1.6 VERSION

ELAN MICROELECTRONICS CORPORATION

Preliminary

USER'S MANUAL

EM73PE00

4-Bit Microcontroller

ELAN MICROELECTRONICS CORPORATION

Hsinchu Headquarters
7F-1, 9 Prosperity Road I
Science-based Industry Park
Hsinchu 30077, Taiwan, ROC
Phone : +886 3 578-7505
Fax : +886 3 577-9095

Taipei Office
3F, No. 178, Sec. 2, Min
Sheng East Road., Taipei,
Taiwan, R.O.C.
Phone : +886 2 2522-3065
Fax : +886 2 2551-6348

Hong Kong Office
Rm. 1005B, 10/F., Empire
Centre, 68 Mody Road.,
Tsimshatsui, Kowloon, Hong
Kong
Phone : +852 2723-3376
Fax : +852 2723-7780

Trademark Acknowledgments

IBM is a registered trademark and PS/2 is a trademark of IBM.

Microsoft, MS, MS-DOS, and Windows are registered trademarks of Microsoft Corporation.

Printed in Taiwan, ROC 4/2000

Chapter 1

Introduction

1.1 Overview

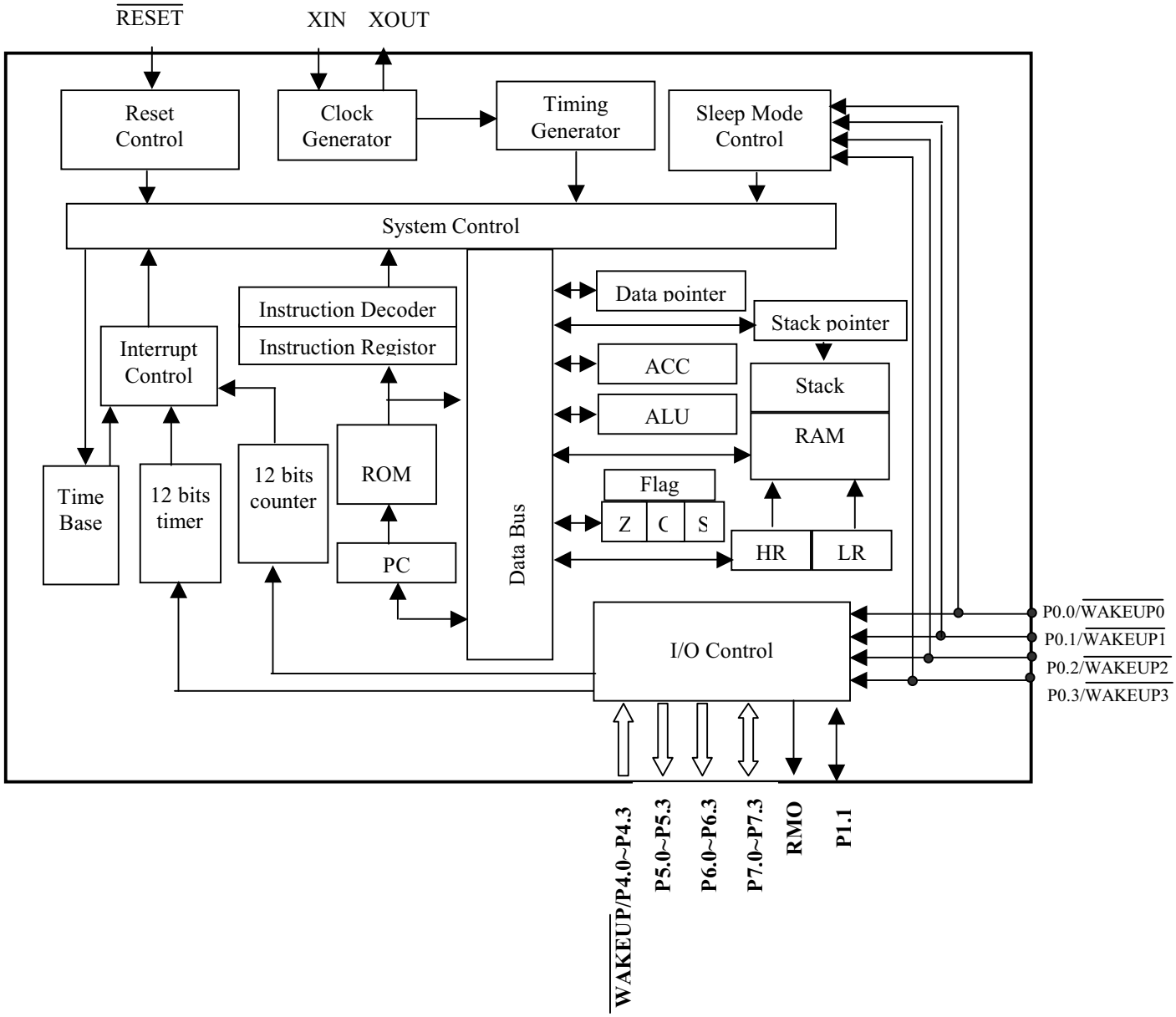
EM73PE00 is an advanced single chip CMOS 4-bit OTP(One Time Programming) microcontroller. It contains 24K-byte ROM, 128-nibble RAM, 4-bit ALU, 16-level subroutine nesting, 13-bit programmable counter, 8-bit watchdog timer, and 13-bit programmable timer on one chip.

1.2 Key Features

- Si gate CMOS process
- Low power consumption
- Single power supply : 2.5V to 5.5V.
- High-speed operation : 4MHz max.
- ROM capacity : 24576 X 8 bits.
- RAM capacity : 128 X 4 bits.
- Built-in time base counter : 22 stages.
- I/P : 8 bits input port
- O/P: 8 bits output port
- I/O: 5 bits bi-direction I/O port
- Programmable counter and timer
- Built-in watchdog timer
- Low voltage detector
- Subroutine nesting : Up to 16 levels.
- Interrupt : Three (Internal)
- Standby function
- Instruction execution time : 1us @4MHz
- Packaging : EM73PE00/01/02AM, EM73PE00/01/02GM 28-pin SOP
- RMO : carrier wave output port of remote controller

1.3 Functional Block Diagram

The figure below illustrates a simplified block diagram of the EM73PE00 followed by pin assignments description.



EM73PE00 Block Diagram

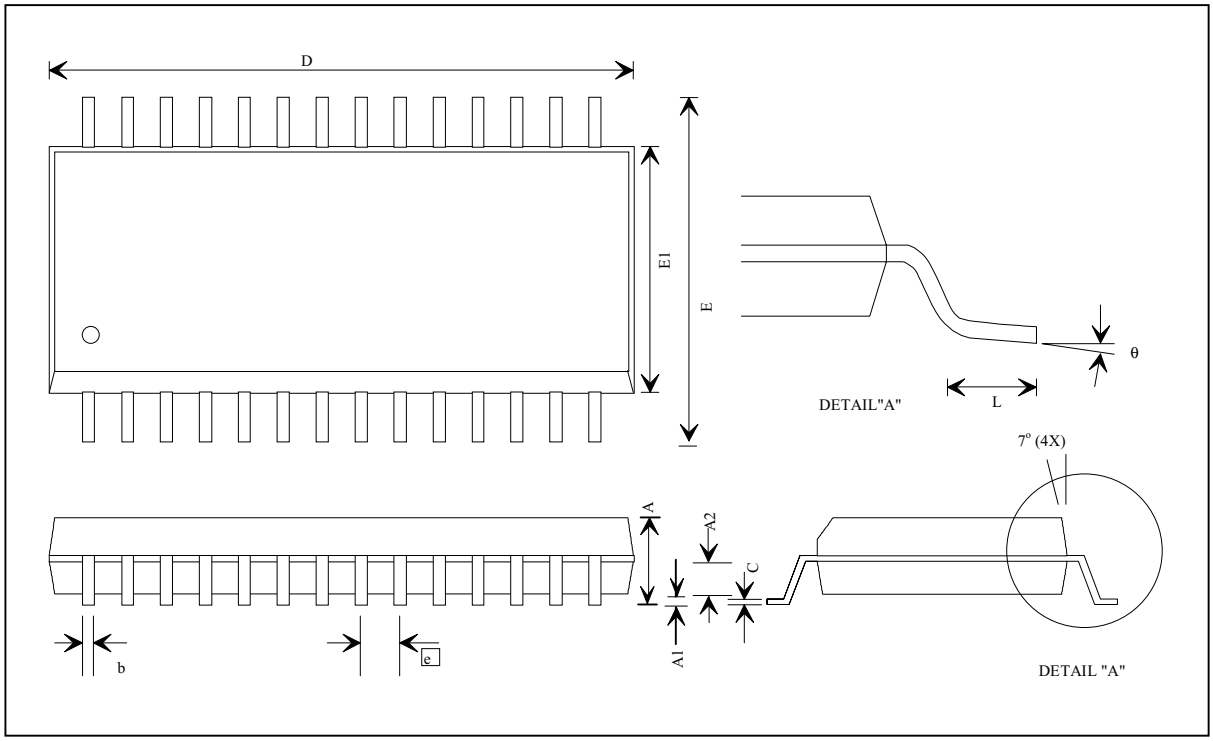
1.4 Pin Assignments

TOP view

P52	1	28	P51
P53	2	27	P50
P60	3	26	P43
P61	4	25	P42
P62	5	24	P41
P63	6	23	P40
P11	7	22	P03 (DCLK)
RMO	8	21	P02 (OEB)
(VPP)TEST	9	20	P01 (PGMB)
(VDD)VDD	10	19	P00 (ALCK)
(Xin)Xin	11	18	P73
Xout	12	17	P72
(GND) Vss	13	16	P71 (DOUT)
(RESET) RESET	14	15	P70 (DIN)

