

BL22P14
LCD Type 8-Bit OTP MCU

USER MANU

V1.0 (2010-4-15)



上海贝岭股份有限公司

Shanghai Belling Co., Ltd.



LCD Type 8-bit OTP MCU BL22P14

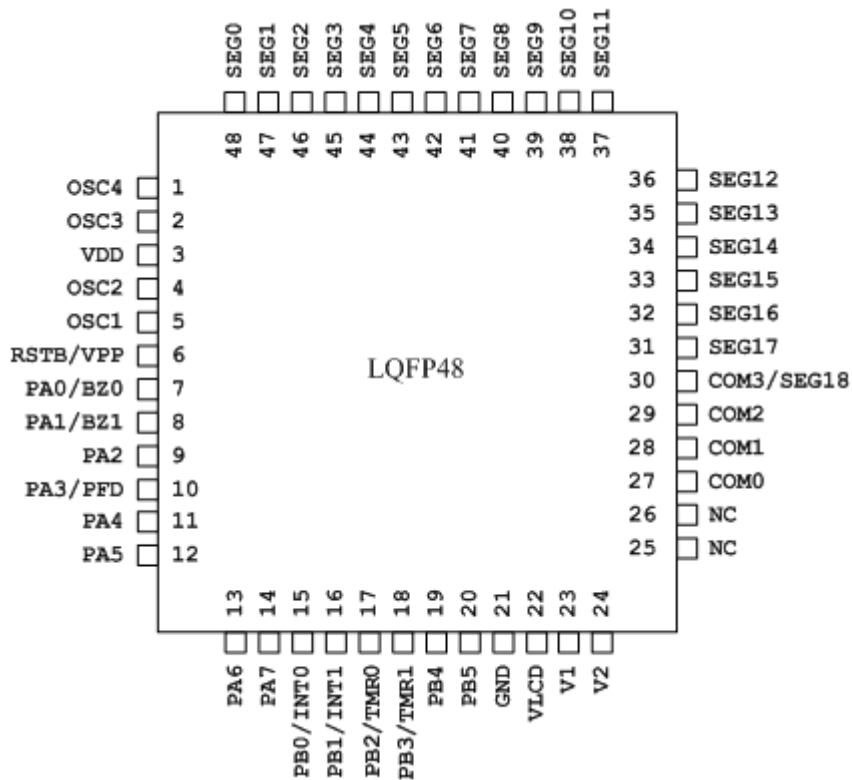
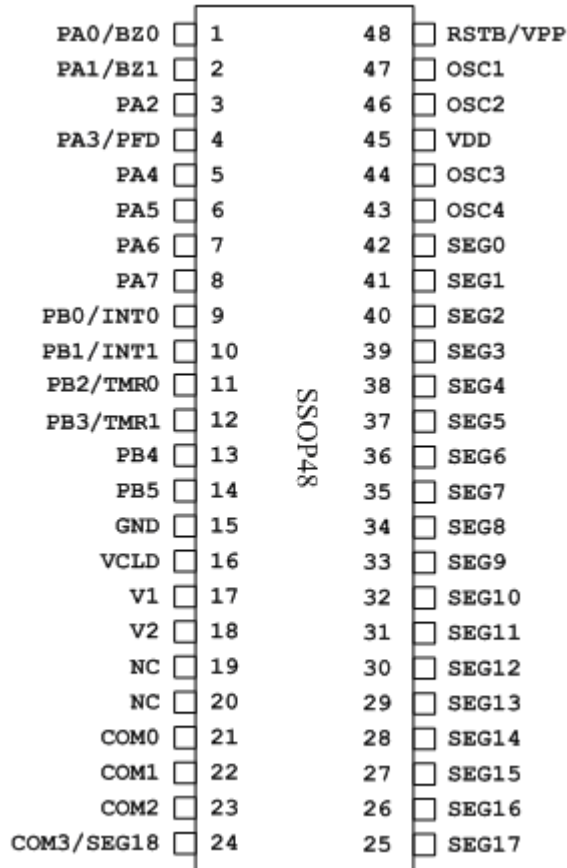
1. General Description

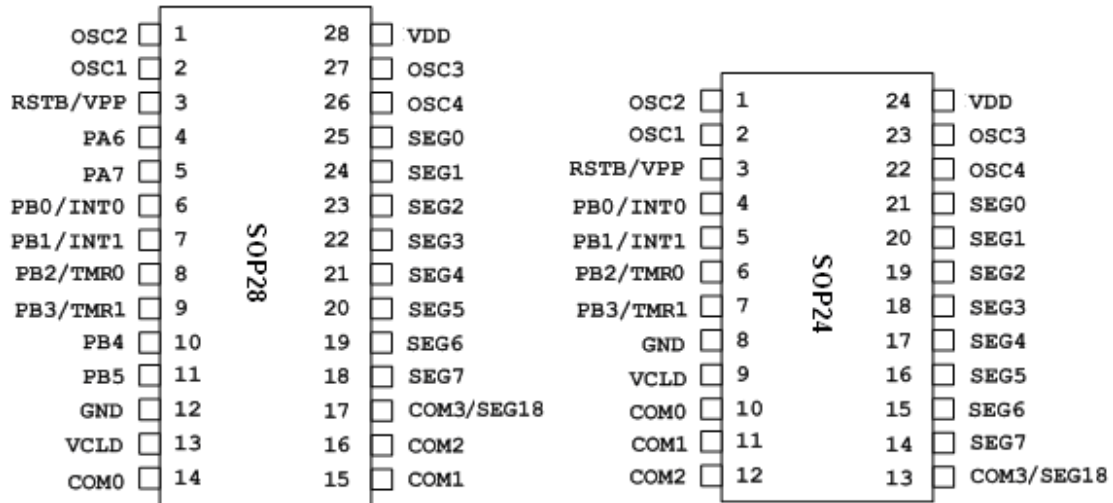
BL22P14 is an 8-bit high performance single chip microcontroller. It provides two group configurable I/O ports, two timers and multiple LCD for household appliances.

2. Features

- ✧ 8-bit CISC core (compatible with Motorola HC05)
- ✧ Low power (1uA@5V in stop mode)
- ✧ Operating Voltage: 2.5-5.5V($\leq 4\text{MHz}$) , 3.5-5.5V($\leq 8\text{MHz}$)
- ✧ Operating temperature: $-40^{\circ}\text{C} \sim 85^{\circ}\text{C}$
- ✧ 4K*8bit ROM
- ✧ 128byte RAM
- ✧ LCD driver with 18*4, 19*3, 19*2 and 19*1
- ✧ 14 bidirectional I/O
- ✧ PA as keyboard interrupt source
- ✧ Two external interrupt
- ✧ An 8-bit Timer
- ✧ A 16-bit Timer
- ✧ Watch dog
- ✧ Buzzer output
- ✧ LVD (typical 3.3V)
- ✧ LVR (typical 3.0V)

