TM8530

4-Bit Microcontroller with LCD Driver

User Manual

AMENDMENT HISTORY

Version	Date	Description
V1.0	Sept, 2010	New release
V1.1	Dec, 2011	Add Ordering Information table

CONTENTS

AMENDMENT HISTORY 2

CHAPTER 1 GENERAL DESCRIPTION 5

1-1. GENERAL DESCRIPTION 5

1-2. FEATURES 5

1-3. BLOCK DIAGRAM 7

1-4. PAD DIAGRAM 8

CHAPTER 2 TM8530 INTERNAL SYSTEM ARCHITECTURE 13

2-1. Power Supply 13

2-2. SYSTEM CLOCK 14

2-3. PROGRAM COUNTER (PC) 22

2-4. PROGRAM/TABLE MEMORY (ROM) 23

2-5. INDEX ADDRESS REGISTER (@HL) 25

2-6. STACK REGISTER (STACK) 25

2-7. DATA MEMORY (RAM) 26

2-8. WORKING REGISTER (WR) 27

2-9. ACCUMULATOR (AC) 27

2-10. ALU (Arithmetic and Logic Unit) 27

2-11. BINARY CONVERTS TO DECIMAL (BCD) 27

2-12. TIMER 1 (TMR1) 29

2-13. TIMER 2 (TMR2) 32

2-14. STATUS REGISTER (STS) 36

2-15. CONTROL REGISTER (CTL) 41

2-16. HALT FUNCTION 44

2-17. BACKUP FUNCTION 44

2-18. STOP FUNCTION (STOP) 45

CHAPTER 3 CONTROL FUNCTION 47

3-1. INTERRUPT FUNCTION 47

3-2. RESET FUNCTION 51

3-3. CLOCK GENERATOR 53

3-4. BUZZER OUTPUT PINS 56

3-5. INPUT / OUTPUT PORTS 58
3-6. EXTERNAL INT PIN 67
3-7. RESISTER TO FREQUENCY CONVERTER (RFC) 68
CHAPTER 4 LCD DRIVER OUTPUT 73
4-1. LCD LIGHTING SYSTEM IN TM8530 73
4-2. SEGMENT PLA CIRCUIT FOR LCD DISPLAY 74
CHAPTER 5 DETAIL EXPLANATION OF TM8530 INSTRUCTIONS 80
5-1. INPUT / OUTPUT INSTRUCTIONS 80
5-2. ACCUMULATOR MANIPULATION INSTRUCTIONS AND MEMORY MANIPULATION INSTRUCTIONS 85
5-3. OPERATION INSTRUCTIONS 86
5-4. LOAD/STORE INSTRUCTIONS 97
5-5. CPU CONTROL INSTRUCTIONS 99
5-6. INDEX ADDRESS INSTRUCTIONS 101
5-7. DECIMAL ARITHMETIC INSTRUCTIONS 102
5-8. JUMP INSTRUCTIONS 104
5-9. MISCELLANEOUS INSTRUCTIONS 106
ORDERING INFORMATION 111
Appendix A TM8530 INSTRUCTION TABLE 112
Appendix B TYPICAL APPLICATION CIRCUIT 121
SYMBOL DESCRIPTION 122

CHAPTER 1 GENERAL DESCRIPTION

1-1. GENERAL DESCRIPTION

The TM8530 is an embedded high-performance 4-bit microcontroller with LCD driver

1-2. FEATURES

1. Powerful instruction set (174 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
- LCD driver with data transfer

2. ROM capacity 3K x 16 bits

- Program ROM Max. capacity 3K x 16 bits
- Table ROM Max. capacity 2K x 8 bits

3. RAM capacity

- Data RAM 128 x 4 bits

4. Input/output ports

- Port IOA 4 pins (with internal pull-low)
- Port IOB 2 pins (with internal pull-low)
- Port IOC 4 pins (with internal pull-low, chattering prevention clock)
- Port IOD 4 pins (with internal pull-low, chattering prevention clock)

5. 8-level subroutine nesting

6. Interrupt function

- External factor 3 (INT pin, Port IOC, IOD)
- Internal factor 4 (Pre-Divider, Timer1, Timer2, RFC)

7. Built-in Alarm, Frequency or Melody generator

8. BZB, BZ [Mux*(Multiplex System)* with IOB3, IOB4]

9. Two 6-bit programmable timers with programmable clock source

10. LCD driver output

- 36 LCD driver outputs (up to 144 LCD segment drivable)
- 1/4 Duty for LCD
- 1/2 Bias or 1/3 Bias for LCD selected in mask option
- Single instruction to turn off all segments

- 18 LCD address

11. Built-in LCD Voltage divider Resistor.

12. Dual clock operation, X'tal type slow oscillation, and fast oscillation can be set as internal RC (500 KHz) or external R in switch mask option.

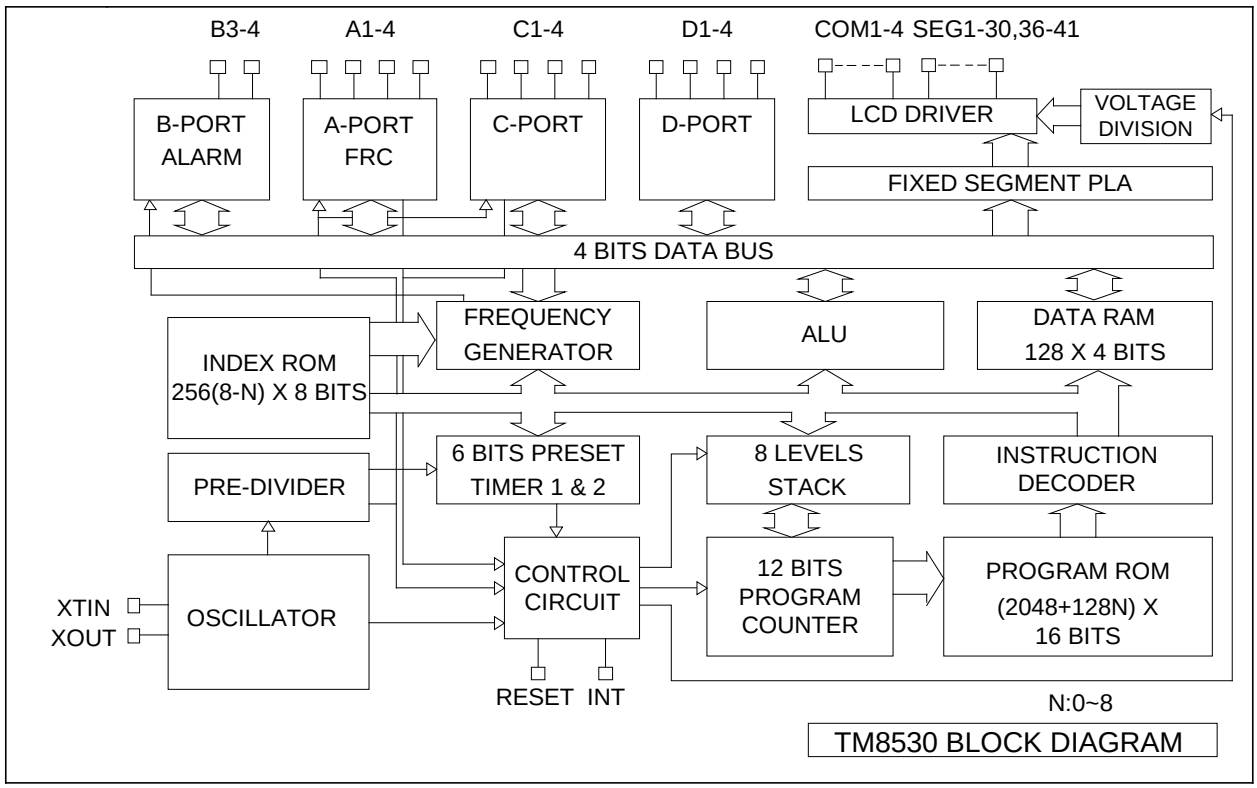
13. Watch dog timer

14. HALT function

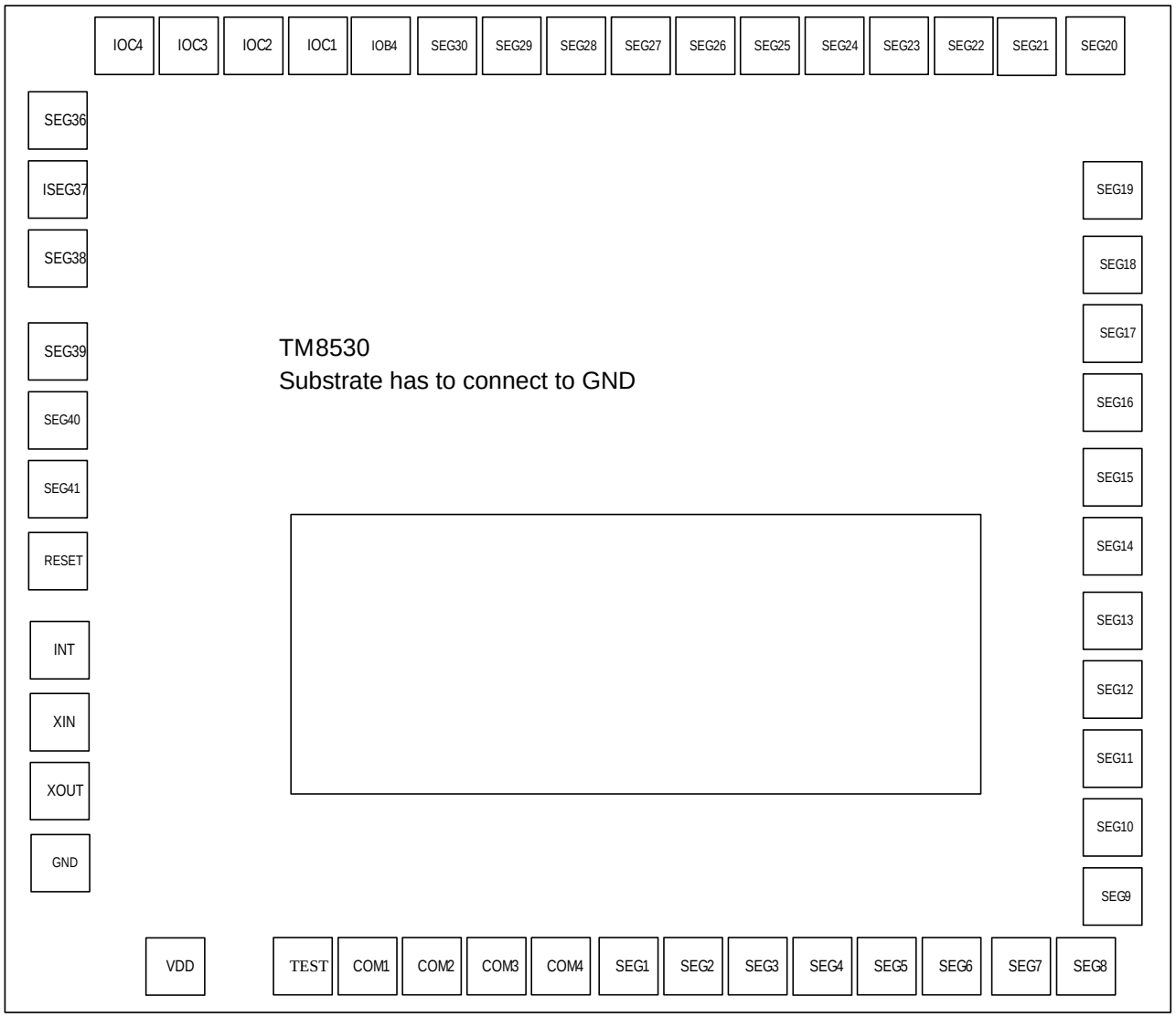
15. STOP function

16. Fixed LCD PLA configuration

1-3. BLOCK DIAGRAM



1-4. PAD DIAGRAM



1-5. PAD COORDINATE

No	Name	No	Name
1	SEG36/IOD1	30	SEG10
2	SEG37/IOD2	31	SEG11
3	SEG38/IOD3	32	SEG12
4	SEG39/IOD4	33	SEG13
5	SEG40	34	SEG14
6	SEG41	35	SEG15
7	RESET	36	SEG16
8	INT	37	SEG17
9	XIN	38	SEG18
10	XOUT	39	SEG19
11	GND	40	SEG20
12	VDD	41	SEG21
13	TEST	42	SEG22
14	COM1	43	SEG23
15	COM2	44	SEG24/IOA1/CX
16	COM3	45	SEG25/IOA2/RR
17	COM4	46	SEG26/IOA3/RT
18	SEG1	47	SEG27/IOA4/RH
19	SEG2	48	SEG28
20	SEG3	49	SEG29
21	SEG4	50	SEG30/IOB3/BZB
22	SEG5	51	IOB4/BZ
23	SEG6	52	IOC1
24	SEG7	53	IOC2
25	SEG8	54	IOC3
26	SEG9	55	IOC4

1-6. PIN DESCRIPTION

Name	I/O	Description
VDD	P	LCD supply voltage, and positive supply voltage. Connect +3.0V battery positive pin to VDD.
RESET	I	Input pin from LSI reset request signal, with internal pull-down resistor.
INT	I I/O	Input pin for external INT request signal. It can be triggered by falling edge or rising edge and is defined in mask option. Internal pull-down or pull-up resistor is selected in mask option. Serial Data for Serial Program/Read Mode.
TEST	I	Test signal input pin.
XIN XOUT	I O	32 KHz Crystal oscillator for Slow Clock. External R oscillation for Fast Clock.
COM1~4	O	Output pins for driving the common pins of the LCD panel.
SEG1~30, 36~41	O	Output pins for driving the LCD panel segment.
IOA1-4	I/O	Input / Output port-A, can use software to define internal pull-low Resistor. This port is multiplexed with SEG24~27, and set in mask option.
IOB3-4	I/O	Input / Output port-B, can use software to define internal pull-low Resistor. This port is multiplexed with BZB, BZ, and set in mask option.
IOC1-4	I/O	Input / Output port-C, can use software to define internal pull-low and chattering clock to reduce input bounce.
IOD1-4	I/O	Input / Output port-D, can use software to define internal pull-low Resistor, and Chattering clock to reduce input bounce. This port is multiplexed with SEG36~39, and set in mask option.
(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 in mask option.
(ALM) BZB/BZ	O	Output port for alarm, frequency or melody generator. This port is multiplexed with IOB3, 4, and set in mask option.
GND	P	Negative supply voltage.

1-7. CHARACTERISTICS

ABSOLUTE MAXIMUM RATINGS (GND= 0V)

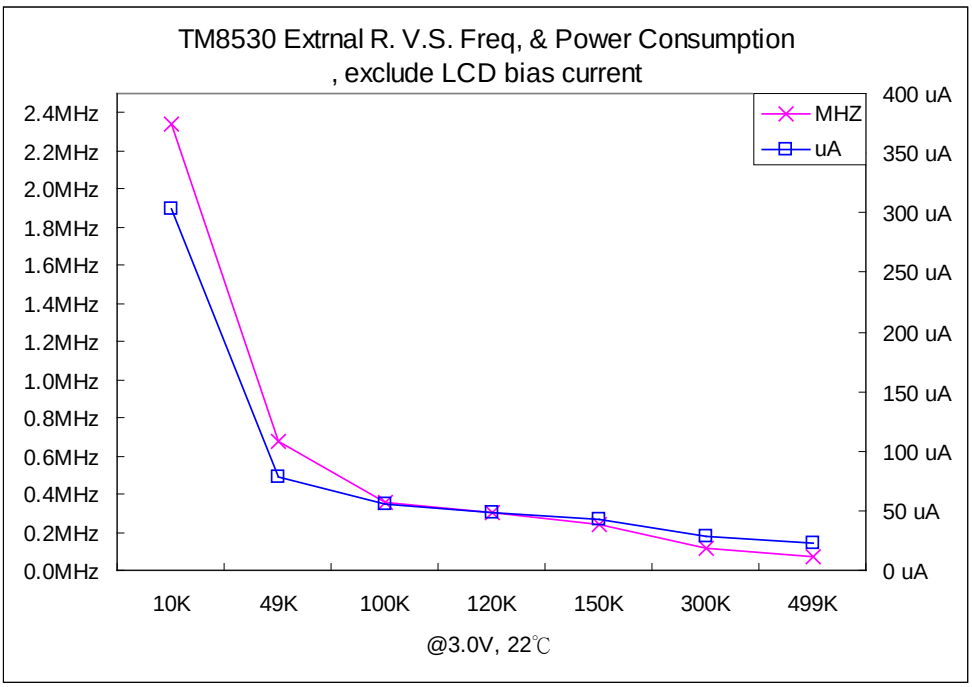
Name	Symbol	Range	Unit
Maximum Supply Voltage	VDD	-0.3 to 3.6	V
Maximum Input Voltage	Vin	-0.3 to VDD+0.3	
Maximum output Voltage	Vout	-0.3 to VDD+0.3	
Maximum Operating Temperature	Topg	-40 to +80	℃
Maximum Storage Temperature	Tstg	-40 to +125	

POWER CONSUMPTION (@ VDD= 3.0V, Ta= 25℃, GND= 0V)

Name	Sym.	Condition	Min.	Typ.	Max.	Unit
Normal Mode	I_{32K}	32.768 KHz Crystal mode, BCF=0, PH0=BCLK **without loading and LCD bias current**		4.5		uA
HALT mode	I_{HALT}			3.5	6	
STOP mode	I_{STOP}				1	
External R	$I_{Ext.\ R}$	R=150 KΩ oscillator operating, BCF=0, PH0=BCLK **without loading and LCD bias current**	42.5			
Pure LCD bias (LCD voltage divider resistor) Current	I_{LCD1}	LCD Low bias (Low Driving)		6		
	I_{LCD2}	LCD Normal bias (Normal Driving)		12		
	I_{LCD3}	LCD High bias (High Driving)		24		
	I_{LCD4}	LCD Higher bias (Higher Driving)		60		

Note: ***1.*** *When operating in External R oscillator mode, the current consumption will depend on the frequency of oscillation.*

2. *Normal Mode & Low Driving: $I_{32K} + I_{LCD1} = (4.5+6)\_uA$ ..(Typ.)*



ALLOWABLE OPERATING CONDITIONS (Ta= 25℃, GND= 0V)

Name	Symb.	Condition	Min.	Max.	Unit
Supply Voltage	VDD		2.0	3.6	V
Oscillator Start-Up Voltage	VDD_{stup}	32.768 KHz Crystal Mode	1.4		
Oscillator Sustain Voltage	VDD_{sut}		1.3		
Input “H” Voltage	Vih1		VDD-0.7	VDD+0.7	
Input “L” Voltage	Vil1		-0.7	0.7	
Operating Freq	Fopg1	32.768 KHz Crystal Mode	32		KHz
	Fopg2	External R mode	10	1000	

DC Output Characteristics (@ VDD= 2.0V, Ta= 25℃, GND= 0V)

Name	Symb.	Condition	Port	Min.	Typ.	Max.	Unit
Output “H” Voltage	Voh	Ioh=-1 mA	IOA,B,C,D	1.5			V
Output “L” Voltage	Vol	Iol=2 mA				0.9	V

ELECTRICAL CHARACTERISTICS

Input Resistance (VDD=3.0V)

Name	Symb.	Condition	Min.	Typ.	Max.	Unit
IOA,B,C Pull-Down Tr	Rmad1	Vi=VDD	200	500	1000	KΩ
INT Pull-Down Tr	Rintu1	Vi=VDD	200	500	1000	
INT Pull-up Tr	Rintd1	Vi=GND	200	500	1000	
RES Pull-Down R	Rres1	Vi=GND or VDD	9	35	90	

Segment Driver Output Characteristics

(@ VDD= 2.0V, Ta= 25℃, GND= 0V)

Name	Symb.	Condition	For	Min.	Typ.	Max.	Unit.
1/2 Bias Display Mode							
Output “H” Voltage	Vohf	Ioh=-1 uA	SEG-n		VDD		V
Output “L” Voltage	Volf	Iol=1 uA			0		
Output “H” Voltage	Vohg	Ioh=-10 uA	COM-n		VDD		
Output “M” Voltage	Vomg	Iol/h=+/-10 uA			0.5 *VDD		
Output “L” Voltage	Volg	Iol=10 uA			0		
1/3 Bias display Mode							
Output “H” Voltage	Vohi	Ioh=-1 uA	SEG-n		VDD		V
Output “M1” Voltage	Vom1i	Iol/h=+/-10 uA			1/3 *VDD		
Output “M2” Voltage	Vom2i	Iol/h=+/-10 uA			2/3 *VDD		
Output “L” Voltage	Voli	Iol=1 uA			0		
Output “H” Voltage	Vohj	Ioh=-10 uA	COM-n		VDD		
Output “M1” Voltage	Vom1j	Iol/h=+/-10 uA			1/3 *VDD		
Output “M2” Voltage	Vom2j	Iol/h=+/-10 uA			2/3 *VDD		
Output “L” Voltage	Volj	Iol=10 uA			0		

CHAPTER 2 TM8530 INTERNAL SYSTEM ARCHITECTURE

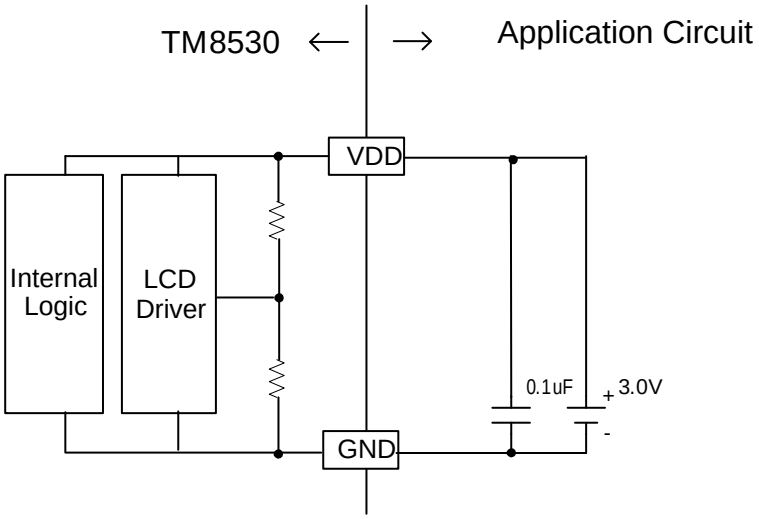
2-1. Power Supply

TM8530 can be operated at Li type voltage. The power supply circuitry also generates the necessary voltage level to drive the LCD panel at different bias. Shown below are the connection diagrams for 1/2 bias, 1/3 bias application.

