GENERAL DESCRIPTION

EM73880 is an advanced single chip CMOS 4-bit micro-controller. It contains 8K-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. EM73880 also contains 6 interrupt sources, 1 input port, 2 bidirection ports, LCD display (32x4), and one high speed timer/counter,sound generator, and speech synthesizer.

EM73880 has plentiful operating modes (SLOW, IDLE, STOP) intended to reduce the power consumption.

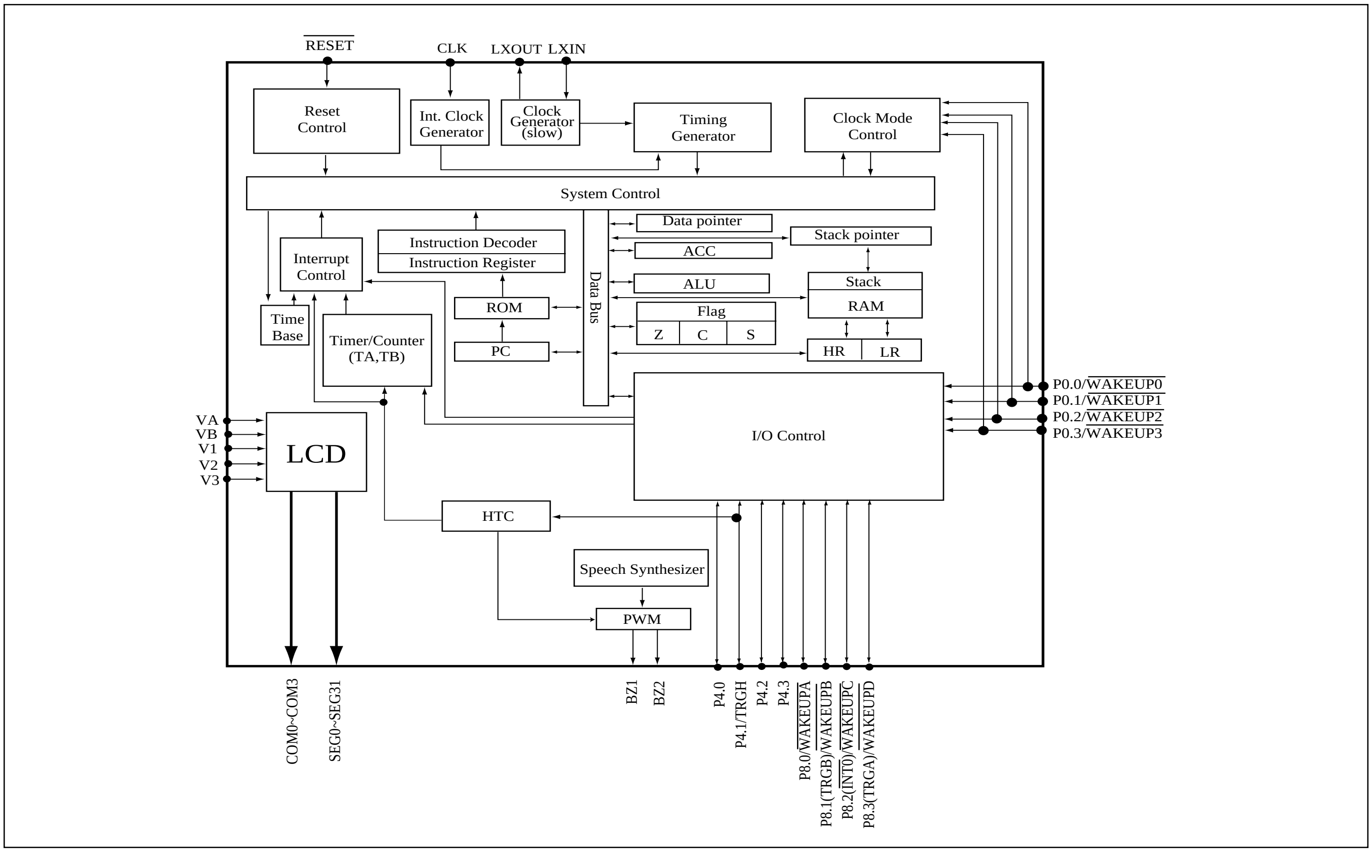
FEATURES

- Operation voltage : 2.4V to 3.6V.
- Clock source : Dual clock system. Low-frequency oscillator (32.768 KHz) could be Crystal or RC oscillator high-frequency oscillator is a built-in for 4.6 MHz.
- Instruction set : 107 powerful instructions.
- Instruction cycle time : 1.7us for 4.6 MHz (high speed clock).
  244 µs for 32768 Hz (low speed clock).
- ROM capacity : 8192 X 8 bits.
- RAM capacity : 244 X 4 bits.
- Input port : 1 port (P0). P0(0..3) and IDLE releasing function are available by mask option.
- Bidirection port : 2 ports (P4, P8). P4.1 is shared with HTC external input. P8(0..3) and IDLE 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.
- High speed timer/counter : One 8-bit high speed timer/counters is programmable for auto load timer, melody output and pulse width measurement.
- Speech synthesizer : 160K Speech ROM.
- Built-in time base counter : 22 stages.
- Subroutine nesting : Up to 13 levels.
- Interrupt : External . . . . . 1 input interrupt sources.
  Internal . . . . . . 2 Timer overflow interrupts, 1 time base interrupt.
  1 high speed timer/counter overflow interrupt.
  1 Speech ending interrupt.
- LCD driver : 32 X 4 dots, 1/4,1/3,1/2, static 4 kinds of duty Type selectable, 1/2 bias, 1/3 bias, 2 kinds bias Type selectable.
- Power saving function : SLOW, IDLE, STOP operation mode.
- Package type : Chip form. . . . .63 pins.

APPLICATIONS

EM73880 is suitable for application in family applicance, consumer products, hand held games and the toy controller.

FUNCTION BLOCK DIAGRAM



PIN DESCRIPTIONS

Symbol	Pin-type	Function
V_{DD}, V_{DD2}		Power supply (+) / speech synthesizer power supply(+)
V_{SS}		Power supply (-)
RESET	RESET-A	System reset input signal, low active mask option : none pull-up
CLK	OSC-G	Capacitor connecting pin for internal high frequency OSC.
LXIN	OSC-B/OSC-H	Crstal/RC connecting pin for low speed clock source
LXOUT	OSC-B	Crstal/RC connecting pin for low speed clock source
P0(0..3)/WAKEUP0..3	INPUT-K	4-bit input port with IDLE releasing function mask option : wakeup enable, negative edge release, pull-up wakeup enable, negative edge release, none wakeup enable, positive edge release, pull-down wakeup enable, positive edge release, none wakeup disable, pull-up wakeup disable, pull-down wakeup disable, none
P4.0	I/O-R	1-bit bidirection I/O port