3. Pin Assignments





4. Pin Descriptions

SSOP48/LQFP48/SOP28/SOP24

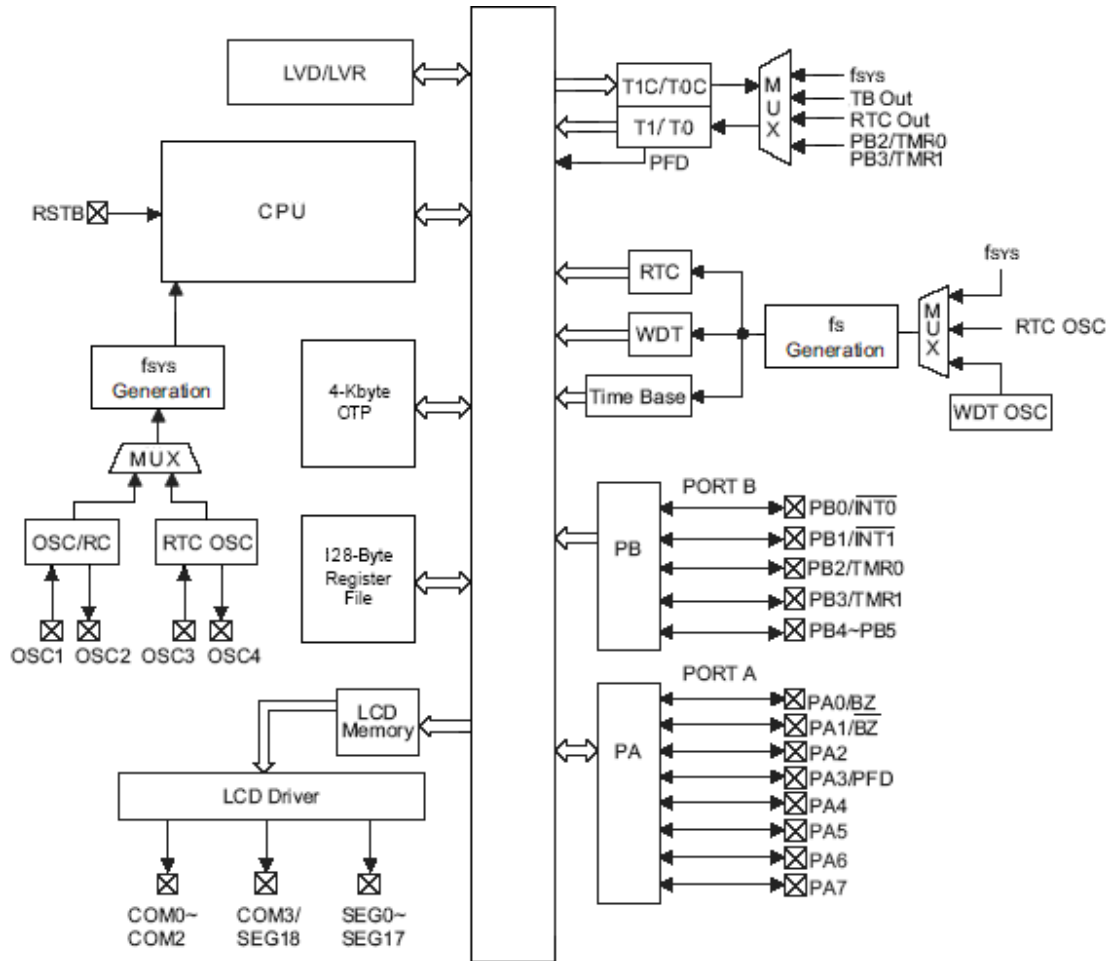
Pin name	In/Out	Share Description	Pin Description
PA0	I/O	BZ0-buzzer output (+)	Bit-programmable I/O port for Schmitt trigger input or push-pull, each bit can be configured as a input with pull-high resistor or keyboard interrupt input
PA1	I/O	BZ1-buzzer output (-)	
PA2	I/O		
PA3	I/O	PFD output	
PA4	I/O		
PA5	I/O		
PA6	I/O		
PA7	I/O		Bit-programmable I/O port for Schmitt trigger input or push-pull, each bit can be configured as a input with pull-high resistor
PB0	I/O	INT0	
PB1	I/O	INT1	
PB2	I/O	TMR0-TIMER0 input	
PB3	I/O	TMR1-TIMER1 input	
PB4	I/O		Ground
PB5	I/O		
VSS			LCD voltage supply
VLCD			
V1			LCD filter capacitor
V2			
COM0	O		COM output of LCD
COM1	O		
COM2	O		
COM3/SEG18	O		COM or SEG of LCD
SEG17~SEG0	O		SEG output of LCD
OSC4	O		32768Hz crystal oscillator
OSC3	I		



VDD			Power supply
OSC2	O		Crystal oscillator or RC clock
OSC1	I		
RSTB	I	VPP	System reset input

5. Function Descriptions

5.1 Block Diagram



5.2 Address Spaces

- \$0000-\$006F: Control registers
- \$0070-\$007F: LCD data
- \$0080-\$00FF: RAM
- \$0100-\$0FFF: Reserved
- \$1000-\$1FFF: OTP ROM

5.3 Control Registers

Register name	Address	R/W	Reset Value
PA	\$00	R/W	0000 0000



DDRA	\$01	R/W	0000 0000
PB	\$02	R/W	--00 0000
DDRBB	\$03	R/W	--00 0000
KBIM	\$04	R/W	0000 1000
PBPR	\$05	R/W	--00 0000
T0D	\$06	R/W	0000 0000
T0C	\$07	R/W	00-0 0000
T1DH	\$08	R/W	0000 0000
T1DL	\$09	R/W	0000 0000
T1C	\$0A	R/W	0000 0000
INTC0	\$0B	R/W	--00 0000
INTC1	\$0C	R/W	0000 0000
BZCR	\$0D	R/W	0000 --00
MCR1	\$0E	R/W	00u- --00
MCR2	\$0F	R/W	0000 0000
RSTFR	\$10	R/W	---- rrrr
LCDD[9:0]	\$79-\$70	W	xxxx xxxx
LCDCON	\$7A	R/W	0000 0000

NOTE:

- : Not used;
- x: Undefined;
- u: Determined by OPBIT[4];
- r: Determined by reset type

5.4 Clock

Device supplies four clock sources: Crystal oscillator, external RC, RTC and WDT. First three clock sources can be as system clock Fsys and all can be as internal clock Fs.

5.4.1 System Clock: Fsys

System clock is main operate clock for device. It is configured by OTP OPTION BIT (FSYS1, FSYS0) to select a clock source.

FSYS1、FSYS0

00 / 11: Fsys=Fosc/2

01: Fsys=Frc/2

10: Fsys=Frtc/2

5.4.2 Internal Clock: Fs

Fs is the clock of Buzzer, RTC interrupt, Time Base interrupt, LCD and Watchdog. It is configured by OTP OPTION BIT (FS1, FS0) to select a clock source. If RTC or WDT is selected, device will still operate in stop mode.

FS1、FS0

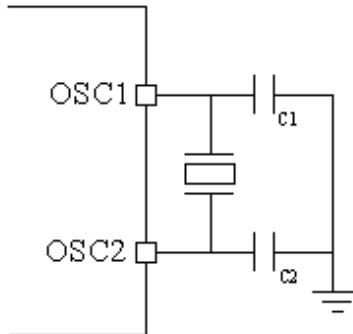
00 / 11: Fs=Fsys



- 01: $F_s = F_{rtc}$
10: $F_s = F_{wdt}$

5.4.3 Crystal Oscillator

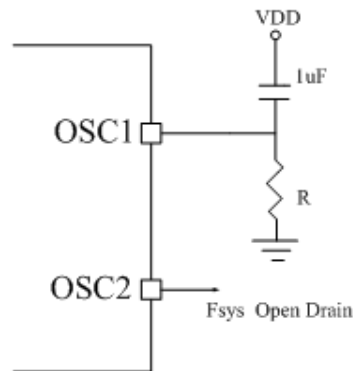
Crystal oscillator provides the clock of 455KHz-8MHz. A crystal should be connected between the pin of OSC1 and OSC2.



Frequency of crystal	Value of C1/C2
8MHz	0/10p
4MHz	0/10p/20p
455KHz	100p/200p

5.4.4 External RC Oscillator

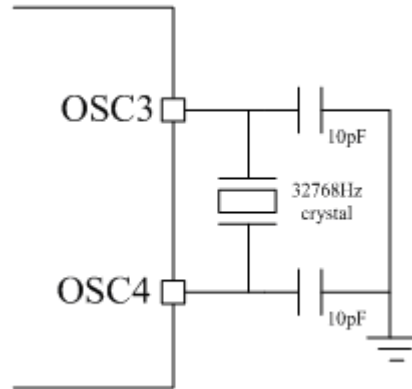
Using external RC oscillator, a resistor between OSC1 and VSS is required. The value of resistor 800K Ω ~40K Ω , the frequency of clock will be 400KHz~8MHz. OSC2 can output clock of F_{sys} if connecting a resistor from OSC2 to VDD. However, the frequency of oscillator may vary with VDD, temperature and chip itself due to process variations. It is therefore, not suitable for timing sensitive operations where accurate frequency is desired.



5.4.5 RTC Oscillator

RTC oscillator provides the clock of 32768Hz. A crystal should be connected between the pin of OSC3 and OSC4. If it is the clock source of F_s , RTC oscillator will still work in stop mode.

The RTC Oscillator circuit can be controlled to oscillate quickly by setting the "QOSC" bit (bit 3 of register of MCR2). It is recommend to turn on the quick oscillating function upon power on and then turn it off after 2 seconds.



5.4.6 WDT oscillator

WDT is a dedicated RC oscillator and output clock of $T=65\mu s@5V$. If it is the source of F_s , RTC oscillator will still work in stop mode. The power is only $3\mu A@5V$.

5.5 Low Power Mode

There are two low power modes: STOP and WAIT.

In stop mode, F_{sys} is turn off and can be waked up by :

- ◇ Keyboard interrupt.
- ◇ External interrupt.
- ◇ Interrupt from module whose clock source is from RTC oscillator or WDT.

In wait mode, only turn off clock of CPU and all interrupt can wake it up.

5.6 RESET

BL22P14 can be reset in four ways:

- 1) by external power-on-reset
- 2) by the external reset input pin(P12) pulled low
- 3) by the digital watchdog peripheral timing out
- 4) by Low Voltage reset (LVR)

There is a dedicated register to write reset type.

RSTFR (\$10): Reset Flag Register

.7-.4 Not used

.3 RSTF3

0: no WDT reset

1: WDT reset

Write "0" to clear the bit, write "1" is null.

