Patent Number : 61007 (R.O.C)
Patent Pending : 83216083 (R.O.C)

GENERAL DESCRIPTION

EM73460 is an advanced single chip CMOS 4-bit micro-controller. It contains 4K-byte ROM, 244-nibble RAM, 4-bit ALU, 13-level subroutine nesting, 22-stage time base, two 12-bit timer/counters for the kernel function. EM73460 also contains 5 interrupt sources, 1 input port, 2 bidirection ports, LCD driver (32x4), and sound generator.
Except low-power consumption and high speed, EM73460 also have a sleep mode for power saving function.

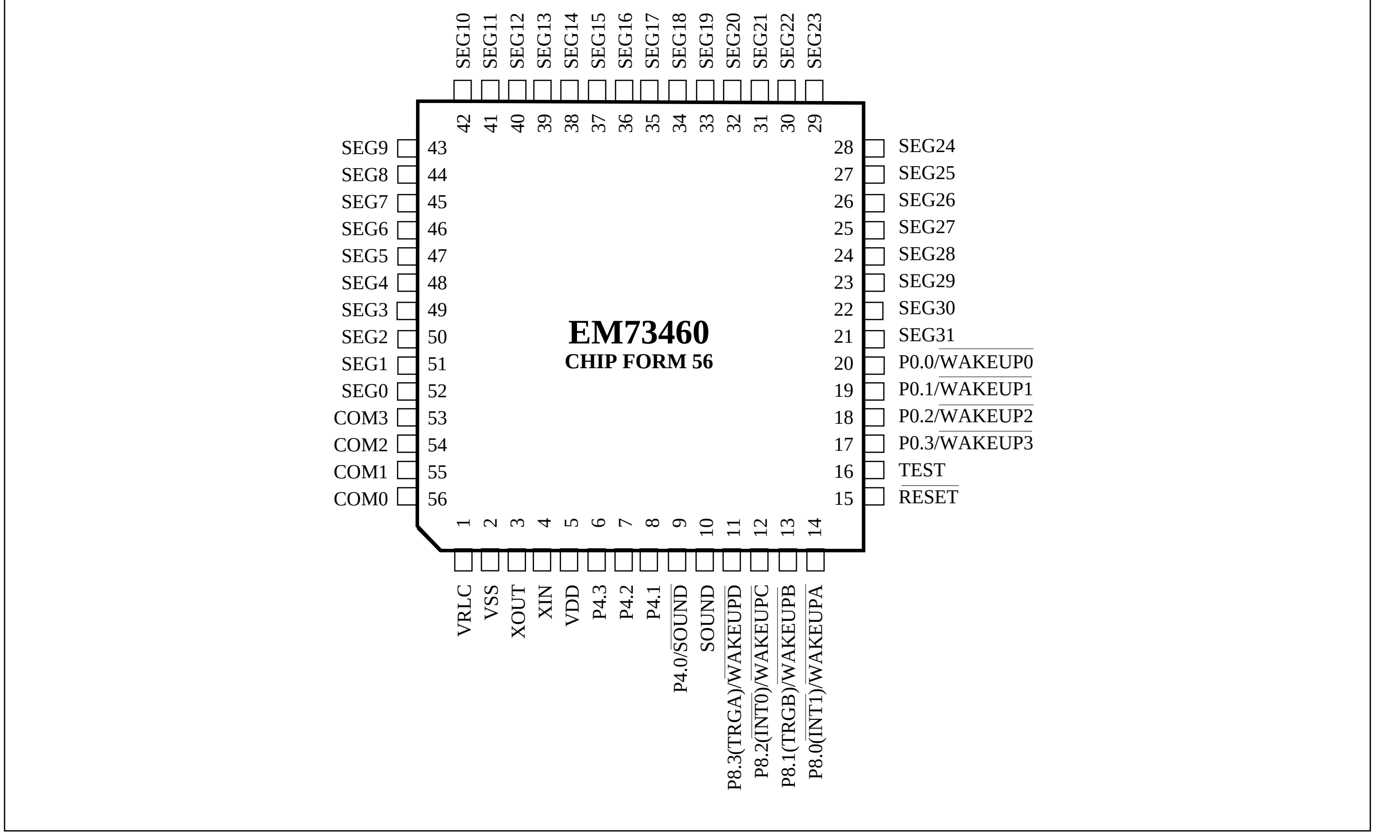
FEATURES

• Operation voltage	: 2.4V to 5.0V.
• Clock source	: Single clock system for RC, Crystal and external clock source available by mask option.
• Oscillation frequency	: 480K, 1M, 2M and 4M Hz are available by mask option.
• Instruction set	: 109 powerful instructions.
• Instruction cycle time	: Up to 2us for 4 MHz.
• ROM capacity	: 4096 X 8 bits.
• RAM capacity	: 244 X 4 bits.
• Input port	: 1 port (P0.0-P0.3) and sleep/hold releasing function are available by mask option. (each input pin is pull-up and pull-down resistor available by mask option).
• Bidirection port	: 2 ports (P4, P8). P4.0 and $\overline{\text{SOUND}}$ is available by mask option. P8(0..3) and sleep/ hold releasing function are available by mask option.
• 12-bit timer/counter	: Two 12-bit timer/counters are programmable for timer, event counter and pulse width measurement.
• Built-in time base counter	: 22 stages.
• Subroutine nesting	: Up to 13 levels.
• Interrupt	: External . . . . . 2 input interrupt sources.
	Internal . . . . . . 2 Timer overflow interrupts.
	1 Time base interrupt.
• LCD driver	: 32 X 4 dots, 1/4,1/3,1/2 static four kinds of duty selectable.
• Sound effect	: There is a built-in sound generator.
• Power saving function	: Sleep mode and Hold mode.
• Package type :	EM73460H Chip form 56 pins.

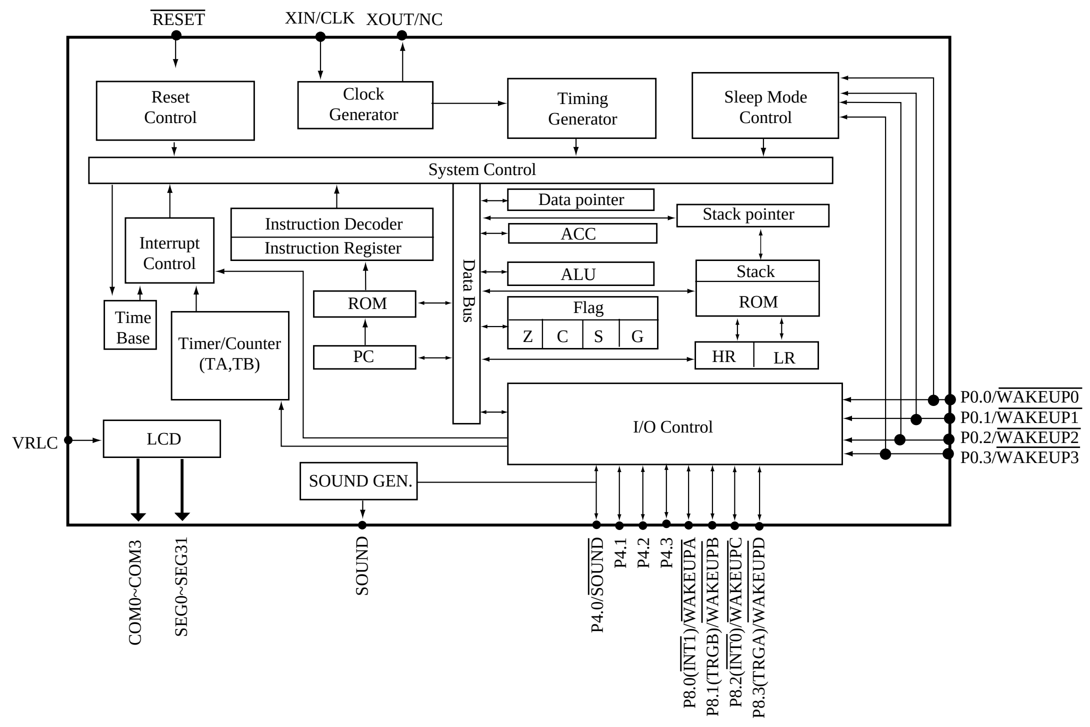
APPLICATIONS

EM73460 is suitable for application in family applicance, consumer products, hand held games and the toy controller.