PIN DESCRIPTIONS

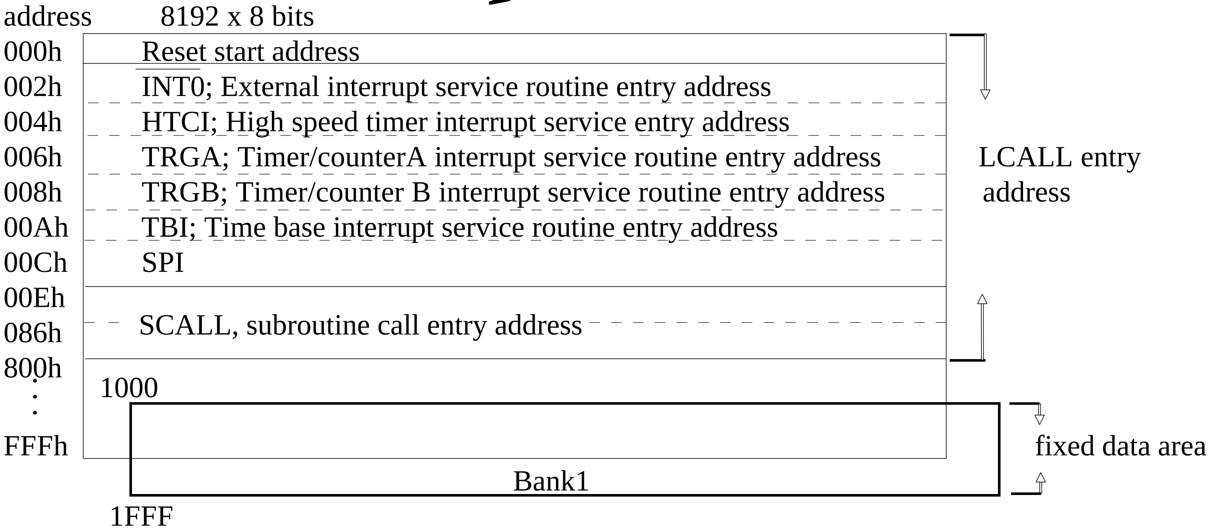
Symbol	Pin-type	Function
P4.1/TRGH	I/O-Q	1-bit bidirection I/O port with HTC external input mask option : NMOS open-drain PMOS open-drain low current push-pull normal current push-pull high current push-pull
P4(2,3)	I/O-Q	2-bit bidirection I/O port with high current source mask option : NMOS open-drain PMOS open-drain low current push-pull normal current push-pull high current push-pull
P8.0/$\overline{\text{WAKEUPA}}$, P8.2/$\overline{\text{INT0}}$/$\overline{\text{WAKEUPC}}$	I/O-S	2-bit bidirection I/O port with external interrupt source input only P8.2 and IDLE releasing function mask option : wakeup enable, low current push-pull wakeup enable, normal current push-pull wakeup disable, open-drain wakeup disable, low current push-pull wakeup disable, normal current push-pull
P8.1(TRGB)/$\overline{\text{WAKEUPB}}$ P8.3(TRGA)/WAKEUPD	I/O-S	2-bit bidirection I/O port with time/counter A,B external input and IDLE releasing function mask option : wakeup enable, low current push-pull wakeup enable, normal current push-pull wakeup disable, open-drain wakeup disable, low current push-pull wakeup disable, normal current push-pull
BZ1, BZ2		Speech output pin
VA,VB, V1, V2, V3		Connect the capacitors for LCD bias voltage
COM0~COM3		LCD common output pins
SEG0~SEG31		LCD segment output pins
TEST		Tie Vss as package type, no connecting as COB type

FUNCTION DESCRIPTIONS

PROGRAM ROM (8K X 8 bits)

8 K x 8 bits program ROM contains user's program and some fixed data.
The basic structure of program ROM can be divided into 6 parts.

1. Address 000h: Reset start address.
2. Address 002h - 00Ch : 6 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 - 1FFFh : Except used as above function, the other region can be used as user's program region.
6. Address 1000h - 1FFFh : Fixed data stortage area.



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.
The program counter is a 13-bit binary counter. The PC can defined 8K ROM.

Table -look-up instruction is depended on the Data Pointer (DP) to indicate to ROM address, then to get the ROM code data. The fixed data only can be put in bank1.

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

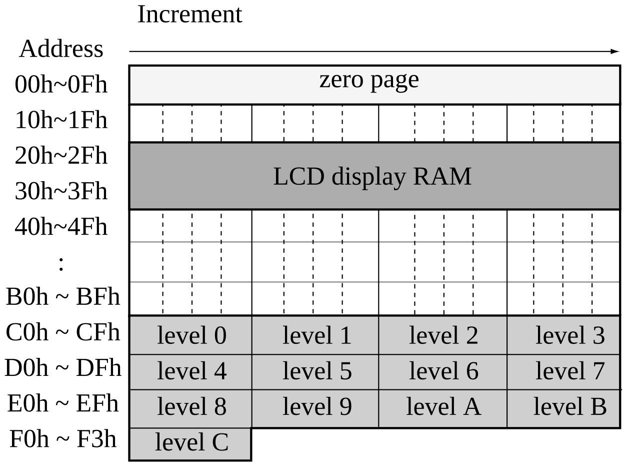
DP is a 12-bit data register which can store the program ROM address to be the pointer for the ROM code data. First, user load ROM address into DP by instruction "STADPL, STADPM, STADPH", then user can get the lower nibble of ROM code data by instruction "LDAX" and higher nibble by instruction "LDAXI"

PROGRAM EXAMPLE: Read out the ROM code of address 1777h by table-look-up instruction.

```
LDIA #07h;
STADPL    ; DP2-0 ← 07h
STADPM    ; DP5-3 ← 07h
STADPH    ; DP8-6 ← 07h, Load DP=777h
:
LDL #00h;
LDH #03h;
LDAX      ; ACC ← 6h
STAMI     ; RAM[30] ← 6h
LDAXI     ; ACC ← 5h
STAM      ; RAM[31] ← 5h
;
ORG 1777H
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.



LCD display RAM:

RAM address from 20h ~ 3Fh are the LCD display RAM area, the RAM data of this region can't be operated by instruction LDHL xx and EXHL.

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 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 (8K ROM)

Program counter (PC) is composed by a 13-bit counter, which indicates the next executed address for the instruction of program ROM.

For a 8K - byte size ROM, PC can indicate address form 0000h - 1FFFh, 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

HTC (High speed counter)

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

SPI (Speech Interrupt)

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
For 3-byte instruction execution: PC + 3

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 three kinds of flag, CF (Carry flag), ZF (Zero flag), SF (Status flag), these 3 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 .

PROGRAM EXAMPLE:
Check following arithematic operation for CF, ZF, SF

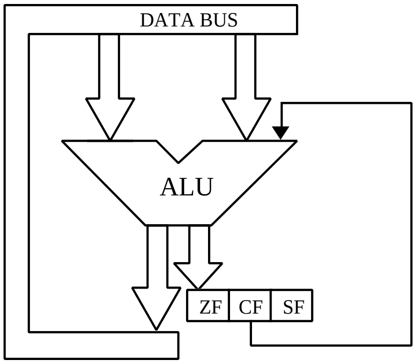
	CF	ZF	SF
LDIA #00h;	-	1	1
LDIA #03h;	-	0	1
ADDA #05h;	-	0	1
ADDA #0Dh;	-	0	0
ADDA #0Eh;	-	0	0

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:

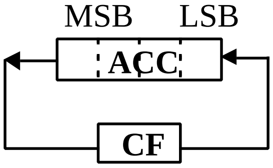
For instruction SUBM #k, SUBA #k, SBCAM, DECM... ALU supports user subtraction function . The subtraction operation can affect CF and ZF, For subtraction operation, if the result is negative, CF will be "0", it means a borrow out, otherwise, if the result is positive, CF will be "1". For ZF, if the result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

EXAMPLE:

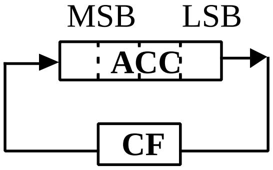
Operation	Carry	Zero
8-4=4	1	0
7-F= -8(1000)	0	0
9-9=0	1	1

(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.



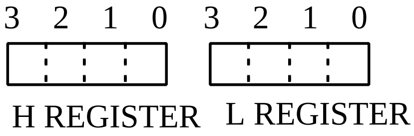
PROGRAM EXAMPLE: To rotate Acc right and shift a "1" into the MSB bit of Acc .

```
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)

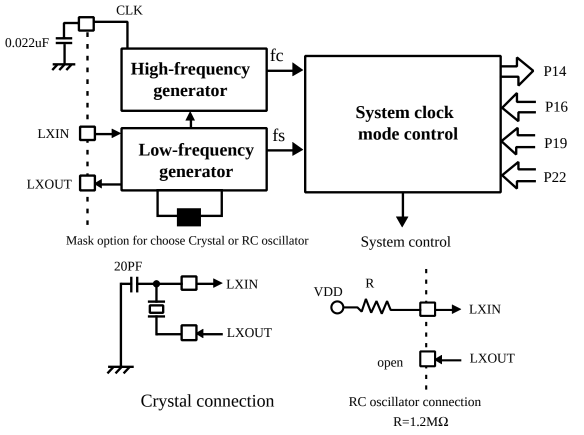
Data pointer is a 12-bit register which stores the address of ROM can indicate the ROM code data specified by user (refer to data ROM).