.2 RSTF2

0: no LVR reset

1: LVR reset

Write "0" to clear the bit, write "1" is null.

.1 RSTF1

0: no RES reset

1: RES reset



Write "0" to clear the bit, write "1" is null.

.0 RSTF0

0: no power on reset

1: power on reset

Write "0" to clear the bit, write "1" is null.

5.7 I/O PORTS

There are two group I/O ports: PA and PB. PA has 8 I/O ports and PB has 6 I/O ports. All ports have pull up resistors. PA has the function of keyboard interrupt.

The control registers are PA, PB, DDRA, DDRB, KBIM and PBPR.

PA (\$00): Data register of PORT A

.7-.0 PA[7:0]

PA is the data register for Port A.

DDRA(\$01): Data direction register of PORT A

.7-.0 DDRA[7:0] 00000000

DDRA is used to select data direction of PA.

When DDRAi is "0", PAi is input; When DDRAi is "1", PAi is output.

PB (\$02): Data register of PORT B

.5-.0 PB[5:0]

PB is the data register for Port B.

DDR(\$03): Data direction register of PORT B

.5-.0 DDRB[5:0]

DDR is used to select data direction of PB.

When DDRBi is "0", PBi is input; When DDRBi is "1", PBi is output.

KBIM(\$04): Keyboard interrupt mask register

.7-.0 KBE[7:0]

KBIM is configured to enable keyboard interrupt. When KBEi is "1", keyboard interrupt of PAi is turn on, PAi keeps input and its pull up resistor is effective.

Besides, if needing keyboard interrupt active, KBIE (bit 3 of register of INTC0) should be "1". When KBEi is "0", keyboard interrupt of PAi is turn off.

PBPR(\$05): PORTB pull-up register

.5-.0 PBP[5:0]

PBPR is configured to enable pull up resistors of PB. When PBPi is "0", resistor of PBi is ineffective. When PBPi is "1", resistor of PBi is effective. When PB is output, PBPR is no effect.

5.8 LCD

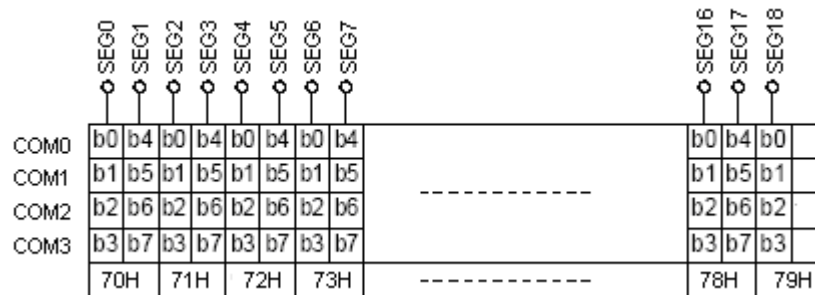
LCD drives 4*18 at most. When the clock of LCD comes from RTC oscillator or WDT oscillator, LCD will still operate in stop mode.

LCD memory

The memory of LCD is located at 70H-79H. Any data written into 70H-79H will affect LCD



display. When written "1", the corresponding bit is turn display on; When written "0", the corresponding bit is turn display off. Before writing data to memory, the memory is inconstant.



LCD register- LCDCON

LCDCON(\$7A): LCD Control Register

.7-.4 PS[3:0] 0000

Dividing frequency of clock source

0000: $1/2^5$

0001: $1/2^6$

0010: $1/2^7$

0011: $1/2^8$

0100: $1/2^9$

0101: $1/2^{10}$

0110: $1/2^{11}$

0111: $1/2^{12}$

1000: $1/2^{13}$

1001: $1/2^{14}$

1010: $1/2^{15}$

.3-.1 DBS [2:0] 000

Duty/Bias select

000: 1/4 Duty 1/3 Bias(COM0-COM3, SEG0-SEG17)

001: 1/3 Duty 1/3 Bias(COM0-COM2, SEG0-SEG18)

010: 1/2 Duty 1/2Bias(COM0-COM1, SEG0-SEG18)

011: 1/3 Duty 1/2Bias(COM0-COM2, SEG0-SEG18)

others: static(COM0-COM2, SEG0-SEG18)

.1 Not used

.0 LCDON 0

0: all COM/SEG output high

1: LCD can display

Clock of LCD

The clock source of LCD is F_s , and dividing frequency is set by bit 4-7 of LCDCON. Usually, the frame frequency is set at 25Hz to 250Hz.

If WDT is the clock source, dividing frequency should be $1/2^5 \sim 1/2^8$.

THE FRAME FREQUENCY WHEN WDT IS CLCOK SOURCE (Hz)



LCD frequency	Static	1/2Duty	1/3Duty	1/4Duty
$F_s/2^5 = 512$ Hz	512	256	171	128
$F_s/2^6 = 256$ Hz	256	128	85	64
$F_s/2^7 = 128$ Hz	128	64	43	32
$F_s/2^8 = 64$ Hz	64	32	21	16

If RTC oscillator is the clock source, dividing frequency should be $1/2^6 \sim 1/2^9$.

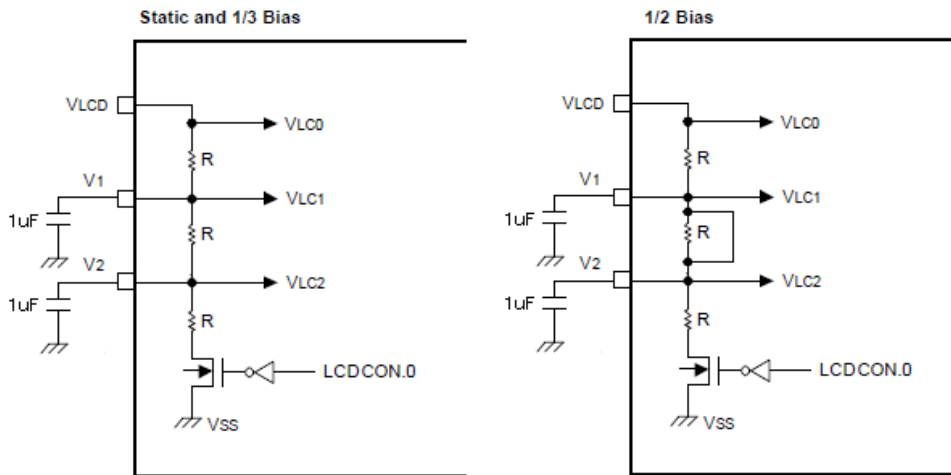
THE FRAME FREQUENCY WHEN RTC IS CLCOK SOURCE(Hz)

LCD frequency	Static	1/2Duty	1/3Duty	1/4Duty
$F_s/2^6 = 512$ Hz	512	256	171	128
$F_s/2^7 = 256$ Hz	256	128	85	64
$F_s/2^8 = 128$ Hz	128	64	43	32
$F_s/2^9 = 64$ Hz	64	32	21	16

Other dividing frequency ($1/2^{10} \sim 1/2^{15}$) is used when OSC or RC is clock source.

LCD drive voltage

V_{LCD} is the voltage source of LCD and divided to four voltage level by internal resistors. V_{LCD} may connect to VDD or other voltage. Two capacitors connecting to V1 and V2 will improve stability for display.



LCD Power Supply	Static Mode	1/2 Bias	1/3 Bias
V _{LCD0}	V _{LCD}	V _{LCD}	V _{LCD}
V _{LCD1}	2/3 V _{LCD}	1/2 V _{LCD}	2/3 V _{LCD}
V _{LCD2}	1/3 V _{LCD}	1/2 V _{LCD}	1/3 V _{LCD}
V _{SS}	0 V	0 V	0 V