PIN CONFIGURATIONS



FUNCTION BLOCK DIAGRAM



PIN DESCRIPTIONS

Symbol	Pin-type	Function
V_{DD}		Power supply (+)
V_{SS}		Power supply (-)
$\overline{\text{RESET}}$	RESET-A	System reset input signal, low active mask option : none pull-up
XIN/CLK	OSC-A/OSC-C	Crystal/RC or external clock source connecting pin
XOUT/NC	OSC-A/OSC-C	Crystal/RC connecting pin
P0(0..3)/$\overline{\text{WAKEUP0..3}}$	INPUT-B	4-bit input port with Sleep/Hold function mask option : wakeup enable, pull-up wakeup enable, none wakeup disable, pull-up wakeup disable, pull-down wakeup disable, none
P4.0/$\overline{\text{SOUND}}$	I/O-O	1-bit bidirection I/O port or inverse sound effect output mask option : SOUND enable, push-pull, high current PMOS $\overline{\text{SOUND}}$ disable, open-drain $\overline{\text{SOUND}}$ disable, push-pull, high current PMOS $\overline{\text{SOUND}}$ disable, push-pull, low current PMOS
P4(1..3)	I/O-N	3-bit bidirection I/O port with high current source mask option : open-drain push-pull, high current PMOS push-pull, low current PMOS
P8.0($\overline{\text{INT1}}$)/$\overline{\text{WAKEUPA}}$ P8.2(INT0)/$\overline{\text{WAKEUPC}}$	I/O-L	2-bit bidirection I/O port with external interrupt sources input and Sleep /Hold releasing function mask option : wakeup enable, push-pull wakeup disable, push-pull wakeup disable, open-drain
P8.1(TRGB)/$\overline{\text{WAKEUPB}}$ P8.3(TRGA)/$\overline{\text{WAKEUPD}}$	I/O-L	2-bit bidirection I/O port with time/counter A,B external input and Sleep /Hold releasing function mask option : wakeup enable, push-pull wakeup disable, push-pull wakeup disable, open-drain
SOUND		Built-in sound effect output
VRLC		LCD regulator voltage input
COM0~COM3		LCD common output pins
SEG0~SEG31		LCD segment output pins
TEST		Test pin must be connected to V_{SS}

FUNCTION DESCRIPTIONS

PROGRAM ROM (4K X 8 bits)

4 K x 8 bits program ROM contains user's program and some fixed data .
The basic structure of program ROM can be divided into 5 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 - FFFh : Except used as above function, the other region can be used as user's program region.

address	4096 x 8 bits
000h	Reset start address
002h	INT0; External interrupt service routine entry address
004h	
006h	TRGA; Timer/counterA 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	
086h	SCALL, subroutine call entry address
:	:
FFFh	

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 two ways.

(1) 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 $\quad$ **Acc ← ROM[DP]$_L$**
LDAXI $\quad$ **Acc ← ROM[DP]$_H$,DP+1**

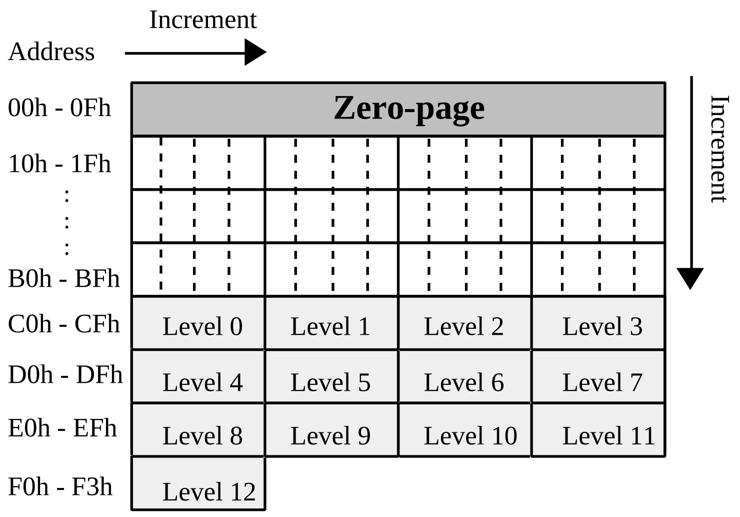
DP is a 12-bit data register which can store the program ROM address to be the pointer for the ROM code data. First, user load ROM address into DP by instruction "LDADPL, LDADPM, LDADPH", then user can get the lower nibble of ROM code data by instruction "LDAX" and higher nibble by instruction "LDAXI"

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 (244-nibble)

There is total 244 - nibble data RAM from address 00 to F3h
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 y,b".

PROGRAM EXAMPLE: To wirte 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 arithematic, comparsion or bit manupulated operation directly.
For example: STD #k,y ; RAM[y] ← #k
ADD #k,y; RAM[y] ← RAM[y] + #k

PROGRAM COUNTER (4K 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 4K - byte size ROM, PC can indicate address form 000h - FFFh, for BRANCH and CALL instrcutions, 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 satisified)

PC	Hold original PC value+1	a	a	a	a	a	a

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

PC	Original PC value + 1

LBR a
Object code: 1100 aaaa aaaa aaaa
Condition: SF=1; PC ← a (branch condition satisified)

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

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

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

LCALL a
Object code: 0100 0 aaa aaaa aaaa
Condition: PC ← a

PC	0	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

RT I
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:

$\overline{\text{INT0}}$ (External interrupt from P8.2)

PC	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	1	1	0

TRGB (Time B overflow interrupt)

PC	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	1	0	1	0

$\overline{\text{INT1}}$ (External interrupt from P8.0)

PC	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

(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 arithematic, logic and comparative opertion .., 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 arithematic, 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. Comparision: CF is as a borrow-in indicator for Comparision 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 TTSFC 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 satisified, otherwise, branch condition will not be satisified 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 arithematic operation for CF, ZF, SF

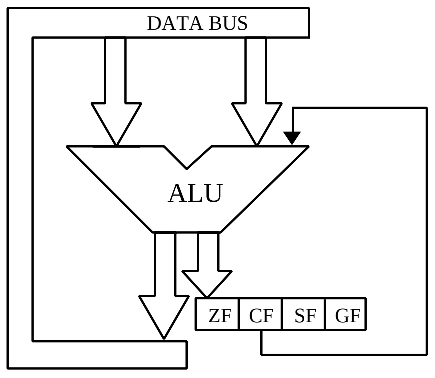
	CF	ZF	SF
LDIA #00h;	-	1	1
LDIA #03h;	-	0	1
ADDA #05h;	-	0	1
ADDA #0Dh;	-	0	0
ADDA #0Eh;	-	0	0

ALU

The arithematic 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 arithematic 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:

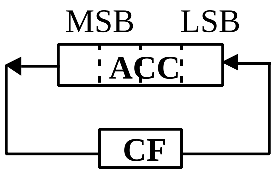
For instruction SUBM #k, SUBA #k, SBCAM, DECM... ALU supports user subtraction function . The subtraction operation can affect CF and ZF, For subtraction operation, if the result is negative, CF will be "0", it means a borrow out, otherwise, if the result is positive, CF will be "1". For ZF, if the result of subtraction operation is "0", the ZF will be "1", otherwise, ZF will be "1".