CLOCK AND TIMING GENERATOR

The clock generator is supported by a dual clock system, the slow clock source comes from crystal (resonator) or RC oscillation is decided by mask option, and it's 32.768 KHz. The high freq OSC is built by a internal clock source that will be 4.6 MHz.

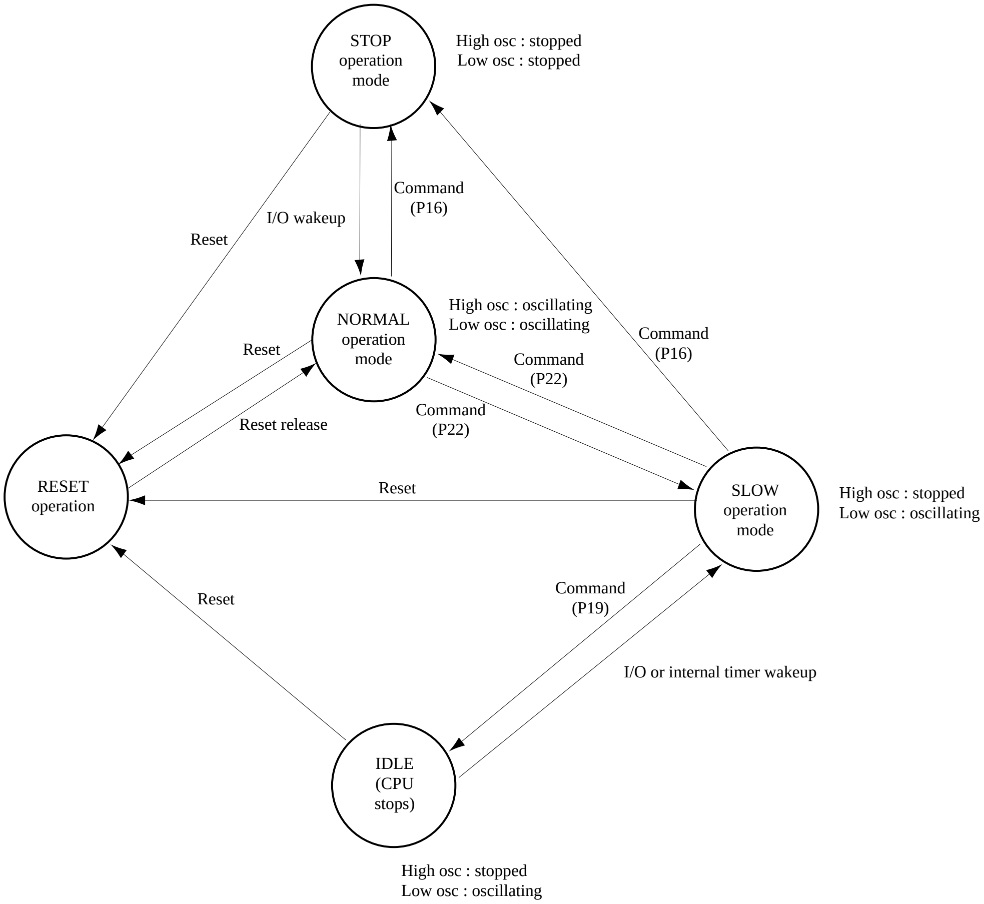
CLOCK GENERATOR STRUCTURE

There are two clock generator for system clock control. P14 is the status register for the CPU status. P16, P19 and P22 are the system clock mode control ports.



SYSTEM CLOCK MODE CONTROL

The system clock mode controller can start or stop the high-frequency and low-frequency clock oscillator and switch between the basic clocks. EM73880 has four operation modes (NORMAL, SLOW,IDLE and STOP operation modes).



Operation Mode	Oscillator	System Clock	Available function	One instruction cycle
NORMAL	High, Low frequency	High frequency clock	LCD, HST, Speech sound	8 / fc
SLOW	Low frequency	Low frequency clock	LCD, HST	8 / fs
IDLE	Low frequency	CPU stops	LCD	-
STOP	None	CPU stops	All disable	-

NORMAL OPERATION MODE

The 4-bit µc is in the NORMAL operation mode when the CPU is reseted. This mode is a dual clock system (high-frequency and low-frequency clocks oscillating). It can be changed to SLOW or STOP operation mode by the command register (P22 or P16).
LCD display and high speed timer/counter with melody output are available for the NORMAL operation mode.

SLOW OPERATION MODE

The SLOW operation mode is a single clock system (low-frequency clock oscillating). It can be changed to the DUAL operation mode with the commoand register (P22), STOP operation mode with P16 and IDLE operation mode with P19.
LCD display and high speed timer/counter with melody output are available for the SLOW operation mode.

P22 Initial value : 0000

3	2	1	0
*	SOM		

SOM	Select operation mode
0 0 0	NORMAL operation mode
1 * *	SLOW operation mode

P14 Initial value : *000

3	2	1	0
*	WKS	LFS	CPUS

LFS	Low-frequency status
0	LXIN source is not stable
1	LXIN source is stable

CPUS	CPU status
0	NORMAL operation mode
1	SLOW operation mode

WKS	Wakeup status
0	Wakeup not by internal timer
1	Wakeup by internal timer

Port14 is the status register for CPU. P14.0 (CPU status) and P14.1 (Low-frequency status) are read-only bits. p14.2 (wakeup status) will be set to '1' when CPU is wake-up by internal timer. P14.2 will be cleared to '0' when user out data to P14.

IDLE OPERATION MODE

The IDLE operation mode suspends all SLOW operations except for the low-frequency clock and LCD driver. It retains the internal status with low power consumption without stopping the clock function and LCD display.

LCD display is available for the IDLE operation mode. Sound generator is disabled in this mode. The IDLE operation mode will be wakeup and return to the SLOW operation mode by the internal timing generator or I/O pins (P0(0..3)/$\overline{\text{WAKEUP 0..3}}$ or P8(0..3)/$\overline{\text{WAKEUPA..D}}$).

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), P8(0..3) pin input
0 1	P0(0..3), P8(0..3) pin input and 1 sec signal
1 0	P0(0..3), P8(0..3) pin input and 0.5 sec signal
1 1	P0(0..3), P8(0..3) pin input and 15.625 ms signal

STOP OPERATION MODE

The STOP operation mode suspends system operation and holds the internal status immediately before the suspension with low power consumption. This mode will be released by reset or I/O pins (P0(0..3)/ WAKEUP 0..3 or P8(0..3)/WAKEUP A..D).

LCD display and high speed timer/counter with melody output are disabled in the mode.

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^{12}/fs
0 1	2^{10}/fs
1 0	2^{14}/fs
1 1	Reserved

