十速

TM8721

with LCD Driver

User Manual

Rev 1.2

AMENDMENT HISTORY

Version	Date	Description
V1.0	May, 2005	New release
V1.1	Dec, 2011	Add Ordering Information table
V1.2	Nov, 2016	P.9~10: Modify Segment1~9, IOB,IOC, RR,RT, Voh1c/Vol1c/Voh12f/Voh12g/Voh12m Driver/sink Value

CONTENTS

1. General Description

1.1 GENERAL DESCRIPTION

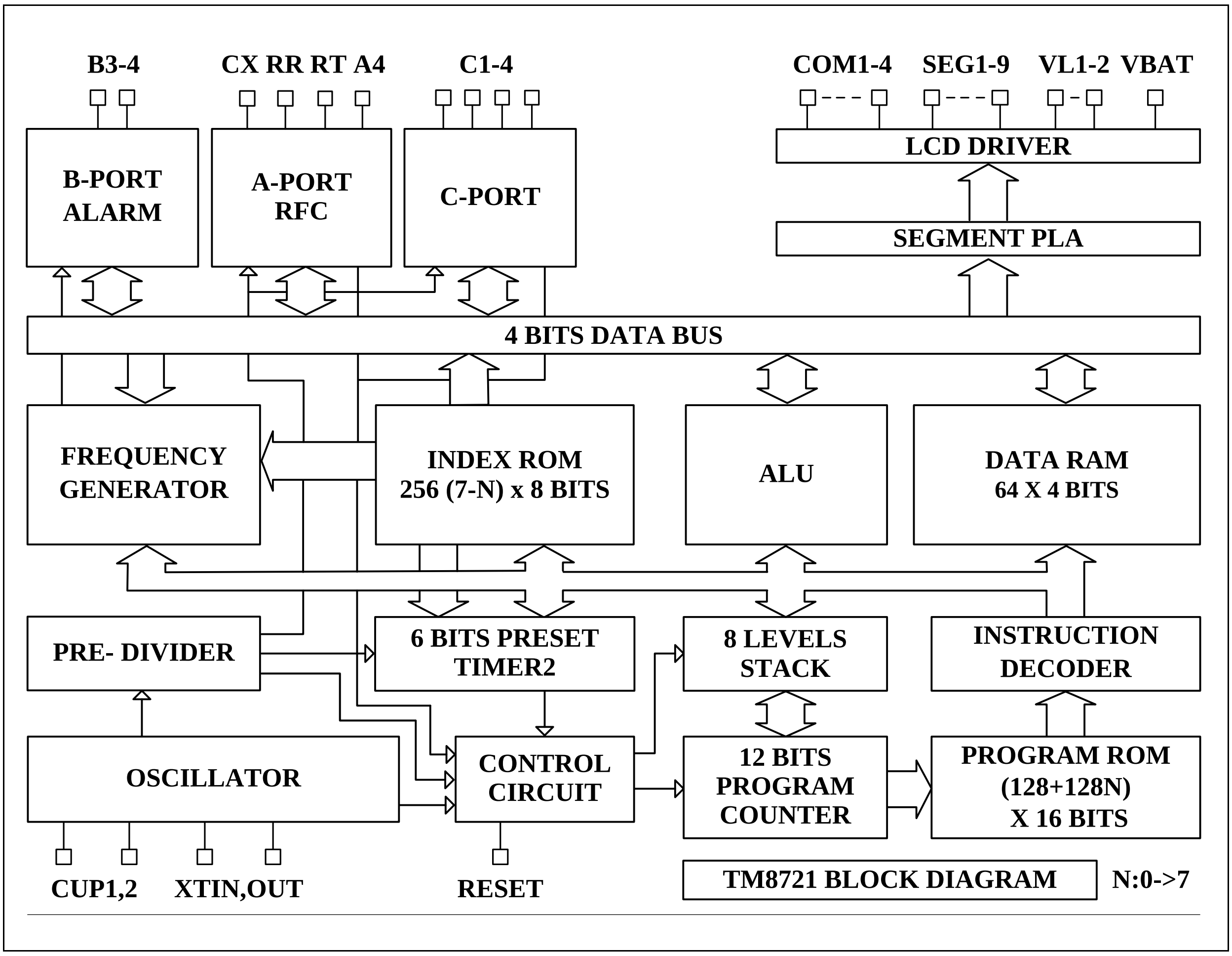
The TM8721 is an embedded high-performance 4-bit microcomputer with LCD driver. It contains all the of the following functions in a single chip: 4-bit parallel processing ALU, ROM, RAM, I/O ports, timer, clock generator, dual clock operation, Resistance to Frequency Converter (RFC), LCD driver, look-up table.

1.2 FEATURES

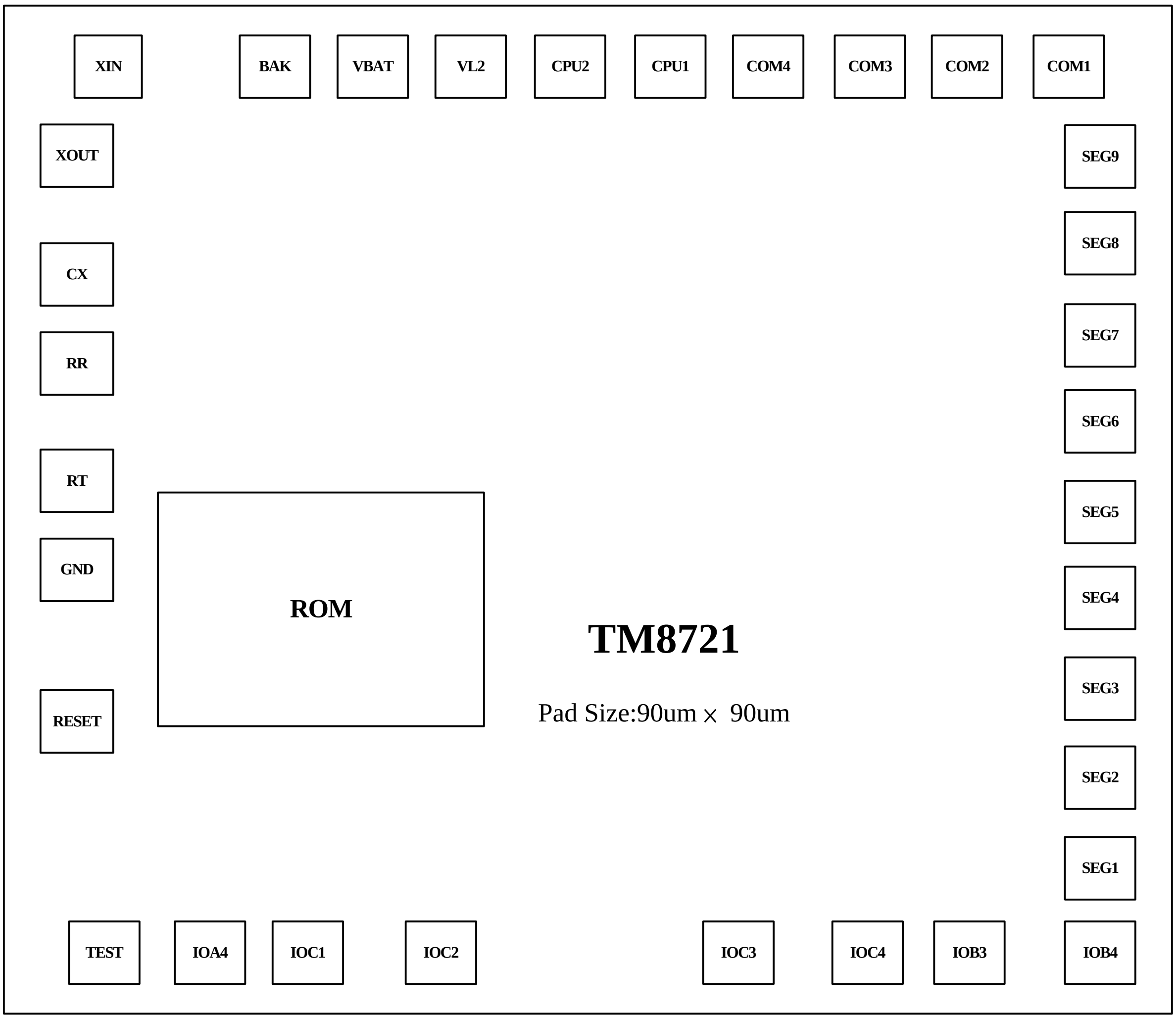
1. **Powerful instruction set (137 instructions)**
   - Binary addition, subtraction, BCD adjustment, logical operation in direct and index addressing mode.
   - Single-bit manipulation (set, reset, decision for branch).
   - Various conditional branches.
   - 16 working registers and manipulation.
   - Look-up table.
   - LCD driver data transfer.
2. **Memory capacity**
   - ROM capacity 1024 x 16 bits.
   - RAM capacity 64 x 4 bits.
3. **Input/output ports**
   - Port IOA4 1 pin (with internal pull-low).
   - IOA4 port has built-in input signal chattering prevention circuitry.
   - Port IOC 4 pins (with internal pull-low, low-level-hold).
   - Port IOB3, 4 2 pins (with internal pull-low), & mask option with BZB, BZ.
4. **8-level subroutine nesting.**
5. **Interrupt function**
   - External factor 1 (Pin IOA4 port).
   - Internal factors 3 (Pre-Divider, Timer2 & RFC).
6. **Built-in Alarm, clock or single tone melody generator (BZB, BZ), & mask option with IOB3, 4.**
7. **Built-in R to F Converter circuit**
   - CX, RR, RT.
8. **One 6-bit programmable timer (Timer 2) with programmable clock source.**

9. **LCD driver output**
   - 9 LCD driver outputs (Up to drive 36 LCD segments) .
   - 1/4 Duty and 1/2 Bias for LCD display.
   - Single instruction to turn off all segments.
   - 9 DC/Open Drain outputs for LED mask option.
   - 16 LCD Address.
10. **Built-in Voltage double charge pump circuit.**
11. **Clock oscillation can be defined as X'tal, external-R or internal-R 2 type oscillators by mask option.**
12. **HALT function.**
13. **STOP function.**

1.3 BLOCK DIAGRAM



1.4 PAD DIAGRAM



The substrate of the chip should be connected to the GND.

1.5 PAD COORDINATE

No	Name	X	Y	No	Name	X	Y
1	VL2	563.50	1267.50	18	IOB4/BZ	1317.40	87.25
2	VBAT	438.50	1267.50	19	SEG1	1327.50	222.50
3	BAK (VL1)	323.50	1267.50	20	SEG2	1327.50	337.50
4	XIN	120.40	1267.50	21	SEG3	1327.50	452.50
5	XOUT	87.50	1119.20	22	SEG4	1327.50	567.50
6	CX	87.50	979.00	23	SEG5	1327.50	682.50
7	RR	87.50	864.00	24	SEG8	1327.50	797.50
8	RT	87.50	700.50	25	SEG7	1327.50	912.50
9	GND	87.50	585.50	26	SEG8	1327.50	1027.50
10	RESET	87.50	330.05	27	SEG9	1327.50	1142.50
11	TEST	115.00	87.25	28	COM1	1253.50	1267.50
12	IOA4	260.00	87.25	29	COM2	1138.50	1267.50
13	IOC1	375.00	87.25	30	COM3	1023.50	1267.50
14	IOC2	530.90	87.25	31	COM4	908.50	1267.50
15	IOC3	890.60	87.25	32	CUP1	793.50	1267.50
16	IOC4	1046.50	87.25	33	CUP2	678.50	1267.50
17	IOB3/BZB	1161.50	87.25				

1.6 PIN DESCRIPTION

Name	I/O	Description
VBAT	P	Positive power supply. Connect a 0.1uF capacitor to GND.
BAK (VL1)	P	Internal logic, RFC & LCD mode level1 supply voltage Connected to VBAT.
VL2	P	LCD mode level2 supply voltage. Connect a 0.1uF capacitor to GND for LCD mode. Short to VBAT for O/P Mode
RESET	I	Input pin for chip reset request signal, with internal pull-down resistor.
TEST	I	Test signal input pin.
CUP1,2	O	Switching pins for supply the LCD driving voltage. Connect the CUP1 and CUP2 pins with a 0.1uf non-polarized electrolytic capacitor for LCD mode.
COM1~4	O	Output pins for driving the common pins of the LCD panel.
SEG1~9	O	Output pins for driving the LCD panel segment.
IOA4	I/O	I/O port pin.
IOB3,4	I/O	I/O port pins, & mask option with BZB,BZ
IOC1~4	I/O	I/O port pins
CX RR, RT	I O	1 input pin and 2 output pins for RFC application.
BZB, BZ	O	Output port for alarm, frequency or melody generator
XIN XOUT	I O	System clock oscillation. Connected with 32KHz crystal oscillator or internal R or external R by mask option.
GND	P	Negative supply voltage.

1.7 CHARACTERIZATION

ABSOLUTE MAXIMUM RATINGS

At Ta= –20°C to 70°C, GND=0V

Name	Symbol	Range	Unit
Maximum Supply Voltage	VBAT	-0.3 to 3.6	V
	BAK	-0.3 to 3.6	
	VL2	-0.3 to 3.6	
Maximum Input Voltage	Vin	-0.3 to VBAT+0.3	
Maximum output Voltage	Vout1	-0.3 to BAK+0.3	
	Vout2	-0.3 to VL2+0.3	
Maximum Operating Temperature	Topg	-20 to+70	°C
Maximum Storage Temperature	Tstg	-25 to+125	

ALLOWABLE OPERATING CONDITIONS

At Ta= –20°C to 70°C, GND=0V

Name	Symb.	Condition	Min.	Typ.	Max.	Unit
Supply Voltage	VBAT	Connected BAK to VBAT	1.2	1.5	1.8	V
	VL2		2xBAKx0.9		2xBAK+0.1	
Input “H” Voltage	Vih1	IOC and IOD port in input mode	VBAT-0.7	-	VBAT+0.7	
Input “L” Voltage	Vil1		-0.7	-	0.7	

ALLOWABLE OPERATING FREQUENCY

At Ta= –20°C to 70°C, GND=0V

Condition	Max. Operating Frequency
BAK=1.5V	800 KHz

ELECTRICAL CHARACTERISTICS INTERNAL RC FREQUENCY RANGE

Option Mode	BAK	Min.	Typical	Max.
350 KHz	1.5V		350 KHz	
650 KHz			650 KHz	

Input Resistance

VBAT=1.5V

Name	Symb.	Condition	Min.	Typ.	Max.	Unit
“L” Level Hold Tr. (IOC1~4)	Rllh1	Vi=0.2VBAT	10	40	70	KΩ
IOA4, IOB3~4, IOC1~4 Pull-Down Tr.	Rmad1	Vi=VBAT	200	500	1100	
RES Pull-Down R	Rres1	Vi=GND or VBAT	50	70	100	

DC Output Characteristics

(VL2=1.2V)

Name	Symb.	Condition	Port	Min.	Typ.	Max.	Unit
Output "H" Voltage	Voh1c	Ioh= -100uA	SEG1~9 , IOB3,4/BZB, BZ, IOC1~4	0.8	0.9	1.0	V
Output "L" Voltage	Vol1c	Iol= 200uA		0.2	0.3	0.4	
Output "H" Voltage	Voh2c	Ioh= -200uA	RR, RT, IOA4	0.8	0.9	1.0	
Output "L" Voltage	Vol2c	Iol= 400uA		0.2	0.3	0.4	

Segment Driver Output Characteristics

Name	Symb.	Condition	For	Min.	Typ.	Max.	Unit
1/2 Bias Display Mode							
Output "H" Voltage	Voh12f	Ioh= -1 uA	SEG-n	2.2			V
Output "L" Voltage	Vol12f	Iol= 1 uA				0.2	
Output "H" Voltage	Voh12g	Ioh= -10 uA	COM-n	2.2			
Output "M" Voltage	Vom12g	Iol/h= +/-10 uA	COM-n	1.0		1.4	
Output "L" Voltage	Vol12g	Iol= 10 uA				0.2	

POWER CONSUMPTION

@Ta= –20°C to 70°C, GND=0V, VBAT=1.5V

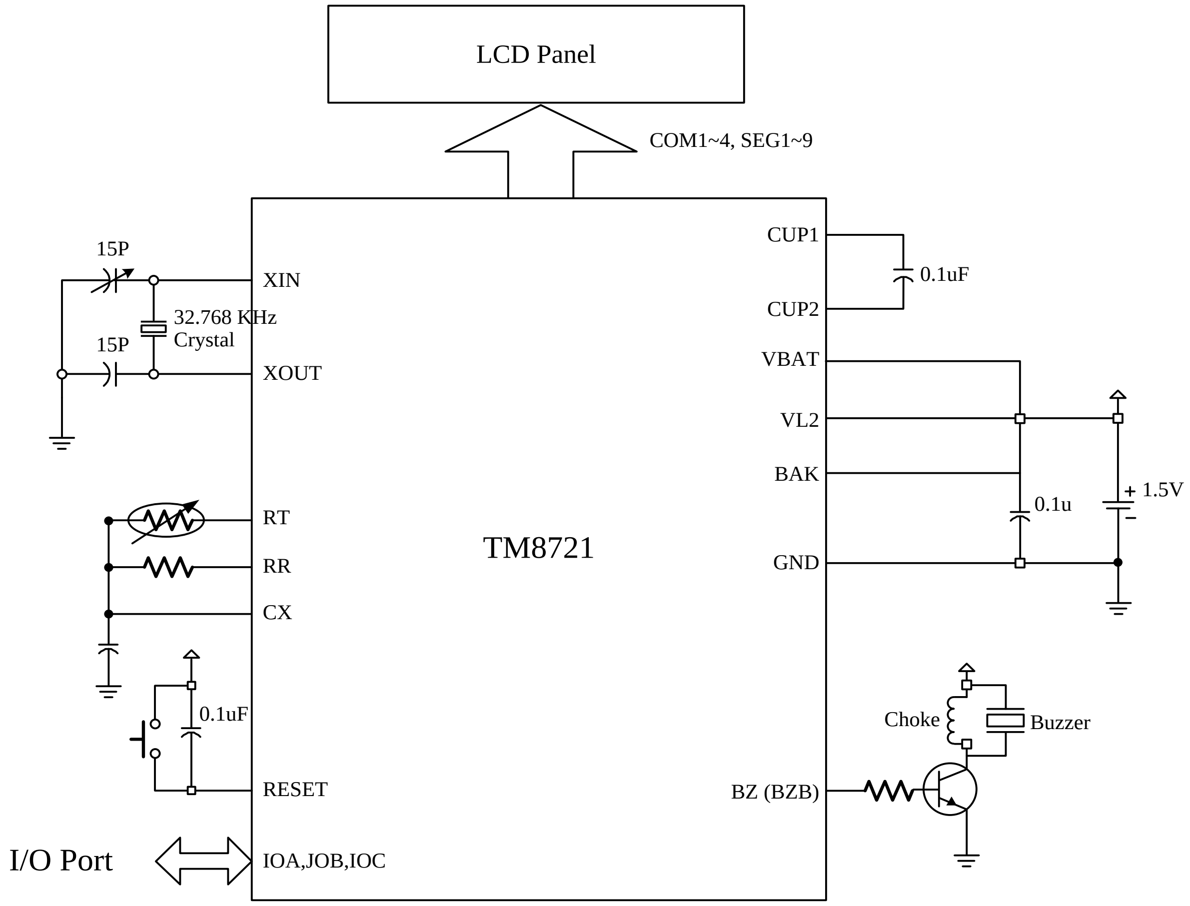
Name	Sym.	Condition	Min.	Typ.	Max.	Unit
HALT mode	IHALT	Only 32.768 KHz Crystal oscillator operating, without loading.		2		uA
STOP mode	ISTOP				1	
Operating current	Iop1	RFC operating *	20	45	60	

*** Rtp and Rref=30 KΩ, Cx=0.001 uF**

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

1.8 TYPICAL APPLICATION CIRCUIT

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



1.5Vpower mode, LCD

2. TM8721 Internal System Architecture

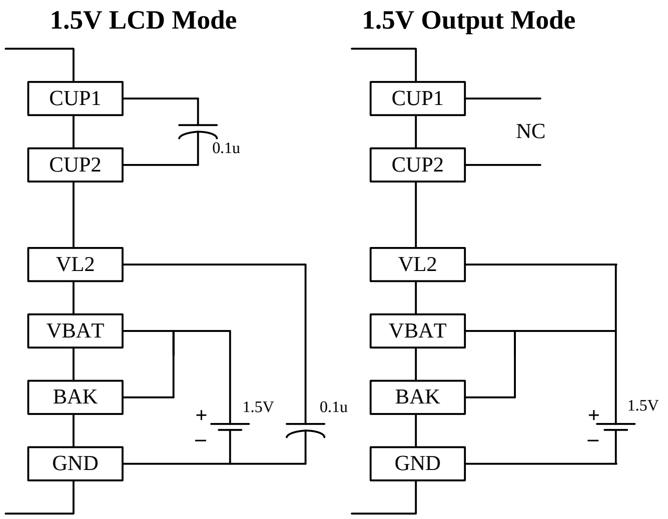
2.1 Power Supply

TM8721 could operate at 1.5V supply voltage. The power supply circuitry also generated the necessary voltage level to drive the LCD panel with 1/2 bias. Shown below are the connection diagrams for application.

