十速科技股份有限公司
tenx technology inc.

TM87P08

4-Bit Micro-Controller with LCD Driver

User's Manual

tenx technology, inc.

CONTENTS

Chapter 1 General Description

1-1. GENERAL DESCRIPTION

The TM87P08 is an EPROM embedded high-performance 4-bit micro controller with LCD/LED driver. It contains all the functions in TM87-series for 3V/5V application, except fixed LCD PLA configuration.

1-2. FEATURES

1. Powerful instruction set (178 instructions).
   - Binary addition, subtraction, BCD adjusts, logical operation in direct and index addressing mode.
   - Single-bit manipulation (set, reset, decision for branch).
   - Various conditional branches.
   - 16 working registers and manipulation.
   - Table look-up.
   - LCD driver data transfer.
2. Memory capacity.

● Program ROM capacity	4096	x 16 bits
● Index ROM capacity	4096	x 8 bits
● Data RAM capacity	256	x 4 bits.

3. Input/output ports.

● Port IOA	4 pins (with internal pull-low).
● Port IOB	4 pins (with internal pull-low).
● Port IOC	4 pins (with internal pull-low, low-level-hold, chattering prevention clock).
● Port IOD	4 pins (with internal pull-low, chattering prevention clock).

4. 8 level subroutine nesting.
5. Interrupt function.

● External factor	4	(INT pin, Port IOC, IOD & KI input).
● Internal factor	4	(Pre-Divider, Timer1, Timer2, RFC).

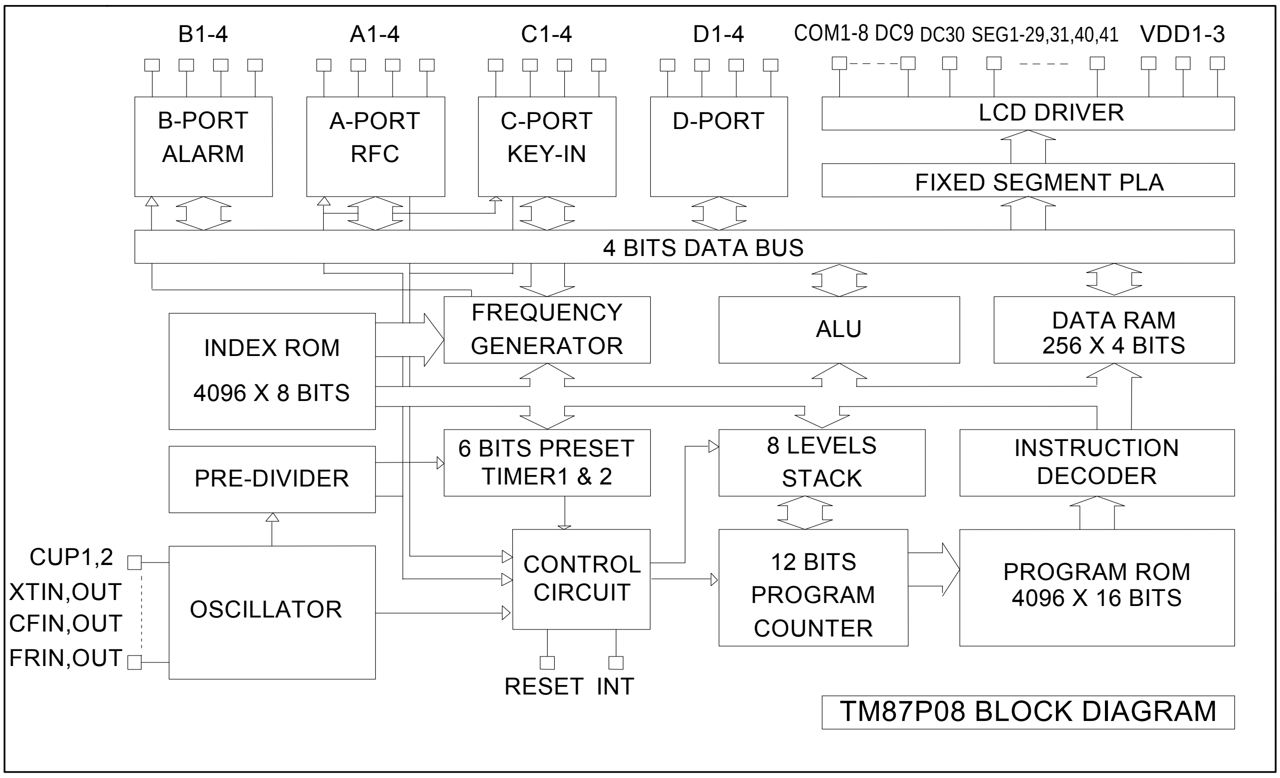
6. Built in Alarm, Frequency or Melody generator.
7. BZB, BZ (Mux with IOB3, IOB4).
8. Built-in R to F Converter circuit.
   - CX, RR, RT, RH (Mux with IOA1~IOA4).
9. Built in KEY_BOARD scanning function.
   - K1~K16 (Share with SEG1~SEG16).
   - KI1~KI4 (Mux with IOC1~IOC4).
10. Two 6-bit programmable timers with programmable clock source.

11. Watch dog timer.
12. LCD driver output.
    - 32 LCD/LED driver outputs (up to 128 or 256 LCD segment drivable).
    - 1/4 or 1/8 Duty for LCD/LED.
    - 1/2 Bias or 1/3 Bias for LCD/LED selected by option.
    - Single instruction to turn off all segments.
    - Option is used to select COM5~8,DC9/OD9,DC30/OD30 as DC outputs/P_open drain.
    - 32 LCD address.
13. Built-in Voltage doubler, halve charge pump circuit.
14. Dual clock operation, and X'tal type slow oscillation, and fast oscillation can set 3.58MHz ceramic resonator or external R by switch option.
15. HALT function.
16. STOP function.
17. ROM code protect fuse.

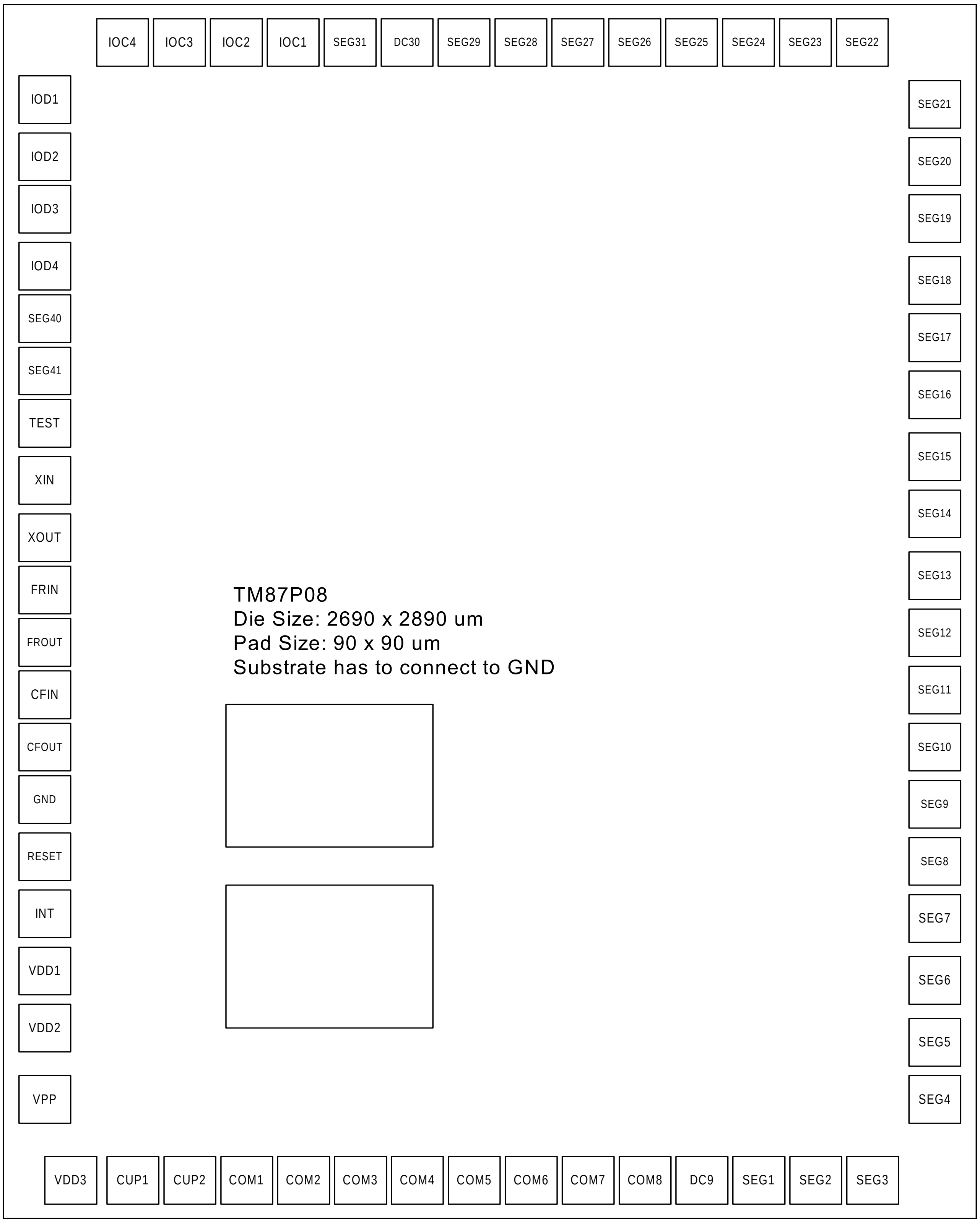
APPLICATION

■ Timer / Calendar / Calculator

1-3. BLOCK DIAGRAM



1-4. PAD DIAGRAM



1-5. PAD COORDINATE

No	Name	X	Y	No	Name	X	Y
1	XIN	102.70	1732.00	34	SEG10 (K10)	2587.30	1101.30
2	XOUT	102.70	1610.20	35	SEG11 (K11)	2587.30	1245.50
3	FRIN	102.70	1495.20	36	SEG12 (K12)	2587.30	1382.90
4	FROUT	102.70	1373.40	37	SEG13 (K13)	2587.30	1527.10
5	CFIN	102.70	1258.40	38	SEG14 (K14)	2587.30	1664.50
6	CFOUT	102.70	1136.60	39	SEG15 (K15)	2587.30	1808.70
7	GND	102.70	1018.85	40	SEG16 (K16)	2587.30	1946.10
8	RESET	102.70	901.10	41	SEG17	2587.30	2090.30
9	INT	102.70	779.10	42	SEG18	2587.30	2227.70
10	VDD1	102.70	662.00	43	SEG19	2587.30	2371.90
11	VDD(2)	102.70	547.00	44	SEG20	2587.30	2509.30
12	VPP	102.70	341.50	45	SEG21	2587.30	2653.50
13	VDD3	145.60	106.70	46	SEG22	2301.15	2787.30
14	CUP1	283.05	102.70	47	SEG23	2156.95	2787.30
15	CUP2	425.35	102.70	48	SEG24/IOA1/CX	1987.05	2787.30
16	COM1	569.75	102.70	49	SEG25/IOA2/RR	1842.85	2787.30
17	COM2	740.15	102.70	50	SEG26/IOA3/RT	1640.45	2787.30
18	COM3	910.55	102.70	51	SEG27/IOA4/RH	1496.25	2787.30
19	COM4	1080.95	102.70	52	SEG28/IOB1	1326.35	2787.30
20	COM5/DC5/OD5	1251.35	102.70	53	SEG29/IOB2	1182.15	2787.30
21	COM6/DC6/OD6	1421.75	102.70	54	DC30/OD30/IOB3/BZB	1053.45	2787.30
22	COM7/DC7/OD7	1592.15	102.70	55	SEG31/IOB4/BZ	902.15	2787.30
23	COM8/DC8/OD8	1762.55	102.70	56	IOC1/KI1	763.45	2787.30
24	DC9/OD9	1907.45	102.70	57	IOC2/KI2	616.25	2787.30
25	SEG1 (K1)	2038.55	102.70	58	IOC3/KI3	496.25	2787.30
26	SEG2 (K2)	2182.75	102.70	59	IOC4/KI4	349.05	2787.30
27	SEG3 (K3)	2320.15	102.70	60	IOD1	102.70	2667.50
28	SEG4 (K4)	2587.30	256.50	61	IOD2	102.70	2523.30
29	SEG5 (K5)	2587.30	400.70	62	IOD3	102.70	2403.30
30	SEG6 (K6)	2587.30	538.10	63	IOD4	102.70	2259.10
31	SEG7 (K7)	2587.30	682.30	64	SEG40	102.70	2130.40
32	SEG8 (K8)	2587.30	819.70	65	SEG41	102.70	1986.20
33	SEG9 (K9)	2587.30	963.90	66	TEST	102.70	1848.50

1-6. PIN DESCRIPTION

Name	I/O	Description
VDD1, 2, 3	P	LCD supply voltage, and positive supply voltage. Connect +3.0V battery positive pin to VDD2. Above 4.0V is need to VDD2 for Serial Program/Read Mode.
RESET	I	Input pin from LSI reset request signal, with internal pull-down resistor. Instruction Reset Time can select "PH15/2" or "PH12/2" by option. Reset Type can select "Level" or "Pulse" by option. Control Signal for Serial Program/Read Mode.
INT	I I/O	Input pin for external INT request signal.Falling edge or rising edge triggered by option.Internal pull-down or pull-up resistor is selected by option. Serial Data for Serial Program/Read Mode.
TEST	I	Test signal input pin. No Connected.
CUP1, 2	O	Switching pins for supply the LCD driving voltage to the VDD1, 2, 3 pins. Connect the CUP1 and CUP2 pins with non-polarized electrolytic capacitor if 1/2 or 1/3 bias mode has been selected. In no BIAS mode, these pins should be open.
XIN XOUT	**I** O	32KHz Crystal oscillator for Slow Clock. ***If XIN pin is unused, it must be connected to VDD2.***
CFIN CFOUT	**I** O	3.58MHz ceramic resonator oscillator for Fast Clock. ***If CFIN pin is unused, it must be connected to VDD.***
FRIN FROUT	**I** O	External R oscillation for Fast Clock. ***If FRIN pin is unused, it must be connected to GND.***
COM1~8	O	Output pins for driving the common pins of the LCD or LED panel. COM5~8 is muxed with DC/Open Drain, and set mask option
DC9	O	DC/Open Drain,
SEG1-29, 31,40, 41	O	Output pins for driving the LCD or LED panel segment.
IOA1-4	I/O	Input / Output port A, can use software to define internal pull-low Resistor. This port is muxed with SEG24~27, and set by option.
IOB1-4	I/O	Input / Output port B, can use software to define internal pull-low Resistor. This port is muxed with SEG28~31 / BZB, BZ, and set by option.
IOC1-4	I/O	Input / Output port C, can use software to define internal pull-low / low-level-hold Resistor and Chattering clock to reduce input bounce. This port is muxed with KI1~4, and set by option.
IOD1-4	I/O	Input / Output port D, can use software to define internal pull-low Resistor, and Chattering clock to reduce input bounce.
(RFC) CX RR/RT/RH	 I O	1 input pin and 3 output pins for RFC application. This port is muxed with SEG24~27 / IOA1~4, and set by option.
(ALM) BZB/BZ	 O	Output port for alarm, frequency or melody generator This port is muxed with DC30, SEG31 / IOB3, 4, and set by option.
KI1~4	I	Keyboard scanning input port. This port is muxed with SEG32~35 / IOC1~4, and set by option.
GND	P	Negative supply voltage. Connect for Serial Program/Read Mode.
VPP	P	Above 11.5V is connected to VPP for Program Mode.

Serial Program/Read Connect Pins:

VPP, VDD2, VDD3, GND, RESET, INT

1-7. CHARACTERIZATION

ABSOLOUTE MAXIMUM RATINGS

(GND= 0V)

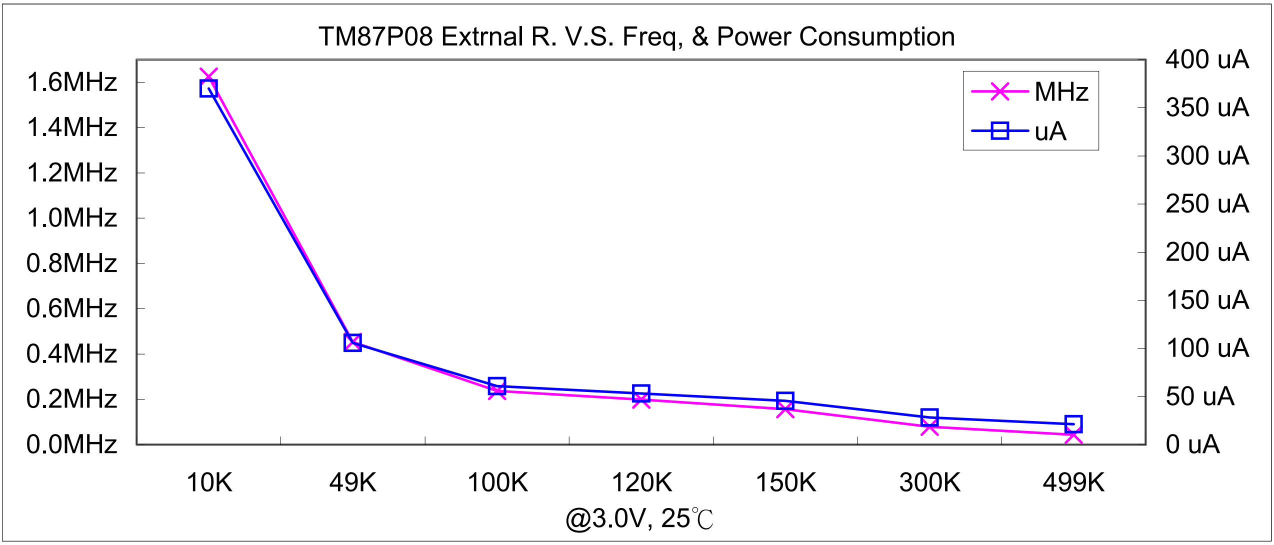
Name	Symbol	Range	Unit
Maximum Supply Voltage	VDD1	-0.3 to 5.5	V
	VDD2	-0.3 to 5.5	V
	VDD3	-0.3 to 8.5	V
	VPP	-0.3 to 13.5	V
Maximum Input Voltage	Vin	-0.3 to VDD1/2+0.3	V
Maximum output Voltage	Vout1	-0.3 to VDD1/2+0.3	V
	Vout2	-0.3 to VDD3+0.3	V
Maximum Operating Temperature	Topg	-20 to +70	℃
Maximum Storage Temperature	Tstg	-25 to +125	℃

POWER CONSUMPTION

at VDD2= 3.0V, Ta=-20℃ to 70℃, GND= 0V

Name	Sym.	Condition	Min.	Typ.	Max.	Unit
HALT mode	I_{HALT}	Only 32.768KHz Crystal oscillator operating, without loading. BCF = 0, 1/4 duty, ph0=BCLK		3	6	uA
STOP mode	I_{STOP}				1	uA
Normal Mode	I_{32K}	Only 32.768KHz Crystal oscillator operating, without loading. BCF = 0, 1/4 duty, ph0=BCLK	8			uA
External R	$I_{Ext. R}$	R = 150KΩ oscillator operating, without loading. BCF = 0, 1/4 duty, ph0=BCLK	36			uA
3.58MHz ceramic resonator	$I_{3.58Mcr}$	Only 3.58MHz ceramic resonator operating, without loading. BCF = 0, 1/4 duty, ph0=BCLK	480			uA

Note : When External R oscillator function is operating, the current consumption will depend on the frequency of
oscillation.



ALLOWABLE OPERATING CONDITIONS

at Ta=-20℃ to 70℃,GND= 0V

Name	Symb.	Condition	Min.	Max.	Unit
Supply Voltage	VDD2		2.4	5.25	V
	VDD3		2.4	8.0	V
	VPP		2.4	12.5	V
Oscillator Start-Up Voltage	VDD_{stup}	32.768KHz Crystal Mode	1.4		V
		3.58 ceramic resonator Mode	1.8		V
Oscillator Sustain Voltage	VDD_{sut}	32.768KHz Crystal Mode	1.3		V
		3.58 ceramic resonator Mode	1.55		V
Supply Voltage	VDD2	EXT-V, Li Mode	2.4	5.25	V
Input "H" Voltage	Vih1	Li Battery Mode	VDD2-0.7	VDD2+0.7	V
Input "L" Voltage	Vil1		-0.7	0.7	V
Input "H" Voltage	Vih2	OSCIN at Li Battery Mode	0.8xVDD2	VDD2	V
Input "L" Voltage	Vil2		0	0.2xVDD2	V
Input "H" Voltage	Vih3	CFIN at Li Battery or EXT-V Mode	0.8xVDD2	VDD2	V
Input "L" Voltage	Vil3		0	0.2xVDD2	V
Operating Freq	Fopg1	32.768KHz Crystal Mode	32		KHZ
	Fopg2	External R mode	10	1000	KHZ

ELECTRICAL CHARACTERISTICS

at#1:VDD2=3.0V(Li); at#2:VDD2=5.0V(Ext-V);

Input Resistance

Name	Symb.	Condition	Min.	Typ.	Max.	Unit
"L" Level Hold Tr(IOC)	Rllh1	Vi=0.2VDD2,#1	10	40	100	KΩ
	Rllh2	Vi=0.2VDD2,#2	5	20	50	KΩ
IOA,B,C Pull-Down Tr	Rmad1	Vi=VDD2,#1	200	500	1000	KΩ
	Rmad2	Vi=VDD2,#2	100	250	500	KΩ
INT Pull-up Tr	Rintu1	Vi=VDD2,#1	200	500	1000	KΩ
	Rintu2	Vi=VDD2,#2	100	250	500	KΩ
INT Pull-Down Tr	Rintd1	Vi=GND,#1	200	500	1000	KΩ
	Rintd2	Vi=GND,#2	100	250	500	KΩ
RES Pull-Down R	Rres1	Vi=GND or VDD2,#1	9	35	90	KΩ
	Rres2	Vi=GND or VDD2,#2	5	18	45	KΩ

at#3:VDD2=2.4V(Li); at#4:VDD2=4.0V(Ext-V);

DC Output Characteristics

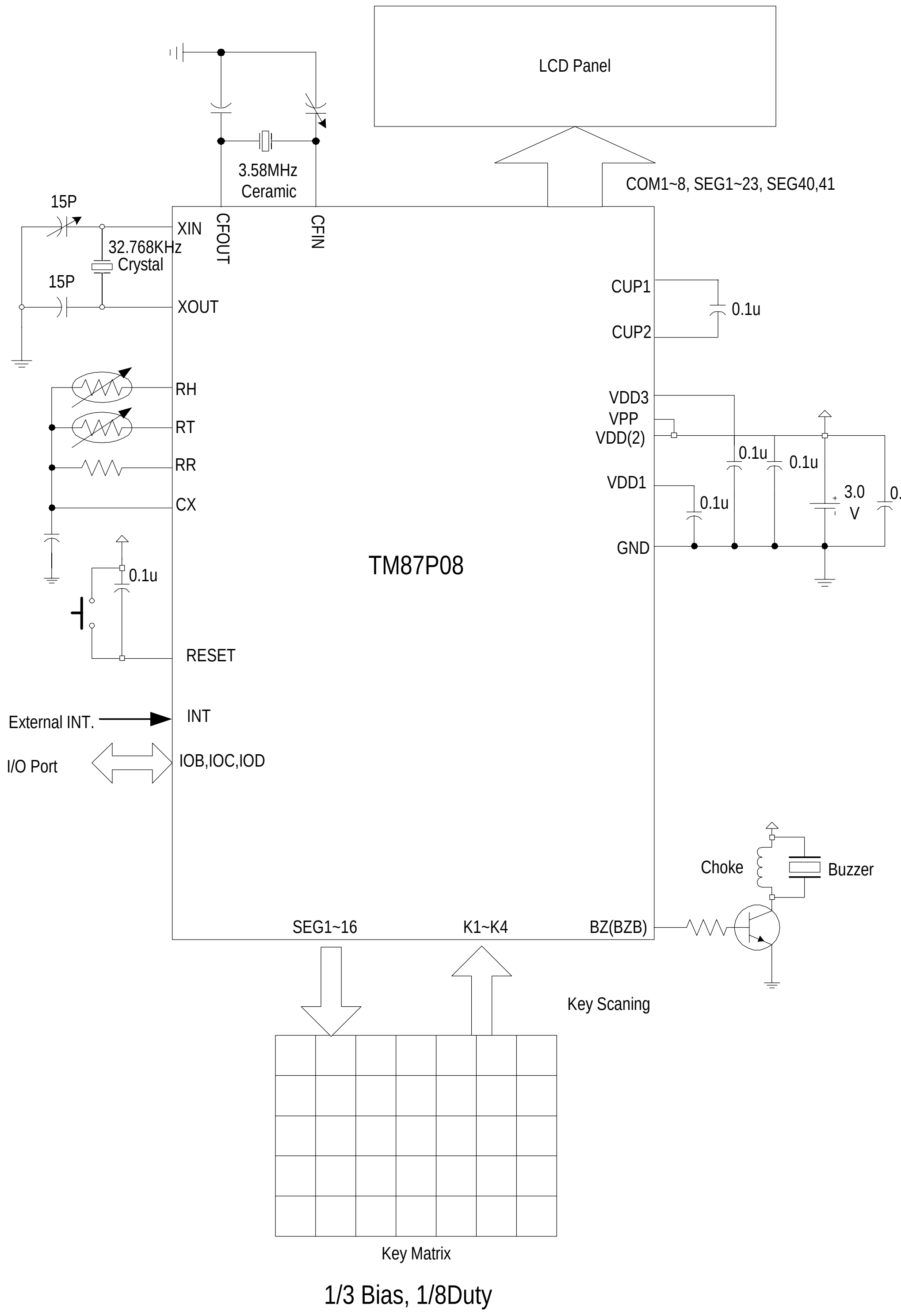
Name	Symb.	Condition	Port	Min.	Typ.	Max.	Unit
Output "H" Voltage	Voh3c	Ioh=-1mA,#3	COM5~9 SEG1~41	1.5	1.8		V
	Voh4c	Ioh=-3mA,#4		2.5	3.0		V
Output "L" Voltage	Vol3c	Iol=2mA,#3			0.6	0.9	V
	Vol4c	Iol=6mA,#4			1.0	1.5	V

Segment Driver Output Characteristics

Name	Symb.	Condition	For	Min.	Typ.	Max.	Unit.
1/2 Bias Display Mode							
Output "H" Voltage	Voh3f	Ioh=-1uA,#3	SEG-n	2.2			V
	Voh4f	Ioh=-1uA,#4		3.8			V
Output "L" Voltage	Vol3f	Iol=1uA,#3				0.2	V
	Vol4f	Iol=1uA,#4				0.2	V
Output "H" Voltage	Voh3g	Ioh=-10uA,#3	COM-n	2.2			V
	Voh4g	Ioh=-10uA,#4		3.8			V
Output "M" Voltage	Vom3g	Iol/h=+/-10uA,#3	COM-n	1.0		1.4	V
	Vom4g	Iol/h=+/-10uA,#4		1.8		2.2	V
Output "L" Voltage	Vol3g	Iol=10uA,#3				0.2	V
	Vol4g	Iol=10uA,#4				0.2	V
1/3 Bias display Mode							
Output "H" Voltage	Voh3i	Ioh=-1uA,#3	SEG-n	3.4			V
	Voh4i	Ioh=-1uA,#4		5.8			V
Output "M1" Voltage	Vom13i	Iol/h=+/-10uA,#3		1.0		1.4	V
	Vom14i	Iol/h=+/-10uA,#4		1.8		2.2	V
Output "M2" Voltage	Vom23i	Iol/h=+/-10uA,#3		2.2		2.6	V
	Vom24i	Iol/h=+/-10uA,#4		3.8		4.2	V
Output "L" Voltage	Vol3i	Iol=1uA,#3				0.2	V
	Vol4i	Iol=1uA,#4				0.2	V
Output "H" Voltage	Voh3j	Ioh=-10uA,#3	COM-n	3.4			V
	Voh4j	Ioh=-10uA,#4		5.8			V
Output "M1" Voltage	Vom13j	Iol/h=+/-10uA,#3		1.0		1.4	V
	Vom14j	Iol/h=+/-10uA,#4		1.8		2.2	V
Output "M2" Voltage	Vom23j	Iol/h=+/-10uA,#3		2.2		2.6	V
	Vom24j	Iol/h=+/-10uA,#4		3.8		4.2	V
Output "L" Voltage	Vol3j	Iol=10uA,#3				0.2	V
	Vol4j	Iol=10uA,#4				0.2	V

1-8. TYPICAL APPLICATION CIRCUIT

This application circuit is simply an example, and is not guaranteed to work.



Chapter 2 TM87P08 Internal System Architecture

2-1 Power Supply

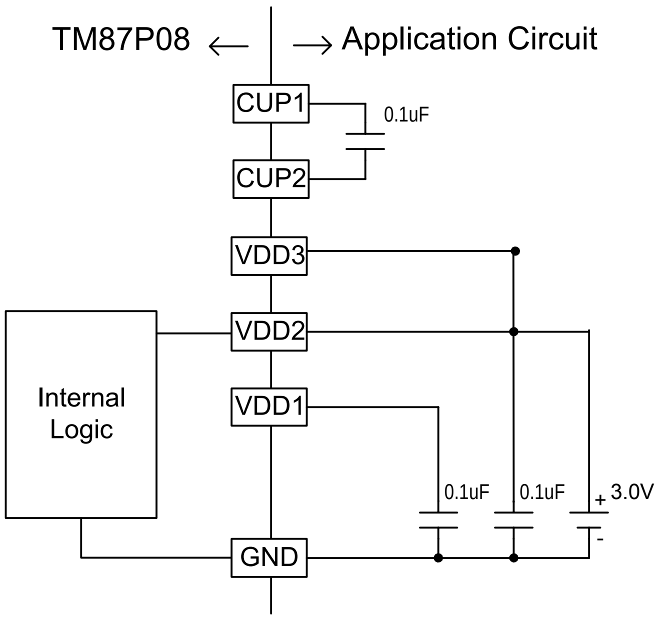
TM87P08 could operate at Li and ExtV 2 types supply voltage, all of these operating types are defined by mask option. The power supply circuitry also generated the necessary voltage level to drive the LCD panel with different bias. Shown below are the connection diagrams for 1/2 bias, 1/3 bias application.

2-1-1. LI BATTERY and ExtV POWER SUPPLY

Operating voltage range : 2.4V ~ 5.25V.
For different LCD bias application, the connection diagrams are shown below :

2-1-1-1. 1/2 BIAS

The backup falg flag (BCF) must be reset after the operation of the halver circuit is fully stabilized and a voltage of approximately 1/2 * VDD2 appears on the VDD1 pin.



MASK OPTION table :

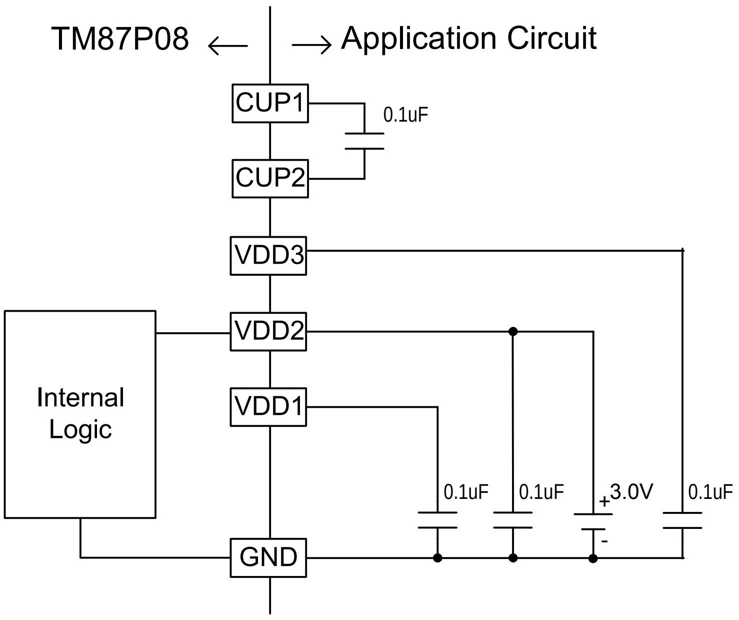
Mask Option name	Selected item
LCD BIAS	(2) 1/2 BIAS

Note 1: The input/output ports operate between GND and VDD2.

Note 2: The backup flag (BCF) is set in the initial clear mode.. When thebackup flag flag is set, the oscillator circuit becomes large in driver size.

When the backup flag is set, the operating current is increased. Therefore, the backup flag must be reset unless otherwise required. For the backup flag, refer to **2-17.**

2-1-2-2. 1/3 BIAS



MASK OPTION table :

Mask Option name	Selected item
LCD BIAS	(3) 1/3 BIAS

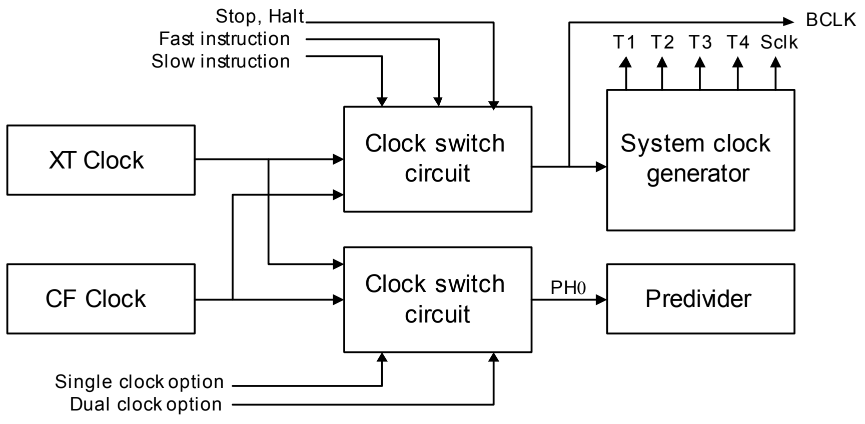
Note 1: The input/output ports operate between GND and VDD2.

Note 2: The backup flag (BCF) is set in the initial clear mode.. When thebackup flag flag is set, the oscillator circuit becomes large in driver size.

When the backup flag is set, the operating current is increased. Therefore, the backup flag must be reset unless otherwise required. For the backup flag, refer to **2-17.**

2-2. SYSTEM CLOCK

XT clock (slow clock oscillator) and CF clock (fast clock oscillator) compose the clock oscillation circuitry and the block diagram is shown below.

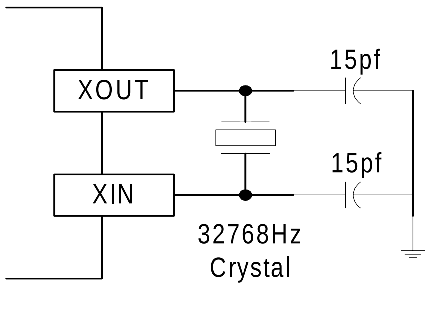


The system clock generator provided the necessary clocks for execution of instruction. The pre-divider generated several clocks with different frequencies for the usage of LCD driver, frequency generator … etc. The following table shows the clock sources of system clock generator and pre-divider in different conditions.

	PH0	BCLK
Slow clock only option	XT clock	XT clock
fast clock only option	CF clock	CF clock
Initial state(dual clock option)	XT clock	XT clock
Halt mode(dual clock option)	XT clock	XT clock
Slow mode(dual clock option)	XT clock	XT clock
Fast mode(dual clock option)	XT clock	CF clock

2-2-1 CONNECTION DIAGRAM OF SLOW CLOCK OSCILLATOR (XT CLOCK)

This clock oscillation circuitry provides the lower speed clock to the system clock generator, pre-divider, timer, chattering prevention of IO port and LCD circuitry. This oscillator will be disabled when the fast clock only option is selected by mask option, or it will be active all the time after the initial reset. In stop mode, this oscillator will be stopped.



(1) X'tal

When backup flag (BCF) is set to 1, the oscillator operates with an extra buffer in parallel in order to shorten the oscillator start-up time but this will increase the power consumption. Therefore, the backup flag should be reset unless required otherwise. If XIIN pin is unused, it must be connected to VDD2.

The following table shows the power consumption of Crystal oscillator in different conditions :

	Li power option	EXT-V power option
BCF=1	Increased	Increased
BCF=0	Normal	Increased
Initial reset	Increased	Increased
After reset	Normal	Increased

2-2-2. CONNECTION DIAGRAM OF FAST CLOCK OSCILLATOR (CF CLOCK)

The CF clock is a multiple type oscillator (mask option) which provide a faster clock source to system. In single clock operation (fast only), this oscillator will provide the clock to the system clock generator, pre-divider, timer, I/O port chattering prevention clock and LCD circuitry. In dual clock operation, CF clock provides the clock to system clock generator only.
When the dual clock option is selected by mask option, this oscillator will be inactive most of the time except when the FAST instruction is executed. After the FAST instruction is executed, the clock source (BCLK) of the system clock generator will be switched to CF clock and the clock source for other functions will still come from XT clock. Halt mode, stop mode or SLOW instruction execution will stop this oscillator and the system clock (BCLK) will be switched to XT clock.

There are 2 type oscillators can be used in fast clock oscillator, selected by mask option:

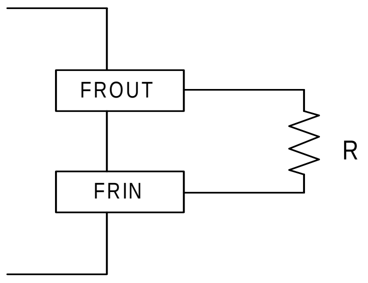
2-2-2-1. RC OSCILLATOR WITH EXTERNAL RESISTOR (CF CLOCK)

This kind of oscillator could only be used in "FAST only" option, the fast clock source of dual clock mode can't use this oscillator. When this oscillator is used, the frequency option of the RC oscillator with internal RC is not cared. If FRIN pin is unused, it must be connected to GND.

MASK OPTION table :

Mask Option name	Selected item
CLOCK SOURCE	(1) FAST ONLY or (3)DuaL

Mask Option name	Selected item
FAST CLOCK OSC TYPE FOR FAST ONLY OR DUAL	(1) EXTERNAL RESISTOR



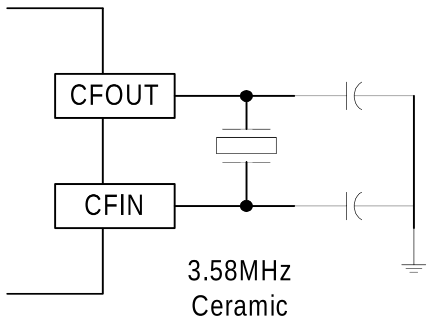
External
Resistor

2-2-2-2. External 3.58MHz Ceramic Resonator oscillator

MASK OPTION table :

Mask Option name	Selected item
CLOCK SOURCE	(1) FAST ONLY or (3) DUAL

Mask Option name	Selected item
FAST CLOCK OSC TYPE FOR FAST ONLY OR DUAL	(2) 3.58MHz CERAMIC RESONATOR



3.58MHz
Ceramic
Resonator

Notes : 1. When the program has to reset the BCF flag to 0 in Li battery power mode, don't use a 3.58MHz Ceramic Resonator as the oscillator.

