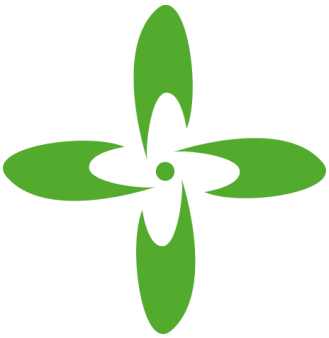


十速

TM56F8152S

DATA SHEET

Rev 0.93

PRECAUTION.

1. **The user must switch RDCTL to “4ns” to enhance the performance of minimal operating voltage**
2. **A sleep instruction is necessary before the event of pin-change otherwise pin-change event may be missed.**
3. **It is suggested to enable the hysteresis function of the comparator.**

AMENDMENT HISTORY

Version	Date	Description
0.9	Dec, 2025	New Release
0.91	Jan, 2026	1. Modify content description 2. Modify ADC Electrical Characteristics
0.92	May, 2026	1. Modify ADC Electrical Characteristics
0.93	May, 2026	1. Modify the use of PWM-triggered ADC

CONTENTS

FEATURES

1. **ROM: 4K x 16 bits Flash**
   - With 8 table readable security keys
2. **RAM: 336 x 8 bits**
3. **STACK: 8 Levels**
4. **System Clock type selections:**
   - Fast clock from Internal RC (FIRC, 16 MHz)
   - Slow clock from Internal RC (SIRC, 95.8 KHz@25℃/V_{CC}=5V)
     - Calibrate through CFC(Capture Frequency Count)
5. **System Clock Prescaler:**
   - System Clock can be divided by 1/2/4/8 option
6. **Power Saving Operation Mode**
   - FAST Mode: Slow-clock is 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, or Wake-up Timer keep running
   - STOP Mode: All clocks stop, Wake-up Timer stop
7. **Independent Timers**
   - Timer0
     - 8-bit timer divided by 1~256 pre-scale option / auto-reload / counter / interrupt / stop function
   - Timer1
     - 8-bit timer divided by 1~256 pre-scale option / auto-reload / interrupt / stop function
     - Overflow and Toggle out
8. **Interrupt**
   - Timer0 / Timer1 / Wake-up Timer Interrupt
   - ADC Interrupt
   - Comparator Interrupt
   - PWM0/PWM1 Interrupt
   - LVD Interrupt
   - All Port Pin-change Wake-up Interrupt
   - CFC(Capture Frequency Count) Interrupt
9. **Wake-up Timer (WKT)**
   - Clocked by built-in RC oscillator with 4 adjustable interrupt times
     - 86 ms / 173 ms / 350 ms / 700 ms @V_{CC}=25℃/5V
     - 100ms / 201ms / 407 ms / 813 ms @V_{CC}=25℃ /3V

10. Watchdog Timer (WDT)

- Clocked by built-in RC oscillator with 4 adjustable reset times
  - 175 ms / 348 ms / 1381 ms / 2761 ms @V_{CC}=25℃/5V
- Watchdog timer can be disabled / enabled in STOP mode

11. One 16-bits PWM and five 8-bits PWMs

- PWM0 is 16-bits PWM with individually adjustable duties and period
- PWM0 clock source: System clock (Fsys), FIRC/256, FIRC (16 MHz), FIRC*2 (32 MHz)
- PWM0 supports complementary output (PWM0P, PWM0N)
- PWM1~5 are 8-bits PWMs with individually adjustable duty and shared adjustable period
- PWM1~5 clock source: System clock (Fsys), FIRC/256, FIRC (16 MHz), FIRC*2 (32 MHz)
  - with 1/2/4/8/16/32/64/128 prescaler
- PWM0N/0P/1/2/3/4/5 has two outputs

12. 12-bit ADC with 22 channels for External Pin Input and 2 channels for Internal Voltage

- Two internal voltage channels: VBG, 1/4VCC
- ADC reference voltage: V_{CC}, V_{BG} (1.183V), V_{BG} (2.53V) , V_{BG} (3V) and V_{BG} (2V)
- PWM0/1 can trigger ADC

13. Comparator

- Comparator x 1
  - With 5-bit special DAC input
  - DAC reference voltage: V_{CC} or 1.183V

14. Reset Sources

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

15. Low Voltage Reset (LVR) and Low Voltage Detection (LVD)

- 15-Level Low Voltage Reset: 1.73V ~ 3.45V, can be disabled
- 15-Level Low Voltage Detection: 1.85V ~ 4.2V, can be disabled

16. Operating Voltage

- Fsys= 16 MHz, PWM0CKS=Fsys, LVR~5.5V. Suggest LVR ≥ 2.30V
- Fsys= 8 MHz, PWM0CKS=Fsys, LVR~5.5V. Suggest LVR ≥ 1.73V

Note: Refer to the "Electrical Characteristics Graphs".

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

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

19. Instruction set: 43 Instructions

20. I/O ports:

- Maximum 22 programmable I/O pins
  - Open-Drain Output
  - CMOS Push-Pull Output
  - Schmitt Trigger Input with pull-up / pull-down resistor option
  - All I/O with High-Sink
  - 1/2 V_{CC} (LCD 1/2 bias) Output
- All pin-change wake-up (falling edge and rising edge trigger) and interrupt

21. LCD Driver

- Maximum 22 software controlled COM
- LCD 1/2 bias

22. Programming connectivity support 4-wire(on-board programming with power), 5-wire (ICP)

23. RDCTL: Read signal delay control for Program ROM

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

24. Trimmed V_{BG}

- Auto-selected V_{BG} trim value by mean of ADC reference voltage selection (ADVREFS) for exact V_{BG}.

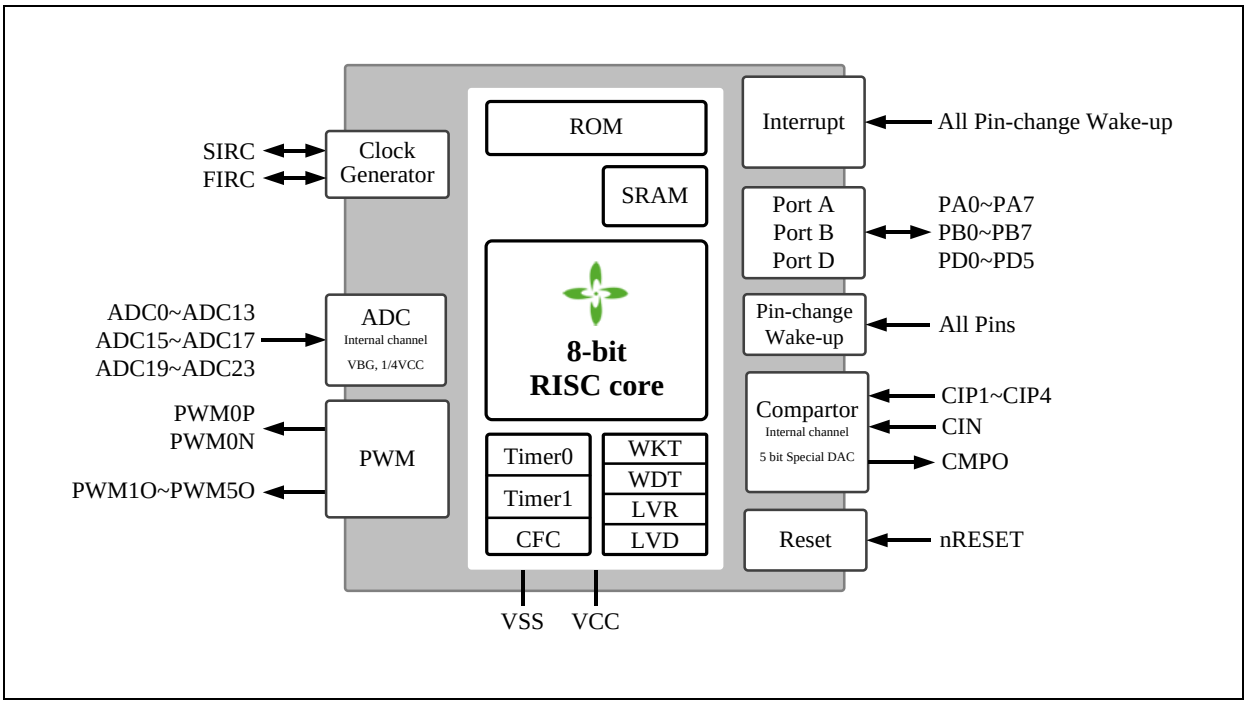
25. ATD: Automatic transient detection(Read signal length control for Program ROM) to enhance the performance of power consumption at slow mode

26. Package Types: Please refer to the chapter of "PACKAGING INFORMATION" (Left click the link to go to that page)

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

- Use PA0/PA1 pin or PD4/PD5 pin

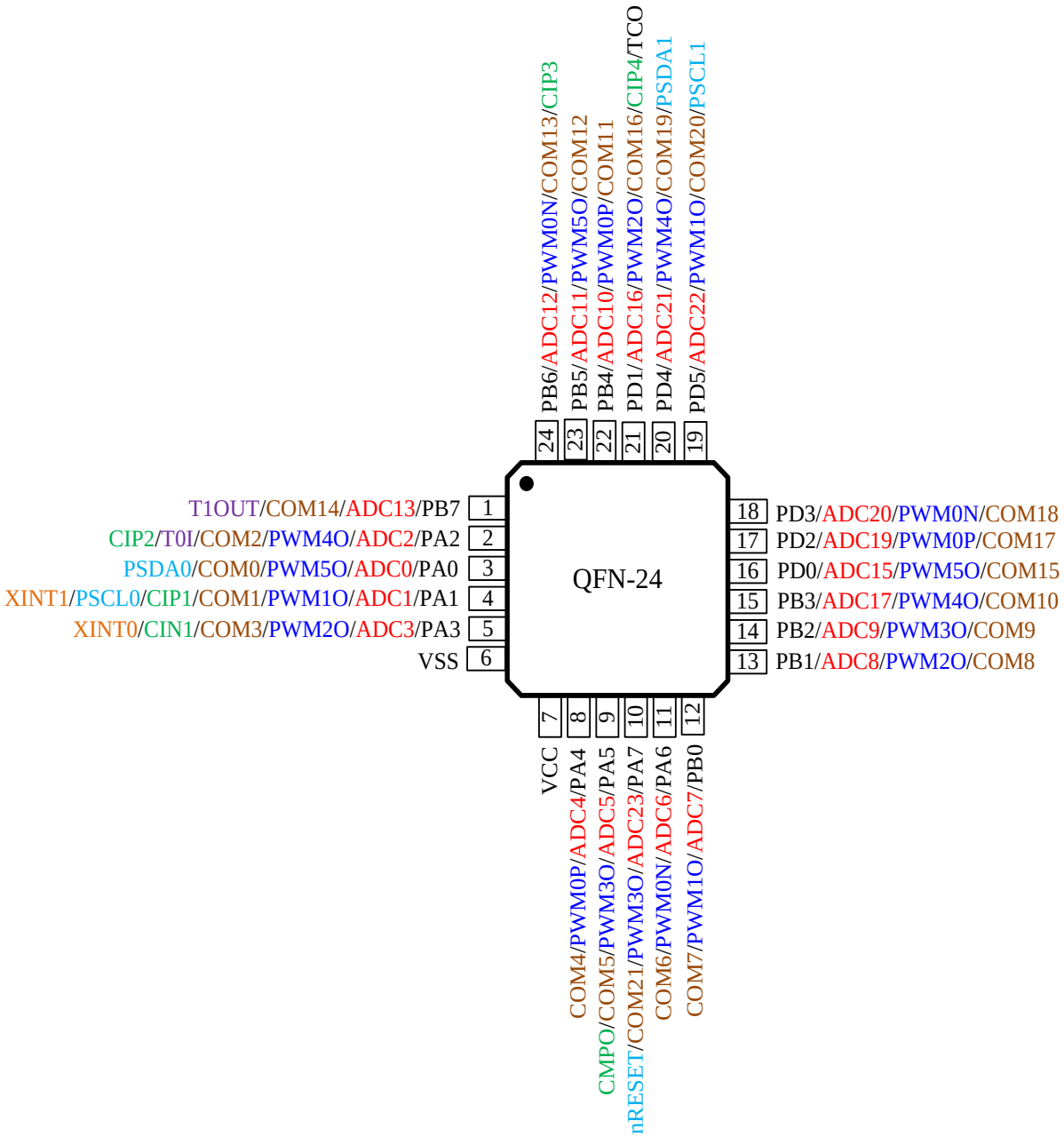
SYSTEM BLOCK DIAGRAM



Block Diagram

PIN ASSIGNMENT

For pads without package, software initialization is required.。



PIN DESCRIPTIONS

Name	In/Out	Pin Description
PA0~PA7 PB0~PB7 PD0~PD5	I/O	Bit-programmable I/O port for Schmitt-trigger input, CMOS push-pull output, open-drain output or 1/2V_{CC}(LCD 1/2 bias) output. Pull-up/Pull-down resistors are assignable by software. **PA7 has no high-sink, 1/2 bias and resistor pull-down capability.**
nRESET	I	External active low reset
VCC, VSS	P	Power Voltage input pin and ground
TM0CKI	I	Timer0's input in counter mode
PWM0P	O	16 bits PWM0 positive output
PWM0N	O	16 bits PWM0 negative output
PWM1O~PWM5O	O	8 bits PWM1~PWM5 output
CMPO	O	Comparator status output
TCOUT	O	Fsys/2 clock output
TM1OUT	O	Timer1 overflow toggle output
ADC0~ADC13 ADC15~ADC17 ADC19~ADC23	I	ADC channel input
CIN	I	Comparator negative port input
CIP1~CIP4	I	Comparator positive port input
COM0~COM16	O	LCD 1/2 bias output
PSCL	I	Clock for programmer
PSDA	I/O	Data for programmer

Programming pins:

Normal mode (5-wire): VCC / VSS / PA0(PSDA) / PA1(PSCL) / PA5

ICP mode (4-wire): VCC / VSS / PA0(PSDA) / PA1(PSCL) - When using ICP (In-Circuit Program) mode, the PCB needs to remove all components of PA0, PA1.

On-board programming with power mode (4-wire): VCC / VSS / PA0(PSDA) / PA1(PSCL) - When using on-board programming with power mode, the PCB needs to remove all components of PA0, PA1.

PIN SUMMARY

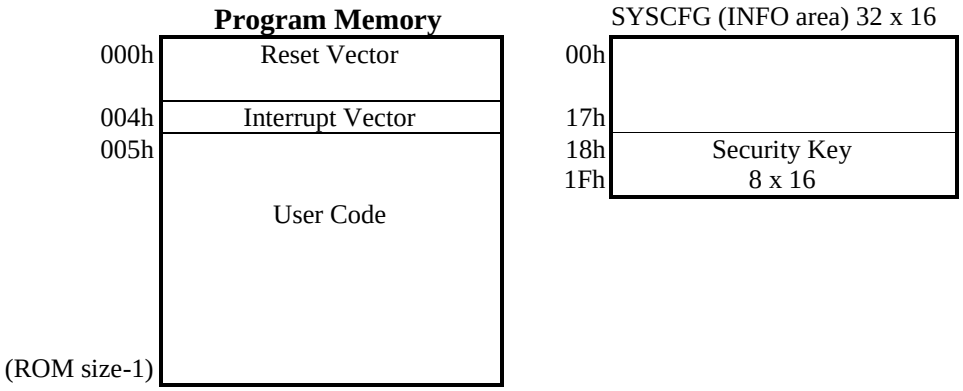
Pin Name	Type	GPIO						Alternate Function			
		Input			Output						
		Pull-up Control	Pull-down Control	Wake-up Interrupt	Open Drain	CMOS Push-Pull	1/2 V_{CC} (LCD 1/2 Bias)	PWM	ADC	Comparator	MISC
VCC	P										
PA4/ADC4/PWM0P/COM4	I/O	●	●	●	●	●	●	●	●		
PA5/ADC5/PWM3O/COM5/CMPO	I/O	●	●	●	●	●	●	●	●	●	
PA7/ ADC23/ PWM3O/COM21/ nRESET	I/O	●		●	●	●	●	●	●		nRESET
PA6/ADC6/PWM0N/COM6	I/O	●	●	●	●	●	●	●	●		
PB0/ADC7/PWM1O/COM7	I/O	●	●	●	●	●	●	●	●		
PB1/ADC8/PWM2O/COM8	I/O	●	●	●	●	●	●	●	●		
PB2/ADC9/PWM3O/COM9	I/O	●	●	●	●	●	●	●	●		
PB3/ADC17/PWM4O/COM10	I/O	●	●	●	●	●	●	●	●		
PD0/ADC15/COM15	I/O	●	●	●	●	●	●	●	●		
PD1/ADC16/COM16/TCOUT/CIP4	I/O	●	●	●	●	●	●	●	●	●	TCOUT
PD2/ADC19/PWM0P/ COM17	I/O	●	●	●	●	●	●	●	●		
PD3/ADC20/PWM0N /COM18	I/O	●	●	●	●	●	●	●	●		
PD4/ADC21/PWM4O /COM19/PSDA1	I/O	●	●	●	●	●	●	●	●		Programming
PD5/ADC22/PWM1O /COM20/PSCL1	I/O	●	●	●	●	●	●	●	●		Programming
PB4/ADC10/PWM0P/COM11	I/O	●	●	●	●	●	●	●	●		
PB5/ADC11/PWM5O/COM12	I/O	●	●	●	●	●	●	●	●		
PB6/ADC12/PWM0N/COM13/CIP3	I/O	●	●	●	●	●	●	●	●	●	
PB7/ADC13/COM14/TM1OUT	I/O	●	●	●	●	●	●	●			TM1OUT
PA2/ADC2/PWM4O/COM2/TM0CKI/ CIP2	I/O	●	●	●	●	●	●	●	●	●	TM0CKI
PA0/ADC0/PWM5O/COM0/PSDA0	I/O	●	●	●	●	●	●	●	●		Programming
PA1/ADC1/PWM1O/COM1/CIP1/PSCL0	I/O	●	●	●	●	●	●	●	●	●	Programming
PA3/ADC3/PWM2O/COM3/CIN	I/O	●	●	●	●	●	●	●	●	●	
VSS	P										

FUNCTION DESCRIPTION

1 CPU Core

1.1 Program ROM (PROM)

The ROM of this device has an extra 32-Word INFO area to store the SYSCFG. There are 8 words of security keys which could be table read in the INFO area. The ROM can be written multi-times and can be read as long as the PROTECT (CFGWH.15) bit of SYSCFG is not set. The SYSCFG can be read no matter PROTECT is set or cleared, but PROTECT bit can be cleared only when User ROM Code area is erased. On the other hand, if PROTECT bit is set, the user ROM code area will not be read by writer, and the user ROM code can't be updated until the PROTECT bit is cleared. The endurance of ROM is 1000 times @Vcc=5V/25℃。



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 INFO area; it contains a 16 bits register (CFGWH). The SYSCFG determines the option for initial condition of CPU. It is written by PROM Write only. User can select LVR operation mode and chip operation mode by SYSCFG register. The 15th bit of CFGWH is code-protected selection bit. If this bit is 1, the data in PROM will be protected when user reads PROM. There also are eight 16-bit security keys which could be written by PROM writer and is table readable.

Bit		**15~0**	
Default Value		**0000_0110_0000_0000**	
Bit		Description	
CFGWH	15	**PROTECT**: Code protection selection	
		0	Disable
		1	Enable
	13-12	**WDTE**: WDT Reset Enable	
		0X	Disable
		10	Enable in FAST/SLOW mode, Disable in IDLE/STOP mode
		11	Always Enable
	11-8	**LVRS**: Low Voltage Reset Voltage Selection Please refer to the table of LVR voltage in the section of "LVR Circuit Characteristics". (Left click the link to go to that page)	
	7	**XRSTE**: External Pin (PA7) Reset Enable	
		0	Disable (PA7 as I/O pin)
		1	Enable
	5	**FIRCPSC**: FIRC Prescaler	
		0	Divided by 1 (16 MHz)
		1	Divided by 2 (8 MHz)
	4	**PORSEL**: POR duty cycle selection	
		0	POR enables at 100% duty cycle(POR is always on)
		1	POR enables at 1/16 duty cycle(This feature can't be emulated)(POR is only on at part of the time)
	3	**ATDOFF**: Automatic transient detection(Read signal length control for Program ROM)	
		0	ATD on(for power saving in SLOW mode)
		1	ATD off(recommend for EFT issue)
	2	**RDCTL**: Read signal delay control for Program ROM **The user must switch this bit to "4ns" to enhance the performance of minimal operating voltage.**	
		0	8ns delay for read signal of Program ROM(default: 8ns)
		1	4ns delay for read signal of Program ROM
	1-0	tenx Reserved	

The usage of the Security Key Read instruction for eight 16-bit security keys is the same as using the Table Read instruction, except that the INFOEN enable bit must be set high. The Security Key Area enable bit can be set by writing 0xF0 to the INFOEN register (191h). Writing any other value to INFOEN (191h) clears the Security Key Area enable bit. To access the Security Key Area, DPTR[11:8] must be set to 0. DPTR[7:0] sets eight 16-bit data addresses from 0xD0 to 0xD7 to read security key data.

◇Example: Read Security Key using assembly method

```
MOVLW    F0h
MOVWX    INFOEN          ; set Security Key Area enable bit
CLRX     DPH             ; set DPH = 00h
MOVLW    D0h             ;
MOVWX    DPL             ; set DPL = D0h, DPTR = 00D0h
TABRL                    ; W = TABR = data low byte of Security Key [0h]
                         ; read Security Key data into W using TABRL
TABRH                    ; W = TABR = data high byte of Security Key [0h]
                         ; read Security Key data into W usingTABRH
MOVLW    00h
MOVWX    INFOEN          ; disable Security Key Area access
```

◇Example: Reading Security Key in C

```
INFOEN=0xF0;             // set Security Key enable bit
DPH=0;                   // set DPH = 00h
DPL=0xD0                 // set DPL = D0h, DPTR = 00D0h
TABR=1;                  // write 01h to TABR  = opcode TABRL
rData=TABR;              // rData = data low byte of  Security Key [0h]
TABR=2;                  // write 02h to TABR  = opcode TABRH
rData=TABR;              // rData = data high byte of Security Key [0h]
INFOEN=0x00;             // disable Security Key access
```

Note: When using a C language lookup table, the lookup table instruction can only be used outside or inside an interrupt. If the lookup table instruction is used inside or outside an interrupt, an error may occur.

191h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INFOEN	INFOEN							
R/W	W							
Reset	0	0	0	0	0	0	0	0

191h.7~0 **INFOEN**: Security Key Area Access Enable
Writing 0xF0 to this register will enable access to the Security Key Area.
Writing any other value to this register will disable access to the Security Key Area.

