

GENERAL DESCRIPTION

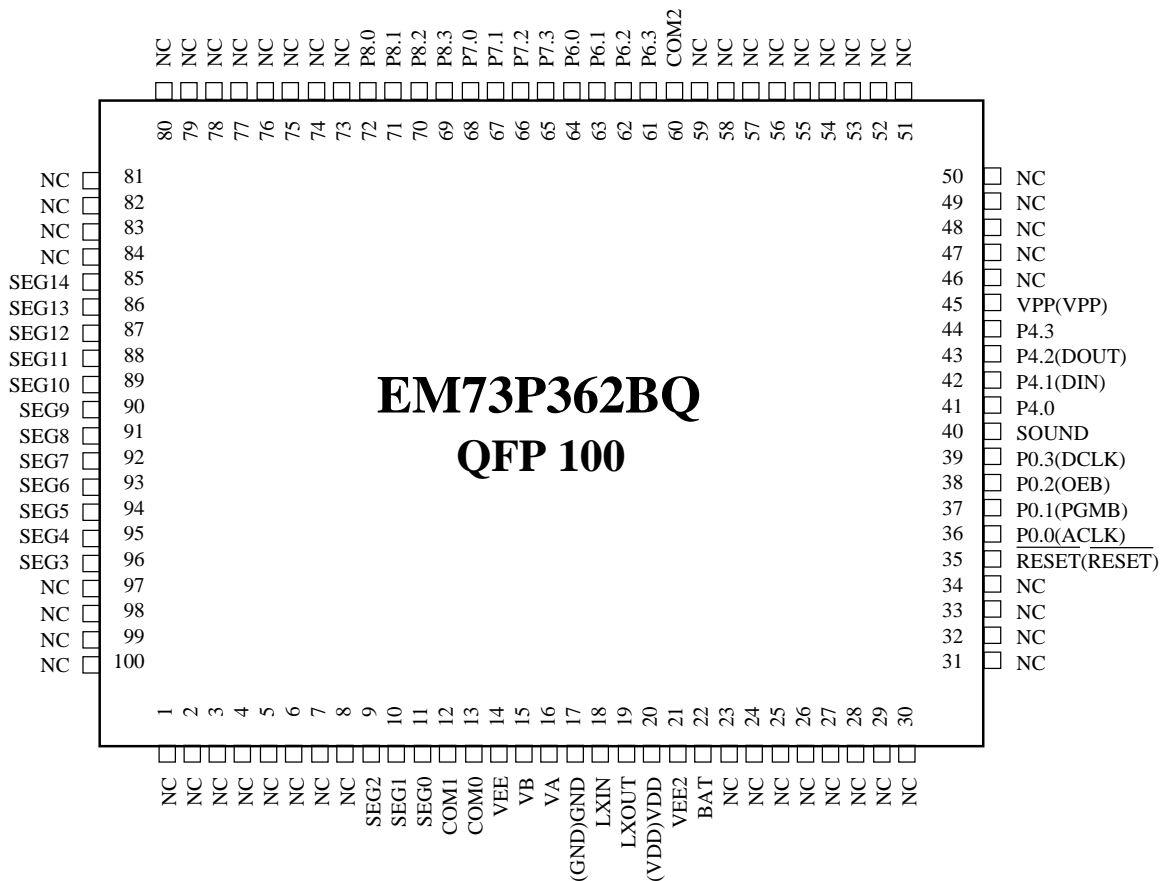
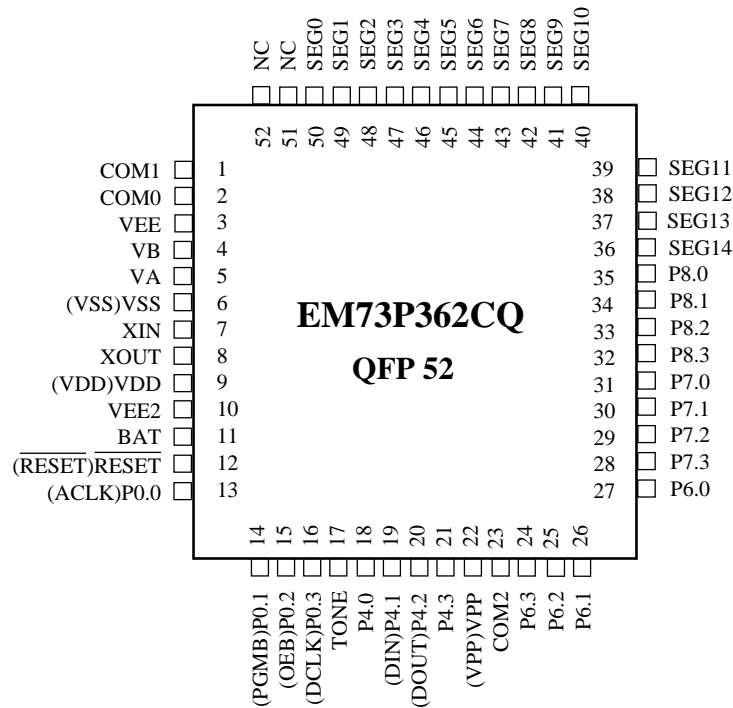
EM73P362 is an advanced single chip CMOS 4-bit one-time-programmable (OTP) micro-controller. It contains 3K-byte ROM, 52-nibble RAM, 4-bit ALU, 13-level subroutine nesting, 22-stage time base, one 12-bit timer for the kernel function and one high speed counter. EM73P362 also contains 5 interrupt sources, 1 input port, 4 bidirection ports, built-in watch-dog-timer and LCD driver (27x3 to 15x3).

Except low-power consumption and high speed, EM73P362 has the STOP mode and IDLE mode operation for power saving function.

FEATURES

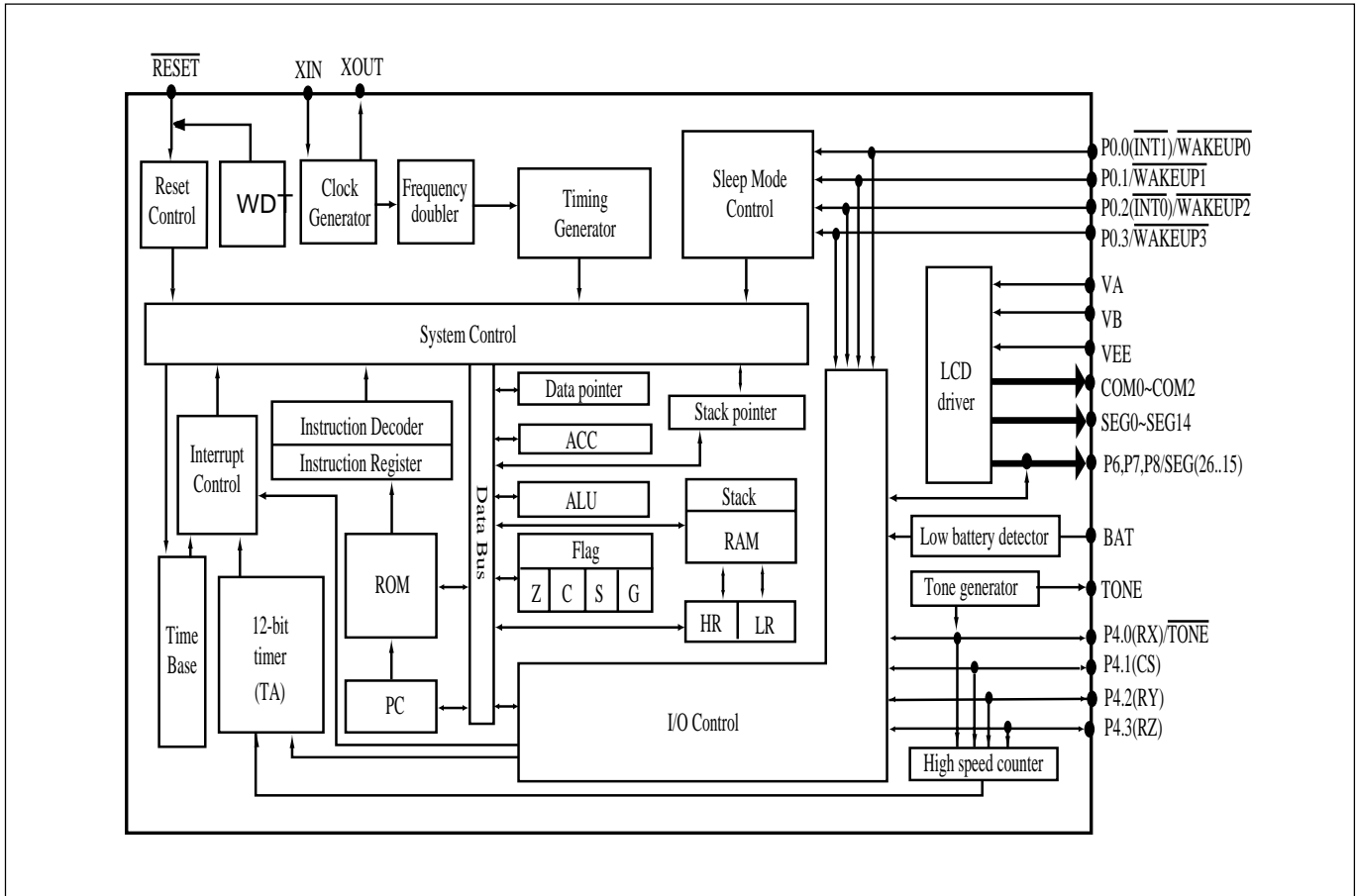
- Operation voltage : 1.3V to 1.8V. (clock frequency : 32K Hz)
- Clock source : Single clock system for crystal, connect a external resistor or external clock source, available by mask option.
- Instruction set : 109 powerful instructions.
- Instruction cycle time : Up to 122 μ s for 32 K Hz.
- ROM capacity : 3072 x 8 bits.
- RAM capacity : 52 x 4 bits.
- Input port : 1 port (4-bit).
- Bidirection port : 4 ports (P4, P6, P7, P8) are available by mask option. P4 is a high current port. (P4.0 and TONE available by mask option. P4.1~P4.3 are shared with the input/output of RFO.) P6, P7 and P8 are shared with SEG15-SEG26.
- 12-bit timer : One 12-bit timer is programmable for timer.
- High speed counter : The high speed counter includes one 8-bit high speed counter, one 12-bit general counter and a resistor frequency oscillator. It has resistor to frequency oscillation mode, melody mode and auto load timer mode.
- Built-in time base counter: 22 stages.
- Subroutine nesting : Up to 13 levels.
- Interrupt : External interrupt 2 input interrupt sources.
Internal interrupt 2 timer overflow interrupts,
1 time base interrupt.
- LCD driver : 27x3 to 15x3 dots available by mask option. 1/3, 1/2 and static three kinds of duty (1/2 bias) selectable. The programming method of LCD driver is RAM mapping.
- Built-in watch-dog-timer is available by mask option.
- Built-in low battery detector.
- Power saving function : STOP mode and IDLE mode.
- Package type : Chip form 50 pins.
QFP 52 pins (CQ).
QFP 100 pins (BQ).

PIN ASSIGNMENT



Remark : In () pin used for OTP programming.

FUNCTION BLOCK DIAGRAM



PIN DESCRIPTIONS

| Pin name | Function | Pin type |
|---|---|---------------|
| VDD | Power supply (+), Power supply (+) for programming OTP | |
| VSS | Power supply (-), Power supply (-) for programming OTP | |
| RESET | System reset input signal, low active mask option : none pull-up | RESET_A |
| XIN | Crystal / external resistor or external clock source connecting pin | OSC_A / OSC_F |
| XOUT | Crystal / external resistor connecting pin | OSC_A / OSC_F |
| P0.0(INT1)/WAKEUP0, P0.2(INT0)/WAKEUP2 | 2-bit input pins with external interrupt sources input and STOP/IDLE releasing function mask option : wake-up enable, pull-up wake-up enable, none wake-up disable, pull-up wake-up disable, pull-down wake-up disable, none In programming OTP mode: P0.0/ACLK : address counter clock for programming OTP P0.2/OEB : data output enable for programming OTP | INPUT_J |

PIN DESCRIPTIONS

| Pin name | Function | PIN type |
|--|---|----------|
| P0(1,3)/ $\overline{\text{WAKEUP1,3}}$ | 2-bit input pins with STOP / IDLE releasing function mask option : wakeup enable, pull-up wakeup enable, none wakeup disable, pull-up wakeup disable, pull-down wakeup disable, none In programming OTP mode : P0.1/PGMB : program data to OTP cells for programming OTP P0.3/DCLK : data in/out clock signal for programming OTP | INPUT_H |
| P4.0(RX)/ $\overline{\text{TONE}}$ | 1-bit bidirection I/O pin or inverse sound effect output or RF oscillation mask option : $\overline{\text{TONE}}$ enable, push-pull, high current PMOS $\overline{\text{TONE}}$ disable, open-drain(apply to RF oscillation) $\overline{\text{TONE}}$ disable, push-pull, high current PMOS $\overline{\text{TONE}}$ disable, push-pull, low current PMOS | I/O_O |
| P4.1(CS) | 1-bit bidirection I/O pin or RF oscillation bias pin mask option : open-drain(apply to RF oscillation) push-pull, high current PMOS push-pull, low current PMOS In programming OTP mode : P4.1/DIN : data input for programming OTP | I/O_X |
| P4.2(RY) | 1-bit bidirection I/O pins or RF oscillation input pins mask option : open-drain(apply to RF oscillation) push-pull, high current PMOS push-pull, low current PMOS In programming OTP mode : P4.2/DOUT : data output for programming OTP | I/O_Y |
| P4.3(RZ) | 1-bit bidirection I/O pins or RF oscillation input pins mask option : open-drain(apply to RF oscillation) push-pull, high current PMOS push-pull, low current PMOS | I/O_Y |
| P6(0..3)/SEG(23..26), P7(0..3)/SEG(19..22), P8(0..3)/SEG(15..18) | 12-bit bidirection I/O pins are shared with LCD segment pin mask option : segment enable, open-drain segment disable, open-drain segment disable, push-pull, high current PMOS segment disable, push-pull, low current PMOS | I/O_O |
| BAT | Connect the capacitor for built-in low battery detector | |
| TONE | Built-in tone generator output | |
| VA, VB, VEE | Connect the capacitors for LCD bias voltage | |
| VEE2 | Used for LCD bias voltage Connect to VDD | |
| COM0 ~ COM2 | LCD common output pins | |
| SEG0 ~ SEG14 | LCD segment output pins | |
| VPP | In normal mode : No connection (Floating) In programming OTP mode : VPP : high voltage (12V) power source for programming OTP | |

FUNCTION DESCRIPTIONS

PROGRAM ROM (3K X 8 bits)

3 K x 8 bits program ROM contains user's program and some fixed data.

The basic structure of program ROM can be divided into 4 parts.