2.1.1 1.5V POWER SUPPLY

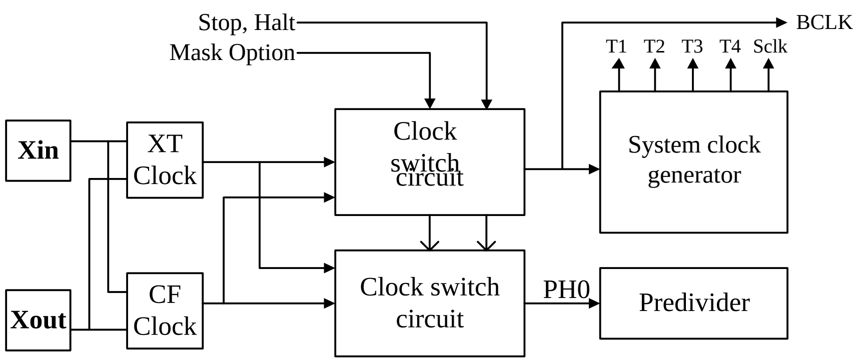
Operating voltage range: 1.2V~1.8V.

For LCD and Output mode application, the connection diagrams are shown below:



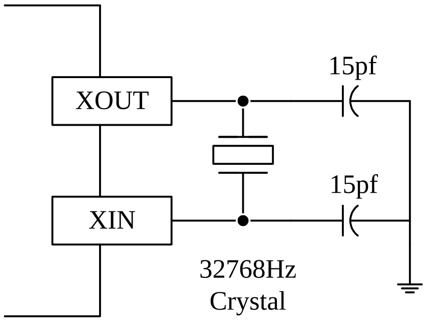
2.2 SYSTEM CLOCK

XT clock (X'TAL oscillator) and CF clock (Internal R and External R oscillator) compose the clock oscillation circuitry and the block diagram is shown below.



2.2.1 CONNECTION DIAGRAM OF X'TAL OSCILLATOR (XT CLOCK)

This clock oscillation circuitry provides the lower speed clock to the system clock generator, pre-divider, timer, chattering prevention of IOA4 port and LCD circuitry. In stop mode, this oscillator will be stopped.



(1) X'tal

2.2.2 CONNECTION DIAGRAM OF INTERNAL R and EXTERNAL R OSCILLATOR (CF CLOCK)

This oscillator will provide the clock to the system clock generator, pre-divider, timer, I/O port chattering prevention clock and LCD circuitry.

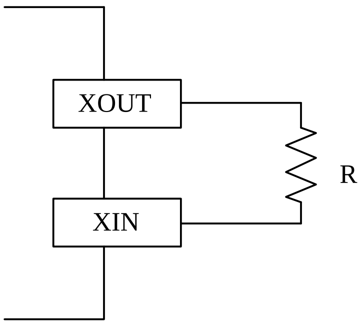
There are 2 type oscillators can be used in CF clock oscillator, selected by mask option:

2.2.2.1 RC OSCILLATOR WITH EXTERNAL RESISTOR (CF CLOCK)

This kind of oscillator could only be used in "EXTERNAL RESISTOR" option. When this oscillator is used, the frequency option of the RC oscillator with internal RC is not cared.

MASK OPTION table:

Mask Option name	Selected item
CLOCK SOURCE	(2) EXTERNAL RESISTOR



External
Resistor

2.2.2.2 RC OSCILLATOR WITH INTERNAL RESISTOR (CF CLOCK)

Two kinds of the frequencies could be selected in this mode of oscillator, the one is 350KHz and the other is 650KHz. When this oscillator is used, leave XIN and XOUT two pins opened.

MASK OPTION table:

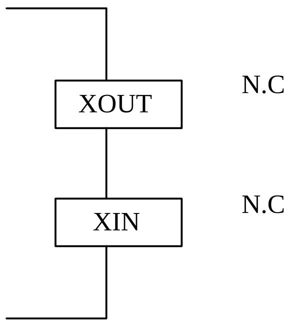
Mask Option name	Selected item
CLOCK SOURCE	(1) INTERNAL RESISTOR

For 350KHz output frequency:

Mask Option name	Selected item
CLOCK FREQUENCY OF INTERNAL RESISTOR MODE:	(1) 350 KHz

For 650KHz output frequency:

Mask Option name	Selected item
CLOCK FREQUENCY OF INTERNAL RESISTOR MODE:	(2) 650 KHz



Internal R

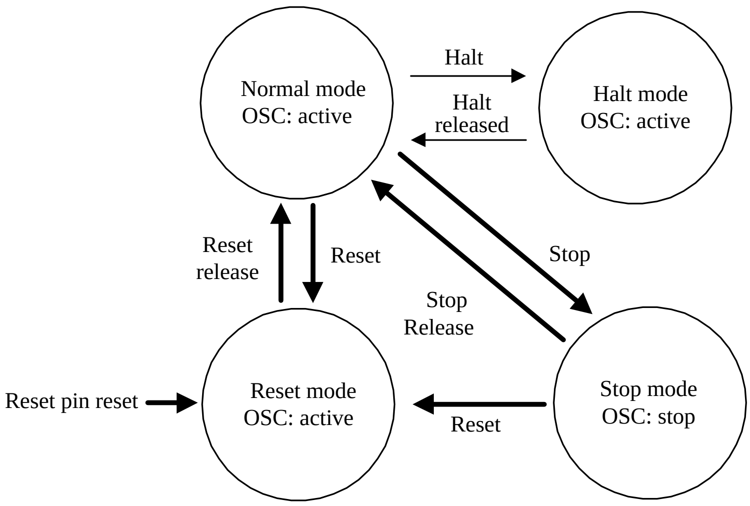
Frequency Range of Interanl RC Oscillator

Option Mode	Min.	Typ.	Max.
350KHz		350 KHz	
650KHz		650 KHz	

2.2.3 SINGLE CLOCK

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

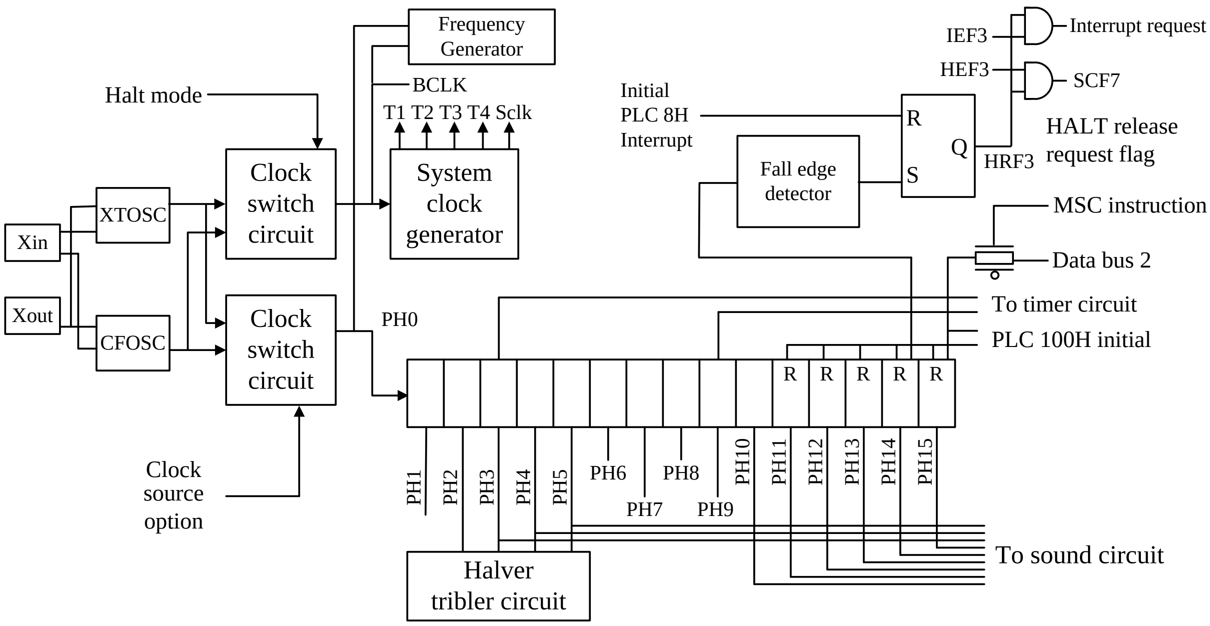
Either XT or CF clock may be selected by mask option in this mode.



This figure shows the State Diagram of Clock

2.2.4 PREDIVIDER

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



This figure shows the Pre-divider and its Peripherals

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

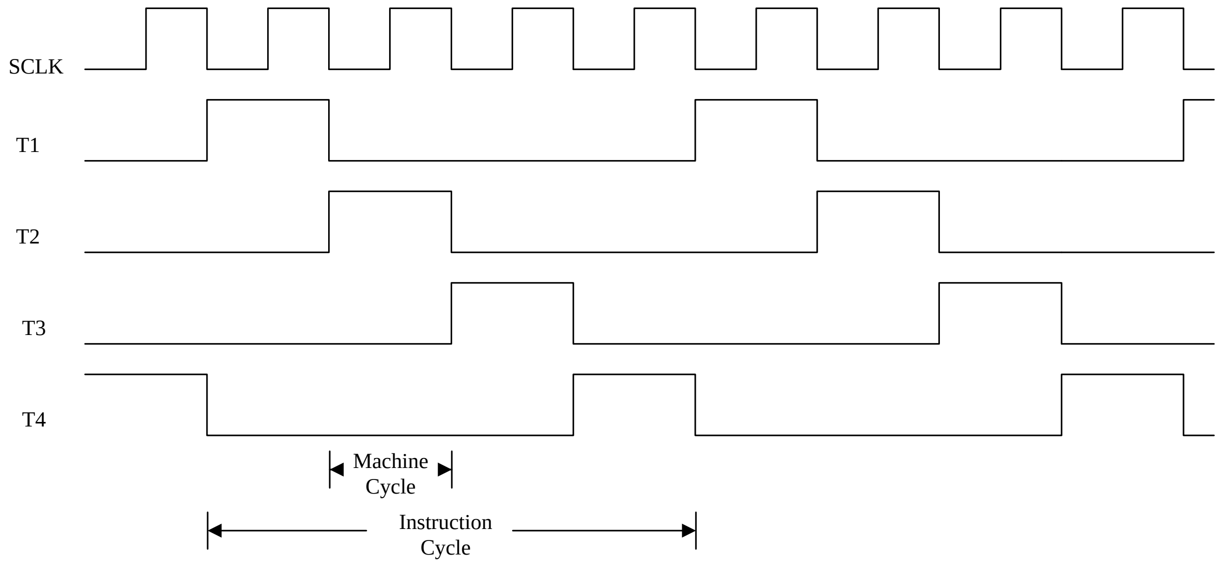
The clock source of pre-divider is PH0, and 4 kinds of frequency of PH0 could be selected by mask option:

MASK OPTION table:

Mask Option name	Selected item
PH0 <-> BCLK FOR INTERNAL OR EXTERNAL RESISTOR	(1) PH0=BCLK
PH0 <-> BCLK FOR INTERNAL OR EXTERNAL RESISTOR	(2) PH0=BCLK/4
PH0 <-> BCLK FOR INTERNAL OR EXTERNAL RESISTOR	(3) PH0=BCLK/8
PH0 <-> BCLK FOR INTERNAL OR EXTERNAL RESISTOR	(4) PH0=BCLK/16

2.2.5 SYSTEM CLOCK GENERATOR

The basic system clock is shown below:



2.3 PROGRAM COUNTER (PC)

This is an 10-bit counter, which addresses the program memory (ROM) up to 1024 addresses.

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

  PC ← PC+1

- When executing JMP instruction, subroutine call instruction (CALL), interrupt service routine or reset occurs, the program counter (PC) loads the specified address corresponding to table 2-1.

  PC ← specified address shows in

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

  PC ← specified address in operand

- Return instruction (RTS)

  PC ← content of stack specified by the stack pointer

  Stack pointer ← stack pointer -1

Table 2-1

	PC9	PC8	PC7	PC6	PC5	PC4	PC3	PC2	PC1	PC0
Initial reset	0	0	0	0	0	0	0	0	0	0
Interrupt 0 (input port A)	0	0	0	0	0	1	0	1	0	0
Interrupt 3 (pre-divider interrupt)	0	0	0	0	0	1	1	1	0	0
Interrupt 4 (timer 2 interrupt)	0	0	0	0	1	0	0	0	0	0
Interrupt 6 (RFC counter interrupt)	0	0	0	0	1	0	1	0	0	0
Jump instruction	P9	P8	P7	P6	P5	P4	P3	P2	P1	P0
Subroutine call	P9	P8	P7	P6	P5	P4	P3	P2	P1	P0

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

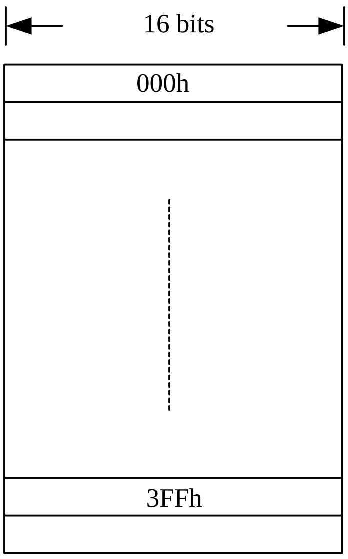
2.4 PROGRAM/TABLE MEMORY

The built-in mask ROM is organized with 1024 x 16 bits.

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

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

Table ROM memory space=256(7 -N) bytes (N=0~7).



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

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

MASK OPTION table:

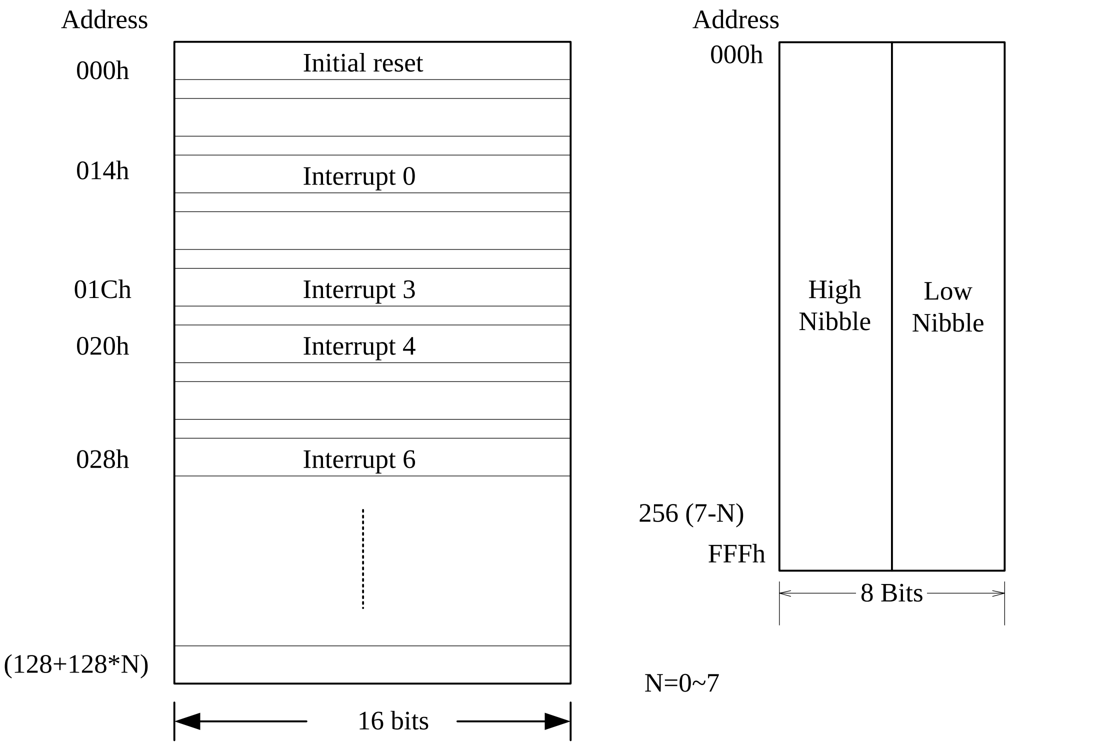
Mask Option name	Selected item	Instruction ROM memory space (Words)	Table ROM memory space (Bytes)
INSTRUCTION ROM <-> TABLE ROM	1 (N=1)	128	1792
INSTRUCTION ROM <-> TABLE ROM	2 (N=2)	256	1536
INSTRUCTION ROM <-> TABLE ROM	3 (N=3)	384	1280
INSTRUCTION ROM <-> TABLE ROM	4 (N=4)	512	1024
INSTRUCTION ROM <-> TABLE ROM	5 (N=5)	640	768
INSTRUCTION ROM <-> TABLE ROM	6 (N=6)	768	512
INSTRUCTION ROM <-> TABLE ROM	7 (N=7)	896	256
INSTRUCTION ROM <-> TABLE ROM	8 (N=8)	1024	0

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

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

2.4.1 INSTRUCTION ROM (PROM)

There are some special locations that serve as the interrupt service routines, such as reset address (000H), interrupt 0 address (014H), interrupt 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)

The table ROM is organized with 256 (7-N) x 8 bits that shared the memory space with instruction ROM, as shown in the figure above. This memory space stores the constant data or look up table for the usage of main program. All of the table ROM addresses are specified by the index address register (@HL). The data width could be 8 bits (256 (7-N) x 8 bits) or 4 bits (512 (7-N) x 4 bits) which depends on the different usage. Refer to the explanation of instruction chapter.