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

P25 3 2 1 0

initial value : 0000

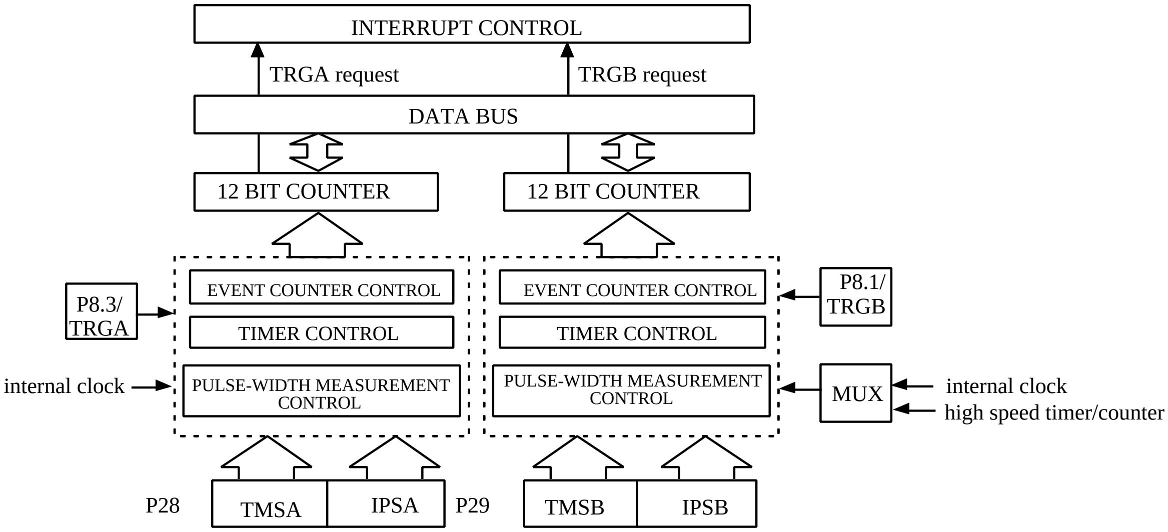
P25	NORMAL operation mode	SLOW operation mode
0 0 x x	Interrupt disable	Interrupt disable
0 1 0 0	Interrupt frequency LXIN / 2^3 Hz	Reserved
0 1 0 1	Interrupt frequency LXIN / 2^4 Hz	Reserved
0 1 1 0	Interrupt frequency LXIN / 2^5 Hz	Reserved
0 1 1 1	Interrupt frequency LXIN / 2^{14} Hz	Interrupt frequency LXIN / 2^{14} Hz
1 1 0 0	Interrupt frequency LXIN / 2^1 Hz	Reserved
1 1 0 1	Interrupt frequency LXIN / 2^6 Hz	Interrupt frequency LXIN / 2^6 Hz
1 1 1 0	Interrupt frequency LXIN / 2^8 Hz	Interrupt frequency LXIN / 2^8 Hz
1 1 1 1	Interrupt frequency LXIN / 2^{10} Hz	Interrupt frequency LXIN / 2^{10} Hz
1 0 x x	Reserved	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	NORMAL mode	SLOW mode
0 0	LXIN/2^3 Hz	Reserved
0 1	LXIN/2^7 Hz	LXIN/2^7 Hz
1 0	LXIN/2^{11} Hz	LXIN/2^{11} Hz
1 1	LXIN/2^{15} Hz	LXIN/2^{15} Hz

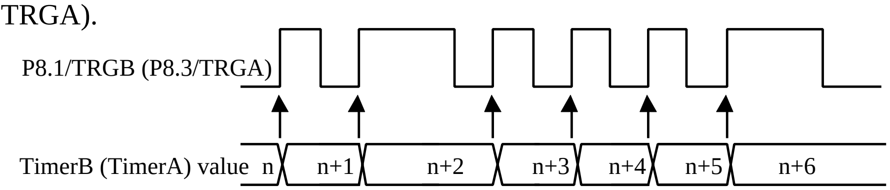
INTERNAL PULSE-RATE SELECTION		
IPSB	NORMAL mode	SLOW mode
0 0	Depend on high speed timer/counter	
0 1	LXIN/2^5 Hz	LXIN/2^5 Hz
1 0	LXIN/2^9 Hz	LXIN/2^9 Hz
1 1	LXIN/2^{13} Hz	LXIN/2^{13} Hz

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).

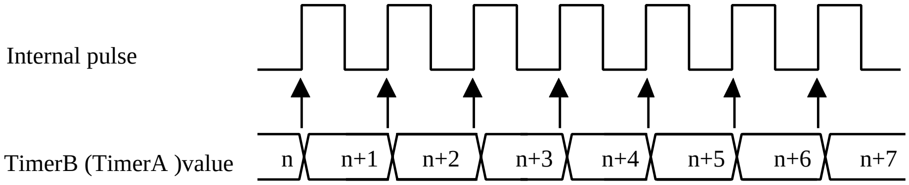


PROGRAM EXAMPLE: Enable timerA with P28

```
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.



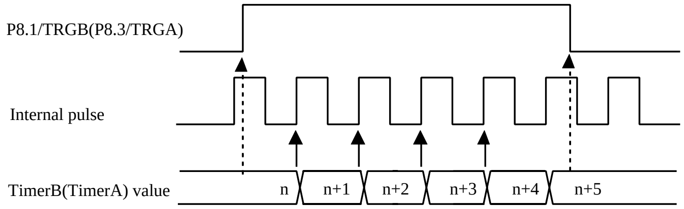
PROGRAM EXAMPLE: To generate TRGA interrupt request after 60 ms with system clock LXlN=32KHz