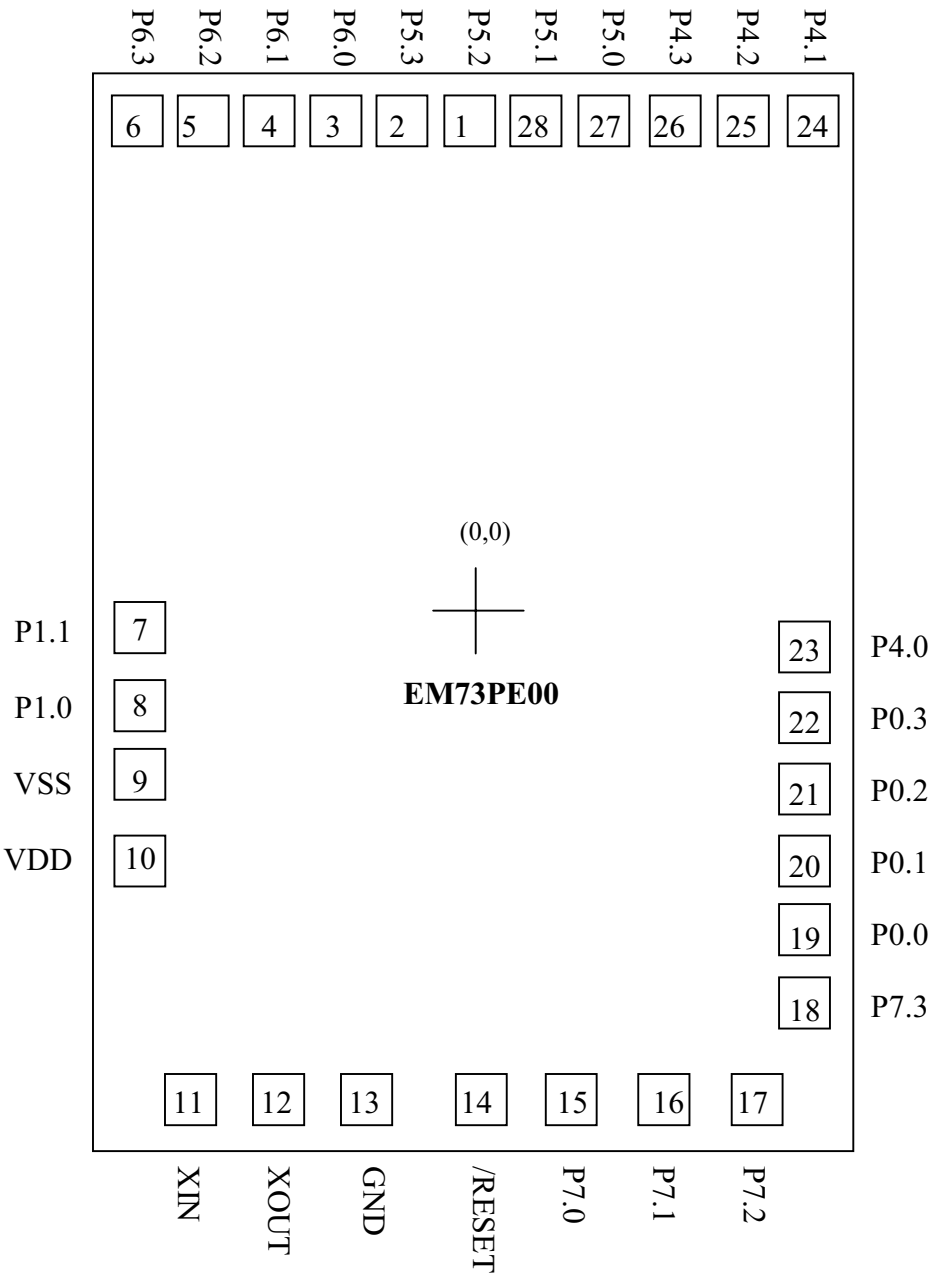
(*): Progrmming OTP signal lines

Physical Dimension of SOP28



Symbol	Dimension in mils			Dimension in mm		
	Min	Nom	Max	Min	Nom	Max
A	92	—	106	2.337	—	2.692
A1	4	—	12	0.102	—	0.305
A2	39	41	43	0.991	1.041	1.092
b	13	16	20	0.330	0.406	0.508
C	9	10	12.5	0.229	0.254	0.318
D	697	705	713	17.704	17.907	18.110
E1	291	295	299	7.392	7.493	7.595
e	—	50	—	—	1.270	—
E	394	410	419	10.008	10.414	10.643
L	16	—	50	0.406	—	1.27
θ	0 °	5 °	8 °	0 °	5 °	8 °

1.5 Pad Diagram



Pad No.	Symbol	X	Y
1	P5.2	5.5	1604.6
2	P5.3	-115.6	1604.6
3	P6.0	-236.7	1604.6
4	P6.1	-357.8	1604.6
5	P6.2	-478.9	1604.6
6	P6.3	-600.0	1604.6
7	P1.1	-610.0	-489.8
8	P1.0	-610.0	-686.4
9	TEST	-610.0	-833.4
10	VDD	-610.0	-1058.6
11	XIN	-537.5	-1625.0
12	XOUT	-365.7	-1625.0
13	VSS	-186.2	-1625.0
14	/RESET	38.6	-1625.0
15	P7.0	210.4	-1625.0
16	P7.1	382.2	-1625.0
17	P7.2	554.0	-1625.0
18	P7.3	610.0	-1351.9
19	P0.0	610.0	-1180.1
20	P0.1	610.0	-1008.3
21	P0.2	610.0	-836.5
22	P0.3	610.0	-664.7
23	P4.0	610.0	-492.9
24	P4.1	612.1	1604.6
25	P4.2	489.9	1604.6
26	P4.3	368.8	1604.6
27	P5.0	247.7	1604.6
28	P5.1	126.6	1604.6

Chip size : 1520 * 3550 um

For PCB layout, IC substrate must be floated or connected to Vss.

1.6 Pin Descriptions

Symbol	Pin-Type	Function	Mask Options
V_{DD}	-	1. Power supply (+) 2. Power supply (+) for programming OTP.	-
V_{SS}	-	1. Power supply (-) 2. Power supply (-) for programming OTP.	-
TEST	-	1. Test pin, high active. 2. VPP:high voltage(12V) power source for programming OTP.	Pull-down resistor
$\overline{\text{RESET}}$	RESET-A	1. System reset input signal, low active 2. RESET: reset input signal for programming OTP.	Pull-up resistor Low voltage detector enable
XIN / OSCin	OSC-A OSC-H	1. Crystal or RC or external clock source connecting pin 2. Xin: oscillator signal input for programming OTP.	Crystal or RC osc type
XOUT	OSC-A	Crystal connecting pin	-
RMO	I/O-RMO	Remote control signal output terminal	-
P7(0..3)	I/O	4-bit bi-direction I/O port P7.1/DOUT: data output for programming OTP. P7.0/DIN: data input for programming OTP.	Output : open-drain or push-pull Input : pull-up resistor or none
P1.1		1-bit bi-direction I/O port	
P0(0..3)	INPUT	4-bit input port P0.0/ALCK: address counter clock for programming OTP. P0.1/PGMB: program data to OTP cells for programming OTP. P0.2/OEB: data output enable for programming OTP. P0.3/DCLK: data in/out clock signal for programming OTP.	Pull-up/pull-down resistor Wakeup
P4(0..3)	INPUT	4-bit input port	Pull-up/pull-down resistor Wakeup
P5(0..3)	O/P	4 bit output port	Push-pull or open-drain
P6(0..3)	O/P	4 bit output port	Push-pull or open-drain

1.7 Instruction Set Overview

The instruction set is optimized to support those instructions most commonly used or executed.

1.7.1 Data Transfer

Mnemonic	Object Code (Binary)	Operation Description	Byte	Cycle	Flag		
					CF	ZF	SF
LDA X	0110 1010 xxxx xxxx	Acc←RAM[x]	2	2	-	Z	1
LDAM	0101 1010	Acc←RAM[HL]	1	1	-	Z	1
LDAX	0110 0101	Acc←ROM[DP]$_L$	1	3	-	Z	1
LDAXI	0110 0111	Acc←ROM[DP]$_H$, DP+1	1	3	-	Z	1
LDH #k	1001 kkkk	HR←k	1	1	-	-	1
LDHL x	0100 1110 xxxx xxx0	LR←RAM[x],HR←RAM[x+1]	2	2	-	-	1
LDIA #k	1101 kkkk	Acc←k	1	1	-	Z	1
LDL #k	1000 kkkk	LR←k	1	1	-	-	1
STA x	0110 1001 xxxx xxxx	RAM[x]←Acc	2	2	-	-	1
STAM	0101 1001	RAM[HL]←Acc	1	1	-	-	1
STAMD	0111 1101	RAM[HL]←Acc, LR-1	1	1	-	Z	C
STAMI	0111 1111	RAM[HL] ←Acc, LR+	1	1	-	Z	C'
STD #k,y	0100 1000 kkkk yyyy	RAM[y]←k	2	2	-	-	1
STDMI #k	1010 kkkk	RAM[HL]←k, LR+	1	1	-	Z	C'
THA	0111 0110	Acc←HR	1	1	-	Z	1
TLA	0111 0100	Acc←LR	1	1	-	Z	1

1.7.2 Rotate

Mnemonic	Object Code (Binary)	Operation Description	Byte	Cycle	Flag		
					CF	ZF	SF
RLCA	0101 0000	$\overline{[\leftarrow CF\leftarrow Acc\leftarrow]}$	1	1	C	Z	C'
RRCA	0101 0001	$\overline{[\rightarrow CF\rightarrow Acc\rightarrow]}$	1	1	C	Z	C'

1.7.3 Arithmetic Operation

Mnemonic	Object Code (Binary)	Operation Description	Byte	Cycle	Flag		
					CF	ZF	SF
ADCAM	0111 0000	Acc←Acc + RAM[HL] + CF	1	1	C	Z	C'
ADD #k,y	0100 1001 kkkk yyyy	RAM[y]←RAM[y] + k	2	2	-	Z	C'
ADDA #k	0110 1110 0101 kkkk	Acc←Acc + k	2	2	-	Z	C'