EXAMPLE:

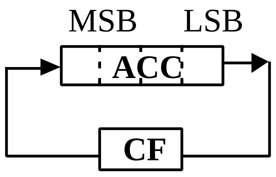
Operation	Carry	Zero
8-4=4	1	0
7-F= -8(1000)	0	0
9-9=0	1	1

(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.



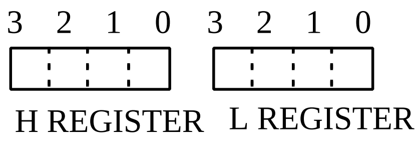
PROGRAM EXAMPLE: To rotate Acc right and shift a "1" into the MSB bit of Acc .

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) .

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 regieter be a pointer to indicate the bit of I/O port.

When LR = 0 indicate P4.0

PROGRAM EXAMPLE: To set bit 0 of Port4 to "1"

```
LDL #00h;
SEPL ; P4.0 ← 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)

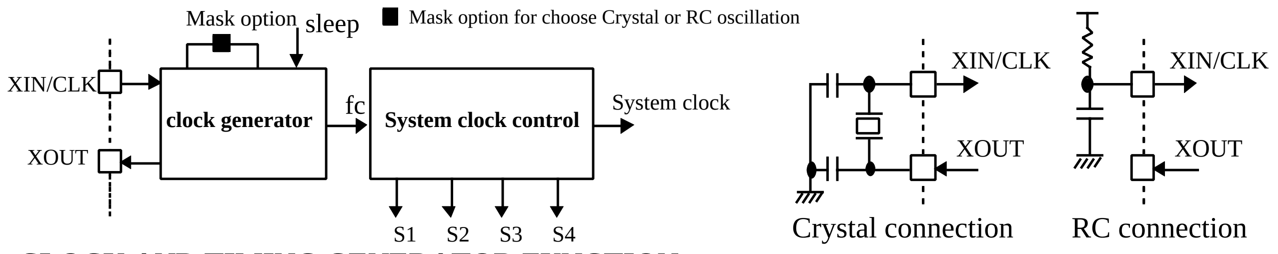
Data pointer is a 12-bit register which stores the address of ROM can indicate the ROM code data specified by user (refer to data ROM).

CLOCK AND TIMING GENERATOR

The clock generator is supported by a single clock system, the clock source comes from crystal (resonator) or RC oscillation is decided by mask option, the working frequency range is 480 K Hz to 4 MHz depending on the working voltage.

CLOCK AND TIMING GENERATOR STRUCTURE

The clock generator connects outside compoments (crystal or resonator by XIN and XOUT pin for crystal osc. type, Resistor and capacitor by CLK pin 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 stoped 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 fc is the oscillation frequency for XIN, XOUT by crystal (resonator) or for CLK by RC osc. When CPU sleeps, the XOUT pin will be in "high" state . When user choose RC osc, XOUT pin is no used. The instruction cycle equal 8 basic clock fc.

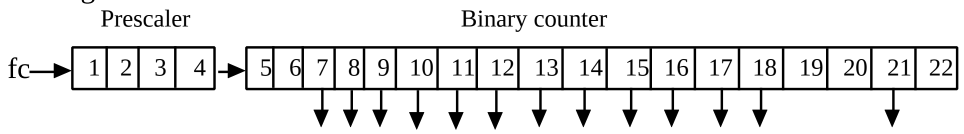
1 instructure cycle = 8 / fc

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 = 8 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 sleep - 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)

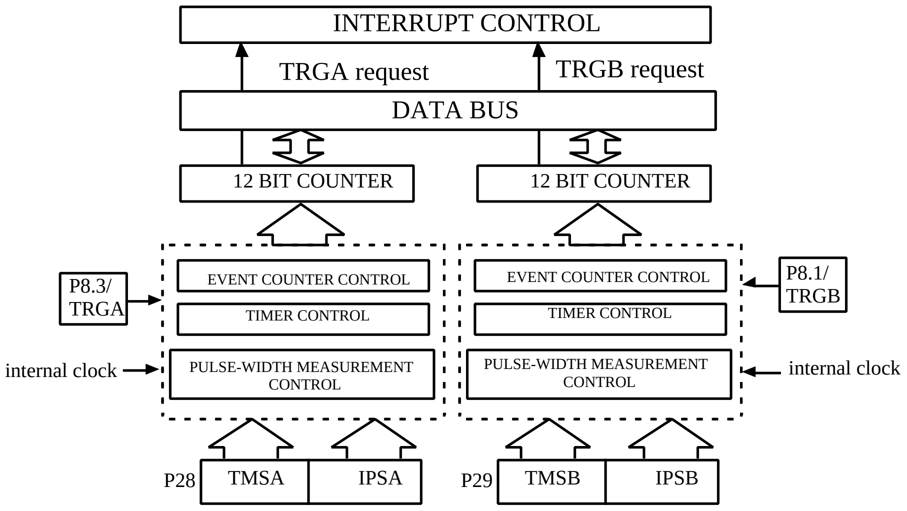
0 0 x x: Interrupt disable
0 1 0 0: Interrupt frequency XIN / 2^{10} Hz
0 1 0 1: Interrupt frequency XIN / 2^{11} Hz
0 1 1 0: Interrupt frequency XIN / 2^{12} Hz
0 1 1 1: Interrupt frequency XIN / 2^{13} Hz
1 1 0 0: Interrupt frequency XIN / 2^{9} Hz
1 1 0 1: Interrupt frequency XIN / 2^{8} Hz
1 1 1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

TIMER / COUNTER (TIMERA, TIMERB)

Timer/counters can support user three special functions:
1. Even counter
2. Timer.
3. Pulse-width measurement.

These three functions can be executed by 2 timer/counter independently.
For timerA, the counter data is saved in timer register TAH, TAM, TAL, which user can set counter initial value and read the counter value by instruction "LDATAH(M,L), STATAH(M,L)" and timer register is TBH, TBM, TBL and W/R instruction "LDATBH (M,L), STATBH (M,L)".

The basic structure of timer/counter is composed by two same structure counter, these two counters can be set initial value and send counter value to timer register, P28 and P29 are the command ports for timerA and timer B, user can choose different operation mode and different internal clock rate by setting these two ports. When timer/counter overflow, it will generate a TRGA(B) interrupt request to interrupt control unit.



TIMER/COUNTER CONTROL

P8.1/TRGB, P8.3/TRGA are the external timer inputs for timerB and timerA, they are used in event counter and pulse-width measurement mode.

Timer/counter command port: P28 is the command port for timer/counterA and P29 is for the timer/counterB.

Port 28

3	2	1	0
TMSA		IPSA	

Initial state: 0000

Port 29

3	2	1	0
TMSB		IPSB	

Initial state: 0000

TIMER/COUNTER MODE SELECTION	
TMSA (B)	Function description
0 0	Stop
0 1	Event counter mode
1 0	Timer mode
1 1	Pulse width measurement mode

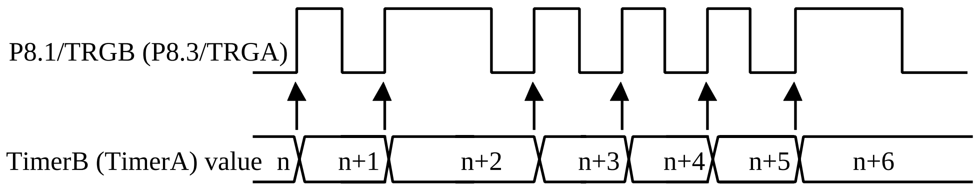
INTERNAL PULSE-RATE SELECTION	
IPSA(B)	Function description
0 0	XIN/2^{10} Hz
0 1	XIN/2^{14} Hz
1 0	XIN/2^{18} Hz
1 1	XIN/2^{22} Hz

TIMER/COUNTER FUNCTION

Timer/counterA can be programmable for timer, event counter and pulse width measurement. Each timer/ counter can execute any one of these functions independly.

EVENT COUNTER MODE

For event counter mode, timer/counter increases one at any rising edge of P8.1/TRGB for timerB (P8.3/ TRGA for timer A). When timerB (timerA) counts overflow, it will give interrupt control an interrupt request TRGB (TRGA).

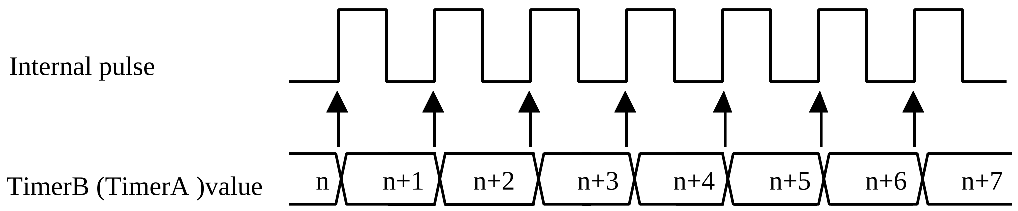


PROGRAM EXAMPLE: Enable timerA with P28