2-1-1. LI BATTERY POWER SUPPLY

Operating voltage range: 2.0V ~ 3.6V (Li-B)

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



OPTION table :

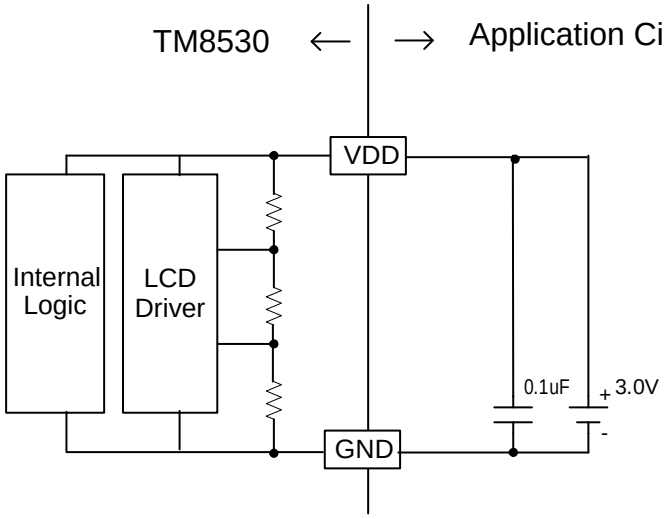
Option name	Selected item
LCD BIAS	(1) 1/2 BIAS

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

Note 2: The backup flag (BCF) is set in the initial clear mode. When the backup 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.**

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



OPTION table :

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

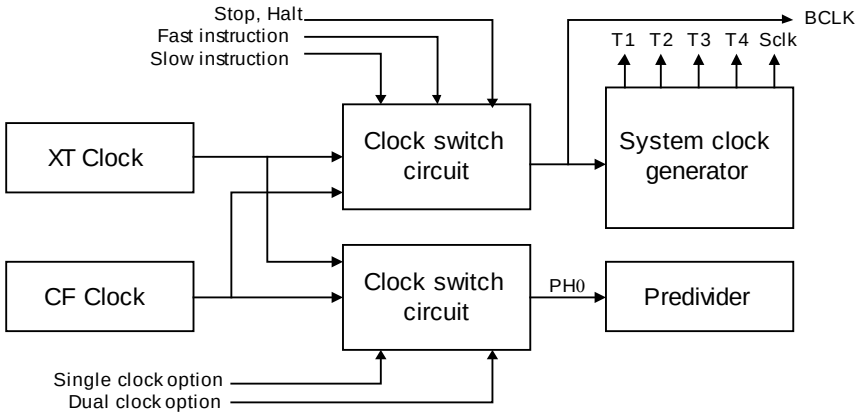
Note 1: The input/output ports operate between GND and VDD.

Note 2: The backup flag (BCF) is set in the initial clear mode. When the backup 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

The 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 provides the necessary clocks for the execution of instructions. The pre-divider generates several clocks with different frequencies for the usage of the LCD driver, the frequency generator, etc….
The following table shows the clock sources of system clock generator and the pre-divider under different conditions.

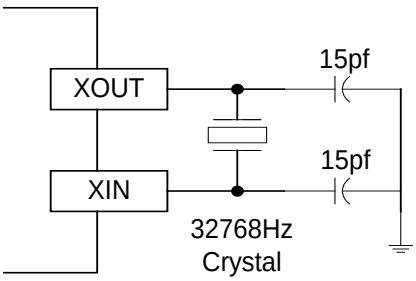
	PH0	BCLK
Slow clock only option	XT clock	XT clock
Fast clock only option	CF clock	CF clock
Initial state (dual clock option)	XT clock	XT clock
Halt mode (dual clock option)	XT clock	XT clock
Slow mode (dual clock option)	XT clock	XT clock
Fast mode (dual clock option)	XT clock	CF clock

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

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

MASK OPTION table :

Mask Option name	Selected item
SLOW CLOCK TYPE FOR SLOW ONLY OR DUAL	(1) X’tal



(1) X'tal

When the backup flag (BCF) is set to 1, the oscillator operates with a higher driving capability in order to reduce the start-up time of the oscillator. However, it will increase the power consumption. Therefore, the backup flag should be reset unless required otherwise.

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

	Li power option	ICE(TM8797)
BCF=1	Increase	Increased
BCF=0	Normal	
Initial reset	Increased	
After reset	Normal	

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

The CF clock consists of multiple types of oscillators (selectable in the option), which provides a faster clock sources to the system. In single clock operation (fast only), this oscillator provides the clock signals to the system clock generator, the pre-divider, the timer, the I/O port chattering prevention clock and the LCD circuitry. In dual clock operation, CF clock provides the clock signals to the system clock generator only.

When the dual clock option is selected in the option, this oscillator is 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 but the clock source for other functions will still come from the XT clock. The Halt mode, the stop mode and the execution of the SLOW instruction will stop this oscillator and the system clock (BCLK) will be switched to the XT clock.

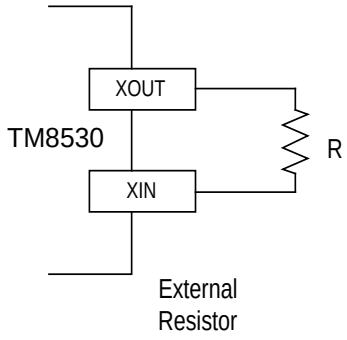
There are 2 type oscillators can be used as the fast clock oscillator, which can be selected in the option:

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

Mask OPTION table :

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

Option name	Selected item
FAST CLOCK OSC TYPE FOR FAST ONLY OR DUAL	(2) EXTERNAL RESISTOR



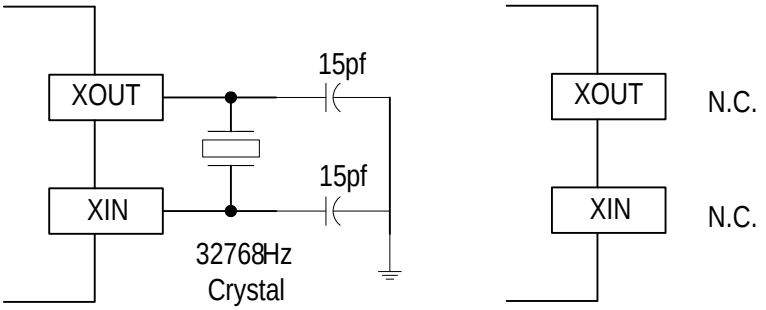
RC OSCILLATOR WITH INTERNAL RESISTOR (CF CLOCK)

This type of oscillator can be used in “FAST only” or “DUAL clock” options.

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) INTERNAL RESISTOR FOR 500 KHz



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

There are three combinations of the clock sources that can be selected in the option:

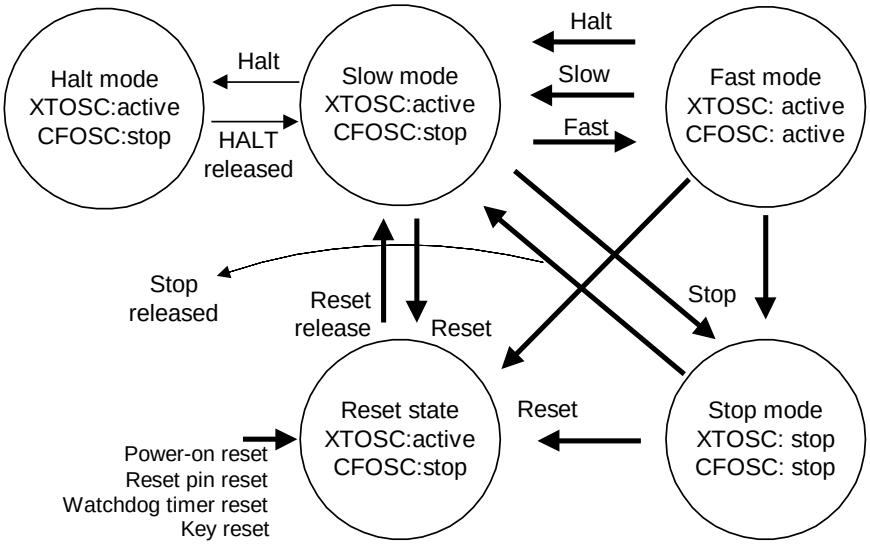
2-2-3-1 DUAL CLOCK

OPTION table :

Option name	Selected item
CLOCK SOURCE	(3) DUAL

The operation of the dual clock mode is shown in the following figure.

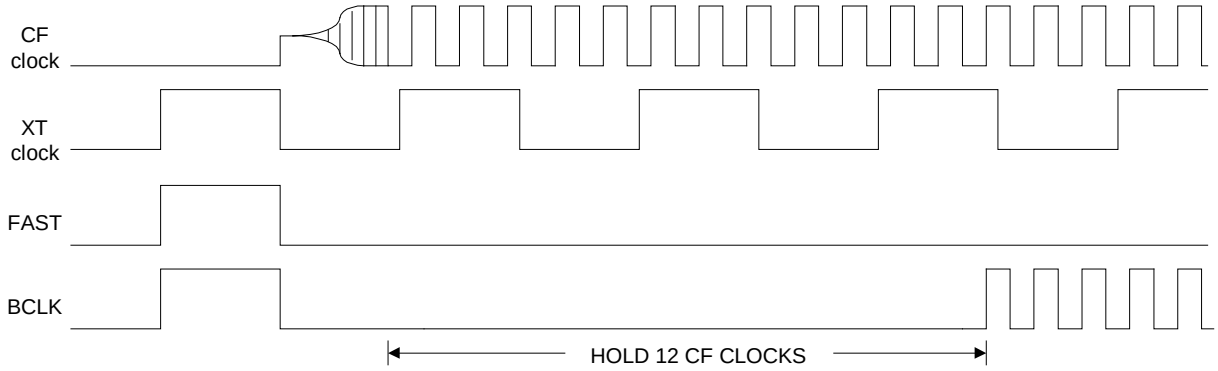
When this mode is selected in the option, the clock source (BCLK) of the system clock generator will switch between XT clock and CF clock according to the user's program. When the HALT and STOP instructions are executed, the clock source (BCLK) will switch to XT clock automatically.



The XT clock provides the clocks to the pre-divider, the timer, the I/O port chattering prevention and the LCD circuitry in this mode.

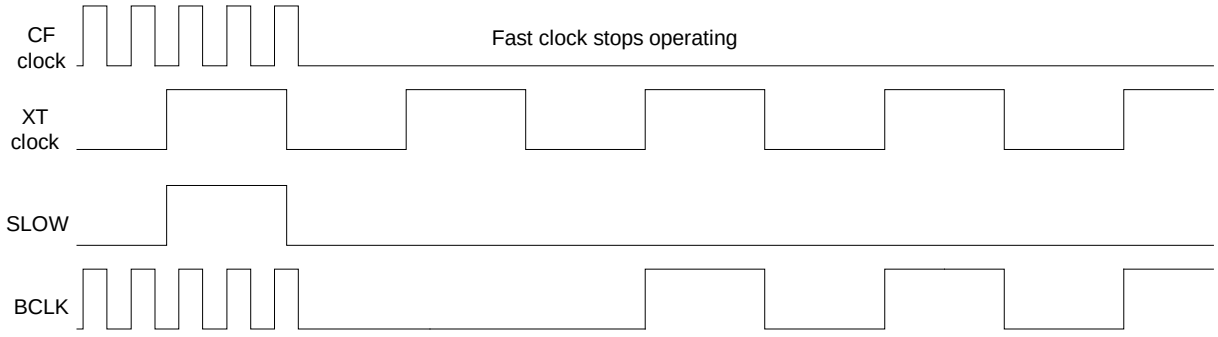
The state diagram of the dual clock mode is shown in the above figure.

After the execution of the FAST instruction, the system clock generator will hold for 12 CF clock cycles after the CF clock oscillator starts up and then BCLK will switch to the CF clock. It prevents the delivery of incorrect clock signals 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 for 2 XT clock cycles and then BCLK will switch to XT clock.



This figure shows the System Clock Switches from Fast to Slow

2-2-3-2 SINGLE CLOCK

OPTION table :

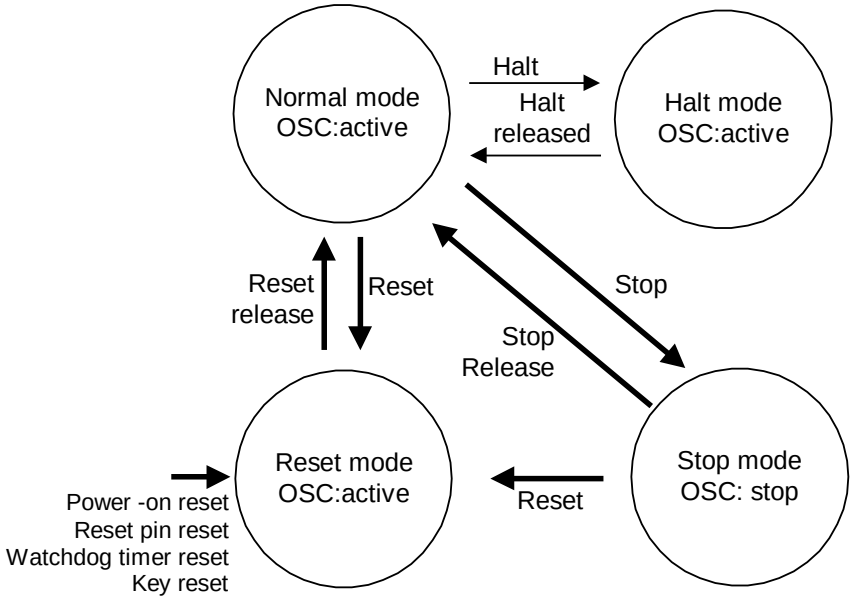
For Fast clock oscillator only

Option name	Selected item
CLOCK SOURCE	(1) FAST ONLY

For slow clock oscillator only

Option name	Selected item
CLOCK SOURCE	(2) SLOW ONLY

The operation of the single clock option is shown in the following figure.



This figure shows the State Diagram of Single Clock Option

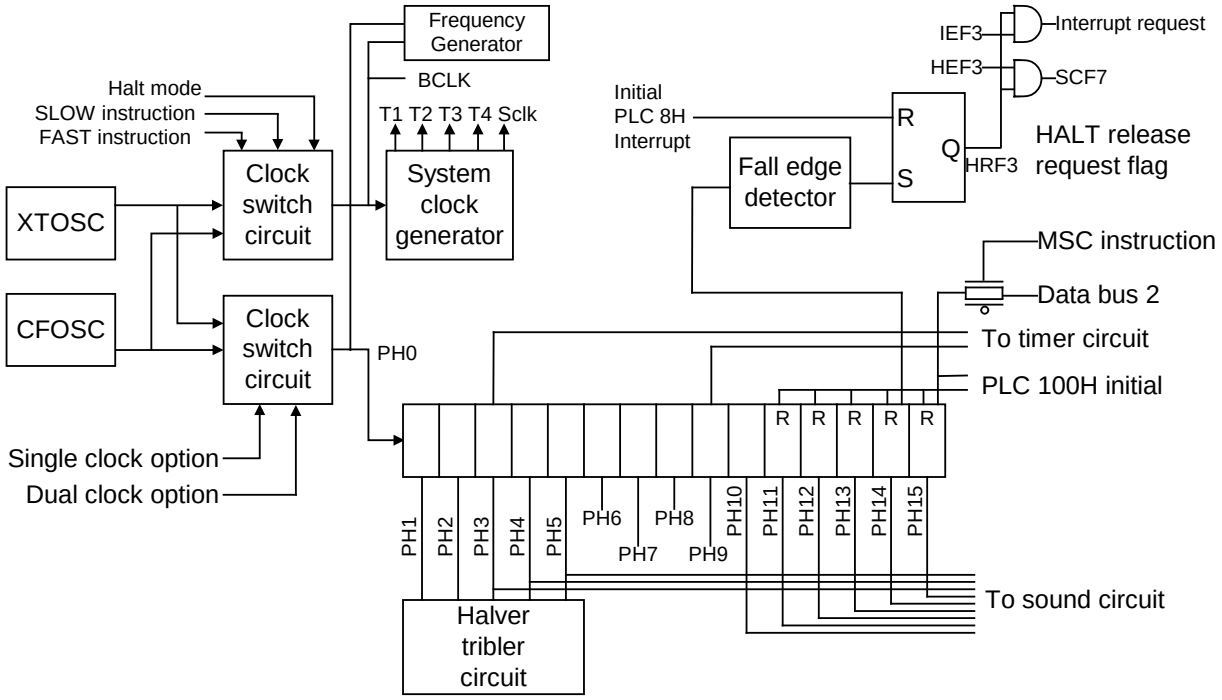
Either XT or CF clock can be selected in the option in this mode. The FAST and SLOW instructions will be treated as the NOP instruction in this mode.

The backup flag (BCF) will be set to 1 automatically before the program enters the stop mode. It can ensure that the Crystal oscillator starts up in a favorable condition.

2-2-4 PREDIVIDER

The pre-divider is a 15-stage counter that receives the clock signals from the output of the clock switch circuitry (PH0) as input. When PH0 changes from "H" level to "L" level, the content of this counter will change accordingly. The PH11 to PH15 of the pre-divider are reset to "0" when the PLC 100H instruction is executed or in the initial reset cylce.

The pre-divider delivers the signals to the halver / triple circuit, alternating frequency for LCD display, system clock, sound generator and halt release request signal (the I/O port chattering prevention function).



This figure shows the pre-divider and its peripherals

The falling edge of PH14 will set the halt mode release request flag (HRF3) to 1, in this case, if the pre-divider interrupt enable mode (IEF3) is set in advance, the interrupt comes from predivider is accepted; and if the halt release enable mode (HEF3) is set in advance, then the halt release request signal will be delivered and the start condition flag 7 (SCF7) in status register 3 (STS3) will be set.

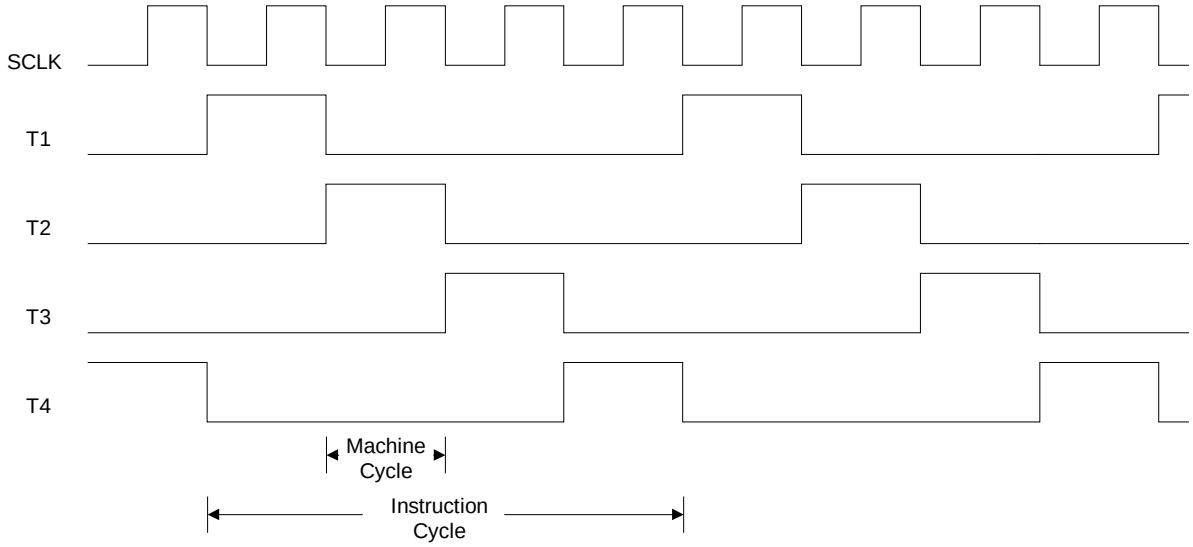
The clock source of the pre-divider is PH0; there are 4 kinds of frequencies of PH0 that can be selected in the option:

OPTION table :

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

The system clock generator provides the necessary clocks to control the execution of instructions.
The FAST and SLOW instructions can also be used to switch the clock input of the system clock generator.
The basic system clock is shown below:



2-3. PROGRAM COUNTER (PC)

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

- The program counter (PC) is normally incremented by one (+1) for every instruction execution.

  PC ← PC + 1

- When executing JMP instruction, subroutine call instruction (CALL), interrupt service routine or when reset occurs, the program counter (PC) will be loaded with the corresponding address in table 2-1.

  PC ← corresponding address shown in Table 2- 1

- When executing a jump instruction except JMP and CALL, the program counter (PC) will be loaded with the specified address in the operand of the instruction.