2-2-3. THE COMBINATION OF THE CLOCK SOURCES

There are three types of combination of the clock sources that can be selected by mask option:

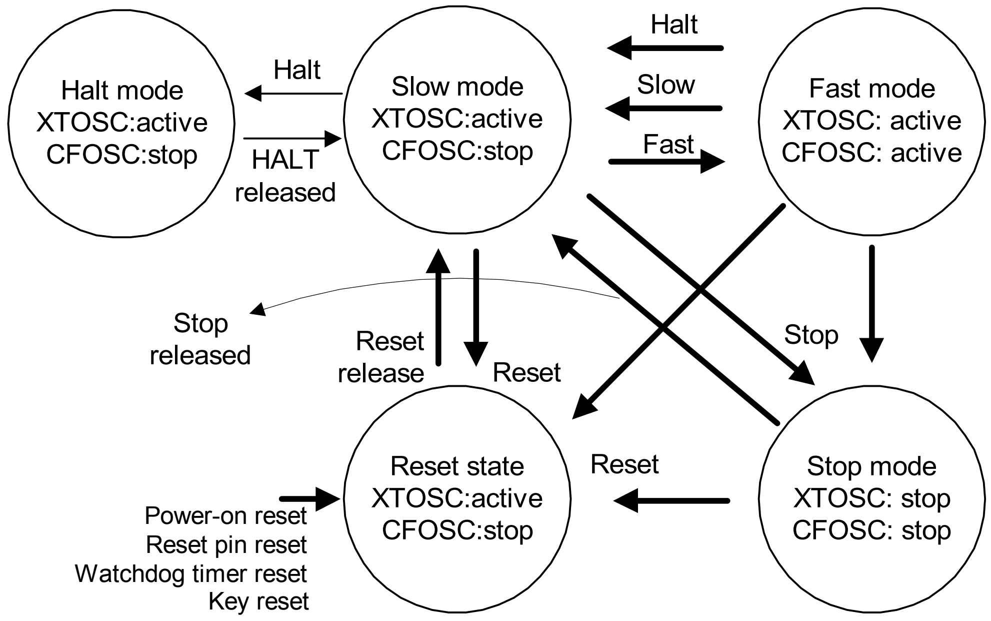
2-2-3-1 DUAL CLOCK

MASK OPTION table :

Mask Option name	Selected item
CLOCK SOURCE	(3) DUAL

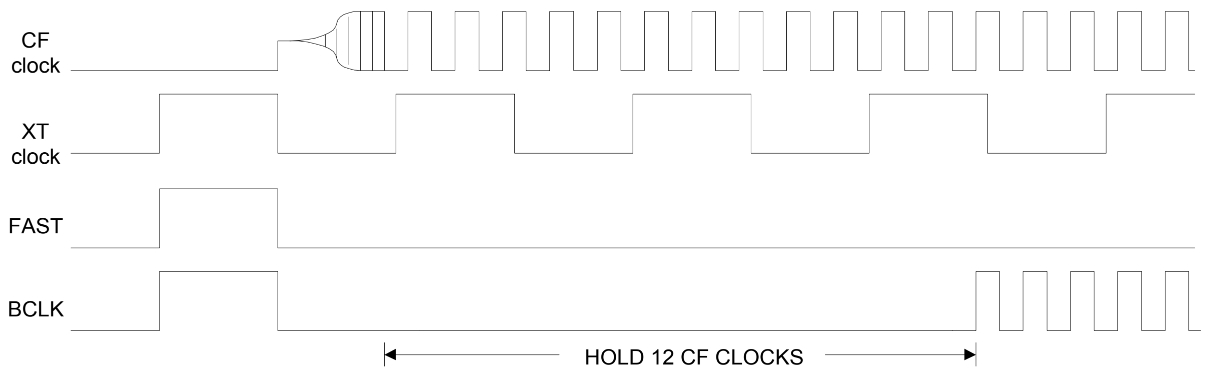
The operation of the dual clock option is shown in the following figure.
When this option is selected by mask option, the clock source (BCLK) of system clock generator will switch between XT clock and CF clock according to the user's program.

When the halt and stop instructions are executed, the clock source (BCLK) will switch to XT clock automatically.
The XT clock provides the clock to the pre-divider, timer, I/O port chattering prevention and LCD circuitry in this option.



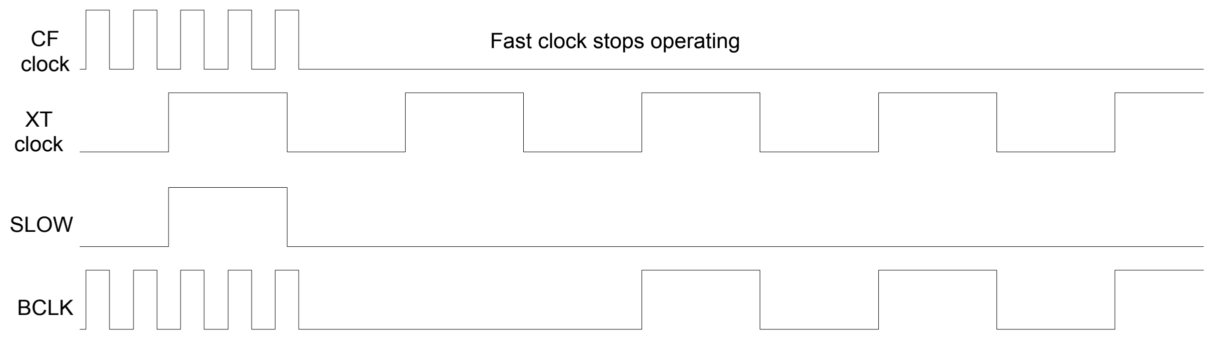
State Diagram of Dual Clock Option was shown on above figure.

After executing FAST instruction, the system clock generator will hold 12 CF clocks after the CF clock oscillator starts up and then switches CF clock to BCLK. This will prevent the incorrect clock from delivering to the system clock in the start-up duration of the fast clock oscillator.



This figure shows the System Clock Switches from Slow to Fast

After executing SLOW instruction, the system clock generator will hold 2 XT clocks and then switches XT clock to BCLK.



This figure shows the System Clock Switches from Fast to Slow

2-2-3-2 SINGLE CLOCK

MASK OPTION table :

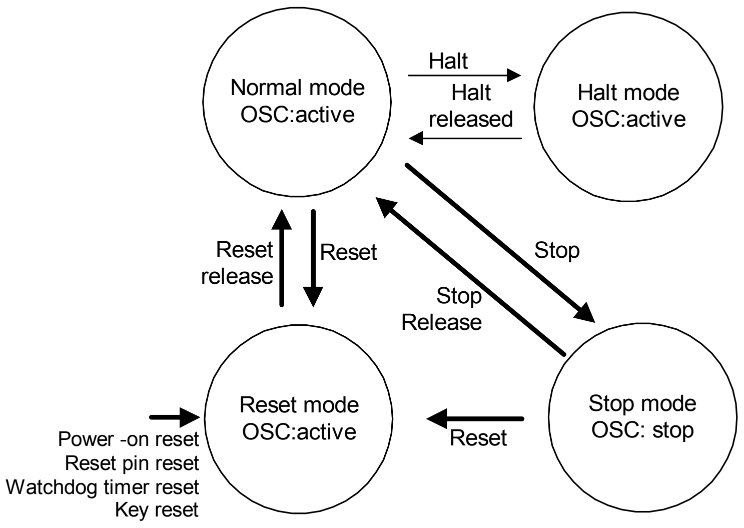
For Fast clock oscillator only

Mask Option name	Selected item
CLOCK SOURCE	(1) FAST ONLY

For slow clock oscillator only

Mask Option name	Selected item
CLOCK SOURCE	(2) SLOW ONLY

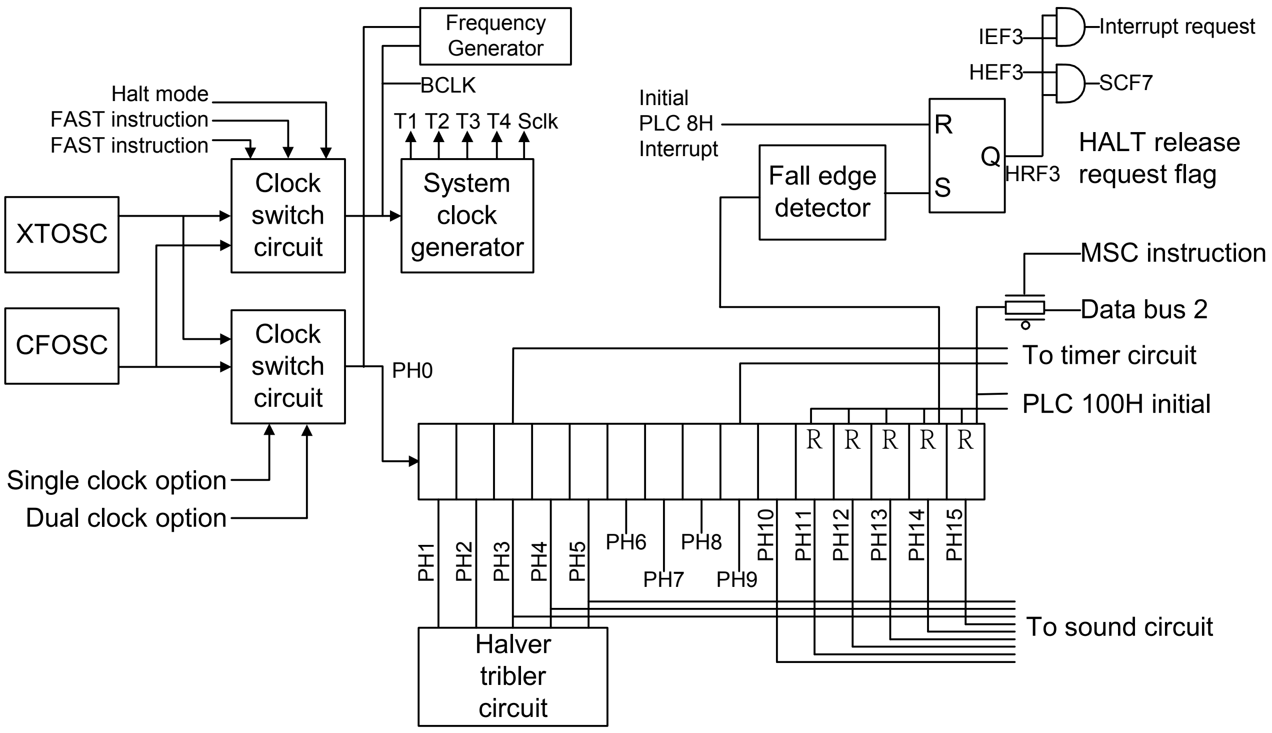
The operation of the single clock option is shown in the following figure.
Either XT or CF clock may be selected by mask option in this mode. The FAST and SLOW instructions will perform as the NOP instruction in this option.
The backup flag (BCF) will be set to 1 automatically before the program enters the stop mode. This could ensure the Crystal oscillator would start up in a better condition.



This figure shows the State Diagram of Single Clock Option

2-2-4 PREDIVIDER

The pre-divider is a 15-stage counter that receives the clock from the output of clock switch circuitry (PH0) as input. When PH0 is changed from "H" level to "L" level, the content of this counter changes. The PH11 to PH15 of the pre-divider are reset to "0" when the PLC 100H instruction is executed or at the initial reset mode. The pre-divider delivers the signal to the halver / tripler circuit, alternating frequency for LCD display, system clock, sound generator and halt release request signal (I/O port chattering prevention clock).



This figure shows the Pre-divider and its Peripherals

The PH14 delivers the halt mode release request signal, setting the halt mode release request flag (HRF3). In this case, if the pre-divider interrupt enable mode (IEF3) is provided, the interrupt is accepted; and if the halt release enable mode (HEF3) is provided, the halt release request signal is delivered, setting the start condition flag 7 (SCF7) in status register 3 (STS3).

The clock source of pre-divider is PH0, and 4 kinds of frequency of PH0 could be selected by mask option :

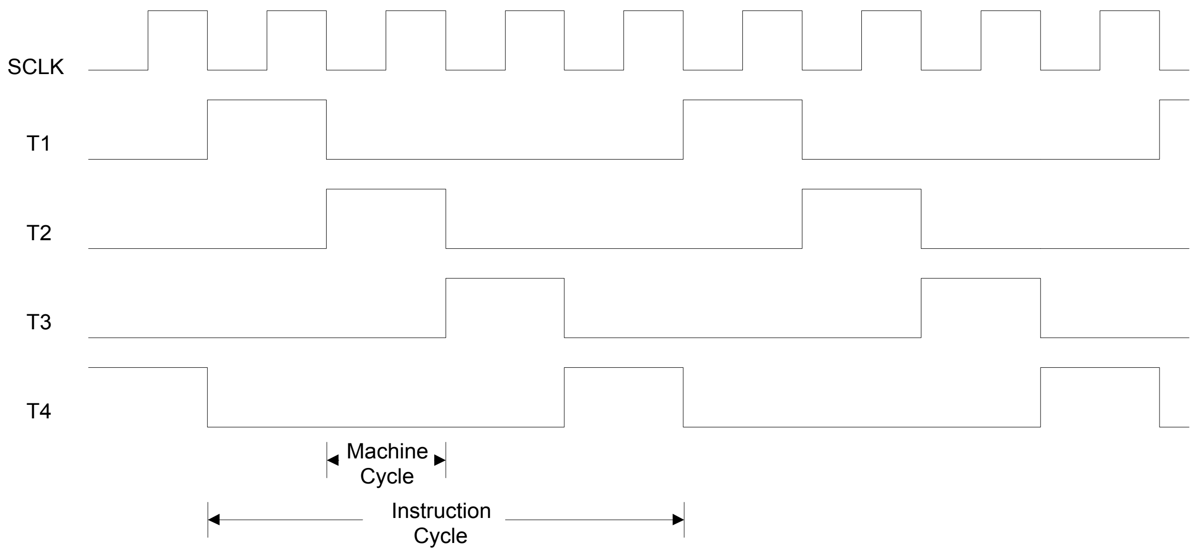
MASK OPTION table :

Mask Option name	Selected item
PH0 <-> BCLK FOR FAST ONLY	(1) PH0 = BCLK
PH0 <-> BCLK FOR FAST ONLY	(2) PH0 = BCLK/4

PH0 <-> BCLK FOR FAST ONLY	(3) PH0 = BCLK/8
PH0 <-> BCLK FOR FAST ONLY	(4) PH0 = BCLK/16

2-2-5 SYSTEM CLOCK GENERATOR

For the system clock, the clock switch circuit permits the different clock input from XTOSC and CFOSC to be selected. The FAST and SLOW instructions can switch the clock input of the system clock generator (SGC).
The basic system clock is shown below:



2-3 PROGRAM COUNTER (PC)

This is an 11-bit counter, which addresses the program memory (ROM) up to 2048 addresses.

- The program counter (PC) is normally increased by one (+1) with every instruction execution.
  PC ← PC + 1
- When executing JMP instruction, subroutine call instruction (CALL), interrupt service routine or reset occurs, the program counter (PC) loads the specified address corresponding to table 2-1.
  PC ← specified address shows in
- When executing a jump instruction except JMP and CALL, the program counter (PC) loads the specified address in the operand of instruction.
  PC ← current page (PC11) + specified address in operand
- Return instruction (RTS)
  PC ← content of stack specified by the stack pointer
  Stack pointer ← stack pointer - 1

Table 2- 1

	PC11	PC10	PC9	PC8	PC7	PC6	PC5	PC4	PC3	PC2	PC1	PC0
Initial reset	0	0	0	0	0	0	0	0	0	0	0	0
Interrupt 2 (INT pin)	0	0	0	0	0	0	0	1	0	0	0	0
Interrupt 0 (input port C & D)	0	0	0	0	0	0	0	1	0	1	0	0
Interrupt 1 (timer 1 interrupt)	0	0	0	0	0	0	0	1	1	0	0	0
Interrupt 3 (pre-divider interrupt)	0	0	0	0	0	0	0	1	1	1	0	0
Interrupt 4 (timer 2 interrupt)	0	0	0	0	0	0	1	0	0	0	0	0
Interrupt 5 (Key Scanning interrupt)	0	0	0	0	0	0	1	0	0	1	0	0
Interrupt 6 (RFC counter interrupt)	0	0	0	0	0	0	1	0	1	0	0	0
Jump instruction	P11	P10	P9	P8	P7	P6	P5	P4	P3	P2	P1	P0
Subroutine call	P11	P10	P9	P8	P7	P6	P5	P4	P3	P2	P1	P0

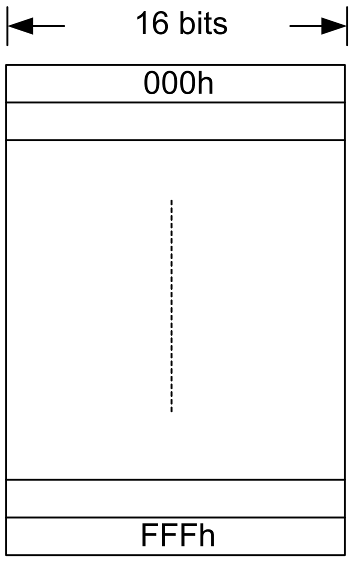
P10 to P0 : Low-order 11 bits of instruction operand.

When executing the subroutine call instruction or interrupt service routine, the contents of the program counter (PC) are automatically saved to the stack register (STACK).

2-4 PROGRAM/TABLE MEMORY (ROM)

The built-in mask ROM is organized with 4096 x 16 bits.
The partition formula for PROM and TROM is shown below :
I

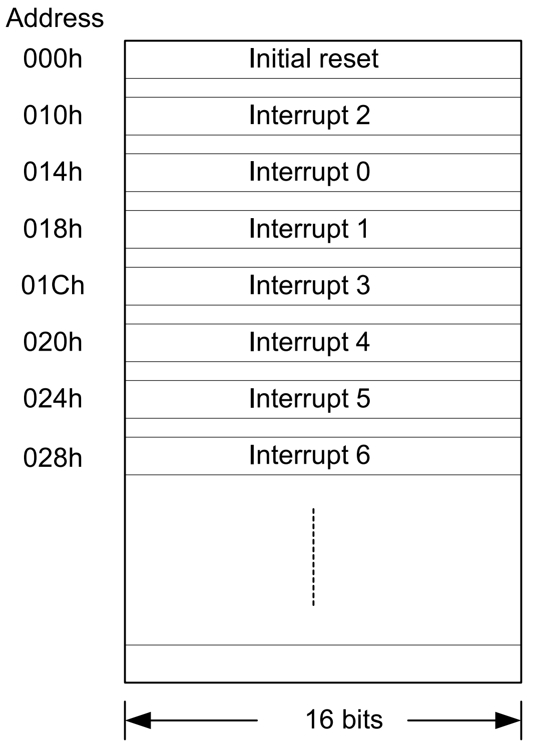
Note : The data width of table ROM is 8-bit



The partition of memory space is defined by mask option, the table is shown below :

2-4-1. INSTRUCTION ROM (PROM)

There are some special locations that serve as the interrupt service routines, such as reset address (000H), interrupt 0 address (014H), interrupt 1 address (018H), interrupt 2 address (010H), interrupt 3 address (01CH), interrupt 4 address (020H), interrupt 5 address (024H), and interrupt 6 address (028H) in the program memory.



Instruction ROM (PROM) organization

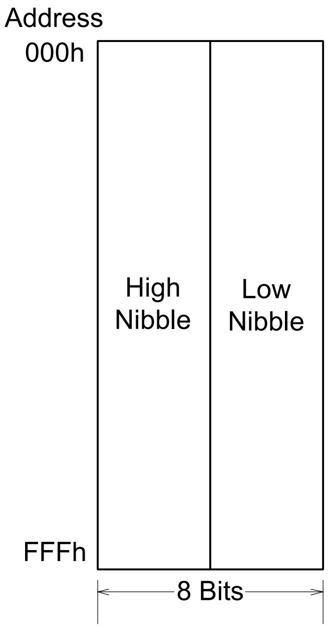


Table ROM (TROM) organization

This figure shows the Organization of ROM

2-4-2. TABLE ROM (TROM)

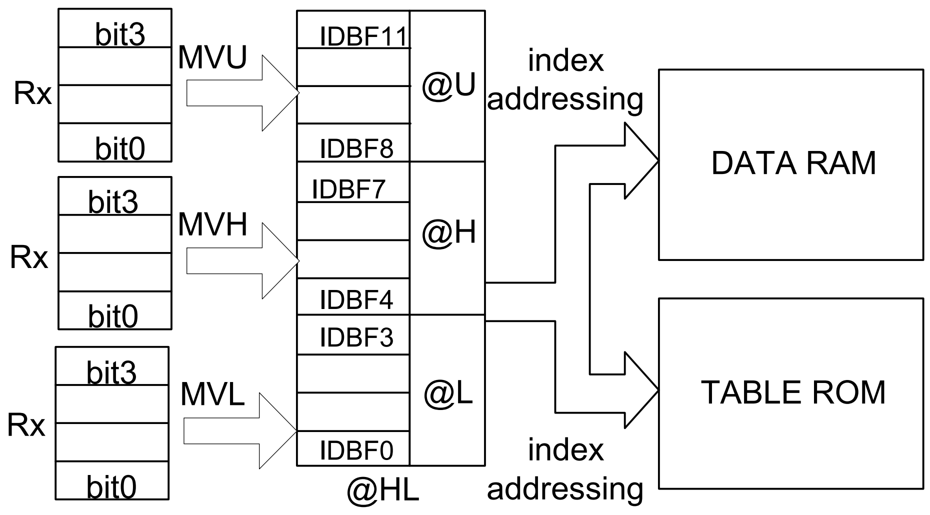
This memory space stores the constant data or look up table for the usage of main program. All of the table ROM addresses are specified by the index address register (@HL). The data width could be 8 bits or 4 bits which depends on the different usage. Refer to the explanation of instruction chapter.

2-5 INDEX ADDRESS REGISTER (@HL)

This is a versatile address pointer for the data memory (RAM) and table ROM (TROM). The index address register (@HL) is a 12-bit register, and the contents of the register can be modified by executing MVU, MVH and MVL instructions. Executed MVL instruction will load the content of specified data memory to the lower nibble of the index register (@L). In the same manner, executed MVH and MVU instructions will load the contents of the data RAM (Rx) to the higher nibble of the register @H and @U, respectively.

@U register				@H register				@L register			
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	Bit3	Bit2	Bit1	Bit0
IDBF11	IDBF10	IDBF9	IDBF8	IDBF7	IDBF6	IDBF5	IDBF4	IDBF3	IDBF2	IDBF1	IDBF0

The index address register can specify the full range addresses of the table ROM and data memory.



This figure shows the diagram of the index address register

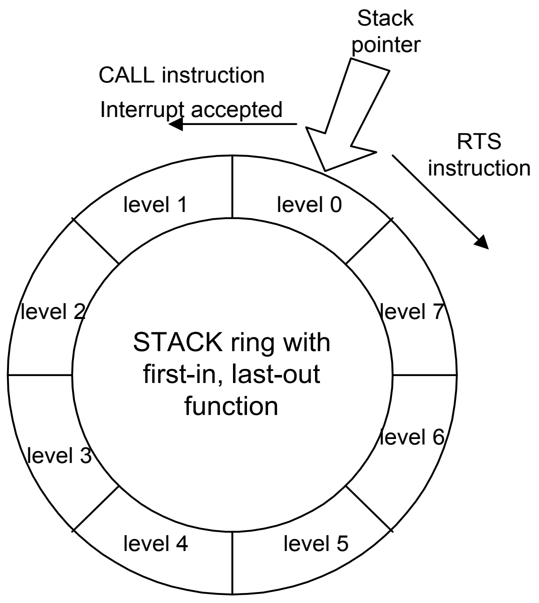
2-6 STACK REGISTER (STACK)

Stack is a special design register following the first-in-last-out rule. It is used to save the contents of the program counter sequentially during subroutine call or execution of the interrupt service routine.
The contents of stack register are returned sequentially to the program counter (PC) while executing return instructions (RTS).

The stack register is organized using 11 bits by 8 levels but with no overflow flag; hence only 8 levels of subroutine call or interrupt are allowed (If the stacks are full, and either interrupt occurs or subroutine call executes, the first level will be overwritten).

Once the subroutine call or interrupt causes the stack register (STACK) overflow, the stack pointer will return to 0 and the content of the level 0 stack will be overwritten by the PC value.
The contents of the stack register (STACK) are returned sequentially to the program counter (PC) during execution of the RTS instruction.
Once the RTS instruction causes the stack register (STACK) underflow, the stack pointer will return to level 7 and the content of the level 7 stack will be restored to the program counter.

The following figure shows the diagram of the stack.



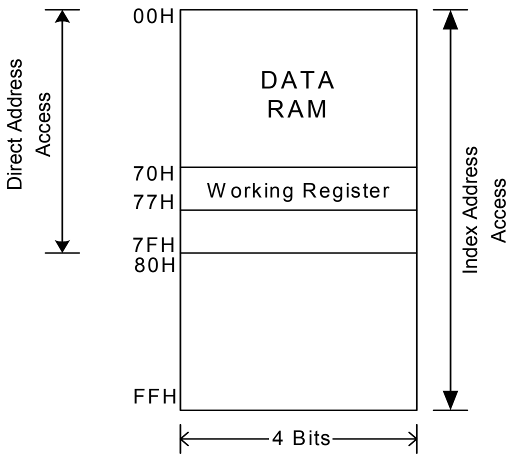
2-7 DATA MEMORY (RAM)

The static RAM is organized with 256 addresses x 4 bits and is used to store data.

The data memory may be accessed using two methods:

1. Direct addressing mode
   The address of the data memory is specified by the instruction and the addressing range is from 00H to 7FH.
2. Index addressing mode
   The index address register (@HL) specifies the address of the data memory and all address space from 00H to FFH can be accessed.

The 8 specified addresses (70H to 77H) in the direct addressing memory are also used as 8 working registers. The function of working register will be described in detail in section 2-8.



This figure shows the Data Memory (RAM) and Working Register Organization

2-8 WORKING REGISTER (WR)

The locations 70H to 77H of the data memory (RAM) are not only used as general-purpose data memory but also as the working register (WR). The following will introduce the general usage of working registers:

1. Be used to perform operations on the contents of the working register and immediate data. Such as :
   ADCI, ADCI*, SBCI, SBCI*, ADDI, ADDI*, SUBI, SUBI*, ADNI, ADNI*, ANDI, ANDI*, EORI, EORI*, ORI, ORI*

2. Be transferred the data between the working register and any address in the direct addressing data memory (RAM). Such as :
   MWR Rx, Ry; MRW Ry, Rx

3. Decode (or directly transfer) the contents of the working register and output to the LCD PLA circuit. Such as :
   LCT, LCB, LCP

2-9 ACCUMULATOR (AC)

The accumulator (AC) is a register that plays the most important role in operations and controls. By using it in conjunction with the ALU (Arithmetic and Logic Unit), data transfer between the accumulator and other registers or data memory can be performed.

2-10 ALU (Arithmetic and Logic Unit)

This is a circuitry that performs arithmetic and logic operation. The ALU provides the following functions:

Binary addition/subtraction	(INC, DEC, ADC, SBC, ADD, SUB, ADN, ADCI, SBUI, ADNI)
Logic operation	(AND, EOR, OR, ANDI, EORI, ORI)
Shift	(SR0, SR1, SL0, SL1)
Decision	(JB0, JB1, JB2, JB3, JC, JNC, JZ, and JNZ)
BCD operation	(DAA, DAS)

2-11 HEXADECIMAL CONVERT TO DECIMAL (HCD)

Decimal format is another number format for TM87P08. When the content of the data memory has been assigned as decimal format, it is necessary to convert the results to decimal format after the execution of ALU instructions. When the decimal converting operation is processing, all of the operand data (including the contents of the data memory (RAM), accumulator (AC), immediate data, and look-up table) should be in the decimal format, or the results of conversion will be incorrect.

Instructions DAA, DAA*, DAA @HL can convert the data from hexadecimal to decimal format after any addition operation. The conversion rules are shown in the following table and illustrated in example 1.

AC data before DAA execution	CF data before DAA execution	AC data after DAA execution	CF data after DAA execution
$0 \leq AC \leq 9$	CF = 0	no change	no change
$A \leq AC \leq F$	CF = 0	AC= AC+ 6	CF = 1
$0 \leq AC \leq 3$	CF = 1	AC= AC+ 6	no change

Example 1:

```
        LDS   10h, 9        ; Load immediate data"9"to data memory address 10H.
        LDS   11h, 1        ; Load immediate data"1"to data memory address 11H
                            ; and AC.
        RF    1h            ; Reset CF to 0.
        ADD*  10h           ; Contents of the data memory address 10H and AC are
                            ; binary-added; the result loads to AC & data memory address
                            ; 10H. (R10 = AC = AH, CF = 0)
        DAA*  10h           ; Convert the content of AC to
                            ; decimal format.
                            ; The result in the data memory address 10H is"0"and in
                            ; the CF is "1". This represents the decimal number"10".
```

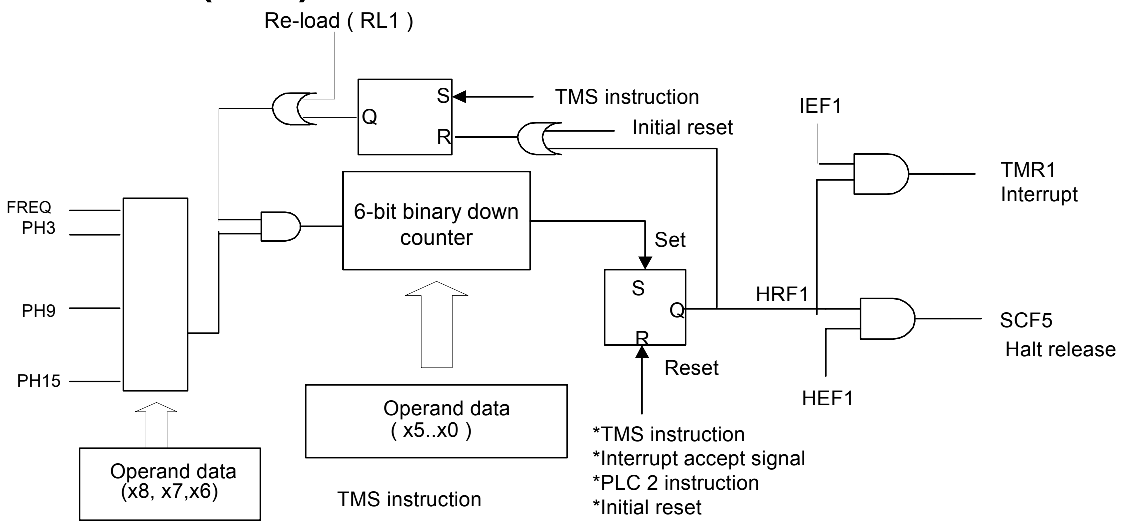
Instructions DAS, DAS*, DAS @HL can convert the data from hexadecimal format to decimal format after any subtraction operation. The conversion rules are shown in the following table and illustrated in Example 2.

AC data before DAS execution	CF data before DAS execution	AC data after DAS execution	CF data after DAS execution
$0 \leq AC \leq 9$	CF = 1	No change	no change
$6 \leq AC \leq F$	CF = 0	AC= AC+A	no change

Example 2:

```
   LDS10h, 1        ; Load immediate data"1"to the data memory address 10H.
   LDS11h, 2        ; Load immediate data"2"to the data memory address 11H and AC.
   SF   1h          ; Set CF to 1, which means no borrowing has occurred.
   SUB*       10h   ; Content of data memory address 10H is binary-subtracted;
                    ; the result loads to data memory address
                    ; 10H. (R10 = AC = FH, CF = 0)
   DAS*       10h   ; Convert the content of the data memory address 10H to decimal
                       format.
                    ; The result in the data memory address 10H is"9"and in
                    ; the CF is "0". This represents the decimal number"–1".
```

2-12 TIMER 1 (TMR1)



This figure shows the TMR1 organization.

2-12-1 NORMAL OPERATION

TMR1 consists of a programmable 6-bit binary down counter, which is loaded and enabled by executing TMS or TMSX instruction.
Once the TMR1 counts down to 3Fh, it generates an underflow signal to set the halt release request flag1 (HRF1) to 1 and then stop to count down.
When HRF1 = 1, and the TMR1 interrupt enable flag (IEF1) = 1, the interrupt is generated.
When HRF1 = 1, if the IEF1 = 0 and the TMR1 halt release enable (HEF1) = 1, program will escapes from halt mode (if CPU is in halt mode) and then set the start condition flag 5 (SCF5) to 1 in the status register 3 (STS3).
After power on reset, the default clock source of TMR1 is PH3.
If watchdog reset occurred, the clock source of TMR1 will still keep the previous selection.

The following table shows the definition of each bit in TMR1 instructions

OPCODE	Select clock			Initiate value of timer					
TMSX X	X8	X7	X6	X5	X4	X3	X2	X1	X0
TMS Rx	0	AC3	AC2	AC1	AC0	Rx3	Rx2	Rx1	Rx0
TMS @HL	0	bit7	bit6	bit5	Bit4	bit3	bit2	bit1	bit0

The following table shows the clock source setting for TMR1.

X8	X7	X6	clock source
0	0	0	PH9
0	0	1	PH3
0	1	0	PH15
0	1	1	FREQ
1	0	0	PH5
1	0	1	PH7
1	1	0	PH11
1	1	1	PH13

Notes:

1. When the TMR1 clock is PH3
   TMR1 set time = (Set value + error) * 8 * 1/fosc (KHz) (ms)
2. When the TMR1 clock is PH9
   TMR1 set time = (Set value + error) * 512 * 1/fosc (KHz) (ms)
3. When the TMR1 clock is PH15
   TMR1 set time = (Set value + error) * 32768 * 1/fosc (KHz) (ms)
4. When the TMR1 clock is PH5
   TMR1 set time = (Set value + error) * 32 * 1/fosc (KHz) (ms)
5. When the TMR1 clock is PH7
   TMR1 set time = (Set value + error) * 128 * 1/fosc (KHz) (ms)
6. When the TMR1 clock is PH11
   TMR1 set time = (Set value + error) * 2048 * 1/fosc (KHz) (ms)
7. When the TMR1 clock is PH13
   TMR1 set time = (Set value + error) * 8192 * 1/fosc (KHz) (ms)

Set value: Decimal number of timer set value

error: the tolerance of set value, $0 < \text{error} < 1$.

fosc:	Input of the predivider
PH3:	The 3rd stage output of the predivider
PH5:	The 5th stage output of the predivider
PH7:	The 7th stage output of the predivider
PH9:	The 9th stage output of the predivider
PH11:	The 11th stage output of the predivider
PH13:	The 13th stage output of the predivider
PH15:	The 15th stage output of the predivider

8. When the TMR1 clock is FREQ
   TMR1 set time = (Set value + error) * 1/FREQ (KHz) (ms).

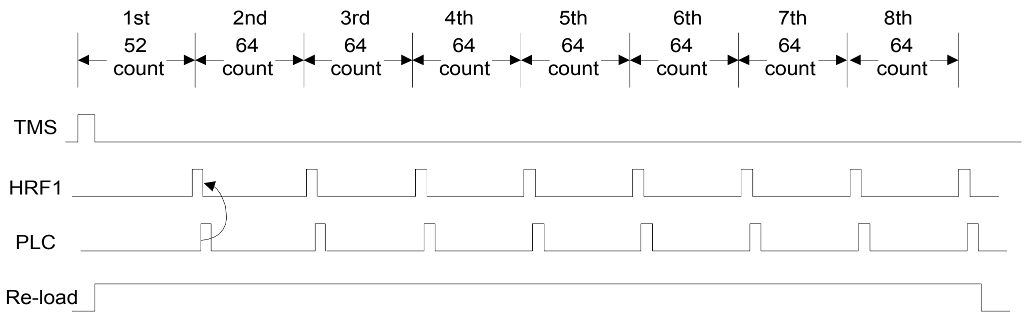
FREQ: refer to section 3-3-4.

2-12-2 RE-LOAD OPERATION

TMR1 provides the re-load function which can extend any time interval greater than 3Fh. The SF 80h instruction enables the re-load function and RF 80h instruction disables it. When the re-load function is enabled, the TMR1 will not stop counting until the re-load function is disabled and TMR1 underflows again. During this operation, the program must use the halt release request flag or interrupt to check the wanted counting value.

- It is necessary to execute the TMS or TMSX instruction to set the down count value before the re-load function is enabled, because TMR1 will automatically count down with an unknown value once the re-load function is enabled.
- Never disable the re-load function before the last expected halt release or interrupt occurs. If TMS related instructions are not executed after each halt release or interrupt occurs, the TMR1 will stop operating immediately after the re-load function is disabled.

For example, if the expected count down value is 500, it may be divided as 52 + 7 * 64. First, set the initiate count down value of TMR1 to 52 and start counting, then enable the TMR1 halt release or interrupt function. Before the first time underflow occurs, enable the re-load function. The TMR1 will continue operating even though TMR1 underflow occurs. When halt release or interrupt occurs, clear the HRF1 flag by PLC instruction. After halt release or interrupt occurs 8 times, disable the re-load function and the counting is completed.

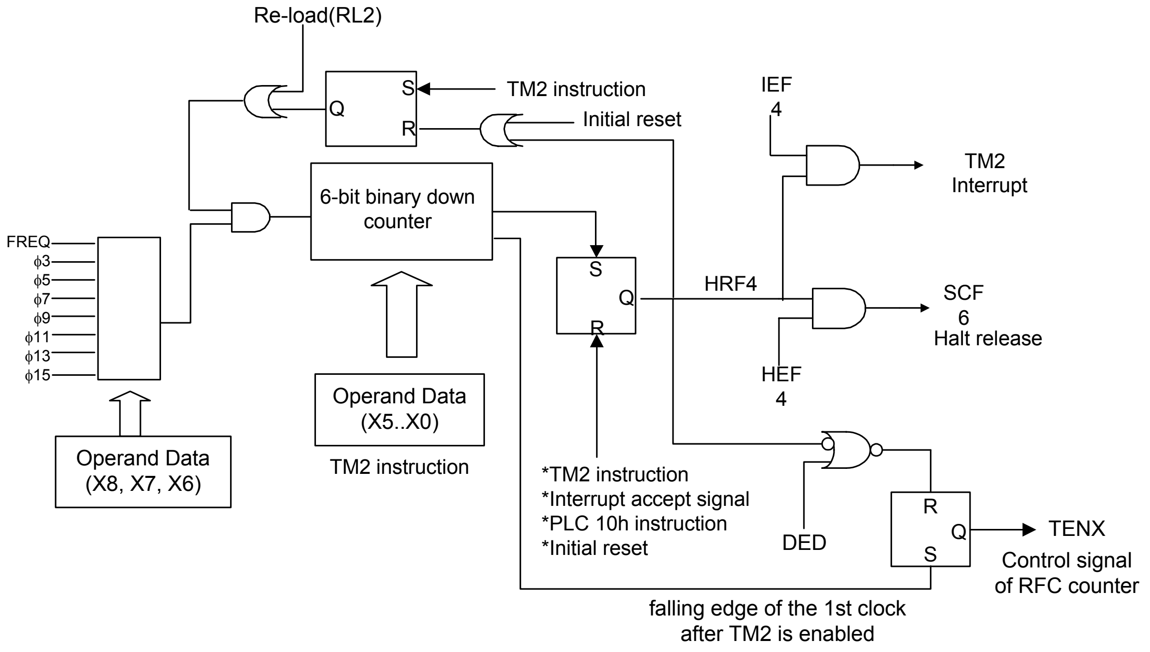


In the following example, S/W enters the halt mode to wait for the underflow of TMR1.

```
            LDS   0, 0          ;initiate the underflow counting register
            PLC   2
            SHE   2             ;enable the HALT release caused by TMR1
            TMSX  34h           ;initiate the TMR1 value (52) and clock source is ϕ9
            SF    80h           ;enable the re-load function
RE_LOAD:
            HALT
            INC*  0             ;increase the underflow counter
            PLC   2             ;clear HRF1
            JB3   END_TM1       ;if the TMR1 underflow counter is equal to 8, exit subroutine
            JMP   RE_LOAD
END_TM1:
            RF    80h           ;disable the re-load function
```

2-13 TIMER 2 (TMR2)

The following figure shows the TMR2 organization.