```
LDIA #0100B;
OUTA P28; Enable timerA with event counter mode
```

TIMER MODE

For timer mode ,timer/counter increase one at any rising edge of internal pulse . User can choose 4 kinds of internal pulse rate by setting IPSB for timerB (IPSA for timerA).
When timer/counter counts overflow, TRGB (TRGA) will be generated to interrupt control unit.



PROGRAM EXAMPLE: To generate TRGA interrupt request after 60 ms with system clock XlN=4MHz

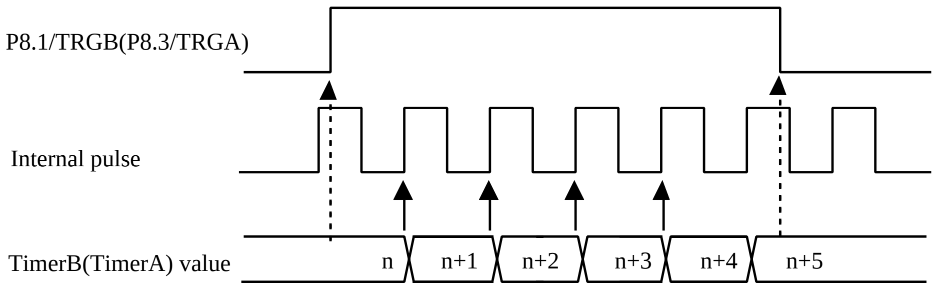
```
LDIA #0100B;
EXAE;     enable mask 2
EICIL 110111B; interrupt latch ←0, enable EI
LDIA #06H;
```