ADDAM	0111 0001	Acc←Acc + RAM[HL]	1	1	-	Z	C'
ADDH #k	0110 1110 1001 kkkk	HR←HR + k	2	2	-	Z	C'
ADDL #k	0110 1110 0001 kkkk	LR←LR + k	2	2	-	Z	C'
ADDM #k	0110 1110 1101 kkkk	RAM[HL]←RAM[HL] + k	2	2	-	Z	C'
DECA	0101 1100	Acc←Acc-1	1	1	-	Z	C
DECL	0111 1100	LR←LR-1	1	1	-	Z	C
DECM	0101 1101	RAM[HL]←RAM[HL] -1	1	1	-	Z	C
INCA	0101 1110	Acc←Acc +1	1	1	-	Z	C'
INCL	0111 1110	LR←LR +1	1	1	-	Z	C'
INCM	0101 1111	RAM[HL]←RAM[HL]+1	1	1	-	Z	C'
SUBA #k	0110 1110 0111 kkkk	Acc←k-Acc	2	2	-	Z	C
SBCAM	0111 0010	Acc←RAM[HLl - Acc - CF'	1	1	C	Z	C
SUBM #k	0110 1110 1111 kkkk	RAM[HL]←k - RAM[HL]	2	2	-	Z	C

1.7.4 Logical Operation

Mnemonic	Object Code (Binary)	Operation Description	Byte	Cycle	Flag		
					CF	ZF	SF
ANDA #k	0110 1110 0110 kkkk	Acc←Acc&k	2	2	-	Z	Z'
ANDAM	0111 1011	Acc←Acc & RAM[HL]	1	1	-	Z	Z'
ANDM #k	0110 1110 1110 kkkk	RAM[HL]←RAM[HL]&k	2	2	-	Z	Z'
ORA #k	0110 1110 0100 kkkk	Acc←Acc | k	2	2	-	Z	Z'
ORAM	0111 1000	Acc←Acc | RAM[HL]	1	1	-	Z	Z'
ORM #k	0110 1110 1100 kkkk	RAM[HL]←RAM[HL] | k	2	2	-	Z	Z'
XORAM	0111 1001	Acc←Acc^RAM[HL]	1	1	-	Z	Z'

1.7.5 Exchange

Mnemonic	Object Code (Binary)	Operation Description	Byte	Cycle	Flag		
					CF	ZF	SF
EXA x	0110 1000 xxxx xxxx	Acc↔RAM[x]	2	2	-	Z	1
EXAH	0110 0110	Acc↔HR	1	1	-	Z	1
EXAL	0110 0100	Acc↔LR	1	1	-	Z	1
EXAM	0101 1000	Acc↔RAM[HL]	1	1	-	Z	1
EXHL x	0100 1100 xxxx xxx0	LR↔RAM[x],HR↔RAM[x+1]	2	3	-	-	1

1.7.6 Branch

Mnemonic	Object Code (Binary)	Operation Description	Byte	Cycle	Flag		
					CF	ZF	SF
BR	000a aaaa aaaa aaaa	If SF=1 than PC← a Else null	2	2	-	-	1
JHL	0100 0110	PC←RAM[HL+3],RAM[HL+2],RAM[HL+1],RAM[HL]	1	2	-	-	-

1.7.7 Compare

Mnemonic	Object Code (Binary)	Operation Description	Byte	Cycle	Flag		
					CF	ZF	SF
CMP #k,y	0100 1011 kkkk yyyy	k-RAM[y]	2	2	C	Z	Z'
CMPA x	0110 1011 xxxx xxxx	RAM[x]-Acc	2	2	C	Z	Z'
CMPAM	0111 0011	RAM[HL] - Acc	1	1	C	Z	Z'
CMPH #k	0110 1110 1011 kkkk	k - HR	2	2	-	Z	C
CMPIA #k	1011 kkkk	k - Acc	1	1	C	Z	Z'
CMPL #k	0110 1110 0011 kkkk	k-LR	2	2	-	Z	C

1.7.8 Bit Manipulation

Mnemonic	Object Code (Binary)	Operation Description	Byte	Cycle	Flag		
					CF	ZF	SF
CLM b	1111 00bb	$RAM[HL]_b \leftarrow 0$	1	1	-	-	1
CLP p,b	0110 1101 11bb pppp	$PORT[p]_b \leftarrow 0$	2	2	-	-	1
CLPL	0110 0000	$PORT[LR_{3-2}+4]_{LR1-0} \leftarrow 0$	1	2	-	-	1
CLR y,b	0110 1100 11bb yyyy	$RAM[y]_b \leftarrow 0$	2	2	-	-	1
SEM b	1111 01bb	$RAM[HL]_b \leftarrow 1$	1	1	-	-	1
SEP p,b	0110 1101 01bb pppp	$PORT[p]_b \leftarrow 1$	2	2	-	-	1
SEPL	0110 0010	$PORT[LR_{3-2}+4]_{LR1-0} \leftarrow 1$	1	2	-	-	1
SET y,b	0110 1100 01bb yyyy	$RAM[y]_b \leftarrow 1$	2	2	-	-	1
TF y,b	0110 1100 00bb yyyy	$SF \leftarrow RAM[y]_b'$	2	2	-	-	*
TFA b	1111 10bb	$SF \leftarrow Acc_b'$	1	1	-	-	*
TFM b	1111 11bb	$SF \leftarrow RAM[HL]_b'$	1	1	-	-	*
TFP p,b	0110 1101 00bb pppp	$SF \leftarrow PORT[p]_b'$	2	2	-	-	*
TFPL	0110 0001	$SF \leftarrow PORT[LR_{3-2}+4]_{LR1-0}'$	1	2	-	-	*
TT y,b	0110 1100 10bb yyyy	$SF \leftarrow RAM[y]_b$	2	2	-	-	*
TTP p,b	0110 1101 10bb pppp	$SF \leftarrow PORT[p]_b$	2	2	-	-	*

1.7.9 Subroutine

Mnemonic	Object Code (Binary)	Operation Description	Byte	Cycle	Flag		
					CF	ZF	SF
LCALL a	001a aaaa aaaa aaaa	SP←SP -1, PC←a STACK[SP]←PC,	2	2	-	-	-
SCALL a	1110 nnnn	SP←SP - 1, PC←a, a = 8n + 6 (n =1~15),a=86h(n=0) STACK[SP]←PC,	1	2	-	-	-
RET	0100 1111	PC←STACK[SP], SP=SP + 1	1	2	-	-	-

1.7.10 Input/Output

Mnemonic	Object Code (Binary)	Operation Description	Byte	Cycle	Flag		
					CF	ZF	SF
INA p	0110 1111 0100 pppp	Acc←PORT[p]	2	2	-	Z	Z'
INM p	0110 1111 1100 pppp	RAM[HL]←PORT[p]	2	2	-	-	Z'
OUT #k,p	0100 1010 kkkk pppp	PORT[p]←k	2	2	-	-	1
OUTA p	0110 1111 000p pppp	PORT[p]←Acc	2	2	-	-	1
OUTM p	0110 1111 100p pppp	PORT[p]←RAM[HL]	2	2	-	-	1

1.7.11 Flag Manipulation

Mnemonic	Object Code (Binary)	Operation Description	Byte	Cycle	Flag		
					CF	ZF	SF
TFCFC	0101 0011	SF←CF', CF←0	1	1	0	-	*
TTCFS	0101 0010	SF←CF, CF←1	1	1	1	-	*
TZS	0101 1011	SF←ZF	1	1	-	-	*

1.7.12 Interrupt Control

Mnemonic	Object Code (Binary)	Operation Description	Byte	Cycle	Flag		
					CF	ZF	SF
CIL r	0110 0011 00rr rrrr	IL←IL & r	2	2	-	-	1
DICIL r	0100 0100 00rr rrrr	EIF←0, IL←IL&r	2	2	-	-	1
EICIL r	0100 0101 00rr rrrr	EIF←1, IL←IL&r	2	2	-	-	1
EXAE	0111 0101	MASK↔Acc	1	1	-	-	1
RTI	0100 1101	FLAG.PC← STACK[SP] SP←SP+1, EIF←1	1	2	*	*	*

1.7.13 CPU Control

Mnemonic	Object Code (Binary)	Operation Description	Byte	Cycle	Flag		
					CF	ZF	SF
NOP	0101 0110	no operation	1	1	-	-	-
HALT	0100 0111	CPU halt, wait interrupt release	1	1	-	-	-

1.7.14 Timer/Counter, Data Pointer, and Stack Pointer Control

Mnemonic	Object Code (Binary)	Operation Description	Byte	Cycle	Flag		
					CF	ZF	SF
LDADPL	0110 1010 1111 1100	Acc←$[DP]_L$	2	2	-	Z	1
LDADPM	0110 1010 1111 1101	Acc←$[DP]_M$	2	2	-	Z	1
LDADPH	0110 1010 1111 1110	Acc←$[DP]_H$	2	2	-	Z	1
LDASP	0110 1010 1111 1111	Acc←SP	2	2	-	Z	1
LDATAL	0110 1010 1111 0100	Acc←$[TA]_L$	2	2	-	Z	1
LDATAM	0110 1010 1111 0101	Acc←$[TA]_M$	2	2	-	Z	1
LDATAH	0110 1010 1111 0110	Acc←$[TA]_H$	2	2	-	Z	1
LDATBL	0110 1010 1111 1000	Acc←$[TB]_L$	2	2	-	Z	1
LDATBM	0110 1010 1111 1001	Acc←$[TB]_M$	2	2	-	Z	1
LDATBH	0110 1010 1111 1010	Acc←$[TB]_H$	2	2	-	Z	1
STADPL	0110 1001 1111 1100	$[DP]_L$←Acc	2	2	-	-	1
STADPM	0110 1001 1111 1101	$[DP]_M$←Acc	2	2	-	-	1
STADPH	0110 1001 1111 1110	$[DP]_H$←Acc	2	2	-	-	1
STASP	0110 1001 1111 1111	SP←Acc	2	2	-	-	1
STATAL	0110 1001 1111 0100	$[TA]_L$←Acc	2	2	-	-	1
STATAM	0110 1001 1111 0101	$[TA]_M$←Acc	2	2	-	-	1
STATAH	0110 1001 1111 0110	$[TA]_H$←Acc	2	2	-	-	1
STATBL	0110 1001 1111 1000	$[TB]_L$←Acc	2	2	-	-	1
STATBM	0110 1001 1111 1001	$[TB]_M$←Acc	2	2	-	-	1
STATBH	0110 1001 1111 1010	$[TB]_H$←Acc	2	2	-	-	1

1.8 Notation Conventions

The following are the notation conventions used in the above tables and throughout this manual unless otherwise specified.