1. Address 000h: Reset start address.
2. Address 002h - 00Ch : 5 kinds of interrupt service routine entry addresses.
3. Address 00Eh-086h : SCALL subroutine entry address, only available at 00Eh,016h,01Eh,026h, 02Eh, 036h, 03Eh, 046h, 04Eh, 056h, 05Eh, 066h, 06Eh, 076h, 07Eh, 086h.
4. Address 000h - 7FFh : LCALL subroutine entry address.
5. Address 000h - BFFh : Except used as above function, the other region can be used as user's program region.

| address | 3072 x 8 bits |
|---------|---|
| 000h | Reset start address |
| 002h | INT0; External interrupt service routine entry address |
| 004h | |
| 006h | TRGA; Timer/counter A interrupt service routine entry address |
| 008h | TRGB; Timer/counter B interrupt service routine entry address |
| 00Ah | TBI; Time base interrupt service routine entry address |
| 00Ch | INT1; External interrupt service routine entry address |
| 00Eh | - - - SCALL, subroutine call entry address |
| 086h | |
| ⋮ | |
| BFFh | |

User's program and fixed data are stored in the program ROM. User's program is according the PC value to send next executed instruction code. Fixed data can be read out by table-look-up instruction. Table-look-up instruction is depended on the Data Pointer (DP) to indicate to ROM address, then to get the ROM code data.

LDAX **Acc** ← **ROM[DP]_L**
LDAXI **Acc** ← **ROM[DP]_H, DP+1**

DP is a 12-bit data register which can store the program ROM address to be the pointer for the ROM code data. First, user load ROM address into DP by instruction "STADPL, STADPM, STADPH", then user can get the lower nibble of ROM code data by instruction "LDAX" and higher nibble by instruction "LDAXI".

PROGRAM EXAMPLE: Read out the ROM code of address 777h by table-look-up instruction.

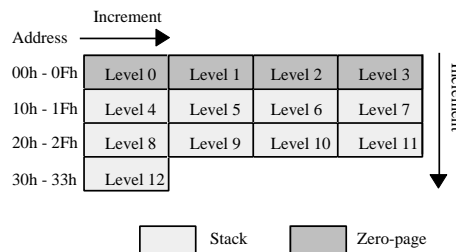
```

LDIA #07h;
STADPL   ; [DP]L ← 07h
STADPM   ; [DP]M ← 07h
STADPH   ; [DP]H ← 07h, Load DP=777h
:
LDL #00h;
LDH #03h;
LDAX     ; ACC ← 6h
STAMI    ; RAM[30] ← 6h
LDAXI    ; ACC ← 5h
STAM     ; RAM[31] ← 5h
;
ORG 777h
DATA 56h;
:

```

DATA RAM (52-nibble)

There is total 52 - nibble data RAM from address 00 to 33h
Data RAM includes 3 parts: zero page region, stacks and data area.



ZERO- PAGE:

From 00h to 0Fh is the location of zero-page. It is used as the pointer in zero-page addressing mode for the instruction of "STD #k,y; ADD #k,y; CLR y,b; CMP k,y".

PROGRAM EXAMPLE: To write immediate data "07h" to address "03h" of RAM and to clear bit 2 of RAM.

```

STD #07h, 03h ; RAM[03] ← 07h
CLR 0Eh,2 ; RAM[0Eh]2 ← 0

```

STACK:

There are 13-level (maximum) stack for user using for subroutine (including interrupt and CALL). User can assign any level be the starting stack by giving the level number to stack pointer (SP).

When user using any instruction of CALL or subroutine, before entry the subroutine, the previous PC address will be saved into stack until return from those subroutines, the PC value will be restored by the data saved in stack.

DATA AREA:

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

ADDRESSING MODE
(1) Indirect addressing mode:

Indirect addressing mode indicates the RAM address by specified HL register.

For example: LDAM ; Acc ← RAM[HL]
 STAM ; RAM[HL] ← Acc

(2) Direct addressing mode:

Direct addressing mode indicates the RAM address by immediate data.

For example: LDA x ; Acc ← RAM[x]
 STA x ; RAM[x] ← Acc

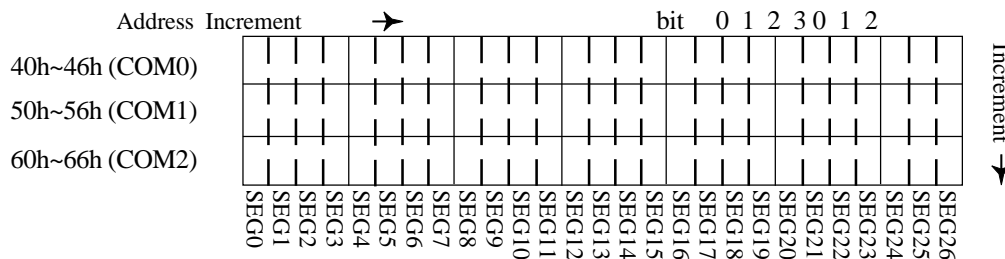
(3) Zero-page addressing mode

For zero-page region, user can using direct addressing to write or do any arithmetic, comparison or bit manipulated operation directly.

For example: STD #k,y ; RAM[y] ← #k
 ADD #k,y ; RAM[y] ← RAM[y] + #k

LCD DISPLAY RAM

RAM address from 40h ~ 46h, 50h ~ 56h, 60h ~ 66h are LCD display RAM, the RAM data of this region can't be operated by instruction "LDHL xx" and "EXHL".


PROGRAM COUNTER (3K ROM)

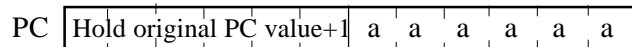
Program counter (PC) is composed by a 12-bit counter, which indicates the next executed address for the instruction of program ROM.

For a 3K - byte size ROM, PC can indicate address form 000h - BFFh, for BRANCH and CALL instructions, PC is changed by instruction indicating.

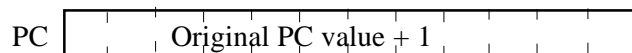
(1) Branch instruction:
SBR a

Object code: 00aa aaaa

Condition: SF=1; PC ← PC_{11-6,a} (branch condition satisfied)

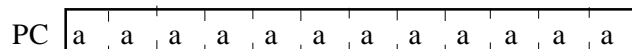


SF=0; PC ← PC + 1 (branch condition not satisfied)


LBR a

Object code: 1100 aaaa aaaa aaaa

Condition: SF=1; PC ← a (branch condition satisfied)



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

PC

| |
|-----------------------|
| Original PC value + 2 |
|-----------------------|

(2) Subroutine instruction:

SCALL a

Object code: 1110 nnnn

Condition : PC ← a ; a=8n+6 ; n=1..15 ; a=86h, n=0

PC

| | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | a | a | a | a | a | a | a | a | a |
|---|---|---|---|---|---|---|---|---|---|---|---|---|

LCALL a

Object code: 0100 0aaa aaaa aaaa

Condition: PC ← a

PC

| | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 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 |
|------------------------------------|

RTI

Object code: 0100 1101

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

PC

| |
|------------------------------------|
| The return address stored in stack |
|------------------------------------|

(3) Interrupt acceptance operation:

When an interrupt is accepted, the original PC is pushed into stack and interrupt vector will be loaded into PC. The interrupt vectors are as following:

INT0 (External interrupt from P0.2)

PC

| | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|

TRGA (Timer A overflow interrupt)

PC

| | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|

TRGB (Time B overflow interrupt)

PC

| | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 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 | 1 | 0 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|

INTI (External interrupt from P0.0)

PC

| | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|

(4) Reset operation:

PC

| | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|

(5) Other operations:

For 1-byte instruction execution: PC + 1

For 2-byte instruction execution: PC + 2

ACCUMULATOR

Accumulator is a 4-bit data register for temporary data. For the arithmetic, logic and comparative operation ..., ACC plays a role which holds the source data and result.

FLAGS

There are four kinds of flag, CF (Carry flag), ZF (Zero flag), SF (Status flag) and GF (General flag), these 4 1-bit flags are affected by the arithmetic, logic and comparative operation.

All flags will be put into stack when an interrupt subroutine is served, and the flags will be restored after RTI instruction executed.

(1) Carry Flag (CF)

The carry flag is affected by following operation :

- a. Addition : CF as a carry out indicator, when the addition operation has a carry-out, CF will be "1", in another word, if the operation has no carry-out, CF will be "0".
- b. Subtraction : CF as a borrow-in indicator, when the subtraction operation must has a borrow, in the CF will be "0", in another word, if no borrow-in, CF will be "1".
- c. Comparison: CF is as a borrow-in indicator for Comparison operation as the same as subtraction operation.
- d. Rotation: CF shifts into the empty bit of accumulator for the rotation and holds the shift out data after rotation.
- e. CF test instruction : For TFCFC instruction, the content of CF sends into SF then clear itself "0". For TTSEFC instruction, the content of CF sends into SF then set itself "1".

(2) Zero Flag (ZF)

ZF is affected by the result of ALU, if the ALU operation generate a "0" result, the ZF will be "1", otherwise, the ZF will be "0".

(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 will be satisfied, otherwise, branch condition will not be satisfied by SF = 0.

(4) General Flag (GF)

GF is a one bit general purpose register which can be set, clear, test by instruction SGF, CGF and TGS.
PROGRAM EXAMPLE :

Check following arithmetic operation for CF, ZF, SF

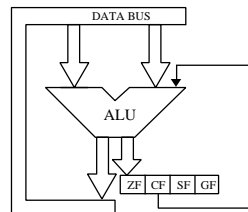
| | CF | ZF | SF |
|------------|----|----|----|
| LDIA #00h; | - | 1 | 1 |
| LDIA #03h; | - | 0 | 1 |
| ADDA #05h; | - | 0 | 1 |
| ADDA #0Dh; | - | 0 | 0 |
| ADDA #0Eh; | - | 0 | 0 |

ALU

The arithmetic operation of 4 - bit data is performed in ALU unit. There are 2 flags can be affected by the result of ALU operation, ZF and SF. The operation of ALU can be affected by CF only.

ALU STRUCTURE

ALU supported user arithmetic operation function, including : addition, subtraction and rotaion.



ALU FUNCTION

(1) Addition:

For instruction ADDAM, ADCAM, ADDM #k, ADD #k,y ALU supports addition function. The addition operation can affect CF and ZF. For addition operation, if the result is "0", ZF will be "1", otherwise, not equal "0", ZF will be "0". When the addition operation has a carry-out, CF will be "1", otherwise, CF will be "0".

EXAMPLE:

| Operation | Carry | Zero |
|-----------|-------|------|
| 3+4=7 | 0 | 0 |
| 7+F=6 | 1 | 0 |
| 0+0=0 | 0 | 1 |
| 8+8=0 | 1 | 1 |

(2) Subtraction:

For instruction SUBM #k, SUBA #k, SBCAM, DECM... ALU supports user subtraction function. The subtraction operation can affect CF and ZF, For subtraction operation, if the result is negative, CF will

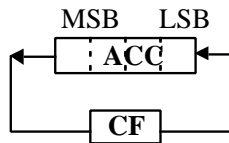
be "0", it means a borrow out, otherwise, if the result is positive, CF will be "1". For ZF, if the result of subtraction operation is "0", the ZF will be "1", otherwise, ZF will be "1".

EXAMPLE:

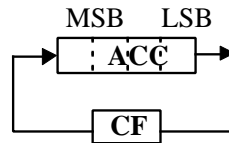
| Operation | Carry | Zero |
|---------------|-------|------|
| 8-4=4 | 1 | 0 |
| 7-F= -8(1000) | 0 | 0 |
| 9-9=0 | 1 | 1 |

(3) Rotation:

There are two kinds of rotation operation, one is rotation left, the other is rotation right. RLCA instruction rotates Acc value to left, shift the CF value into the LSB bit of Acc and the shift out data will be hold in CF.