```
LDIA #0100B;
EXAE;     enable mask 2
EICIL 110111B; interrupt latch ←0, enable EI
LDIA #0AH;
STATAL;
LDIA #00H;
STATAM;
LDIA #0FH;
STATAH;
LDIA #1000B;
OUTA P28; enable timerA with internal pulse rate: LXIN/2³ Hz
```

NOTE: The preset value of timer/counter register is calculated as following procedure.
Internal pulse rate: LXIN/2^3 ; LXIN = 32KHz
The time of timer counter count one = 2^3 /LXIN = 8/32768=0.244ms
The number of internal pulse to get timer overflow = 60 ms/ 0.244ms = 245.901= 0F6H
The preset value of timer/counter register = 1000H - 0F6H = 0F0AH

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.



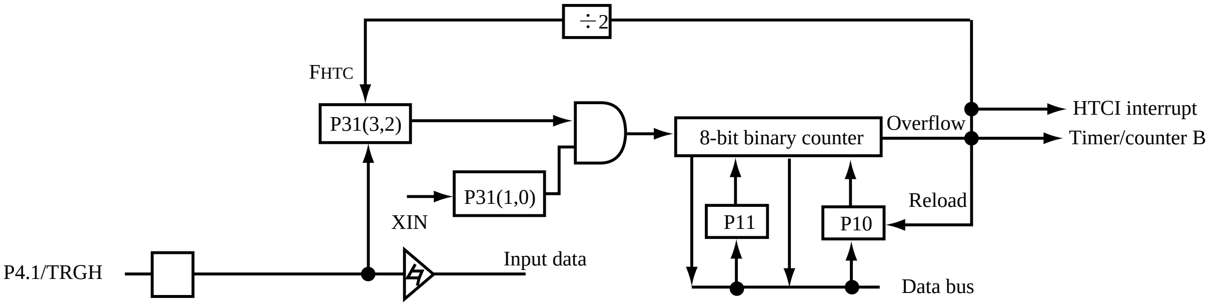
PROGRAM EXAMPLE : Enable timerA by pulse width measurement mode .

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

HIGH SPEED TIMER/COUNTER

8-bit high speed timer/counter (HTC) supports three special functions : auto load timer, melody output and pulse width measurement modes. The HTC is available for the NORMAL and SLOW operation mode.

The HTC can be set initial value and send counter value to counter registers (P11 and P10), P31 is the command port for HTC, user can choose different operation mode and different internal clockrate by setting the port. The timer/counter increase one at the rising edge of internal pulse. The HTC can generate an overflow interrupt (HTCI) when it overflows.



P31 is the command register of the 8-bit high speed timer/counter.

P31 3 2 1 0 Initial value : 0000
HTMS HIPS

HTMS	Mode selection
0 0	Stop
0 1	Event counter mode
1 0	Auto load timer mode
1 1	Pulse width measurement mode

HIPS	Clock rate selection	
	NORMAL mode	SLOW mode
0 0	LXIN/2^0 Hz	LXIN/2^0 Hz
0 1	LXIN/2^2 Hz	LXIN/2^2 Hz
1 0	CLK/2^4 Hz	Reserved
1 1	CLK/2^6 Hz	Reserved

* CLK=4.6MHz

P11 and P10 are the counter registers of the 8-bit high speed timer/counter. P10 is the lower nibble register and P11 is the higher nibble register. (HT is the value of counter registers.)

P11 3 2 1 0 P10 3 2 1 0 Initial value : 0000 0000 (HT)
Higher nibble register Lower nibble register

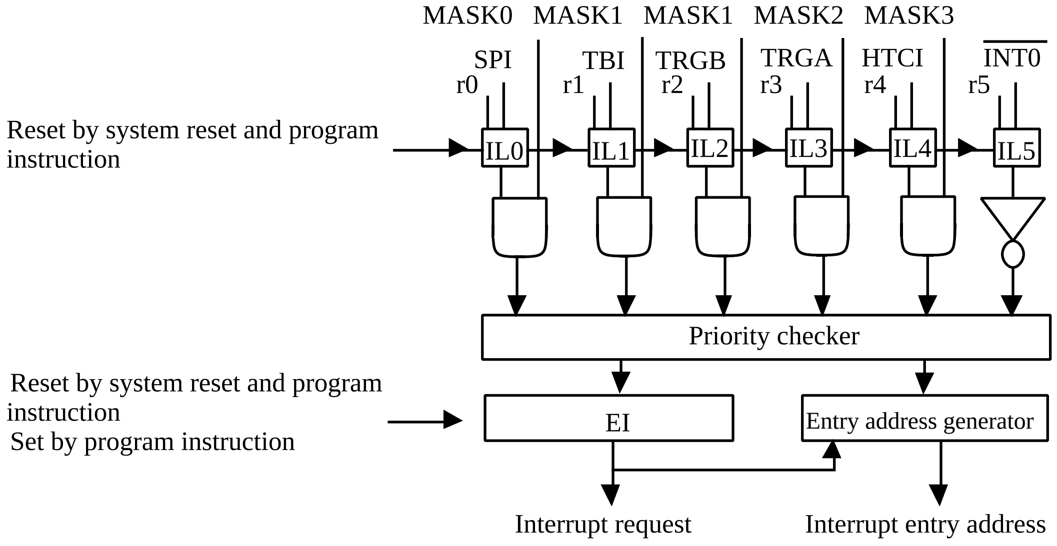
The value of 8-bit binary up counter can be presetted by P10 and P11. The value of registers can loaded into the HTC when the counter starts counting or occurs overflow. If user write value to the registers before the next overflow occurs, the preset value can be changed.
The preset value will be changed when users output the different data to P10. Thus, the data must be output to P11 before P10 when users want to change the preset value.
The count value of HTC can be read from P10 and P11. The value is unstable when user read the value during counting. Thus, user must disable the counter before reading the value.
The P4.1/RGH pin will be the input pin in the event counter and pulse width measurement mode. User must output high to P4. 1/TRGH and then it can be the HTC external input pin. When the HTC is disabled, the P4. 1 pin is a normal I/O pin.

INTERRUPT FUNCTION

There are 6 interrupt sources, 2 external interrupt sources, 4 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($\overline{\text{INT0}}$)	1	IL5	EI=1	002H
Internal	High speed timer overflow interrupt (HTCI)	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
Internal	SPI (Speech interrupt)	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 "INT0, TRGA"

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

SPEECH SYNTHESIZER OPERATES PROCEDURE

1. Send the speech start address to the address latch by writing P6 four times.
2. Choose the sampling rate, enable the speech synthesizer by writing P5.
3. The ROM address counters send the ROM address A6 .. A18 to the speech ROM.
4. ACT is the speech acknowledge signal. When the speech synthesizer has voice output. ACT is high . When ACT is changed from high to low, the speech synthesizer can generate the speech ending interrupt SPI. The ACT signal can be read from P5.3.