Symbol	Operation	Symbol	Operation
HR	H register	LR	L register
PC	Program counter	DP	Data pointer
SP	Stack pointer	STACK[SP]	Stack specified by SP
Acc	Accumulator	FLAG	All flags
CF	Carry flag	ZF	Zero flag
SF	Status flag	LR_{3-2}	Bit 3 to 2 of LR
EI	Enable interrupt register	IL	Interrupt latch
MASK	Interrupt mask	PORT[p]	Port (address : p)
TA	Timer/counter A	TB	Timer/counter B
RAM[HL]	Data memory (address : HL)	RAM[x]	Data memory (address : x)
$ROM[DP]_L$	Low 4-bit of program memory	$ROM[DP]_H$	High 4-bit of program memory
$[DP]_L$	Low 4-bit of data pointer register	$[DP]_M$	Middle 4-bit of data pointer register
$[DP]_H$	High 4-bit of data pointer register	$[TA]_L([TB]_L)$	Low 4-bit of CounterA/TimerB register
$[TA]_M([TB]_M)$	Middle 4-bit of CounterA/TimerB register	$[TA]_H([TB]_H)$	High 4-bit of CounterA/TimerB register
←	Transfer	↔	Exchange
+	Addition	-	Subtraction
&	Logic AND	|	Logic OR
^	Logic XOR	'	Inverse operation
.	Concatenation	#k	4-bit immediate data
x	8-bit RAM address	y	4-bit zero-page address
p	4-bit or 5-bit port address	b	Bit address
r	6-bit interrupt latch	PC_{12-6}	Bit 12 to 6 of program counter
LR_{1-0}	Contents of bit assigned by bit 1 to 0 of LR	a_{5-0}	Bit 5 to 0 of destination address for branch instruction

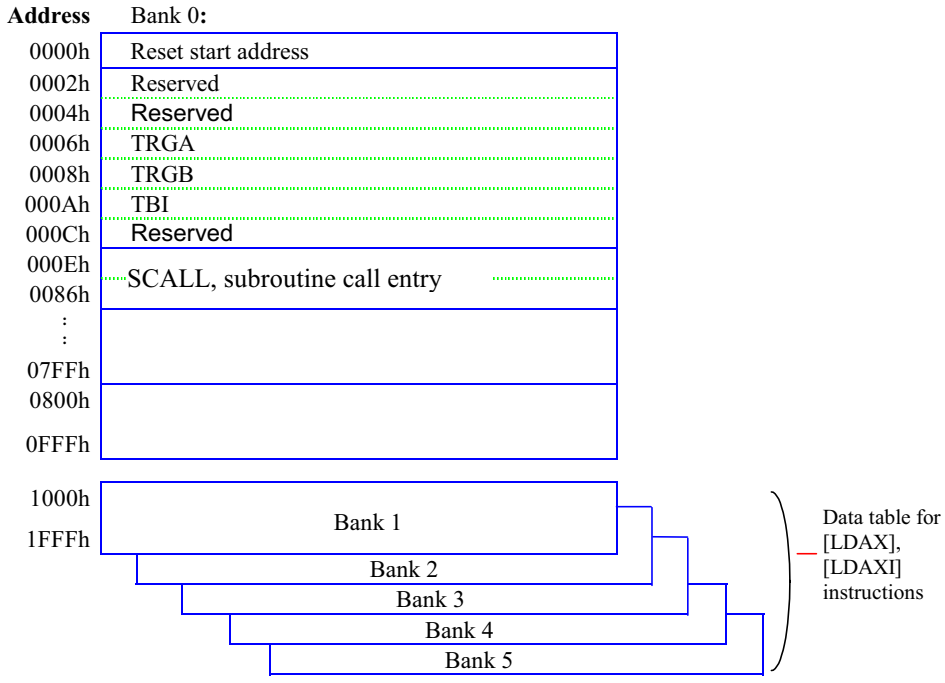
Chapter 2

Programming Model

2.1 Program ROM (24K x 8bits) for EM73PE00

24K x 8 bits program ROM contains user's program and some fixed data. The basic structure of the program may be categorized into 4 partitions.

1. Address 0000h: Reset start address.
2. Address 0002h ~ 000Ch: 3 kinds of interrupt service routine entry addresses.
3. Address 000Eh-0086h: SCALL subroutine entry address, only available at 000Eh, 0016h, 001Eh, 0026h, 002Eh, 0036h, 003Eh, 0046h, 004Eh, 0056h, 005Eh, 0066h, 006Eh, 0076h, 007Eh. 0086h.
4. Address 0000h ~ 1FFFh: Except for the regions used for the above functions, other region can be used as user's program region.



2.1.1 User's Program and Fixed Data

User's program and fixed data are stored in the program ROM. User's program is executed using the PC value to fetch an instruction code.

User's Program

The 24Kx8 bits program ROM can be divided into 4 banks at 4Kx8 bits per bank.

The program ROM bank is selected by P3 (2..0). The program counter is a 13-bit binary counter. The PC and P3 are initialized to "0" during reset.
When P3(2..0) = 000B, program ROM Bank0 and Bank1, are selected.
When P3(2..0) = 001B, Bank0 and Bank2 are selected.
When P3(2..0) = 010B, Bank0 and Bank3 are selected.

Address	P3 = x000B	P3 = x001B	P3 = x010B	P3 = x011B	P3 = x100B
0000h : : 0FFFh	Bank0				
1000h : : 1FFFh	Bank1	Bank2	Bank3	Bank4	Bank5

Example:

```
            BANK   0
START:      :
            :
            :
            LDIA   #00H          ; set program ROM to bank1
            OUTA   P3
            BR     XA1
            :
XA :        :
            :
            LDIA   #01H          ; set program ROM to bank2
            OUTA   P3
            BR     XB1
            :
XB :        :
            :
            LDIA   #02H          ; set program ROM to bank3
            OUTA   P3
            BR     XC1
            :
XC :        :
            :
```

```
            BR    XD
XD :        :
            :
            :
; - - - - - - - - - - - - - - - - - - - - - - - - - - - -
            BANK  1
XA1 :       :
            :
            BR    XA
            :
XA2 :       :
            BR    XA2
; - - - - - - - - - - - - - - - - - - - - - - - - - - - -
            BANK  2
XB1 :       :
            :
            BR    XB
            :
XB2 :       :
            BR    XB2
            :
; - - - - - - - - - - - - - - - - - - - - - - - - - - - -
            BANK  3
XC1 :       :
            :
            BR    XC
            :
      XC2 :       :
                  B     XC2
```

Fixed Data

Fixed Data is read out by table-look-up instruction. The instruction requires the Data Pointer (DP) to indicate the ROM address in obtaining the ROM code data (except Bank0)**:**

LDAX **Acc← ROM[DP]$_L$**

LDAXI **Acc← ROM[DP]$_H$,DP+1**

Where**: DP** is a 12-bit data register that stores the program ROM address as pointer for the ROM code data. User must to initially load ROM address into DP register with instructions "LDADPL", "LDADPM", and "LDADPH" and assign the bank # by out (bank #-1) into P3.

To obtain the lower nibble of ROM code data, use instruction "LDAX", and use "LDAXI." to obtain the higher nibble. After "LDAXI" instruction. The DP value will be increased one automatically

Example:

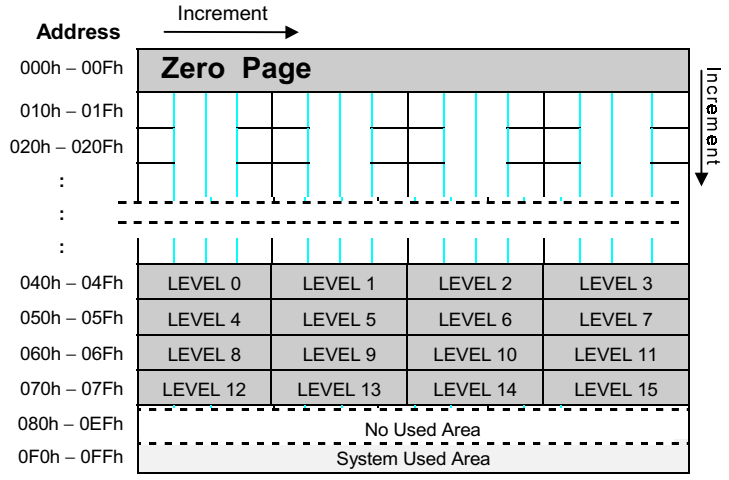
Read out the ROM code of address 777h in Bank 1 and 111h in Bank 2 by table-look-up instruction.

```
:
OUT #00h,P3; set bank 1
```

```
LDIA #07h;
STADPL      ; DP2-0← 07h
STADPM      ; DP5-3← 07h
STADPH      ; DP8-6← 07h, Load DP=777h
:
LDL #00h;
LDH #03h;
LDAX        ; ACC← 6h
STAMI       ; RAM[30]← 6h
LDAXI       ; ACC← 5h, increase DP=778h
STAMI       ; RAM[31]← 5h
;
OUT #01h,P3; set bank 1
LDIA #01h;
STADPL      ; DP2-0← 01h
STADPM      ; DP5-3← 01h
STADPH      ; DP8-6← 01h, Load DP=111h
:
LDL #02h;
LDH #03h;
LDAX        ; ACC← Ch
STAMI       ; RAM[32]← Ch
LDAXI       ; ACC← Ah, increase DP=112h
STAMI       ; RAM[33]←Ah
;
BANK 1
ORG 0777h
DATA 56h;
:
BANK 2
ORG 111h
DATA ACh;
```

2.2 Data RAM (128-nibble)

A total of 128-nibble data RAM is available from address 000 to 7Fh. Data RAM includes the Zero Page region, Stacks and Data areas.



2.2.1 Zero Page

From 000h to 00Fh is the zero-page location. It is used as the zero-page address mode pointer for instructions "STD #k,y; ADD #k,y; CLR y,b; and CMP y,b".

Example:

Write immediate data "07h" to address "03h" of RAM, and clear bit 2 from RAM.

STD #07h, 03h ; RAM[03]← 07h

CLR 0Eh,2 ; RAM[0Eh]2← 0

2.2.2 Stack

There are 16 (maximum) stack levels that user can use as subroutines (including interrupt and CALL). User can assign any level as the starting stack by providing the level number to stack pointer (SP).

When an instruction (CALL or subroutine) is invoked before the subroutine is entered, the previous PC address is saved into the stack until its return from those subroutines. The PC value is restored by the data saved in stack.