COM/SEG output

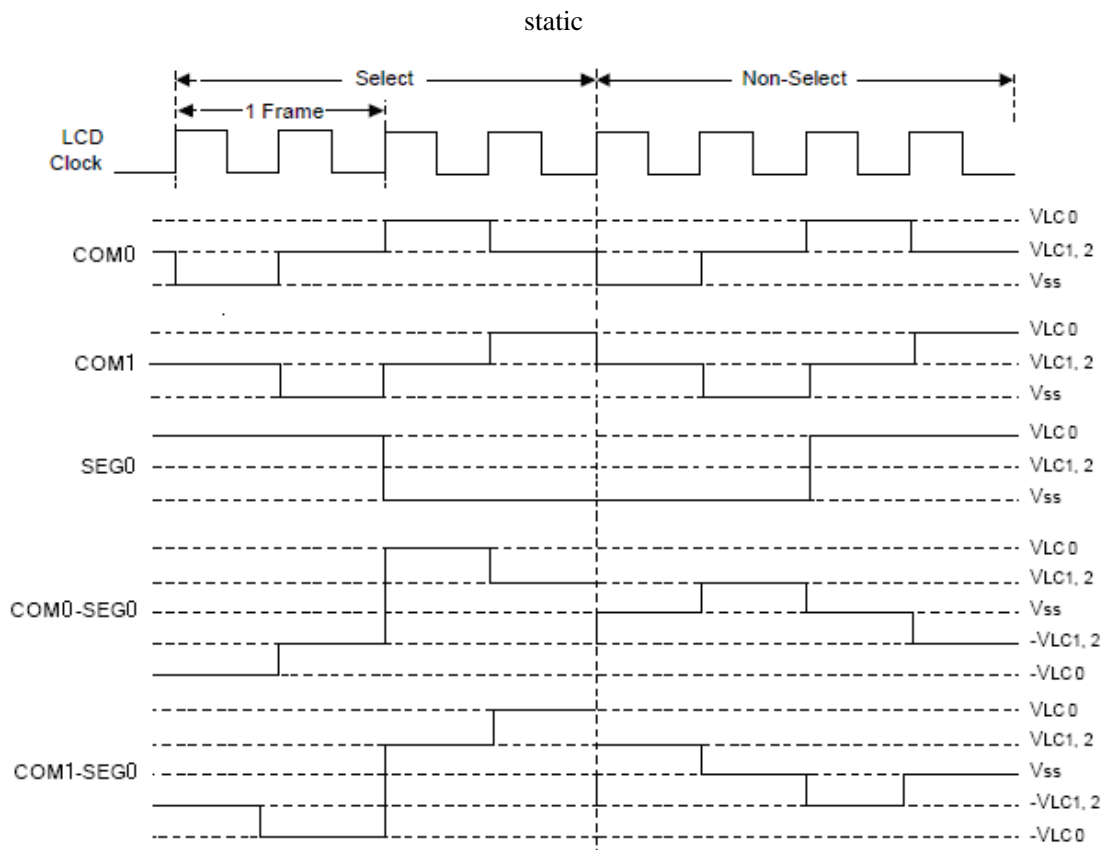
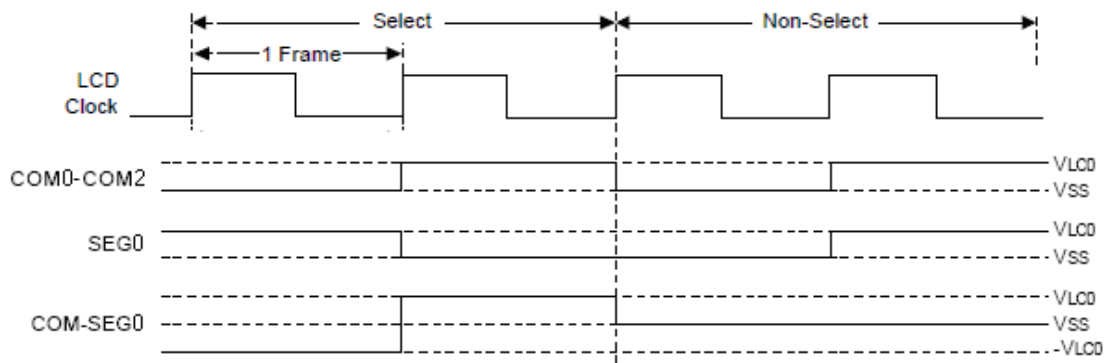
The output number can be 4 COM and 18 SEG (48pin), 4 COM and 8 SEG (28pin and 24pin). COM3 and SEG18 are at common pin. COM3 is effective only at 1/4duty mode and other mode is SEG18.

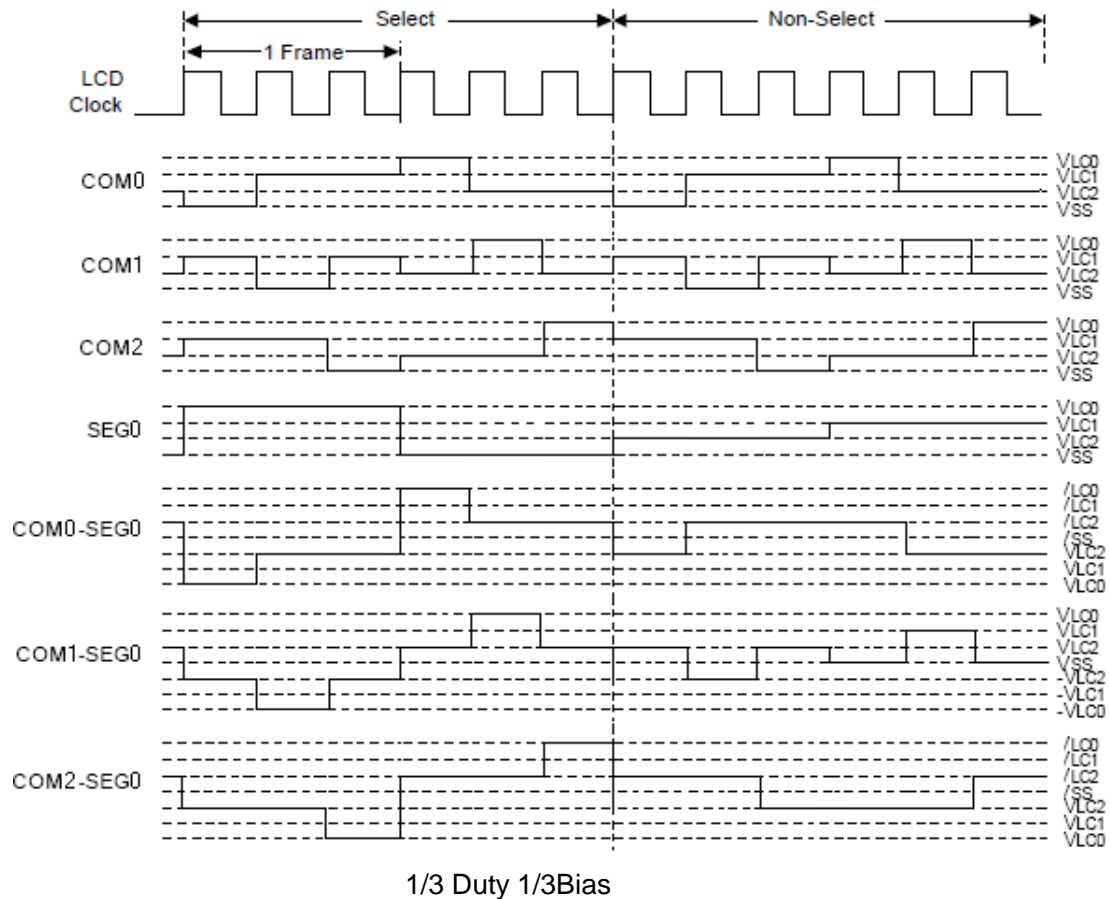
pin	duty	Drive number	bias
-----	------	--------------	------



48	1 (static)	19x1	1/3
	1/2	19x2	1/2
	1/3	19x3	1/3
	1/4	18x4	1/3
24 or 28	1 (static)	9x1	1/3
	1/2	9x2	1/2
	1/3	9x3	1/3
	1/4	8x4	1/3

COM and SEG output wave





5.9 Timer - TIMER0

TIMER0 is an 8bit count-up counter. The counter clock source may come from Fsys or RTC time-out signal or external source. Using external clock input from TMR0 (PB2) allows the user to count external events, measure time internals or pulse widths or generate an accurate time base. While using the internal clock allows the user to generate an accurate time base.

There is a data register - TOD. Before use, a data must be written to TOD. TIMER0 will count from \$00 to this data and generate an interrupt. Then counter restart from \$00.

TIMER0 has three operate mode:

- ◇ **Timer mode** The timer clock comes from internal selected clock source.
- ◇ **Event mode** To count external signal from TMR0, may count up when the signal is from low to high or high to low.
- ◇ **Pulse width measure mode** To measure pulse width from TMR0. When TE is "1", TIMER0 will start counting after TMR0 receives a transient from low to high(from high to low when TE is "0") until TMR0 returns to the original level and resets the TON. (Note: TON is the enable of TIMER0, and only automatic reset at pulse width measure mode. If other two modes, resetting TON must be finished by program.) In other words, only one cycle measurement can be made until TON is set. Then data reading from TOD is the value of pulse width. In the case of counter overflows, the counter is reload from the counter preload register and issues an interrupt request as other two modes. And the value of pulse width



must add the time of generating the interrupt.

It has two registers: TOD、T0C。

TOD (\$06): Timer0 data register

.7-.0 TOD[7:0] 00000000

The data register of Timer0 is writable and readable. When writing it, this data is the overflow data and TIMER0 will count from \$00 to this data driven by internal or external signal. When reading, it is the real-time counting data. If writing to TOD, new overflow data will be effective in next counting cycle.

