十速

TM56F6922

DATA SHEET

Rev 1.0

(Please read the precautions on the second page before use)

PRECAUTIONS

1. **Please refer to DC Electrical Characteristics (page 113) before using IO Current**
2. **Please refer to EEPROM Electrical Characteristics (page 115) before using EEPROM**
3. **ADDWXC/ SUBXW / SUBXWB / SUBWXB 4 instructions do not support indirect addressing, only direct addressing.**

AMENDMENT HISTORY

Version	Date	Description
0.90	Mar, 2025	New release
0.91	Apr, 2025	Modify DC IO Current characteristics
0.92	May, 2025	PRECAUTIONS Added: The 4 newly added instructions do not support indirect addressing
1.0	Jun, 2026	Modify INT0/INT1/INT2 Pin Modify Mode Settin Remove PWM Open Drain pin state description Corrected ADCVREFS initial value to 0 Update EEPROM characteristics Add SOP-20 Pin Package

CONTENTS

FEATURES

1. **ROM: 4K x 16-bit Flash program memory**
2. **EEPROM: 256 x 8-bit**
3. **RAM: 592 x 8-bit**
4. **STACK: 8 level**
5. **System clock selection:**
   - Fast-clock:
     - ✧ Internal RC fast clock: Max. 18.432 MHz (can be calibrated)
     - ✧ FXT (external Fast crystal oscillator): 1M~18MHz
   - Slow-clock
     - ✧ SIRC internal RC slow clock: 50 KHz or 60KHz @VCC=5V
     - ✧ SXT external slow clock: 32.768KHz
6. **System clock prescaler:**
   - The system clock can be divided into 1/2/4/8 options
7. **Dual system clock:**
   - FIRC+SIRC
   - FIRC+SXT
   - FXT+SIRC
8. **I/O port: up to 26 programmable I/O pins**
   - Open drain output
   - CMOS push-pull output
   - Schmitt trigger input with pull-up resistor options
   - Support high sink current mode : 70mA @VCC=5V, V_{OL}=0.5V; 140mA@VCC=5V, V_{OL}=1.2V
   - Support constant current drive mode: 19mA@VCC=5V,V_{OH}=2.5V~3.8V
9. **Power Saving Operation Mode**
   - FAST Mode: Slow-clock can be disabled or enabled, Fast-clock keeps CPU running
   - SLOW Mode: Fast-clock can be disabled or enabled, Slow-clock keeps CPU running
   - IDLE Mode: Fast-clock and CPU stop. Slow-clock, T2, or Wake-up Timer keep running
   - STOP Mode: All clocks stop, T2 and Wake-up Timer stop
10. **3 Independent Timers**
    - Timer0
      - 8-bit timer divided by 1~256 pre-scaler option, Reload/Interrupt/Stop function
    - Timer1
      - 8-bit timer divided by 1~256 pre-scaler option, Reload/Interrupt/Stop function

- Overflow and Toggle out

- T2
  - 15-bit timer with 4 interrupt interval time options
  - IDLE mode wake-up timer or used as one simple 15-bit time base
  - Clock source: Slow-clock (SIRC), Fsys/128

11. Interrupt

- Three external interrupt pins
  - 1 pin is a falling edge wake-up trigger and interrupt
  - 2 pins for rising or falling edge wake-up trigger and interrupt
- Timer0 / Timer1 / T2 / WKT Interrupt
- ADC Interrupt
- I2C Interrupt
- UART Interrupt
- All pin interrupts
- LVD Interrupt
- EEP programming completion interrupt

12. Wake-up timer (WKT)

- Clocked by built-in RC oscillator with 4 adjustable interrupt times

  20.5 ms/41 ms/82 ms/164 ms @50KHz SRC VCC=3.6V

13. Watchdog Timer (WDT)

- Clocked by built-in RC oscillator with 4 adjustable reset times

  164ms/328ms/655ms/1311ms @50KHz SRC VCC=3.6V

- Watchdog timer can be disabled/enabled in STOP mode

14. PWM

- PWM0 :
  - ✧ 16 bits, duty-adjustable, period-adjustable controlled PWM
  - ✧ PWM0 clock source: system clock source or FIRC/256 or FIRC or FIRC*2
  - ✧ PWM0 with 4 output modes
  - ✧ Complementary PWM0 output (PWM0P, PWM0N)
  - ✧ Non overlap time durations adjustable. (0~15 PWM CLK)
- PWM1~5:
  - ✧ 16 bits, duty-adjustable (Independent) , period-adjustable controlled (shared with PWM0)
  - ✧ Clock source (Shared with PWM0)

15. I2C Interface

- Specific purpose slave I2C interface with interrupt function

16. UART Interface

- 7/8/9 bits mode TX/RX selectable
- Supported Baud-Rate range from 9600bps to 115200 bps with proper selected oscillation frequency and baud rate clock divide
- Automatic parity generation and detection
- Detects Overrun, Frame Error and Parity Error

17. 12-bit ADC converter with 26 input channels and 1 internal reference voltage

- Internal Reference Voltage: VBG 1.18V±1% @VCC=5V~2.5V, 25℃
- Internal reference voltage::1/4VCC
- ADC reference voltage:VCC, VBG

18. All pin change wake up (negedge and posedge trigger)

19. Reset Sources

- Power On Reset/Watchdog Reset/Low Voltage Reset/External Pin Reset

20. Low Voltage Reset (LVR) /Low Voltage Detection Flag (LVD) Option:

- 16-Level Low Voltage Reset: 1.6~ 3.5 V
- 16-Level Low Voltage Detection Flag (Disable, 1.7V ~ 3.5V)

21.Operating Voltage:

- Fsys= 1 MHz, LVR ~5.5V
- Fsys=18.432 MHz, 2.5~5.5V

Note: Power-on VCC must exceed POR 1.6V and selected LVR level, refer to "Electrical Characteristics Graph" to avoid entering ROM dead zone.

22. Operating Temperature Range : -40℃ to + 105℃

23. RDCTL: Read signal delay control for Program ROM

- **The user must switch this register to "4ns" to enhance the performance of minimal operating voltage**

24. Integrated 16-bit Cyclic Redundancy Check (CRC)

25. Table Read Instruction: 16-bit ROM data lookup table

26. Instruction set: 43 Instructions

27. Instruction Execution Time

- 2 system clocks (Fsys) per instruction except branch

28. Built-in 1/2 bias for software LCD display (4 COM)

29. Programming connectivity support 4-wire (ICP) or 5-wire program

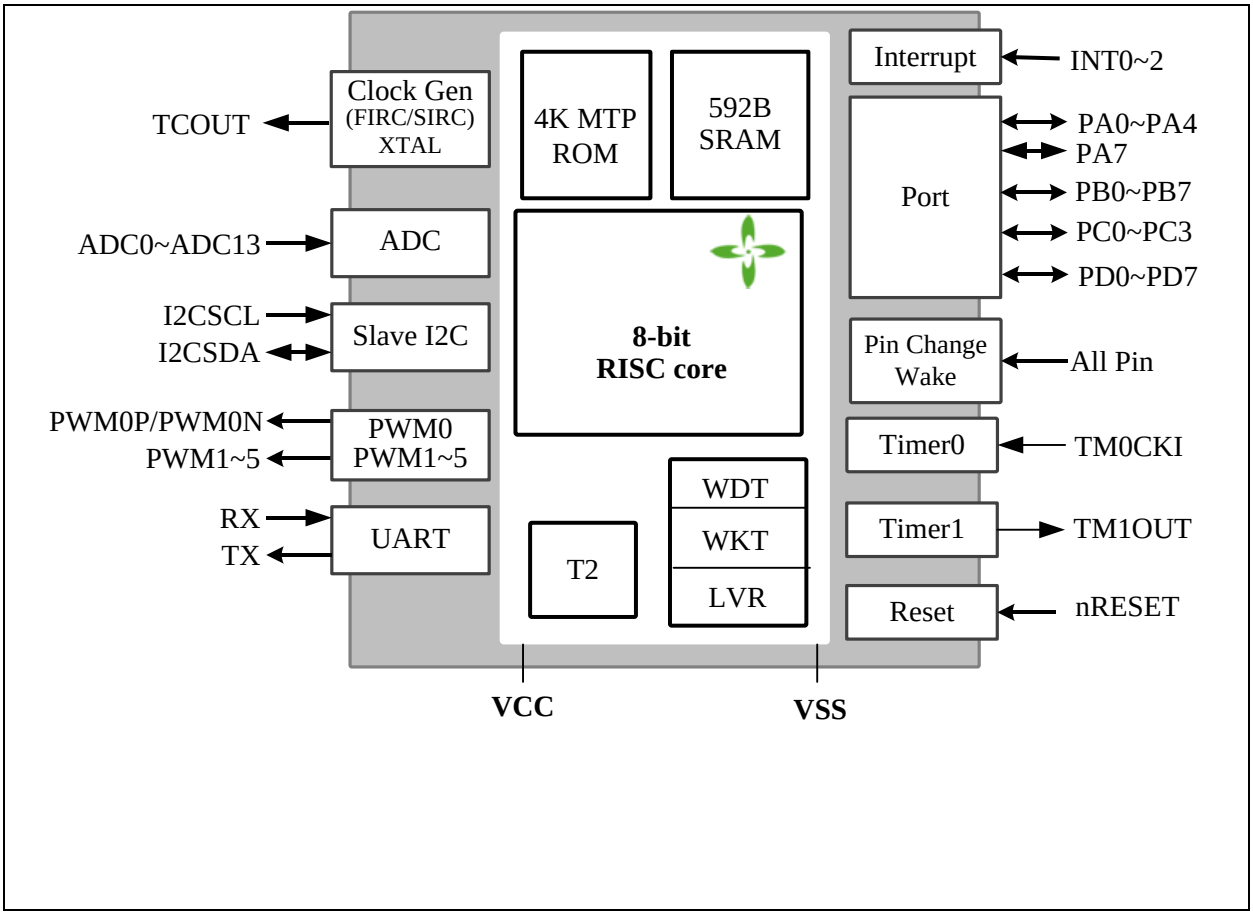
30. Package Types:

- SOP-28/SSOP-28/TSSOP-20/SOP-20

31. Supported EV board (ICE) on Real Chip Debug

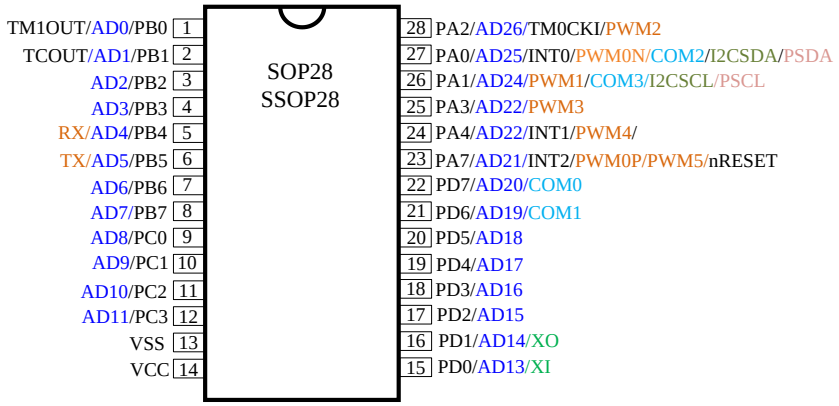
- Use PA0/PA1 pin or PC0/PC1 pin
- Share with ICP programming pin

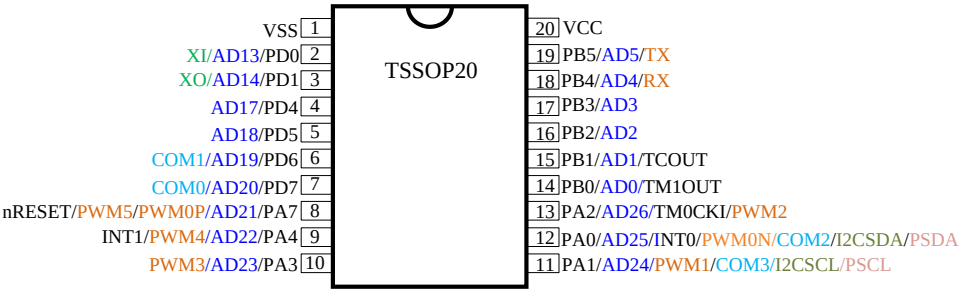
SYSTEM BLOCK DIAGRAM

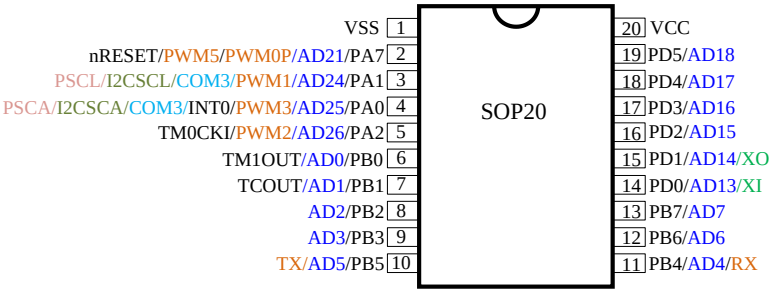


TM56F6922 Block Diagram

PIN ASSIGNMENT DIAGRAM







PIN DESCRIPTIONS

Name	In/Out	Pin Description
PA7, PA4–PA0	I/O	Bit-programmable I/O port for Schmitt-trigger input, CMOS push-pull output or open-drain output. Pull-up resistor are assignable by software.
nRESET	I	External active low reset/ Schmitt-trigger input
VCC, VSS	P	Power input pin and ground
INT0~INT2	I	External interrupt input
PB7-PB0	I/O	Bit-programmable I/O port for Schmitt-trigger input, CMOS push-pull output or open-drain output. Pull-up resistor are assignable by software.
PC3-PC0	I/O	Bit-programmable I/O port for Schmitt-trigger input, CMOS push-pull output or open-drain output. Pull-up resistor are assignable by software.
PD7-PD0	I/O	Bit-programmable I/O port for Schmitt-trigger input, CMOS push-pull output or open-drain output. Pull-up resistor are assignable by software.
PWM0P/PWM0N PWM1 PWM2 PWM3 PWM4 PWM5	O	PWM0/PWM1/PWM2/PWM3/PWM4/PWM5 outputs
AD26~AD0	I	Analog to Digital Convert input pin
COM0~COM3	O	1/2 bias for software LCD display
TM0CKI	I	Timer0's input pin in counter mode
TCOUT	O	Fsys/2 时钟输出
TM1OUT	O	Timer1 overflow toggle output
I2CSCL	I	I2C Serial clock input
I2CSDA	I/O	I2C serial data pin
PSCL	I	I2C Serial clock input for program/ICE
PSDA	I/O	I2C serial data pin for program/ICE
XI, XO	-	Crystal/Resonator oscillator connection for System clock (FXT or SXT)
RX	I	UART receive pin
TX	O	UART transmit pin

Programming pins:
Normal mode:VCC / VSS / PA0 / PA1 / PB0
ICP mode (4-wire):VCC / VSS / PA0 / PA1-When using ICP (In-circuit Program) mode, the PCB needs to remove all components of PA0, PA1.。

Pin Summary

Pin number			Pin Name	Type	GPIO				Function after reset	Alternate Function				
					Input		Output							
SOP28/SSOP28	TSSOP20	SOP20			Weak Pull-up	Ext. Interrupt	O.D.	P.P.		PWM	ADC	I2C	UART	MISC
1	14	6	PB0/TM1OUT/AD0	I/O	✓		✓	✓	PB0		✓			TM1OU
2	15	7	PB1/TCOUT/AD1	I/O	✓		✓	✓	PB1		✓			TCOUT
3	16	8	PB2/AD2	I/O	✓		✓	✓	PB2		✓			
4	17	9	PB3/AD3	I/O	✓		✓	✓	PB3		✓			
5	18	11	PB4/AD4/RX	I/O	✓		✓	✓	PB4		✓		✓	
6	19	10	PB5/AD5/TX	I/O	✓		✓	✓	PB5		✓		✓	
7	-	12	PB6/AD6	I/O	✓		✓	✓	PB6		✓			
8	-	13	PB7/AD7	I/O	✓		✓	✓	PB7		✓			
9	-	-	PC0/AD8/PSDA	I/O	✓		✓	✓	PC0		✓			
10	-	-	PC1/AD9/PSCL	I/O	✓		✓	✓	PC1		✓			
11	-	-	PC2/AD10	I/O	✓		✓	✓	PC2		✓			
12	-	-	PC3/AD11	I/O	✓		✓	✓	PC3		✓			
13	1	1	VSS	P										
14	20	20	VCC	P										
15	2	14	PD0/AD13/XI	I/O	✓		✓	✓	PD0		✓			XI
16	3	15	PD1/AD14/XO	I/O	✓		✓	✓	PD1		✓			XO
17	-	16	PD2/AD15	I/O	✓		✓	✓	PD2		✓			
18	-	17	PD3/AD16	I/O	✓		✓	✓	PD3		✓			
19	4	18	PD4/AD17	I/O	✓		✓	✓	PD4		✓			
20	5	19	PD5/AD18	I/O	✓		✓	✓	PD5		✓			
21	6	-	PD6/AD19/COM1	I/O	✓		✓	✓	PD6		✓			
22	7	-	PD7/AD20/COM0	I/O	✓		✓	✓	PD7		✓			
23	8	2	PA7AD21/PWM0P/PWM5/INT2	I/O	✓	✓	✓	✓	PA7	✓	✓			nRESET
24	9	-	PA4/AD22/PWM4/INT1	I/O	✓	✓	✓	✓	PA4	✓	✓			
25	10	-	PA3/AD23/PWM3	I/O	✓		✓	✓	PA3	✓	✓			
26	11	3	PA1/AD24/PWM1/COM3/I2CSCL/PSCL	I/O	✓		✓	✓	PA1	✓	✓	✓		
27	12	4	PA0/AD25/PWM0N/INT0/COM2/I2CSDA/PSDA	I/O	✓	✓	✓	✓	PA0	✓	✓	✓		
28	13	5	PA2/AD26/TM0CKI/PWM2	I/O	✓		✓	✓	PA2	✓	✓			TM0CKI

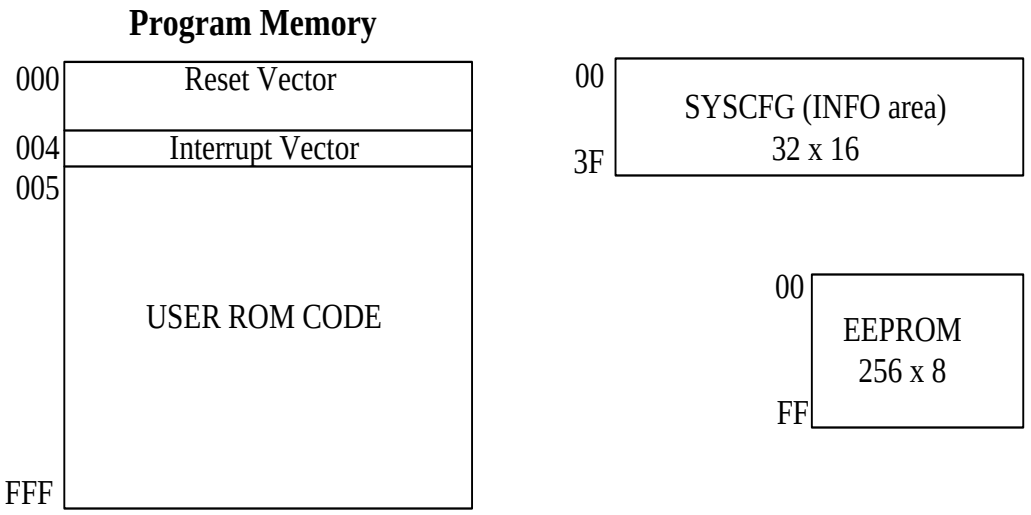
Symbol：O.D. = Open Drain
P.P. = Push-Pull Output

FUNCTION DESCRIPTION

1.CPUCore

1.1 ProgramROM (PROM)

The flash program ROM of this chip is 4K words (4Kx16bit), with an additional 32-word INFO area to store SYSCFG and another additional 256-byte EEPROM. The ROM can be written multiple times and can be read as long as the PROTECT bit of SYSCFG is not set. SYSCFG can be read regardless of whether PROTECT is set or cleared, but the PROTECT bit can only be cleared when erasing the user ROM code area. On the other hand, if the PROTECT bit is set, the writer will not read the user ROM code area, and the user ROM code cannot be updated until the PROTECT bit is cleared.。



106h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
RDCTL	–	–	–	–	–	–	RDCTL	
R/W	–	–	–	–	–	–	R/W	R/W
Reset	–	–	–	–	–	–	0	1

106h.1~0 **RDCTL:** Read signal delay control for Program ROM
00: 4ns delay for read signal of Program ROM
01: 8ns delay for read signal of Program ROM
10: 12ns delay for read signal of Program ROM
11: 16ns delay for read signal of Program ROM
Change this register at slow clock for safety
The user must switch this register to “4ns” to enhance the performance of minimal operating voltage.
This feature can’t be emulated.

1.1.1 Reset Vector (000H)

After reset, system will restart the program counter (PC) at the address 000h, all registers will revert to the default value。

1.1.2 Interrupt Vector (004H)

When an interrupt occurs, the program counter (PC) will be pushed onto the stack and jumps to address 004H。

1.2 System Configuration Register (SYSCFG)

The system configuration register (SYSCFG) is located in the Flash INFO area; it contains two 16-bit registers (CFGWL/CFGWH). SYSCFG determines the options for the CPU initial conditions. Users can select the LVR operating mode and chip operating mode through the SYSCFG register. The 15th bit of CFGWH is the code protection selection bit. If this bit is 1, the data in the PROM will be protected when the user reads the PROM.。

位		15~0	
默认值		00000000000000	
位		Description	
CFGWL (INFO area Address 00)	15~14	Reserved	
CFGWH (INFO area Address 01)	15	**PROTECT**: Code protection selection	
		1	Enable
		0	Disable
	14	**XRSTE**: External Pin (PA7) Reset Enable	
		1	Enable
		0	Disable
	13~10	**LVR**: Low Voltage Reset Mode	
		1111	LV Reset 3.51V
		1110	LV Reset 3.39V
		….	….
		0001	LV Reset 1.75V
		0000	LV Reset 1.63V
	9~8	**WDTE**: WDT Reset Enable	
		11	Always Enable
		10	Enable in FAST/SLOW mode, Disable in IDLE/STOP mode
		0X	Disable
	4	PORSEL: POR duty selection	
		0	POR enables at 100% duty cycle
		1	POR enables at 1/8 duty cycle
	3	ATDOFF: Automatic transient detection	
		0	ATD ON
		1	ATD OFF
	Others	Reserved	

1.3 EEPROM

TM56F6922 contains 256 bytes of EEPROM data memory. It is a separate data space where single bytes can be read and written. Due to the physical characteristics of EEPROM, it requires longer access time than program ROM, at least 30K write/erase cycles。

1.3.1 The Precaution of EEPROM Programming

1.3.1.1 The Programming Characteristics of EEPROM

（1）The EEPROM programming time is not fixed. Programming different data requires different times。

（2）The programming time is affected by voltage, temperature, and whether the data is inverted. The programming time is longer for higher temperatures, lower VCC and larger number of data inversion。

（3）This chip has a built-in EEPROM write time-out function to ensure that the system can execute the program normally when write time-out occurred。

1.3.1.2 EEPROM Write Endurance

The number of times of EEPROM programming is related to voltage and temperature. The write endurance is at least 30,000(VCC = 5V, -40℃~105℃). Please refer to the table of “EEPROM Characteristics” in the chapter of “ELECTRICAL CHARACTERISTICS”。

1.3.1.3 EEPROM Write Validation

Depending on the specific application, it is generally required to read back the data written into EEPROM for verification。

1.3.1.4 Protection against Miswriting

When a write operation is initiated, the following operations can prevent miswriting：

（1）Low voltage detection: when writing EEPROM, VCC must be >3.0V@9.2MHz, you can use the LVD function to monitor the voltage. (LVD monitoring voltage is recommended to be 3.5V to prevent power outage and allow sufficient time for writing EEPROM)

（2）Clear the watchdog (WDT) every time a byte is written to prevent the watchdog from being reset when multiple bytes are written sequentially。

（3）When writing/reading data, temporarily turn off all of the interrupt and resume them after the completion of writing/reading。

（4）Software failure: add EEPROM read-back mechanism to the program to ensure that data is written correctly。

（5）Timeout protection: enable the write timeout function (EEPTE) in the program to protect the system from getting stuck when write time-out occurred。

（6）To reduce power supply glitches: connect capacitors between VCC and GND to stabilize the system power supply。

(7) Must confirm EEPBUSY=0 before writing the next byte

The usage of EEPROM read is the same as using table read instruction, but the EEPROM enable bit must be set high and DPH is always set to 0. The interrupt function needs to be stopped during the process. Writing EEPEN (192h) by writing 0xE2 will set the EEPROM enable bit, writing any other value to EEPEN (192h) will clear the EEPROM enable bit.

◇Example: ReadEEPROMData@Addrss 0x23

```
CLRX        INTIE       ; Turn off INTIE
CLRX        INTIE1      ; Turn off INTIE1
MOVLW       E2h         ;
MOVWX       EEPEN       ; Set the EEPROM enable bit
CLRX        DPH         ; Set DPH=0 for EEPROM write/read
MOVLW       00h
MOVWX       DPH
MOVLW       23h
MOVWX       DPL         ; Set DPTR = 0023h
; Use the opcode tabl to read the EEPROM @Address 23h data into W
TABRL
….
; Another way to read EEPROM data into W
MOVLW       01h         ;
MOVWX       TABR        ; TABR=01h=opcode TABRL
MOVXW       TABR        ; Read EEPROM data to W
…
```

The usage of EEPROM writing is similar to that of reading EEPROM. The interrupt function needs to be stopped during the process. When the S/W writes data to the register EEPDT (193h), the data will also be written to the EEPROM. When the power supply VCC < 3V, the S/W must set CPUPSC=2 and lower the system clock to avoid EEPROM write errors. In addition, the S/W must clear the WDT timer before writing EEPROM.

◇Example: Write EEPROM data A5 to address 23h

```
CLRX        INTIE       ; Turn off INTIE
CLRX        INTIE1      ; Turn off INTIE1
CLRWDT
MOVLW       E2h         ;
MOVWX       EEPEN       ; Set EEPROM enable bit
CLRX        DPH         ; Set DPH=0 for EEPROM write/read
MOVLW       23h
MOVWX       DPL         ; Set DPTR = 0023h (set EEPROM address)
MOVLW       00000010b
MOVWX       EEPCTL      ; Set EEPROM write with 10mS time out
MOVLW       A5h
MOVWX       EEPDT       ; Write data A5h EEPDT (193h)
                        ; Data is also saved to EEPROM @ address 23h
NOP
NOP                     ; Wait EEPBUSY goes High
BTXSC       EEPBUSY     ; Check EEPBUSY
LGOTO       $ -1        ; Wait EEPBUSY goes low
BTXSC       EEPTO       ; Check EEPROM write timeout flag
LGOTO       TIMEOUT
MOVXW       CLKCTL
ANDLW       FCh
IORLW       02h         ; If VCC is lower than 3V, S/W must set CPUPSC=2
MOVXW       CLKCTL      ; Before writing to EEPROM, Fsys=FIRC/2
MOVLW       35h         ; Write next EEPROM address 0x35
MOVWX       DPL         ; Set DPTR = 0035h
LCALL       WAIT100us   ; If VCC drops and approaches LVR, you need to wait
                        ;100us before continuing to write down the next
                        ; EEPROM address
```

```
MOVLW     A5h
MOVWX     EEPDT     ; Write data A5h EEPDT (193h)
                    ; Data is also saved to EEPROM @ address 35h
CLRX      EEPEN     ; Protect EEPROM from abnormal writing
```

191h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
EEPCTL	EEPTO	EEPBUSY	–	–	–	–	EEPTE	
R/W	R	R	–	–	–	–	R/W	R/W
Reset	0	0	–	–	–	–	0	0

191h.7 **EEPTO:** EEPROM Write Time-Out flag
191h.6 **EEPBUSY:** EEP Busy Flag, EEP Busy Flag is set high when write EEP
191h.1~0 **EEPTE:** Enable EEPROM write timeout (access wait time)
00:Disable 01: 2.5ms 10:10ms 11:20ms

192h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
EEPEN	EEPEN							
R/W	W							
Reset	0							

192h.7~0 **EEPEN**: EEPROM Access Enable
write 0xE2 to this register will enable EEPROM access
write others value to this register will disable EEPROM access

193h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
EEPDT	EEPDT							
R/W	W							
Reset	0							

193h.7~0 **EEPDT**: EEPROM Data to write
write data to this register will let H/W write the data to EEPROM when EEPROM access is enable
After write EEPDT, must wat 2 NOP

107h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
LVRPD	LVRPD						PORPDF	LVRPDF
R/W	W						R/W	R/W
Reset	0							

107.7~0 **LVRPD:** LVR/POR power down register
Write 0x37 to force LVR+POR be disabled (when read PORPDF=LVRPDF=1).
Write 0x38 to force LVR be disabled, POR still enable (when read PORPDF=0, LVRPDF=1).
Write 0x39 to force POR be disabled, LVR still enable (when read PORPDF=1, LVRPDF=0)
Write other value to enable LVR+POR (when read PORPDF=LVRPDF=0)

18Ch	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TABR	TABR							
R/W	R/W							
Reset	0							

18C.7~0 1. TABR write 01h = opcode TABRL
2. TABR write 02h = opcode TABRH
3. After setp1 or step2, read TABR to get main ROM table read value
After step1, read TABR to get EEPROM value (When EEPEN=E2h)
Table Read for ASM: instruction TABRL/TABRH or register TABR
Table Read for C : using register TABR ***and it is forbidden to use in interrupt***

1.4 RAM Addressing Mode

TM56F6922 has 336 bytes of data memory space and an additional 256 bytes of data memory in the CPU. The 336 bytes of memory are divided into four banks of memory planes. Each bank extends to 7Fh (128 bytes). The lower position of each bank is reserved for (SFR). Above the SFR are general purpose registers implemented as static RAM. All implemented banks contain special function registers. Some frequently used special function registers can be mirrored from one bank to another for reduced code and faster access.

- The memory is 592 bytes in total, divided into BANK0~5。
- BANK0~BANK3 can be addressed by value or indirectly by **FSR** and **INDF**。
- BANK4~BANK5 need to be indirectly addressed through **FSR1** and **INDF1**。

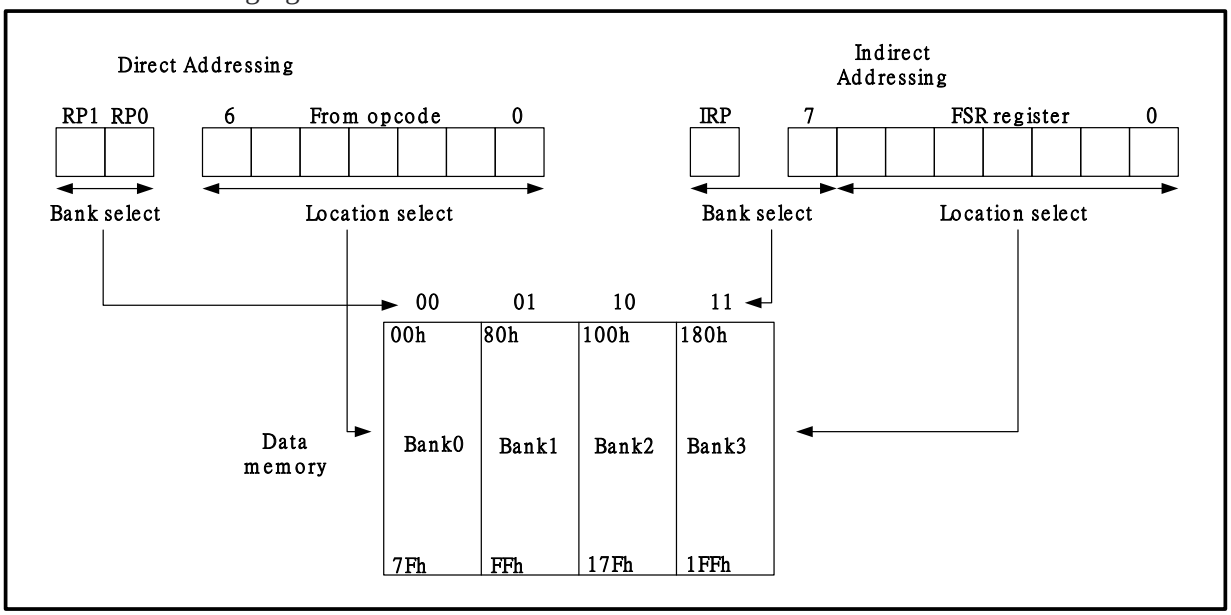
1.4.1 BANK0/BANK1/BANK2/BANK3

Bit RP1 and RP0 (STATUS[6:5]) BANK 0/1/2/3 selection bits

RP1 : RP0	BANK
00	0
01	1
10	2
11	3

The 336-byte memory plane can be addressed directly or indirectly for bank0/1/2/3. The INDFx registers are not physical registers. Addressing an INDFx register will result in indirect addressing. Indirect addressing is achieved by using the INDFx registers. Any instruction using the INDFx register actually accesses the register pointed to by the File Select Register (FSR). Indirectly reading the INDF register itself (FSR = '0') will read 00h. Writing to the INDFx register indirectly results in a no-op (possibly the status bits will be affected). BANK0/1/2/3 addressing is achieved by concatenating the 8-bit FSR register and the IRP bit (STATUS [7]) to obtain an effective 9-bit address.

Refer to the following figure.



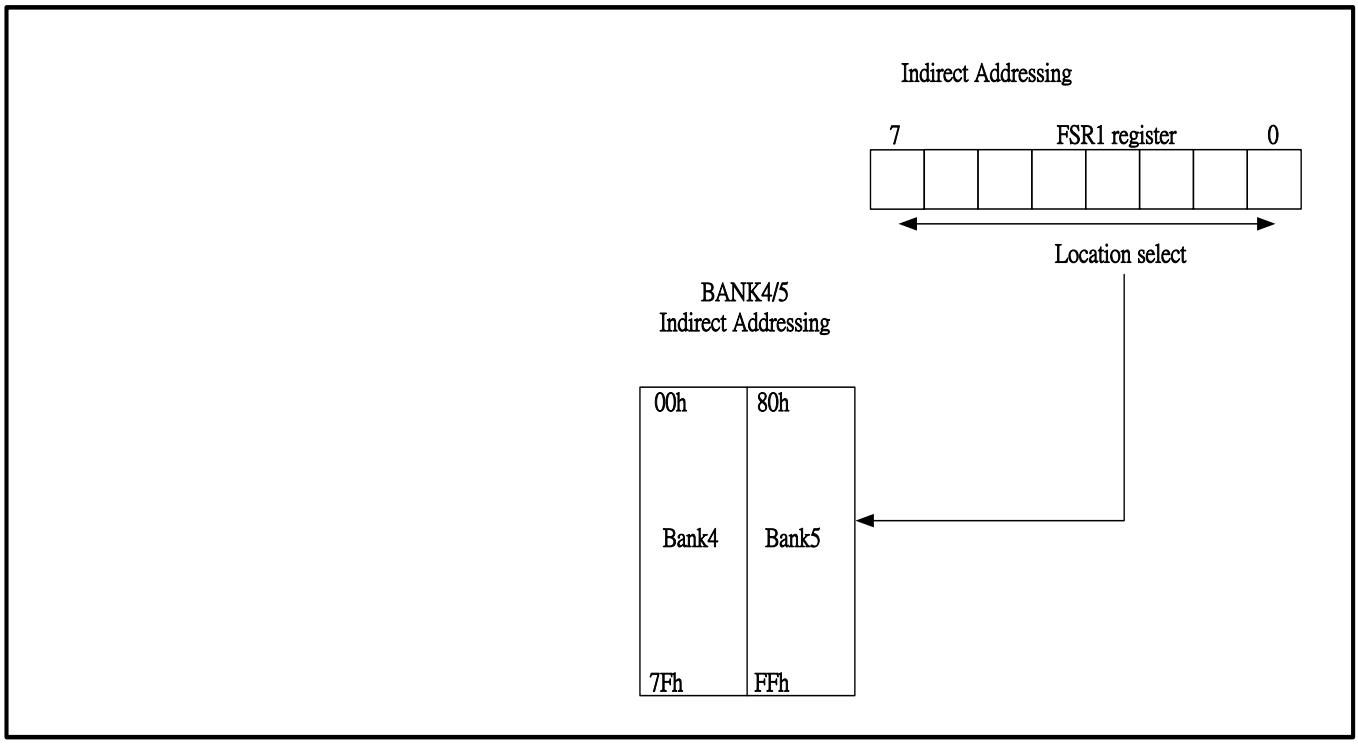
BANK 0/1/2/3 direct/indirect addressing

Keep RP0=RP1=0 at the beginning of the S/W code and use the new instruction set.
The advantage of using the new instruction is that the user can ignore the BANK position of the register and can save code size. The new instruction is almost the same as the old instruction. By replacing "F" with "X" in the instruction set, the new instruction can be easily used without switching banks.
Example:

BC**F**	TM0IE	➔	BC**X**	TM0IE
DEC**F**	CNT, 1	➔	DEC**X**	CNT,1
INC**F**SZ	RAM25, 0	➔	INC**X**SZ	RAM25, 0
MOVW**F**	PAMODL	➔	MOVW**X**	PAMODL
RL**F**	RAMA0, 0	➔	RL**X**	RAMA0, 0
SWAP**F**	ADCTL, 0	➔	SWAP**X**	ADCTL, 0

1.4.2 BANK4/BANK5

There is an additional 256-byte RAM space (BANK 4/5) that can only be accessed by using indirect addressing (using INDF1 and FSR1). INDF1 is used to select BANK 4/5 and FSR1 is used as an address pointer. For BANK4, the memory addresses are from 00H to 7FH and for BANK5, the memory addresses are from 80H to FFH



Indirect addressing of BANK 4/5

	BANK0 00~7Fh		BANK1 80h~FFh		BANK2 100h~17Fh		BANK3 180h~1FFh		BANK4 00~7Fh		BANK5 80~FFh
00h	INDF0	80h	INDF0	100h	INDF0	180h	INDF0				
01h	TM0	81h	OPTION	101h	TM0	181h	OPTION				
02h	PCL	82h	PCL	102h	PCL	182h	PCL				
03h	STATUS	83h	STATUS	103h	STATUS	183h	STATUS				
04h	FSR	84h	FSR	104h	FSR	184h	FSR				
05h	PAD	85h	PAMODH	105h	TESTREG	185h	DPL				
06h	PBD	86h	PAMODL	106h	RDCTL	186h	DPH				
07h	PCD	87h	PBMODH	107h	LVRPD	187h	UARTCTL				
08h	PDD	88h	PBMODL	108h	LOE	188h	UARTSTA				
09h	INDF1	89h	INDF1	109h	INDF1	189h	INDF1				
0Ah	PCLATH	8Ah	PCLATH	10Ah	PCLATH	18Ah	PCLATH				
0Bh	INTIE	8Bh	INTIE	10Bh	INTIE	18Bh	INTIE				
0Ch	INTIF	8Ch	PCMODL	10Ch	PCH	18Ch	TABR				
0Dh	FSR1	8Dh	PDMODH	10Dh	UBAUD	18Dh	URDATA				
0Eh	INTIE1	8Eh	PDMODL	10Eh	BGTRIM	18Eh	CRCDL				
0Fh	CLKCTL	8Fh	OPTION2	10Fh	IRCF	18Fh	CRCDH				
10h	TM0RLD	90h	PWMOE	110h	I2CTXD0	190H	CRCIN				
11h	TM0CTL	91h	PWMCTL	111h	I2CTXD1	191h	EEPCTL				
12h	TM1	92h	PWMPRDH	112h	I2CCTL	192h	EEPEN				
13h	TM1RLD	93h	PWMPRDL	113h	I2CFLG	193h	EEPDT				
14h	TM1CTL	94h	PWM0DH	114h	I2CRCD0	194h					
15h	T2CTL	95h	PWM0DL	115h	I2CRCD1	195h					
16h	LVCTL	96h	PWM1DH	116h		196h					
17h	ADCDH	97h	PWM1DL	117h		197h					
18h	ADCTL	98h	PWM2DH	118h		198h					
19h	ADCTL2	99h	PWM2DL	119h		199h					
1Ah	INTIF1	9Ah	PWM3DH	11Ah		19Ah					
1Bh	IOCCTRL	9Bh	PWM3DL	11Bh		19Bh					
1Ch	PAWKE	9Ch	PWM4DH	11Ch		19Ch					
1Dh	PBWKE	9Dh	PWM4DL	11Dh		19Dh					
1Eh	PCWKE	9Eh	PWM5DH	11Eh		19Eh	VTEMPDL				
1Fh	PDWKE	9Fh	PWM5DL	11Fh		19Fh	VTEMPDH				
20h ~ 6Fh	General Purpose SRAM 80 Bytes	A0h ~ EFh	General Purpose SRAM 80 Bytes	120h ~ 16Fh	General Purpose SRAM 80 Bytes	1A0h ~ 1EFh	General Purpose SRAM 80 Bytes				
70h ~ 7Fh	Common Area 16 Bytes	F0h ~ FFh	accesses 70h~7Fh	170h ~ 17Fh	accesses 70h~7Fh	1F0h ~ 1FFh	accesses 70h~7Fh	7Fh	Indirect Addressing SRAM 128 Bytes BANK4	FFh	Indirect Addressing SRAM 128 Bytes BANK5

