window integration function that interlocks source program, disassemble display, and memory display to the trace result. In addition, debugging efficiency for programs using real time OS can be enhanced by incorporating extension modules such as task debugger and system performance analyzer. Used in combination with optional device file (DF780228).
	Part Number : μSxxxxID78K0

Remark xxxx in the part number differs depending on the host machine and OS used.

μSxxxxID78K0

xxxx	Host Machine	OS	Distribution Medium
AA13	PC-9800 Series	MS-DOS (Ver.3.30 to Ver.6.2[Note]) + Windows (Ver.3.1)	3.5-inch 2HD
AB13	IBM PC/AT and their compatible machines (Windows Japanese version)	Refer to **B.4**	3.5-inch 2HC
BB13	IBM PC/AT and their compatible machines (Windows English version)		3.5-inch 2HC
3P16	HP9000 series 700	HP-UX (rel.9.01)	Digital audio tape (DAT)
3K15	SPARCstation	SunOS (rel.4.1.1)	Cartridge tape (QIC-24)
3K13			3.5-inch 2HC
3R16	NEWS™ (RISC)	NEWS-OS™ (6.1x)	1/4-inch CGMT
3R13			3.5-inch 2HC
3M15	EWS4800 series (RISC)	EWS-UX/V (rel.4.0)	Cartridge tape (QIC-24)

Note Although MS-DOS Ver. 5.0 and above have a task swap function, this function cannot be used with this software.

B.4 OS for IBM PC

The following OSs for the IBM PC are supported.

To operate SM78K0, ID78K/0, and FE9200 (refer to **C.2 Fuzzy Inference Development Support System**), Windows (Ver. 3.0 to Ver. 3.1) is necessary.

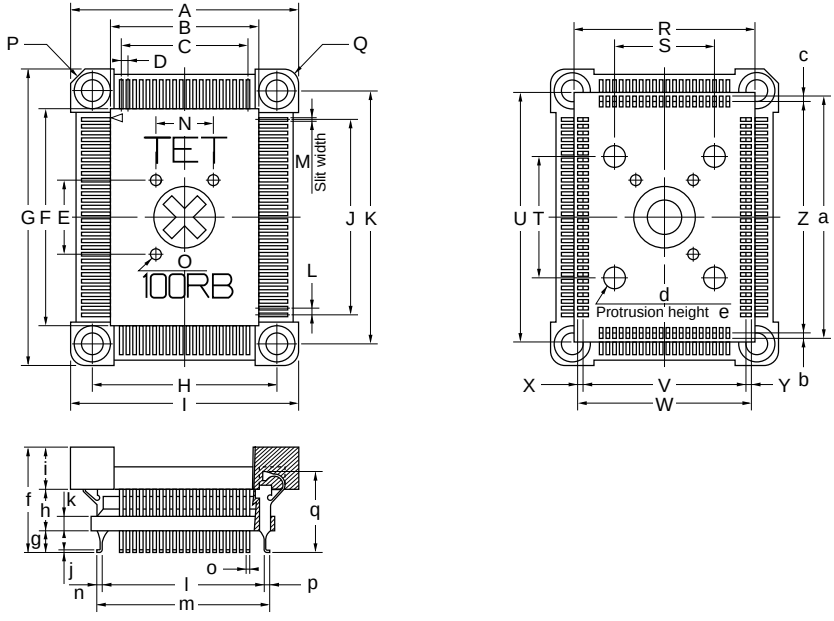
OS	Version
PC DOS	Ver. 5.02 to Ver. 6.3
	J6.1/V[Note] to J6.3/V[Note]
IBM DOS™	J5.02/V[Note]
MS-DOS	Ver. 5.0 to Ver. 6.22
	5.0/V[Note] to 6.2/V[Note]

Note Only English mode is supported.

Caution Although MS-DOS Ver. 5.0 and above have a task swap function, this function cannot be used with this software.