When TON is “0”, writing to TOD will clear the counting data, and count from \$00 after TON is set. But if do not writing to TOD when TON is “0”, it will count from previous data which TIMER0 stop at.

T0C (\$07): Timer 0 control register

.7-.6 TM[1:0] 00

Mode select

00: TIMER0 not work

01: Event mode, and setting PB2 input

10: Timer mode

11: Pulse wide mode, and setting PB2 input

.5 Not used

.4 TS 0

Internal clock source select, ineffective in event mode.

0: Fsys as clock source

1: RTC time-out signal as clock source

.3 TON 0

Counting enable bit

0: TIMER0 disable

1: TIMER0 enable

.2 TE 0

Defining active edge of counting, Ineffective in timer mode.

0: active on low to high

1: active on high to low

.1-.0 PS[1:0] 00

Dividing frequency select, ineffective in event mode. The clock source is select by TS.

00: 1

01: 1/4

10: 1/16

11: 1/64

The overflow of TIMER0 can be applied to a PFD output (Programmable frequency divider)



at PA3 by option.

5.10 Timer – TIMER1

TIMER1 is a 16bit count-up counter. The counter clock source may come from Fsys or RTC time-out signal or external source or overflow from TIMER0. Using external clock input from TMR1 (PB3) allows the user to count external events, measure time internals or pulse widths or generate an accurate time base.

There are two data register- T1DH and T1DL. Before use, data must be written to T1DH and T1DL. TIMER1 will count from \$0000 to this data and generate an interrupt. Then counter restart from \$0000.

TIMER1 has three operate mode:

- ◇ **Timer mode** The timer clock comes from internal selected clock source.
- ◇ **Event mode** To count external signal from TMR1, may count up when the signal is from low to high or high to low.
- ◇ **Pulse width measure mode** To measure pulse width from TMR1. When TE is “1”, TIMER0 will start counting after TMR1 receives a transient from low to high(from high to low when TE is “0”) until TMR1 returns to the original level and resets the TON. (Note: TON is the enable of TIMER1, and only automatic reset at pulse width measure mode. If other two modes, resetting TON must be finished by program.) In other words, only one cycle measurement can be made until TON is set. Then data reading from T1DH and T1DL is the value of pulse width. In the case of counter overflows, the counter is reload from the counter preload register and issues an interrupt request as other two modes. And the value of pulse width must add the time of generating the interrupt.

It has three registers: T1DH、T1DL、T1C。

T1DH (\$08): Timer 1 data register (High byte)

.7-.0 T1DH[7:0]

T1DL (\$09): Timer 1 data register (Low byte)

.7-.0 T1DL[7:0]

The data registers of Timer1 are writable and readable. When writing it, this data is the overflow data and TIMER1 will count from \$0000 to this data driven by internal or external signal. When reading, it is the real-time counting data. If writing to the data registers, new overflow data will be effective in next counting cycle.

When TON is “0”, writing to the data registers will clear the counting data, and count from \$00 after TON is set. But if do not writing to the data registers when TON is “0”, it will count from previous data which TIMER1 stop at.

One point must be note: there are two data registers, so it can not read or write



data synchronously. When writing, it must write T1DL at first and then T1DH. When reading, it must read T1DH at first and then T1DL.

T1C (\$0A): Timer 1 control register

.7-.6 TM[1:0] 00

Mode select

00: TIMER0 not work

01: Event mode, and setting PB3 input

10: Timer mode

11: Pulse wide mode, and setting PB3 input

.5-.4 TS[1:0] 00

Internal clock source select, ineffective in event mode.

00: Fsys as clock source

01: RTC time-out signal as clock source

10: Ftbi (time base interrupt) as clock source

11: Ftmr0 (overflow from TIMER0) as clock source

.3 TON 0

Counting enable bit

0: TIMER1 disable

1: TIMER1 enable

.2 TE 0

Defining active edge of counting, Ineffective in timer mode.

0: active on low to high

1: active on high to low

.1-.0 PS[1:0] 00

Dividing frequency select, ineffective in event mode. The clock source is select by TS.

00: 1

01: 1/4

10: 1/16

11: 1/64

The overflow of TIMER0 can be applied to a PFD output (Programmable frequency divider) at PA3 by option.

5.11 Interrupt

The device provides two external interrupts, two internal timer interrupts, an internal time base interrupt, an internal real time clock interrupt, eight keyboard interrupts, a software interrupt and an external reset. Interrupts PC address and priority are in the following table (priority is from low to high).

Address	Interrupt
1FE0:1FE1	Reserved
1FE2:1FE3	Reserved



1FE4:1FE5	Reserved
1FE6:1FE7	Reserved
1FE8:1FE9	Reserved
1FEA:1FEB	Reserved
1FEC:1FED	Reserved
1FEE:1FEF	TBI
1FF0:1FF1	RTCI
1FF2:1FF3	T1I
1FF4:1FF5	T0I
1FF6:1FF7	KBI
1FF8:1FF9	INT1
1FFA:1FFB	INT0
1FFC:1FFD	SWI
1FFE:1FFF	RESET

The interrupt control registers INTC0 and INTC1 are used to set enable/disable status and interrupt request flags. Once an interrupt subroutine is serviced, other interrupts are all blocked. This scheme may prevent any further interrupt nesting.

All these interrupts except software interrupt support a wake-up function.

The request flags of KBI, INT0 and INT1 are "0" if relevant interrupt is turn off. But the request flags of T0I, T1I, RTCI and TBI are not affected by enable bit.

INTC0 (\$0B): Interrupt control register 0

.7-.6 not used

.5 KBIE 0

Keyboard interrupt enable/disable

0: KBI disable

1: KBI enable

.4 KBIF 0

Keyboard interrupt flag

0: KBI has not interrupt request

1: KBI has interrupt request

.3 INT1E 0

External interrupt 1 enable/disable

0: INT1 disable

1: INT1 enable and setting PB1 input with pull-up resistor

.2 INT1F 0

External interrupt 1 flag

0: INT1 has not interrupt request

1: INT1 has interrupt request

.1 INT0E 0

External interrupt 0 enable/disable

0: INT0 disable

1: INT0 enable and setting PB0 input with pull-up resistor



.0 INT0F 0

External interrupt 0 flag

0: INT0 has not interrupt request

1: INT0 has interrupt request

INTC1 (\$0C): Interrupt control register 1

.7 TBIE 0

Time base interrupt enable/disable

0: TBI disable

1: TBI enable

.6 TBIF 0

Time base interrupt flag

0: TBI has not interrupt request

1: TBI has interrupt request

.5 RTCIE 0

Real time clock (RTC) interrupt enable/disable

0: RTCI disable

1: RTCI enable

.4 RTCIF 0

RTC interrupt flag

0: RTCI has not interrupt request

1: RTCI has interrupt request

.3 T1IE 0

TIMER1 interrupt enable/disable

0: T1I disable

1: T1I enable

.2 T1IF 0

TIMER1 interrupt flag

0: T1I has not interrupt request

1: T1I has interrupt request

.1 T0IE 0

TIMER0 interrupt enable/disable

0: T0I disable

1: T0I enable

.0 T0IF 0

TIMER0 interrupt flag

0: T0I has not interrupt request

1: T0I has interrupt request

5.12 Low Voltage Detect - LVD

LVD provides the function to monitor voltage supply of chip. The typical LVD voltage is 3.3V. LVDE (bit 7 of MCR1) is enable bit and makes LVD active if it is set. LVDF (bit 5 of MCR1) is flag of LVD and is "1" when VDD is lower than 3.3V.



5.13 Watch Dog Timer - WDT

The WDT clock source is implemented by Fs. The timer is designed to prevent a software malfunction or sequence from jumping to an unknown location with unpredictable results. WDT can be disabled by clearing WDTE (bit 7 of MCR2) and be cleared by WDTC (bit 5 of MCR2).

WDT is a 16bit timer and will make the device reset when counting to 2^{16} . Overflow time is:

$$65536 * 65\mu s = 4.2s \quad (Fs = F_{wdt})$$

$$65536 * 30.5\mu s = 2s \quad (Fs = F_{rtc})$$

$$65536 * 0.5\mu s = 32.7ms \quad (4MHz \text{ crystal}, Fs = F_{osc}/2)$$

5.14 Buzzer

BZ0 and BZ1 are buzzer driving output pair. If willing to use the function, the related PA port should be set as an output. The clock source is from Fs. The buzzer is controlled by register BZCR.