◇Example: Use direct addressing (RP0=RP1=0) to read/write registers in bank0/1/2/3

```
TM1             equ     12h     ;SFR in Bank0
PWM1DH          equ     96h     ;SFR in Bank1
I2CTXD0         equ     110h    ;SFR in Bank2
RAM20           equ     20h     ;RAM in Bank0
RAMA0           equ     A0h     ;RAM in Bank1
RAM120          equ     120h    ;RAM in Bank2
RAM1A0          equ     1A0h    ;RAM in Bank3

MOVXW   TM1             ; Read TM1 (Bank0) to W
MOVXW   PWM1DH          ; Read PWM1PRD (Bank1) to W
MOVXW   I2CTXD0         ; Read I2CTXD0 (Bank2) to W


MOVLW   16h
MOVWX   RAM20           ; W=16h write to RAM[0x20]
MOVWX   RAMA0           ; W=16h write to RAM[0xA0]
MOVWX   RAM120          ; W=16h write to RAM[0x120]
MOVWX   RAM1A0          ; W=16h write to RAM[0x1A0]
.
MOVLW   037H            ; W=37H
MOVWX    LVRPD          ; LVRPD  = W = 037H , force LVR+POR disable
```

◇Example: read/write register by using indirect addressing (RP0=RP1=0, IRP=1) for BANK 0/1/2/3

```
BSX     IRP             ; IRP=1  =>Bank2/3
MOVLW   10h             ; W=10H
MOVWX   FSR             ; FSR = W =10H
MOVXW   INDF            ; Read SFR I2CTXD0 (110h) to W
.
BSX     IRP             ; IRP=1 =>Bank2/3
MOVLW   10H             ; W=10H
MOVWX   FSR             ; FSR = W =10H
MOVLW   37H             ; W=37H
MOVWX    INDF           ; I2CTXD0 (110H) = W = 037H
```

◇Example: read/write register by using indirect addressing (RP0=RP1=0, IRP=0) for BANK 0/1/2/3

```
BCX     IRP             ; IRP=0  =>Bank0/1
MOVLW   10h             ; W=10H
MOVWX   FSR             ; FSR = W =10H
MOVXW   INDF            ; Read SFR TM0RLD (10h) to W
.
BCX     IRP             ; IRP=0 =>Bank0/1
MOVLW   10H             ; W=10H
MOVWX   FSR             ; FSR = W =10H
MOVLW   37H             ; W=37H
MOVWX    INDF           ; TM0RLD (10H) = W = 037H
```

◇Example: read/write register by using indirect addressing for BANK 4/5

```
; Write data to BANK4 memory
MOVLW   20h             ; W=20H
MOVWX   FSR1            ; FSR1 = W =20H
MOVLW   5Ah             ; W=5AH
```

```
MOVWX   INDF1           ; write data 5AH to BANK4 memory   @20h

; Write data to BANK5memory
MOVLW   A7h             ; W=A0H
MOVWX   FSR1            ; FSR1 = W =A7H
MOVLW   69h             ; W=69H
MOVWX   INDF1           ; write data 69H to BANK5 memory   @A7h

; Read data from BANK4 memory
MOVLW   20h             ; W=20H
MOVWX   FSR1            ; FSR1 = W =20H
MOVXW   INDF1           ; read memory BANK4 data @20h to W

; Read data from BANK5 memory
MOVLW   A0h             ; W=A0H
MOVWX   FSR1            ; FSR1 = W =A0H
MOVXW   INDF1           ; read memory BANK5 data @A0h to W
```

0h/80h/100h/180h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INDF	INDF							
R/W	R/W							
Reset	0							

INDF : addressing INDF actually point to the register whose address is contained in the FSR register

04h/84h/104h/184h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
FSR	FSR							
R/W	R/W							
Reset	0							

FSR : **File Select Register**, indirect address mode pointer

09h/89h/109h/189h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INDF1	INDF1							
R/W	R/W							
Reset	0							

INDF1 : addressing INDF1 actually point to the register whose address is contained in the FSR1 register
Used for BANK4/BANK5 memory only

0Dh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
FSR1	FSR1							
R/W	R/W							
Reset	0							

FSR1 : **File Select Register1**, indirect address mode pointer
Used for BANK4/BANK5 memory only

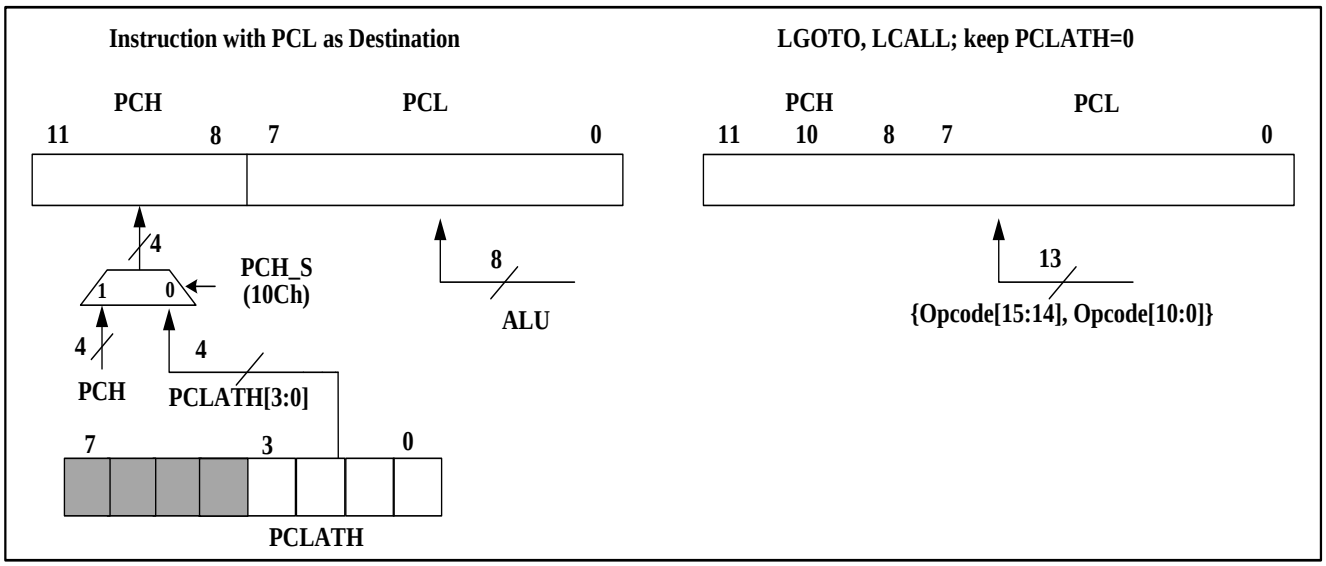
1.5 Programming Counter (PC) and Stack

The program counter is 12 bits wide and is capable of addressing a 4K x 16 Flash ROM. The lower byte comes from the PCL register, which is a readable and writable register. The upper bits (PC[11:8]) are not readable but can be written indirectly through the PCLATH register, or left unchanged when PCH_S is set to "1" (by writing 1Ch to the PCH_S register). On any reset, the upper bits of the PC will be cleared. When a program instruction is executed, the PC will contain the address of the next program instruction to be executed. The PC value is normally incremented by 1, except in the following cases. The reset vector (000h) and interrupt vector (004h) are used for PC initialization and interrupts. For LCALL/LGOTO instructions, the PC is loaded with the 12-bit address from the instruction word. For RET/RETI/RETLW instructions, the PC retrieves its contents from the top stack.

When PCH_S is cleared to '0' (by writing 0h to the PCH_S register), executing any instruction targeting the PCL register simultaneously causes the Program Counter PC[11:8] bits (PCH) to be replaced by the contents of the PCLATH register. This allows the entire contents of the Program Counter to be changed by writing the desired upper 4 bits to the PCLATH register. When the lower 8 bits are written to the PCL register, all 12 bits of the Program Counter will become the value contained in the PCLATH register and the value being written to the PCL register.

When PCH_S is set to '1', executing any instruction targeting the PCL register, the lower 8 bits are written to the PCL register without changing the program counter PC[11:8] bits (PCH).

For assembly code, set PCH_S to "1" at the beginning of S/W code, and use the new instruction LGOTO/LCALL (HI-TECH C does not support encoding). The advantage of using the new instruction is that the user can ignore setting the PCLATH register, and the system will automatically read INST[15:14] directly from the 8K ROM. The new instruction can be easily used without setting PCLATH.



10Ch	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PCH_S	PCH_S							
R/W	R/W							
Reset	0	0	0	0	0	0	0	0

10Ch PCH_S: Program Counter Upper bits selection when instruction with PCL as destination is executed
write 0x1C to set PCH_S= 1: PCH keep the original value
write others to clear PCH_S=0: PCH is from PCLATH[3:0]

81h/181h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
OPTION	HWAUTO	INT0EDG	INT1EGE	FREQL	WDTPSC		WKTPSC	
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	1	1	1	1	1

81h.7 **HWAUTO:** Enter/Exit interrupt subroutine, hardware automatic save/restore WREG, FSR, FSR 1, TABR, PCLATH, DPL, DPH and STATUS (excluding TO, PD);
Only work for ASM, not for C
0: disable
1: enable; (only work for ASM, not for C)

The STACK is 12 bits wide and 8 levels deep. LCALL instructions and hardware interrupts push STACK levels in sequence. RET/RETI/RETLW instructions pop stack levels in sequence.

For table lookup, the device offer the powerful table read instructions TABRL, TABRH to return the 16-bit ROM data into W register by setting DPTR={DPH, DPL} registers. It also offers another way to read the 16-bit ROM data into W register by setting TABR (18Ch) for C language.

◇Example: To look up the PROM data located “TABLE1” and “TABLE2”

```
        ORG         00h             ; Reset Vector
        LGOTO       START
        ORG         04h             ; Interrupt Entry Address
        MOVWX       W_TMP           ; if HWAUTO =1, can omit this line
        MOVXW       STATUS          ; if HWAUTO =1, can omit this line
        MOVWX       STATUS_TMP      ; if HWAUTO =1, can omit this line
        MOVXW       PCLATH
        MOVWX       PCLATH_TMP

        BTXSC       TM0IF
        LCALL       TM0INT_TASK
        …….
        MOVXW       PCLATH_TMP
        MOVWX       PCLATH
        MOVXW       STATUS_TMP      ; if HWAUTO =1, can omit this line
        MOVWX       STATUS          ; if HWAUTO =1, can omit this line
        SWAPX       W_TMP, f        ; if HWAUTO =1, can omit this line
        SWAPX       W_TMP           ; if HWAUTO =1, can omit this line
        MOVXW       W_TMP           ; if HWAUTO =1, can omit this line
        RETI

START:
        MOVLW       00h
        MOVWX       INDEX           ; Set lookup table’s address
        MOVLW       1Ch
        MOVWX       PCH_S
        MOVLW       00h
        MOVWX       RAM20           ; RAM20 as PCL counter
LOOP:
        MOVXW       RAM20
        LCALL       TABLE1          ; To lookup data, W=55h
        ……
        INCX        RAM20, 1
        ……
```

```
        LGOTO       LOOP            ; Go to LOOP label
        ……
        MOVLW       (TABLE2>>8) & 0xff
        MOVWX       DPH             ; DPH register (F186.4~0)
        MOVLW       (TABLE2) & 0xff
        MOVWX       DPL             ; DPL register (F185.7~0)
; Table Read by opcode TABRL / TABRH
        TABRL                       ; read PROM low byte data to W (W=86h)
        TABRH                       ; read PROM high byte data to W (W=19h)
        ……
; Table Read by SFR TABR
        MOVLW       01h             ; TABR=01h= opcode TABRL
        MOVWX       TABR
        MOVXW       TABR            ; read PROM low byte data to W (W=86h)

        MOVLW       02h             ; TABR=02h=opcode TABRH
        MOVWX       TABR
        MOVXW       TABR            ; read PROM high byte data to W (W=19h)

        ORG         F68h
TABLE1:
        ADDWX       PCL, 1          ; Add the W with PCL, the result back in PCL
        RETLW       55h             ; W=55h when return
        RETLW       56h             ; W=56h when return
        RETLW       57h             ; W=57h when return
        RETLW       58h             ; W=58h when return

        ……
        ORG         368h
TABLE2:
        .DT 0x1986, 0x3719, 0x2983  ; 16-bit ROM data
        ….
        ORG         100h
TM0INT_TASK:
        MOVLW       11101111B
        MOVWX       INTIF
        …….
        RET
```

1.6 ALU and Working (W) Register

The ALU is 8-bit wide and capable of addition, subtraction, shift and logical operations. In two-operand instructions, typically one operand is the W register, which is an 8-bit non-addressable register used for ALU operations. The other operand is either a file register or an immediate constant. In single operand instructions, the operand is either W register or a file register. Depending on the instruction executed, the ALU may affect the values of Carry (C), Digit Carry (DC), and Zero (Z) Flags in the STATUS register. The C and DC flags operate as a /Borrow and /Digit Borrow, respectively, in subtraction.

Note: /Borrow represents inverted of Borrow register.
/Digit Borrow represents inverted of Digit Borrow register

1.7 STATUS Register (03H/83H/103H/183H)

This register contains the arithmetic status of ALU and the Reset status. The STATUS register can be the destination for any instruction, as with any other register. If the STATUS register is the destination for an instruction that affects the Z, DC or C bits, then the write to these three bits is disabled. These bits are set or cleared according to the device logic. It is recommended, therefore, that only BCX, BSX and MOVWX instructions are used to alter the STATUS Register because these instructions do not affect those bits.

<table>
<tr><th>STATUS</th><td>Bit 7</td><td>Bit 6</td><td>Bit 5</td><td>Bit 4</td><td>Bit 3</td><td>Bit 2</td><td>Bit 1</td><td>Bit 0</td></tr>
<tr><th>Reset value</th><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr>
<tr><th>R/W</th><td>R/W</td><td>R/W</td><td>R/W</td><td>R</td><td>R</td><td>R/W</td><td>R/W</td><td>R/W</td></tr>
<tr><td>Bit</td><td colspan="8">描述</td></tr>
<tr><td>7</td><td colspan="8">IRP: Register Bank Select bit (used for indirect addressing)
0 = Bank 0,1 (00h - FFh)
1 = Bank 2,3 (100h - 1FFh)</td></tr>
<tr><td>6:5</td><td colspan="8">RP1:RP0: Register Bank Select bits (used for direct addressing)
00 = Bank 0 (00h - 7Fh)
01 = Bank 1 (80h - FFh)
10 = Bank 2 (100h - 17Fh)
11 = Bank 3 (180h - 1FFh)
Each bank is 128 bytes</td></tr>
<tr><td>4</td><td colspan="8">TO: Time Out Flag
0: after Power On Reset, LVR Reset or CLRWDT/SLEEP instruction
1: WDT time out occurs</td></tr>
<tr><td>3</td><td colspan="8">PD: Power Down Flag
0: after Power On Reset, LVR Reset or CLRWDT instruction
1: after SLEEP instruction</td></tr>
<tr><td>2</td><td colspan="8">Z: Zero Flag
0: the result of a logic operation is not zero
1: the result of a logic operation is zero</td></tr>
<tr><td rowspan="3">1</td><td colspan="8">DC: Decimal Carry Flag or Decimal / Borrow Flag</td></tr>
<tr><td colspan="4">ADD 指令</td><td colspan="4">SUB 指令</td></tr>
<tr><td colspan="4">0: no carry
1: a carry from the low nibble bits of the result occurs</td><td colspan="4">0: a borrow from the low nibble bits of the result occurs
1: no borrow</td></tr>
<tr><td rowspan="3">0</td><td colspan="8">C: Carry Flag or /Borrow Flag</td></tr>
<tr><td colspan="4">ADD instruction</td><td colspan="4">SUB instruction</td></tr>
<tr><td colspan="4">0: no carry
1: a carry occurs from the MSB</td><td colspan="4">0: a borrow occurs from the MSB
1: no borrow</td></tr>
</table>

◇Example：Write immediate data into STATUS register

```
MOVLW   00h
MOVWX   STATUS      ; Clear STATUS register
```

◇Example：Bit addressing set and clear STATUS register.

```
BSX     STATUS, 0   ; Set C=1.
BCX     STATUS, 0   ; Clear C=0.
```

◇Example：Determine the C flag by BTXSS instruction.

```
BTXSS   STATUS, 0   ;  Check the carry flag
LGOTO   LABEL_1     ; If C=0, goto label_1
LGOTO   LABEL_2     ; If C=1, goto label_2
```

2. Reset

This device can be RESET in four ways。

- Power-On-Reset (POR)
- Low Voltage Reset (LVR)
- External Pin Reset (XRST)
- Watchdog Reset (WDTR)

Resets can be caused by Power on Reset (POR), External Pin Reset (XRST), Watchdog Timer Reset (WDTR) , or Low Voltage Reset (LVR) . The CFGWH controls the Reset functionality. After Reset, the SFRs are returned to their default value, the program counter (PC) is cleared, and the system starts running from the reset vector 000H place. The TO and PD flags at status register (STATUS) are indicate system reset status.

2.1 Power on Reset (POR)

After Power-On-Reset, all system and peripheral control registers are then set to their default hardware Reset values. The clock source, LVR level and chip operation mode are selected by the SYSCFG register value.

2.2 Low Voltage Reset (LVR)

The function of low voltage reset is to perform static reset when the power supply voltage is below the threshold level. 16 threshold levels can be selected. The operation mode of LVR is defined by the CFGWH register. SFR LVDS can select 16 levels of LVD (Low Voltage Detection).

SYSCFG 01h	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
CFGWH	PROT	XRSTE	LVRE				WDTE	

CFGWH.13~10 LVRE: Low Voltage Reset function select

0000: Set LVR at 1.63V
0001: Set LVR at 1.75V
0010: Set LVR at 1.89V
0011: Set LVR at 2.01V
0100: Set LVR at 2.15V
0101: Set LVR at 2.27V
0110: Set LVR at 2.39V
0111: Set LVR at 2.51V
1000: Set LVR at 2.64V
1001: Set LVR at 2.75V
1010: Set LVR at 2.88V
1011: Set LVR at 3.01V
1100: Set LVR at 3.14V
1101: Set LVR at 3.27V
1110: Set LVR at 3.39V
1111: Set LVR at 3.51V

16h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
LVCTL	LVDF	BGEN	LVRSAV	LVDSAV	LVDS			
R/W	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	-	1	1	1	0	0	0	1

16h.7 **LVDF:** Low Voltage Detect output Flag

16h.6 **BGEN:** Band Gap BG 1.18V enable
0: Disable,
1: Enable and Auto disable in STOP/IDLE mode

16h.5 **LVRSAV:** when set to 1, LVR auto power off in STOP/IDLE mode

16h.4 **LVDSAV:** when set to 1, LVD auto power off in STOP/IDLE mode

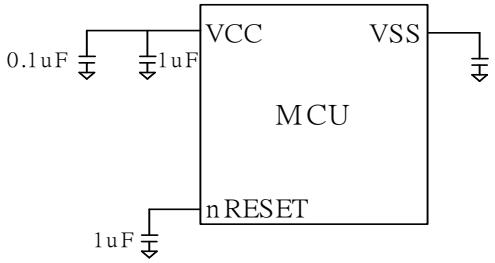
16h.3~0 **LVDS:** Low Voltage Detect select
0000: Disable LVD
0001: Set LVD at 1.75V
0010: Set LVD at 1.89V
0011: Set LVD at 2.01V
0100: Set LVD at 2.15V
0101: Set LVD at 2.27V
0110: Set LVD at 2.39V
0111: Set LVD at 2.51V
1000: Set LVD at 2.64V
1001: Set LVD at 2.75V
1010: Set LVD at 2.88V
1011: Set LVD at 3.01V
1100: Set LVD at 3.14V
1101: Set LVD at 3.27V
1110: Set LVD at 3.39V
1111: Set LVD at 3.51V

Different Fsys have different system minimum operating voltage, reference to Operating Voltage of DC characteristics, if current system voltage is low than minimum operating voltage and lower LVR is selected, then the system maybe enters dead-band and error occurs.

2.3 External Pin Reset (XRST)

The External Pin Reset can be disabled or enabled by the SYSCFG register (XRSTE). It needs to keep at least 2 SIRC clock cycle long to be seen by the chip. External Pin Reset also set all the control registers to their default reset value. The TO/PD flags are not affected by these resets.

External reset pin is low level active. The system is running when reset pin is high level voltage input. The reset pin receives the low voltage and the system is reset. The external reset can reset the system during power on duration and good external reset circuit can protect the system to avoid operating at inappropriate power condition.



SYSCFG 01h	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
CFGWH	PROT	XRSTE	LVRE				WDTE	

CFGWH.14 **XRSTE:** External Pin Reset control
0: Disable External Pin Reset
1: Enable External Pin Reset

2.4 Watchdog Timer Reset (WDTR)

WDT overflow Reset can be disabled or enabled by the SYSCFG register. It runs in Fast/Slow mode and runs or stops in IDLE/STOP mode. WDT overflow speed can be defined by WDTPSC SFR. WDT is cleared by device Reset or CLRWDT SFR bit WDT overflow Reset also set all the control registers to their default reset value. The TO/PD flags are not affected by these resets

SYSCFG 01h	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
CFGWH	PROT	XRSTE	LVRE				WDTE	

CFGWH.9~8 **WDTE:** WDT overflow flow Reset control
0x: Disable WDT Reset
10: Enable WDT Reset in Fast/Slow Mode, Disable in IDLE/STOP Mode
11: Always Enable WDT Reset

81h/181h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
OPTION	HWAUTO	INT0EDG	INT1EGE	FREQL	WDTPSC		WKTPSC	
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	1	1	1	1	1

81h.4 **FREQL:** SIRC frequency selection
0: 60Khz
1: 50Khz

81h.3~2 **WDTPSC:** WDT 周期(@VCC=5V)
00: 164ms WDT overflow rate
01: 328ms WDT overflow rate
10: 655ms WDT overflow rate
11: 1311ms WDT overflow rate

◇Example: Defining Reset Vector

```
        ORG     000H
        LGOTO   START       ; Jump to user program address

        ORG     010H

START:
        …                   ; 010H, The head of user program
        …
        LGOTO   START
```

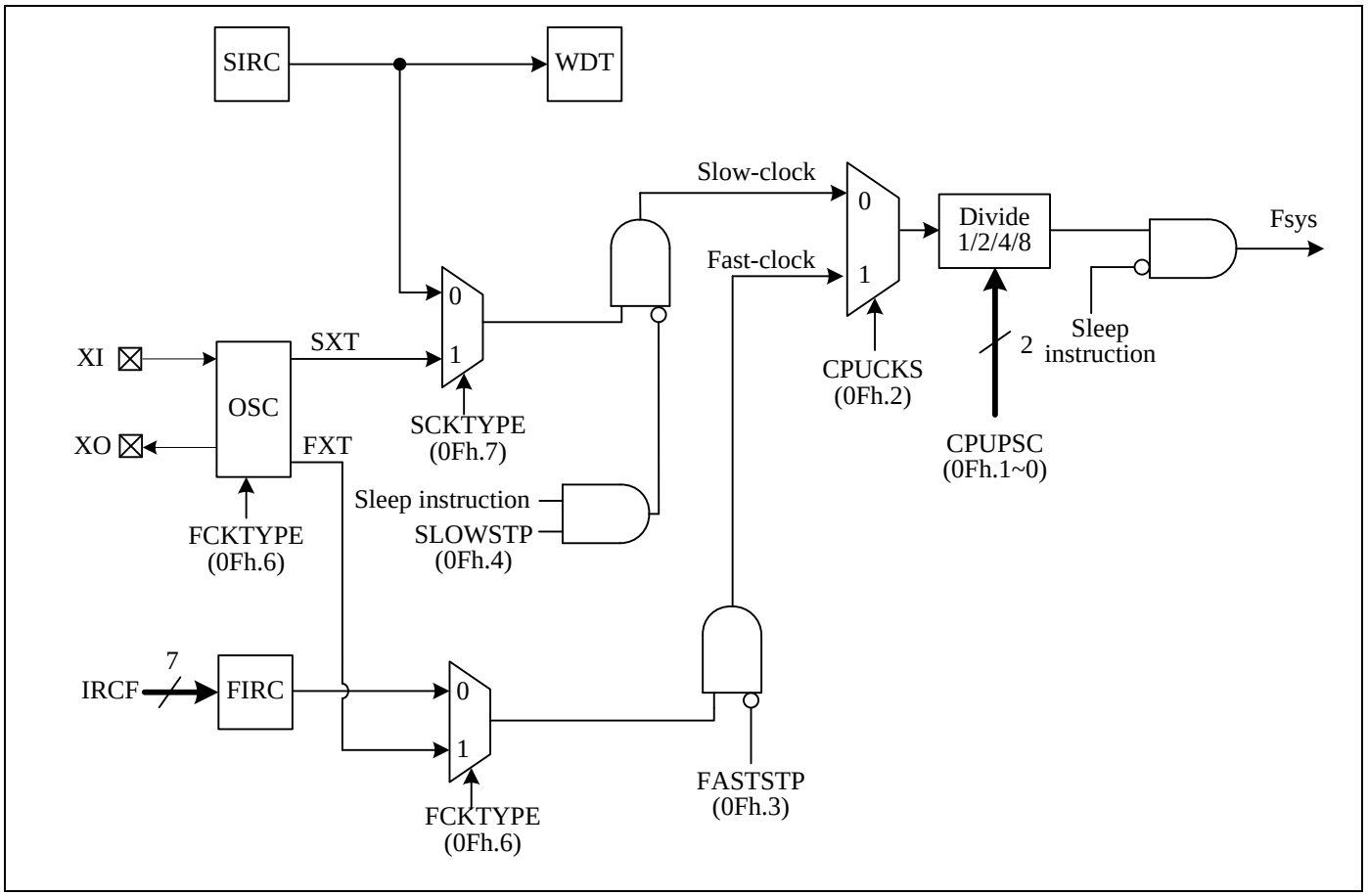
3. Clock Circuitry and Operation Mode

3.1 System Clock

The device is designed with dual-clock system. There are two kinds of clock source, i.e. Slow clock or fast clock. The Slow clock can be selected as SIRC (Slow Internal RC) or SXT (Slow Crystal, 32KHz). The Fast clock can be selected as FIRC (Fast Internal RC) or FXT (Fast Crystal 18MHz). Each clock source can be applied to CPU kernel as system clock. When in IDLE mode, only Slow-clock can be configured to keep oscillating to provide clock source to T2 block. Refer to the figure below.

After Reset, the device is running at Slow mode with 50 KHz SIRC. 50 KHz SIRC can be switch to 60 KHz by clear FREQL (81h.4). S/W should select the proper clock rate for chip operation safety. The higher V_{CC} allows the chip to run at a higher System clock frequency. In a typical condition, an 18.432 MHz System clock rate requires V_{CC} > 2.8V

The CLKCTL SFR controls the System clock operating. H/W automatically blocks the S/W abnormally setting for this register. S/W can only change the Slow-clock type in Fast mode and change the Fast-clock type in Slow mode. Never to write both FASTSTP=1 & CPUCKS=1. It is recommended to write this SFR bit by bit.



Clock Scheme Block Diagram

The frequency of FIRC (internal RC fast clock) can be adjusted through IRCF. When IRCF=00h, the frequency is the lowest. When IRCF=7Fh, the frequency is the highest. Because each IC may have a different IRCF default value, we can ensure the frequency of FIRC=18.432MHz after power-on reset.

0Fh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
CLKCTL	SCKTYPE	FCKTYPE	–	SLOWSTP	FASTSTP	CPUCKS	CPUPSC	
R/W	R/W	R/W	–	R/W	R/W	R/W	R/W	R/W
Reset	0	0	–	0	1	0	1	1

0Fh.7 **SCKTYPE:** Slow clock type. This bit can be changed only in Fast mode(CPUCKS=1)
0: SIRC
1: SXT, PD0 and PD1 are crystal pins

0Fh.6 **FCKTYPE:** Fast clock type. This bit can be changed only in Slow mode(CPUCKS=0)
0: FIRC
1: FXT, PD0 and PD1 are crystal pins， oscillator gain is high for FXT

0Fh.4 **SLOWSTP**: Stop Slow-clock in Stop Mode
0: Not stop slow clock after the SLEEP instruction.
1: Stop slow clock after the SLEEP instruction.

0Fh.3 **FASTSTP**: Stop Fast Clock
0: Fast Clock Running
1: Fast Clock Stop

0Fh.2 **CPUCKS**: System Clock selection
0: Slow Clock as system clock
1: Fast Clock as system clock

0Fh.1~0 **CPUPSC**: System clock Prescaler. System clock
00: divided by 8 01: divided by 4
10: divided by 2 11: divided by 1

FAST Mode:
In this mode, the program will be executed using FIRC or FXT as the CPU clock (Fsys). Timer0, Timer1 blocks are driven by the fast clock. PWM0 block can be driven by Fsys, FIRC (18.4 MHz) or FIRC*2 (36.8 MHz) by setting PWMCKS (91h.5~4). T2 block can be driven by slow clock, Fsys/128 by setting T2CKS (15h.2).

SLOW Mode:
After power-on or reset, device enters SLOW mode, the default Slow-clock is SIRC. In this mode, the Fast-clock can stopped (by FASTSTP=1, for power saving) or running (by FASTSTP=0), and Slow-clock is enabled. All peripheral blocks (Timer0, Timer1etc...) clock sources are Slow-clock in the SLOW mode.

IDLE Mode:
If Slow-clock is enabled (SLOWSTP=0) and T2CKS=0 before executing the SLEEP instruction, the CPU enters the IDLE mode. In this mode, the Slow-clock source keeps T2 block running. CPU stop fetching code and all blocks are stop except T2 related circuits. Idle mode is terminated by Reset or enabled Interrupts wake up.

Another way to keep clock oscillation in IDLE mode is setting WKTIE=1 before executing the SLEEP instruction. In such condition, the WKT keeps working and wake up CPU periodically.

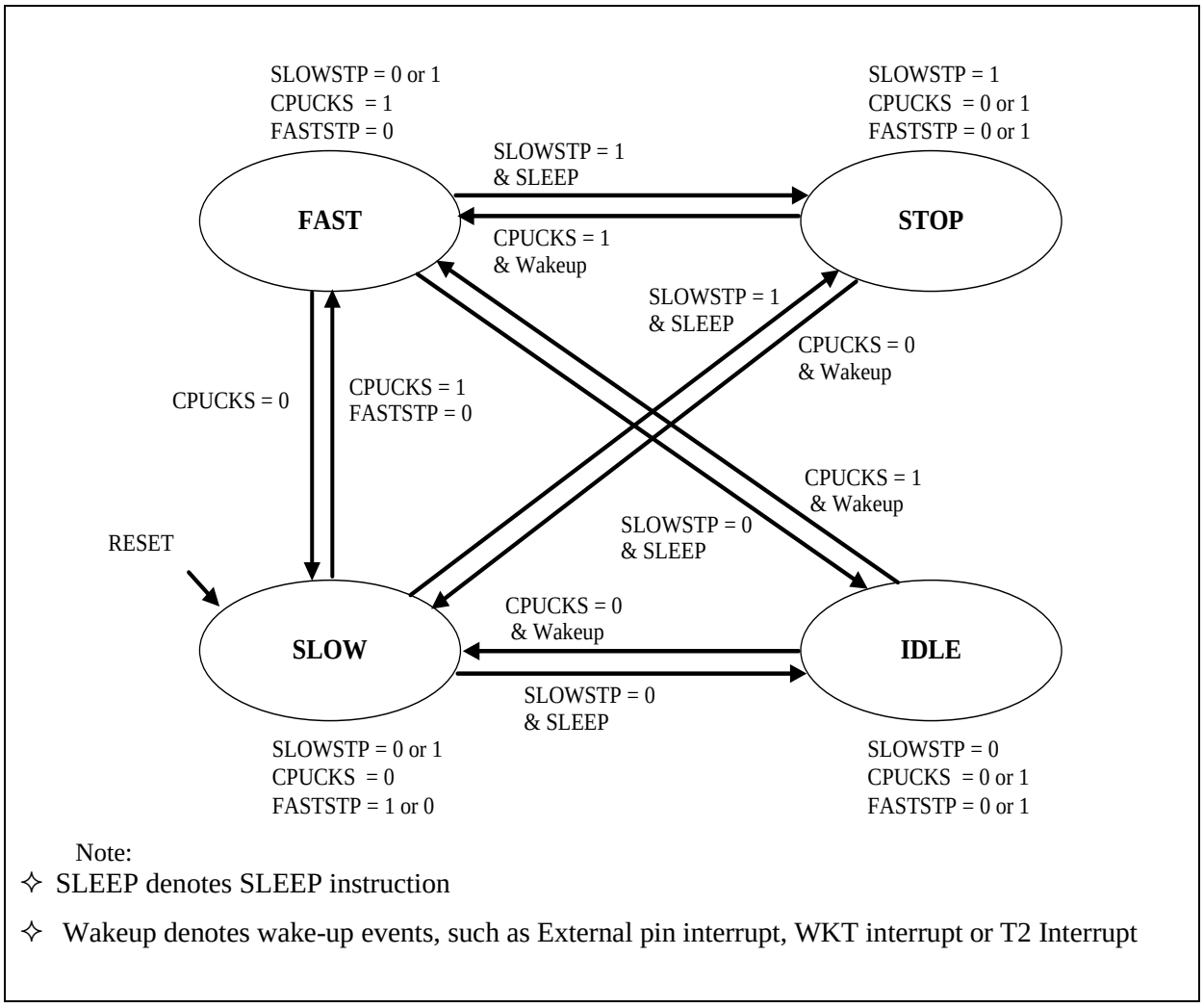
T2 and WKT/WDT are independent and have their own control registers. It is possible to keep both T2 and WKT working and wake-up in the IDLE mode.

STOP Mode:

If SLOWSTP (0Fh.4) is set, WKTIE (0Bh.3) is cleared and WDTE = 0x or 10 before executing the SLEEP instruction, the chip will enter the stop mode and all modules will stop. The stop mode is similar to the idle mode, except that all clock oscillators (fast clock or slow clock) are stopped and no clock is generated.

3.2 Dual System Clock Modes Transition

The device is operated in one of four modes: FAST mode, SLOW mode, IDLE mode, and STOP mode



Note:

✧ SLEEP denotes SLEEP instruction

✧ Wakeup denotes wake-up events, such as External pin interrupt, WKT interrupt or T2 Interrupt

CPU Operation Block Diagram

CPU Mode & Clock Functions Table:

Mode	Oscillator	Fsys	Fast-clock	Slow-clock	TM0/TM1	T2	Wakeup event
FAST	FIRC, FXT	Fast-clock	Run	Set by SLOWSTP	Run	Run	X
SLOW	SIRC, SXT	Slow-clock	Set by FASTSTP	Run	Run	Run	X
IDLE	SIRC, SXT	Stop	Stop	Run	Stop	Run	WKT/IO/T2
STOP	Stop	Stop	Stop	Stop	Stop	Stop	IO

- **FAST mode switches to SLOW mode**

  When switching from FAST mode to SLOW mode, it is recommended to perform the following steps in sequence:

(1) Enable Slow-clock (SLOWSTP=0)

(2) Switch to Slow-clock (CPUCKS=0)

(3) Stop Fast-clock (FASTSTP=1)

◇ Example: 将 Switch FAST mode to SLOW mode.

```
BCX     SLOWSTP     ; Enable Slow-clock.
NOP
BCX     CPUCKS      ; Fsys=Slow-clock.
BSX     FASTSTP     ; Disable Fast-clock.
```

- **SLOW mode switches to FAST mode**

SLOW mode can be enabled by CPUCKS=0 in CLKCTL register. The following steps are suggested to be executed by order when SLOW mode switches to FAST mode:

(1) Enable Fast-clock (FASTSTP=0)

(2) Switch to Fast-clock (CPUCKS=1)

◇Example: Switch SLOW mode to FAST mode (The Fast-clock stop)

```
BCX     FASTSTP     ; EnableFast-clock.
NOP
BSX     CPUCKS      ; Fsys=Fast-clock
```

- **IDLE mode Setting**

The IDLE mode can be configured by following setting in order:

(1) Enable Slow-clock (SLOWSTP=0) or WKT(WKTIE=1)

(2) Switch T2clock source to Slow-clock (T2CKS=0)

(3) Execute SLEEP instruction

IDLE mode can be wakened up by External interrupt, WKT interrupt and T2 interrupt。.

◇Example: Switch FAST/SLOW mode to IDLE mode.

```
BCX     SLOWSTP     ; Enable Slow-clock.
MOVLW   00000000B
MOVWX   T2CTL       ; T2  Clock source=Slow-clock. T2PSC=除以 32768
SLEEP               ; Enter IDLE mode.
```

- **STOP Mode Setting**

The STOP mode can be configured by following setting in order:

(1) Stop Slow-clock (SLOWSTP=1)

(2) Stop WKT (WKTIE=0, WDTE=10 or 0X)

(3) Execute SLEEP instruction

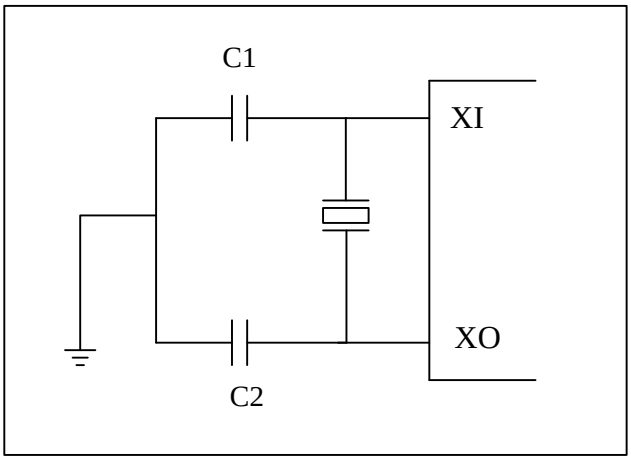
STOP mode can be wakened up only by External pin interrupt or Pin Change。

◇Example: Switch FAST/SLOW mode to STOP mode.

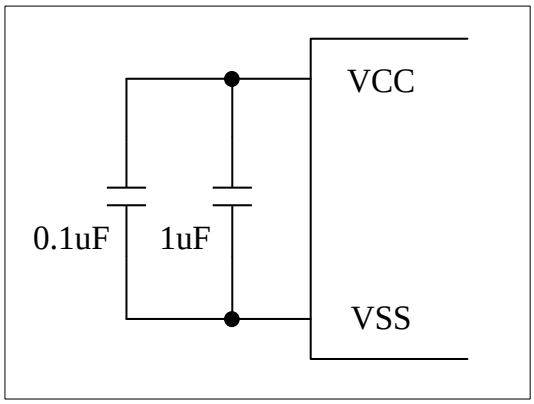
```
BSX     SLOWSTP     ; Disable Slow-clock.
MOVLW   00000000B   ; Disable WKT counting
MOVWX   INTIE
SLEEP               ; Enteer STOP mode.
```

3.3 System Clock Oscillator

System clock can be operated in four different oscillation modes. Four oscillation modes are FIRC, FXT, SIRC and SXT respectively. In the Fast Internal RC (FIRC) mode, the on-chip oscillator generates 18.432 MHz system clock. In Slow/Fast Crystal (SXT/FXT) mode, a crystal or ceramic resonator is connected to XI and XO pins to establish oscillation. Since power noise degrades the performance of Internal Clock Oscillator, placing power supply bypass capacitors 1uF and 0.1uF very close to VCC/VSS pins improves the stability of clock and the overall system。



External Oscillator Circuit
(Crystal or Ceramic)



Internal RC Mode

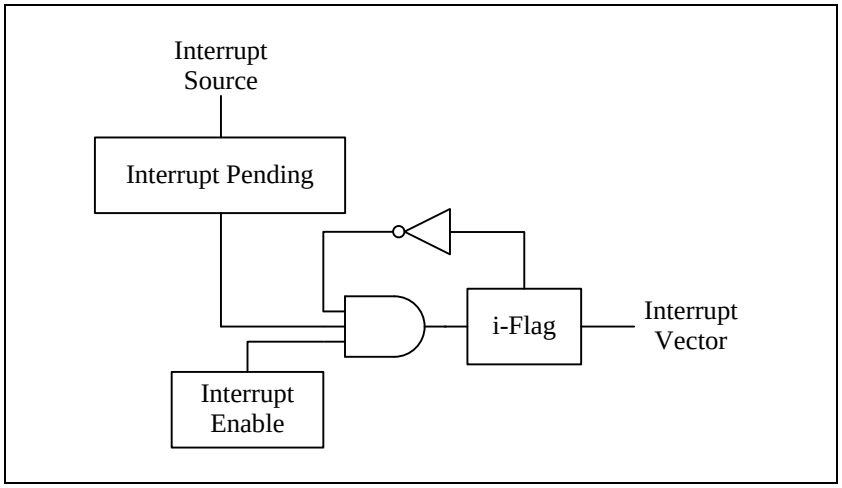
4. Interrupt

TM56F6922 has 1 level, 1 vector and 14 interrupt sources. Each interrupt source has its own enable control bit. An interrupt event will set its individual pending flag; no matter its enable control bit is 0 or 1。

If the corresponding interrupt enable bit (INTIE[7:0] or INTIE1[7:0]) has been set, it would trigger CPU to service the interrupt. CPU accepts interrupt in the end of current executed instruction cycle. In the mean while, a “LCALL 004” instruction is inserted to CPU, and i-Flag is set to prevent recursive interrupt nesting。

Before writing/reading EEPROM data, the user should disable all interrupts and enable them again after writing/reading is completed.