SPEECH SYNTHESIZER CONTROL

Speech sample rate control register (P5 write) :

3	2	1	0
SINT	SR		

Initial value : *000

SR	Sample rate selection		Sample rate
000	PWM on	CLK/64/1/3	24K
001		CLK/64/1/4	18K
010		CLK/64/2/3	12K
011		CLK/64/2/4	9K
100		CLK/64/3/3	8K
101		CLK/64/3/4	6K
111	PWM off		

port 5 -- initialization is "*111".
port 6 -- initialization is pointed to the low-nibble of start address latch.

CLK=4.6 MHz

SINT	Select interrupt source
0	Select HTC overflow interrupt
1	Select speech ending interrupt

Speech active flag (P5 read) :

3	2	1	0
ACT	*	*	*

Initial value : 0***

ACT is the speech acknowledge signal. When the speech synthesizer has voice output, ACT is high. When ACT is high → low, the speech synthesizer can generate the speech ending interrupt SPI, or when HTC overflow interrupt SPI.

Speech start address register (P6 write) :

3	2	1	0
Port 6			

Initial value : 1111

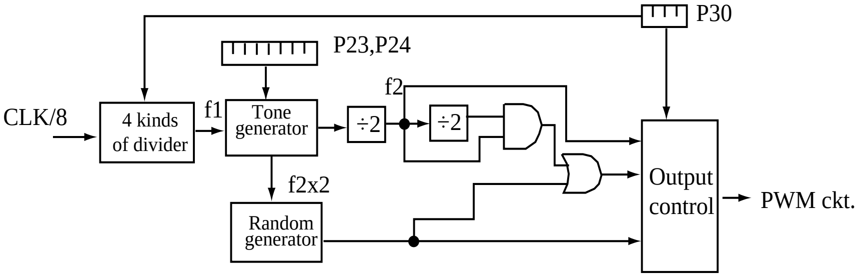
P6L1				P6L2				P6L3				P6L4			
A9	A8	A7	A6	A13	A12	A11	A10	A17	A16	A15	A14	-	-	-	-

Send the speech start address to the speech synthesizer by writing P6 four times. There is a pointer counter to point the address latch (P6L1, P6L2, P6L3, P6L4). It will increase one when write P6. So, the first time writing P6 to P6L1, the second time is P6L2, the third time is P6L3, the fourth time is P6L4 and the fifth time is P6L1 latch again, ... etc. The pointer counter point to P6L1 when CPU is reset or P5 is writen.
In the NORMAL operation mode, the speech synthesizer is available. In the other operation modes, it is disable.

MELODY (SOUND EFFECT) CONTROL

One channel melody/sound effect output, controlled by port 23, 24, 17, and 30.

There is a built-in sound effect. It includes the tone generator and random generator. The tone generator is a binary down counter and the random generator is a 9-bit liner feedback shift register.



Sound effect command register (P30)

There are 4 kinds of basic frequency 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

BFREQ	Basic frequency (f1) select
0 0	CLK/4
0 1	CLK/8
1 0	CLK/16
1 1	CLK/32

(CLK=4.6MHz)

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

Tone frequency register (P23, P24)

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 users want to change the 8-bit tone frequency (TF).

Port24 3 2 1 0
Higher nibble register

Port23 3 2 1 0
Lower nibble register

Initial value : 1111 1111

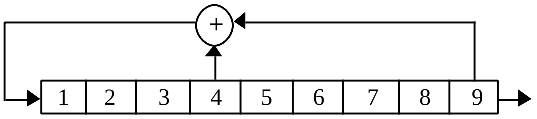
** $f1=CLK/2^X$, $f2=f1/(TF+1)/2$, $TF=1\sim255$, $TF\neq0$

** Example : CLK=4.6 MHz, BFREQ=11, TF=00110001B.

$\Rightarrow$ f1=143.75K Hz, f2=143.75K Hz/50/2=1430 Hz

Random generator

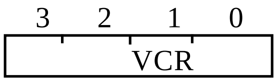
$f(x)=x^9+x^4+1$



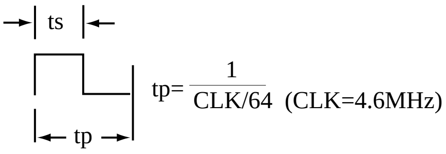
Volume control register (P17)

The are 16 levels of volume for sound generator. P17 is the volume control register.

Port17 Initial value : 1111



VCR				ts/tp
1	1	1	1	15/16
1	1	1	0	14/16
:				:
0	0	0	1	1/16
0	0	0	0	0/16



PROGRAM EXAMPLE:

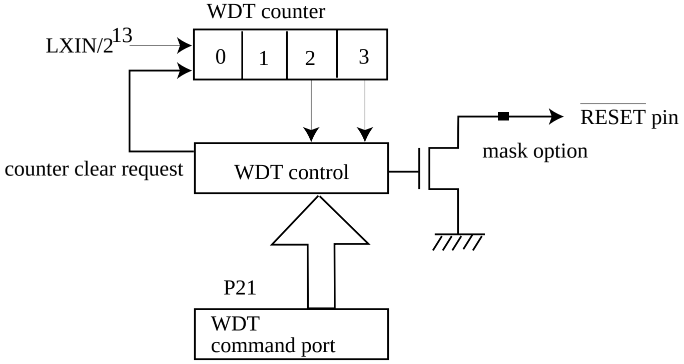
```
LDIA        #1101B          ; basic frequency : CLK/32, tone output
OUTA        P30
LDIA        #0111B          ; volume control
OUTA        P17
LDIA        #0011B          ; 1430 Hz tone output
OUTA        P24
LDIA        #0001B
OUTA        P23
```

WATCH-DOG-TIMER (WDT)

Watch-dog-timer can help user to detect the malfunction (runaway) of CPU and give system a timeup 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 continue 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 $\overline{\text{RESET}}$.

Port 21 Initial value :0000

3	2	1	0
CWC	*	*	WDT

CWC	Clear watchdog timer counter
0	Clear counter then return to 1
1	Nothing

WDT	Set watch-dog-timer detect time