  PC ← current page (PC11) + specified address in the operand

- Return instruction (RTS)

  PC ← content of stack specified by the stack pointer

  Stack pointer ← stack pointer – 1

Table 2- 1

Interrupt No. / Interrupt Source	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 6 (RFC counter interrupt)	0	0	0	0	0	0	1	0	1	0	0	0
Jump instruction / Subroutine call	P11	P10	P9	P8	P7	P6	P5	P4	P3	P2	P1	P0

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

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

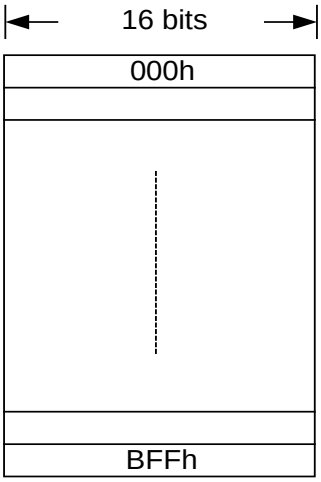
2-4. PROGRAM/TABLE MEMORY (ROM)

The built-in mask ROM is organized into 3072 x 16 bits.

Both instruction ROM (PROM) and table ROM (TROM) share this memory space together. The partition formula for PROM and TROM is as shown below:

Instruction ROM memory space = (2048+128 * N) words,

Table ROM memory space = 256(8 - N) bytes (N = 0~8).



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

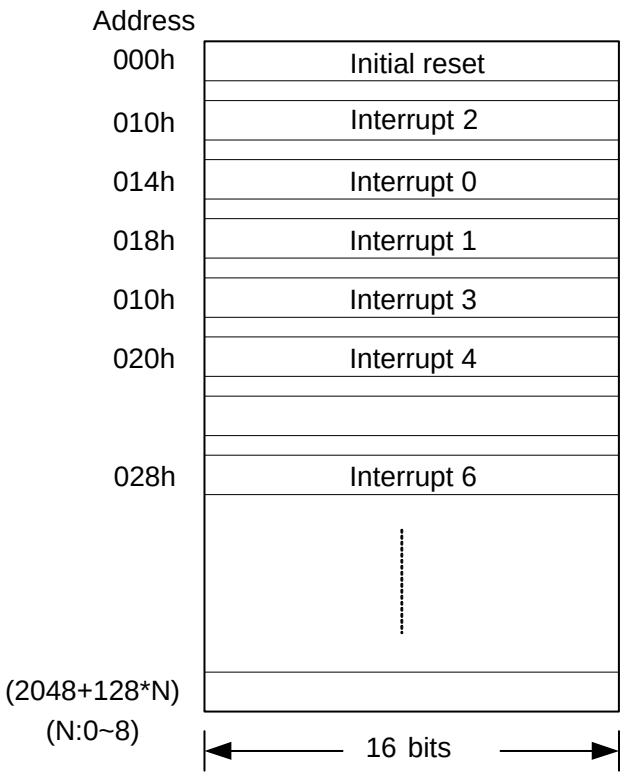
The partition of memory space is defined in mask option, the table is shown below:

MASK OPTION table :

Mask Option name	Selected item	Instruction ROM memory space (Words)	Table ROM memory space (Bytes)
INSTRUCTION ROM <-> TABLE ROM	1 (N=0)	2048	2048
INSTRUCTION ROM <-> TABLE ROM	2 (N=1)	2176	1792
INSTRUCTION ROM <-> TABLE ROM	3 (N=2)	2304	1536
INSTRUCTION ROM <-> TABLE ROM	4 (N=3)	2432	1280
INSTRUCTION ROM <-> TABLE ROM	5 (N=4)	2560	1024
INSTRUCTION ROM <-> TABLE ROM	6 (N=5)	2688	768
INSTRUCTION ROM <-> TABLE ROM	7 (N=6)	2816	512
INSTRUCTION ROM <-> TABLE ROM	8 (N=7)	2944	256
INSTRUCTION ROM <-> TABLE ROM	9 (N=8)	3072	0

2-4-1. INSTRUCTION ROM (PROM)

There are some special locations to 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), and interrupt 6 address (028H) in the program memory.



Instruction ROM (PROM) organization

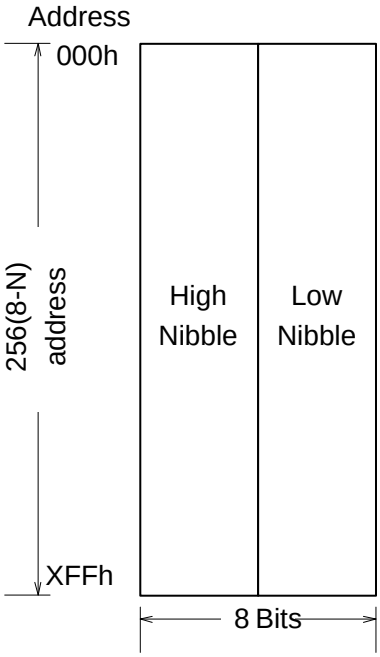


Table ROM (TROM) organization

This figure shows the Organization of ROM

2-4-2. TABLE ROM (TROM)

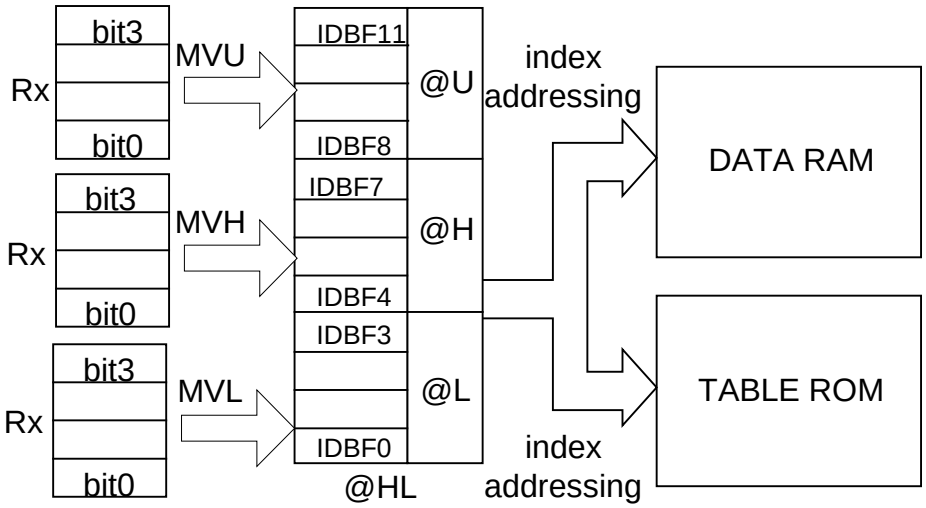
This memory space stores the constant data or look up tables for the usage of the main program. All the table ROM addresses can be specified by the index address register (@HL). The data width can be 8 bits or 4 bits depending on the different usage. Please refer to the explanation in the instruction chapter for details.

2-5. INDEX ADDRESS REGISTER (@HL)

It 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 executing MVU, MVH and MVL instructions can modify the contents of the register. The execution of the MVL instruction will load the content of the specified data memory to the lower nibble of the index register (@L). In the same manner, the execution of the 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)

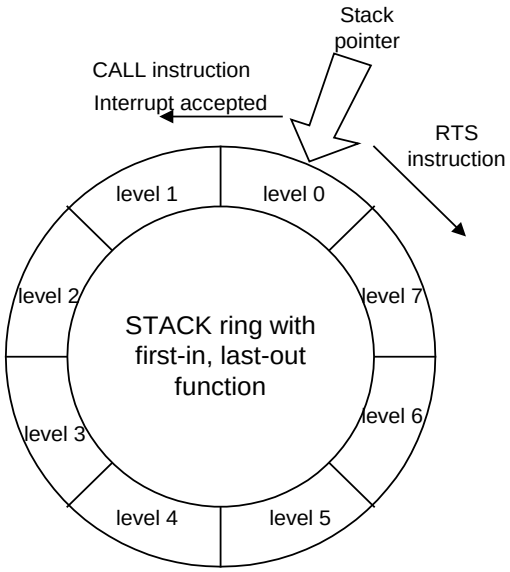
The Stack is a special data structure that follows the first-in-last-out rule. It is used to save the contents of the program counter sequentially during subroutine calls or the execution of the interrupt service routines.
The contents of the stack registers are returned sequentially to the program counter (PC) when return instructions (RTS) is executed.

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

Once a subroutine call or interrupt causes the stack registers (STACK) to 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 registers (STACK) are returned sequentially to the program counter (PC) during execution of the RTS instruction.
Once a RTS instruction causes the stack register (STACK) to 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.



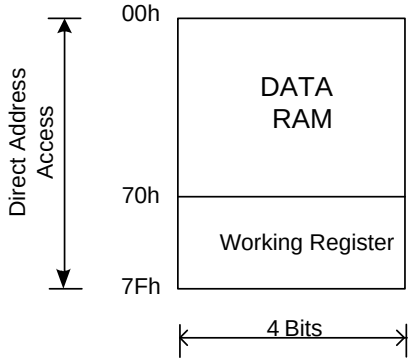
2-7. DATA MEMORY (RAM)

The static RAM is organized into 128 addresses x 4 bits and is used to store data.

The data memory may be accessed by 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) can address the data memory from 00H to FFH.

In addition, the 16 specified addresses (70H to 7FH) in the direct addressing memory are also used as 16 working registers. The function of working registers 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 7FH of the data memory (RAM) are not only used as general-purpose data memory but also used as the working register. The following will introduce the general usage of working registers:

1. To perform the arithmetic and logic operations on the contents of a working register and immediate data. Such as: ADCI, ADCI*, SBCI, SBCI*, ADDI, ADDI*, SUBI, SUBI*, ADNI, ADNI*, ANDI, ANDI*, EORI, EORI*, ORI, ORI*

2. To transfer the data between a working register and any address in the direct addressing data memory (RAM). Such as:

   MWR Rx, Ry; MRW Ry, Rx

3. To decode (or directly transfer) the contents of a 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 operations. 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. BINARY CONVERTS TO DECIMAL (BCD)

Decimal format is another numerical format supported by TM8530. When the content of the data memory is assigned as decimal format, it is necessary to convert the results into decimal format after the execution of ALU instructions. When the decimal converting operation is in execution, all the operands (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 binary 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 ≤ 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

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           ; The contents of the data memory  at address 10H and AC are
                     ; binary-added; the result is loaded into AC & the data memory
; address 10H. (R10 = AC = A(binary), CF = 0)
DAA*   10h           ; Convert the content of AC into decimal format.
                     ; The result in the data memory at address 10H is”0”and in
                     ; the CF is “1”. This represents the decimal number”10”.
```

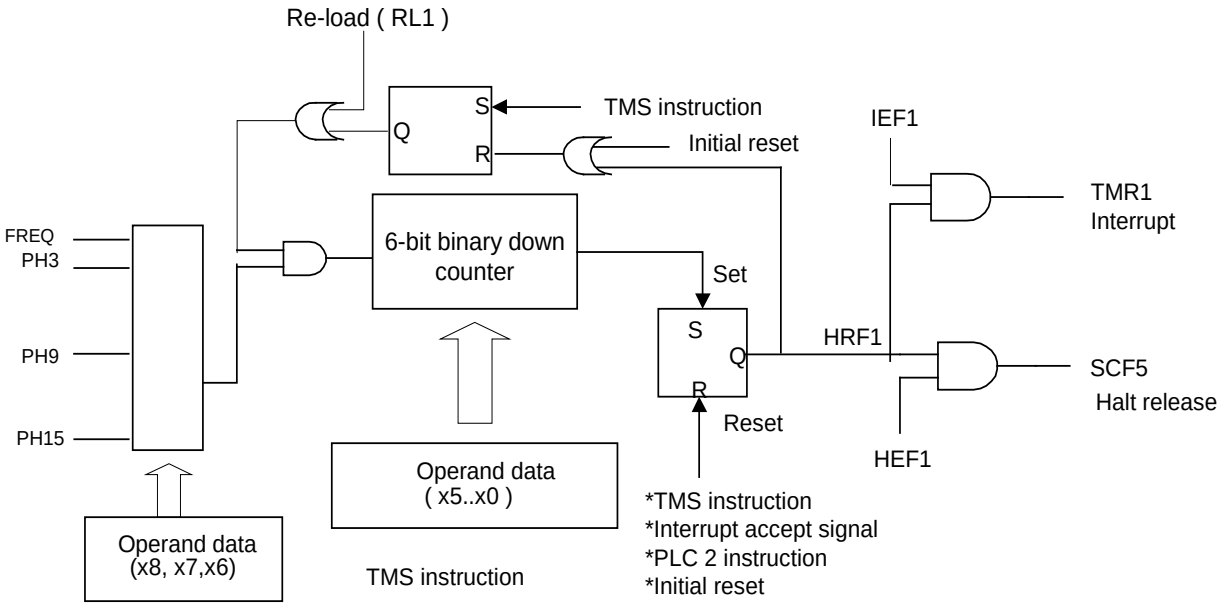
Instructions DAS, DAS*, DAS @HL can convert the data from binary format to decimal format after any subtraction operation. The conversion rules are shown in the following table and illustrated in Example 2.

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

Example 2:

```
LDS    10h, 1  ; Load immediate data”1”to the data memory address 10H.
LDS    11h, 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     ; The content of data memory address 10H is binary-subtracted;
               ; the result is loaded into data memory address
               ; 10H. (R10 = AC = F(binary), CF = 0)
DAS*   10h     ; Convert the content of the data memory at address 10H to decimal
               ; format.
               ; The result in the data memory at address 10H is”9” and in
               ; 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 can be loaded and enabled by executing the TMS and the TMSX instruction.

Once the TMR1 counts down to 3Fh, it will generate 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, an interrupt is generated.

When HRF1 = 1, if the IEF1 = 0 and the TMR1 halt release enabled (HEF1) = 1, the program will exit from the halt mode (if CPU is in the 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 a watchdog reset occurs, the clock source of TMR1 will still stay with the previous selection.

The following table shows the definition of each operand 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:

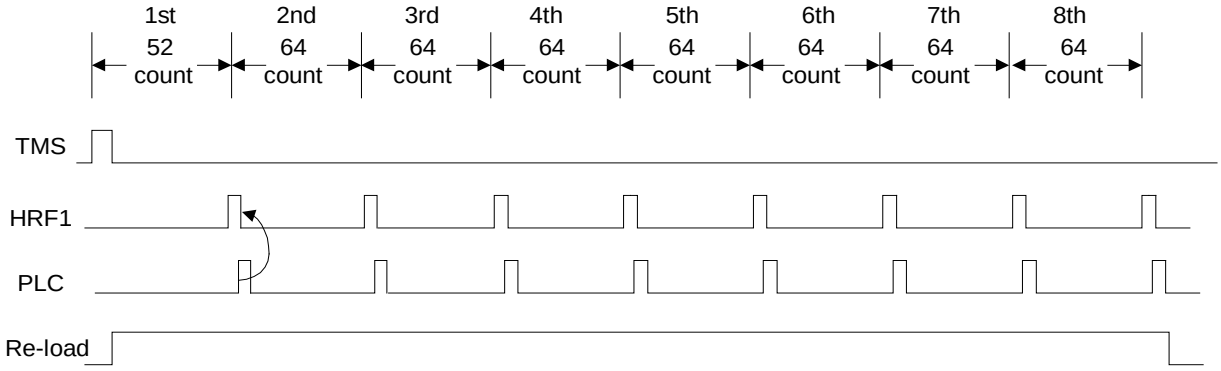
1. When the clock source of TMR1 is PH3
   TMR1 set time = (Set value + error) * 8 * 1/fosc (KHz) (ms)
2. When the clock source of TMR1 is PH9
   TMR1 set time = (Set value + error) * 512 * 1/fosc (KHz) (ms)
3. When the clock source of TMR1 is PH15
   TMR1 set time = (Set value + error) * 32768 * 1/fosc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

When the re-load function is enabled, the TMR1 will count down with a 3Fh initial data automatically if TMR1's underflow occurs. Once the re-load function has been disabled, TMR1's underflow will stop TMR1 immediately. During this operation, the program must use the halt release request flag or interrupt to calculate the desired counting value.

- It is necessary to execute either the TMS or the TMSX instructions to initialize the count down value before the re-load function is enabled, otherwise, TMR1 will automatically count down with an unknown value.
- Do not disable the re-load function before the last expected halt release or interrupt occurs. If the TMS related instructions are executed before a halt release or an 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 initial count down value of TMR1 to 52 and start counting, then enable the TMR1 halt release or interrupt function. Before the first underflow occurs, enable the re-load function. The TMR1 will continue operating even though TMR1 underflow occurs. When a halt release or an interrupt occurs, clear the HRF1 flag by executing the PLC instruction. After a halt release or an interrupt occurs 8 times, disable the re-load function and then the counting is completed.