2-13-1 NORMAL OPERATION

TMR2 consists of a programmable 6-bit binary down counter, which is loaded and enabled by executing TM2 or TM2X instruction.

Once the TMR2 counts down to 3Fh, it stops counting, then generates an underflow signal and the halt release request flag 4 (HRF4) will be set to 1.

. When HRF4 = 1, and the TMR2 interrupt enabler (IEF4) is set to 1, the interrupt occurred.

. When HRF4 =1, IEF4 = 0, and the TMR2 halt release enabler (HEF4) is set to 1, program will escapes from halt mode (if CPU is in halt mode) and then HRF4 sets the start condition flag 6 (SCF6) to 1 in the status register 4 (STS4).

After power on reset, the default clock source of TMR2 is PH7.

If watchdog reset occurred, the clock source of TMR2 will still keep the previous selection.

The following table shows the definition of each bit in TMR2 instructions

OPCODE	Select clock			Initiate value of timer					
TM2X X	X8	X7	X6	X5	X4	X3	X2	X1	X0
TM2 Rx	0	AC3	AC2	AC1	AC0	Rx3	Rx2	Rx1	Rx0
TM2 @HL	0	bit7	bit6	bit5	Bit4	bit3	bit2	bit1	bit0

The following table shows the clock source setting for TMR2

X8	X7	X6	clock source
0	0	0	PH9
0	0	1	PH3
0	1	0	PH15
0	1	1	FREQ
1	0	0	PH5
1	0	1	PH7
1	1	0	PH11
1	1	1	PH13

Notes:

1. When the TMR2 clock is PH3
   TMR2 set time = (Set value + error) * 8 * 1/fosc (KHz) (ms)
2. When the TMR2 clock is PH9
   TMR2 set time = (Set value + error) * 512 * 1/fosc (KHz) (ms)
3. When the TMR2 clock is PH15
   TMR2 set time = (Set value + error) * 32768 * 1/fosc (KHz) (ms)
4. When the TMR2 clock is PH5
   TMR2 set time = (Set value + error) * 32 * 1/fosc (KHz) (ms)
5. When the timer clock is PH7
   TMR2 set time = (Set value + error) * 128 * 1/fosc (KHz) (ms)
6. When the TMR2 clock is PH11
   TMR2 set time = (Set value + error) * 2048 * 1/fosc (KHz) (ms)
7. When the TMR2 clock is PH13
   TMR2 set time = (Set value + error) * 8192 * 1/fosc (KHz) (ms)

Set value: Decimal number of timer set value

error: the tolerance of set value, 0 < error <1.

fosc:	Input of the predivider
PH3:	The 3rd stage output of the predivider
PHn:	The nth stage output of the predivider (n=5,7,9,11,13)

8. When the TMR2 clock is FREQ
   TMR2 set time = (Set value + error) * 1/FREQ (KHz) (ms).

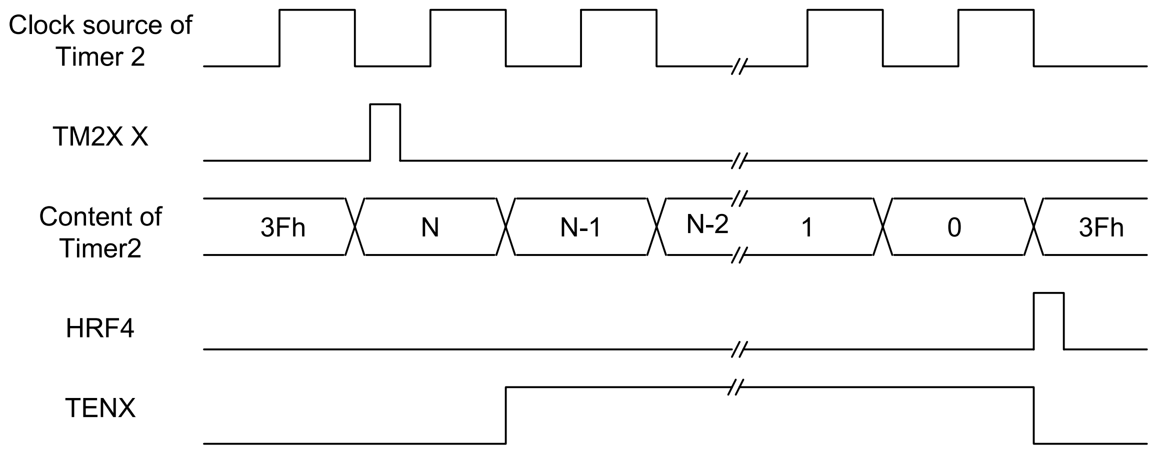
FREQ: refer to section 3-3-4.

2-13-2 RE-LOAD OPERATION

TMR2 also provides the re-load function is the same as TMR1. The instruction SF2 1 enables the re-load function; the instruction RF2 1 disables it.

2-13-3 TIMER 2 (TMR2) IN RESISTOR TO FREQUENCY CONVERTER (RFC)

TMR2 also controlled the operation of RFC function.
TMR2 will set TENX flag to 1 to enable the RFC counter; once the TMR2 underflows, the TENX flag will be reset to 0 automatically. In this case, Timer 2 could set an accurate time period without setting a value error like the other operations of TMR1 and TMR2. Refer to 2-16 for detailed information on controlling the RFC counter. The following figure shows the operating timing of TMR 2 in RFC mode.



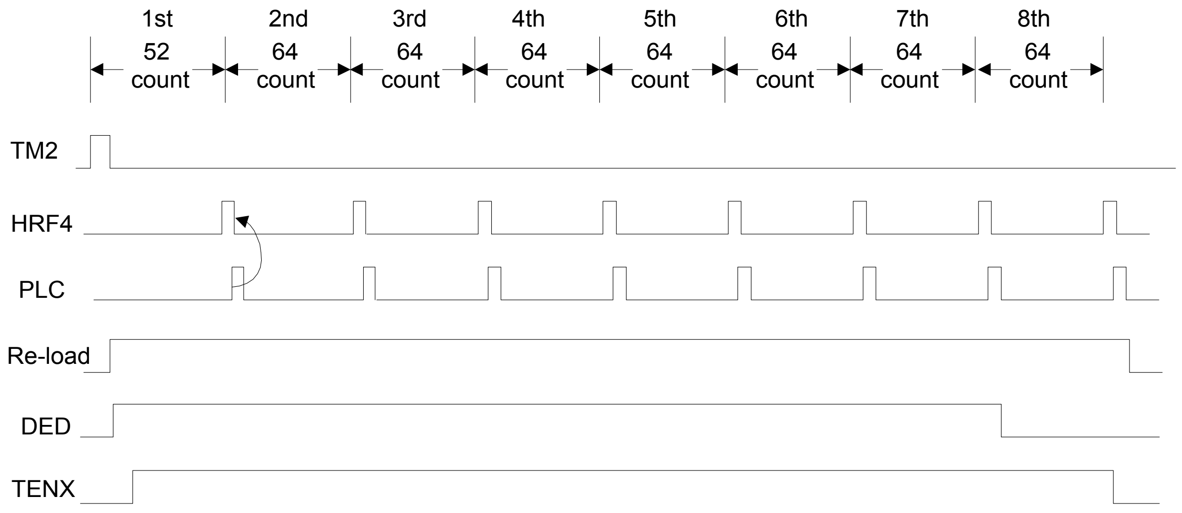
TMR2 also provides the re-load function when controlled the RFC function.
The SF2 1h instruction enables the re-load function, and the DED flag should be set to 1 by SF2 2h instruction. Once DED flag had been set to 1, TENX flag will not be cleared to 0 while TMR2 underflows (but HRF4 will be set to1). The DED flag must be cleared to 0 by executing RF2 2h instruction before the last HRF4 occurs; thus, the TENX flag will be reset to 0 when the last HRF4 flag delivery. After the last underflow (HRF4) of TMR2 occurred, disable the re-load function by executing RF2 1h instruction.

For example, if the target set value is 500, it will be divided as 52 + 7 * 64.
1. Set the initiate value of TMR2 to 52 and start counting.
2. Enable the TMR2 halt release or interrupt function.
3. Before the first underflow occurs, enable the re-load function and set the DED flag. The TMR2 will continue counting even if TMR2 underflows.
4. When halt release or interrupt occurs, clear the HRF4 flag by PLC instruction and increase the counting value to count the underflow times.
5. When halt release or interrupt occurs for the 7th time, reset the DED flag.
6. When halt release or interrupt occurs for the 8th time, disable the re-load function and the counting is completed.

In the following example, S/W enters the halt mode to wait for the underflow of TM2

```
LDS    0,0              ;initiate the underflow counting register
PLC    10h
```

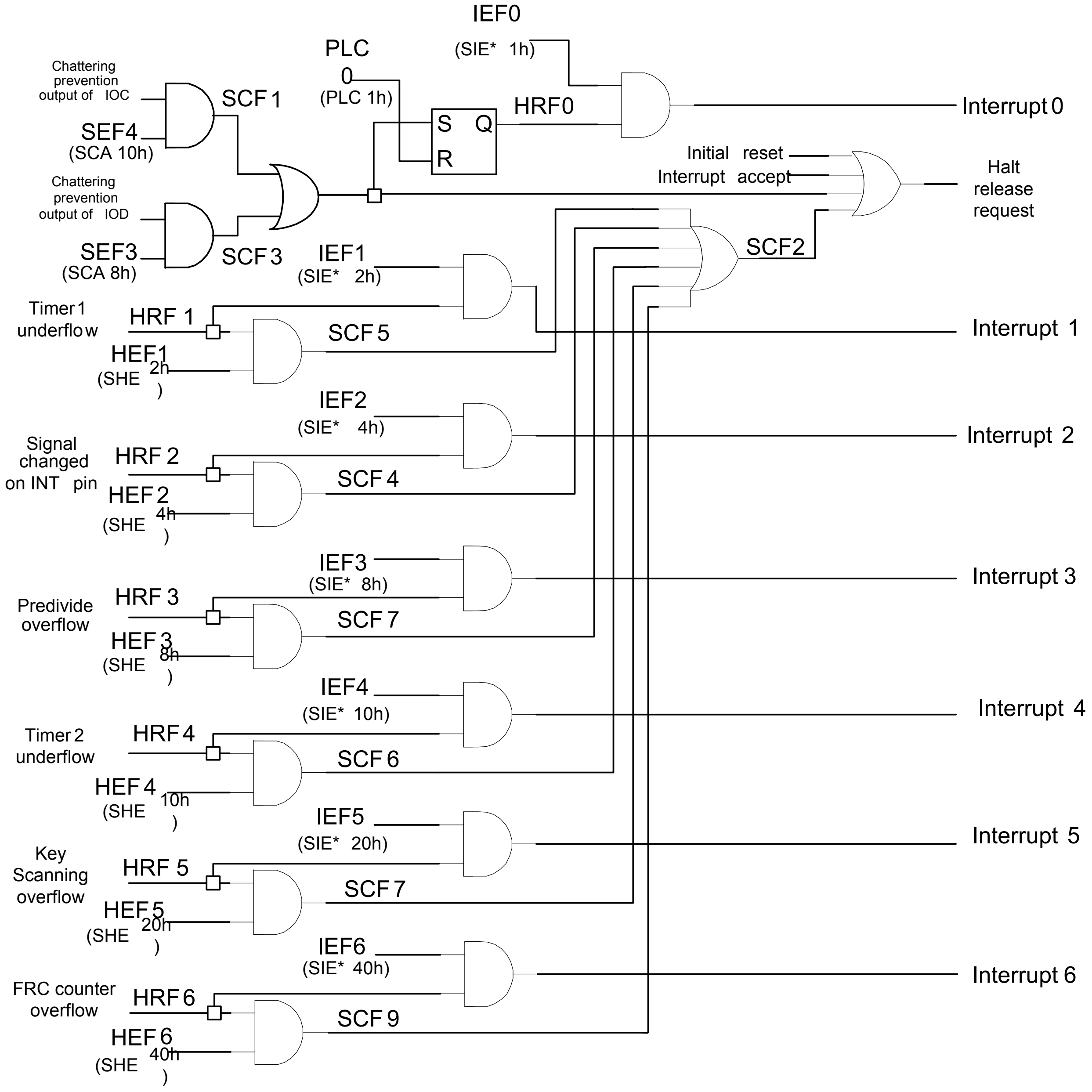
```
            SHE   10h          ;enable the halt release caused by TM2
            SRF   19h          ;enable RFC, and controlled by TM2
            TM2X 34h           ;initiate the TM value(52) and clock source is φ9
            SF2   3h           ;enable the re-load function and set DED flag to 1
RE_LOAD:
            HALT
            INC*  0            ;increase the underflow counter
            PLC   10h          ;clear HRF4
            LDS   20h, 7
            SUB   0            ;when halt is released for the 7th time, reset DED flag
            JNZ   NOT_RESET_DED
            RF2   2            ;reset DED flag
NOT_RESET_DED:
            LDA   0            ;store underflow counter to AC
            JB3   END_TM1      ;if the TM2 underflow counter is equal to 8, exit this
subroutine
            JMP   RE_LOAD
END_TM1:
            RF2   1            ;disable the re-load function
```



This figure shows the operating timing of TMR2 re-load function for RFC

2-14 STATUS REGISTER (STS)

The status register (STS) is organized with 4 bits and comes in 4 types: status register 1 (STS1) to status register 4 (STS4). The following figure shows the configuration of the start condition flags for TM87P08.



2-14-1 STATUS REGISTER 1 (STS1)

Status register 1 (STS1) consists of 2 flags:

1. Carry flag (CF)
   The carry flag is used to save the result of the carry or borrow during the arithmetic operation.
2. Zero flag(Z)
   Indicates the accumulator (AC) status. When the content of the accumulator is 0, the Zero flag is set to 1. If the content of the accumulator is not 0, the zero flag is reset to 0.
3. The MAF instruction can be used to transfer data in status register 1 (STS1) to the accumulator (AC) and the data memory (RAM).
4. The MRA instruction can be used to transfer data of the data memory (RAM) to the status register 1 (STS1).

The bit pattern of status register 1 (STS1) is shown below.

Bit 3	Bit 2	Bit 1	Bit0
Carry flag (AC)	Zero flag (Z)	NA	NA
Read / write	Read only	Read only	Read only

2-14-2 STATUS REGISTER 2 (STS2)

Status register 2 (STS2) consists of start condition flag 1, 2, 3 (SCF1, SCF2, SCF3) and the backup flag.The MSB instruction can be used to transfer data of status register 2 (STS2) to the accumulator (AC) and the data memory (RAM), but it is impossible to transfer data of the data memory (RAM) to status register 2 (STS2).

The following table shows the bit pattern of each flag in status register 2 (STS2).

Bit 3	Bit 2	Bit 1	Bit 0
Start condition flag 3 (SCF3)	Start condition flag 2 (SCF2)	Start condition flag 1 (SCF1)	Backup flag (BCF)
Halt release caused by the IOD port	Halt release caused by SCF4,5,6,7,9	Halt release caused by the IOC port	The backup mode status
Read only	Read only	Read only	Read only

Start condition flag 1 (SCF1)
When the SCA instruction specified signal change occurs at port IOC to release the halt mode, SCF1 will be set. Executing the SCA instruction will cause SCF1 to be reset to 0
Start condition flag 2 (SCF2)
When a factor other than port IOC causes the halt mode to be released, SCF2 will be set to1. In this case, if one or more start condition flags in SCF4, 5, 6, 7, 9 is set to 1, SCF2 will also be set to 1 simultaneously. When all of the flags in SCF4, 5, 6, 7, 9 are clear, start condition flag 2 (SCF2) is reset to 0.

Note: If start condition flag is set to 1, the program will not be able to enter halt mode.

Start condition flag 3 (SCF3)

When the SCA instruction specified signal change occurs at port IOD to release the halt mode, SCF3 will be set. Executing the SCA instruction will cause SCF3 to be reset to 0.

Backup flag (BCF)

This flag could be set / reset by executing the SF 2h / RF 2h instruction.

2-14-3 STATUS REGISTER 3 (STS3)

When the halt mode is released caused by the start condition flag 2 (SCF2), status register 3 (STS3) will store the status of the factor in the release of the halt mode.

Status register 3 (STS3) consists of 4 flags:

1. Start condition flag 4 (SCF4)

   Start condition flag 4 (SCF4) is set to 1 when the signal change at the INT pin causes the halt release request flag 2 (HRF2) to be outputted and the halt release enable flag 2 (HEF2) is set beforehand. To reset start condition flag 4 (SCF4), the PLC instruction must be used to reset the halt release request flag 2 (HRF2) or the SHE instruction must be used to reset the halt release enable flag 2 (HEF2).

2. Start condition flag 5 (SCF5)

   Start condition flag 5 (SCF5) is set when an underflow signal from Timer 1 (TMR1) causes the halt release request flag 1 (HRF1) to be outputted and the halt release enable flag 1 (HEF1) is set beforehand. To reset start condition flag 5 (SCF5), the PLC instruction must be used to reset the halt release request flag 1 (HRF1) or the SHE instruction must be used to reset the halt release enable flag 1 (HEF1).

3. Start condition flag 7 (SCF7)

   Start condition flag 7 (SCF7) is set when an overflow signal from the pre-divider causes the halt release request flag 3 (HRF3) to be outputted and the halt release enable flag 3 (HEF3) is set beforehand. To reset start condition flag 7 (SCF7), the PLC instruction must be used to reset the halt release request flag 3 (HRF3) or the SHE instruction must be used to reset the halt release enable flag 3 (HEF3).

4. The 15th stage's content of the pre-divider.

The MSC instruction is used to transfer the contents of status register 3 (STS3) to the accumulator (AC) and the data memory (RAM).

The following table shows the Bit Pattern of Status Register 3 (STS3)

Bit 3	Bit 2	Bit 1	Bit 0
Start condition flag 7 (SCF7)	15th stage of the pre-divider	Start condition flag 5 (SCF5)	Start condition flag 4 (SCF4)
Halt release caused by pre-divider overflow		Halt release caused by TMR1 underflow	Halt release caused by INT pin
Read only	Read only	Read only	Read only

2-14-4 STATUS REGISTER 3X (STS3X)

When the halt mode is released with start condition flag 2 (SCF2), status register 3X (STS3X) will store the status of the factor in the release of the halt mode.

Status register 3X (STS3X) consists of 3 flags:

1. Start condition flag 8 (SCF8)

   SCF8 is set to 1 when any one of KI1~4 =1/0 (KI1~4=1 in LED mode / KI1~4=0 in LCD mode) causes the halt release request flag 5 (HRF5) to be outputted and the halt release enable flag 5 (HEF5) is set beforehand. To reset the start condition flag 8 (SCF8), the PLC instruction must be used to reset the halt release request flag 5 (HRF5) or the SHE instruction must be used to reset the halt release enable flag 5 (HEF5).

2. Start condition flag 6 (SCF6)

SCF6 is set to 1 when an underflow signal from timer 2 (TMR2) causes the halt release request flag 4 (HRF4) to be outputted and the halt release enable flag 4 (HEF4) is set beforehand. To reset the start condition flag 6 (SCF6), the PLC instruction must be used to reset the halt release request flag 4 (HRF4) or the SHE instruction must be used to reset the halt release enable flag 4 (HEF4).

3. Start condition flag 9 (SCF9)

SCF9 is set when a finish signal from mode 3 of RFC function causes the halt release request flag 6 (HRF6) to be outputted and the halt release enable flag 9 (HEF9) is set beforehand. In this case, the 16-counter of RFC function must be controlled by CX pin; please refer to 2-16-9. To reset the start condition flag 9 (SCF9), the PLC instruction must be used to reset the halt release request flag 6 (HRF6) or the SHE instruction must be used to reset the halt release enable flag 6 (HEF6).

The MCX instruction can be used to transfer the contents of status register 3X (STS3X) to the accumulator (AC) and the data memory (RAM).

The following table shows the Bit Pattern of Status Register 3X (STS3X)

Bit 3	Bit 2	Bit 1	Bit 0
Start condition flag 9 (SCF9)	NA	Start condition flag 6 (SCF6)	Start condition flag 8 (SCF8)
Halt release caused by RFC counter finish	NA	Halt release caused by TMR2 underflow	Halt release caused by SKI underflow
Read only	Read only	Read only	Read only

2-14-5 STATUS REGISTER 4 (STS4)

Status register 4 (STS4) consists of 3 flags:

1. System clock selection flag (CSF)

The system clock selection flag (CSF) indicates which clock source of the system clock generator (SCG) is used. Executing SLOW instruction will change the clock source (BCLK) of the system clock generator (SCG) to the slow speed oscillator (XT clock), and the system clock selection flag (CSF) is reset to 0. Executing FAST instruction will change the clock source (BCLK) of the system clock generator (SCG) to the fast speed oscillator (CF clock), and the system clock selection flag (CSF) is set to 1. For the operation of the system clock generator, refer to 3-3.

2. Watchdog timer enable flag (WTEF)

The watchdog timer enable flag (WDF) indicates the operating status of the watchdog timer.

3. Overflow flag of 16-bit counter of RFC (RFOVF)

The overflow flag of 16-bit counter of RFC (RFOVF) is set to 1 when the overflow of the 16-bit counter of RFC occurs. The flag will reset to 0 when this counter is initiated by executing SRF instruction.

The MSD instruction can be used to transfer the contents of status register 4 (STS4) to the accumulator (AC) and the data memory (RAM).

The following table shows the Bit Pattern of Status Register 4 (STS4)

Bit 3	Bit 2	Bit 1	Bit 0
NA	The overflow flag of 16-bit counter of RFC (RFVOF)	Watchdog timer Enable flag (WDF)	System clock selection flag (CSF)
Read only	Read only	Read only	Read only

2-14-6 START CONDITION FLAG 11 (SCF11)

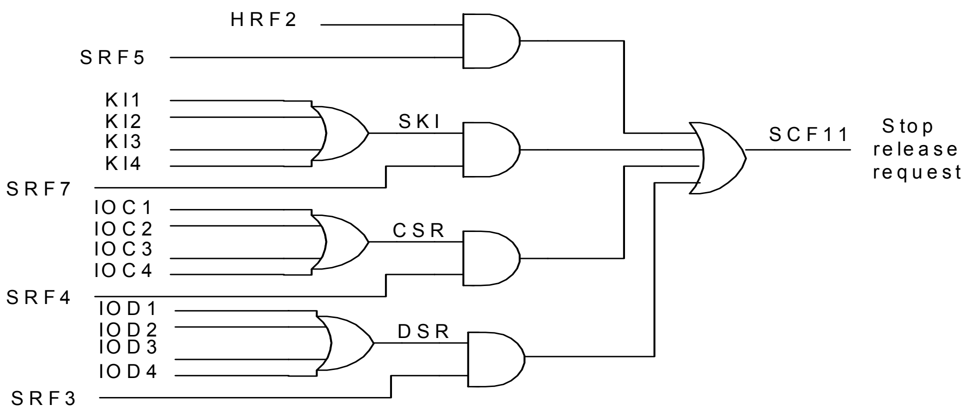
Start condition flag 11 (SCF11) will be set to 1 in STOP mode when the following conditions are met :

. A high level signal comes from the OR-ed output of the pins defined as input mode in IOC port, which causes the stop release flag of IOC port (CSR) to output, and stop release enable flag 4 (SRF4) is set beforehand.

. A high level signal comes from the OR-ed output of the pins defined as input mode in IOD port, which causes the stop release flag of IOD port (DSR) to output. The stop release enable flag3 (SRF 3) must be set beforehand.

. The signal change from the INT pin causes the halt release flag 2 (HRF2) to output and the stop release enable flag 5 (SRF5) is set beforehand.

The following figure shows the organization of start condition flag 11 (SCF 11).



The stop release flags (SKI, CSR, DSR, HRF2) were specified by the stop release enable flags (SRFx). These flags should be clear before the chip enters stop mode. All of the pins in the IOA and IOC ports have to be set in input mode and keep in 0 state before the chip enters the STOP mode, otherwise the program can not enter STOP mode.
Instruction SRE is used to set or reset the stop release enable flags (SRF4,5,7).

The following table shows the stop release request flags.

	The OR-ed latched signals for KI1~4	The OR-ed input mode pins of IOC(IOD) port	The rising or falling edge on INT pin
Stop release request flag	SKI	CSR(DSR)	HRF2
Stop release enable flag	SRF7	SRF4(SRF3)	SRF5

2-15 CONTROL REGISTER (CTL)

The control register (CTL) comes in 4 types: control register 1 (CTL1) to control register 4 (CTL4).

2-15-1 CONTROL REGISTER 1 (CTL1)

The control register 1 (CTL1), being a 1-bit register:

1. Switch enable flag 4 (SEF4)

   Stores the status of the input signal change at pins of IOC set in input mode that causes the halt mode or stop mode to be released.

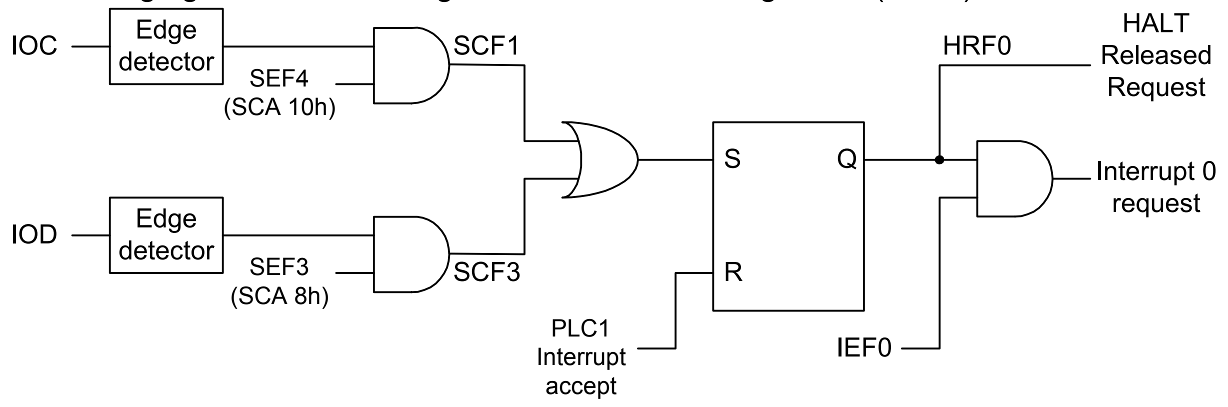
2. Switch enable flag 3 (SEF3)

   Stores the status of the input signal change at pins of IOD set in input mode that causes the halt mode or stop mode to be released.

Executed the SCA instruction may set or reset these flags.
The following table shows Bit Pattern of Control Register 1 (CTL1).

Bit 4	Bit 3
Switch enable flag 4 (SEF 4)	Switch enable flag 3 (SEF 3)
Enables the halt release caused by the signal change on IOC port	Enables the halt release caused by the signal change on IOD port
Write only	Write only

The following figure shows the organization of control register 1 (CTL1).



2-15-1-1 The Setting for Halt Mode

If the SEF4 is set to 1, the signal changed on IOC port will cause the halt mode to be released, and set SCF1 to 1. Because the input signal of IOC port were ORed, so it is necessary to keep the unchanged input signals at " 0 " state and only one of the input signal could change state.

2-15-1-2 The Setting for Stop Mode

If SRF4 and SEF4 are set, the stop mode will be released to set the SCF1 when a high level signal is applied to one of the input mode pins of IOC port and the other pins stay in "0" state.
After the stop mode is released, TM87P08 enters the halt condition.
The high level signal must hold for a while to cause the chattering prevention circuitry of IOC port to detect this signal and then set SCF1 to release the halt mode, or the chip will return to the stop mode again.

2-15-1-3 Interrupt for CTL1

The control register 1 (CTL1) performs the following function in the execution of the SIE instruction to enable the interrupt function.
The input signal changes at the input pins in IOC port will deliver the SCF1 when SEF4 has been set to 1 by executing SCA instruction. Once the SCF1 is delivered, the halt release request flag (HRF0) will be set to 1. In this case, if the interrupt enable flag 0 (IEF0) is set to 1 by executing SIE instruction, the interrupt request flag 0 (interrupt 0) will be delivered to interrupt the program.
If the interrupt 0 is accepted by SEF4 and IEF0, the interrupt 0 request to the next signal change at IOC will be inhibited. To release this mode, SCA instruction must be executed again. Refer to 2-16-1-1.

2-15-2 CONTROL REGISTER 2 (CTL2)

Control register 2 (CTL2) consists of halt release enable flags 1, 2, 3, 4, 5, 6 (HEF1, 2, 3, 4, 5, 6) and is set by SHE instruction. The bit pattern of the control register (CTL2) is shown below.

Halt release enable flag	HEF6	HEF5	HEF4
Halt release condition	Enable the halt release caused by RFC counter to be finished (HRF6)	Enable the halt release caused by Key Scanning(HRF5)	Enable the halt release caused by TMR2 underflow (HRF4)
Halt release enable flag	HEF3	HEF2	HEF1
Halt release condition	Enable the halt release caused by pre-divider overflow (HRF3)	Enable the halt release caused by INT pin (HRF2)	Enable the halt release caused by TM1 underflow (HRF1)

When the halt release enable flag 6 (HEF6) is set, a finish signal from the 16-bit counter of RFC causes the halt mode to be released. In the same manner, when HEF1 to HEF4 are set to 1, the following conditions will cause the halt mode to be released respectively: an underflow signal from TMR1, the signal change at the INT pin, an overflow signal from the pre-divider and an underflow signal from TMR2.
When the stop release enable flag 5 (SRF5) and the HEF2 are set, the signal change at the INT pin can cause the stop mode to be released.

2-15-3 CONTROL REGISTER 3 (CTL3)

Control register 3 (CTL3) is organized with 6 bits of interrupt enable flags (IEF) to enable / disable interrupts.
The interrupt enable flag (IEF) is set / reset by SIE* instruction. The bit pattern of control register 3 (CTL3) is shown below.

Enable flag	IEF6	IEF4
Request flag	Enable the interrupt request caused by RFC counter to be finished (HRF6)	Enable the interrupt request caused by TMR2 underflow (HRF4)
Interrupt No.	Interrupt 6	Interrupt 4
Enable flag	IEF3	IEF2
Request flag	Enable the interrupt request caused by predivider overflow (HRF3)	Enable the interrupt request caused by INT pin (HRF2)
Interrupt No.	Interrupt 3	Interrupt 2
Enable flag	IEF1	IEF0
Request flag	Enable the interrupt request caused by TM1 underflow (HRF1)	Enable the interrupt request caused by IOC or IOD port signal to be changed (HRF0)
Interrupt No.	Interrupt 1	Interrupt 0

When any of the interrupts are accepted, the corresponding HRFx and the interrupt enable flag (IEF) will be reset to 0 automatically. Therefore, the desirable interrupt enable flag (IEFx) must be set again before exiting from the interrupt routine.

2-15-4 CONTROL REGISTER 4 (CTL4)

Control register 4 (CTL4), being a 3-bit register, is set / reset by SRE instruction.
The following table shows the Bit Pattern of Control Register 4 (CTL4)

Stop release enable flag	SRF5	SRF4	SRF3
Stop release request flag	Enable the stop release request caused by signal change on INT pin (HRF2)	Enable the stop release request caused by signal change on IOC	Enable the stop release request caused by signal change on IOD

When the stop release enable flag 5 (SRF5) is set to 1, the input signal change at the INT pins causes the stop mode to be released. In the same manner, when SRF4 (SRF3) is set to 1, the input signal change at the input mode pins of IOC port and the signal changed on INT pin causes the stop mode to be released respectively.

Example:
This example illustrates the stop mode released by port IOC and INT pin. Assume all of the pins in IOD and IOC have been defined as input mode.

```
PLC         05h         ; Reset the HRF0 and HRF2.
SHE         04h         ; HEF2 and HEF5 is set so that the signal change at INT pin
                        ; causes start condition flag 4 or 8 to be set.
SCA         18h         ; SEF4 is set so that the signal changes at port IOC and IOD
                        ; cause the start conditions SCF1 to be set.
SRE         038h        ; SRF5,4 are set so that the signal changes at port
                        ; IOC, IOD and INT pin cause the stop mode to be released.
STOP                    ; Enter the stop mode.

      ……………            ;STOP release

MSC         10h         ; Check the signal change at INT pin that causes the stop mode
                        : to be
                        ; released.
MSB         11h         ; Check the signal change at port IOC, IOD that causes the
                        : stop mode to be
                        ; released.
```

2-16 HALT FUNCTION

The halt function is provided to minimize the current dissipation of the TM87P08 when LCD is operating. During the halt mode, the program memory (ROM) is not in operation and only the oscillator circuit, pre-divider circuit, sound circuit, I/O port chattering prevention circuit, and LCD driver output circuit are in operation. (If the timer has started operating, the timer counter still operates in the halt mode).

After the HALT instruction is executed and no halt release signal (SCF1, SCF3, HRF1 ~ 6) is delivered, the CPU enters the halt mode.

The following 3 conditions are available to release the halt mode.

(1) An interrupt is accepted.

When an interrupt is accepted, the halt mode is released automatically, and the program will enter halt mode again by executing the RTS instruction after completion of the interrupt service.

When the halt mode is released and an interrupt is accepted, the halt release signal is reset automatically.

(2) The signal change specified by the SCA instruction is applied to port IOC(SCF1).

(3) The halt release condition specified by the SHE instruction is met (HRF1 ~ HRF6).

When the halt mode is released in either (2) or (3), it is necessary that the MSB, MSC, or MCX instruction is executed in order to test the halt release signal and that the PLC instruction is then executed to reset the halt release signal (HRF).

Even when the halt instruction is executed in the state where the halt release signal is delivered, the CPU does not enter the halt mode.

2-17 BACK UP FUNCTION

TM87P08 provides a back up mode to avoid system malfunction when heavy loading occurs, such as buzzer activation, LED illumination... etc. Since heavy loading will cause a large voltage drop in the supply voltage, the system will malfunction in this condition. Once the program enters back up mode (BCF = 1), 32.768KHz Crystal oscillator will operate in a large driver condition and the internal logic function operates with a higher supply voltage. TM87P08 will get a higher power supply noise margin while back up mode is active, but it will also receive an increase in power consumption.
The back up flag (BCF) indicates the status of the back up function. BCF flag can be set or reset by executing the SF or RF instructions, respectively.
The back up function has different performance corresponding to different power mode options, shown in the following table.

TM87P08 status	3V battery or higher mode
	BCF flag status
Initial reset cycle	BCF = 1 (hardware controlled)
After initial reset cycle	BCF = 1 (hardware controlled)
Executing SF 2h instruction	BCF = 1
Executing RF 2h instruction	BCF = 0
HALT mode	Previous state
STOP mode	BCF = 1 (hardware controlled)

TM87P08 Oscillation	3V battery or higher mode	
	BCF = 0	BCF = 1
32.768KHz Crystal Oscillator	Small driver	Large driver

Note: For power saving reasons, it is recommended to reset BCF flag to 0 when back up mode is not used.

2-18 STOP FUNCTION (STOP)

The stop function is another solution to minimize the current dissipation for TM87P08. In stop mode, all of functions in TM87P08 are held including oscillators. All of the LCD corresponding signals (COM and Segment) will output "L" level. In this mode, TM87P08 does not dissipate any power in the stop mode. Because the stop mode will set the BCF flag to 1 automatically, it is recommended to reset the BCF flag after releasing the stop mode in order to reduce power consumption.
Before the stop instruction is executed, all of the signals on the pins defined as input mode of IOC port must be in the "L" state, and no stop release signal (SRFn) should be delivered. The CPU will then enter the stop mode.
The following conditions cause the stop mode to be released.

. One of the signals on the input mode pin of IOC port is in "H" state and holds long enough to cause the CPU to be released from halt mode.
. A signal change in the INT pin.
. The stop release condition specified by the SRE instruction is met.(INT pin is exclusive)

When the TM87P08 is released from the stop mode, the TM87P08 enters the halt mode immediately and will process the halt release procedure. If the "H" signal on the IOC port does not hold long enough to set the SCF1, once the signal on the IOC port returns to "L", the TM87P08 will enter the stop mode immediately. The backup flag (BCF) will be set to 1 automatically after the program enters the stop mode.

The following diagram shows the stop release procedure:

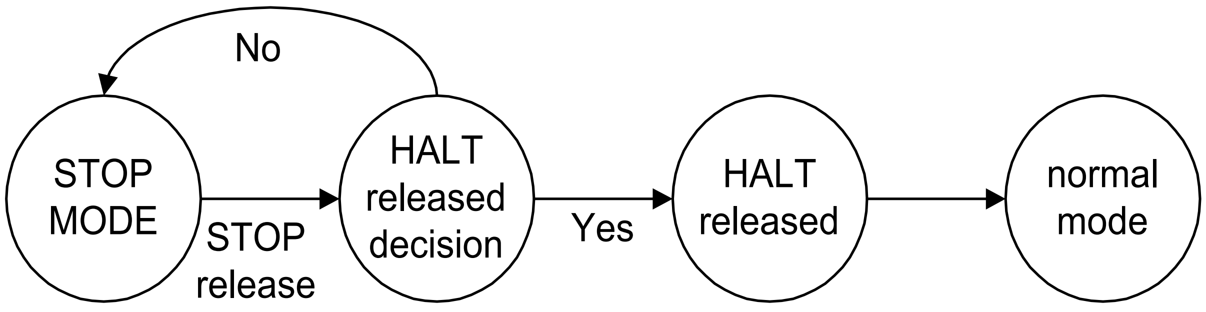


Figure 3- 16 The stop release state machine

Before the stop instruction is executed, the following operations must be completed:
. Specify the stop release conditions by the SRE instruction.
. Specify the halt release conditions corresponding to the stop release conditions if needed.
. Specify the interrupt conditions corresponding to the stop release conditions if needed.
When the stop mode is released by an interrupt request, the TM87P08 will enter the halt mode immediately. While the interrupt is accepted, the halt mode will be released by the interrupt request. The stop mode returns by executing the RTS instruction after completion of interrupt service.
After the stop release, it is necessary that the MSB, MSC or MCX instruction be executed to test the halt release signal and that the PLC instruction then be executed to reset the halt release signal. Even when the stop instruction is executed in the state where the stop release signal (SRF) is delivered, the CPU does not enter the stop mode but the halt mode. When the stop mode is released and an interrupt is accepted, the halt release signal (HRF) is reset automatically.