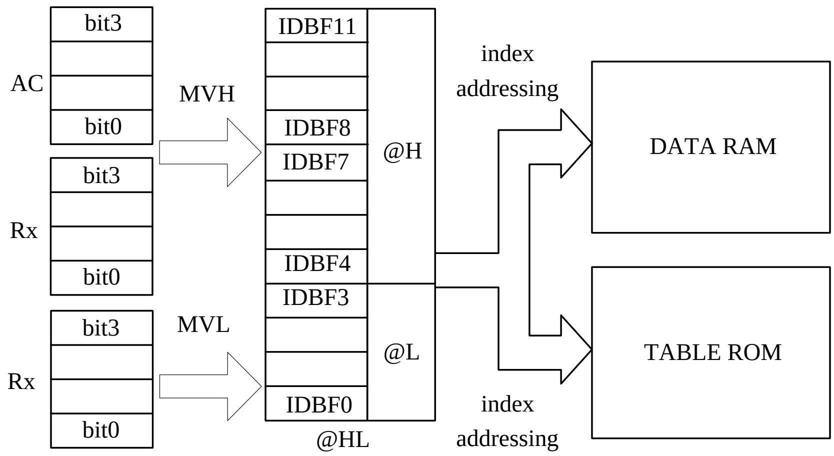
2.5 INDEX ADDRESS REGISTER (@HL)

This is a versatile address pointer for the data memory (RAM) and table ROM (TROM). The index address register (@HL) is a 12-bit register, and the contents of the register can be modified by executing MVH and MVL instructions. Executed MVL instruction will load the content of specified data memory to the lower nibble of the index register (@L). In the same manner, executed MVH instructions will load the contents of the data RAM (Rx) and AC to the higher 2 nibbles of the index register (@H)

@L is a 4-bit register and @H is an 8-bit register.

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

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



This figure shows the diagram of the index address register

2.6 STACK REGISTER (STACK)

Stack is a special design register following the first-in-last-out rule. It is used to save the contents of the program counter sequentially during subroutine call or execution of the interrupt service routine.

The contents of stack register are returned sequentially to the program counter (PC) while executing return instructions (RTS).

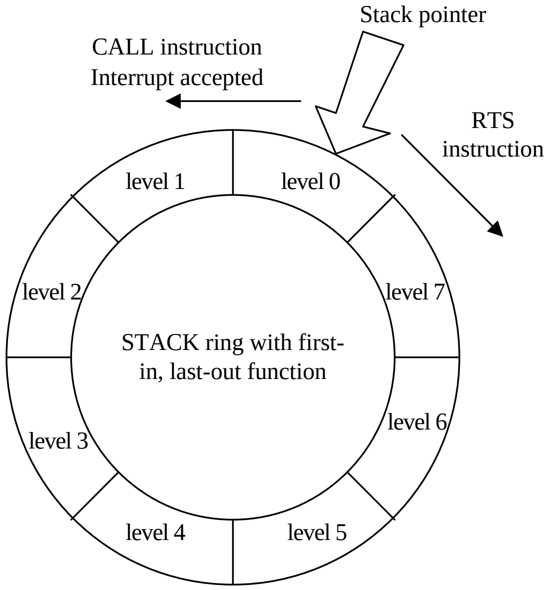
The stack register is organized using 11 bits by 8 levels but with no overflow flag; hence only 8 levels of subroutine call or interrupt are allowed (If the stacks are full, and either interrupt occurs or subroutine call executes, the first level will be overwritten).

Once the subroutine call or interrupt causes the stack register (STACK) overflow, the stack pointer will return to 0 and the content of the level 0 stack will be overwritten by the PC value.

The contents of the stack register (STACK) are returned sequentially to the program counter (PC) during execution of the RTS instruction.

Once the RTS instruction causes the stack register (STACK) underflow, the stack pointer will return to level 7 and the content of the level 7 stack will be restored to the program counter.

The following figure shows the diagram of the stack.



2.7 DATA MEMORY (RAM)

The static RAM is organized with 64 addresses x 4 bits and is used to store data.

The data memory may be accessed using two methods:

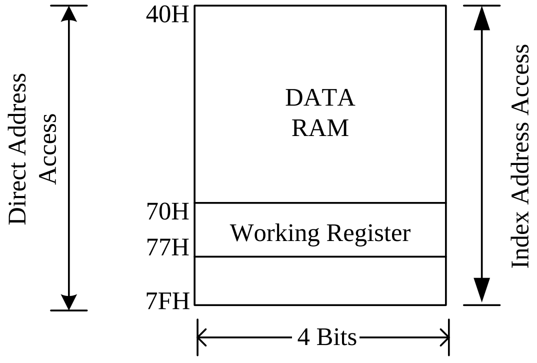
1. **Direct addressing mode**

   The address of the data memory is specified by the instruction and the addressing range is from 40H to 7FH.

2. **Index addressing mode**

   The index address register (@HL) specifies the address of the data memory and all address space from 40H to FFH can be accessed.

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



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

2.8 WORKING REGISTER (WR)

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

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

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

   MWR Rx, Ry; MRW Ry, Rx

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

   LCT, LCB, LCP

2.9 CCUMULATOR (AC)

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

2.10 ALU (Arithmetic and Logic Unit)

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

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

2.11 HEXADECIMAL CONVERT TO DECIMAL (HCD)

Decimal format is another number format for TM8721. When the content of the data memory has been assigned as decimal format, it is necessary to convert the results to decimal format after the execution of ALU instructions. When the decimal converting operation is processing, all of the operand data (including the contents of the data memory (RAM), accumulator (AC), immediate data, and look-up table) should be in the decimal format, or the results of conversion will be incorrect.

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

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

Example 1:

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

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

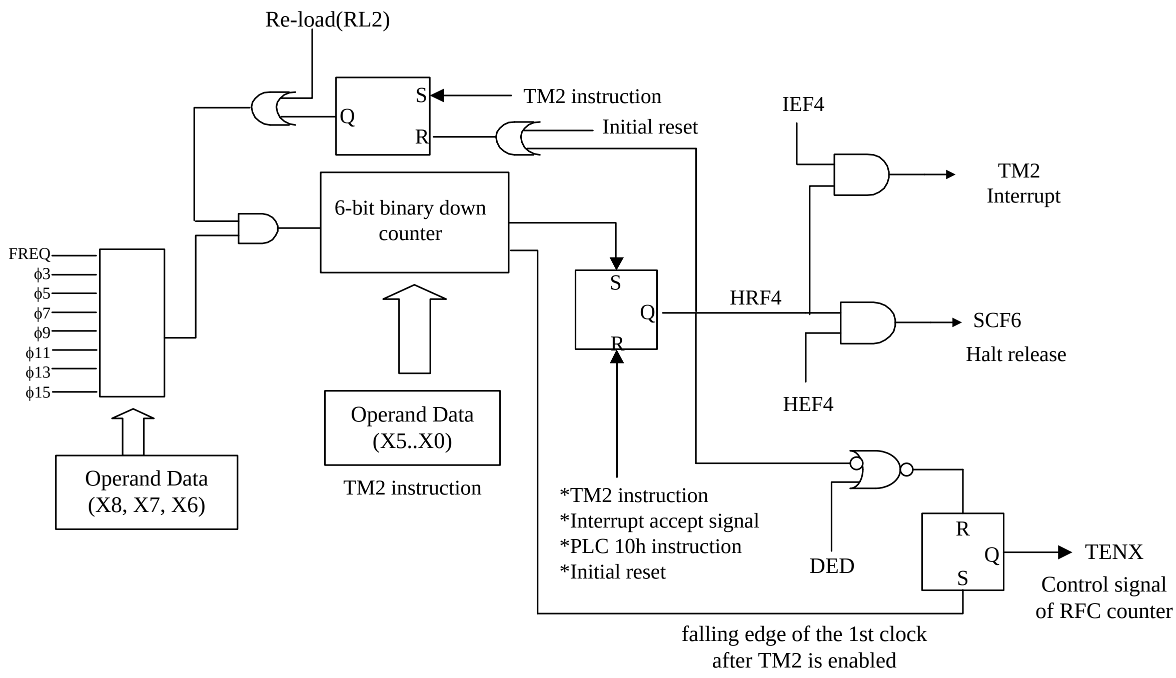
AC data before DAS execution	CF data before DAS execution	AC data after DAS execution	CF data after DAS execution
$0 \le AC \le 9$	CF=1	No change	no change
$6 \le AC \le F$	CF=0	AC=AC+A	no change

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          ; Content of data memory address 10H is binary-subtracted;
                 ; the result loads to data memory address
                 ; 10H. (R10=AC=FH, CF=0)
DAS*10h          ; Convert the content of the data memory address 10H to decimal
                   format.
                 ; The result in the data memory address 10H is"9"and in
                 ; the CF is "0". This represents the decimal number"–1".
```

2.12 TIMER 2 (TMR2)

The following figure shows the TMR2 organization.



2.12.1 NORMAL OPERATION

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

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

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

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

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

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

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

The following table shows the clock source setting for TMR2.

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

Notes:

1. When the TMR2 clock is PH3
   TMR2 set time = (Set value + error) * 8 * 1/fosc (KHz) (ms)
2. When the TMR2 clock is PH9
   TMR2 set time = (Set value + error) * 512 * 1/fosc (KHz) (ms)
3. When the TMR2 clock is PH15
   TMR2 set time = (Set value + error) * 32768 * 1/fosc (KHz) (ms)
4. When the TMR2 clock is PH5
   TMR2 set time = (Set value + error) * 32 * 1/fosc (KHz) (ms)
5. When the timer clock is PH7
   TMR2 set time = (Set value + error) * 128 * 1/fosc (KHz) (ms)
6. When the TMR2 clock is PH11
   TMR2 set time = (Set value + error) * 2048 * 1/fosc (KHz) (ms)
7. When the TMR2 clock is PH13
   TMR2 set time = (Set value + error) * 8192 * 1/fosc (KHz) (ms)
   Set value: Decimal number of timer set value
   error: the tolerance of set value, 0 < error <1.
   fosc: Input of the predivider
   PH3: The 3rd stage output of the predivider
   PHn: The nth stage output of the predivider (n =5, 7, 9, 11 ,13, 15)
8. When the TMR2 clock is FREQ
   TMR2 set time = (Set value + error) * 1/FREQ (KHz) (ms).
   FREQ: ***refer to section 3-3-1.***

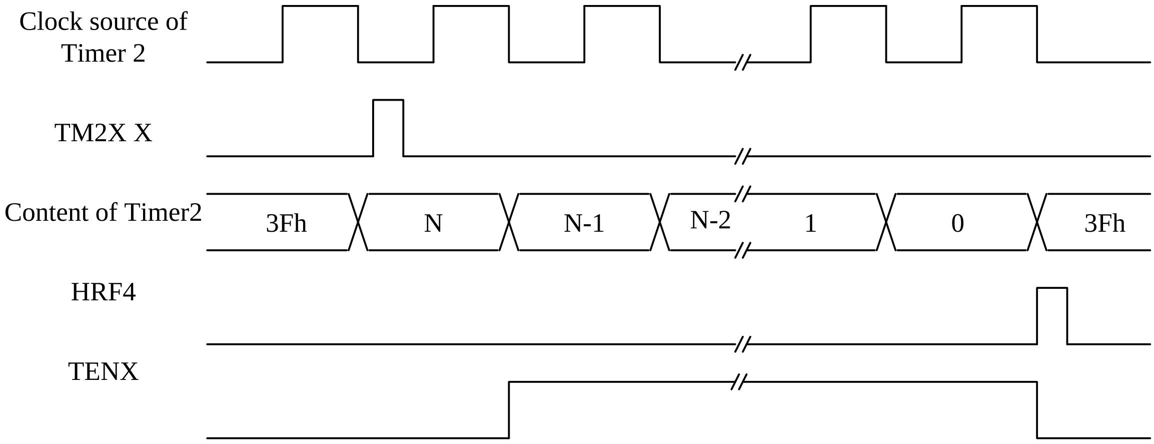
2.12.2 RE-LOAD OPERATION

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

2.12.3 TIMER 2 (TMR2) IN RESISTOR TO FREQUENCY CONVERTER (RFC)

TMR2 also controlled the operation of RFC function.

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



TMR2 also provides the re-load function when controlled the RFC function.

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

For example, if the target set value is 500, it will be divided as 52+7*64.

1. Set the initiate value of TMR2 to 52 and start counting.

2. Enable the TMR2 halt release or interrupt function.

3. Before the first underflow occurs, enable the re-load function and set the DED flag. The TMR2 will continue counting even if TMR2 underflows.

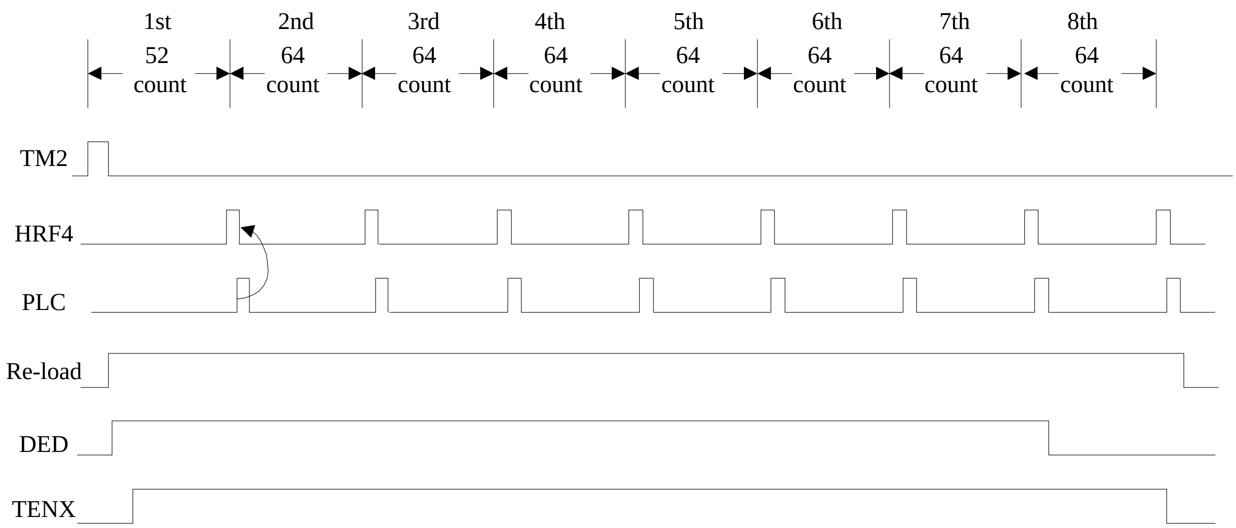
4. When halt release or interrupt occurs, clear the HRF4 flag by PLC instruction and increase the counting value to count the underflow times.

5. When halt release or interrupt occurs for the 7th time, reset the DED flag.

6. When halt release or interrupt occurs for the 8th time, disable the re-load function and the counting is completed.

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

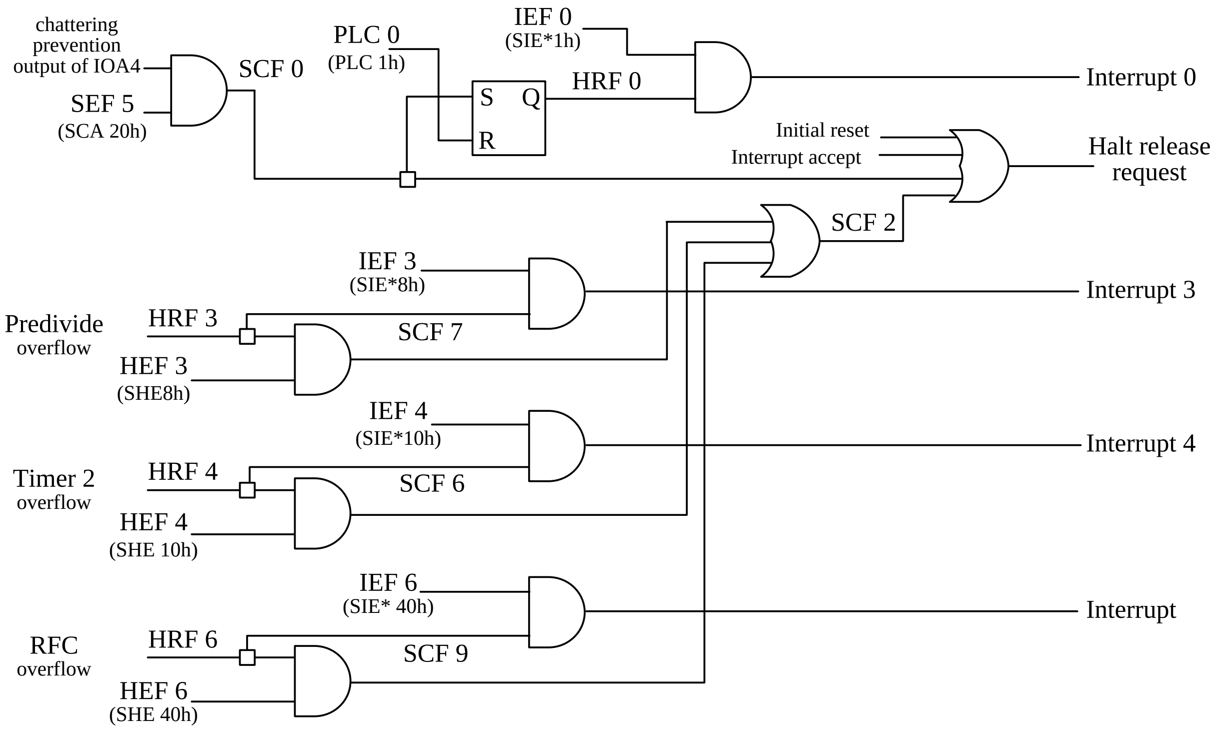
```
        LDS     0,0             ; initiate the underflow counting register
        PLC     10h
        SHE     10h             ; enable the halt release caused by TM2
        SRF     19h             ; enable RFC, and controlled by TM2
        TM2X    34h             ; initiate the TM value (52) and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



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

2.13 STATUS REGISTER (STS)

The status register (STS) is organized with 4 bits and comes in 4 types: status register 1 (STS1) to status register 4 (STS4). The following figure shows the configuration of the start condition flags for TM8721.