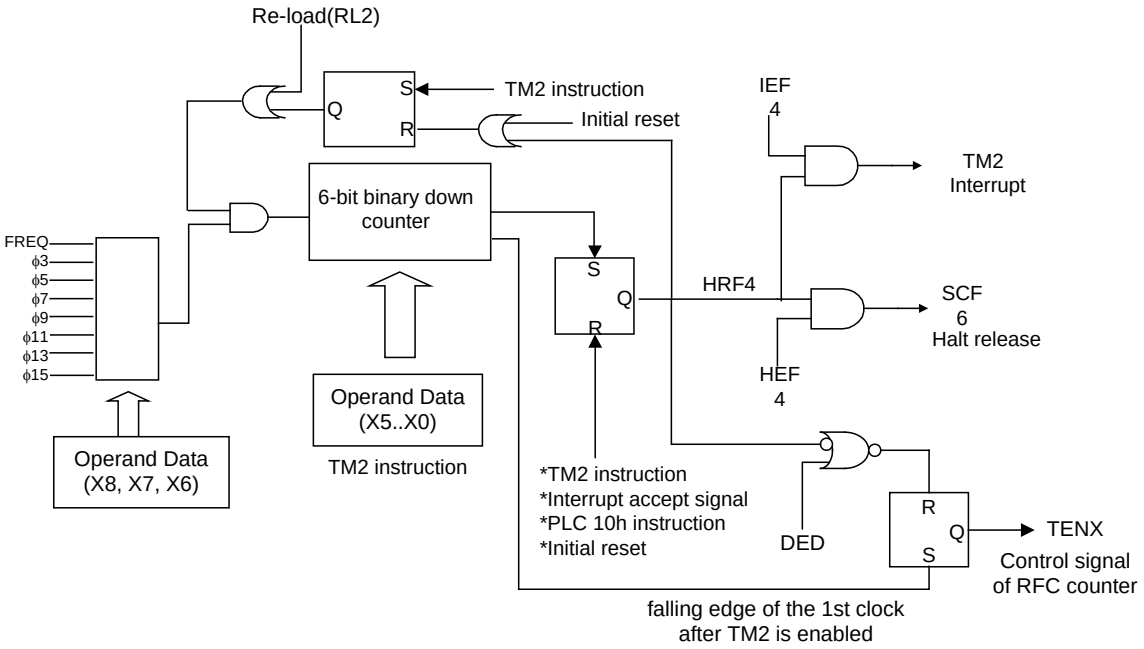


In the following example, S/W enters the halt mode to wait for the underflow of TMR1.

```
        LDS   0, 0          ;initiate the underflow counting register
        PLC   2
        SHE   2             ;enable the HALT release caused by TMR1
        TMSX  34h           ;initiatilize the TMR1 value (52) and the clock source is PH9
        SF    80h           ;enable the re-load function
RE_LOAD:
        HALT
        INC*  0             ;increment 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 can be loaded and enabled by executing the TM2 or the TM2X instruction.

Once the TMR2 counts down to 3Fh, it stops counting, and 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 occurs.

. When HRF4 =1, IEF4 = 0, and the TMR2 halt release enabler (HEF4) is set to 1, program will exit from the halt mode (if CPU is in the 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 a watchdog reset occurs, the clock source of TMR2 will stay with 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 clock source of TMR2 is PH3
   TMR2 set time = (Set value + error) * 8 * 1/fosc (KHz) (ms)
2. When the clock source of TMR2 is PH9
   TMR2 set time = (Set value + error) * 512 * 1/fosc (KHz) (ms)
3. When the clock source of TMR2 is PH15
   TMR2 set time = (Set value + error) * 32768 * 1/fosc (KHz) (ms)
4. When the clock source of TMR2 is PH5
   TMR2 set time = (Set value + error) * 32 * 1/fosc (KHz) (ms)
5. When the clock source of timer is PH7
   TMR2 set time = (Set value + error) * 128 * 1/fosc (KHz) (ms)
6. When the clock source of TMR2 is PH11
   TMR2 set time = (Set value + error) * 2048 * 1/fosc (KHz) (ms)
7. When the clock source of TMR2 is PH13
   TMR2 set time = (Set value + error) * 8192 * 1/fosc (KHz) (ms)
   Set value: Decimal number of the 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 clock source of TMR2 is FREQ
   TMR2 set time = (Set value + error) * 1/FREQ (KHz) (ms).
   FREQ: refer to section 3-3-4.

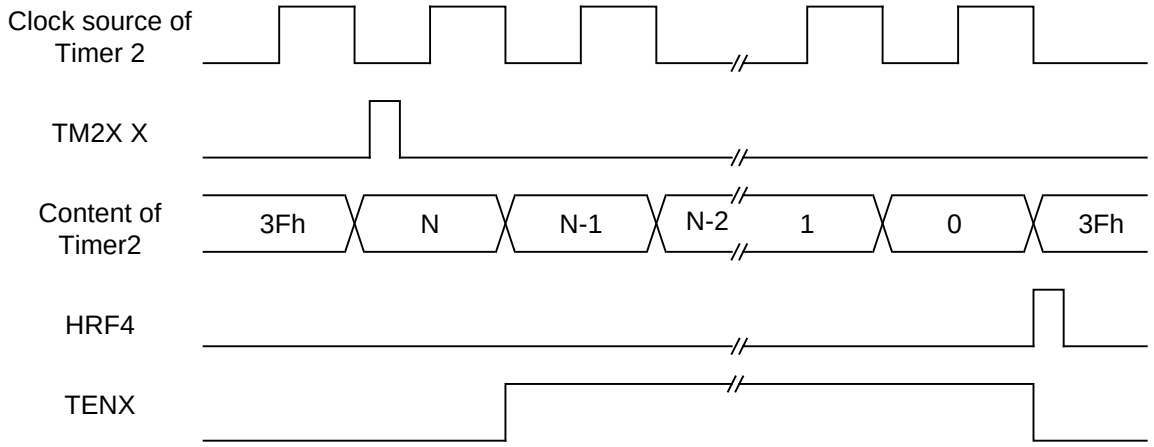
2-13-2 RE-LOAD OPERATION

TMR2 also provides a re-load function which works in the same fashion 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 controls the operation of the RFC function.

TMR2 sets TENX flag to 1 to enable the RFC counter. Once the TMR2 underflows, the TENX flag will be reset to 0 automatically. In behaving this way, Timer 2 can set an accurate time period without setting a value error like the other operations of TMR1 and TMR2. Refer to section 2-16 for more detail information on controlling the RFC counter. The following figure shows the operating timing of TMR 2 in RFC mode.



TMR2 can also controll the RFC function through the re-load function.

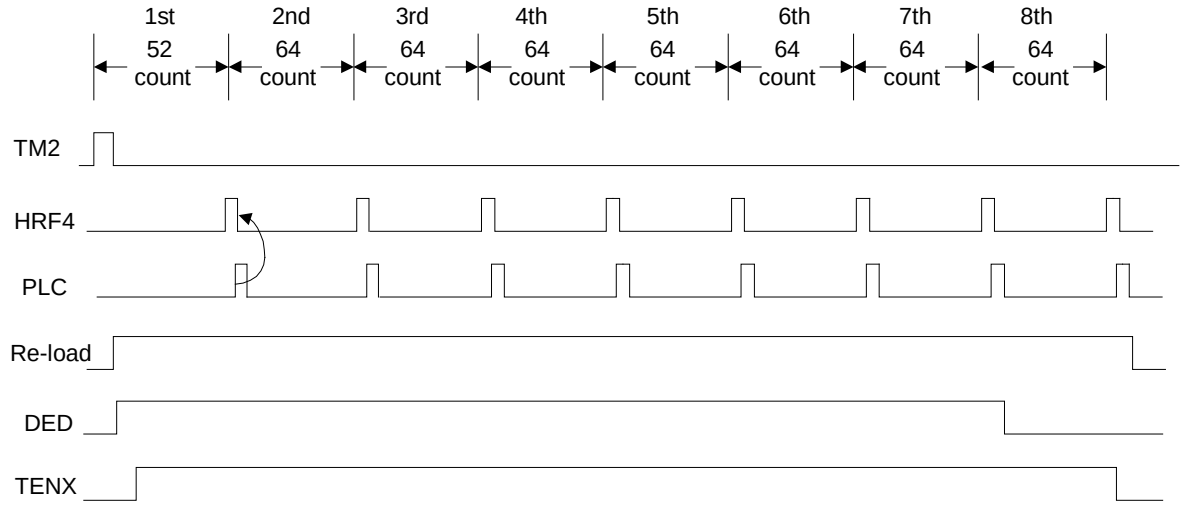
The SF2 1h instruction enables the re-load function, and the DED flag should be set to 1 by the SF2 2h instruction. Once the DED flag is set to 1, the TENX flag will not be cleared to 0 when TMR2 underflows (but HRF4 will be set to1). The DED flag must be cleared to 0 by executing RF2 2h instruction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

For example, if the expected count down value is 500, it will be divided as 52 + 7 * 64.

1. Set the initial value of TMR2 to 52 and start counting.
2. Enable the TMR2 halt release or the 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 a halt release or an interrupt occurs, clear the HRF4 flag by PLC instruction and increment the counting value to count the underflow times.
5. When a halt release or an interrupt occurs for the 7^{th} time, reset the DED flag.
6. When a halt release or an interrupt occurs for the 8^{th} time, disable the re-load function and then the counting is completed.

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

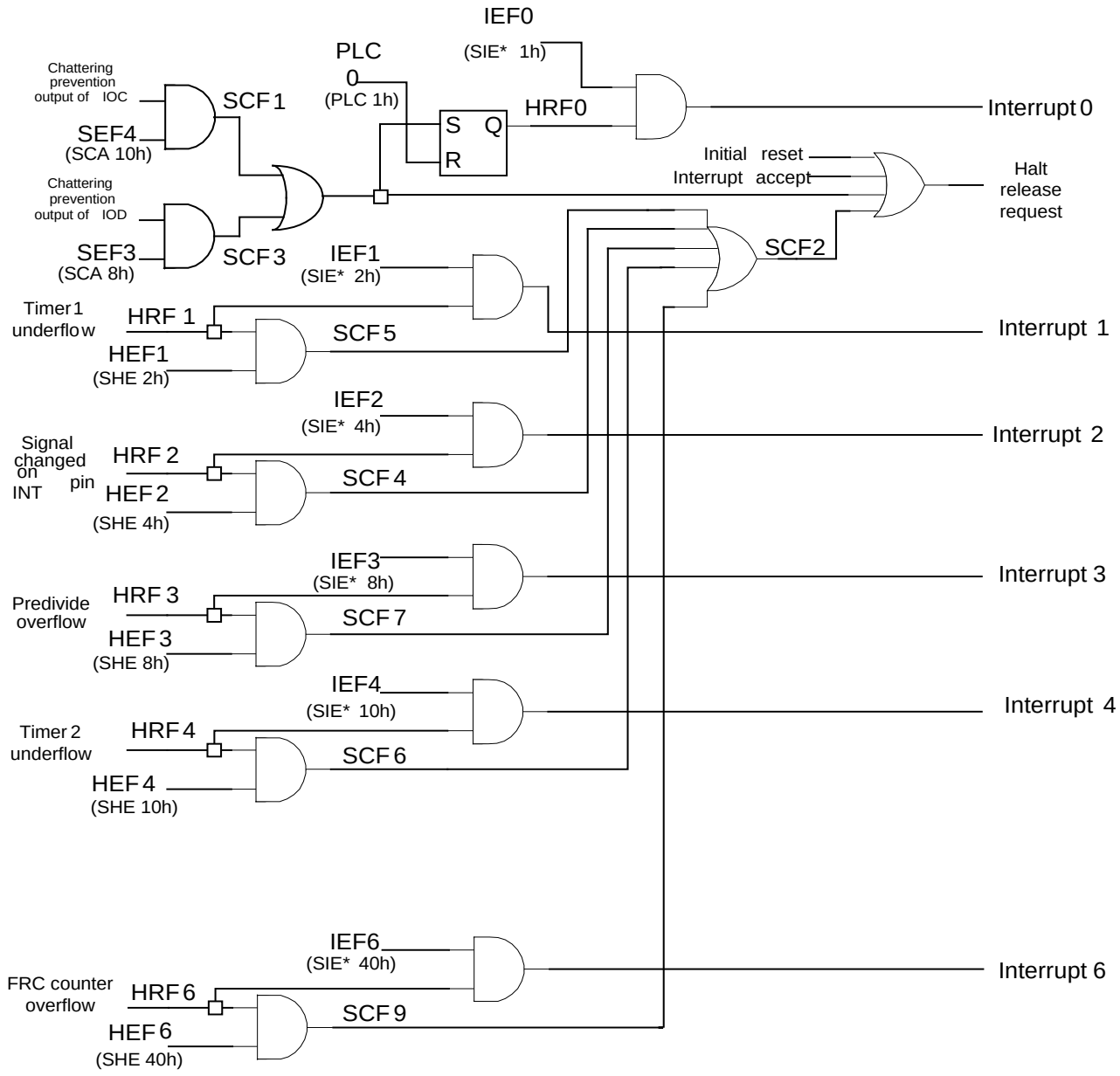
```
        LDS   0,0          ;initiate the underflow counting register
        PLC   10h
        SHE   10h          ;enable the halt release caused by TM2
        SRF   19h          ;enable RFC, and controlled by TM2
        TM2X  34h          ;initiate the TM value(52) and the clock source is PH9
        SF2   3h           ;enable the re-load function and set the DED flag to 1
RE_LOAD:
        HALT
        INC*  0            ;increment the underflow counter
        PLC   10h          ;clear HRF4
        LDS   20h, 7
        SUB   0            ;when halt is released at the 7th time, reset the DED flag
        JNZ   NOT_RESET_DED
        RF2   2            ;reset the DED flag
NOT_RESET_DED:
        LDA   0            ;restore 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 a 4-bit register 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 TM8530.



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)

It 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 transfers the data of the status register 1 (STS1) to the accumulator (AC) and the data memory (RAM).

4. The MRA instruction transfers the 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 the start condition flag 1, 2, 3 (SCF1, SCF2, SCF3) and the backup flag. The MSB instruction transfers the data of the status register 2 (STS2) to the accumulator (AC) and the data memory (RAM); the status register 2 (STS2) is read-only.

The following table shows the bit pattern of each flag in the 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 a signal change occurs on port IOC due to that the execution of SCA instruction and the halt mode is released as a result, 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 to 1. Also, if one or more start condition flags in SCF4, 5, 6, 7, 9 are set to 1, SCF2 will be set to 1 synchronously. When all the flags in SCF4, 5, 6, 7, 9 are cleared, the start condition flag 2 (SCF2) is reset to 0.
Note: If the start condition flag is set to 1, the program will not be able to enter the halt mode.

Start Condition Flag 3 (SCF3)

When a signal change occurs on port IOD due to that the execution of SCA instruction and the halt mode is released as a result, SCF3 will be set. Executing the SCA instruction will cause SCF3 to be reset to 0.

Backup Flag (BCF)

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

2-14-3 STATUS REGISTER 3 (STS3)

When the halt mode is released by the start condition flag 2 (SCF2), the status register 3 (STS3) will update in the corresponding status flag wherein the cause for the release of the halt mode.

Status register 3 (STS3) consists of 4 flags:

1. The Start Condition Flag 4 (SCF4)
   The Start condition flag 4 (SCF4) is set to 1 when the signal change on the INT pin causes the halt release request flag 2 (HRF2) to be output 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 enables flag 2 (HEF2).

2. Start Condition Flag 5 (SCF5)
   The 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 output 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 enables flag 1 (HEF1).

3. Start Condition Flag 7 (SCF7)
   The 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 output 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 enables flag 3 (HEF3).

4. The 15th stage's content of the pre-divider.
   The MSC instruction is used to transfer the contents of the 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 by the start condition flag 2 (SCF2), the status register 3X (STS3X) will update in the corresponding status flag wherein the cause for the release of the halt mode.
The Status register 3X (STS3X) consists of 2 flags:

1. 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 output 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).

2. 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 output 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 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)	NA
Halt release caused by RFC counter finish	NA	Halt release caused by TMR2 underflow	NA
Read only	Read only	Read only	Read only

2-14-5 STATUS REGISTER 4 (STS4)

The Status register 4 (STS4) consists of 3 flags:

1. The System Clock Selection Flag (CSF)
   The system Clock Selection Flag (CSF) shows which clock source of the system clock generator is selected by the system. Executing SLOW instruction will change the clock source (BCLK) of the system clock generator to the slow speed oscillator (XT clock), and the system Clock Selection Flag (CSF) will be reset to 0. Executing FAST instruction will change the clock source (BCLK) of the system clock generator to the fast speed oscillator (CF clock), and the system Clock Selection Flag (CSF) will be set to 1. For the operation of the system clock generator, refer to 3-3.

2. The Watchdog Timer Enable Flag (WTEF)
   The watchdog timer enable flag (WDF) shows the operating status of the watchdog timer.

3. The Overflow flag of the 16-bit counter of RFC (RFOVF)
   The overflow flag of the 16-bit counter of RFC (RFOVF) is set to 1 when the overflow of the 16-bit counter of RFC occurs. The flag will be 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 (WTEF)	System clock selection flag (CSF)
Read only	Read only	Read only	Read only

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

The 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 on the IOC port, which causes the stop release flag of the IOC port (CSR) to output, and the stop release enable flag 4 (SRF4) has to be set beforehand.

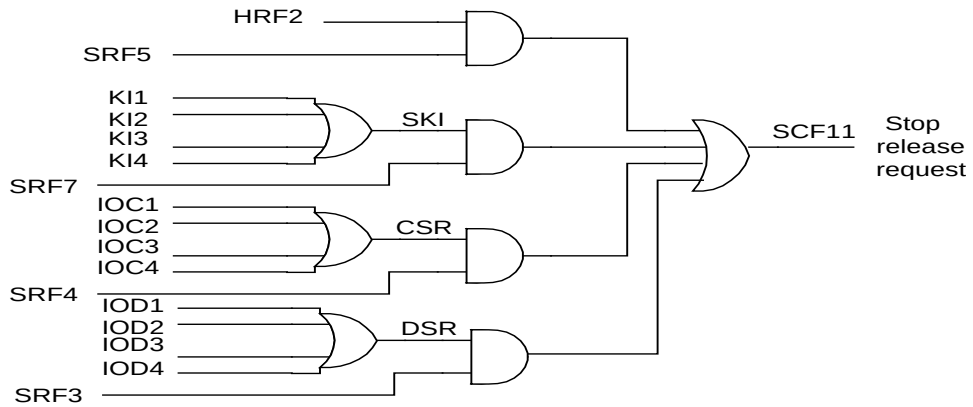
. A high level signal comes from the OR-ed output of the pins defined as input mode on IOD port, which causes the stop release flag of IOD port (DSR) to output. The stop release enable flag3 (SRF 3) must be set beforehand.

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

The stop release flags (CSR, DSR, HRF2) can be set by the stop release enable flags (SRFx). These flags should be cleared before the MCU enters the stop mode. All the pins corresponding to the IOA and IOC ports have to be defined as input mode and kept in 0 state before the MCU enters the STOP mode, otherwise the program cannot enter STOP mode.

The SRE Instruction SRE can set or reset the stop release enable flags (SRF4,5,7).

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



The following table shows the stop release request flags.

	The OR-ed input mode	The rising or falling

	pins of IOC(IOD) port	edge on INT pin
Stop release request flag	CSR(DSR)	HRF2
Stop release enable flag	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), is a 1-bit register:

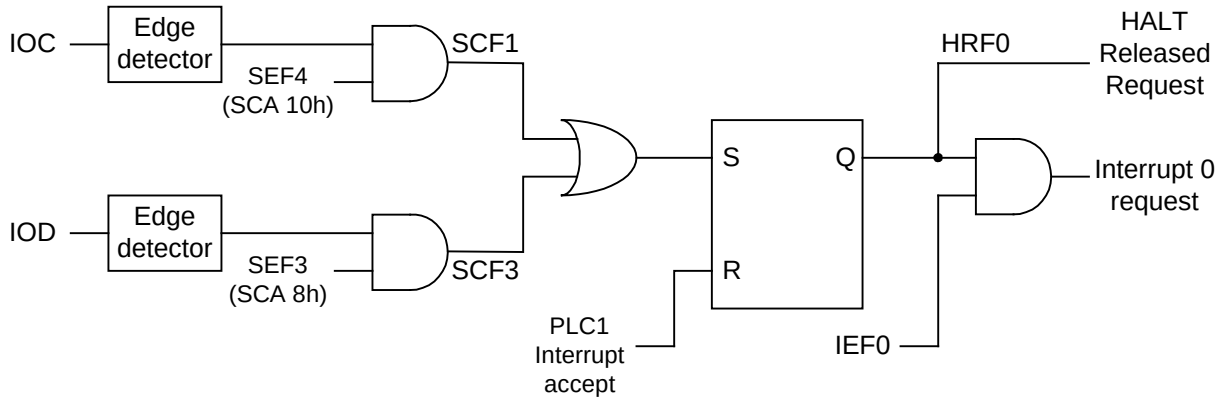
1. Switch Enable Flag 4 (SEF4)
   It stores the status of the input signal change on IOC pins in input mode that causes the halt mode or the stop mode to be released.
2. Switch Enable Flag 3 (SEF3)
   It stores the status of the input signal change on IOD pins in input mode that causes the halt mode or stop mode to be released.

Executed a SCA instruction can 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 Releasing the Halt Mode

If SEF4 is set to 1, a signal change on IOC port will cause the halt mode to be released, and set SCF1 to 1. Because the signal change on the IOC port is an ORed output of IOC1~4, it is necessary to keep the rest of input pins in “0” state and only one of the input signal can change the state.

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

If SRF4 and SEF4 are set, the stop mode will be released and set the SCF1 when a high level signal is applied to one of input mode pins of the IOC port and the other pins stay in “0” state.
After the stop mode is released, TM8530 enters the halt mode.
The high level signal must hold for a period long enough to cause the chattering prevention circuitry of the 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 functions by the execution of the SIE instruction to enable the interrupt function.

An input signal change on the IOC port pins will deliver the SCF1 when SEF4 has been set to 1 by executing the SCA instruction. After the SCF1 is delivered, the halt release request flag (HRF0) will be set to 1.
In behaving this way, if the interrupt enable flag 0 (IEF0) is set to 1 by executing the SIE instruction beforehand, the interrupt request flag 0 (interrupt 0) will be delivered to interrupt the program.
Once the interrupt 0 is accepted by SEF4 and IEF0, the interrupt 0 request to the next signal change on IOC will be inhibited. To release this mode, the SCA instruction must be executed again. Refer to 2-16-1-1.

2-15-2 CONTROL REGISTER 2 (CTL2)

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

Halt release enable flag	HEF6		HEF4
Halt release condition	Enable the halt release caused by RFC counter stop counting (HRF6)		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, the stop counting signal from the 16-bit counter of RFC will cause 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: e.g. an underflow signal from TMR1, the signal changes on 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, a signal change on the INT pin can cause the stop mode to be released.

2-15-3 CONTROL REGISTER 3 (CTL3)

The Control register 3 (CTL3) is organized into 6 bits of Interrupt Enable Flags (IEF) to enable / disable interrupts.
The Interrupt Enable Flag (IEF) is set / reset by the SIE* instruction. The bit pattern of control register 3 (CTL3) is shown below.

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

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

2-15-4 CONTROL REGISTER 4 (CTL4)

The 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 changes on the INT pins will cause the stop mode to be released. In the same manner, when SRF4 (SRF3) is set to 1, the input signal changes on the IOC port pin in the input mode and the signal changed on the INT pin causes the stop mode to be released respectively.

Example:

This example illustrates the stop mode released by port IOC and INT pin. Assuming all the IOD port pins and IOC port pins have been defined as input mode.

```
PLC     05h     ; Reset the HRF0 and HRF2.
SHE     04h     ; Set HEF2 and HEF5, a signal change on INT pin
                ; will cause the start condition flag 4 or 8 to be set.
SCA     18h     ; Set SEF4, the signal changes on port IOC and IOD
```

```
                        ; will cause the start conditions SCF1 to be set.
SRE     038h            ; SRF5,4 are set so that the signal changes on the port
                        ;IOC, IOD and INT pin will cause the stop mode to be released.
STOP                            ; Enter the stop mode.

  ……                    ;STOP release

MSC     10h             ; Check the signal change on INT pin that causes the stop
                        : mode to be released.

MSB     11h             ; Check the signal change on port IOC, IOD that causes the
                        : stop mode to be released.
```

2-16. HALT FUNCTION

The halt function can minimize the current dissipation of the TM8530 when the LCD is still 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 circuits are in operation (If the timer has started operating, the timer counter still operates in the halt mode).

After executing the HALT instruction and no halt release signals (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 the halt mode again by executing the RTS instruction after the completion of the interrupt service.

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

(2) A signal change specified by the SCA instruction is applied to port IOC (SCF1) /IOD (SCF3).

(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 to execute the MSB, MSC, or MCX instructions to test the halt release signal and that the PLC instruction is then executed to reset the halt release signal (HRF).

Even the HALT instruction is executed in the state that the halt release signal is delivered; the MCU does not enter the halt mode.

2-17. BACKUP FUNCTION

TM8530 provides a backup mode to avoid system malfunctioning under heavy loading occurs, such as active buzzer, lighting LED, etc..., since heavy loading will cause a large voltage drop in the supply voltage, the system will malfunction under this condition.

Once the program enters backup mode (BCF = 1), 32.768 KHz Crystal oscillator will operate in a large driver condition and the internal logic function operates with a higher supply voltage. TM8530 will get a higher power supply noise margin while backup mode is active, but it will also receive an increase in power consumption.

The backup flag (BCF) indicates the status of the backup function. When the BCF flag is set to 1, the MCU will enter the backup mode. The BCF flag can be set or reset by executing the SF or RF instructions, respectively.

The backup function performs differently with different power mode options as shown in the following table.

TM8530 status	3V battery
	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)

TM8530 Oscillation	3V battery or higher mode	
	BCF = 0	BCF = 1
32.768 KHz Crystal Oscillator	Small driver	Large driver

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

2-18. STOP FUNCTION (STOP)

The stop function is another way to minimize the current dissipation for TM8530. In stop mode, all the functions in TM8530 are put into hold state including oscillators. All of the LCD corresponding signals (COM and Segment) will output "L" level. In this mode, TM8530 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 the signals on the IOC port pins which are defined as input mode must be in the "L" state, and no stop release signals (SRFn) will be delivered. The CPU will then enter the stop mode by executing the STOP instruction.