RRCA instruction operation rotates Acc value to right, shift the CF value into the MSB bit of Acc and the shift out data will be hold in CF.



PROGRAM EXAMPLE: To rotate Acc right and shift a "1" into the MSB bit of Acc.

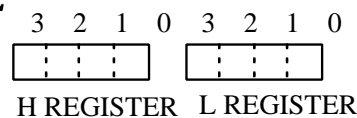
TTCFS; CF ← 1

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

HL REGISTER

HL register are two 4-bit registers, they are used as a pair of pointer for the address of RAM memory and also 2 independent temporary 4-bit data registers. For some instruction, L register can be a pointer to indicate the pin number (Port4, Port6, Port7).

HL REGISTER STRUCTURE



HL REGISTER FUNCTION

(1) For instruction : LDL #k, LDH #k, THA, THL, INCL, DECL, EXAL, EXAH, HL register used as a temporary register.

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

LDL #05h;

LDH #0Dh;

(2) For instruction LDAM, STAM, STAMI ..., HL register used as a pointer for the address of RAM memory.

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

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

(3) For instruction : SELP, CLPL, TFPL, L register be a pointer to indicate the bit of I/O port.

When LR = 0 - 1, indicate P4.0 - P4.1.

PROGRAM EXAMPLE: To set bit 1 of Port 4 to "1"

```
LDL #01h;
SEPL ; P4.1 ← 1
```

STACK POINTER (SP)

Stack pointer is a 4-bit register which stores the present stack level number.

Before using stack, user must set the SP value first, CPU will not initiate the SP value after reset condition.

When a new subroutine is accepted, the SP will be decreased one automatically, in another word, if returning from a subroutine, the SP will be increased one.

The data transfer between ACC and SP is by instruction of "LDASP" and "STASP".

DATA POINTER (DP)

Data pointer is a 12-bit register which stores the address of ROM can indicate the ROM code data specified by user (refer to data ROM).

CLOCK AND TIMING GENERATOR

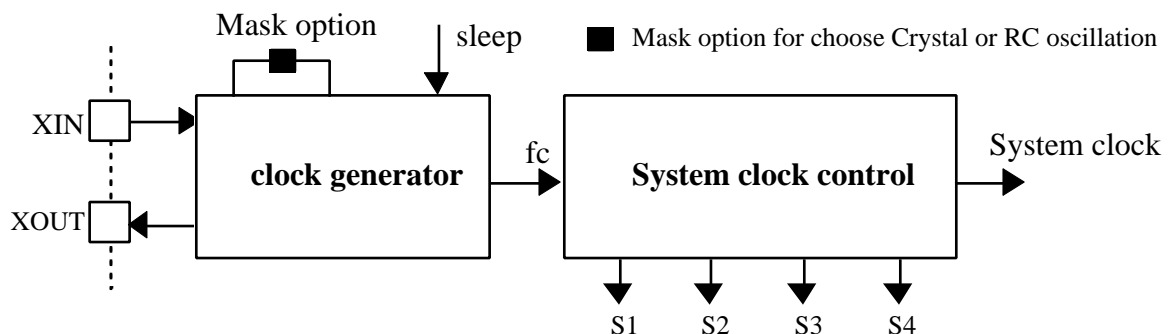
The clock generator is supported by a single clock system, the clock source comes from crystal (resonator) or RC oscillation, the working frequency range is 32 KHz to 100 KHz depending on the working voltage.

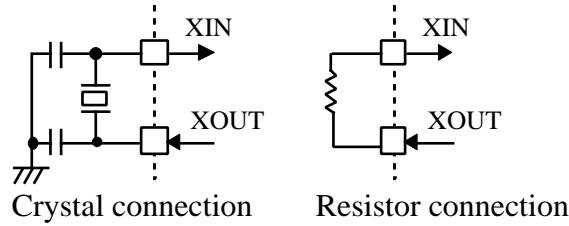
CLOCK AND TIMING GENERATOR STRUCTURE

The clock generator connects outside components (crystal or resonator by XIN and XOUT pin for crystal osc type, capacitor for RC osc type, these two type is decided by mask option) the clock generator generates a basic system clock "fc".

When CPU sleeping, the clock generator will be stopped until the sleep condition released.

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





CLOCK AND TIMING GENERATOR FUNCTION

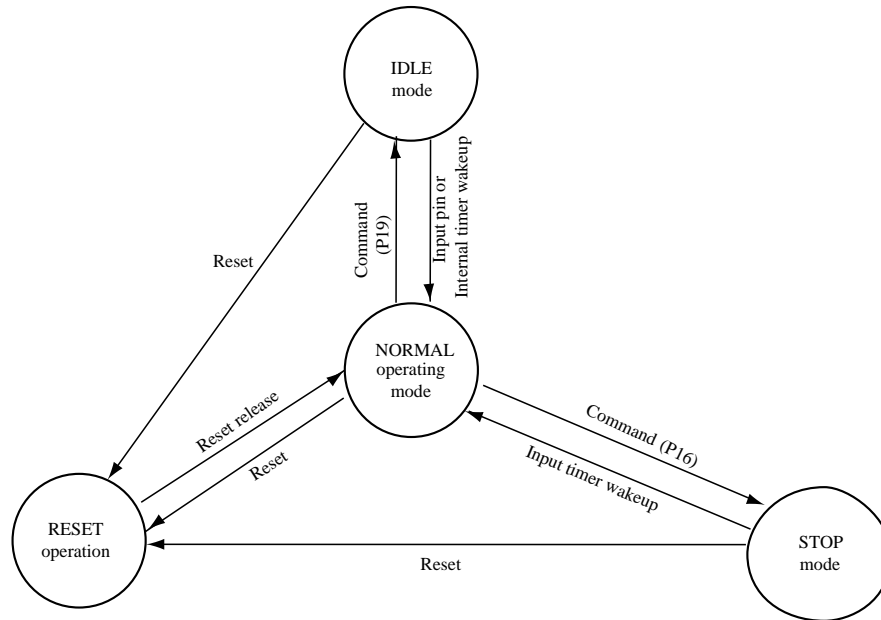
The frequency of f_c is the oscillation frequency for XIN, XOUT by crystal (resonator) or by RC osc. When CPU sleeps, the XOUT pin will be in "high" state.

The instruction cycle equal 4 basic clock f_c .

$$1 \text{ instruction cycle} = 4 / f_c$$

OPERATION MODE CONTROL

EM73P362 has 3 operation modes. They are Normal, Idle, and Stop mode.



| Operation Mode | Oscillator | CPU | Available Function |
|----------------|-------------|------|--------------------------------|
| Normal | Oscillating | Run | LCD, RFC, Low battery detector |
| Idle | Oscillating | Run | LCD |
| Stop | Stop | Stop | All disable |

STOP OPERATION MODE

During STOP operation mode, CPU holds the system's internal status with a low power consumption, for the STOP mode, the system clock will be stopped in the STOP condition and system need a warm up time for the stability of system clock running after wakeup.

The STOP operation mode is controlled by Port 16 and released by P0(0..3)/ $\overline{\text{WAKEUP}}_{0..3}$.

P16 3 2 1 0 Initial value : 0000

| | |
|------|------|
| SPME | SWWT |
|------|------|

| SPME | Enable STOP mode |
|------|------------------|
| 0 1 | Enable STOP mode |
| * * | Reserved |

| SWWT | Set wake-up warm-up time |
|------|--------------------------|
| 0 0 | $2^9 / \text{XIN}$ |
| 0 1 | $2^{14} / \text{XIN}$ |
| 1 0 | $2^{16} / \text{XIN}$ |
| 1 1 | Reserved |

STOP operation mode condition :

1. Osc stop and CPU internal status held.
2. Internal time base clear to "0".
3. CPU internal memory, flags, register, I/O held original states.
4. Program counter hold the executed address after STOP release.

Release condition :

1. Release STOP operation mode by the falling edge of any one of P0(0..3)/ $\overline{\text{WAKEUP}}_{0..3}$.
2. Osc start to oscillating.
3. Warm-up time passing.
4. According PC to execute the following program.

Note : There are 4 independent mask options for wakeup function in EM73P362. So, the wakeup function of P0(0..3)/ $\overline{\text{WAKEUP}}_{0..3}$ are enabled or disabled independently.

IDLE OPERATION MODE

The IDLE operation mode retains the internal status with low power consumption without stopping the system clock function and LCD display.

The IDLE operation mode is controlled by Port 19 and released by P0(0..3)/ $\overline{\text{WAKEUP}}_{0..3}$ or the internal timing generator.

P19 3 2 1 0 Initial value : 0000

| | |
|------|------|
| IDME | SIDR |
|------|------|

| IDME | Enable IDLE mode |
|------|------------------|
| 0 1 | Enable IDLE mode |
| * * | Reserved |

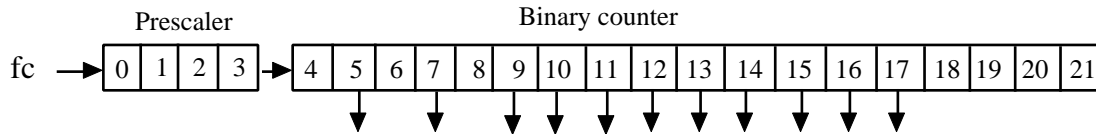
| SIDR | Select IDLE releasing condition |
|------|---|
| 0 0 | P0(0..3) pin input |
| 0 1 | P0(0..3) pin input and 1 sec signal |
| 1 0 | P0(0..3) pin input and 0.5 sec signal |
| 1 1 | P0(0..3) pin input and 15.625m sec signal |

TIMING GENERATOR AND TIME BASE

The timing generator produces the system clock from basic clock pulse which can be normal mode or slow mode clock.

1 instruction cycle = 4 basic clock pulses

There are 22 stages time base.



When working in the single clock mode, the timebase clock source is come from fc.

Time base provides basic frequency for following function:

1. TBI (time base interrupt).
2. Timer/counter, internal clock source.
3. Warm-up time for STOP - mode releasing.

TIME BASE INTERRUPT (TBI)

The time base can be used to generate a fixed frequency interrupt. There are 8 kinds of frequencies can be selected by setting "P25"

Single clock mode

P25

| | | | |
|---|---|---|---|
| 3 | 2 | 1 | 0 |
|---|---|---|---|

 (initial value 0000)

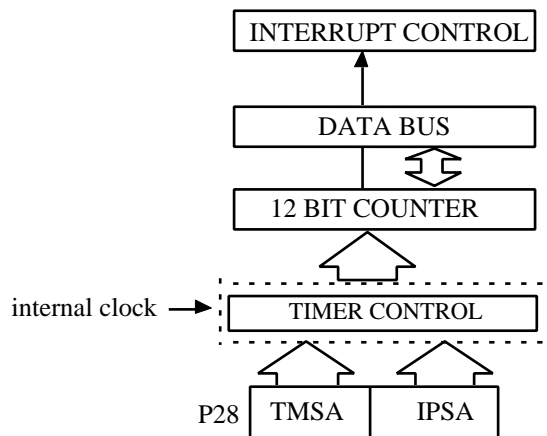
- 0 0 x x: Interrupt disable
- 0 1 0 0: Interrupt frequency $XIN / 2^9$ Hz
- 0 1 0 1: Interrupt frequency $XIN / 2^{10}$ Hz
- 0 1 1 0: Interrupt frequency $XIN / 2^{12}$ Hz
- 0 1 1 1: Interrupt frequency $XIN / 2^{13}$ Hz
- 1 1 0 0: Interrupt frequency $XIN / 2^{14}$ Hz
- 1 1 0 1: Interrupt frequency $XIN / 2^{15}$ Hz
- 1 1 1 0: Interrupt frequency $XIN / 2^{16}$ Hz
- 1 1 1 1: Interrupt frequency $XIN / 2^{17}$ Hz
- 1 0 x x: Reserved

TIMER (TIMERA, TIMERB)

EM73P362 only can support timer function for timerA.

For timerA, the timer data is saved in timer register TAH, TAM, TAL, which user can set timer initial value and read the timer value by instruction "LDATAH(M,L), STATAH(M,L)".

This counter can be set initial value and send counter value to timer register, P28 is the command port for timerA, user can choose different internal clock rate by setting this port. When timer overflows, it will generate a TRGA interrupt request to interrupt control unit.



TIMER CONTROL

Timer command port: P28 is the command port for timerA.

Port 28 3 2 1 0



Initial state: 0000

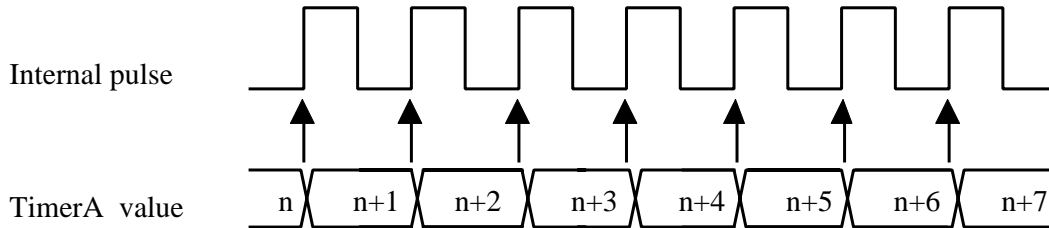
| TMSA | Mode Selection |
|------|----------------|
| 0 0 | Stop |
| 0 1 | Reserved |
| 1 0 | Timer mode |
| 1 1 | Reserved |

| IPSA | Clock rate Selection |
|------|----------------------|
| 0 0 | $XIN/2^5$ Hz |
| 0 1 | $XIN/2^7$ Hz |
| 1 0 | $XIN/2^{11}$ Hz |
| 1 1 | $XIN/2^{15}$ Hz |

TIMER FUNCTION

For timer mode, timerA increase one at any rising edge of internal pulse. User can choose 4 kinds of internal pulse rate by setting IPSA for timerA.