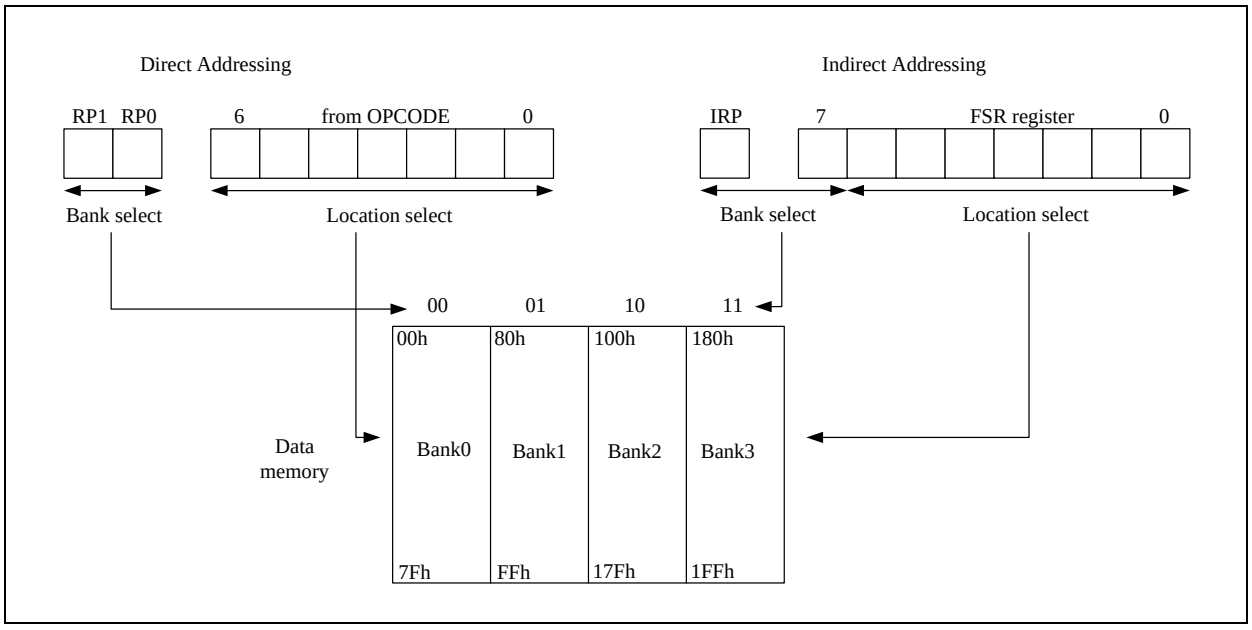
1.3 RAM Addressing Mode

There is one Data Memory Plane in CPU. The Plane is partitioned into four banks. Each bank extends up to 7Fh (128 bytes). The lower locations of each bank are reserved for Special Function Register (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 from one bank may be mirrored in another bank for code reduction and quicker access.

Bit RP1 and RP0 (STATUS[6:5]) are the bank select bits.

[RP1, RP0]	BANK
00	0
01	1
10	2
11	3

The plane can be addressed directly or indirectly. The INDF register is not a physical register. Addressing the INDF register will cause indirect addressing. Indirect addressing is possible by using the INDF register. Any instruction using the INDF register actually accesses the register pointed to by File Select Register, FSR. Reading the INDF register itself, indirectly (FSR = '0') will read 00h. Writing to the INDF register indirectly (FSR = '0') results in a no operation (although status bit may be affected). An effective 9-bit address is obtained by concatenating the 8-bit FSR register and the IRP bit (STATUS[7]). Refer to the figure below.



Direct / Indirect Addressing

Keeping RP0=RP1=0 in the beginning of the F/W code and using the new instruction set.
The advantage of using new instruction is user can ignore the bank location of registers and the code size can be saved. The new instruction is almost the same as the old instruction. By replacing the "F" to "X" in the instruction set can easily use the new instruction without switching the bank.

For 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**	PAMOD10	➔	MOVW**X**	PAMOD10
RL**F**	RAMA0, 0	➔	RL**X**	RAMA0, 0
SWAP**F**	ADCTL, 0	➔	SWAP**X**	ADCTL, 0

【BANK0】 000~07Fh		【BANK1】 080h~0FFh		【BANK2】 100h~17Fh		【BANK3】 180h~1FFh	
000h	**INDF**	080h	**INDF**	100h	**INDF**	180h	**INDF**
001h	TM0	081h	OPTION	101h	TM0	181h	OPTION
002h	**PCL**	082h	**PCL**	102h	**PCL**	182h	**PCL**
003h	**STATUS**	083h	**STATUS**	103h	**STATUS**	183h	**STATUS**
004h	**FSR**	084h	**FSR**	104h	**FSR**	184h	**FSR**
005h	PAD	085h	PAMOD10	105h		185h	DPL
006h	PBD	086h	PAMOD32	106h		186h	DPH
007h	PDD	087h	PAMOD54	107h	PWMCTL2	187h	
008h		088h	PAMOD76	108h		188h	
009h		089h	PWMCTL	109h	LVRPD	189h	
00Ah	**PCLATH**	08Ah	**PCLATH**	10Ah	**PCLATH**	18Ah	**PCLATH**
00Bh	**INTIE**	08Bh	**INTIE**	10Bh	**INTIE**	18Bh	**INTIE**
00Ch	INTIF	08Ch	PBMOD10	10Ch	PCH	18Ch	TABR
00Dh	INTIE1	08Dh	PBMOD32	10Dh		18Dh	CMPCTL
00Eh	INTIF1	08Eh	PBMOD54	10Eh	BGTRIM	18Eh	CMPPNS
00Fh	CLKCTL	08Fh	PBMOD76	10Fh	IRCF	18Fh	DACTL
010h	TM0RLD	090h	PDMOD10	110h		190h	INFOEN
011h	TM0CTL	091h	PWM1PRD	111h		191h	
012h	TM1	092h	PWM0PRDH	112h		192h	
013h	TM1RLD	093h	PWM0PRDL	113h		193h	
014h	TM1CTL	094h	PWM0DH	114h		194h	
015h		095h	PWM0DL	115h		195h	
016h	LVCTL	096h	PWM1D	116h	CFCTL	196h	
017h	ADCDH	097h	PWM2D	117h	CFCNT	197h	
018h	ADCTL	098h	PWM3D	118h		198h	
019h	ADCTL2	099h	PWM4D	119h		199h	
01Ah		09Ah	PWM5D	11Ah		19Ah	
01Bh		09Bh	PDMOD32	11Bh		19Bh	
01Ch		09Ch	PDMOD54	11Ch		19Ch	
01Dh		09Dh	PWMCMP_CTL	11Dh		19Dh	
01Eh		09Eh	PWM_CMPH	11Eh		19Eh	
01Fh		09Fh	PWM_CMPL	11Fh		19Fh	
020h	RAM Bank0 area (80 Bytes)	0A0h	RAM Bank1 area (80 Bytes)	120h	RAM Bank2 area (80 Bytes)	1A0h	RAM Bank3 area (80 Bytes)
06Fh		0EFh		16Fh		1EFh	
070h	**common area (16 Bytes)**	0F0h	**accesses 070h~07Fh**	170h	**accesses 070h~07Fh**	1F0h	**accesses 070h~07Fh**
07Fh		0FFh		17Fh		1FFh	

◇ Example: read / write register by using direct addressing **(force RP0=RP1=0)**

```
CLKCTL      equ     00Fh        ; SFR in Bank0
TM1         equ     012h        ; SFR in Bank0
PWM1PRD     equ     091h        ; SFR in Bank1
LVRPD       equ     109h        ; SFR in Bank2
IRCF        equ     10Fh        ; SFR in Bank2
DPL         equ     185h        ; SFR in Bank3
RAM020      equ     020h        ; RAM in Bank0
RAM0A0      equ     0A0h        ; RAM in Bank1

MOVXW       TM1                 ; read TM1 (Bank0) to W
MOVXW       PWMCTL              ; read PWMCTL (Bank1) to W
MOVXW       IRCF                ; read IRCF (Bank2) to W
MOVXW       DPL                 ; read DPL (Bank3) to W

MOVLW       16h                 ; W = 16h
MOVWX       RAM020              ; RAM[0x020] = W = 16h
MOVWX       RAM0A0              ; RAM[0x0A0] = W = 16h

MOVLW       37h                 ; W = 37h
MOVWX       LVRPD               ; LVRPD = W = 37h, force LVR/POR disable

MOVXW       CLKCTL              ; read SFR CLKCTL (00Fh) to W
MOVXW       IRCF                ; read SFR IRCF (10Fh) to W

MOVLW       0Bh                 ; W = 0Bh
MOVWX       CLKCTL              ; CLKCTL (00Fh) = W = 0Bh
MOVWX       IRCF                ; IRCF (10Fh) = W = 0Bh
```

◇ Example: read / write register by using indirect addressing **(force RP0=RP1=0)**

```
BSX         IRP                 ; IRP = 1 => Bank2/3
MOVLW       0Fh                 ; W = 0Fh
MOVWX       FSR                 ; FSR = W = 0Fh
MOVXW       INDF                ; read SFR IRCF (10Fh) to W

BSX         IRP                 ; IRP = 1 => Bank2/3
MOVLW       0Fh                 ; W = 0Fh
MOVWX       FSR                 ; FSR = W = 0Fh
MOVLW       0Bh                 ; W = 0Bh
MOVWX       INDF                ; IRCF (10Fh) = W = 0Bh

BCX         IRP                 ; IRP = 0 => Bank0/1
MOVLW       0Fh                 ; W = 0Fh
MOVWX       FSR                 ; FSR = W = 0Fh
MOVXW       INDF                ; read SFR CLKCTL (00Fh) to W

BCX         IRP                 ; IRP = 0 => Bank0/1
MOVLW       0Fh                 ; W = 0Fh
MOVWX       FSR                 ; FSR = W = 0Fh
MOVLW       0Bh                 ; W = 0Bh
MOVWX       INDF                ; CLKCTL (00Fh) = W = 0Bh
```

1.4 Programming Counter (PC) and Stack

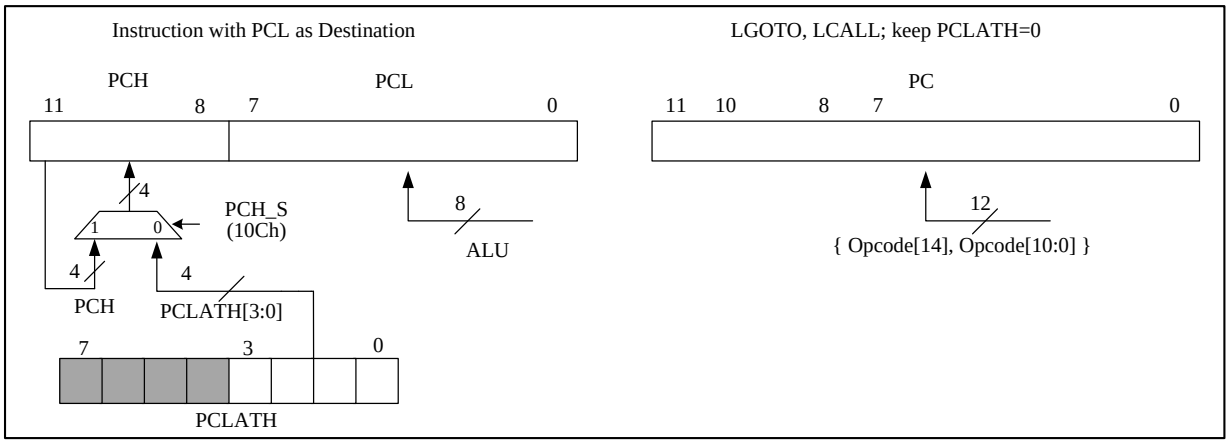
The Programming Counter is 12-bit wide. As a program instruction is executed, the PC will contain the address of the next program instruction to be executed. The PC value is normally increased by one except for the following cases. The Reset Vector (000h) and the Interrupt Vector (004h) are provided for PC initialization and Interrupt. For CALL/GOTO instruction, PC loads lower 11 bits address from instruction word and upper 1 bit from PCLATH[3]. For RET/RETI/RETLW instruction, PC retrieves its content from the top level STACK.

Before CALL/GOTO instruction is executed, the PCLATH[3] must be set if the destination address more than 2K, otherwise the PCLATH[3] must be cleared. Similar as RAM Addressing Mode (refer section 1.3), the Chip provides new instruction set LCALL/LGOTO to replace CALL/GOTO instruction set. When using LCALL/LGOTO, user don't care about the destination address, just only keep PCLATH[3] cleared.

The low byte data of the Programming Counter (PC[7:0]) can be read and written by PCL register (002h/082h/102h/182h). The high byte data of Programming Counter (PC[11:8]) can only be read by PCH register (10Ch). The internal flag PCH_S is used to select the source of PCH, when executing any instruction with the PCL register as the destination. Write 0x1C to PCH register can set PCH_S, write others value to PCH register will clear PCH_S. After reset, the PCH_S is cleared.

When PCH_S is cleared to '0', executing any instruction with the PCL register as the destination simultaneously causes PCH to be replaced by the contents of the PCLATH (00Ah/08Ah/10Ah/18Ah) register. This allows the entire contents of the program counter to be changed by writing the desired high byte to the PCLATH register. When the low byte is written to the PCL register, all contents of program counter will change to the values contained in the PCLATH register and those being written to the PCL register.

When PCH_S is set to '1', executing any instruction with the PCL register as the destination the low byte is written to the PCL register and will not change the PCH. It is recommended to setting PCH_S to '1' when using any instruction with the PCL register as the destination, but C language doesn't support this function.



002h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PCL	PCL							
R/W	R/W							
Reset	0	0	0	0	0	0	0	0

002h.7~0 **PCL:** Programming Counter data bit 7~0

00Ah	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

PCLATH	GPR				PCLATH			
R/W	R/W				R/W			
Reset	0	0	0	0	0	0	0	0

00Ah.3~0 **PCLATH:** Programming Counter high byte data when instruction with PCL as destination is executed, and PCH_S is cleared

00Ah.3 **PCLATH:** Programming Counter upper 1 bit when CALL/GOTO instruction is executed
Note: When using LCALL/LGOTO instruction must keep cleared

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

10Ch.7~0 **PCH (W):** Programming Counter high byte source 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

10Ch.3~0 **PCH (R):** Programming Counter data bit 11~8

The STACK is 12-bit wide and 8-level in depth. The LCALL instruction and hardware interrupt will push STACK level in order, while the RET/RETI/RETLW instruction pops STACK level in order. 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 the16-bit ROM data into W register by setting TABR (18Ch) for C language.

◇ Example: To look up the PROM data located RETLW.

```
ORG       000h                    ;  Reset Vector
          LGOTO     START

START:
          MOVLW     00h
          MOVWX     RAM020          ; Set lookup table's address
          MOVLW     1Ch             ; Write 1Ch to PCH to set PCH_S flag
          MOVWX     PCH
LOOP:
          MOVXW     RAM020          ; Move index value to W register
          LCALL     TABLE1          ; To lookup data
          …
          INCX      RAM020, 1       ; Increment the index address for next address
          …
          LGOTO     LOOP            ; Go to LOOP label
          …
ORG       X00h
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     58h             ; W=58h when return
```

Note: The chip define 256 ROM address as one page, so that ROM has 16 pages, 000h~0FFh, 100h~1FFh, …, F00h~FFFh. On the other words, PC[11:8] can be define as page. A lookup table must be located at the same page to avoid getting wrong data. Thus, the lookup table has maximum 255 data

for above example with starting a lookup table at X00h (X = 1, 2, 3, …, E, F). If a lookup table has fewer data, it needs not setting the starting address at X00h, but only confirms all lookup table data are located at the same page.

◇ Example: To look up the PROM data located using TABRL / TABRH in assembly method.

```
            MOVLW     (TABLE2 >>8) & 0xff
            MOVWX     DPH
            MOVLW     (TABLE2) & 0xff
            MOVWX     DPL                 ; DPTR = {DPH, DPL} = TABLE2
; Table Read by instructions TABRL / TABRH
            TABRL                         ; Read PROM low byte data to W (W = 86h)
            TABRH                         ; Read PROM high byte data to W (W = 19h)
            …

TABLE2:
            .DT       0x1986              ; 16-bit ROM data
            …
```

◇ Example: To look up the PROM data located using TABR register in C language method.

```
; Table Read by SFR TABR
            const unsigned int TABLE[]={0x1234};
            DPL = &TABLE;
            DPH = &TABLE >> 8;
            TABR=1;                       // write 01h to TABR  = opcode TABRL
                                          // TABR= low byte data of TABLE[0] =34h
            rData1=TABR;                  // rData1 = TABR= low byte data of TABLE[0] =34h
            TABR=2;                       // write 02h to TABR  = opcode TABRH
                                          // TABR= high byte data of TABLE[0] =12h
            rData2=TABR;                  // rData2 = TABR= high byte data of TABLE[0] =12h
```

Note: When using TABR register in C language to look up the table, the lookup table instruction can only be used outside the interrupt or within the interrupt, if the lookup table instruction is used inside and outside the interrupt, it may cause an error.

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	0	0	0	0	0	0	0

18Ch.7~0 The TABR register is used for lookup tables when using the C language as following step.
1. TABR write 01h to get PROM low byte data to W and TABR register (as instruction TABRL)
2. TABR write 02h to get PROM high byte data to W and TABR register (as instruction TABRH)
3. After step.1 or step.2, read TABR to get main ROM table read value
4. After step.1 or step.2, read TABR to get main ROM table read value for C language

Table Read for ASM: Support instruction TABRL / TABRH or register TABR. Suggest not using the method of register TABR. SFR HWAUTO=1 is also suggested.
Table Read for C: using register TABR. Only be used outside or inside the interrupt service routine. Don't utilize it inside and outside interrupt service routine simultaneously. Otherwise, something will be wrong.

1.4.1 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.

1.4.2 STATUS Register (003h/083h/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><td>STATUS</td><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><td>Reset Value</td><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><td>R/W</td><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">Description</td></tr>
<tr><td>7</td><td colspan="8">IRP: Register Bank Select bit (used for indirect addressing)
0 = Bank 0,1 (000h - 0FFh)
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 (000h - 07Fh)
01 = Bank 1 (080h - 0FFh)
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 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 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 instruction</td><td colspan="4">SUB instruction</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 Timer 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.

2.2 Low Voltage Reset (LVR)

The Low Voltage Reset features static reset when supply voltage is below a threshold level. There are 15 threshold levels can be selected. The LVR's operation mode is defined by the CFGWH register. Please refer to the table of LVR voltage in the chapter of Electrical Characteristics. Vcc must be below maximum operating voltage and above LVR voltage. The user must consider the minimal operating voltage of corresponding operating frequency such that LVR voltage must be above minimum operating voltage.

Different F_{sys} have different system minimum operating voltage. Please refer to the table and figure of minimal operating voltage in the chapter of "Electrical Characteristics". If too low LVR voltage is selected such that it is lower than the minimum operating voltage of the corresponding F_{sys}, the system may enter a dead zone and an error occurs.

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

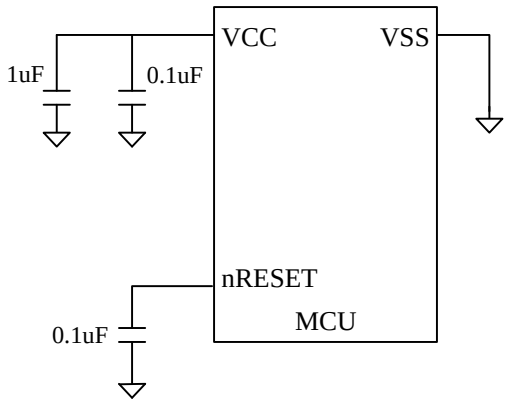
16h.5 **LVRSAV**: POR/LVR auto power off in STOP/IDLE mode
0: disable POR/LVR auto power off in STOP/IDLE mode
1: enable POR/LVR auto power off in STOP/IDLE mode

2.3 External Pin Reset (XRST)

The External Pin Reset (XRST) can be disabled or enabled by XRSTE at CFGWH register. External pin reset should be kept low for at least 2 SIRC clock cycles to ensure reset can active. The External Pin Reset also sets all the control registers to their default value but the TO/PD flags will not affected by these resets.

External reset pin (nRESET) 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.



2.4 Watchdog Timer Reset (WDTR)

The WDT reset can be disabled or enabled through the CFGWH register. Set WDTPSC to define the period during which WDT reset occurs. WDT reset counter can be cleared by device Reset or CLRWDT bit. WDT reset also set all the control registers to their default value. The TO/PD flags are not affected by WDT resets.