The i-Flag is cleared in the instruction after the “RETI” instruction. That is, at least one instruction in main program is executed before service the pending interrupt. The interrupt event is level triggered. S/W must clear the interrupt event register while serving the interrupt routine。



◇Example: Setup INT1 (PA4) interrupt request with rising edge trigger.

```
        ORG     00h         ; Reset Vector
        LGOTO   START       ; Goto user program address

        ORG     004h        ; All interrupt vector
        LGOTO   INT         ; If INT1 (PA4) input occurred rising edge
        ORG     100h
START:
        MOVLW   xxxxxx00B
        MOVWX   PAMODH      ;  select INT1 Pin Mode as Mode0

        MOVLW   xxx1xxxxB
        MOVWX   PAD         ; Release INT1, it becomes Schmitt-trigger
                            ; input with pull-up resistor
        MOVLW   xx1xxxxxB
        MOVWX   OPTION      ; Set INT1 interrupt trigger as rising edge
        MOVLW   111111 01B
        MOVWX   INTIF       ; Clear INT1 interrupt request flag
```

```
        MOVLW       00000010B
        MOVWX       INTIE           ;  Enable INT1 interrupt
MAIN:
        ….
        LGOTO       MAIN
INT:
        MOVWX       RAM20h          ; Store W data to RAM 20h
        MOVXW       STATUS          ; Get STATUS data
        MOVWX       RAM21h          ; Store SATAUS data to RAM 21h
CHKI1:
        BTXSC       INT1IE          ;
        BTXSS       INT1IF          ;
        LGOTO       END_CHK         ; if INT1IE=0 or INT1IE=1 & INT1IF=0
        LCALL       INT1INT         ; if INT1IE=1 & INT1IF=1
        ….                          ; CALL INT1 interrupt service routine
END_CHK:
        MOVXW       RAM21h          ; Get RAM 21h data
        MOVWX       STATUS          ; Restore STATUS data
        MOVXW       RAM20h          ; Restore W data
        RETI                        ; Return from interrupt
INT1INT:
        MOVLW       11111101B
        MOVWX       INTIF           ; clear INT1IF
        …..
        RET
```

0Bh/8Bh/10Bh/18Bh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTIE	ADCIE	T2IE	TM1IE	TM0IE	WKTIE	INT2IE	INT1IE	INT0IE
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

INTIE.7 **ADCIE:** ADC interrupt enable
0: disable
1: enable

INTIE.6 **T2IE:** T2 interrupt enable
0: disable
1: enable

INTIE.5 **TM1IE:** Timer1 interrupt enable
0: disable
1: enable

INTIE.4 **TM0IE:** Timer0 interrupt enable
0: disable
1: enable

INTIE.3 **WKTIE:** Wakeup Timer interrupt enable
0: disable
1: enable

INTIE.2 **INT2IE:** INT2 interrupt enable
0: disable
1: enable

INTIE.1 **INT1IE:** INT1 interrupt enable
0: disable
1: enable

INTIE.0 **INT0IE:** INT0 interrupt enable
0: disable
1: enable

0Ch	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTIF	ADCIF	T2IF	TM1IF	TM0IF	WKTIF	INT2IF	INT1IF	INT0IF
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

0Ch.7 **ADCIF:** ADC interrupt event pending flag
This bit is set by H/W after end of ADC conversion , S/W write 7Fh to INTIF to clear this flag

0Ch.6 **T2IF:** T2 interrupt event pending flag
This bit is set by H/W while T2 overflows, S/W write BFh to INTIF to clear this flag

0Ch.5 **TM1IF:** Timer1 interrupt event pending flag
This bit is set by H/W while Timer1 overflows, S/W write DFh to INTIF to clear this flag

0Ch.4 **TM0IF:** Timer0 interrupt event pending flag
This bit is set by H/W while Timer0 overflows, S/W write EFh to INTIF to clear this flag

0Ch.3 **WKTIF:** Wakeup Timer interrupt event pending flag
This bit is set by H/W while Wakeup Timer is timeout, S/W write F7h to INTIF to clear this flag

0Ch.2 **INT2IF:** INT2 pin falling interrupt pending flag
This bit is set by H/W at INT2 pin's falling edge, S/W write FBh to INTIF to clear this flag

0Ch.1 **INT1IF:** INT1 pin falling/rising interrupt pending flag
This bit is set by H/W at INT1 pin's falling/rising edge, S/W write FDh to INTIF to clear this flag

0Ch.0 **INT0IF:** INT0 pin falling/rising interrupt pending flag
This bit is set by H/W at INT0 pin's falling/rising edge, S/W write FEh to INTIF to clear this flag

81h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
OPTON	HWAUTO	INT0EDG	INT1EGE	FREQL	WDTPSC		WKTPSC	
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	1	1	1	1	1

81h.6 INT0EDG**:** INT0 interrupt trigger edge
0: falling edge trigger, 1: rising edge trigger

81h.5 INT1EDG**:** INT1 interrupt trigger edge
0: falling edge trigger, 1: rising edge trigger

0Eh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTIE1	LVDIE	EEPIE				I2CIE	UARTIE	PXIE
R/W	R/W	R/W				R/W	R/W	R/W
Reset	0	0				0	0	0

INTIE1.7 **LVDIE:** LVD interrupt enable
0: disable
1: enable

INTIE1.6 **EEPIE:** EEP write finished interrupt enable
0: disable
1: enable

INTIE1.2 **I2CIE:** Slave I2C interrupt enable
0: disable
1: enable

INTIE1.1 **UARTIE:** UART TX/RX complete interrupt enable
0: disable
1: enable

INTIE1.0 **PXIE:** Pin Change Wakeup interrupt enable
0: disable
1: enable

1Ah	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTIF1	LVDIF	EEPIF				I2CIF	UARTIF	PXIF
R/W	R/W	R/W				R/W	R/W	R/W
Reset	0	0				0	0	0

1Ah.7 **LVDIF:** Low Voltage Detect interrupt pending flag
This bit is set by H/W , S/W writes 7Fh to INTIF1 to clear this flag

1Ah.6 **EEPIF:** EEP write complete interrupt pending flag
This bit is set by H/W after EEP is written. S/W writes BFh to INTIF1 to clear this flag.

1Ah.2 **I2CIF:** Slave I2C interrupt pending flag
This bit is set by H/W while

- ✧ I2CRCD0 or I2CRCD1 receive data finished
- ✧ I2CRCD0 or I2CRCD1 data overflow occurred
- ✧ I2CTXD0 or I2CTXD1 data transmit finished

S/W writes FBh to INTIF1 to clear this flag and slave I2C related flags.

1Ah.1 **UARTIF:** UART interrupt pending flag
This bit is set by H/W while TX/RX transfer is completed; S/W writes FDh to INTIF1 to clear this flag.

1Ah.0 **PXIF:** Pin Change interrupt pending flag
This bit is set by H/W while the corresponding pin change, S/W writes FEh to INTIF1 to clear this flag.

113h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
I2CFLG	-	-	TXD1F	TXD0F	RCD1OVF	RCD1F	RCD0OVF	RCD0F
R/W	-	-	R/W	R/W	R/W	R/W	R/W	R/W
Reset	-	-	0	0	0	0	0	0

113h.5 **TXD1F:** Slave I2C transmitting data register 1 flag
This bit is set by H/W while I2CTXD1 data transmitting finished, S/W write DFh to I2CFLG to clear this flag

113h.4 **TXD0F:** Slave I2C transmitting data register 0 flag
This bit is set by H/W while I2CTXD0 data transmitting finished, S/W write EFh to I2CFLG to clear this flag

113h.3 **RCD1OVF:** Slave I2C receiving data register 1 overflow
This bit is set by H/W while receiving I2CRCD1 overflow, S/W write F7h to I2CFLG to clear this flag

113h.2 **RCD1F:** Slave I2C receiving data register 1 flag
This bit is set by H/W while data receiving to I2CRCD1 finished, S/W write FBh to I2CFLG to clear this flag

113h.1 **RCD0OVF:** Slave I2C receiving data register 0 overflow
This bit is set by H/W while receiving I2CRCD0 overflow, S/W write FDh to I2CFLG to clear this flag

113h.0 **RCD0F:** Slave I2C receiving data register 0 flag
This bit is set by H/W while data receiving to I2CRCD0 finished, S/W write FEh to I2CFLG to clear this flag

5. I/O Port

5.1 PA0-4, PA7, PB0-7, PC0-3 and PD0-7

These pins can be used as Schmitt-trigger input, CMOS push-pull output. The pull-up resistor is assignable to each pin by S/W setting. To use the pin in Schmitt-trigger input mode, S/W needs to set the I/O pin to Mode0 or Mode1 and PxD=1 (x=A, B, C or D). Reading the pin data (PxD) has different meaning. In "Read-Modify-Write" instruction, CPU actually reads the output data register. In the others instructions, CPU reads the pin state. The so-called "Read-Modify-Write" instruction includes BSX, BCX and all instructions.

These pins can operate in four different modes as below.

Mode	PA0~PA4, PA7, PB, PC, PD pin function	PxD SFR data	Pin State	Resistor Pull-up	Digital Input
Mode 0	Open Drain	0	Drive Low	N	N
	Input	1	Pull-up	Y	Y
Mode 1	Open Drain	0	Drive Low	N	N
		1	Hi-Z	N	Y
Mode 2	CMOS Output	0	Drive Low	N	N
		1	Drive High	N	N
Mode 3	ADC	X (don't care)	–	N	N

I/O Pin Function Table

The chip support I/O Pin Current control (High-Sink/LED Current Control). For efficient control, we divided the High-sink pin or LED Pins into four groups (PA, PB, PC and PD group) independently. It is enable by setting IOCCTRL register.

1Bh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
IOCCTRL	LED				HSNK			
R/W	R/W				R/W			
Reset	0	0	0	0	0	0	0	0

1Bh.7~4 **LED:** IO Pin Current Control for LED application
LED[0]: 1: enable PA as LED application
LED[1]: 1: enable PB as LED application
LED[2]: 1: enable PC as LED application
LED[3]: 1: enable PD as LED application

1Bh.3~0 **HSNK:** IO Pin Current for High Sink Control
HSNK[0]: 1: enable PA as High Sink application
HSNK[1]: 1: enable PB as High Sink application
HSNK[2]: 1: enable PC as High Sink application
HSNK[3]: 1: enable PD as High Sink application

Beside I/O port function, each pin has one or more alternative functions, such as LCD and ADC Key

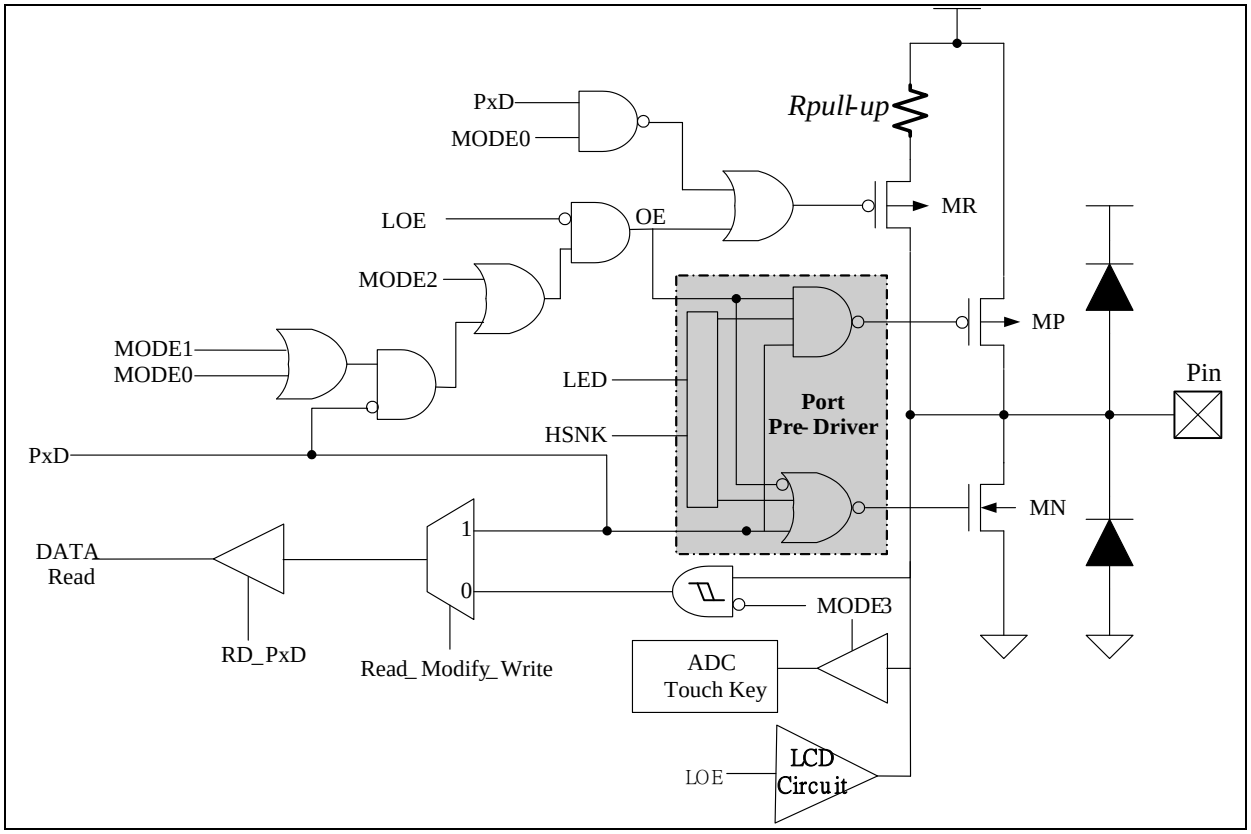
Pin Name	Wake-up	CKO	ADC	LCD	PWM	others	Mode3
PA0	Wakeup / INT0		AD25	COM2	PWM0N	I2CSDA	AD25
PA1	Wakeup		AD24	COM3	PWM1	I2CSCL	AD24
PA2	Wakeup		AD26		PWM2	TM0CKI	AD26
PA3	Wakeup		AD23		PWM3		AD23
PA4	Wakeup / INT1		AD22		PWM4		AD22
PA7	Wakeup / INT2		AD21		PWM0P/ PWM5		AD21
PB0	Wakeup	TM1OUT	AD0				AD0
PB1	Wakeup	TCOUT	AD1				AD1
PB2	Wakeup		AD2				AD2
PB3	Wakeup		AD3				AD3
PB4	Wakeup		AD4			UART RX	AD4
PB5	Wakeup		AD5			UART TX	AD5
PB6	Wakeup		AD6				AD6
PB7	Wakeup		AD7				AD7
PC0	Wakeup		AD8				AD8
PC1	Wakeup		AD9				AD9
PC2	Wakeup		AD10				AD10
PC3	Wakeup		AD11				AD11
PD0	Wakeup		AD13			XI	AD13
PD1	Wakeup		AD14			XO	AD14
PD2	Wakeup		AD15				AD15
PD3	Wakeup		AD16				AD16
PD4	Wakeup		AD17				AD17
PD5	Wakeup		AD18				AD18
PD6	Wakeup		AD19	COM1			AD19
PD7	Wakeup		AD20	COM0			AD20

PortA/B/C/D multi-function Table

The necessary SFR setting for pin's alternative function is list below.

Alternative Function	Mode	PxD SFR data	Pin State	Other necessary SFR setting
INT0, INT1, INT2 TM0CKI	**0**	1	Input with Pull-up resistor	INTxIE TM0CTL
	1	1	Input high impedance state	
AD0~AD26	**3**	X	High impedance state, used for analog signals.	ADCHS
LCD0~LCD3	**X**	X	1/2 Vcc Bias Output	LOE
UART RX	**0**	X	Input with Pull-up resistor	UART
	1	X	Input high impedance state	
UART TX	**2**	X	CMOS output	UART
PWM0N, PWM0P, PWM1/2/3/4/5	**2**	X	COMS Output	PWMOE PWM0N PWM0P PWM1/2/3/4/5
I2CSCL	**0**	1	Input with Pull-up	I2CCTL
	1	1	Input	
I2CSDA	**0**	X	Input with Pull-up/ Open Drain Output	
	1	X	Input / Open Drain Output	
XI, XO	**0**	1	Crystal oscillation	CLKCTL

Mode Setting for Port Alternative Function



PA0 Pin Structure

05h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PAD	PAD7			PAD4	PAD3	PAD2	PAD1	PAD0
R/W	R/W			R/W	R/W	R/W	R/W	R/W
Reset	1			1	1	1	1	1

05h.7~0 **PAD:** PA7~PA0 data

06h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PBD	PBD							
R/W	R/W							
Reset	1	1	1	1	1	1	1	1

06h.7~0 **PBD:** PB7~PB0 data

07h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PCD					PCD			
R/W					R/W			
Reset					1	1	1	1

07h.3~0 **PCD:** PC3~PC0 data

08h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PDD	PDD							
R/W	R/W							
Reset	1	1	1	1	1	1	1	1

08h.7~0 **PDD:** PD7~PD0 data

85h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PAMODH	PA7MOD						PA4MOD	
R/W	R/W						R/W	
Reset	0	1					0	1

85h.7~6 **PA7MOD:** PA7 Pin Mode Control
00: Mode0
01: Mode1
10: Mode2
11: Mode3

85h.1~0 **PA4MOD:** PA4 Pin Mode Control
00: Mode0
01: Mode1
10: Mode2
11: Mode3

86h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PAMODL	PA3MOD		PA2MOD		PA1MOD		PA0MOD	
R/W	R/W		R/W		R/W		R/W	
Reset	0	1	0	1	0	1	0	1

86h.7~6 **PA3MOD:** PA3 Pin Mode Control
00: Mode0
01: Mode1
10: Mode2
11: Mode3

86h.5~4 **PA2MOD:** PA2 Pin Mode Control
00: Mode0
01: Mode1
10: Mode2
11: Mode3

86h.3~2 **PA1MOD:** PA1 Pin Mode Control
00: Mode0
01: Mode1
10: Mode2
11: Mode3

86h.1~0 **PA0MOD:** PA0 Pin Mode Control
00: Mode0
01: Mode1
10: Mode2
11: Mode3

87h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PBMODH	PB7MOD		PB6MOD		PB5MOD		PB4MOD	
R/W	R/W		R/W		R/W		R/W	
Reset	0	1	0	1	0	1	0	1

87h.7~6 **PB7MOD:** PB7 Pin Mode Control
00: Mode0
01: Mode1
10: Mode2
11: Mode3

87h.5~4 **PB6MOD:** PB6 Pin Mode Control
00: Mode0
01: Mode1
10: Mode2
11: Mode3

87h.3~2 **PB5MOD:** PB5 Pin Mode Control
00: Mode0
01: Mode1
10: Mode2
11: Mode3

87h.1~0 **PB4MOD:** PB4 Pin Mode Control
00: Mode0
01: Mode1
10: Mode2
11: Mode3

88h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PBMODL	PB3MOD		PB2MOD		PB1MOD		PB0MOD	
R/W	R/W		R/W		R/W		R/W	
Reset	0	1	0	1	0	1	0	1

88h.7~6 **PB3MOD:** PB3 Pin Mode Control
00: Mode0
01: Mode1
10: Mode2
11: Mode3

88h.5~4 **PB2MOD:** PB2 Pin Mode Control
00: Mode0
01: Mode1
10: Mode2
11: Mode3

88h.3~2 **PB1MOD:** PB1 Pin Mode Control
00: Mode0
01: Mode1
10: Mode2
11: Mode3

88h.1~0 **PB0MOD:** PB0 Pin Mode Control
00: Mode0
01: Mode1
10: Mode2
11: Mode3

8Ch	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PCMODL	PC3MOD		PC2MOD		PC1MOD		PC0MOD	
R/W	R/W		R/W		R/W		R/W	
Reset	0	1	0	1	0	1	0	1

8Ch.7~6 **PC3MOD:** PC3 Pin Mode Control
00: Mode0
01: Mode1
10: Mode2
11: Mode3

8Ch.5~4 **PC2MOD:** PC2 Pin Mode Control
00: Mode0
01: Mode1
10: Mode2
11: Mode3

8Ch.3~2 **PC1MOD:** PC1 Pin Mode Control
00: Mode0
01: Mode1
10: Mode2
11: Mode3

8Ch.1~0 **PC0MOD:** PC0 Pin Mode Control
00: Mode0
01: Mode1
10: Mode2
11: Mode3

8Dh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PDMODH	PD7MOD		PD6MOD		PD5MOD		PD4MOD	
R/W	R/W		R/W		R/W		R/W	
Reset	0	1	0	1	0	1	0	1

8Dh.7~6 **PD7MOD:** PD7 Pin Mode Control
00: Mode0
01: Mode1
10: Mode2
11: Mode3

8Dh.5~4 **PD6MOD:** PD6 Pin Mode Control
00: Mode0
01: Mode1
10: Mode2
11: Mode3

8Dh.3~2 **PD5MOD:** PD5 Pin Mode Control
00: Mode0
01: Mode1
10: Mode2
11: Mode3

8Dh.1~0 **PD4MOD:** PD4 Pin Mode Control
00: Mode0
01: Mode1
10: Mode2
11: Mode3

8Eh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PDMODL	PD3MOD		PD2MOD		PD1MOD		PD0MOD	
R/W	R/W		R/W		R/W		R/W	
Reset	0	1	0	1	0	1	0	1

8Eh.7~6 **PD3MOD:** PD3 Pin Mode Control
00: Mode0
01: Mode1
10: Mode2
11: Mode3

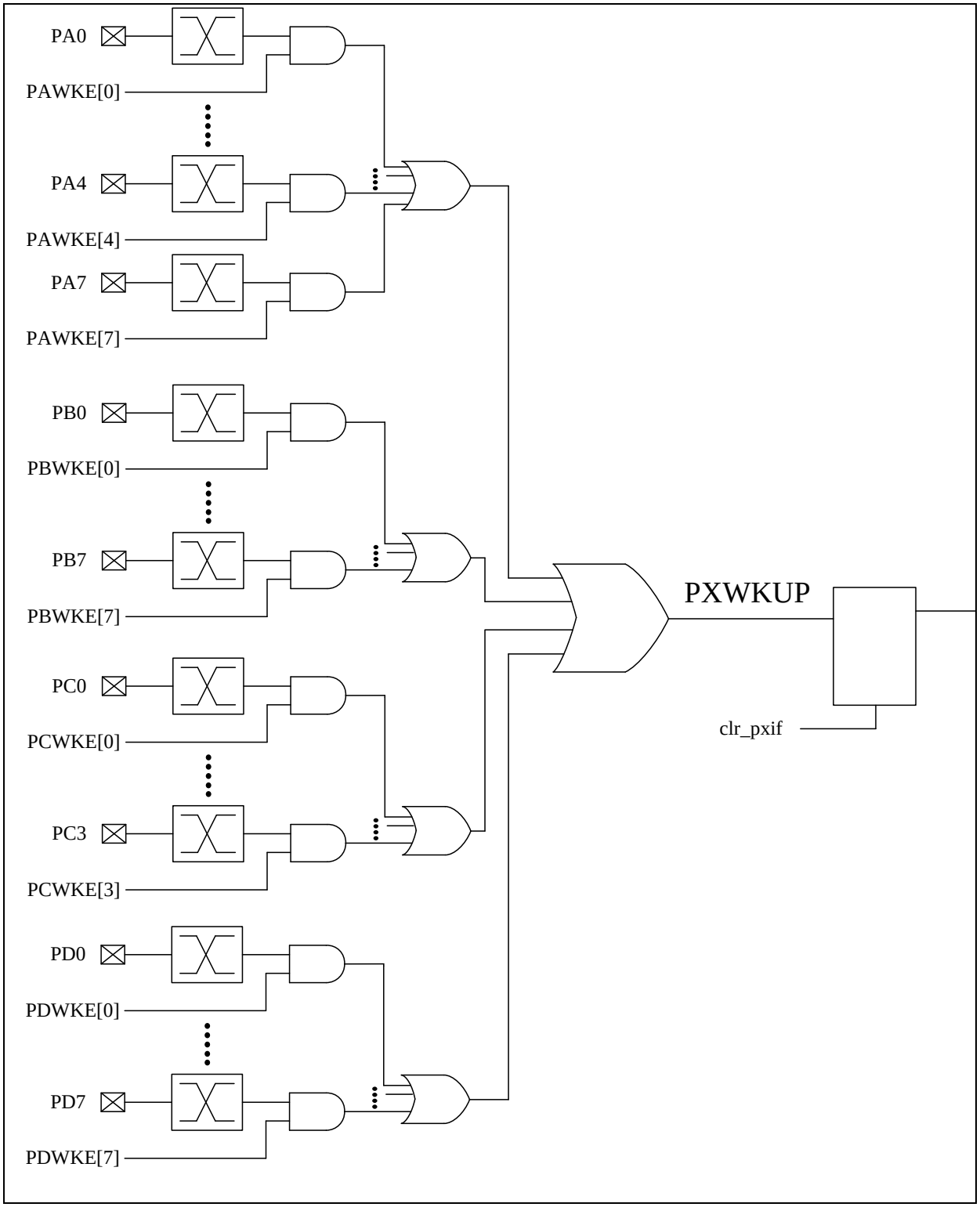
8Eh.5~4 **PD2MOD:** PD2 Pin Mode Control
00: Mode0
01: Mode1
10: Mode2
11: Mode3

8Eh.3~2 **PD1MOD:** PD1 Pin Mode Control
00: Mode0
01: Mode1
10: Mode2
11: Mode3

8Eh.1~0 **PD0MOD:** PD0 Pin Mode Control
00: Mode0
01: Mode1
10: Mode2
11: Mode3

5.2 Pin Interrupt

Pin interrupts include INT0, INT1,INT2 and PA0-4, PA7, PB0-7, PC0-3 and PD0-7 Change Interrupt. These pins also have the STOP mode wake up capability. INT0 and INT1 are falling edge or rising edge triggered, INT2 is falling edge triggered. All of the IO pins Change Interrupt is triggered by any pin state change when the pin wakeup register PxWKE (x=A, B, C or D) and Pin Change Interrupt Enable bit (PXIE) is set to enable the function.



◇Example: Setup Port B as Pin Change interrupts

```
        ORG         00h             ; Reset Vector
        LGOTO       START           ; Goto user program address

        ORG         004h            ; All interrupt vector
        LGOTO       INT             ; If PXINT (PB) pin change occurred
        ORG         100h
START:
        MOVLW       00000000B
        MOVWX       PBMODH          ; set PB Pin Mode as Mode0
        MOVWX       PBMODL

        MOVLW       11111111B
        MOVWX       PBWKE           ; set PB as Pin Change wakeup
                                    ;  input with pull-up resistor
        BSX         PXIE            ; Enable PXIE (Pin Change Interrupt Enable)

MAIN:
        ….
        SLEEP
        ….
        LGOTO       MAIN
INT:
        MOVWX       RAM20h          ; Store W data to RAM 20h
        MOVXW       STATUS          ; Get STATUS data
        MOVWX       RAM21h          ; Store SATAUS data to RAM 21h
        …
CHKPX:
        BTXSC       PXIE
        BTXSS       PXIF
        LGOTO       END_CHK         ; if PXIE=0 or PXIE=1 & PXIF=0
        LCALL       PXINT           ; if PXIE=1 & PXIF=1

END_CHK:
        MOVXW       RAM21h          ; Get RAM 21h data
        MOVWX       STATUS          ; Restore STATUS data
        MOVXW       RAM20h          ; Restore W data
        RETI                        ; Return from interrupt


PXINT:
        MOVLW       11111110B
        MOVWX       INTIF1          ; clear PXIF
        …..
        RET
```

1Ch	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PAWKE	PAWKE7			PAWKE 4	PAWKE 3	PAWKE 2	PAWKE 1	PAWKE 0
R/W	R/W			R/W	R/W	R/W	R/W	R/W
Reset	0			0	0	0	0	0

1Ch.7~0 **PAWKE:** PA7~PA0 individual wakeup enable

1Dh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PBWKE	PBWKE							
R/W	R/W							
Reset	0	0	0	0	0	0	0	0

1Dh.7~0 **PBWKE:** PB7~PB0 individual wakeup enable

1Eh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PCWKE					PCWKE			
R/W					R/W			
Reset					0	0	0	0

1Eh.3~0 **PCWKE:** PC3~PC0 individual wakeup enable

1Fh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PDWKE	PDWKE							
R/W	R/W							
Reset	0	0	0	0	0	0	0	0

1Fh.7~0 **PDWKE:** PD7~PD0 individual wakeup enable

0Eh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTIE1	LVDIE	EEPIE				I2CIE	UARTIE	PXIE
R/W	R/W	R/W				R/W	R/W	R/W
Reset	0	0				0	0	0

INTIE1.0 **PXIE:** Pin Change Wakeup interrupt enable
0:disable
1:enable

1Ah	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTIF1	LVDIF	EEPIF				I2CIF	UARTIF	PXIF
R/W	R/W	R/W				R/W	R/W	R/W
Reset	0	0				0	0	0

1Ah.0 **PXIF:** Pin Change interrupt pending flag
This bit is set by H/W while the corresponding pin change, S/W write FEh to INTIF1 to clear this flag

108h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
LOE					LOE			
R/W					R/W			
Reset					0	0	0	0

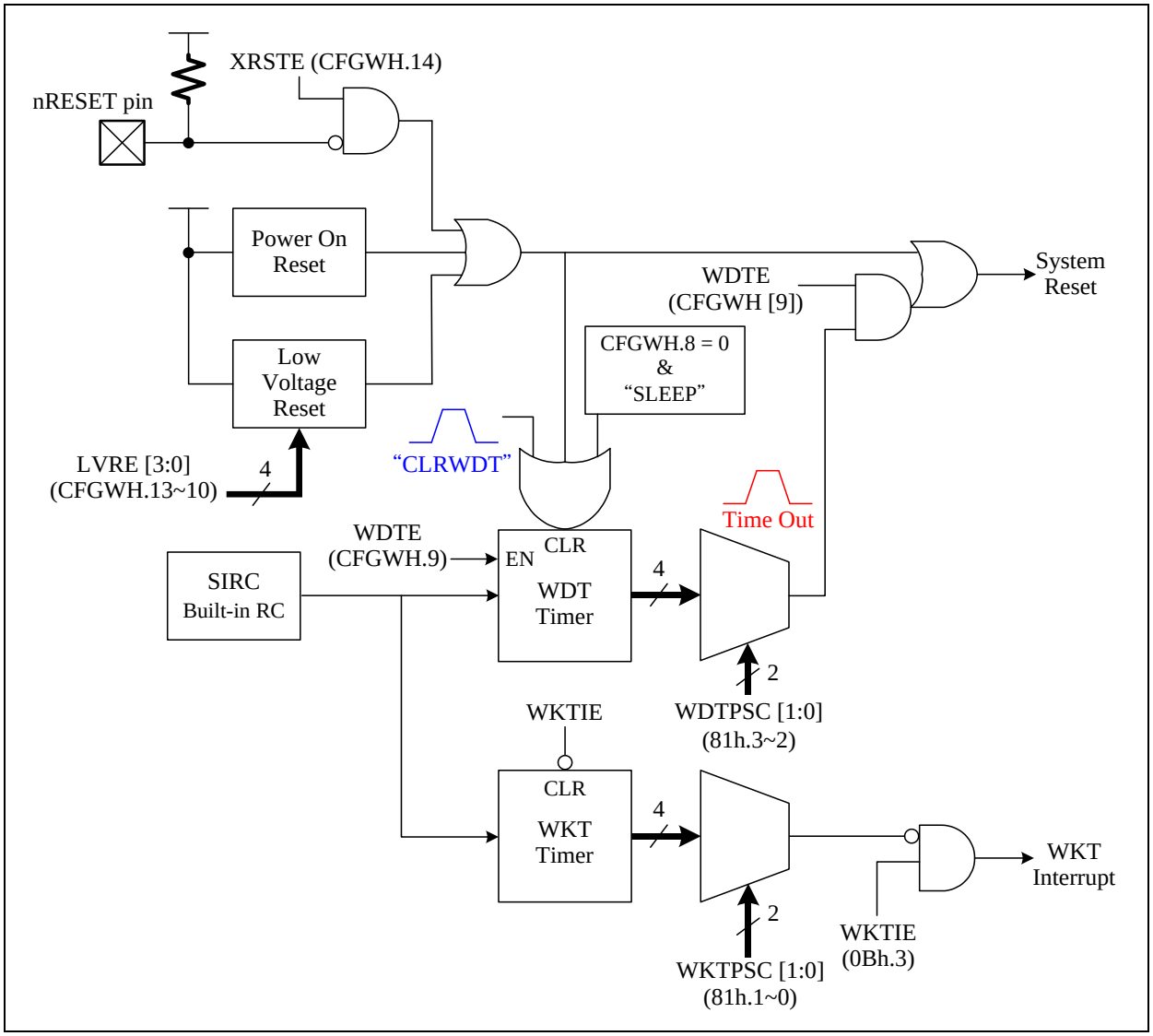
108h.3~0 COM0~COM3 LCD 1/2 bias output enable control
0001: COM0 (PD7) enable
0010: COM1 (PD6) enable
0100: COM2 (PA0) enable
1000: COM3 (PA1) enable

6. Peripheral Functional Block

6.1 Watchdog (WDT) /Wakeup (WKT) Timer

The WDT and WKT share the same built-in internal RC Oscillator and have individual own counters. The overflow period of WDT, WKT can be selected by individual prescaler (WDTPSC [1:0] , WKTPSC [1:0]) . The WDT timer is cleared by the CLRWDT instruction. If the Watchdog is enabled (CFGWH.9=WDTE=1) , the WDT generates the chip reset signal. Set CFGWH.8 to '0' can let WDT timer stop counting after executing SLEEP instruction, i.e. CFGWH.8=1 WDT timer is always keep counting even if the SLEEP instruction is executed.

The WKT timer is an interval timer, WKT time out will generate WKT Interrupt Flag (WKTIF). The WKT timer is cleared/stopped by WKTIE=0. Set WKTIE=1, the WKT timer will always count regardless at any CPU operating mode.



WDT/WKT 框图

Watchdog clear is controlled by CLRWDT instruction and moving any value into WDTCLR is to clear watchdog timer.\

◇Example: Clear watchdog timer by CLRWDT instruction

```
MAIN:
        …                       ; Execute program
        CLRWDT                  ; Execute CLRWDT instruction.
        …
        GOTO    MAIN
```

◇ Example: Setup WDT time and disable after executing SLEEP instruction

```
        MOVLW   0000**01**11B
        MOVWX   OPTION          ; Select WDT Time out=328 ms @5V

        SLEEP
```

◇ Example: Set WKT period and interrupt function

```
        MOVLW   000101**10**B
        MOVWX   OPTION          ; Select WKT period=82 ms @5V.

        MOVLW   1111**0**111B   ; Clear WKT interrupt request flag by using byte operation
                                ; Don't use bit operation "BCX WKTIF" clear interrupt flag
        MOVWX   INTIF           ;

        MOVLW   0000**1**000B   ; Enable WKT interrupt function
        MOVWX   INTIE
```

0Ch	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTIF	ADCIF	T2IF	TM1IF	TM0IF	WKTIF	INT2IF	INT1IF	INT0IF
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

0Ch.3 **WKTIF:** Wakeup Timer interrupt event pending flag
This bit is set by H/W while Wakeup Timer is timeout, S/W write F7h to INTIF to clear this flag

0Bh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTIE	ADCIE	T2IE	TM1IE	TM0IE	WKTIE	INT2IE	INT1IE	INT0IE
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

0Bh.3 **WKTIE:** Wakeup Timer interrupt enable
0: disable
1: enable

81h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
OPTON	HWAUTO	INT0EDG	INT1EGE	FREQL	WDTPSC		WKTPSC	
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	1	1	1	1	1

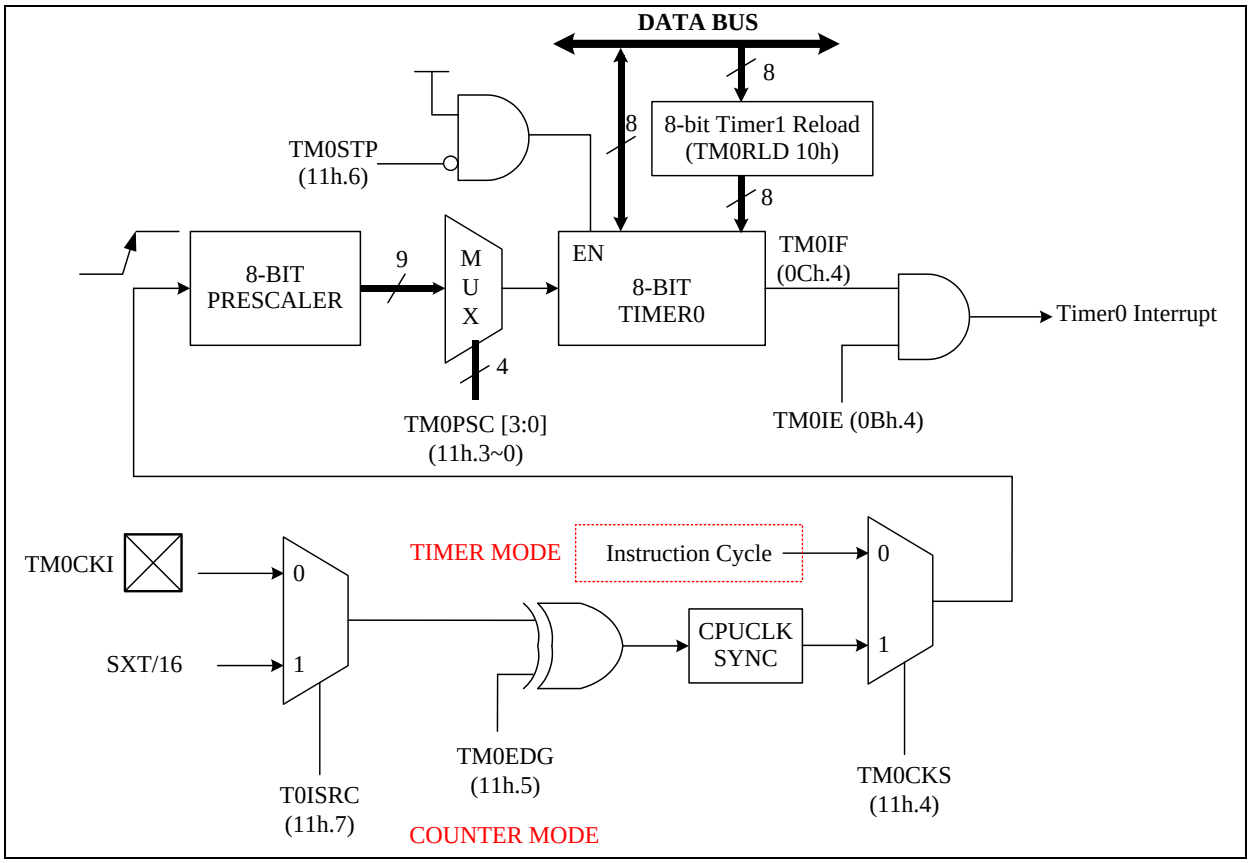
81h.4 **FREQL:** SIRC frequency selection;
0: SIRC 60KHz,
1: SIRC 50Khz

81h.3~2 **WDTPSC:** WDT period (@VCC=5V)
00: 164 ms 01: 328 ms 10: 655 ms 11: 1311 ms; when FREQL=1
00: 137ms 01: 273 ms 10: 546 ms 11: 1092 ms; when FREQL=0

81h.1~0 **WKTPSC:** WKT period (@VCC=5V)
00: 21ms 01: 41 ms 10: 82 ms 11: 164 ms; when FREQL=1
00: 17ms 01: 34 ms 10: 68 ms 11: 137 ms; when FREQL=0

6.2 Timer0

The Timer0 is an 8-bit wide register 01h (TM0). It can be read or written as any other register. Besides, Timer0 increases itself periodically and automatically rolls over a new "offset value" (TM0RLD) while it rolls over based on the pre-scaled clock source, which can be Fsys/2 when TM0CKS=0 or TM0CKI (PA2) rising/falling input when TM0CKS=1 and T0ISRC=0 or SXT/16 when TM0CKS=1 and T0ISRC=1. The Timer0 increase rate is determined by “Timer0 Pre-Scale” (TM0PSC) register. The Timer0 always generates TM0IF when its count rolls over. It generates Timer0 Interrupt if (TM0IE) is set. Timer0 can be stopped counting if the TM0STP bit is set.