When timerA counts overflow, TRGA will be generated to interrupt control unit.



PROGRAM EXAMPLE: To generate TRGA interrupt request after 60 ms with system clock XIN=32K Hz

```
LDIA #0100B;
EXAE; enable mask 2
EICIL 110111B; interrupt latch ←0, enable EI
LDIA #04H;

STATAL;
LDIA #0CH;
STATAM;
LDIA #0FH;
STATAH;
LDIA #1000B;
OUTA P28; enable timerA with internal pulse rate: XIN/25 Hz
```

NOTE: The preset value of timer/counter register is calculated as following procedure.

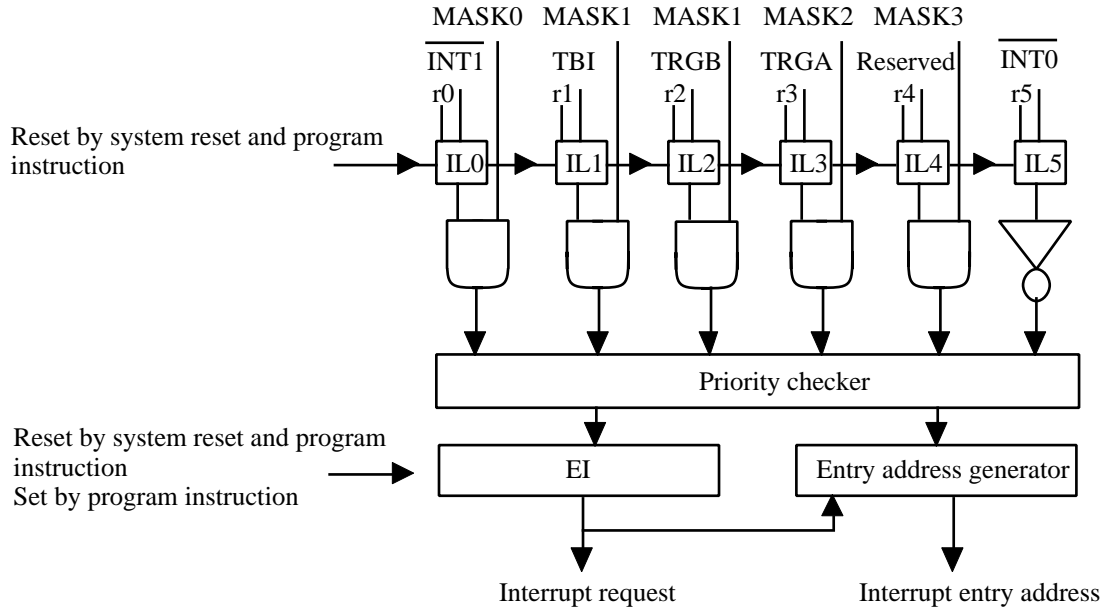
- Internal pulse rate: $XIN/2^5$; $XIN = 32KHz$
- The time of timer counter count one = $2^5 / XIN = 32/32K = 1ms$
- The number of internal pulse to get timer overflow = $60 ms / 1ms = 60 = 03CH$
- The preset value of timer/counter register = $1000H - 03CH = 0FC4H$

INTERRUPT FUNCTION

There are 3 internal interrupt sources and 2 external interrupt sources. Multiple interrupts are admitted according to the priority.

| Type | Interrupt source | Priority | Interrupt Latch | Interrupt Enable condition | Program ROM entry address |
|----------|--|----------|-----------------|----------------------------|---------------------------|
| External | External interrupt ($\overline{INT0}$) | 1 | IL5 | EI=1 | 002H |
| Internal | Reserved | 2 | IL4 | EI=1, MASK3=1 | 004H |
| Internal | TimerA overflow interrupt (TRGA) | 3 | IL3 | EI=1, MASK2=1 | 006H |
| Internal | TimerB overflow interrupt (TRGB) | 4 | IL2 | EI=1, MASK1=1 | 008H |
| Internal | Time base interrupt (TBI) | 5 | IL1 | | 00AH |
| External | External interrupt ($\overline{INT1}$) | 6 | IL0 | EI=1, MASK0=1 | 00CH |

INTERRUPT STRUCTURE



Interrupt controller:

IL0-IL5 : Interrupt latch. Hold all interrupt requests from all interrupt sources. ILr can not be set by program, but can be reset by program or system reset, so IL only can decide which interrupt source can be accepted.

MASK0-MASK3 : MASK register can permit or inhibit all interrupt sources.

EI : Enable interrupt Flip-Flop can permit or inhibit all interrupt sources, when interrupt happened, EI is cleared to "0" automatically, after RTI instruction happened, EI will be set to "1" again.

Priority checker: Check interrupt priority when multiple interrupts happened.

INTERRUPT FUNCTION

The procedure of interrupt operation:

1. Push PC and all flags to stack.
2. Set interrupt entry address into PC.
3. Set SF= 1.
4. Clear EI to inhibit other interrupts happened.
5. Clear the IL for which interrupt source has already be accepted.
6. To 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.

PROGRAM EXAMPLE: To enable interrupt of "TRGA"

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

LCD DRIVER

EM73P362 can directly drive the liquid crystal display (LCD) and has 27 segment, 3 common output pins. There are total 27 x 3 dots can be display. The VDD, VEE, VA, VB and VSS pins are the bias voltage inputs of the LCD driver. The method of LCD programming is RAM mapping.

CONTROL OF LCD DRIVER

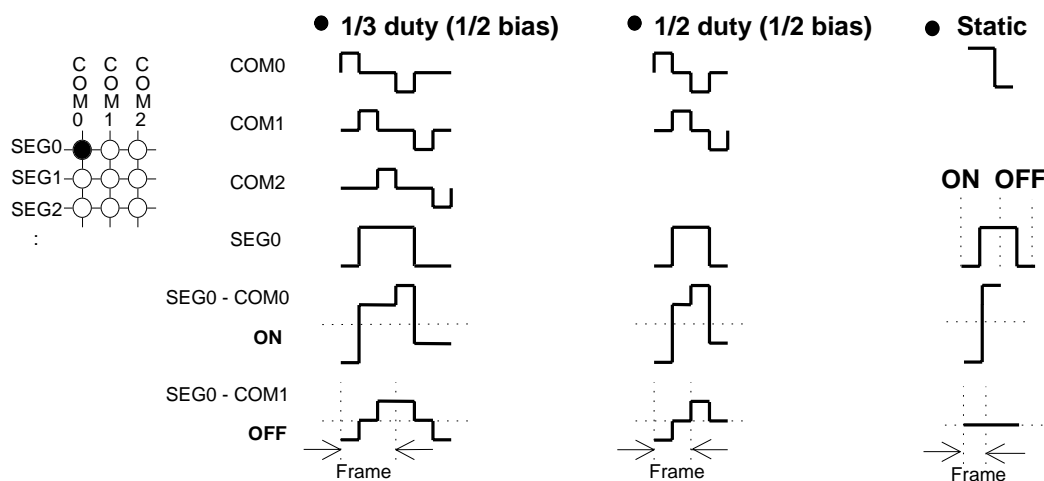
The LCD driver control command register is P27. When LDC is 00, the LCD is disabled and changes the duty only. When LDC is 01, the LCD is blanking, the COM pins are inactive and the SEG pins continuously output the display data. When LDC is 11, the LCD driver enables, the power switch is turned on and it cannot be turned off forever except the CPU is reseted or in the STOP operation mode. Users must enable the LCD driver by self when the CPU is woke up.

P27 3 2 1 0 Initial value : 0000

| LDC | | DUTY | |
|-----|---------------------|------|-----------------------|
| LDC | LCD display control | DUTY | Driving method select |
| 0 0 | LCD display disable | 0 0 | Reserved |
| 0 1 | Blanking | 0 1 | 1/3 duty (1/2 bias) |
| 1 0 | Reserved | 1 0 | 1/2 duty (1/2 bias) |
| 1 1 | LCD display enable | 1 1 | Static |

LCD driving methods

There are four kinds of driving methods can be selected by DUTY (P27.0 ~ P27.1). The driving wave forms of LCD driver are as below :

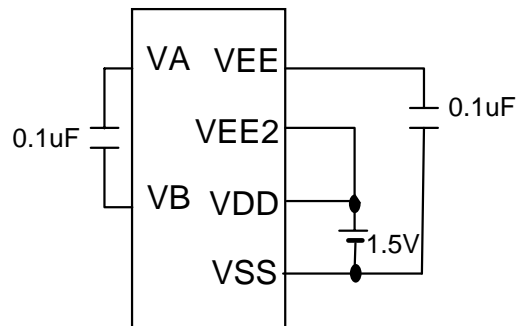


LCD frame frequency : According to the drive method to set the frame frequency.

| Driving method | Frame frequency (Hz) |
|----------------|-------------------------|
| 1/3 duty | $86 \times (3/3) = 86$ |
| 1/2 duty | $86 \times (3/2) = 129$ |
| Static | 86 |

LCD drive voltage

When the power supply is 1.5V, the VEE is connected a capacitor to VSS and the VA is connected a capacitor to VB for the voltage doubler. The output of VEE is $2 \times 1.5V$ for LCD bias voltage.



PROGRAM EXAMPLE

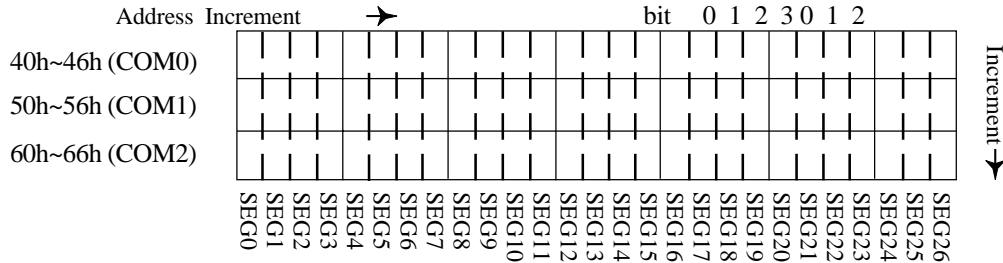
```

LDIA    #0001B           ; set LCD mode 1/3 duty 1/2 bias
OUTA    P27
LDIA    #1100B
OUTA    P27             ; enable LCD
:
:

```

LCD DISPLAY DATA AREA

The LCD display data is stored in RAM from address 40h ~ 46h, 50h ~ 56h and 60h ~ 66h. The relation of data area and COM / SEG pin is as below :



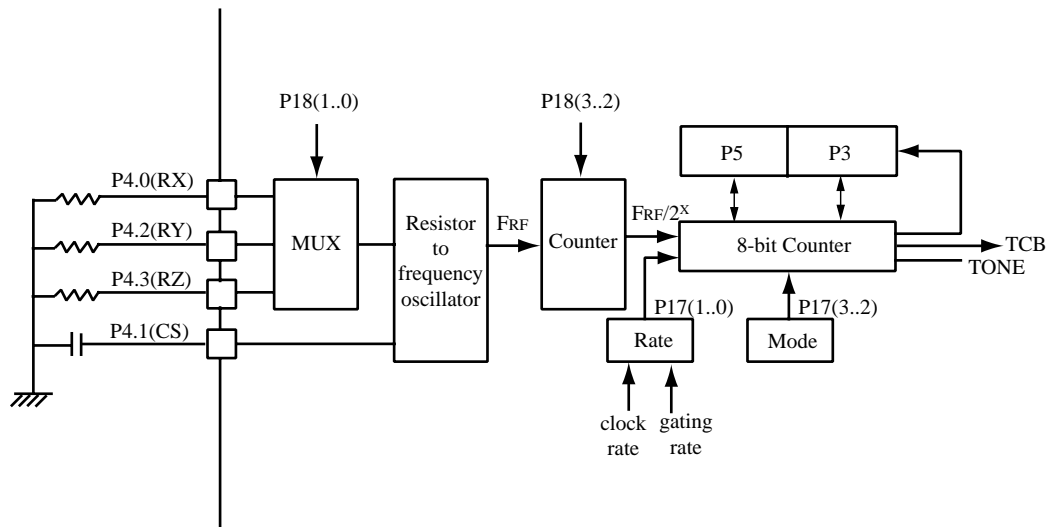
Read automatically by the display data from the display data area and send to the LCD driver by the hardware. Therefore, the display patterns can be changed only by overwriting the contents of the display data area with the software.