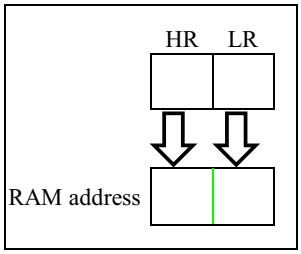
2.2.3 Data Area

Except for the area used for user application, the whole RAM can be used as data area for storing and loading general data.

2.2.4 Address Mode

The Data Memory also consisted of three Address Modes, namely-

1) Indirect Address Mode

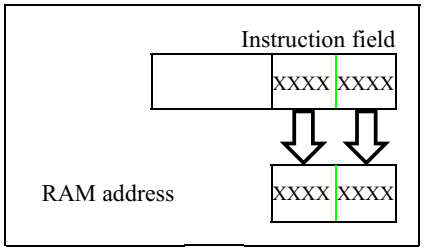


The address is specified by the HL registers.

Example:

```
LDAM        ; Acc← RAM[HL]
STAM        ; RAM[HL]← Acc
```

2) Direct Address Mode

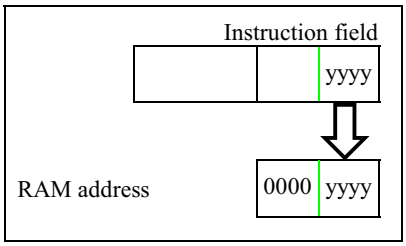


The address is specified by 8 bits of the second byte in the instruction field.

Example:

```
LDA  43h    ; Acc← RAM[43h]
STA  23h    ; RAM[23h] ← Acc
```

3) Zero-Page Address Mode



The zero-page is in address 000h~00Fh. The address are the lower nibble of the second byte in the instruction field.

Example:

```
STD   #k, y   ; RAM[y]← #k
```

2.3 Program Counter (24K ROM) for EM73PE00

The Program Counter (PC) consisted of 13-bit counter. It indicates the next address to be executed by program ROM instruction.

For an 8K-byte size ROM, PC indicates address 0000h - 1FFFh as BRANCH and CALL instructions. PC is changed when indicating the desired instruction.

2.3.1 Branch Instruction

BR a

Object code: 000a aaaa aaaa aaaa

Condition: SF=1; PC← $PC_{13.a}$ (branch condition satisfied)

PC	a	a	a	a	a	a	a	a	a	a	a	a	a

SF=0; PC← PC +2(branch condition not satisfied)

PC	Original PC value+2

2.3.2 JUMP instruction

JHL

Object code: 0100 0110

Condition: PC← RAM[HL+3].RAM[HL+2].RAM[HL+1].RAM[HL]; HL%4=0

PC	a	a	a	a	a	a	a	a	a	a	a	a	a

2.3.3 Subroutine Instruction

SCALL a

Object Code: 1110 nnnn

Condition: PC← a ; a=8n+6 , n=1..Fh ;a=86h, n=0

PC	0	0	0	0	0	a	a	A	a	a	1	1	0

LCALL a

Object code: 001a aaaa aaaa aaaa

Condition: PC← a

PC	a	A	a	a	a	a	a	a	a	a	a	a	a

RET

Object code: 0100 1111

Condition: PC← STACK[SP]; SP + 1

PC	The return address stored in stack

2.3.4 Interrupt Acceptance Operation

When an interrupt is received, the original PC is entered into stack and interrupt vector is loaded into PC. The interrupt vectors are as follows:

TRGA (Counter overflow interrupt)

PC	0	0	0	0	0	0	0	0	0	0	1	1	0

TRGB (Timer overflow interrupt)

PC	0	0	0	0	0	0	0	0	0	0	1	0	0	0

TBI (Time base interrupt)

PC	0	0	0	0	0	0	0	0	0	0	1	0	1	0

2.3.5 Interrupt Return Operation

Object code: 0100 1101

Condition: FLAG. PC← STACK[SP]; EI← 1; SP + 1

PC	The return address stored in stack

2.3.6 Reset Operation

PC	0	0	0	0	0	0	0	0	0	0	0	0	0	0

2.3.7 Other Operations

For 1-byte instruction execution: PC + 1

For 2-byte instruction execution: PC + 2

For 3-byte instruction execution: PC + 3

Chapter 3

Signal and Function Descriptions

3.1 Accumulator (ACC)

Accumulator (ACC) is a 4-bit data register for temporary data. ACC plays the role of holding the source data and result for arithmetic, logic, and comparison operations.

3.2 Flags

There are three types of flag:

- Carry Flag (CF)
- Zero Flag (ZF)
- Status Flag (SF)

These three 1-bit flags are influenced by the arithmetic, logic, and comparison operation.

All flags are loaded into stack when an interrupt subroutine is served, and the flags are restored after RTI instruction is executed.

3.2.1 Carry Flag (CF)

The Carry Flag is affected by the following operations:

a) **Addition:** CF as a carry out indicator. Under Addition operation, when a carry-out occurs, CF is "1". Likewise, if the operation has no carry-out, CF is "0".

b) **Subtraction:** CF as a borrow-in indicator. Under Subtraction operation, when a borrow-in occurs, the CF is "0". Likewise, if there is no borrow-in, CF will be "1".

c) **Comparison:** CF is as a borrow-in indicator for Comparison operation as in the Subtraction operation.

d) **Rotation:** CF shifts into the empty bit of accumulator for rotation and holds the shift out data after rotation.

e) **CF Test Instruction:**
Under TFCFC instruction, the CF' content is sent into SF, then clears itself as "0". For TTCFS instruction, the content of CF is sent into SF then sets itself as "1".

3.2.2 Zero Flag (ZF)

ZF is affected by the result of ALU. If the ALU operation generate a "0" result, the ZF is "1". Otherwise, the ZF is "0".

3.2.3 Status Flag (SF)

The SF is affected by instruction operation and system status.

a) SF is initiated to "1" for reset condition.

b) Branch instruction is decided by SF. When SF=1, branch condition is satisfied. Likewise, when SF=0, branch condition is dissatisfied.

3.2.4 Examples

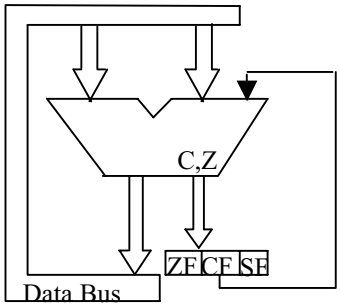
Check the following arithmetic operation for CF, ZF, and SF

Arithmetic Operation	CF	ZF	SF
LDIA #00h	-	1	1
LDIA #03h	-	0	1
ADDA #05h	-	0	1
ADDA #0Dh	-	0	0
ADDA #0Eh	-	0	0

3.3 ALU

The Arithmetic and logic operations of 4-bit data are performed in ALU unit.

3.3.1 ALU Structure



ALU supports user arithmetic operation functions, including Addition, Subtraction, and Rotation.The C, Z affect the ZF, CF, and SF results.

3.3.2 ALU Supported Functions

1) Addition:

ALU supports Addition function with instructions ADDAM, ADCAM, ADDM #k, and ADD #k,y. The Addition operation affects signal C and signal Z.

Under Addition operation, if the result is "0", Z will be "1". Otherwise, Z will be "0". When Addition operation has a carry-out, C is "1". Otherwise, C will be "0".

Example:

Operation	C	Z
3+4=7	0	0
7+F=6	1	0
0+0=0	0	1
8+8=0	1	1

2) Subtraction:

ALU supports Subtraction function with instructions SUBM #k, SUBA #k, SBCAM, and DECM. The Subtraction operation affects C and Z.

Under Subtraction operation, if the result is negative, C is equal to "0" or borrow out. If the result is positive, C will be "1".

Likewise, if the result of subtraction operation is "0", the Z is "1". Otherwise, Z will be "1".

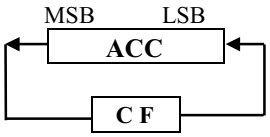
Example:

Operation	C	Z
8-4=4	1	0
7-F= -8(1000)	0	0
9-9=0	1	1

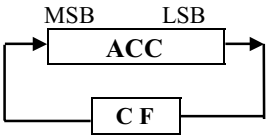
3) Rotation:

Two types of Rotation operations are available. One is Rotation Left, the other is Rotation Right.

RLCA instruction rotates ACC value counter-clockwise, shift the CF value into the LSB bit of ACC, and then hold the shift out data in CF.



RRCA instruction rotates ACC value clockwise, shift the CF value into the MSB bit of ACC, and then hold the shift out data in CF.



Example: Rotate ACC clockwise (right) and shift "1" into the MSB bit of ACC.

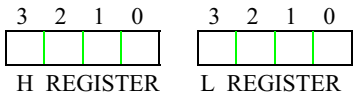
TTCFS**;** CF← 1

RRCA**;** rotate ACC right and shift CF=1 into MSB.

3.4 HL Register

HL Register are two 4-bit registers. They are used as a pair of pointer for RAM memory address. They are also used as 2 independent temporary 4-bit data registers. For certain instructions, L Register can be a pointer to indicate the pin number (Port 4).

The following illustrates the HL Register structure.



3.4.1 HL Register Function

1) Temporary Register Function

HL Register is used as a temporary register for instructions LDL #k, LDH #k, LDHL, THA, THL, INCL, DECL, EXAL, EXAH.

LDHL legal address should be even number.

Example: Load immediate data "5h" into L register, "Dh" into H register.

LDL #05h;

LDH #0Dh;

2) Memory Pointer Function

HL Register is used as a pointer of RAM memory address for instructions LDAM, STAM, STAMI.

Example: Store immediate data #Ah into RAM of address 35h.

```
LDL     #5h;
LDH     #3h;
STDMI   #0Ah    ; RAM[35]← Ah
```

3) I/O Port Pointer Function

HL Register is used as a pointer to indicate the I/O port bit for instructions SELP, CLPL, TFPL (when LR = 0, it indicates P4.0).

Example: Set bit 0 of Port 4 to "1"

```
LDL     #00h;
SEPL            ; P4.0← 1
```

4)direct Jump function

HL Register is used as a pointer to indicate the destination address for instructions JHL. HL should be limited on xxxxxx00.