2.13.1 STATUS REGISTER 1 (STS1)

Status register 1 (STS1) consists of 2 flags:

1. **Carry flag (CF)**

   The carry flag is used to save the result of the carry or borrow during the arithmetic operation.

2. Zero flag (Z)

   Indicates the accumulator (AC) status. When the content of the accumulator is 0, the Zero flag is set to 1. If the content of the accumulator is not 0, the zero flag is reset to 0.

3. The MAF instruction can be used to transfer data in status register 1 (STS1) to the accumulator (AC) and the data memory (RAM).

4. The MRA instruction can be used to transfer data of the data memory (RAM) to the status register 1 (STS1).

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

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

2.13.2 STATUS REGISTER 2 (STS2)

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

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

Bit 3	Bit 2	Bit 1	Bit 0
NA	Start condition flag 2 (SCF2)	NA	NA
NA	Halt release caused by SCF4,5,6,7,9	NA	NA
Read only	Read only	Read only	Read only

Start condition flag 2 (SCF2)

When a factor other than port IOA4 causes the halt mode to be released, SCF2 will be set to1. In this case, if one or more start condition flags in SCF6, 7, 9 is set to 1, SCF2 will also be set to 1 simultaneously. When all of the flags in SCF6, 7, 9 are clear, start condition flag 2 (SCF2) is reset to 0.

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

2.13.3 STATUS REGISTER 3 (STS3)

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

Status register 3 (STS3) consists of 2 flags:

Start condition flag 7 (SCF7)

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

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

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

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

Bit 3	Bit 2	Bit 1	Bit 0
Start condition flag 7 (SCF7)	15th stage of the pre-divider	NA	NA
Halt release caused by pre-divider overflow		NA	NA
Read only	Read only	Read only	Read only

2.13.4 STATUS REGISTER 3X (STS3X)

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

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

Start condition flag 0 (SCF0)

When the SCA instruction specified signal change occurs at port IOA4 to release the halt mode, SCF0 will be set. Executing the SCA instruction will cause SCF0 to be reset to 0

Start condition flag 6 (SCF6)

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

Start condition flag 9 (SCF9)

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

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

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

Bit 3	Bit 2	Bit 1	Bit 0
Start condition flag 9 (SCF9)	Start condition flag 0 (SCF0)	Start condition flag 6 (SCF6)	NA
Halt release caused by RFC counter finish	Halt release caused by the IOA4 port	Halt release caused by TMR2 underflow	NA
Read only	Read only	Read only	Read only

2.13.5 STATUS REGISTER 4 (STS4)

Status register 4 (STS4) consists of 1 flag:

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

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

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

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

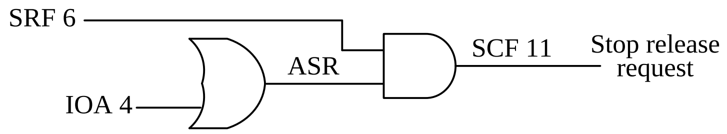
Bit 3	Bit 2	Bit 1	Bit 0
NA	The overflow flag of 16-bit counter of RFC (RFVOF)	NA	NA
Read only	Read only	Read only	Read only

2.13.6 START CONDITION FLAG 11 (SCF11)

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

A high level signal comes from the OR-ed output of the pins defined as input mode in IOA4 port, which causes the stop release flag of IOA4 port (ASR) to output, and stop release enable flag 6 (SRF6) is set beforehand.

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



The stop release flags (ASR) were specified by the stop release enable flags (SRF6) and these flags should be clear before the chip enters the stop mode. The IOA4 port had to be defined as the input mode and keep in 0 state before the chip enters the STOP mode, or the program can not enter the STOP mode.

Instruction SRE is used to set or reset the stop release enable flags (SRF6).

The following table shows the stop release request flags

	The input mode pins of IOA4 port
Stop release request flag	ASR
Stop release enable flag	SRF6

2.14 CONTROL REGISTER (CTL)

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

2.14.1 CONTROL REGISTER 1 (CTL1)

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

Switch enable flag 5 (SEF5)

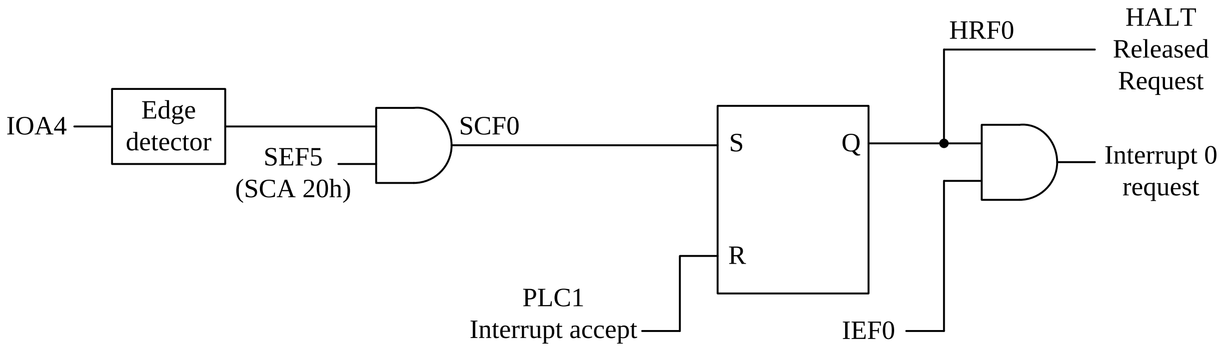
Stores the status of the input signal change at pins of IOA4 set in input mode that causes the halt mode or stop mode to be released.

Executed the SCA instruction may set or reset these flags.

The following table shows Bit Pattern of Control Register 1 (CTL1).

Bit 5
Switch enable flag 5 (SEF 5)
Enables the halt release caused by the signal change on IOA4 port
Write only

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



2.14.1.1 The Setting for Halt Mode

If the SEF5 is set to 1, the signal changed on IOA4 port will cause the halt mode to be released, and set SCF0 to 1.

2.14.1.2 The Setting for Stop Mode

If SRF5 and SEF5 are set, the stop mode will be released to set the SCF0 when a high level signal is applied to one of the input mode pins of IOA4 port.

After the stop mode is released, TM8721 enters the halt condition.

The high level signal must hold for a while to cause the chattering prevention circuitry of IOA4 port to detect this signal and then set SCF0 to release the halt mode, or the chip will return to the stop mode again.

2.14.1.3 Interrupt for CTL1

The control register 1 (CTL1) performs the following function in the execution of the SIE instruction to enable the interrupt function.

The input signal changes at the input pins in IOA4 port will deliver the SCF0 when SEF5 has been set to 1 by executing SCA instruction. Once the SCF0 is delivered, the halt release request flag (HRF0) will be set to 1. In this case, if the interrupt enable flag 0 (IEF0) is set to 1 by executing SIE instruction, the interrupt request flag 0 (interrupt 0) will be delivered to interrupt the program.

If the interrupt 0 is accepted by SEF5 and IEF0, the interrupt 0 request to the next signal change at IOA4 will be inhibited. To release this mode, SCA instruction must be executed again.

2.14.1.4 CONTROL REGISTER 2 (CTL2)

Control register 2 (CTL2) consists of halt release enable flags 3, 4, 6 (HEF3, 4, 6) and is set by 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 to be finished (HRF6)	Enable the halt release caused by TMR2 underflow (HRF4)
Halt release enable flag	HEF3	
Halt release condition	Enable the halt release caused by pre-divider overflow (HRF3)	

When the halt release enable flag 6 (HEF6) is set, a finish signal from the 16-bit counter of RFC causes the halt mode to be released. In the same manner, when HEF3 or HEF4 are set to 1, the following conditions will cause the halt mode to be released respectively: an overflow signal from the pre-divider and an underflow signal from TMR2.

2.14.1.5 CONTROL REGISTER 3 (CTL3)

Control register 3 (CTL3) is organized with 4 bits of interrupt enable flags (IEF) to enable/disable interrupts.

The interrupt enable flag (IEF) is set/reset by SIE* instruction. The bit pattern of control register 3 (CTL3) is shown below.

Interrupt enable flag	IEF6	IEF4
Interrupt request flag	Enable the interrupt request caused by RFC counter to be finished (HRF6)	Enable the interrupt request caused by TMR2 underflow (HRF4)
Interrupt flag	Interrupt 6	Interrupt 4
Interrupt enable flag	IEF3	
Interrupt request flag	Enable the interrupt request caused by predivider overflow (HRF3)	
Interrupt flag	Interrupt 3	
Interrupt enable flag	IEF0	
Interrupt request flag	Enable the interrupt request caused by IOA4 port signal to be changed (HRF0)	
Interrupt flag	Interrupt 0	

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

2.14.1.6 CONTROL REGISTER 4 (CTL4)

Control register 4 (CTL4), being a 1-bit register, is set/reset by SRE instruction.

The following table shows the Bit Pattern of Control Register 4 (CTL4)

Stop release enable flag	SRF6
Stop release request flag	Enable the stop release request caused by signal change on IOA4 pin (HRF0)

When the stop release enable flag 6 (SRF6) is set to 1, the input signal change at the IOA4 pins causes the stop mode to be released.

Example:

This example illustrates the stop mode released by port IOA4 pin. Assume IOA4 have been defined as input mode.

```
PLC     01h          ; Reset the HRF0.
SCA     20h          ; SEF5 is set so that the signal changes at port IOA4
                     ; cause the start conditions SCF0 to be set.
SRE     40h          ; SRF6 is set so that the signal changes at port
                     ; IOA4 pin cause the stop mode to be released.
STOP                 ; Enter the stop mode.
 ……………               ; STOP release
MCX     10h          ; Check the signal change at port IOA4 that causes the
                     : stop mode to be released.
```

2.15 HALT FUNCTION

The halt function is provided to minimize the current dissipation of the TM8721 when LCD is operating. During the halt mode, the program memory (ROM) is not in operation and only the oscillator circuit, pre-divider circuit, sound circuit, I/O port chattering prevention circuit, and LCD driver output circuit are in operation. (If the timer has started operating, the timer counter still operates in the halt mode).

After the HALT instruction is executed and no halt release signal (SCF0, HRF3, 4, 6) is delivered, the CPU enters the halt mode.

The following 2 conditions are available to release the halt mode.

(1) The signal change specified by the SCA instruction is applied to port IOA4 (SCF0).

(2) The halt release condition specified by the SHE instruction is met (HRF3, 4 ,6).

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

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

2.16 STOP FUNCTION (STOP)

The stop function is another solution to minimize the current dissipation for TM8721. In stop mode, all of functions in TM8721 are held including oscillators. All of the LCD corresponding signals (COM and Segment) will output "L" level. In this mode, TM8721 does not dissipate any power in the stop mode.

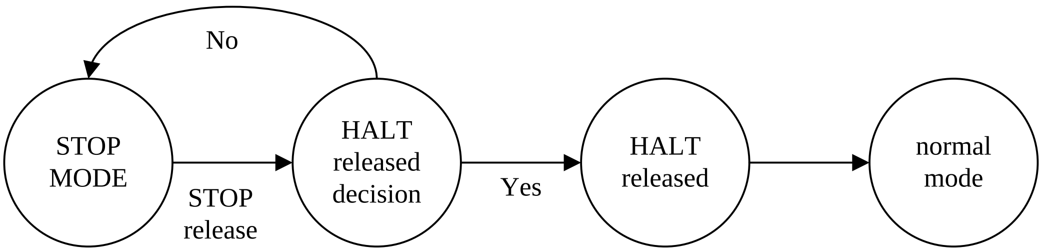
Before the stop instruction is executed, all of the signals on the pins defined as input mode of IOC port must be in the "L" state, and no stop release signal (SRFn) should be delivered. The CPU will then enter the stop mode.

The following conditions cause the stop mode to be released.

- One of the signals on the input mode pin of IOA4 port is in "H" state and holds long enough to cause the CPU to be released from halt mode.
- The stop release condition specified by the SRE instruction is met.

When the TM8721 is released from the stop mode, the TM8721 enters the halt mode immediately and will process the halt release procedure. If the "H" signal on the IOA4 port does not been held long enough to set the SCF0, once the signal on the IOA4 port returns to "L", the TM8721 will be entered the stop mode immediately.

The following diagram shows the stop release procedure:



The stop release state machine

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

- Specify the stop release conditions by the SRE instruction.
- Specify the halt release conditions corresponding to the stop release conditions if needed.
- Specify the interrupt conditions corresponding to the stop release conditions if needed.

When the stop mode is released by an interrupt request, the TM8721 will enter the halt mode immediately. While the interrupt is accepted, the halt mode will be released by the interrupt request. The stop mode returns by executing the RTS instruction after completion of interrupt service.

After the stop release, it is necessary that the MSB, MSC or MCX instruction be executed to test the halt release signal and that the PLC instruction then be executed to reset the halt release signal. Even when the stop instruction is executed in the state where the stop release signal (SRF) is delivered, the CPU does not enter the stop mode but the halt mode. When the stop mode is released and an interrupt is accepted, the halt release signal (HRF) is reset automatically.

3. Control Function

3.1 INTERRUPT FUNCTION

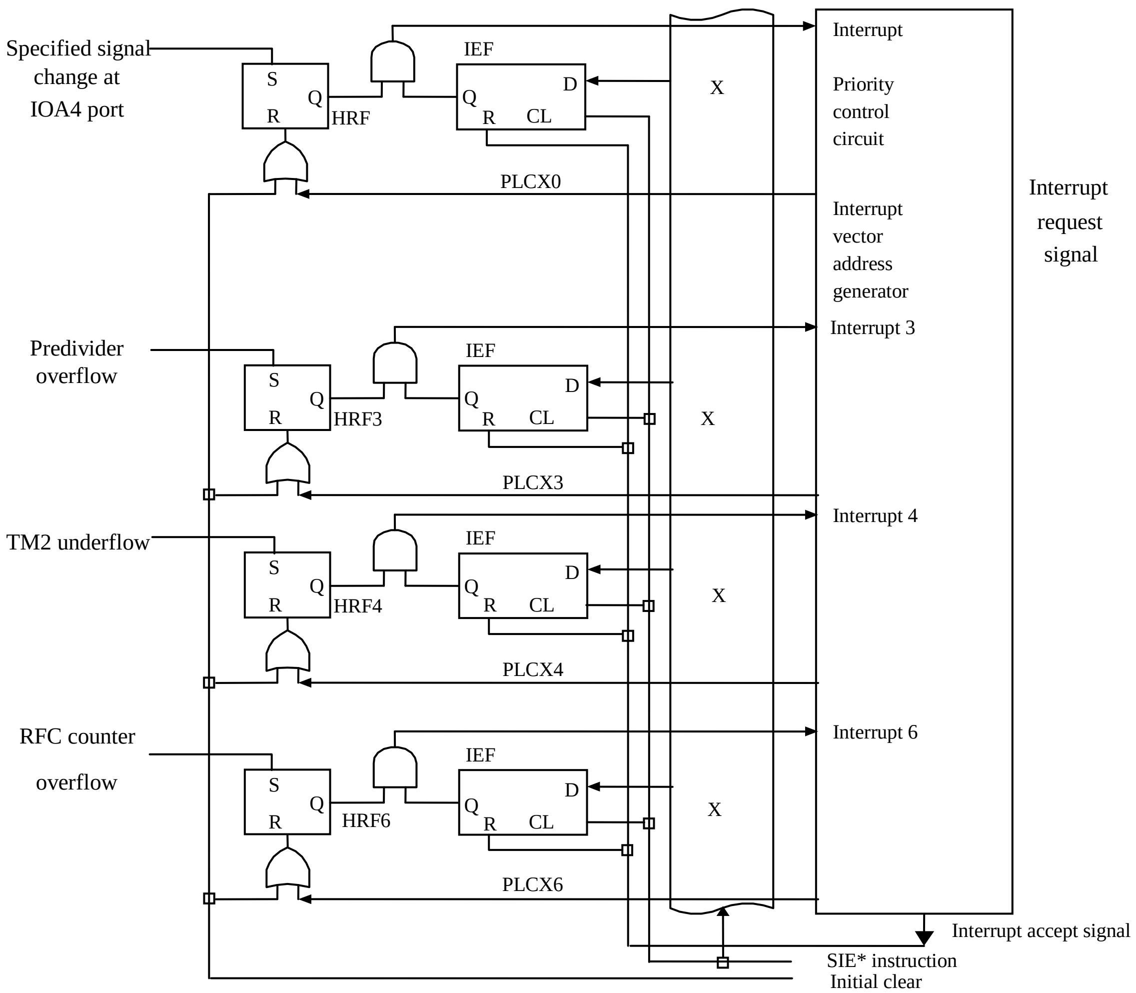
There are 4 interrupt resources: 1 external interrupt factors and 3 internal interrupt factors. When an interrupt is accepted, the program in execution is suspended temporarily and the corresponding interrupt service routine specified by a fix address in the program memory (ROM) is called.

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

Table 3.1 Interrupt information:

Interrupt source	IOA4 port	Pre-divider overflow	TMR2 underflow	RFC counter overflow
Interrupt vector	014H	01CH	020H	028H
Interrupt enable flag	IEF0	IEF3	IEF4	IEF6
Interrupt priority	5th	1st	3rd	4th
Interrupt request flag	Interrupt 0	Interrupt 3	Interrupt 4	Interrupt 6

The following figure shows the Interrupt Control Circuit:



3.1.1 INTERRUPT REQUEST AND SERVICE ADDRESS

3.1.1.1 External interrupt factor

The external interrupt factor involves the use of the IOA4 ports.

I/O port IOA4 interrupt request.

An interrupt request signal (HRF0) is delivered when the input signal changes at I/O port IOA4 specified by the SCA instruction. In this case, if the interrupt enabled by flag 0 (IEF0) is set to 1, interrupt 0 is accepted and the instruction at address 14H is executed automatically.

3.1.1.2 Internal interrupt factor

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

1. **Timer 2 (TMR 2) interrupt request**

   An interrupt request signal (HRF 4) is delivered when timer 2 (TMR2) underflows. In this case, if the interrupt enable flag 4 (IEF4) is set, interrupt 4 is accepted and the instruction at address 20H is executed automatically.

2. **Pre-divider interrupt request**

3. 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.

4. **16-bit counter of RFC (CX pin control mode) interrupt request**

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

3.1.2 INTERRUPT PRIORITY

If all interrupts are requested simultaneously during a state when all interrupts are enabled, the pre-divider interrupt is given the first priority and other interrupts are held. When the interrupt service routine is initiated, all of the interrupt enable flags (IEF0, 3, 4, 6) are cleared and should be set with the next execution of the SIE instruction. Refer to Table 3-1.

Example:

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

```
PLC     59h          ; Clear all of the HRF flags
SCA     20h          ; enable the interrupt request of IOA4
SIE*    58h          ; enable all interrupt requests

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

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

```
SIE*    51h          ; Enable the interrupt request (except the predivider).