The relation between LCD display RAM and driving method

| Driving method | LCD display RAM address | | |
|----------------|-------------------------|-----------|-----------|
| | 40h ~ 46h | 50h ~ 56h | 50h ~ 66h |
| 1/3 duty | COM0 | COM1 | COM2 |
| 1/2 duty | COM0 | COM1 | - |
| Static | COM0 | - | - |

HIGH SPEED COUNTER

EM73P362 has one high speed counter for resistor to frequency oscillation mode, melody mode and auto load timer mode. The resistor to frequency oscillation (RFO) circuit as show below :



CONTROL OF HIGH SPEED COUNTER

The high speed counter is controlled by the command registers (P17, P18) :

P17 3 2 1 0 Initial value : 0000



| MODE | Selection of HTC mode |
|------|--|
| 0 0 | Disable HTC |
| 0 1 | Melody mode |
| 1 0 | Auto load timer mode |
| 1 1 | Resistor to frequency oscillation mode |

| RATE (Hz) | Internal pulse rate / Counter start request frequency | |
|----------------|---|--|
| | Resistor to frequency oscillation mode | Auto load timer mode / Melody mode internal pulse rate |
| 0 0 | $XIN / 2^{10}$ | $XIN / 2^0$ |
| 0 1 | $XIN / 2^{12}$ | $XIN / 2^2$ |
| 1 0 | $XIN / 2^{14}$ | $XIN / 2^4$ |
| 1 1 | $XIN / 2^{15}$ | $XIN / 2^6$ |

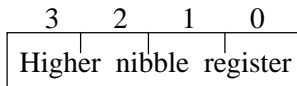
P18 3 2 1 0 Initial value : 0000



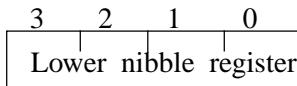
| RFIP | Input frequency of RFO | RFIN | Selection of RFO Pin |
|------|------------------------|------|----------------------|
| 0 0 | $F_{RF} / 2$ | 0 0 | Normal I/O |
| 0 1 | $F_{RF} / 4$ | 0 1 | P4.0 (RX) for RFO |
| 1 0 | $F_{RF} / 8$ | 1 0 | P4.2 (RY) for RFO |
| 1 1 | $F_{RF} / 16$ | 1 1 | P4.3 (RZ) for RFO |

P3 and P5 are the 8-bit binary counter registers of the HTC. P3 is lower nibble register and P5 is higher nibble register.

P5



P3



Initial value : 0000 0000

The HTC consist of one auto-reload and presetable 8-bit binary counter, 12-bit general counter and clock source selectors. The command register can select the internal clock pulse for the melody, auto load counter and resistor to frequency oscillation modes. The HTC increases one at the rising edge of the clock pulses. When the first rising edge occurs by the HTC enabled, the HTC starts counting.

8-BIT BINARY COUNTER

Write the preset value to the registers

The value of 8-bit binary counter can be presetted by P3 and P5. The value of registers can be loaded into the 8-bit binary counter when the counter starts counting or occurs overflow. If you write values to the registers before the next overflow occurs, the preset value can be changed.

Read the count value from the registers

The count value of 8-bit binary counter can be read out from P3 and P5. The value is unstable when you read out the value during counting. Thus, you must disable the counter before reading out the value.

12-BIT GENERAL COUNTER (TCB)

Write the initial value to the registers

The initial value can be written into the 12-bit counter registers by using STATBL, STATBM and STATBH instructions. The value of registers can be loaded into the 12-bit binary counter (TCB) and the TCB increases one when the 8-bit binary counter overflows.

Read the count value from the registers

The count value of 12-bit binary counter can be read out from the counter registers by using LDATBL, LDATBM and LDATBH instructions.

20-BIT COUNTER FUNCTION

The 8-bit binary counter is connected to TCB which is one 12-bit general counter and becomes to the 20-bit counter. The TCB increases one when the 8-bit binary counter overflows and generates an overflow interrupt (TRGB) when the TCB overflows. In this case, the TCB cannot be used as a 12-bit counter alone.

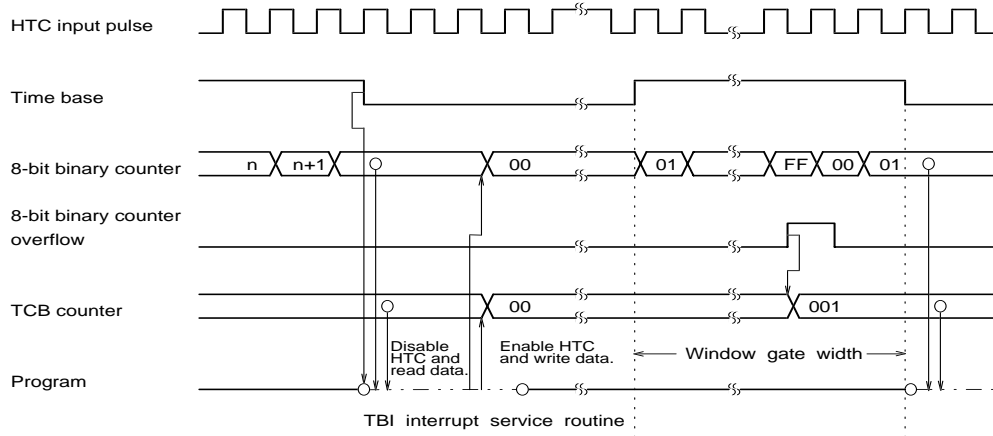
FUNCTION OF HIGH SPEED COUNTER

The HTC has three modes which are RFO mode, melody mode and auto load timer mode. In these mode, the HTC loads the initial values from the counter registers (P3, P5) when it is enabled by P17 and it also can be auto-reloaded the initial values when it overflows. The HTC is counted by the internal pulse and the value of TCB increases one when 8-bit binary counter overflows. The TCB can generate an overflow interrupt (TRGB) when it overflows. The TRGB cannot be generated when the HTC is in the melody mode or disabled.

The HTC is disabled when the CPU is reseted or in the STOP/IDLE operation mode. Users must enable it by self when the CPU is waked up.

Resistor to frequency oscillation mode

In this case, the window gate width interval is from the time base output fall to rise and the value of window gate width setting is the same as the time base interrupt frequency. The time base can be generated a fixed frequency interrupt when the time base interrupt (TBI) is enabled. The content of the HTC can be read and initialized by the TBI interrupt service routine.



ex. TBI interrupt frequency is $XIN/2^{15}$ Hz (P25=1101B). The pulse rate of RFO is $XIN/2^{15}$ Hz (P17=1111B).
The window gate width of RFO is $2^{14}/XIN$ sec.

PROGRAM EXAMPLE

```

DSEG
ORG      00H
RFCON:  RES      1
:
CSEG
ORG      00H
LBR      MAIN          ;initial jump
ORG      0AH
LBR      TBI           ;timebase interrupt vector address
:
:                       ;timebase interrupt service routine
TBI:     CMP      #00H,RFCON
B         TBI1
STD      #01H,RFCON
LDIA     #00H          ;initial TCB & HTC register
OUTA     P5
OUTA     P3
STATBL
STATBM
STATBH
B         TBIEND

```



```

TBI1:  LDIA    #00H           ;disable RFO before reading the counter value
        OUTA   P17
        INA    P3           ;store the counter value to RAM[00] - RAM[04]
        STA    00H
        INA    P5
        STA    01H
        LDATBL
        STA    02H
        LDATBM
        STA    03H
        LDATBH
        STA    04H

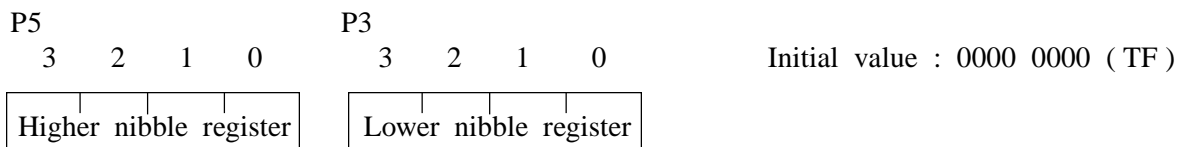
TBIEND: RTI

MAIN:  STD     #00H,RFCON    ;main program
        LDIA   #0001B       ;P4.0 (RX) output
        OUTA   P18
        LDIA   #0010B       ;enable timebase interrupt
        EXAE
        EICIL  0
        LDIA   #1111B       ;enable RFO mode, the window gate width of RFO=214/XIN sec.
        OUTA   P17
        LDIA   #1101B       ;enable timebase, interrupt frequency : XIN / 215 Hz
        OUTA   P25
        :
    
```

Melody mode

The P4.0/ $\overline{\text{TONE}}$ and TONE pins will output the square wave in the melody mode. When the CPU is not in the melody mode, the P4.0/ $\overline{\text{TONE}}$ is high and TONE is low.

The 8-bit tone frequency register is P5 and P3. The tone frequency will be changed when users output the different data to P3. Thus, the data must be output to P5 before P3 when users want to change the 8-bit tone frequency (TF).



** $F_{\text{TONE}} = [(XIN / 2^X) / (100H - TF)] / 2, TF = 0 \sim 255$

** Example : XIN = 32KHz, RATE = 01, TF = 11110000B = 0F0H.

$F_{\text{TONE}} = [(32K \text{ Hz} / 2^2) / (100H - 0F0H)] / 2 = 256 \text{ Hz.}$

PROGRAM EXAMPLE :

```
LDIA          #0FH
OUTA          P5
LDIA          #00H
OUTA          P3
OUTA          P18
LDIA          #0101B   ;enable melody mode
OUTA          P17
:
```

Auto load timer mode

In this mode, there are four different internal pulse rates can be selected by P17. The HTC loads the initial values by the counter registers (P3, P5) and increases at the rising edges of internal pulse generated by the time base. The value of TCB increases one when the high speed counter overflows and generates an overflow interrupt (TRGB) when the TCB overflows.

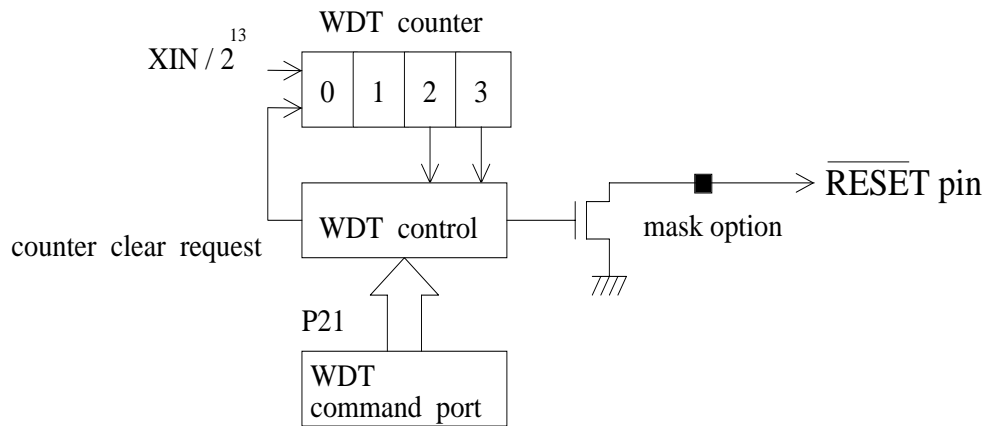
PROGRAM EXAMPLE :

```
LDIA          #00H           ; initial TCB & HTC register
STATBL
STATBM
STATBH
OUTA          P5
OUTA          P3
OUTA          P18
LDIA          #1011B        ; auto load timer mode, internal pulse rate : XIN/26
OUTA          P17
:
LDIA          #00H           ; disable timer mode
OUTA          P17
INA          P3              ; store the counter value to RAM[00] - RAM[04]
STA          00H
INA          P5
STA          01H
LDATBL
STA          02H
LDATBM
STA          03H
LDATBH
STA          04H
```

WATCH-DOG-TIMER (WDT)

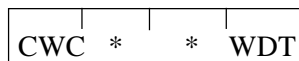
Watch-dog-timer can help user to detect the malfunction (runaway) of CPU and give system a time up signal every certain time. User can use the time up signal to give system a reset signal when system is fail. This function is available by mask option. If the mask option of WDT is enabled, it will stop counting when CPU is reseted or in the STOP operation mode.

The basic structure of watch-dog-timer control is composed by a 4-stage binary counter and a control unit. The WDT counter counts for a certain time to check the CPU status, if there is no malfunction happened, the counter will be cleared and counting. Otherwise, if there is a malfunction happened, the WDT control will send a WDT signal (low active) to reset CPU. The WDT checking period is assign by P21 (WDT command port).



P21 is the control port of watch-dog-timer, and the WDT time up signal is connected to $\overline{\text{RESET}}$.