◇ Example: Defining Reset Vector

```
        ORG     000h        ; Reset Vector
        LGOTO   START       ; Jump to user program address.

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

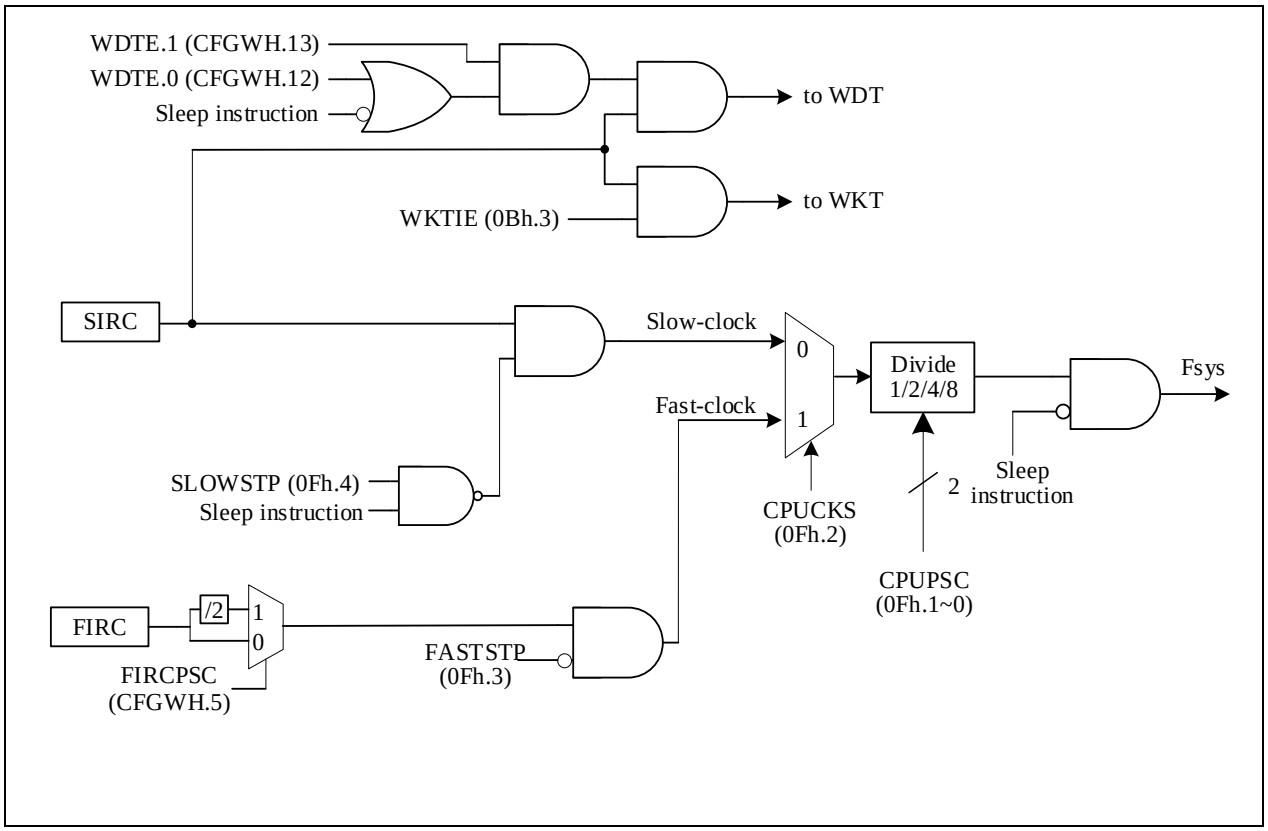
3 Clock Circuitry and Operation Mode

3.1 System Clock

The device is designed with dual-clock system. There are two kinds of clock source: SIRC (Slow Internal RC) Clock and FIRC (Fast Internal RC) Clock. Each clock source can be applied to CPU kernel as system clock. When in IDLE mode, the SIRC provides clock source to WKT/WDT block. Refer to the Figure as below.

After Reset, the device is running at SLOW mode with SIRC. 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, a 16 MHz System clock rate requires V_{CC} > 2V@(25℃).

The CLKCTL (0Fh) SFR controls the System clock operating. H/W automatically blocks the S/W abnormally setting for this register. Never to write both FASTSTP=1 and CPUCKS=1. It is recommended to write this SFR bit by bit.



Clock Scheme Block Diagram

The frequency of FIRC can be adjusted by IRCF (10Fh). When IRCF=00h, frequency is the lowest. When IRCF=7Fh, frequency is the highest. With this function, we can adjust the frequency of FIRC after power on. Each IC may have different default value of IRCF, to make sure the frequency of FIRC=16 MHz after Power on Reset.

FAST Mode:

In this mode, the program is executed using FIRC as CPU clock (Fsys). The Timer0, Timer1 blocks are also driven by Fast-clock. The PWM0 block can be driven by Fsys, FIRC/256, FIRC (16 MHz), or FIRC*2 (32 MHz) by setting PWM0CKS (89h.5~4). The PWM1 block can only be driven by FIRC (16 MHz).

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, Timer1, etc…) clock source are Slow-clock in the SLOW mode except PWM which can select other clock source. There is only one kinds of SLOW clock can be selected, SIRC.

IDLE Mode:

After executing the SLEEP instruction, if SIRC is still oscillating, it means entering IDLE mode. IDLE mode is terminated by Reset or enabled Interrupts wake-up. There are two ways to keep SIRC oscillating in IDLE mode.

(1) Set SLOWSTP=0, before executing the SLEEP instruction, the SIRC can still oscillate

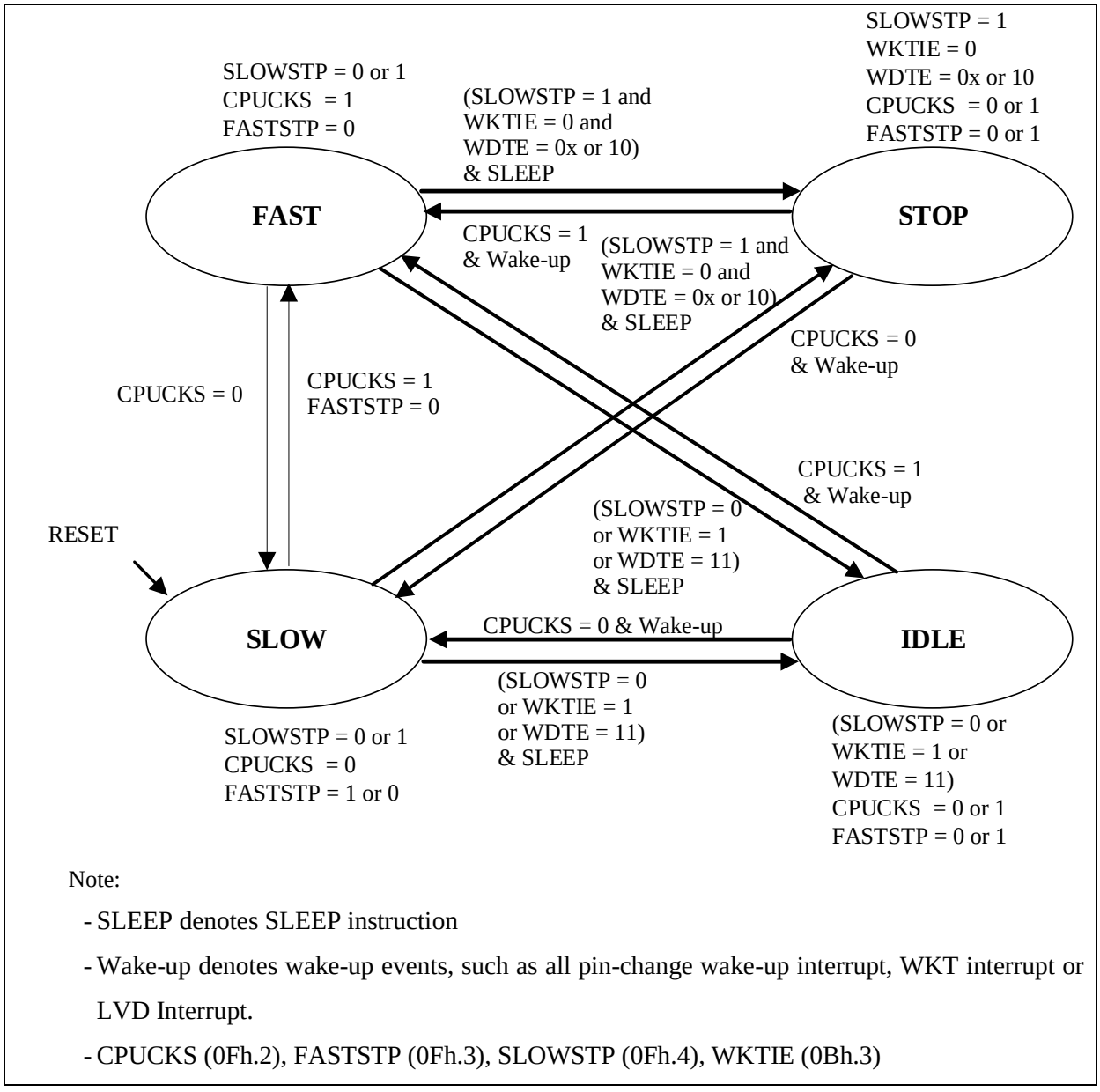
(2) Set WKTIE=1 or WDTE=11, before executing the SLEEP instruction, the SIRC can still oscillate to keep WKT/WDT operating in IDLE mode.

STOP Mode:

When SLOWSTP (0Fh.4) is set, WKTIE (0Bh.3) is cleared and WDTE=0x or 10, all blocks will be turned off and the Chip will enter the “STOP Mode” after executing the SLEEP instruction. STOP mode is similar to IDLE mode. The difference is all clock oscillators either Fast-clock or Slow-clock are stopped and no clocks are 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
- Wake-up denotes wake-up events, such as all pin-change wake-up interrupt, WKT interrupt or LVD Interrupt.
- CPUCKS (0Fh.2), FASTSTP (0Fh.3), SLOWSTP (0Fh.4), WKTIE (0Bh.3)

CPU Operation Block Diagram

CPU Mode & Clock Functions Table:

Mode	Fsys	Fast-clock	Slow-clock	TM0/TM1	WKT	WDT	Wake-up event
FAST	Fast-clock	Run	Run	Run	Run	Run	X
SLOW	Slow-clock	Set by FASTSTP	Run	Run	Run	Run	X
IDLE	Stop	Stop	Run	Stop	Set by WKTIE	Set by WDTE	WKT/IO
STOP	Stop	Stop	Stop	Stop	Stop	Stop	IO

- **FAST mode switches to SLOW mode**

The following steps are suggested to be executed by order when FAST mode switches to SLOW mode:

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

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

◇ Example: Switch FAST mode to SLOW mode.

```
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     ; Enable Fast-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) or WDT (WDTE=11b)

(2) Execute SLEEP instruction

IDLE mode can be waked up by all port pin-change wake-up interrupt, WKT interrupt and LVD interrupt.

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

```
BCX     SLOWSTP     ; Enable Slow-clock after execute SLEEP instruction
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)

(3) Execute SLEEP instruction

STOP mode can be woken up only by pin-change.

Note: CPU will not enter STOP mode if WDTE=11b

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

```
BSX     SLOWSTP     ; Disable Slow-clock after execute SLEEP instruction
MOVLW   00000000b   ; Disable WKT counting
MOVWX   INTIE
SLEEP               ; Enter STOP mode.
```

0Bh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTIE	ADCIE	–	TM1IE	TM0IE	WKTIE	–	–	–
R/W	R/W	–	R/W	R/W	R/W	–	–	–
Reset	0	–	0	0	0	–	–	–

0Bh.3 **WKTIE:** Wake-up Timer interrupt enable and Wake-up Timer enable
0: disable
1: enable

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

0Fh.4 **SLOWSTP**: Stop Slow-clock after execute SLEEP instruction
0: Slow-clock keeps running after execute SLEEP instruction
1: Slow-clock stops running after execute SLEEP instruction

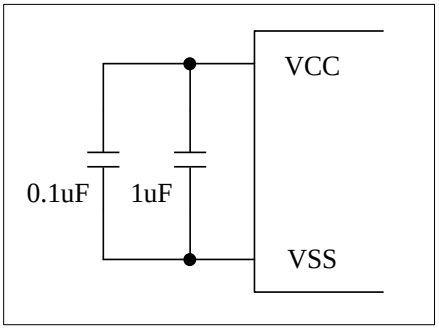
0Fh.3 **FASTSTP**: Fast-clock stop
0: Fast-clock is running
1: Fast-clock stops running

0Fh.2 **CPUCKS**: System clock source select
0: Slow-clock
1: Fast-clock

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

3.3 System Clock Oscillator

In the Fast Internal RC (FIRC) mode, the on-chip oscillator generates 16 MHz system clock. Since power noise degrades the performance of Internal Clock Oscillator, placing power supply bypass capacitors 1 uF and 0.1 uF very close to VCC/VSS pins improves the stability of clock and the overall system.



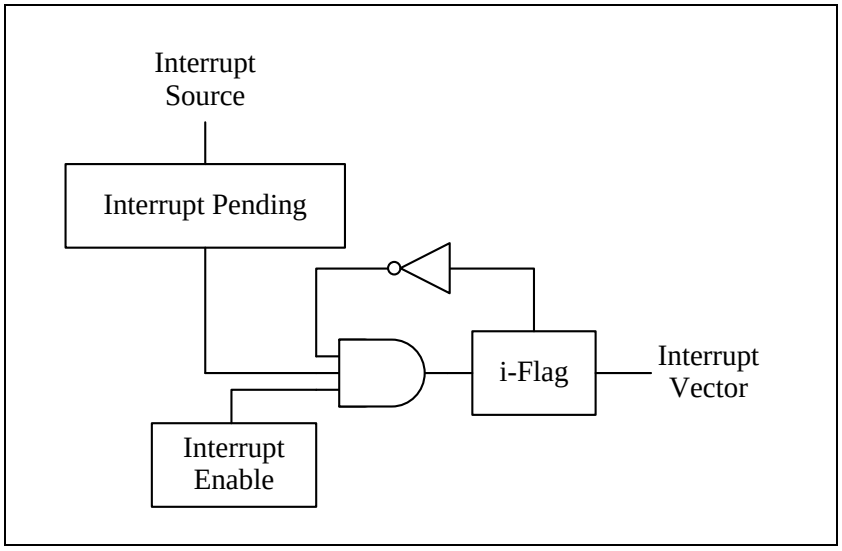
Internal RC Mode

4 Interrupt

The Chip has 1 level, 1 vector and 11 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], INTIE[5:3], INTIE[1:0], INTIE1[7:6], INTIE1[4], INTIE1[2:0]) has been set, it would trigger CPU to service the interrupt. CPU accepts interrupt at the end of current executed instruction cycle. In the meanwhile, a “LCALL 004” instruction is inserted to CPU, and i-flag is set to prevent recursive interrupt nesting.

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. F/W must clear the interrupt event register while serving the interrupt routine.



◇ Example: Setup pin-change wake-up interrupt request

```
        ORG     000h        ; Reset Vector
        LGOTO   START       ; Goto user program address

        ORG     004h        ; All interrupt vector
        LGOTO   INT         ; If PA, PB and PD change during sleep

        ORG     005h
START:
        MOVLW   37h
        MOVWX   LVRPD       ; Disable POR and LVR
        MOVLW   03h
        MOVWX   CLKCTL      ; FIRC runs
        MOVLW   17h
        MOVWX   CLKCTL      ; Fsys=FIRC, SIRC stops after sleep
        MOVLW   10001000b
        MOVWX   PAMOD10     ; Select PA, PB and PD Pin Mode as mode 1000b
        MOVWX   PAMOD32     ; Open drain output low or input with Pull-up
        MOVWX   PAMOD54     ; & pin-change wake-up
        MOVWX   PAMOD76
        MOVWX   PBMOD10
        MOVWX   PBMOD32
        MOVWX   PBMOD54
        MOVWX   PBMOD76
        MOVWX   PDMOD10
        MOVWX   PDMOD32
        MOVWX   PDMOD54
        MOVLW   11111111b
        MOVWX   PAD         ; Release PA, PB and PD, it becomes Schmitt-trigger
        MOVWX   PBD         ; input with input pull-up resistor
        MOVWX   PDD
        MOVLW   10111111b
        MOVWX   INTIF1      ; Clear pin-change wake-up interrupt request flag
        MOVLW   01000000b
        MOVWX   INTIE1      ; Enable pin-change wake-up interrupt
MAIN:   …
        SLEEP               ; Wait pin-change of PA, PB or PD to wake-up
        LGOTO   MAIN
INT:
        MOVWX   20h         ; Store W data to SRAM 20h
        MOVXW   STATUS      ; Get STATUS data
        MOVWX   21h         ; Store STATUS data to SRAM 21h

        BTXSC   PCIF        ; Check PCIF bit
        LCALL   PC_SUB      ; PCIF = 1, jump to pin-change wake-up interrupt
        …                   ; service routine
EXIT_INT:
```

```
        MOVXW   21h             ; Get SRAM 21h data
        MOVWX   STATUS          ; Restore STATUS data
        MOVXW   20h             ; Restore W data
        RETI                    ; Return from interrupt