Chapter 3 Control Function

3-1 INTERRUPT FUNCTION

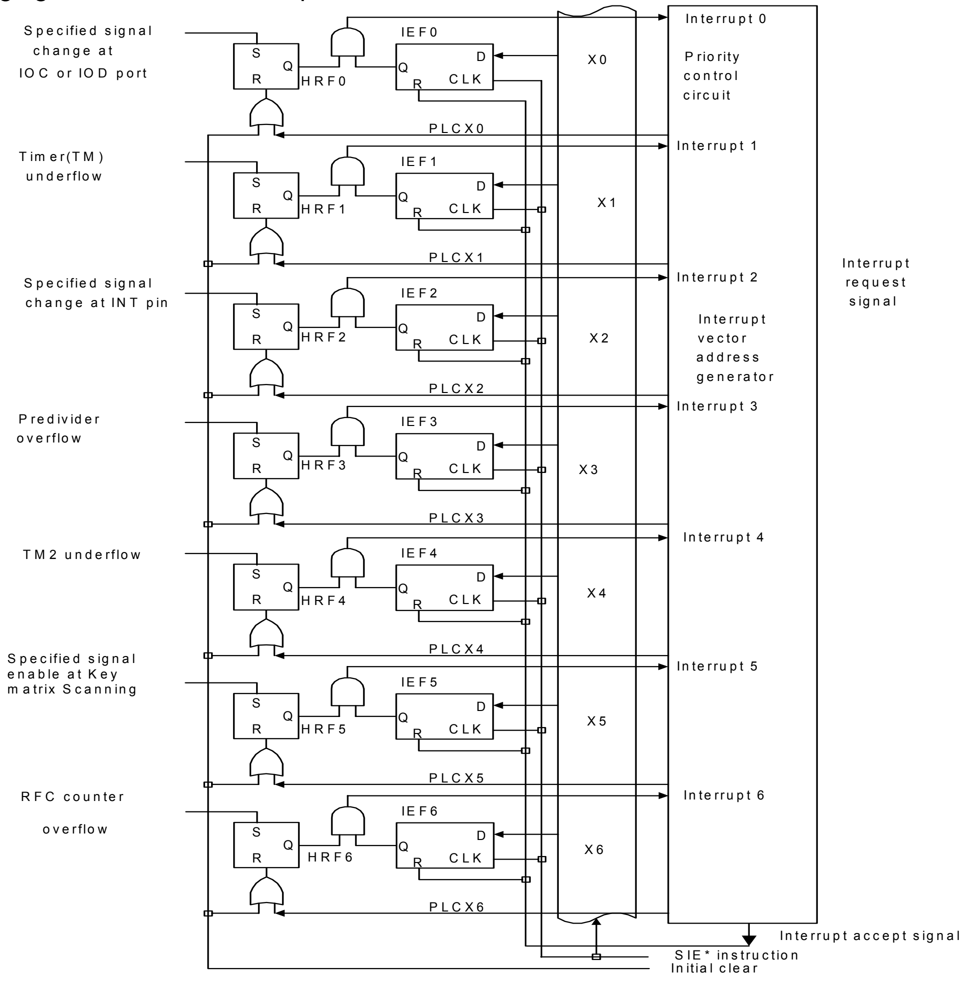
There are 7 interrupt resources: 3 external interrupt factors and 4 internal interrupt factors. When an interrupt is accepted, the program in execution is suspended temporarily and the corresponding interrupt service routine specified by a fix address in the program memory (ROM) is called.

The following table shows the flag and service of each interrupt:

Table 3-1 Interrupt information

Interrupt source	INT pin	IOC, IOD port	TMR1 underflow	Pre-divider overflow	TMR2 underflow	RFC counter overflow
Interrupt vector	010H	014H	018H	01CH	020H	028H
Interrupt enable flag	IEF2	IEF0	IEF1	IEF3	IEF4	IEF6
Interrupt priority	6th	5th	2nd	1st	3rd	4th
Interrupt request flag	Interrupt 2	Interrupt 0	Interrupt 1	Interrupt 3	Interrupt 4	Interrupt 6

The following figure shows the Interrupt Control Circuit



3-1-1 INTERRUPT REQUEST AND SERVICE ADDRESS

3-1-1-1 External interrupt factor

The external interrupt factor involves the use of the INT pin, IOC ports, or IOD ports.

1. External INT pin interrupt request
By using mask option, either a rise or fall of the signal at the INT pin can be selected for applying an interrupt. If the interrupt enable flag 2 (IEF2) is set and the signal on the INT pin change that matches the mask option will issue the HRF2, interrupt 2 is accepted and the instruction at address10H is executed automatically. It is necessary to apply level "L" before the signal rises and level "H" after the signal rises to the INT pin for at least 1 machine cycle.

2. I/O port IOC, IOD interrupt request.
An interrupt request signal (HRF0) is delivered when the input signal changes at I/O port IOC, IOD specified by the SCA instruction. In this case, if the interrupt enabled by flag 0 (IEF0) is set to 1, interrupt 0 is accepted and the instruction at address 14H is executed automatically.

3. Key matrix Scanning interrupt request.
An interrupt request signal (HRF5) is delivered when the input signal is generated in the scanning interval. If the interrupt enable flag 5 (IEF5) is set to 1 and interrupt 5 is accepted, the instruction at address 24H will be executed automatically.

3-1-1-2 Internal interrupt factor

The internal interrupt factor involves the use of timer 1 (TMR1), timer 2 (TMR2), RFC counter and the pre-divider.

1. Timer1 / 2 (TMR1 / 2) interrupt request
   An interrupt request signal (HRF1 / 4) is delivered when timer1 / 2 (TMR1/ 2) underflows. In this case, if the interrupt enable flag 1 / 4 (IEF1 / 4) is set, interrupt 1 / 4 is accepted and the instruction at address 18H / 20H is executed automatically.

2. Pre-divider interrupt request
   An interrupt request signal (HRF3) is delivered when the pre-divider overflows. In this case, if the interrupt enable flag3 (IEF3) is set, interrupt 3 is accepted and the instruction at address 1CH is executed automatically.

3. 16-bit counter of RFC (CX pin control mode) interrupt request
   An interrupt request signal (HRF6) is delivered when the 2nd falling edge applied on CX pin and 16-bit counter stops to operate. In this case, if the interrupt enable flag6 (IEF6) is set, interrupt 6 is accepted and the instruction at address 28H is executed automatically.

3-1-2 INTERRUPT PRIORITY

If all interrupts are requested simultaneously during a state when all interrupts are enabled, the pre-divider interrupt is given the first priority and other interrupts are held. When the interrupt service routine is initiated, all of the interrupt enable flags (IEF0 ~ IEF6) are cleared and should be set with the next execution of the SIE instruction. Refer to Table 3-1.

Example:

```
; Assume all interrupts are requested simultaneously when all interrupts are enabled, and
all of the
; the pins of IOC have been defined as input mode.

        PLC    7Fh             ;Clear all of the HRF flags
        SCA    18h             ;enable the interrupt request of IOC, IOD
        SIE*   5Fh             ;enable all interrupt requests

        ;……………………………;all interrupts are requested simultaneously.

        ;Interrupt caused by the predivider overflow occurs, and interrupt  service is concluded.

        SIE*   57h             ;Enable the interrupt request (except  the predivider).

        ;Interrupt caused by the TM1 underflow occurs, and interrupt service is concluded.

        SIE*   55h             ;Enable the interrupt request (except  the predivider and TMR1).

        ;Interrupt caused by the TM2 underflow occurs, and interrupt service is concluded.

        SIE*   45h             ;Enable the interrupt request(except the predivider, TMR1
        and ;TMR2).

        ;Interrupt caused by the RFC counter overflow occurs, and interrupt service is concluded.

        SIE*   05h             ;Enable the interrupt request (except the predivider, TMR1,
                               ;TMR2, and the RFC counter).

        ;Interrupt caused by the IOC port, and interrupt service is concluded.

        SIE*   04h             ;Enable the interrupt request (except  the predivider, TMR1,
                               ;TMR2, RFC counter, and IOC, IOD port)

        ;Interrupt caused by the INT pin, and interrupt service is concluded.
        ;All interrupt requests have been processed.
```

3-1-3 INTERRUPT SERVICING

When an interrupt is enabled, the program in execution is suspended and the instruction at the interrupt service address is executed automatically(Refer to Table 3-1). In this case, the CPU performs the following services automatically.

(1) As for the return address of the interrupt service routine, the addresses of the program counter (PC) installed before interrupt servicing began are saved in the stack register (STACK).

(2) The corresponding interrupt service routine address is loaded in the program counter (PC).
The interrupt request flag corresponding to the interrupt accepted is reset and the interrupt enable flags are all reset.

When the interrupt occurs, the TM87P08 will follow the procedure below:

```
Instruction 1       ;In this instruction, interrupt is accepted.
NOP                 ;TM87P08 stores the program counter data into the STACK. At this
time,
                    ;no instruction will be executed, as with NOP instruction.
Instruction A       ;The program jumps to the interrupt service routine.
Instruction B
Instruction C
 .............
RTS                 ;Finishes the interrupt service routine
Instruction 1*      ;re-executes the instruction which was interrupted.
Instruction 2
```

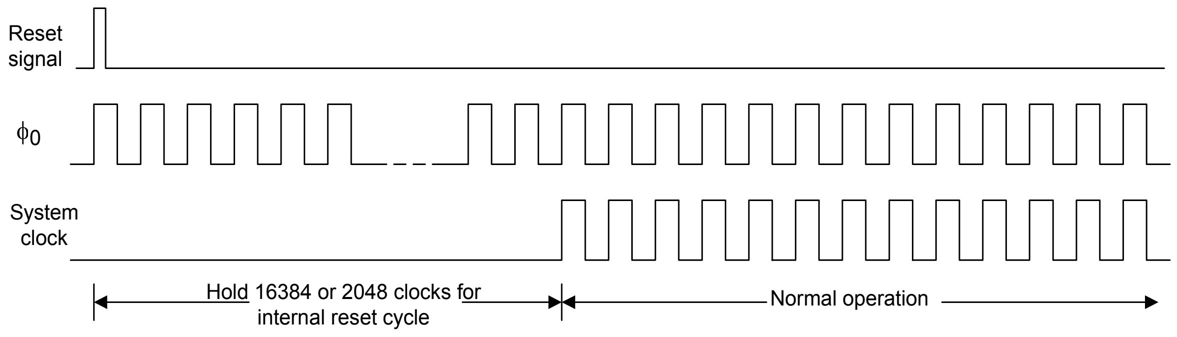
Note: If instruction 1 is "halt" instruction, the CPU will return to "halt" after interrupt.

When an interrupt is accepted, all interrupt enable flags are reset to 0 and the corresponding HRF flag will be cleared; the interrupt enable flags(IEF) must be set again in the interrupt service routine as required.

3-2 RESET FUNCTION

TM87P08 contains four reset sources: power-on reset, RESET pin reset, IOC port reset and watchdog timer reset.

When reset signal is accepted, TM87P08 will generate a time period for internal reset cycle and there are two types of internal reset cycle time could be selected by mask option, the one is PH15/2 and the other is PH12/2.



.

Internal reset cycle time is PH15/2

MASK OPTION table :

Mask Option name	Selected item
RESET TIME	(1) PH15/2

In this option, the reset cycle time will be extended 16384 clocks (clock source comes form pre-divider) long at least.

. Internal reset cycle time is PH12/2

MASK OPTION table :

Mask Option name	Selected item
RESET TIME	(2) PH12/2

In this option, the reset cycle time will be extended 2048 clocks (clock source comes form pre-divider) long at least.

3-2-1 RESET PIN RESET

When "H" level is applied to the reset pin, the reset signal will issue. Built in a pull down resistor on this pin.
Two types of reset method for RESET pin and the type could be mask option, the one is level reset and other is pulse reset.
It is recommended to connect a capacitor (0.1uf) between RESET pin and VBAT. This connection will prevent the bounce signal on RESET pin.

3-2-1-1 Level Reset

Once a “1” signal applied on the RESET pin, TM87P08 will not release the reset cycle until the signal on RESET pin returned to “0”. After the signal on reset pin is cleared to 0, TM87P08 begins the internal reset cycle and then release the reset status automatically.

MASK OPTION table :

Mask Option name	Selected item
RESET PIN TYPE	(1) LEVEL

3-2-1-2 Pulse Reset

Once a “1” signal applied on the RESET pin, TM87P08 will escape from reset state and begin the normal operation after internal reset cycle automatically no matter what the signal on RESET pin returned to “0” or not.

MASK OPTION table :

Mask Option name	Selected item
RESET PIN TYPE	(2) PULSE

The following table shows the initial condition of TM87P08 in reset cycle.

Program counter	(PC)	Address 000H
Start condition flags 1 to 7	(SCF1-7)	0
Backup flag	(BCF)	1 (Li-B option)
Stop release enable flags 4,5,7	(SRF3,4,5,7)	0
Switch enable flags 4	(SEF3,4)	0
Halt release request flag	(HRF 0~6)	0
Halt release enable flags 1 to 3	(HEF1-6)	0
Interrupt enable flags 0 to 3	(IEF0-6)	0
Alarm output	(ALARM)	DC 0
Pull-down flags in I/OC, I/OD port		1(with pull-down resistor)
Input/output ports I/OA, I/OB, I/OC, I/OD	(PORT I/OA, I/OB, I/OC, I/OD)	Input mode
I/OC, I/OD port chattering clock	Cch	PH10*
Frequency generator clock source and duty cycle	Cfq	PH0, duty cycle is 1/4, output is inactive
Resistor frequency converter	(RFC)	Inactive, RR/RT/RH output 0
LCD driver output		All lighted (mask option)*
Timer 1/2		Inactive
Watchdog timer	(WDT)	Reset mode, WDF = 0
Clock source	(BCLK)	XT clock (slow speed clock in dual clock option)

Notes: PH3: the 3rd output of predivider.   PH10: the 10th output of predivider

Mask option can unlighted all of the LCD output

3-2-3 IOC Port RESET

Key reset function is selected by mask option. When IOC port is in used, the '0' signal applied to all these pins that had be set as input mode in the same time, reset signal is delivered.

MASK OPTION table :

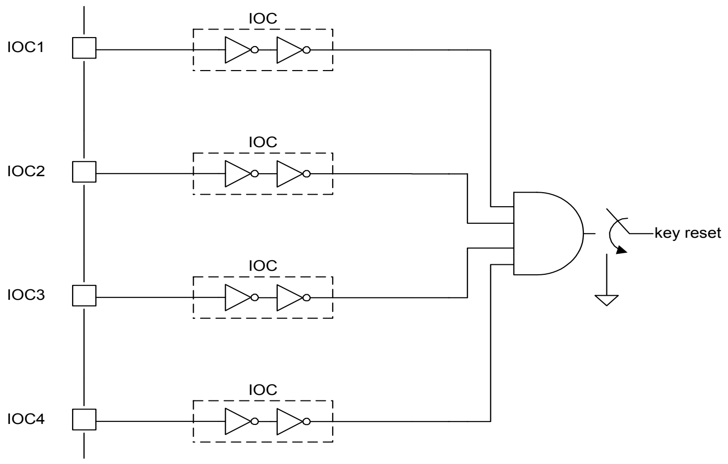
IOC or KI pins are used as key reset :

Mask Option name	Selected item
IOC1 FOR KEY RESET	(1) USE
IOC2 FOR KEY RESET	(1) USE
IOC3 FOR KEY RESET	(1) USE
IOC4 FOR KEY RESET	(1) USE

IOC pins aren't used as key reset :

Mask Option name	Selected item
IOC1 FOR KEY RESET	(2) NO USE
IOC2 FOR KEY RESET	(2) NO USE
IOC3 FOR KEY RESET	(2) NO USE
IOC4 FOR KEY RESET	(2) NO USE

The following figure shows the key reset organization.

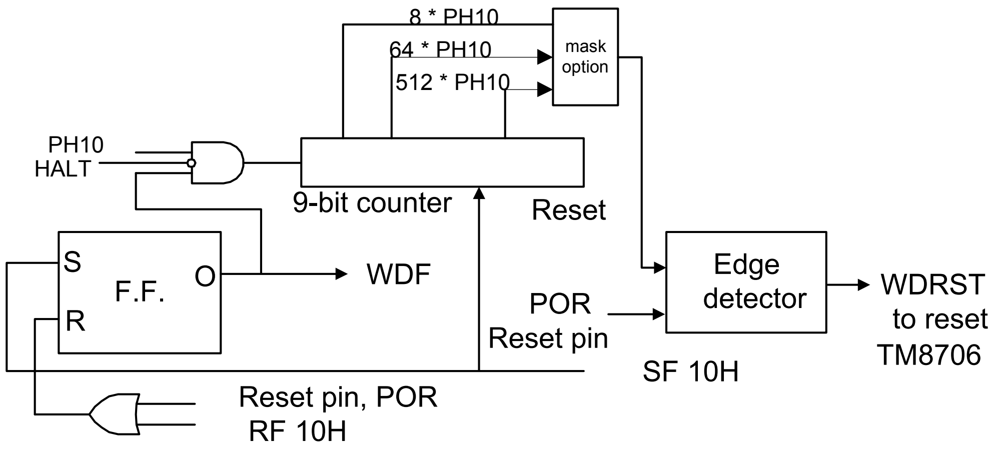


3-2-4 WATCHDOG RESET

The timer is used to detect unexpected execution sequence caused by software run-away. The watchdog timer consists of a 9-bit binary counter. The timer input (PH10) is the 10th stage output of the pre-divider.

When the watchdog timer overflows, it generates a reset signal to reset TM87P08 and most of the functions in TM87P08 will be initiated except for the watchdog timer (which is still active), WDF flag will not be affected and PH0 ~ PH10 of the pre-divider will not be reset.

The following figure shows the watchdog timer organization.



During initial reset (power on reset [POR] or reset pin), the timer is inactive and the watchdog flag (WDF) is reset. Instruction SF 10h will enable the watchdog timer and set the watchdog flag (WDF) to 1. At the same time, the content of the timer will be cleared. Once the watchdog timer is enabled, the timer will be paused when the program enters the halt mode or stop mode. When the TM87P08 wakes up from the halt or stop mode, the timer operates continuously. It is recommended to execute SF 10h instruction before the program enters the halt or stop mode in order to initialize the watchdog timer.
Once the watchdog timer is enabled, the program must execute SF 10h instruction periodically to prevent the timer overflowed.
The overflow time interval of watchdog timer is selected by mask option :
MASK OPTION table :

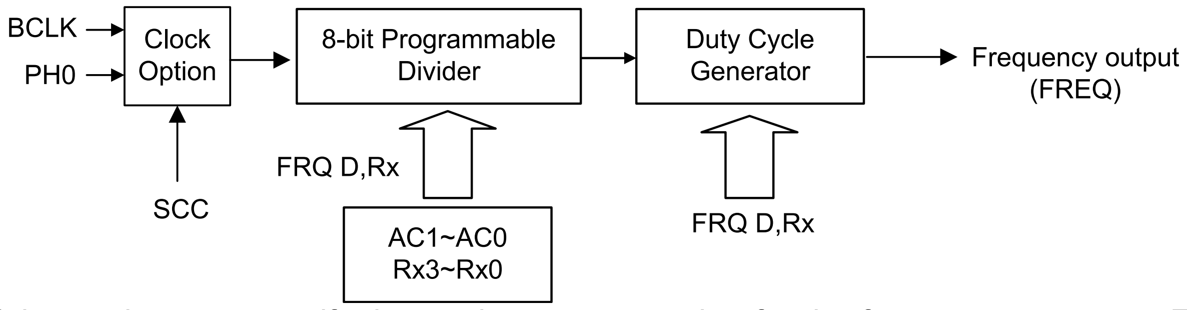
Mask Option name	Selected item
WATCHDOG TIMER OVERFLOW TIME INTERVAL	(1) 8 x PH10
WATCHDOG TIMER OVERFLOW TIME INTERVAL	(2) 64 x PH10
WATCHDOG TIMER OVERFLOW TIME INTERVAL	(3) 512 x PH10

Note : timer overflow time interval is about 16 seconds when PH0 = 32.768KHz

3-3 CLOCK GENERATOR

3-3-1 FREQUENCY GENERATOR

The Frequency Generator is a versatile programmable divider that is capable of delivering a clock with wide frequency range and different duty cycles. The output of the frequency generator may be the clock source for the alarm function, timer1, timer2 and RFC counter. The following shows the organization of the frequency generator.



SCC instruction may specify the clock source selection for the frequency generator. The frequency generator outputs the clock with different frequencies and duty cycles corresponding to the presetting data of FRQ related instructions. The FRQ related instructions preset a letter N into the programming divider and letter D into the duty cycle generator. The frequency generator will then output the clock using the following formula:
FREQ=(clock source) / ((N+1) * X) Hz. (X=1,2,3,4 for 1/1,1/2,1/3,1/4 duty)

This letter N is a combination of data memory and accumulator (AC), or the table ROM data or operand data specified in the FRQX instruction. The following table shows the bit pattern of the combination.

The following table shows the bit pattern of the preset letter N

	The bit pattern of preset letter N							
Programming divider	bit7	Bit6	bit 5	bit 4	bit 3	Bit 2	bit 1	bit 0
FRQ D,Rx	AC3	C2	AC1	AC0	Rx3	Rx2	Rx1	Rx0
FRQ D,@HL	T7	T6	T5	T4	T3	T2	T1	T0
FRQX D,X	X7	X6	X5	X4	X3	X2	X1	X0

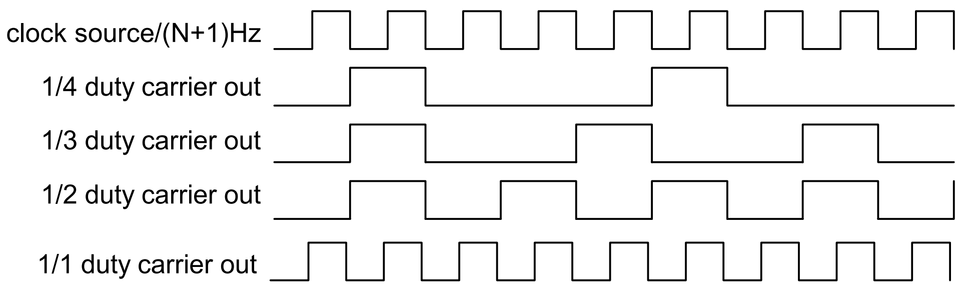
Notes: 1. T0 ~ T7 represents the data of table ROM.

2. X0 ~ X7 represents the data specified in operand X.

The following table shows the bit pattern of the preset letter D

Preset Letter D		Duty Cycle
D1	D0	
0	0	1/4 duty
0	1	1/3 duty
1	0	1/2 duty
1	1	1/1 duty

The following diagram shows the output waveform for different duty cycles.



3-3-2 Melody Output

The frequency generator may generate the frequency for melody usage. When the frequency generator is used to generate the melody output, the tone table is shown below:

1. The clock source is PH0, i.e. 32.768KHz
2. The duty cycle is 1/2 Duty (D=2)
3. “FREQ” is the output frequency
4. “ideal” is the ideal tone frequency
5. “%” is the frequency deviation

The following table shows the note table for melody application

Tone	N	FREQ	Ideal	%	Tone	N	FREQ	Ideal	%
C2	249	65.5360	65.4064	0.19	C4	62	260.063	261.626	-0.60
#C2	235	69.4237	69.2957	0.18	#C4	58	277.695	277.183	0.18
D2	222	73.4709	73.4162	0.07	D4	55	292.571	293.665	-0.37
#D2	210	77.6493	77.7817	-0.17	#D4	52	309.132	311.127	-0.64
E2	198	82.3317	82.4069	-0.09	E4	49	327.680	329.628	-0.59
F2	187	87.1489	87.3071	-0.18	F4	46	348.596	349.228	-0.18
#F2	176	92.5650	92.4986	0.07	#F4	43	372.364	369.994	0.64
G2	166	98.1078	97.9989	0.11	G4	41	390.095	391.995	-0.48
#G2	157	103.696	103.826	-0.13	#G4	38	420.103	415.305	1.16
A2	148	109.960	110.000	-0.04	A4	36	442.811	440.000	0.64
#A2	140	116.199	116.541	-0.29	#A4	34	468.114	466.164	0.42
B2	132	123.188	123.471	-0.23	B4	32	496.485	493.883	0.53
C3	124	131.072	130.813	0.20	C5	30	528.516	523.251	1.01
#C3	117	138.847	138.591	0.19	#C5	29	546.133	554.365	-1.48
D3	111	146.286	146.832	-0.37	D5	27	585.143	587.330	-0.37
#D3	104	156.038	155.563	0.31	#D5	25	630.154	622.254	1.27
E3	98	165.495	164.814	0.41	E5	24	655.360	659.255	-0.59
F3	93	174.298	174.614	-0.18	F5	22	712.348	698.456	1.99
#F3	88	184.090	184.997	-0.49	#F5	21	744.727	739.989	0.64
G3	83	195.048	195.998	-0.48	G5	20	780.190	783.991	-0.48
#G3	78	207.392	207.652	-0.13	#G5	19	819.200	830.609	-1.37
A3	73	221.405	220.000	0.64	A5	18	862.316	880.000	-2.01
#A3	69	234.057	233.082	0.42	#A5	17	910.222	932.328	-2.37
B3	65	248.242	246.942	0.53	B5	16	963.765	987.767	-2.43

Note:

1. Above variation does not include X'tal variation.
2. If PH0 = 65536Hz, C3 - B5 may have more accurate frequency.

During the application of melody output, sound effect output or carrier output of remote control, the frequency generator needs to combine with the alarm function (BZB, BZ). For detailed information about this application, refer to section 3-4.

3-3-3 Halver / Doubler / Tripler

The halver / doubler / tripler circuits are used to generate the bias voltage for LCD and are composed of a combination of PH2, PH3, PH4, PH5.

3-3-4 Alternating Frequency for LCD

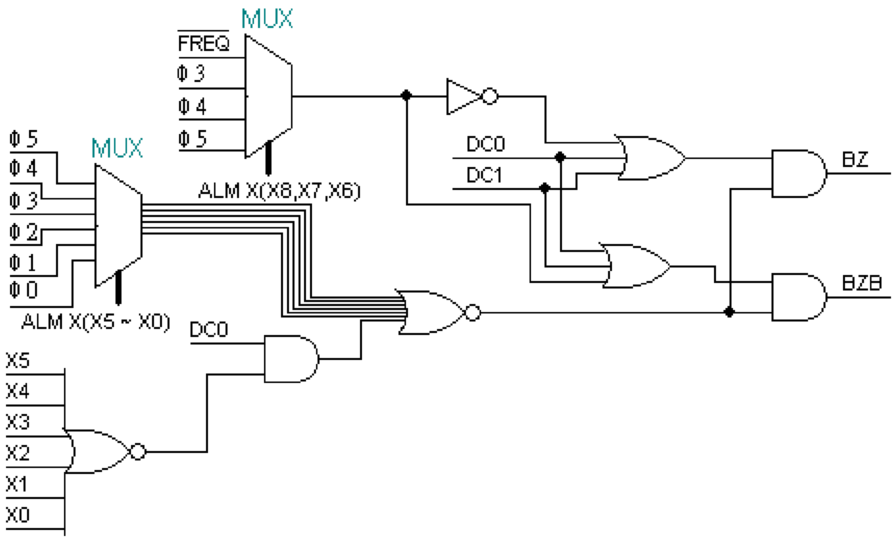
The alternating frequency for LCD is a frequency used to make the LCD waveform.

3-4 BUZZER OUTPUT PINS

There are two output pins, BZB and BZ. Each are MUXed with IOB3 and IOB4 by mask option, respectively. BZB and BZ pins are versatile output pins with complementary output polarity. When buzzer output function combined with the clock source comes from the frequency generator, this output function may generate melody, sound effect or carrier output of remote control.

MASK OPTION table :

Mask Option name	Selected item
SEG30/IOB3/BZB	(3) BZB
SEG31/IOB4/BZ	(3) BZ



This figure shows the organization of the buzzer output.

3-4-1 BASIC BUZZER OUTPUT

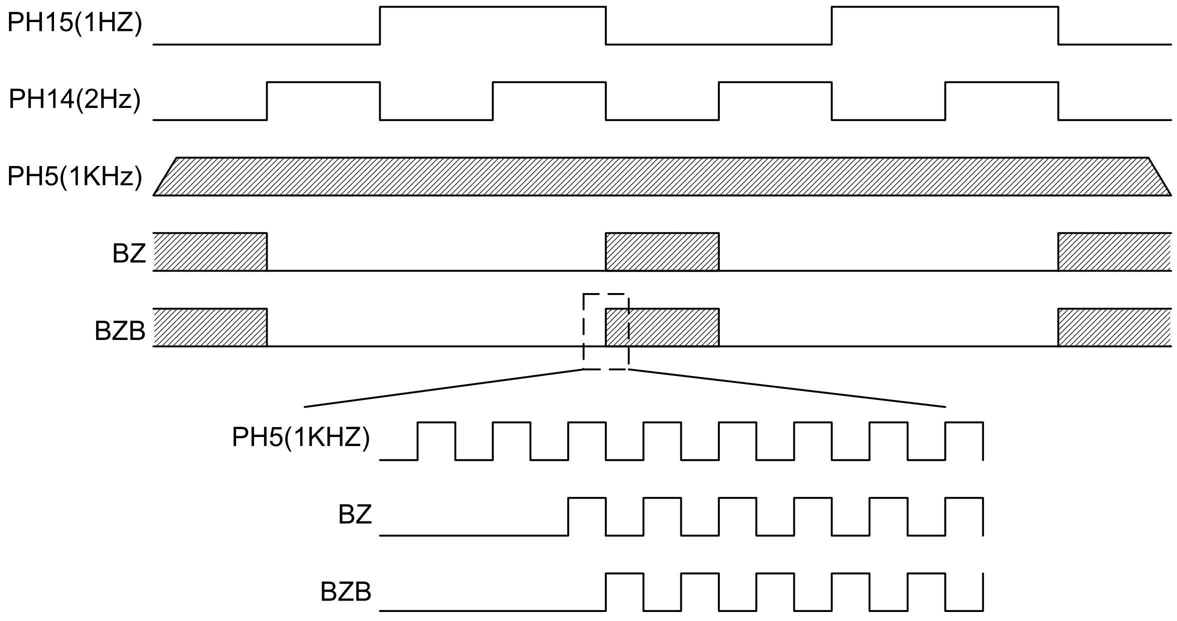
The buzzer output (BZ, BZB) is suitable for driving a transistor for the buzzer with one output pin or driving a buzzer with BZ and BZB pins directly. It is capable of delivering a modulation output in any combination of one signal of FREQ, PH3(4096Hz), PH4(2048Hz), PH5(1024Hz) and multiple signals of PH10(32Hz), PH11 (16Hz), PH12(8Hz), PH13(4Hz), PH14(2Hz), PH15(1Hz). The ALM instruction is used to specify the combination. The higher frequency clock is the carrier of modulation output and the lower frequency clock is the envelope of the modulation output.

Note:
1. The high frequency clock source should only be one of PH3, PH4, PH5 or FREQ, and the lower frequency may be any/all of the combinations from PH10 ~ PH15.
2. The frequencies in () corresponding to the input clock of the pre-divider (PH0) is 32768Hz.
3. The BZ and BZB pins will output DC0 after the initial reset.

Example:
Buzzer output generates a waveform with 1KHz carrier and (PH15 + PH14) envelope.

```
        LDS    20h, 0Ah
        ……….
        ALM    70h            ; Output the waveform.
        ………
```

In this example, the BZ and BZB pins will generate the waveform as shown in the following figure :



3-4-2 THE CARRIER FOR REMOTE CONTROL

If buzzer output combines with the timer and frequency generator, the output of the BZ pin may deliver the waveform for the IR remote controller. For remote control usage, the setting value of the frequency generator must be greater than or equal to 3, and the ALM instruction must be executed immediately after the FRQ related instructions in order to deliver the FREQ signal to the BZ pin as the carrier for IR remote controller.

Example:

```
        SHE         2       ;Enable timer 1 halt release enable flag.
        TMSX        3Fh     ;Set value for timer 1 is 3Fh and the clock source is PH9.
        SCC         40h     ;Set the clock source of  the frequency generator as BCLK.
        FRQX        2, 3    ;FREQ = BCLK / (4*2), setting value for the frequency
```

```
                               generator
                              ;is 3 and duty cycle is 1/2.
        ALM          1C0h     ;FREQ signal is outputted. This instruction must be executed
                              ;after the FRQ related instructions.
        HALT                  ;Wait for the halt release caused by timer 1.
        ……………………              ;Halt released.
        ALM          0        ;Stop the buzzer output.
```

3-5 INPUT / OUTPUT PORTS

Three I/O ports are available in TM87P08 : IOA, IOB and IOC. Each I/O port is composed of 4 bits and has the same basic function.
When the I/O pins are defined as non-IO function by mask option, the input / output function of the pins will be disabled.

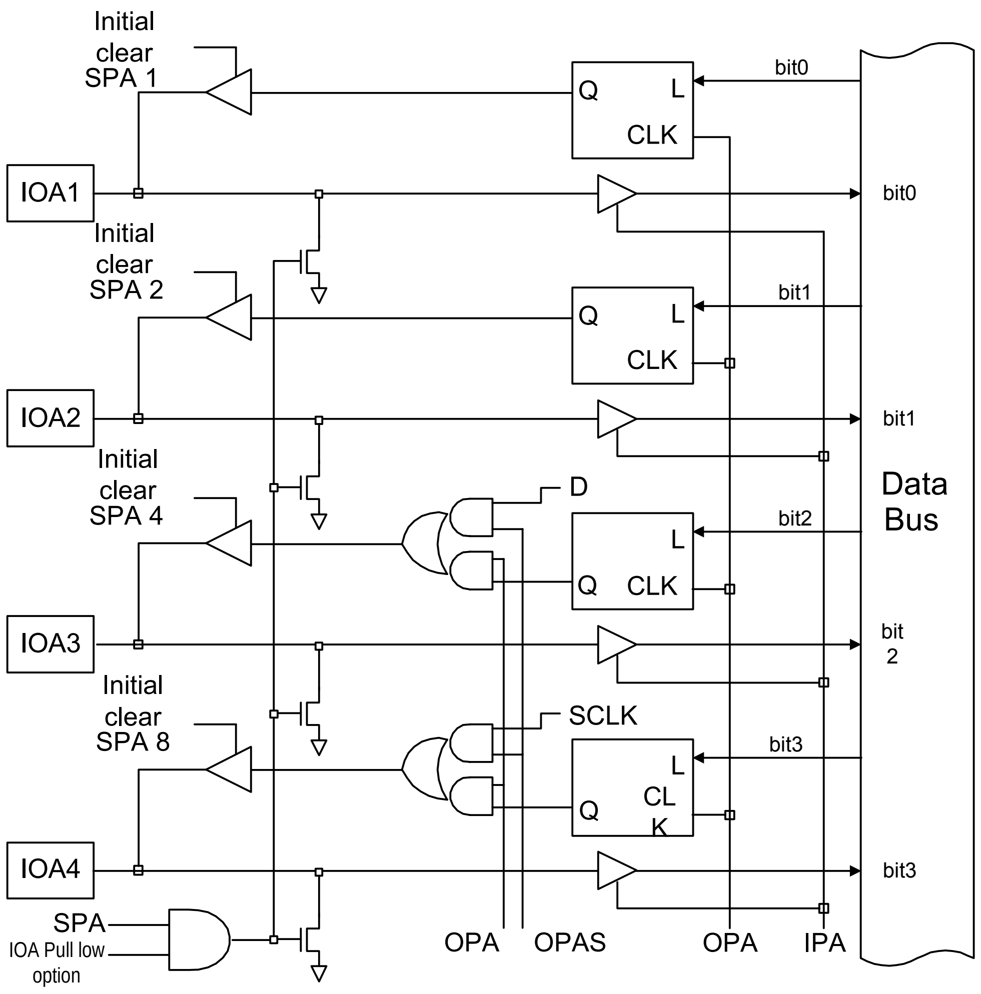
3-5-1 IOA PORT

IOA1 ~ IOA4 pins are MUX with CX / SEG24, RR / SEG25, RT / SEG26 and RH / SEG27 pins respectively by mask option.

MASK OPTION table :

Mask Option name	Selected item
SEG24/IOA1/CX	(2) IOA1
SEG25/IOA2/RR	(2) IOA2
SEG26/IOA3/RT	(2) IOA3
SEG27/IOA4/RH	(2) IOA4

In initial reset cycle, the IOA port is set as input mode and each bit of port can be defined as input mode or output mode individually by executing SPA instructions. Executing OPA instructions may output the content of specified data memory to the pins defined as output mode; the pins defined as the input mode will still remain the input mode.
Executing IPA instructions may store the signals applied to the IO pins into the specified data memory. When the IO pins are defined as the output mode, executing IPA instruction will store the content that stored in the latch of the output pin into the specified data memory.
Before executing SPA instruction to define the I/O pins as the output mode, the OPA instruction must be executed to output the data to those output latches beforehand. This will prevent the chattering signal on the I/O pin when the I/O mode changed.
IOA port had built-in pull-down resistor and executing SPA instruction to enable / disable this device.



This figure shows the organization of IOA port.

Note: If the input level is in the floating state, a large current (straight-through current) flows to the input buffer. The input level must not be in the floating state.

3-5-1-1 Pseudo Serial Output

IOA port may operate as a pseudo serial output port by executing OPAS instruction. IOA port must be defined as the output mode before executing OPAS instruction.

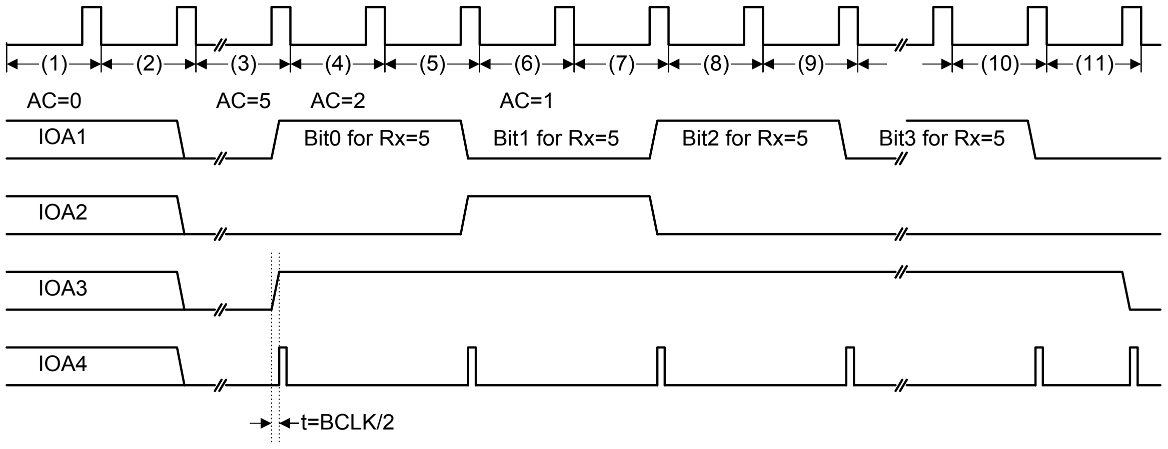
1. BIT0 and BIT1 of the port deliver RAM data.
2. BIT2 of the port delivers the constant value of the OPAS.
3. BIT3 of the port delivers pulses.