0	3 x 2^{13}/LXIN = 3 x 2^{13}/32K Hz = 0.75 sec
1	7 x 2^{13}/LXIN = 7 x 2^{13}/32K Hz = 1.75 sec

PROGRAM EXAMPLE

To enable WDT with 7 x 2^{13}/LXIN detection time.

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

LCD DRIVER

EM73880 can directly drive the liquid crystal display (LCD) and has 32 segment, 4 common output pins (1/2 bias, 1/3 bias). There are total 32x4 dots can be display. The V1, V2, V3, VA, VB, VDD and VSS pins are the LCD bias generator.

CONTROL OF LCD DRIVER

The LCD driver control command register is P27. When LDC is 0, the LCD is disabled, and user could change Duty in this situation only the COM and SEG pins are VSS. When LDC is 1, the LCD driver enables. When the CPU is reseted or during the STOP operation mode, the LCD driver is disabled.

Port27 Initial value : 0000

3	2	1	0
LDC	DUTY		

LDC	LCD display control
0	LCD display disable and change Duty
1	LCD display enable

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

The LCD display data is stored in the display data area of the data memory (RAM).
The display data area begins with address 20H during reset. The LCD display data area ia as below :

	RAM address	COM3	COM2	COM1	COM0
		bit3	bit2	bit1	bit0
SEG0	20H				
SEG1	21H				
SEG2	22H				
:	:				
:	:				
SEG30	3EH				
SEG31	3FH				

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

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

Duty	Frame frequency (Hz)
1/4 duty	64 x (4/4) = 64
1/3 duty	64 x (4/3) = 85
1/2 duty	64 x (4/2) = 128
Static	64

PROGRAM EXAMPLE :

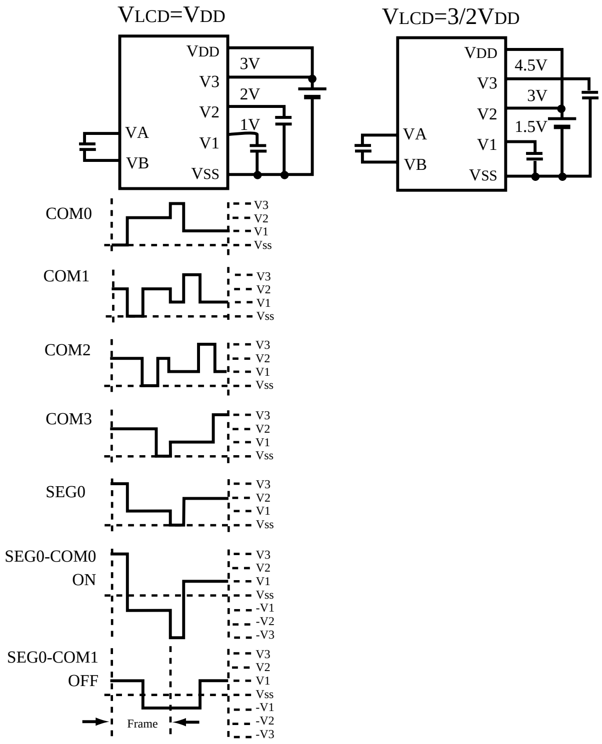
```
  LDIA     #0001B      ; 1/4 duty, 1/2 bias
  OUTA     P27
  LDIA     #1001B      ; enable LCD
  OUTA     P27
```

LCD DRIVING METHODS

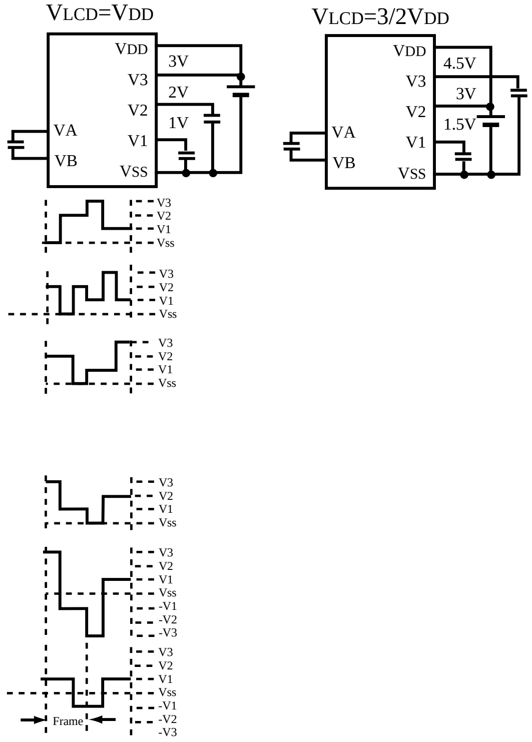
There are six kinds of driving methods can be selected by DUTY (P27.0~P27.2). The drivinf waveforms of LCD driver are as below :

• VDD=3V

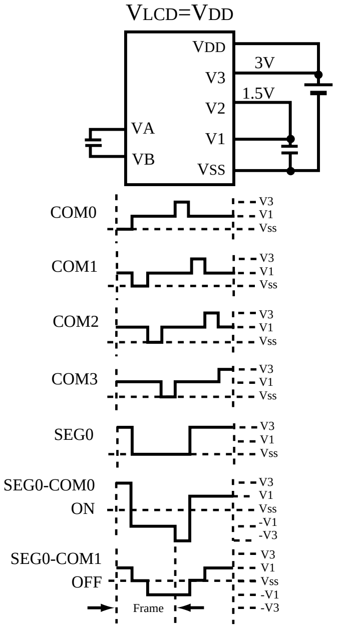
(1) 1/4 duty (1/3 bias)



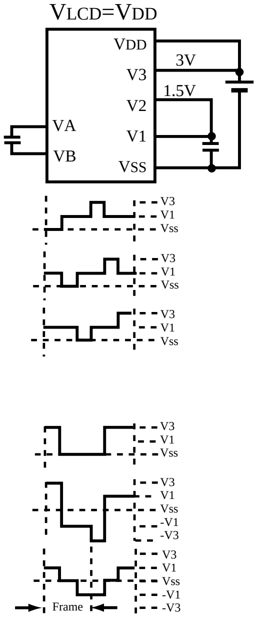
(2) 1/3 duty (1/3 bias)



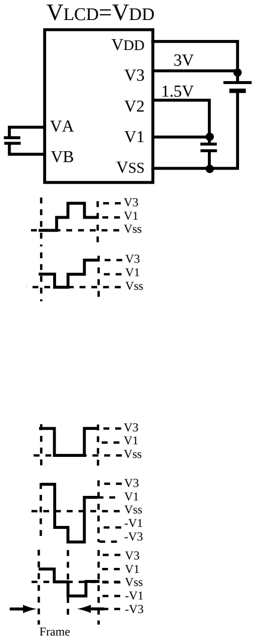
(3) 1/4 duty (1/2 bias)



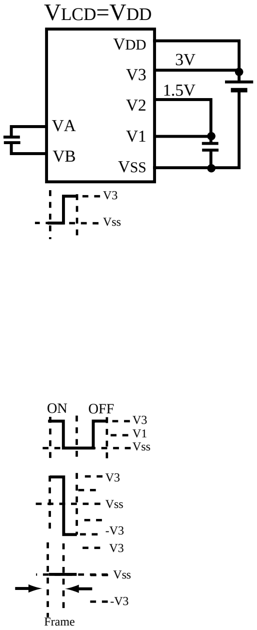
(4) 1/3 duty (1/2 bias)



(5) 1/2 duty (1/2 bias)



(6) static

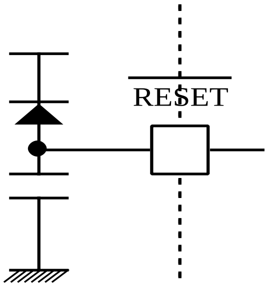


RESETTING FUNCTION

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	0000h
Status flag	01h
Interrupt enable flip-flop (EI)	00h
MASK0 ,1, 2, 3	00h
Interrupt latch (IL)	00h
P4, P5, P10, 11,14, 16, 19, 25, 27, 28, 29, 31	00h
P4, 8, 17, 23, 24	0Fh
Both oscillator	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}.



EM73880 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, P4.1/HST	
5	I	Speech active signal	I	Speech sample rate control register	--
6		--	I	Speech address address	
7		--		--	
8	E	Input port, wakeup function,	E	Output port	
9		--		--	
10		--	I	High speed timer/counter	low nibble
11		--	I	High speed timer/counter	high nibble
12		--		--	
13		--		--	
14	I	CPU status	I	Clear P14.0 to 0	
15		--		--	
16			I	STOP mode control register	
17				Volumn control register	
18				--	
19			I	IDLE mode control register	
20				--	
21				WDT counter	
22			I	Slow mode control register	
23				--	
24				--	
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				--	
31			I	HTC control register	

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}	2.2V to 3.6V	
Input Voltage	V_{IH}	0.90xV_{DD} to V_{DD}	
	V_{IL}	0V to 0.10xV_{DD}	
Operating Frequency	F_C	4.6MHz	CLK
	Fs	32KHz	LXIN, LXOUT

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

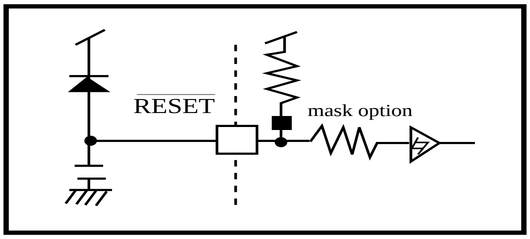
Parameters	Sym.	Min.	Typ.	Max.	Unit	Conditions
Supply current	I_{DD}	-	700	1200	µA	V_{DD}=3.3V,no load,NORMAL mode, Fc=4.6MHz, Fs=32KHz
		-	12	20	µA	V_{DD}=3.3V,no load,SLOW mode,RC osc. (R=1.2MΩ)
		-	7	15	µA	V_{DD}=3.3V,no load,SLOW mode,Crystal osc. (Fs=32KHz)