Example: Jump to address 1234h

```
LDL     #07H;
LDH     #01H;
LDIA    #01H;
STAMD           ;[17] ← 01H
LDIA    #02H;
STAMD           ;[16] ← 02H
LDIA    #03H;
STAMD           ;[15] ← 03H
LDIA    #04H;
STAM            ;[14] ← 04H
JHL             ; PC ←1234H
```

3.5 Stack Pointer (SP)

Stack Pointer is a 4-bit register that stores the present stack level number. The SP value must be set first before using stack. CPU will not initiate the SP value after a reset condition. When a new subroutine is received, the SP value is automatically decreased by one. Likewise, if a subroutine is sent, the SP value is increased by one.

The data transfer between ACC and SP is done with instructions "LDASP" and "STASP".

Example:

```
LDIA    #0FH    ;
STASP           ; SP ←0FH
LCALL   NEXT1   ; SP[0EH] ←PC ,PC ←NEXT1
```

```
:
NEXT1:
:
RET                         ; PC ← SP[0EH]
```

3.6 Data Pointer (DP)

Data Pointer is a 12-bit register that stores the ROM address indicating the ROM code data specified by user (refer to Chapter 2, *Program ROM)*.

3.7 Reset Function

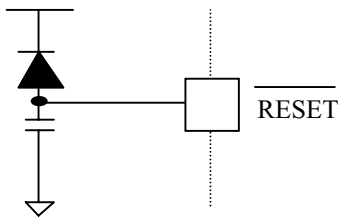
When $\overline{\text{RESET}}$ pin is held in low level for at least 3 instruction cycles with the CPU working normally, the CPU will begin to initialize the whole internal states. When $\overline{\text{RESET}}$ pin changes to high level, the CPU will begin to work in normal condition.

The following table shows the CPU internal state during reset condition:

Hardware Condition Under RESET (f1) State	Initial value
Program counter	0000h
P3	00h
Status flag	01h
Interrupt enable flip-flop (EI)	00h
MASK0 ,1, 2	00h
Interrupt latch (IL)	00h
RMO	0h
P7, P1.1	input
P5,P6	0Fh
XIN	Start oscillation

The $\overline{\text{RESET}}$ pin is a hysteresis input pin and it has a pull-up resistor.

The simplest RESET circuit is to connect $\overline{\text{RESET}}$ pin with a capacitor to V_{SS} and a diode to V_{DD}.

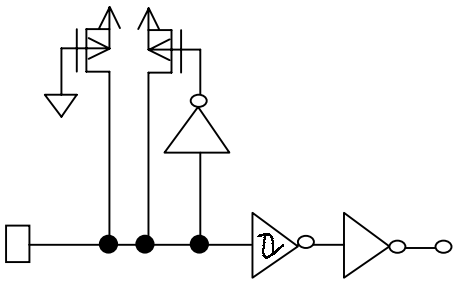


Chapter
4

Pin Type Circuity Diagrams

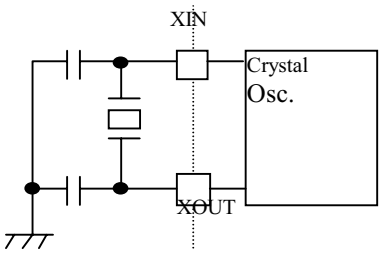
4.1 Pin Type Circuitry Diagrams

4.1.1 Reset Pin Type

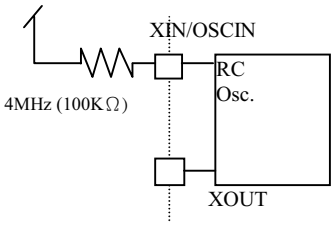


Type Reset-A

4.1.2 Oscillation Pin Types

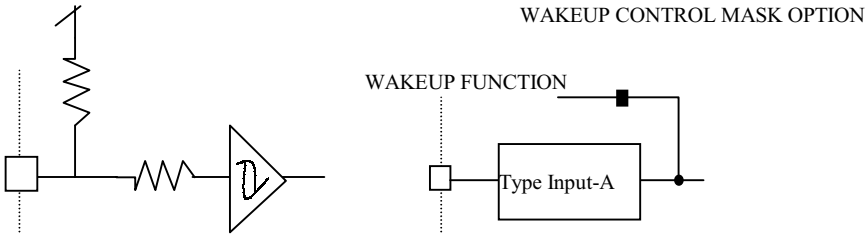


Type Osc-A



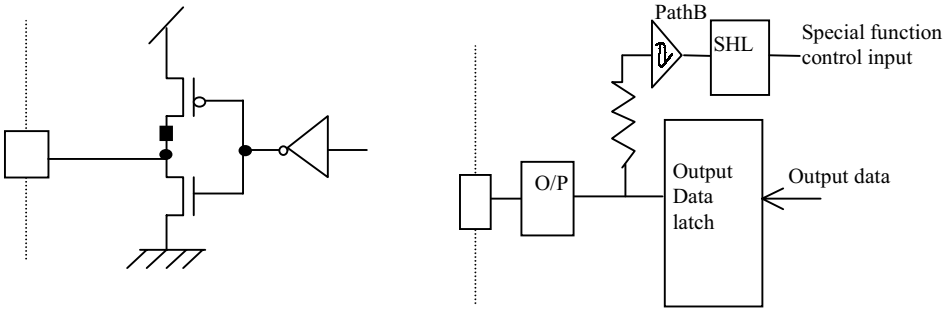
Type Osc-H

4.1.3 Input Pin Types

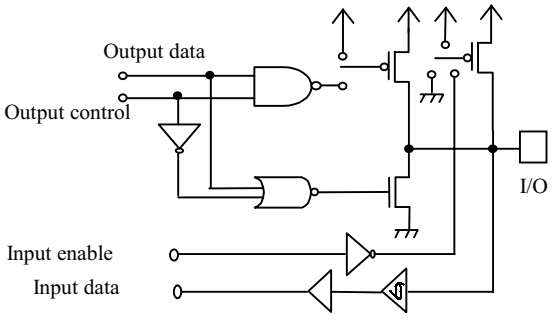


Type Input-A Type Input-B

4.1.4 I/O Pin Types



Type O/P Type I/O-RMO



Type O/P

4.1.5 Bi-direction I/O Program Controlling

P10 is the command port for controlling P7.

Port 10

3	2	1	0
P7C3	P7C2	P7C1	P7C0

Initial state: 0000

Input /Output mode selection	
P7Cn	**Function Description**
0	P7.n as input port.
1	P7.n as output port.

n:3~0

P11.1 is the command port for controlling P1.1.

Port 11

3	2	1	0
TIMF	CNTF	P1C	-

Initial state: 0000

Input /Output mode selection	
P1C	**Function Description**
0	P1.1 as input port.
1	P1.1 as output port.

```
Example: P7 output by program control.
         LDIA #1110B      ;P7C=1110
         OUTA P10         ;P7(3..1) define output port
         :
         :
         LDIA #0111B;
         OUTA P7          ;P7=0110B
         :
         :
```

EM73PE00 I/O Port Descriptions

Port	Input Function		Output Function		Remarks
0	E	Input port , wake-up function			
1		P1.1 input		P1.0 :RMO, P1.1 output	
2		--			
3		--	I	P3(1..0) : ROM bank selection	
4	E	Input port, wake-up function		--	
5		--	E	Output port	
6		--	E	Output port	
7	E	Input port	E	Output port	
8		--		--	
9		--		--	
10	I	P7C	I	P7C	P7 output by program control
11	I	TIMF+CNTF+P1C	I	TIMFC+CNTFC+P1C	Ref. Tmr/Cnt
12		--		--	
13		--		--	
14		--		--	
15		--		--	
16			I	Sleep/Off mode control register	
17					
18				--	
19				--	
20				--	
21			I	Watch Dog Timer control register	
22				--	
23					
24					
25			I	Timebase control register	
26					
27					
28			I	Counter control register	
29			I	Timer control register	
30					
31			--		

Chapter
5

Clock and Timing

5.1 Overview

This chapter summarizes the clock and timing for EM73PE00 CPU.

5.2 Clock and Timing Generator

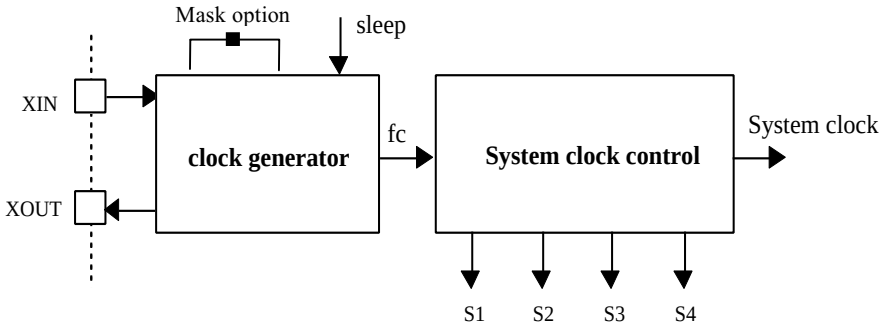
The clock generator is supported by a single clock system. The clock sources from crystal. The working frequency range is 200KHz to 4MHz depending on the actual working voltage.

5.2.1 Clock and Timing Generator Structure

The clock generator unit generates a basic system clock "fc".

When CPU is in off mode, the clock generator will remain disabled until the off condition ends.

The system clock control unit generates 4 basic clock phase (S1, S2, S3, S4) and a system clock signal.



5.2.2 Clock and Timing Generator Function

The “fc” frequency is equal to the oscillation frequency of pins XIN and XOUT with crystal (resonator).

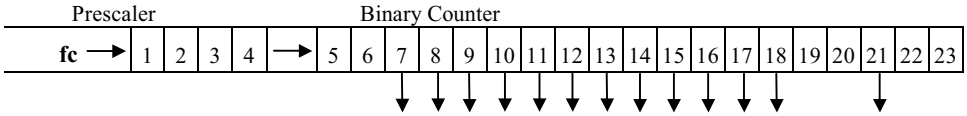
When CPU is under off mode, the XOUT pin is in "high" state.

The instruction cycle equals to 4 basic clock “fc”.

1 instruction cycle = 4 / fc

5.3 Internal Time Base

As illustrated below, a time base has 23 stages.