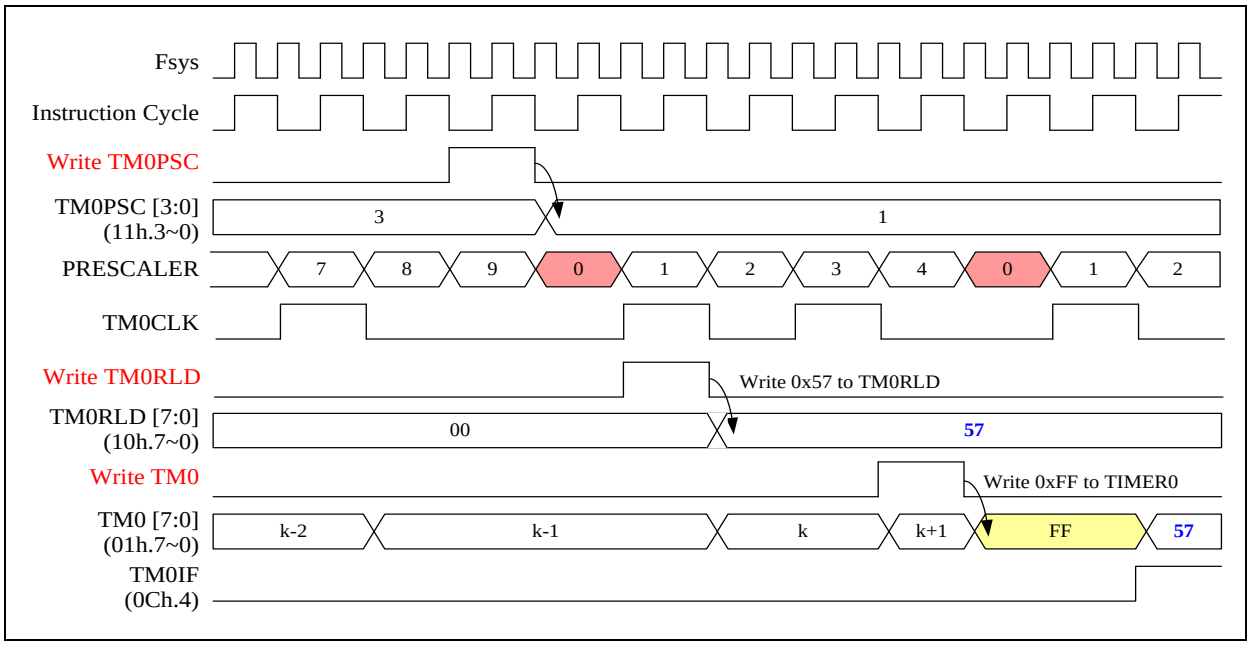
Timer0 框图

	Timer Mode	Counter Mode TM0CKI(PA2)	Counter Mode SXT/16
TM0CKS	0	1	1
T0ISRC	x	0	1

Timer0 Mode Control Signal Table

The following timing diagram describes the Timer0 works in pure Timer mode

When the Timer0 prescaler (TM0PSC) is written, the internal 8-bit prescaler will be cleared to 0 to make the counting period correct at the first Timer0 count. TM0CLK is the internal signal that causes the Timer0 to increase by 1 at the end of TM0CLK. TM0WR is also the internal signal that indicates the Timer0 is directly written by instruction; meanwhile, the internal 8-bit prescaler will be cleared. When Timer0 counts from FFh to TM0RLD, TM0IF (Timer0 Interrupt Flag) will be set to 1 and generate interrupt if TM0IE (Timer0 Interrupt Enable) is set.



Timer0 works in Timer mode (TM0CKS=0)

The equation of TM0 interrupt time value is as following:

TM0 interrupt interval cycle time=Fsys/2/TM0PSC/ (256-TM0)

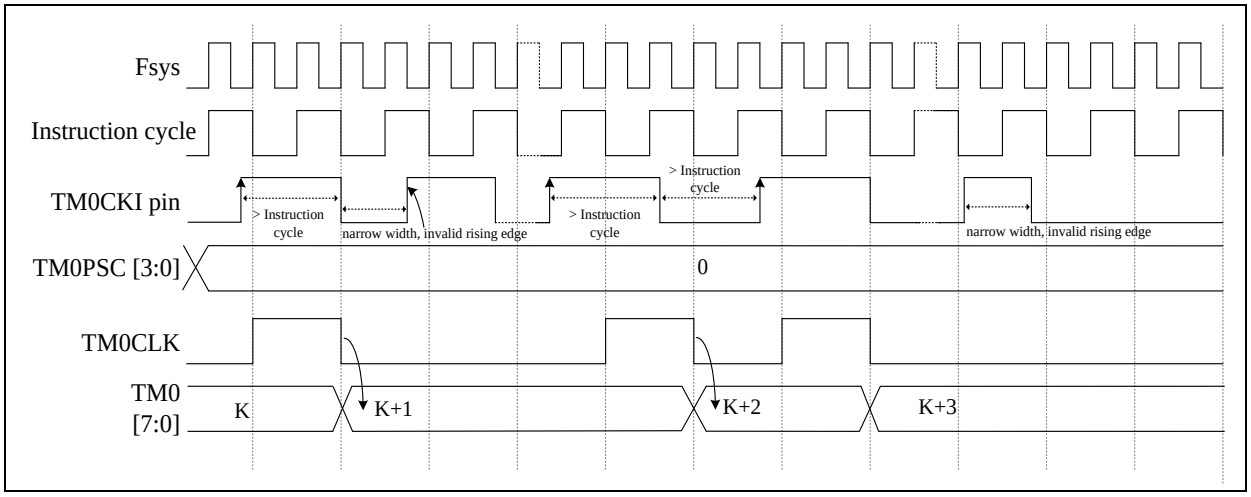
◇Example: Setup TM0 work in Timer mode

```
; Setup TM0 clock source and divider
        MOVLW   0000 0101B      ; TM0CKS=0, Setup TM0 clock= Fsys/2
        MOVWX   TM0CTL          ; TM0PSC=5, TM0PSC= Fsys/64
; Set TM0 timer.
        BSX     TM0STP          ; Disable TM0 counting (Default "0")..
        MOVLW   156
        MOVWX   TM0             ; Write 156 into TM0 register
        MOVLW   124
        MOVWX   TM0RLD          ; Write 156 into TM0RLD register

; Enable TM0 timer and interrupt function.
        MOVLW   1110 1111B      ; Clear TM0 request interrupt flag by byte operation
        MOVWX   INTIF           ; 0Ch
        MOVLW   0001 0000B      ; Enable TM0 interrupt function
        MOVWX   INTIE           ; 0Bh
        BCX     TM0STP          ; Enable TM0 counting (Default “0”).
```

The following timing diagram describes the Timer0 works in Counter mode.

If TM0CKS=1 then Timer0 counter source clock is from TM0CKI pin. TM0CKI signal is synchronized by instruction cycle (Fsys/2) that means the high/low time durations of TM0CKI must be longer than one instruction cycle time (Fsys/2) to guarantee each TM0CKI's change will be detected correctly by the synchronizer.



Timer0 works in Counter mode for TM0CKI (TM0EDG=0) , TM0CKS=1

◇Example: Setup TM0 work in Counter mode and clock source from TM0CKI pin (PA2)

```
; Setup TM0 clock source from TM0CKI pin (PA2) and divider.
        MOVLW   00110000B
        MOVWX   TM0CTL          ; TM0EDG=1
                                ; Select TM0 prescaler counting edge=falling edge.
                                ; TM0CKS=1, Setup TM0 clock=TM0CKI pin (PA2)
                                ; TM0PSC=0
                                ; TM0 clock prescaler= TM0CKI divided by 1
; Set TM0 timer and stop TM0 counting.
        BSX     TM0STP          ; Disable TM0 counting (Default "0").
        MOVLW   00H
        MOVWX   TM0             ; Write 0 into TM0 register 01H.
; Start TM0 count and read TM0 counter.
        BCX     TM0STP          ; Enable TM0 counting.
        NOP
        NOP
        NOP
        BSX     TM0STP          ; Disable TM0 counting (Default "0")

        MOVXW   TM00
```

01h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TM0	TM0							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

01h **TM0:** Timer0 content

0Bh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTIE	ADCIE	T2IE	TM1IE	TM0IE	WKTIE	INT2IE	INT1IE	INT0IE
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

0Bh.4 **TM0IE:** Timer0 interrupt enable
0:disable
1:enable

0Ch	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTIF	ADCIF	T2IF	TM1IF	TM0IF	WKTIF	INT2IF	INT1IF	INT0IF
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

0Ch.4 **TM0IF:** Timer0 interrupt event pending flag
This bit is set by H/W while Timer0 overflows, S/W write EFh to INTIF to clear this flag

10h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TM0RLD	TM0RLD							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

10h **TM0RLD:** Timer0 Reload Data

11h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TM0CTL	T0ISRC	TM0STP	TM0EDG	TM0CKS	TM0PSC			
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

11h.7 **T0ISRC:** Timer0 Counter mode source
0: TM0CKI pin (PA2)
1: SXT/16

11h.6 **TM0STP:** Timer0 counter stop
0: Release
1: Stop counting

11h.5 **TM0EDG:** Timer0 prescaler counting edge for TM0CKI pin
0: rising edge
1: falling edge

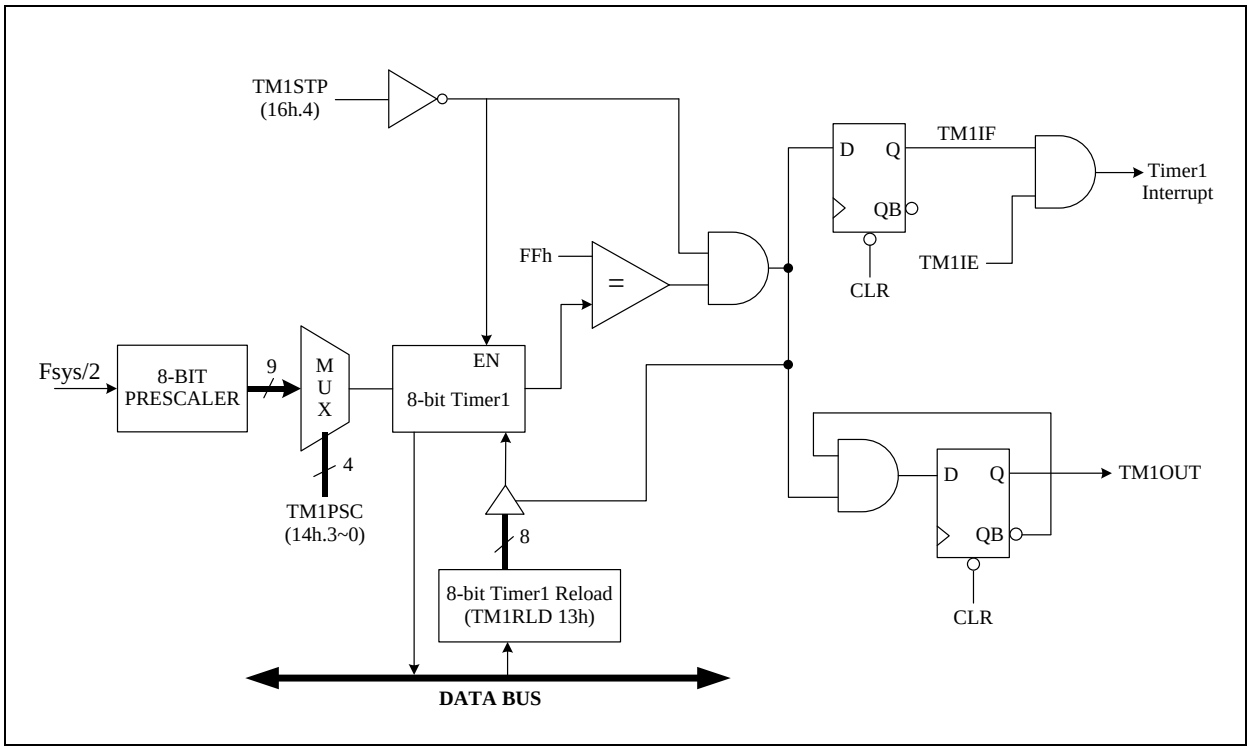
11h.4 **TM0CKS**: Timer0 prescaler clock source
0:Fsys/2
1:TM0CKI pin (PA2 pin)

11h.3~0 **TM0PSC**: Timer0 prescaler. Timer0 prescaler clock source divided by

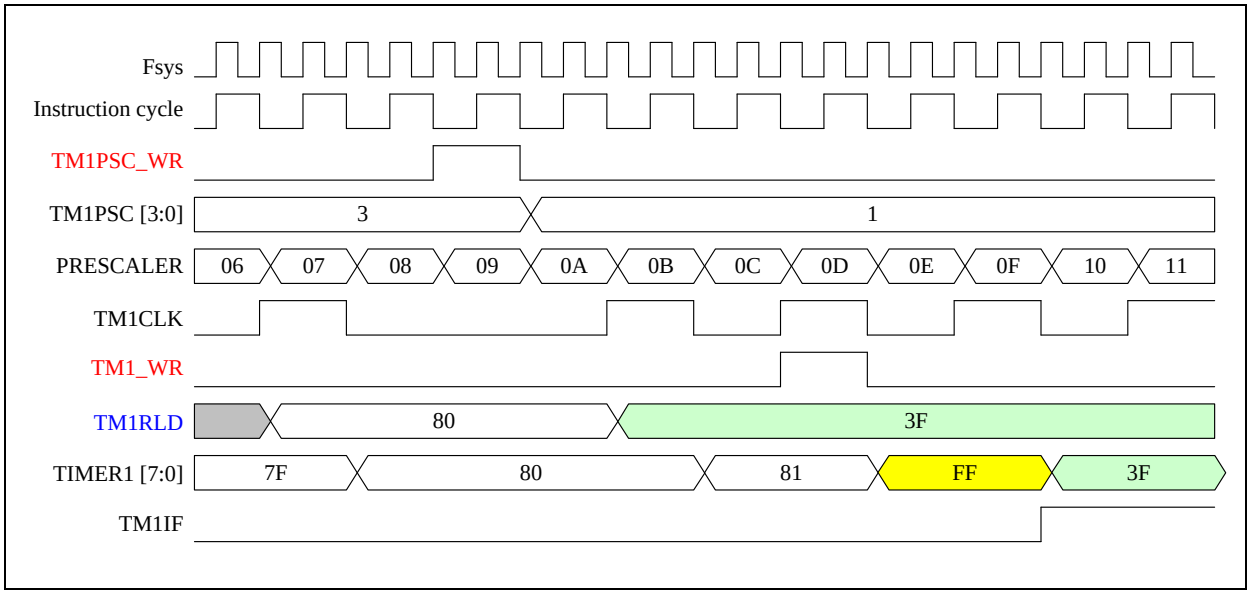
0000: /1	0001: /2	0010: /4	0011: /8
0100: /16	0101: /32	0110: /64	0111: /128
1000: /256	1001: /512	1010: /1024	1011: /2048
1100: /4096	1101: /8192	1110: /16384	1111: /32768

6.3 Timer1

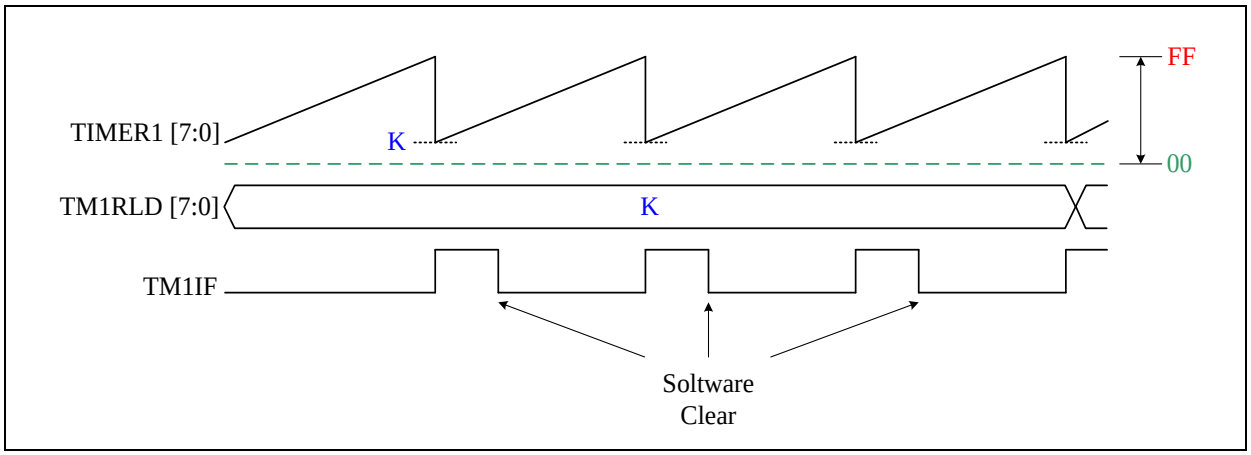
TM1 (12h.7~0) is an 8-bit wide register. It can be read or written like any other register. Additionally, Timer1 increments itself periodically and automatically reloads a new "offset value" (TM1RLD) on rollover based on the prescaled instruction clock (Fsys/2). The increase rate of Timer1 is selected by the TM1PSC register. If the TM1IE bit is set, it will generate a Timer1 interrupt. Timer1 counting can be stopped if the TM1STP bit is set. TM1OUT is an output signal that toggles when Timer1 overflow.



Timer1 Block Diagram



Timer1 Timing Diagram



Timer1 Reload Diagram

◇Example: Setup TM1 work in Timer mode and counting overflow toggle out to TM1OUT (PB0) configuration.

```
; Setup TM1 clock source, divider and enable TM1OUT
        MOVLW   0000 0101B
        MOVWX   TM1CTL      ; TM1PSC=5 , Select TM1 clock=Fsys/64.
        BSX     TM1OE       ; Enable TM1OUT function pin (PB0).

; Set TM1 timer offset and stops TM1 counting
        BSX     TM1STP      ; Stop TM1 counting (Default "0").
        MOVLW   F0H
        MOVWX   TM1         ; Write F0H into TM1 counter

; Enable TM1 timer and interrupt function.
        MOVLW   11011111B   ; Clear TM1 request interrupt flag by byte operation
        MOVWX   INTIF       ; 09H

        MOVLW   00100000B   ; Enable TM1 interrupt function.
        MOVWX   INTIE       ;

        BCX     TM1STP      ; Enable TM1 counting (Default “0”).
```

Example:

Fsys=4 MHz, TM1PSC=1, TM1 clock source=Fsys/4=1 MHz

TM1RLD=0xF0,

TM1 interrupt time= (1/1 MHz) * (0xFF – 0xF0) =1 us*16=16 us

TM1OUT output time period=16 us *2=32 us.

TM1OUT output frequency=1/32 us=31.250 KHz.

0Bh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTIE	ADCIE	T2IE	TM1IE	TM0IE	WKTIE	INT2IE	INT1IE	INT0IE
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

0Bh.5 **TM1IE:** Timer1 interrupt enable
0: disable
1: enable

0Ch	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTIF	ADCIF	T2IF	TM1IF	TM0IF	WKTIF	INT2IF	INT1IF	INT0IF
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

0Ch.5 **TM1IF:** Timer1 interrupt event pending flag
This bit is set by H/W while Timer1 overflows, S/W write DFh to INTIF to clear this flag

12h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TM1	TM1							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

12h **TM1:** Timer1 content

13h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TM1RLD	TM1RLD							
R/W	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

13h.7~0 **TM1RLD:** Timer1 reload offset value while it rolls over

14h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TM1CTL	–	–	–	TM1STP	TM1PSC			
R/W	–	–	–	R/W	W	W	W	W
Reset	–	–	–	0	0	0	0	0

14h.4 **TM1STP:** Timer1 counter stop
0: Timer1 Running
1: Timer1 Stop counting

14h.3~0 **TM1PSC:** Timer1 prescaler. Timer1 clock source divided by
0000: Fsys/2 0001: Fsys/4 0010: Fsys/8 0011: Fsys/16
0100: Fsys/32 0101: Fsys/64 0110: Fsys/128 0111: Fsys/256
1xxx: Fsys/512

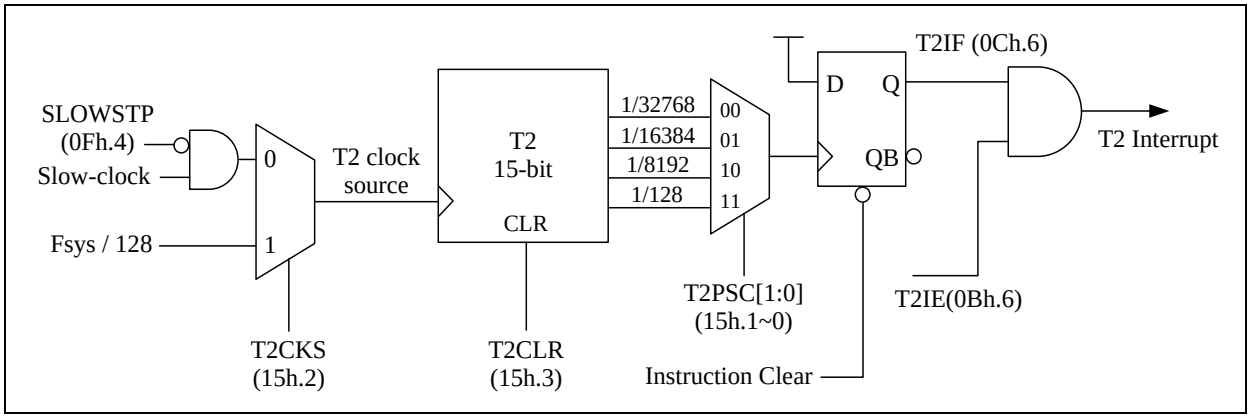
98h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
MF098	TCOE	TM1OE	PWM0OE	PWM2OE	PWM1AOE	PWM1BOE	PWM1COE	-
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	-
Reset	0	0	0	0	0	0	0	-

F1B.5 **TM1OE:** Enable Timer1 overflow toggle output to PB0 pin (TM1OUT)

6.4 T2:15-bit Timer

The T2 is a 15-bit counter and the clock sources are from either Fsys/128 or Slow-clock. It is used to generate time base interrupt and T2 counter block clock. The T2 content cannot be read by instructions. It generates interrupt flag T2IF (0Ch.6) with the clock divided by 32768/16384/8192/128 depends on T2PSC[1:0] (15h.1~0) register bits.

The following figure shows the block diagram of T2.



T2 Block Diagram

范例:

[CPU 以快速模式运行, Fsys=Fast-clock= FIRC/4= 4.6 MHz]

◇Example:

```
; Setup T2 clock source and divider .
        MOVLW   0000 1 101B     ;15h.3 (T2CLR)=1, Stop T2 counting
        MOVWX   T2CTL           ;15h.2 (T2CKS) = 1, T2 clock source = Fsys/128
                                ;15h.1~0 (T2PSC) =1, Divided by 16384

; Enable T2 timer and interrupt function.
        MOVLW   10111111B       ; Clear T2 request interrupt flag by byte operation
        MOVWX   INTIF           ;

        MOVLW   01000000B       ; Enable T2 interrupt function.
        MOVWX   INTIE           ;

        BCX     T2CLR           ; (T2CLR)=0, Enable T2 counting (Default “0”).
```

T2 clock source is Fsys/128 = 4.6 MHz/128 = 36000 Hz, T2PSC = /16384

T2 frequency = 36000 Hz / 16384 ≒ 2.197 Hz

◇Example:

[CPU running at SLOW mode, Fsys = Slow-clock = SIRC= 50KHz]

◇Example:

```
; Setup T2 clock source and divider
        MOVLW   0000 1 000B     ; 15h.3 (T2CLR)=1, Stop T2 counting
        MOVWX   T2CTL           ; 15h.2 (T2CKS) = 0, T2clock source = Slow-clock
                                ; 15.1~0 (T2PSC) =0, Divided by32768
; Enable T2 timer and interrupt function.
        MOVLW   10111111B       ; Clear T2 request interrupt flag
        MOVWX   INTIF

        MOVLW   01000000B       ; Enable T2 interrupt function.
        MOVWX   INTIE

        BCX     T2CLR           ; (T2CLR)=0, Enable T2 counting (Default “0”).
```

T2clock source is Slow-clock = 50KHz, T2PSC = /32768,

T2 frequency = 50000Hz / 32768 ≒ 1.53Hz

0Bh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTIE	ADCIE	T2IE	TM1IE	TM0IE	WKTIE	INT2IE	INT1IE	INT0IE
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

0Bh.6 **T2IE:** 2 interrupt enable
0: disable
1: enable

0Ch	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTIF	ADCIF	T2IF	TM1IF	TM0IF	WKTIF	INT2IF	INT1IF	INT0IF
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

0Ch.6 **T2IF:** T2 interrupt event pending flag
This bit is set by H/W while T2 overflows, S/W write BFh to INTIF to clear this flag

0Fh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
CLKCTL	SCKTYPE	FCKTYPE		SLOWSTP	FASTSTP	CPUCKS	CPUPSC	
R/W	R/W	R/W		R/W	R/W	R/W	R/W	R/W
Reset	0	0		0	1	0	1	1

0Fh.4 **SLOWSTP:** Stop Slow-clock in Stop Mode
0: No Stop。1:Stop

15h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
T2CTL					T2CLR	T2CKS	T2PSC	
R/W					R/W	R/W	R/W	R/W
Reset					1	0	0	0

15h.3 **T2CLR:** T2 counter clear
0: Release1: Stop counting
15h.2 **T2CKS:** “T2 clock source” selection.
0: Slow-clock ; 1: Fsys/128
15h.1~0 **T2PSC:** T2 prescaler. “T2 clock source” divided by
00: 32768 01: 16384 10: 8192 11: 128

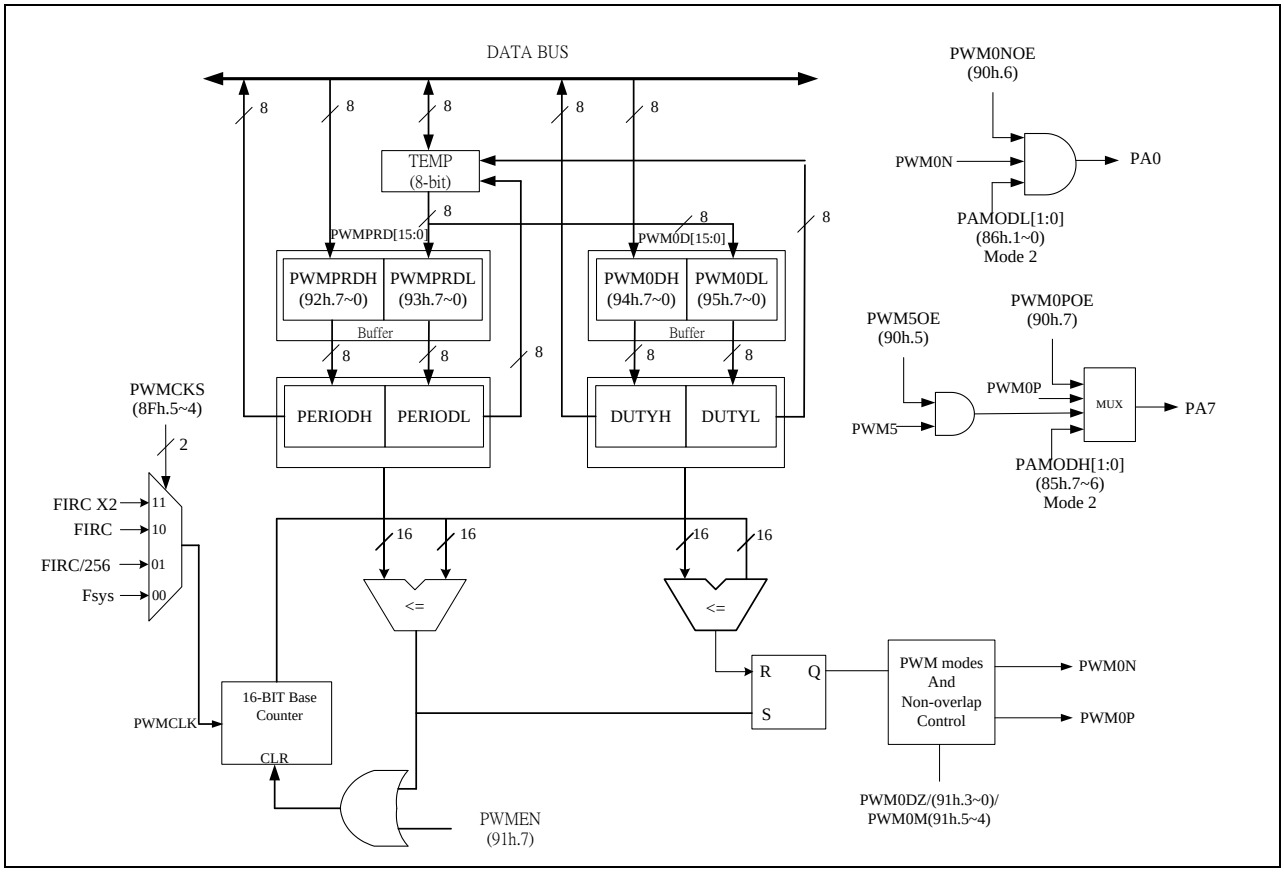
6.5 PWM: Six 16-bit PWM Module

This Chip has six 16-bit PWM modules, PWM0 to PWM5. PWM0~PWM5 have independent 16-bit duty control register, and share a set of 16-bit period register. The PWM can generate various frequency waveforms with 65536 duty resolution based on PWMCLK, which frequency can select Fsys or FIRC or FIRC*2 or FIRC/256, decided by PWMCKS (8Fh.5~4).

The 16-bit PWM period PWMPRD and PWM duty PWM0D~PWM5D registers all have a low and high byte structure. **Writing to these register pairs must be carried out in a specific way. That is, write low byte register first and then writes high byte register**. For example, writing period value to PWMPRDL register (93h) and then PWMPRDH (92h). The PWMPRD will immediately change to the new values when high byte data has been written to the register. Writing PWM0D~PWM5D are the same as writing PWMPRD.

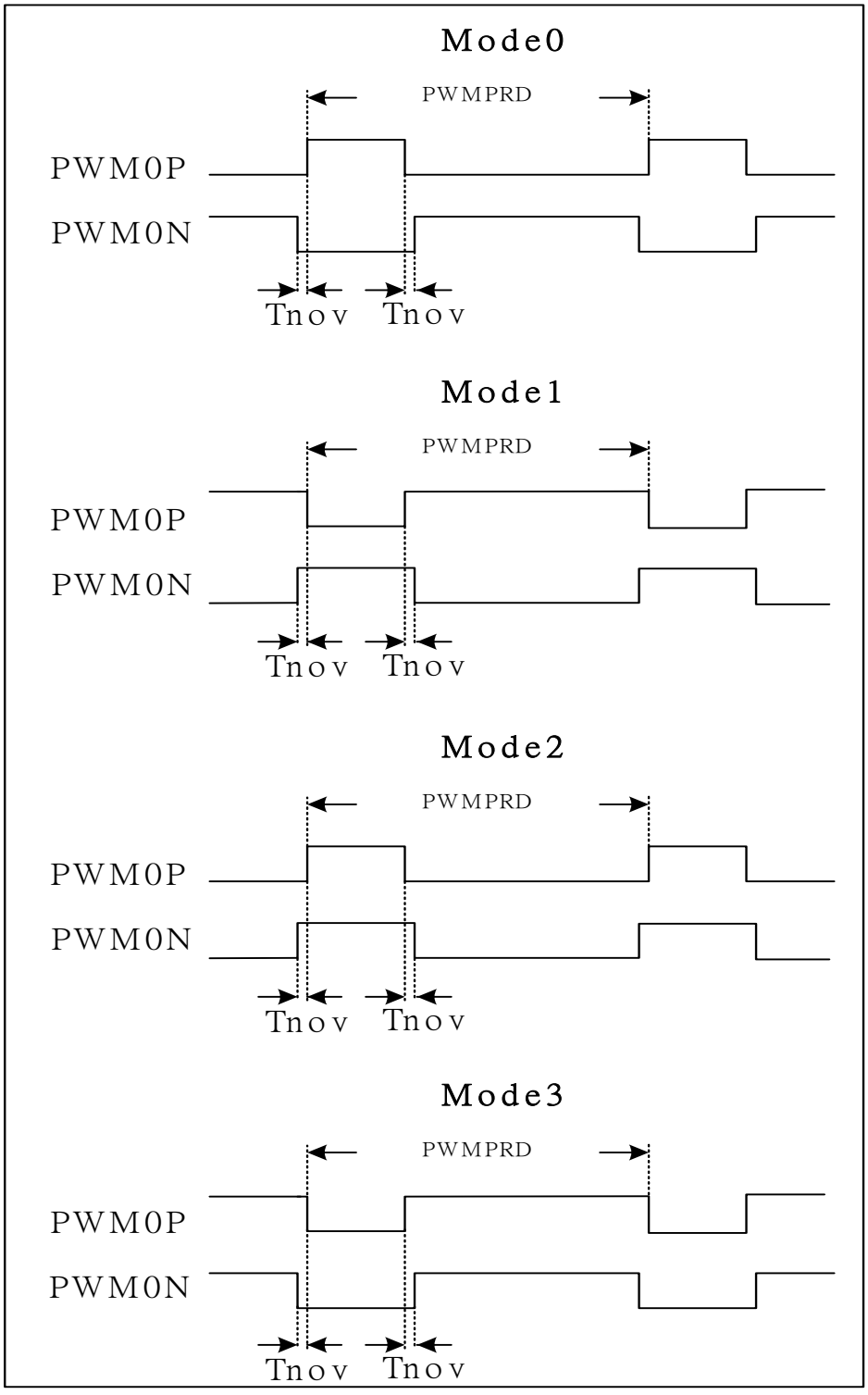
If PWMEN is cleared, the PWM0~5 will be cleared and stopped, otherwise the PWM0~5 remain running. The PWM0 structure is shown as follow. The PWM0 duty cycle can be changed by writing to PWM0DL and PWM0DH. The PWM0 period can be set by writing the period value to PWMPRDL and PWMPRDH register. There is a digital comparator that compares the PWM counter and PWMRD, if PWM counter is larger than PWMPRD the PWM counter will be cleared and PWM output signal will be set to high level. The PWM output signals PWM0~PWM5 reset to low level whenever the 16-bit base counter matches the PWM duty register PWM0D ~PWM5D individually.

Only PWM0 has dead-zone control (PWM0DZ), and is divided into PWM0P and PWM0N outputs with 4 modes (PWM0OM), and the remaining PWM1~5 have no non-overlap control. The PWM1~5 outputs are PWM1O~PWM5O.



PWM0 Block Diagram

Only PWM0 can be output via PWM0P and PWM0N with four different modes control by PWM0OM (91h.5~4). The edges of PWM pulse can be separated with 16 different time non-overlap clock intervals (Tnov). The width of Tnov can be selected by PWM0DZ (91h.3~0) within 0~15 pwm clock. The default output form is Mode0. The waveforms of the four output modes are shown below.



PWM0 Waveform Modes

◇Example: [CPU running at Fast mode, Fsys=FIRC 18.432Mhz]

; Setup PWM0 clock prescaler

```
MOVLW     00000000B     ;
MOVWX     OPTION2       ; PWMCKS=00B, PWM clk source = Fsys=FIRC 18.432MHz
                          ; Tpwmclk=(1/18.432M)
MOVLW     20h
MOVWX     PWMPRDL       ; set PWM period Low byte =20h
                          ; low byte data is stored in TEMP, not in PWMPRD register
MOVLW     03h
MOVWX     PWMPRDH       ; 设置 PWM 周期高字节 =03h
                          ; PWMPRD = 0320h (TEMP 数据也保存到 PWMPRD 中)
MOVLW     90h
MOVWX     PWM0DL        ; 设置 PWM0 周期低字节  =90h
                          ; 低字节数据存储在 TEMP 中，而不是 PWM0D 寄存器中
MOVLW     01h           ;
MOVWX     PWM0DH        ; set PWM0 周期高字节   =01h
                            ; PWM0D = 0190h (TEMP数据也保存到PWM0D中)
MOVLW     10010100B
MOVWX     PWMCTL        ; 设置 PWMEN=1, PWM0OM=10B=Mode 2,
                          ; PMW0DZ=0100B= non-overlap 4*Tpwmclk
MOVLW     10xxxx10B     ;
MOVWX     PAMODL        ; 设置 PA7, PA0 as CMOS 输出
MOVLW     11000000B
MOVWX     PWMOE         ; 使能 PWM0P 输出到 PA7, 使能 PWM0N 输出到 PA0
```

90h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWMOE	PWM0POE	PWM0NOE	PWM5OE	PWM4OE	PWM3OE	PWM2OE	PWM1OE	-
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	-
Reset	0	0	0	0	0	0	0	-

90h.7 **PWM0POE:** PWM0P Output Enable
0: Disable
1: Enable, PWM0P output to PA7

90h.6 **PWM0NOE:** PWM0N Output Enable
0: Disable
1: PWM0N output to PA0

90h.5 **PWM5OE:** PWM5 Output Enable
0: Disable
1: Enable, PWM5 output to PA7; PWM0POE has higher priority to output PA7

90h.4 **PWM4OE:** PWM4 Output Enable
0: Disable
1: Enable, PWM4 output to PA4

90h.3 **PWM3OE:** PWM3 Output Enable
0: Disable
1: Enable, PWM3 output to PA3

90h.2 **PWM2OE:** PWM2 Output Enable
0: Disable
1: Enable, PWM2 output to PA2

90h.1 **PWM1OE:** PWM1 Output Enable
0: Disable
1: Enable, PWM1 output to PA1

91h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWMCTL	PWMEN	-	PWM0OM		PWM0DZ			
R/W	R/W	-	R/W		R/W			
Reset	0	-	0	0	0	0	0	0

91h.7 **PWMEN:** PWM Clock Enable
0: Disable
1: Enable

91h.5~4 **PWM0OM:** PWM0 Output Mode
00: Mode 0
01: Mode 1
10: Mode 2
11: Mode 3

91h.3~0 **PWM0DZ:** PWM0 Dead Zone (non-overlap) control
0000: non-overlap 0*Tpwmclk; Original PWM0
0001: non-overlap 1*Tpwmclk
0010: non-overlap 2*Tpwmclk
……
1111: non-overlap 16*Tpwmclk

92h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWMPRDH	PWMPRDH							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	1	1	1	1	1	1	1

92h.7~0 **PWMPRDH:** PWM period data high byte

93h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWMPRDL	PWMPRDL							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	1	1	1	1	1	1	1

93h.7~0 **PWMPRDL:** PWM period data low byte

94h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWM0DH	PWM0DH							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	0	0	0	0	0	0	0

94h.7~0 **PWM0DH:** PWM0 duty data high byte

95h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWM0DL	PWM0DL							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

95h.7~0 **PWM0DL:** PWM0 duty data low byte

96h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWM1DH	PWM1DH							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	0	0	0	0	0	0	0

96h.7~0 **PWM1DH:** PWM1 duty data high byte

97h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWM1DL	PWM1DL							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

97h.7~0 **PWM1DL:** PWM1 duty data low byte

98h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWM2DH	PWM2DH							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	0	0	0	0	0	0	0

98h.7~0 **PWM2DH:** PWM2 duty data high byte

99h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWM2DL	PWM2DL							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

99h.7~0 **PWM2DL:** PWM2 duty data low byte

9Ah	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWM3DH	PWM3DH							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	0	0	0	0	0	0	0

9Ah.7~0 **PWM3DH:** PWM3 duty data high byte

9Bh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWM3DL	PWM3DL							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

9Bh.7~0 **PWM3DL:** PWM3 duty data low byte

9Ch	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWM4DH	PWM4DH							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	0	0	0	0	0	0	0

9Ch.7~0 **PWM4DH:** PWM4 duty data high byte

9Dh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWM4DL	PWM4DL							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

9Dh.7~0 **PWM4DL:** PWM4 duty data low byte

9Eh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWM5DH	PWM5DH							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	0	0	0	0	0	0	0

9Eh.7~0 **PWM5DH:** PWM5 duty data high byte

9Fh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWM5DL	PWM5DL							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

9Fh.7~0 **PWM5DL:** PWM5 duty data low byte

6.6 UART

TM56F6922 has built-in standard UART (Universal Asynchronous Receiver/Transmitter), which is responsible for performing serial communications among computers. Transmitting device changes the parallel data to serial data and sends the bit-stream data via TX line, the receiving device receives the data via RX line in serial form and converts to parallel data stored in register for MCU reading. The TX/RX module also handles data synchronization, parity bit generation and detection, frame error, overrun error, and interrupts generation. Figure 6.6.1 shows the data format of UART

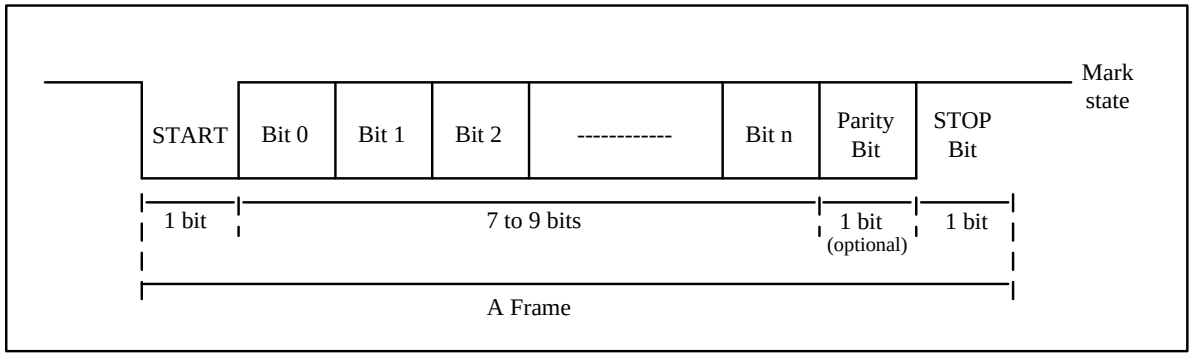


Figure 6.6.1 UART UART data format

The transmission line is normally kept in mark state (logic 1) until the transmission or reception start with a transition to space state (logic 0). The first bit transmitted is START bit and the length is 1 bit wide (1 bps second), followed by the transmitted bit stream of the data. The LSB (least Significant Bit) is sent first, the number or transmitted bit is defined by UMODE control bits that can have 7, 8 or 9 bits mode to be transmitted or received. The parity bit is followed by data bit and is selectable to be sent or not, but in 9-bit mode transmission, the parity bit is always not to be transmitted.