PACKAGE DRAWINGS OF NQPACK100RB, HQPACK100RB, AND YQPACK100RB

Figure B-2. NQPACK100RB (Target-Connected Side) Package Drawings (Reference)



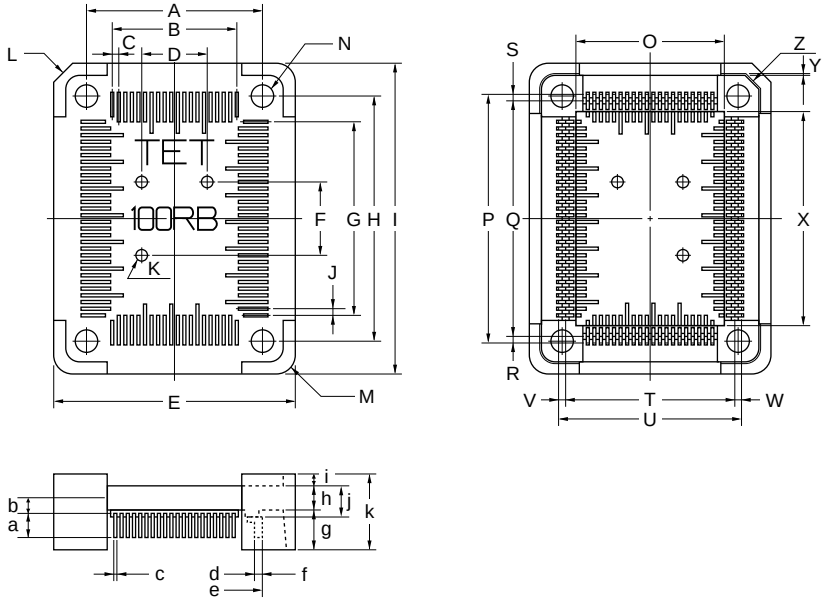
ITEM	MILLIMETERS	INCHES
A	21.75	0.856
B	14.25	0.561
C	0.65x19=12.35	0.026x0.748=0.486
D	0.65	0.026
E	7.0	0.276
F	20.75	0.817
G	28.25	1.112
H	17.4	0.685
I	21.75	0.856
J	0.65x29=18.85	0.026x1.142=0.742
K	23.9	0.941
L	0.65	0.026
M	0.4	0.016
N	6.0	0.236
O	3-ϕ1.0	3-ϕ0.039
P	C 1.5	C 0.059
Q	3-R 1.5	3-R 0.059
R	17.15	0.675
S	10.0	0.394
T	12.0	0.472
U	23.65	0.931
V	15.25	0.600
W	16.25	0.640
X	0.5	0.020
Y	0.5	0.020
Z	21.75	0.856

ITEM	MILLIMETERS	INCHES
a	22.75	0.896
b	0.5	0.020
c	0.5	0.020
d	4-ϕ2.0	4-ϕ0.079
e	1.8	0.071
f	9.45	0.372
g	1.85	0.073
h	3.7	0.146
i	3.9	0.154
j	0.2	0.008
k	1.2	0.047
l	15.25	0.600
m	16.25	0.640
n	0.5	0.020
o	0.25	0.010
p	0.5	0.020
q	6.95	0.274

NQPACK100RB-G1E

Remark Manufactured by Tokyo Eletech Corp.

Figure B-3. HQPACK100RB (Lid for Device Mounted) Package Drawings (Reference)



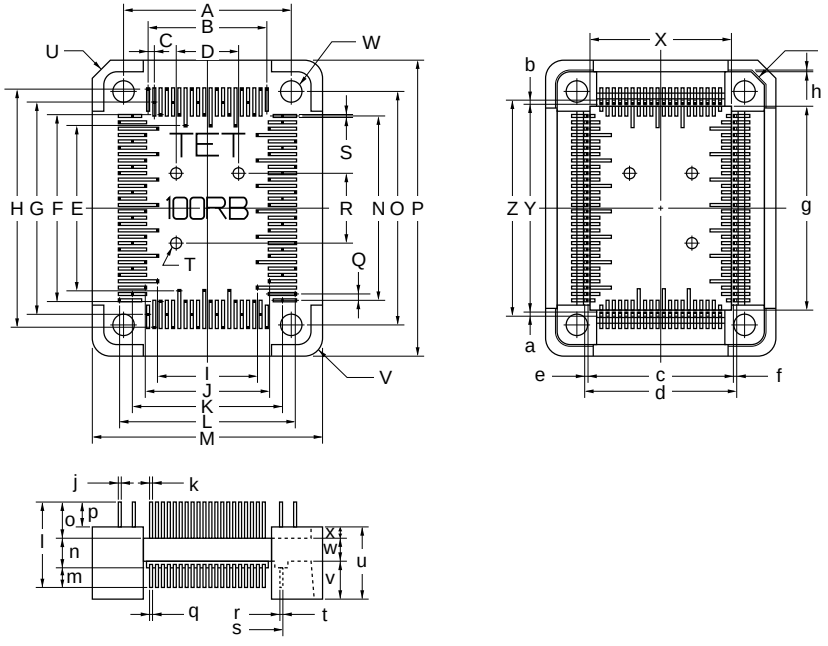
ITEM	MILLIMETERS	INCHES
A	17.4	0.685
B	0.65x19=12.35	0.026x0.748=0.486
C	0.65	0.026
D	6.0	0.236
E	23.75	0.935
F	7.0	0.276
G	0.65x29=18.85	0.026x1.142=0.742
H	23.9	0.941
I	30.25	1.191
J	0.65	0.026
K	3-ϕ1.0	3-ϕ0.039
L	C 2.0	C 0.079
M	3-R 2.5	3-R 0.098
N	4-ϕ2.2	4-ϕ0.087
O	14.1	0.555
P	24.07	0.948
Q	23.07	0.908
R	0.5	0.020
S	0.5	0.020
T	16.57	0.652
U	17.57	0.692
V	0.5	0.020
W	0.5	0.020
X	20.1	0.791
Y	0.2	0.008
Z	C 1.5	C 0.059