P21 3 2 1 0 Initial value : 0000



| | |
|-----|--------------------------------|
| CWC | Clear watch-dog-timer counter |
| 0 | Clear counter then return to 1 |
| 1 | Nothing |

| | |
|-----|---|
| WDT | Set watch-dog-timer detect time |
| 0 | $3 \times 2^{13}/\text{XIN} = 3 \times 2^{13}/32\text{K Hz} = 0.75 \text{ sec}$ |
| 1 | $7 \times 2^{13}/\text{XIN} = 7 \times 2^{13}/32\text{K Hz} = 1.75 \text{ sec}$ |

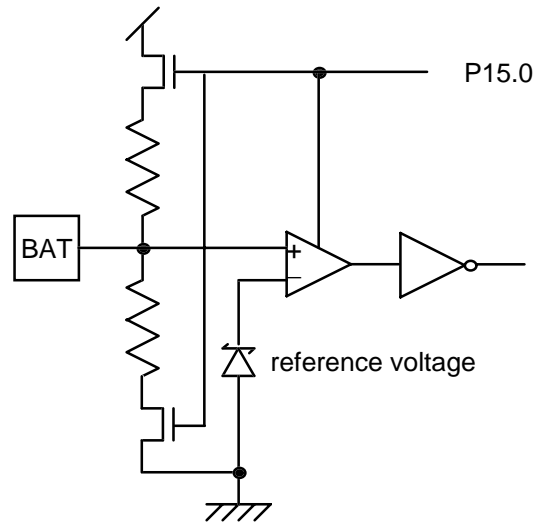
PROGRAM EXAMPLE

To enable WDT with $7 \times 2^{13}/\text{XIN}$ detection time.

```
LDIA    #0001B
OUTA    P21      ;set WDT detection time and clear WDT counter
:
:
```

LOW BATTERY DETECTOR (LBD)

EM73P362 has a built-in low battery detector. This function is disabled when CPU is reseted or in the STOP /IDLE operation mode. User must enable the low battery detector by self when the CPU is waked up. If the low battery detector is enabled, the operating current of whole chip will increase.



CONTROL OF LOW BATTERY DETECTOR

Port15 is the control register of low battery detector. P15.1 (Low battery detector status) is a read-only bit. When LBE is 1, the low battery detector is enabled. When $VDD < 1.25 \pm 0.05V$, SLB is 1.

P15 (write port)

| | | | |
|---|---|---|-----|
| 3 | 2 | 1 | 0 |
| * | * | * | LBE |

P15 (read port)

| | | | |
|---|---|-----|---|
| 3 | 2 | 1 | 0 |
| * | * | SLB | * |

Initial value : **00

| LBE | Low battery detector control |
|-----|------------------------------|
| 0 | Low battery detector disable |
| 1 | Low battery detector enable |

| SLB | Status of low battery detector |
|-----|--|
| 0 | $VDD \geq 1.25 \pm 0.05V$ |
| 1 | $VDD < 1.25 \pm 0.05V$ (Low battery) |

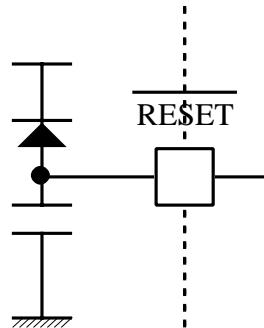
RESETTING FUNCTION

When CPU in normal working condition and $\overline{\text{RESET}}$ pin holds in low level for three instruction cycles at least, then CPU begins to initialize the whole internal states, and when $\overline{\text{RESET}}$ pin changes to high level, CPU begins to work in normal condition.

The CPU internal state during reset condition is as following table :

| Hardware condition in RESET state | Initial value |
|---------------------------------------|-------------------|
| Program counter | 000h |
| Status flag | 01h |
| Interrupt enable flip-flop (EI) | 00h |
| MASK0 ,1, 2, 3 | 00h |
| Interrupt latch (IL) | 00h |
| P3, 5, 15, 16, 17, 18, 19, 21, 25, 27 | 00h |
| P4, 6, 7, 8 | 0Fh |
| XIN | Start oscillation |

The $\overline{\text{RESET}}$ pin is a hysteresis input pin and it has a pull-up resistor available by mask option. The simplest RESET circuit is connect $\overline{\text{RESET}}$ pin with a capacitor to V_{SS} and a diode to V_{DD} .



EM73P362 PORT DESCRIPTION :

| Port | Input function | Output function | Note |
|------|--|--|-------------|
| 0 | E Input port , wake-up function , external interrupt input | | |
| 1 | -- | -- | |
| 2 | -- | -- | |
| 3 | -- | I High speed counter register | low nibble |
| 4 | E Input port , Resistor to frequency oscillation | E Output port, P4.0/TONE | |
| 5 | -- | I High speed counter register | high nibble |
| 6 | E Input port | E Output port , LCD segment pin | |
| 7 | E Input port | E Output port , LCD segment pin | |
| 8 | E Input port | E Output port , LCD segment pin | |
| 9 | -- | -- | |
| 10 | -- | -- | |
| 11 | -- | -- | |
| 12 | -- | -- | |
| 13 | -- | -- | |
| 14 | -- | -- | |
| 15 | I P15.1 (low battery detector status) | I P15.1 (low battery detector control) | |
| 16 | | I Stop mode control register | |
| 17 | | I HTC control register | |
| 18 | | I HTC control register | |
| 19 | | I Idle mode control register | |
| 20 | | -- | |
| 21 | | I WDT control register | |
| 22 | | -- | |
| 23 | | -- | |
| 24 | | -- | |
| 25 | | I Timebase control register | |
| 26 | | -- | |
| 27 | | I LCD control register | |
| 28 | | I Timer A control register | |
| 29 | | -- | |
| 30 | | -- | |
| 31 | | -- | |

ABSOLUTE MAXIMUM RATING

| Items | Sym. | Ratings | Conditions |
|-----------------------|-----------|--------------------------|--------------------------------------|
| Supply voltage | V_{DD} | -0.5V to 2V | |
| Input voltage | V_{IN} | -0.5V to $V_{DD} + 0.5V$ | |
| Output voltage | V_O | -0.5V to $V_{DD} + 0.5V$ | |
| Power dissipation | P_D | 100mW | $T_{OPR} = 50\text{ }^\circ\text{C}$ |
| Operating temperature | T_{OPR} | 0 °C to 50 °C | |
| Storage temperature | T_{STG} | -55 °C to 125 °C | |

RECOMMENDED OPERATING CONDITIONS

| Items | Sym. | Ratings | Conditions |
|----------------|----------|----------------------------------|----------------------|
| Supply voltage | V_{DD} | 1.2V to 1.8V | $F_c = 32\text{KHz}$ |
| Input voltage | V_{IH} | $0.90 \times V_{DD}$ to V_{DD} | |
| | V_{IL} | 0V to $0.10 \times V_{DD}$ | |

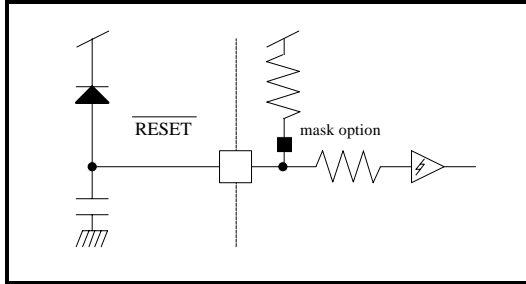
DC ELECTRICAL CHARACTERISTICS ($V_{DD} = 1.5 \pm 0.2V$, $V_{SS} = 0V$, $T_{OPR} = 25^\circ\text{C}$)

| Parameters | Sym. | Min. | Typ. | Max. | Unit | Conditions |
|------------------------|------------|---------------|-----------|---------------|---|---|
| Supply current | I_{DD} | - | 6 | 10 | μA | RC osc. $V_{DD}=1.7V$, $F_c=32\text{KHz}$, no load, |
| | | - | 5 | 8 | μA | X'tal osc. RFO off, LBD off |
| | | - | 4 | 8 | μA | RC osc. $V_{DD}=1.7V$, $F_c=32\text{KHz}$, IDLE |
| | | - | 3 | 6 | μA | X'tal osc. mode, no load |
| | | - | 0.1 | 1 | μA | $V_{DD}=1.7V$, STOP mode |
| | | - | 80 | 250 | μA | X'tal osc. $V_{DD}=1.5V$, $F_c=32\text{KHz}$, RFO on |
| | | 40 | 60 | μA | $V_{DD}=1.5V$, $F_c=32\text{KHz}$, LBD on | |
| Low battery detector | V_{LBD} | 1.30 | 1.35 | 1.40 | V | |
| Frequency of RFO | F_{RF1} | 98 | 115 | 132 | KHz | $R1 = 10\text{K}\Omega$, $V_{DD}=1.5V$ |
| | F_{RF2} | 10.2 | 12 | 13.8 | KHz | $R2 = 100\text{K}\Omega$, $V_{DD}=1.5V$ |
| | F_{RF3} | 1.12 | 1.32 | 1.51 | KHz | $R3 = 1\text{M}\Omega$, $V_{DD}=1.5V$ |
| Frequency ratio of RFO | F_{RF1} | 9 | 9.5 | 10 | | $V_{DD}=1.5V$, $R1=10\text{K}\Omega$, $R2=100\text{K}\Omega$, $R3=1\text{M}\Omega$ |
| | F_{RF2} | | | | | |
| | F_{RF3} | 8.5 | 9 | 9.5 | | |
| Hysteresis voltage | V_{HYS+} | $0.50V_{DD}$ | - | $0.75V_{DD}$ | V | RESET, P0 |
| | V_{HYS-} | $0.20V_{DD}$ | - | $0.40V_{DD}$ | V | |
| Input current | I_{IH} | - | 10 | 15 | μA | P0, Pull-down, $V_{IH}=V_{DD}$ |
| | | -15 | -10 | - | μA | P0, Pull-up, $V_{IH}=V_{SS}$ |
| | | - | - | 1 | μA | P0, None |
| Output voltage | V_{OH} | 1.1 | - | - | V | Push-pull : P4 high current PMOS, TONE $V_{DD}=1.3V$, $I_{OH}=-500\mu\text{A}$ |
| | | 1.1 | - | - | V | Push-pull : P4 low current PMOS, others, $V_{DD}=1.3V$, $I_{OH}=-30\mu\text{A}$ |
| | V_{OL} | - | - | 0.2 | V | $V_{DD}=1.3V$, $I_{OL}=500\mu\text{A}$ |
| Leakage current | I_{LO} | - | - | 1 | μA | Open-drain, $V_{DD}=1.7V$, $V_O=1.7V$ |
| Input resistor | R_{IN} | 50 | 100 | 200 | $\text{K}\Omega$ | RESET |
| LCD bias voltage | V_{EE} | $2V_{DD}-0.1$ | $2V_{DD}$ | $2V_{DD}+0.1$ | V | Voltage doubler |

| Parameters | Sym. | Min. | Typ. | Max. | Unit | Conditions |
|---------------------------------|--------|--------------|----------|--------------|------|--|
| COM, SEG pins output current | Vo_1 | $V_{EE}-0.1$ | V_{EE} | - | V | $Io_1 = -5\mu A$ |
| | Vo_2 | $V_{DD}-0.1$ | V_{DD} | $V_{DD}+0.1$ | V | $Io_2 = \pm 5\mu A$ |
| | Vo_3 | - | V_{SS} | $V_{SS}+0.1$ | V | $Io_3 = 5\mu A$ |
| Frequency stability | | - | 20 | - | % | $F_c=32K$ Hz, RC osc, R=620K Ω , [F(1.5V)-F(1.3V)]/F(1.5V) |
| Frequency variation | | - | 20 | - | % | $F_c=32K$ Hz, $V_{DD}=1.5V$, RC osc, R=620K Ω , [F(typical)-F(worse case)]/F(typical) |