The following conditions will cause the stop mode to be released.

. One of the signals on the IOC port pin in input mode is in "H" state and holds long enough to cause the CPU to be released from the halt mode.

. A signal is changing on the INT pin.

. The stop release condition specified by the SRE instruction is met (INT pin is exclusive).

When the TM8530 is released from the stop mode, the TM8530 will enter 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

TM8530 will enter the stop mode immediately. The backup flag (BCF) will be set to 1 automatically after the MCU enters the stop mode.

The following diagram shows the stop release procedure:

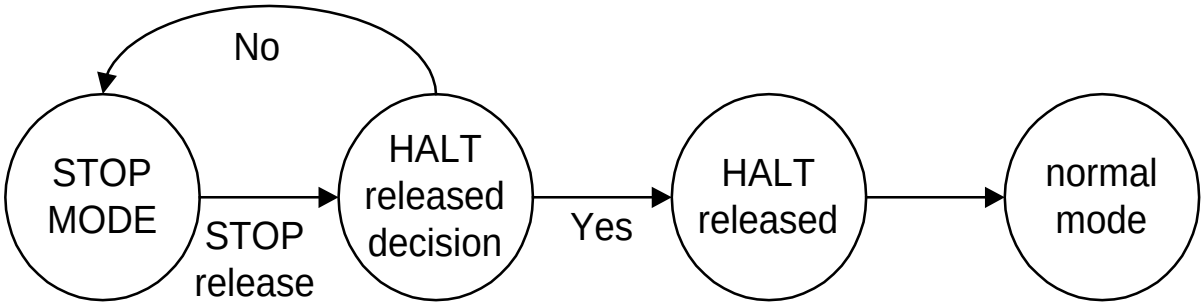


Figure 3- 16 The stop release state machine

Before the STOP instruction is executed, the following operations must be completed:

. Set the stop release conditions by the execution of the SRE instruction.

. Set the halt release conditions corresponding to the stop release conditions if needed.

. Set the interrupt conditions corresponding to the stop release conditions if needed.

When the stop mode is released by an interrupt request, the TM8530 will enter the halt mode immediately. Once the interrupt is accepted, the halt mode will be released and then enters the interrupt service routine. The MCU will return to the stop mode again by executing the RTS instruction after the interrupt service is completed.

Once the MCU is released from the stop release mode, the execution of MSB, MSC or MCX instruction can test the halt release signals and the execution of the PLC instruction can reset the halt release signals. If the stop instruction is executed in the state that the stop release signal (SRF) is delivered, the CPU will not enter the stop mode but enter 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

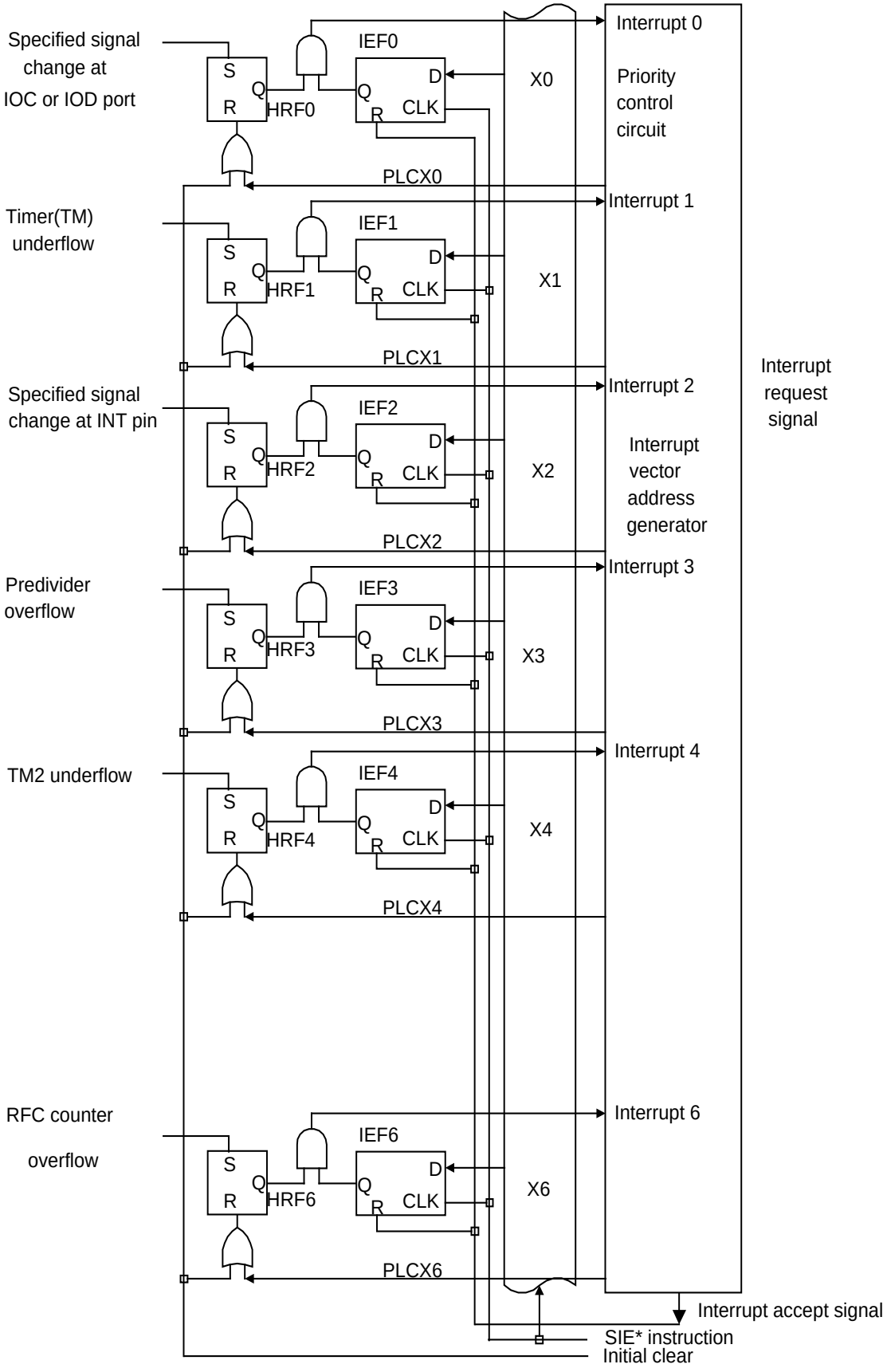
There are 7 different kinds of interrupt: 3 external interrupt factors and 4 internal interrupt factors. When an interrupt is accepted, the program in execution is suspended temporarily and the corresponding interrupt service routine specified by a pre-determined address in the program memory will be 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	6^{th}	5^{th}	2^{nd}	1^{st}	3^{rd}	4^{th}
Interrupt request flag	Interrupt 2	Interrupt 0	Interrupt 1	Interrupt 3	Interrupt 4	Interrupt 6

The following figure shows the Interrupt Control Circuit


Specified signal change at IOC or IOD port
Timer(TM) underflow
Specified signal change at INT pin
Predivider overflow
TM2 underflow
RFC counter overflow
IEF0
IEF1
IEF2
IEF3
IEF4
IEF6
HRF0
HRF1
HRF2
HRF3
HRF4
HRF6
PLCX0
PLCX1
PLCX2
PLCX3
PLCX4
PLCX6
X0
X1
X2
X3
X4
X6
Interrupt 0
Interrupt 1
Interrupt 2
Interrupt 3
Interrupt 4
Interrupt 6
Priority control circuit
Interrupt vector address generator
Interrupt request signal
Interrupt accept signal
SIE* instruction
Initial clear

3-1-1 INTERRUPT REQUEST AND SERVICE ADDRESS

3-1-1-1 External Interrupt Factor

The external interrupts are generated by the INT pin, the IOC ports, or the IOD ports.

1. External INT pin interrupt request
   In the option, either a rising or falling edge of the signal on the INT pin can be selected for generating an interrupt. If the Interrupt Enable Flag 2 (IEF2) is set beforehand and a signal change on the INT pin change that matches the option, it will generate a HRF2, the interrupt 2. Once the interrupt request is accepted, the instruction at address 10H will be executed automatically. It is necessary to hold the signal level for at least 1 machine cycle after the signal edge changes.

2. I/O port IOC, IOD interrupt request.
   An interrupt request signal (HRF0) will be generated when an input signal changes at I/O port IOC, IOD matches what is specified by the SCA instruction. In this case, if the interrupt enable flag 0 (IEF0) is set to 1, interrupt 0 is accepted and the instruction at address 14H 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 generated when Timer1 / 2 (TMR1/ 2) underflows. In this case, if the interrupt enable flag 1 / 4 (IEF1 / 4) is set beforehand and the 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. The 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 is inactive and the 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 requests are requested simultaneously and all interrupts are enabled beforehand, 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 they can be set by executing the SIE instruction again. Refer to Table 3-1.

Example:

; Assume all interrupts are requested simultaneously and all interrupts are enabled, and all the IOC port pins 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.

                    ; an interrupt caused by the predivider overflow occurs, and the
                    ; 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) The address of the suspended instruction will be stored into the stack register (STACK) as the return address of the interrupt service routine.

(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 cleared.

When an interrupt occurs, the TM8530 will follow the procedure below:

```
Instruction 1       ;an interrupt is accepted by the MCU.
NOP                 ;TM8530 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.
```

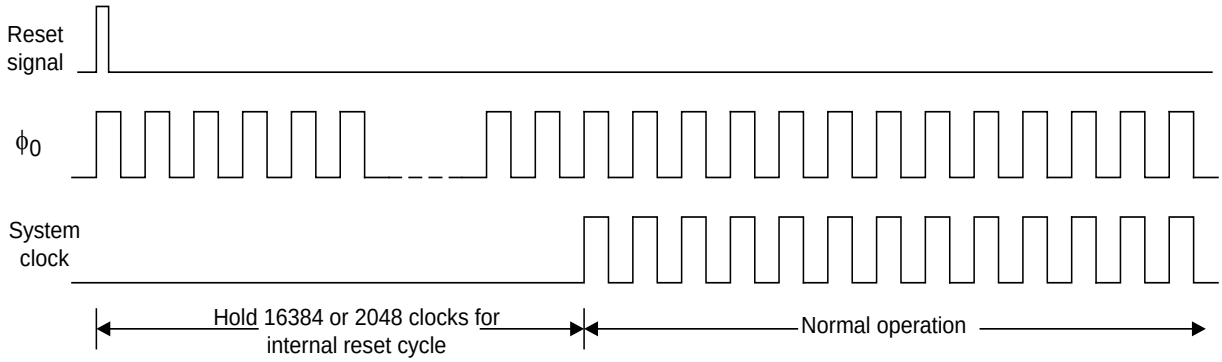
```
Instruction B
Instruction C
 .............
RTS                 ;finish the interrupt service routine
Instruction 1*      ;re-execute the instruction which was interrupted.
Instruction 2
```

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

TM8530 provides four kinds of reset functions: power-on reset, RESET pin reset, IOC port reset and watchdog timer reset.
When a reset signal is accepted, TM8530 will generate a time period for the initial reset cycle and there are two types of initial reset cycle time could be selected in the option, the one is PH15/2. The reset cycle time will be extended 16384 clocks (clock source comes form pre-divider) long at least.



3-2-1 RESET PIN RESET

When "H" level is applied on the reset pin, a reset signal will be generated. There is a built -in pull down resistor on this pin.
It is recommended to connect a capacitor (0.1 uf) between the RESET pin and VDD. This connection can prevent the signal bounce on the RESET pin.

3-2-1-1 Level Reset

Once an “H” signal is applied on the RESET pin, TM8530 will not enter the initial reset cycle until the signal on the RESET pin returns to “0”. After the signal on reset pin is cleared to 0, TM8530 starts the internal reset cycle and then release the reset status automatically.

The following table shows the initial condition of TM8530 in reset cycle.

Program counter	(PC)	Address 000H

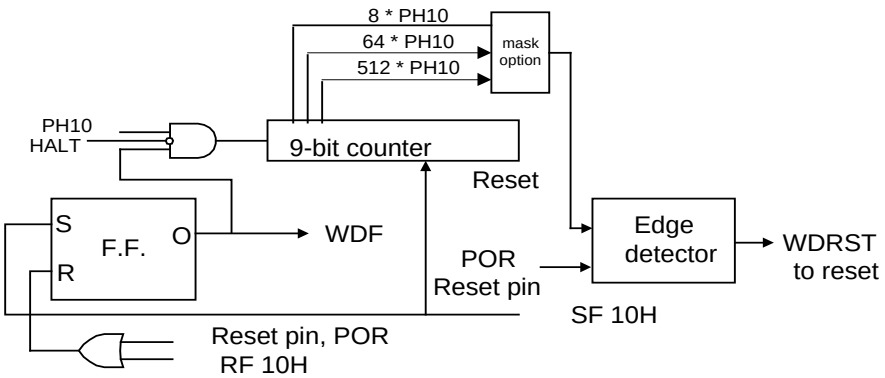
Program counter	(PC)	Address 000H
Start condition flags	(SCF1-7)	0
Backup flag	(BCF)	1 (Li-B option)
Stop release enable flag	(SRF3,4,5,7)	0
Switch enable flags	(SEF3,4)	0
Halt release request flag	(HRF 0~6)	0
Halt release enable flag	(HEF1-6)	0
Interrupt enable flag	(IEF0-6)	0
Alarm output	(ALARM)	DC 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 (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: 1. PH3 is the 3rd output of the predivider.
2. PH10 is the 10th output of predivider
3. All the LCD segment pins can be set to output all-ON or all-OFF signals during reset cycle in the option

3-2-4 WATCHDOG RESET

The watchdog timer can prevent the unexpected execution sequences caused by run-away software. 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 will generate a reset signal to reset TM8530. Most of the functions in TM8530 will be re-initiated except for the watchdog timer itself (which is still active), the 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 the initial reset cycle (power on reset [POR] or reset pin), the timer is inactive and the watchdog flag (WDF) is reset. The instruction SF 10h will enable the watchdog timer and set the watchdog flag (WDF) to 1. At the same time, the content of the watchdog timer will be cleared. Once the watchdog timer is enabled, the timer will pause when the program enters the halt mode or the stop mode. When the TM8530 wakes up from the halt or the stop mode, the timer operates continuously. It is recommended to execute a SF 10h instruction before the program enters the halt or the stop mode. This will keep the MCU away from the unexpected reset when it releases from halt or stop mode.

Once the watchdog timer is enabled, the program must execute the SF 10h instruction to clear the watchdog timer periodically. This will prevent the watchdog timer from overflow.

The overflow time interval of watchdog timer is selected in the option:

OPTION table :

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

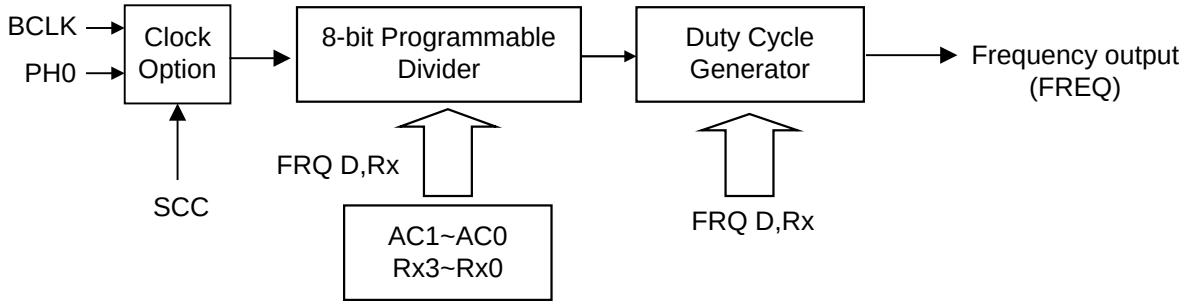
Note: timer overflow time interval is about 16 seconds when PH0 = 32.768 KHz

3-3. CLOCK GENERATOR

3-3-1 FREQUENCY GENERATOR

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

The following shows the organization of the frequency generator.



Executing the SCC instruction can select clock source 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 a letter D into the duty cycle generator. The frequency generator will then output the clock signals with 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)

The scaling data N is preset by the content of data memory and the accumulator (AC), the table ROM data or the 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 scaling data N

	The bit pattern of preset scaling data 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

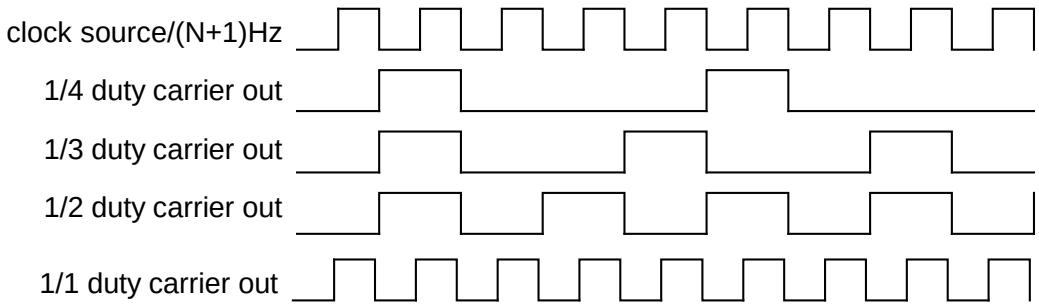
Notes: 1. T0 ~ T7 represents the data of table ROM.

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

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

Preset data 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 APPLICATION

The frequency generator may generate specified frequencies to compose melody music and the note table for those specified frequenies is shown below:

1. The clock source is PH0, i.e. 32.768 KHz
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. The above variation does not include X'tal variation.
2. If PH0 = 65536 Hz, C3 - B5 may have more accurate frequency.

For the melody application, the output signal of frequency generator has to be conveyed to the buzzer output (BZB, BZ) in order to accomplish the whole function. For more detail information about Buzzer output function, refer to section 3-4.

3-3-3 Halved / Doubler / Tripler

The halver / doubler / tripler circuitry generates the necessary bias voltage for LCD driver, this circuitry consists of a combination of PH2, PH3, PH4, PH5.

3-3-4 Alternating Clock for LCD Driver

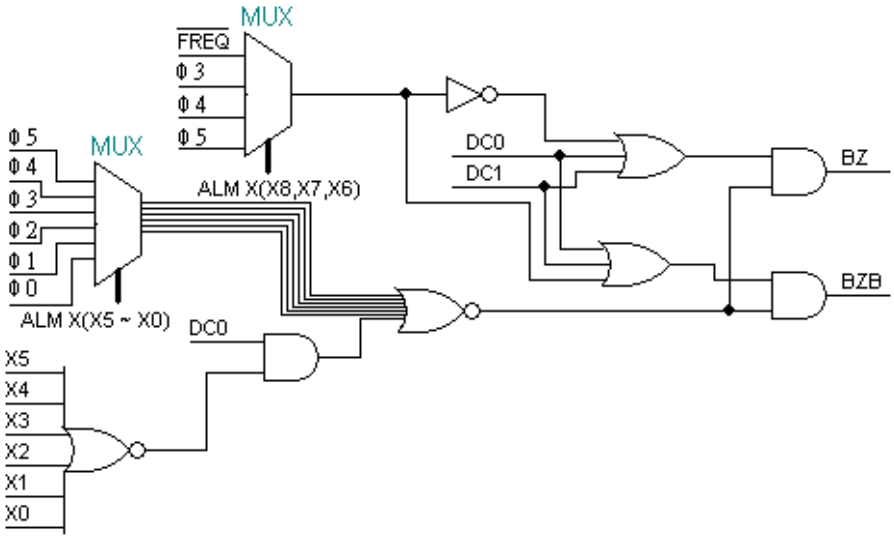
The alternating clock is the basic clock for LCD driver. Both COM and SEG pins shall change their output waveforms according to the alternating clock.

3-4. BUZZER OUTPUT PINS

TM8530 provides a pair of buzzer output pins known as BZB and BZ, which are pin-shared with I/O pins, IOB3 and IOB4, and can be configured in mask option respectively. BZB and BZ pins are versatile output pins with complementary output polarity. When the buzzer output function combined with the clock source comes from the frequency generator, it can generate a melody, a sound effect or the carrier output for the remote controller.

OPTION table :

Option name	Selected item
*SEG30(DC30)/IOB3/BZB	(3) BZB
SEG31/IOB4/BZ	(3) BZ



This figure shows the organization of the buzzer output.