ITEM	MILLIMETERS	INCHES
a	2.25	0.089
b	1.6	0.063
c	0.25	0.010
d	16.57	0.652
e	17.57	0.692
f	0.5	0.020
g	3.9	0.154
h	2.3	0.091
i	1.2	0.047
j	3.1	0.122
k	7.4	0.291

HQPACK100RB-G0E

Remark Manufactured by Tokyo Eletech Corp.

Figure B-4. YQPACK100RB (Probe Side) Package Drawings (Reference)



ITEM	MILLIMETERS	INCHES
A	17.4	0.685
B	0.65x19=12.35	0.026x0.748=0.486
C	0.65	0.026
D	6.0	0.236
E	16.85	0.663
F	19.25	0.758
G	21.65	0.852
H	24.05	0.947
I	10.35	0.407
J	12.75	0.502
K	15.15	0.596
L	17.55	0.691
M	23.75	0.935
N	0.65x29=18.85	0.026x1.142=0.742
O	23.9	0.941
P	30.25	1.191
Q	0.65	0.026
R	7.0	0.276
S	0.25	0.010
T	3-ϕ1.0	3-ϕ0.039
U	C 2.0	C 0.079
V	3-R 2.5	3-R 0.098
W	4-ϕ2.2	4-ϕ0.087
X	14.1	0.555
Y	21.25	0.837
Z	22.05	0.868

ITEM	MILLIMETERS	INCHES
a	0.4	0.685
b	0.4	0.026
c	14.75	0.748
d	15.55	0.486
e	0.4	0.016
f	0.4	0.016
g	20.1	0.791
h	0.2	0.008
i	C 1.5	C 0.059
j	0.3	0.012
k	0.25x0.3	0.010x0.012
l	9.0	0.354
m	2.2	0.087
n	3.1	0.122
o	3.7	0.146
p	2.5	0.098
q	0.25	0.010
r	14.75	0.581
s	15.55	0.612
t	0.4	0.016
u	7.4	0.291
v	3.9	0.154
w	2.3	0.091
x	1.2	0.047

YQPACK100RB-G0E

Remark Manufactured by Tokyo Eletech Corp.

[MEMO]

APPENDIX C EMBEDDED SOFTWARE

For efficient program development and maintenance of the μPD780228 subseries, the following embedded software products are available.

C.1 Real-time OS (1/2)

RX78K/0 Real-time OS	RX78K/0 is a real-time OS which is based on the μITRON specification. Supplied with the RX78K/0 nucleus and a tool to prepare multiple information tables (configurator). Used in combination with optional assembler package (RA78K/0) and device file (DF780228).
	Part Number: μSxxxxRX78013-△△△△

Caution When purchasing the RX78K/0, fill in the purchase application form in advance, and sign the User Agreement.

Remark xxxx and △△△△ in the part number differs depending on host machine and OS, etc.