PC_SUB:                         ; pin-change wake-up interrupt service routine
        …
        MOVLW   10111111b
        MOVWX   INTIF1          ; Clear pin-change wake-up interrupt request flag
        RET
```

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

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

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

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

0Bh.3 **WKTIE:** Wake-up Timer interrupt enable and Wake-up Timer enable
0: disable
1: enable

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

0Bh.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	–	TM1IF	TM0IF	WKTIF	–	INT1IF	INT0IF
R/W	R/W	–	R/W	R/W	R/W	–	R/W	R/W
Reset	0	–	0	0	0	–	0	0

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

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

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

0Ch.3 **WKTIF:** Wake-up Timer interrupt event pending flag
This bit is set by H/W while Wake-up Timer is timeout, write F7h to INTIF will 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

0Dh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTIE1	CFCIE	PCIE	–	CMPIE	–	PWM1IE	PWM0IE	LVDIE
R/W	R/W	R/W	–	R/W	–	R/W	R/W	R/W
Reset	0	0	0	0	–	0	0	0

0Dh.7 **CFCIE:** Capture frequency count interrupt enable
0: disable
1: enable

0Dh.6 **PCIE:** All port pin-change wake-up interrupt enable
0: disable
1: enable

0Dh.5 **Must be kept at 0**

0Dh.4 **CMPIE:** Comparator interrupt enable
0: disable
1: enable

0Dh.2 **PWM1IE:** PWM1~5 interrupt enable
0: disable
1: enable

0Dh.1 **PWM0IE:** PWM0 interrupt enable
0: disable
1: enable

0Dh.0 **LVDIE:** LVD interrupt enable
0: disable
1: enable

0Eh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTIF1	CFCIF	PCIF	–	CMPIF	–	PWM1IF	PWM0IF	LVDIF
R/W	R/W	R/W	–	R/W	–	R/W	R/W	R/W
Reset	0	0	–	0	–	0	0	0

0Eh.7 **CFCIF:** Capture frequency count interrupt event pending flag
This bit is set by H/W while capture frequency count is finished, write 7Fh to INTIF1 will clear this flag

0Eh.6 **PCIF:** All port pin-change wake-up interrupt event pending flag
This bit is set by H/W at all pin's falling/rising edge, write BFh to INTIF1 will clear this flag. A sleep instruction is necessary before the event of pin-change otherwise pin-change event may be missed.

0Eh.4 **CMPIF:** Comparator interrupt event pending flag
This bit is set by H/W while CMPO match trigger condition, write EFh to INTIF1 will clear this flag

0Eh.2 **PWM1IF:** PWM1~5 interrupt event pending flag
This bit is set by H/W after PWM1~5 period counter roll over, write FBh to INTIF1 will clear this flag

0Eh.1 **PWM0IF:** PWM0 interrupt event pending flag
This bit is set by H/W after PWM0 period counter roll over, write FDh to INTIF1 will clear this flag

0Eh.0 **LVDIF:** LVD interrupt event pending flag
This bit is set by H/W after $V_{CC} < V_{LVD}$, write FEh to INTIF1 will clear this flag

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

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

16h.7 **LVDF**: Low voltage detection flag
0: $V_{CC} > V_{LVD}$
1: $V_{CC} < V_{LVD}$

16h.6 **LVDHYS**: LVD Hysteresis
0: disable
1: enable

16h.4 **LVDSAV**: LVD auto power off in STOP/IDLE mode
0: disable LVD auto power off in STOP/IDLE mode
1: enable LVD auto power off in STOP/IDLE mode

16h.3~0 **LVDS**: LVD voltage (V_{LVD}) select
Please refer to the table of LVD voltage in the section of "LVD Circuit Characteristics". (Left click the link to go to that page)

5 I/O Port

5.1 PA0-PA7, PB0-PB7, PD0-PD5

Each IO has 4 bits as the mode setting. The mode setting can include the following functions: open drain output, CMOS output, pull-up resistor, pull-down resistor, pin-changed wake-up, PWMO, TCOUT, TM1OUT and so on. All IO support two sink current options, which are defined by the HSINK (81h.4).

These pins can be operated in different modes as below table.

PAxMOD PBxMOD PDxMOD	PADx PBDx PDDx	PA0~PA7, PB0~PB7, PD0~PD5 pin function	Pin State	Resistor Pull-up	Digital Input	Pin-changed Wake-up
0000b	0	Open Drain	Drive Low	-	-	-
	1	Input	Pull-up	Y	Y	-
0001b	0	Open Drain	Drive Low	-	-	-
	1	Input	Hi-Z	-	Y	-
0010b	0	CMOS Output (except PWMx)	Drive Low	-	-	-
	1		Drive High	-	-	-
0011b	X	Analog input/output for ADCx / CINx / CIPx	Hi-Z	-	-	-

I/O Pin Function Table 1

PAxMOD PBxMOD PDxMOD	PADx PBDx PDDx	PA0~PA7, PB0~PB7, PD0~PD5 pin function	Pin State	Resistor Pull-down	Digital Input	Pin-changed Wake-up
0100b	0	Open Drain	Drive Low	-	-	-
	1	Input	Pull-down*	Y*	Y	-
0101b	0	Open Drain	Drive Low	-	-	-
	1	Input	Hi-Z	-	Y	-
0110b	0	CMOS Output (except PWMx)	Drive Low	-	-	-
	1		Drive High	-	-	-
0111b	X	Function CMOS output for PWMx / TCOUT / TM1OUT	-	-	-	-

I/O Pin Function Table 2

PAxMOD PBxMOD PDxMOD	PADx PBDx PDDx	PA0~PA7, PB0~PB7, PD0~PD5 pin function	Pin State	Resistor Pull-up	Digital Input	Pin-changed Wake-up
1000b	0	Open Drain	Drive Low	-	-	-
	1	Input	Pull-up	Y	Y	Y
1001b	0	Open Drain	Drive Low	-	-	-
	1	Input	Hi-Z	-	Y	Y
1010b	0	CMOS Output (except PWMx)	Drive Low	-	-	-
	1		Drive High	-	-	-
1011b		Reserved				

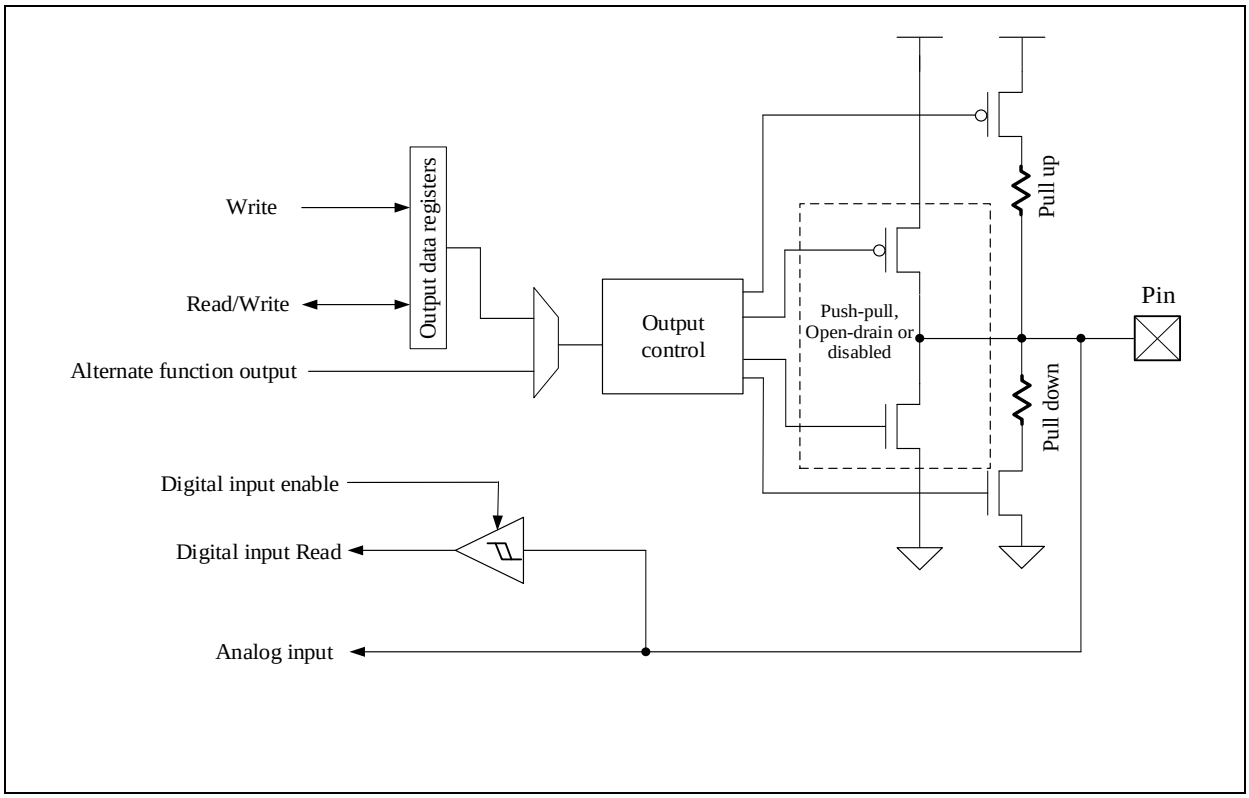
I/O Pin Function Table 3

PAxMOD PBxMOD PDxMOD	PADx PBDx PDDx	PA0~PA7, PB0~PB7, PD0~PD1 pin function	Pin State	Resistor Pull-down	Digital Input	Pin-changed Wake-up
1100b	0	Open Drain	Drive Low	-	-	-
	1	Input	Pull-down*	Y*	Y	Y
1101b	0	Open Drain	Drive Low	-	-	-
	1	Input	Hi-Z	-	Y	Y
1110b	0	CMOS Output (except PWMx)	Drive Low	-	-	-
	1		Drive High	-	-	-
1111b	X	Analog output for 1/2 V_{CC} (LCD 1/2 bias) Does not contain PA7	1/2 V_{CC}	-	-	-
		- (PA7)	Pull-up			

I/O Pin Function Table 4

Pin Name	PAxMOD / PBxMOD / PDxMOD Setting		
	0011b (Analog in/out)	0111b (Digital output)	1111b (Analog output)
PA0	ADC0	PWM5O	COM0 (LCD1/2 bias)
PA1	ADC1 CIP1	PWM1O	COM1 (LCD1/2 bias)
PA2	ADC2 CIP2	PWM4O	COM2 (LCD1/2 bias)
PA3	ADC3 CIN	PWM2O	COM3 (LCD1/2 bias)
PA4	ADC4	PWM0P	COM4 (LCD1/2 bias)
PA5	ADC5	PWM3O	COM5 (LCD1/2 bias)
PA6	ADC6	PWM0N	COM6 (LCD1/2 bias)
PA7	ADC23	PWM3O	COM21 (LCD1/2 bias)
PB0	ADC7	PWM1O	COM7 (LCD1/2 bias)
PB1	ADC8	PWM2O	COM8 (LCD1/2 bias)
PB2	ADC9	PWM3O	COM9 (LCD1/2 bias)
PB3	ADC17	PWM4O	COM10 (LCD1/2 bias)
PB4	ADC10	PWM0P	COM11 (LCD1/2 bias)
PB5	ADC11	PWM5O	COM12 (LCD1/2 bias)
PB6	ADC12 CIP3	PWM0N	COM13 (LCD1/2 bias)
PB7	ADC13	TM1OUT	COM14 (LCD1/2 bias)
PD0	ADC15		COM15 (LCD1/2 bias)
PD1	ADC16 CIP4	TCOUT	COM16 (LCD1/2 bias)
PD2	ADC19	PWM0P	COM17 (LCD 1/2 bias)
PD3	ADC20	PWM0N	COM18 (LCD 1/2 bias)
PD4	ADC21	PWM4O	COM19 (LCD 1/2 bias)
PD5	ADC22	PWM1O	COM20 (LCD 1/2 bias)

Special function for PAxMOD/PBxMOD/PDxMOD Table



General Pin Structure

85h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PAMOD10	PA1MOD				PA0MOD			
R/W	R/W				R/W			
Reset	0	0	0	1	0	0	0	1

86h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PAMOD32	PA3MOD				PA2MOD			
R/W	R/W				R/W			
Reset	0	0	0	1	0	0	0	1

87h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PAMOD54	PA5MOD				PA4MOD			
R/W	R/W				R/W			
Reset	0	0	0	1	0	0	0	1

88h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PAMOD76	PA7MOD				PA6MOD			
R/W	R/W				R/W			
Reset	0	0	0	0	0	0	0	1

88h.7~4 **PA7MOD ~ PA0MOD**: PA7~PA0 Pin Mode Control
88h.3~0 0000: Open drain or digital input with pull-up
87h.7~4 0001: Open drain or digital input
87h.3~0 0010: CMOS Push-pull
86h.7~4 0011: Analog input/output
86h.3~0 0100: Open drain or digital input with pull-down
85h.7~4 0101: Open drain or digital input
85h.3~0 0110: CMOS Push-pull

0111: Alternate function output
1000: Open drain or digital input with pull-up and pin-changed wake-up
1001: Open drain or digital input and pin-changed wake-up
1010: CMOS Push-pull
1011: Reserved
1100: Open drain or digital input with pull-down and pin-changed wake-up
1101: Open drain or digital input and pin-changed wake-up
1110: CMOS Push-pull
1111: 1/2 V_{CC} (LCD 1/2 bias)

8Ch	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PBMOD10	PB1MOD				PB0MOD			
R/W	R/W				R/W			
Reset	0	0	0	1	0	0	0	1

8Dh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PBMOD32	PB3MOD				PB2MOD			
R/W	R/W				R/W			
Reset	0	0	0	1	0	0	0	1

8Eh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PBMOD54	PB5MOD				PB4MOD			
R/W	R/W				R/W			
Reset	0	0	0	1	0	0	0	1

8Fh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PBMOD76	PB7MOD				PB6MOD			
R/W	R/W				R/W			
Reset	0	0	0	1	0	0	0	1

8Fh.7~4 **PB7MOD ~ PB0MOD**: PB7~PB0 Pin Mode Control
8Fh.3~0 0000: Open drain or digital input with pull-up
8Eh.7~4 0001: Open drain or digital input
8Eh.3~0 0010: CMOS Push-pull
8Dh.7~4 0011: Analog input
8Dh.3~0 0100: Open drain or digital input with pull-down
8Ch.7~4 0101: Open drain or digital input
8Ch.3~0 0110: CMOS Push-pull
0111: Alternate function output
1000: Open drain or digital input with pull-up and pin-changed wake-up
1001: Open drain or digital input and pin-changed wake-up
1010: CMOS Push-pull
1011: Reserved
1100: Open drain or digital input with pull-down and pin-changed wake-up
1101: Open drain or digital input and pin-changed wake-up
1110: CMOS Push-pull
1111: 1/2 V_{CC} (LCD 1/2 bias)

90h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PDMOD10	PD1MOD				PD0MOD			
R/W	R/W				R/W			
Reset	0	0	0	1	0	0	0	1

9Bh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PDMOD32	PD3MOD				PD2MOD			
R/W	R/W				R/W			
Reset	0	0	0	1	0	0	0	1

9Ch	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PDMOD54	PD5MOD				PD4MOD			
R/W	R/W				R/W			
Reset	0	0	0	1	0	0	0	1

90h.7~4 **PD5MOD ~ PD0MOD**: PD5~PD0 Pin Mode Control
90h.3~0 0000: Open drain or digital input with pull-up
9Bh.7~4 0001: Open drain or digital input
9Bh.3~0 0010: CMOS Push-pull
9Ch.7~4 0011: Analog input
9Ch.3~0 0100: Open drain or digital input with pull-down
0101: Open drain or digital input
0110: CMOS Push-pull
0111: Alternate function output
1000: Open drain or digital input with pull-up and pin-changed wake-up
1001: Open drain or digital input and pin-changed wake-up
1010: CMOS Push-pull
1011: Reserved
1100: Open drain or digital input with pull-down and pin-changed wake-up
1101: Open drain or digital input and pin-changed wake-up
1110: CMOS Push-pull
1111: 1/2 V_{CC} (LCD 1/2 bias)

05h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PAD	PAD							
R/W	R/W							
Reset	1	1	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
PDD	–	–	PDD					
R/W	–	–	R/W					
Reset	–	–	1	1	1	1	1	1

07h.1~0 **PDD**: PD1~PD0 data

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

81h.4 **HSINK**: All IO ports high sink current enable
0: low sink current
1: high sink current

105h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PINMOD	IODR		TCOE	–	–	–	Reserved	Reserved
R/W	R/W		R/W	–	–	–	R/W	R/W
Reset	1	0	0	–	–	–	0	0

105h.7~6 **IODR:** All IO port drive current settings
00: 4mA
01: 8mA
10: 12mA(default setting)
11: 18mA

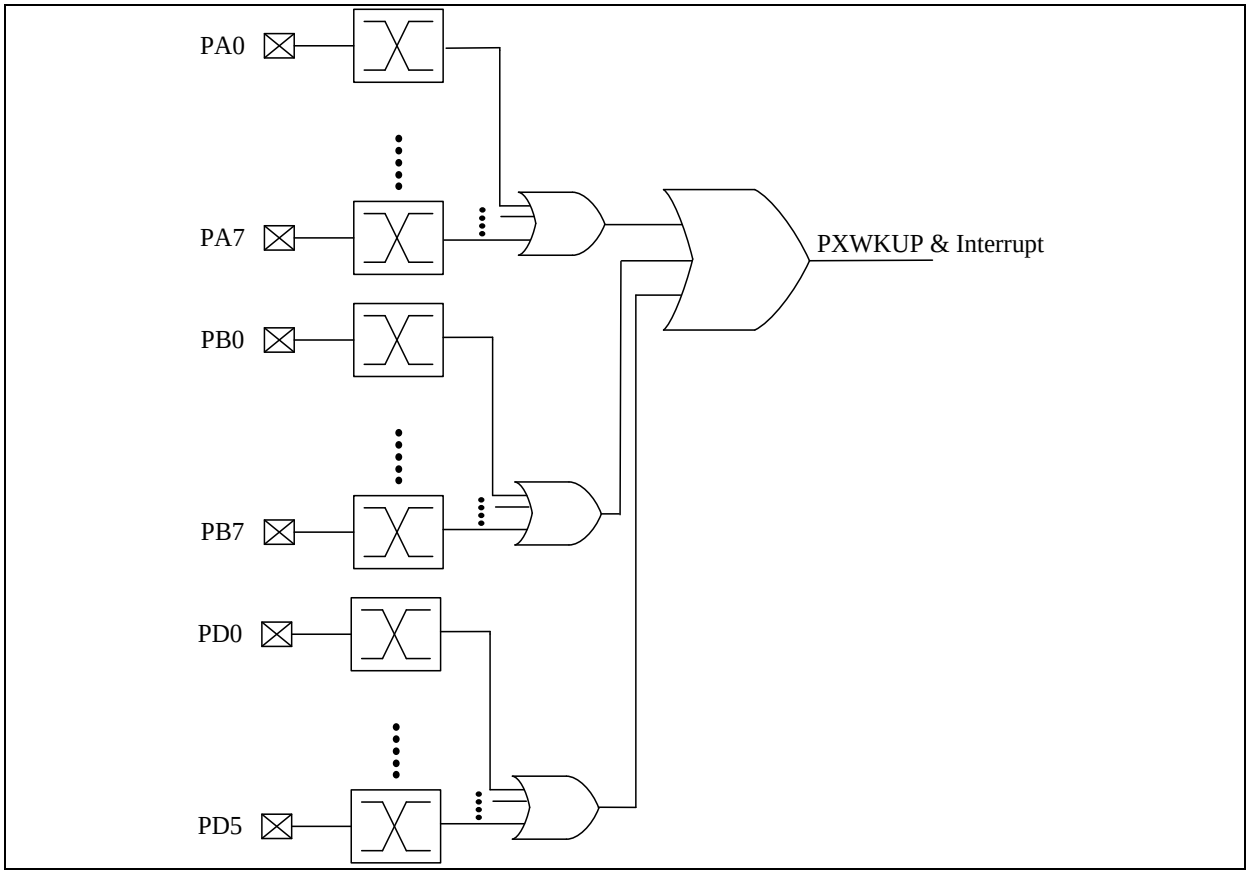
105h.5 **TCOE:** TCO(Fsys/2) allows output**.**
0: disable TCO(Fsys/2) output to PD1, PD1 as normal IO
1: enable TCO(Fsys/2) output to PD1

105h.1 **Reserved**: must be kept at 0

105h.0 **Reserved**: must be kept at 0

5.2 Pin-change Wake-up & Interrupt

All of the IO pins also have the pin-change wake-up and interrupt capability. A sleep instruction is necessary before the event of pin-change otherwise pin-change event may be missed.

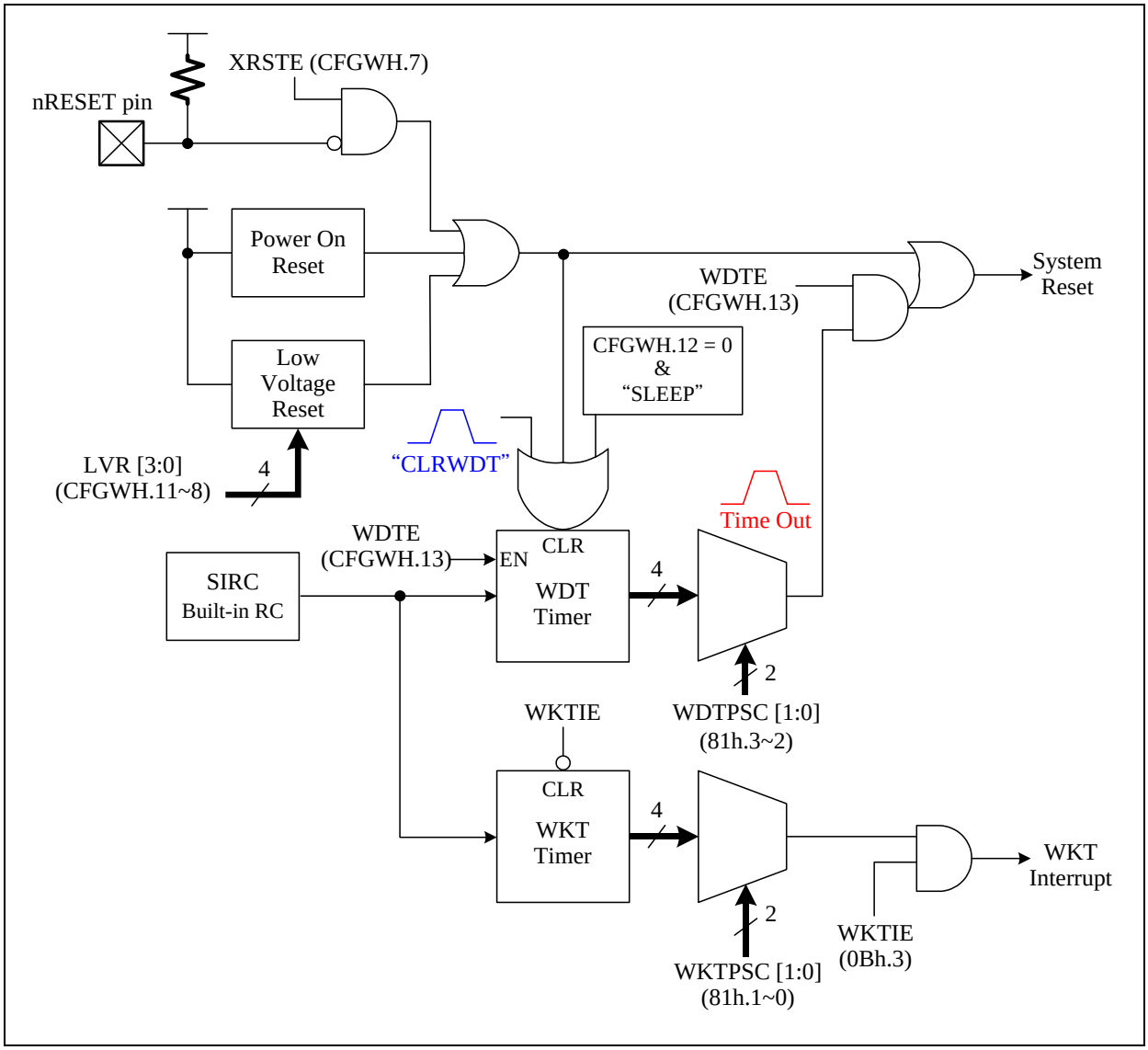


6 Peripheral Functional Block

6.1 Watchdog (WDT) /Wake-up (WKT) Timer

The WDT and WKT share the same built-in internal RC Oscillator and have individual 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, the WDT generates the chip reset signal.

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 Block Diagram

The WDT's behavior in different Mode is shown as below table.

Mode	CFGWH[13:12]		WDT
	WDTE[1]	WDTE[0]	
Normal Mode	0	0	Stop
	0	1	Stop
	1	0	Run
	1	1	Run
Power-down Mode (SLEEP)	0	0	Stop
	0	1	Stop
	1	0	Stop
	1	1	Run

Watchdog clear is controlled by CLRWDT instruction.

◇ Example: Clear watchdog timer by CLRWDT instruction.

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

◇ Example: Setup WDT time.

```
        MOVLW   0000**01**11b
        MOVWX   OPTION          ; Select WDT Time out=348 ms @5V
        …
```

◇ Example: Set WKT period and interrupt function.

```
        MOVLW   000001**10**b
        MOVWX   OPTION          ; Select WKT period=172 ms @5V
        MOVLW   1111**0**111b   ; Clear WKT interrupt flag by using byte operation
        MOVWX   INTIF           ; Don't use bit operation "BCX WKTIF" to clear

        BSX     WKTIE           ; Enable WKT interrupt function
```

03h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
STATUS	IRP	RP1	RP0	TO	PD	Z	DC	C
R/W	R/W	R/W	R/W	R	R	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

03h.4 **TO:** WDT time out flag, read-only
0: after Power On Reset or CLRWDT / SLEEP instructions
1: WDT time out occurs

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

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

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

0Bh.3 **WKTIE:** Wake-up Timer interrupt enable and Wake-up Timer enable
0: disable
1: enable

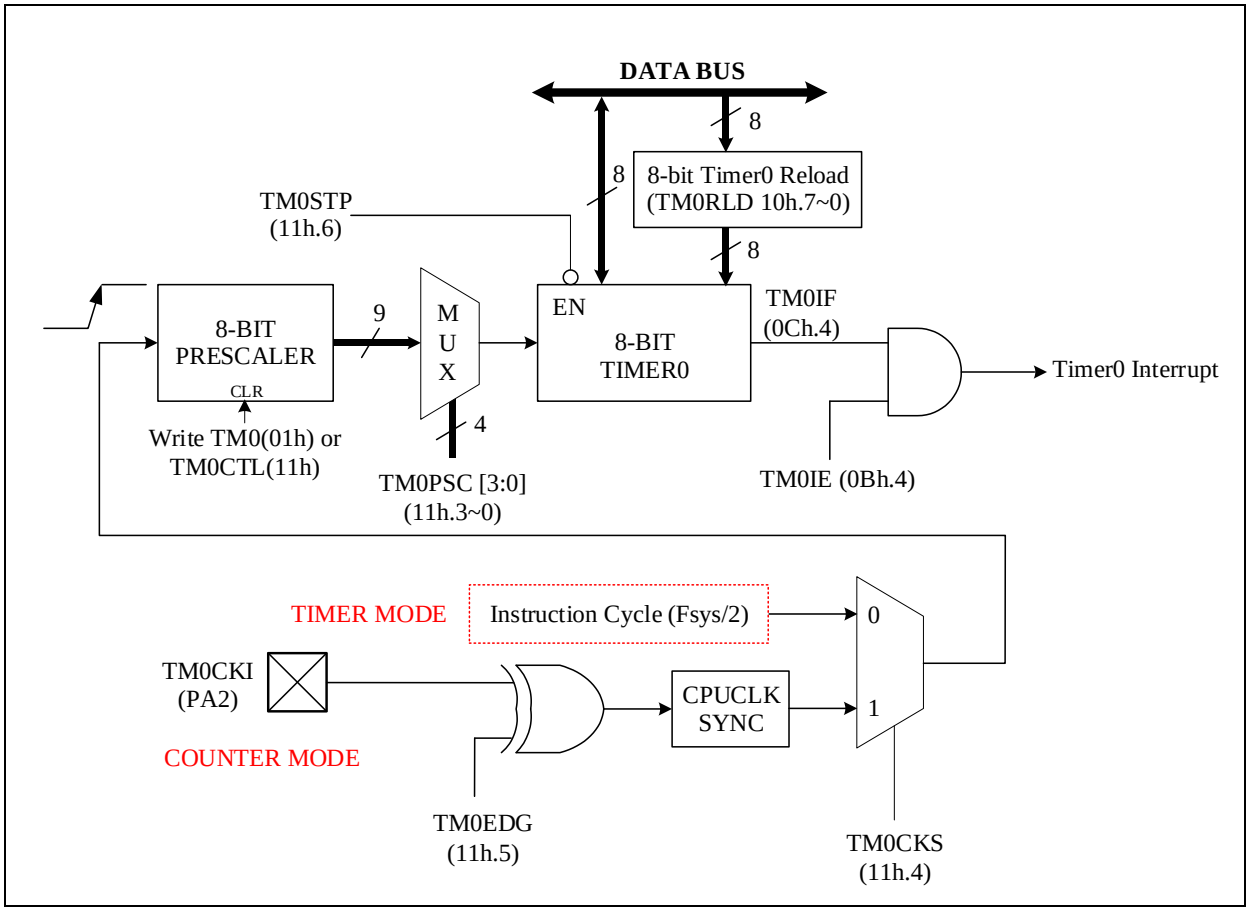
81h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
OPTION	HWAUTO	INT0EDG	INT1EGE	HSINK	WDTPSC		WKTPSC	
R/W	R/W	R/W	R/W	R/W	R/W		R/W	
Reset	0	0	0	0	1	1	1	1

81h.3~2 **WDTPSC:** WDT period (@V_{CC}=5V)
00: 175 ms
01: 348 ms
10: 1381 ms
11: 2761 ms

81h.1~0 **WKTPSC:** WKT period (@V_{CC}=5V/3V)
00: 86 ms /100ms
01: 172 ms /201ms
10: 343 ms /407ms
11: 686 ms /813ms

6.2 Timer0

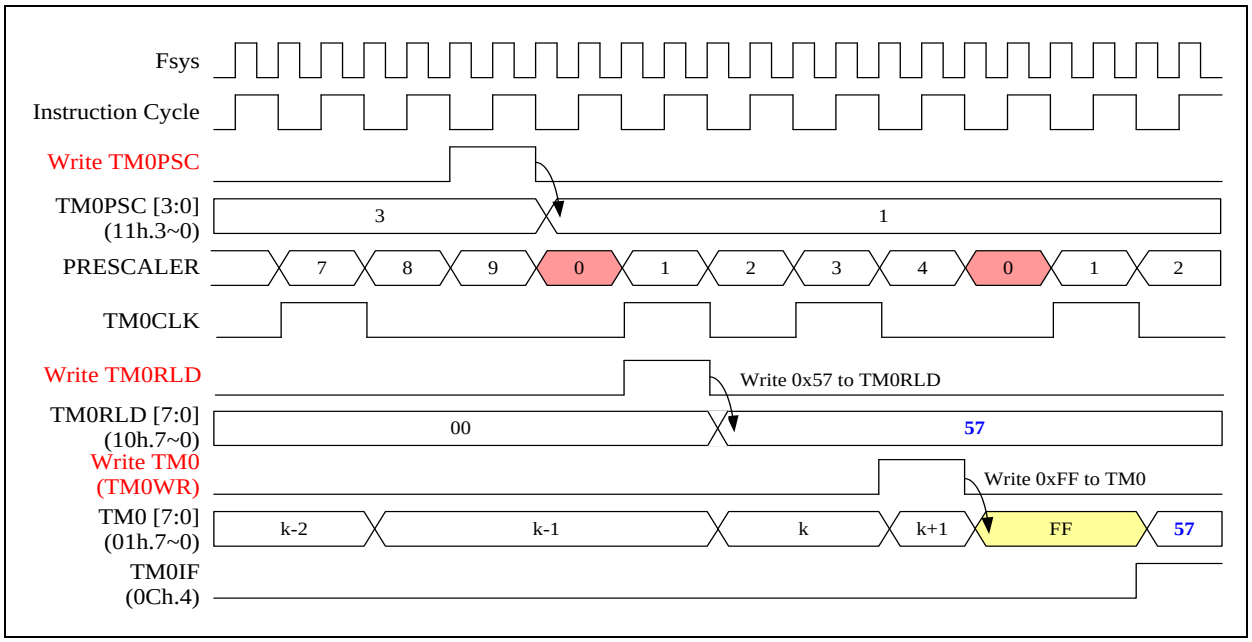
Timer0(TM0) (01h.7~0) is an 8-bit wide register. It can be read or written as any other register. Besides, Timer0 increases itself periodically and automatically rolls over a new "offset value" (TM0RLD) while it rolls over based on the pre-scaled clock source, which can be Fsys/2 or TM0CKI (PA2) rising/falling input. The Timer0 increase rate is determined by “Timer0 Pre-Scale” (TM0PSC) register. The Timer0 always generates TM0IF (0Ch.4) when its count rolls over. It generates Timer0 Interrupt if TM0IE (0Bh.4) is set. Timer0 can be stopped counting if the TM0STP (11h.6) bit is set.



Timer0 Block Diagram

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 Timer0 interrupt time value is as following:

Timer0 interrupt interval cycle time = Fsys / 2 / TM0PSC / (256-TM0RLD)

◇ Example: Setup Timer0 work in Timer mode, if Fsys = 8 MHz