Shown below is a sample program using the OPAS instruction.

```
(1)     LDS   0AH, 0
(2)     OPA   0AH
        SPA   0FH
         :
         :
        LDS   1,5
(3)     OPAS 1,1      ;Bit 0 output, shift gate open
```

```
(4)     SR0   1       ;Shifts bit 1 to bit 0
(5)     OPAS 1,1      ;Bit 1 output
(6)     SR0   1       ;Shifts bit2 to bit 0
(7)     OPAS 1,1      ;Bit 2 output
(8)     SR0   1       ;Shifts bit 3 to bit 0
(9)     OPAS 1,1      ;Bit 3 output
          :
          :
(10)    OPAS 1,1      ;Last data
(11)    OPAS 1,0      ;Shift gate closes
```

The timing chart below illustrates the above program.



If IOA1 pin is used as the CX pin for RFC function and the other pins (IOA2 ~ IOA3) are used for normal IO pins, IOA1 pin must always be defined as the output mode to avoid the influence from the CX when the input chattering prevention function is active. On the other hand, the RFC counter can receive the signal changes on IOA1 when the RFC counter is enabled.

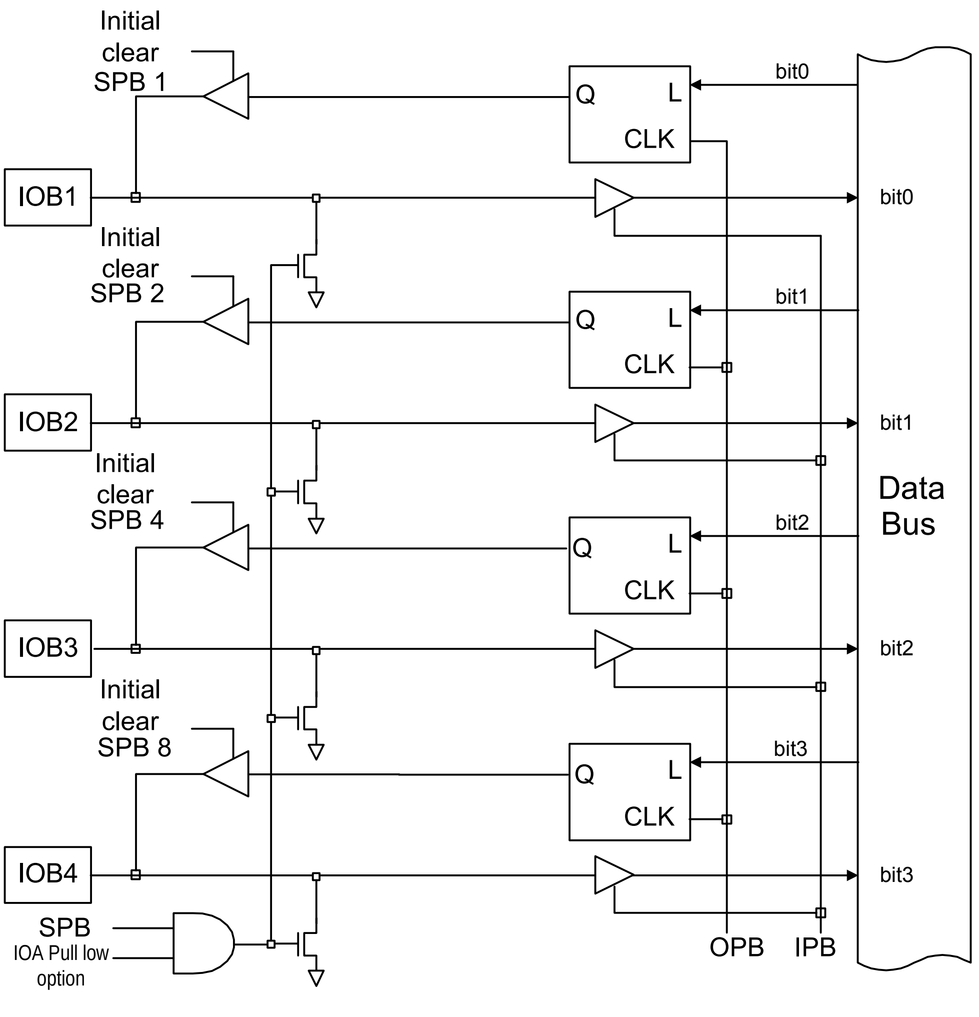
3-5-2 IOB PORT

IOB1 ~ IOB4 pins are MUXed with SEG28, SEG29, BZB / SEG30 and BZ / SEG31 pins respectively by mask option.

MASK OPTION table :

Mask Option name	Selected item
SEG28/IOB1	(2) IOB1
SEG29/IOB2	(2) IOB2
SEG30/IOB3/BZB	(2) IOB3
SEG31/IOB4/BZ	(2) IOB4

The following figure shows the organization of IOB port.



Note: If the input level is in the floating state, a large current (straight-through current) flows to the input buffer. The input level must not be in the floating state.

After the reset cycle, the IOB port is set as input and each bit of port can be defined as input or output individually by executing SPB instructions. Executing OPB instructions may

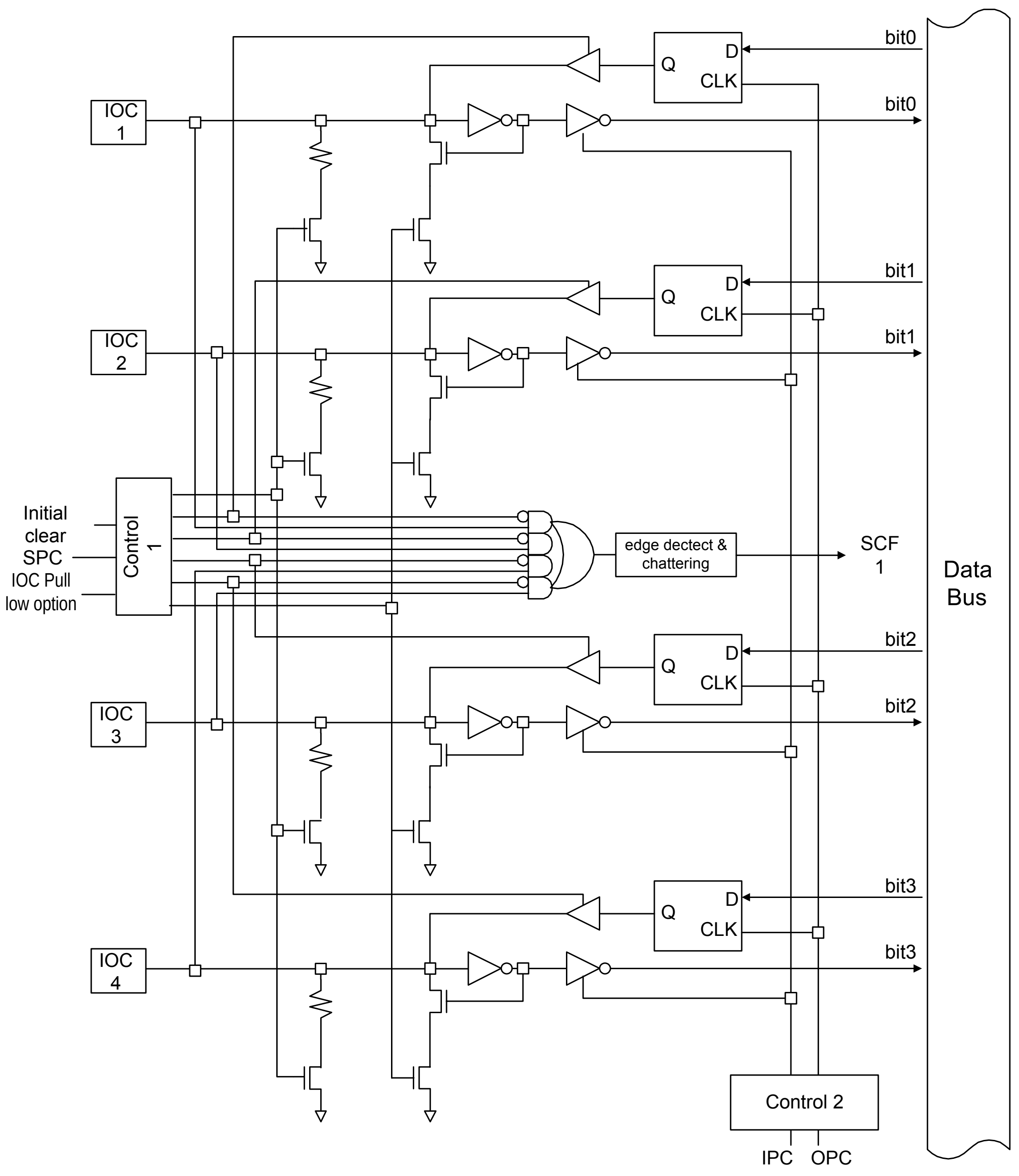
output the content of specified data memory to the pins defined as output mode; the other pins which are defined as the input will still be input.
Executed IPB instructions may store the signals applied on the IOB pins into the specified data memory. When the IOB pins are defined as the output, executing IPB instruction will save the data stored in the output latch into the specified data memory.
Before executing SPB instruction to define the I/O pins as output, the OPB instruction must be executed to output the data to the output latches. This will prevent the chattering signal on the I/O pin when the I/O mode changed.
IOB port had built-in pull-down resistor and executing SPB instruction to enable / disable this device.

3-5-3 IOC PORT

IOC1 ~ IOC4 pins are MUXed with SEG32/KI1, SEG33/KI2, SEG34/KI3 and SEG35/KI4 pins respectively by mask option.
MASK OPTION table :

Mask Option name	Selected item
SEG32/IOC1/KI1	(2) IOC1
SEG33/IOC2/KI2	(2) IOC2
SEG34/IOC3/KI3	(2) IOC3
SEG35/IOC4/KI4	(2) IOC4

After the reset cycle, the IOC port is set as input mode and each bit of port can be defined as input mode or output mode individually by executing SPC instruction. Executed OPC instruction may output the content of specified data memory to the pins defined as output; the other pins which are defined as the input will still remain the input mode.
Executed IPC instructions may store the signals applied to the IOC pins in the specified data memory. When the IOC pins are defined as the output, executing IPC instruction will save the data stored in the output latches in the specified data memory.
Before executing SPC instruction to define the IOC pins as output, the OPC instruction must be executed to output the data to those output latches. This will prevent the chattering signal when the IOC pins change to output mode.



This figure shows the organization of IOC port.

Note: If the input level is in the floating state, a large current (straight-through current) flows to the input buffer when both the pull low and L-level hold devices are disabled. The input level must not be in the floating state

IOC port had built-in pull-down resistor and executing SPC instruction to enable / disable this device.

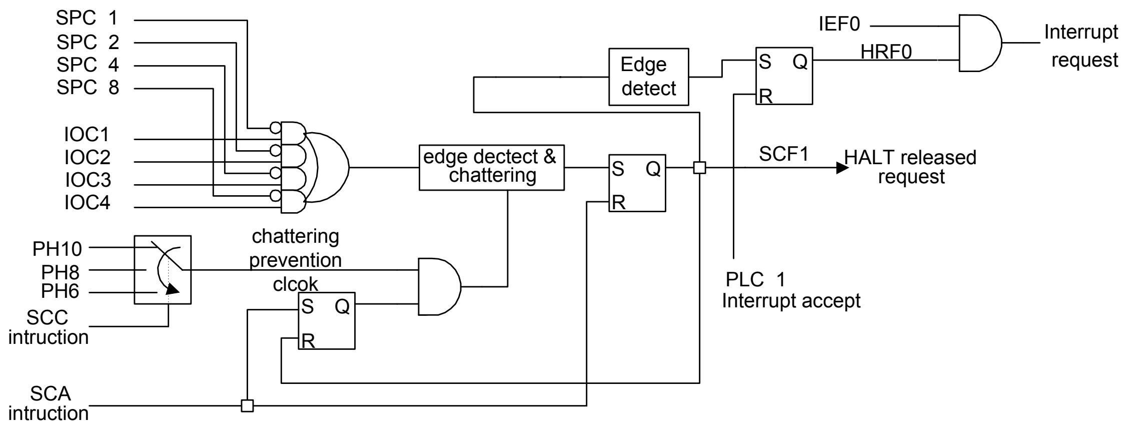
IOC port also built in the pull-low device for each pin, but these devices are enable by mask option. The pull-down resistor and low-level-hold device in each IOC pin can't exist in the same time. When the pull-down resistor is enabled, the low-level-hold device will be disable, vise versa. Executing SPC 10h instruction to enable the pull-low device and

disable the low-level hold device, executing SPC 0h to disable the pull-low device and enable the low-level hold device.
When the low-level hold device is enabled by mask option, the initial reset will enable the pull-low device and disable the low-level hold device.
When the IOC pin has been defined as the output mode, both the pull-low and low-level hold devices will be disabled.
Low-level-hold function option

Mask Option name	Selected item
C PORT LOW LEVEL HOLD	(1) USE
C PORT LOW LEVEL HOLD	(2) NO USE

3-5-3-1 Chattering Prevention Function and Halt Release

The port IOC is capable of preventing high / low chattering of the switch signal applied on IOC1 to IOC4 pins. The chattering prevention time can be selected as PH10 (32ms), PH8 (8ms) or PH6 (2ms) by executing SCC instruction, and the default selection is PH10 after the reset cycle. When the pins of the IOC port are defined as output, the signals applied to the output pins will be inhibited for the chattering prevention function. The following figure shows the organization of chattering prevention circuitry.



Note: The default prevention clock is PH10

This chattering prevention function works when the signal at the applicable pin (ex. IOC1) is changed from "L" level to "H" level or from "H" level to "L" level, and the remaining pins (ex, IOC2 to IOC4) are held at "L" level.
When the signal changes at the input pins of IOC port specified by the SCA instruction occur and keep the state for at least two chattering clock (PH6, PH8, PH10) cycles, the control circuit at the input pins will deliver the halt release request signal (SCF1). At that time, the chattering prevention clock will stop due to the delivery of SCF1. The SCF1 will be reset to 0 by executing SCA instruction and the chattering prevention clock will be enabled at the same time. If the SCF1 has been set to 1, the halt release request flag 0 (HRF0) will be delivered. In this case, if the port IOC interrupt enable mode (IEF0) is provided, the interrupt is accepted.

Since no flip-flop is available to hold the information of the signal at the input pins IOC1 to IOC4, the input data at the port IOC must be read into the RAM immediately after the halt mode is released.

3-5-4 IOD PORT

IOD1 ~ IOD4 pins are MUXed with SEG36, SEG37, SEG38 and SEG39 pins respectively by mask option.

MASK OPTION table :

Mask Option name	Selected item
SEG36/IOD1	(2) IOD1
SEG37/IOD2	(2) IOD2
SEG38/IOD3	(2) IOD3
SEG39/IOD4	(2) IOD4

After the reset cycle, the IOD port is set to input mode; each bit of port can be set to input or output mode individually by executing SPD instructions. Executing the OPD instruction outputs the contents of specified data memory locations to the pins set as output; the other pins which are set as input will still remain the in the input mode.

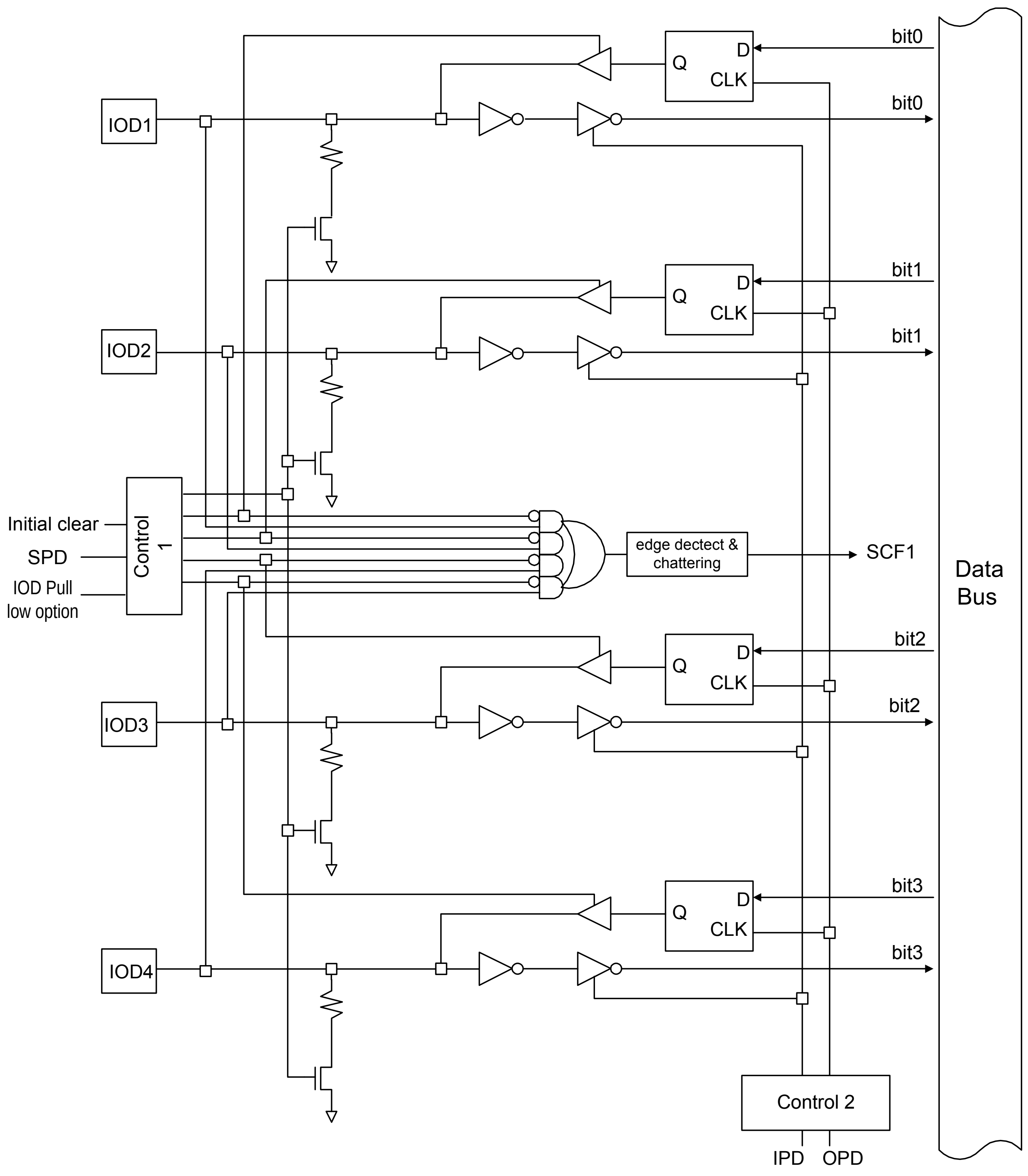
Executing IPD instructions will store the signals applied to the IOD pins in the specified data memory locations. When the IOD pins are set as output, executing IPD instructions will save the data stored in the output latches in the specified data memory locations. Before executing SPD instructions to define the IOD pins as output, the OPD instructions must be executed to output the data to those output latches. This will prevent the chattering signal when the IOD pins change to output mode.

IOD port has a built in pull-low device for each pin that is selected by mask option. To enable or disable this device, execute the SPD instruction.
When the IOD pin has been set to the output mode, the pull-low device will be disabled.

MASK OPTION table :
Pull-low function option

Mask Option name	Selected item
IOC PULL LOW RESISTOR	(1) USE
IOC PULL LOW RESISTOR	(2) NO USE

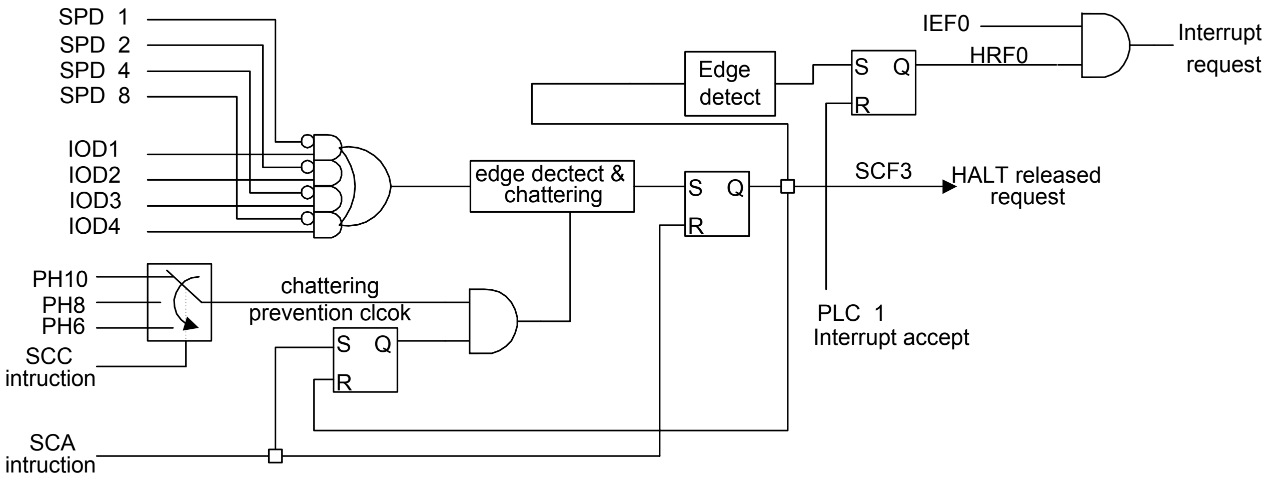
Note: If the input level is in the floating state, a large current (straight-through current) flows to the input buffer when both the pull low and L-level hold devices are disabled. The input level must not be in the floating state



This figure shows the organization of IOD port.

3-5-4-1 Chattering Prevention Function and Halt Release

The port IOD is capable of preventing high / low chattering of the switch signal applied on the IOD1 to IOD4 pins. Chattering prevention time can be selected as PH10 (32ms), PH8 (8ms) or PH6 (2ms) by executing the SCC instruction; the default selection is PH10 after the reset cycle. When the pins of the IOD port are set as output, the signals applied to the output pins will be inhibited for the chattering prevention function. The following figure shows the organization of chattering prevention circuitry.



This figure shows the organization of chattering prevention circuitry.

Note: The default prevention clock is PH10

This chattering prevention function works when the signal at the applicable pin (ex. IOD1) is changed from ”L” level to ”H” level or from ”H” level to ”L” level, and the remaining pins (ex, IOD2 to IOD4) are held at ”L” level.
When the signal changes at the input pins of IOD port specified by the SCA instruction occur and keep the state for at least two chattering clock (PH6, PH8, PH10) cycles, the control circuit at the input pins will deliver the halt release request signal (SCF3). At that time, the chattering prevention clock will stop due to the delivery of SCF3. The SCF3 will be reset to 0 by executing SCA instruction and the chattering prevention clock will be enabled at the same time. If the SCF3 has been set to 1, the halt release request flag 0 (HRF0) will be delivered. In this case, if the port IOD interrupt enable mode (IEF0) is provided, the interrupt is accepted.
Since no flip-flop is available to hold the information of the signal at the input pins IOD1 to IOD4, the input data at the port IOD must be read into the RAM immediately after the halt mode is released.

3-6 EXTERNAL INT PIN

The INT pin can be selected as pull-up or pull-down or open type by mask option. The signal change (either rising edge or falling edge by mask option) sets the interrupt flag, delivering the halt release request flag 2 (HRF2). In this case, if the halt release enable flag (HEF2) is provided, the start condition flag 2 is delivered. If the INT pin interrupt enable mode (IEF2) is provided, the interrupt is accepted.

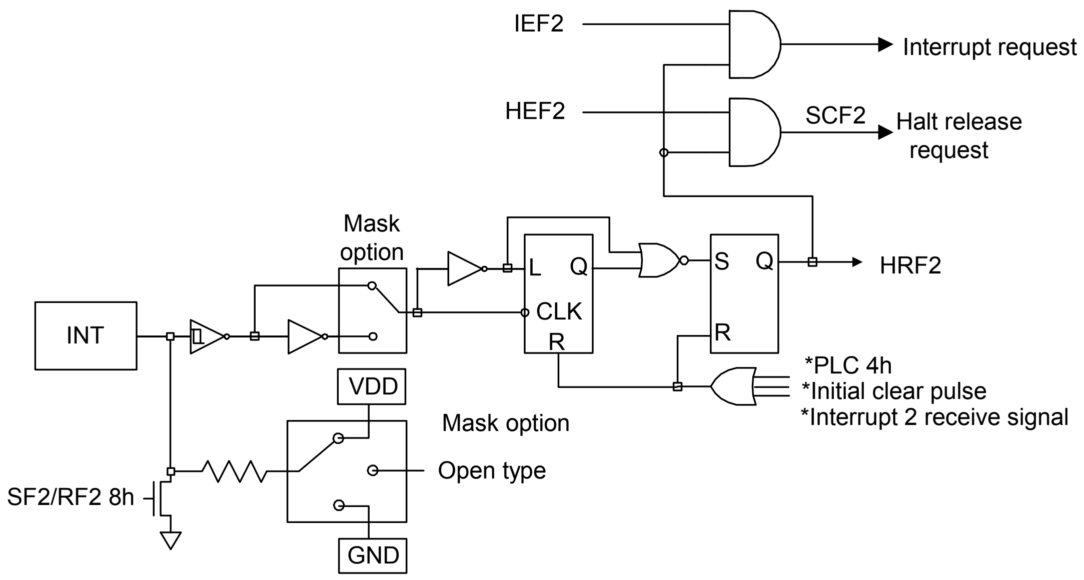
MASK OPTION table :

For internal resistor type :

Mask Option name	Selected item
INT PIN INTERNAL RESISTOR	(1) PULL HIGH
INT PIN INTERNAL RESISTOR	(2) PULL LOW
INT PIN INTERNAL RESISTOR	(3) OPEN TYPE

For input triggered type :

Mask Option name	Selected item
INT PIN TRIGGER MODE	(1) RISING EDGE
INT PIN TRIGGER MODE	(2) FALLING EDGE

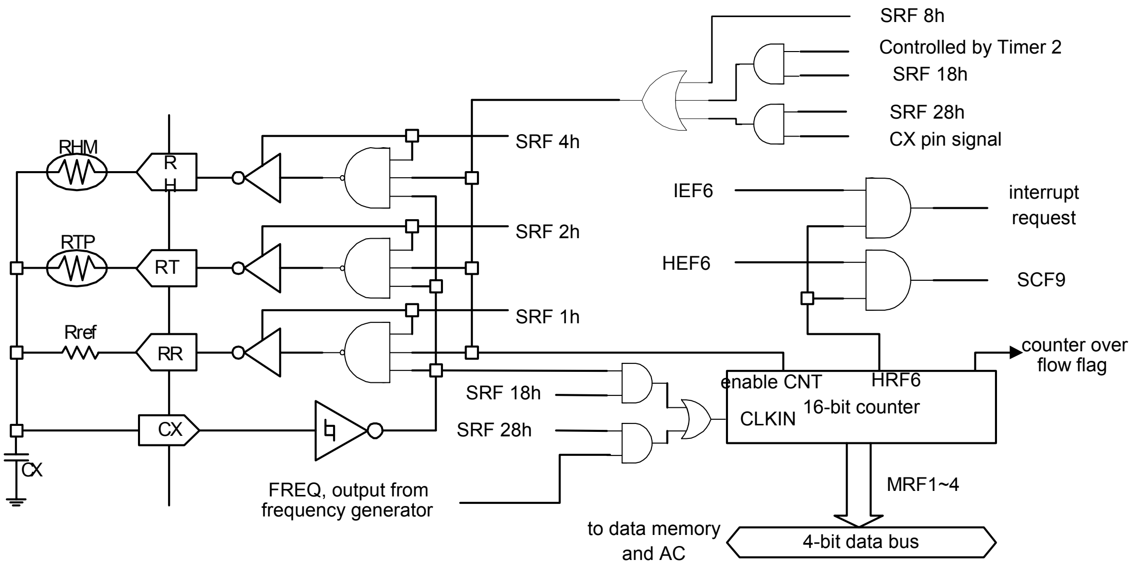


.

This figure shows the INT Pin Configuration

3-7 Resister to Frequency Converter (RFC)

The resistor to frequency converter (RFC) can compare two different sensors with the reference resister separately. This figure shows the block diagram of RFC.



This RFC contains four external pins:
CX: the oscillation Schemmit trigger input
RR: the reference resister output pin
RT: the temperature sensor output pin
RH: the humidity sensor output pin (this can also be used as another temperature sensor or can even be left floating)
These CX, RR, RT and RH pins are MUXed with IOA1 / SEG37 to IOA4 / SEG40 respectively and selected by mask option.
MASK OPTION table :

Mask Option name	Selected item
SEG24/IOA1/CX	(3) CX
SEG25/IOA2/RR	(3) RR
SEG26/IOA3/RT	(3) RT
SEG27/IOA4/RH	(3) RH

3-7-1 RC Oscillation Network

The RFC circuitry may build up 3 RC oscillation networks through RR, RT or RH and CX pins with external resistors. Only one RC oscillation network may be active at a time. When the oscillation network is built up (executing SRF 1h, SRF 2h, SRF 4h instructions to enable RR, RT, RH networks respectively), the clock will be generated by the oscillation network and transferred to the 16-bit counter through the CX pin. It will then enable or disable the 16-bit counter in order to count the oscillation clock.
Build up the RC oscillation network:

1. Connect the resistor and capacitor on the RR, RT, RH and CX pins. Fig. 2-24 illustrates the connection of these networks.
2. Execute SRF 1h, SRF 2h, or SRF 4h instructions to activate the output pins for RC networks respectively. The RR, RT, RH pins will become of a tri-state type when these networks are disabled.
3. Execute SRF 8, SRF 18h or SRF 28h instructions to enable the RC oscillation network and 16-bit counter. The RC oscillation network will not operate if these instructions have not been executed, and the RR, RT, RH pins output 0 state at this time.

To get a better oscillation clock from the CX pin, activate the output pin for each RC network before the counter is enabled.
The RFC function provides 3 modes for the operation of the 16-bit counter. Each mode will be described in the following sections:

3-7-2 Enable/Disable the Counter by Software

The clock input of the 16-bit counter comes from the CX pin and is enabled / disabled by the S/W. When SRF 8h instruction is executed, the counter will be enabled and will start to count the signals on the CX pin. The counter will be disabled when SRF 0 instruction is executed. Executing MRF1 ~ 4 instructions may load the result of the counter into the specified data memory and AC.
Each time the 16-bit counter is enabled, the content of the counter will be cleared automatically.

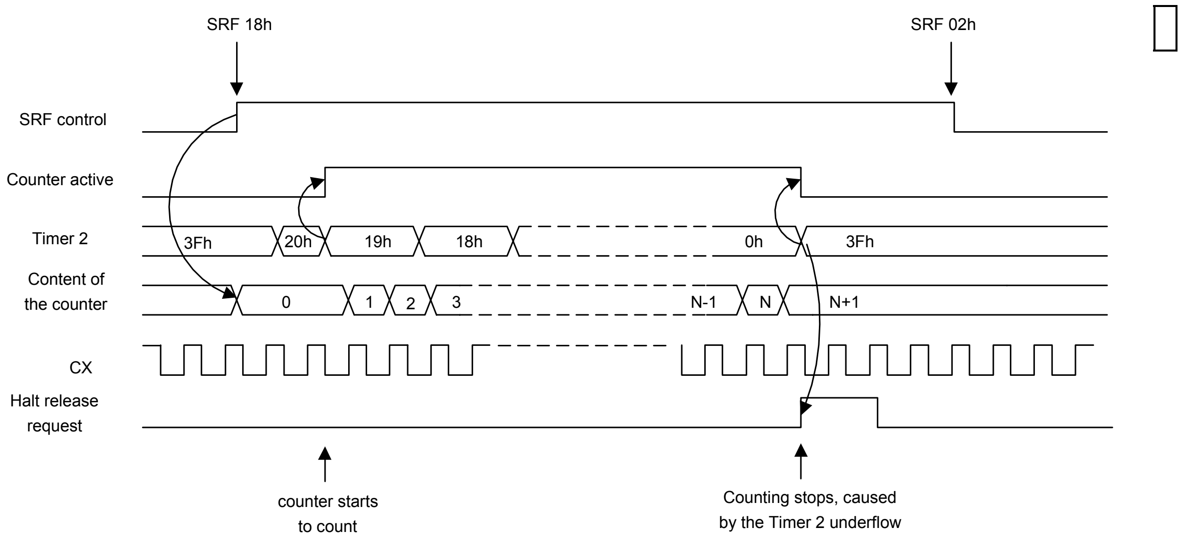
Example:

If you intend to count the clock input from the CX pin for a specified time period, you can enable the counter by executing SRF 8 instruction and setting timer1 to control the time period. Check the overflow flag (RFOVF) of this counter when the time period elapses. If the overflow flag is not set to 1, read the content of the counter; if the overflow flag has been set to 1, you must reduce the time period and repeat the previous procedure again.
In this example, use the RR network to generate the clock source.

```
;Timer 1 is used to enable/disable the counter
   LDS       0, 0            ;Set the TMR1 clock source (PH9)
   LDS       1, 3            ;initiate TMR1 setting value to 3F
   LDS       2, 0Fh
   SHE       2               ;enable halt release by TMR1
RE_CNT:
   LDA       0
   OR*       1               ;combine the TMR1 setting value
   TMS       2               ;enable the TMR1
   SRF       9               ;build up the RR network and enable the counter
   HALT
   SRF       1               ;stop the counter when TMR1 underflows
   MRF1      10h             ;read the content of the counter
   MRF2      11h
   MRF3      12h
   MRF4       13h
   MSD        20h
   JB2        CNT1_OF          ;check the overflow flag of counter
   JMP        DATA_ACCEPT
CNT1_OF:
   DEC*       2                ;decrease the TM1 value
   LDS        20h, 0
   SBC*       1
   JZ         CHG_CLK_RANGE  ;change the clock source of TMR1
   PLC        1                   ;clear the halt release request flag of TMR1
   JMP        RE_CNT
```

3-7-3 Enable / Disable the Counter by Timer 2

TMR2 will control the operation of the counter in this mode. When the counter is controlled by SRF 18 instruction, the counter will start to operate until TMR2 is enabled and the first falling edge of the clock source gets into TMR2. When the TMR2 underflow occurs, the counter will be disabled and will stop counting the CX clock at the same time. This mode can set an accurate time period with which to count the clock numbers on the CX pin. For a detailed description of the operation of TMR2, please refer to 2-12.
Each time the 16-bit counter is enabled, the content of the counter will be cleared automatically.



This figure shows the timing of the RFC counter controlled by timer 2

Example:

```
; In this example, use the RT network to generate the clock source.
SRF          1Ah             ;Build up the RT network and enable the counter
                             ;controlled by TM2
SHE          10h             ;enable the halt release caused by TM2
TM2X         20h             ;set the PH9 as the clock source of TM2 and the down
```

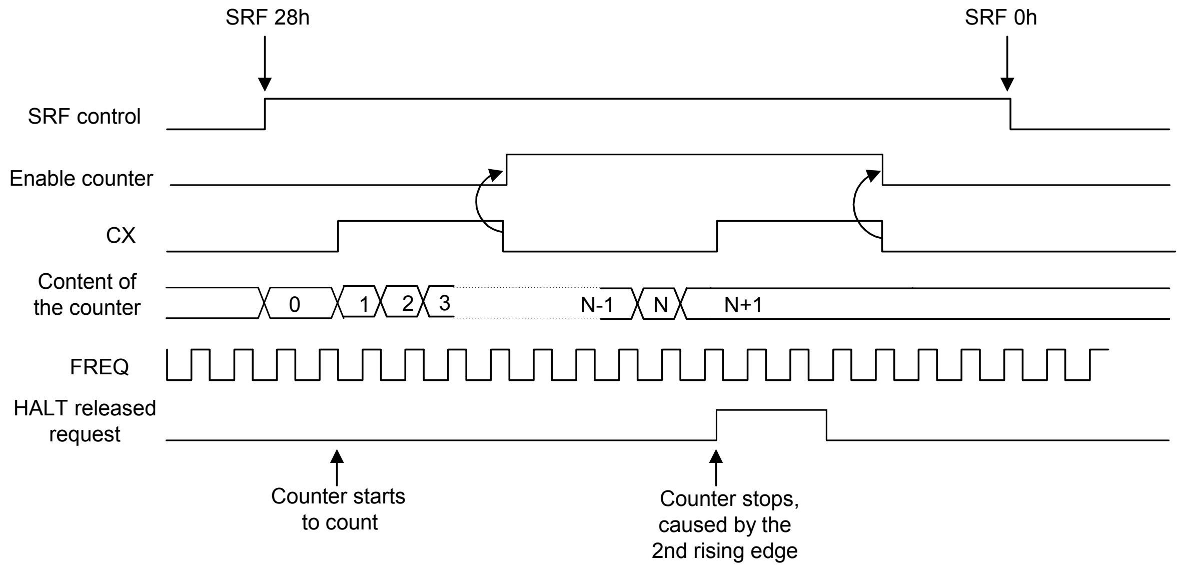
```
                        ;count value is 20h.
HALT
PLC         10h         ;Clear the halt release request flag of TM2
MRF1        10h         ;read the content of the counter.
MRF2        11h
MRF3        12h
MRF4        13h
```

3-7-4 Enable / Disable the Counter by CX Signal

This is another use for the 16-bit counter. In previous modes, CX is the clock source of the counter and the program must specify a time period by timer or subroutine to control the counter. In this mode, however, the counter has a different operation method. CX pin becomes the controlled signal to enable / disable the counter and the clock source of the counter comes from the output of the frequency generator (FREQ).

The counter will start to count the clock (FREQ) after the first rising edge signal applied on the CX pin when the counter is enabled. Once the second rising edge is applied to the CX pin after the counter is enabled, the halt release request (HRF6) will be delivered and the counter will stop counting. In this case, if the interrupt enable mode (IEF6) is provided, the interrupt is accepted; and if the halt release enable mode (HEF6) is provided, the halt release request signal is delivered, setting the start condition flag 9 (SCF9) in status register 4 (STS4).

Each time the 16-bit counter is enabled, the content of the counter will be cleared automatically.