UART Modes

The UAT can operate in one of three modes, e.g. Mode 0 (7-bit data), Mode 1 (8-bit data) and Mode 2 (9-bit data). The data format of Mode 0 and Mode 1 can transmit/receive parity bit according to PRE bit. If PRE=1, the TX data format will add the parity bit after the data bits, otherwise, the TX only transmits the data bits. Mode 2 does not support parity bit transmit/receive. Note that the setting of UART between two communicating devices must be the same so that the RX part can check the data format and parity the same as TX part. The parity select bit is EVEN bit in UARTS control register. It uses even parity if EVEN=1, else it uses odd parity. Every UART transfer is ended with a STOP bit. If the STOP bit is not seen by RX part, the RX part will set the FMERR bit in UARTS control register to indicate the transfer may not be properly received/transmitted, it must be re-sent again or firmware handling. Figure 6.6.2 shows the data formats of three UART modes.

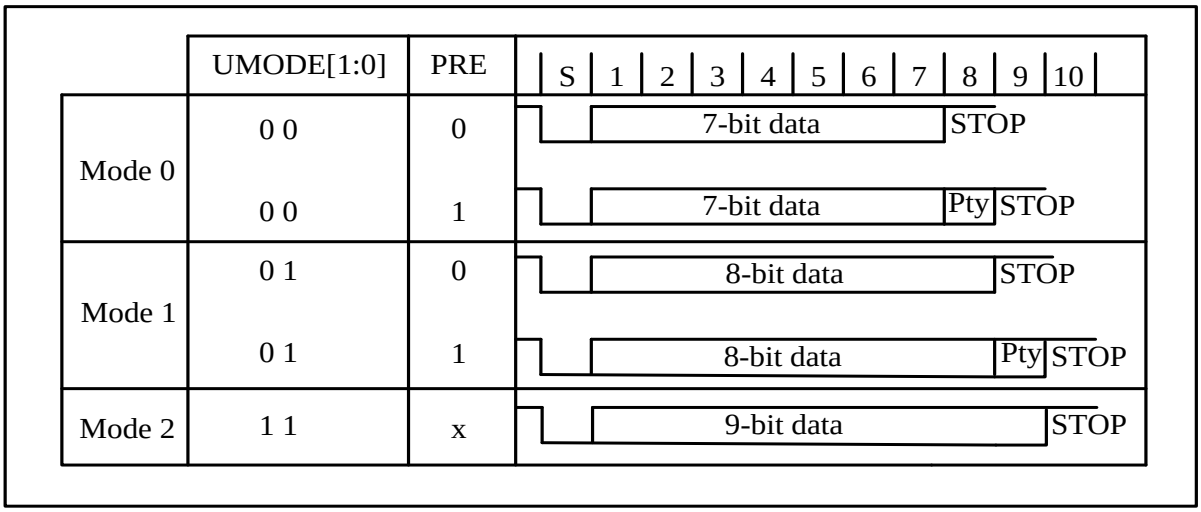


Figure 6.6.2 UART Modes

UART System Block

Figure 6.6.3 shows the system block of the UART built in TM56F1543. The Baud Clock Generator divides the Fsys to produce the proper frequency of the UART baud rate. The divisor is UBAUD register. Every bit of data stream is sampled by 16 Baud clocks which are generated from Baud Clock Generator. The desired Baud rate can be achieved by setting UBAUD with the following equation:

$$\text{Baud Rate} = \frac{\text{Fsys}}{32*\text{UBAUD}}$$

The divided clock (Baud clock) is then sent to TX and RX control logic and TX/RX shift buffers. The TX control logic start sending data out to TX pin (PB5) once the URDATA is written, URTD9 will be sent if the UART mode is Mode 2. When transmit completes, the “TX buffer Empty” signal is asserted, then interrupt will be generated if INTIE1.UARTIE is set. UINVEN will invert the output when it is set to 1, the RX will be inverted when UINVEN is set.

The UART RX signal is input from RX pin (PB4) and then it is inverted if UINVEN is set, passing through a synchronization circuit then clocked to RX shifter. Number of bits to be shifted into RX shifter depends on what UART Mode is operated, and then RX control logic handle them, including frame error detection (FMERR bit), parity detection (PRERR) and overrun error (OVERR) detection. Frame error means that the incoming frame STOP bit is not detected. Parity error means the incoming data parity bit does not equal to the parity of data bits that are evaluated by the RX control logic. Overrun error means the previous transfer that is stored in URDATA has not been read as a new incoming transfer is completed, the old URDATA will be lost when overrun error occurs.

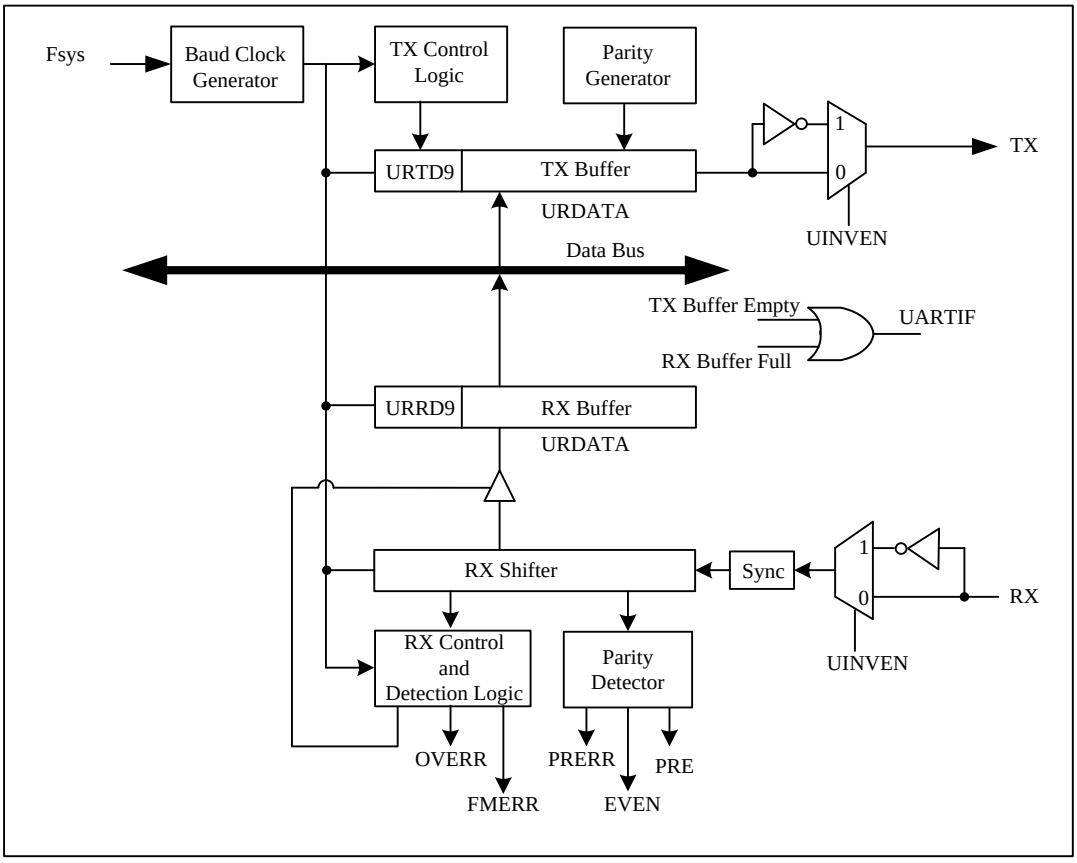


Figure 6.6.3 UART Block Diagram

Table 6.6. lists the common used operating frequency to generate common used Baud Rates under Fsys is 18.432MHz. Some Baud Rates cannot be achieved because frequency error is too large to use.

Fsys (Hz)	Desired Baud Rate (bps)	UBAUD	Actual Generate Baud Rate	Frequency Error(%)	
18432000	2400	240	2400	0	
18432000	4800	120	4800	0	
18432000	9600	60	9600	0	
18432000	14400	40	14400	0	
18432000	19200	30	19200	0	
18432000	28800	20	28800	0	
18432000	38400	15	38400	0	
18432000	57600	10	57600	0	
18432000	76800	8	72000	6.25%	Don’t use
18432000	76800	7	82285.7	7.14%	Don’t use
18432000	115200	5	115200	0	
18432000	192000	3	192000	0	

Table 6.6 UART Baud Rates setting

◇Example:

[CPU running at FAST mode , Fsys=FIRC 18.432 MHz]

UART Baud Rate = 115200 bps, RX pin (PB4), TX pin (PB5).
Transmit 91H data to TX (PB5)

◇Example:

```
BCX     FASTSTP     ; Fsys=18.432 MHz
BSX     CPUCKS      ;

MOVLW   1001 0101B  ; RX (PB4) Pin Mode=1, TX (PB5) Pin Mode=2
MOVWX   PAMODL;

MOVLW   00000101B
MOVWX   UBAUD       ; 115200 bps (@Fsys=18.432MHz)

BSX     UARTIE      ; enable UART interrupt
MOVLW   00101010B   ;
MOVWX   UARTCTL     ; 8bit mode, UTXE=1, URXE=0, UARTE=1, UINVEN=0
MOVLW   10010001B   ;
MOVWX   URDATA      ; TX Buffer=91h, and then H/W send to TX pin via TX shifter

BTXSS   UTBE
LGOTO   $-1         ; check TX buffer until buffer is empty (transmit finished)
……
…...
```

0Eh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTIE1	LVDIE	EEPIE				I2CIE	UARTIE	PXIE
R/W	R/W	R/W				R/W	R/W	R/W
Reset	0	0				0	0	0

INTIE1.1 **UARTIE:** UART TX/RX complete interrupt enable
0: disable
1: enable

1Ah	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTIF1	LVDIF	EEPIF				I2CIF	UARTIF	PXIF
R/W	R/W	R/W				R/W	R/W	R/W
Reset	0	0				0	0	0

1Ah.1 **UARTIF:** UART interrupt pending flag
This bit is set by H/W while TX/RX transfer is completed; S/W write FDh to INTIF1 to clear this flag

10Dh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
UBAUD	UBAUD							
R/W	R/W							
Reset	0	0	0	0	0	0	0	0

10Dh **UBAUD:** UART Baud Rate Divider

187h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
UARTCTL	URTD9	UMODE		URINTTF	UTXE	URXE	UARTE	UINVEN
R/W	R/W	R/W	R/W	R	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

187h.7 **URTD9:** UART Transmitted 9th bit
187h.6~5 **UMODE:** UART Mode Selection
00: 7-bit
01: 8-bit
10: 9-bit
11: Reserved

187h.4 **URINTTF:** UART TXTransfer completion flag
This bit is set to 1 by H/W when UART TX transmission is completed; clearing UARTIF will also clear this flag

187h.3 **UTXE:** UART transmission enable
0: Disable
1: Enable

187h.2 **URXE:** UART reception enable
0: Disable
1: Enable

187h.1 **UARTE**: UART function enable
0: Disable
1: Enable

187h.0 **UINVEN**: UART TX/RX output/input Inversion enable
0: Disable
1: Enable

188h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
UARTSTA	UTBE	URRD9	EVEN	PRE	PREERR	OVERR	FMERR	URBF
R/W	R	R	R/W	R/W	R/W	R/W	R/W	R
Reset	0	0	0	0	0	0	0	0

188h.7 **UTBE:** UART Transmitting status
0: TX is transmitting
1: TX buffer is empty

188h.6 **URRD9**: UART Received 9th bit

188h.5 **EVEN:** UART parity
0: ENEN
1: ODD

188h.4 **PRE:**UART UART Parity Addition
0: Disable
1: Enable

188h.3 **PRERR**: UART Parity Error
Set to 1 when parity error occurs, clear to 0 by S/W

188h.2 **OVERR**: UART Overrun Error
Set to 1 when overrun error occurs, clear to 0 by S/W

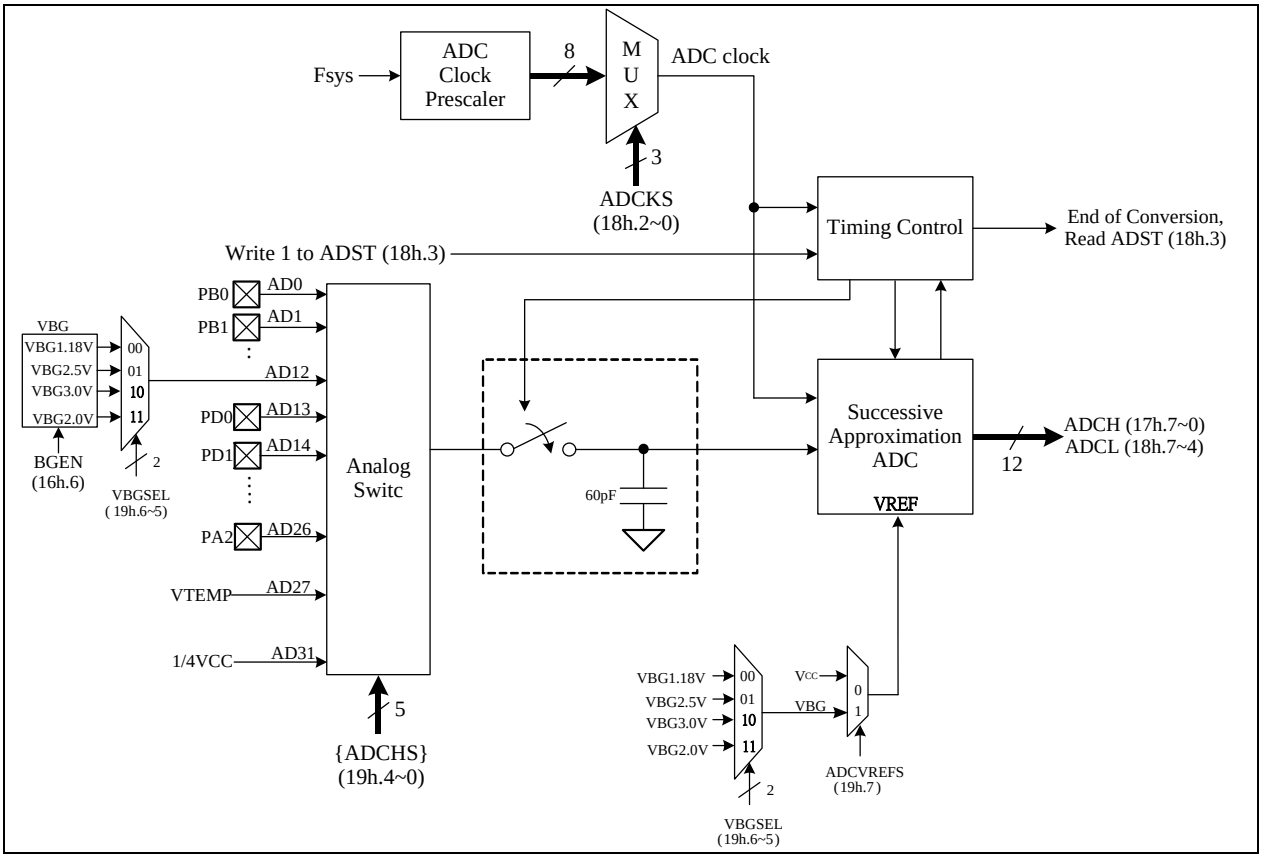
188h.1 **FMERR:** UART Frame Error
Set to 1 when frame error occurs, clear to 0 by S/W

188h.0 **URBF:** UART buffer status
Set to 1 when 1 byte (or 9 bits) is received
S/W read URDATA will clear to 0 automatically and then to receive the succeeding data without overrun error

18Dh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
URDATA	URDATA							
R/W	R/W							
Reset	0	0	0	0	0	0	0	0

18Dh **URDATA:** UART Transmission/Reception Data Buffer
write: to transmit buffer
read: from receive buffer

6.7 Analog-to-Digital Converter

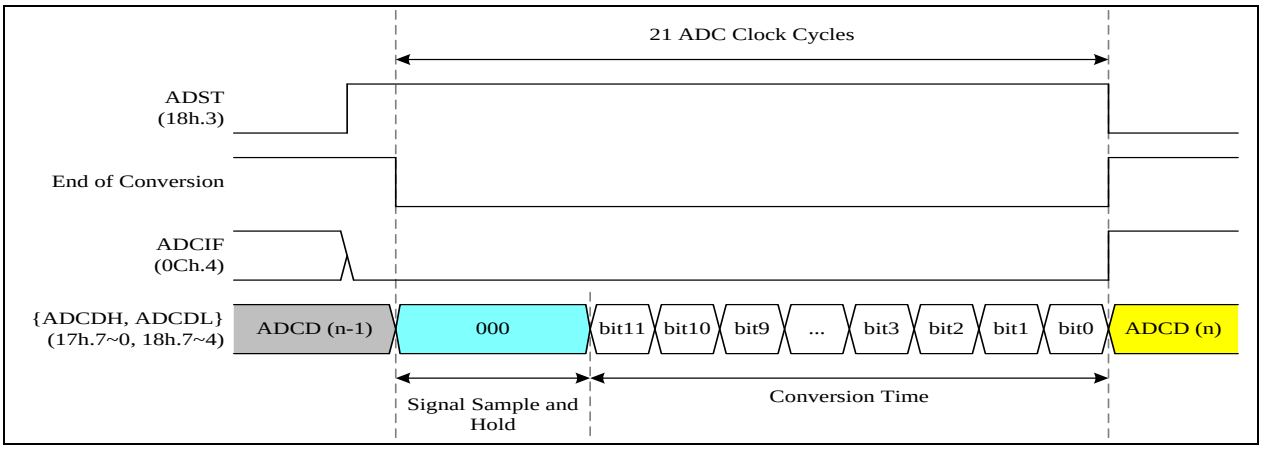


The 12-bit ADC (Analog-to-Digital Converter) consists of a 14-channel analog input multiplexer, control register, clock generator, 12-bit successive approximation register, and output data register. To use the ADC, the user needs to set ADCKS (18h.2~0) to select the appropriate ADC clock frequency, which must be less than 1MHz. Then, the user starts the ADC conversion by setting the ADST (18h.3) control bit. After the conversion is completed, the H/W automatically clears the ADST (18h.3) bit. The user can poll this bit to know the conversion status. The PxMODx control register is used for ADC pin configuration. When the pin is used as ADC input, the user must set the pin mode = 3. This setting can disable the pin logic input path to save power. The user needs to set {ADCHS} (19h.4~0) to select the input channel of the ADC. Among them, AD14 is the ADC VBG input and AD31 is the ADC 1/4VCC input. VBG is enabled by BGEN control (16h.6). ADC reference voltage is selected by ADCVREFS (19h.7) and VBGSEL (19h.6~5).

TM56F6922 has a built-in temperature sensor that can generate a VTEMP voltage with a positive temperature coefficient. After the temperature sensor channel is enabled (ADCHS=1Bh), a delay of 100uS is required before ADC sampling and conversion is performed and then this VTEMP voltage is converted into 12-bit digital information. Before the chip leaves the factory, TM56F6922 performs ADC conversion on the VTEMP voltage of VREF=3V (VBGSEL=2'b10) at room temperature and stores it in the INFO area. After power-on reset, this 12-bit converted digital information is copied to VTEMPDH (19Fh.7~0) and VTEMPDL (19Eh.7~4).

Users can use the following formula to convert the temperature value

$$\textbf{Temperature} = 25 + \frac{\{ADCDH, ADCDL\} - \{VTEMPDH, VTEMPDL\}}{6.9}$$



◇Example:

[CPU running at FAST mode , Fsys=FIRC 18.432 MHz]

ADC clock frequency=FIRC/32, ADC channel=AD10 (PC2).

◇Example:

```
MOVLW   00000111B       ; Fsys=18.432 MHz
MOVWX   CLKCTL          ;

MOVLW   0111 0101B      ; AD10 (PC2) Pin Mode=3=ADC input
MOVWX   PCMODL;

MOVLW   00000011B       ; Fsys=18.432 MHz
MOVWX   ADCTL           ; 18h.2~0 (ADCKS) =ADC clock=Fsys/32=576KHz

MOVLW   01110011B       ; 16h.6=1, enable VBG;
MOVWX   LVCTL           ;

MOVLW   1 00 01010B     ; 19h.7=1; 19h.6~5=10, VBG=3.0V
MOVWX   ADCTL2          ; 19h.4~0 (ADCHS [4:0]) =1010, ADC select AD10 (PC2 pin).

 BSX    ADST            ; 18h.3 (ADST) , ADC start conversion

WAIT_ADC:
        BTXSC   ADST            ; Wait ADC conversion finish.
        LGOTO   WAIT_ADC

        MOVXW   ADCDH           ; 17h.7~0, Read ADC result [11:4] 到 W
        …
        MOVXW   ADCTL           ; 18h.7~4, Read ADC result [3:0] 到 W
        …
```

16h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
LVCTL	LVDF	BGEN	LVRSAV	LVDSAV	LVDS			
R/W	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	-	1	1	1	0	0	0	1

16h.6 **BGEN:** Band Gap BG enable
0: Disable,
1: Disable

17h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
ADCDH	ADCDH							
R/W	R	R	R	R	R	R	R	R
Reset	–	–	–	–	–	–	–	–

17h.7~0 **ADCDH:** ADC output data MSB, ADCQ [11:4]

18h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
ADCTL	ADCDL				ADST	ADCKS		
R/W	R	R	R	R	R/W	R/W	R/W	R/W
Reset	–	–	–	–	0	0	0	0

18h.7~4 **ADCDL:** ADC output data LSB, ADCQ [3:0]
18h.3 **ADST:** ADC start bit。
0: H/W clear after end of conversion
1: ADC start conversion
18h.2~0 **ADCKS:** ADC clock frequency selection:
000: Fsys/256 100: Fsys/16
001: Fsys/128 101: Fsys/8
010: Fsys/64 110: Fsys/4
011: Fsys/32 111: Fsys/

19h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
ADCTL2	ADCVREFS	VBGSEL		ADCHS				
R/W	R/W	R/W		R/W				
Reset	0	0			0	0	0	0

F19h.7 **ADCVREFS:** ADC reference voltage select
0: VCC
1: VBG
F19h.6~5 **VBGSEL:** VBG Voltage Selection
00: 1.18V
01: 2.5V
10: 3.0V
11: 2.0V
F19h.4~0 **ADCHS:** ADC channel select

00000: AD0 (PB0)	01000: AD8 (PC0)	10000: AD16 (PD3)	11000: AD24 (PA1)
00001: AD1 (PB1)	01001: AD9 (PC1)	10001: AD17 (PD4)	11001: AD25 (PA0)
00010: AD2 (PB2)	01010: AD10 (PC2)	10010: AD18 (PD5)	11010: AD26 (PA2)
00011: AD3 (PB3)	01011: AD11 (PC3)	10011: AD19 (PD6)	11011: AD27(VTEMP)
00100: AD4 (PB4)	01100: AD12 (VBGO)	10100: AD20 (PD7)	11100: reserved
00101: AD5 (PB5)	01101: AD13 (PD0)	10101: AD21 (PA7)	11101: reserved
00110: AD6 (PB6)	01110: AD14 (PD1)	10110: AD22 (PA4)	11110: reserved
00111: AD7 (PB7)	01111: AD15 (PD2)	10111: AD23 (PA3)	11111: 1/4 VCC

19Eh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
VTEMPDL	VTEMPDL							
R/W	R/W							
Reset	–	–	–	–	–	–	–	–

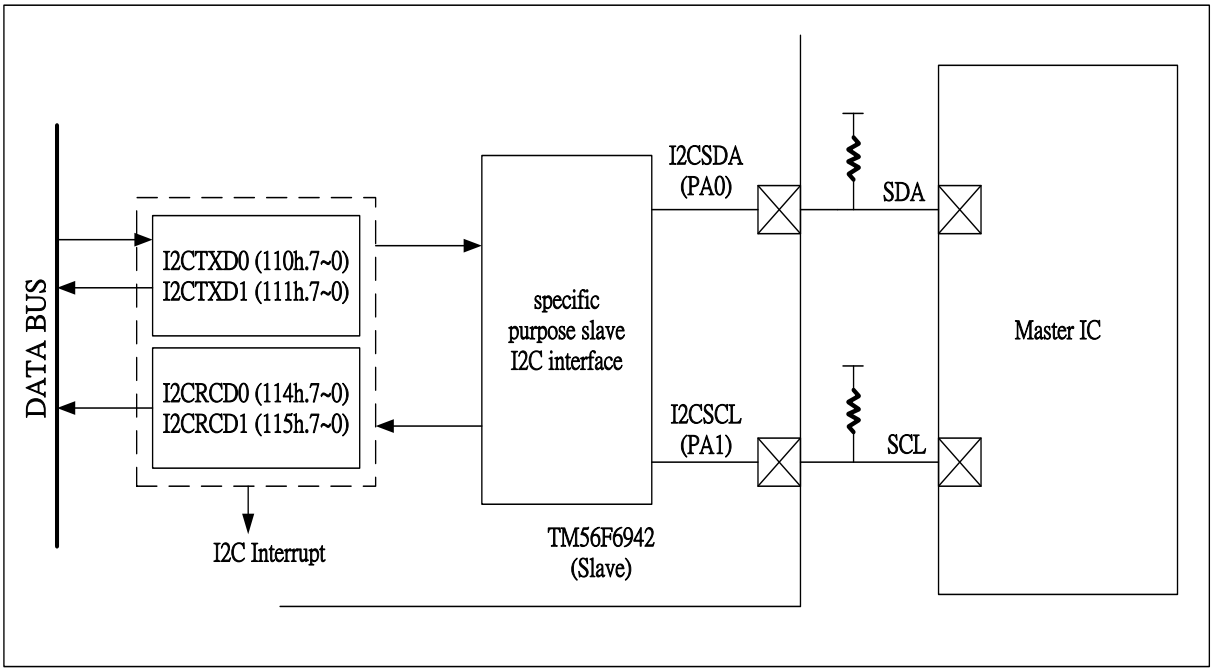
19Eh.2~0 VTEMPDL: ADC VTEMP[3:0] : Factory read value

19Fh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
VTEMPDH	VTEMPDH							
R/W	R/W							
Reset	–	–	–	–	–	–	–	–

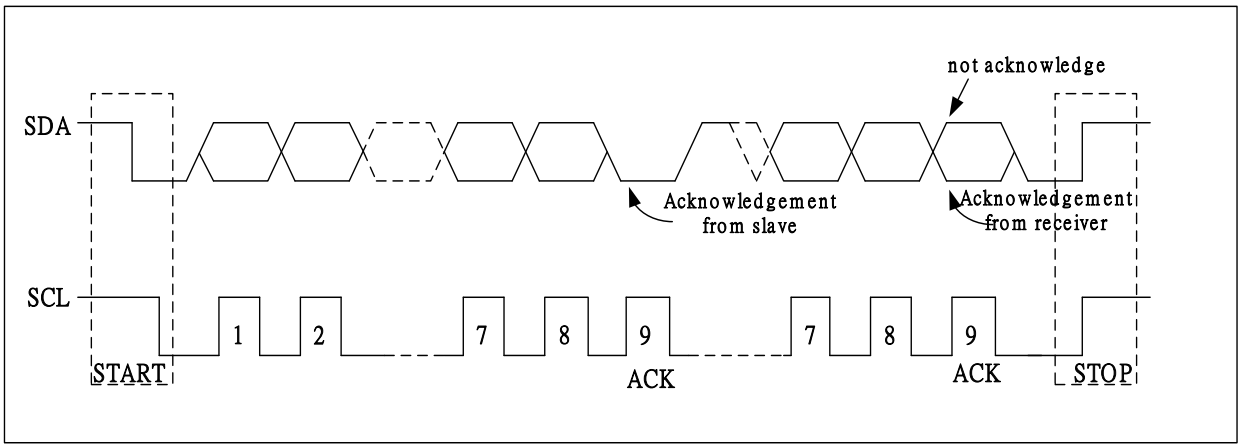
19Fh.7~0 VTEMPDH: ADC VTEMP[11:4]: Factory read value

6.8 Specific Purpose Slave I2C Interface

The dedicated slave I2C interface in TM56F6922 can be used for data transfer. This interface is based on standard I2C (Inter-Integrated Circuit), and TM56F6922 always works as slave mode. When the master node (another IC or device) sends the correct ID via I2C, it can read data from registers I2CTXD0 (110h.7~0) and I2CTXD1 (111h.7~0) of TM56F6922, and can also write data from registers I2CRCD0 (1114h .7~0) and I2CRCD1 (115h.7~0) of TM56F6922.

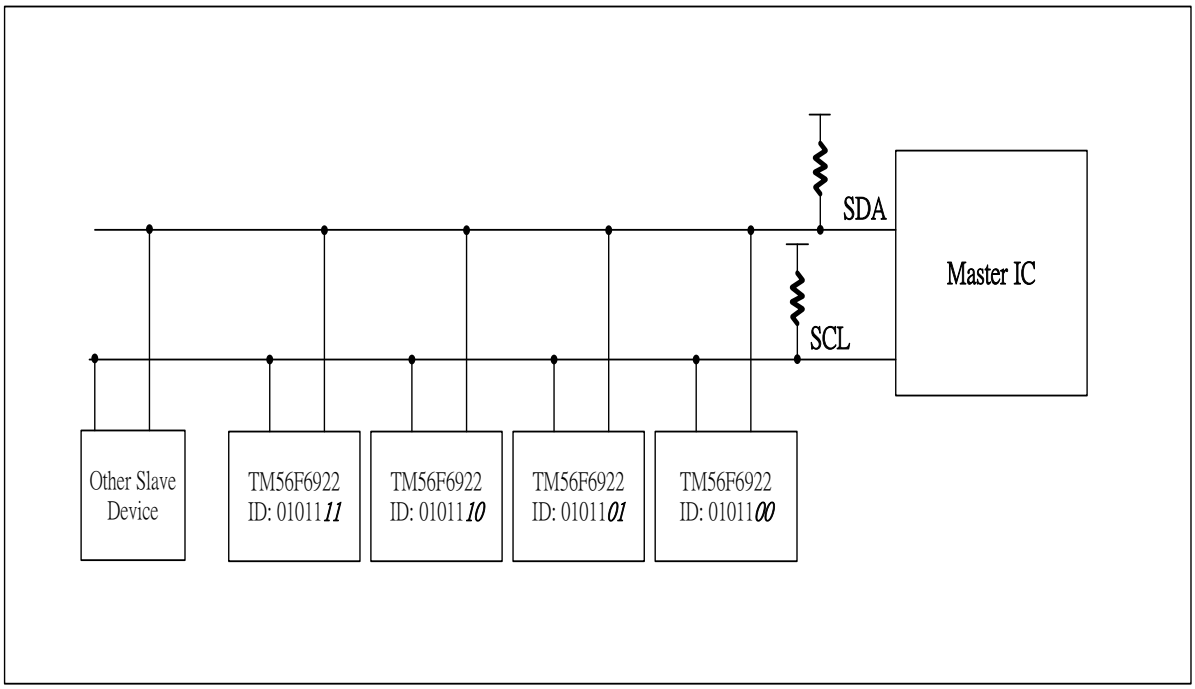


Slave I2C Interface Block Diagram

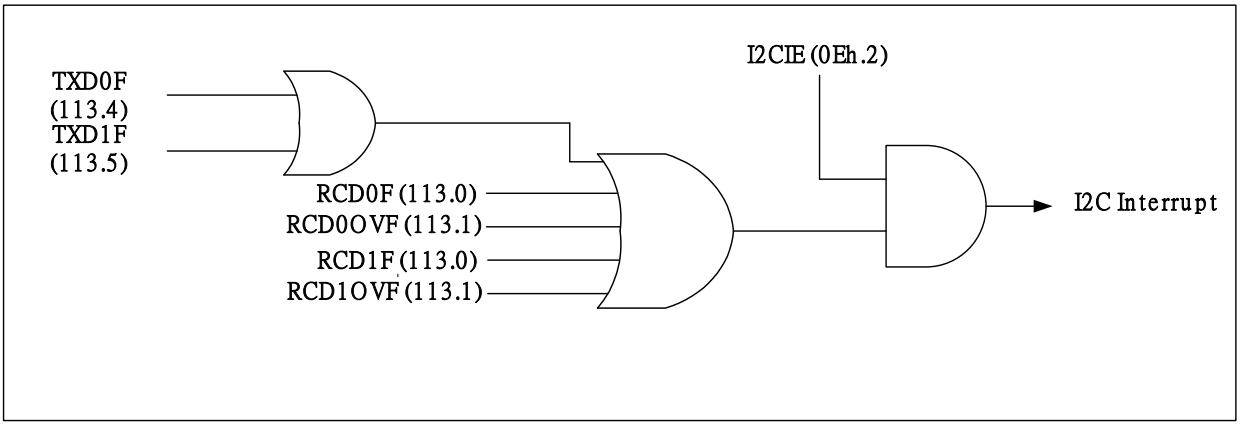


I2C Protocol

To use the slave I2C interface, the I2C (112h.3) bit must be set. TM56F6922 supports 4 slave device IDs by setting I2CID (112.1~0). TM56F6922 can generate transmit flags TXD0F (113h.4) and TXD1F (113h.5) when data transmission is completed. When data reception is completed, receive flags RCD0F (113h.0) and RCD1F (113h.2) are generated. It can also generate receive overflow flags RCD0OVF (113h.1) and RCD1OVF (113h.3) when data reception is completed but the receive flag is not cleared. If one of the I2C flags is set, the I2C interrupt flag I2CIF (1Ah.2) will be generated. If the I2CIE (0Eh.2) bit is set, an I2C interrupt is generated. Refer to the following table and figure

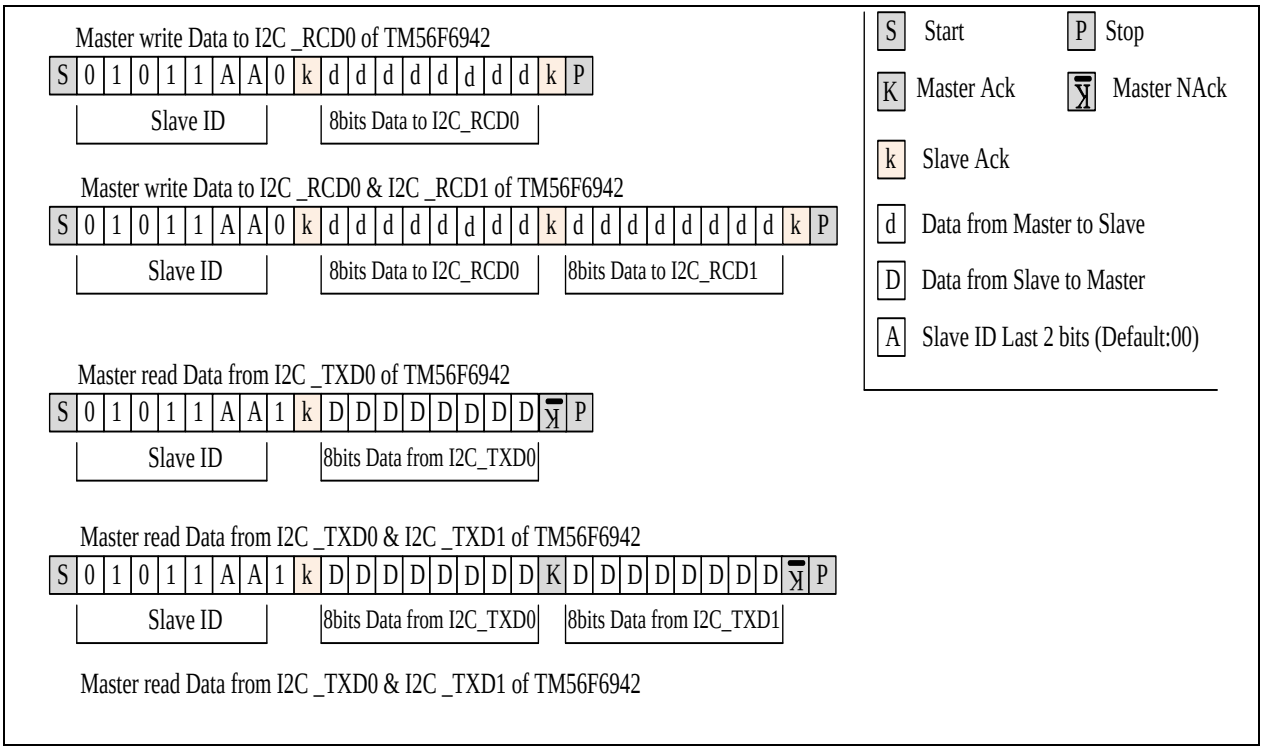


I2C Parallel Connection Application Circui



Slave I2C Interrupt Block Diagram

RCDxOVF	RCDxF	I2CIF	STATE
0	0	0	IDLE
0	1	1	Date received to I2CRCDx register
1	1	1	Data overflow occurred at I2CRCDx register



TM56F6922 I2C Commands

0Eh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTIE1	LVDIE					I2CIE	UARTIE	PXIE
R/W	R/W					R/W	R/W	R/W
Reset	0					0	0	0

INTIE1.2 **I2CIE:** Slave I2C interrupt enable
0:Disable
1:Enable

1Ah	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTIF1	LVDIF	EEPIF				I2CIF	UARTIF	PXIF
R/W	R/W	R/W				R/W	R/W	R/W
Reset	0	0				0	0	0

1Ah.2 **I2CIF:** Slave I2C interrupt pending flag; refer to 113h (I2CFLG)
This bit is set by H/W while

- ✧ I2CRCD0 or I2CRCD1 receive data finished
- ✧ I2CRCD0 or I2CRCD1 data overflow occurred
- ✧ I2CTXD0 or I2CTXD1 data transmit finished

S/W writes FBh to INTIF1 to clear this flag and slave I2C related flags.