```
; Setup Timer0 clock source and divider
        MOVLW   00x00101b       ; TM0CKS = 0, Timer0 clock is instruction cycle
        MOVWX   TM0CTL          ; TM0PSC = 0101b, divided by 32

; Setup Timer0 reload data
        MOVLW   80h
        MOVWX   TM0RLD          ; Set Timer0 reload data = 128

; Setup Timer0
        BSX     TM0STP          ; Timer0 stops counting
        CLRX    TM0             ; Clear Timer0 content

; Enable Timer0 and interrupt function
        MOVLW   1110 1111b
        MOVWX   INTIF           ; Clear Timer0 request interrupt flag
        BSX     TM0IE           ; Enable Timer0 interrupt function
        BCX     TM0STP          ; Enable Timer0 counting
```

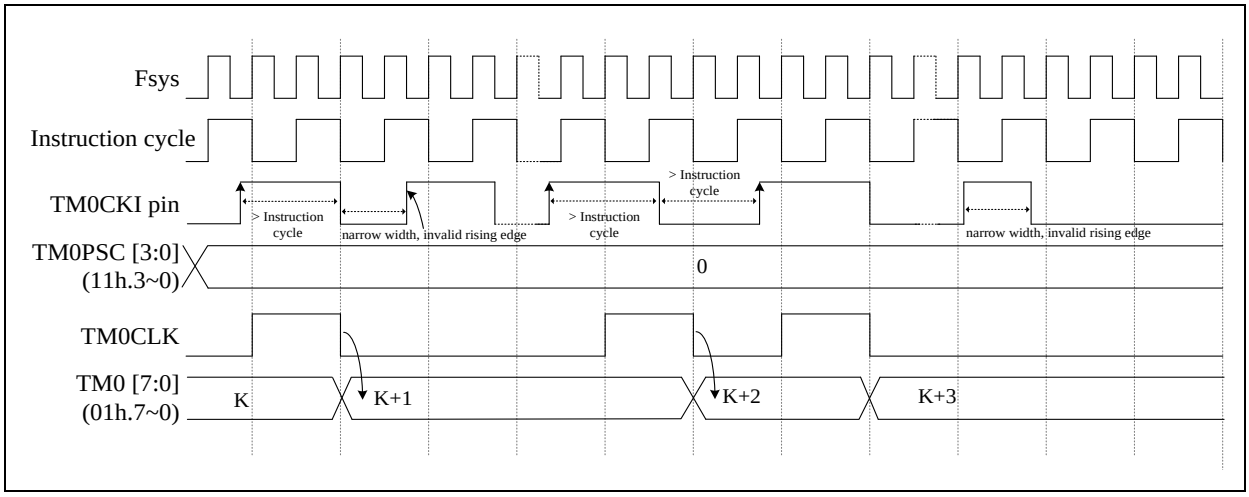
Timer0 interrupt frequency = Fsys / 2 / TM0PSC / (256-TM0RLD),

Fsys = 8 MHz, TM0PSC = div 32, TM0RLD = 128

Timer0 interrupt frequency = 8 MHz / 2 / 32 / (256-128) = 0.976 KHz

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 Timer0 clock source and divider
        MOVLW   00110000B       ; TM0EDG = 1, counting edge is falling edge
        MOVWX   TM0CTL          ; TM0CKS = 1, Timer0 clock is TM0CKI
                                ; TM0PSC = 0000b, divided by 1
; Setup Timer0
        BSX     TM0STP          ; Timer0 stops counting
        CLRX    TM0             ; Clear Timer0 content

; Enable Timer0 and read Timer0 counter
        BCX     TM0STP          ; Enable Timer0 counting
        …
        BSX     TM0STP          ; Timer0 stops counting
        MOVXW   TM0             ; Read Timer0 content
```

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

01h.7~0 **TM0:** Timer0 content

0Bh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTIE	ADCIE	–	TM1IE	TM0IE	WKTIE	–	INT1IE	INT0IE
R/W	R/W	–	R/W	R/W	R/W	–	R/W	R/W
Reset	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	–	TM1IF	TM0IF	WKTIF	–	INT1IF	INT0IF
R/W	R/W	–	R/W	R/W	R/W	–	R/W	R/W
Reset	0	–	0	0	0	–	0	0

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

10h.7~0 **TM0RLD:** Timer0 reload data

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

11h.6 **TM0STP:** Stop Timer0
0: Timer0 runs
1: Timer0 stops

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)

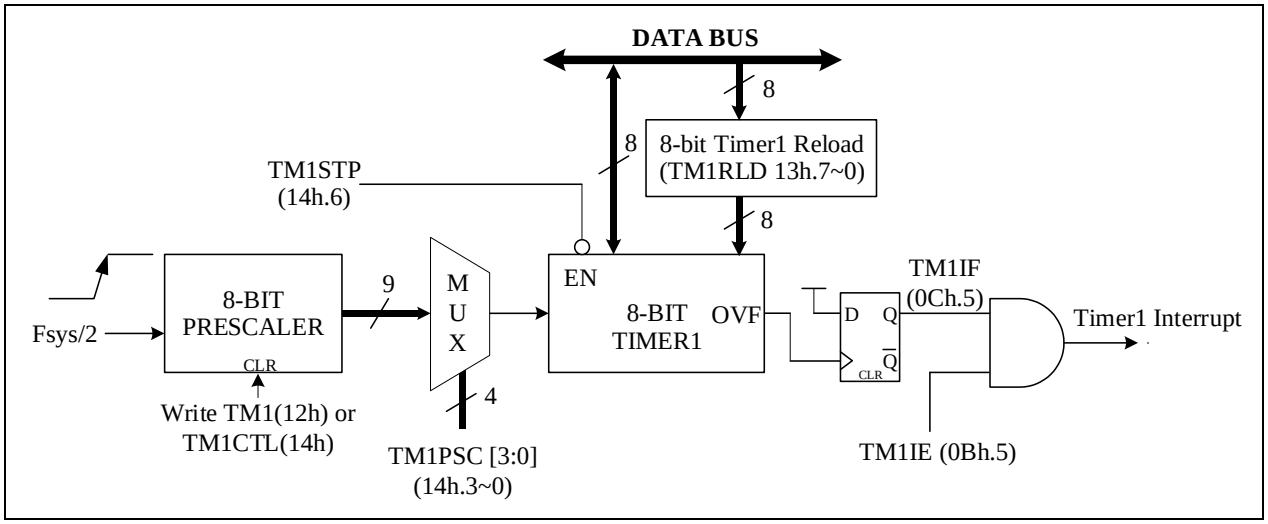
11h.3~0 **TM0PSC:** Timer0 prescaler. Timer0 prescaler clock source divided by

0000: 1	0001: 2	0010: 4	0011: 8
0100: 16	0101: 32	0110: 64	0111: 128

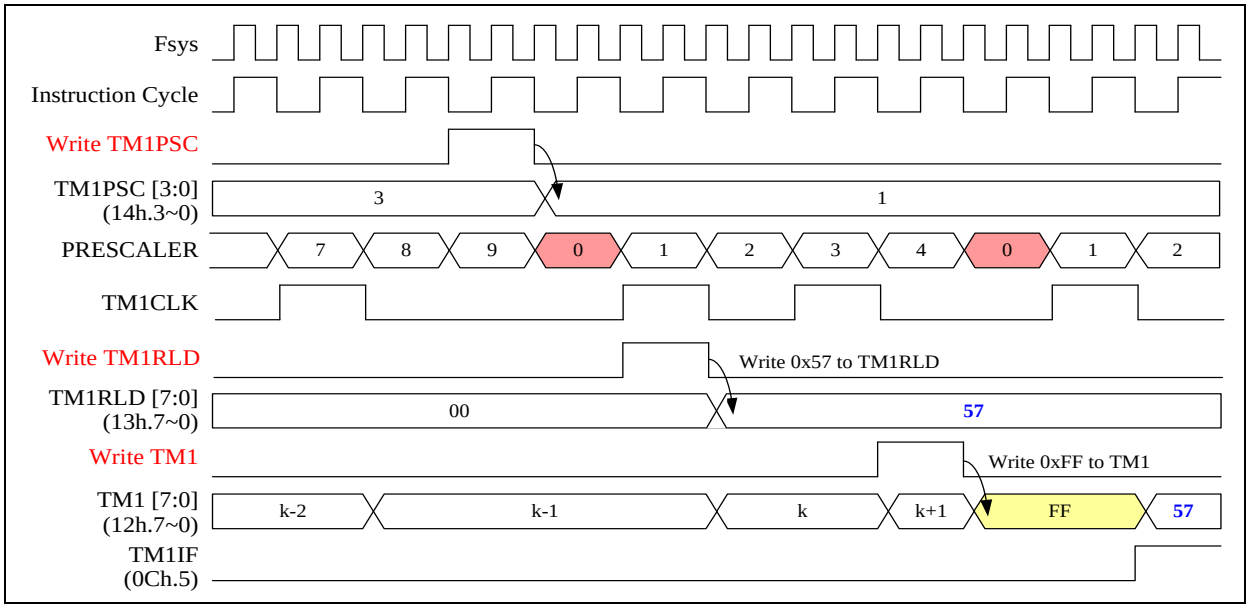
1xxx: 256

6.3 Timer1

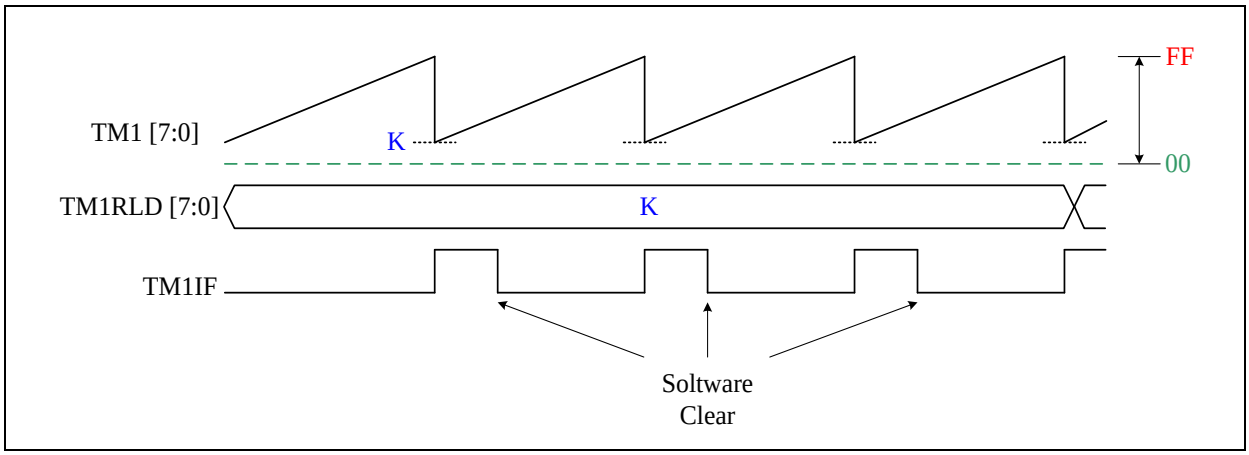
Timer1(TM1) (12h.7~0) is an 8-bit wide register. It can be read or written as any other register. Besides, Timer1 increases itself periodically and automatically reloads a new "offset value" (TM1RLD) while it rolls over based on the pre-scaled instruction clock (Fsys/2). The Timer1 increase rate is determined by TM1PSC register. It generates Timer1 interrupt if the TM1IE bit is set. Timer1 can be stopped counting if the TM1STP bit is set.



Timer1 Block Diagram



Timer1 Timing Diagram



Timer1 Reload Diagram

◇ Example: CPU is running in SLOW mode, Fsys = Slow-clock / CPUPSC = 92.8 KHz / 2 = 46.4 KHz

```
; Setup Timer1 clock source and divider
        MOVLW   0000 0011b
        MOVWX   TM1CTL          ; TM1PSC = 0011b, divided by 8

; Setup Timer1 reload data
        MOVLW   FFh
        MOVWX   TM1RLD          ; Set Timer1 reload data = 255

; Setup Timer1
        BSX     TM1STP          ; Timer1 stops counting
        CLRX    TM1             ; Clear Timer1 content

; Enable Timer1 and interrupt function
        MOVLW   11011111b
        MOVWX   INTIF           ; Clear Timer1 request interrupt flag
        BSX     TM1IE           ; Enable Timer1 interrupt function
        BCX     TM1STP          ; Enable Timer1 counting
```

Timer1 interrupt frequency = Fsys / 2 / TM1PSC / (256-TM1RLD),
Fsys = 46.4 KHz, TM1PSC = div 8, TM1RLD = 255
Timer1 interrupt frequency = 46.4 KHz / 2 / 8 / (256-255) = 2.9 KHz

0Bh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTIE	ADCIE	–	TM1IE	TM0IE	WKTIE	–	INT1IE	INT0IE
R/W	R/W	–	R/W	R/W	R/W	–	R/W	R/W
Reset	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	–	TM1IF	TM0IF	WKTIF	–	INT1IF	INT0IF
R/W	R/W	–	R/W	R/W	R/W	–	R/W	R/W
Reset	0	–	0	0	0	–	0	0

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

12h.7~0 **TM1:** Timer1 content

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

13h.7~0 **TM1RLD:** Timer1 reload data

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

14h.6 **TM1STP:** Stop Timer1
0: Timer1 runs
1: Timer1 stops

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

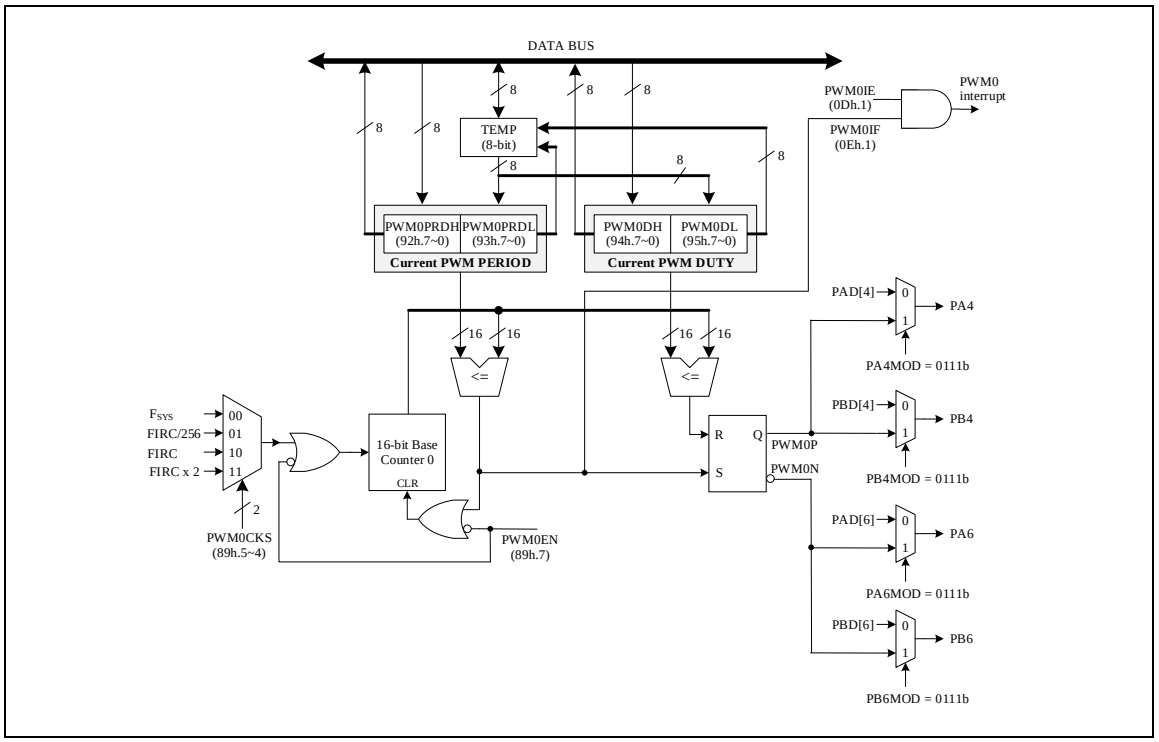
6.4 PWM: One 16-bit PWM and Five 8-bit PWMs

There are six PWMs in this chip. They could be divided into two groups. The first group is PWM0 which is 16-bit PWM. The second group is PWM1~5 which are 8-bit PWM. Each group has its own PWM clock source, period register and enable control.

PWM0~PWM5 have no dead-zone(non-overlap) control. PWM0 output are PWM0P and PWM0N. PWM1~5 outputs are PWM1O~PWM5O. User can use pin mode setting to output PWM0P, PWM0N or PWMxO to the corresponding IO pin. Please refer to Chapter 5 for more information on pin settings.

6.4.1 PWM0

PWM0 has 16-bit duty control register and 16-bit period register. PWM0 can generate varies frequency waveform with 65536 duty resolution on the basis of the PWM0 clock. The PWM0 clock can select Fsys, FIRC/256, FIRC (16 MHz), or FIRC*2 (32 MHz) as its clock source. The PWM0 structure is shown as follow.



PWM0 Block Diagram

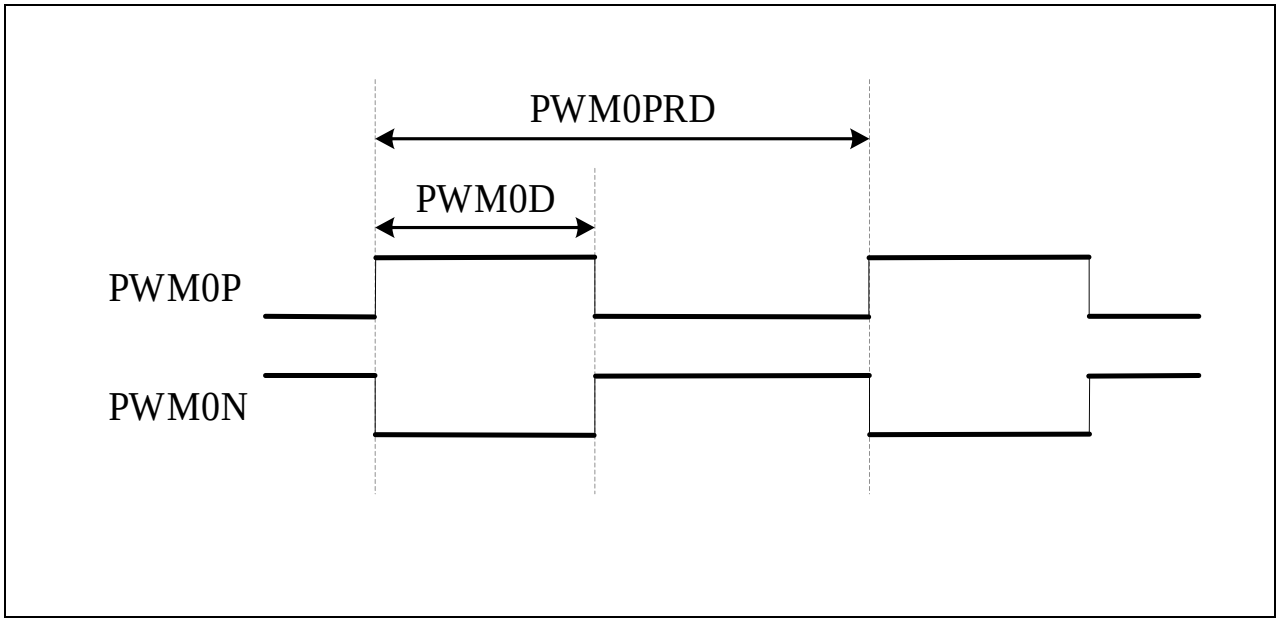
The 16-bit PWM0PRD and PWM0D registers both have a low byte and high byte structure. The high byte can be directly accessed, but the low byte can only be accessed via an internal 8-bit buffer. Reading or writing these register pairs must be carried out in a specific way. The important point to notes is that data transfer to and from the 8-bit buffer and its related low byte only takes place when writing or reading operation to its corresponding high bytes is executed. ***Briefly speaking, write low byte first and then high byte; read high byte first and then low byte***.

For reading and writing of 16-bit PWM0 period and duty, it is recommended to update the data only in the main program, or only update the data in the interrupt, to avoid possible errors.

If PWM0EN is cleared, the PWM0 will be cleared and stopped, otherwise the PWM0 remain running. The PWM0 duty cycle can be changed by writing to PWM0DH and PWM0DL. The PWM0 output signal resets to a low level whenever the 16-bit base counter matches the 16-bit PWM0 duty register {PWM0DH, PWM0DL}. The PWM0 period can be set by writing the period value to the PWM0PRDH

and PWM0PRDL registers. After writing the PWM0DH or PWM0PRDH register, H/W will update PWM0 period and duty immediately. PWM0 has an interrupt flag, PWM0IF, which is generated at the end of period.

PWM0 is a simple structure, which switches its output high and low at uniform repeatable intervals. The PWM0D is the output duty cycle, and the output period is PWM0PRD+1.



PWM0 Output

◇ Example:

```
; Setup Pin mode
        MOVLW   xxxx0111b       ;
        MOVWX   PAMOD54         ; PA4 Pin as PWM0P

        MOVLW   xxxx0111b       ;
        MOVWX   PAMOD76         ; PA6 Pin as PWM0N

; Setup PWM0 clock source select
        MOVLW   0010 0000b      ;
        MOVWX   PWMCTL          ; Set 16-MHz FIRC as PWM0 clock source

; Setup PWM0 period and duty setting
        MOVLW   FFh
        MOVWX   PWM0PRDL        ; write sequence: PWM0PRDL then PWM0PRDH
        MOVLW   7Fh
        MOVWX   PWM0PRDH        ; Set PWM0 period = 7FFFh

        MOVLW   00h
        MOVWX   PWM0DL          ; write sequence: PWM0DL then PWM0DH
        MOVLW   40h
        MOVWX   PWM0DH          ; Set PWM0 duty = 4000h
```

```
; Setup PWM0 enable and dead-zone(non-overlap) control
        MOVLW   1010 0000b          ; 89h.7 = 1, PWM0 enable
        MOVWX   PWMCTL              ; 89h.5~4 = 10, Set 16-MHz FIRC as PWM0 clock
                                    ; source
```

Example:

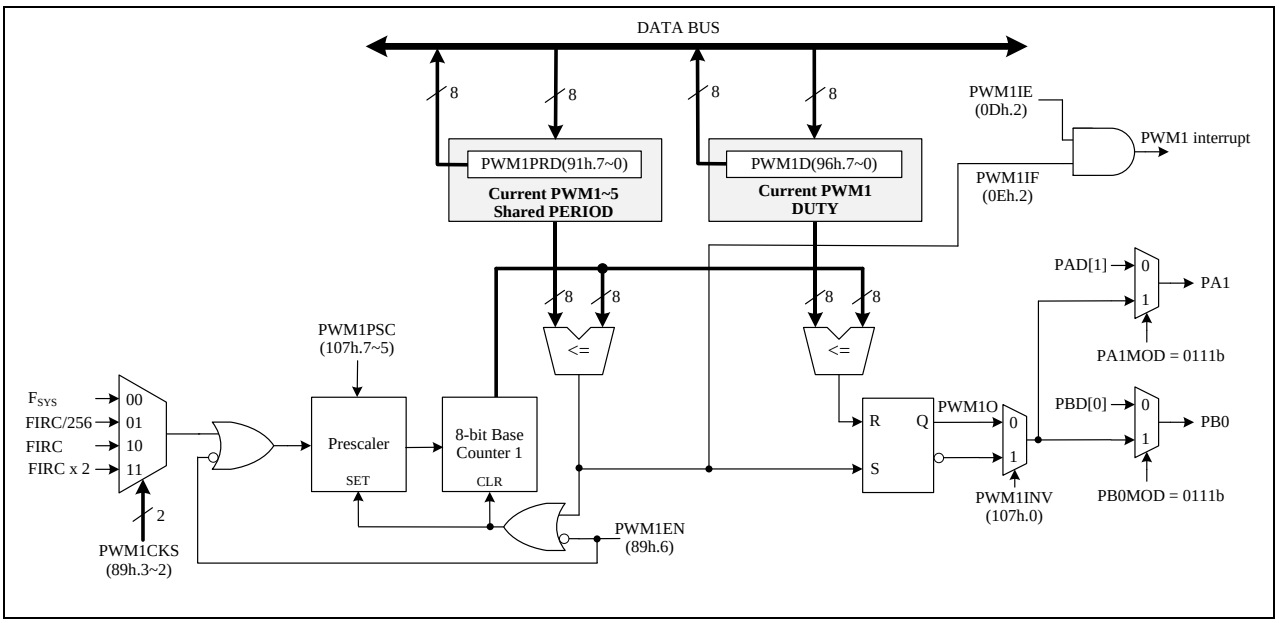
PWM0 clock source = FIRC 16 MHz, PWM0 period = 7FFFh, PWM0 duty = 4000h

PWM0 output frequency = 16 MHz / (period+1) = 16 MHz / 32768 = 488 Hz.