3-4-1 SOUND EFFECT APPLICATION

The buzzer output pins (BZ, BZB) are suitable for driving the buzzer through a transistor with one output pin or driving the buzzer with both BZ and BZB pins directly. It is capable of outputting a modulation waveform of any combination of the frequency generator's output signal, PH3 (4096 Hz), PH4 (2048 Hz), PH5 (1024 Hz) and multiple signals of PH10 (32 Hz), PH11 (16 Hz), PH12 (8 Hz), PH13 (4 Hz), PH14 (2 Hz), PH15 (1 Hz). 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:

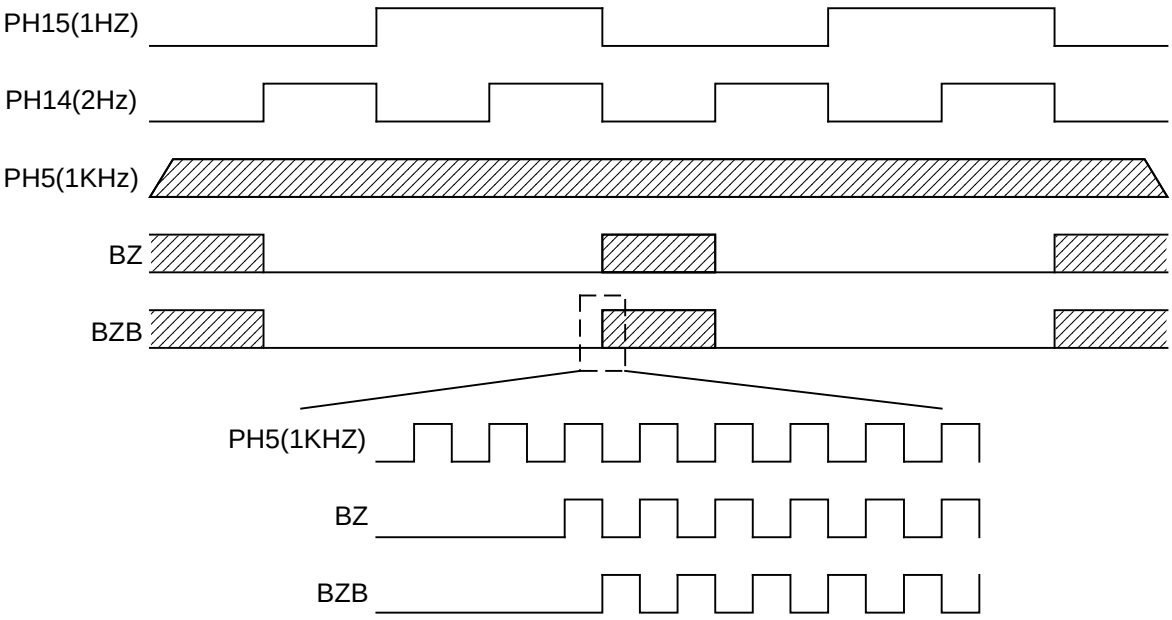
1. The higher frequency clock source should be only one of PH3, PH4, PH5 or FREQ, and the lower frequency may be any/all of the combinations from any/all of PH10 ~ PH15.
2. The frequencies in () corresponding to the input clock of the pre-divider (PH0) is 32768 Hz.
3. The BZ and BZB pins will output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

Example:

Buzzer output generates a waveform with 1 KHz 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 REMOTE CONTROLLER APPLICATION

If the 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 the usage of remote controller, the preset scaling data N of the frequency generator must be greater than or equal to 3, and the ALM instruction must be executed immediately following 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 the initial value of timer 1 to 3Fh and the clock source to
                      ;PH9.
SCC          40h      ;set the clock source of  the frequency generator to BCLK.
FRQX         2, 3     ;FREQ = BCLK / (4*2), preset scaling data of the frequency
                      ;generator
                      ;to 3 and duty cycle to 1/2.
ALM          1C0h     ;FREQ signal is output. This instruction must be executed
                      ;following the FRQ related instructions.
HALT                  ;waiting for the halt release (timer 1 underflows).
……                    ;Halt released.
ALM          0        ;stop the buzzer output.
```

3-5. INPUT / OUTPUT PORTS

Three I/O ports are available in TM8530: IOA, IOB and IOC. Each I/O port has the same basic function and consists of 4 bits (or 2 bits)
When the I/O pins are defined as non-IO function in the 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 as defined in the option.

OPTION table :

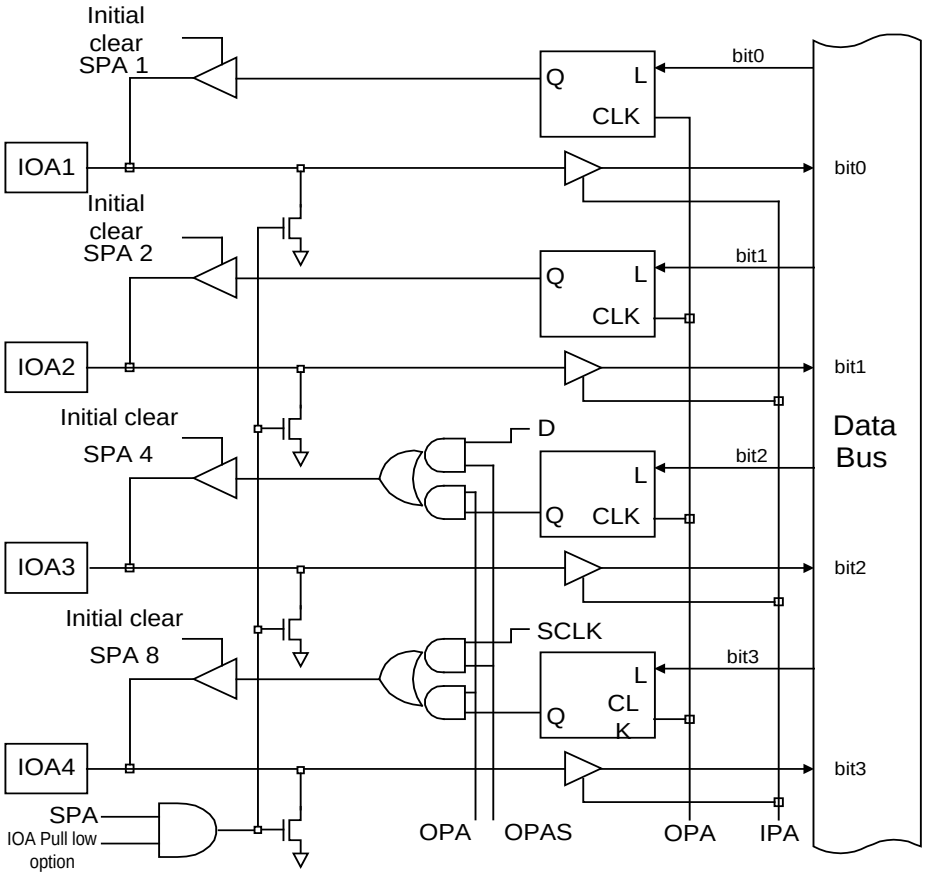
Option name	Selected item
SEG24/IOA1/CX	(2) IOA1
SEG25/IOA2/RR	(2) IOA2
SEG26/IOA3/RT	(2) IOA3
SEG27/IOA4/RH	(2) IOA4

The default setting of IOA port is input mode in initial reset cycle, each bit of the port can be defined as input mode or output mode respectly by executing a SPA instruction. Executing an OPA instruction can output the content of the specified data memory to the pins which have been defined as output mode.

Executing an IPA instruction can store the I/O pins' signal into the specified data memory locations. When the IO pins are defined as output mode, executing an IPA instruction will store the content of the latch of the output pin into the specified data memory location.

Before executing the SPA instruction to set the I/O pins to 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 changes.

The IOA port has a built-in pull-low resistor which can be selected in mask option and be enabled / disabled by executing a SPA instruction.



This figure shows the organization of IOA port.

Note: The pins in input mode should not be in floating, or a large current (straight-through current) will flow into the input buffer.

3-5-1-1 Pseudo Serial Output

The IOA port can operate as a pseudo serial output port by executing an OPAS instruction. The IOA port must be defined as output mode before executing an OPAS instruction.

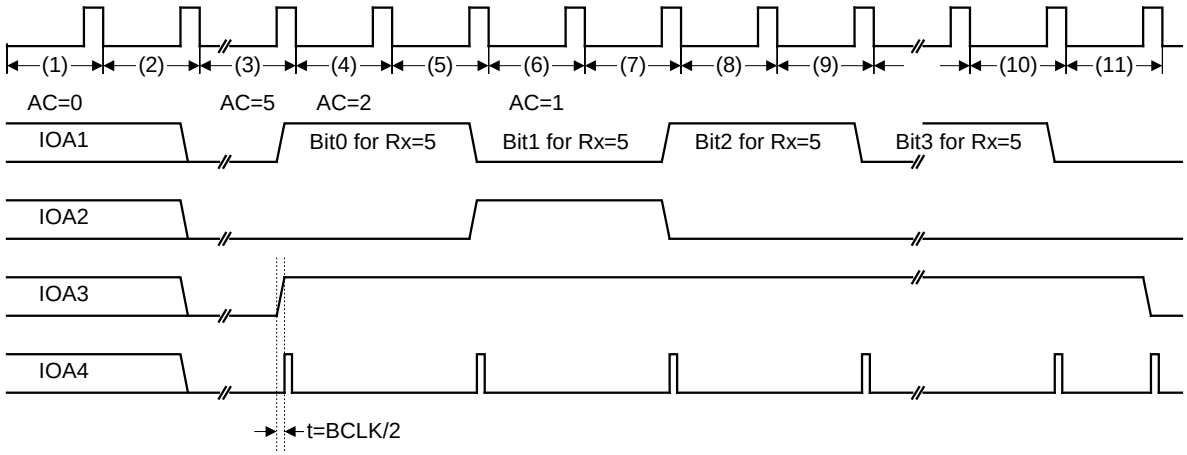
1. BIT0 and BIT1 of the port deliver RAM data.
2. BIT2 of the port delivers the constant data (D) in operand.
3. BIT3 of the port delivers a pulse.

Shown below is a sample program using the OPAS instruction to perform a serial output function.

```
(1) LDS   0AH, 0
(2) OPA   0AH    ;IOA1 is the serial output data pin,
    SPA   0FH    ;IOA3 enables the serial output function
     :           ;IOA4 is the serial output clock pin.
     :
```