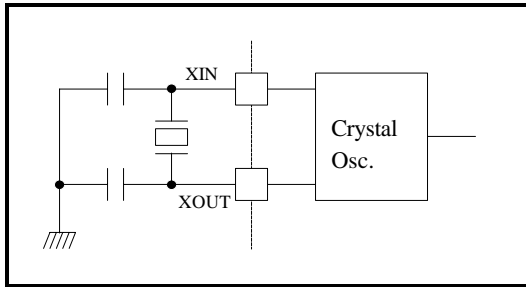
RESET PIN TYPE

TYPE RESET_A

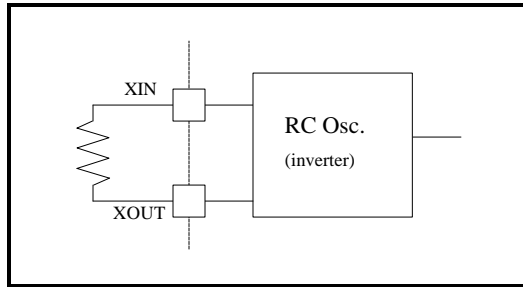


OSCILLATION PIN TYPE

TYPE OSC_A

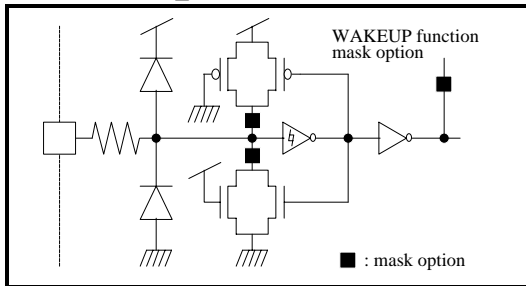


TYPE OSC_F

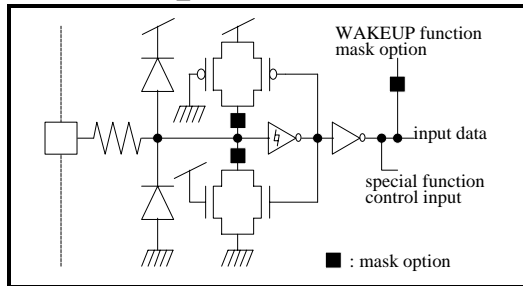


INPUT PIN TYPE

TYPE INPUT_H

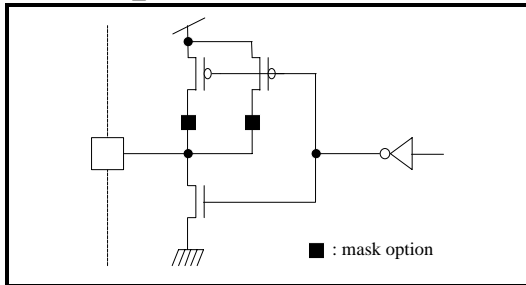


TYPE INPUT_J

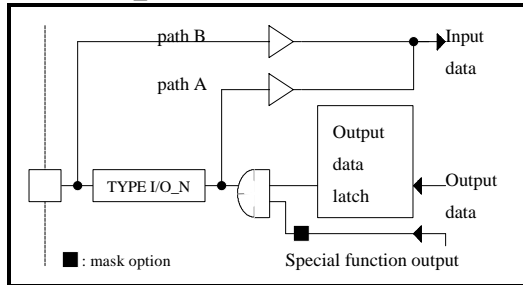


I/O PIN TYPE

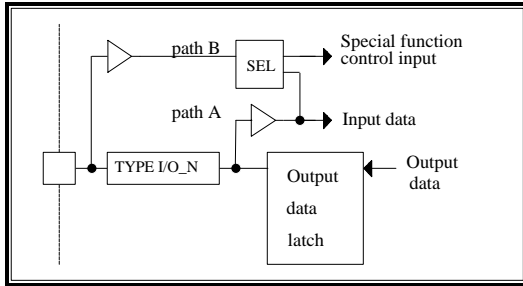
TYPE I/O_N



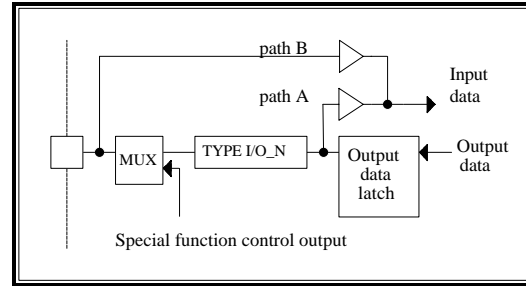
TYPE I/O_O



TYPE I/O_X

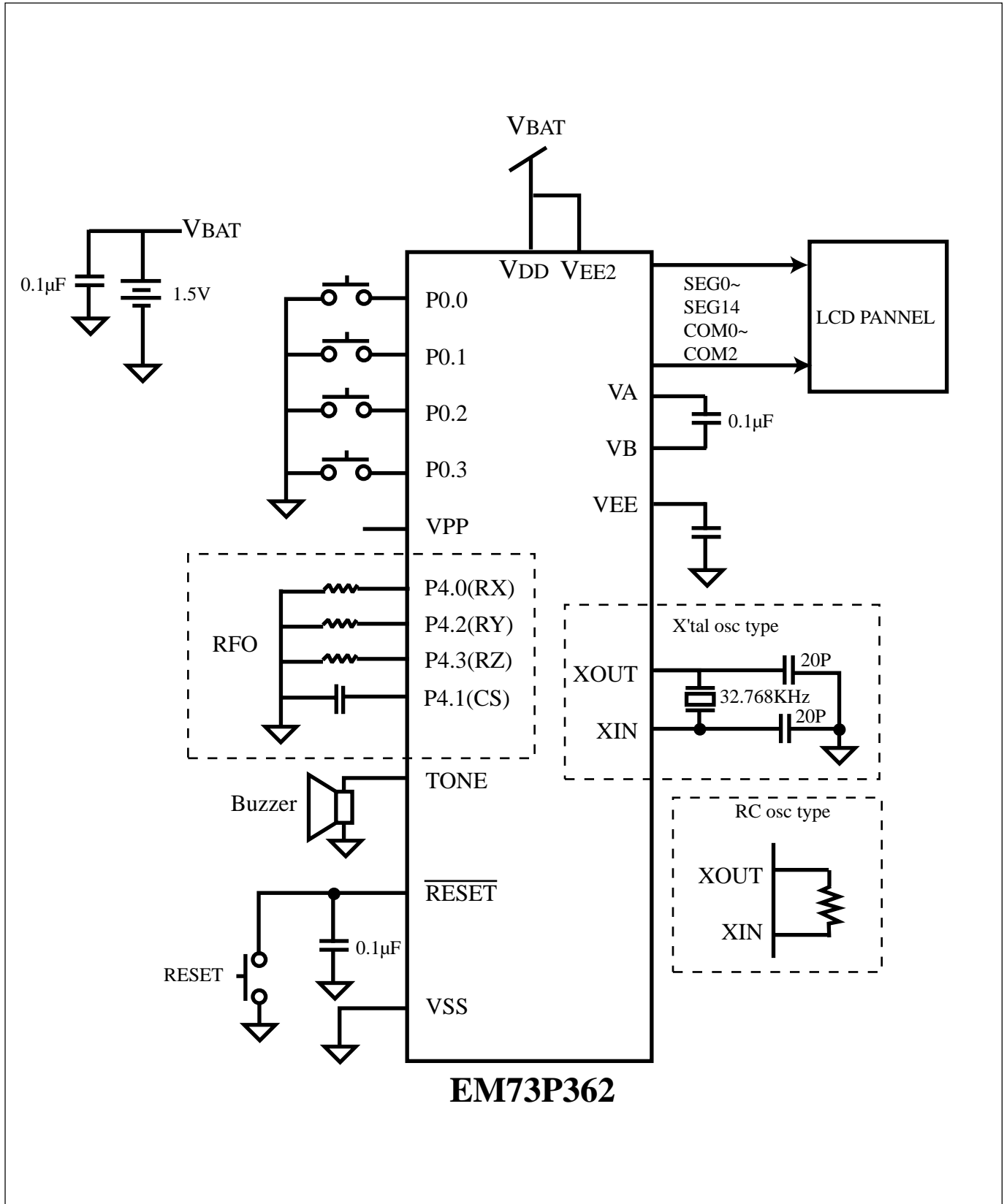


TYPE I/O_Y

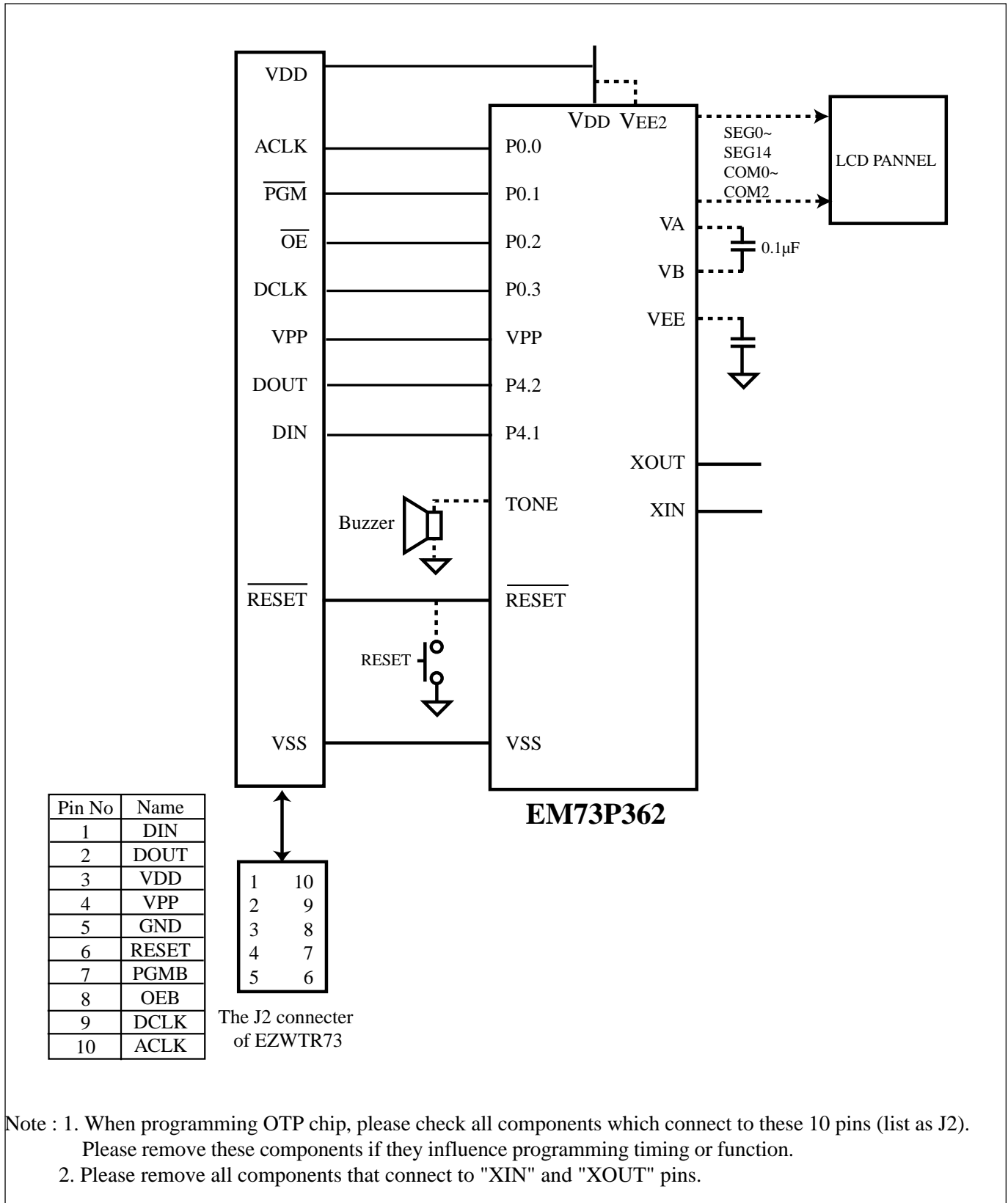


PATH A :For set and clear bit of port instructions, data goes through path A from output data latch to CPU.
 PATH B :For input and test instructions, data from output pin go through path B to CPU and the output data latch will be set to high.

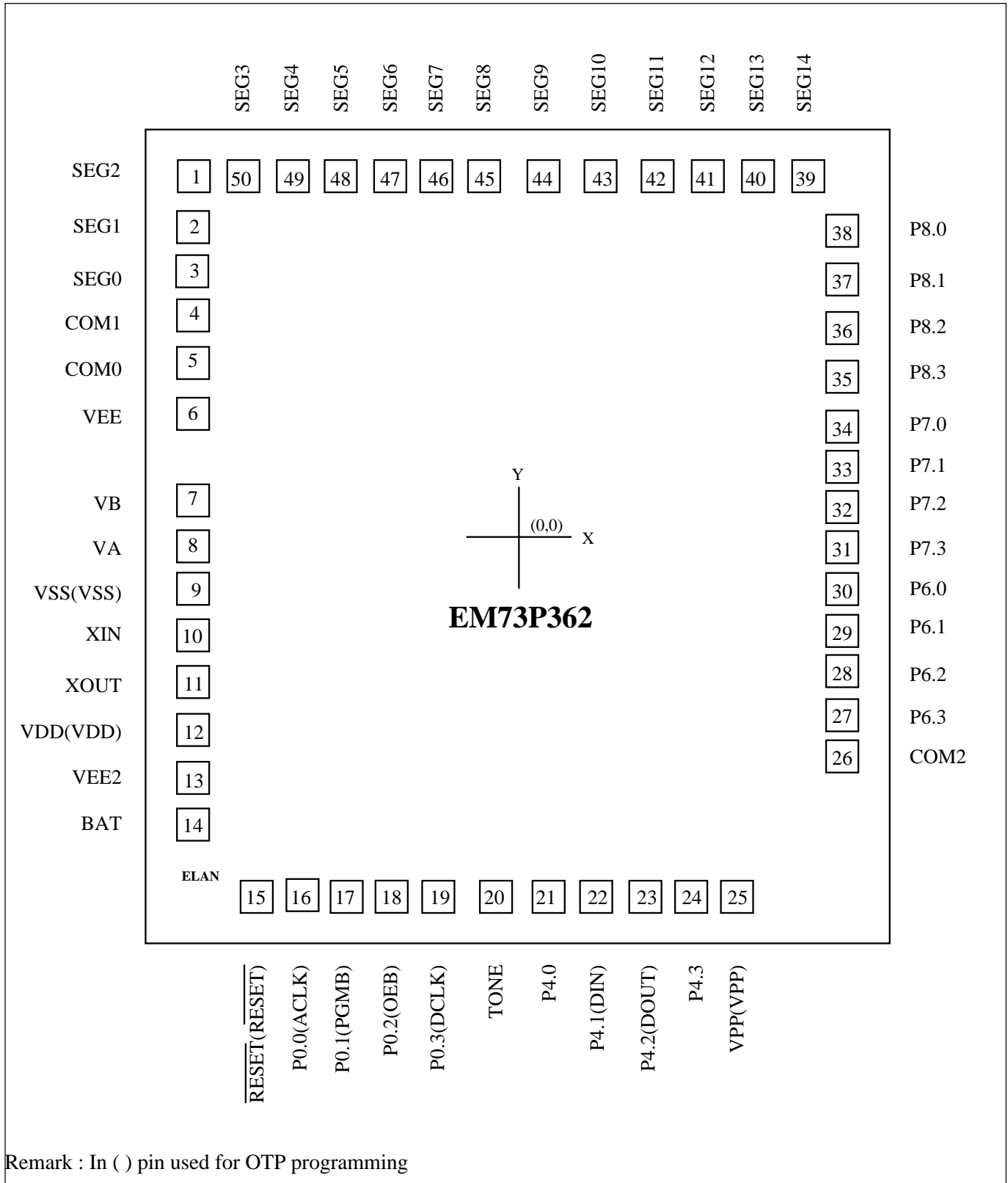
APPLICATION CIRCUIT (NORMAL MODE)



APPLICATION CIRCUIT (OTP PROGRAMMING MODE)



PAD DIAGRAM



Chip Size : 2110 x 2420 μm.