PWM0 output duty = duty / (period+1) = 50 %.

6.4.2 PWM1~5

PWM1~5 have 8-bit duty control register respectively, and **share a set of 8-bit period register and clock source**. PWM1~5 can generate varies frequency waveform with 256 duty resolution on the basis of their shared clock. The shared clock source of PWM1~5 can select Fsys, FIRC/256, FIRC (16 MHz), or FIRC*2 (32 MHz) as its clock source. The structure of PWM1 is shown as follow, and PWM2~5 are similar.



PWM1 Block Diagram

PWM1PRD and PWMxD(x=1~5) registers are 8-bit structure.

For reading and writing of 8-bit PWM1~5 period and duty, it is recommended to update the data only in the main program, or only update the data in the interrupt, to avoid possible errors.

If PWM1EN is cleared, the PWM1~5 will be cleared and stopped, otherwise the PWM1~5 remain running. The PWM1~5 duty cycle can be changed by writing to PWMxD(x=1~5). The PWM1~5 output signal resets to a low level whenever the 8-bit base counter matches the 8-bit PWM1~5 duty register PWMxD(x=1~5). The PWM1~5 periods can be set by writing the period value to the PWM1PRD registers. After writing the PWMxD(x=1~5) or PWM1PRD register, H/W will update PWM1~5 period and duty immediately. PWM1~5 share an interrupt flag, PWM1IF, which is generated at the end of the period.

0Dh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTIE1	CFCIE	PCIE	–	CMPIE	–	PWM1IE	PWM0IE	LVDIE
R/W	R/W	R/W	–	R/W	–	R/W	R/W	R/W
Reset	0	0	0	0	–	0	0	0

0Dh.2 **PWM1IE:** PWM1~5 interrupt enable
0: disable
1: enable

0Dh.1 **PWM0IE:** PWM0 interrupt enable
0: disable
1: enable

0Eh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTIF1	CFCIF	PCIF	–	CMPIF	–	PWM1IF	PWM0IF	LVDIF
R/W	R/W	R/W	–	R/W	–	R/W	R/W	R/W
Reset	0	0	–	0	–	0	0	0

0Eh.2 **PWM1IF:** PWM1~5 interrupt event pending flag
This bit is set by H/W after PWM1~5 period counter roll over, write FBh to INTIF1 will clear this flag

0Eh.1 **PWM0IF:** PWM0 interrupt event pending flag
This bit is set by H/W after PWM0 period counter roll over, write FDh to INTIF1 will clear this flag

89h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWMCTL	PWM0EN	PWM1EN	PWM0CKS		PWM1CKS			
R/W	R/W	R/W	R/W		R/W			
Reset	0	0	0	0	0	0		

89h.7 **PWM0EN:** PWM0 enable
0: disable
1: enable

89h.6 **PWM1EN:** PWM1~5 enable
0: disable
1: enable

89h.5~4 **PWM0CKS:** PWM0 clock source select
00: Fsys
01: FIRC/256
10: FIRC (16 MHz)
11: FIRC x 2 (32 MHz)

89h.3 ~2 **PWM1CKS:** PWM1 clock source select
00: Fsys
01: FIRC/256
10: FIRC (16 MHz)
11: FIRC x 2 (32 MHz)

91h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWM1PRD	PWM1PRD							
R/W	R/W							
Reset	1	1	1	1	1	1	1	1

91h.7~0 **PWM1PRD:** PWM1~5 period byte

92h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWM0PRDH	PWM0PRDH							
R/W	R/W							

Reset	1	1	1	1	1	1	1	1

92h.7~0 **PWM0PRDH:** PWM0 period high byte
write sequence: PWM0PRDL then PWM0PRDH
read sequence: PWM0PRDH then PWM0PRDL

93h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWM0PRDL	PWM0PRDL							
R/W	R/W							
Reset	1	1	1	1	1	1	1	1

93h.7~0 **PWM0PRDL:** PWM0 period low byte
write sequence: PWM0PRDL then PWM0PRDH
read sequence: PWM0PRDH then PWM0PRDL

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

94h.7~0 **PWM0DH:** PWM0 duty high byte
write sequence: PWMxDL then PWMxDH
read sequence: PWMxDH then PWMxDL

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

95h.7~0 **PWM0DL:** PWM0 duty low byte
write sequence: PWMxDL then PWMxDH
read sequence: PWMxDH then PWMxDL

96h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWM1D	PWM1D							
R/W	R/W							
Reset	1	0	0	0	0	0	0	0

96h.7~0 **PWM1D:** PWM1 duty byte

97h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWM2D	PWM2D							
R/W	R/W							
Reset	1	0	0	0	0	0	0	0

97h.7~0 **PWM2D:** PWM2 duty byte

98h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWM3D	PWM3D							
R/W	R/W							
Reset	1	0	0	0	0	0	0	0

98h.7~0 **PWM3D:** PWM3 duty byte

99h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWM4D	PWM4D							
R/W	R/W							
Reset	1	0	0	0	0	0	0	0

99h.7~0 **PWM4D:** PWM4 duty byte

9Ah	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWM5D	PWM5D							
R/W	R/W							
Reset	1	0	0	0	0	0	0	0

9Ah.7~0 **PWM5D:** PWM5 duty byte

107h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWMCTL2	PWM1PSC			PWM5INV	PWM4INV	PWM3INV	PWM2INV	PWM1INV
R/W	R/W			R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

107h.7~5 **PWM1PSC:** PWM1~5 Prescaler
000: DIV1 001: DIV2 010: DIV4 011: DIV8
100: DIV16 101: DIV32 110: DIV64 111: DIV128

107h.4 **PWM5INV:** PWM5 Inversion Selection
0: Disable Inversion
1: Enable Inversion

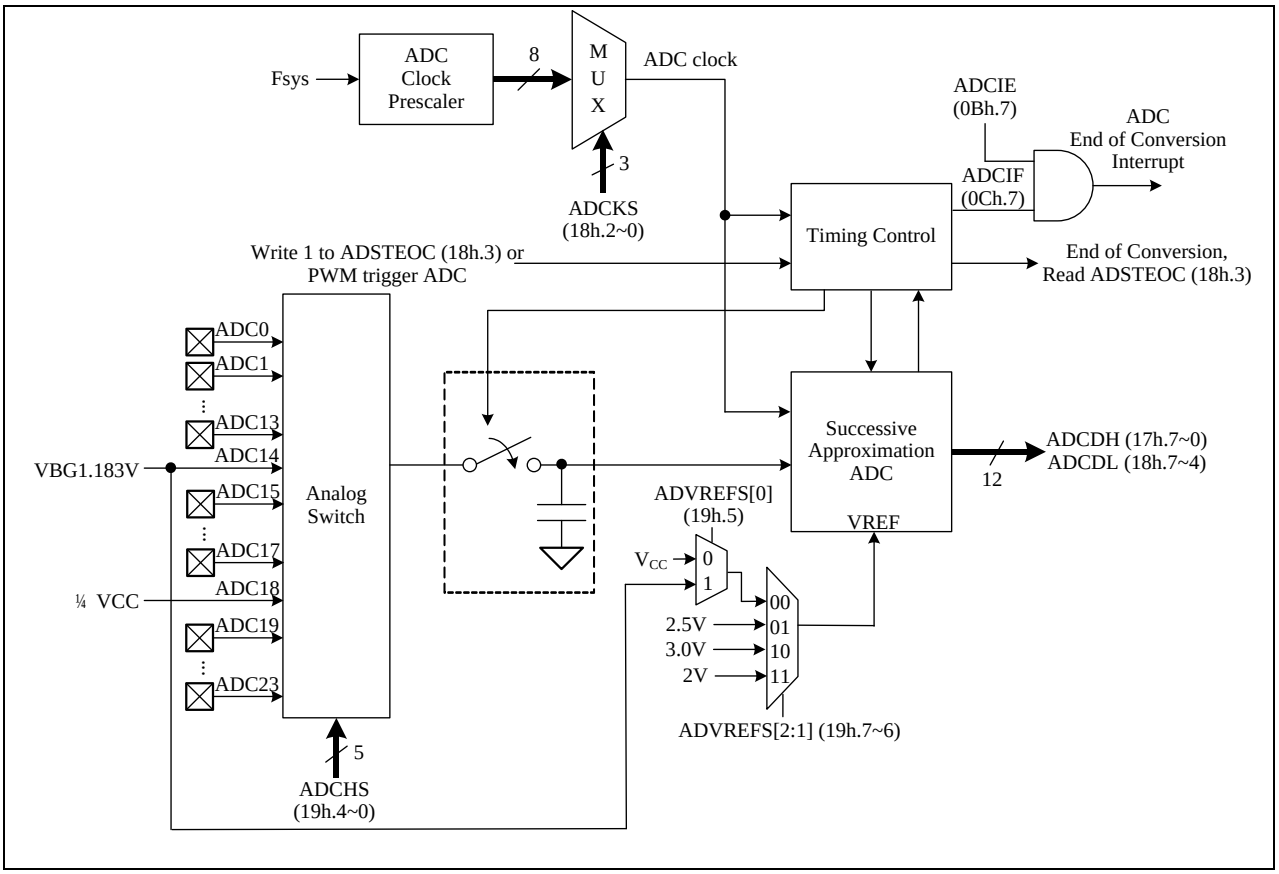
107h.3 **PWM4INV:** PWM4 Inversion Selection
0: Disable Inversion
1: Enable Inversion

107h.2 **PWM3INV:** PWM3 Inversion Selection
0: Disable Inversion
1: Enable Inversion

107h.1 **PWM2INV:** PWM2 Inversion Selection
0: Disable Inversion
1: Enable Inversion

107h.0 **PWM1INV:** PWM1 Inversion Selection
0: Disable Inversion
1: Enable Inversion

6.5 Analog-to-Digital Converter

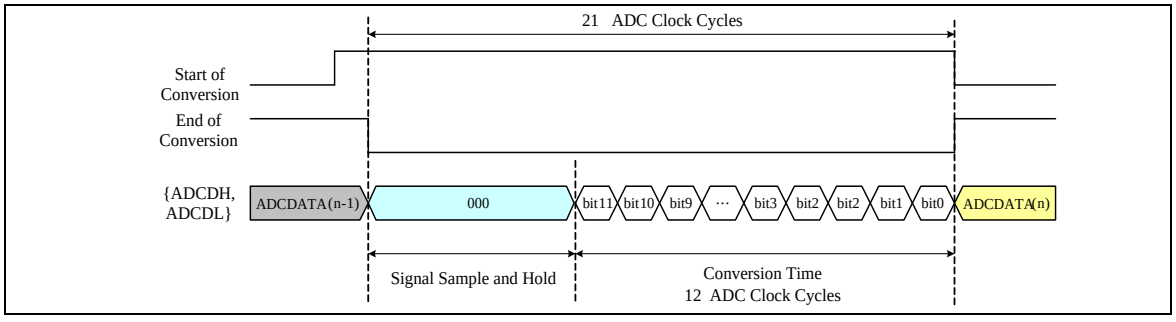


ADC Block Diagram

6.5.1 ADC Normal Mode

The 12-bit ADC (Analog to Digital Converter) consists of a 19-channel analog input multiplexer, control register, clock generator, 12-bit successive approximation register, and output data register.

To use the ADC, user needs to set ADCKS (18h.2~0) to choose a proper ADC clock frequency, which must be less than 2 MHz. User then launches the ADC conversion by setting the ADSTEOC (18h.3) control bit. After end of conversion, H/W automatic clears the ADSTEOC (18h.3) bit. User can poll this bit to know the conversion status. When the IO pin is used as the ADC input pin, the corresponding pin mode should be set to 0011b. User needs to set ADCHS (19h.4~0) to choose the input channel of ADC. Besides, there are some reference input channel can be selected, ADC14 is VBG and ADC18 is 1/4VCC for ADC. ADC reference voltage can be configured as V_{CC} or V_{BG} by ADVREFS (19h.7~5). Furthermore, if ADVREFS is changed to 2.53V, 3.0V or 2V, it will need 200uS warm-up stable time. When ADCHS is selected to VBG, ADVREFS must be set to V_{CC}, otherwise ADC conversion will be invalid.



ADC Timing Diagram

Example:

[CPU running at FAST mode , Fsys = FIRC 16 MHz]
ADC clock frequency = 1 MHz, ADC channel = ADC2 (PA2).

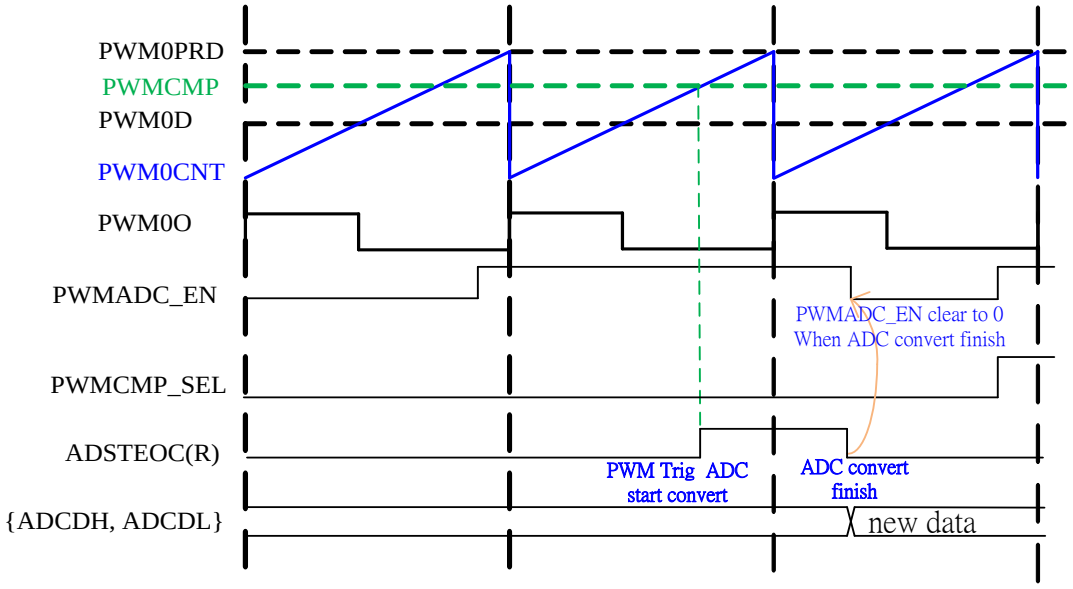
◇ Example:

```
        MOVLW   xxxx0011b       ; ADC2 (PA2) as ADC input
        MOVWX   PAMOD32

        MOVLW   00000100b       ; ADCKS = Fsys/16, ADC clock = 1 MHz
        MOVWX   ADCTL

        MOVLW   00000010b       ; ADC reference voltage select VCC
        MOVWX   ADCTL2          ; ADC input channel select ADC2

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

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

        MOVXW   ADCDH           ; Read ADC output data bit 11~4
        MOVXW   ADCTL           ; Read ADC output data bit 3~0
        …
```

6.5.2 PWM Trigger ADC

The PWM trigger ADC mechanism can be implemented by setting the PWMCMP_CTL and PWMCMPH and PWMCMPL registers. When PWMADC_EN is set to 1, the ADC conversion is started when the internal PWM counter reaches the PWMCMP{PWMCMPH, PWMCMPL} setting value. When the ADC conversion is completed, PWMADC_EN is automatically cleared to 0. PWMCMP_SEL selects the internal counter of PWM0 or PWM1 to compare with PWMCMP. When PWMCMP_SEL = 0, the 16-bit PWM0 counter is compared with PWMCMP. When PWMCMP_SEL = 1, the 8-bit PWM1 counter is compared with PWMCMPL.

Note: When PWMADC_EN=1, PWMxCKS (x=0,1) is disabled and FIRC*2 (32MHz) is selected.



PWM Trig ADC convert

Example:

[CPU running at FAST mode , Fsys = FIRC 16 MHz]
ADC clock frequency = 1 MHz, ADC channel = ADC2 (PA2).

◇ Example:

```
MOVLW   xxxx0011b       ; ADC2 (PA2) as ADC input
MOVWX   PAMOD32
MOVLW   00000100b       ; ADCKS = Fsys/16, ADC clock = 1 MHz
MOVWX   ADCTL
MOVLW   00000010b       ; ADC reference voltage select VCC
MOVWX   ADCTL2          ; ADC input channel select ADC2
MOVLW   02h
MOVWX   PWM0PRDL
MOVLW   01h
MOVWX   PWM0PRDH        ; set PWM0PRD=102h
MOVLW   50h
MOVWX   PWM0DL
MOVLW   00h
MOVWX   PWM0DH          ; set PWM0DUTY = 50h
MOVLW   40h
MOVWX   PWMCMPL
MOVLW   00h
MOVWX   PWMCMPH         ; set PWMCMP = 40h
MOVLW   0000000B
MOVWX   PWMCTL          ; set PWM0CLK = CPUCLK(Fsys)
BCX     PWMCMP_SEL      ; select  PWM0 to Trig ADC
BSX     PWM0EN          ; enable  PWM0 running

BSX     PWMADC_EN       ; set 9Dh.0 , enable PWM trigger ADC conversion .
```

```
        NOP
WAIT_PWMADC:
        BTXSC   PWMADC_EN       ; Wait ADC conversion finish.
        LGOTO   WAIT_PWMADC

        MOVXW   ADCDH           ; Read ADC output data bit 11~4
        MOVXW   ADCTL           ; Read ADC output data bit 3~0
        …
```

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

0Bh.7 **ADCIE:** ADC 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	–	TM1IF	TM0IF	WKTIF	–	INT1IF	INT0IF
R/W	R/W	–	R/W	R/W	R/W	–	R/W	R/W
Reset	0	–	0	0	0	–	0	0

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

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

17h.7~0 **ADCDH:** ADC output data bit 11~4

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

18h.7~4 **ADCDL:** ADC output data bit 3~0
18h.3 **ADSTEOC(W):** ADC start bit.
0: H/W clear after end of conversion
1: ADC start conversion (Don't write ADSTEOC when PWM trigger ADC mode is utilized)
ADSTEOC(R): ADC conversion or end of conversion bit.
0: ADC Idle (H/W clear after end of conversion)
1: ADC is in 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/2

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

19h.7~5 **ADVREFS:** ADC reference voltage and V_{BG} output voltage select
000: ADC reference voltage is V_{CC}.
001: ADC reference voltage is 1.183V.
01x: ADC reference voltage is 2.53V.
10x: ADC reference voltage is 3.0V.
11x: ADC reference voltage is 2.00V.
19h.4~0 **ADCHS:** ADC channel select
00000: ADC0 (PA0) 01000: ADC8 (PB1) 10000: ADC16 (PD1)
00001: ADC1 (PA1) 01001: ADC9 (PB2) 10001: ADC17 (PB3)
00010: ADC2 (PA2) 01010: ADC10 (PB4) 10010: 1/4 VCC
00011: ADC3 (PA3) 01011: ADC11 (PB5) 10011: ADC19 (PD2)

00100: ADC4 (PA4) 01100: ADC12 (PB6) 10100: ADC20 (PD3)
00101: ADC5 (PA5) 01101: ADC13 (PB7) 10101: ADC21 (PD4)
00110: ADC6 (PA6) 01110: V_{BG} 10110: ADC22 (PD5)
00111: ADC7 (PB0) 01111: ADC15 (PD0) 10111: ADC23 (PA7)
others: Reserved

9Dh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWMCMP_CTL	–	–	–	–	–	–	PWMCMP_SEL	PWMADC_EN
R/W	–	–	–	–	–	–	R/W	R/W
Reset	–	–	–	–	–	–	0	0

9Dh.1 **PWMCMP_SEL:** PWM trigger ADC source selection
0: Select PWM0 to trigger ADC conversion, use PWM0 internal 16-bit counter to compare PWMCMP
1: Select PWM1 to trigger ADC conversion, use PWM1 internal 8-bit counter to compare PWMCMPL

9Dh.0 **PWMADC_EN:** PWM triggers ADC conversion to enable (PWMADC_EN will automatically clear to 0 when ADC conversion is completed)
0: disable
1: enable

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

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

9Eh.7~0 **PWMCMPH:** PWMCMP[15:8] High byte value
9Fh.7~0 **PWMCMPL:** PWMCMP[7:0] Low byte value
write sequence: PWMCMPL then PWMCMPH
read sequence: PWMCMPH then PWMCMPL

6.6 Comparator

There is a Comparator (CMP) in this device.

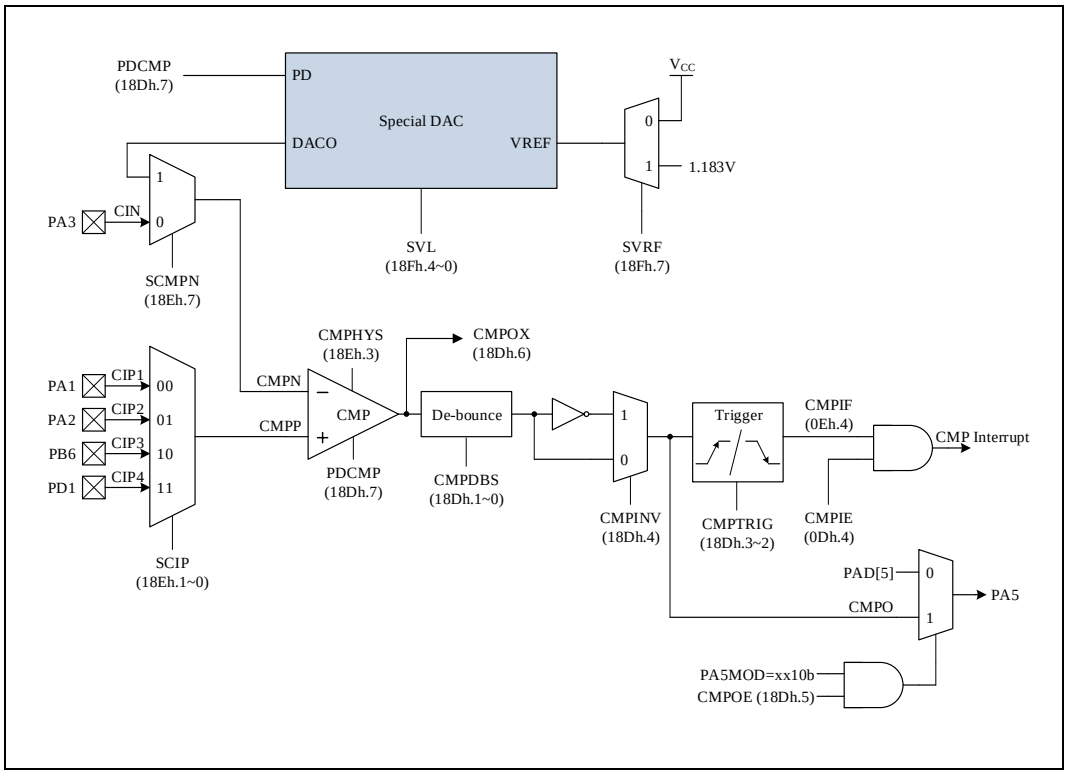
The CMP built in a 5-bit Special DAC module, which output can be accessed to negative input port of the CMP.