BZCR (\$0D): Buzzer control register

.7-.5 PS[3:0] 0000

Dividing frequency of Fs

0000: $Fs/2^{12}$

0001: $Fs/2^{11}$

0010: $Fs/2^{10}$

0011: $Fs/2^9$

0100: $Fs/2^8$

0101: $Fs/2^7$

0110: $Fs/2^6$

0111: $Fs/2^5$

1000: $Fs/2^4$

1001: $Fs/2^3$

1010: $Fs/2^2$

Others: $Fs/2$

.3-.2 not used

.1 BZ1E 0

0: BZ1 disable

1: BZ1 enable

.0 BZ0E 0

0: BZ0 disable

1: BZ0 enable

5.15 Timer Base

Timer base offers a periodic time out period to generate a regular internal interrupt. Its time out period ranges from $Fs/2^{12}$ to $Fs/2^{15}$ selected by bit 5 and bit 4 of MCR2. If time



base time out occurs, the related interrupt request flag is set.

5.16 Real Time Clock - RTC

RTC is operated in the same manner as time base that is used to supply a regular internal interrupt. Its time out period ranges from $Fs/2^8$ to $Fs/2^{15}$ selected by bit 2 and bit 0 of MCR2. If RTC time out occurs, the related interrupt request flag is set. RTC time out signal also can be applied to be a clock source of timer for getting a longer time out period.

RTC and time base use the same 15bit counter which is cleared only by system reset. So the first time out period of RTC or time base is uncertain.

5.17 Miscellaneous Control Register

There are two miscellaneous control registers to control some functions.

MCR1 (\$0E): Miscellaneous control register 1

.7 LVDE 0

0: LVD disable

1: LVD enable

.6 LVDF 0

0: VDD is higher than V_{LVD}

1: VDD is lower than V_{LVD}

.5 LVRE

0: LVR disable

1: LVR enable

.4-.2 not used

.1 PFDE 0

0: PFD disable

1: PFD enable

.0 PFDC 0

0: PFD clock is from time out of TIMER0

1: PFD clock is from time out of TIMER1

Note: PFD is share with PA3. If outputting PFD, PA3 should be output and data register of PA3 must be written "0".

MCR2 (\$0F): Miscellaneous control register 2

.7 WDTE 0

0: WDT disable

1: WDT enable

.6 WDTC 0

0: useless

1: clear WDT

Note: when read this bit, it is always "0".

.5-.4 TBPS[1:0] 00

Time base interrupt period select

00: $2^{15} * Fs$

01: $2^{14} * Fs$



- 10: $2^{13} \times F_s$
- 11: $2^{12} \times F_s$
- .3 QOSC 0
 - Quickly oscillating select
 - 0: Quickly oscillating enable
 - 1: Quickly oscillating disable
- .2-.0 RT[2:0] 000
 - RTC interrupt period select
 - 000: $2^{15} \times F_s$
 - 001: $2^{14} \times F_s$
 - 010: $2^{13} \times F_s$
 - 011: $2^{12} \times F_s$
 - 100: $2^{11} \times F_s$
 - 101: $2^{10} \times F_s$
 - 110: $2^9 \times F_s$
 - 111: $2^8 \times F_s$

5.18 OPBIT

OPBIT is a special byte in OTP ROM and used to configure some initial functions for the device. OPBIT is set when OTP written.

- .7 ENCR
 - 0: OTP read protection
 - 1: OTP can be read
- .6 Not used
- .5 LCD control in stop mode
 - 0: LCD on in stop mode
 - 1: LCD off in stop mode
- .4 LVREO
 - 0: LVR off
 - 1: LVR on
- .3-.2 FS[1:0]
 - 00、11: $F_s = F_{sys}$
 - 01: $F_s = F_{rtc}$
 - 10: $F_s = F_{wdt}$
- .1-.0 FSYS[1:0]
 - 00、11: $F_{sys} = F_{osc}/2$
 - 01: $F_{sys} = F_{rc}/2$
 - 10: $F_{sys} = F_{rtc}/2$

6. ELECTRICAL DATA

6.1 Absolute Maximum Ratings

($T_A = 25^\circ\text{C}$)



Parameter	Symbol	Condition	Rating	Unit
Supply voltage	V_{DD}	-	$V_{SS}-0.3$ to $V_{SS}+6.5$	V
Input voltage	V_I	All ports	$V_{SS}-0.3$ to $V_{DD}+0.3$	V
Operating temperature	T_A	-	-40 to +85	°C
Storage temperature	T_S	-	-65 to +150	°C

6.2 DC Electrical Characteristics

($T_A=25^\circ\text{C}$ $V_{DD}=2.7-5.5\text{V}$)

Parameter	Sym.	Condition	Min	Typ	Max	Unit
Operate voltage	V_{DD}	Oscillator Frequency $\leq 4\text{MHz}$	2.5	-	5.5	V
		Oscillator Frequency $\leq 8\text{MHz}$	3.5	-	5.5	V
LCD operate voltage	V_{LCD}	-	2.5	-	5.5	V
Input high voltage	V_{IH1}	All ports	$0.7V_{DD}$	-	V_{DD}	V
Input low voltage	V_{IL1}	All ports	0	-	$0.3V_{DD}$	V
I/O ports source current	I_{OH}	$V_{OH}=0.9V_{DD}$	5	12	-	mA
I/O ports sink current	I_{OL}	$V_{OL}=0.1V_{DD}$	10	20	-	mA
Pull-up resistors	R_{PH}	PA, PB	10	25	40	k Ω
LVR	V_{LVR}	-	2.7	3.0	3.3	V
LVD	V_{LVD}	-	3.0	3.3	3.6	V
Dynamic working current	I_{DD}	4MHz clock	-	3	5	mA
Standby working current	I_{STB}	STOP mode, LVR off, LCD off	-	-	1	μA
		STOP mode, LVR on, LCD off	-	10	15	μA
		STOP mode, LVR off, LCD on	-	20	30	μA

7. Instruction Set

7.1 Addressing Modes

The addressing modes define the manner in which an instruction is to obtain the data



required for its execution. There are 8 modes:

- 1) Inherent
- 2) Immediate
- 3) Direct
- 4) Extended
- 5) Indexed, no offset
- 6) Indexed, 8-bit offset
- 7) Indexed, 16-bit offset
- 8) Relative

7.1.1 Inherent Addressing Mode

In inherent addressing mode, all information required for the operation is already inherently known to the CPU, and no external operand from memory or from the program is needed. The operands, if any, are only the index register and accumulator, and are always 1-byte instructions.

7.1.2 Immediate Addressing Mode

In the immediate addressing mode, the operand is contained in the byte immediately following the opcode. This mode is used to hold a value or constant which is known at the time the program is written and which is not changed during program execution. These are 2-byte instructions, one for the opcode and one for the immediate data byte.

7.1.3 Direct Addressing Mode

The direct addressing mode is similar to the extended addressing mode except the upper byte of the operand address is assumed to be \$00. Thus, only the lower byte of the operand address needs to be included in the instruction. Direct addressing allows you to efficiently address the lowest 256 bytes in memory. This area of memory is called the direct page and includes on-chip RAM and I/O registers. Direct addressing is efficient in both memory and time. Direct addressing mode instructions are usually two bytes, one for the opcode and one for the low-order byte of the operand address.

7.1.4 Extended Addressing Mode

In the extended addressing mode, the address of the operand is contained in the two bytes following the opcode. Extended addressing references any location in the MCU memory space including I/O, RAM, ROM and EPROM. Extended addressing mode instructions are three bytes, one for the opcode and two for the address of the operand.

7.1.5 Indexed, No Offset Addressing Mode

In the indexed, no-offset addressing mode, the effective address of the instruction is contained in the 8-bit index register. Thus, this addressing mode can access the first 256 memory locations. These instructions are only one byte.

7.1.6 Indexed, 8-bit Offset Addressing Mode

In the indexed, 8-bit offset addressing mode, the effective address is obtained by adding



the contents of the byte following the opcode to the contents of the index register. This mode of addressing is useful for selecting the kth element in an n element table. To use this mode, the table must begin in the lowest 256 memory locations and may extend through the first 511 memory locations (IFE is the last location which the instruction may access). Indexed 8-bit offset addressing can be used for ROM, RAM, or I/O. This is a 2-byte instruction with the offset contained in the byte following the opcode. The content of the index register (X) is not changed. The offset byte supplied in the instruction is an unsigned 8-bit integer.

7.1.7 Indexed, 16-bit Offset Addressing Mode

In the indexed, 16-bit offset addressing mode, the effective address is the sum of the contents of the 8-bit index register and the two bytes following the opcode. The content of the index register is not changed. These instructions are three bytes, one for the opcode and two for a 16-bit offset.

7.1.8 Relative Addressing Mode

The relative addressing mode is used only for branch instructions. Branch instructions, other than the branching versions of bit-manipulation instructions, generate two machine-code bytes: one for the opcode and one for the relative offset. Because it is desirable to branch in either direction, the offset byte is a signed twos-complement offset with a range of -127 to +128 bytes (with respect to the address of the instruction immediately following the branch instruction). If the branch condition is true, the contents of the 8-bit signed byte following the opcode (offset) are added to the contents of the program counter to form the effective branch address; otherwise, control proceeds to the instruction immediately following the branch instruction.