μSxxxxRX78013-△△△△

△△△△	Product Outline	Maximum Number for Use in Mass Production
001	Evaluation object	Do not use for mass-produced product.
100K	Mass production object	100,000
001M		1,000,000
010M		10,000,000
S01	Source program	Source program of mass production object

xxxx	Host Machine	OS	Distribution Medium
5A13	PC-9800 series	MS-DOS (Ver. 3.30 to Ver.6.2[Note])	3.5-inch 2HD
5A10			5-inch 2HD
7B13	IBM PC/AT and their compatible machines	Refer to **B.4**	3.5-inch 2HC
7B10			5-inch 2HC
3H15	HP9000 series 300	HP-UX (rel.7.05B)	Cartridge tape (QIC-24)
3P16	HP9000 series 700	HP-UX (rel.9.01)	Digital audio tape (DAT)
3K15	SPARCstation	SunOS (rel.4.1.1)	Cartridge tape (QIC-24)
3M15	EWS4800 series (RISC)	EWS-UX/V (rel.4.0)	

Note Although MS-DOS Ver. 5.0 and above have a task swap function, this function cannot be used with this software.

C.1 Real-time OS (2/2)

MX78K0 OS	μITRON-specification subset OS. Nucleus of MX78K0 is supplied. This OS performs task management, event management, and time management. It controls the task execution sequence for task management and selects the task to be executed next.
	Part number: μSxxxxMX78K0-△△△

Remark xxxx and △△△ in the part number differs depending on the host machine and OS, etc.

μSxxxxMX78K0-△△△

△△△	Product Outline	Note
001	Evaluation object	Use for trial product.
XX	Mass production object	Use for mass-produced product.
S01	Source program	Can be purchased only when object for mass-produced product is purchased

xxxx	Host Machine	OS	Distribution Medium
5A13	PC-9800 series	MS-DOS (Ver. 3.30 to Ver. 6.2[Note])	3.5-inch 2HD
5A10			5-inch 2HD
7B13	IBM PC/AT and their compatible machines	Refer to **B.4**	3.5-inch 2HC
7B10			5-inch 2HC
3H15	HP9000 series 300	HP-UX (rel.7.05B)	Cartridge tape (QIC-24)
3P16	HP9000 series 700	HP-UX (rel.9.01)	Digital audio tape (DAT)
3K15	SPARCstation	SunOS (rel.4.1.1)	Cartridge tape (QIC-24)
3M15	EWS4800 series (RISC)	EWS-UX/V (rel.4.0)	

Note Although MS-DOS Ver. 5.0 and above have a task swap function, this function cannot be used with this software.

C.2 Fuzzy Inference Development Support System

FE9000/FE9200 Fuzzy Knowledge Data Preparation Tool	Program supporting input of fuzzy knowledge data (fuzzy rule and membership function), editing (edit), and evaluation (simulation). FE9200 operations on Windows.
	Part Number (Product Name): μSxxxxFE9000 (PC-9800 series) μSxxxxFE9200 (IBM PC/AT and compatible machines)
FT9080/FT9085 Translator	Program converting fuzzy knowledge data obtained by using fuzzy knowledge data preparation tool to the assembler source program for the RA78K/0.
	Part Number (Product Name): μSxxxxFT9080 (PC-9800 series) μSxxxxFT9085 (IBM PC/AT and compatible machines)
FI78K0 Fuzzy Inference Module	Program executing fuzzy inference. Fuzzy inference is executed by linking fuzzy knowledge data converted by translator.
	Part Number (Product Name): μSxxxxFI78K0 (PC-9800 series, IBM PC/AT and compatible machines)
FD78K0 Fuzzy Inference Debugger	Support software evaluating and adjusting fuzzy knowledge data at hardware level by using in-circuit emulator.
	Part Number (Product Name): μSxxxxFD78K0 (PC-9800 series, IBM PC/AT and compatible machines)

Remark xxxx in the part number differs depending on the host machine and OS.

μSxxxxFE9000
μSxxxxFT9080
μSxxxxFI78K0
μSxxxxFD78K0

xxxx	Host Machine	OS	Distribution Medium
5A13	PC-9800 series	MS-DOS	3.5-inch 2HD
5A10		(Ver. 3.30 to Ver. 6.2[Note])	5-inch 2HD

Note Although MS-DOS Ver. 5.0 and above have a task swap function, this function cannot be used with this software.