```
STATAL;
LDIA #01H;
STATAM;
LDIA #0FH;
STATAH;
LDIA #1000B;
OUTA P28; enable timerA with internal pulse rate: XIN/2^10 Hz
```

NOTE: The preset value of timer/counter register is calculated as following procedure.
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 = 0F16H

PULSE WIDTH MEASUREMENT MODE

For the pulse width measurement mode, the counter only incresed by the rising edge of internal pulse rate as external timer/counter input (P8.1/TRGB, P8.3/TRGA), interrupt request will be generated as soon as timer/counter count overflow.



PROGRAM EXAMPLE: Enable timerA by pulse width measurement mode .

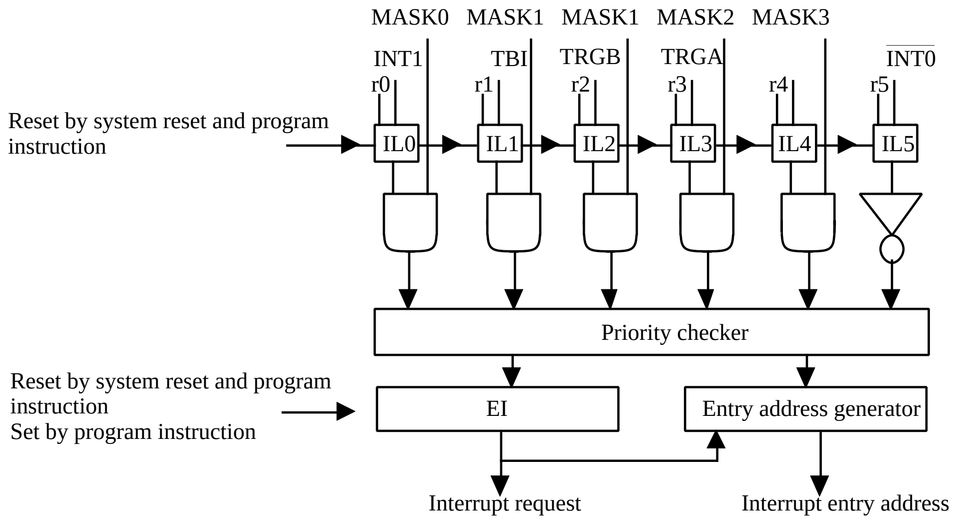
```
LDIA #1100B;
OUTA P28; Enable timerA with pulse width measurement mode.
```

INTERRUPT FUNCTION

There are 5 interrupt sources, 2 external interrupt sources, 3 internal interrupt sources . Multiple interrupts are admitted according 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	: Except $\overline{\text{INT0}}$,MASK register can promit or inhibit all interrupt sources.
EI	: Enable interrupt Flip-Flop can promit 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 excute 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 "$\overline{\text{INT0}}$, TRGA"

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

POWER SAVING FUNCTION (Sleep / Hold functlon)

During sleep and hold condition, CPU holds the system’s internal status with a low power consumption, for the sleep mode, the system clock will be stoped in the sleep condition and system need a warm up time for the stability of system clock running after wakeup . In the other way, for the hold mode, the system clock

does not stop at all and it does not need a warm-up time any way.
The sleep and hold mode is controlled by Port 16 and released by P0(0..3)/$\overline{\text{WAKEUP0..3}}$ or P8(0..3)/ $\overline{\text{WAKEUPA..D}}$.

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

SWWT	Set wake-up warm-up time
0 0	2^{18} /XIN
0 1	2^{14} /XIN
1 0	2^{16} /XIN
1 1	Hold mode

SE	Enable sleep/hold
0	Reserved
1	Enable sleep / hold rnode

Sleep and hold condition:

1. Osc stop (sleep only) 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 sleep release.

Release condition:

1. Osc start to oscillating.(sleep only).
2. Warm-up time passing (sleep only).
3. According PC to execute the following program.

There is one kind of sleep/hold release mode .

1. Edge release mode:
   Release sleep/hold condition by the falling edge of any one of P0(0..3)/$\overline{\text{WAKEUP0..3}}$ or P8(0..3)/ $\overline{\text{WAKEUPA..D}}$.

   Note : There are 8 independent mask options for wakeup function in EM73460. So, the wakeup function of P0(0..3)/$\overline{\text{WAKEUP0..3}}$ and P8(0..3)/$\overline{\text{WAKEUPA..D}}$ are enabled or disabled independently.

LCD DRIVER

EM73460 can directly drive the liquid crystal display (LCD) and has 32 segment, 4 common output pins. There are total 32 x 4 dots can be display. The VRLC pin is the LCD driver power input, there is the voltage of (V_{CC}-VRLC) to LCD.

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 off forever except the CPU is reseted or sleeping. User must enable the LCD driver by self when the CPU is waked up.

Port27 3 2 1 0 Initial value : 0000

LDC	DUTY

LDC	LCD display control
0 0	LCD display disable & change duty
0 1	Blanking
1 0	Reserved
1 1	LCD display enable

DUTY	Driving method select
0 0	1/4 duty (1/3 bias)
0 1	1/3 duty (1/2 bias)
1 0	1/2 duty (1/2 bias)
1 1	Static

The frame frequency of LCD driver can be selected by P26.

Port26 3 2 1 0 Initial value : 0010

LDA	LFF

LCD DISPLAY AREA	
LDA	RAM address
0 0 0	00 - 1FH
0 0 1	20 - 3FH
0 1 0	40 - 5FH
0 1 1	60 - 7FH
1 0 0	80 - 9FH
1 0 1	A0 - BFH
1 1 0	C0 - DFH
1 1 1	Reserved

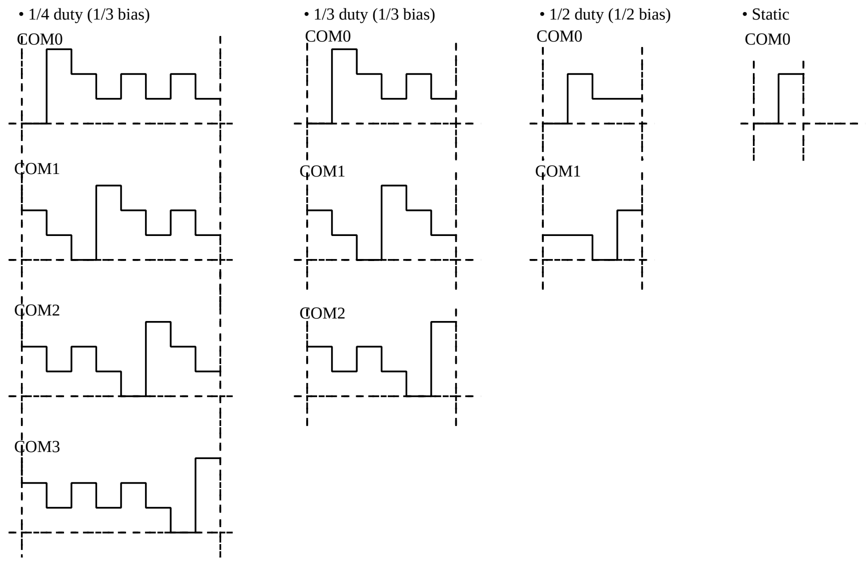
LCD FRAME FREQUENCY	
LFF	Base frequency
0	4096 Hz
1	8192 Hz

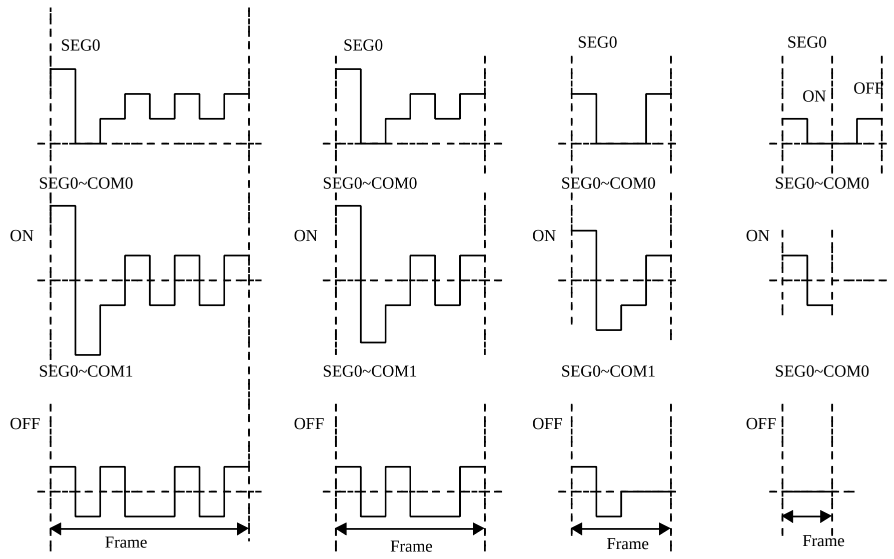
LCD frame frequency : According to the drive method to set the frame frequency and base frequency.

Base frequency	Frame frequency (Hz)			
	1/4 duty	1/3 duty	1/2 duty	Static
4096 Hz	32 x (4/4)=32	32 x (4/3)=43	32 x (4/2)=64	32
8192 Hz	64 x (4/4)=64	64 x (4/3)=85	64 x (4/2)=128	64

LCD driving methods

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





The storing region of data RAM for LCD display data

	RAM address	COM3 bit3	COM2 bit2	COM1 bit1	COM0 bit0
SEG0	32 x LDA + 00h				
SEG1	32 x LDA + 01h				
:	:				
SEG30	32 x LDA + 1Eh				
SEG31	32 x LDA + 1Fh				

The relation between LCD display data and driving method

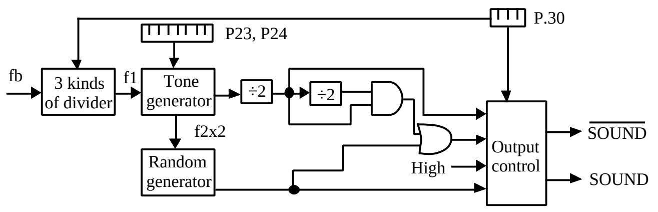
Driving method	bit3	bit2	bit1	bit0
1/4 duty	COM3	COM2	COM1	COM0
1/3 duty	-	COM2	COM1	COM0
1/2 duty	-	-	COM1	COM0
Static	-	-	-	COM0

PROGRAM EXAMPLE:

```
LDIA    #0011B
OUTA    P26     ; Select LCD display area & LCD frame frequency
LDIA    #0000B
OUTA    P27     ; Set LCD duty
LDIA    #1100B
OUTA    P27     ; Enable LCD
LDIA    #1010B
STA     24H
```

SOUND EFFECT

EM73460 has a built-in sound generator. It includes the tone generator and random generator. The tone generator is a binary down counter and the random generator is a 9-bit linear feedback shift register. When the CPU is reseted or sleeping, the sound generator is disabled and the output (P4.0/$\overline{\text{SOUND}}$) is high .



Sound effect command register

The basic frequency is selected by mask option. There are 3 kinds of basic tone for sound generator which can be selected by P30. The output of sound effect is tone and random combination.

Port30	3	2	1	0
	BFREQ		SMODE	

Initial value : 0000

XIN	Basic frequency
480K	fb=XIN/2
1M	fb=XIN/2^2
2M	fb=XIN/2^3
4M	fb=XIN/2^4

BFREQ	Basic tone select
0 0	fb
0 1	fb/2
1 0	fb/2^2
1 1	don't care

SMODE	Sound generator mode
0 0	Disable
0 1	Tone output
1 0	Random output
1 1	Tone+random output

Tone frequency register

The 8-bit tone frequency register is P24 and P23. The tone frequency will be changed when user output the different data to P23. Thus, the data must be output to P24 before P23 when user want to change the 8-bit tone frequency (TF).

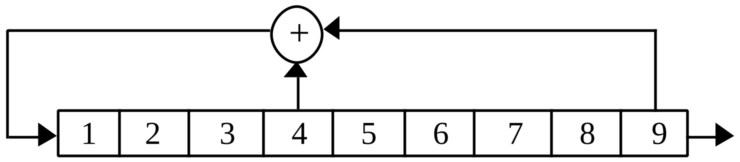
Port24 3 2 1 0	Port23 3 2 1 0
Higher nibble register	Lower nibble register

Initial value : 1111 1111

** f1=fb/2^X, f2=f1/(TF+1)/2, TF=1~255, TF≠0
** Example : XIN=480KHz, BFREQ=10, TF=00110001B.
$\Rightarrow$ fb=240KHz, f1=60K Hz, f2=60K Hz/50/2=600 Hz

Random generator

f(x)=x^9+x^4+1



PROGRAM EXAMPLE:

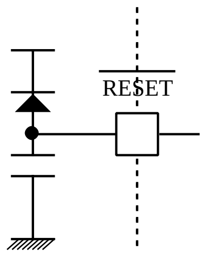
```
LDIA     #1001B  ; basic frequency : 60 KHz tone output
OUTA     P30
LDIA     #0011B  ; 600 Hz tone output
OUTA     P24
LDIA     #0001B
OUTA     P23
```

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 (f1) state	Initial value
Program counter	0000h
Status flag	01h
Interrupt enable flip-flop (EI)	00h
MASK0 ,1, 2, 3	00h
Interrupt latch (IL)	00h
P16, 25, 27, 28, 29, 30	00h
P4, 8, 23, 24	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}.



EM73460 I/O PORT DESCRIPTION :

Port	Input function		Output function		Note
0	E	Input port , wakeup function			
1		--		--	
2		--		--	
3		--		--	
4	E	Input port	E	Output port, P4.0/$\overline{\text{SOUND}}$	
5		--		--	
6		--		--	
7		--		--	
8	E	Input port, wakeup function,	E	Output port	
9		--		--	
10		--		--	
11		--		--	
12		--		--	
13		--		--	
14		--		--	
15		--		--	
16			I	Sleep/Hold mode control register	
17				--	
18				--	
19				--	
20				--	
21				--	
22				--	
23			I	Sound effect frequency register	low nibble
24			I	Sound effect command register	high nibble
25			I	Timebase control register	
26				--	
27			I	LCD control register	
28			I	Timer/counter A control register	