```
    LDS    1,5
(3) OPAS   1,1     ;Bit 0 output, enable the serial output function
(4) SR0    1       ;shift bit 1 to bit 0
(5) OPAS   1,1     ;Bit 1 output
(6) SR0    1       ;shift bit2 to bit 0
(7) OPAS   1,1     ;Bit 2 output
(8) SR0    1       ;shift bit 3 to bit 0
(9) OPAS   1,1     ;Bit 3 output
     :
     :
(10)       OPAS   1,1     ;Output the Last bit data
(11)       OPAS   1,0     ;Inactive the serial output function
```

The above program is illustrated by the timing chart below:



If the IOA1 pin is defined as the CX pin for RFC function and the other IOA pins (IOA2 ~ IOA3) are used for normal IO pins in mask option, the IOA1 function must be set as output mode in the begining of program in order to prevent the signal change on CX pin get into the IOA1 function within input mode.

On the other hand, the IOA1 function can not change the output signal within output mode because the output signal of IOA1 function will affect the counting of RFC counter through the CX pin when the RFC counter function is enabled.

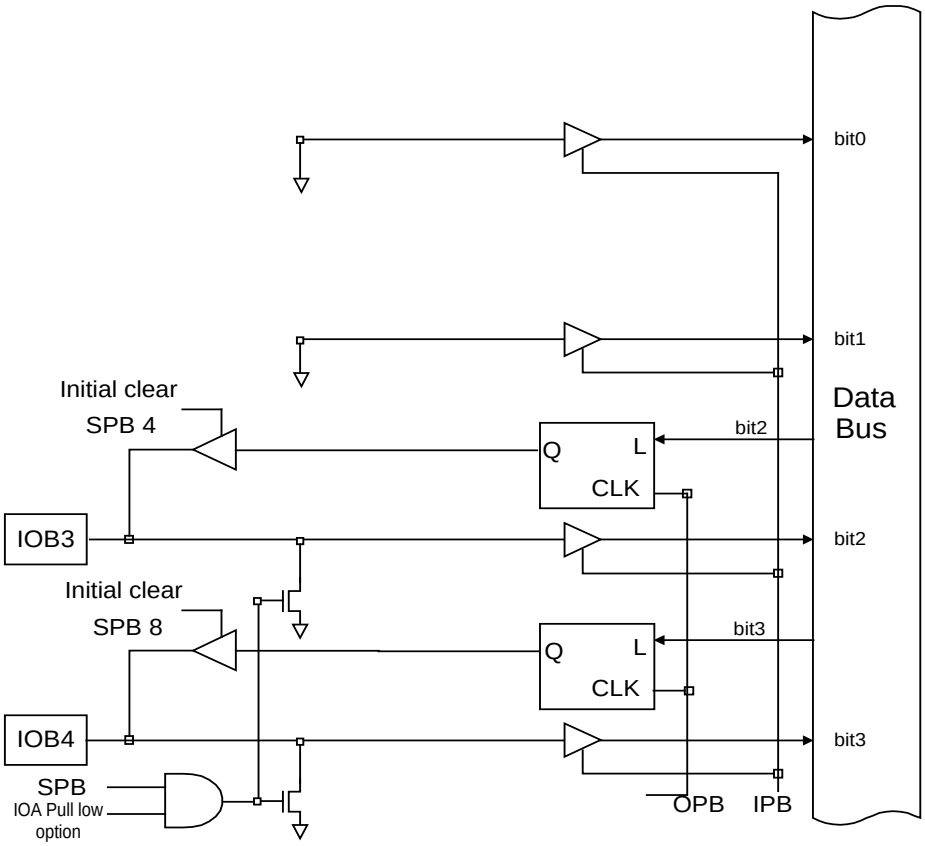
3-5-2 IOB PORT

IOB3 ~ IOB4 pins are MUXed with BZB / SEG30 and BZ pins respectively as defined in the option.

OPTION table :

Option name	Selected item
SEG30/IOB3/BZB	(2) IOB3
IOB4/BZ	(1) IOB4

The following figure shows the organization of IOB port.



Note: The pins in input mode should not be in floating, or a large current (straight-through current) will flow into the input buffer.

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.

Executing the IPB instruction can store the signals applied to the IOB pins into the specified data memory. When the IOB pins are defined as the output, executing an IPB instruction will save the data stored in the output latch into the specified data memory.

Before executing the 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 changing the I/O mode.

IOB port has a built-in pull-down resistor which can be selected in the option and can be enabled / disabled by executing a SPB instruction.

3-5-3 IOC PORT

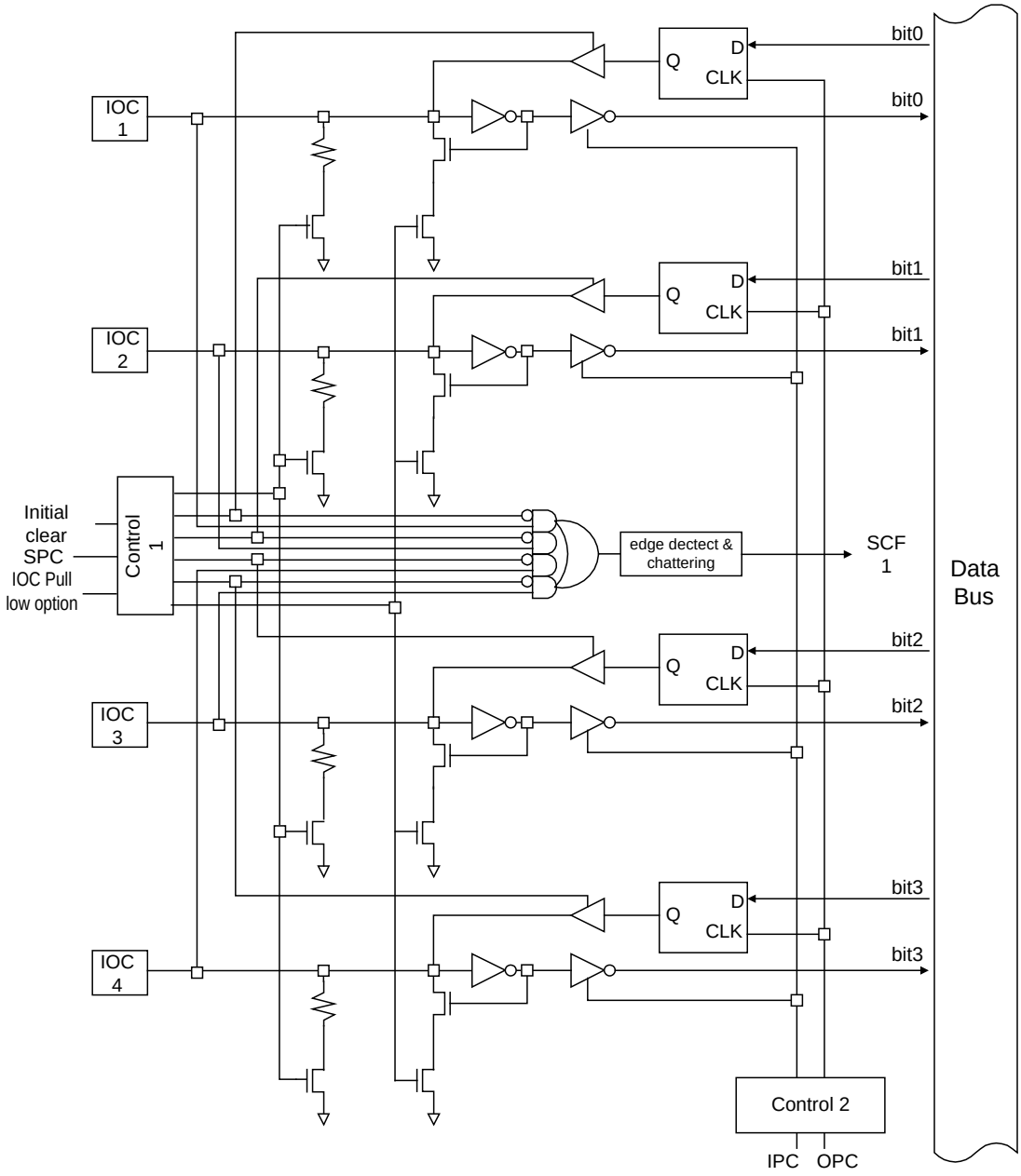
After the reset cycle, the IOC port is set as input mode and each bit of port can be defined as input mode or output mode individually by executing SPC instruction. Executed OPC instruction

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.

Executing the IPC instruction can store the signals applied to the IOC pins in the specified data memory. When the IOC pins are defined as the output, executing an IPC instruction will save the data stored in the output latch 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.

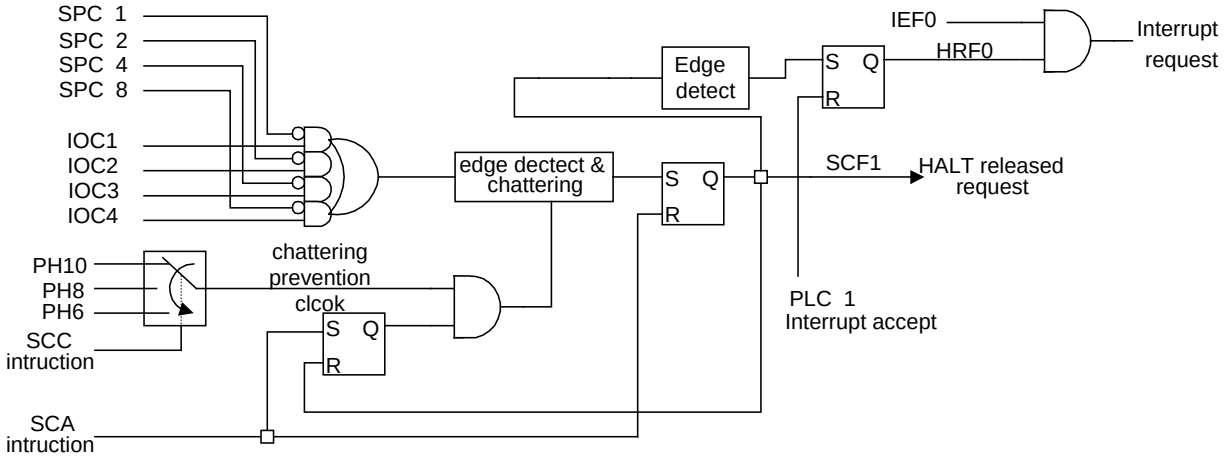


Note: The pins in input mode should not be in floating, or a large current (straight-through current) will flow into the input buffer when both the pull low device and the L-level hold device are disabled.

IOC port has a built-in pull-down resistor which can be selected in mask option and can be enabled / disabled by executing a SPC instruction,IOC port also has built-in the pull-low device for each pin.

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

The port IOC is capable of preventing the high / low chattering (bounce) of the switch signal applied on IOC1 to IOC4 pins. The chattering prevention time can be selected as PH10 (32 ms), PH8 (8 ms) or PH6 (2 ms) by executing a SCC instruction. 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 will be invoked when the signal on the applicable pin (ex. IOC1) changes 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 on the IOC port pins in input mode specified by the SCA instruction and stays in that state for at least two chattering clock (PH6, PH8, PH10) cycles, the control circuit that operates on the input pins will transmit a 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 a SCA instruction and the chattering prevention clock will be enabled at the same time. If the SCF1 has been set to 1, a halt release request flag 0 (HRF0) will be generated. In this case, if the interrupt enable mode (IEF0) of the port IOC is set, the interrupt will be accepted.

Since no flip-flop is available to hold the information of the signal on the input pins IOC1 to IOC4, the input data on the port IOC should be stored 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 in the option.

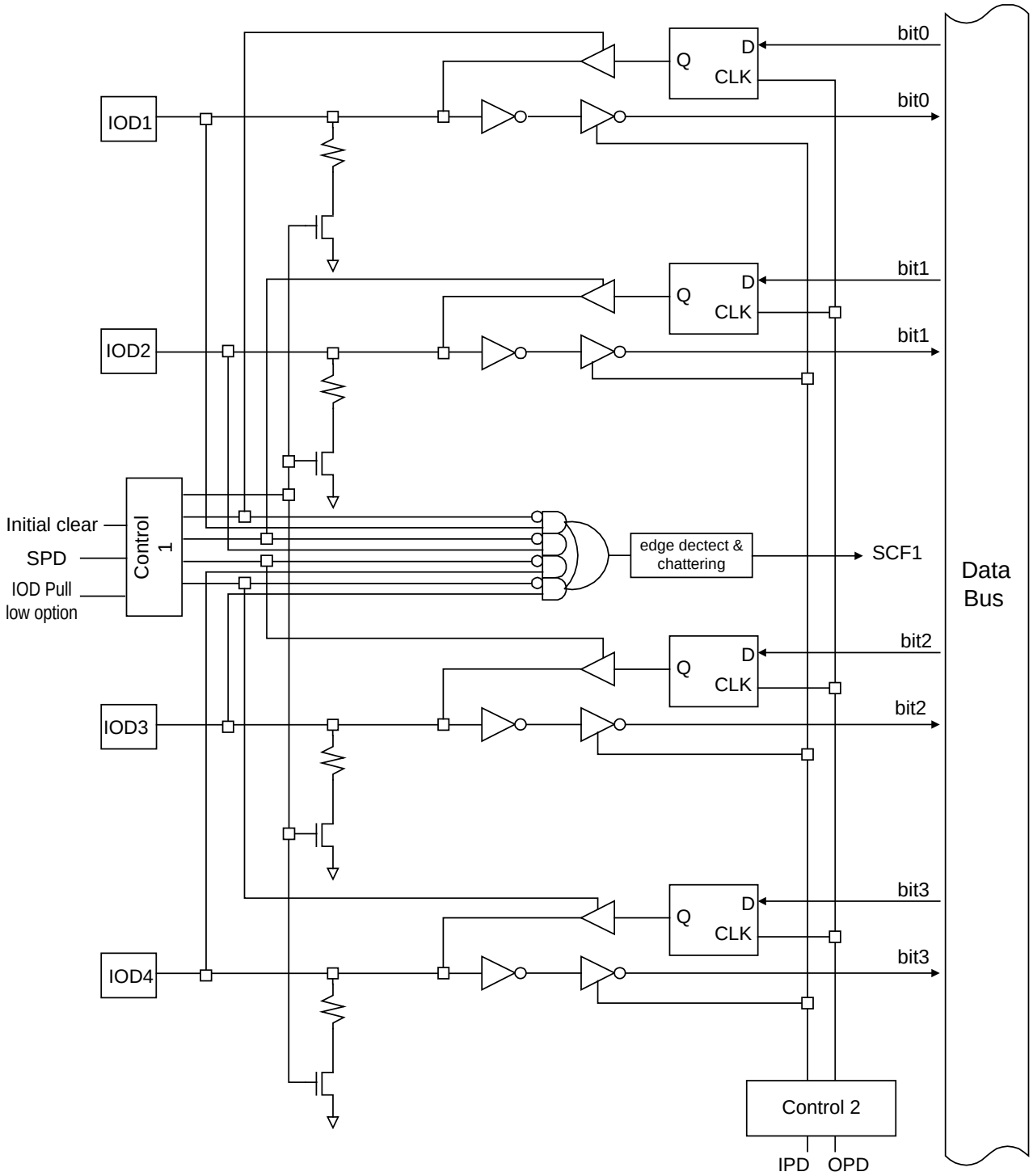
OPTION table :

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 the IPD instruction can store the signals applied to the IOD pins into the specified data memory. When the IOD pins are set as output, executing an IPD instruction will save the data stored in the output latches into the specified data memory.
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.

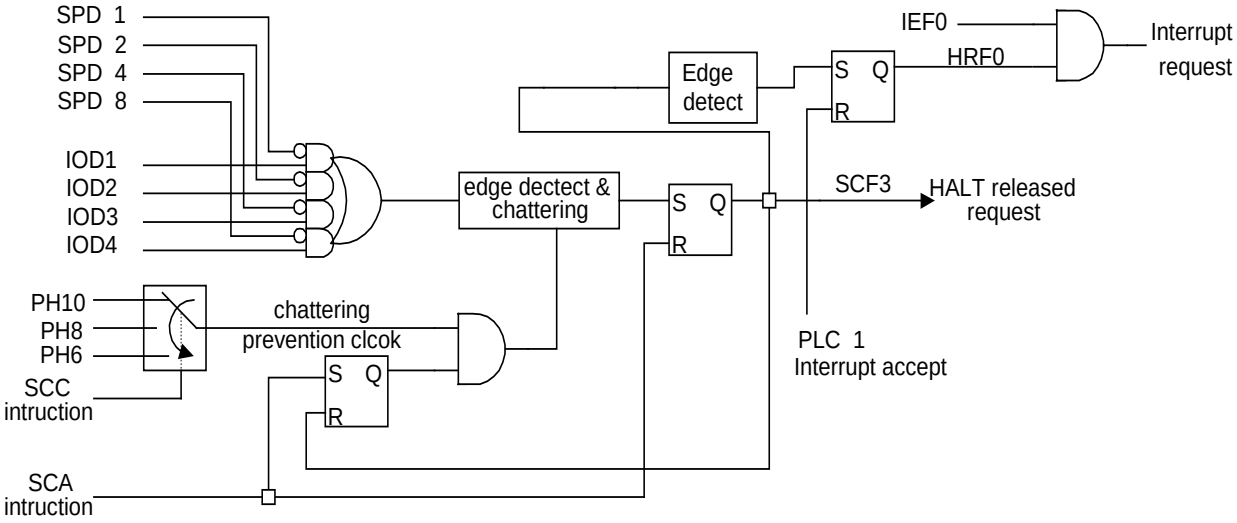
IOD port has a built-in pull-low device for each pin. To enable or disable this device, execute the SPD instruction.
When the IOD pin is set to the output mode, the pull-low device will be disabled.



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 (32 ms), PH8 (8 ms) or PH6 (2 ms) 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 will be invoked when the signal on the applicable pin (ex. IOD1) changes 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 on the IOD port pins in input mode specified by the SCA instruction occur and keep the state for at least two chattering clock (PH6, PH8, and PH10) cycles, the control circuit that operates on the input pins will transmit a 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 a SCA instruction and the chattering prevention clock will be enabled at the same time. If the SCF3 has been set to 1, a halt release request flag 0 (HRF0) will be delivered. In this case, if the interrupt enable flag (IEF0) of the port IOD is set, the interrupt will be accepted.

Since no flip-flop is available to hold the information of the signal on the input pins IOD1 to IOD4, the input data on the port IOD should be stored into the RAM immediately after the halt mode is released.

3-6. EXTERNAL INT PIN

There are 3 kinds of input type can be selected in mask option for the INT pin, pull-up, pull-down and high impendance. A signal change (either rising edge or falling edge in the option) will set the interrupt flag to deliver the halt release request flag 2 (HRF2). In this case, if the halt release enable flag (HEF2) is set, the start condition flag 2 will be set and a corresponding sigal is delivered. If the INT pin interrupt enable mode (IEF2) is set, the interrupt will be accepted.

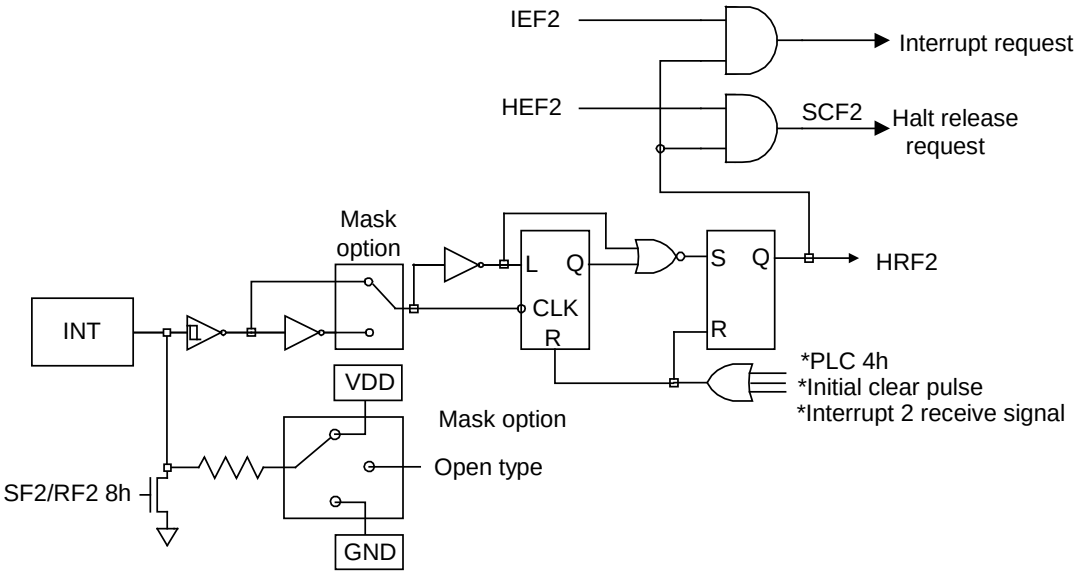
OPTION table:

For internal resistor type :

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:

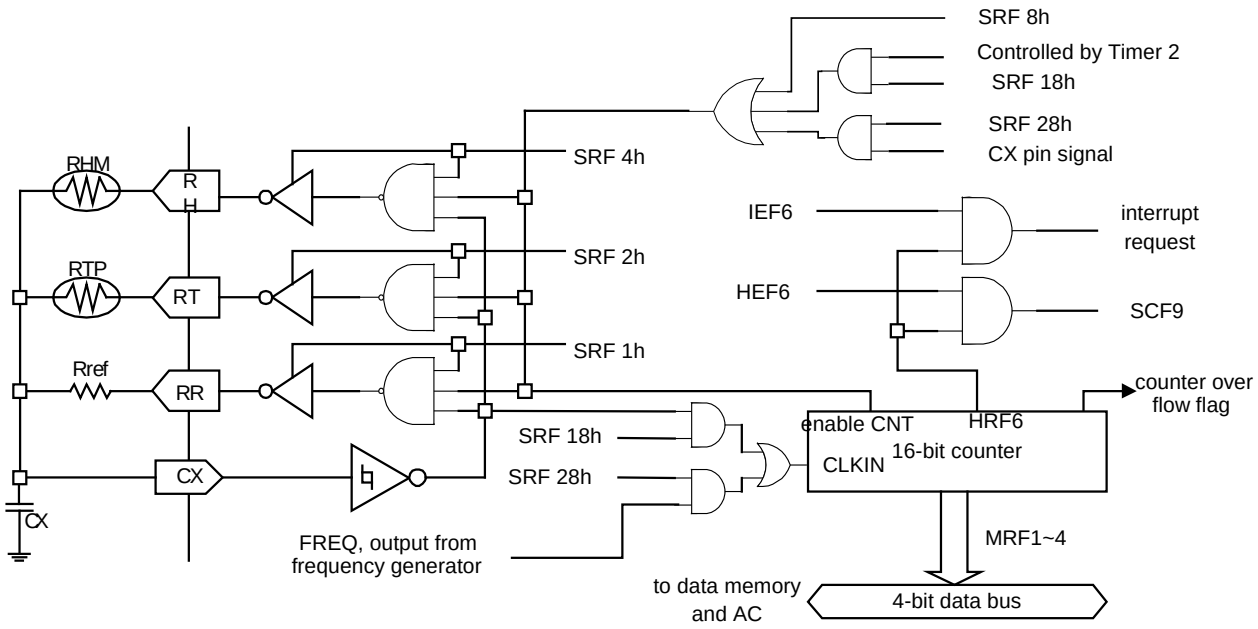
Option name	Selected item
INT PIN TRIGGER MODE	(1) RISING EDGE
INT PIN TRIGGER MODE	(2) FALLING EDGE



This figure shows the INT Pin Configuration

3-7. RESISTER TO FREQUENCY CONVERTER (RFC)

The resistor to frequency converter (RFC) converts a specified resistance to a corresponding frequency. This figure shows the block diagram of *RFC.



This RFC contains four external pins:

CX: the oscillation Schmitt trigger input pin

RR: the reference resister output pin

RT: the temperature sensor output pin

RH: the humidity sensor output pin (this pin can also be used with another temperature sensor or left floating)

These CX, RR, RT and RH pins are MUXed with IOA1 / SEG37 to IOA4 / SEG40 respectively and selected in the option.

OPTION table :

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 can build up to 3 RC oscillation networks by connecting sensors or resistors between CX and one of RR, RT or RH and CX pins. Only one RC oscillation network can be active at a time. When the oscillation network is built up (by executing SRF 1h, SRF 2h, SRF 4h instructions to enable RR, RT, RH networks respectively), a clock signal with specified frequency corresponding to the resistance will be generated and transferred to the 16-bit

counter through the CX pin as the clock source. It will then enable or disable the 16-bit counter in order to count the oscillation clock.

How to 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 (RR, RT, RH) for RC networks respectively. The inactive pins will become tri-state type output pins.
3. Execute SRF 8, SRF 18h or SRF 28h instructions to enable the RC oscillation network and the 16-bit counter. The RC oscillation network will not active until these instructions are executed. The output pin of RC oscillation network (one of the RR, RT, RH pins) is set to output 0 state before this network is activated.

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

In this mode, the clock input of the 16-bit counter received from the CX pin and the counter is enabled/disabled by the S/W. When the SRF 8h instruction is executed, the counter will be enabled and started to count the clocks from the CX pin. The counter will be disabled when the SRF 0 instruction is executed. Executing MRF1 ~ 4 instructions will load the content of the 16-bit 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.

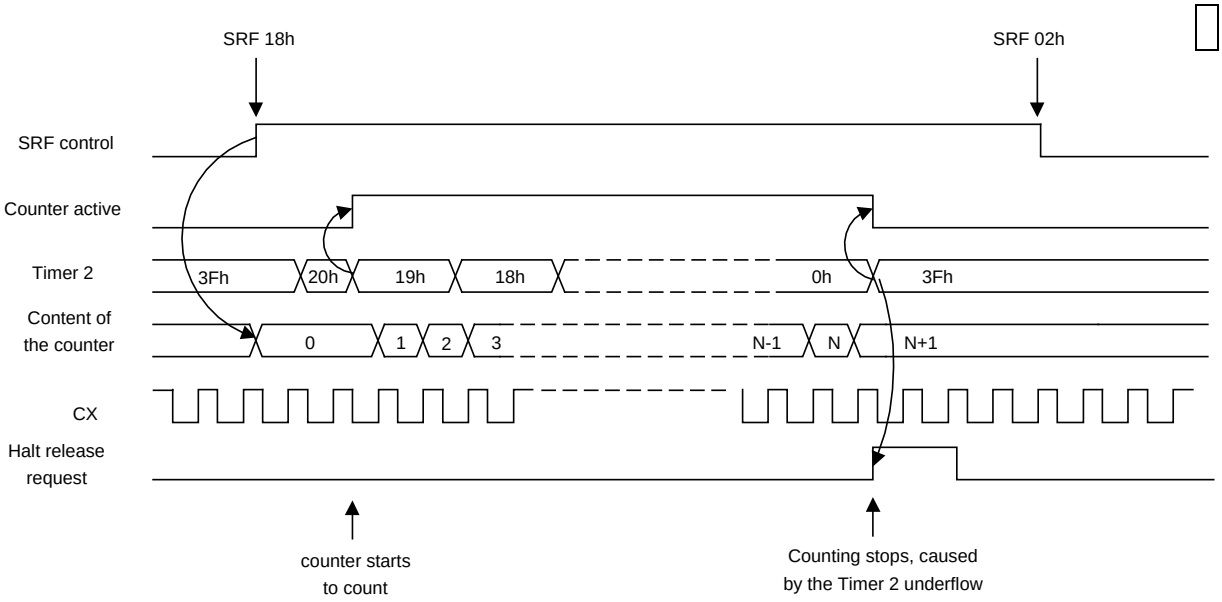
Example:
If you intend to count the number of clock input from the CX pin for a time period, you can enable the counter by executing a SRF 8 instruction and setting the timer1 to control the time period. The overflow flag (RFOVF) of this counter will be checked during the time period. If the overflow flag is not set to 1, read the content of the counter directly; if the overflow flag has been set to 1, the program is required to reduce the time period and repeat the previous procedure again. In the following example, the RR network generates the clock source on CX pin.

```
;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                   ;decrement 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, the RT network is used 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 signals for TM2 and the
;count down 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
```

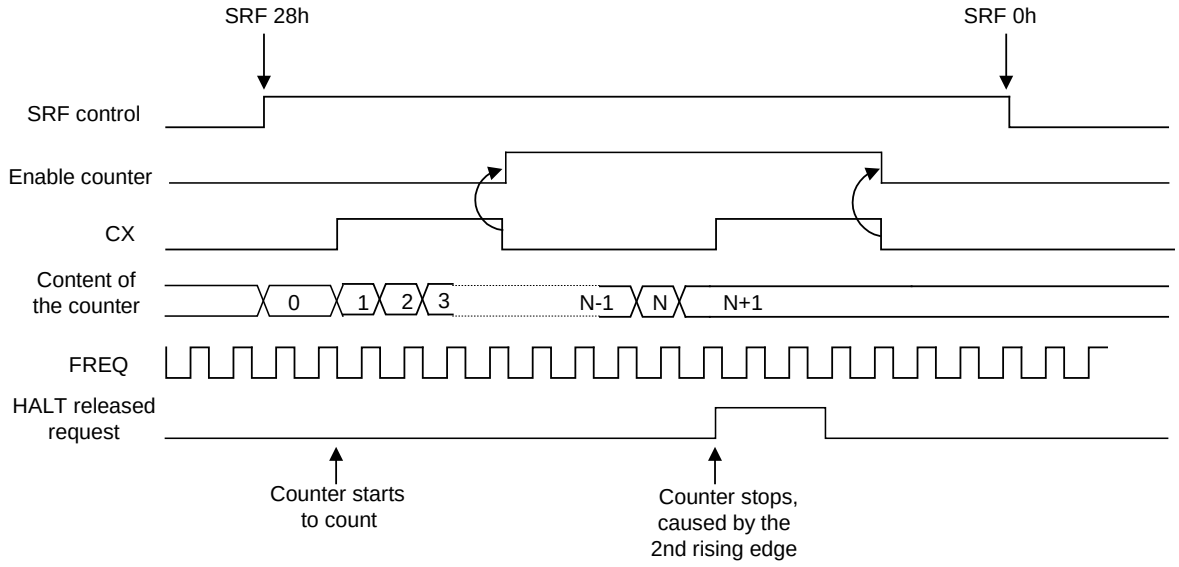
3-7-4 Enable / Disable the Counter by CX Signal

This is another usage for the 16-bit counter but doesn't relate to the RFC function. In applications described in the previous section, CX is used as the clock source for the 16-bit counter and using the S/W or TMR2 to produce a time period in order to control the 16-bit counter.

In this mode, however, the 16-bit counter operates differently, the clock signal on CX pin turns into the controlled signal to enable / disable the 16-bit counter and the clock source of the 16-bit counter received from the output of the frequency generator (FREQ).

When the 16-bit counter is enabled, it counts the clock (FREQ) after the first rising edge signal applies to the CX pin. Once the second rising edge applies to the CX pin, a halt release request (HRF6) will be delivered and the 16-bit counter stops counting. In this case, if the interrupt enable flag 6 (IEF6) is set, the interrupt will be accepted; and if the halt release enable flag 6 (HEF6) is set, the halt release request signal will be delivered to set 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
                        ;is 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
```

CHAPTER 4 LCD DRIVER OUTPUT

TM8530 provides 36 segment pins and 4 common output pins to drive LCD.

OPTION table :

During the initial reset cycle, the LCD patterns can be selected all “ON” or all “OFF” in the option. All the LCD patterns or DC output will keep in the initial setting state until the LCD related instructions are executed to change the output data.

OPTION table :

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

4-1. LCD LIGHTING SYSTEM IN TM8530

There are two settings for the LCD lighting systems that can be selected in the option in TM8530, they are:

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

All of these lighting systems are combined with 2 kinds of mask options; one is “BIAS” and the other is “LCD Driving Capability”.

LCD bias option

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

LCD Driving Capability (LCD voltage divider resistor)

Option name	Selected item	Remark
LCD Driving Capability	(1) Low Driving	*6 uA
	(2) Normal Driving	*12 uA
	(3) High Driving	*24 uA
	(4) Higher Driving	*60 uA

*Typical power consumption of pure LCD voltage divider resistor (@ VDD= 3.0V, Ta= 25℃)

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

The LCD frame frequency in 1/4 duty type

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

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

LCD Lighting System	The Maximum Number of Driving LCD Segments
1/2 bias 1/4 duty	144
1/3 bias 1/4 duty	144

It is recommended to choose the frame frequency higher than 24 Hz. If the frame frequency is lower than 24 Hz, the pattern on the LCD panel will start to flicker.

4-2. SEGMENT PLA CIRCUIT FOR LCD DISPLAY

4-2-1. PRINCIPLE OF OPERATION OF LCD DRIVER SECTION

The figure below illustrates how the LCD driver module operates when the LCD-related instructions are executed.