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

| Pad No. | Symbol | X | Y |
|----------------|---------------|----------|----------|
| 1 | SEG2 | -915.0 | 1069.4 |
| 2 | SEG1 | -915.0 | 919.4 |
| 3 | SEG0 | -915.0 | 779.4 |
| 4 | COM1 | -915.0 | 634.4 |
| 5 | COM0 | -915.0 | 494.4 |
| 6 | VEE | -915.0 | 333.0 |
| 7 | VB | -915.0 | 56.5 |
| 8 | VA | -915.0 | -78.2 |
| 9 | VSS | -915.0 | -204.8 |
| 10 | XIN | -915.0 | -324.8 |
| 11 | XOUT | -915.0 | -444.8 |
| 12 | VDD | -915.0 | -565.9 |
| 13 | VEE2 | -915.0 | -685.9 |
| 14 | BAT | -915.0 | -805.9 |
| 15 | RESET | -650.3 | -1080.0 |
| 16 | P0.0 | -530.3 | -1080.0 |
| 17 | P0.1 | -410.3 | -1080.0 |
| 18 | P0.2 | -290.3 | -1080.0 |
| 19 | P0.3 | -165.3 | -1080.0 |
| 20 | TONE | -13.4 | -1080.0 |
| 21 | P4.0 | 127.2 | -1080.0 |
| 22 | P4.1 | 253.1 | -1080.0 |
| 23 | P4.2 | 378.1 | -1079.6 |
| 24 | P4.3 | 498.1 | -1079.6 |
| 25 | VPP | 623.1 | -1080.0 |
| 26 | COM2 | 921.8 | -646.7 |
| 27 | P6.3 | 921.8 | -514.0 |
| 28 | P6.2 | 921.8 | -394.0 |
| 29 | P6.1 | 921.8 | -274.0 |
| 30 | P6.0 | 921.8 | -154.0 |
| 31 | P7.3 | 921.8 | -34.0 |
| 32 | P7.2 | 921.8 | 86.0 |
| 33 | P7.1 | 921.8 | 206.0 |
| 34 | P7.0 | 921.8 | 326.0 |
| 35 | P8.3 | 922.1 | 474.4 |
| 36 | P8.2 | 922.1 | 614.4 |
| 37 | P8.1 | 922.1 | 759.4 |
| 38 | P8.0 | 922.1 | 899.4 |
| 39 | SEG14 | 792.4 | 1069.4 |
| 40 | SEG13 | 652.4 | 1069.4 |

| Pad No. | Symbol | X | Y |
|----------------|---------------|----------|----------|
| 41 | SEG12 | 517.4 | 1069.4 |
| 42 | SEG11 | 387.3 | 1069.4 |
| 43 | SEG10 | 230.0 | 1069.4 |
| 44 | SEG9 | 70.0 | 1069.4 |
| 45 | SEG8 | -90.0 | 1069.4 |
| 46 | SEG7 | -230.0 | 1069.4 |
| 47 | SEG6 | -370.0 | 1069.4 |
| 48 | SEG5 | -510.0 | 1069.4 |
| 49 | SEG4 | -640.0 | 1069.4 |
| 50 | SEG3 | -775.0 | 1069.4 |

INSTRUCTION TABLE

(1) Data Transfer

| Mnemonic | Object code (binary) | Operation description | Byte | Cycle | Flag | | |
|----------|------------------------|--------------------------------|------|-------|------|---|----|
| | | | | | C | Z | S |
| 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 | 2 | - | Z | 1 |
| LDAXI | 0110 0111 | Acc←ROM[DP] _H ,DP+1 | 1 | 2 | - | Z | 1 |
| LDH #k | 1001 kkkk | HR←k | 1 | 1 | - | - | 1 |
| LDHL x | 0100 1110 xxxx xx00 | 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 | 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 | 1 | - | Z | C' |
| THA | 0111 0110 | Acc←HR | 1 | 1 | - | Z | 1 |
| TLA | 0111 0100 | Acc←LR | 1 | 1 | - | Z | 1 |

(2) Rotate

| Mnemonic | Object code (binary) | Operation description | Byte | Cycle | Flag | | |
|----------|------------------------|-----------------------|------|-------|------|---|----|
| | | | | | C | Z | S |
| RLCA | 0101 0000 | ←CF←Acc← | 1 | 1 | C | Z | C' |
| RRCA | 0101 0001 | →CF→Acc→ | 1 | 1 | C | Z | C' |

(3) Arithmetic operation

| Mnemonic | Object code (binary) | Operation description | Byte | Cycle | Flag | | |
|----------|------------------------|------------------------|------|-------|------|---|----|
| | | | | | C | Z | S |
| 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[HL] - Acc - CF' | 1 | 1 | C | Z | C |
| SUBM #k | 0110 1110 1111 kkkk | RAM[HL]←k - RAM[HL] | 2 | 2 | - | Z | C |

(4) Logical operation

| Mnemonic | Object code (binary) | Operation description | Byte | Cycle | Flag | | |
|----------|------------------------|-----------------------|------|-------|------|---|----|
| | | | | | C | Z | S |
| 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' |

(5) Exchange

| Mnemonic | Object code (binary) | Operation description | Byte | Cycle | Flag | | |
|----------|------------------------|---------------------------|------|-------|------|---|---|
| | | | | | C | Z | S |
| EXA x | 0110 1000 xxxx xxxx | Acc↔RAM[x] | 2 | 2 | - | Z | 1 |
| EXAH | 0110 0110 | Acc↔HR | 1 | 2 | - | Z | 1 |
| EXAL | 0110 0100 | Acc↔LR | 1 | 2 | - | Z | 1 |
| EXAM | 0101 1000 | Acc↔RAM[HL] | 1 | 1 | - | Z | 1 |
| EXHL x | 0100 1100 xxxx xx00 | LR↔RAM[x], HR↔RAM[x+1] | 2 | 2 | - | - | 1 |

(6) Branch

| Mnemonic | Object code (binary) | Operation description | Byte | Cycle | Flag | | |
|----------|------------------------|---|------|-------|------|---|---|
| | | | | | C | Z | S |
| SBR a | 00aa aaaa | If SF=1 then PC←PC ₁₁₋₆ .a ₅₋₀ else null | 1 | 1 | - | - | 1 |
| LBR a | 1100 aaaa aaaa aaaa | If SF= 1 then PC←a else null | 2 | 2 | - | - | 1 |

(7) Compare

| Mnemonic | Object code (binary) | Operation description | Byte | Cycle | Flag | | |
|----------|------------------------|-----------------------|------|-------|------|---|----|
| | | | | | C | Z | S |
| 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 |

(8) Bit manipulation

| Mnemonic | Object code (binary) | Operation description | Byte | Cycle | Flag | | |
|----------|------------------------|--|------|-------|------|---|---|
| | | | | | C | Z | S |
| CLM b | 1111 00bb | RAM[HL] _b ←0 | 1 | 1 | - | - | 1 |
| CLP p,b | 0110 1101 11bb pppp | PORT[p] _b ←0 | 2 | 2 | - | - | 1 |
| CLPL | 0110 0000 | PORT[LR ₃₋₂ +4]LR ₁₋₀ ←0 | 1 | 2 | - | - | 1 |
| CLR y,b | 0110 1100 11bb yyyy | RAM[y] _b ←0 | 2 | 2 | - | - | 1 |
| SEM b | 1111 01bb | RAM[HL] _b ←1 | 1 | 1 | - | - | 1 |
| SEP p,b | 0110 1101 01bb pppp | PORT[p] _b ←1 | 2 | 2 | - | - | 1 |
| SEPL | 0110 0010 | PORT[LR ₃₋₂ +4]LR ₁₋₀ ←1 | 1 | 2 | - | - | 1 |
| SET y,b | 0110 1100 01bb yyyy | RAM[y] _b ←1 | 2 | 2 | - | - | 1 |
| TF y,b | 0110 1100 00bb yyyy | SF←RAM[y] _b ' | 2 | 2 | - | - | * |
| TFA b | 1111 10bb | SF←Acc _b ' | 1 | 1 | - | - | * |
| TFM b | 1111 11bb | SF←RAM[HL] _b ' | 1 | 1 | - | - | * |
| TFP p,b | 0110 1101 00bb pppp | SF←PORT[p] _b ' | 2 | 2 | - | - | * |
| TFPL | 0110 0001 | SF←PORT[LR ₃₋₂ +4]LR ₁₋₀ ' | 1 | 2 | - | - | * |
| TT y,b | 0110 1100 10bb yyyy | SF←RAM[y] _b | 2 | 2 | - | - | * |
| TTP p,b | 0110 1101 10bb pppp | SF←PORT[p] _b | 2 | 2 | - | - | * |

(9) Subroutine

| Mnemonic | Object code (binary) | Operation description | Byte | Cycle | Flag | | |
|----------|------------------------|---|------|-------|------|---|---|
| | | | | | C | Z | S |
| LCALL a | 0100 0aaa aaaa aaaa | STACK[SP]←PC, SP←SP -1, PC←a | 2 | 2 | - | - | - |
| SCALL a | 1110 nnnn | STACK[SP]←PC, SP←SP - 1, PC←a, a = 8n + 6 (n=1~15), 0086h (n=0) | 1 | 2 | - | - | - |
| RET | 0100 1111 | SP←SP + 1, PC←STACK[SP] | 1 | 2 | - | - | - |

(10) Input/output

| Mnemonic | Object code (binary) | Operation description | Byte | Cycle | Flag | | |
|----------|------------------------|-----------------------|------|-------|------|---|----|
| | | | | | C | Z | S |
| 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 |

(11) Flag manipulation

| Mnemonic | Object code (binary) | Operation description | Byte | Cycle | Flag | | |
|----------|------------------------|-----------------------|------|-------|------|---|---|
| | | | | | C | Z | S |
| CGF | 0101 0111 | GF←0 | 1 | 1 | - | - | 1 |
| SGF | 0101 0101 | GF←1 | 1 | 1 | - | - | 1 |

| | | | | | | | |
|-------|-----------|--------------|---|---|---|---|---|
| TFCFC | 0101 0011 | SF←CF', CF←0 | 1 | 1 | 0 | - | * |
| TGS | 0101 0100 | SF←GF | 1 | 1 | - | - | * |
| TTCFS | 0101 0010 | SF←CF, CF←1 | 1 | 1 | 1 | - | * |
| TZS | 0101 1011 | SF←ZF | 1 | 1 | - | - | * |

(12) Interrupt control

| Mnemonic | Object code (binary) | Operation description | Byte | Cycle | Flag | | |
|----------|------------------------|-------------------------------------|------|-------|------|---|---|
| | | | | | C | Z | S |
| CIL r | 0110 0011 11rr rrrr | IL←IL & r | 2 | 2 | - | - | 1 |
| DICIL r | 0110 0011 10rr rrrr | EIF←0,IL←IL&r | 2 | 2 | - | - | 1 |
| EICIL r | 0110 0011 01rr rrrr | EIF←1,IL←IL&r | 2 | 2 | - | - | 1 |
| EXAE | 0111 0101 | MASK↔Acc | 1 | 1 | - | - | 1 |
| RTI | 0100 1101 | SP←SP+1,FLAG.PC ←STACK[SP],EIF←1 | 1 | 2 | * | * | * |

(13) CPU control

| Mnemonic | Object code (binary) | Operation description | Byte | Cycle | Flag | | |
|----------|------------------------|-----------------------|------|-------|------|---|---|
| | | | | | C | Z | S |
| NOP | 0101 0110 | no operation | 1 | 1 | - | - | - |

(14) Timer/Counter & Data pointer & Stack pointer control

| Mnemonic | Object code (binary) | Operation description | Byte | Cycle | Flag | | |
|----------|------------------------|------------------------|------|-------|------|---|---|
| | | | | | C | Z | S |
| 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 |

****** SYMBOL DESCRIPTION**

| Symbol | Description | Symbol | Description |
|--|--|--|--|
| HR | H register | LR | L register |
| PC | Program counter | DP | Data pointer |
| SP | Stack pointer | STACK[SP] | Stack specified by SP |
| A _{CC} | Accumulator | FLAG | All flags |
| CF | Carry flag | ZF | Zero flag |
| SF | Status flag | GF | General flag |
| 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 timer/counter A (timer/counter B) register |
| [TA] _M ([TB] _M) | Middle 4-bit of timer/counter A (timer/counter B) register | [TA] _H ([TB] _H) | High 4-bit of timer/counter A (timer/counter B) register |
| ← | Transfer | ↔ | Exchange |
| + | Addition | - | Substraction |
| & | 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 ₁₁₋₆ | Bit 11 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 |
| LR ₃₋₂ | Bit 3 to 2 of LR | | |