		-	10	15	µA	V_{DD}=3.3V,IDLE mode,RC osc.(R=1.2MΩ)
		-	5	10	µA	V_{DD}=3.3V,IDLE mode,Crystal osc.
		-	0.1	1	µA	V_{DD}=3.3V, STOP mode
Hysteresis voltage	V_{HYS+}	0.50V_{DD}	-	0.75V_{DD}	V	$\overline{RESET}$, P0, P8
	V_{HYS-}	0.20V_{DD}	-	0.40V_{DD}	V	
Input current	I_{IH}	-	20	30	µA	P0, Pull-down, V_{IH}=V_{DD}
		-30	-20	-	µA	P0, Pull-up, V_{IH}=V_{SS}
		-	-	1	µA	P0, None
		-	-	±1	µA	$\overline{RESET}$, Open-drain, V_{DD}=3.3V,V_{IH}=3.3/0V
	I_{IL}	-	-40	-70	µA	Low current Push-pull, V_{DD}=3.3V,V_{IL}=0.4V
Output voltage	V_{OH}	2.4	2.6	-	V	High current push-pull, SOUND, V_{DD}=2.7V, I_{OH}=-1mA
		2.0	2.4	-	V	Normal current push-pull, V_{DD}=2.7V, I_{OH}=-60µA
	V_{OL}	-	0.1	0.3	V	V_{DD}=2.7V,I_{OL}=1mA
Leakage current	I_{LO}	-	-	1	µA	Open-drain, V_{DD}=3.3V, V_O=3.3V
Input resistor	R_{IN}	50	100	300	KΩ	$\overline{RESET}$

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

Parameters	Sym.	Min.	Typ.	Max.	Unit	Conditions
LCD bias voltage ($^1/_2$ bias)	V1	$^1/_2V_{DD}$-0.1	$^1/_2V_{DD}$	$^1/_2V_{DD}$+0.1	V	I1=5μA
	V2	$^1/_2V_{DD}$-0.1	$^1/_2V_{DD}$	$^1/_2V_{DD}$+0.1	V	I2=5μA
	V3	-	V_{DD}	V_{DD}+0.1	V	I3=5μA
LCD bias voltage ($^1/_3$ bias)	V1	$^1/_3V_{DD}$-0.1	$^1/_3V_{DD}$	-	V	I1=5μA
	V2	$^2/_3V_{DD}$-0.1	$^2/_3V_{DD}$	$^2/_3V_{DD}$+0.1	V	I2=5μA
	V3	-	V_{DD}	V_{DD}+0.1	V	I3=5μA
Output current of BZ1, BZ2	I_{OH}	30	-	-	mA	V_{DD}=3V, V_{BZ}=1.5V
	I_{OL}	30	-	-	mA	

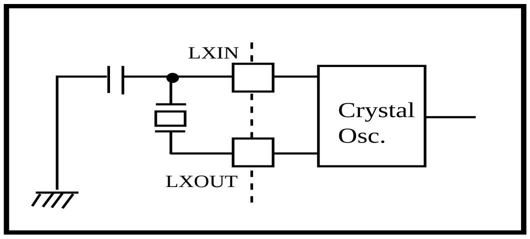
$\overline{\text{RESET}}$ PIN TYPE

TYPE RESET-A

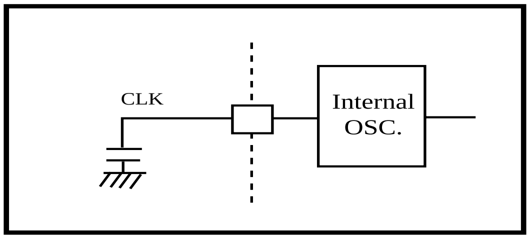


OSCILLATION PIN TYPE

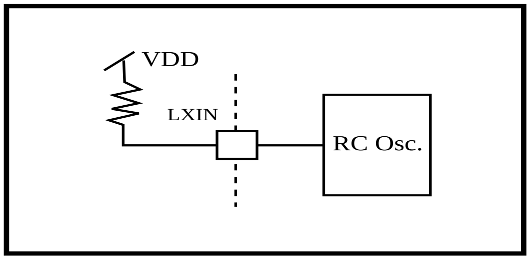
TYPE OSC-B



TYPE OSC-D

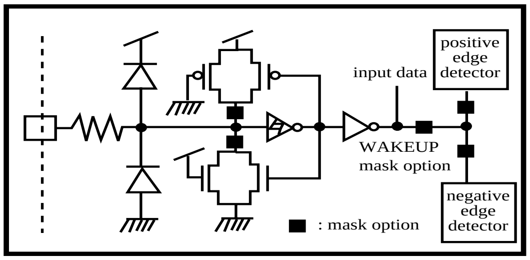


TYPE OSC-H



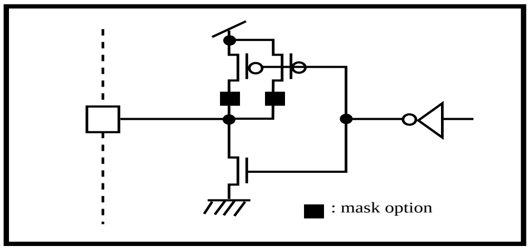
INPUT PIN TYPE

TYPE INPUT-K

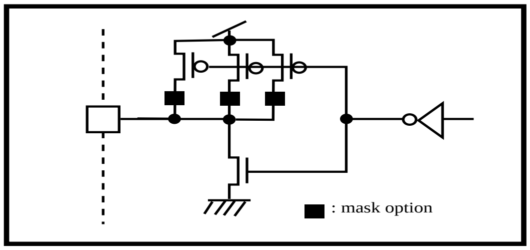


I/O PIN TYPE

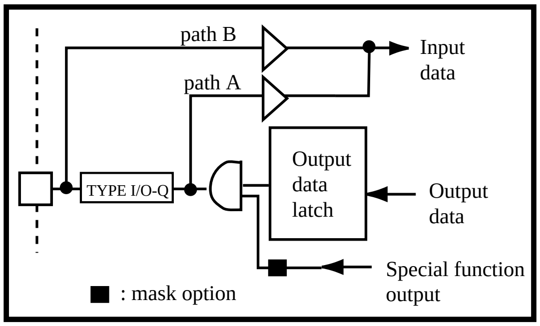
TYPE I/O-N



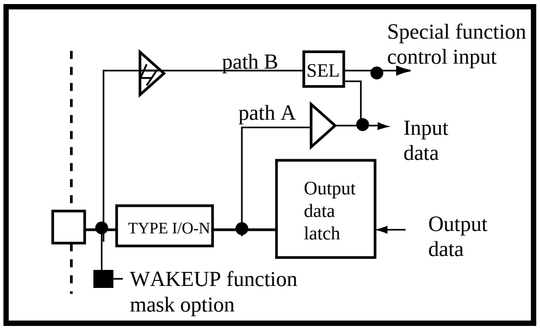
TYPE I/O-Q



TYPE I/O-R

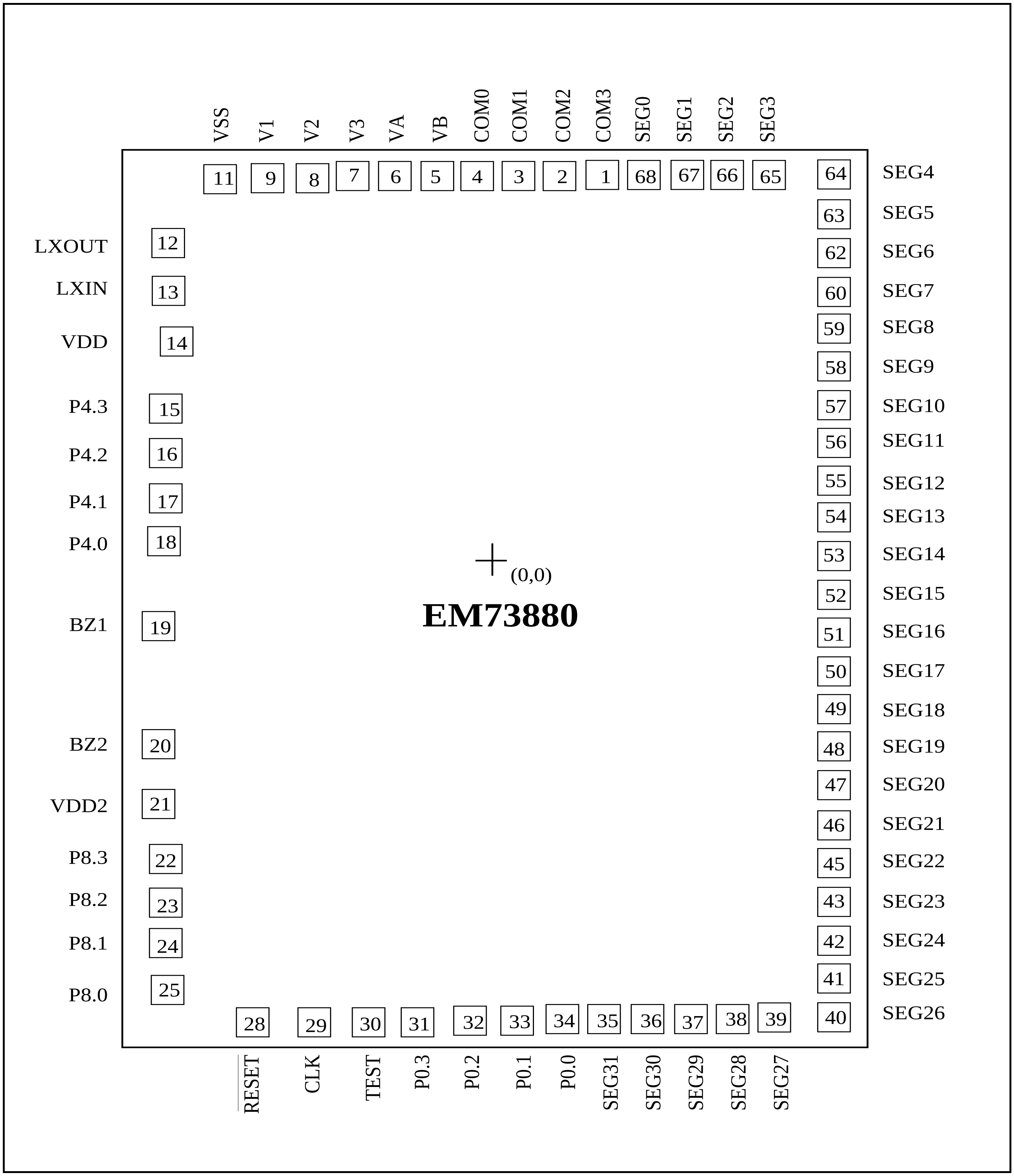


TYPE I/O-S



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



Unit : µm

Chip Size : 2490 x 2960 µm

Note : For PCB layout, IC substrate must be floated or connected to V_{SS}.

Pad No.	Symbol	X	Y
1	COM3	357.5	1322.4
2	COM2	237.0	1322.4
3	COM1	82.4	1322.4
4	COM0	-38.2	1322.4
5	VB	-192.8	1322.4
6	VA	-313.4	1322.4
7	V3	-468.0	1322.4
8	V2	-588.5	1322.4
9	V1	-743.2	1322.4
10			
11	VSS	-934.0	1294.8
12	LXOUT	-1079.9	1019.3
13	LXIN	-1079.9	898.8
14	VDD	-1055.5	745.5
15	P4.3	-1078.5	526.5
16	P4.2	-1078.5	404.4
17	P4.1	-1078.5	282.2
18	P4.0	-1078.5	160.1
19	BZ1	-1091.5	-67.4
20	BZ2	-1091.5	-424.5
21	VDD2	-1094.9	-666.2
22	P8.3	-1074.7	-902.5
23	P8.2	-1074.7	-1024.7
24	P8.1	-1074.7	-1146.8
25	P8.0	-1074.7	-1267.3
26			
27			
28	/RESET	-812.3	-1327.4
29	CLK	-654.9	-1327.4
30	TEST	-487.7	-1327.4
31	P0.3	-322.5	-1327.4
32	P0.2	-154.2	-1327.4
33	P0.1	15.9	-1327.4
34	P0.0	184.3	-1327.4
35	SEG31	346.6	-1327.4
36	SEG30	467.2	-1327.4
37	SEG29	621.8	-1327.4
38	SEG28	742.4	-1327.4
39	SEG27	897.0	-1327.4
40	SEG26	1085.0	-1329.4

Pad No.	Symbol	X	Y
41	SEG25	1085.0	-1028.8
42	SEG24	1085.0	-1088.3
43	SEG23	1085.0	-967.8
44			
45	SEG22	1085.0	-847.2
46	SEG21	1085.0	-726.7
47	SEG20	1085.0	-606.1
48	SEG19	1085.0	-485.6
49	SEG18	1085.0	-365.1
50	SEG17	1085.0	-244.5
51	SEG16	1085.0	-124.0
52	SEG15	1085.0	-3.4
53	SEG14	1085.0	117.1
54	SEG13	1085.0	237.6
55	SEG12	1085.0	358.2
56	SEG11	1085.0	478.7
57	SEG10	1085.0	599.3
58	SEG9	1085.0	719.8
59	SEG8	1085.0	840.3
60	SEG7	1085.0	960.9
61			
62	SEG6	1085.0	1081.4
63	SEG5	1085.0	1202.0
64	SEG4	1085.0	1322.5
65	SEG3	907.9	1322.4
66	SEG2	787.4	1322.4
67	SEG1	632.7	1322.4
68	SEG0	512.2	1322.4

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[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
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		
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		