This figure shows the timing of the counter controlled by the CX pin

Example:

```
SCC         0h          ;Select the base clock of the frequency generator that comes
from
                        ;PH0 (XT clock)
FRQX        1, 5        ;set the frequency generator to FREQ = (PH0/6) / 3
```

```
                        ;the setting value of the frequency generator is 5 and FREQ
                        ;has 1/3 duty waveform.
SHE         40h         ;enable the halt release caused by 16-bit counter
SRF         28h         ;enable the counter controlled by the CX signal
HALT
PLC         40h         ;halt release is caused by the 2nd rising edge on CX pin and
                        ;then clear the halt release request flag
MRF1        10h         ;read the content of the counter
MRF2        11h
MRF3        12h
MRF4        13h
```

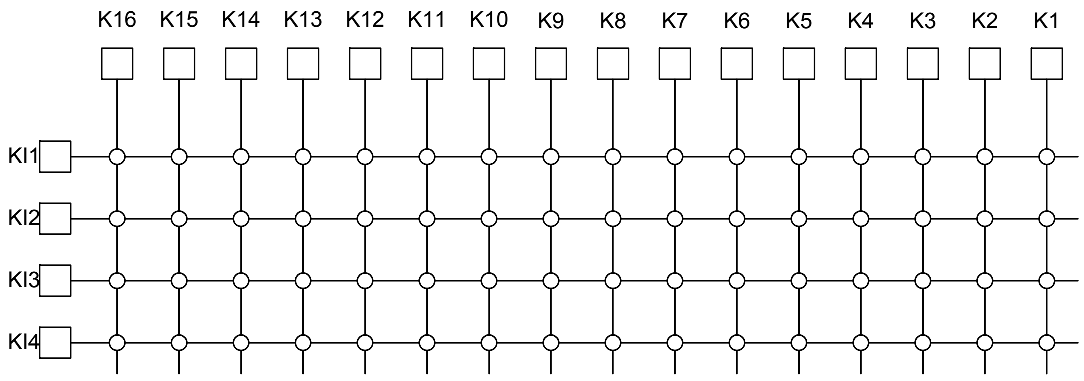
3-8 Key Matrix Scanning

TM8706 shares the timing of the LCD waveform to scan the key matrix circuitry. These scanning output pins are SEG1~16(for easy to understand, named these pins as K1 ~ K16). The time sharing of the LCD waveform will not affect the display of the LCD panel. The input port of the key matrix circuitry is composed of KI1 ~ KI4 pins (these pins are muxed with SEG32 ~ SEG35 pins and selected by mask option).

MASK OPTION table :

Mask Option name	Selected item
SEG32/IOC1/KI1	(3) KI1
SEG33/IOC2/KI2	(3) KI2
SEG34/IOC3/KI3	(3) KI3
SEG35/IOC4/KI4	(3) KI4

The typical application circuit of the key matrix scanning is shown below:
Executing SPKX X, SPK Rx, and SPK @HL instructions could set the scanning type of



the key matrix. The bit pattern of these 3 instructions are shown below :

Instruction	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
SPKX X	X7	X6	X5	X4	X3	X2	X1	X0
SPK Rx	AC3	AC2	AC1	AC0	Rx3	Rx2	Rx1	Rx0
SPK @HL	T@HL7	T@HL6	T@HL5	T@HL4	T@HL3	T@HL2	T@HL1	T@HL0

The following description shows the bit definitions in the operand of the SPKX instruction.

X_6 = " 0 ", when HEF5 is set to 1, the HALT release request (HRF5) will be set to 1 after the key depressed on the key matrix, and then SCF7 will be set to 1.
" 1 ", when HEF5 is set to 1, the HALT released request (HRF5) will be set to 1 after each scanning cycle regardless of key depression, and then SCF7 will be set to 1.

$X_7X_5X_4$ = 000, in this setting, each scanning cycle only checks one specified column (K1 ~ K16) on the key matrix. The specified column is defined by the setting of X_3 ~ X_0.
X_3 ~ X_0 = 0000, activates K1 column
X_3 ~ X_0 = 0001, activates K2 column
……………………………………..
X_3 ~ X_0 = 1110, activates K15 column
X_3 ~ X_0 = 1111, activates K16 column

$X_7X_5X_4$ = 001, in this setting, all of the matrix columns (K1 ~ K16) will be checked simultaneously in each scanning cycle. X_3 ~ X_0 are not a factor.

$X_7X_5X_4$ = 010, in this setting, the key matrix scanning function will be disabled. X_3 ~ X_0 are not a factor.

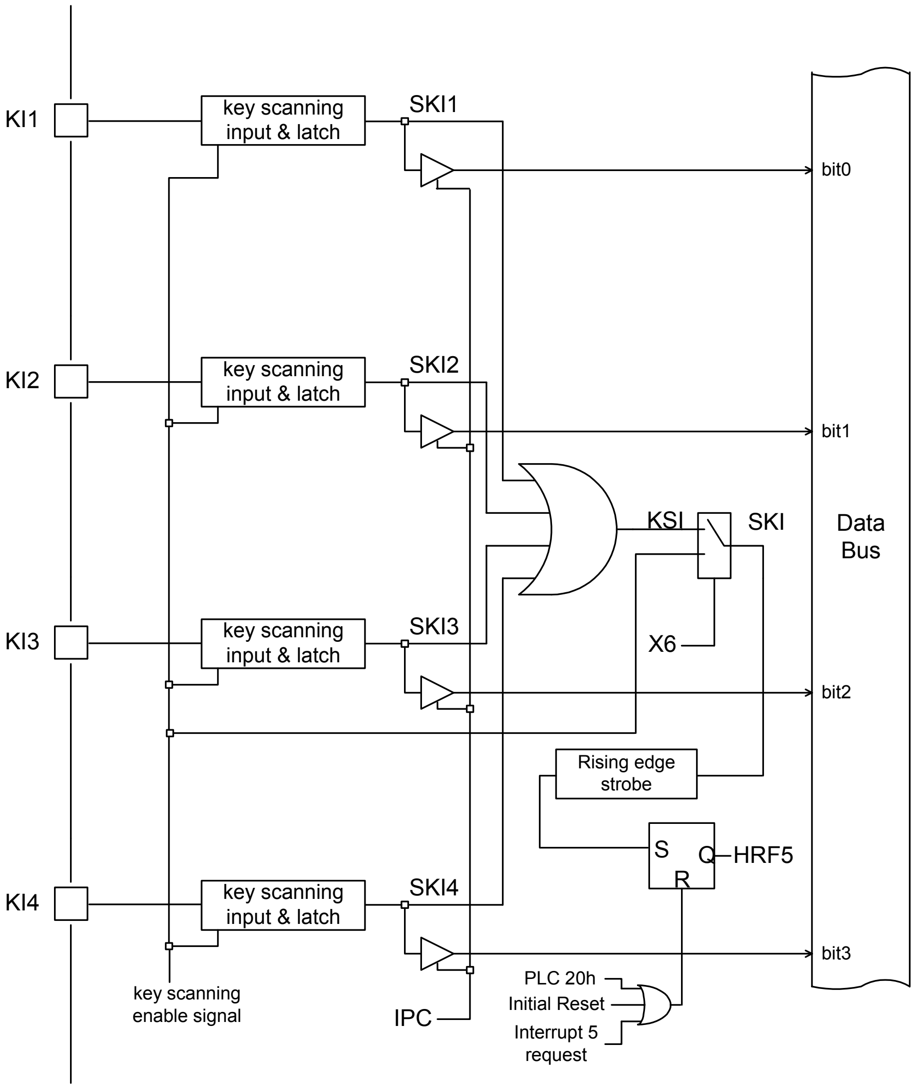
$X_7X_5X_4$ = 10X, in this setting, each scanning cycle checks 8 specified columns on the key matrix. The specified column is defined by the setting of X_3.
X_3 = 0, activates K1 ~ K8 columns simultaneously
X_3 = 1, activates K9 ~ K16 columns simultaneously
X_2 ~ X_0 don't care.

$X_7X_5X_4$ = 110, in this setting, each scanning cycle checks four specified columns on key matrix. The specified columns are defined by the setting of X_3 and X_2.
X_3X_2 = 00, activates K1 ~ K4 columns simultaneously
X_3X_2 = 01, activates K5 ~ K8 columns simultaneously
X_3X_2 = 10, activates K9 ~ K12 columns simultaneously
X_3X_2 = 11, activates K13 ~ K16 columns simultaneously
X_1, X_0 don't care.

$X_7X_5X_4$ = 111, in this setting, each scanning cycle checks two specified columns on key matrix. The specified columns are defined by the setting of X_3, X_2 and X_1.
$X_3X_2X_1$ = 000, activates K1 ~ K2 columns simultaneously
$X_3X_2X_1$ = 001, activates K3 ~ K4 columns simultaneously
…………………………………….
$X_3X_2X_1$ = 110, activates K13 ~ K14 columns simultaneously
$X_3X_2X_1$ = 111, activates K15 ~ K16 columns simultaneously
X_0 is not a factor.

When KI1~4 is defined for the Key matrix scanning input by mask option, it is necessary to execute the SPC instruction to set the internal unused IOC port to output mode before the key matrix scanning function is activated. Fig 2-27 shows the organization of the Key matrix scanning input port. Each one of the SKI1~4 changed to "High" will set HRF5 to 1. If HEF5 has been set to 1 beforehand, this will cause SCF7 to be set, as well as releasing the HALT mode. After the key scanning cycle, the states of SKI1 ~ 4 will be latched and executing the IPC instruction could store these states into data RAM. Executing the PLC 20h instruction clears the HRF5 flag.

Since the key matrix scanning function shares the timing of LCD waveform, the scanning frequency corresponds to the LCD frame frequency and the LCD duty cycle. The formula for the key matrix scanning frequency is shown below :
key matrix scanning frequency (Hz) = (LCD frame frequency) x (LCD duty cycle) x 2
Note : "2" is a factor
For example, if the LCD frame frequency is 32Hz, and duty cycle is 1/5 duty, the scanning frequency for the key matrix is : 320Hz(32 x 5 x 2).



This figure shows the organization of Key matrix scanning input

Example:

SPC 0fh ; Disables all the pull-down devices on the internal IOC port.
; Sets all of the IOC pins as the output mode.
SPKX 10h ;Generates HALT release request when a key is depressed

```
                                ; Scanns every column simultaneously in each cycle.
        PLC         20h         ; Clears HRF5
        SHE         20h         ;Sets HEF5.
        HALT                    ;waits for the halt release caused by the key matrix.
        MCX         10h         ;Checks SCF8 (SKI).
        JB0         ski_release
        ………….
        …………
ski_release:
        IPC         10h         ;reads the KI1~4 input latch state.
        JB0         ki1_release
        JB1         ki2_release
        JB2         ki3_release
        JB3         ki4_release
        .
        .
ki1_release:
        SPKX        40h         ; Checks the key depressed on K1 column.
        PLC         20h         ; Clears HRF5 to avoid the false HALT release
        CALL        wait_scan_again; Waits for the next key matrix scanning cycle.
                                ; The waiting period must be longer than the key matrix
                                  scanning
                                ; cycle.
        IPC         10h         ; Reads  the KI1 input latch state.
        JB0         ki1_seg1
        ………….
        ………….
        SPK         4fh         ; Enables only the SEG16 scanning output.
        PLC         20h         ; Clear HRF5 to avoid the false HALT released
        CALL        wait_scan_again     ; Waits for the time over the halt LCD clock cycle
        to ensure,  and scans again.
        IPC         10h         ; Reads the KI1 input latch state.
        JB0         kil_seg16
        ………….
        ………….
wait_scan_again:
        HALT
        PLC   20h
        RTS
```

CHAPTER 4 LCD DRIVER OUTPUT

There are 41 segment pins with 9 common pins in the LCD driver outputs in TM87P08. All of these output pins can also be used as DC output ports (through the mask option).

MASK OPTION table :

During the initial reset cycle, the LCD lighting system may be lit or extinguished by mask option. All of the LCD or DC output will remain in the initial setting until instructions relative to the LCD are executed to change the output data.

MASK OPTION table :

Mask Option name	Selected item
LCD DISPLAY IN RESET CYCLE	(1) ON
LCD DISPLAY IN RESET CYCLE	(2) OFF

4-1. LCD LIGHTING SYSTEM IN TM87P08

There are several LCD lighting systems that can be selected by mask option in TM87P08, they are :

- 1/2 bias 1/4 duty, 1/2 bias 1/8 duty
- 1/3 bias 1/4 duty, 1/3 bias 1/8 duty

All of these lighting systems are combined with 2 kinds of mask options; one is "LCD DUTY CYCLE" and the other is "BIAS".

MASK OPTION table :

LCD duty cycle option

Mask Option Name	Selected Item
LCD DUTY CYCLE	(4) 1/4 DUTY
LCD DUTY CYCLE	(8) 1/8 DUTY

LCD bias option

Mask Option name	Selected item
BIAS	(2) 1/2 BIAS
BIAS	(3) 1/3 BIAS

The frame frequency for each lighting system is shown below; these frequencies can be selected by mask option. (All of the LCD frame frequencies in the following tables are based on the clock source frequency of the pre-divider (PH0) is 32768Hz).

The LCD alternating frequency in 1/4 duty type

Mask Option name	Selected item	Remark (alternating frequency)
LCD frame frequency	(1) SLOW	16Hz
LCD frame frequency	(2) TYPICAL	32Hz
LCD frame frequency	(2) FAST	64Hz

The LCD alternating frequency in 1/8 duty type

Mask Option name	Selected item	Remark (alternating frequency)
LCD frame frequency	(1) SLOW	32Hz
LCD frame frequency	(2) TYPICAL	64Hz
LCD frame frequency	(2) FAST	128Hz

The following table shows the relationship between the LCD lighting system and the maximum number of driving LCD segments.

LCD Lighting System	Maximum Number of Driving LCD Segments	Remarks
1/2 bias 1/4 duty	128	Connect VL3 to VL2
1/2 bias 1/8 duty	256	Connect VL3 to VL2
1/3 bias 1/4 duty	128	
1/3 bias 1/8 duty	256	

When choosing the LCD frame frequency, it is recommended to chose a frequency higher than 24Hz. If the frame frequency is lower than 24Hz, the pattern on the LCD panel will start to flash.

4-2. DC OUTPUT

TM87P08 permits LCD driver output pins (COM5 ~ COM8 , DC9 and DC30) to be defined as CMOS type DC output or P open-drain DC output ports by mask option. In these cases, it is possible to use some LCD driver output pins as DC output and the rest of the LCD driver output pins as LCD drivers. Refer to 4-3-4.
The configurations of CMOS output type and P open-drain type are shown below.
When the LCD driver output pins (SEG) are defined as DC output ports, the output data on those ports will not be affected when the program enters stop mode or LCD turn-off mode.

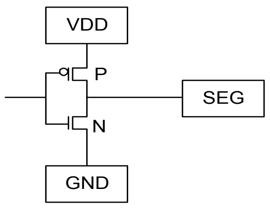


Figure 5-1 CMOS Output Type

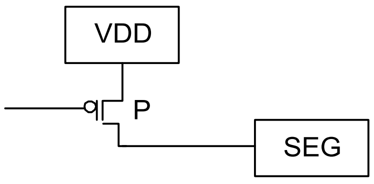


Figure 5-2 P Open-Drain Output Type

Only unused COM and SEG pads can be defined as DC output pins. The COM pad sequence for LCD drivers cannot be interrupted when the COM pads are defined as DC output ports.

For example, when the LCD lighting system is specified as 1/4 duty, the COM pad used for LCD driver must be COM1 ~ COM4. Each of COM6 ~ COM8, DC9 pad can be defined as DC output ports.

4-3. SEGMENT PLA CIRCUIT FOR LCD DISPLAY

4-3-1. PRINCIPLE OF OPERATION OF LCD DRIVER SECTION

The explanation below explains how the LCD driver section operates when the instructions are executed.

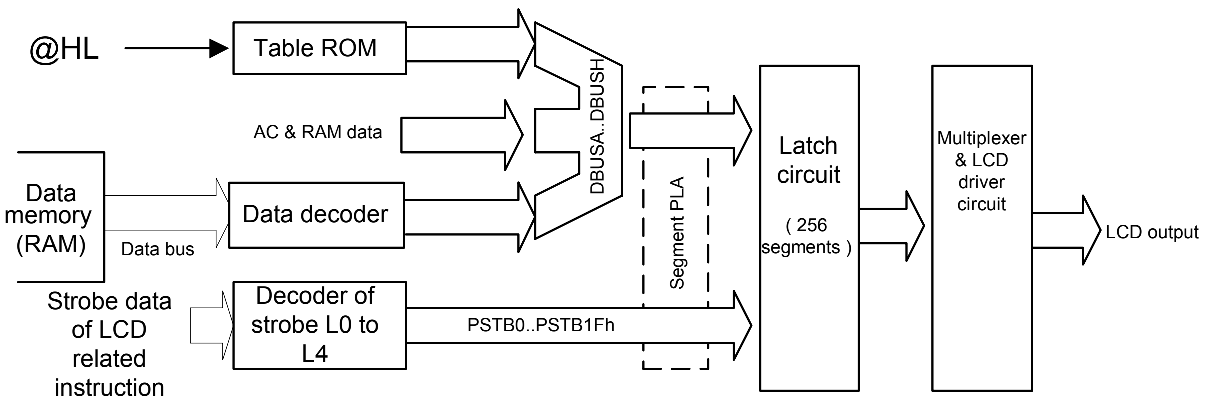


Figure 5-3 Principal Drawing of LCD Driver Section

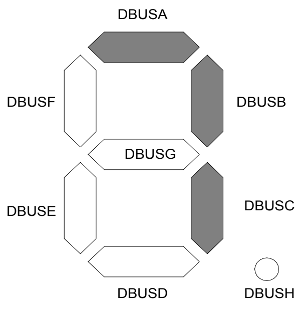
The LCD driver section consists of the following units:

- Data decoder to decode data supplied from RAM or table ROM
- Latch circuit to store LCD lighting information
- L0 to L4 decoder to decode the Lz-specified data in LCD-related instructions which specifies the strobe of the latch circuit
- Multiplexer to select 1/4 duty and 1/8 duty
- LCD driver circuitry
- Segment PLA circuit connected between data decoder, L0 to L4 decoder and latch circuit.

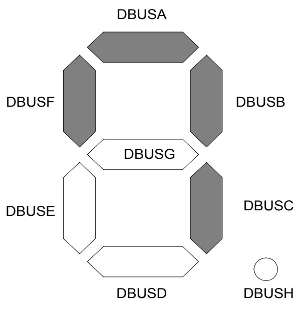
The data decoder is used for decoding the contents of the working registers as specified in LCD-related instructions. They are decoded as 7-segment patterns on the LCD panel. The decoding table is shown below:

Content of data memory	Output of data decoder							
	DBUSA	DBUSB	DBUSC	DBUSD	DBUSE	DBUSF	DBUSG	DBUSH
0	1	1	1	1	1	1	0	1
1	0	1	1	0	0	0	0	1
2	1	1	0	1	1	0	1	1
3	1	1	1	1	0	0	1	1
4	0	1	1	0	0	1	1	1
5	1	0	1	1	0	1	1	1
6	1	0	1	1	1	1	1	1
7	1	1	1	0	0	*note	0	1
8	1	1	1	1	1	1	1	1
9	1	1	1	1	0	1	1	1
A-F	0	0	0	0	0	0	0	0

* Note : The DBUSF of decoded output can be selected as 0 or 1 by mask option. The LCD pattern of this option is shown below :



DBUSF=0



DBUSF=1

The following table shows the options table for displaying the digit "7" pattern:

MASK OPTION table :

Mask Option name	Selected item
F SEGMENT FOR DISPLAY " 7 "	(1) ON
F SEGMENT FOR DISPLAY " 7 "	(2) OFF

Both LCT and LCB instructions use the data-decoder table to decode the content of the specified data memory location. When the content of the data memory location specified by the LCB instruction is "0", the decoded outputs of DBUSA ~ DBUSH are all "0". (this is used for blanking the leading digit "0" on the LCD panel).
The LCP instruction transfers data about the RAM (Rx) and accumulator (AC) directly from " DBUSA" to " DBUSH" without passing through the data decoder.
The LCD instruction transfers the table ROM data (T@HL) directly from "DBUSA" to "DBUSH" without passing through the data decoder.

Table 2- 2 The mapping table of LCP and LCD instructions

	DBUSA	DBUSB	DBUSC	DBUSD	DBUSE	DBUSF	DBUSG	DBUSH
LCP	Rx0	Rx1	Rx2	Rx3	AC0	AC1	AC2	AC3
LCD	T@HL0	T@HL1	T@HL2	T@HL3	T@HL4	T@HL5	T@HL6	T@HL7

There are 8 data decoder outputs from ”DBUSA” to ”DBUSH” and 32 L0 to L4 decoder outputs from PSTB 0h to PSTB 1Fh. The input data and clock signal of the latch circuit are ”DBUSA” to ”DBUSH” and PSTB 0h to PSTB 1Fh, respectively. Each segment pin has 8 latches corresponding to COM1-8.
The segment PLA performs the function of combining ”DBUSA” outputs to ”DBUSH” inputs and then sending them to each latch and strobe; PSTB 0h to PSTB 1Fh is selected freely by mask option.
Of the 512 signals obtainable by combining ”DBUSA” to ”DBUSH” and PSTB 0h to PSTB 1Fh, any one of 256 (corresponding to the number of latch circuits incorporated in the hardware) signals can be selected by programming the aforementioned segment PLA. Table 2-7 shows the PSTB 0h to PSTB 1Fh signals.

Table 2- 3 Strobe Signal for LCD Latch in Segment PLA and Strobe in LCT Instruction

strobe signal for LCD latch	Strobe in LCT, LCB, LCP, LCD instructions The values of Lz in”LCT Lz, Q": *
PSTB0	0H
PSTB1	1H
PSTB2	2H
PSTB3	3H
PSTB4	4H
PSTB5	5H
…………	……………..
PSTB1Ah	1AH
PSTB1Bh	1BH
PSTB1Ch	1CH
PSTB1Dh	1DH
PSTB1Eh	1EH
PSTB1Fh	1FH

Note: The values of Q are the addresses of the working register in the data memory (RAM). In the LCD instruction, Q is the index address in the table ROM.

The LCD outputs can be turned off without changing segment data. The execution of the SF2 4h instruction may turn off the displays simultaneously. The execution of the RF2 4h instruction may turn on the display with the patterns turned off. These two instructions will not affect the data stored in the latch circuitry. When executing the RF2 4h instruction to turn off the LCD, the program can still execute LCT, LCB, LCP and LCD instructions to update the data in the latch circuitry. The new content will be outputted to the LCD while the display is being turned on again.
In the stop state, all COM and SEG outputs of LCD drivers will automatically switch to the GND state to avoid DC voltage bias on the LCD panel.

4-3-2. Relative Instructions

1. LCT Lz, Ry
Decodes the content specified in Ry with the data decoder and transfers the DBUSA ~ H to the LCD latch specified by Lz.

2. LCB Lz, Ry
Decodes the content specified in Ry with the data decoder and transfers the DBUSA ~ H to the LCD latch specified by Lz. "DBUSA" to "DBUSH" are all set to 0 when the input data of the data decoder is 0.

3. LCD Lz, @HL
Transfers the table ROM data specified by @HL directly to "DBUSA" through "DBUSH" without passing through the data decoder. The mapping table is shown in table 2-32.

4. LCP Lz, Ry
The data in the RAM and accumulator (AC) are transferred directly to "DBUSA" through "DBUSH" without passing through the data decoder. The mapping table is shown below:

5. LCT Lz, @HL
Decodes the index RAM data specified in @HL with the data decoder and transfers DBUSA ~ H to the LCD latch specified by Lz.

6. LCB Lz, @HL
Decodes the index RAM data specified in @HL with the data decoder and transfers the DBUSA ~ H to the LCD latch specified by Lz. The "DBUSA" to "DBUSH" are all set to 0 when the input data of the data decoder is 0.

7. LCP Lz, @HL
The data of the index RAM and accumulator (AC) are transferred directly to "DBUSA" through "DBUSH" without passing through the data decoder. The mapping table is shown below:

Table 2- 4 The mapping table of LCP and LCD instructions

	DBUSA	DBUSB	DBUSC	DBUSD	DBUSE	DBUSF	DBUSG	DBUSH
LCP	Rx0	Rx1	Rx2	Rx3	AC0	AC1	AC2	AC3
LCD	T@HL0	T@HL1	T@HL2	T@HL3	T@HL4	T@HL5	T@HL6	T@HL7

5. SF2 4h
Turns off the LCD display.

6. RF2 4h
Turns on the LCD display.

4-3-3. CONCRETE EXPLANATION

Each LCD driver output corresponds to the LCD 1/98duty panel and has 8 latches (refer to Figure : 4-3 Sample Organization of Segment PLA Option). Since the latch input and the signal to be applied to the clock (strobe) are selected with the segment PLA, the combination of segments in the LCD driver outputs is fixed. In other words, one of the data decoder outputs from "DBUSA" to "DBUSH" is applied to the latch input L, and one of the PSTB0 to PSTB 1Fh outputs is applied to clock CLK.

TM87P08 provide a flash type instruction to update the LCD pattern. When the LCTX D, LCBX D, LCPX D and LCDX D instructions are executed, the pattern of DBUS will be outputted to the 16 latches (Lz) specified by D simultaneously.

D	Specified range of latched
00	Lz = 00h ~ 0Fh
01	Lz = 10h ~ 1Fh

Refer to Chapter 5 for detailed description of these instructions.

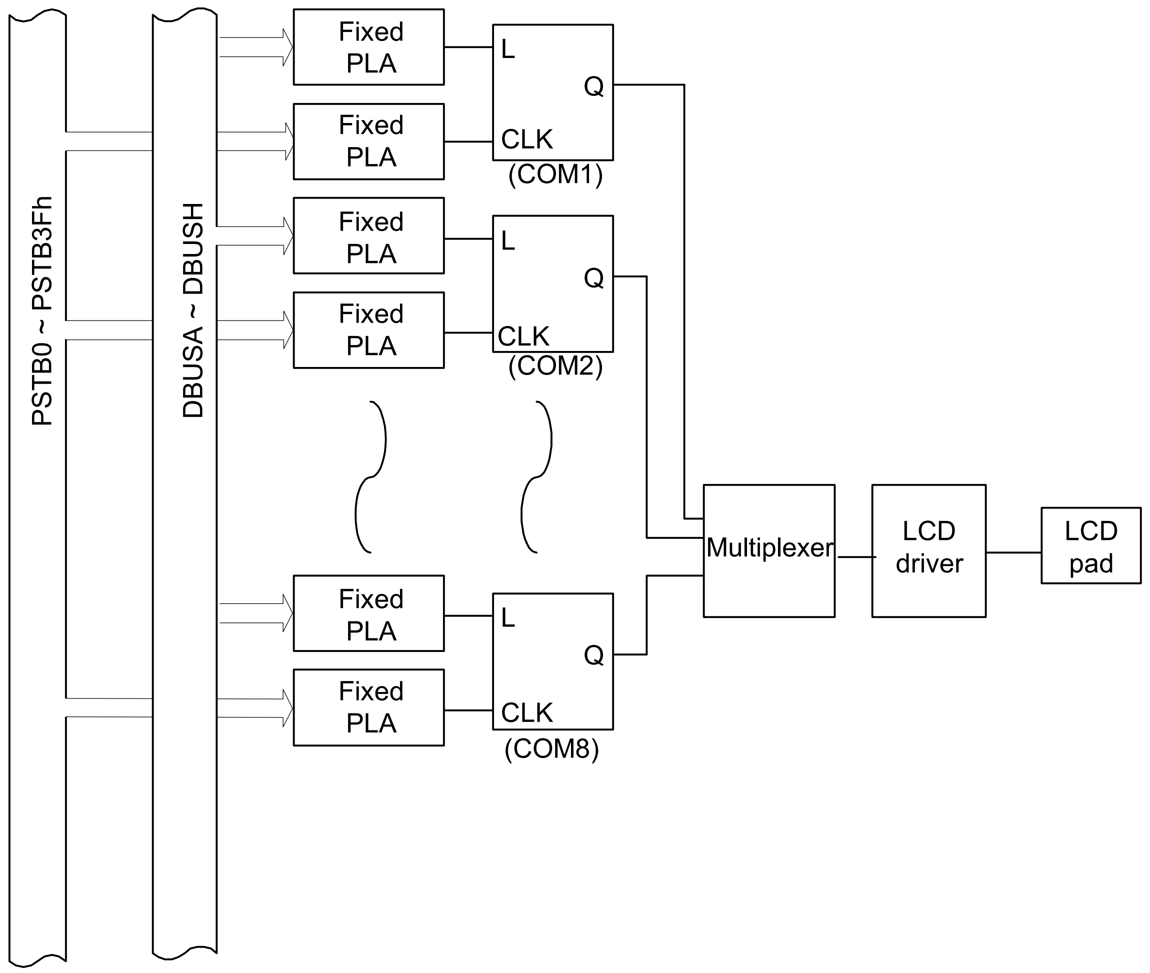


Figure : 4-3 Sample Organization of Segment PLA Option

4-3-4. THE CONFIGURATION FORMAT FOR FIXED PLA MAPPING

The TM87P08 is fixed PLA structure

PLA table is shown below:

	< 1/8 Duty >								
	< 1/4 Duty >								
SEGx	**Lz**	COM1	COM2	COM3	COM4	**Lz**	COM5	COM6	COM7	COM8
SEG1	00H	DBUSA	DBUSB	DBUSC	DBUSD	10H	DBUSA	DBUSB	DBUSC	DBUSD
SEG2		DBUSE	DBUSF	DBUSG	DBUSH		DBUSE	DBUSF	DBUSG	DBUSH
SEG3	01H	DBUSA	DBUSB	DBUSC	DBUSD	11H	DBUSA	DBUSB	DBUSC	DBUSD
SEG4		DBUSE	DBUSF	DBUSG	DBUSH		DBUSE	DBUSF	DBUSG	DBUSH
SEG5	02H	DBUSA	DBUSB	DBUSC	DBUSD	12H	DBUSA	DBUSB	DBUSC	DBUSD
SEG6		DBUSE	DBUSF	DBUSG	DBUSH		DBUSE	DBUSF	DBUSG	DBUSH
SEG7	03H	DBUSA	DBUSB	DBUSC	DBUSD	13H	DBUSA	DBUSB	DBUSC	DBUSD
SEG8		DBUSE	DBUSF	DBUSG	DBUSH		DBUSE	DBUSF	DBUSG	DBUSH
SEG9	04H	DBUSA	DBUSB	DBUSC	DBUSD	14H	DBUSA	DBUSB	DBUSC	DBUSD
SEG10		DBUSE	DBUSF	DBUSG	DBUSH		DBUSE	DBUSF	DBUSG	DBUSH
SEG11	05H	DBUSA	DBUSB	DBUSC	DBUSD	15H	DBUSA	DBUSB	DBUSC	DBUSD
SEG12		DBUSE	DBUSF	DBUSG	DBUSH		DBUSE	DBUSF	DBUSG	DBUSH
SEG13	06H	DBUSA	DBUSB	DBUSC	DBUSD	16H	DBUSA	DBUSB	DBUSC	DBUSD
SEG14		DBUSE	DBUSF	DBUSG	DBUSH		DBUSE	DBUSF	DBUSG	DBUSH
SEG15	07H	DBUSA	DBUSB	DBUSC	DBUSD	17H	DBUSA	DBUSB	DBUSC	DBUSD
SEG16		DBUSE	DBUSF	DBUSG	DBUSH		DBUSE	DBUSF	DBUSG	DBUSH
SEG17	08H	DBUSA	DBUSB	DBUSC	DBUSD	18H	DBUSA	DBUSB	DBUSC	DBUSD
SEG18		DBUSE	DBUSF	DBUSG	DBUSH		DBUSE	DBUSF	DBUSG	DBUSH
SEG19	09H	DBUSA	DBUSB	DBUSC	DBUSD	19H	DBUSA	DBUSB	DBUSC	DBUSD
SEG20		DBUSE	DBUSF	DBUSG	DBUSH		DBUSE	DBUSF	DBUSG	DBUSH
SEG21	0AH	DBUSA	DBUSB	DBUSC	DBUSD	1AH	DBUSA	DBUSB	DBUSC	DBUSD
SEG22		DBUSE	DBUSF	DBUSG	DBUSH		DBUSE	DBUSF	DBUSG	DBUSH
SEG23	0BH	DBUSA	DBUSB	DBUSC	DBUSD	1BH	DBUSA	DBUSB	DBUSC	DBUSD
SEG24		DBUSE	DBUSF	DBUSG	DBUSH		DBUSE	DBUSF	DBUSG	DBUSH
SEG25	0CH	DBUSA	DBUSB	DBUSC	DBUSD	1CH	DBUSA	DBUSB	DBUSC	DBUSD
SEG26		DBUSE	DBUSF	DBUSG	DBUSH		DBUSE	DBUSF	DBUSG	DBUSH
SEG27	0DH	DBUSA	DBUSB	DBUSC	DBUSD	1DH	DBUSA	DBUSB	DBUSC	DBUSD
SEG28		DBUSE	DBUSF	DBUSG	DBUSH		DBUSE	DBUSF	DBUSG	DBUSH
SEG29	0EH	DBUSA	DBUSB	DBUSC	DBUSD	1EH	DBUSA	DBUSB	DBUSC	DBUSD
SEG31		DBUSE	DBUSF	DBUSG	DBUSH		DBUSE	DBUSF	DBUSG	DBUSH
SEG40	0FH	DBUSA	DBUSB	DBUSC	DBUSD	1FH	DBUSA	DBUSB	DBUSC	DBUSD
SEG41		DBUSE	DBUSF	DBUSG	DBUSH		DBUSE	DBUSF	DBUSG	DBUSH

Lz	**1/4 Duty**			**Lz**	**1/8 Duty**	
1FH	**DC5/OD5**	**DC6/OD6**	**DC7/OD7**	1FH	**DC9/OD9**	**DC30/OD30**
	DBUSA	DBUSB	DBUSC			
	DC8/OD8	**DC9/OD9**	DC30/OD30		DBUSE	DBUSH
	DBUSD	DBUSE	DBUSH			

Chapter 5 Detail Explanation of TM87P08 Instructions

- Before using the data memory, it is necessary to initiate the content of data memory because the initial value is unknown.
- The working registers are part of the data memory (RAM), and the relationship between them can be shown as follows:
  [The absolute address of working register Rx=Ry+70H]*

Note: Ry: Address of working register, the range of addresses specified by Rx is from 70H to 7FH.
Rx: Address of data memory, the range of addresses specified by Ry is from 0H to FH.
Ry use for LCD instruction only 0H~7H

Address of working registers specified by Ry	Absolute address of data memory (Rx)
0H	70H
1H	71H
2H	72H
. .	. .
DH	7DH
EH	7EH
FH	7FH

- Lz represents the address of the latch of LCD PLA; the address range specified by Lz is from 00H to 1FH.

5-1 INPUT / OUTPUT INSTRUCTIONS

LCT Lz, Ry

function:	LCD latch Lz ← data decoder ← (Ry)
description:	The working register contents specified by Ry are loaded to the LCD latch specified by Lz through the data decoder.

LCB Lz, Ry

function:	LCD latch Lz ← data decoder ← (Ry)
description:	The working register contents specified by Ry are loaded to the LCD latch specified by Lz through the data decoder. If the content of Ry is "0", the outputs of the data decoder are all "0".

LCP Lz, Ry

function:	LCD latch Lz ← (Ry), (AC)
description:	The working register contents specified by Ry and the contents of AC are loaded to the LCD latch specified by Lz.

LCD Lz, @HL

function:	LCD latch Lz ← (T@HL)

description: @HL indicates an index address of table ROM.
The contents of table ROM specified by @HL are loaded to the LCD latch specified by Lz directly.

LCT Lz, @HL
function: LCD latch Lz ← data decoder ← (R@HL)
description: The contents of index RAM specified by @HL are loaded to the LCD latch specified by Lz through the data decoder.

LCB Lz, @HL
function: LCD latch Lz ← data decoder ← (R@HL)
description: The contents of index RAM specified by @HL are loaded to the LCD latch specified by Lz through the data decoder.
If the content of @HL is "0", the outputs of the data decoder are all "0".

LCP Lz, @HL
function: LCD latch Lz ← (R@HL), (AC)
description: The contents of index RAM specified by @HL and the contents of AC are loaded to the LCD latch specified by Lz.

LCDX D
Function : Mullti-LCD latches [Lz(s)] ← TAB[@HL]
Description : @HL indicates an index address of table ROM.
The content of table ROM, specified by @HL, are loaded to several LCD latches(Lz) simultaneously. Refer to Table 5-2. The range of multi-Lz is specified by data "D".
D : 0 ~ 3.

D=0	Multi-Lz=00H~0FH
D=1	Multi-Lz=10H~1FH

Table 5-2 The range of multi-Lz latches

LCTX D
Function : Mullti-LCD latch [Lz] ← data decoder ← [@HL]
Description : The contents of index RAM, specified by @HL, are loaded to several LCD latches(Lz) simultaneously. The range of multi-Lz is specified by data "D". Refer to Table 5-2.
D : 0 ~ 3.

LCBX D
Function : Mullti- LCD latch [Lz] ← data decoder ← [@HL]
Description : The contents of index RAM, specified by @HL, are loaded to the LCD latch specified by Lz through the data decoder. The range of multi-Lz is specified by data "D". Refer to Table 5-2.

D : 0 ~ 3.

LCPX D

Function : Mullti- LCD latch [Lz] ← [@HL],AC

Description : The contents of index RAM, specified by @HL, and the contents of AC are loaded to several LCD latches(Lz) simultaneously. Refer to Table 5-2. The range of multi-Lz is specified by data "D".
D : 0 ~ 3.

SPA X

function: Defines the input/output mode of each pin for IOA port and enables / disables the pull-low device.

description: Sets the I/O mode and turns on/off the pull-low device. The input pull-low device will be enabled when the I/O pin was set as input mode.
The meaning of each bit of X(X3 X2 X1 X0) is shown below:

Bit pattern	Setting	Bit pattern	Setting
X4=1	Enable IOA pull low R	X4=0	Disable IOA pull low R
X3=1	IOA4 as output mode	X3=0	IOA4 as input mode
X2=1	IOA3 as output mode	X2=0	IOA3 as input mode
X1=1	IOA2 as output mode	X1=0	IOA2 as input mode
X0=1	IOA1 as output mode	X0=0	IOA1 as input mode

OPA Rx

function: I/OA ← (Rx)

description: The content of Rx is outputted to I/OA port.

OPAS Rx, D

function: IOA1,2 ← (Rx), IOA3 ← D, IOA4 ← pulse

description: Content of Rx is outputted to IOA port. D is outputted to IOA3, pulse is outputted to IOA4.
D = 0 or 1

IPA Rx

function: Rx, AC ← (IOA)

description: The data of I/OA port is loaded to AC and data memory Rx.

SPB X

function: Defines the input/output mode of each pin for IOB port and enables / disables the pull-low device.

description: Sets the I/O mode and turns on/off the pull-low device. The input pull-low device will be enabled when the I/O pin was set as input mode.