7.2 Instruction Type

There are 65 instructions in CPU, and can be divided into 5 types.

- 1) Register/Memory Instructions
- 2) Read/Modify-Write Instructions
- 3) Branch Instructions
- 4) Control Instructions
- 5) bit manipulate Instructions

7.3 Instruction Set

Instruction	Operation	Function	Status					Addressing MODE	Opcode	Opdata	#Cycle
			H	I	N	Z	C				
ADC #opr	Add with Carry	$A \leftarrow (A)+(M)+(C)$						IMM	A9	ii	2
ADC opr								DIR	B9	dd	3
ADC opr			*	-	*	*	*	EXT	C9	hh	4
ADC opr,X								IX2	D9	ll	5



ADC opr,X								IX1	E9	ee	4
ADC ,X								IX	F9	ff	3
										ff	
ADD #opr								IMM	AB	ii	2
ADD opr								DIR	BB	dd	3
ADD opr	Add without Carry	$A \leftarrow (A)+(M)$	*	-	*	*	*	EXT	CB	hh	4
ADD opr,X								IX2	DB	ll	5
ADD opr,X								IX1	EB	ee	4
ADD ,X								IX	FB	ff	3
										ff	
AND #opr								IMM	A4	ii	2
AND opr								DIR	B4	dd	3
AND opr	Logical AND	$A \leftarrow (A) \wedge (M)$	-	-	*	*	-	EXT	C4	hh	4
AND opr,X								IX2	D4	ll	5
AND opr,X								IX1	E4	ee	4
AND ,X								IX	F4	ff	3
										ff	
ASL opr								DIR	38	dd	5
ASLA	Arithmetic Shift Left		-	-	*	*	*	INH	48		3
ASLX	(Same as LSL)							INH	58		3
ASL opr,X								IX1	68	ff	6
ASL ,X								IX	78		5
ASR opr								DIR	37	dd	5
ASRA	Arithmetic Shift Right		-	-	*	*	*	1.1.1.1.1	47		3
ASRX										57	
ASR opr,X									67	ff	6
ASR ,X								INH	77		5
								IX1			
								IX			
BCC rel	Branch if Carry Bit Clear	$PC \leftarrow (PC)+2+rel$? C=0	-	-	-	-	-	REL	24	rr	3
BCLR n opr	Clear Bit n	$Mn \leftarrow 0$	-	-	-	-	-	DIR(b0)	11	dd	5
								DIR(b1)	13	dd	5
								DIR(b2)	15	dd	5
								DIR(b3)	17	dd	5
								DIR(b4)	19	dd	5
								DIR(b5)	1B	dd	5



								DIR(b6)	1D	dd	5
								DIR(b7)	1F	dd	5
BCS rel	Branch if Carry Bit Set (Same as BLO)	PC ← (PC)+2+rel ? C=1 (与 BLO 相同)	-	-	-	-	-	REL	25	rr	3
BEQ rel	Branch if Equal	PC ← (PC)+2+rel ? Z=1	-	-	-	-	-	REL	27	rr	3
BHCC rel	Branch if Half Carry Bit Clear	PC ← (PC)+2+rel ? H=0	-	-	-	-	-	REL	28	rr	3
BHCS rel	Branch if Half Carry Bit Set	PC ← (PC)+2+rel ? H=1	-	-	-	-	-	REL	29	rr	3
BHI rel	Branch if Higher	PC ← (PC)+2+rel ? (C ∨ Z)=0	-	-	-	-	-	REL	22	rr	3
BHS rel	Branch if Higher or Same	PC ← (PC)+2+rel ? C=0	-	-	-	-	-	REL	24	rr	3
BIT #opr	Bit Test Accumulator with Memory Byte	(A)^(M)	-	-	*	*	-	IMM	A5	ii	2
BIT opr								DIR	B5	dd	3
BIT opr								EXT	C5	hh	4
BIT opr,X								IX2	D5	ll	5
BIT opr,X								IX1	E5	ee	4
BIT ,X								IX	F5	ff	3
										ff	
BLO rel	Branch if Lower (Same as BCS)	PC ← (PC)+2+rel ? C=1	-	-	-	-	-	REL	25	rr	3
BLS rel	Branch if Lower or Same	PC ← (PC)+2+rel ? (C ∨ Z)=1	-	-	-	-	-	REL	23	rr	3
BMC rel	Branch if Interrupt Mask Clear	PC ← (PC)+2+rel ? I=0	-	-	-	-	-	REL	2C	rr	3
BMI rel	Branch if Minus	PC ← (PC)+2+rel ? N=1	-	-	-	-	-	REL	2B	rr	3
BMS rel	Branch if Interrupt Mask Set	PC ← (PC)+2+rel ? I=1	-	-	-	-	-	REL	2D	rr	3
BNE rel	Branch if Not Equal	PC ← (PC)+2+rel ? Z=0	-	-	-	-	-	REL	26	rr	3
BPL rel	Branch if Plus	PC ← (PC)+2+rel ? N=0	-	-	-	-	-	REL	2A	rr	3

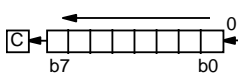
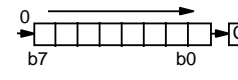


BSET n opr	Set Bit n	$M_n \leftarrow 1$	-	-	-	-	-	DIR(b1)	12	dd	5
								DIR(b2)	14	dd	5
								DIR(b3)	16	dd	5
								DIR(b4)	18	dd	5
								DIR(b5)	1A	dd	5
								DIR(b6)	1C	dd	5
								DIR(b7)	1E	dd	5
BSR rel	Branch to Subroutine	$PC \leftarrow (PC)+2$	-	-	-	-	-	REL	AD	rr	6
		push(PCL); $SP \leftarrow (SP)-1$									
		push(PCH); $SP \leftarrow (SP)-1$									
		$PC \leftarrow (PC)+rel$									
CLC	Clear Carry Bit	$C \leftarrow 0$	-	-	-	-	0	INH	98		2
CLI	Clear Interrupt Mask	$I \leftarrow 0$	-	0	-	-	-	INH	9A		2
CLR opr	Clear Byte	$M \leftarrow \$00$	-	-	0	1	-	DIR	3F	dd	5
CLRA		$A \leftarrow \$00$						INH	4F		3
CLR X		$X \leftarrow \$00$						INH	5F		3
CLR opr,X		$M \leftarrow \$00$						IX1	6F	ff	6
CLR ,X		$M \leftarrow \$00$						IX	7F		5
CMP #opr	Compare Accumulator with Memory Byte	$(A) -(M)$	-	-	*	*	*	IMM	A1	ii	2
CMP opr								DIR	B1	dd	3
CMP opr								EXT	C1	hh	4
CMP opr,X								IX2	D1	ll	5
CMP opr,X								IX1	E1	ee	4
CMP ,X								IX	F1	ff	3
COM opr	Complement Byte (One's Complement)	$M \leftarrow \$FF-(M)$	-	-	*	*	1	DIR	33	dd	5
COMA		$A \leftarrow \$FF-(A)$						INH	43		3
COM X		$X \leftarrow \$FF-(X)$						INH	53		3
COM opr,X		$M \leftarrow \$FF-(M)$						IX1	63	ff	6
COM ,X		$M \leftarrow \$FF-(M)$						IX	73		5
CPX #opr	Compare Index Register with Memory Byte	$(X) -(M)$	-	-	*	*	*	IMM	A3	ii	2
CPX opr								DIR	B3	dd	3
CPX opr								EXT	C3	hh	4
CPX opr,X								IX2	D3	ll	5