μSxxxxFE9200
μSxxxxFT9085
μSxxxxFI78K0
μSxxxxFD78K0

xxxx	Host Machine	OS	Distribution Medium
7B13	IBM PC/AT and compatible machines	Refer to **B.4**	3.5-inch 2HC
7B10			5-inch 2HC

APPENDIX D REGISTER INDEX

D.1 Register Index

★ APPENDIX E REVISION HISTORY

This table shows the revision history of this manual. The column under the heading “Chapter” indicates the chapter of the preceding edition in which a revision has been made.

Edition	Major Revision from Previous Edition	Chapter
2nd edition	Change of μPD780226 and 780228 from “under development” to “developed”	Throughout
	Change of I/O circuit types of ports 7, 8, 9, and 10 of mask ROM model as follows: Ports 7, 8, and 9: Type 15-D to Type 15-F Port 10 : Type 14-D to Type 14-F	CHAPTER 2 PIN FUNCTIONS
	Addition of internal expansion RAM size select register (IXS)	CHAPTER 3 CPU ARCHITECTURE CHAPTER 14 RESET FUNCTION CHAPTER 15 μPD78F0228
	Correction of block diagrams of ports 7, 8, 9, and 10	CHAPTER 4 PORT FUNCTIONS
	Change of notes on inputting external clock	CHAPTER 5 CLOCK GENERATOR
	Change of count clock value by setting of TLC2 and TCL1 in Figure 6-2 Timer Mode Control Register 1 Format	CHAPTER 6 8-BIT REMOTE CONTROL TIMER
	Change of oscillation stabilization time by setting of OSTS2-OSTS0 in Format of Oscillation Stabilization Time Select Register	CHAPTER 8 WATCHDOG TIMER CHAPTER 13 STANDBY FUNCTION
	Change of setting of FOUT5-FOUT0, and addition of notes on FIP output pins	CHAPTER 11 FIP CONTROLLER/ DRIVER
	Addition of **11.7 Calculation of Total Power Dissipation**	
	Change of product name of dedicated flash writer as follows: Flashpro to Falshpro II (part No.: FL-PR2)	CHAPTER 15 μPD78F0228 APPENDIX B DEVELOPMENT TOOLS
	Addition of **APPENDIX A DIFFERENCES BETWEEN μPD78044H, 780228, AND 780208 SUBSERIES**	APPENDIX A DIFFERENCES BETWEEN μPD78044H, 780228, AND 780208 SUBSERIES
	Change of hardware for debugging tools as follows: Change of IE-780000-SL to IE-78001-R-A Deletion of IE-78K0-SL-EM (CPU core board) Change of EP-100GF-SL from “under development” to “developed”	APPENDIX B DEVELOPMENT TOOLS
	Deletion of **“Upgrading Your In-Circuit Emulator to Emulator for 78K/0 Series”**	

[MEMO]

NEC

Facsimile Message

Although NEC has taken all possible steps to ensure that the documentation supplied to our customers is complete, bug free and up-to-date, we readily accept that errors may occur. Despite all the care and precautions we've taken, you may encounter problems in the documentation. Please complete this form whenever you'd like to report errors or suggest improvements to us.

From:

Name

Company

Tel. FAX

Address

Thank you for your kind support.

North America
NEC Electronics Inc.
Corporate Communications Dept.
Fax: 1-800-729-9288
1-408-588-6130

Europe
NEC Electronics (Europe) GmbH
Technical Documentation Dept.
Fax: +49-211-6503-274

South America
NEC do Brasil S.A.
Fax: +55-11-889-1689

Hong Kong, Philippines, Oceania
NEC Electronics Hong Kong Ltd.
Fax: +852-2886-9022/9044

Korea
NEC Electronics Hong Kong Ltd.
Seoul Branch
Fax: 02-528-4411

Taiwan
NEC Electronics Taiwan Ltd.
Fax: 02-719-5951

Asian Nations except Philippines
NEC Electronics Singapore Pte. Ltd.
Fax: +65-250-3583

Japan
NEC Corporation
Semiconductor Solution Engineering Division
Technical Information Support Dept.
Fax: 044-548-7900

I would like to report the following error/make the following suggestion:

Document title: _____

Document number: _____ Page number: _____

If possible, please fax the referenced page or drawing.

Document Rating	Excellent	Good	Acceptable	Poor
Clarity	❑	❑	❑	❑
Technical Accuracy	❑	❑	❑	❑
Organization	❑	❑	❑	❑