29			I	Timer/counter B control register	
30			I	Sound effect command register	
31				--	

ABSOLUTE MAXIMUM RATINGS

Items	Sym.	Ratings	Conditions
Supply Voltage	V_{DD}	-0.5V to 6V	
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}	300mW	T_{OPR}=50°C
Operating Temperature	T_{OPR}	0°C to 50°C	
Storage Temperature	T_{STG}	-55°C to 125°C	

RECOMMANDED OPERATING CONDITIONS

Items	Sym.	Ratings	Condition
Supply Voltage	V_{DD}	2.4V to 5.0V	480KHz<Fc<4MHz
Input Voltage	V_{IH}	0.90xV_{DD} to V_{DD}	
	V_{IL}	0V to 0.10xV_{DD}	
Operating Frequency	F_{C}	480K to 4MHz	

DC ELECTRICAL CHARACTERISTICS (V_{DD}=3±0.3V, V_{SS}=0V, T_{OPR}=25°C)

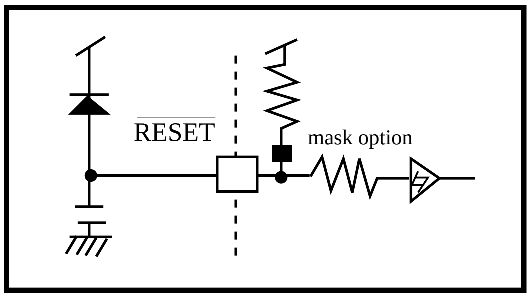
Parameters	Sym.	Min.	Typ.	Max.	Unit	Conditions
Supply current	I_{DD}	-	0.7	1.1	mA	V_{DD}=3.3V,no load,Fc=4MHz (RC osc : R=6.2KΩ, C=20pF)
		-	0.1	1	μA	V_{DD}=3.3V, sleep mode
Hysteresis voltage	V_{HYS+}	0.50V_{DD}	-	0.75V_{DD}	V	$\overline{\text{RESET}}$, P0, P8
	V_{HYS-}	0.20V_{DD}	-	0.40V_{DD}	V	
Input current	I_{IH}	-	-	±1	μA	P0, $\overline{\text{RESET}}$, V_{DD}=3.3V,V_{IH}=3.3/0V
		-	-	±1	μA	Open-drain, V_{DD}=3.3V,V_{IH}=3.3/0V
	I_{IL}	-	-	-500	μA	Push-pull, V_{DD}=3.3V ,V_{IL}=0.4V,except P4
Output voltage	V_{OH}	2.4	-	-	V	Push-pull, V_{DD}=2.7V,P4(high current PMOS), SOUND,I_{OH}=-0.9mA
		2.0	-	-	V	Push-pull, V_{DD}=2.7V,P4 (low current PMOS), P8,I_{OH}=-40μA
	V_{OL}	-	-	0.3	V	V_{DD}=2.7V,I_{OL}=0.9mA,P4,SOUND
Leakage current	I_{LO}	-	-	1	μA	Open-drain, V_{DD}=3.3V, V_{O}=3.3V
Input resistor	R_{IN}	100	200	300	KΩ	P0
		300	600	900	KΩ	$\overline{\text{RESET}}$
Frequency stability		-	15	-	%	Fc=4MHz,RC osc,[F(3V)-F(2.7V)]/F(3V)
Frequency variation		-	20	-	%	Fc=4MHz, V_{DD}=3V,RC osc, [F(typical)-F(worse case)]/F(typical)

(V_{DD}=4.5±0.5V, V_{SS}=0V, T_{OPR}=25°C)

Parameters	Sym.	Min.	Typ.	Max.	Unit	Conditions
Supply current	I_{DD}	-	4.5	5.5	mA	V_{DD}=5V,no load,Fc=4.19MHz(crystal osc)
		-	1.5	2	mA	V_{DD}=5V,no load,Fc=4MHz, (RC osc : R=7.5KΩ, C=20pF)
		-	0.1	1	µA	V_{DD}=5V, sleep mode
Hysteresis voltage	V_{HYS+}	0.50V_{DD}	-	0.75V_{DD}	V	$\overline{\text{RESET}}$, P0, P8
	V_{HYS-}	0.20V_{DD}	-	0.40V_{DD}	V	
Input current	I_{IH}	-	-	±1	µA	P0, $\overline{\text{RESET}}$, V_{DD}=5V,V_{IH}=5/0V
		-	-	±1	µA	Open-drain, V_{DD}=5V,V_{IH}=5/0V
	I_{IL}	-	-	-1	mA	Push-pull, V_{DD}=5V ,V_{IL}=0.4V,except P4
Output voltage	V_{OH}	3.0	-	-	V	Push-pull, P4(high current PMOS), SOUND,I_{OH}=-4mA
		2.4	-	-	V	Push-pull, V_{DD}=4V,P4(low current PMOS), P8,I_{OH}=-200µA
	V_{OL}	-	-	1.0	V	V_{DD}=4V,I_{OL}=4mA,P4,SOUND
Leakage current	I_{LO}	-	-	1	µA	Open-drain, V_{DD}=5V, V_O=5V
Input resistor	R_{IN}	30	90	150	KΩ	P0
		100	300	450	KΩ	$\overline{\text{RESET}}$
Frequency stability		-	10	-	%	Fc=4MHz,RC osc,[F(4.5V)-F(3.6V)]/F(4.5V)
Frequency variation		-	20	-	%	Fc=4MHz, V_{DD}=4.5V,RC osc, [F(typical)-F(worse case)]/F(typical)