The meaning of each bit of X(X3 X2 X1 X0) is shown below:

Bit pattern	Setting	Bit pattern	Setting
X4=1	Enable IOB pull low R	X4=0	Disable IOB pull low R
X3=1	IOB4 as output mode	X3=0	IOB4 as input mode
X2=1	IOB3 as output mode	X2=0	IOB3 as input mode
X1=1	IOB2 as output mode	X1=0	IOB2 as input mode
X0=1	IOB1 as output mode	X0=0	IOB1 as input mode

OPB Rx

function: I/OB ← (Rx)

description: The contents of Rx are outputted to I/OB port.

IPB Rx

function: Rx, AC ← (IOB)

description: The data of I/OB port is loaded to AC and data memory Rx.

SPC X

function: Defines the input/output mode of each pin for IOC port and enables / disables the pull-low device or low-level-hold device.

description: Sets the I/O mode and turns on/off the pull-low device. The input pull-low device will be enabled when the I/O pin was set as input mode.

The meaning of each bit of X(X4 X3 X2 X1 X0) is shown below:

Bit pattern	Setting	Bit pattern	Setting
X4=1	Enables all of the pull-low and disables the low-level hold devices	X4=0	Disables all of the pull-low and enables the low-level hold devices
X3=1	IOC4 as output mode	X3=0	IOC4 as input mode
X2=1	IOC3 as output mode	X2=0	IOC3 as input mode
X1=1	IOC2 as output mode	X1=0	IOC2 as input mode
X0=1	IOC1 as output mode	X0=0	IOC1 as input mode

OPC Rx

function: I/OC ← (Rx)

description: The content of Rx is outputted to I/OC port.

IPC Rx

function: Rx, AC ← (IOC)

description: The data of I/OC port is loaded to AC and data memory Rx.

SPD X

Function : Defines the input/output mode of each pin for IOD port and enables or disables the pull-low device.

Description : Sets the I/O mode and turns the pull-low device on or off. The meaning of each bit of X(X4, X3, X2, X1, X0) is shown below:

Bit pattern	Setting	Bit pattern	Setting
X4=1	Enable the pull-low device on IOD1~IOD4 simultaneously	X4=0	Disable the pull-low device on IOD1~IOD4 simultaneously
X3=1	IOD4 as output mode	X3=0	IOD4 as input mode
X2=1	IOD3 as output mode	X2=0	IOD3 as input mode
X1=1	IOD2 as output mode	X1=0	IOD2 as input mode
X0=1	IOD1 as output mode	X0=0	IOD1 as input mode

OPD Rx

Function : I/OD ← [Rx]

Description : The content of Rx is outputted to I/OD port.

IPD Rx

Function : [Rx], AC ← [I/OD]

Description : The data of the I/OD port is loaded to AC and data memory Rx.

SPKX X

Function : Sets the Key matrix scanning output state.

Description : When SEG1~16 is(are) used for LCD driver pin(s), set X(X7~0) to specify the key matrix scanning output state for each SEGn pin in the scanning interval.

X_6 = " 0 ", when HEF5 is set to 1, the HALT released request (HRF5) will be set to 1 after the key is depressed on the key matrix, and then SCF7 will be set to 1.

" 1 ", when HEF5 is set to 1, the HALT released request (HRF5) will be set to 1 after each scanning cycle regardless of key depression, and then SCF7 will be set to 1.

$X_7X_5X_4$ = 000, in this setting, each scanning cycle only checks one specified column (K1 ~ K16) on the key matrix. The specified column is defined by the setting of X_3 ~ X_0.

X_3 ~ X_0 = 0000, activates the K1 column

X_3 ~ X_0 = 0001, activates the K2 column

……………………………………..

X_3 ~ X_0 = 1110, activates the K15 column

X_3 ~ X_0 = 1111, activates the K16 column

$X_7X_5X_4$ = 001, in this setting, all of the matrix columns (K1 ~ K16) will be checked simultaneously in each scanning cycle. X_3 ~ X_0 are not a factor.

$X_7X_5X_4$ = 010, in this setting, the key matrix scanning function will be disabled. X_3 ~ X_0 are not a factor.

$X_7X_5X_4$ = 10X, in this setting, each scanning cycle checks 8 specified columns on the key matrix. The specified column is defined by the setting of X_3.
X_3 = 0, activates the K1 ~ K8 columns simultaneously
X_3 = 1, activates the K9 ~ K16 columns simultaneously
(X_2 ~ X_0 are not a factor)

$X_7X_5X_4$ = 110, in this setting, each scanning cycle checks four specified columns on the key matrix. The specified columns are defined by the setting of X_3 and X_2.
X_3X_2 = 00, activates the K1 ~ K4 columns simultaneously
X_3X_2 = 01, activates the K5 ~ K8 columns simultaneously
X_3X_2 = 10, activates the K9 ~ K12 columns simultaneously
X_3X_2 = 11, activates the K13 ~ K16 columns simultaneously
(X_1, X_0 are not a factor)

$X_7X_5X_4$ = 111, in this setting, each scanning cycle checks two specified columns on the key matrix. The specified columns are defined by the setting of X_3, X_2 and X_1.
$X_3X_2X_1$ = 000, activates the K1 ~ K2 columns simultaneously
$X_3X_2X_1$ = 001, activates the K3 ~ K4 columns simultaneously
……………………………………
$X_3X_2X_1$ = 110, activates the K13 ~ K14 columns simultaneously
$X_3X_2X_1$ = 111, activates the K15 ~ K16 columns simultaneously
(X_0 is not a factor)

SPK Rx

Function :	Sets the Key matrix scanning output state.
Description :	When SEG1~16 is(are) used for LCD driver pin(s), sets the contents of AC and Rx to specify the key matrix scanning output state for each SEGn pin in the scanning interval. The bit setting is the same as the SPKX instruction. The bit patterns of AC and Rx corresponding to SPKX are shown below:

Instruction	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
SPK Rx	AC3	AC2	AC1	AC0	Rx3	Rx2	Rx1	Rx0
SPKX X	X7	X6	X5	X4	X3	X2	X1	X0

SPK @HL

Function :	Sets the Key matrix scanning output state.
Description :	When SEG1~16 is(are) used for LCD driver pin(s), sets the content of table ROM([@HL]) to specify the key matrix scanning output state for each SEGn pin in the scanning interval. The bit setting is the same as the SPKX instruction. The bit pattern of the table ROM corresponding to SPKX is shown below:

Instruction	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
SPK @HL	(T@HL)7	(T@HL)6	(T@HL)5	(T@HL)4	(T@HL)3	(T@HL)2	(T@HL)1	(T@HL)0
SPKX X	X7	X6	X5	X4	X3	X2	X1	X0

ALM X

function:	Sets buzzer output frequency.
description:	The waveform specified by X(X8 ~ X0) is delivered to the BZ and BZB pins.The output frequency could be any combination in the following table.

The bit pattern of X (for higher frequency clock source):

X8	X7	X6	clock source (higher frequency)
1	1	1	FREQ*
1	0	0	DC1
0	1	1	ϕ3(4KHz)
0	1	0	ϕ4(2KHz)
0	0	1	ϕ5(1KHz)
0	0	0	DC0

The bit pattern of X(for lower frequency clock source)*:

Bit	clock source(lower frequency)
X5	ϕ15(1Hz)
X4	ϕ14(2Hz)
X3	ϕ13(4Hz)
X2	ϕ12(8Hz)
X1	ϕ11(16Hz)
X0	ϕ10(32Hz)

Notes: 1. FREQ is the output of frequency generator.

2. When the buzzer output does not need the envelope waveform, X5 ~ X0 should be set to 0.

3. The frequency inside the () bases on the ϕ0 is 32768Hz.

SRF X

function: The operation control for RFC.

description: The meaning of each control bit(X5 ~ X0) is shown below:

X0=1	enables the RC oscillation network of RR	X0=0	disables the RC oscillation network of RR
X1=1	enables the RC oscillation network of RT	X1=0	disables the RC oscillation network of RT
X2=1	enables the RC oscillation network of RH	X2=0	disables the RC oscillation network of RH
X3=1	enables the 16-bit counter	X3=0	disables the 16-bit counter
X4=1	Timer 2 controls the 16-bit counter. X3 must be set to 1 when this bit is set to 1.	X4=0	Disables timer 2 to control the 16-bit counter.
X5=1	The 16-bit counter is controlled by the signal on CX pin. X3 must be set to 1 when this bit is set to 1.	X5=0	Disables the CX pin to control the 16-bit counter.

Note: X4 and X5 can not be set to 1 at the same time.

5-2 ACCUMULATOR MANIPULATION INSTRUCTIONS AND MEMORY MANIPULATION INSTRUCTIONS

MRW Ry, Rx

function: AC, Rx ← (Rx)

description: The content of Rx is loaded to AC and the working register specified by Ry.

MRW @HL, Rx

function: AC, R@HL ← (Rx)

description: The content of data memory specified by Rx is loaded to AC and data memory specified by @HL.

MRW# @HL, Rx

Function : AC, R[@HL] ← [Rx], @HL ← HL + 1

Description : The content of data memory specified by Rx is loaded to AC and the data memory specified by @HL.
The content of the index register (@HL) will be incremented automatically after executing this instruction.

MWR Rx, Ry

function: AC, Rx ← (Ry)

description: The content of working register specified by Ry is loaded to AC and data memory specified by Rx.

MWR Rx, @HL

function:	AC, Rx ← (R@HL)
description:	The content of data memory specified by @HL is loaded to AC and data memory specified by Rx.

MWR# Rx, @HL

Function :	AC, [Rx] ← R[@HL] , @HL ← HL + 1
Description :	The content of the data memory specified by @HL is loaded to AC and the data memory specified by Rx. The content of the index register (@HL) will be incremented automatically after executing this instruction.

SR0 Rx

function:	Rxn, ACn ← Rx(n+1),AC(n+1) Rx3, AC3 ← 0
description:	The Rx content is shifted right and 0 is loaded to the MSB. The result is loaded to the AC. 0 → Rx3 → Rx2 → Rx1 → Rx0 →

SR1 Rx

function:	Rxn, ACn ← Rx(n+1),AC(n+1) Rx3, AC3 ← 1
description:	The Rx content is shifted right and 1 is loaded to the MSB. The result is loaded to the AC. 1 → Rx3 → Rx2 → Rx1 → Rx0 →

SL0 Rx

function:	Rxn, ACn ← Rx(n-1),AC(n-1) Rx0, AC0 ← 0
description:	The Rx content is shifted left and 0 is loaded to the LSB. The results are loaded to the AC. ← Rx3 ← Rx2 ← Rx1 ← Rx0 ← 0

SL1 Rx

function:	Rxn, ACn ← Rx(n-1),AC(n-1) Rx0, AC0 ← 1
description:	The Rx content is shifted left and 1 is loaded to the LSB. The results are loaded to the AC. ← Rx3 ← Rx2 ← Rx1 ← Rx0 ← 1

MRA Rx

function:	CF ← (Rx)3
description:	Bit3 of the content of Rx is loaded to carry flag(CF).

MAF Rx

function: AC,Rx ← CF

description: The content of CF is loaded to AC and Rx. The content of AC and meaning of bit after execution of this instruction are as follows:

Bit 3 .... CF
Bit 2 .... (AC)=0, zero flag
Bit 1 .... (No Use)
Bit 0 .... (No Use)

5-3 OPERATION INSTRUCTIONS

INC* Rx

function: Rx,AC ← (Rx)+1

description: Add 1 to the content of Rx; the result is loaded to data memory Rx and AC.
* Carry flag (CF) will be affected.

INC* @HL

function: R@HL,AC ← (R@HL)+1

description: Add 1 to the content of data memory specified by @HL; the result is loaded to data memory specified by @HL and AC.
* Carry flag (CF) will be affected.

INC*# @HL

Function : [@HL],AC ← R[@HL]+1, @HL ← HL + 1

Description : Adds 1 to the content of @HL; the result is loaded to the data memory @HL and AC. The content of the index register (@HL) will be incremented automatically after executing this instruction.
* The carry flag (CF) will be affected.
• @HL indicates an index address of data memory.

DEC* Rx

function: Rx, AC ← (Rx)-1

description: Substrate 1 from the content of Rx; the result is loaded to data memory Rx and AC.
• Carry flag (CF) will be affected.

DEC* @HL

function: R@HL, AC ← (R@HL)-1

description: Substrate 1 from the content of data memory specified by @HL; the result is loaded to data memory specified by @HL and AC.
* Carry flag (CF) will be affected.

DEC*# @HL

Function :	R@HL, AC ← R[@HL] -1, @HL ← HL + 1
Description :	Substrates 1 from the content of @HL; the result is loaded to the data memory @HL and AC. The content of the index register (@HL) will be incremented automatically after executing this instruction. * The carry flag (CF) will be affected. • @HL indicates an index address of data memory.

ADC Rx

function:	AC ← (Rx)+(AC)+CF
description:	The contents of Rx, AC and CF are binary-added; the result is loaded to AC. * Carry flag (CF) will be affected.

ADC @HL

function:	AC ← (R@HL)+(AC)+CF
description:	The contents of data memory specified by @HL, AC and CF are binary-added; the result is loaded to AC. * Carry flag (CF) will be affected.

ADC# @HL

Function :	AC ← [@HL]+AC+CF, @HL ← HL + 1
Description :	Binary-adds the contents of @HL,AC and CF; the result is loaded to AC. The content of the index register (@HL) will be incremented automatically after executing this instruction. * The carry flag (CF) will be affected. . @HL indicates an index address of data memory.

ADC* Rx

function:	AC, Rx ← (Rx)+(AC)+CF
description:	The contents of Rx, AC and CF are binary-added; the result is loaded to AC and data memory Rx. * Carry flag (CF) will be affected.

ADC* @HL

function:	AC,R@HL ← (R@HL)+(AC)+CF
description:	The contents of data memory specified by @HL,AC and CF are binary-added; the result is loaded to AC and data memory specified by @HL. * Carry flag (CF) will be affected.

ADC*# @HL

Function :	AC, [@HL] ← [@HL]+AC+CF, @HL ← HL + 1

Description :	Binary-adds the contents of @HL,AC and CF; the result is loaded to AC and the data memory @HL. The content of the index register (@HL) will be incremented automatically after executing this instruction. * The carry flag (CF) will be affected. . @HL indicates an index address of data memory.

SBC Rx

function:	AC ← (Rx)+ (AC)B+CF
description:	The contents of AC and CF are binary-subtracted from content of Rx; the result is loaded to AC. . Carry flag (CF) will be affected.

SBC @HL

function:	AC ← (R@HL)+ (AC)B+CF
description:	The contents of AC and CF are binary-subtracted from content of data memory specified by @HL; the result is loaded to AC. * Carry flag (CF) will be affected.

SBC# @HL

Function :	AC ← [@HL]+ (AC)B+CF, @HL ← HL + 1
Description :	Binary-subtracts the contents of AC and CF from the content of @HL; the result is loaded to AC. The content of the index register (@HL) will be incremented automatically after executing this instruction. . @HL indicates an index address of data memory. * The carry flag (CF) will be affected.

SBC* Rx

function:	AC, Rx ← (Rx)+(AC)B+CF
description:	The contents of AC and CF are binary-subtracted from content of Rx; the result is loaded to AC and data memory Rx. . Carry flag (CF) will be affected.

SBC* @HL

function:	AC,R@HL ← (R@HL)+ (AC)B+CF
description:	The contents of AC and CF are binary-subtracted from content of data memory specified by @HL; the result is loaded to AC and data memory specified by @HL. * Carry flag (CF) will be affected.

SBC*# @HL

Function :	AC,[@HL] ← [@HL]+ (AC)B+CF, @HL ← HL + 1
Description :	Binary-subtracts the contents of AC and CF from the content of @HL; the result is loaded to AC and the data memory @HL. The content of

	the index register (@HL) will be incremented automatically after executing this instruction. . @HL indicates an index address of data memory. * The carry flag (CF) will be affected.

ADD Rx

Function :	AC $\leftarrow$ [Rx]+AC
Description :	Binary-adds the contents of Rx and AC; the result is loaded to AC. * The carry flag (CF) will be affected.

ADD @HL

Function :	AC $\leftarrow$ [@HL]+AC
Description :	Binary-adds the contents of @HL and AC; the result is loaded to AC. . @HL indicates an index address of data memory. * The carry flag (CF) will be affected.

ADD# @HL

Function :	AC $\leftarrow$ [@HL]+AC, @HL $\leftarrow$ HL + 1
Description :	Binary-adds the contents of @HL and AC; the result is loaded to AC. The content of the index register (@HL) will be incremented automatically after executing this instruction. . @HL indicates an index address of data memory. * The carry flag (CF) will be affected.

ADD* Rx

Function :	AC, [Rx] $\leftarrow$ [Rx]+AC
Description :	Binary-adds the contents of Rx and AC; the result is loaded to AC and the data memory Rx. * The carry flag (CF) will be affected.

ADD* @HL

Function :	AC,[@HL] $\leftarrow$ [@HL]+AC
Description :	Binary-adds the contents of @HL and AC; the result is loaded to AC and the data memory @HL. . @HL indicates an index address of data memory. * The carry flag (CF) will be affected.

ADD*# @HL

Function :	AC,[@HL] $\leftarrow$ [@HL]+AC, @HL $\leftarrow$ HL + 1
Description :	Binary-adds the contents of @HL and AC; the result is loaded to AC and the data memory @HL. The content of the index register (@HL) will be incremented automatically after executing this instruction. . @HL indicates an index address of data memory. * The carry flag (CF) will be affected.

SUB Rx

Function : AC ← [Rx]+ (AC)B+1

Description : Binary-subtracts the content of AC from the content of Rx; the result is loaded to AC.
* The carry flag (CF) will be affected.

SUB @HL

Function : AC ← [@HL]+ (AC)B+1

Description : Binary-subtracts the content of AC from the content of @HL; the result is loaded to AC.
. @HL indicates an index address of data memory.
* The carry flag (CF) will be affected.

SUB# @HL

Function : AC ← [@HL]+ (AC)B+1, @HL ← HL + 1

Description : Binary-subtracts the content of AC from the content of @HL; the result is loaded to AC. The content of the index register (@HL) will be incremented automatically after executing this instruction.
. @HL indicates an index address of data memory.
* The carry flag (CF) will be affected.

SUB* Rx

Function : AC,[Rx] ← [Rx]+ (AC)B+1

Description : Binary-subtracts the content of AC from the content of Rx; the result is loaded to AC and Rx.
* The carry flag (CF) will be affected.

SUB* @HL

Function : AC, [@HL] ← [@HL]+ (AC)B+1

Description : Binary-subtracts the content of AC from the content of @HL; the result is loaded to AC and the data memory @HL.
. @HL indicates an index address of data memory.
* The carry flag (CF) will be affected.

SUB*# @HL

Function : AC, [@HL] ← [@HL]+ (AC)B+1, @HL ← HL + 1

Description : Binary-subtracts the content of AC from the content of @HL; the result is loaded to AC and the data memory @HL. The content of the index register (@HL) will be incremented automatically after executing this instruction.
. @HL indicates an index address of data memory.
* The carry flag (CF) will be affected.

ADN Rx

Function : AC ← [Rx]+AC

Description : Binary-adds the contents of Rx and AC; the result is loaded to AC.

* The result will not affect the carry flag (CF).

ADN @HL

Function : AC ← [@HL]+AC

Description : Binary-adds the contents of @HL and AC; the result is loaded to AC.
* The result will not affect the carry flag (CF).
. @HL indicates an index address of data memory.

ADN# @HL

Function : AC ← [@HL]+AC, @HL ← HL + 1

Description : Binary-adds the contents of @HL and AC; the result is loaded to AC. The content of the index register (@HL) will be incremented automatically after executing this instruction.
* The result will not affect the carry flag (CF).
. @HL indicates an index address of data memory.

ADN* Rx

Function : AC, [Rx] ← [Rx]+AC

Description : Binary-adds the contents of Rx and AC; the result is loaded to AC and data memory Rx.
* The result will not affect the carry flag (CF).

ADN* @HL

Function : AC, [@HL] ← [@HL]+AC

Description : Binary-adds the contents of @HL and AC; the result is loaded to AC and the data memory @HL.
* The result will not affect the carry flag (CF).
. @HL indicates an index address of data memory.

ADN*# @HL

Function : AC, [@HL] ← [@HL]+AC, @HL ← HL + 1

Description : Binary-adds the contents of @HL and AC; the result is loaded to AC and the data memory @HL. The content of the index register (@HL) will be incremented automatically after executing this instruction.
* The result will not affect the carry flag (CF).
. @HL indicates an index address of data memory.

AND Rx

Function : AC ← [Rx] & AC

Description : Binary-ANDs the contents of Rx and AC; the result is loaded to AC.

AND @HL

Function : AC ← [@HL] & AC

Description : Binary-ANDs the contents of @HL and AC; the result is loaded to AC.

. @HL indicates an index address of data memory.

AND# @HL

Function : AC ← [@HL] & AC, @HL ← HL + 1

Description : Binary-ANDs the contents of @HL and AC; the result is loaded to AC. The content of the index register (@HL) will be incremented automatically after executing this instruction.
. @HL indicates an index address of data memory.

AND* Rx

Function : AC, [Rx] ← [Rx] & AC

Description : Binary-ANDs the contents of Rx and AC; the result is loaded to AC and the data memory Rx.

AND* @HL

Function : AC, [@HL] ← [@HL] & AC

Description : Binary-ANDs the contents of @HL and AC; the result is loaded to AC and the data memory @HL.
. @HL indicates an index address of data memory.

AND*# @HL

Function : AC, [@HL] ← [@HL] & AC, @HL ← HL + 1

Description : Binary-ANDs the contents of @HL and AC; the result is loaded to AC and the data memory @HL. The content of the index register (@HL) will be incremented automatically after executing this instruction.
. @HL indicates an index address of data memory.

EOR Rx

Function : AC ← [Rx] ⊕ AC

Description : Exclusive-Ors the contents of Rx and AC; the result is loaded to AC.

EOR @HL

Function : AC ← [@HL] ⊕ AC

Description : Exclusive-Ors the contents of @HL and AC; the result is loaded to AC.
. @HL indicates an index address of data memory.

EOR# @HL

Function : AC ← [@HL] ⊕ AC, @HL ← HL + 1

Description : Exclusive-Ors the contents of @HL and AC; the result is loaded to AC. The content of the index register (@HL) will be incremented automatically after executing this instruction.
. @HL indicates an index address of data memory.

EOR* Rx	
Function :	AC, Rx $\leftarrow$ [Rx] $\oplus$ AC
Description :	Exclusive-Ors the contents of Rx and AC; the result is loaded to AC and the data memory Rx.

EOR* @HL	
Function :	AC, [@HL] $\leftarrow$ [@HL] $\oplus$ AC
Description :	Exclusive-Ors the contents of @HL and AC; the result is loaded to AC and the data memory @HL. . @HL indicates an index address of data memory.

EOR*# @HL	
Function :	AC, [@HL] $\leftarrow$ [@HL] $\oplus$ AC, @HL $\leftarrow$ HL + 1
Description :	Exclusive-Ors the contents of @HL and AC; the result is loaded to AC and the data memory @HL. The content of the index register (@HL) will be incremented automatically after executing this instruction. . @HL indicates an index address of data memory.

OR Rx	
Function :	AC $\leftarrow$ [Rx] | AC
Description :	Binary-Ors the contents of Rx and AC; the result is loaded to AC.

OR @HL	
Function :	AC $\leftarrow$ [@HL] | AC
Description :	Binary-Ors the contents of @HL and AC; the result is loaded to AC. . @HL indicates an index address of data memory.

OR# @HL	
Function :	AC $\leftarrow$ [@HL] | AC, @HL $\leftarrow$ HL + 1
Description :	Binary-Ors the contents of @HL and AC; the result is loaded to AC. The content of the index register (@HL) will be incremented automatically after executing this instruction. . @HL indicates an index address of data memory.

OR* Rx	
Function :	AC, Rx $\leftarrow$ [Rx] | AC
Description :	Binary-Ors the contents of Rx and AC; the result is loaded to AC and the data memory Rx.

OR* @HL	
Function :	AC,[@HL] $\leftarrow$ [@HL] | AC
Description :	Binary-Ors the contents of @HL and AC; the result is loaded to AC and the data memory @HL.

. @HL indicates an index address of data memory.

OR*# @HL

Function : AC,[@HL] ← [@HL] | AC, @HL ← HL + 1

Description : Binary-Ors the contents of @HL and AC; the result is loaded to AC and the data memory @HL. The content of the index register (@HL) will be incremented automatically after executing this instruction.
. @HL indicates an index address of data memory.

ADCI Ry, D

Function : AC ← [Ry]+D+CF

Description : . D represents the immediate data.
Binary-ADDs the contents of Ry, D and CF; the result is loaded to AC.
* The carry flag (CF) will be affected.
D = 0H ~ FH

ADCI* Ry, D

Function : AC,[Ry] ← [Ry]+D+CF

Description : . D represents the immediate data.
Binary-ADDs the contents of Ry, D and CF; the result is loaded to AC and the working register Ry.
* The carry flag (CF) will be affected.
D = 0H ~ FH

SBCI Ry, D

Function : AC ← [Ry]+#(D)+CF

Description : . D represents the immediate data.
Binary-subtracts the CF and immediate data D from the working register Ry; the result is loaded to AC.
* The carry flag (CF) will be affected.
D = 0H ~ FH

SBCI* Ry, D

Function : AC,[Ry] ← [Ry]+#(D)+CF

Description : . D represents the immediate data.
Binary-subtracts the CF and immediate data D from the working register Ry; the result is loaded to AC and the working register Ry.
* The carry flag (CF) will be affected.
D = 0H ~ FH

ADDI Ry, D

Function : AC ← [Ry]+D

Description : . D represents the immediate data.
Binary-ADDs the contents of Ry and D; the result is loaded to AC.
* The carry flag (CF) will be affected.

D = 0H ~ FH

ADDI* Ry, D

Function : AC,[Ry] ← [Ry]+D

Description : . D represents the immediate data.
Binary-ADDs the contents of Ry and D; the result is loaded to AC and the working register Ry.
* The carry flag (CF) will be affected.
D = 0H ~ FH

SUBI Ry, D

Function : AC ← [Ry]+#(D)+1

Description : . D represents the immediate data.
Binary-subtracts the immediate data D from the working register Ry; the result is loaded to AC.
* The carry flag (CF) will be affected.
D = 0H ~ FH

SUBI* Ry, D

Function : AC,[Ry] ← [Ry]+#(Y)+1

Description : . D represents the immediate data.
Binary-subtracts the immediate data D from the working register Ry; the result is loaded to AC and the working register Ry.
* The carry flag (CF) will be affected.
D = 0H ~ FH

ADNI Ry, D

Function : AC ← [Ry]+D

Description : . D represents the immediate data.
Binary-ADDs the contents of Ry and D; the result is loaded to AC.
* The result will not affect the carry flag (CF).
D = 0H ~ FH

ADNI* Ry, D

Function : AC, [Ry] ← [Ry]+D

Description : . D represents the immediate data.
Binary-ADDs the contents of Ry and D; the result is loaded to AC and the working register Ry.
* The result will not affect the carry flag (CF).
D = 0H ~ FH

ANDI Ry, D

Function : AC ← [Ry] & D

Description : . D represents the immediate data.
Binary-ANDs the contents of Ry and D; the result is loaded to AC.

	D = 0H ~ FH

ANDI* Ry, D

Function :	AC,[Ry] ← [Ry] & D
Description :	. D represents the immediate data. Binary-ANDs the contents of Ry and D; the result is loaded to AC and the working register Ry. D = 0H ~ FH

EORI Ry, D

Function :	AC ← [Ry] EOR D
Description :	. D represents the immediate data. Exlusive-Ors the contents of Ry and D; the result is loaded to AC. D = 0H ~ FH

EORI* Ry, D

Function :	AC,[Ry] ← [Ry] ⊕ D
Description :	. D represents the immediate data. Exclusive-Ors the contents of Ry and D; the result is loaded to AC and the working register Ry. D = 0H ~ FH

ORI Ry, D

Function :	AC ← [Ry] | D
Description :	. D represents the immediate data. Binary-Ors the contents of Ry and D; the result is loaded to AC. D = 0H ~ FH

ORI* Ry, D

Function :	AC,[Ry] ← [Ry] | D
Description :	. D represents the immediate data. Binary-Ors the contents of Ry and D; the result is loaded to AC and the working register Ry. D = 0H ~ FH

5-4 LOAD/STORE INSTRUCTIONS

STA Rx

function:	Rx ← (AC)
description:	The content of AC is loaded to data memory specified by Rx.

STA @HL

function:	R@HL ← (AC)

description:	The content of AC is loaded to data memory specified by @HL.

STA# @HL

Function :	[@HL] ← AC, @HL ← HL + 1
Description :	The content of AC is loaded to the data memory specified by @HL. The content of the index register (@HL) will be incremented automatically after executing this instruction. . @HL indicates an index address of data memory.

LDS Rx, D

function:	AC,Rx ← D
description:	Immediate data D is loaded to the AC and data memory specified by Rx. D = 0H ~ FH

LDA Rx

function:	AC ← (Rx)
description:	The content of Rx is loaded to AC.

LDA @HL

function:	AC ← (R@HL)
description:	The content of data memory specified by @HL is loaded to AC.

LDA# @HL

Function :	AC ← [@HL] , @HL ← HL + 1
Description :	The content specified by @HL is loaded to AC. The content of the index register (@HL) will be incremented automatically after executing this instruction. . @HL indicates an index address of data memory.

LDH Rx, @HL

function:	Rx , AC ← H(T@HL)
description:	The higher nibble data of Table ROM specified by @HL is loaded to data memory specified by Rx.

LDH* Rx, @HL

function:	Rx , AC ← H(T@HL), @HL←(@HL)+1
description:	The higher nibble data of Table ROM specified by @HL is loaded to data memory specified by Rx and then is increased in @HL.

LDL Rx, @HL

function:	Rx , AC ← L(T@HL)
description:	The lower nibble data of Table ROM specified by @HL is loaded to the data memory specified by Rx.

LDL* Rx, @HL

function: Rx, AC ← L(T@HL), @HL ← (@HL)+1

description: The lower nibble data of Table ROM specified by @HL is loaded to the data memory specified by Rx and then incremented the content of @HL.

MRF1 Rx

function: Rx , AC ← RFC[3 ~ 0]

description: Loads the lowest nibble data of 16-bit counter of RFC to AC and data memory specified by Rx.
Bit 3 ← RFC[3]
Bit 2 ← RFC[2]
Bit 1 ← RFC[1]
Bit 0 ← RFC[0]

MRF2 Rx

function: Rx , AC ← RFC[7 ~ 4]

description: Loads the 2nd nibble data of 16-bit counter of RFC to AC and data memory specified by Rx.
Bit 3 ← RFC[7]
Bit 2 ← RFC[6]
Bit 1 ← RFC[5]
Bit 0 ← RFC[4]

MRF3 Rx

function: Rx , AC ← RFC[11 ~ 8]

description: Loads the 3rd nibble data of 16-bit counter of RFC to AC and data memory specified by Rx.
Bit 3 ← RFC[11]
Bit 2 ← RFC[10]
Bit 1 ← RFC[9]
Bit 0 ← RFC[8]

MRF4 Rx

function: Rx , AC ← RFC[15 ~ 12]

description: Loads the highest nibble data of 16-bit counter of RFC to AC and data memory specified by Rx.
Bit 3 ← RFC[15]
Bit 2 ← RFC[14]
Bit 1 ← RFC[13]
Bit 0 ← RFC[12]

5-5 CPU CONTROL INSTRUCTIONS

NOP

function:	no operation
description:	no operation

HALT

function: Enters halt mode

description: The following 3 conditions cause the halt mode to be released.
1) An interrupt is accepted.
2) The signal change specified by the SCA instruction is applied to IOC.
3) The halt release condition specified by SHE instruction is met.
When an interrupt is accepted to release the halt mode, the halt mode returns by executing the RTS instruction after completion of interrupt service.

STOP

function: Enters stop mode and stops all oscillators

description: Before executing this instruction, all signals on IOC port must be set to low.
The following 3 conditions cause the stop mode to be released.
1) One of the signal on KI1~4 is "H"/"L"(LED/LCD) in scanning interval.
2) A signal change in the INT pin.
3) One of the signals on IOC port is "H".

SCA X

function: The data specified by X causes the halt mode to be released.

description: The signal change at port IOA,IOC is specified. The bit meaning of X(X4) is shown below:

Bit pattern	Description
X4=1	Halt mode is released when signal applied to IOC

X7~5,X3~0 is reserved

SIE* X
function: Set/Reset interrupt enable flag
description:

X0=1	The IEF0 is set so that interrupt 0(Signal change at port IOC specified by SCA) is accepted.
X1=1	The IEF1 is set so that interrupt 1 (underflow from timer 1) is accepted.
X2=1	The IEF2 is set so that interrupt 2(the signal change at the INT pin) is accepted.
X3=1	The IEF3 is set so that interrupt 3(overflow from the predivider) is accepted.
X4=1	The IEF4 is set so that interrupt 4(underflow from timer 2) is accepted.
X6=1	The IEF6 is set so that interrupt 6(overflow from the RFC counter) is accepted.

X7 is reserved

SHE X
function: Set/Reset halt release enable flag
description:

X1=1	The HEF1 is set so that the halt mode is released by TMR1 underflow.
X2=1	The HEF2 is set so that the halt mode is released by signal changed on INT pin.
X3=1	The HEF3 is set so that the halt mode is released by predivider overflow.
X4=1	The HEF4 is set so that the halt mode is released by TMR2 underflow.
X6=1	The HEF6 is set so that the halt mode is released by RFC counter overflow.

X7 is reserved

SRE X
function: Set/Reset stop release enable flag
description:

X4=1	The SRF4 is set so that the stop mode is released by the signal changed on IOC port.
X5=1	The SRF5 is set so that the stop mode is released by the signal changed on INT pin.

X6,X3~0 is reserved

FAST
function: Switches the system clock to CFOSC clock.
description: Starts up the CFOSC(high speed osc.) and then switches the system clock to high speed clock.

SLOW
function: Switches the system clock to XTOSC clock(low speed osc).
description: Switches the system clock to low speed clock, and then stops the CFOSC.

MSB Rx

function: AC, Rx ← SCF3,SCF1,SCF2,BCF

description: The SCF1, SCF2 and BCF flag contents are loaded to AC and the data memory specified by Rx.
The content of AC and meaning of bit after execution of this instruction are as follows:

Bit 3	Bit 2	Bit 1	Bit 0
Start condition flag 3 (SCF 3)	Start condition flag 2 (SCF2)	Start condition flag 1 (SCF1)	Backup flag (BCF)
Halt release caused by the IOD port	Halt release caused by SCF4,5,6,7,8,9	Halt release caused by the IOC port	The Backup mode status in TM87P08

MSC Rx

function: AC, Rx ← SCF4..7

description: The SCF4 to SCF7 contents are loaded to AC and the data memory specified by Rx.
The content of AC and meaning of bit after execution of this instruction are as follows:

Bit 3	Bit 2	Bit 1	Bit 0
Start condition flag 7 (SCF7)	The content of 15th stage of the predivider	Start condition flag 5 (SCF5)	Start condition flag 4 (SCF4)
Halt release caused by predivider overflow		Halt release caused by TM1 underflow	Halt release caused by INT pin

MCX Rx

function: AC, Rx ← SCF8,SCF6,SCF9

description: The SCF8,SCF6,SCF9 contents are loaded to AC and the data memory specified by Rx.

The content of AC and meaning of bit after execution of this instruction are as follows:

Bit 3	Bit 2	Bit 1	Bit 0
Start condition flag 9 (SCF9)	NA	Start condition flag 6 (SCF6)	Start condition flag 8 (SCF8)
Halt release caused by RFC counter overflow	NA	Halt release caused by TM2 underflow	Halt release caused by the signal change to "L" applied on KI1~4 in scanning interval

MSD Rx

function: Rx, AC ← WDF,CSF,RFOVF

description: The watchdog flag, system clock status, overflow flag of RFC counter and low battery detected flag are loaded to data memory specified by Rx and AC.
The content of AC and meaning of bit after execution of this instruction are as follows:

Bit 3	Bit 2	Bit 1	Bit 0
NA	The overflow flag of 16-bit counter of RFC (RFVOF)	Watchdog timer enable flag (WDF)	System clock selection flag (CSF)

5-6 INDEX ADDRESS INSTRUCTIONS

MVU Rx

Function :	[@U] ← (Rx)
Description :	Loads content of Rx to the index address buffer @U. U3=[Rx]3, U2=[Rx]2, U1=[Rx]1, U0=[Rx]0

MVH Rx

function:	(@H) ← (Rx)
description:	Loads content of Rx to higher nibble of index address buffer @H. H3=[Rx]3, H2=[Rx]2, H1=[Rx]1, H0=[Rx]0,

MVL Rx

function:	(@L) ← (Rx)
description:	Loads content of Rx to lower nibble of index address buffer @L. L3=[Rx]3, L2=[Rx]2, L1=[Rx]1, L0=[Rx]0

CPHL X

Function :	If @HL = X, force the next instruction as NOP.
Description :	Compare the content of the index register @HL in lower 8 bits (@h and @L) with the immediate data X.

Note : In the duration of the comparison of the index address, all the interrupt enable flags(IEF) have to be cleared to avoid malfunction.If the compared result is equal, the next executed instruction that is behind the CPHL instruction will be forced as NOP.If the compared result is not equal, the next executed instruction that is behind CPHL instruction will operate normally.

The comparison bit pattern is shown below :

CPHL X	X7	X6	X5	X4	X3	X2	X1	X0
@HL	IDBF7	IDBF6	IDBF5	IDBF4	IDBF3	IDBF2	IDBF1	IDBF0

5-7 DECIMAL ARITHMETIC INSTRUCTIONS

DAA

function:	AC ← BCD(AC)
description:	Converts the content of AC to binary format, and then restores to AC. When this instruction is executed, the AC must be the result of any added instruction. * The carry flag (CF) will be affected.

DAA* Rx

function:	AC, Rx ← BCD(AC)
description:	Converts the content of AC to binary format, and then restores to AC and data memory specified by Rx.

When this instruction is executed, the AC must be the result of any added instruction.
* The carry flag (CF) will be affected.

DAA* @HL

function: AC,R@HL ← BCD(AC)

description: Converts the content of AC to decimal format, and then restores to AC and data memory specified by @HL.
When this instruction is executed, the AC must be the result of any added instruction.
* The carry flag (CF) will be affected.

AC data before DAA execution	CF data before DAA execution	AC data after DAA execution	CF data after DAA execution
0 ≤ AC ≤ 9	CF = 0	no change	no change
A ≤ AC ≤ F	CF = 0	AC= AC+ 6	CF = 1
0 ≤ AC ≤ 3	CF = 1	AC= AC+ 6	no change

DAA*# @HL

Function : AC,[@HL] ← BCD[AC], @HL = @HL + 1

Description : Converts the content of AC to binary format, and then restores to AC and the data memory specified by @HL. The content of the index register (@HL) will be incremented automatically after executing this instruction. When this instruction is executed, the AC must be the result of any added instruction.
* The carry flag (CF) will be affected.