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

SIE*    41h          ; Enable the interrupt request (except the predivider, TMR2).

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

SIE*    01h          ; Enable the interrupt request (except the predivider,
; TMR2, and the RFC counter).

; Interrupt caused by the IOA4 port, and interrupt service is concluded.

SIE*    04h          ; Enable the interrupt request (except the predivider, TMR1,
; TMR2, RFC counter, and IOA4 port)

; All interrupt requests have been processed.
```

3.1.3 INTERRUPT SERVICING

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

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

(2) The corresponding interrupt service routine address is loaded in the program counter (PC).

The interrupt request flag corresponding to the interrupt accepted is reset and the interrupt enable flags are all reset.

When the interrupt occurs, the TM8721 will follow the procedure below:

```
Instruction 1        ; In this instruction, interrupt is accepted.
NOP                  ; TM8721 stores the program counter data into the STACK.
                     ; At this time, no instruction will be executed, as with NOP instruction.
Instruction A        ; The program jumps to the interrupt service routine.
Instruction B
Instruction C
 ............
RTS                  ; Finishes the interrupt service routine
Instruction 1*       ; re-executes the instruction which was interrupted.
Instruction 2
```

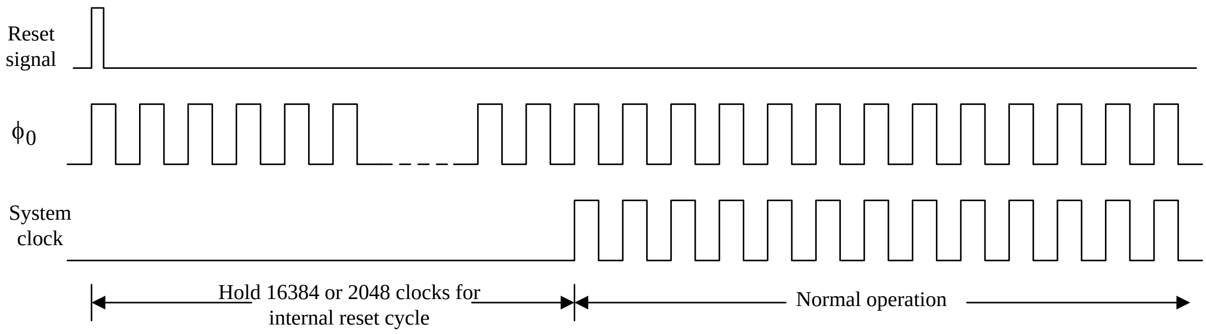
Note: If instruction 1 is "halt" instruction, the CPU will return to "halt" after interrupt.

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

3.2 RESET FUNCTION

TM8721 contains one reset sources: RESET pin reset.

When reset signal is accepted, TM8721 will generate a time period for internal reset cycle and there are two types of internal reset cycle time could be selected by mask option, the one is PH12/2.



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

3.2.1 RESET PIN RESET

When "H" level is applied to the reset pin, the reset signal will issue. Built in a pull down resistor on this pin.

Two types of reset method for RESET pin and the type could be mask option, the one is level reset and other is pulse reset.

It is recommended to connect a capacitor (0.1uf) between RESET pin and VBAT. This connection will prevent the bounce signal on RESET pin.

Once a “1” signal applied on the RESET pin, TM8721 will escape from reset state and begin the normal operation after internal reset cycle automatically no matter what the signal on RESET pin returned to “0” or not.

The following table shows the initial condition of TM8721 in reset cycle.

Program counter	(PC)	Address 000H
Start condition flags 1 to 7	(SCF1-7)	0
Stop release enable flags 4,5,7	(SRF3,4,5,7)	0
Switch enable flags 5	(SEF5)	0
Halt release request flag	(HRF0, 3, 4, 6)	0
Halt release enable flags 1 to 3	(HEF3, 4, 6)	0
Interrupt enable flags 0 to 3	(IEF0, 3, 4, 6)	0
Alarm output	(ALARM)	DC 0
Pull-down flags in I/OA4 port		1(with pull-down resistor)
Input/output ports I/OA4, I/OB3,4, I/OC	(PORT I/OA, I/OB, I/OC)	Input mode
I/OA4 port chattering clock	Cch	PH10*
Frequency generator clock source and duty	Cfq	PH0, duty cycle is 1/4, output

Program counter	(PC)	Address 000H
cycle		is inactive
Resistor frequency converter	(RFC)	Inactive, RR/RT output 0
LCD driver output		All lighted (mask option)*
Timer 2		Inactive

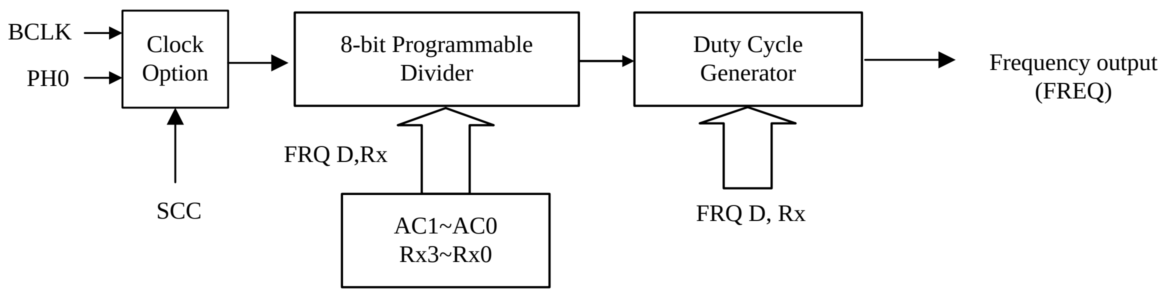
Notes: PH3: the 3rd output of predivider
PH10: the 10th output of predivider
Mask option can unlighted all of the LCD output

3.3 CLOCK GENERATOR

3.3.1 FREQUENCY GENERATOR

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

The following shows the organization of the frequency generator.



SCC instruction may specify the clock source selection for the frequency generator. The frequency generator outputs the clock with different frequencies and duty cycles corresponding to the presetting data of FRQ related instructions. The FRQ related instructions preset a letter N into the programming divider and letter D into the duty cycle generator. The frequency generator will then output the clock using the following formula:

FREQ= (clock source) / ((N+1) * X) Hz. (X=1, 2, 3, 4 for 1/1, 1/2, 1/3, 1/4 duty)

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

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

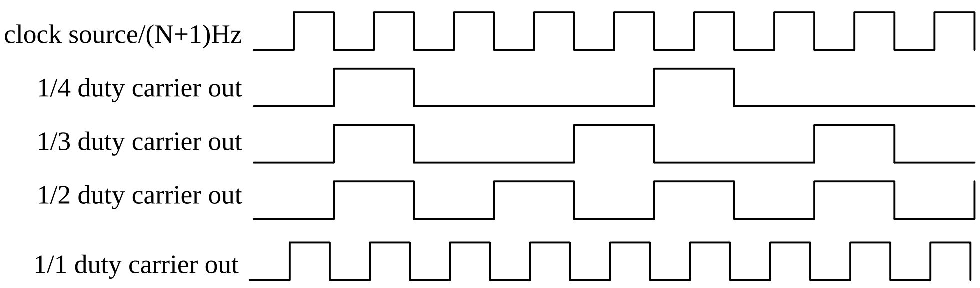
	The bit pattern of preset letter N							
Programming divider	bit7	Bit6	bit 5	bit 4	bit 3	Bit 2	bit 1	bit 0
FRQ D,Rx	AC3	C2	AC1	AC0	Rx3	Rx2	Rx1	Rx0
FRQ D,@HL	T7	T6	T5	T4	T3	T2	T1	T0
FRQX D,X	X7	X6	X5	X4	X3	X2	X1	X0

Notes: **(1).** T0 ~ T7 represents the data of table ROM.
(2). X0 ~ X7 represents the data specified in operand X.

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

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

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



3.3.2 Melody Output

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

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

The following table shows the note table for melody application.

Tone	N	FREQ	Ideal	%	Tone	N	FREQ	Ideal	%
C2	249	65.5360	65.4064	0.19	C4	62	260.063	261.626	-0.60
#C2	235	69.4237	69.2957	0.18	#C4	58	277.695	277.183	0.18
D2	222	73.4709	73.4162	0.07	D4	55	292.571	293.665	-0.37
#D2	210	77.6493	77.7817	-0.17	#D4	52	309.132	311.127	-0.64
E2	198	82.3317	82.4069	-0.09	E4	49	327.680	329.628	-0.59
F2	187	87.1489	87.3071	-0.18	F4	46	348.596	349.228	-0.18
#F2	176	92.5650	92.4986	0.07	#F4	43	372.364	369.994	0.64
G2	166	98.1078	97.9989	0.11	G4	41	390.095	391.995	-0.48
#G2	157	103.696	103.826	-0.13	#G4	38	420.103	415.305	1.16
A2	148	109.960	110.000	-0.04	A4	36	442.811	440.000	0.64
#A2	140	116.199	116.541	-0.29	#A4	34	468.114	466.164	0.42
B2	132	123.188	123.471	-0.23	B4	32	496.485	493.883	0.53
C3	124	131.072	130.813	0.20	C5	30	528.516	523.251	1.01

Tone	N	FREQ	Ideal	%	Tone	N	FREQ	Ideal	%
#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

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

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

3.3.3 Halver/Doubler/Tripler

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

3.3.4 Alternating Frequency for LCD

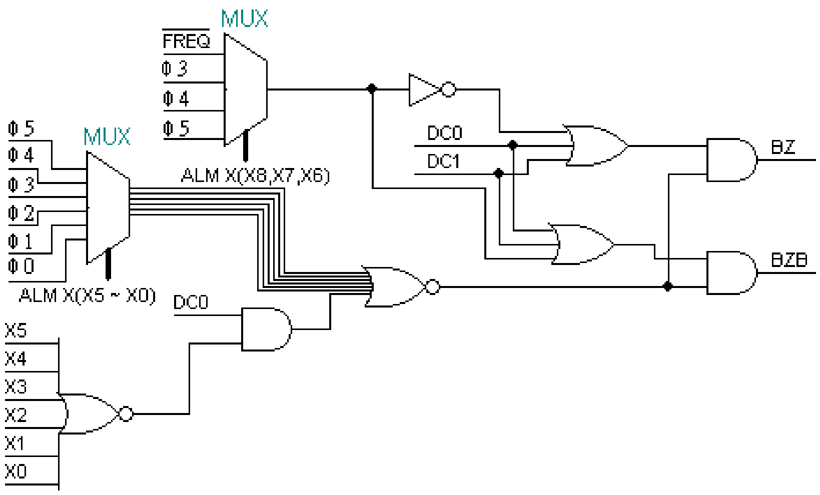
The alternating frequency for LCD is a frequency used to make the LCD waveform.

3.4 BUZZER OUTPUT PINS

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

MASK OPTION table:

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



This figure shows the organization of the buzzer output.

3.4.1 BASIC BUZZER OUTPUT

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

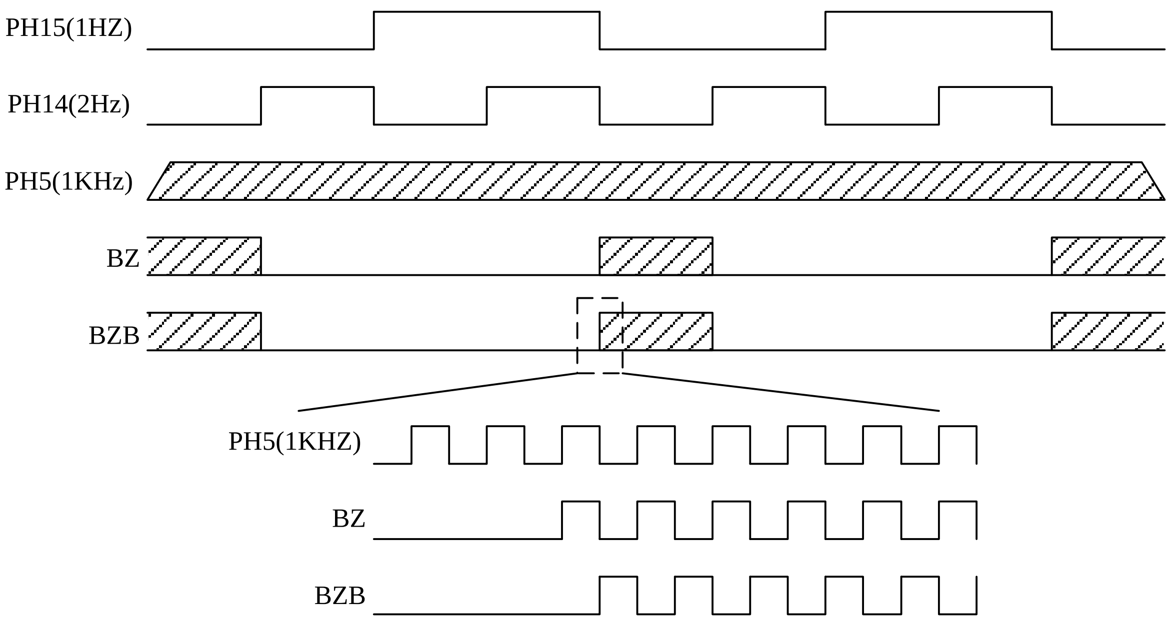
Notes: 1. The high frequency clock source should only be one of PH3, PH4, PH5 or FREQ, and the lower frequency may be any/all of the combinations from PH10 ~ PH15.
2. The frequencies in () corresponding to the input clock of the pre-divider (PH0) is 32768Hz.
3. The BZ and BZB pins will output DC0 after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

Example:

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

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

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



3.4.2 THE CARRIER FOR REMOTE CONTROL

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

Example:

```
SHE     10h          ; Enable timer 2 halt release enable flag.
TM2X    3Fh          ; Set value for timer 2 is 3Fh and the clock source is PH9.
SCC     40h          ; Set the clock source of  the frequency generator as BCLK.
FRQX    2, 3         ; FREQ=BCLK/ (4*2), setting value for the frequency
                     ; generator is 3 and duty cycle is 1/2.
ALM     1C0h         ; FREQ signal is outputted. This instruction must be executed
                     ; after the FRQ related instructions.
HALT                 ; Wait for the halt release caused by timer 2.
……………………….         ; Halt released.
ALM     0            ; Stop the buzzer output.
```

3.5 INPUT/OUTPUT PORTS

Three I/O ports are available in TM8721: IOA4, IOB and IOC.

When the I/O pins are defined as non-IO function by mask option, the input/output function of the pins will be disabled.

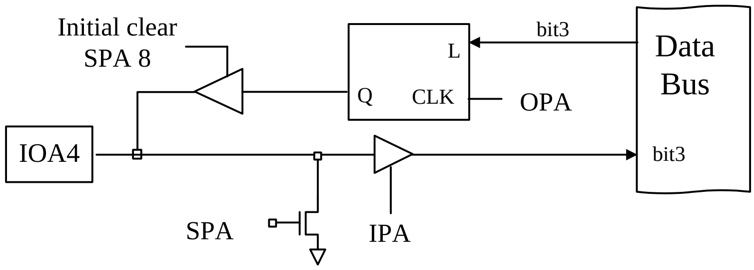
3.5.1 IOA4 PORT

In initial reset cycle, the IOA4 port is set as input mode and can be defined as input mode or output mode by executing SPA instructions. Executing OPA instructions may output the content of specified data memory to the pins defined as output mode; the pins defined as the input mode will still remain the input mode.

Executing IPA instructions may store the signals applied to the IO pins into the specified data memory. When the IO pins are defined as the output mode, executing IPA instruction will store the content that stored in the latch of the output pin into the specified data memory.

Before executing SPA instruction to define the I/O pins as the output mode, the OPA instruction must be executed to output the data to those output latches beforehand. This will prevent the chattering signal on the I/O pin when the I/O mode changed.

IOA$ port had built-in pull-down resistor and executing SPA instruction to enable/disable this device.



This figure shows the organization of IOA4 port.

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

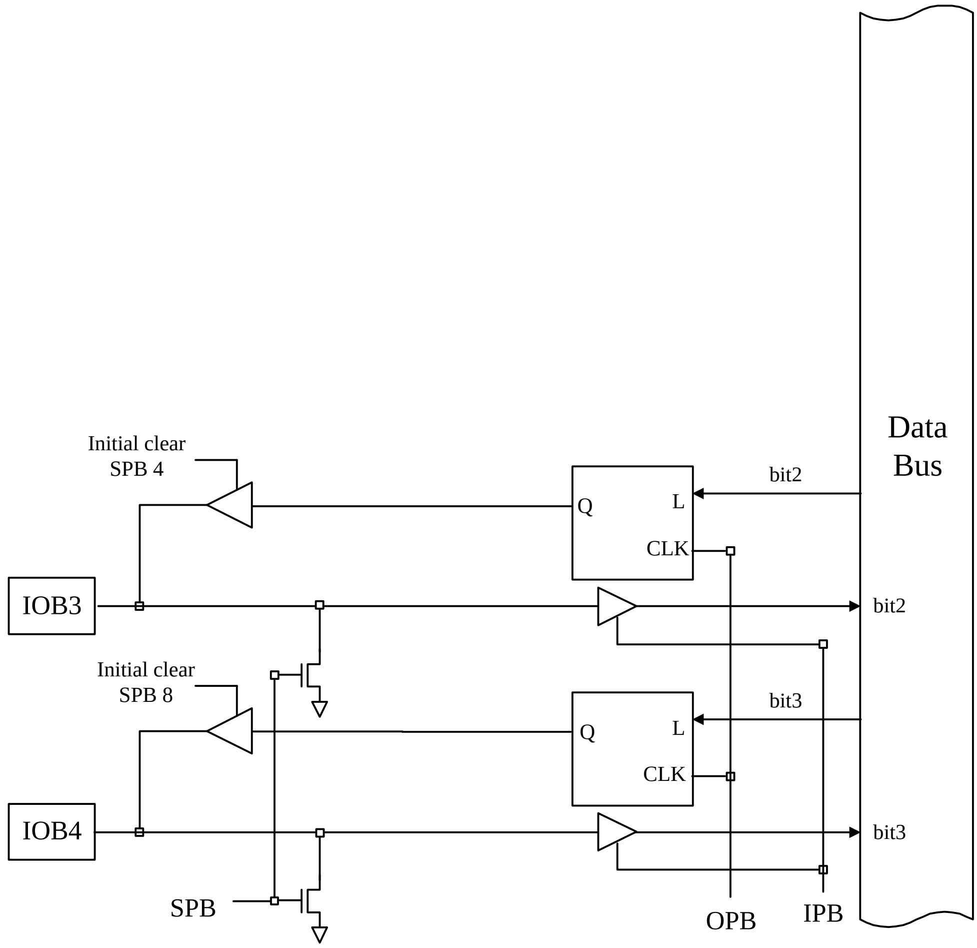
3.5.2 IOB PORT

IOB3, IOB4 pins are MUXed with BZB and BZ pins respectively by mask option.

MASK OPTION table:

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

The following figure shows the organization of IOB port.



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

After the reset cycle, the IOB port is set as input and each bit of port can be defined as input or output individually by executing SPB instructions. Executing OPB instructions may output the content of specified data memory to the pins defined as output mode; the other pins which are defined as the input will still be input.

Executed IPB instructions may store the signals applied on the IOB pins into the specified data memory. When the IOB pins are defined as the output, executing IPB instruction will save the data stored in the output latch into the specified data memory.

Before executing SPB instruction to define the I/O pins as output, the OPB instruction must be executed to output the data to the output latches. This will prevent the chattering signal on the I/O pin when the I/O mode changed.

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

3.5.3 IOC PORT

After the reset cycle, the IOC port is set as input mode and each bit of port can be defined as input mode or output mode individually by executing SPC instruction. Executed OPC instruction may output the content of specified data memory to the pins defined as output; the other pins which are defined as the input will still remain the input mode.

Executed IPC instructions may store the signals applied to the IOC pins in the specified data memory. When the IOC pins are defined as the output, executing IPC instruction will save the data stored in the output latches in the specified data memory.

Before executing SPC instruction to define the IOC pins as output, the OPC instruction must be executed to output the data to those output latches.

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

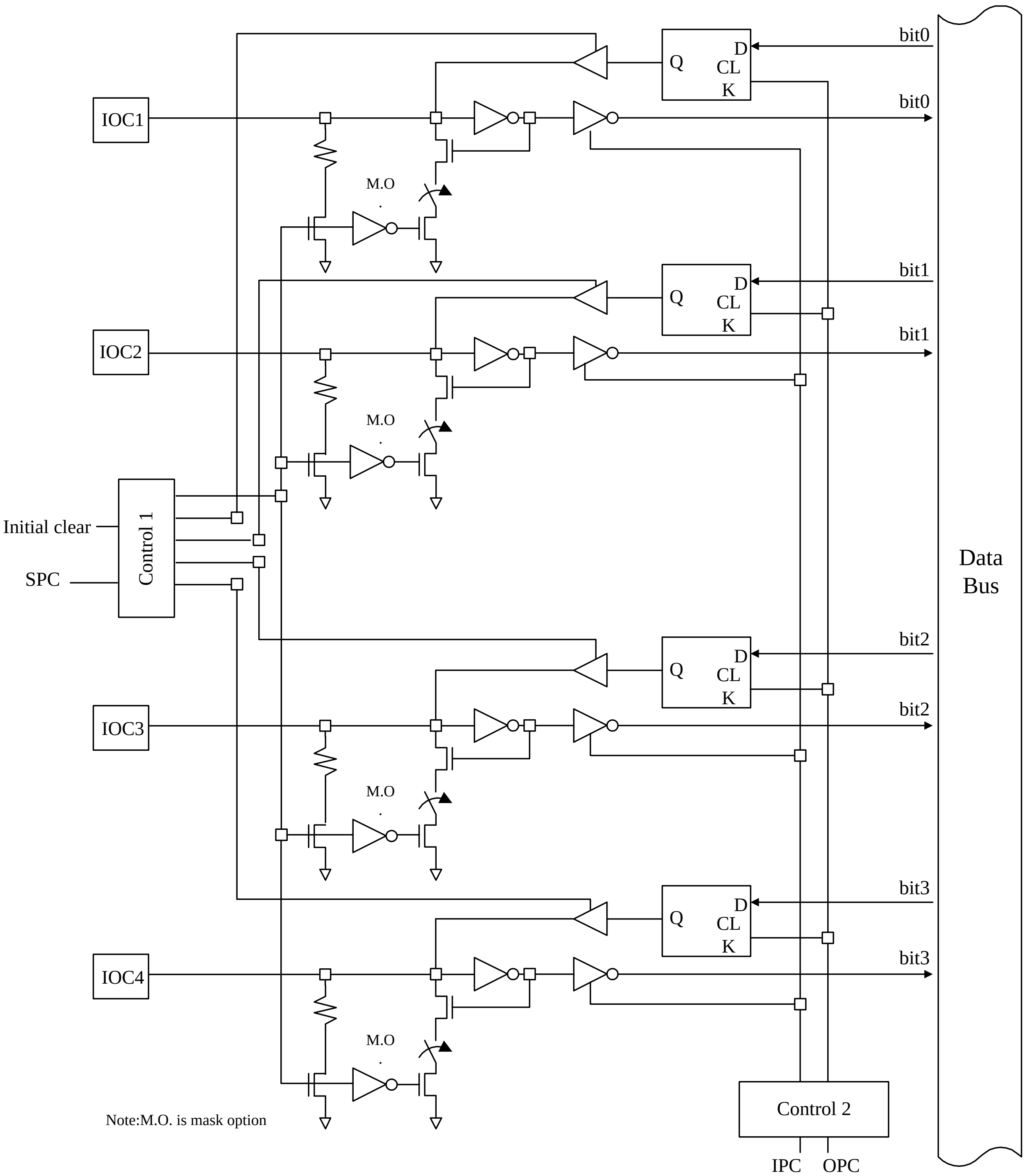
IOC port also built in the pull-low device for each pin, but these devices are enabled by mask option. The pull-down resistor and low-level-hold device in each IOC pin can't exist in the same time. When the pull-down resistor is enabled, the low-level-hold device will be disabled. Executing SPC 10h instruction to enable the pull-low device and disable the low-level hold device, executing SPC 0h to disable the pull-low device and enable the low-level hold device.

When the low-level hold device is enabled by mask option, the initial reset will enable the pull-low device and disable the low-level hold device.

When the IOC pin has been defined as the output mode, both the pull-low and low-level hold devices will be disabled.

Low-level-hold function option:

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

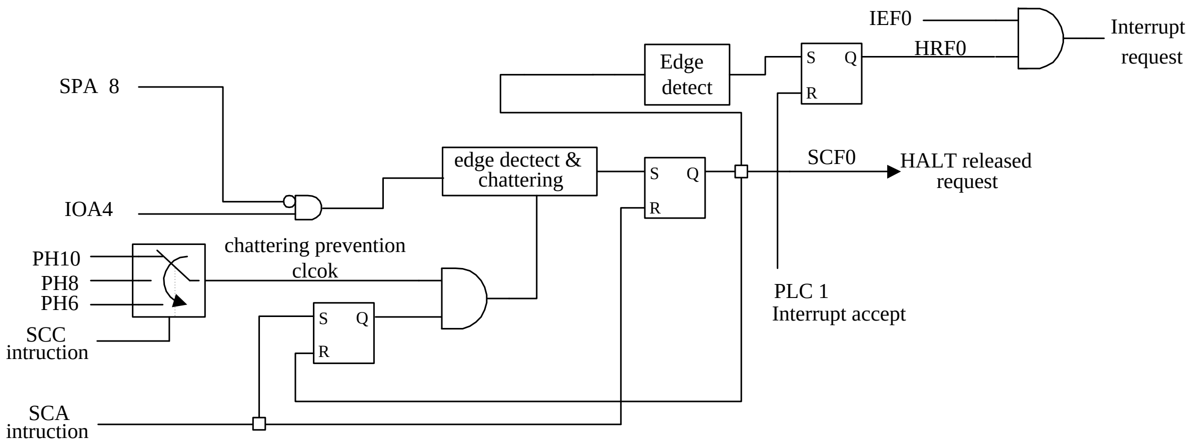


This figure shows the organization of IOC port.

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

3.5.4 IOA4 Chattering Prevention Function and Halt Release

The IOA4 pin is capable of preventing high/low chattering of the switch signal. The chattering prevention time can be selected as PH10 (32ms), PH8 (8ms) or PH6 (2ms) by executing SCC instruction, and the default selection is PH10 after the reset cycle. When IOA4 pin is defined as output, the signals applied to the output pins will be inhibited for the chattering prevention function. The following figure shows the organization of chattering prevention circuitry.



Note: The default prevention clock is PH10.

This chattering prevention function works when the signal of IOA4 is changed from “L” level to “H” level or from “H” level to “L” level.

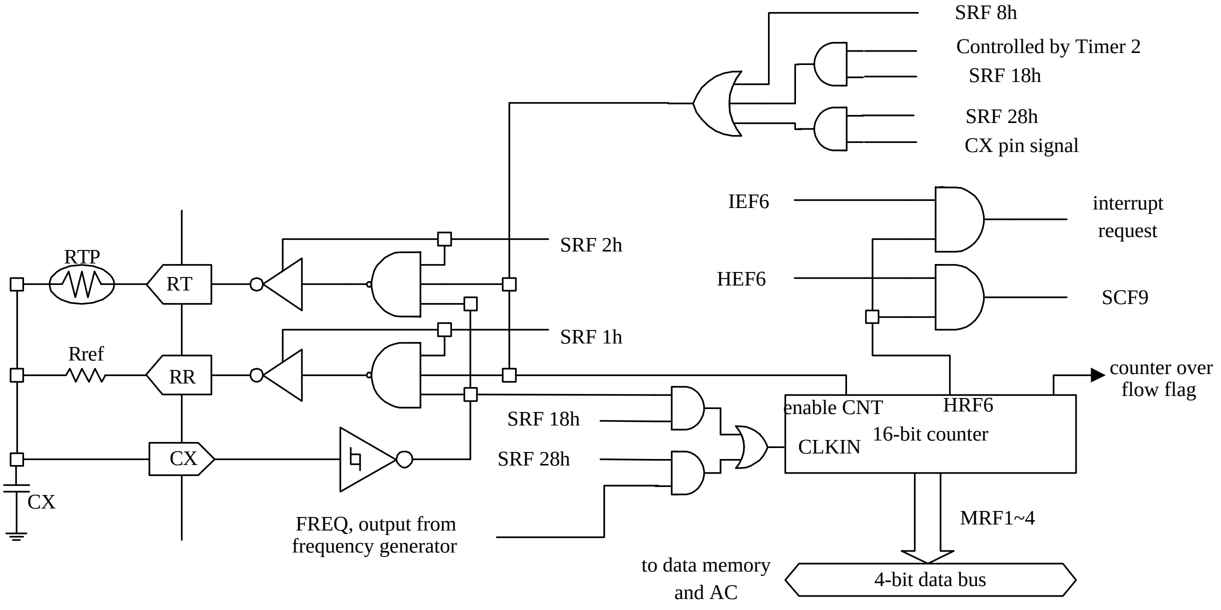
When the signal changes at the input pins of IOA4 port specified by the SCA instruction occur and keep the state for at least two chattering clock (PH6, PH8, PH10) cycles, the control circuit at the input pins will deliver the halt release request signal (SCF0). At that time, the chattering prevention clock will stop due to the delivery of SCF0. The SCF0 will be reset to 0 by executing SCA instruction and the chattering prevention clock will be enabled at the same time. If the SCF0 has been set to 1, the halt release request flag 0 (HRF0) will be delivered. In this case, if the port IOA4 interrupt enable mode (IEF0) is provided, the interrupt is accepted.

Since no flip-flop is available to hold the information of the signal at the input pins IOA4, the input data at the port IOA4 must be read into the RAM immediately after the halt mode is released.

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

3.6 Resister to Frequency Converter (RFC)

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



This RFC contains four external pins:

CX: the oscillation Schemmit trigger input

RR: the reference resister output pin

RT: the temperature sensor output pin

3.6.1 RC Oscillation Network

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

Build up the RC oscillation network:

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

To get a better oscillation clock from the CX pin, activate the output pin for each RC network before the counter is enabled.

The RFC function provides 3 modes for the operation of the 16-bit counter. Each mode will be described in the following sections:

3.6.2 Enable/Disable the Counter by Software

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

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

Example:

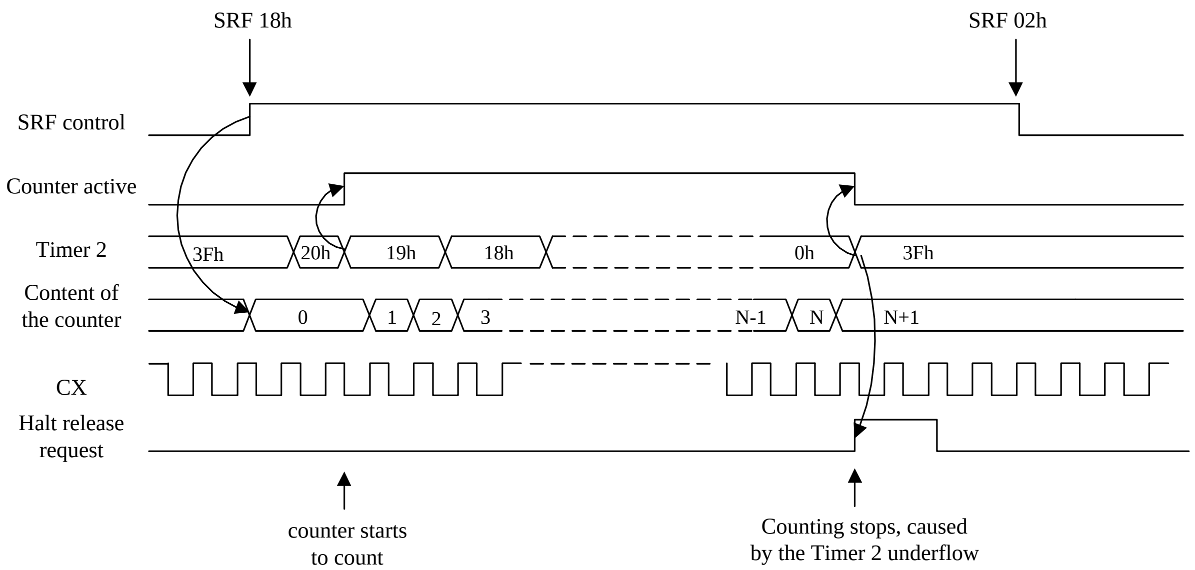
If you intend to count the clock input from the CX pin for a specified time period, you can enable the counter by executing SRF 8 instruction and setting timer1 to control the time period. Check the overflow flag (RFOVF) of this counter when the time period elapses. If the overflow flag is not set to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

```
;Timer 1 is used to enable/disable the counter
        LDS     0, 0            ;Set the TMR2 clock source (PH9)
        LDS     1, 3            ;initiate TMR2 setting value to 3F
        LDS     2, 0Fh
        SHE     2               ;enable halt release by TMR2
RE_CNT:
        LDA     0
        OR*     1               ;combine the TMR2 setting value
        TM2                     2       ;enable the TMR2
        SRF     9               ;build up the RR network and enable the counter
        HALT
        SRF     1               ;stop the counter when TMR2 underflows
        MRF1    10h             ;read the content of the counter
        MRF2    11h
        MRF3    12h
        MRF4    13h
        MSD     20h
        JB2     CNT1_OF         ;check the overflow flag of counter
        JMP     DATA_ACCEPT
CNT1_OF:
        DEC*    2               ;decrease the TM2 value
        LDS     20h, 0
        SBC*    1
        JZ      CHG_CLK_RANGE   ;change the clock source of TMR2
        PLC     10h             ;clear the halt release request flag of TMR2
        JMP     RE_CNT
```

3.6.3 Enable/Disable the Counter by Timer 2

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

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



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

Example:

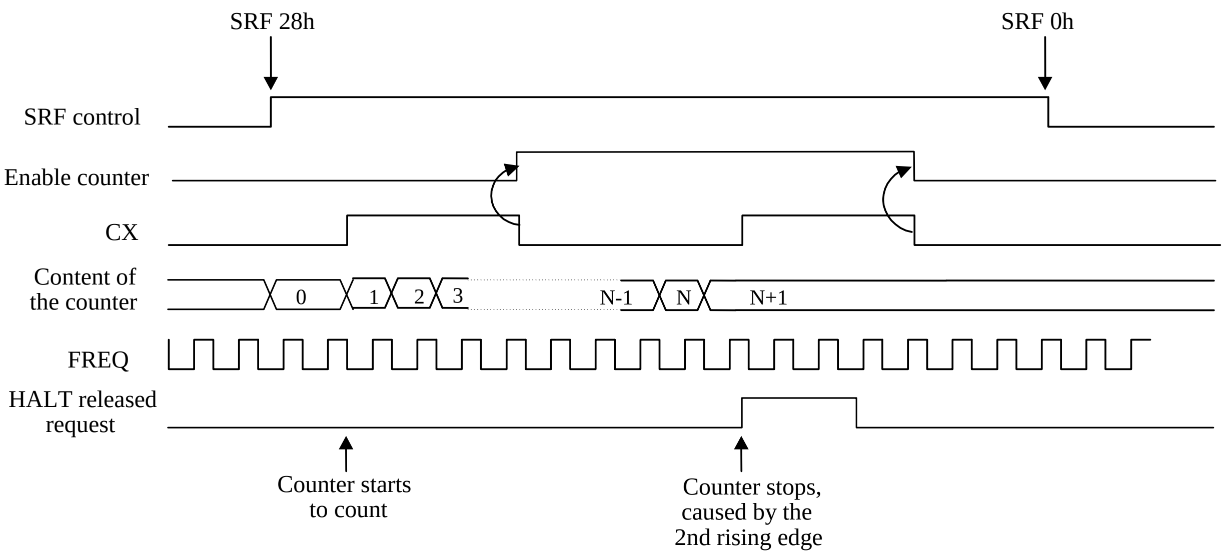
```
; In this example, use the RT network to generate the clock source.
        SRF     1Ah             ;Build up the RT network and enable the counter
                                ;controlled by TMR2
        SHE     10h             ;enable the halt release caused by TMR2
        TM2X    20h             ;set the PH9 as the clock source of TMR2 and the down
                                ;count value is 20h.
        HALT
        PLC     10h             ;Clear the halt release request flag of TMR2
        MRF1    10h             ;read the content of the counter.
        MRF2    11h
        MRF3    12h
        MRF4    13h
```

3.6.4 Enable/Disable the Counter by CX Signal

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

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

Each time the 16-bit counter is enabled; the content of the counter will be cleared automatical.



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

Example:

```
SCC     0h      ;Select the base clock of the frequency generator that comes
                ;from PH0 (XT clock)
FRQX    1, 5    ;set the frequency generator to FREQ= (PH0/6) /3
                ;the setting value of the frequency generator is 5 and FREQ
                ;has 1/3 duty waveform.
SHE     40h     ;enable the halt release caused by 16-bit counter
SRF     28h     ;enable the counter controlled by the CX signal
HALT
PLC     40h     ;halt release is caused by the 2nd rising edge on CX pin and
                ;then clear the halt release request flag
MRF1    10h     ;read the content of the counter
MRF2    11h
MRF3    12h
MRF4    13h
```

4. LCD DRIVER OUTPUT

There are 9 segment pins with 4 common pins in the LCD driver outputs in TM8721. All of 9 segment pins can also be used as DC output ports (through the mask option).

MASK OPTION table:

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

MASK OPTION table:

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

4.1 LCD LIGHTING SYSTEM IN TM8721

TM8721 is 1/2 bias and 1/4 Duty LCD lighting system.

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

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

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

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

4.2 DC OUTPUT

TM8721 permits LCD driver output pins (SEG1~9) to be defined as CMOS type DC output or P open-drain DC output ports by mask option. In these cases, it is possible to use some LCD driver output pins as DC output and the rest of the LCD driver output pins as LCD drivers. Refer to 4-3-4.

The configurations of CMOS output type and P open-drain type are shown below.

When the LCD driver output pins (SEG) are defined as DC output ports, the output data on those ports will not be affected when the program enters stop mode or LCD turn-off mode.

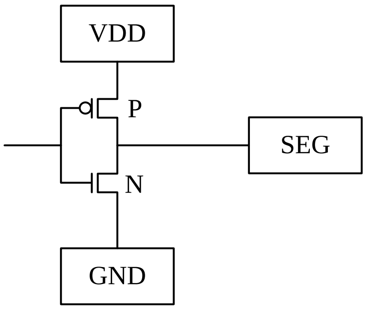


Figure 5-1 CMOS Output Type

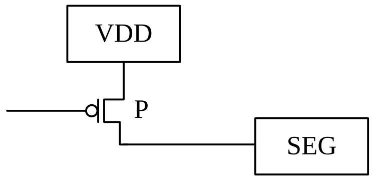
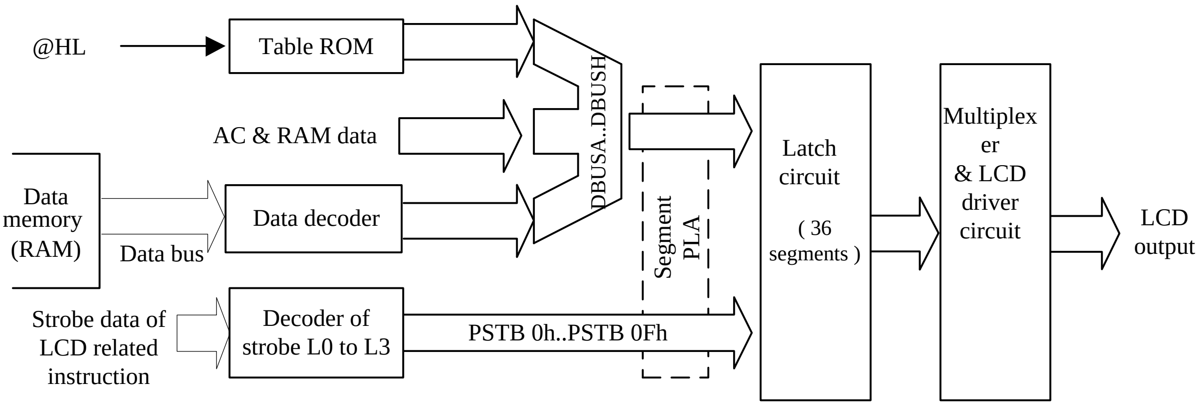


Figure 5-2 P Open-Drain Output Type

4.3 SEGMENT PLA CIRCUIT FOR LCD DISPLAY

4.3.1 PRINCIPLE OF OPERATION OF LCD DRIVER SECTION

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



Principal Drawing of LCD Driver Section

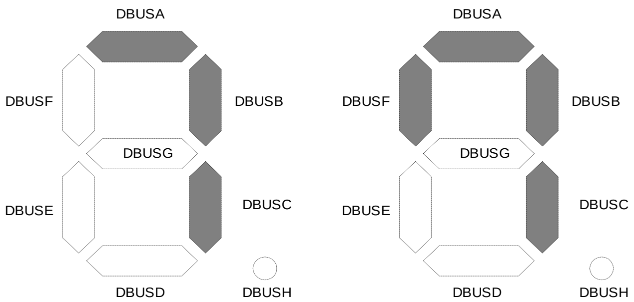
The LCD driver section consists of the following units:

- Data decoder to decode data supplied from RAM or table ROM
- Latch circuit to store LCD lighting information
- L0 to L3 decoder to decode the Lz-specified data in LCD-related instructions which specifies the strobe of the latch circuit
- LCD driver circuitry
- Segment PLA circuit connected between data decoder, L0 to L3 decoder and latch circuit.

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

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

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



DBUSF=0 DBUSF=1

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

MASK OPTION table:

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

Both LCT and LCB instructions use the data-decoder table to decode the content of the specified data memory location. When the content of the data memory location specified by the LCB instruction is "0", the decoded outputs of DBUSA ~ DBUSH are all "0". (This is used for blanking the leading digit "0" on the LCD panel).

The LCP instruction transfers data about the RAM (Rx) and accumulator (AC) directly from "DBUSA" to "DBUSH" without passing through the data decoder.

The LCD instruction transfers the table ROM data (T@HL) directly from "DBUSA" to "DBUSH" without passing through the data decoder.

Table 2- 1 The mapping table of LCP and LCD instructions:

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

There are 8 data decoder outputs from ”DBUSA” to ”DBUSH” and 16 L0 to L3 decoder outputs from PSTB 0h to PSTB 0Fh. The input data and clock signal of the latch circuit are ”DBUSA” to ”DBUSH” and PSTB 0h to PSTB 0Fh, respectively. Each segment pin has 4 latches corresponding to COM1-4.

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 0Fh is selected freely by mask option.

Of the 36 signals obtainable by combining ”DBUSA” to ”DBUSH” and PSTB 0h to PSTB 0Fh, any one of 36 (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 0Fh signals.

Table 2- 2 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
…………	…………….
PSTB0Ah	0AH
PSTB0Bh	0BH
PSTB0Ch	0CH
PSTB0Dh	0DH
PSTB0Eh	0EH
PSTB0Fh	0FH

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

The LCD outputs can be turned off without changing segment data. The execution of the SF2 4h instruction may turn off the displays simultaneously. The execution of the RF2 4h instruction may turn on the display with the patterns turned off. These two instructions will not affect the data stored in the latch circuitry. When executing the RF2 4h instruction to turn off the LCD, the program can still execute LCT, LCB, LCP and LCD instructions to update the data in the latch circuitry. The new content will be outputted to the LCD while the display is being turned on again.

In the stop state, all COM and SEG outputs of LCD drivers will automatically switch to the GND state to avoid DC voltage bias on the LCD panel.

4.3.2 Relative Instructions

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.

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.

LCD Lz, @HL
Transfer 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.

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:

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.

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.

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- 3 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

SF2 4h
Turns off the LCD display.

RF2 4h
Turns on the LCD display.

4.3.3 CONCRETE EXPLANATION

Each LCD driver output corresponds to the LCD 1/4 duty panel and has 4 latches (Refer to Figure: Sample Organization of Segment PLA Option). Since the latch input and the signal to be applied to the clock (strobe) are selected with the segment PLA, the combination of segments in the LCD driver outputs is 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 0Fh outputs is applied to clock CLK.

Refer to Chapter 5 for detailed description of these instructions.

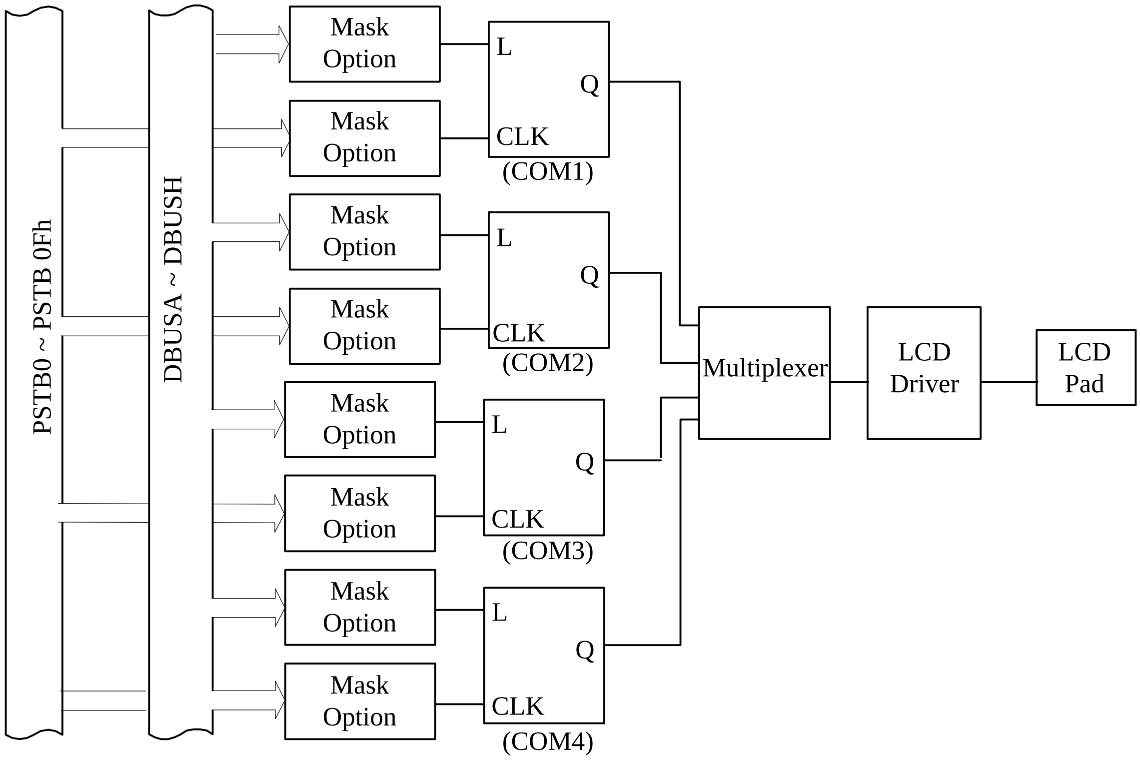


Figure: Sample Organization of Segment PLA Option

4.3.4 THE CONFIGURATION FILE FOR MASK OPTION

When configuring the mask option of LCD PLA, the *.cfg file provides the necessary format for editing the LCD configuration.

The syntax in the *.cfg file is as follows:

SEG COM PSTB DBUS

SEG: Specifies the segment pin No. "1" ~ "9"

COM: Specifies the corresponding latch in each segment pin. Only 0, 1, 2, 3, 4 ~ 10 can be specified in this column. "1" ~ "4" represents COM1 latch ~ COM4 latch respectively. "0" is for CMOS type DC output option and "10" is for P open-drain DC output option.

PSTB: Specifies the strobe data for the latch.

DBUS: Specifies the DBUS data for the latch.

5. Detail Explanation of TM8721 Instructions

- Before using the data memory, it is necessary to initiate the content of data memory because the initial value is unknown.

- The working registers are part of the data memory (RAM), and the relationship between them can be shown as follows:

[The absolute address of working register Rx=Ry+70H]*

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

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

- Lz represents the address of the latch of LCD PLA; the address range specified by Lz is from 00H to 0FH.

5.1 INPUT/OUTPUT INSTRUCTIONS

LCT Lz, Ry

Function: LCD latch Lz ← data decoder ← (Ry)

Description: The working register contents specified by Ry are loaded to the LCD latch specified by Lz through the data decoder.

LCB Lz, Ry

Function: LCD latch Lz ← data decoder ← (Ry)

Description: The working register contents specified by Ry are loaded to the LCD latch specified by Lz through the data decoder.

If the content of Ry is "0", the outputs of the data decoder are all "0".

LCP Lz, Ry

Function: LCD latch Lz ← (Ry), (AC)

Description: The working register contents specified by Ry and the contents of AC are loaded to the LCD latch specified by Lz.

LCD Lz, @HL

Function: LCD latch Lz ← (T@HL)

Description: @HL indicates an index address of table ROM.

The contents of table ROM specified by @HL are loaded to the LCD latch specified by Lz directly.

LCT Lz, @HL

Function: LCD latch Lz ← data decoder ← (R@HL)

Description: The contents of index RAM specified by @HL are loaded to the LCD latch specified by Lz through the data decoder.

LCB Lz, @HL

Function: LCD latch Lz ← data decoder ← (R@HL)

Description: The contents of index RAM specified by @HL are loaded to the LCD latch specified by Lz through the data decoder.

If the content of @HL is "0", the outputs of the data decoder are all "0".

LCP Lz, @HL

Function: LCD latch Lz ← (R@HL), (AC)

Description: The contents of index RAM specified by @HL and the contents of AC are loaded to the LCD latch specified by Lz.

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(X4 X3) is shown below:

Bit pattern	Setting	Bit pattern	Setting
X4=1	Enable IOA4 pull low R	X4=0	Disable IOA4 pull low R
X3=1	IOA4 as output mode	X3=0	IOA4 as input mode

OPA Rx

Function: I/OA ← (Rx)

Description: The content of Rx is outputted to I/OA4 port.

IPA Rx

Function: Rx, AC ← (IOA4)

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

SPB X

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

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

The meaning of each bit of X(X3 X2) 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 outputted to I/OB port.

IPB Rx

Function: Rx, AC ← (IOB)

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

SPC X

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

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

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

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

OPC Rx

Function: I/OC ← (Rx)

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

IPC Rx

Function: Rx, AC ← (IOC)

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

ALM X

Function: Sets buzzer output frequency.

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

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

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

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

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

Notes: 1. FREQ is the output of frequency generator.
2. When the buzzer output does not need the envelope waveform, X5 ~ X0 should be set to 0.
3. The frequency inside the () bases on the ϕ0 is 32768Hz.

SRF X
Function: The operation control for RFC.
Description: The meaning of each control bit(X5 X4 X3 X1 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
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.

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.

SR0 Rx

Function: Rxn, ACn ← Rx(n+1),AC(n+1)

Rx3, AC3 ← 0

Description: The Rx content is shifted right and 0 is loaded to the MSB.
The result is loaded to the AC.
0 → Rx3 → Rx2 → Rx1 → Rx0 →

SR1 Rx

Function: Rxn, ACn ← Rx(n+1),AC(n+1)
Rx3, AC3 ← 1

Description: The Rx content is shifted right and 1 is loaded to the MSB. The result is loaded to the AC.
1 → Rx3 → Rx2 → Rx1 → Rx0 →

SL0 Rx

Function: Rxn, ACn ← Rx(n-1),AC(n-1)
Rx0, AC0 ← 0

Description: The Rx content is shifted left and 0 is loaded to the LSB. The results are loaded to the AC.
← Rx3 ← Rx2 ← Rx1 ← Rx0 ← 0

SL1 Rx

Function: Rxn, ACn ← Rx(n-1),AC(n-1)
Rx0, AC0 ← 1

Description: The Rx content is shifted left and 1 is loaded to the LSB. The results are loaded to the AC.
← Rx3 ← Rx2 ← Rx1 ← Rx0 ← 1

MRA Rx

Function: CF ← (Rx)3

Description: Bit3 of the content of Rx is loaded to carry flag(CF).

MAF Rx

Function: AC, Rx ← CF

Description: The content of CF is loaded to AC and Rx. The content of AC and meaning of bit after execution of this instruction are as follows:
Bit 3 .... CF
Bit 2 .... (AC) =0, zero flag
Bit 1 .... (No Use)
Bit 0 .... (No Use)

5.3 OPERATION INSTRUCTIONS

INC* Rx

Function: Rx,AC ← (Rx) +1

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

INC* @HL

Function: R@HL,AC ← (R@HL) +1

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

DEC* Rx

Function: Rx, AC ← (Rx) -1

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

DEC* @HL

Function: R@HL, AC ← (R@HL) -1

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

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* 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.

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* 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.

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* 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.

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* 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.

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* 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.

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* 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.

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* 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.

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* 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.

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.

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.

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

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

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

LDL* Rx, @HL

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

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

MRF1 Rx

Function: Rx , AC ← RFC[3 ~ 0]

Description: Loads the lowest nibble data of 16-bit counter of RFC to AC and data memory specified by Rx.

Bit 3 ← RFC[3]
Bit 2 ← RFC[2]
Bit 1 ← RFC[1]
Bit 0 ← RFC[0]

MRF2 Rx

Function: Rx , AC ← RFC[7 ~ 4]

Description: Loads the 2nd nibble data of 16-bit counter of RFC to AC and data memory specified by Rx.

Bit 3 ← RFC[7]
Bit 2 ← RFC[6]
Bit 1 ← RFC[5]
Bit 0 ← RFC[4]

MRF3 Rx

Function: Rx , AC ← RFC[11 ~ 8]

Description: Loads the 3rd nibble data of 16-bit counter of RFC to AC and data memory specified by Rx.

Bit 3 ← RFC[11]
Bit 2 ← RFC[10]
Bit 1 ← RFC[9]
Bit 0 ← RFC[8]

MRF4 Rx

Function: Rx , AC ← RFC[15 ~ 12]

Description: Loads the highest nibble data of 16-bit counter of RFC to AC and data memory specified by Rx.

Bit 3 ← RFC[15]
Bit 2 ← RFC[14]
Bit 1 ← RFC[13]
Bit 0 ← RFC[12]

5.5 CPU CONTROL INSTRUCTIONS

NOP
Function: no operation
Description: no operation

HALT
Function: Enters halt mode
Description: The following 3 conditions cause the halt mode to be released.
1). The signal change specified by the SCA instruction is applied to IOA4.
2). The halt release condition specified by SHE instruction is met.
When an interrupt is accepted to release the halt mode, the halt mode returns by executing the RTS instruction after completion of interrupt service.

STOP
Function: Enters stop mode and stops all oscillators
Description: Before executing this instruction, all signals on IOA4 port must be set to low.
The following 3 conditions cause the stop mode to be released.