Time base provides basic frequency for the following functions:

- TBI (time base interrupt)
- Timer/counter, internal clock source
- Warm-up time for off-mode disable

5.3.1 Time Base Interrupt (TBI)

The time base is used to generate a single fixed frequency interrupt. Eight types of frequencies can be selected with the "P25" setting:

Single clock mode

P25

3	2	1	0

(initial value 0000)

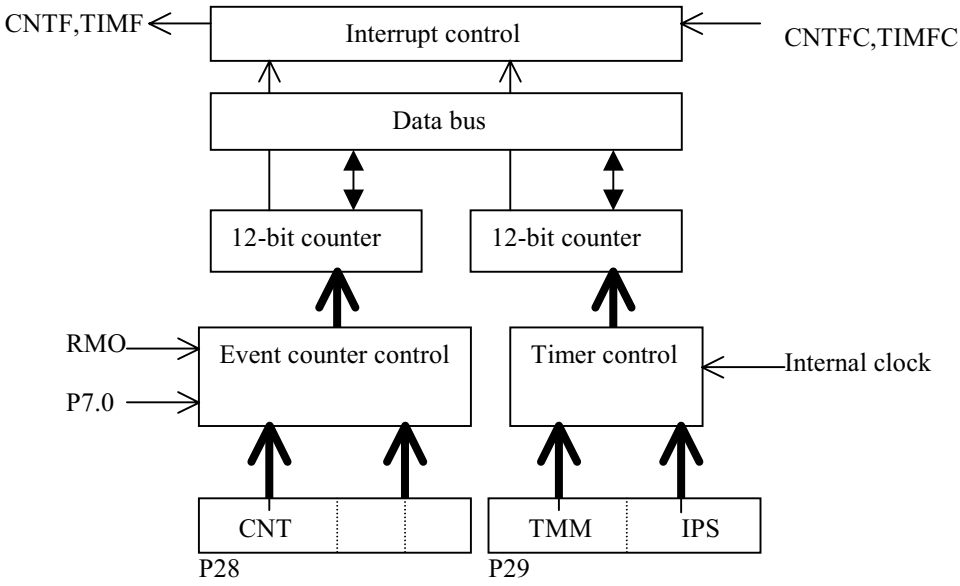
0 0 x x: Interrupt disable
0 1 0 0: Interrupt frequency XIN / 2^{10} Hz
0 1 0 1: Interrupt frequency XIN / 2^{11} Hz
0 1 1 0: Interrupt frequency XIN / 2^{12} Hz
0 1 1 1: Interrupt frequency XIN / 2^{13} Hz
1 1 0 0: Interrupt frequency XIN / 2^{9} Hz
1 1 0 1: Interrupt frequency XIN / 2^{8} Hz
1 1 1 0: Interrupt frequency XIN / 2^{15} Hz
1 1 1 1: Interrupt frequency XIN / 2^{17} Hz
1 0 x x: Reserved

5.4 Timer/Counter

With Counter, the counter data are saved in the timer registers TAH, TAM, and TAL. User can set initial counter value. Such value is loaded or stored by executing instructions "LDATAH(M,L), STATAH(M,L)."

With Timer, the timer data are saved in the TBH, TBM, and TBL. User can set initial counter value. Such value is loaded or stored by executing instructions "LDATBH (M,L), STATBH (M,L)".

The timer/counter basic structure (see figure below) consisted of two identical modules. The timer or counter module can be set as initial timer or counter values which are sent to the register. P28 and P29 are the command ports for Counter and Timer. User can choose different operating modes and internal clock rates to set these two ports. When timer/counter overflows, it will generate a TRGB/TRGA interrupt request to interrupt control unit.



5.4.1 Counter Function

RMO and P7.0are the external inputs for Counter

Port 28

3	2	1	0
CNT		*	*

Initial state: 0000

COUNTER MODE SELECTION	
CNT	**Function Description**
0 0	Stop
1 0	RMO input counter
1 1	P7.0 input counter

Under counter , the counter increased by one at any falling edge of RMO and P7.0.

When counter counts overflow, it will provide an interrupt request TRGA to interrupt control.

P11	3	2	1	0
Read	TIMF	CNTF	P1C	-
Write	TIMFC	CNTFC		

CNTF: When MASK2=0, if Counter overflow then read P11.2 whose value (CNTF) is 1.

CNTFC: When write 1 to P11.2 ,then Flag is reset (CNTFC =0).

Example: Enable Counter with P28

```
LDIA #1000B;
OUTA P28;          Enable Counter with counter mode
EICIL 110111b;     interrupt latch ←0, enable EI
```

5.4.2 Timer Function

P29 is the command port for timer

Port 29

3	2	1	0
TMM	*	IPS	

Initial state: 0000

Counter mode selection	
TMM	**Function Description**
0	Stop
1	Timer mode

Internal pulse rate selection	
IPS	**Function Description**
0 0	XIN/2^8Hz
0 1	XIN/2^{10}Hz
1 0	XIN/2^{12}Hz
1 1	XIN/2^{14}Hz

Under Timer , Timer increased by one at any falling edge of internal pulse.user can choose above 4 types of internal pulse rate by setting IPS to timer command.

When timer counts overflow, it will provide an interrupt request TRGB to interrupt control.

P11	3	2	1	0
Read	TIMF	CNTF	P1C	-
Write	TIMFC	CNTFC		

TIMF: When MASK1=0, if Timer overflow then read P11.3 whose value (TIMF) is 1.

TIMFC: When write 1 to P11.3 ,then Flag is reset (TIMFC =0).

Example: To generate TRGB interrupt request after 60ms with system clock XIN=4MHz

```
LDIA #0010B;
EXAE;              Enable mask 1
```

```
EICIL 111011B;  interrupt latch <-0,enable EI
;
LDIA #06h;
STATBL;
LDIA #01h;
STATBM;
LDIA #0Fh;
STATBH;
LDIA #1001B;
OUTA P29;          enable Timer with internal pulse rate: XIN/2^10 Hz
```

NOTE

The preset value of Timer/Counter Register is calculated as follows:

Internal pulse rate: XIN/2^10 ; XIN = 4MHz

The time of Timer/Counter count one = 2^10 /XIN = 1024/4000=0.256ms

The number of internal pulse to get timer overflow = 60 ms/ 0.256ms = 234.375 = 0EAh

The preset value of Timer/Counter Register = 1000h - 0EAh = F16h

Chapter
6

Interrupt and Power Saving Functions

6.1 Overview

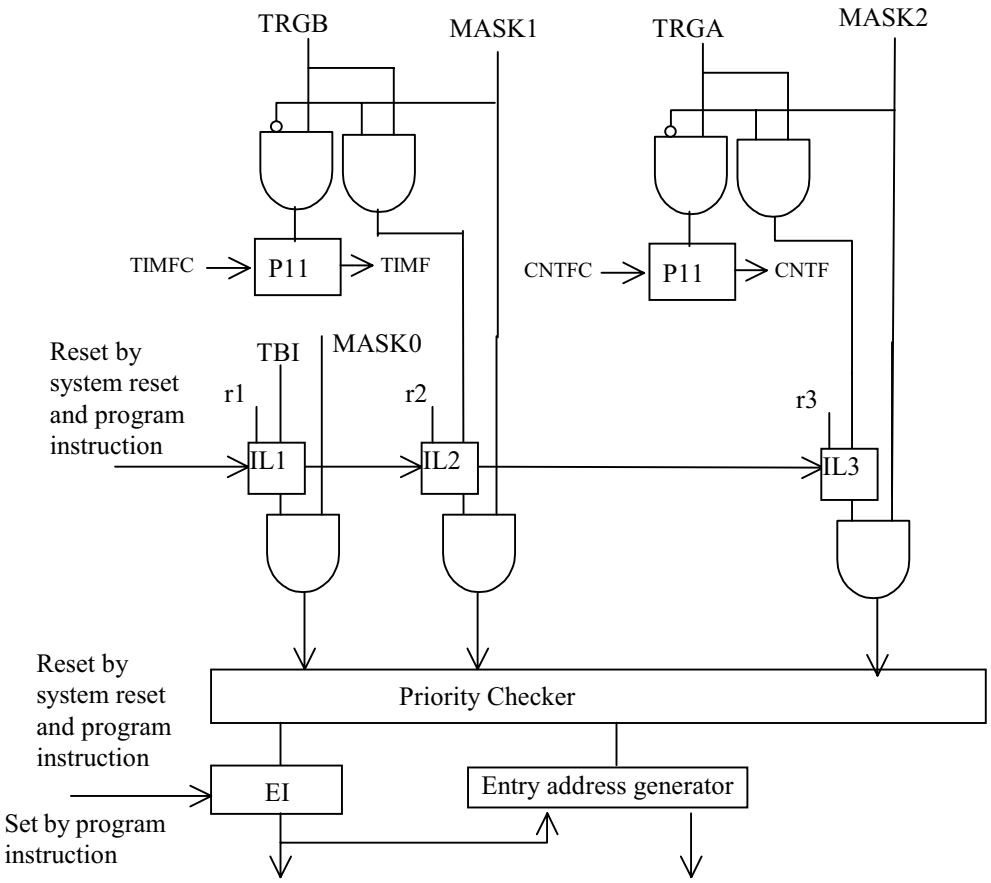
This chapter summarizes the interrupt and power saving functions of EM73PE00 CPU.

6.2 Interrupt Function

They are 3 internal interrupt sources. Multiple interrupts are admitted according to their priority .

Type	Interrupt Source	Priority	Interrupt Latch	Interrupt Enable Condition	Program ROM Entry Address
Internal	Counter overflow interrupt (TRGA)	1	IL3	EI=1, MASK2=1	006h
Internal	Timer overflow interrupt (TRGB)	2	IL2	EI=1, MASK1=1	008h
Internal	Time base interrupt(TBI)	3	IL1	EI=1, MASK0=1	00Ah

6.2.1 Interrupt Structure



Interrupt Controller

IL1-IL3	Interrupt latch. Holds all interrupt requests from all interrupt sources. IL can not be set by the program, but can be reset by program or system reset. So IL can only decide which interrupt source can be accepted.
MASK0 - MASK2	MASK register may promote or inhibit all interrupt sources.
EI	Enable interrupt Flip-Flop may promote or inhibit all interrupt sources. When interrupt occurs, EI is auto cleared to "0". After RTI instruction is executed, EI is auto set to "1" again .
Priority Checker	Check interrupt priority when multiple interrupts occur.