110h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
I2CTXD0	I2CTXD0							
R/W	R/W							
Reset	0	0	0	0	0	0	0	0

110h.7~0 **I2CTXD0:** The transmitting register 0 of slave I2C

111h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
I2CTXD1	I2CTXD1							
R/W	R/W							
Reset	0	0	0	0	0	0	0	0

111h.7~0 **I2CTXD1:** The transmitting register 1 of slave I2C

112h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
I2CCTL	-	-	-	-	I2CEN	-	I2CID	
R/W	-	-	-	-	R/W	-	R/W	
Reset	-	-	-	-	0	-	0	0

112h.3 **I2CEN:** Slave I2C interface enable
0:Disable
1:Enable

112h.1~0 **I2CID:** Slave I2C ID last 2 bits

113h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
I2CFLG	-	-	TXD1F	TXD0F	RCD1OVF	RCD1F	RCD1OVF	RCD0F
R/W	-	-	R/W	R/W	R/W	R/W	R/W	R/W
Reset	-	-	0	0	0	0	0	0

113h.5 **TXD1F:** Slave I2C transmitting data register 1 flag
This bit is set by H/W while I2CTXD1 data transmitting finished, S/W writes DFh to I2CFLG to clear this flag

113h.4 **TXD0F:** Slave I2C transmitting data register 0 flag
This bit is set by H/W while I2CTXD0 data transmitting finished, S/W writes EFh to I2CFLG to clear this flag

113h.3 **RCD1OVF:** Slave I2C receiving data register 1 overflow
This bit is set by H/W while receiving data to I2CRCD1 overflow, S/W writes F7h to I2CFLG to clear this flag

113h.2 **RCD1F:** Slave I2C receiving data register 1 flag
This bit is set by H/W while data receiving to I2CRCD1 finished, S/W writes FBh to I2CFLG to clear this flag

113h.1 **RCD0OVF:** Slave I2C receiving data register 0 overflow
This bit is set by H/W while receiving data to I2CRCD0 overflow, S/W writes FDh to I2CFLG to clear this flag

113h.0 **RCD0F:** Slave I2C receiving data register 0 flag
This bit is set by H/W while data receiving to I2CRCD0 finished, S/W writes FEh to I2CFLG to clear this flag

114h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
I2CRCD0	I2CRCD0							
R/W	R/W							
Reset	0	0	0	0	0	0	0	0

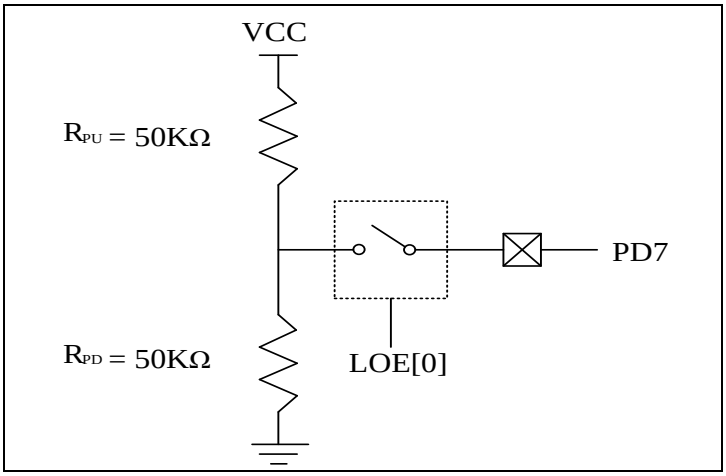
114h.7~0 **I2CRCD0:** The receiving register 0 of slave I2C

115h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
I2CRCD1	I2CRCD1							
R/W	R/W							
Reset	0	0	0	0	0	0	0	0

115h.7~0 **I2CRCD1:** The receiving register 1 of slave I2C

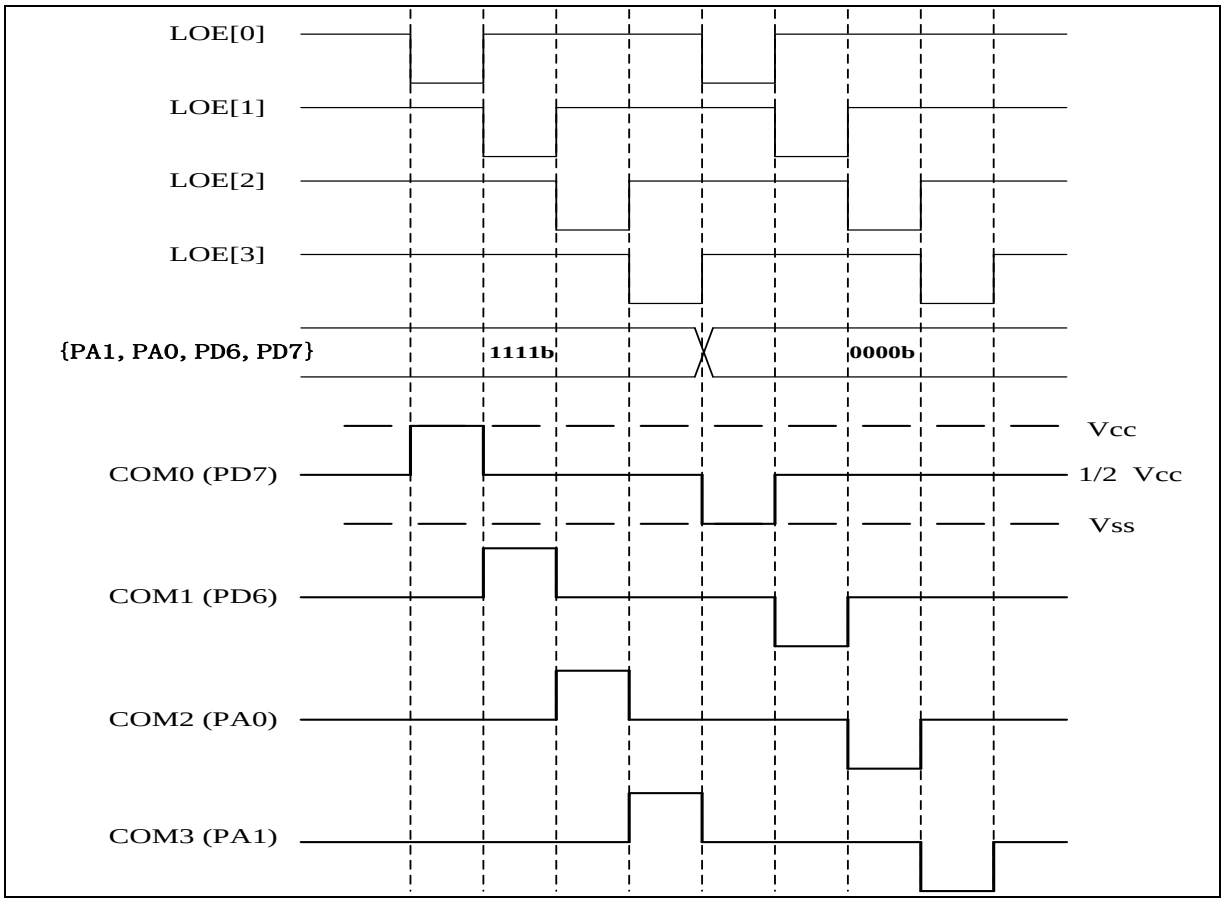
6.9 S/W Control LCD Driver

The chip support an S/W controlled method to driving LCD. Four IO pins can be the Common Pins. The four pin are PA0, PA1, PD6 and PD7. Common pins are capable of driving 1/2 bias by setting the register LOE. The COM0 circuit is shown below.

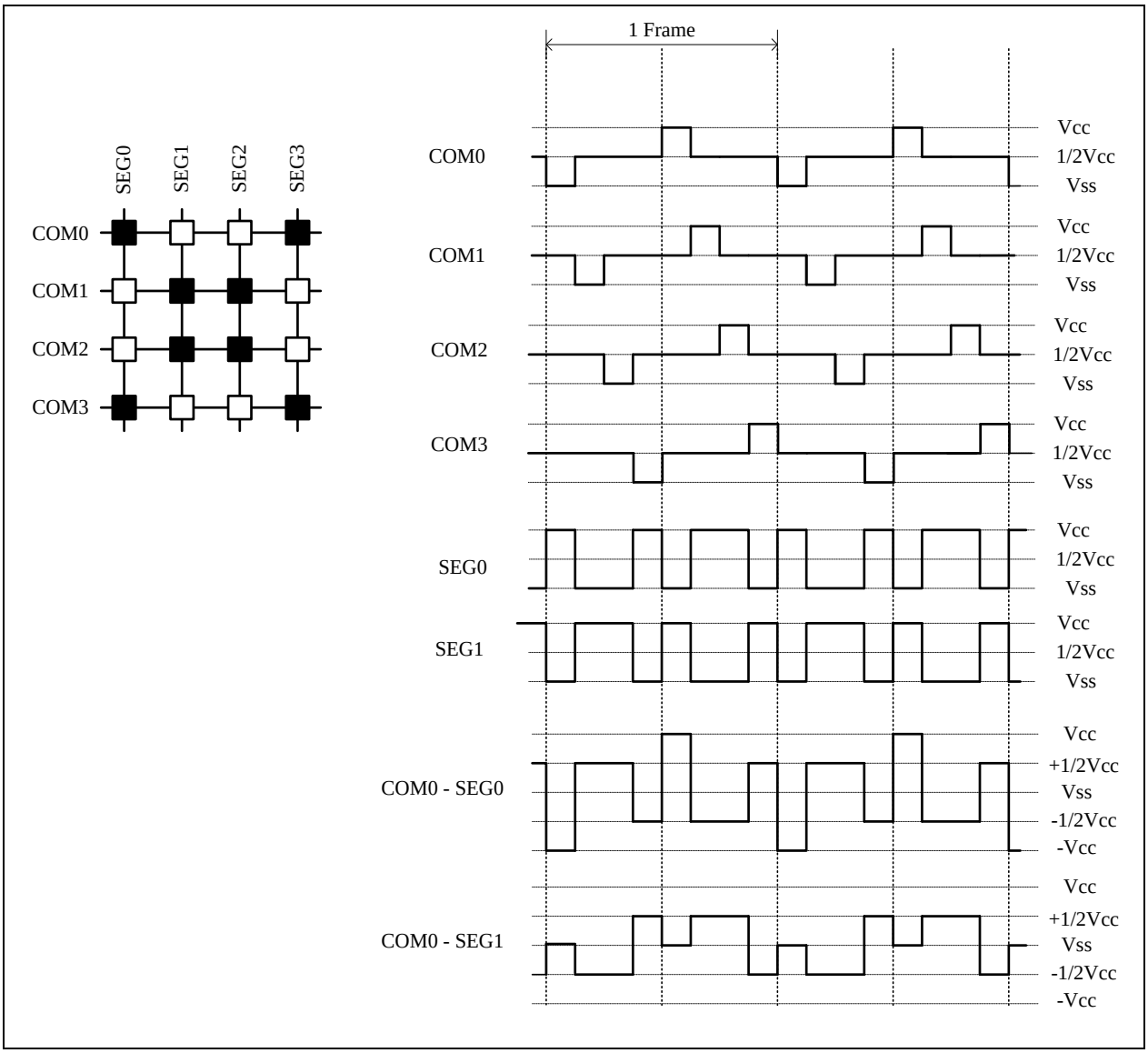


LCD COM0 Circuit

The frequency of any repeating waveform output on the COM pin can be used to represent the LCD frame rate. The figure below shows an LCD frame.



S/W Controlled LCD COM0 ~ COM3 Scanning

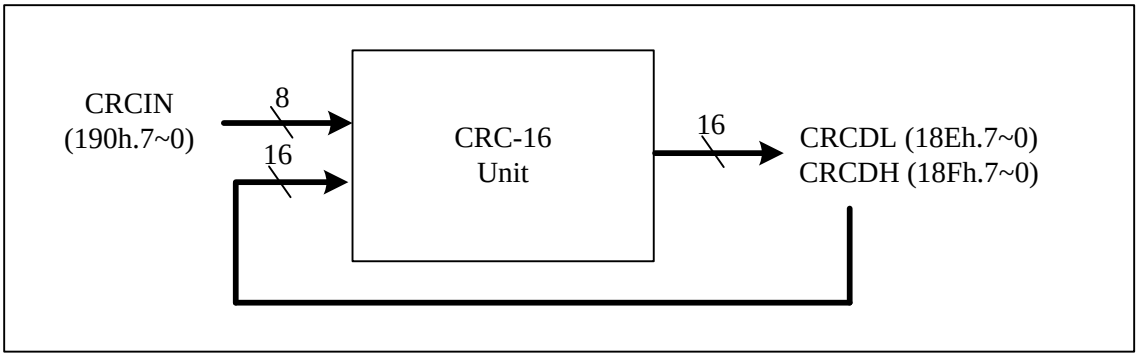


108h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
LOE					LOE			
R/W					R/W			
Reset					0	0	0	0

108h.3~0 **LOE:** COM0~COM3 LCD 1/2 bias output enable control
0001: COM0 (PD7) Enable
0010: COM1 (PD6) Enable
0100: COM2 (PA0) Enable
1000: COM3 (PA1) Enable

6.10 Cyclic Redundancy Check (CRC)

The chip supports an integrated 16-bit Cyclic Redundancy Check function. The Cyclic Redundancy Check (CRC) calculation unit is an error detection technique test algorithm and uses to verify data transmission or storage data correctness. The CRC calculation takes an 8-bit data stream or a block of data as input and generates a 16-bit output remainder. The data stream is calculated by the same generator polynomial.



CRC16 Block Diagram

The CRC generator provides the 16-bit CRC result calculation based on the CRC-16-IBM polynomial. In this CRC generator, there is only one polynomial available for the numeric values calculation. It can't support the 16-bit CRC calculations based on any other polynomials. Each write operation to the CRCIN register creates a combination of the previous CRC value stored in the CRCDH and CRCDL registers. It will take one MCU instruction cycle to calculate.

CRC-16-IBM (Modbus) Polynomial representation: $X^{16} + X^{15} + X^2 + 1$

18Eh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
CRCDL	CRCDL							
R/W	R/W							
Reset	1	1	1	1	1	1	1	1

18Eh.7~0 **CRCDL**: 16-bit CRC checksum data bit 7~0

18Fh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
CRCDH	CRCDH							
R/W	R/W							
Reset	1	1	1	1	1	1	1	1

18Fh.7~0 **CRCDH:** 16-bit CRC checksum data bit 15~8

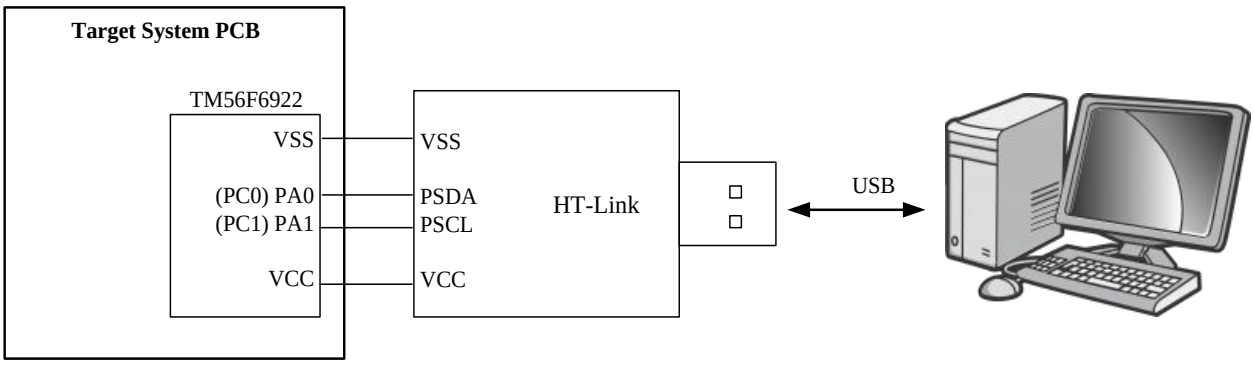
190h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
CRCIN	CRCIN							
R/W	W							
Reset	-	-	-	-	-	-	-	-

190h.7~0 CRC data input, write this register to start CRC calculation

6.11 In Circuit Emulation (ICE) Mode

This device can support the In Circuit Emulation Mode. To use ICE Mode, user needs to connect PA0 and PA1 pin to the tenx proprietary EV Module. The benefit is that user can emulate the whole system without changing the on board target device. But there are some limits for the ICE mode as below.

- ✧ The device must be un-protect.
- ✧ The device's PA0 and PA1 pins must work in input Mode.
- ✧ The HT-Link communication Pin's function cannot be emulated.
- ✧ The PA0 and PA1 pins can be replaced by PC0 and PC1 (only in ICE Mode)
- ✧ The VCC level is controlled by HT-Link module.



MEMORY MAP

Name	Address	R/W	Rst	Description
INDF (00h/80h/100h/180h)				**Function related to: RAM W/R**
INDF	00.7~0	R/W	-	Not a physical register, addressing INDF actually point to the register whose address is contained in the FSR register
TM0 (01h/101h)				**Function related to: Timer0**
TM0	01.7~0	R/W	0	Timer0 content
PCL (02h/82h/105h/182h)				**Function related to: PROGRAM COUNT**
PCL	02.7~0	R/W	0	Programming Counter LSB [7~0]
STATUS (03h/83h/103h/183h)				**Function related to: STATUS**
IRP	03.7	R/W	0	Register Bank Select bit (used for indirect addressing)
RP1	03.6	R/W	0	Register Bank Select bit 1 (used for direct addressing)
RP0	03.5	R/W	0	Register Bank Select bit 0 (used for direct addressing)
TO	03.4	R	0	WDT timeout flag, cleared by PWRST, ‘SLEEP’ or ‘CLRWDT’ instruction
PD	03.3	R	0	Power down flag, set by ‘SLEEP’, cleared by ‘CLRWDT’ instruction
Z	03.2	R/W	0	Zero flag
DC	03.1	R/W	0	Decimal Carry flag
C	03.0	R/W	0	Carry flag
FSR (04h/84h/104h/184h)				**Function related to: RAM W/R**
FSR	04.7~0	R/W	-	File Select Register, indirect address mode pointer
PAD (05h)				**Function related to: Port A**
PAD	05.7~0	R	-	Port A pin or “data register” state
		W	FF	Port A output data register
PBD (06h)				**Function related to: Port B**
PBD	06.7~0	R	-	Port B pin or “data register” state
		W	FF	Port B output data register
PCD (07h)				**Function related to: Port C**
PCD	07.3~0	R	-	Port C pin or “data register” state
		W	FF	Port C output data register
PDD (08h)				**Function related to: Port D**
PDD	08.7~0	R	-	Port D pin or “data register” state
		W	FF	Port D output data register
INDF1 (09h/89h/109h/189h)				**Function related to: RAM (BANK4/5) W/R**
INDF1	09.7~0	R/W	0	Not a physical register, addressing INDF1 actually point to the register whose address is contained in the FSR1 register
PCLATH (0Ah/8Ah/10Ah/18Ah)				**Function related to: PROGRAM COUNT**
PCLATH	0A.3~0	R/W	0	Write Buffer for the upper 4 bits of the Program Counter
INTIE (0Bh/8Bh/10Bh/18Bh)				**Function related to: Interrupt Enable**
ADCIE	0B.7	R/W	0	ADC interrupt enable, 1=enable, 0=disable
T2IE	0B.6	R/W	0	T2 interrupt enable, 1=enable, 0=disable
TM1IE	0B.5	R/W	0	Timer1 interrupt enable, 1=enable, 0=disable
TM0IE	0B.4	R/W	0	Timer0 interrupt enable, 1=enable, 0=disable
WKTIE	0B.3	R/W	0	Wakeup Timer interrupt enable, 1=enable, 0=disable Set 0 to clear & disable WKT timer
INT2IE	0B.2	R/W	0	INT2 pin (PA7 or PB7) interrupt enable, 1=enable, 0=disable

Name	Address	R/W	Rst	Description
INT1IE	0B.1	R/W	0	INT1 pin (PA4 or PB4) interrupt enable, 1=enable, 0=disable
INT0IE	0B.0	R/W	0	INT0 pin (PA0 or PB0) interrupt enable, 1=enable, 0=disable
INTIF (0Ch)				**Function related to: Interrupt Flag**
ADCIF	0C.7	R	-	ADC interrupt flag, set by H/W after end of ADC conversion
		W	0	S/W write 7Fh to INTIF to clear this flag
T2IF	0C.6	R	-	T2 interrupt event pending flag, set by H/W while T2 overflows
		W	0	S/W write BFh to INTIF to clear this flag
TM1IF	0C.5	R	-	Timer1 interrupt event pending flag, set by H/W while Timer1 overflows
		W	0	S/W write DFh to INTIF to clear this flag
TM0IF	0C.4	R	-	Timer0 interrupt event pending flag, set by H/W while Timer0 overflows
		W	0	S/W write EFh to INTIF to clear this flag
WKTIF	0C.3	R	-	WKT interrupt event pending flag, set by H/W while WKT time out
		W	0	S/W write F7h to INTIF to clear this flag
INT2IF	0C.2	R	-	INT2 (PA7 or PB7) interrupt event pending flag, set by H/W at INT2 pin's falling edge
		W	0	S/W write FBh to INTIF to clear this flag
INT1IF	0C.1	R	-	INT1 (PA4 or PA4) interrupt event pending flag, set by H/W at INT1 pin's falling/rising edge
		W	0	S/W write FDh to INTIF to clear this flag
INT0IF	0C.0	R	-	INT0 (PA0 or PB0) interrupt event pending flag, set by H/W at INT0 pin's falling/rising edge
		W	0	S/W write FEh to INTIF to clear this flag
FSR1 (0Dh)				**Function related to: RAM (BANK4/5) W/R**
FSR1	0D.7~0	R/W	-	File Select 1 Register, indirect address mode pointer
INTIE1 (0Eh)				**Function related to: Interrupt Enable Group 1**
LVDIE	0E.7	R/W	0	LVD interrupt enable, 0=disable , 1=enable
EEPIE	0E.6	R/W	0	EEP write finish interrupt enable 0=disable , 1=enable
I2CIE	0E.2	R/W	0	I2C interrupt enable, 0=disable , 1=enable
UARTIE	0E.1	R/W	0	UART TX/RX interrupt enable, 0=disable , 1=enable
PXIE	0E.0	R/W	0	Pin change interrupt enable, 0=disable , 1=enable
CLKCTL (0Fh)				**Function related to: Fsys**
SCKTYPE	0F.7	R/W	0	Slow-clock type 0: SIRC 1: SXT
FCKTYPE	0F.6	R/W	0	Fast-clock type 0: FIRC 1: FXT
SLOWSTP	0F.4	R/W	0	Stop Slow-clock in Stop Mode 0: no Stop 1: Stop
FASTSTP	0F.3	R/W	1	Stop Fast-clock 0:no Stop 1:Stop
CPUCKS	0F.2	R/W	0	Select Fast-clock 0: Fsys=Slow-clock 1: Fsys=Fast-clock
CPUPSC	0F.1~0	R/W	11	Fsys Prescaler, 0: div 8, 1: div 4, 2: div 2, 3: div 1

Name	Address	R/W	Rst	Description
TM0RLD (10h)				**Function related to: TM0**
TM0RLD	10.7~0	R/W	0	Timer0 reload Data
TM0CTL (11h)				**Function related to: TM0**
T0ISRC	11.7	R/W	0	Timer0 Counter mode source 0: TM0CKI pin (PA2) 1: SXT divided 16
TM0STP	11.6	R/W	0	Timer0 counter stop 0: Release 1: Stop counting
TM0EDG	11.5	R/W	0	Timer0 prescaler counting edge for TM0CKI pin 0: rising edge 1: falling edge
TM0CKS	11.4	R/W	0	Timer0 prescaler clock source 0: Fsys/2 1: TM0CKI pin (PA2 pin)
TM0PSC	11.3~0	R/W	0	Timer0 prescaler. Timer0 prescaler clock source divided by 0000: /1 0101: /32 1010: /1024 1111: /32768 0001: /2 0110: /64 1011: /2048 0010: /4 0111: /128 1100: /4096 0011: /8 1000: /256 1101: /8192 0100: /16 1001: /512 1110: /16384
TM1 (12h)				**Function related to: Timer1**
TM1	12.7~0	R/W	0	Timer1 content
TM1RLD (13h)				**Function related to: Timer1**
TM1RLD	13.7~0	R/W	0	Timer1 reload Data
TM1CTL (14h)				**Function related to: Timer1**
TM1STP	14.4	R/W	0	Timer1 counter stop 0: Release 1: Stop counting
TM1PSC	14.3~0	R/W	0	Timer1 prescaler. Timer1 clock source 0000: Fsys/2 0001: Fsys/4 0010: Fsys/8 0011: Fsys/16 0100: Fsys/32 0101: Fsys/64 0110: Fsys/128 0111: Fsys/256 1xxx: Fsys/512
T2CTL (15h)				**Function related to: T2**
T2CLR	15.3	R/W	1	T2 counter clear 0: Release 1: Stop counting
T2CKS	15.2	R/W	0	T2 clock source selection. 1: Fsys/128 0: Slow-clock
T2PSC	15.1~0	R/W	0	T2 prescaler. T2 clock source divided by - 00: 32768 01: 16384 10: 8192 11: 128
LVCTL (16h)				**Function related to: LVR/LVD**
LVDF	16.7	R	-	Low voltage detection flag, set by H/W while Vcc ≦ LVD
BGEN	16.6	R/W	1	Band Gap BG1.18V enable 0: Disable 1: Enable and Auto disable in STOP/IDLE mode
LVRSAV	16.5	R/W	1	POR/LVR power save 1: POR/LVR auto power off in STOP/IDLE mode 0: POR/LVR enable in STOP/IDLE mode
LVDSAV	16.4	R/W	1	LVD auto power off in STOP/IDLE mode 0= LVD enable 1= LVD enable in slow/Fast mode; disable in STOP/IDLE mode
LVDS	16.3~0	R/W	01	LVD select; 0000: Disable, 0001: 1.5V, …. 1111: 3.5V

Name	Address	R/W	Rst	Description
ADCDH (17h)				**相关功能: ADC**
ADCDH	17.7~0	R	-	ADC 输出数据位 11~4
ADCTL (18h)				**相关功能: ADC**
ADCDL	18.7~4	R	-	ADC 输出数据位 3~0
ADST	18.3	R/W	0	ADC 起始位 0: 转换结束后由 H/W 清除 1: ADC 开始转换
ADCKS	18.2~0	R/W	0	ADC 时钟频率选择: 000: Fsys/256 100: Fsys/16 001: Fsys/128 101: Fsys/8 010: Fsys/64 110: Fsys/4 011: Fsys/32 111: Fsys/2 (TC)
ADCTL2 (19h)				**相关功能: ADC**
ADCVREFS	19.7	R/W	0	ADC VREF 设置, 0: VCC, 1: VBG
VBGSEL	19.6~5	R/W	0	VBG 选择设置, 00: 1.18V, 01: 2.5V, 10: 3.0V, 11: 2.0V
ADCHS	19.4~0	R/W	0	ADC 通道选择 00000: ADC0 (PB0) 00001: ADC1 (PB1) 00010: ADC2 (PB2) 00011: ADC3 (PB3) 00100: ADC4 (PB4) 00101: ADC5 (PB5) 00110: ADC6 (PB6) 00111: ADC7 (PB7) 01000: ADC8 (PC0) 01001: ADC9 (PC1) 01010: ADC10(PC2) 01011: ADC11(PC3) 01100: ADC12(VBG) 01101: ADC13(PD0) 01110: ADC14(PD1) 01111: ADC15(PD2) 10000: ADC16(PD3) 10001: ADC17(PD4) 10010: ADC18(PD5) 10011: ADC19(PD6) 10100: ADC20(PD7) 10101: ADC21(PA7) 10110: ADC22(PA4) 10111: ADC23(PA3) 11000: ADC24(PA1) 11001: ADC25(PA0) 11010: ADC26(PA2) 11011:ADC27(VTEMP) others: reserved 11111: ADC31(1/4 VCC)
INTIF1 (1Ah)				**Function related to: Interrupt Flag**

Name	Address	R/W	Rst	Description
LVDIF	1A.7	R/W	0	LVD interrupt event pending flag, set by H/W while LV Detected S/W write 7Fh to INTIF1 to clear this flag
EEPIF	1A.6	R/W	0	EEP write complete interrupt pending flag This bit is set by H/W after EEP is written, S/W write BFh to INTIF1 to clear this flag.
I2CIF	1A.2	R/W	0	Slave I2C interrupt pending flag This bit is set by H/W while ✧ I2CRCD0 or I2CRCD1 receive data finished ✧ I2CRCD0 or I2CRCD1 data overflow occurred ✧ I2CTXD0 or I2CTXD1 data transmit finished. S/W write FBh to INTIF1 to clear this flag
UARTIF	1A.1	R/W	0	UART Interrupt Flag, set by H/W while TX/RX transfers complete; S/W write FDh to INTIF1 to clear this flag
PXIF	1A.0	R/W	0	Port A/B/C/D Pin Change Interrupt Flag S/W write FEh to INTIF1 to clear this flag
IOCCTRL (1Bh)				**Function related to: IO Port**
LED	1B.7~4	R/W	0	IO current Control for LED usage enable LED[3]: PD port LED[2]: PC port LED[1]: PB port LED[0]: PA port 1: enable, 0 disable
HSNK	1B.3~0	R/W	0	IO current Control for High Sink usage enable HSNK[3]: PD port HSNK[2]: PC port HSNK[1]: PB port HSNK[0]: PA port 1: enable, 0: disable
PAWKE (1Ch)				**Function related to: Port A**
PAWKE	1C.7~0	R/W	0	PA pin Individual Wake up Enable 0: disable 1: enable
PBWKE (1Dh)				**Function related to: Port B**
PBWKE	1D.7~0	R/W	0	PB pin Individual Wake up Enable 0: disable 1: enable
PCWKE (1Eh)				**Function related to: Port C**
PCWKE	1E.7~0	R/W	0	PC pin Individual Wake up Enable 0: disable 1: enable
PDWKE (1Fh)				**Function related to: Port D**
PDWKE	1F.7~0	R/W	0	PD pin Individual Wake up Enable 0: disable 1: enable
User Data Memory				
RAM	20~6F	R/W	-	RAM Bank0 area (80 Bytes)
RAM	70~7F	R/W	-	RAM common area (16 Bytes)

OPTION (81h/181h)				Function related to: STATUS/INT0/INT1/WDT/WKT
HWAUTO	81.7	R/W	0	Enter interrupt vector, HW auto save/restore WREG 、FSR、TABR、PCLATH、DPL、DPH and STATUS w/o TO, PD 0:disable 1: Enable ; (work for ASM not for C)
INT0EDG	81.6	R/W	0	INT0 pin edge interrupt event 0: falling edge to trigger 1: rising edge to trigger
INT1EDG	81.5	R/W	0	INT1 pin edge interrupt event 0: falling edge to trigger 1: rising edge to trigger
FREQL	81.4	R/W	1	SIRC Frequency selection 0: 60KHz 1: 50KHz
WDTPSC	81.3~2	R/W	11	WDT pre-scale selections: 00: 164mS 01: 328mS 10: 655mS 11: 1311mS
WKTPSC	81.1~0	R/W	11	WKT pre-scale selections: 00: 21mS 01: 41mS 10: 82mS 11: 164mS
PAMODH (85h)				**Function related to: Port A**
PA7MOD	85.7~6	R/W	01	PA7, PA4 I/O mode control 00: Mode0 01: Mode1 10: Mode2 11: Mode3
PA4MOD	85.1~0	R/W	01	
PAMODL (86h)				**Function related to: Port A**
PA3MOD	86.7~6	R/W	01	PA3~PA0 I/O 模式控制 00: Mode0 01: Mode1 10: Mode2 11: Mode3
PA2MOD	86.5~4	R/W	01	
PA1MOD	86.3~2	R/W	01	
PA0MOD	86.1~0	R/W	01	
PBMODH (87h)				**Function related to: Port B**
PB7MOD	87.7~6	R/W	01	PB7~PB4 I/O mode control 00: Mode0 01: Mode1 10: Mode2 11: Mode3
PB6MOD	87.5~4	R/W	01	
PB5MOD	87.3~2	R/W	01	
PB4MOD	87.1~0	R/W	01	
PBMODL (88h)				**Function related to: Port B**
PB3MOD	88.7~6	R/W	01	PB3~PB0 I/O mode control 00: Mode0 01: Mode1 10: Mode2 11: Mode3
PB2MOD	88.5~4	R/W	01	
PB1MOD	88.3~2	R/W	01	
PB0MOD	88.1~0	R/W	01	
PCMODL (8Ch)				**Function related to: Port C**
PC3MOD	8C.7~6	R/W	01	PC3~PC0 I/O mode control 00: Mode0 01: Mode1 10: Mode2 11: Mode3
PC2MOD	8C.5~4	R/W	01	
PC1MOD	8C.3~2	R/W	01	
PC0MOD	8C.1~0	R/W	01	

PDMODH (8Dh)				Function related to: Port D
PD7MOD	8D.7~6	R/W	01	PD7~PD4 I/O mode control 00: Mode0 01: Mode1 10: Mode2 11: Mode3
PD6MOD	8D.5~4	R/W	01	
PD5MOD	8D.3~2	R/W	01	
PD4MOD	8D.1~0	R/W	01	
PDMODL (8Eh)				**Function related to: Port D**
PD3MOD	8E.7~6	R/W	01	PD3~PD0 I/O mode control 00: Mode0 01: Mode1 10: Mode2 11: Mode3
PD2MOD	8E.5~4	R/W	01	
PD1MOD	8E.3~2	R/W	01	
PD0MOD	8E.1~0	R/W	01	
OPTION2 (8Fh)				**Function related to: PWM0/PWM1**
TCOE	8F.7	R/W	0	TCOUT Output Enable 0: Disable 1: Enable, output to PB1
TM1OE	8F.6	R/W	0	Timer1 overflow toggle Output Enable 0: Disable 1:Enable, output to PB0
PWMCKS	8F.5~4	R/W	0	PWM0 Clock Source 0x: Fsys 10:FIRC 11:FIRC*2
INT2SEL	8F.2	R/W	0	INT2 Pin Select 0: PA7 1: PB7
INT1SEL	8F.1	R/W	0	INT1 Pin Select 0: PA4 1: PB4
INT0SEL	8F.0	R/W	0	INT0 Pin Select 0: PA0 1: PB0
PWMOE (90h)				**Function related to: PWM0/PWM1**
PWM0POE	90.7	R/W	0	PWM0P Output Enable to PA7 0: disable 1: enable
PWM0NOE	90.6	R/W	1	PWM0N Output Enable to PA0 0: disable 1: enable
PWM5OE	90.5	R/W	1	PWM5 Output Enable to PA7 0: disable 1: enable (PWM0POE has higher priority)
PWM4OE	90.4	R/W	1	PWM4 Output Enable to PA4 0: disable 1: enable
PWM3OE	90.3	R/W	1	PWM3 Output Enable to PA3 0: disable 1: enable
PWM2OE	90.2	R/W	0	PWM2 Output Enable to PA2 0: disable 1: enable
PWM1OE	90.1	R/W	0	PWM1 Output Enable to PA1 0: disable 1: enable

PWMCTL (91h)				**Function related to: PWM**
PWMEN	91.7	R/W	0	PWM Clock Enable 0: Disable 1: Enable
PWM0OM	91.5~4	R/W	0	PWM0 output mode 00~11: Mode0 ~Mode3
PWM0DZ	91.3~0	R/W	0	PWM0 dead zone (non-overlap) 0000~1111: 0~14, 16 *Tpwmclk
PWMPRDH (92h)				**Function related to: PWM**
PWMPRDH	92.7~0	R/W	0	PWM Period MSB data
PWMPRDL (93h)				**Function related to: PWM**
PWMPRDL	93.7~0	R/W	0	PWM Period LSB data
PWM0DH (94h)				**Function related to: PWM**
PWM0DH	94.7~0	R/W	0	PWM0 Duty MSB data
PWM0DL (95h)				**Function related to: PWM**
PWM0DL	95.7~0	R/W	0	PWM0 Duty LSB data
PWM1DH (96h)				**Function related to: PWM**
PWM1DH	96.7~0	R/W	0	PWM1 Duty MSB data
PWM1DL (97h)				**Function related to: PWM**
PWM1DL	97.7~0	R/W	0	PWM1 Duty LSB data
PWM2DH (98h)				**Function related to: PWM**
PWM2DH	98.7~0	R/W	0	PWM2 Duty MSB data
PWM2DL (99h)				**Function related to: PWM**
PWM2DL	99.7~0	R/W	0	PWM2 Duty LSB data
PWM3DH (9Ah)				**Function related to: PWM**
PWM3DH	9A.7~0	R/W	0	PWM3 Duty MSB data
PWM3DL (9Bh)				**Function related to: PWM**
PWM3DL	9B.7~0	R/W	0	PWM3 Duty LSB data
PWM4DH (9Ch)				**Function related to: PWM**
PWM4DH	9C.7~0	R/W	0	PWM4 Duty MSB data
PWM4DL (9Dh)				**PWM4DL (9Dh)**
PWM4DL	9D.7~0	R/W	0	PWM4 Duty LSB data
PWM5DH (9Eh)				**Function related to: PWM**
PWM5DH	9E.7~0	R/W	0	PWM5 Duty MSB data
PWM5DL (9Fh)				**Function related to: PWM**
PWM5DL	9F.7~0	R/W	0	PWM5 Duty LSB data
User Data Memory				
RAM	A0~EF	R/W	-	RAM Bank1area (80 Bytes)

TESTREG (105h)				**Function related to: TEST**
TESTREG	105.7~0	R/W	00	Test bits, Keep 00
RDCTL (106h)				**Function related to：ROM Read TEST**
RDCTL	106.1~0	W	1	Read signal delay control for Program ROM 00: 4ns delay for read signal of Program ROM 01: 8ns delay for read signal of Program ROM 10: 12ns delay for read signal of Program ROM 11: 16ns delay for read signal of Program ROM Change this register at slow clock for safety. The user must switch this register to “4ns” to enhance the performance of minimal operating voltage. This feature can’t be emulated.
LVRPD (107h)				**Function related to： LVR**
LVRPD	107	W	0	LVR power down control Write 0x37 to force LVR+POR disable Write 0x38 to force LVR disable, POR still enable Write 0x39 to force POR disable, LVR still enable Write other value to enable LVR/POR
PORPDF	107.1	R	0	1: when Reg. 107h write 0x37 or 0x39; 0: when Reg. 107h write other value
LVRPDF	107.0	R	0	1: when Reg. 107h write 0x37 or 0x38; 0: when Reg. 107h write other value
LOE (108h)				**Function related to: Software LCD COM**
LOE	108	R/W	0	Software COM3~0 LCD 1/2 bias Output Enable
	108.3	R/W	0	1: PA1 as LCD COM3 1/2 bias Output Enable
	108.2	R/W	0	1: PA0 as LCD COM2 1/2 bias Output Enable
	108.1	R/W	0	1: PD6 as LCD COM1 1/2 bias Output Enable
	108.0	R/W	0	1: PD7 as LCD COM0 1/2 bias Output Enable
PCH (10Ch)				**Function related to: PC**
PCH	10C.3~0	R	0	Program Counter [11:8]
		W	0	Write 1Ch to this register, then it’s not necessary to write PCLATCH for table read lookup
UBAUD (10Dh)				**Function related to: UART**
UBAUD	10D	R/W	0	UART Baud Rate divider
BGTRIM (10Eh)				**Function related to: VBG**
BGTRIM	10E.4~0	R/W		VBG voltage adjustment 00h: Lowest voltage 1Fh: Highest voltage
I2CTXD0 (110h)				**Function related to: Slave I2C**
I2CTXD0	110.7~0	R/W	0	The transmitting register0 of slave I2C
I2CTXD1 (111h)				**Function related to: Slave I2C**
I2CTXD1	111.7~0	R/W	0	The transmitting register1 of slave I2C
I2CCTL (112h)				**Function related to: Slave I2C**
I2CEN	112.3	R/W	0	Slave I2C interface enable 0: Disable 1: Enable
I2CID	112.1~0	R/W	0	Slave I2C ID last 2 bits

I2CFLG (113h)				Function related to: Slave I2C
TXD1F	113.5	R/W	0	Slave I2C transmitting data register 1 flag, set by H/W while I2CTXD1 data transmitting finished S/W write DFh to I2CFLG to clear this flag
TXD0F	113.4	R/W	0	Slave I2C transmitting data register 0 flag, set by H/W while I2CTXD0 data transmitting finished S/W write EFh to I2CFLG to clear this flag
RCD1OVF	113.3	R/W	0	Slave I2C receiving data register 1 overflow, set by H/W while receiving data to I2CRCD1 overflow. S/W write F7h to I2CFLG to clear this flag
RCD1F	113.2	R/W	0	Slave I2C receiving data register 1 flag, set by H/W while I2CRCD1 data receiving finished S/W write FBh to I2CFLG to clear this flag
RCD0OVF	113.1	R/W	0	Slave I2C receiving data register 0 overflows, set by H/W while receiving data to I2CRCD0 overflow. S/W write FDh to I2CFLG to clear this flag
RCD0F	113.0	R/W	0	Slave I2C receiving data register 0 flag, set by H/W while I2CRCD0 data receiving finished S/W write FEh to I2CFLG to clear this flag
I2CRCD0 (114h)				**Function related to: Slave I2C**
I2CRCD0	114.7~0	R	0	The receiving register0 of slave I2C
User Data Memory				
RAM	120~16F	R/W	-	RAM Bank2 area (80 Bytes)

DPL (185h)				**Function related to: Table Read**
DPL	185.7~0	R/W	0	Table read low address, data ROM pointer (DPTR) low byte
DPH (186h)				**Function related to: Table Read**
DPH	186.4~0	R/W	0	Table read high address, data ROM pointer (DPTR) high byte
UARTCTL (187h)				**Function related to: UART Control**
URTD9	187.7	R/W	0	UART Transmitted 9th bit
UMODE	187.6~5	R/W	0	UART Mode Selection 0: 7-bit, 1: 8-bit, 2: 9-bit, 3: Reserved
UTXE	187.3	R/W	0	UART Transmit Enable 0: Disable 1: Enable
URXE	187.2	R/W	0	UART Receive Enable 0: Disable 1: Enable
UARTE	187.1	R/W	0	UART Function Enable 0: Disable 1: Enable
UINVEN	187.0	R/W	0	UART TX/RX output/input Inversion Enable 0: Disable 1: Enable
UARTSTA (188h)				**Function related to: UART Status and Control**
UTBE	188.7	R	0	0: TX is transmitting 1: TX buffer is empty;
URRD9	188.6	R	0	Received 9th bit
EVEN	188.5	R/W	0	UART Parity ; 0: ODD 1: EVEN,
PRE	188.4	R/W	0	UART Parity addition; 0: Disable 1: Enable,
PRERR	188.3	R/W	0	set to 1 when parity error occurs, S/W write 0 to clear it
OVERR	188.2	R/W	0	set to 1 when overrun error occurs, S/W write 0 to clear it
FMERR	188.1	R/W	0	set to 1 when frame error occurs, S/W write 0 to clear it
URBF	188.0	R	0	set to 1 when 1 byte (or 9-bit) is received Read UARTDAT register will clear it to 0
TABR (18Ch)				**Function related to: Table Read**
TABR	18C.7~0	R/W	0	1: TABR write 1 = opcode TABRL 2: TABR write 2 = opcode TABRH 3: after step1 or step2, read TABR to get main ROM table read value after step1, read TABR to get EEPROM value (when EEPEN=E2h) *Table Read for ASM: TABRL/TABRH or TABR* *Table Read for C: using register TABR* ***and it is forbidden to us in interrupt***
URDATA (18Dh)				**Function related to: UART DATA**
URDATA	18D.7~0	R/W	0	UART TX/RX Data Buffer
		R	0	from Receive Buffer
		W	0	to Transmit shifter

CRCDL (18Eh)				**Function related to: CRC16**
CRCDL	18E.7~0	R/W	0	CRC16 Data 7~0
CRCDH(18Fh)				**Function related to: CRC16**
CRCDH	18F.7~0	R/W	0	CRC16 Data 15~8
CRCIN(190h)				**Function related to: CRC16**
CRCIN	190.7~0	W	0	CRC input data
EEPCTL (191h)				**Function related to: EEPROM**
EEPTO	191.7	R	0	EEPROM Write Time-out flag
EEPTE	191.1~0	R/W	11	EEPROM Write Time-out enable; 00: Disable; 01:2.5ms; 10:10ms; 11:20.5ms
EEPEN (192h)				**Function related to: EEPROM**
EEPEN	192.7~0	W	0	EEPROM Enable 0xE2: Enable Others: Disable
EEPDT (193h)				**Function related to: EEPROM**
EEPDT	193.7~0	W	0	EEPROM Data to write in
VTEMPDL (19Eh)				**Function related to: VTEMPDL**
VTEMPDL	19E.2~0	R/W	0	ADC VTEMP[3:0] : Factory read value
VTEMPDH(19Fh)				**Function related to: VTEMPDH**
VTEMPDH	19F.7~0	R/W	0	ADC VTEMP[11:4] : Factory read value
User Data Memory				
RAM	1A0~1EF	R/W	-	RAM Bank3 area (80 Bytes)
Indirect Addressing User Data Memory				**Use INDF1/FSR1 to access RAM Bank 4/5**
RAM	00~ 7F	R/W	-	Indirect addressing RAM Bank4 area (128 bytes); use **INDF1/FSR1**
RAM	80~FF	R/W	-	Indirect addressing RAM Bank5 area (128 bytes); use **INDF1/FSR1**

INSTRUCTION SET

Each instruction is a 16-bit word divided into an Op Code, which specifies the instruction type, and one or more operands, which further specify the operation of the instruction. The instructions can be categorized as byte-oriented, bit-oriented and literal operations list in the following table.