CPX opr,X								IX1	E3	ee	4
CPX ,X								IX	F3	ff	3
										ff	
DEC opr		$M \leftarrow (M)-1$						DIR	3A	dd	5
DECA		$A \leftarrow (A)-1$						INH	4A		3
DECX	Decrement Byte	$X \leftarrow (X)-1$	-	-	*	*	-	INH	5A		3
DEC opr,X		$M \leftarrow (M)-1$						IX1	6A	ff	6
DEC ,X		$M \leftarrow (M)-1$						IX	7A		5
EOR #opr								IMM	A8	ii	2
EOR opr								DIR	B8	dd	3
EOR opr	EXCLUSIVE OR	$A \leftarrow (A) \oplus (M)$	-	-	*	*	-	EXT	C8	hh	4
EOR opr,X	Accumulator with							IX2	D8	ll	5
EOR opr,X	Memory Byte							IX1	E8	ee	4
EOR ,X								IX	F8	ff	3
										ff	
INC opr		$M \leftarrow (M)+1$						DIR	3C	dd	5
INCA		$A \leftarrow (A)+1$						INH	4C		3
INCX	Increment Byte	$X \leftarrow (X)+1$	-	-	*	*	-	INH	5C		3
INC opr,X		$M \leftarrow (M)+1$						IX1	6C	ff	6
INC ,X		$M \leftarrow (M)+1$						IX	7C		5
JMP opr								DIR	BC	dd	2
JMP opr								EXT	CC	hh	3
JMP opr,X	Unconditional Jump	$PC \leftarrow \text{Jump Address}$	-	-	-	-	-	IX2	DC	ll	4
JMP opr,X								IX1	EC	ee	3
JMP ,X								IX	FC	ff	2
										ff	
JSR opr		$PC \leftarrow (PC)+n(n=1,2,\text{or } 3)$						DIR	BD	dd	5
JSR opr	Jump to Subroutine							EXT	CD	hh	6
JSR opr,X		push	-	-	-	-	-	IX2	DD	ll	7
JSR opr,X		$(PCL); SP \leftarrow (SP)-1$						IX1	ED	ee	6
JSR ,X		push(PCH); $SP \leftarrow (SP)-1$						IX	FD	ff	5
		$PC \leftarrow \text{Effective Address}$								ff	
LDA #opr								IMM	A6	ii	2
LDA opr	Load Accumulator with							DIR	B6	dd	3



LDA opr	Memory Byte	$A \leftarrow (M)$	-	-	*	*	-	EXT	C6	hh	4
LDA opr,X								IX2	D6	ll	5
LDA opr,X								IX1	E6	ee	4
LDA ,X								IX	F6	ff	3
										ff	
LDX #opr								IMM	AE	ii	2
LDX opr	Load Index Register with							DIR	BE	dd	3
LDX opr	Memory Byte	$X \leftarrow (M)$	-	-	*	*	-	EXT	CE	hh	4
LDX opr,X								IX2	DE	ll	5
LDX opr,X								IX1	EE	ee	4
LDX ,X								IX	FE	ff	3
										ff	
LSL opr								DIR	38	dd	5
LSLA								INH	48		3
LSLX	Logical Shift Left (Same		-	-	*	*	*	INH	58		3
LSL opr,X	as ASL)							IX1	68	ff	6
LSL ,X								IX	78		5
LSR opr								DIR	34	dd	5
LSRA								INH	44		3
LSRX	Logical Shift Right		-	-	0	*	*	INH	54		3
LSR opr,X								IX1	64	ff	6
LSR ,X								IX	74		5
MUL	Unsigned Multiply	$X:A \leftarrow (X)X(A)$	0	-	-	-	0	INH	42		1
											1
NEG opr		$M \leftarrow \neg(M)$						DIR	30	dd	5
NEGA	Negate Byte (Two's	$A \leftarrow \neg(A)$						INH	40		3
NEGX	Complement)	$X \leftarrow \neg(X)$	-	-	*	*	*	INH	50		3
NEG opr,X		$M \leftarrow \neg(M)$						IX1	60	ff	6
NEG ,X		$M \leftarrow \neg(M)$						IX	70		5
NOP	No Operation		-	-	-	-	-	INH	9D		2
ORA #opr								IMM	AA	ii	2
ORA opr	Logical OR Accumulator							DIR	BA	dd	3
ORA opr	with Memory	$A \leftarrow (A) \vee (M)$	-	-	*	*	-	EXT	CA	hh	4
ORA opr,X								IX2	DA	ll	5
ORA opr,X								IX1	EA	ee	4
ORA ,X								IX	FA	ff	3



									ff		
ROL opr	Rotate Byte Left through Carry Bit		-	-	*	*	*	DIR	39	dd	5
ROLA			INH	49		3					
ROLX			INH	59		3					
ROL opr,X			IX1	69	ff	6					
ROL ,X			IX	79		5					
ROR opr	Rotate Byte Right through Carry Bit		-	-	*	*	*	DIR	36	dd	5
RORA			INH	46		3					
RORX			INH	56		3					
ROR opr,X			IX1	66	ff	6					
ROR ,X			IX	76		5					
RSP	Reset Stack Pointer	$SP \leftarrow \$00FF$	-	-	-	-	-	INH	9C		2
RTI	Return from Interrupt	$SP \leftarrow (SP)+1;$ Pull(CCR) $SP \leftarrow (SP)+1; \text{ Pull}(A)$ $SP \leftarrow (SP)+1; \text{ Pull}(X)$ $SP \leftarrow (SP)+1;$ Pull(PCH) $SP \leftarrow (SP)+1;$ Pull(PCL)	*	*	*	*	*	INH	80		9
RTS	Return from Subroutine	$SP \leftarrow (SP)+1;$ Pull(PCH) $SP \leftarrow (SP)+1;$ Pull(PCL)	-	-	-	-	-	INH	81		6
SBC #opr	Subtract Memory Byte and Carry Bit from Accumulator	$A \leftarrow (A)-(M)-(C)$	-	-	*	*	*	IMM	A2	ii	2
SBC opr			DIR	B2	dd	3					
SBC opr			EXT	C2	hh	4					
SBC opr,X			IX2	D2	ll	5					
SBC opr,X			IX1	E2	ee	4					
SBC ,X			IX	F2	ff	3					
SEC	Set Carry Bit	$C \leftarrow 1$	-	-	-	-	1	INH	99		2
SEI	Set Interrupt Mask	$I \leftarrow 1$	-	1	-	-	-	INH	9B		2
STA opr	Store Accumulator in Memory	$M \leftarrow (A)$	-	-	*	*	-	DIR	B7	dd	4
STA opr			EXT	C7	hh	5					
STA opr,X			IX2	D7	ll	6					



STA opr,X								IX1	E7	ee	5
STA ,X								IX	F7	ff	4
STOP	Stop Oscillator and Enable IRQ Pin		-	0	-	-	-	INH	8E		2
STX opr	Store Index Register In Memory	$M \leftarrow (X)$	-	-	*	*	-	DIR	BF	dd	4
STX opr								EXT	CF	hh	5
STX opr,X								IX2	DF	ll	6
STX opr,X								IX1	EF	ee	5
STX ,X								IX	FF	ff	4
SUC #opr	Subtract Memory Byte from Accumulator	$A \leftarrow (A) - (M)$	-	-	*	*	*	IMM	A0	ii	2
SUB opr								DIR	B0	dd	3
SUB opr								EXT	C0	hh	4
SUB opr,X								IX2	D0	ll	5
SUB opr,X								IX1	E0	ee	4
SUB ,X								IX	F0	ff	3
SWI	Software Interrupt	$PC \leftarrow (PC) + 1$; Push(PC L) $SP \leftarrow (SP) - 1$; Push(PC H) $SP \leftarrow (SP) - 1$; Push(X) $SP \leftarrow (SP) - 1$; ush(CCR) $SP \leftarrow (SP) - 1$; $I \leftarrow 1$ $PCH \leftarrow$ Interrupt Vector High Byte $PCL \leftarrow$ Interrupt Vector Low Byte	-	1	-	-	-	INH	83		1
											0
TAX	Transfer Accumulator to Index Register	$X \leftarrow (A)$	-	-	-	-	-	INH	97		2
TST opr	Test Memory Byte for Negative or Zero	$(M) - \$00$	-	-	*	*	-	DIR	3D	dd	4
TSTA								INH	4D		3
TSTX								INH	5D		3
TST opr,X								IX1	6D	ff	5



TST ,X								IX	7D		4
TXA	Transfer Index Register to Accumulator	A ← (X)	-	-	-	-	-	INH	9F		2
WAIT	Stop CPU Clock and Enable Interrupts		-	0	-	-	-	INH	8F		2

- | | |
|--|--|
| <ul style="list-style-type: none"> A Accumulator C Carry/borrow flag CCR Condition code register dd Direct address of operand dd rr Direct address of operand and relative offset of branch instruction DIR Direct addressing mode offset byte ee ff High and low bytes of offset in indexed, 16-bit offset addressing byte EXT Extended addressing mode ff Offset byte in indexed, 8-bit offset addressing H Half-carry flag hh ll High and low bytes of operand address in extended addressing I Interrupt mask ii Immediate operand byte IMM Immediate addressing mode INH Inherent addressing mode IX Indexed, no offset addressing mode IX1 Indexed, 8-bit offset addressing mode IX2 Indexed, 16-bit offset addressing mode M Memory location N Negative flag n Any bit — Not affected | <ul style="list-style-type: none"> opr Operand (one or two bytes) PC Program counter PCH Program counter high byte PCL Program counter low byte REL Relative addressing mode rel Relative program counter rr Relative program counter offset SP Stack pointer X Index register Z Zero flag # Immediate value ∧ Logical AND ∨ Logical OR ⊕ Logical EXCLUSIVE OR () Contents of -() Negation (twos complement) ← Loaded with ? If : Concatenated with ↔ Set or cleared — Not affected |
|--|--|