1). IOA4 port is "H".

SCA X
Function: The data specified by X causes the halt mode to be released.
Description: The signal change at port IOA is specified.
The bit meaning of X(X4) is shown below:

Bit pattern	Description
X4=1	Halt mode is released when signal applied to IOA4

X7~5,X3~0 is reserved

SIE* X
Function: Set/Reset interrupt enable flag
Description:

X0=1	The IEF0 is set so that interrupt 0(Signal change at port IOA4 specified by SCA) is accepted.
X3=1	The IEF3 is set so that interrupt 3(overflow from the predivider) is accepted.
X4=1	The IEF4 is set so that interrupt 4(underflow from timer 2) is accepted.
X6=1	The IEF6 is set so that interrupt 6(overflow from the RFC counter) is accepted.

X7, 5, 2, 1 is reserved

SHE X
Function: Set/Reset halt release enable flag
Description:

X3=1	The HEF3 is set so that the halt mode is released by predivider overflow.
X4=1	The HEF4 is set so that the halt mode is released by TMR2 underflow.
X6=1	The HEF6 is set so that the halt mode is released by RFC counter overflow.

X7, 5, 2, 1, 0 is reserved

SRE X

Function: Set/Reset stop release enable flag

Description:

X4=1	The SRF6 is set so that the stop mode is released by the signal changed on IOA4 port.

X7, X5~0 is reserved

MSB Rx

Function: AC, Rx ← 0, SCF2, 0, 0

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

Bit 3	Bit 2	Bit 1	Bit 0
NA	Start condition flag 2 (SCF2)	NA	NA
NA	Halt release caused by SCF4,5,6,7,8,9	NA	NA

MSC Rx

Function: AC, Rx ← SCF7, PH15, 0, 0

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

Bit 3	Bit 2	Bit 1	Bit 0
Start condition flag 7 (SCF7)	The content of 15th stage of the predivider		
Halt release caused by predivider overflow			

MCX Rx

Function: AC, Rx ← SCF9, SCF0, SCF6, 0

Description: The SCF9, 0 , 6 contents are loaded to AC and the data memory specified by Rx.
The content of AC and meaning of bit after execution of this instruction are as follows:

Bit 3	Bit 2	Bit 1	Bit 0
Start condition flag 9 (SCF9)	Start condition flag 0 (SCF0)	Start condition flag 6 (SCF6)	NA
Halt release caused by RFC counter overflow	Halt release caused by IOA4	Halt release caused by TM2 underflow	NA

MSD Rx

Function: Rx, AC ← 0, RFOVF, 0, 0

Description: The overflow flag of RFC counter is loaded to data memory specified by Rx and AC.
The content of AC and meaning of bit after execution of this instruction are as follows:

Bit 3	Bit 2	Bit 1	Bit 0
NA	The overflow flag of 16-bit counter of RFC (RFVOF)	NA	NA

5.6 INDEX ADDRESS INSTRUCTIONS

MVH Rx

Function: (@H) ← (Rx &AC)

Description: Loads content of Rx to higher nibble of index address buffer @H.
H7=[AC]3, H6=[AC]2, H5=[AC]1, H4=[AC]0,
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

5.7 DECIMAL ARITHMETIC INSTRUCTIONS

DAA

Function: AC ← BCD(AC)

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

DAA* Rx

Function: AC, Rx ← BCD(AC)

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

DAA* @HL

Function: AC,R@HL ← BCD(AC)

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

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

DAS

Function: AC ← BCD[AC]

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

DAS* Rx

Function: AC, Rx ← BCD(AC)

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

DAS* @HL

Function: AC, @HL ← BCD[AC]

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

5.8 JUMP INSTRUCTIONS

JB0 X

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

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

JB1 X

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

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

JB2 X

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

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

JB3 X

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

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

JNZ X

Function: Program counter jumps to X if (AC)! =0.

Description: If the content of AC is not 0, jump occurs.
If 0, the PC increases by 1.
The range of X is from 000H to 7FFH or 800H to FFFH.

JNC X

Function: Program counter jumps to X if CF=0.

Description: If the content of CF is 0, jump occurs.
If 1, the PC increases by 1.
he range of X is from 000H to 7FFH or 800H to FFFH.

JZ X

Function: Program counter jumps to X if (AC) =0.

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

JC X

Function: Program counter jumps to X if CF=1.

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

JMP X

Function: Program counter jumps to X.

Description: Unconditional jump.
The range of X is from 000H to FFFH.

CALL X

Function: STACK ← (PC) +1
Program counter jumps to X.

Description: A subroutine is called.
The range of X is from 000H to 3FFH.

RTS

Function: PC ← (STACK)

Description: A return from a subroutine occurs.

5.9 MISCELLANEOUS INSTRUCTIONS

SCC X

Function: Setting the clock source for IOA4 hattering prevention, PWM output and frequency generator.

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

Bit pattern	Clock source setting	Bit pattern	Clock source setting
X6=1	The clock source comes from the system clock(BCLK).	X6=0	The clock source comes from the ϕ0. Refer to section 3-3-4 for ϕ0.
(X2,X1,X0) =001	Chattering prevention clock= ϕ10	(X2,X1,X0) =010	Chattering prevention clock= ϕ8
(X2,X1,X0) =100	Chattering clock = ϕ6		

X7,5,4,3 is reserved

FRQ D, Rx

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

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

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

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

FRQ D, @HL

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

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

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

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

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

FRQX D, X

Function: Frequency generator ← D, X

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

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

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

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

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

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.

X7~1 is reserved

RF X

Function: Resets flag

Description: Description of each flag

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

X7~1 is reserved

SF2 X

Function: Sets flag

Description: Description of each flag

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.

X7~3 is reserved

RF2 X

Function: Resets flag

Description: Description of each flag

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.

X7~3 is reserved

PLC

Function: Pulse control

Description: The pulse corresponding to the data specified by X is generated.

X0: "1" Halt release request flag HRF0 caused by the signal at I/O port C is reset.

X3: "1" Halt release request flag HRF3 caused by overflow from the predivider is reset.

X4: "1" Halt release request flag HRF4 caused by underflow from the timer 2 is reset and stops the operating of timer 2(TM2).

X6: "1" Halt release request flag HRF6 caused by overflow from the RFC counter is reset.

X8: "1" The last 5 bits of the predivider (15 bits) are reset. When executing this instruction, X3 must be set to "1".

X7, 5, 2, 1 is reserved

ORDERING INFORMATION

The ordering information:

Ordering number	Package
TM8721-COD	Wafer/Dice with code

Appendix A TM8721 Instruction Table

Instruction		Machine Code	Function		Flag/Remark
NOP		0000 0000 0000 0000	No Operation		
LCT	Lz, Ry	0000 0010 ZZZZ YYYY	Lz	← (7SEG ← Ry)	
LCB	Lz, Ry	0000 0100 ZZZZ YYYY	Lz	← (7SEG ← Ry)	Blank Zero
LCP	Lz, Ry	0000 0110 ZZZZ YYYY	Lz	← Ry & AC	
LCD	Lz, @HL	0000 1000 ZZZZ 0000	Lz	← T@HL	
LCT	Lz, @HL	0000 1000 ZZZZ 0001	Lz	← (7SEG ← @HL)	
LCB	Lz, @HL	0000 1000 ZZZZ 0010	Lz	← (7SEG ← @HL)	Blank Zero
LCP	Lz, @HL	0000 1000 ZZZZ 0011	Lz	← @HL & AC	
OPA	Rx	0000 1010 01XX XXXX	PortA (IOA4)	← Rx	
OPB	Rx	0000 1100 01XX XXXX	PortB (IOB)	← Rx	
OPC	Rx	0000 1101 01XX XXXX	PortC (IOC)	← Rx	
FRQ	D,Rx	0001 00DD 01XX 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 01XX XXXX	@L	← Rx	
MVH	Rx	0001 1101 01XX XXXX	@H	← Rx & AC	
ADC	Rx	0010 0000 01XX XXXX	AC	← Rx+AC+CF	CF
ADC	@HL	0010 0000 1000 0000	AC	← @HL+AC+CF	CF
ADC*	Rx	0010 0001 01XX XXXX	AC, Rx	← Rx+AC+CF	CF
ADC*	@HL	0010 0001 1000 0000	AC, @HL	← @HL+AC+CF	CF
SBC	Rx	0010 0010 01XX XXXX	AC	← Rx+ACB+CF	CF
SBC	@HL	0010 0010 1000 0000	AC	← @HL+ACB+CF	CF
SBC*	Rx	0010 0011 01XX XXXX	AC, Rx	← Rx+ACB+CF	CF
SBC*	@HL	0010 0011 1000 0000	AC, @HL	← @HL+ACB+CF	CF
ADD	Rx	0010 0100 01XX XXXX	AC	← Rx+AC	CF
ADD	@HL	0010 0100 1000 0000	AC	← @HL+AC	CF
ADD*	Rx	0010 0101 01XX XXXX	AC, Rx	← Rx+AC	CF
ADD*	@HL	0010 0101 1000 0000	AC, @HL	← @HL+AC	CF
SUB	Rx	0010 0110 01XX XXXX	AC	← Rx+ACB+1	CF
SUB	@HL	0010 0110 1000 0000	AC	← @HL+ACB+1	CF
SUB*	Rx	0010 0111 01XX XXXX	AC, Rx	← Rx+ACB+1	CF
SUB*	@HL	0010 0111 1000 0000	AC, @HL	← @HL+ACB+1	CF
ADN	Rx	0010 1000 01XX XXXX	AC	← Rx+AC	
ADN	@HL	0010 1000 1000 0000	AC	← @HL+AC	
ADN*	Rx	0010 1001 01XX XXXX	AC, Rx	← Rx+AC	
ADN*	@HL	0010 1001 1000 0000	AC, @HL	← @HL+AC	
AND	Rx	0010 1010 01XX XXXX	AC	← Rx AND AC	
AND	@HL	0010 1010 1000 0000	AC	← @HL AND AC	

Instruction		Machine Code	Function		Flag/Remark
AND*	Rx	0010 1011 01XX XXXX	AC, Rx	← Rx AND AC	
AND*	@HL	0010 1011 1000 0000	AC, @HL	← @HL AND AC	
EOR	Rx	0010 1100 01XX XXXX	AC	← Rx EOR AC	
EOR	@HL	0010 1100 1000 0000	AC	← @HL EOR AC	
EOR*	Rx	0010 1101 01XX XXXX	AC, Rx	← Rx EOR AC	
EOR*	@HL	0010 1101 1000 0000	AC,@HL	← @HL EOR AC	
OR	Rx	0010 1110 01XX XXXX	AC	← Rx OR AC	
OR	@HL	0010 1110 1000 0000	AC	← @HL OR AC	
OR*	Rx	0010 1111 01XX XXXX	AC, Rx	← Rx OR AC	
OR*	@HL	0010 1111 1000 0000	AC, @HL	← @HL OR AC	
ADCI	Ry, D	0011 0000 DDDD YYYY	AC	← Ry+D+CF	CF
ADCI*	Ry, D	0011 0001 DDDD YYYY	AC, Ry	← Ry+D+CF	CF
SBCI	Ry, D	0011 0010 DDDD YYYY	AC	← Ry+DB+CF	CF
SBCI*	Ry, D	0011 0011 DDDD YYYY	AC, Ry	← Ry+DB+CF	CF
ADDI	Ry, D	0011 0100 DDDD YYYY	AC	← Ry+D	CF
ADDI*	Ry, D	0011 0101 DDDD YYYY	AC, Ry	← Ry+D	CF
SUBI	Ry, D	0011 0110 DDDD YYYY	AC	← Ry+DB+1	CF
SUBI*	Ry, D	0011 0111 DDDD YYYY	AC, Ry	← Ry+DB+1	CF
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 01XX XXXX	AC, Rx	← Rx+1	CF
INC*	@HL	0100 0000 1000 0000	AC, @HL	← @HL+1	CF
DEC*	Rx	0100 0001 01XX XXXX	AC, Rx	← Rx - 1	CF
DEC*	@HL	0100 0001 1000 0000	AC, @HL	← @HL - 1	CF
IPA	Rx	0100 0010 01XX XXXX	AC, Rx	← PortA (IOA4)	
IPB	Rx	0100 0100 01XX XXXX	AC, Rx	← PortB (IOB4~1)	
IPC	Rx	0100 0111 01XX XXXX	AC, Rx	← PortC (IOC4~1)	
MAF	Rx	0100 1010 01XX XXXX	AC, Rx	← STS1	B3: CF B2: ZERO B1: (Unused) B0: (Unused)
MSB	Rx	0100 1011 01XX XXXX	AC,Rx	← STS2	B3: (Unused) B2: SCF2(HRx) B1: (Unused) B0: (Unused)
MSC	Rx	0100 1100 01XX XXXX	AC, Rx	← STS3	B3: SCF7 (PDV) B2: PH15 B1: (Unused) B0: (Unused)
MCX	Rx	0100 1101 01XX XXXX	AC, Rx	← STS3X	B3: SCF9 (RFC)

Instruction		Machine Code	Function		Flag/Remark
					B2: SCF0 (APT) B1: SCF6 (TM2) B0: (Unused)
MSD	Rx	0100 1110 01XX XXXX	AC, Rx	← STS4	B3: (Unused) B2: RFOVF B1: (Unused) B0: (Unused)
SR0	Rx	0101 0000 01XX XXXX	ACn, Rxn AC3, Rx3	← Rx (n+1) ← 0	
SR1	Rx	0101 0001 01XX XXXX	ACn, Rxn AC3, Rx3	← Rx (n+1) ← 1	
SL0	Rx	0101 0010 01XX XXXX	ACn, Rxn AC0, Rx0	← Rx (n-1) ← 0	
SL1	Rx	0101 0011 01XX XXXX	ACn, Rxn AC0, Rx0	← Rx (n-1) ← 1	
DAA		0101 0100 0000 0000	AC	← BCD (AC)	
DAA*	Rx	0101 0101 01XX XXXX	AC, Rx	← BCD (AC)	
DAA*	@HL	0101 0101 1000 0000	AC, @HL	← BCD (AC)	
DAS		0101 0110 0000 0000	AC	← BCD (AC)	
DAS*	Rx	0101 0111 01XX XXXX	AC, Rx	← BCD (AC)	
DAS*	@HL	0101 0111 1000 0000	AC, @HL	← BCD (AC)	
LDS	Rx, D	0101 1DDD D1XX XXXX	AC, Rx	← D	
LDH	Rx, @HL	0110 0000 01XX XXXX	AC, Rx	← H (T@HL)	
LDH*	Rx, @HL	0110 0001 01XX XXXX	AC, Rx HL	← H (T@HL) ← HL+1	
LDL	Rx, @HL	0110 0010 01XX XXXX	AC, Rx	← L (T@HL)	
LDL*	Rx, @HL	0110 0011 01XX XXXX	AC, Rx HL	← L (T@HL) ← HL+1	
MRF1	Rx	0110 0100 01XX XXXX	AC, Rx	← RFC3-0	
MRF2	Rx	0110 0101 01XX XXXX	AC, Rx	← RFC7-4	
MRF3	Rx	0110 0110 01XX XXXX	AC, Rx	← RFC11-8	
MRF4	Rx	0110 0111 01XX XXXX	AC, Rx	← RFC15-12	
STA	Rx	0110 1000 01XX XXXX	Rx	← AC	
STA	@HL	0110 1000 1000 0000	@HL	← AC	
LDA	Rx	0110 1100 01XX XXXX	AC	← Rx	
LDA	@HL	0110 1100 1000 0000	AC	← @HL	
MRA	Rx	0110 1101 01XX XXXX	CF	← Rx3	
MRW	@HL, Rx	0110 1110 01XX XXXX	AC, @HL	← Rx	
MWR	Rx, @HL	0110 1111 01XX XXXX	AC, Rx	← @HL	
MRW	Ry, Rx	0111 0YYY Y1XX XXXX	AC, Ry	← Rx	
MWR	Rx, Ry	0111 1YYY Y1XX XXXX	AC, Rx	← Ry	
JB0	X	1000 00XX XXXX XXXX	PC	← X	if AC0=1
JB1	X	1000 10XX XXXX XXXX	PC	← X	if AC1=1
JB2	X	1001 00XX XXXX XXXX	PC	← X	if AC2=1
JB3	X	1001 10XX XXXX XXXX	PC	← X	if AC3=1
JNZ	X	1010 00XX XXXX XXXX	PC	← X	if AC≠0
JNC	X	1010 10XX XXXX XXXX	PC	← X	if CF=0
JZ	X	1011 00XX XXXX XXXX	PC	← X	if AC=0

Instruction		Machine Code	Function		Flag/Remark
JC	X	1011 10XX XXXX XXXX	PC	← X	if CF=1
CALL	X	1100 00XX XXXX XXXX	STACK PC	← PC + 1 ← X	
JMP	X	1101 00XX XXXX XXXX	PC	← X	
RTS		1101 1000 0000 0000	PC	← STACK	CALL Return
SCC	X	1101 1001 0X10 0XXX	X6=1 X6=0 X2, 1, 0=001 X2, 1, 0=010 X2, 1, 0=100	: Cfq=BCLK : Cfq=PH0 : Cch=PH10 : Cch=PH8 : Cch=PH6	
SCA	X	1101 1010 00X0 0000	X5	: Enable SEF5	
SPA	X	1101 1100 000X X111	X4 X3	: Set A4 Pull-Low : Set A4 I/O	1: Output, 0: Input
SPB	X	1101 1101 000X XX01	X4 X3~0	: Set B4-3 Pull-Low : Set B4-3 I/O	1: Output, 0: Input
SPC	X	1101 1110 000X XXXX	X4 X3-0	: Set C4-1 Pull-Low /Low-Level-Hold : Set C4-1 I/O	1: Output, 0: Input X: 0000 0000b~ 000x xxxxb
TM2	Rx	1110 0100 01XX 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 X000	X6 X4 X3	: Enable HEF6 : Enable HEF4 : Enable HEF3	RFC TMR2 PDV
SIE*	X	1110 1001 0X0X X00X	X6 X4 X3 X0	: Enable IEF6 : Enable IEF4 : Enable IEF3 : Enable IEF0	RFC TMR2 PDV APT
PLC	X	1110 101X 0X0X X00X	X8 X6 X4 X3 X0	: Reset PH15~11 : Reset HRF6 : Reset HRF4 : Reset HRF3 : Reset HRF0	 RFC TMR2 PDV APT
SRF	X	1110 1100 00XX X0XX	X5 X4 X3 X1 X0	: Enable Cx Control : Enable TM2 Control : Enable Counter : Enable RT Output : Enable RR Output	 ENX ETP ERR
SRE	X	1110 1101 0X00 0000	X6	: Enable SRF6	SRF6 (APT)
SF		1111 0000 0000 0001	X0	: CF Set	CF
RF		1111 0100 0000 0001	X0	: CF Reset	CF
SF2	X	1111 1000 0000 0XXX	X2 X1 X0	: Close all Segments : Dis-ENX Set : Reload 2 Set	RSOFF DED RL2
RF2	X	1111 1001 0000 0XXX	X2 X1	: Release Segments : Dis-ENX Reset	RSOFF DED

Instruction		Machine Code	Function		Flag/Remark
			X0	: Reload 2 Reset	RL2
ALM	X	1111 101X 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	
HALT		1111 1110 0000 0000	Halt Operation		
STOP		1111 1111 0000 0000	Stop Operation		

NOTE: Rx: 40~7Fh

Symbol Description

AC	: Accumulator
ACn	: Accumulator bit n
X	: Address
Rx	: Memory of address X
Rxn	: Memory bit n of address X
Ry	: Memory of working register Y
HRFn	: HALT Release Flag
HEFn	: HALT Release Enable Flag
PDV	: Pre-Divider
Lz	: LCD Latch
@HL	: Address of Index
@L	: Low address of Index
@H	: High address of Index
L (T@HL)	: Low Nibble of Index ROM
H (T@HL)	: High Nibble of Index ROM
T@HL	: Address of Index ROM
RFOVF	: RFC Overflow Flag
D	: Immediate Data
PC	: Program Counter
CF	: Carry Flag
ZERO	: Zero Flag
HL	: Index Register
BCLK	: System clock, stop only in STOP condition
IEFn	: Interrupt Enable Flag
SRFn	: STOP Release Enable Flag
SCFn	: Start Condition Flag
Cch	: Clock Source of Chattering Detector
Cfq	: Clock Source of Frequency Generator
SEFn	: Switch Enable Flag
FREQ	: Frequency Generator setting Value
( )	: Content of Register
TMR	: Timer Overflow Release Flag
Ctm	: Clock Source of Timer