For byte-oriented instructions, “f” represents the address designator and “d” represents the destination designator. The address designator is used to specify which address in Program memory is to be used by the instruction. The destination designator specifies where the result of the operation is to be placed. If “d” is “0”, the result is placed in the W register. If “d” is “1”, the result is placed in the address specified in the instruction.

For bit-oriented instructions, “b” represents a bit field designator, which selects the number of the bit affected by the operation, while “f” represents the address designator. For literal operations, “k” represents the literal or constant value.

Field/Legend	Description
f	Register File Address
b	Bit address
k	Literal. Constant data or label
d	Destination selection field, 0: Working register, 1: Register file
W	Working Register
Z	Zero Flag
C	Carry Flag or/Borrow Flag
DC	Decimal Carry Flag or Decimal/Borrow Flag
PC	Program Counter
TOS	Top Of Stack
GIE	Global Interrupt Enable Flag (i-Flag)
[]	Option Field
( )	Contents
.	Bit Field
B	Before
A	After
←	Assign direction

Mnemonic		Op Code	Cycle	Flag Affect	Description
Byte-Oriented File Register Instruction					
ADDWX	f, d	ff00 0111 dfff ffff	1	C, DC, Z	Add W and "f"
ANDWX	f, d	ff00 0101 dfff ffff	1	Z	AND W with "f"
CLRX	f	ff00 0001 1fff ffff	1	Z	Clear "f"
CLRW		0000 0001 0100 0000	1	Z	Clear W
COMX	f, d	ff00 1001 dfff ffff	1	Z	Complement "f"
DECX	f, d	ff00 0011 dfff ffff	1	Z	Decrement "f"
DECXSZ	f, d	ff00 1011 dfff ffff	1 or 2	-	Decrement "f", skip if zero
INCX	f, d	ff00 1010 dfff ffff	1	Z	Increment "f"
INCXSZ	f, d	ff00 1111 dfff ffff	1 or 2	-	Increment "f", skip if zero
IORWX	f, d	ff00 0100 dfff ffff	1	Z	OR W with "f"
MOVX	f,d	ff00 1000 dfff ffff	1	Z	Move "f"
MOVXW	f	ff00 1000 0fff ffff	1	Z	Move "f" to W
MOVWX	f	ff00 0000 1fff ffff	1	-	Move W to "f"
RLX	f, d	ff00 1101 dfff ffff	1	C	Rotate left "f" through carry
RRX	f, d	ff00 1100 dfff ffff	1	C	Rotate right "f" through carry
SUBWX	f, d	ff00 0010 dfff ffff	1	C, DC, Z	Subtract W from "f"
SWAPX	f, d	ff00 1110 dfff ffff	1	-	Swap nibbles in "f"
TSTX	f	ff00 1000 1fff ffff	1	Z	Test if "f" is zero
XORWX	f, d	ff00 0110 dfff ffff	1	Z	XOR W with "f"
Bit-Oriented File Register Instruction					
BCX	f, b	ff11 00bb bfff ffff	1	-	Clear "b" bit of "f"
BSX	f, b	ff11 01bb bfff ffff	1	-	Set "b" bit of "f"
BTXSC	f, b	ff11 10bb bfff ffff	1 or 2	-	Test "b" bit of "f", skip if clear
BTXSS	f, b	ff11 11bb bfff ffff	1 or 2	-	Test "b" bit of "f", skip if set
Literal and Control Instruction					
ADDLW	k	0001 1100 kkkk kkkk	1	C, DC, Z	Add Literal "k" and W
ANDLW	k	0001 1011 kkkk kkkk	1	Z	AND Literal "k" with W
LCALL	k	kk10 0kkk kkkk kkkk	2	-	Call subroutine "k"
CLRWDT		0001 1110 0000 0100	1	TO, PD	Clear Watch Dog Timer
LGOTO	k	kk10 1kkk kkkk kkkk	2	-	Jump to branch "k"
IORLW	k	0001 1010 kkkk kkkk	1	Z	OR Literal "k" with W
MOVLW	k	0001 1001 kkkk kkkK	1	-	Move Literal "k" to W
NOP		0000 0000 0000 0000	1	-	No operation
RET		0000 0000 0100 0000	2	-	Return from subroutine
RETI		0000 0000 0110 0000	2	-	Return from interrupt
SUBXW		ff01 0010 dfff ffff	1	C, DC, Z	Subtract "f" from W
SUBXWB		ff01 0011 dfff ffff	1	C, DC, Z	Subtract "f" from W with Borrow
SUBWXB		ff01 0100 dfff ffff	1	C, DC, Z	Subtract W from "f" with Borrow
ADDWXC		ff01 0111 dfff ffff	1	C, DC, Z	Add W and "f" and C
RETLW	k	0001 1000 kkkk kkkk	2	-	Return with Literal in W
SLEEP		0001 1110 0000 0011	1	TO, PD	Go into Power-down mode, Clock oscillation stops
SUBLW	k	0001 1111 kkkk kkkk	1	C, DC, Z	Subtract W from literal
TABRH		0000 0000 0101 1000	2	-	Lookup ROM high data to W
TABRL		0000 0000 0101 0000	2	-	Lookup ROM low data to W
XORLW	k	0001 1101 kkkk kkkk	1	Z	XOR Literal "k" with W

Note: ADDWXC/ SUBXW / SUBXWB / SUBWXB 4 instructions do not support indirect addressing, only support direct addressing

ADDLW	**Add Literal "k" and W**	
Syntax	ADDLW k	
Operands	k : 00h ~ FFh	
Operation	(W) ← (W) + k	
Status Affected	C, DC, Z	
OP-Code	0001 1100 kkkk kkkk	
Description	The contents of the W register are added to the eight-bit literal 'k' and the result is placed in the W register.	
Cycle	1	
Example	ADDLW 0x15	B : W =0x10 A : W =0x25

ADDWX	**Add W and "f"**	
Syntax	ADDWX f [,d]	
Operands	f : 00h ~ 1FFh, d : 0, 1	
Operation	(destination) ← (W) + (f)	
Status Affected	C, DC, Z	
OP-Code	ff00 0111 dfff ffff	
Description	Add the contents of the W register with register 'f'. If 'd' is 0, the result is stored in the W register. If 'd' is 1, the result is stored back in register 'f'.	
Cycle	1	
Example	ADDWX FSR, 0	B : W =0x17, FSR =0xC2 A : W =0xD9, FSR =0xC2

ADDWXC	**Add W and "f"**	
Syntax	ADDWXC f [,d]	
Operands	f : 000h ~ 1FFh, d : 0, 1	
Operation	(destination) ← (W) + (f) + C	
Status Affected	C, DC, Z	
OP-Code	ff01 0111 dfff ffff	
Description	Add the contents of the W register with register 'f' and C. If 'd' is 0, the result is stored in the W register. If 'd' is 1, the result is stored back in register 'f'	
Cycle	1	
Example	ADDWXC FSR, 0	B : W =0x17, FSR =0xC2, C=0 A : W =0xD9, FSR =0xC2, C=0
Example	ADDWXC FSR, 0	B : W =0x17, FSR =0xC2, C=1 A : W =0xDA, FSR =0xC2, C=0

ANDLW	**Logical AND Literal "k" with W**	
Syntax	ANDLW k	
Operands	k : 00h ~ FFh	
Operation	(W) ← (W) AND k	
Status Affected	Z	
OP-Code	0001 1011 kkkk kkkk	
Description	The contents of W register are AND'ed with the eight-bit literal 'k'. The result is placed in the W register.	
Cycle	1	
Example	ANDLW 0x5F	B : W =0xA3 A : W =0x03

ANDWX	AND W with "f"	
Syntax	ANDWX f [,d]	
Operands	f : 00h ~ 1FFh, d : 0, 1	
Operation	(destination) ← (W) AND (f)	
Status Affected	Z	
OP-Code	ff00 0101 dfff ffff	
Description	AND the W register with register 'f'. If 'd' is 0, the result is stored in the W register. If 'd' is 1, the result is stored back in register 'f'.	
Cycle	1	
Example	ANDWX FSR, 1	B : W =0x17, FSR =0xC2 A : W =0x17, FSR =0x02

BCX	Clear "b" bit of "f"	
Syntax	BCX f [,b]	
Operands	f : 00h ~ 1FFh, b : 0 ~ 7	
Operation	(f.b) ← 0	
Status Affected	-	
OP-Code	ff11 00bb bfff ffff	
Description	Bit 'b' in register 'f' is cleared.	
Cycle	1	
Example	BCX FLAG_REG, 7	B : FLAG_REG =0xC7 A : FLAG_REG =0x47

BSX	Set "b" bit of "f"	
Syntax	BSX f [,b]	
Operands	f : 00h ~ 1FFh, b : 0 ~ 7	
Operation	(f.b) ← 1	
Status Affected	-	
OP-Code	ff11 01bb bfff ffff	
Description	Bit 'b' in register 'f' is set.	
Cycle	1	
Example	BSX FLAG_REG, 7	B : FLAG_REG =0x0A A : FLAG_REG =0x8A

BTXSC	Test "b" bit of "f", skip if clear(0)	
Syntax	BTXSC f [,b]	
Operands	f : 00h ~ 1FFh, b : 0 ~ 7	
Operation	Skip next instruction if (f.b) =0	
Status Affected	-	
OP-Code	ff11 10bb bfff ffff	
Description	If bit 'b' in register 'f' is 1, then the next instruction is executed. If bit 'b' in register 'f' is 0, then the next instruction is discarded, and a NOP is executed instead, making this a 2nd cycle instruction.	
Cycle	1 or 2	
Example	LABEL1 BTXSC FLAG, 1 TRUE LGOTO SUB1 FALSE ...	B : PC =LABEL1 A : if FLAG.1 =0, PC =FALSE if FLAG.1 =1, PC =TRUE

BTXSS	Test "b" bit of "f", skip if set(1)	
Syntax	BTXSS f [,b]	
Operands	f : 00h ~ 1FFh, b : 0 ~ 7	
Operation	Skip next instruction if (f.b) =1	
Status Affected	-	
OP-Code	ff11 11bb bfff ffff	
Description	If bit 'b' in register 'f' is 0, then the next instruction is executed. If bit 'b' in register 'f' is 1, then the next instruction is discarded, and a NOP is executed instead, making this a 2nd cycle instruction.	
Cycle	1 or 2	
Example	LABEL1 BTXSS FLAG, 1 TRUE LGOTO SUB1 FALSE ...	B : PC =LABEL1 A : if FLAG.1 =0, PC =TRUE if FLAG.1 =1, PC =FALSE

LCALL	Call subroutine "k"	
Syntax	LCALL k	
Operands	k : 000h ~ 1FFFh	
Operation	Operation: TOS ← (PC) + 1, PC.10~0 ← k	
Status Affected	-	
OP-Code	kk10 0kkk kkkk kkkk	
Description	Call Subroutine. First, return address (PC+1) is pushed onto the stack. The 13-bit immediate address is loaded into PC bits <12:0>. The upper bits of PC are loaded from PCLATH. CALL is a two-cycle instruction.	
Cycle	2	
Example	LABEL1 LCALL SUB1	B : PC =LABEL1 A : PC =SUB1, TOS =LABEL1 + 1

CLRX	Clear "f"	
Syntax	CLRX f	
Operands	f : 00h ~ 1FFh	
Operation	(f) ← 00h, Z ← 1	
Status Affected	Z	
OP-Code	ff00 0001 1fff ffff	
Description	The contents of register 'f' are cleared and the Z bit is set.	
Cycle	1	
Example	CLRX FLAG_REG	B : FLAG_REG =0x5A A : FLAG_REG =0x00, Z =1

CLRW	Clear W	
Syntax	CLRW	
Operands	-	
Operation	(W) ← 00h, Z ← 1	
Status Affected	Z	
OP-Code	0000 0001 0100 0000	
Description	W register is cleared and Z bit is set.	
Cycle	1	
Example	CLRW	B : W =0x5A A : W =0x00, Z =1

CLRWDT	Clear Watchdog Timer	
Syntax	CLRWDT	
Operands	-	
Operation	WDT Timer ← 00h	
Status Affected	TO, PD	
OP-Code	0001 1110 0000 0100	
Description	CLRWDT instruction clears the Watchdog Timer	
Cycle	1	
Example	CLRWDT	B : WDT counter =? A : WDT counter =0x00

COMX	Complement "f"	
Syntax	COMX f [,d]	
Operands	f : 00h ~ 1FFh, d : 0, 1	
Operation	(destination) ← ($\bar{f}$)	
Status Affected	Z	
OP-Code	ff00 1001 dfff ffff	
Description	The contents of register 'f' are complemented. If 'd' is 0, the result is stored in W. If 'd' is 1, the result is stored back in register 'f'.	
Cycle	1	
Example	COMX REG1, 0	B : REG1 =0x13 A : REG1 =0x13, W =0xEC

DECX	Decrement "f"	
Syntax	DECX f [,d]	
Operands	f : 00h ~ 1FFh, d : 0, 1	
Operation	(destination) ← (f) - 1	
Status Affected	Z	
OP-Code	ff00 0011 dfff ffff	
Description	Decrement register 'f'. If 'd' is 0, the result is stored in the W register. If 'd' is 1, the result is stored back in register 'f'.	
Cycle	1	
Example	DECX CNT, 1	B : CNT =0x01, Z =0 A : CNT =0x00, Z =1

DECXSZ	Decrement "f", Skip if 0	
Syntax	DECXSZ f [,d]	
Operands	f : 00h ~ 1FFh, d : 0, 1	
Operation	(destination) ← (f) - 1, skip next instruction if result is 0	
Status Affected	-	
OP-Code	ff00 1011 dfff ffff	
Description	The contents of register 'f' are decremented. If 'd' is 0, the result is placed in the W register. If 'd' is 1, the result is placed back in register 'f'. If the result is 1, the next instruction is executed. If the result is 0, then a NOP is executed instead, making it a 2 cycle instruction.	
Cycle	1 or 2	
Example	LABEL1 DECXSZ CNT, 1 LGOTO LOOP CONTINUE	B : PC =LABEL1 A : CNT =CNT – 1 if CNT =0, PC =CONTINUE if CNT ≠0, PC =LABEL1 + 1

LGOTO	Unconditional Branch	
Syntax	LGOTO k	
Operands	k : 000h ~ 1FFFh	
Operation	PC.10~0 ← k	
Status Affected	-	
OP-Code	kk10 1kkk kkkk kkkk	
Description	LGOTO is an unconditional branch. The 13-bit immediate value is loaded into PC bits <12:0>.The upper bits of PC are loaded from PCLATH. LGOTO is a two-cycle instruction.	
Cycle	2	
Example	LABEL1 LGOTO SUB1	B : PC =LABEL1 A : PC =SUB1

INCX	Increment "f"	
Syntax	INCX f [,d]	
Operands	f : 00h ~ 1FFh	
Operation	(destination) ← (f) + 1	
Status Affected	Z	
OP-Code	ff00 1010 dfff ffff	
Description	The contents of register 'f' are incremented. If 'd' is 0, the result is placed in the W register. If 'd' is 1, the result is placed back in register 'f'.	
Cycle	1	
Example	INCX CNT, 1	B : CNT =0xFF, Z =0 A : CNT =0x00, Z =1

INCXSZ	Increment "f", Skip if 0	
Syntax	INCXSZ f [,d]	
Operands	f : 00h ~ 1FFh, d : 0, 1	
Operation	(destination) ← (f) + 1, skip next instruction if result is 0	
Status Affected	-	
OP-Code	ff00 1111 dfff ffff	
Description	The contents of register 'f' are incremented. If 'd' is 0, the result is placed in the W register. If 'd' is 1, the result is placed back in register 'f'. If the result is 1, the next instruction is executed. If the result is 0, a NOP is executed instead, making it a 2 cycle instruction.	
Cycle	1 or 2	
Example	LABEL1 INCXSZ CNT, 1 LGOTO LOOP CONTINUE	B : PC =LABEL1 A : CNT =CNT + 1 if CNT =0, PC =CONTINUE if CNT ≠0, PC =LABEL1 + 1

IORLW	Inclusive OR Literal with W	
Syntax	IORLW k	
Operands	k : 00h ~ FFh	
Operation	(W) ← (W) OR k	
Status Affected	Z	
OP-Code	0001 1010 kkkk kkkk	
Description	The contents of the W register are OR'ed with the eight-bit literal 'k'. The result is placed in the W register.	
Cycle	1	
Example	IORLW 0x35	B : W =0x9A A : W =0xBF, Z =0

IORWX	Inclusive OR W with "f"	
Syntax	IORWF f [,d]	
Operands	f : 00h ~ 1FFh, d : 0, 1	
Operation	(destination) ← (W) OR k	
Status Affected	Z	
OP-Code	ff00 0100 dfff ffff	
Description	Inclusive OR the W register with register 'f'. If 'd' is 0, the result is placed in the W register. If 'd' is 1, the result is placed back in register 'f'.	
Cycle	1	
Example	IORWX RESULT, 0	B : RESULT =0x13, W =0x91 A : RESULT =0x13, W =0x93, Z =0

MOVX	Move f	
Syntax	MOVX f,d	
Operands	f : 00h ~ 1FFh	
Operation	(destination) ← (f)	
Status Affected	Z	
OP-Code	ff00 1000 dfff ffff	
Description	The contents of register 'f' are moved to a destination dependent upon the status of d. If d=0, destination is W register. If d =1, the destination is file register f itself. d=1 is useful to test a file register, since status flag Z is affected.	
Cycle	1	
Example	MOVX FSR,0	B : FSR =0xC2, W =? A : FSR =0xC2, W 0xC2

MOVXW	Move "f" to W	
Syntax	MOVXW f	
Operands	f : 00h ~ 1FFh	
Operation	(W) ← (f)	
Status Affected	Z	
OP-Code	ff00 1000 0fff ffff	
Description	The contents of register 'f' are moved to W register.	
Cycle	1	
Example	MOVXW FSR	B : FSR =0xC2, W =? A : FSR =0xC2, W 0xC2

MOVLW	Move Literal to W	
Syntax	MOVLW k	
Operands	k : 00h ~ FFh	
Operation	(W) ← k	
Status Affected	-	
OP-Code	0001 1001 kkkk kkkk	
Description	The eight-bit literal 'k' is loaded into W register. The don’t cares will assemble as 0’s.	
Cycle	1	
Example	MOVLW 0x5A	B : W =? A : W =0x5A

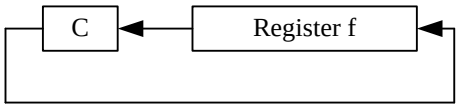
MOVWX	Move W to "f"	
Syntax	MOVWX f	
Operands	f : 00h ~ 1FFh	
Operation	(f) ← (W)	
Status Affected	-	
OP-Code	ff00 0000 1fff ffff	
Description	Move data from W register to register 'f'.	
Cycle	1	
Example	MOVWX REG1	B : REG1 =0xFF, W =0x4F A : REG1 =0x4F, W =0x4F

NOP	No Operation	
Syntax	NOP	
Operands	-	
Operation	No Operation	
Status Affected	-	
OP-Code	0000 0000 0000 0000	
Description	No Operation	
Cycle	1	
Example	NOP	-

RET	Return from Subroutine	
Syntax	RET	
Operands	-	
Operation	PC ← TOS	
Status Affected	-	
OP-Code	0000 0000 0100 0000	
Description	Return from subroutine. The stack is POPed and the top of the stack (TOS) is loaded into the program counter. This is a two-cycle instruction.	
Cycle	2	
Example	RET	A : PC =TOS

RETI	Return from Interrupt	
Syntax	RETI	
Operands	-	
Operation	PC ← TOS, GIE ← 1	
Status Affected	-	
OP-Code	0000 0000 0110 0000	
Description	Return from Interrupt. Stack is POPed and Top-of-Stack (TOS) is loaded in to the PC. Interrupts are enabled. This is a two-cycle instruction.	
Cycle	2	
Example	RETI	A : PC =TOS, GIE =1

RETLW	Return with Literal in W
Syntax	RETLW k
Operands	k : 00h ~ FFh
Operation	PC ← TOS, (W) ← k
Status Affected	-
OP-Code	0001 1000 kkkk kkkk
Description	The W register is loaded with the eight-bit literal 'k'. The program counter is loaded from the top of the stack (the return address). This is a two-cycle instruction.
Cycle	2
Example	LCALL TABLE : TABLE ADDWX PCL, 1 RETLW k1 RETLW k2 : RETLW kn B : W =0x07 A : W =value of k8

RLX	Rotate Left "f" through Carry
Syntax	RLX f [,d]
Operands	f : 00h ~ 1FFh, d : 0, 1
Operation	
Status Affected	C
OP-Code	ff00 1101 dfff ffff
Description	The contents of register 'f' are rotated one bit to the left through the Carry Flag. If 'd' is 0, the result is placed in the W register. If 'd' is 1, the result is stored back in register 'f'.
Cycle	1
Example	RLX REG1, 0 B : REG1 =1110 0110, C =0 A : REG1 =1110 0110 W =1100 1100, C =1

RRX	Rotate Right "f" through Carry
Syntax	RRX f [,d]
Operands	f : 00h ~ 1FFh, d : 0, 1
Operation	
Status Affected	C
OP-Code	ff00 1100 dfff ffff
Description	The contents of register 'f' are rotated one bit to the right through the Carry Flag. If 'd' is 0, the result is placed in the W register. If 'd' is 1, the result is placed back in register 'f'.
Cycle	1
Example	RRX REG1, 0 B : REG1 =1110 0110, C =0 A : REG1 =1110 0110 W =0111 0011, C =0

SLEEP	Go into Power-down mode, Clock oscillation stops	
Syntax	SLEEP	
Operands	-	
Operation	-	
Status Affected	TO, PD	
OP-Code	001 1110 0000 0011	
Description	Go into Power-down mode with the oscillator stops.	
Cycle	1	
Example	SLEEP	-

SUBLW	Subtract W from Literal	
Syntax	SUBLW k	
Operands	k : 00h ~ FFh	
Operation	(W) ← k – (W)	
Status Affected	C, DC, Z	
OP-Code	0001 1111 kkkk kkkk	
Description	The W register is subtracted (2’s complement method) from the eight-bit literal “k”. The result is placed in the W register.	
Cycle	1	
Example	SUBLW 0x25	B : W =0x15 A : W =0x10

SUBWX	Subtract W from "f"	
Syntax	SUBWX f [,d]	
Operands	f : 00h ~ 1FFh, d : 0, 1	
Operation	(destination) ← (f) – (W)	
Status Affected	C, DC, Z	
OP-Code	ff00 0010 dfff ffff	
Description	Subtract (2’s complement method) W register from register 'f'. If 'd' is 0, the result is stored in the W register. If 'd' is 1, the result is stored back in register 'f'.	
Cycle	1	
Example	SUBWX REG1, 1	B : REG1 =0x03, W =0x02, C =?, Z =? A : REG1 =0x01, W =0x02, C =1, Z =0
	SUBWX REG1, 1	B : REG1 =0x02, W =0x02, C =?, Z =? A : REG1 =0x00, W =0x02, C =1, Z =1
	SUBWX REG1, 1	B : REG1 =0x01, W =0x02, C =?, Z =? A : REG1 =0xFF, W =0x02, C =0, Z =0

SUBWXB Subtract W and $\overline{C}$ from "f"

Syntax	SUBWXB f [,d]
Operands	f : 000h ~ 1FFh, d : 0, 1
Operation	(destination) ← (f) – (W) – $\overline{C}$
Status Affected	C, DC, Z
OP-Code	ff01 0100 dfff ffff
Description	Subtract (2's complement method) W register and $\overline{C}$ from register 'f'. If 'd' is 0, the result is stored in the W register. If 'd' is 1, the result is stored back in register 'f'
Cycle	1
Example	SUBWXB REG1, 1 — B : REG1 =0x19, W =0x0D, C =1, Z =? A : REG1 =0x0C,W =0x0D, C =1, Z =0
	SUBWXB REG1, 1 — B : REG1 =0x1B, W =0x1A, C =0 Z =? A : REG1 =0x00, W =0x1A, C =1, Z =1
	SUBWXB REG1, 1 — B : REG1 =0x03, W =0x0E, C =1, Z =? A : REG1 =0xF5, W =0x0E, C =0, Z =0

SUBXW Subtract "f" from W

Syntax	SUBXW f [,d]
Operands	f : 000h ~ 1FFh, d : 0, 1
Operation	(destination) ←(W) – (f)
Status Affected	C, DC, Z
OP-Code	ff01 0010 dfff ffff
Description	Subtract (2's complement method) register 'f' from W. If 'd' is 0, the result is stored in the W register. If 'd' is 1, the result is stored back in register 'f'
Cycle	1
Example	SUBXW REG1, 1 — B : REG1 =0x02, W =0x03, C =?, Z =? A : REG1 =0x01, W =0x03, C =1, Z =0
	SUBXW REG1, 1 — B : REG1 =0x02, W =0x02, C =?, Z =? A : REG1 =0x00, W =0x02, C =1, Z =1
	SUBXW REG1, 1 — B : REG1 =0x02, W =0x01, C =?, Z =? A : REG1 =0xFF, W =0x02, C =0, Z =0

SUBXWB Subtract "f" and $\overline{C}$ from W

Syntax	SUBXWB f [,d]
Operands	f : 000h ~ 1FFh, d : 0, 1
Operation	(目标) ←(W) – (f) – $\overline{C}$
Status Affected	C, DC, Z
OP-Code	ff01 0011 dfff ffff
Description	Subtract (2's complement method) register 'f' and $\overline{C}$ from W. If 'd' is 0, the result is stored in the W register. If 'd' is 1, the result is stored back in register 'f'
Cycle	1
Example	SUBXWB REG1, 1 — B : REG1 =0x03, W =0x02, C =1, Z =? A : REG1 =0xFF, W =0x02, C =0, Z =0
	SUBXWB REG1, 1 — B : REG1 =0x02, W =0x05, C =1, Z =? A : REG1 =0x03, W =0x05, C =1, Z =0
	SUBXWB REG1, 1 — B : REG1 =0x01, W =0x02, C =0, Z =? A : REG1 =0x00, W =0x02, C =1, Z =1

SWAPX	Swap Nibbles in "f"
Syntax	SWAPX f [,d]
Operands	f : 00h ~ 1FFh, d : 0, 1
Operation	(destination,7~4) ← (f.3~0), (destination.3~0) ← (f.7~4)
Status Affected	-
OP-Code	ff00 1110 dfff ffff
Description	The upper and lower nibbles of register 'f' are exchanged. If 'd' is 0, the result is placed in W register. If 'd' is 1, the result is placed in register 'f'.
Cycle	1
Example	SWAPX REG, 0 B : REG1 =0xA5 A : REG1 =0xA5, W =0x5A

TABRH	Return DPTR high byte to W
Syntax	TABRH
Operands	-
Operation	(W) ← ROM[DPTR] high byte content, Where DPTR = {DPH[max:8], FSR[7:0]}
Status Affected	-
OP-Code	0000 0000 0101 1000
Description	The W and TABR register is loaded with high byte of ROM[DPTR]. This is a two-cycle instruction.
Cycle	2
Example	MOVLW (TAB1&0xFF) MOVWX DPL ;Where DPL is register MOVLW (TBA1>>8)&0xFF MOVWX DPH ;Where DPH is register TABRL ;W =0x89, TABR=0x89 TABRH ;W =0x37, TABR=0x37 ORG 0234H TAB1: DT 0x3789, 0x2277 ;ROM data 16 bits

TABRL	Return DPTR low byte to W
Syntax	TABRL
Operands	-
Operation	(W) ← ROM[DPTR] low byte content, Where DPTR = {DPH[max:8], FSR[7:0]}
Status Affected	-
OP-Code	0000 0000 0101 0000
Description	The W and TABR register is loaded with low byte of ROM[DPTR]. This is a two-cycle instruction.
Cycle	2
Example	MOVLW (TAB1&0xFF) MOVWX DPL ;Where DPL register MOVLW (TBA1>>8)&0xFF MOVWX DPH ;Where DPH register TABRL ;W =0x89, TABR=0x89 TABRH ;W =0x37, TABR=0x37 ORG 0234H TAB1: DT 0x3789, 0x2277 ;ROM data 16 bits

TSTX	Test if "f" is zero	
Syntax	TSTX f	
Operands	f : 00h ~ 1FFh	
Operation	Set Z flag if (f) is 0	
Status Affected	Z	
OP-Code	ff00 1000 1fff ffff	
Description	If the content of register 'f' is 0, Zero flag is set to 1.	
Cycle	1	
Example	TSTX REG1	B : REG1 =0, Z =? A : REG1 =0, Z =1

XORLW	Exclusive OR Literal with W	
Syntax	XORLW k	
Operands	k : 00h ~ FFh	
Operation	(W) ← (W) XOR k	
Status Affected	Z	
OP-Code	0001 1101 kkkk kkkk	
Description	The contents of the W register are XOR'ed with the eight-bit literal 'k'. The result is placed in the W register.	
Cycle	1	
Example	XORLW 0xAF	B : W =0xB5 A : W =0x1A

XORWX	Exclusive OR W with "f"	
Syntax	XORWX f [,d]	
Operands	f : 00h ~ 1FFh, d : 0, 1	
Operation	(destination) ← (W) XOR (f)	
Status Affected	Z	
OP-Code	ff00 0110 dfff ffff	
Description	Exclusive OR the contents of the W register with register 'f'. If 'd' is 0, the result is stored in the W register. If 'd' is 1, the result is stored back in register 'f'.	
Cycle	1	
Example	XORWX REG, 1	B : REG =0xAF, W =0xB5 A : REG =0x1A, W =0xB5

ELECTRICAL CHARACTERISTICS

1. Absolute Maximum Ratings (T_A =25℃)

Parameter	Rating	Unit
Supply voltage	V_{SS} -0.3 to V_{SS} +5.5	V
Input voltage	V_{SS} -0.3 to V_{CC} +0.3	
Output voltage	V_{SS} -0.3 to V_{CC} +0.3	
Output current high per 1 PIN	-25	mA
Output current high per all PIN	-80	
Output current low per 1 PIN	+30	
Output current low per all PIN	+150	
Maximum operating voltage	5.5	V
Operating temperature	-40 to +105	℃
Storage temperature	-65 to +150	

2. DC Characteristics (T_A =25℃, V_{DD} =5.0V, unless otherwise specified)

Parameter	Sym	Conditions		Min	Typ	Max	Unit
Operating Voltage	V_{cc}	Fsys=18.432Mhz (RDCTL=4ns)		2.6		5.5	
		Fsys= 9.2Mhz (RDCTL=4ns)		1.7		5.5	
Input High Voltage	V_{IH}	All Input	V_{CC} = 3~5V	0.6Vcc	–	Vcc	V
Input Low Voltage	V_{IL}	All Input	V_{CC} = 3~5V	Vss	–	0.2Vcc	V
Output High Current	I_{OH}	PB6/PB7/PC0/ PC1/PC2/PC4/ PD0/PD1/PD2/ PD3/PD4/PD5 Output	V_{CC} = 5V,V_{OH} = 4.5V LED[x]=0	12.6	12.7	–	mA
			V_{CC} = 5V,V_{OH} = 3.5V LED[x]=1	18.5	18.8	–	
			V_{CC} = 3V,V_{OH} = 2.7V LED[x]=0	5.2	5.3	–	
			V_{CC} = 3V,V_{OH} = 2.1V LED[x]=1	8.1	8.2	–	
		PA0/PA1/PA2/ PA3/PA4/PA7/ PB0/PB1/PB2/ PB3/PB4/PB5/ PD6/PD7 Output	V_{CC} = 5V,V_{OH} = 4.5V LED[x]=0	10.7	10.9	–	
			V_{CC} = 5V,V_{OH} = 3.5V LED[x]=1	17.4	17.6	–	
			V_{CC} = 3V,V_{OH} = 2.7V LED[x]=0	4.6	4.7	–	
			V_{CC} = 3V,V_{OH} = 2.1V LED[x]=1	7.7	7.8	–	
Output Low Current	I_{OL}	PB6/PB7/PC0/ PC1/PC2/PC4/ PD0/PD1/PD2/ PD3/PD4/PD5 Output	V_{CC} = 5V,V_{OL} = 0.5V HSNK[x]=0	42.3	44.7	–	mA
			V_{CC} = 5V,V_{OL} = 0.5V HSNK[x]=1	71.5	80.7	–	
			V_{CC} = 3V,V_{OL} = 0.3V HSNK[x]=0	18.8	19.5	–	
			V_{CC} = 3V,V_{OL} = 0.3V HSNK[x]=1	33	36.1	–	
		PA0/PA1/PA2/ PA3/PA4/PA7/ PB0/PB1/PB2/ PB3/PB4/PB5/ PD6/PD7 Output	V_{CC} = 5V,V_{OL} = 0.5V HSNK[x]=0	30.5	31.2	–	
			V_{CC} = 5V,V_{OL} = 0.5V HSNK[x]=1	44	44.8	–	
			V_{CC} = 3V,V_{OL} = 0.3V HSNK[x]=0	14.5	14.8	–	
			V_{CC} = 3V,V_{OL} = 0.3V HSNK[x]=1	22.2	22.6	–	

Parameter	Sym	Conditions		Min	Typ	Max	Unit
Input Leakage Current (pin high)	I_{ILH}	All Input	$V_{IN} = V_{CC}$	–	–	1	uA
Input Leakage Current (pin low)	I_{ILL}	All Input	$V_{IN} = 0V$	–	–	–1	uA
Power Supply Current (No Load) ATD ON	I_{CC}	FAST mode FXT 20MHz	$V_{CC} = 5V$	–	6.6	–	mA
			$V_{CC} = 3V$	–	3.6	–	
		FAST mod FIRC 18.4 MHz	$V_{CC} = 5V$	–	6.2	–	
			$V_{CC} = 3V$	–	3.5	–	
		FAST mode FIRC 9.2 MHz	$V_{CC} = 5V$	–	4.1	–	
			$V_{CC} = 3V$	–	2.4	–	
		FAST mode FIRC 4.6 MHz	$V_{CC} = 5V$	–	2.7	–	
			$V_{CC} = 3V$	–	1.8	–	
		FAST mode FIRC 2.3 MHz	$V_{CC} = 5V$	–	1.9	–	
			$V_{CC} = 3V$	–	1.3	–	
		SLOW mode SXT 32KHz FIRC Stop POR/LVR ON	$V_{CC} = 5.0V$	–	0.25	–	mA
			$V_{CC} = 3.0V$	–	0.2	–	
		SLOW mode SXT 32KHzz FIRC Stop POR/LVR OFF	$V_{CC} = 5.0V$	–	0.02	–	
			$V_{CC} = 3.0V$	–	0.04	–	
		SLOW mode SIRC 50KHz FIRC Stop POR/LVR ON	$V_{CC} = 5.0V$	–	0.2	–	
			$V_{CC} = 3.0V$	–	0.17	–	
		SLOW mode SIRC 50KHz FIRC Stop POR/LVR OFF	$V_{CC} = 5.0V$	–	0.03	–	
			$V_{CC} = 3.0V$	–	0.02	–	
		IDLE mode SIRC 50 KHz POR/LVR OFF	$V_{CC} = 5.0V$	–	7	–	uA
			$V_{CC} = 3.0V$	–	2	–	
		STOP mode POR/LVR OFF	$V_{CC} = 5.0V$	–	-	1	uA
			$V_{CC} = 3.0V$	–	-	1	
Pull-up Resistor	R_{UP}	V_{IN} = 0 V Ports A/B/C/D	$V_{CC} = 5.0V$	–	33	–	KΩ
			$V_{CC} = 3.0V$	–	57	–	

3. Clock Timing (T_A = -40°C to +85°C)

Parameter	Condition	Min	Typ	Max	Unit
Parameter FIRC Frequency (*)	**Condition**	**Min**	**Typ**	**Max**	MHz
	25℃, V_{CC} =5.0 V	-1%	18.432	+1%	
	-40℃ ~ 105℃, V_{CC} = 5.0 V	-1.5%	18.432	+1.5%	

(*) FIRC frequency can be divided by 1/2/4/8.