Reference Voltage of DAC can be selected as V_{CC} or 1.183V by setting SVRF(18Fh.7). A suitable level of voltage can be selected for proper operation of user application by setting SVL (18Fh.4~0), which will change the resistance to transform the value of voltage. Setting the PDCMP=1 (18Dh.7) will let DAC and CMP enter power down mode. By configuring SCMPN(18Eh.7), negative port input source will be external pin input or DAC output. And positive port input source is external pin input. The SCIP(18Eh.1~0) register determine positive port external input source.

Because the input module of the CMP is composed of PMOS, the input voltage range will be affected by Vth of the PMOS. Thus, the maximum input voltage of the CMP will be (V_{CC}-0.5) V. Meanwhile, the Comparator's hysteresis voltage is V_{HYS_CMP} which could be disabled by clearing CMPHYS(18Eh.3). Please refer to V_{HYS_CMP} in the table of "Comparator Characteristics" (Left click the link to go to that page). It is suggested to enable the hysteresis function of the comparator.

The Comparator original output (CMPOX) can be read by CMPOX (18Dh.6) bit. The Chip provides a de-bounce module to de-bounce the CMPOX signal, user can select de-bounce time by CMPDBS (18Dh.1~0). The de-bounce output signal can select invert or not by CMPINV (18Dh.4) to generate CMPO signal. The CMPO can be output to pin (PA5) by set CMPOE (18Dh.5) and the PA5MOD should be set to xx10b.

The CMPO is also a trigger source for the interrupt trigger module to generate interrupt flag CMPIF (0Eh.4). The trigger mode is selected by CMPTRIG (18Dh.3~2). When Comparator power down, the interrupt flag will still be produced. Therefore, it is necessary to clear the interrupt flag first after turning on the CMP module each time to prevent using the fake flag.



Comparator Block Diagram

0Dh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTIE1	CFCIE	PCIE	–	CMPIE	–	PWM1IE	PWM0IE	LVDIE
R/W	R/W	R/W	–	R/W	–	R/W	R/W	R/W
Reset	0	0	0	0	–	0	0	0

0Dh.4 **CMPIE:** Comparator interrupt enable
0: disable
1: enable

0Eh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTIF1	CFCIF	PCIF	–	CMPIF	–	PWM1IF	PWM0IF	LVDIF
R/W	R/W	R/W	–	R/W	–	R/W	R/W	R/W
Reset	0	0	–	0	–	0	0	0

0Eh.4 **CMPIF:** Comparator interrupt event pending flag
This bit is set by H/W while CMPO match trigger condition, write EFh to INTIF1 will clear this flag

18Dh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
CMPCTL	PDCMP	CMPOX	CMPOE	CMPINV	CMPTRIG		CMPDBS	
R/W	R/W	R	R/W	R/W	R/W		R/W	
Reset	1	1	0	0	0	0	0	0

18Dh.7 **PDCMP:** Comparator & DAC power down enable control
0: disable Comparator & DAC power down
1: enable Comparator & DAC power down
18Dh.6 **CMPOX:** Comparator original output (CMPOX) status
0: $V_{CMPP} < V_{CMPN}$
1: $V_{CMPP} > V_{CMPN}$ or PDCMP =1
18Dh.5 **CMPOE:** Comparator output (CMPO) signal output to PA5
0: disable
1: enable, PA5MOD should be set to xx10b
18Dh.4 **CMPINV:** Comparator de-bounce output invert select
0: no invert
1: invert
18Dh.3~2 **CMPTRIG:** Comparator interrupt trigger mode
00: Rising edge
01: Falling edge
10: Both edge
11: High level
18Dh.1~0 **CMPDBS:** Comparator original output (CMPOX) de-bounce time
00: none
01: 4 Fsys
10: 8 Fsys
11: 16 Fsys

18Eh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
CMPPNS	SCMPN	–	–	–	CMPHYS	–	SCIP	
R/W	R/W	–	–	–	R/W	–	R/W	
Reset	1	–	–	–	0	–	1	1

18Eh.7 **SCMPN:** Comparator CMPN source select
0: Comparator CMPN source is external input (CIN)
1: Comparator CMPN source is a special DAC output
18Eh.3 **CMPHYS:** Comparator Hysteresis select
0: disable hysteresis
1: enable hysteresis (suggested use)

18Eh.1~0 **SCIP:** Comparator CMPP external input select
00: Comparator CMPP external input is CIP1 (PA1)
01: Comparator CMPP external input is CIP2 (PA2)
10: Comparator CMPP external input is CIP3 (PB6)
11: Comparator CMPP external input is CIP4 (PD1)

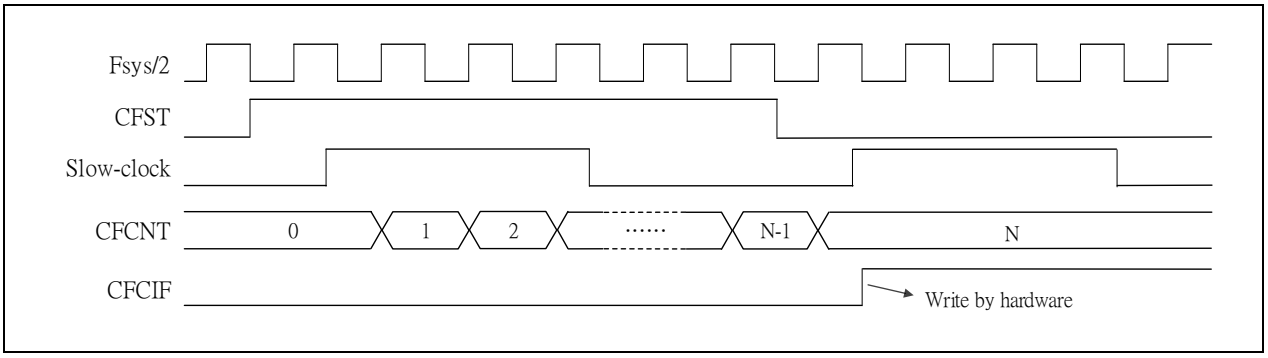
18Fh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
DACTL	SVRF	–	–	SVL				
R/W	R/W	–	–	R/W				
Reset	0	–	–	0	0	0	0	0

18Fh.7 **SVRF:** Special DAC reference voltage select
0: V_{CC}
1: 1.183V

18Fh.4~0 **SVL:** Special DAC output voltage select (reference source can be selected as V_{CC} or 1.183V)
Output voltage of special DAC = Reference voltage of special DAC * ratio
The following is the **table of ratio** when SVRF=1:
0_0000: 0.01296, 0_0001: 0.01634, 0_0010: 0.02029 , 0_0011: 0.0248
0_0100: 0.02874, 0_0101: 0.03325, 0_0110: 0.03719, 0_0111: 0.04114
0_1000: 0.04565, 0_1001: 0.05015, 0_1010: 0.05410, 0_1011: 0.05861
0_1100: 0.06255, 0_1101: 0.06706, 0_1110: 0.07101, 0_1111: 0.75402
1_0000: 0.05917, 1_0001: 0.06312, 1_0010: 0.06706, 1_0011: 0.07101
1_0100: 0.07495, 1_0101: 0.07890, 1_0110: 0.08340, 1_0111: 0.08679
1_1000: 0.09129 ,1_1001: 0.09580, 1_1010: 0.09975, 1_1011: 0.10369
1_1100: 0.10707, 1_1101: 0.11214, 1_1110: 0.11609, 1_1111: 0.76641
The **table of ratio** when SVRF=Vcc=5V is the following:
0_0000: 0.03287 , 0_0001: 0.06424 , 0_0010: 0.09545, 0_0011: 0.12667
0_0100: 0.15773, 0_0101: 0.18898, 0_0110: 0.22058, 0_0111: 0.25192
0_1000: 0.28314, 0_1001: 0.31449, 0_1010: 0.34597, 0_1011: 0.37731
0_1100: 0.40835, 0_1101: 0.43944, 0_1110: 0.47094, 0_1111: 0.50214
1_0000: 0.53348, 1_0001: 0.56443, 1_0010: 0.59538, 1_0011: 0.62687
1_0100: 0.65807, 1_0101: 0.68956, 1_0110: 0.72117, 1_0111: 0.75214
1_1000: 0.78374, 1_1001: 0.81456, 1_1010: 0.84573, 1_1011: 0.87722
1_1100: 0.90951, 1_1101: 0.94085, 1_1110: 0.97273, 1_1111: --

6.7 Capture Frequency Count(CFC)

The chip supports a mechanism to calibrate the frequency of slow-clock based on accurate fast-clock.



CFC Sketch Map

Setting CFST(116h.0) will start capture frequency counter which will count one cycle of slow-clock with Fsys/2 as the clock source. CFST(116h.0) will be cleared, counter will be stopped, and CFCIF(0Eh.7) will also be set automatically after finishing count.

The recommended Fsys is 16MHz fast-clock. Slow-clock and fast-clock must be turned-on by yourself when capture-frequency-count (CFC) is utilized. CFCIE(0Dh.7) and CFST(116h.0) will not turn-on SIRC and FIRC.

0Dh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTIE1	CFCIE	PCIE	–	CMPIE	–	PWM1IE	PWM0IE	LVDIE
R/W	R/W	R/W	–	R/W	–	R/W	R/W	R/W
Reset	0	0	0	0	–	0	0	0

0Dh.7 **CFCIE:** Capture frequency count interrupt enable
0: disable
1: enable

0Eh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTIF1	CFCIF	PCIF	–	CMPIF	–	PWM1IF	PWM0IF	LVDIF
R/W	R/W	R/W	–	R/W	–	R/W	R/W	R/W
Reset	0	0	–	0	–	0	0	0

0Eh.7 **CFCIF:** Capture frequency count interrupt event pending flag
This bit is set by H/W while capture frequency count is finished. write 7Fh to INTIF1 will clear this flag

116h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
CFCCTL	–	–	–	–	–	–	–	CFST
R/W	–	–	–	–	–	–	–	R/W
Reset	–	–	–	–	–	–	–	0

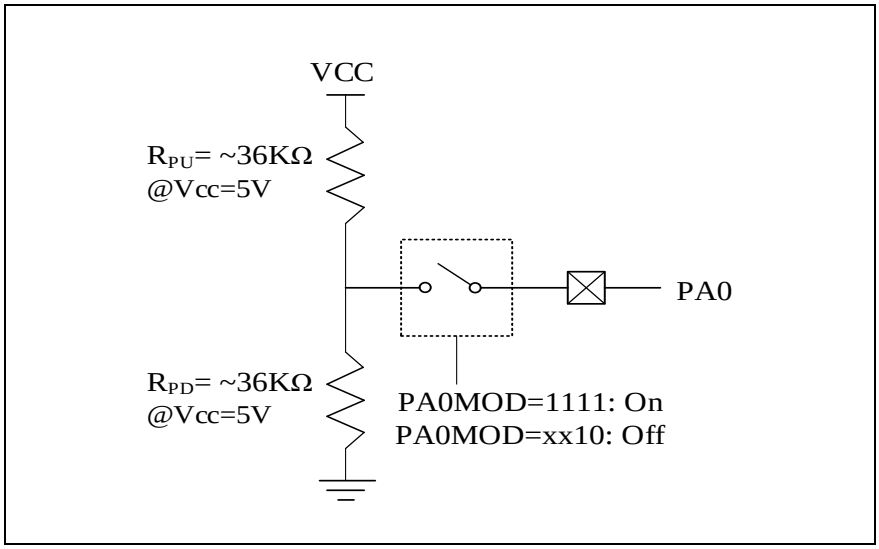
116h.0 **CFST:** Capture frequency start flag
0: Clear by hardware after finishing count
1: Start capture frequency count
Fsys=16MHz fast-clock is recommended when this function is used. If Fsys=fast-clock/2 is used, the accuracy of this function will be lower.

117h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
CFCNT	CFCNT							
R	R							
Reset	0	0	0	0	0	0	0	0

117h.7~0 **CFCNT:** Capture frequency counter w/o overflowing
When CFCNT is equal to 0xFF, it will stop count to prevent overflow.
The calculated frequency of slow-clock is (Fsys/2)/CFCNT

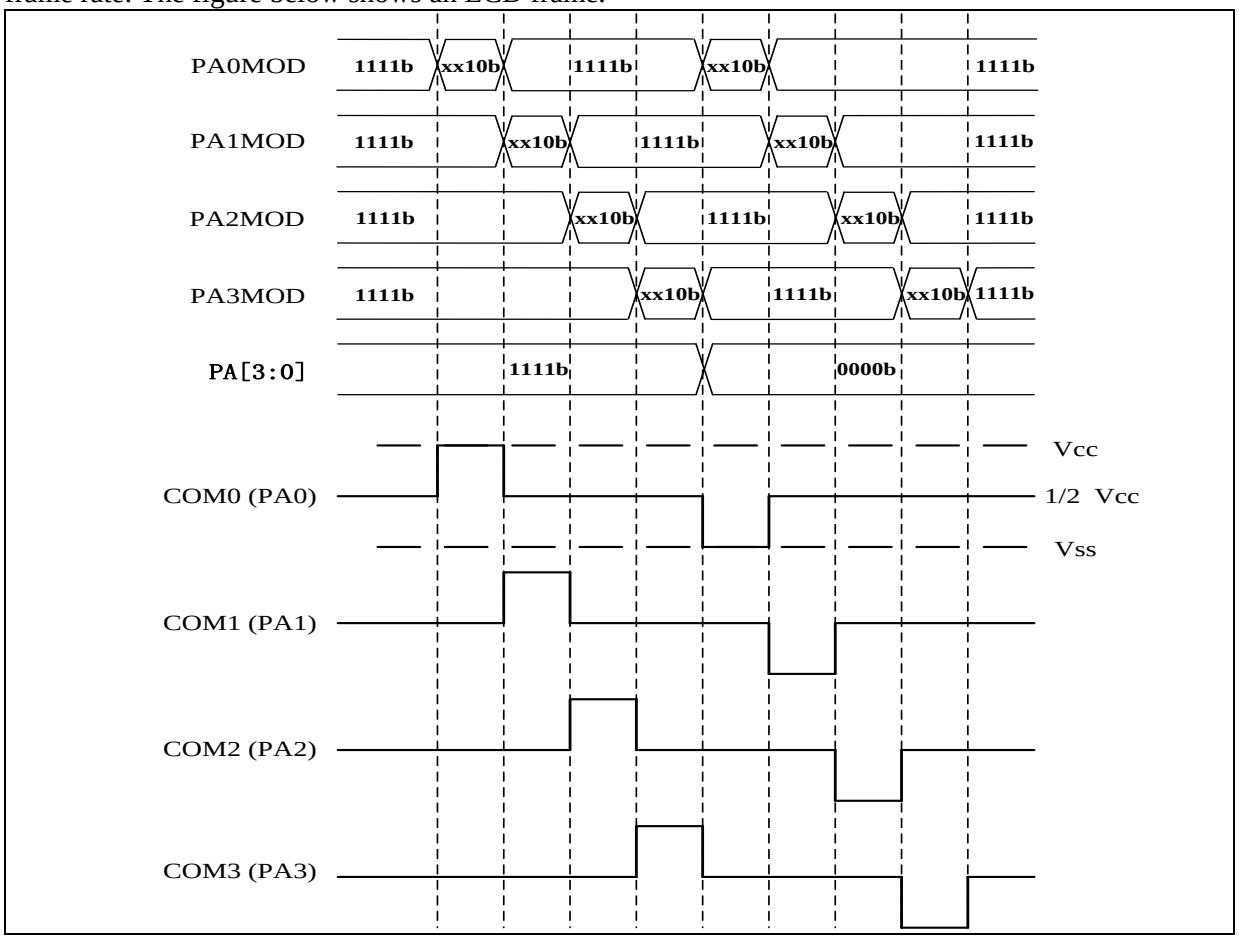
6.8 S/W Control LCD Driver

The chip support an S/W controlled method to driving LCD. Common pins are capable of driving 1/2 bias by setting the register PAxMOD/PBxMOD/PDxMOD. The COM0 circuit is shown below.

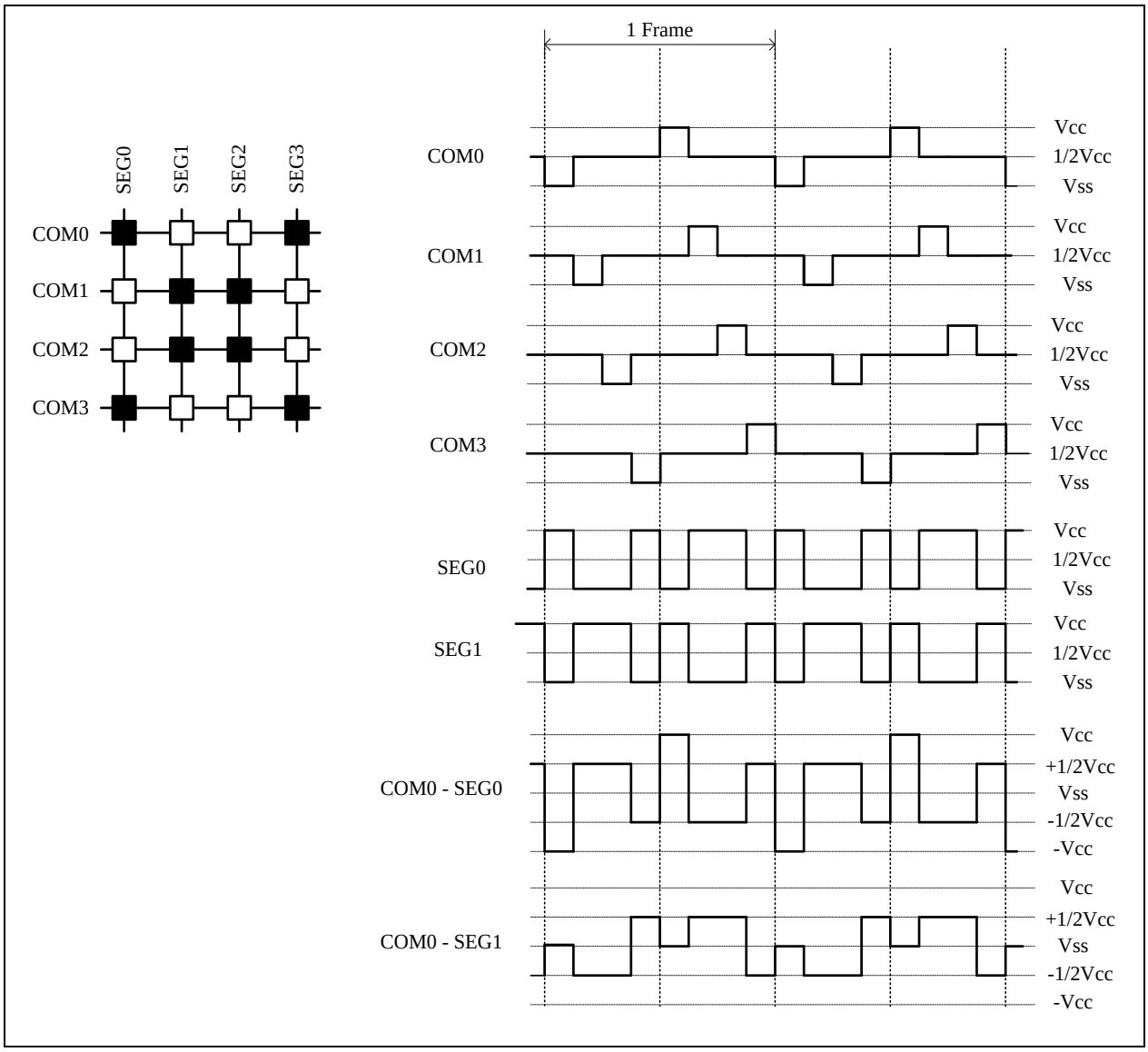


LCD COM0 Circuit

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



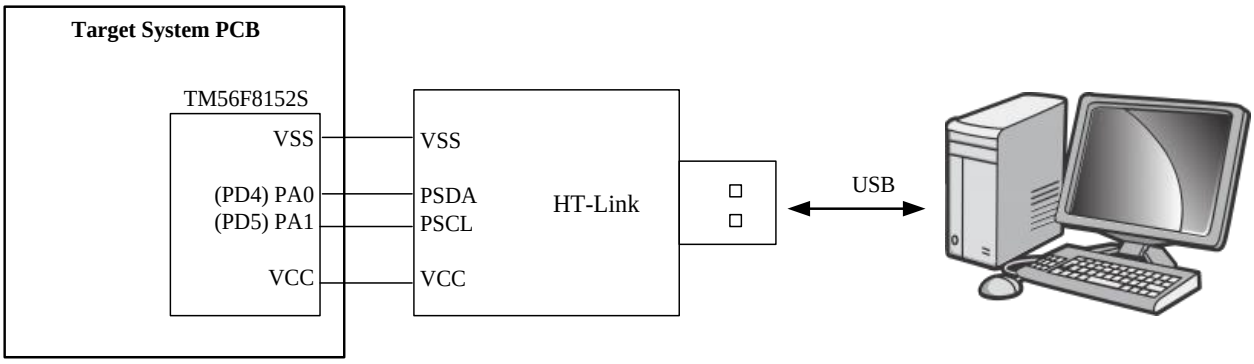
1/4 Duty, 1/2 Bias Output Waveform

6.9 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 PD4 and PD5 (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	00	Timer0 content
PCL (02h/82h/102h/182h)				**Function related to: PROGRAM COUNT**
PCL	02.7~0	R/W	00	Programming Counter data bit 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 for direct addressing
RP0	03.5	R/W	0	Register Bank Select bit 0 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
PDD (07h)				**Function related to: Port D**
PDD	07.1~0	R	-	Port D pin or "data register" state
		W	3	Port D output data register
PCLATH (0Ah/8Ah/10Ah/18Ah)				**Function related to: PROGRAM COUNT**
GPR	0A.7~4	R/W	0	General Purpose Register
PCLATH	0A.3~0	R/W	0	Write Buffer for the high byte of the Program Counter
INTIE (0Bh/8Bh/10Bh/18Bh)				**Function related to: Interrupt Enable**
ADCIE	0B.7	R/W	0	ADC interrupt enable 0: disable 1: enable
TM1IE	0B.5	R/W	0	Timer1 interrupt enable 0: disable 1: enable
TM0IE	0B.4	R/W	0	Timer0 interrupt enable 0: disable 1: enable
WKTIE	0B.3	R/W	0	Wake-up Timer interrupt enable and Wake-up Timer enable 0: disable 1: enable
INT1IE	0B.1	R/W	0	INT1 (PA1) interrupt enable 0: disable 1: enable
INT0IE	0B.0	R/W	0	INT1 (PA3) interrupt enable 0: disable 1: enable
INTIF (0Ch)				**Function related to: Interrupt Flag**
ADCIF	0C.7	R	-	ADC interrupt flag, set by H/W after ADC end of conversion
		W	0	write 7Fh to INTIF will clear this flag
TM1IF	0C.5	R	-	Timer1 interrupt event pending flag, set by H/W while Timer1 overflows
		W	0	write DFh to INTIF will clear this flag
TM0IF	0C.4	R	-	Timer0 interrupt event pending flag, set by H/W while Timer0 overflows
		W	0	write EFh to INTIF will clear this flag
WKTIF	0C.3	R	-	WKT interrupt event pending flag, set by H/W while WKT time out