6.2.2 Interrupt Operation

Follow the steps below for interrupt operation:

1. Locate PC and all flags to stack.
2. Set interrupt entry address to PC.
3. Set SF= 1.
4. Clear EI to inhibit other interrupts happened.
5. Clear the IL with which interrupt source has already been accepted.
6. Execute interrupt subroutine from the interrupt entry address.
7. CPU accept RTI, restore PC and flags from stack . Set EI to accept other interrupt requests.

Example: Enable interrupt “TRGA".

```
LDIA #1100B;
EXAE; set mask register "1100B"
EICIL 111111B  ; enable interrupt F.F.
```

6.3 Power Saving (OFF / Sleep) Function

During Off and Sleep Modes, CPU keeps the systems internal status under low power consumption condition.

Under Off Mode, the system clock changes into snooze state. The system needs a warm up period to stabilize the system clock back to running condition after wake-up.

Under Sleep Mode, the system clock does not turn into snooze state. It needs no warm-up period.

6.3.1 Off / Sleep Control

The Off and Sleep Modes are controlled by Port 16, and released by P0(0..3) or P4(0...3), dependent on Mask Option.

P16 3 2 1 0

WM	SE	SWWT

Initial value: 0000

WM	Set wake-up release mode
0	Wake-up in edge release mode
1	Wake-up in level release mode

SE	Enable off/sleep
0	Reserved
1	Enable off/sleep mode

SWWT	Set wake-up warm-up time	OSC	Note
00	2^{16}/XIN	Stop	Off mode 0
01	2^{12}/XIN	Stop	Off mode 1
10	2^{14}/XIN	Stop	Off mode 2
11	Sleep mode	No Stop	No Need warm-up time

6.3.2 Off and Sleep Conditions

The following conditions exist when system is under Off/Sleep Mode:

- Osc stop (off only) and CPU internal status held
- Internal time base clear to "0"
- CPU internal memory, flags, register, and I/O held original states
- Program Counter freezes at the executed address until CPU released.

Example: Entry the Off/Sleep mode

```
LDIA #0101B       ;
OUTA P16          ;entry Off mode1,wakeup in edge.
```

6.3.3 Release Condition

The following conditions exists when system is wakes-up from Off /Sleep Mode:

- Osc starts to oscillate (off only)
- Warm-up period in process (off only)
- According PC to execute the following program

6.3.4 Edge Off/Sleep Mode:

Releases the Off/Sleep condition by the falling edge of any of the following pins**:**

- P0(0..3)
- P4(0..3)

When P0/P4 wakeup in edge/level release mode, must attend to the other P0/P4 whether enable wakeup release mode, because the wakeup result decide by wakeup function gating. (Ref 4.1.3)

NOTE

There are 8 independent mask options for wake-up function in EM73PE00. So, the wake-up function of P0(0..3) or P4(0...3) is enabled or disabled independently.

HALT condition: Program Counter execute HALT instruction ,OSC not stop and CPU internal status held. CPU internal memory, flags, register and I/O held original state. When Interrupt happen the HALT condition released.

Chapter 7

Miscellaneous Function

7.1 Overview

This chapter describes the remote control signal output, low voltage detector and watch dog timer features of EM73PE00.

7.2 Remote control signal output port

This RMO is a dedicate terminal for carrier wave output of remote controller. When a remote control data is written into the remote control output data register, P1.0, the data is output from RMO terminal as it is.

Port 1

3	2	1	0
*	*	*	RMO

Initial state: ***0

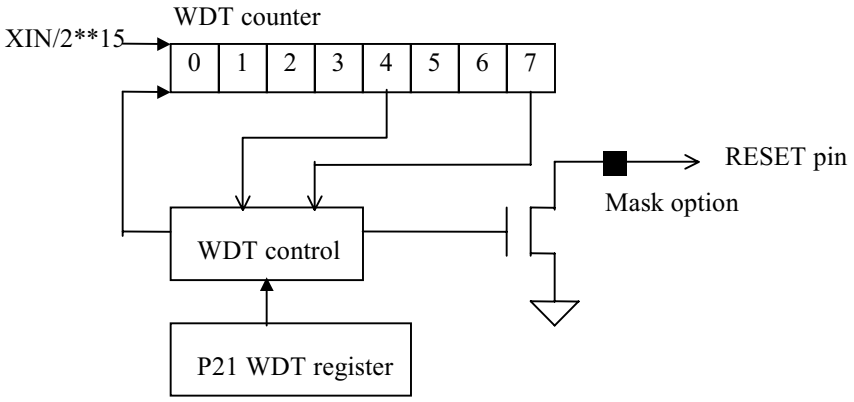
REG	Value	Function Description
RMO	0	Output low
	1	Output high

7.3 Low Voltage Detector

If power supply voltage is lower than normally operating voltage, automatically initializes internal system. This function can be used by masking option.

7.4 Watch Dog Timer

The watch dog timer is an 8-bit timer that protects from runaway programs. The block diagram of the watch dog timer are shown below. The watch dog timer is enabled or disabled by mask option. If watch dog timer is enabled, each interval time whice choosing by P21, the system will be reset.



P21 is the control port of watch dog timer, and the WDT time up signal will reset the system.

Port 21

3	2	1	0
CWC	*	WDT	

Initial state: 0*00

REG	Value	Function Description
CWC	0	Clear WDT
	1	Nothing
WDT	00	2^{19}/XIN
	01	2^{22}/XIN
	10	2^{15}/XIN
	11	2^{12}/XIN

Chapter 8

Electrical Properties

8.1 Overview

This chapter provides information on electrical, thermal and timing properties of the EM73PE00.

8.2 Absolute Maximum Ratings

Items	Sym.	Ratings	Conditions
Supply Voltage	V_{DD}	-0.3V to 6V	T_{OPR} = 25°C
Input Voltage	V_{IN}	V_{ss} -0.3V to V_{DD}+0.3V	T_{OPR} = 25°C
Output Voltage	V_O	V_{ss} to V_{DD}	T_{OPR} = 25°C
Power Dissipation	P_D	300mW	T_{OPR} = 25°C
Operating Temperature	T_{OPR}	-20°C to 70°C	
Storage Temperature	T_{STG}	-40°C to 125°C	

8.3 Recommended Operating Conditions

Items	Sym.	Ratings	Condition
Supply Voltage	V_{DD}	2.5V to 5.5V	EM73PE00/01/02AM
Supply Voltage	V_{DD}	3.2V to 5.5V	EM73PE00/01/02GM
Input Voltage	V_{IH}	0.80xV_{DD} to V_{DD}	
	V_{IL}	0V to 0.20xV_{DD}	
Operating Frequency		200KHz to 4MHz	

8.4 DC Electrical Characteristics

V_{DD}=3V, V_{SS}=0V, T_{OPR}=--20°C to 70°C

Parameters	Sym.	Min	Typ.	Max.	Unit	Conditions
Operation current consumpion	I_{VDD}	-	0.4	1	mA	no load, Fc=4MHz
Standby current consumption	I_{DDs}	-	0.4	1	uA	no load, at OFF mode, No use low voltage detector
Standby current consumption	I_{DDs}	-	4	6	uA	no load, at OFF mode, Use low voltage detector
Input Current	I_{IH}	3	10	30	uA	TEST
High level input current 1	I_{IH1}	-	-	1	uA	P0, P4, P7, V_{IH}= V_{DD}
High level input current 2	I_{IH2}	-8	-5	-2	uA	$\overline{\text{RESET}}$, Vin=Vdd-0.3V
Low level input current 1	I_{IL1}	-120	-50	-20	uA	P0, P4, P7 with pull-up resistor
Low level input current 2	I_{IL2}	-10	-4	-1.5	uA	$\overline{\text{RESET}}$, Vin=Vss
High level output current 1	I_{OH1}	-20	-10	-5	mA	RMO, Vout = 2.1V
High level output current 2	I_{OH2}	-7	-3	-1.5	mA	P1.1,P7, Vout = 2.1V
High level output current 3	I_{OH3}	-900	-450	-200	uA	P5, P6, Vout = 2.6V
Low level output current 1	I_{OL1}	2	5	15	mA	RMO, Vout = 0.9V
Low level output current 2	I_{OL2}	4	8	16	mA	P1.1,P7, Vout = 0.9V
Low level output current 3	I_{OL3}	1.0	1.5	3	mA	P5, P6, Vout = 0.4V
Hysteresis width	V_{HYS-}	-	1	-	V	
Low battery detector	Vlb	-	2.3	-	V	25°C

V_{DD}=4.5V, V_{SS}=0V, T_{OPR}=--20°C to 70°C

Parameters	Sym.	Min	Typ.	Max.	Unit	Conditions
Operation current consumpion	I_{VDD}	-	1	2	mA	No load, Fc=4MHz
Standby current consumption	I_{DDs}	-	0.8	1.5	uA	No load, at OFF mode, No use low voltage detector
Standby current consumption	I_{DDs}	-	4	6	uA	No load, at OFF mode, Use low voltage detector
Input Current	I_{IH}	20	30	50	uA	TEST
High level input current 1	I_{IH1}	-	-	1	uA	P0, P4, P7, V_{IH}= V_{DD}
High level input current 2	I_{IH2}	-11	-8	-5	uA	$\overline{\text{RESET}}$, Vin=Vdd-0.3V
Low level input current 1	I_{IL1}	-200	-150	-100	uA	P0, P4, P7 with pull-up resistor
Low level input current 2	I_{IL2}	-20	-15	-10	uA	$\overline{\text{RESET}}$, Vin=Vss
High level output current 1	I_{OH1}	-30	-20	-10	mA	RMO, Vout = 3.2V
High level output current 2	I_{OH2}	-10	-7	-3	mA	P1.1,P7, Vout = 3.2V
High level output current 3	I_{OH3}	-1.2	-0.9	-0.6	mA	P5, P6, Vout = 3.9V
Low level output current 1	I_{OL1}	5	15	25	mA	RMO, Vout = 0.9V
Low level output current 2	I_{OL2}	8	12	20	mA	P1.1,P7, Vout = 0.9V
Low level output current 3	I_{OL3}	1.0	1.5	3	mA	P5, P6, Vout = 0.4V
Hysteresis width	V_{HYS-}	-	1	-	V	
Low battery detector	Vlb	-	2.3	-	V	25°C