4. Reset Timing Characteristics (T_A = 25°C)

Parameter	Conditions	Min	Typ	Max	Unit
RESET Input Low width	Input V_{CC} = 5 V ±10 %	-	30	–	μs
WDT time	V_{CC} = 3 V, WDTPSC = 11	–	14140	–	ms
	V_{CC} = 5 V, WDTPSC = 11		1277		
WKT time	V_{CC} = 3 V, WKTPSC = 11	–	177	–	ms
	V_{CC} = 5 V, WKTPSC = 11		160		
CPU start up time	V_{CC} = 5 V	–	38	–	ms

5. EEPROM Characteristics

Parameter	Conditions	Min	Typ	Max	Unit
Write Voltage V_{EEWR}	-40°C ~ 105° Fsys=FRC/1, V_{CC} /47uF	2.5		5.5	V
*Write Endurance N_{EE}	V_{CC} =3.0~5.0V, 25℃	50K	–	–	cycles
	V_{CC} =3.0~5V -20℃ ~105℃	20K	–	–	

Note: The value of this parameter is based on the characteristics of tested samples

6. LVR Circuit Characteristics (T_A = 25℃)

Parameter	Sym	Conditions	Min	Typ	Max	Unit
LVR Reference Voltage	LVR_{th}	T_A = 25℃	–	3.51	–	V
			–	3.39	–	
			–	3.27	–	
			–	3.14	–	
			–	3.01	–	
			–	2.88	–	
			–	2.75	–	
			–	2.64	–	
			–	2.51	–	
			–	2.39	–	
			–	2.27	–	
			–	2.15	–	
			–	2.01	–	
			–	1.89	–	
			–	1.75	–	
			–	1.63	–	
LVD Reference Voltage	LVD_{th}	T_A = 25℃	–	3.51	–	V
			–	3.39	–	
			–	3.27	–	
			–	3.14	–	
			–	3.01	–	
			–	2.88	–	
			–	2.75	–	
			–	2.64	–	
			–	2.51	–	
			–	2.39	–	
			–	2.27	–	
			–	2.15	–	
			–	2.01	–	
			–	1.89	–	
			–	1.75	–	
			–	–	–	
LVR Hysteresis Voltage	V_{HYS_LVR}	T_A = 25℃	–	±0.1	–	V
Low Voltage Detection time	T_{LVR}	T_A = 25℃	100	–	–	μs

7. ADC Electrical Characteristics (T$_A$ = 25℃, VCC = 3.0V to 5.5V, VSS = 0V)

Total Accuracy	Conditions	Min	Typ	Max	Units
Integral Non-Linearity	V_{CC} = 5V, V_{SS} = 0V, F_{ADC} = 1 MHz	–	±3	±13	LSB
Differential Non-linearity		–	±3.2	±15	
Max Input Clock freq. (F_{ADC})		–	±1	±4	
Conversion Time	Source impedance (Rs<10K omh)	–	–	2	MHz
	Source impedance (Rs<20K omh)	–	–	1	
	Source impedance (Rs<50K omh)	–	–	0.5	
	Source is VBG (ADCHS=01100)	–	–	2	
BandGap Voltage Reference (VBG)	F_{ADC} = 1 MHz	–	50	–	μs
ADC reference voltage(V_{REF}) (VBGSEL =01b)	-20℃~105℃, V_{CC} = 3.0~5.0V	-2%	1.18	+1.5%	V
Vcc/4 reference voltage	-20℃~105℃, V_{CC} = 3.0~5.0V	-2.5%	2.5	+2%	
Input Voltage	25℃, V_{CC} = 3.0~5.0V	-1%	0.25*Vcc	1%	
Total Accuracy	–	V_{SS}	–	Vcc	

8. Temperature Sensor Characteristics

Parameter	Condition	Min	Typ	Max	Unit
Supply voltage	–	3.2	–	5.5	V
Temperature slope Volt_slope note1	–		5		mV/℃
ADC slope ADC_slpoe note 1	ADCVREF=VBG3.0V	–	6.9	–	LSB/℃
Relative temperature linearity T_L note 4	Temp = 0~70℃, Factory Calibration	-2		2	℃
	Temp = -20~+100℃, Factory Calibration	-4		4	℃
VTEMPDH (19Fh.7~0) VTEMPDL (19Eh.7~4) note 4	Temperature sensor channel ADC data @25℃ (+/-3℃), ADCVREF=VBG3.0V				

Note 1: Guaranteed by design, not production tested

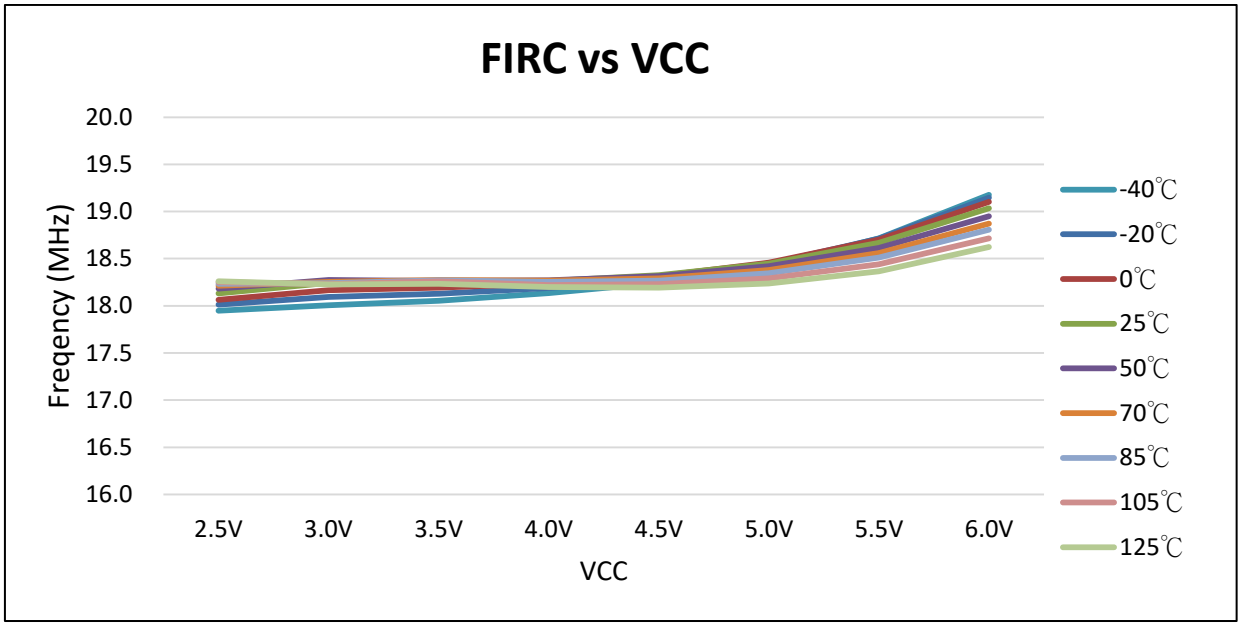
Note 2: When the temperature sensor channel is enabled, the ADC can be sampled after a delay of 100uS

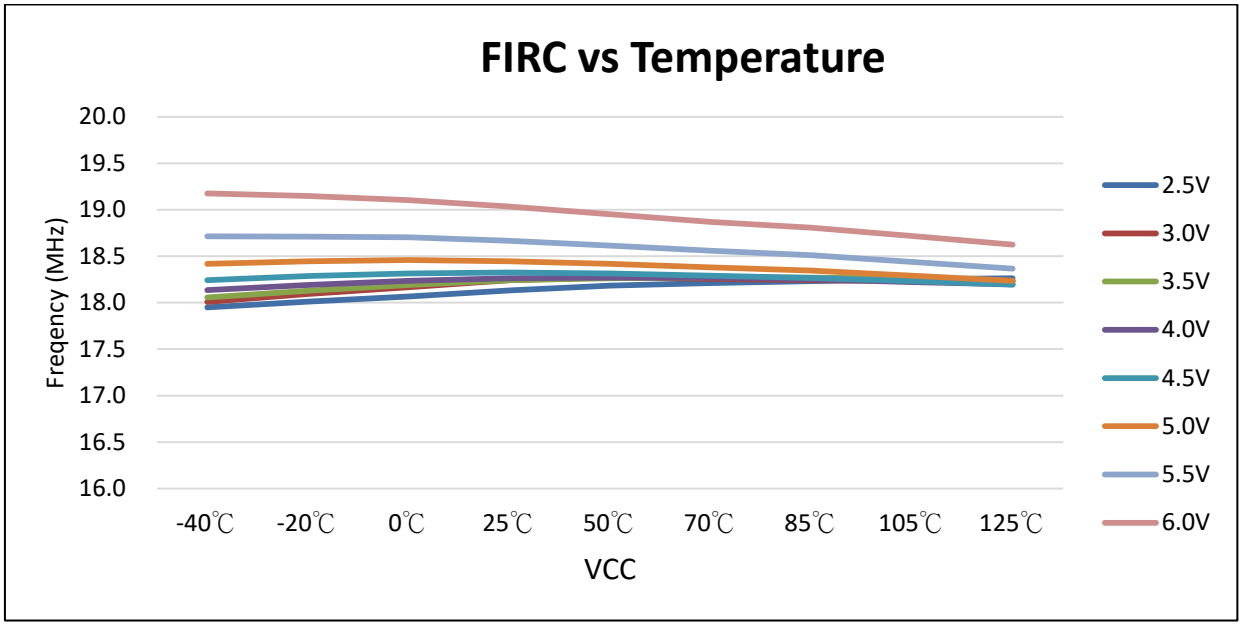
Note 3: Due to process differences, the offset of the internal temperature sensor depends on the chip. It is not calibrated and is only suitable for detecting relative temperature changes.

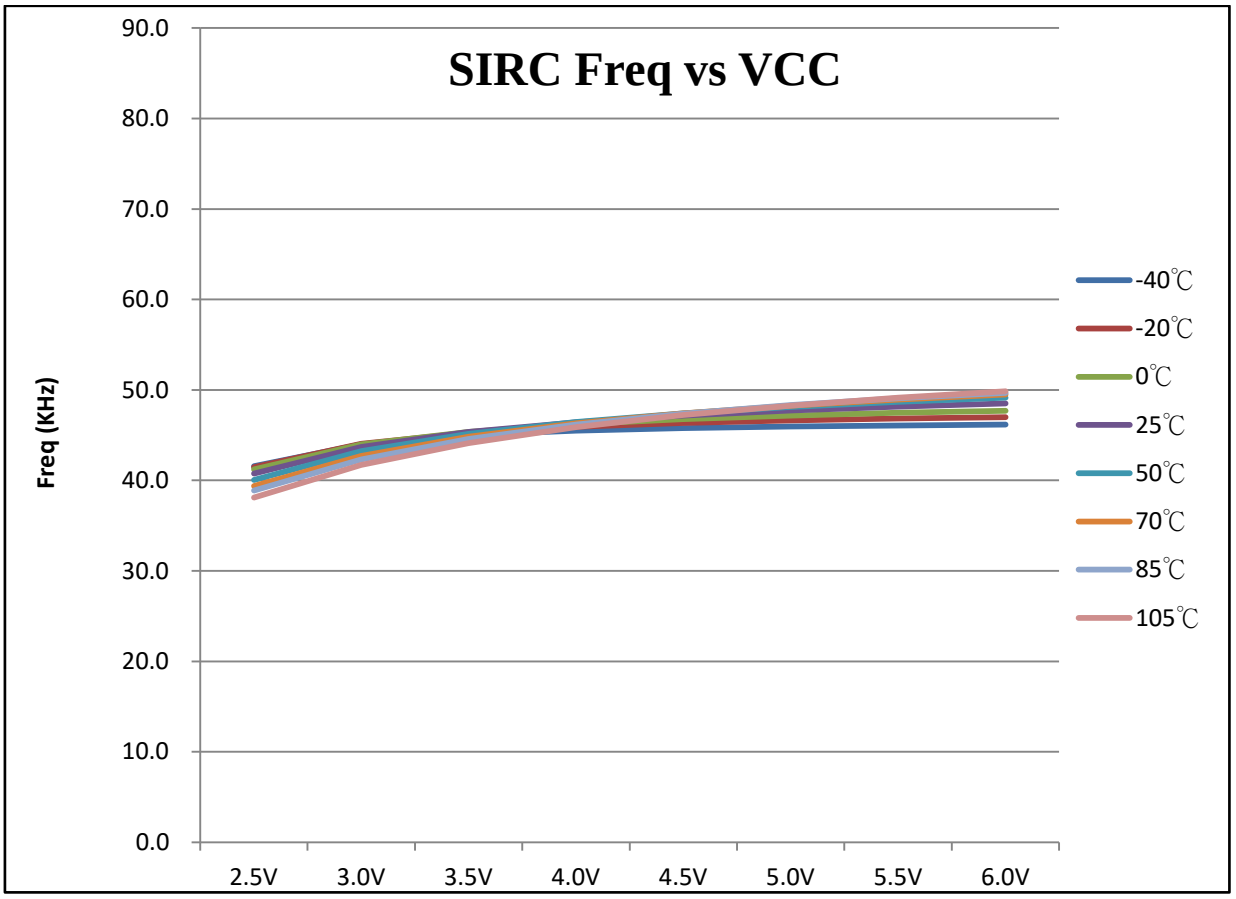
Note 4: To improve the measurement accuracy, each chip is calibrated independently before leaving the factory and stored in VTEMPDH/L (read only)

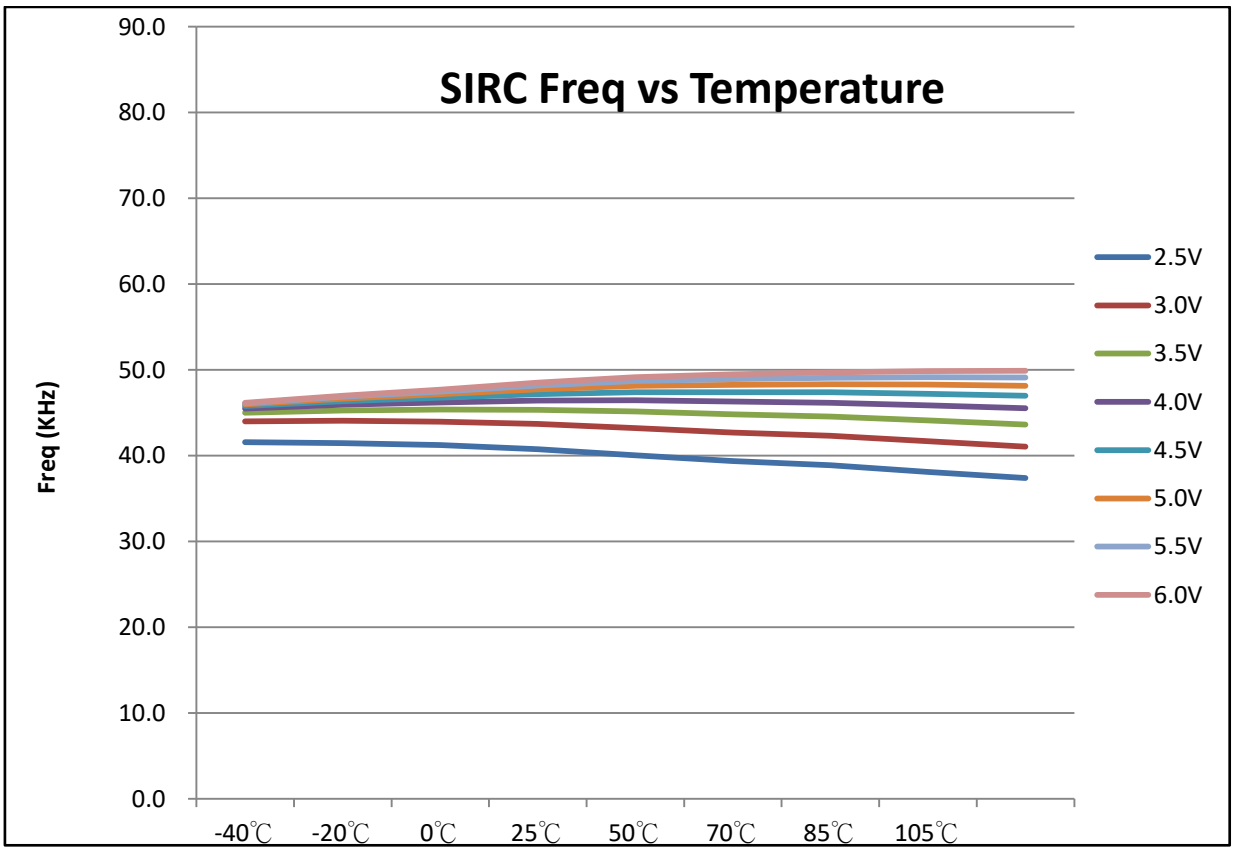
Note 5: Temp = 25 ℃ + (ADC data-VTEMPDHL) / 6.9

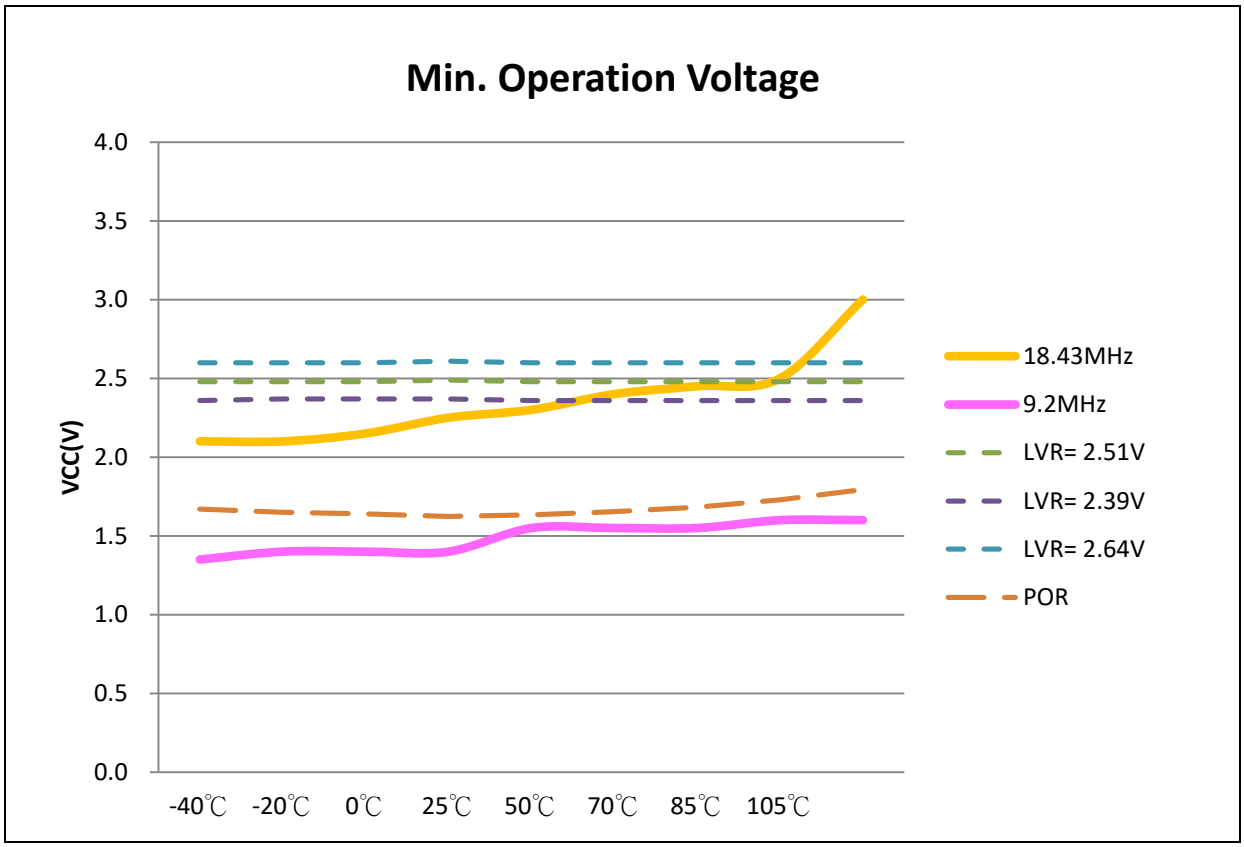
9. Characteristic Graphs



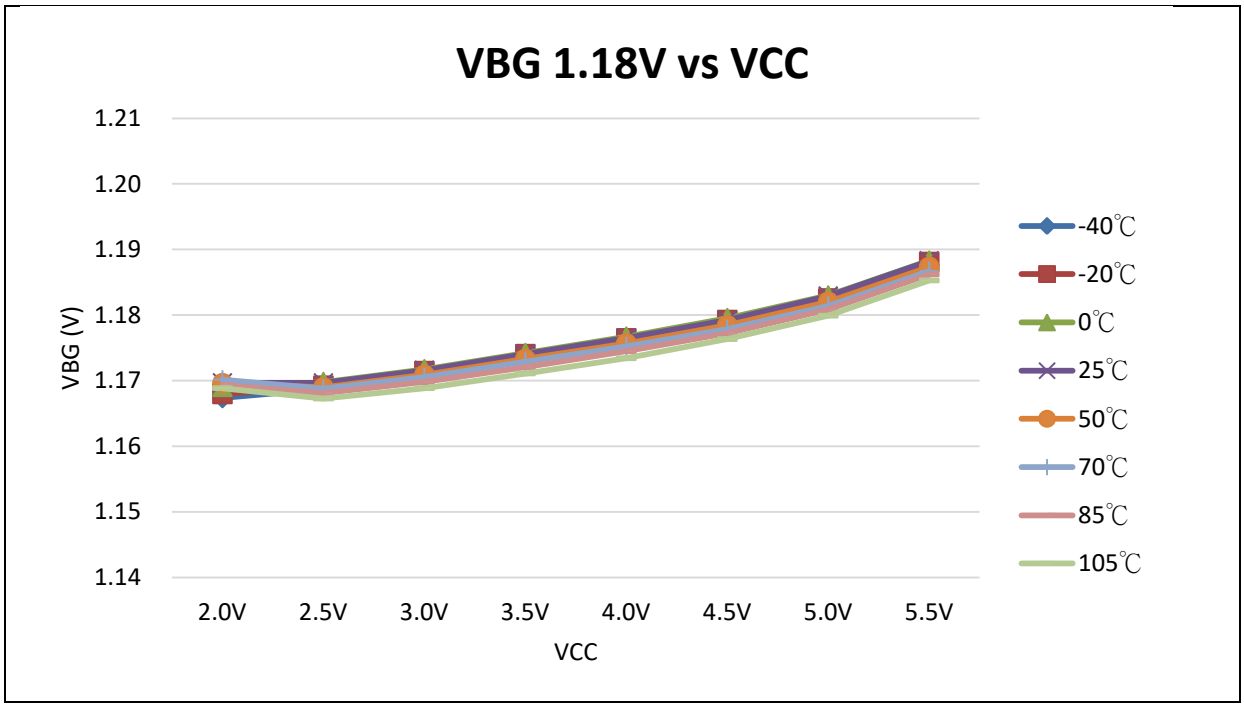



SIRC Freq vs VCC
Freq (KHz)
90.0
80.0
70.0
60.0
50.0
40.0
30.0
20.0
10.0
0.0
2.5V
3.0V
3.5V
4.0V
4.5V
5.0V
5.5V
6.0V
-40℃
-20℃
0℃
25℃
50℃
70℃
85℃
105℃


SIRC Freq vs Temperature
Freq (KHz)
90.0
80.0
70.0
60.0
50.0
40.0
30.0
20.0
10.0
0.0
-40℃
-20℃
0℃
25℃
50℃
70℃
85℃
105℃
2.5V
3.0V
3.5V
4.0V
4.5V
5.0V
5.5V
6.0V



User must switch RDCTL to "4ns" to improve the performance at the lowest operating voltage



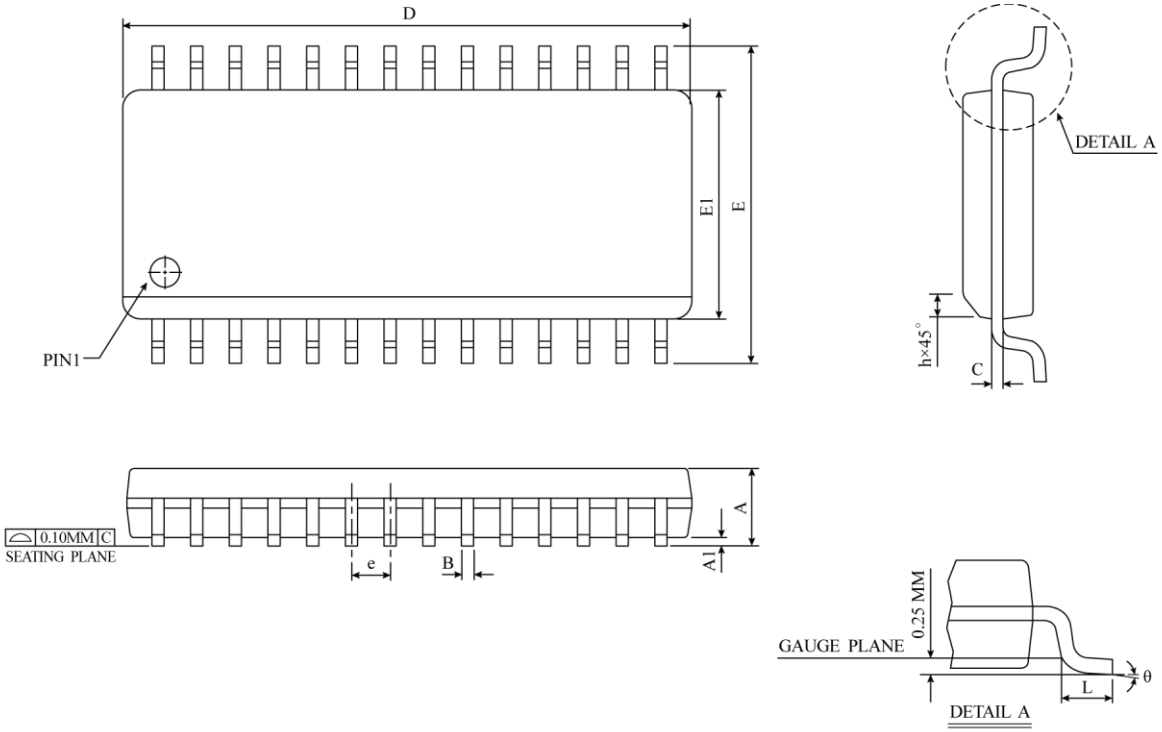
PACKAGING INFORMATION

Please note that the package information provided is for reference only. Since this information is frequently updated, users can contact Sales to consult the latest package information and stocks.

The ordering information:

Ordering number	Package
TM56F6922-MTP	Wafer/Dice blank chip
TM56F6922-COD	Wafer/Dice with code
TM56F69225S	SOP 28-pin (300 mil)
TM56F69225E	SSOP 28-pin (150 mil)
TM56F69223T	TSSOP 20-pin (173 mil)
TM56F69223S4	SOP 20-pin (300 mil)

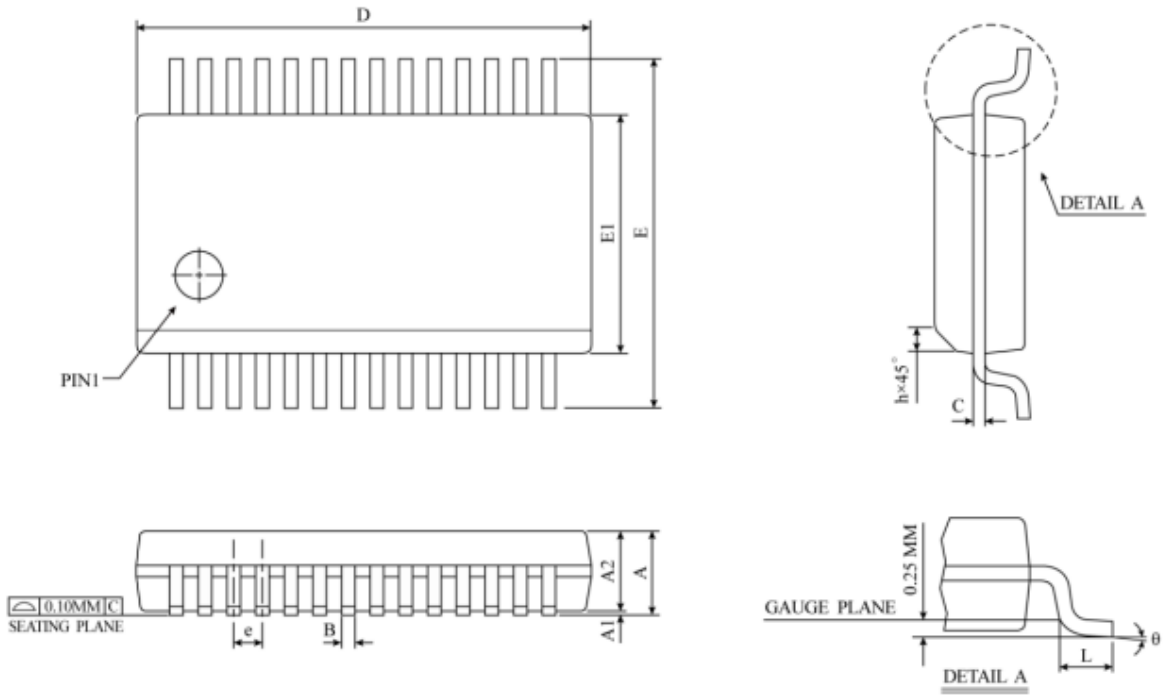
✧ SOP-28 (300mil) Package Dimension



SYMBOL	DIMENSION IN MM			DIMENSION IN INCH		
	MIN	NOM	MAX	MIN	NOM	MAX
A	2.35	2.50	2.65	0.0926	0.0985	0.1043
A1	0.10	0.20	0.30	0.0040	0.0079	0.0118
B	0.33	0.42	0.51	0.0130	0.0165	0.0200
C	0.23	0.28	0.32	0.0091	0.0108	0.0125
D	17.70	17.90	18.10	0.6969	0.7047	0.7125
E	10.00	10.33	10.65	0.3940	0.4425	0.4910
E1	7.40	7.50	7.60	0.2914	0.2953	0.2992
e	1.27 BSC			0.050 BSC		
h	0.25	0.50	0.75	0.0100	0.0195	0.0290
L	0.40	0.84	1.27	0.0160	0.0330	0.0500
θ	0°	4°	8°	0°	4°	8°
JEDEC	MS-013 (AE)					

⚠ * NOTES : DIMENSION " D " DOES NOT INCLUDE MOLD FLASH, PROTRUSIONS OR GATE BURRS. MOLD FLASH, PROTRUSIONS AND GATE BURRS SHALL NOT EXCEED 0.15 MM (0.006 INCH) PER SIDE.

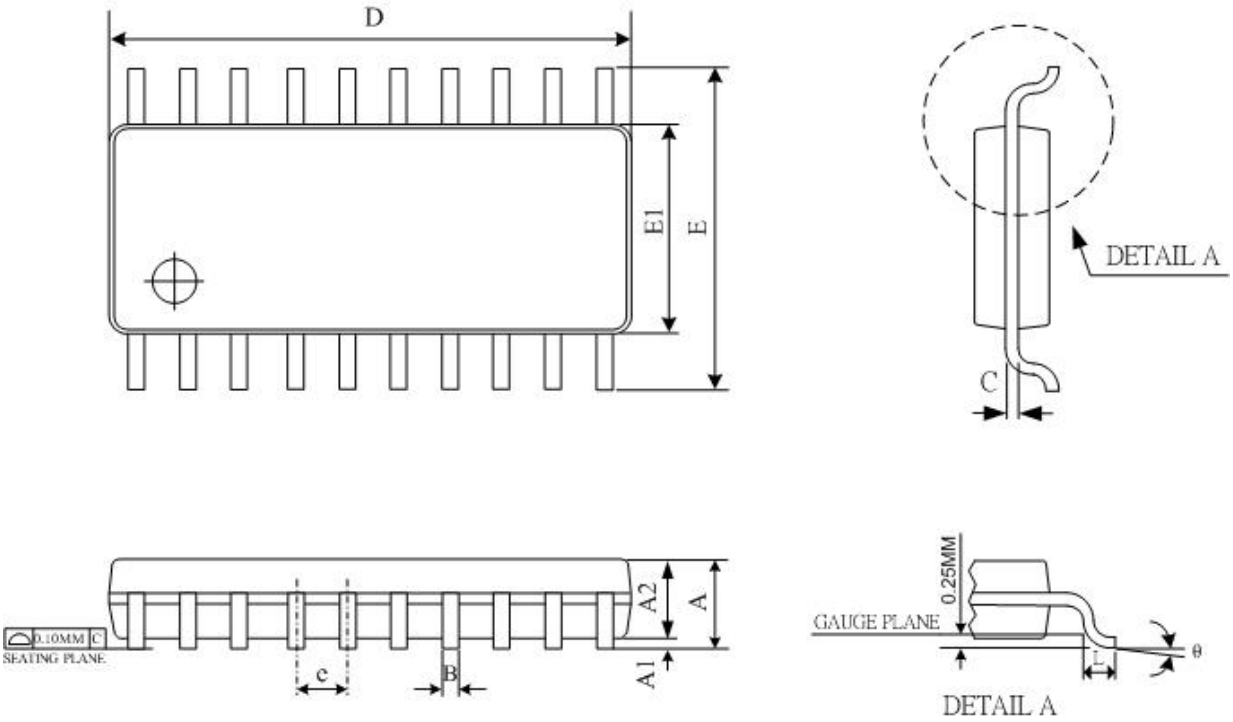
✧ SSOP-28 (150mil) Package Dimension



SYMBOL	DIMENSION IN MM			DIMENSION IN INCH		
	MIN	NOM	MAX	MIN	NOM	MAX
A	1.50	1.65	1.80	0.06	0.06	0.07
A1	0.102	0.176	0.249	0.004	0.007	0.010
A2	1.40	1.475	1.55	0.06	0.06	0.06
B	0.20	0.25	0.30	0.01	0.01	0.01
C	0.2TYP			0.008TYP		
e	0.635TYP			0.025TYP		
D	9.804	9.881	9.957	0.386	0.389	0.392
E	5.842	6.020	6.198	0.230	0.237	0.244
E1	3.86	3.929	3.998	0.152	0.155	0.157
L	0.406	0.648	0.889	0.016	0.026	0.035
θ	0°	4°	8°	0°	4°	8°
JEDEC	M0-137(AF)					

⚠*NOTES：DIMENSION "D" DOES NOT INCLUDE MOLD PROTRUSIONS OR GATE BURRS.
MOLD PROTRUSIONS AND GATE BURRS SHALL NOT EXCEED 0.006 INCH PER SIDE.

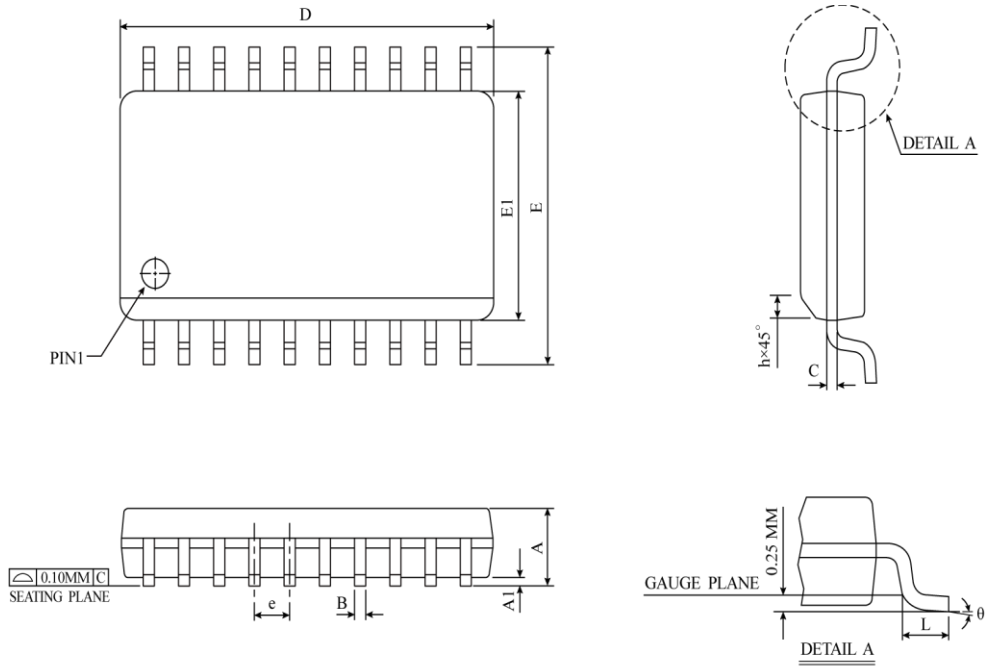
✧ TSSOP-20 (173mil) Package Dimension



SYMBOL	DIMENSION IN MM			DIMENSION IN INCH		
	MIN	NOM	MAX	MIN	NOM	MAX
A	-	-	1.2	-	-	0.047
A1	0.05	0.10	0.15	0.002	0.004	0.006
A2	0.8	0.93	1.05	0.031	0.036	0.041
B	0.19	-	0.3	0.007	-	0.012
D	6.4	6.5	6.6	0.252	0.256	0.260
E	6.25	6.4	6.55	0.246	0.252	0.258
E1	4.3	4.4	4.5	0.169	0.173	0.177
e	0.65 BSC			0.026 BSC		
L	0.45	0.60	0.75	0.018	0.024	0.030
θ	0 °		8 °	0 °		8 °
JEDEC	MO-153 AC REV.F					

Notes：
1.DIMENSION "D" DOES NOT INCLUDE MOLD FLASH, PROTRUSIONS OR GATE BURRS. MOLD FLASH, PROTRUSIONS OR GATE BURRS SHALL NOT EXCEED 0.15 PER SIDE.
2.DIMENSION "E1" DOES NOT INCLUDE INTERLEAD FLASH OR PROTRUSION. INTERLEAD FLASH OR PROTRUSION SHALL NOT EXCEED 0.25 PER SIDE.
3.DIMENSION "B" DOES NOT INCLUDE DAMBAR PROTRUSION.ALLOWABLE DAMBAR PROTRUSION SHALL BE 0.08MM TOTAL IN EXCESS OF THE "B" DIMENSION AT MAXIMUM METERIAL CONDITION. DAMBAR CANNOT BE LOCATED ON THE LOWER RADIUS OF THE FOOT. MINIMUM SPACE BETWEEN PROTRUSION AND ADJACENT LEAD IS 0.07MM.

- SOP-20 (300mil) Package Dimension



SYMBOL	DIMENSION IN MM			DIMENSION IN INCH		
	MIN	NOM	MAX	MIN	NOM	MAX
A	2.35	2.50	2.65	0.0926	0.0985	0.1043
A1	0.10	0.20	0.30	0.0040	0.0079	0.0118
B	0.33	0.42	0.51	0.0130	0.0165	0.0200
C	0.23	0.28	0.32	0.0091	0.0108	0.0125
D	12.60	12.80	13.00	0.4961	0.5040	0.5118
E	10.00	10.33	10.65	0.3940	0.4425	0.4910
E1	7.40	7.50	7.60	0.2914	0.2953	0.2992
e	1.27 BSC			0.050 BSC		
h	0.25	0.50	0.75	0.0100	0.0195	0.0290
L	0.40	0.84	1.27	0.0160	0.0330	0.0500
θ	0°	4°	8°	0°	4°	8°
JEDEC	MS-013 (AC)					

* NOTES : DIMENSION " D " DOES NOT INCLUDE MOLD FLASH, PROTRUSIONS OR GATE BURRS. MOLD FLASH, PROTRUSIONS AND GATE BURRS SHALL NOT EXCEED 0.15 MM (0.006 INCH) PER SIDE.

-