AC data before DAA execution	CF data before DAA execution	AC data after DAA execution	CF data after DAA execution
0 ≤ AC ≤ 9	CF = 0	no change	no change
A ≤ AC ≤ F	CF = 0	AC= AC+ 6	CF = 1
0 ≤ AC ≤ 3	CF = 1	AC= AC+ 6	no change

DAS

Function : AC ← BCD[AC]

Description : Converts the content of AC to binary format, and then restores to AC. When this instruction is executed, the AC must be the result of any subtracted instruction.
* The carry flag (CF) will be affected.

DAS* Rx

function: AC, Rx ← BCD(AC)

description: Converts the content of AC to decimal format, and then restores to AC and data memory specified by Rx. When this instruction is executed, the AC must be the result of any subtracted instruction.
* The carry flag (CF) will be affected.

DAS* @HL

Function : AC, @HL ← BCD[AC]

Description : Converts the content of AC to binary format, and then restores to AC and the data memory @HL. When this instruction is executed, the AC must be the result of any subtracted instruction.
* The carry flag (CF) will be affected.

DAS*# @HL

Function : AC, @HL ← BCD[AC], @HL = @HL + 1

Description : Converts the content of AC to binary format, and then restores to AC and the data memory @HL. The content of the index register (@HL) will be incremented automatically after executing this instruction. When this instruction is executed, the AC must be the result of any subtracted instruction.
* The carry flag (CF) will be affected.

AC data before DAS execution	CF data before DAS execution	AC data after DAS execution	CF data after DAS execution
0 ≤ AC ≤ 9	CF = 1	No change	no change
6 ≤ AC ≤ F	CF = 0	AC= AC+A	no change

5-8 JUMP INSTRUCTIONS

JB0 X

function: Program counter jumps to X if AC0=1.

description: If bit0 of AC is 1 , jump occurs.
If 0, the PC increases by 1.
The range of X is from 000H to 7FFH or 800H to FFFH.

JB1 X

function: Program counter jumps to X if AC1=1.

description: If bit1 of AC is 1 , jump occurs.
If 0, the PC increases by 1.
The range of X is from 000H to 7FFH or 800H to FFFH.

JB2 X

function: Program counter jumps to X if AC2=1.

description: If bit2 of AC is 1 , jump occurs.
If 0 , the PC increases by 1.
The range of X is from 000H to 7FFH or 800H to FFFH.

JB3 X

function: Program counter jumps to X if AC3=1.

description: If bit3 of AC is 1 , jump occurs.
If 0, the PC increases by 1.
The range of X is from 000H to 7FFH or 800H to FFFH.

JNZ X

function:	Program counter jumps to X if (AC) != 0.
description:	If the content of AC is not 0 , jump occurs.
	If 0, the PC increases by 1.
	The range of X is from 000H to 7FFH or 800H to FFFH.

JNC X

function:	Program counter jumps to X if CF=0.
description:	If the content of CF is 0 , jump occurs.
	If 1, the PC increases by 1.
	The range of X is from 000H to 7FFH or 800H to FFFH.

JZ X

function:	Program counter jumps to X if (AC)=0.
description:	If the content of AC is 0 , jump occurs.
	If 1, the PC increases by 1.
	The range of X is from 000H to 7FFH or 800H to FFFH.

JC X

function:	Program counter jumps to X if CF=1.
description:	If the content of CF is 1 , jump occurs.
	If 0, the PC increases by 1.
	The range of X is from 000H to 7FFH or 800H to FFFH.

JMP X

function:	Program counter jumps to X.
description:	Unconditional jump.
	The range of X is from 000H to FFFH.

CALL X

function:	STACK $\leftarrow$ (PC)+1
	Program counter jumps to X.
description:	A subroutine is called.
	The range of X is from 000H to FFFH.

RTS

function:	PC $\leftarrow$ (STACK)
description:	A return from a subroutine occurs.

5-9 MISCELLANEOUS INSTRUCTIONS

SCC X

function: Setting the clock source for IOC,IOD chattering prevention, PWM output and frequency generator.

description: The following table shows the meaning of each bit for this instruction:

Bit pattern	Clock source setting	Bit pattern	Clock source setting
X6=1	The clock source comes from the system clock(BCLK).	X6=0	The clock source comes from the ϕ0. Refer to section 3-3-4 for ϕ0.

Bit pattern	Clock source setting	Bit pattern	Clock source setting
(X4,X3) = 01 (X2,X1,X0)=001	Chattering prevention clock of IOD port = PH0	(X4,X3) = 10 (X2,X1,X0)=001	Chattering prevention clock of IOC port = PH0
(X4,X3) = 01 (X2,X1,X0)=010	Chattering prevention clock of IOD port = PH8	(X4,X3) = 10 (X2,X1,X0)=010	Chattering prevention clock of IOC port = PH8
(X4,X3) = 01 (X2,X1,X0)=100	Chattering prevention clock of IOD port = PH6	(X4,X3) = 10 (X2,X1,X0)=100	Chattering prevention clock of IOC port = PH6

FRQ D, Rx

function: Frequency generator ← D, (Rx), (AC)

description: Loads the content of AC and data memory specified by Rx and D to frequency generator to set the duty cycle and initial value. The following table shows the preset data and the duty cycle setting:

	The bit pattern of preset letter N							
Programming divider	Bit7	Bit6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
FRQ D, Rx	AC3	AC2	AC1	AC0	Rx3	Rx2	Rx1	Rx0

Preset Letter D		Duty Cycle
D1	D0	
0	0	1/4 duty
0	1	1/3 duty
1	0	1/2 duty
1	1	1/1 duty

FRQ D, @HL

function: Frequency generator ← D, (T@HL)

description: Loads the content of Table ROM specified by @HL and D to frequency generator to set the duty cycle and initial value. The following table shows the preset data and the duty cycle setting:

	The bit pattern of preset letter N							
Programming divider	Bit7	Bit6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
FRQ D,@HL	T7	T6	T5	T4	T3	T2	T1	T0

Note: T0 ~ T7 represents the data of table ROM.

Preset Letter D		Duty Cycle
D1	D0	
0	0	1/4 duty
0	1	1/3 duty
1	0	1/2 duty
1	1	1/1 duty

FRQX D, X

function:	Frequency generator ← D, X
description:	Loads the data X(X7 ~ X0) and D to frequency generator to set the duty cycle and initial value. The following table shows the preset data and the duty cycle setting:

	The bit pattern of preset letter N							
Programming divider	Bit7	Bit6	Bit 5	Bit 4	Bit 3	Bit 2	bit 1	bit 0
FRQX D,X	X7	X6	X5	X4	X3	X2	X1	X0

Note: X0 ~ X7 represents the data specified in operand X.

Preset Letter D		Duty Cycle
D1	D0	
0	0	1/4 duty
0	1	1/3 duty
1	0	1/2 duty
1	1	1/1 duty

1. FRQ D, Rx
   The content of Rx and AC as preset data N.
2. FRQ D, @HL
   The content of tables TOM specified by index address buffer as preset data N.
3. FRQX D, X
   The data of operand in the instruction assigned as preset data N.

TMS Rx

function:	Select timer 1 clock source and preset timer 1.
description:	The content of data memory specified by Rx and AC are loaded to timer 1 to start the timer.

The following table shows the bit pattern for this instruction:

	Select clock		Setting value					
TMS Rx	AC3	AC2	AC1	AC0	Rx3	Rx2	Rx1	Rx0

The clock source option for timer 1

AC3	AC2	Clock source
0	0	PH9
0	1	PH3

1	0	PH15
1	1	FREQ

TMS @HL

function: Select timer 1 clock source and preset timer 1.

description: The content of table ROM specified by @HL is loaded to timer 1 to start the timer.

The following table shows the bit pattern for this instruction:

	Select clock		Setting value					
TMS @HL	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0

The clock source option for timer 1

Bit7	Bit6	Clock source
0	0	PH9
0	1	PH3
1	0	PH15
1	1	FREQ

TMSX X

function: Selects timer 1 clock source and preset timer 1.

description: The data specified by X(X8 ~ X0) is loaded to timer 1 to start the timer.

The following table shows the bit pattern for this instruction:

OPCODE	Select clock			Initiate value of timer					
TMSX X	X8	X7	X6	X5	X4	X3	X2	X1	X0

The clock source setting for timer 1

X8	X7	X6	clock source
0	0	0	PH9
0	0	1	PH3
0	1	0	PH15
0	1	1	FREQ
1	0	0	PH5
1	0	1	PH7
1	1	0	PH11
1	1	1	PH13

TM2 Rx

function: Selects timer 2 clock source and preset timer 2.

description: The content of data memory specified by Rx and AC is loaded to timer 2 to start the timer.

The following table shows the bit pattern for this instruction:

OPCODE	Select clock		Initiate value of timer					
TM2 Rx	AC3	AC2	AC1	AC0	Rx3	Rx2	Rx1	Rx0

The clock source setting for timer 2

AC3	AC2	clock source
0	0	PH9
0	1	PH3
1	0	PH15
1	1	FREQ

TM2 @HL

function: Selects timer 2 clock source and preset timer 2.

description: The content of Table ROM specified by @HL is loaded to timer 2 to start the timer.

The following table shows the bit pattern for this instruction:

OPCODE	Select clock		Initiate value of timer					
TM2 @HL	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0

The clock source setting for timer 2

Bit7	Bit6	clock source
0	0	PH9
0	1	PH3
1	0	PH15
1	1	FREQ

TM2X X

function: Selects timer 2 clock source and preset timer 2.

description: The data specified by X(X8 ~ X0) is loaded to timer 2 to start the timer. The following table shows the bit pattern for this instruction:

OPCODE	Select clock			Initiate value of timer					
TM2X X	X8	X7	X6	X5	X4	X3	X2	X1	X0

The clock source setting for timer 2

X8	X7	X6	clock source
0	0	0	PH9
0	0	1	PH3
0	1	0	PH15
0	1	1	FREQ
1	0	0	PH5
1	0	1	PH7
1	1	0	PH11
1	1	1	PH13

SF X

function: Sets flag

description: Description of each flag

X0 : "1" The CF is set to 1.

X1 : "1" The chip enters backup mode and BCF is set to 1.

X4 : "1" The watchdog timer is initiated and active.

X7 : "1" Enables the re-load function of timer 1.

X6,5 is reserved

RF X

machine code: 1111 0100 $X_7$00X_4 00X_1X_0

function: Resets flag

description: Description of each flag

X0 : "1" The CF is reset to 0.

X1 : "1" The chip is out of backup mode and BCF is reset to 0.

X4 : "1" The watchdog timer is inactive.

X7 : "1" Disables the re-load function of timer 1.

X6,5,3 is reserved

SF2 X

function: Sets flag

description: Description of each flag

X4 : "1" Enable low battery detected function

X3 : "1" Enable INT powerful pull-low

X2 : "1" Disables the LCD segment output.

X1 : "1" Sets the DED flag. Refer to 2-12-3 for detail.

X0 : "1" Enables the re-load function of timer 2.

X7~6 is reserved

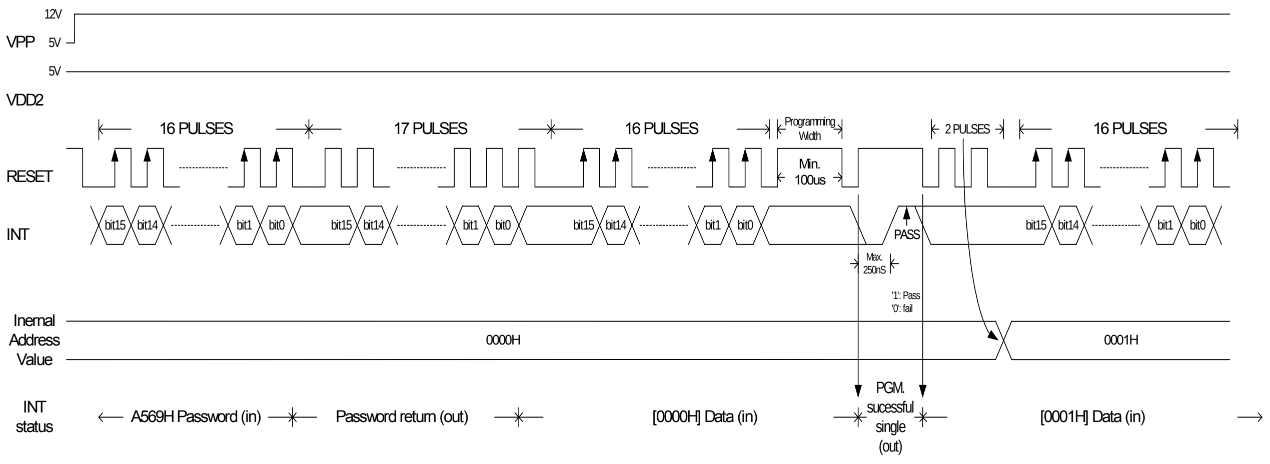
RF2 X

function:	Resets flag
description:	Description of each flag
	X4 : "1" Disable low battery detected function
	X3 : "1" Disable INT powerful pull-low
	X2 : "1" Enables the LCD segment output.
	X1 : "1" Resets the DED flag. Refer to 2-12-3 for detail.
	X0 : "1" Disables the re-load function of timer 2.
	X7~6 is reserved

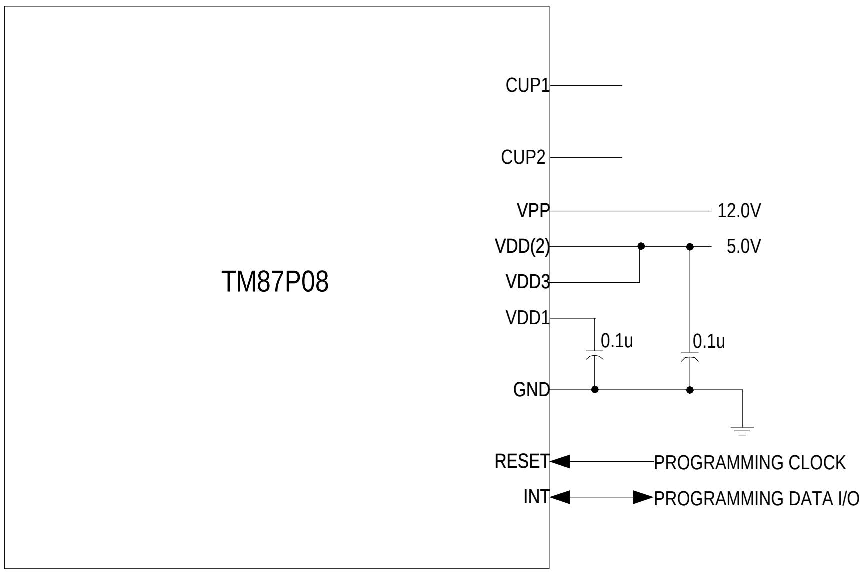
PLC

Function :	Pulse control
Description :	The pulse corresponding to the data specified by X is generated.
	X0 : "1" Halt release request flag HRF0 caused by the signal at I/O port C is reset.
	X1 : "1" Halt release request flag HRF1 caused by underflow from the timer 1 is reset, and stops the operating of timer 1(TM1).
	X2 : "1" Halt or stop release request flag HRF2 caused by the signal change at the INT pin is reset.
	X3 : "1" Halt release request flag HRF3 caused by overflow from the predivider is reset.
	X4 : "1" Halt release request flag HRF4 caused by underflow from the timer 2 is reset and stops the operating of timer 2(TM2).
	X5 : "1" Halt release request flag HRF5 caused by the signal change to "L" on KI1~4 in scanning interval is reset.
	X6 : "1" Halt release request flag HRF6 caused by overflow from the RFC counter is reset.
	X8 : "1" The last 5 bits of the predivider (15 bits) are reset. When executing this instruction, X3 must be set to "1" simultaneously.

Chapter 6 Programming Waveform



This programming application circuit is simply an example.



Appendix A TM87P08 Instruction Table

Instruction		Machine Code	Function		Flag/Remark
NOP		0000 0000 0000 0000	No Operation		
LCT	Lz,Ry	0000 001Z ZZZZ YYYY	Lz	← (7SEG ← Ry)	(Ry=70H~7FH)
LCB	Lz,Ry	0000 010Z ZZZZ YYYY	Lz	← (7SEG ← Ry) Blank Zero	(Ry=70H~7FH)
LCP	Lz,Ry	0000 011Z ZZZZ YYYY	Lz	← Ry & AC	(Ry=70H~7FH)
LCD	Lz,@HL	0000 100Z ZZZZ 0000	Lz	← T@HL	
LCT	Lz,@HL	0000 100Z ZZZZ 0001	Lz	← (7SEG ← @HL)	
LCB	Lz,@HL	0000 100Z ZZZZ 0010	Lz	← (7SEG ← @HL) Blank Zero	
LCP	Lz,@HL	0000 100Z ZZZZ 0011	Lz	← @HL & AC	
LCDX	D	0000 100D D000 0100	Multi-Lz D=00 D=01	← T@HL : Multi-Lz=00H~0FH : Multi-Lz=10H~1FH	D: 0~1
LCTX	D	0000 100D D000 0101	Multi-Lz	← (7SEG ← @HL)	D: 0~1
LCBX	D	0000 100D D000 0110	Multi-Lz	← (7SEG ← @HL) Blank Zero	D: 0~1
LCPX	D	0000 100D D000 0111	Multi-Lz	← @HL & AC	D: 0~1
OPA	Rx	0000 1010 0XXX XXXX	Port(A)	← Rx	
OPAS	Rx,D	0000 1011 DXXX XXXX	A1,2,3,4	← Rx0,Rx1,D,Pulse	
OPB	Rx	0000 1100 0XXX XXXX	Port(B)	← Rx	
OPC	Rx	0000 1101 0XXX XXXX	Port(C)	← Rx	
OPD	Rx	0000 1110 0XXX XXXX	Port(D)	← Rx	
FRQ	D,Rx	0001 00DD 0XXX XXXX	FREQ D=00 D=01 D=10 D=11	← Rx & AC : 1/4 Duty : 1/3 Duty : 1/2 Duty : 1/1 Duty	
FRQ	D,@HL	0001 01DD 0000 0000	FREQ	←T@HL	
FRQX	D,X	0001 10DD XXXX XXXX	FREQ	← X	
MVL	Rx	0001 1100 0XXX XXXX	IDBF0~3	← Rx	
MVH	Rx	0001 1101 0XXX XXXX	IDBF4~7	← Rx	
MVU	Rx	0001 1110 0XXX XXXX	IDBF8~11	← Rx	
ADC	Rx	0010 0000 0XXX XXXX	AC	← Rx + AC + CF	CF
ADC	@HL	0010 0000 1000 0000	AC	← @HL + AC + CF	CF
ADC#	@HL	0010 0000 1100 0000	AC HL	← @HL + AC + CF ←HL+1	CF
ADC*	Rx	0010 0001 0XXX XXXX	AC,Rx	← Rx + AC + CF	CF

ADC*	@HL	0010 0001 1000 0000	AC,@HL	← @HL + AC + CF	CF
ADC*#	@HL	0010 0001 1100 0000	AC,@HL HL	← @HL + AC + CF ←HL+1	CF
SBC	Rx	0010 0010 0XXX XXXX	AC	← Rx + ACB + CF	CF
SBC	@HL	0010 0010 1000 0000	AC	← @HL + ACB + CF	CF
SBC#	@HL	0010 0010 1100 0000	AC HL	← @HL + ACB + CF ←HL+1	CF
SBC*	Rx	0010 0011 0XXX XXXX	AC,Rx	← Rx + ACB + CF	CF
SBC*	@HL	0010 0011 1000 0000	AC,@HL	← @HL + ACB + CF	CF
SBC*#	@HL	0010 0011 1100 0000	AC,@HL HL	← @HL + ACB + CF ←HL+1	CF
ADD	Rx	0010 0100 0XXX XXXX	AC	← Rx + AC	CF
ADD	@HL	0010 0100 1000 0000	AC	← @HL + AC	CF
ADD#	@HL	0010 0100 1100 0000	AC HL	← @HL + AC ←HL+1	CF
ADD*	Rx	0010 0101 0XXX XXXX	AC,Rx	← Rx + AC	CF
ADD*	@HL	0010 0101 1000 0000	AC,@HL	← @HL + AC	CF
ADD*#	@HL	0010 0101 1100 0000	AC,@HL HL	← @HL + AC ←HL+1	CF
SUB	Rx	0010 0110 0XXX XXXX	AC	← Rx + ACB + 1	CF
SUB	@HL	0010 0110 1000 0000	AC	← @HL + ACB + 1	CF
SUB#	@HL	0010 0110 1100 0000	AC HL	← @HL + ACB + 1 ←HL+1	CF
SUB*	Rx	0010 0111 0XXX XXXX	AC,Rx	← Rx + ACB + 1	CF
SUB*	@HL	0010 0111 1000 0000	AC,@HL	← @HL + ACB + 1	CF
SUB*#	@HL	0010 0111 1100 0000	AC,@HL HL	← @HL + ACB + 1 ←HL+1	CF
ADN	Rx	0010 1000 0XXX XXXX	AC	← Rx + AC	
ADN	@HL	0010 1000 1000 0000	AC	← @HL + AC	
ADN#	@HL	0010 1000 1100 0000	AC HL	← @HL + AC ←HL+1	
ADN*	Rx	0010 1001 0XXX XXXX	AC,Rx	← Rx + AC	
ADN*	@HL	0010 1001 1000 0000	AC,@HL	← @HL + AC	
ADN*#	@HL	0010 1001 1100 0000	AC,@HL HL	← @HL + AC ←HL+1	
AND	Rx	0010 1010 0XXX XXXX	AC	← Rx AND AC	
AND	@HL	0010 1010 1000 0000	AC	← @HL AND AC	
AND#	@HL	0010 1010 1100 0000	AC HL	← @HL AND AC ←HL+1	

AND*	Rx	0010 1011 0XXX XXXX	AC,Rx	← Rx AND AC	
AND*	@HL	0010 1011 1000 0000	AC,@HL	← @HL AND AC	
AND*#	@HL	0010 1011 1100 0000	AC,@HL HL	← @HL AND AC ←HL+1	
EOR	Rx	0010 1100 0XXX XXXX	AC	← Rx EOR AC	
EOR	@HL	0010 1100 1000 0000	AC	← @HL EOR AC	
EOR#	@HL	0010 1100 1100 0000	AC HL	← @HL EOR AC ←HL+1	
EOR*	Rx	0010 1101 0XXX XXXX	AC,Rx	← Rx EOR AC	
EOR*	@HL	0010 1101 1000 0000	AC,@HL	← @HL EOR AC	
EOR*#	@HL	0010 1101 1100 0000	AC,@HL HL	← @HL EOR AC ←HL+1	
OR	Rx	0010 1110 0XXX XXXX	AC	← Rx OR AC	
OR	@HL	0010 1110 1000 0000	AC	← @HL OR AC	
OR#	@HL	0010 1110 1100 0000	AC HL	← @HL OR AC ←HL+1	
OR*	Rx	0010 1111 0XXX XXXX	AC,Rx	← Rx OR AC	
OR*	@HL	0010 1111 1000 0000	AC,@HL	← @HL OR AC	
OR*#	@HL	0010 1111 1100 0000	AC,@HL HL	← @HL OR AC ←HL+1	
ADCI	Ry,D	0011 0000 DDDD YYYY	AC	← Ry + D + CF	
ADCI*	Ry,D	0011 0001 DDDD YYYY	AC,Ry	← Ry + D + CF	
SBCI	Ry,D	0011 0010 DDDD YYYY	AC	← Ry + DB + CF	
SBCI*	Ry,D	0011 0011 DDDD YYYY	AC,Ry	← Ry + DB + CF	
ADDI	Ry,D	0011 0100 DDDD YYYY	AC	← Ry + D	
ADDI*	Ry,D	0011 0101 DDDD YYYY	AC,Ry	← Ry + D	
SUBI	Ry,D	0011 0110 DDDD YYYY	AC	← Ry + DB + 1	
SUBI*	Ry,D	0011 0111 DDDD YYYY	AC,Ry	← Ry + DB + 1	
ADNI	Ry,D	0011 1000 DDDD YYYY	AC	← Ry + D	
ADNI*	Ry,D	0011 1001 DDDD YYYY	AC,Ry	← Ry + D	
ANDI	Ry,D	0011 1010 DDDD YYYY	AC	← Ry AND D	
ANDI*	Ry,D	0011 1011 DDDD YYYY	AC,Ry	← Ry AND D	
EORI	Ry,D	0011 1100 DDDD YYYY	AC	← Ry EOR D	
EORI*	Ry,D	0011 1101 DDDD YYYY	AC,Ry	← Ry EOR D	
ORI	Ry,D	0011 1110 DDDD YYYY	AC	← Ry OR D	
ORI*	Ry,D	0011 1111 DDDD YYYY	AC,Ry	← Ry OR D	
INC*	Rx	0100 0000 0XXX XXXX	AC,Rx	← Rx + 1	CF
INC*	@HL	0100 0000 1000 0000	AC,@HL	← @HL + 1	CF
INC*#	@HL	0100 0000 1100 0000	AC,@HL	← @HL + 1	CF

			HL	←HL+1	
DEC*	Rx	0100 0001 0XXX XXXX	AC,Rx	← Rx – 1	CF
DEC*	@HL	0100 0001 1000 0000	AC,@HL	← @HL - 1	CF
DEC*#	@HL	0100 0001 1100 0000	AC,@HL HL	← @HL - 1 ←HL+1	CF
IPA	Rx	0100 0010 0XXX XXXX	AC,Rx	← Port(A)	
IPB	Rx	0100 0100 0XXX XXXX	AC,Rx	← Port(B)	
IPC	Rx	0100 0111 0XXX XXXX	AC,Rx	← Port(C)	
IPD	Rx	0100 1000 0XXX XXXX	AC,Rx	← Port(D)	
MAF	Rx	0100 1010 0XXX XXXX	AC,Rx	← STS1	B3 : CF B2 : ZERO B1 : (No use) B0 : (No use)
MSB	Rx	0100 1011 0XXX XXXX	AC,Rx	← STS2	B3 : SCF3(DPT) B2 : SCF2(HRx) B1 : SCF1(CPT) B0 : BCF
MSC	Rx	0100 1100 0XXX XXXX	AC,Rx	← STS3	B3 : SCF7(PDV) B2 : PH15 B1 : SCF5(TM1) B0 : SCF4(INT)
MCX	Rx	0100 1101 0XXX XXXX	AC,Rx	← STS3X	B3 : SCF9(RFC) B2 : (unused) B1 : SCF6(TM2) B0 : SCF8(SKI)
MSD	Rx	0100 1110 0XXX XXXX	AC,Rx	← STS4	B3 : (No use) B2 : FROVF B1 : WDF B0 : CSF
SR0	Rx	0101 0000 0XXX XXXX	ACn, Rxn AC3, Rx3	← Rx(n+1) ← 0	
SR1	Rx	0101 0001 0XXX XXXX	ACn, Rxn AC3, Rx3	← Rx(n+1) ← 1	
SL0	Rx	0101 0010 0XXX XXXX	ACn, Rxn AC0, Rx0	← Rx(n-1) ← 0	
SL1	Rx	0101 0011 0XXX XXXX	ACn, Rxn AC0, Rx0	← Rx(n-1) ← 1	
DAA		0101 0100 0000 0000	AC	← BCD(AC)	CF
DAA*	Rx	0101 0101 0XXX XXXX	AC,Rx	← BCD(AC)	CF
DAA*	@HL	0101 0101 1000 0000	AC,@HL	← BCD(AC)	CF
DAA*#	@HL	0101 0101 1100 0000	AC,@HL HL	← BCD(AC)	CF

				←HL+1	
DAS		0101 0110 0000 0000	AC	← BCD(AC)	CF
DAS*	Rx	0101 0111 0XXX XXXX	AC,Rx	← BCD(AC)	CF
DAS*	@HL	0101 0111 1000 0000	AC,@HL	← BCD(AC)	CF
DAS*#	@HL	0101 0111 1100 0000	AC,@HL HL	← BCD(AC) ←HL+1	CF
LDS	Rx,D	0101 1DDD DXXX XXXX	AC,Rx	← D	
LDH	Rx,@HL	0110 0000 0XXX XXXX	AC,Rx	← H(T@HL)	
LDH*	Rx,@HL	0110 0001 0XXX XXXX	AC,Rx HL	← H(T@HL) ← HL + 1	
LDL	Rx,@HL	0110 0010 0XXX XXXX	AC,Rx	← L(T@HL)	
LDL*	Rx,@HL	0110 0011 0XXX XXXX	AC,Rx HL	← L(T@HL) ← HL + 1	
MRF1	Rx	0110 0100 0XXX XXXX	AC,Rx	← RFC3-0	
MRF2	Rx	0110 0101 0XXX XXXX	AC,Rx	← RFC7-4	
MRF3	Rx	0110 0110 0XXX XXXX	AC,Rx	← RFC11-8	
MRF4	Rx	0110 0111 0XXX XXXX	AC,Rx	← RFC15-12	
STA	Rx	0110 1000 0XXX XXXX	Rx	← AC	
STA	@HL	0110 1000 1000 0000	@HL	← AC	
STA#	@HL	0110 1000 1100 0000	@HL HL	← AC ←HL+1	
LDA	Rx	0110 1100 0XXX XXXX	AC	← Rx	
LDA	@HL	0110 1100 1000 0000	AC	← @HL	
LDA#	@HL	0110 1100 1100 0000	AC HL	← @HL ←HL+1	
MRA	Rx	0110 1101 0XXX XXXX	CF	← Rx3	
MRW	@HL,Rx	0110 1110 0XXX XXXX	AC,@HL	← Rx	
MRW#	@HL,Rx	0110 1110 1XXX XXXX	AC,@HL HL	← Rx ←HL+1	
MWR	Rx,@HL	0110 1111 0XXX XXXX	AC,Rx	← @HL	
MWR#	Rx,@HL	0110 1111 1XXX XXXX	AC,Rx HL	← @HL ←HL+1	
MRW	Ry,Rx	0111 0YYY YXXX XXXX	AC,Ry	← Rx	
MWR	Rx,Ry	0111 1YYY YXXX XXXX	AC,Rx	← Ry	
JB0	X	1000 0XXX XXXX XXXX	PC	← X	if AC0 = 1
JB1	X	1000 1XXX XXXX XXXX	PC	← X	if AC1 = 1
JB2	X	1001 0XXX XXXX XXXX	PC	← X	if AC2 = 1
JB3	X	1001 1XXX XXXX XXXX	PC	← X	if AC3 = 1
JNZ	X	1010 0XXX XXXX XXXX	PC	← X	if AC ≠ 0

JNC	X	1010 1XXX XXXX XXXX	PC	← X	if CF = 0
JZ	X	1011 0XXX XXXX XXXX	PC	← X	if AC = 0
JC	X	1011 1XXX XXXX XXXX	PC	← X	if CF = 1
CALL	X	1100 PXXX XXXX XXXX	STACK PC	← PC + 1 ← X	
JMP	X	1101 PXXX XXXX XXXX	PC	← X	
TMS	Rx	1110 0000 0XXX XXXX	AC3,2 = 11 AC3,2 = 10 AC3,2 = 01 AC3,2 = 00 AC1,0,PB3 ~0	: Ctm = FREQ : Ctm = PH15 : Ctm = PH3 : Ctm = PH9 : Set Timer1 Value	
TMS	@HL	1110 0001 0000 0000	TD7,6 = 11 TD7,6 = 10 TD7,6 = 01 TD7,6 = 00 TD5~0	: Ctm = FREQ : Ctm = PH15 : Ctm = PH3 : Ctm = PH9 : Set Timer1 Value	
TMSX	X	1110 001X XXXX XXXX	X8,7,6=111 X8,7,6=110 X8,7,6=101 X8,7,6=100 X8,7,6=011 X8,7,6=010 X8,7,6=001 X8,7,6=000 X5~0	: Ctm = PH13 : Ctm = PH11 : Ctm = PH7 : Ctm = PH5 : Ctm = FREQ : Ctm = PH15 : Ctm = PH3 : Ctm = PH9 : Set Timer1 Value	
TM2	Rx	1110 0100 0XXX XXXX	Timer2	← Rx & AC	
TM2	@HL	1110 0101 0000 0000	Timer2	← T@HL	
TM2X	X	1110 011X XXXX XXXX	X8,7,6=111 X8,7,6=110 X8,7,6=101 X8,7,6=100 X8,7,6=011 X8,7,6=010 X8,7,6=001 X8,7,6=000 X5~0	: Ctm = PH13 : Ctm = PH11 : Ctm = PH7 : Ctm = PH5 : Ctm = FREQ : Ctm = PH15 : Ctm = PH3 : Ctm = PH9 : Set Timer2 Value	
SHE	X	1110 1000 0XXX XXX0	X6 X5 X4 X3 X2 X1	: Enable HEF6 : Enable HEF5 : Enable HEF4 : Enable HEF3 : Enable HEF2 : Enable HEF1	RFC KEY_S TMR2 PDV INT TMR1

SIE*	X	1110 1001 0XXX XXXX	X6 X5 X4 X3 X2 X1 X0	: Enable IEF6 : Enable IEF5 : Enable IEF4 : Enable IEF3 : Enable IEF2 : Enable IEF1 : Enable IEF0	RFC KEY_S TMR2 PDV INT TMR1 C, DPT
PLC	X	1110 101X 0XXX XXXX	X8 X6-0	: Reset PH15~11 : Reset HRF6-0	
SRF	X	1110 1100 00XX XXXX	X5 X4 X3 X2 X1 X0	: Enable Cx Control : Enable TM2 Control : Enable Counter : Enable RH Output : Enable RT Output : Enable RR Output	ENX EHM ETP ERR
SRE	X	1110 1101 X0XX X000	X7 X5 X4 X3	: Enable SRF7(key_s) : Enable SRF5(INT) : Enable SRF4(C port) : Enable SRF3(D port)	
FAST		1110 1110 0000 0000	SCLK	: High Speed Clock	
SLOW		1110 1110 1000 0000	SCLK	: Low Speed Clock	
CPHL	X	1110 1111 XXXX XXXX	(PC+1)	← force "NOP" if X7~0=IDBF7~0	
SPK	Rx	1111 0000 0XXX XXXX	KO1~16	← Rx & AC	
SPK	@HL	1111 0001 0000 0000	KO1~16	← T @HL	
SPKX	X	1111 0010 XXXX XXXX	X6=1 X6=0 X7,5,4=000 X7,5,4=001 X7,5,4=010 X7,5,4=10X X7,5,4=110	: KEY_S release by scanning cycle : KEY_S release by normal key scanning : Set one of KO1~16 =1 by X3~0 : Set all = 1 : Set all Hi-z : Set eight of KO1~16 =1 by X3 X3=0 => KO1~8 X3=1 => KO9~16 : Set four of KO1~16 =1 by X3,2 X3,2=00 => KO1~4 X3,2=01 => KO5~8 X3,2=10 => KO9~12 X3,2=11 => KO13~16	

			X7,5,4=111	: Set two of KO1~16 =1 by X3,2,1 X3~1=000=>KO1,2 X3~1=001=>KO3,4 X3~1=010=>KO5,6 X3~1=011=>KO7,8 X3~1=100=>KO9,10 X3~1=101=>KO11,12 X3~1=110=>KO13,14 X3~1=111=>KO15,16	
RTS		1111 0100 0000 0000	PC	← STACK (CALL Return)	
SCC	X	1111 0100 1X0X XXXX	X6 = 1 X6 = 0 X4 = 1 X3 = 1 X2,1,0=001 X2,1,0=010 X2,1,0=100	: Cfq = BCLK : Cfq = PH0 : Set P(C) Cch : Set P(D) Cch : Cch = PH10 : Cch = PH8 : Cch = PH6	
SCA	X	1111 0101 000X X000	X4 X3	: Enable SEF4(C1-4) : Enable SEF3(D1-4)	
SPA	X	1111 0101 100X XXXX	X4 X3~0	: Set A4-1 Pull-Low : Set A4-1 I/O	1:Pull low 1:Output, 0: Input
SPB	X	1111 0101 101X XXXX	X4 X3~0	: Set B4-1 Pull-Low : Set B4-1 I/O	1:Pull low 1:Output, 0: Input
SPC	X	1111 0101 110X XXXX	X4 X3-0	: Set C4-1 Pull-Low / Low-Level-Hold : Set C4-1 I/O	1:Pull low, 0:LLH 1:Output, 0: Input
SPD	X	1111 0101 111X XXXX	X4 X3-0	: Set D4-1 Pull-Low : Set D4-1 I/O	1:Pull low 1:Output, 0: Input
SF	X	1111 0110 X00X 00XX	X7 X4 X1 X0	: Reload 1 Set : WDT Enable : BCF Set : CF Set	
RF	X	1111 0111 X00X 00XX	X7 X4 X1 X0	:Reload 1 Reset : WDT Reset : BCF Reset : CF Reset	
ALM	X	1111 110X XXXX XXXX	X8,7,6=111 X8,7,6=100 X8,7,6=011 X8,7,6=010 X8,7,6=001 X8,7,6=000	: FREQ : DC1 : PH3 : PH4 : PH5 : DC0	

			X5~0	← PH15~10	
SF2	X	1111 1110 0000 XXXX	X3 X2 X1 X0	: Enable INT powerful Pull-low : Close all Segments : Dis-ENX Set : Reload 2 Set	
RF2	X	1111 1110 1000 XXXX	X3 X2 X1 X0	: Disable INT powerful Pull-low : Release Segments : Dis-ENX Reset : Reload 2 Reset	
HALT		1111 1111 0000 0000	Halt Operation		
STOP		1111 1111 1000 0000	Stop Operation		

Symbol Description

Symbol	Description	Symbol	Description
( )	Content of Register	D	Immediate Data
AC	Accumulator	(D)B	Complement of Immediate Data
(AC)n	Content of Accumulator (bit n)	PC	Program Counter
(AC)B	Complement of content of Accumulator	CF	Carry Flag
X	Address of program or control data	ZERO	Zero Flag
Rx	Address X of data RAM	WDF	Watch-Dog Timer Enable Flag
(Rx)n	Bit n content of Rx	7SEG	7 segment decoder for LCD
Ry	Address Y of working register	BCLK	System clock for instruction
R@HL	Address of data RAM specified by @HL	IEFn	Interrupt Enable Flag
BCF	Backup flag	HRFn	HALT Release Flag
@HL	Generic Index address register	HEFn	HALT Release Enable Flag
(@HL)	Content of generic Index address register	Lz	Address of LCD PLA Latch
(@L)	Content of lowest nibble Index register	SRFn	STOP Release Enable Flag
(@H)	Content of middle nibble Index register	SCFn	Start Condition Flag
(@U)	Content of highest nibble Index register	Cch	Clock Source of Chattering prevention ckt.
T@HL	Address of Table ROM	Cfq	Clock Source of Frequency Generator
H(T@HL)	High Nibble content of Table ROM	SEFn	Switch Enable Flag
L(T@HL)	Low Nibble content of Table ROM	FREQ	Frequency Generator setting Value
TMR	Timer Overflow Release Flag	CSF	Clock Source Flag
Ctm	Clock Source of Timer	P	Program Page
PDV	Pre-Divider	RFOVF	RFC Overflow Flag
STACK	Content of stack	RFC	Resistor to Frequency counter
TM1	Timer 1	(RFC)n	Bit data of Resistor to Frequency counter
TM2	Timer 2		