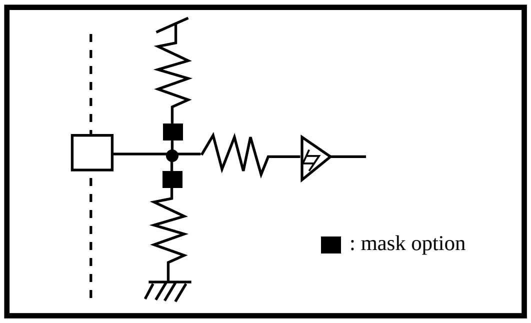
$\overline{\text{RESET}}$ PIN TYPE

TYPE RESET-A

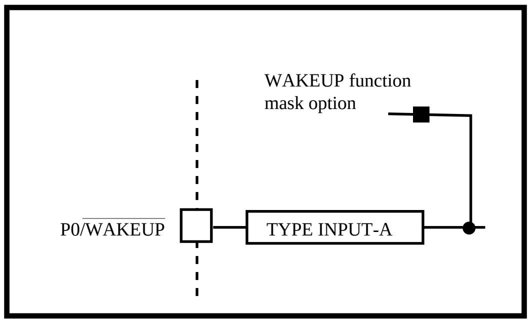


INPUT PIN TYPE

TYPE INPUT-A

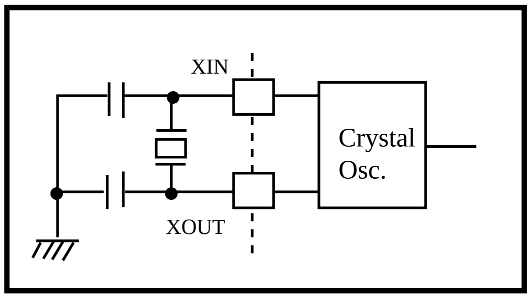


TYPE INPUT-B

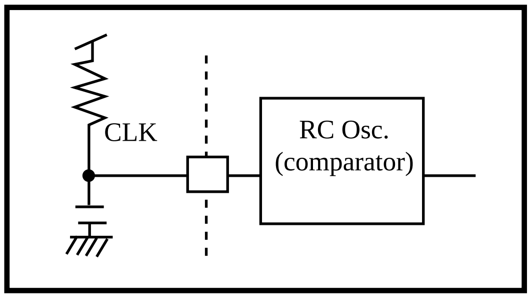


OSCILLATION PIN TYPE

TYPE OSC-A

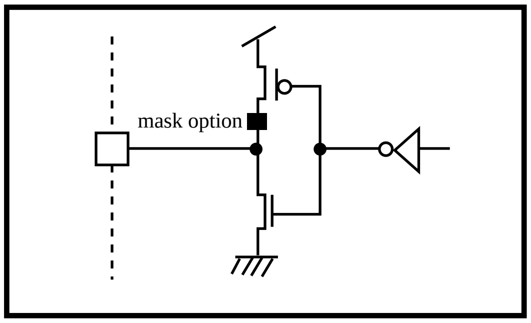


TYPE OSC-C

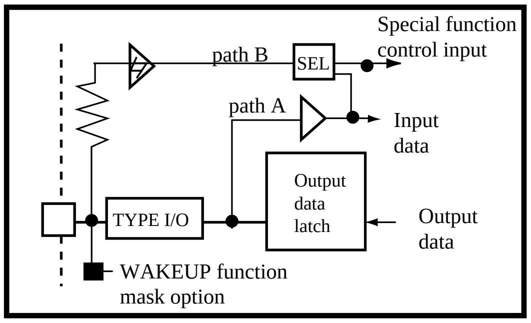


I/O PIN TYPE

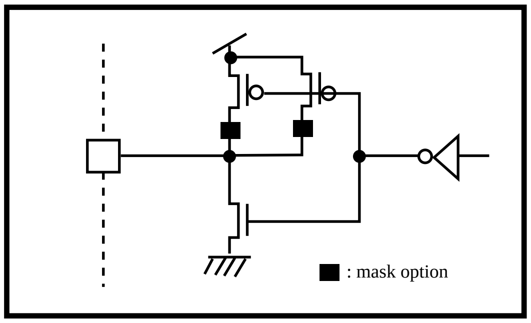
TYPE I/O



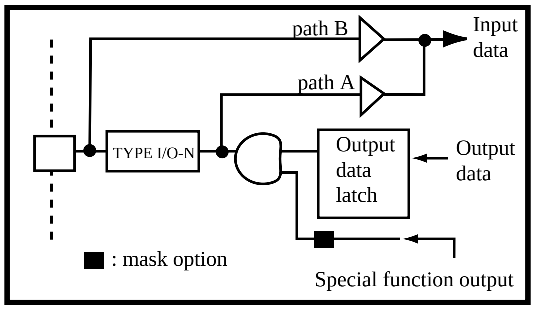
TYPE I/O-L



TYPE I/O-N

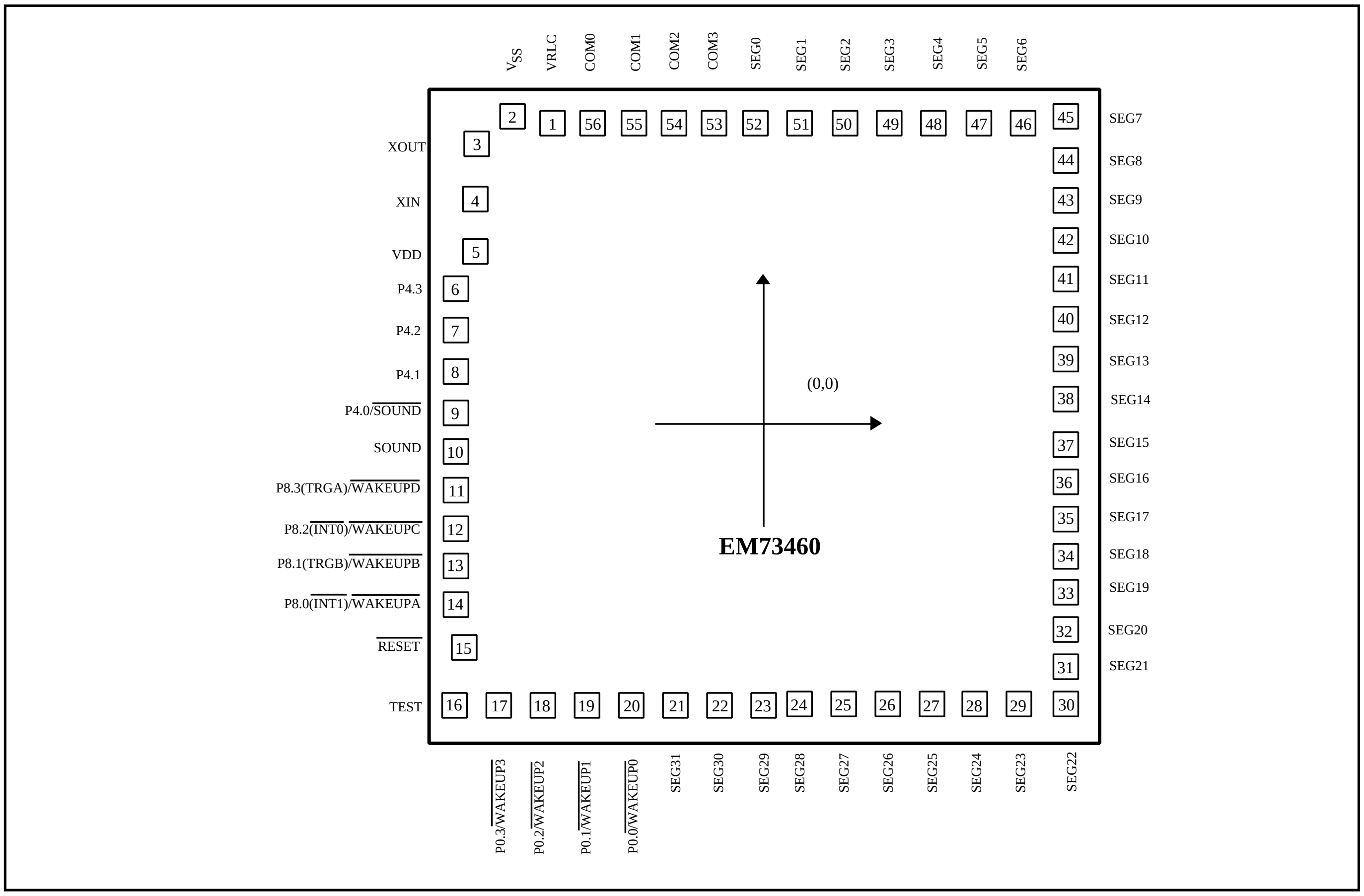


TYPE I/O-O



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.

PAD DIAGRAM



Chip Size : 2560 x 2700 µm

Pad No.	Symbol	X	Y
1	VRLC	-759.0	1142.9
2	V_{SS}	-909.5	1156.0
3	XOUT	-1080.3	1043.4
4	XIN	-1080.3	831.0
5	V_{DD}	-1074.2	610.4
6	P4.3	-1097.9	456.2
7	P4.2	-1097.9	309.2
8	P4.1	-1097.9	146.3
9	P4.0	-1097.9	-0.7
10	SOUND	-1097.9	-163.7
11	P8.3(TRGA)/$\overline{\text{WAKEUPD}}$	-1097.9	-310.7
12	P8.2($\overline{\text{INT0}}$)/$\overline{\text{WAKEUPC}}$	-1097.9	-473.7
13	P8.1(TRGB)/$\overline{\text{WAKEUPB}}$	-1097.9	-620.7
14	P8.0($\overline{\text{INT1}}$)/$\overline{\text{WAKEUPA}}$	-1097.9	-783.6
15	$\overline{\text{RESET}}$	-1072.4	-936.6
16	TEST	-1098.8	-1142.4
17	P0.3/$\overline{\text{WAKEUP3}}$	-932.2	-1142.4

Pad No.	Symbol	X	Y
18	P0.2/$\overline{\text{WAKEUP2}}$	-783.1	-1142.4
19	P0.1/$\overline{\text{WAKEUP1}}$	-625.4	-1142.4
20	P0.0/$\overline{\text{WAKEUP0}}$	-476.3	-1142.4
21	SEG31	-310.8	-1142.4
22	SEG30	-163.8	-1142.4
23	SEG29	-6.2	-1142.4
24	SEG28	140.8	-1142.4
25	SEG27	298.4	-1142.4
26	SEG26	445.4	-1142.4
27	SEG25	603.1	-1142.4
28	SEG24	750.1	-1142.4
29	SEG23	907.7	-1142.4
30	SEG22	1068.9	-1140.5
31	SEG21	1068.9	-982.9
32	SEG20	1068.9	-835.9
33	SEG19	1068.9	-678.3
34	SEG18	1068.9	-531.3
35	SEG17	1068.9	-373.6
36	SEG16	1068.9	-226.6
37	SEG15	1068.9	-69.0
38	SEG14	1068.9	78.0
39	SEG13	1068.9	235.7
40	SEG12	1068.9	382.7
41	SEG11	1068.9	540.3
42	SEG10	1068.9	687.3
43	SEG9	1068.9	844.9
44	SEG8	1068.9	991.9
45	SEG7	1068.9	1157.3
46	SEG6	921.9	1142.9
47	SEG5	764.2	1142.9
48	SEG4	617.2	1142.9
49	SEG3	459.6	1142.9
50	SEG2	312.6	1142.9
51	SEG1	154.9	1142.9
52	SEG0	7.9	1142.9
53	COM3	-149.7	1142.9
54	COM2	-296.7	1142.9
55	COM1	-454.3	1142.9
56	COM0	-601.3	1142.9

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[HLl - 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	0101 0110 1111 1101	Acc←$[DP]_M$	2	2	-	Z	1
LDADPH	0101 0110 1111 1110	Acc←$[DP]_H$	2	2	-	Z	1
LDASP	0101 0110 1111 1111	Acc←SP	2	2	-	Z	1
LDATAL	0110 1010 1111 0100	Acc←$[TA]_L$	2	2	-	Z	1
LDATAM	0101 0110 1111 0101	Acc←$[TA]_M$	2	2	-	Z	1
LDATAH	0101 0110 1111 0110	Acc←$[TA]_H$	2	2	-	Z	1
LDATBL	0110 1010 1111 1000	Acc←$[TB]_L$	2	2	-	Z	1
LDATBM	0101 0110 1111 1001	Acc←$[TB]_M$	2	2	-	Z	1
LDATBH	0101 0110 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		