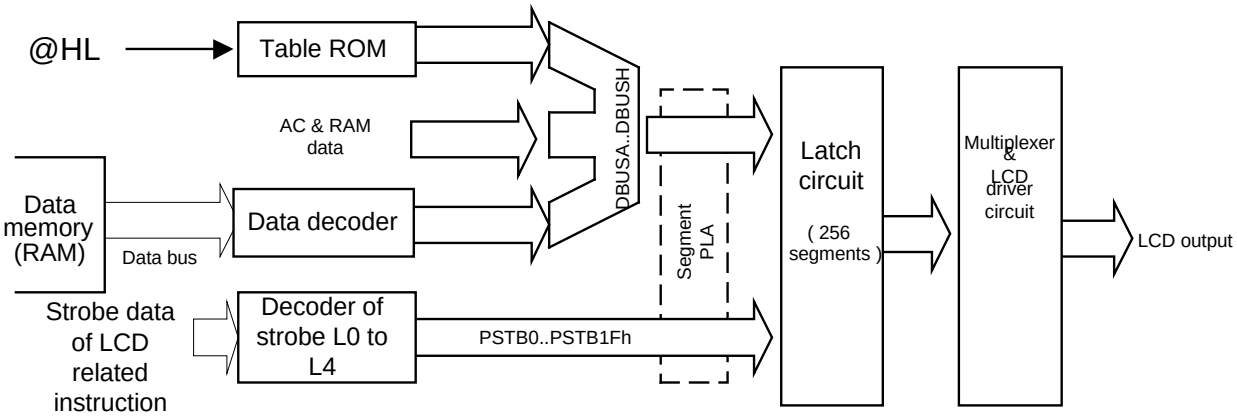


Figure 5-3 Principal Drawing of LCD Driver Section

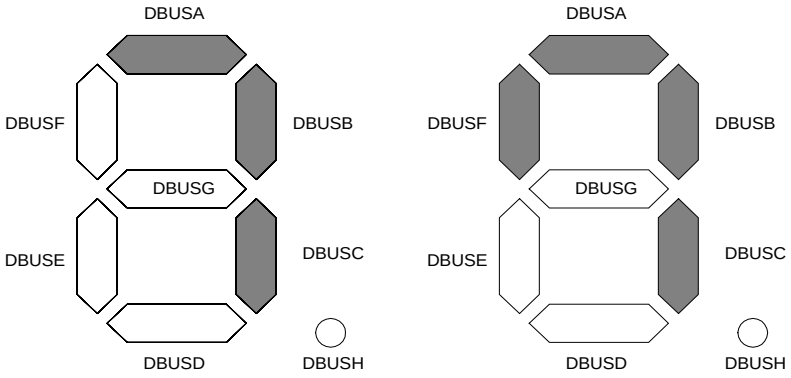
The LCD driver module consists of the following units:

- Data decoder to decode data received from RAM or table ROM
- Latch circuit to store LCD lighting information
- L0 to L4 decoder: decode the Lz data specified in the 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 converts the contents of the working registers specified in LCD-related instructions into the data format of 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 data decoder output, DBUSF, can be selected as 0 or 1 in the option. For example, the muliti-pattern option of digit “7" can be displayed in two shapes as below:



DBUSF=0 DBUSF=1

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

OPTION table :

Option name	Selected item
F SEGMENT FOR DISPLAY “7”	(1) ON
F SEGMENT FOR DISPLAY “7”	(2) OFF

Both the 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 output of DBUSA ~ DBUSH will be all “0” (this is used for blanking the leading digit “0” on the LCD panel).
The LCP instruction transfers the content of the RAM (Rx) and accumulator (AC) to “DBUSA” ~ “DBUSH” directly bypassing the data decoder.
The LCD instruction transfers the table ROM data (T@HL) to “DBUSA” ~ “DBUSH” directly bypassing the data decoder.

Table 4- 2 The bit 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 in the option.

Among the 512 signals obtainable by combining “DBUSA” to “DBUSH” with the address 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 4- 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 can turn off the displays simultaneously. The execution of the RF2 4h instruction can 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 output to the LCD while the display is turned on again.

In the stop mode, all COM and SEG outputs of LCD driver will automatically switch to the GND state to eliminate the DC bias on the LCD panel.

4-2-2. LCD-related 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

8. SF2 4h

Turns off the LCD display.

9. RF2 4h

Turns on the LCD display.

4-2-3. EXPLANATION

Each LCD driver output corresponds to the LCD 1/4 duty panel and has 4 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) can be selected using the segment PLA, to combine segments in the LCD driver outputs is quite flexible. 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.
TM8530 provides 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 output to the 16 latches (Lz) specified by D simultaneously.

D	Specified range of latched
00	Lz = 00h ~ 0Fh
01	Lz = 10h ~ 11h

Refer to Chapter 5 for detailed description of these instructions.

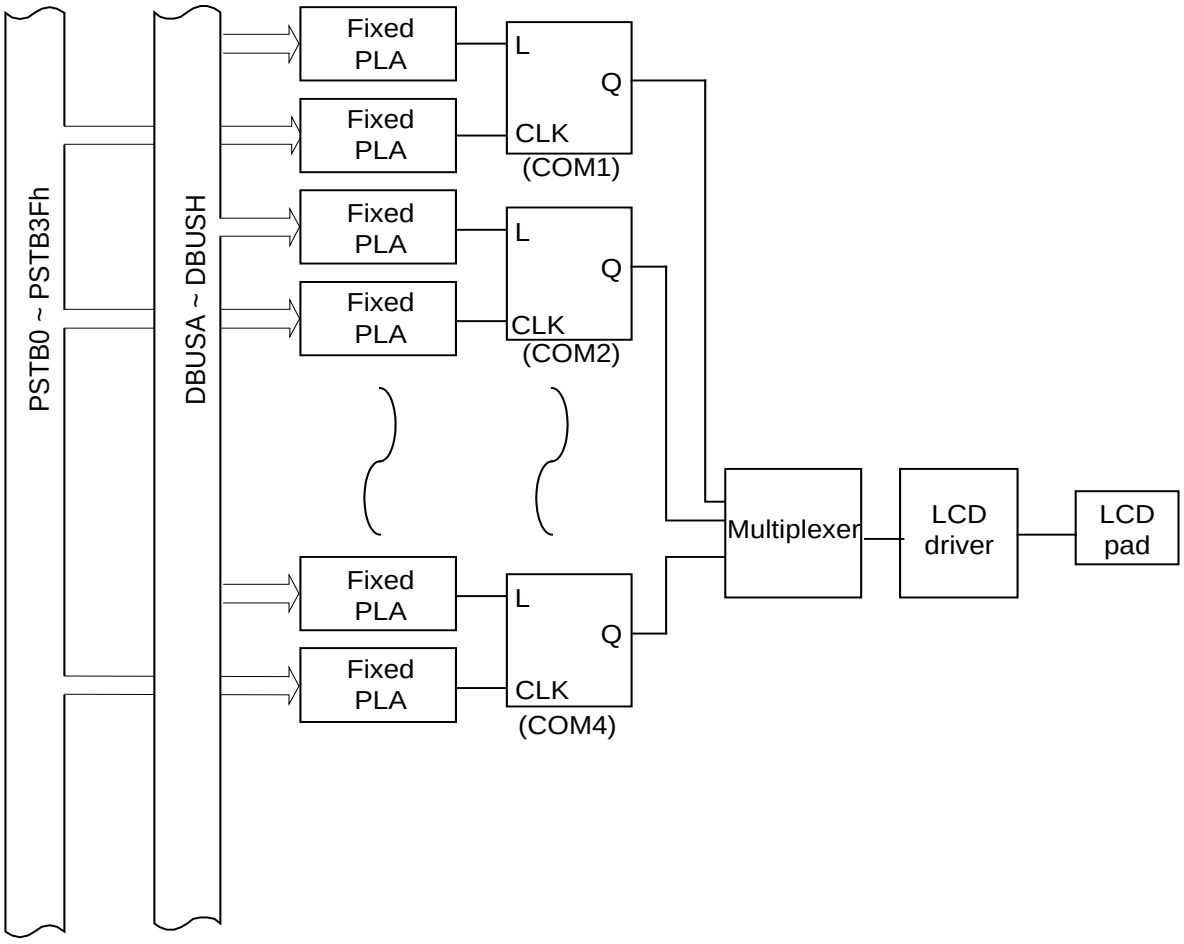


Figure: 4-3 Sample Organization of Segment PLA Option

4-2-4. THE CONFIGURATION FORMAT FOR FIXED PLA MAPPING

The TM8530 is a flexible PLA structure

PLA table is shown below:

SEGn	Lz	COM1	COM2	COM3	COM4
SEG1	00H	DBUSA	DBUSB	DBUSC	DBUSD
SEG2		DBUSE	DBUSF	DBUSG	DBUSH
SEG3	01H	DBUSA	DBUSB	DBUSC	DBUSD
SEG4		DBUSE	DBUSF	DBUSG	DBUSH
SEG5	02H	DBUSA	DBUSB	DBUSC	DBUSD
SEG6		DBUSE	DBUSF	DBUSG	DBUSH
SEG7	03H	DBUSA	DBUSB	DBUSC	DBUSD
SEG8		DBUSE	DBUSF	DBUSG	DBUSH
SEG9	04H	DBUSA	DBUSB	DBUSC	DBUSD
SEG10		DBUSE	DBUSF	DBUSG	DBUSH
SEG11	05H	DBUSA	DBUSB	DBUSC	DBUSD
SEG12		DBUSE	DBUSF	DBUSG	DBUSH
SEG13	06H	DBUSA	DBUSB	DBUSC	DBUSD
SEG14		DBUSE	DBUSF	DBUSG	DBUSH
SEG15	07H	DBUSA	DBUSB	DBUSC	DBUSD
SEG16		DBUSE	DBUSF	DBUSG	DBUSH
SEG17	08H	DBUSA	DBUSB	DBUSC	DBUSD
SEG18		DBUSE	DBUSF	DBUSG	DBUSH
SEG19	09H	DBUSA	DBUSB	DBUSC	DBUSD
SEG20		DBUSE	DBUSF	DBUSG	DBUSH
SEG21	0AH	DBUSA	DBUSB	DBUSC	DBUSD
SEG22		DBUSE	DBUSF	DBUSG	DBUSH
SEG23	0BH	DBUSA	DBUSB	DBUSC	DBUSD
SEG24		DBUSE	DBUSF	DBUSG	DBUSH
SEG25	0CH	DBUSA	DBUSB	DBUSC	DBUSD
SEG26		DBUSE	DBUSF	DBUSG	DBUSH
SEG27	0DH	DBUSA	DBUSB	DBUSC	DBUSD
SEG28		DBUSE	DBUSF	DBUSG	DBUSH
SEG29	0EH	DBUSA	DBUSB	DBUSC	DBUSD
SEG30		DBUSE	DBUSF	DBUSG	DBUSH
SEG36	0FH	DBUSA	DBUSB	DBUSC	DBUSD
SEG37		DBUSE	DBUSF	DBUSG	DBUSH
SEG38	10H	DBUSA	DBUSB	DBUSC	DBUSD
SEG39		DBUSE	DBUSF	DBUSG	DBUSH
SEG40	11H	DBUSA	DBUSB	DBUSC	DBUSD
SEG41		DBUSE	DBUSF	DBUSG	DBUSH

CHAPTER 5 DETAIL EXPLANATION OF TM8530 INSTRUCTIONS

- Before using the data memory, it is necessary to initialize the content of data memory because the initial values of them are 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 0FH (TM87P04) or 1FH (TM8530).

5-1. INPUT / OUTPUT INSTRUCTIONS

LCT Lz, Ry
Function: LCD latch Lz ← data decoder ← (Ry)
Description: The content of 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 content of 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 content of 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 content 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 content 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 content 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 content 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 output to I/OA port.

OPAS Rx, D

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

Description: Content of Rx is output to IOA port. D is output to IOA3, pulse is output 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(X4 X3 X2) 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

OPB Rx

Function: I/OB ← (Rx)

Description: The contents of Rx are output 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 output 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 output 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.

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 can 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(4 KHz)
0	1	0	ϕ4(2 KHz)
0	0	1	ϕ5(1 KHz)
0	0	0	DC0

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

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

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 () is bases on the PH0 is 32768 Hz.

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 all the bits that after the 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: Substract 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: Substract 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: Substract 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 ← [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 ← [@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 ← [@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.
. @HL indicates an index address of data memory.
* The carry flag (CF) will be affected.

ADD* Rx

Function: AC, [Rx] ← [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] ← [@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] ← [@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.
. @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 ← [Rx] ⊕ 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] ← [@HL] ⊕ 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] ← [@HL] ⊕ AC, @HL ← 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 ← [Rx] | AC

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

OR @HL

Function: AC ← [@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 ← [@HL] | AC, @HL ← 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 ← [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] ← [@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 the 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 the stop mode and stops all oscillators
Description: Before executing this instruction, all signals on IOC port should be set to low.
The following 3 conditions cause the stop mode to be released.
1)One of the signals on KI1~4 is “H”/”L”(LED/LCD) in scanning interval.
2) A signal change on 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 on 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 the interrupt enable flag
Description:

X0=1	The IEF0 is set so that interrupt 0 (Signal change on port IOC specified by SCA) is accepted.
X1=1	The IEF1 is set so that interrupt 1 (underflow from timer 1) can be accepted.
X2=1	The IEF2 is set so that interrupt 2 (the signal change at the INT pin) can be accepted.
X3=1	The IEF3 is set so that interrupt 3 (overflow from the predivider) can be accepted.
X4=1	The IEF4 is set so that interrupt 4 (underflow from timer 2) can be accepted.
X6=1	The IEF6 is set so that interrupt 6 (overflow from the RFC counter) can be 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 can be released by TMR1 underflow.
X2=1	The HEF2 is set so that the halt mode can be released by signal changed on INT pin.
X3=1	The HEF3 is set so that the halt mode can be released by predivider overflow.
X4=1	The HEF4 is set so that the halt mode can be released by TMR2 underflow.
X6=1	The HEF6 is set so that the halt mode can be released by RFC counter overflow.

X7 is reserved;

SRE X

Function: Set/Reset stop release enable flag

Description:

X3=1	The SRF3 is set so that the stop mode is released by the signal changed on IOD port.
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.

X7,6,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 content of the SCF1, SCF2 and BCF flags are loaded to AC and the data memory specified by Rx.
The content of AC and the meaning of all the bits that after the 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 TM8530

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 all the bits that after the 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 all the bits that after the 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)	NA
Halt release caused by RFC counter overflow	NA	Halt release caused by TM2 underflow	NA

MSD Rx

Function: Rx, AC ← WDF,CSF,RFOVF

Description: The watchdog flag, system clock status and 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 all the bits after the 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 an add 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 an add 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 an add 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 a subtract 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 a subtract 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 a subtract 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 a subtract 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 bit0 of AC is 0, the PC will be incremented 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 bit1 of AC is 0, the PC will be incremented 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 the bit2 of AC is 1, jump occurs.
If the bit2 of AC is 0, the PC will be incremented 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 bit3 of AC is 0, the PC will be incremented 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 the content of AC is 0, the PC will be incremented 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 the content of CF is 1, the PC will be incremented 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 the content of AC is 1, the PC will be incremented by 1.
The range of X is from 000H to 7FFH or 800H to BFFH.

JC X

Function: Program counter jumps to X if CF=1.
Description: If the content of CF is 1, jump occurs.
If the content of CF is 0, the PC will be incremented by 1.
The range of X is from 000H to 7FFH or 800H to BFFH.

JMP X

Function: Program counter jumps to X.
Description: Unconditional jump.
The range of X is from 000H to BFFH.

CALL X

Function: STACK ← (PC)+1
Program counter jumps to X.
Description: A subroutine is called.
The range of X is from 000H to BFFH.

RTS

Function: PC ← (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 PH0. Refer to section 3-3-4 for PH0.

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 the 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 the initial value. The following table shows the preset data and the duty cycle settings:

	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
   Use the contents of Rx and AC as preset data N.
2. FRQ D, @HL
   Use the contents of table TROM specified by index address buffer as preset data N.
3. FRQX D, X
   Use 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 X700X4 00X1X0
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 the strong pull-low device on INT pin
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 flags
Description: Description of each flag
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.
X7~4 is reserved

PLC

Function: Pulse control
Description: The pulse corresponding to the data specified by X is generated.
X0 : "1" the Halt release request flag HRF0 caused by the signal at I/O port C is reset.
X1 : "1" the Halt release request flag HRF1 caused by underflow from the timer 1 is reset, and stops the operating of timer 1(TM1).
X2 : "1" the Halt or stop release request flag HRF2 caused by the signal change at the INT pin is reset.
X3 : "1" the Halt release request flag HRF3 caused by overflow from the predivider is reset.
X4 : "1" the Halt release request flag HRF4 caused by underflow from the timer 2 is reset and stops the operating of timer 2(TM2).
*X5 : "1" the Halt release request flag HRF5 caused by the signal change to ”L” on KI1~4 d scanning interval is reset.
*X6 : "1" the 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.

ORDERING INFORMATION

The ordering information:

Ordering number	Package
TM8530-COD	Wafer / Dice with code

Appendix A TM8530 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	

Instruction		Machine Code	Function		Flag/Remark
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	

Instruction		Machine Code	Function		Flag/Remark
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	

Instruction		Machine Code	Function		Flag/Remark
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	← @HL + 1 ←HL+1	CF
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 : (Unused) B0 : (Unused)

Instruction		Machine Code	Function		Flag/Remark
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 : (Unused)
MSD	Rx	0100 1110 0XXX XXXX	AC,Rx	← STS4	B3 : (Unused) 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) ←HL+1	CF
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	

Instruction		Machine Code	Function		Flag/Remark
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

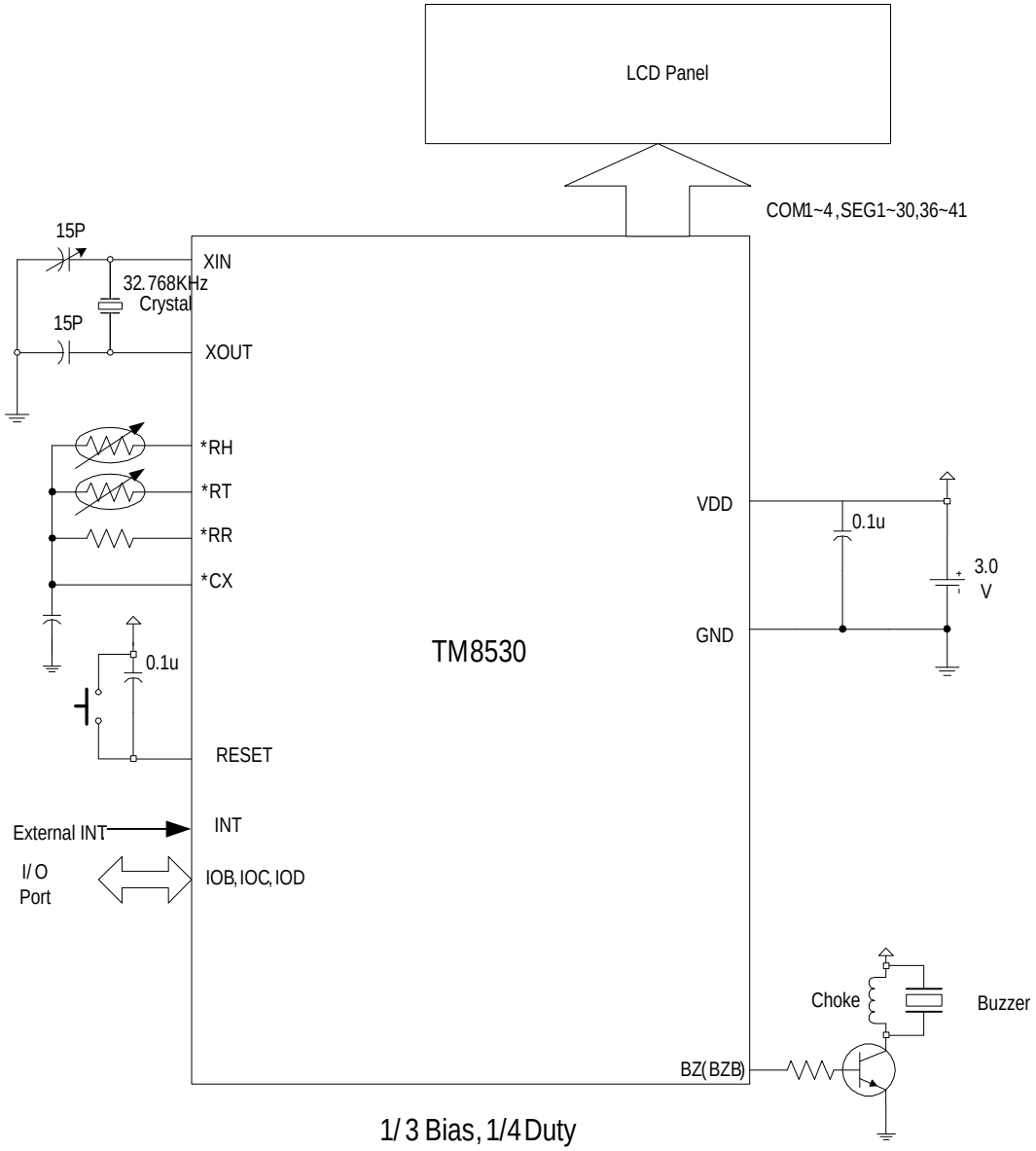
Instruction		Machine Code	Function		Flag/Remark
JC	X	1011 1XXX XXXX XXXX	PC	← X	if CF = 1
CALL	X	1100 PXXX XXXX XXXX	STACK PC	← PC + 1 ← X	
JMP	X	1101 PXXX XXXX XXXX	PC	← X	
TMS	Rx	1110 0000 0XXX XXXX	AC3,2 = 11 AC3,2 = 10 AC3,2 = 01 AC3,2 = 00 AC1,0,PB3~0	: Ctm = FREQ : Ctm = PH15 : Ctm = PH3 : Ctm = 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 0X0X XXX0	X6 X4 X3 X2 X1	: Enable HEF6 : Enable HEF4 : Enable HEF3 : Enable HEF2 : Enable HEF1	RFC TMR2 PDV INT TMR1

Instruction		Machine Code	Function		Flag/Remark
SIE*	X	1110 1001 0X0X XXXX	X6 X4 X3 X2 X1 X0	: Enable IEF6 : Enable IEF4 : Enable IEF3 : Enable IEF2 : Enable IEF1 : Enable IEF0	RFC 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 00XX X000	X5 X4 X3	: 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	
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 XX00	X4 X3,2	: Set B4-1 Pull-Low : Set B4-3 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	

Instruction		Machine Code	Function		Flag/Remark
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 X5~0	: FREQ : DC1 : PH3 : PH4 : PH5 : DC0 ← 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		

Appendix B TYPICAL APPLICATION CIRCUIT

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



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		