Name	Address	R/W	Rst	Description
		W	0	write F7h to INTIF will clear this flag
INT1IF	0C.1	R	-	INT1 (PA1) event pending flag, set by H/W at INT1 pin’s falling/rising edge
		W	0	write FDh to INTIF will clear this flag
INT0IF	0C.0	R	-	INT0 (PA3) event pending flag, set by H/W at INT1 pin’s falling/rising edge
		W	0	write FEh to INTIF will clear this flag
INTIE1 (0Dh)				**Function related to: Interrupt Enable**
CFCIE	0D.7	R/W	0	Capture frequency count interrupt enable 0: disable 1: enable
PCIE	0D.6	R/W	0	All port pin-change wake-up interrupt enable 0: disable 1: enable
CMPIE	0D.4	R/W	0	Comparator interrupt enable 0: disable 1: enable
PWM1IE	0D.2	R/W	0	PWM1~5 interrupt enable 0: disable 1: enable
PWM0IE	0D.1	R/W	0	PWM0 interrupt enable 0: disable 1: enable
LVDIE	0D.0	R/W	0	LVD interrupt enable 0: disable 1: enable
INTIF1 (0Eh)				**Function related to: Interrupt Flag**
CFCIF	0E.7	R	-	Capture frequency count interrupt event pending flag, set by H/W while capture frequency count is finished
		W	0	write 7Fh to INTIF1 will clear this flag
PCIF	0E.6	R	-	All port pin-change wake-up interrupt event pending flag, set by H/W at all pin’s falling/rising edge. A sleep instruction is necessary before the event of pin-change otherwise pin-change event may be missed.
		W	0	write BFh to INTIF1 will clear this flag
CMPIF	0E.4	R	-	Comparator interrupt event pending flag, set by H/W while CMPO match trigger condition
		W	0	write EFh to INTIF1 will clear this flag
PWM1IF	0E.2	R	-	PWM1~5 interrupt event pending flag, set by H/W after PWM1~5 period counter roll over
		W	0	write FBh to INTIF1 will clear this flag
PWM0IF	0E.1	R	-	PWM0 interrupt event pending flag, set by H/W after PWM0 period counter roll over
		W	0	write FDh to INTIF1 will clear this flag
LVDIF	0E.0	R	-	LVD interrupt event pending flag, set by H/W while $V_{CC} < V_{LVD}$
		W	0	write FEh to INTIF1 will clear this flag
CLKCTL (0Fh)				**Function related to: Fsys**
SLOWSTP	0F.4	R/W	0	Stop Slow-clock after execute SLEEP instruction 0: Slow-clock keeps running after execute SLEEP instruction 1: Slow-clock stop running after execute SLEEP instruction
FASTSTP	0F.3	R/W	1	Stop Fast-clock 0: Fast-clock is running 1: Fast-clock stops running

Name	Address	R/W	Rst	Description
CPUCKS	0F.2	R/W	0	System clock source select 0: Slow-clock 1: Fast-clock
CPUPSC	0F.1~0	R/W	11	System clock source prescaler. System clock source 00: div 8 01: div 4 10: div 2 11: div 1
TM0RLD (10h)				**Function related to: Timer0**
TM0RLD	10.7~0	R/W	00	Timer0 reload data
TM0CTL (11h)				**Function related to: Timer0**
TM0STP	11.6	R/W	0	Stop Timer0 0: Timer0 runs 1: Timer0 stops
TM0EDG	11.5	R/W	0	TM0CKI (PA2) edge 0: rising edge 1: falling edge
TM0CKS	11.4	R/W	0	Timer0 prescaler clock source 0: Fsys/2 1: TM0CKI (PA2)
TM0PSC	11.3~0	R/W	0	Timer0 prescaler. Timer0 prescaler clock source divided by 0000: 1 0011: 8 0110: 64 0001: 2 0100: 16 0111: 128 0010: 4 0101: 32 1xxx: 256
TM1 (12h)				**Function related to: Timer1**
TM1	12.7~0	R/W	00	Timer1 content
TM1RLD (13h)				**Function related to: Timer1**
TM1RLD	13.7~0	R/W	00	Timer1 reload data
TM1CTL (14h)				**Function related to: Timer1**
TM1STP	14.6	R/W	0	Stop Timer1 0: Timer1 runs 1: Timer1 stops
TM1PSC	14.3~0	R/W	0	Timer1 prescaler. Timer1 clock source (Fsys/2) divided by 0000: 1 0011: 8 0110: 64 0001: 2 0100: 16 0111: 128 0010: 4 0101: 32 1xxx: 256
LVCTL (16h)				**Function related to: LVD/LVR**
LVDF	16.7	R	0	Low voltage detection flag 0: $V_{CC} > V_{LVD}$ 1: $V_{CC} < V_{LVD}$
LVDHYS	16.6	R/W	0	LVD Hysteresis 0: disable 1: enable
LVRSAV	16.5	R/W	1	POR/LVR auto power off in STOP/IDLE mode
LVDSAV	16.4	R/W	1	LVD auto power off in STOP/IDLE mode
LVDS	16.3~0	R/W	0	LVD voltage (V_{LVD}) select Please refer to the table of LVD voltage in the section of "LVD Circuit Characteristics". (Left click the link to go to that page)
ADCDH (17h)				**Function related to: ADC**
ADCDH	17.7~0	R	-	ADC output data bit 11~4

ADCTL (18h)				Function related to: ADC
ADCDL	18.7~4	R	-	ADC output data bit 3~0
ADSTEOC	18.3	W	0	ADC start bit. (Don't write ADSTEOC when PWM trigger ADC mode) 0: H/W clear after end of conversion 1: ADC start conversion
		R	0	0：ADC idle 1：ADC is in convertsion
ADCKS	18.2~0	R/W	0	ADC clock frequency selection. 1MHz(Typ.) 000: Fsys/256 010: Fsys/64 100: Fsys/16 110: Fsys/4 001: Fsys/128 011: Fsys/32 101: Fsys/8 111: Fsys/2
ADCTL2 (19h)				**Function related to: ADC**
ADVREFS	19.7~5	R/W	00	ADC reference voltage and V_{BG} output voltage select 000: ADC reference voltage is $V_{CC.}$ 001: ADC reference voltage is 1.183V. 01x: ADC reference voltage is 2.53V. 10x: ADC reference voltage is 3.0V. 11x: ADC reference voltage is 2.00V.
ADCHS	19.4~0	R/W	1F	ADC channel select 00000: ADC0 (PA0) 01000: ADC8 (PB1) 10000: ADC16 (PD1) 00001: ADC1 (PA1) 01001: ADC9 (PB2) 10001: ADC17 (PB3) 00010: ADC2 (PA2) 01010: ADC10 (PB4) 10010: 1/4 VCC 00011: ADC3 (PA3) 01011: ADC11 (PB5) 10011: ADC19 (PD2) 00100: ADC4 (PA4) 01100: ADC12 (PB6) 10100: ADC20 (PD3) 00101: ADC5 (PA5) 01101: ADC13 (PB7) 10101: ADC21 (PD4) 00110: ADC6 (PA6) 01110: VBG 10110: ADC22 (PD5) 00111: ADC7 (PB0) 01111: ADC15 (PD0) 10111: ADC23 (PA7) others: Reserved
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/Exit interrupt subroutine, HW auto Save/Restore WREG, FSR, TABR, PCLATH, DPL, DPH, and STATUS w/o TO, PD 0:disable 1: enable
INT0EDG	81.6	R/W	0	INT0 pin interrupt edge selection 0: falling edge to trigger 1: rising edge to trigger
INT1EDG	81.5	R/W	0	INT1 pin interrupt edge selection 0: falling edge to trigger 1: rising edge to trigger
HSINK	81.4	R/W	0	All IO port high sink current enable 0: low sink current 1: high sink current
WDTPSC	81.3~2	R/W	3	WDT period selections: 00: 175ms 01: 348ms 10: 1381ms 11: 2761ms @5V
WKTPSC	81.1~0	R/W	3	WKT period selections: 00: 86ms 01: 173ms 10: 350ms 11: 700ms @5V 00: 100ms 01: 201ms 10: 407ms 11: 813ms @3V
PAMOD10 (85h)				**Function related to: Port A**
PA1MOD	85.7~4	R/W	1	PA1 I/O mode control
PA0MOD	85.3~0	R/W	1	PA0 I/O mode control
PAMOD32 (86h)				**Function related to: Port A**
PA3MOD	86.7~4	R/W	1	PA3 I/O mode control
PA2MOD	86.3~0	R/W	1	PA2 I/O mode control

PAMOD54 (87h)				**Function related to: Port A**
PA5MOD	87.7~4	R/W	1	PA5 I/O mode control
PA4MOD	87.3~0	R/W	1	PA4 I/O mode control
PAMOD76 (88h)				**Function related to: Port A**
PA7MOD	88.7~4	R/W	0	PA7 I/O mode control PA7 has no high-sink, 1/2 bias and resistor pull-down capability.
PA6MOD	88.3~0	R/W	1	PA6 I/O mode control
PWMCTL (89h)				**Function related to: PWM0**
PWM0EN	89.7	R/W	0	PWM0 Enable 0: Disable 1: Enable
PWM1EN	89.6	R/W	0	PWM1~5 Enable 0: Disable 1: Enable
PWM0CKS	89.5~4	R/W	0	PWM0 Clock Source 00: Fsys 01: FIRC/256 10: FIRC (16 MHz) 11: FIRC*2 (32 MHz)
PWM1CKS	89.3~2	R/W	0	PWM1 Clock Source 00: Fsys 01: FIRC/256 10: FIRC (16 MHz) 11: FIRC*2 (32 MHz)
PBMOD10 (8Ch)				**Function related to: Port B**
PB1MOD	8C.7~4	R/W	1	PB1 I/O mode control
PB0MOD	8C.3~0	R/W	1	PB0 I/O mode control
PBMOD32 (8Dh)				**Function related to: Port B**
PB3MOD	8D.7~4	R/W	1	PB3 I/O mode control
PB2MOD	8D.3~0	R/W	1	PB2 I/O mode control
PBMOD54 (8Eh)				**Function related to: Port B**
PB5MOD	8E.7~4	R/W	1	PB5 I/O mode control
PB4MOD	8E.3~0	R/W	1	PB4 I/O mode control
PBMOD76 (8Fh)				**Function related to: Port B**
PB7MOD	8F.7~4	R/W	1	PB7 I/O mode control
PB6MOD	8F.3~0	R/W	1	PB6 I/O mode control
PDMOD10 (90h)				**Function related to: Port D**
PD1MOD	90.7~4	R/W	1	PD1 I/O mode control
PD0MOD	90.3~0	R/W	1	PD0 I/O mode control
PWM1PRD (91h)				**Function related to: PWM1~5**
PWM1PRD	91.7~0	R/W	FF	PWM1~5 period byte
PWM0PRDH (92h)				**Function related to: PWM0**
PWM0PRDH	92.7~0	R/W	FF	PWM0 period high byte
PWM0PRDL (93h)				**Function related to: PWM0**
PWM0PRDL	93.7~0	R/W	FF	PWM0 period low byte
PWM0DH (94h)				**Function related to: PWM0**
PWM0DH	94.7~0	R/W	80	PWM0 duty high byte
PWM0DL (95h)				**Function related to: PWM0**
PWM0DL	95.7~0	R/W	00	PWM0 duty low byte
PWM1D (96h)				**Function related to: PWM1**
PWM1D	96.7~0	R/W	80	PWM1 duty byte

PWM2D (97h)				**Function related to: PWM2**
PWM2D	97.7~0	R/W	80	PWM2 duty byte
PWM3D (98h)				**Function related to: PWM3**
PWM3D	98.7~0	R/W	80	PWM3 duty byte
PWM4D (99h)				**Function related to: PWM4**
PWM4D	99.7~0	R/W	80	PWM4 duty byte
PWM5D (9Ah)				**Function related to: PWM5**
PWM5D	9A.7~0	R/W	80	PWM5 duty byte
PDMOD32 (9Bh)				**Function related to: Port D**
PD3MOD	9B.7~4	R/W	1	PD3 I/O mode control
PD2MOD	9B.3~0	R/W	1	PD2 I/O mode control
PDMOD54 (9Ch)				**Function related to: Port D**
PD5MOD	9C.7~4	R/W	1	PD5 I/O mode control
PD4MOD	9C.3~0	R/W	1	PD4 I/O mode control
PDMCMP_CTL (9Dh)				**Function related to: Port D**
PWMCMP_SEL	9D.1	R/W	0	PWM trigger ADC select 0: select PWM0 trigger ADC convert 1: select PWM1 trigger ADC convert
PWMADC_EN	9D.0	R/W	0	PWM triggers ADC conversion to enable (When ADC conversion is completed, PWMADC_EN will automatically clear to 0) 0: disable 1: enable
PWMCMPH (9Eh)				**Function related to: PWM0/1**
PWMCMPH	91E.7~0	R/W	FF	PWMCMP High byte
PWMCMPL (9Fh)				**Function related to: PWM0/1**
PWMCMPL	9F.7~0	R/W	FF	PWMCMP low byte
User Data Memory				
RAM	A0~EF	R/W	-	RAM Bank1 area (80 Bytes)
PINMOD (105h)				**Function related to: IO Port**
IODR	105.7~6	R/W	2	All IO port drive current settings 00: 4mA, 01: 8mA, 10: 12mA (default) , 11: 18mA
TCOE	105.5	R/W	0	TCO(Fsys/2) allows output: 0: disabled. 1: enabled, output to PD1.
Reserved	105.1	R/W	0	must be kept at 0
Reserved	105.0	R/W	0	must be kept at 0
PWMCTL2 (107h)				**Function related to: PWM0**
PWM1PSC	107.7~5	R/W	0	PWM1~5 Prescaler 000: DIV1 001: DIV2 010: DIV4 011: DIV8 100: DIV16 101: DIV32 110: DIV64 111: DIV128
PWM5INV	107.4	R/W	0	PWM5 Inversion Selection 0: Disable Inversion 1: Enable Inversion
PWM4INV	107.3	R/W	0	PWM4 Inversion Selection 0: Disable Inversion 1: Enable Inversion
PWM3INV	107.2	R/W	0	PWM3 Inversion Selection 0: Disable Inversion 1: Enable Inversion

PWM2INV	107.1	R/W	0	PWM2 Inversion Selection 0: Disable Inversion 1: Enable Inversion
PWM1INV	107.0	R/W	0	PWM1 Inversion Selection 0: Disable Inversion 1: Enable Inversion
LVRPD (109h)				**Function related to: LVR/POR**
LVRPD	109.7~0	W	0	Write 37h to force LVR+POR Disable Write 38h to force LVR Disable, POR still enable Write 39h to force POR Disable, LVR still enable Write others LVR and POR enable
PORPDF	109.1	R	0	POR force power down flag 0: POR enable 1: POR is forced power down
LVRPDF	109.0	R	0	LVR force power down flag 0: LVR enable 1: LVR is forced power down
PCH (10Ch)				**Function related to: PCH**
PCH	10C.7~0	W	00	Programming Counter high byte source 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 After reset, the PCH_S is cleared
PCH	10C.3~0	R	0	Program Counter data bit 11~8
BGTRIM (10Eh)				**Function related to: Bandgap**
BGTRIM	10E.4~0	R/W	CFG	VBG trim value
IRCF (10Fh)				**Function related to: Internal RC**
IRCF	10F.6~0	R/W	CFG	FIRC trim value
CFCCTL (116h)				**Function related to: Capture Frequency Count**
CFST	116.0	R/W	0	Capture frequency start flag 0: Clear by hardware after finishing count 1: Start capture frequency count Fsys=16MHz fast-clock is recommended when this function is used. If Fsys=fast-clock/2 is used, the accuracy of this function will be lower.
CFCNT (117h)				**Function related to: Capture Frequency Count**
CFCNT	117.7~0	R	0	Capture frequency counter w/o overflowing When CFCNT is equal to 0xFF, it will stop count to prevent overflow. The calculated frequency of slow-clock is (Fsys/2)/CFCNT
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	00	TBL Data Pointer bit 7~0
DPH (186h)				**Function related to: Table Read**
DPH	186.3~0	R/W	00	TBL Data Pointer bit 11~8

TABR (18Ch)				Function related to: Table Read
TABR	18C.7~0	R/W	0	1. TABR write 01h = instruction TABRL (Read PROM low byte data to W and TABR) 2. TABR write 02h = instruction TABRH (Read PROM high byte data to W and TABR) 3. Don't write the value other than 01h or 02h into register TABR 4. After step.1 or step.2, read TABR to get main ROM table read value for C language *Table Read for ASM: Support instruction TABRL / TABRH or register TABR. Suggest not using the method of register TABR. SFR HWAUTO=1 is also suggested.* *Table Read for C: using register TABR. Only be used outside or inside the interrupt service routine. Don't utilize it inside and outside interrupt service routine simultaneously. Otherwise, something will be wrong.*
CMPCTL (18Dh)				**Function related to: Comparator**
PDCMP	18D.7	R/W	1	Comparator & DAC power down enable control 0: disable Comparator & DAC power down 1: enable Comparator & DAC power down
CMPOX	18D.6	R	1	Comparator original output (CMPOX) status 0: $V_{CMPP} < V_{CMPN}$ 1: $V_{CMPP} > V_{CMPN}$ or PDCMP =1
CMPOE	18D.5	R/W	0	Comparator output (CMPO) signal output to PA5 0: disable 1: enable, PA5MOD should be set to xx10b
CMPINV	18D.4	R/W	0	Comparator de-bounce output invert select 0: no invert 1: invert
CMPTRIG	18D.3~2	R/W	0	Comparator interrupt trigger mode 00: Rising edge 01: Falling edge 10: Both edge 11: High level
CMPDBS	18D.1~0	R/W	0	Comparator original output (CMPOX) de-bounce time 00: none 01: 4 Fsys 10: 8 Fsys 11: 16 Fsys
CMPPNS (18Eh)				**Function related to: Comparator/DAC**
SCMPN	18E.7	R/W	1	Comparator CMPN source select 0: Comparator CMPN source is external input (CIN) 1: Comparator CMPN source is a special DAC output
CMPHYS	18E.3	R/W	0	Comparator Hysteresis select 0: disable hysteresis 1: enable hysteresis (suggested use)
SCIP	18E.1~0	R/W	3	Comparator CMPP external input select 00: Comparator CMPP external input is CIP1 (PA1) 01: Comparator CMPP external input is CIP2 (PA2) 10: Comparator CMPP external input is CIP3 (PB6) 11: Comparator CMPP external input is CIP4 (PD1)

DACTL (18Fh)				Function related to: DAC/Comparator
SVRF	18F.7	R/W	0	Special DAC reference voltage select 0: V_{CC} 1: 1.183V
SVL	18F.4~0	R/W	0	Special DAC output voltage select (reference source can be selected as V_{CC} or 1.183V) Output voltage of special DAC = Reference voltage of special DAC * ratio The following is the **table of ratio** when SVRF=1: 0_0000: 0.01296, 0_0001: 0.01634, 0_0010: 0.02029 , 0_0011: 0.0248 0_0100: 0.02874, 0_0101: 0.03325, 0_0110: 0.03719, 0_0111: 0.04114 0_1000: 0.04565, 0_1001: 0.05015, 0_1010: 0.05410, 0_1011: 0.05861 0_1100: 0.06255, 0_1101: 0.06706, 0_1110: 0.07101, 0_1111: 0.75402 1_0000: 0.05917, 1_0001: 0.06312, 1_0010: 0.06706, 1_0011: 0.07101 1_0100: 0.07495, 1_0101: 0.07890, 1_0110: 0.08340, 1_0111: 0.08679 1_1000: 0.09129 ,1_1001: 0.09580, 1_1010: 0.09975, 1_1011: 0.10369 1_1100: 0.10707, 1_1101: 0.11214, 1_1110: 0.11609, 1_1111: 0.76641 The **table of ratio** when SVRF=Vcc=5V is the following: 0_0000: 0.03287 , 0_0001: 0.06424 , 0_0010: 0.09545, 0_0011: 0.12667 0_0100: 0.15773, 0_0101: 0.18898, 0_0110: 0.22058, 0_0111: 0.25192 0_1000: 0.28314, 0_1001: 0.31449, 0_1010: 0.34597, 0_1011: 0.37731 0_1100: 0.40835, 0_1101: 0.43944, 0_1110: 0.47094, 0_1111: 0.50214 1_0000: 0.53348, 1_0001: 0.56443, 1_0010: 0.59538, 1_0011: 0.62687 1_0100: 0.65807, 1_0101: 0.68956, 1_0110: 0.72117, 1_0111: 0.75214 1_1000: 0.78374, 1_1001: 0.81456, 1_1010: 0.84573, 1_1011: 0.87722 1_1100: 0.90951, 1_1101: 0.94085, 1_1110: 0.97273, 1_1111: --
INFOEN (191h)				**Function related to: Security Key Area**
INFOEN	191.7~0	R/W	0	Security Key Area Access Enable Writing 0xF0 to this register enables Security Key Area access Writing other value to this register disables Security Key Area access
User Data Memory				
RAM	1A0~1EF	R/W	-	RAM Bank3 area (80 Bytes)

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
ADDWXC	f, d	ff01 0111 dfff ffff	1	C, DC, Z	W 和 "f" 和 C 相加
SUBWXB	f, d	ff01 0100 dfff ffff	1	C, DC, Z	"f"减掉 W 及 $\overline{C}$
SUBXW	f, d	ff01 0010 dfff ffff	1	C, DC, Z	W 减掉"f"
SUBXWB	f, d	ff01 0011 dfff ffff	1	C, DC, Z	W 减掉"f"及 $\overline{C}$
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

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 : 000h ~ 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 : 000h ~ 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 : 000h ~ 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 : 000h ~ 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 : 000h ~ 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 : 000h ~ 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

CLRX	Clear "f"	
Syntax	CLRX f	
Operands	f : 000h ~ 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 : 000h ~ 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 : 000h ~ 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 : 000h ~ 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, “LGOTO LOOP” is replace with NOP if CNT ≠0, “LGOTO LOOP” will be executed

INCX	**Increment "f"**	
Syntax	INCX f [,d]	
Operands	f : 000h ~ 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 : 000h ~ 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, “LGOTO LOOP” is replace with NOP if CNT ≠0, “LGOTO LOOP” will be executed

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	IORWX f [,d]	
Operands	f : 000h ~ 1FFh, d : 0, 1	
Operation	(destination) ← (W) OR (f)	
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

LCALL	Call subroutine "k"	
Syntax	LCALL k	
Operands	k : 0000h ~ 1FFFh	
Operation	Operation: TOS ← (PC) + 1, PC.12~0 ← k	
Status Affected	-	
OP-Code	kk10 0kkk kkkk kkkk	
Description	LCALL Subroutine. First, return address (PC+1) is pushed onto the stack. The 13-bit immediate address is loaded into PC bits <12:0>. LCALL is a two-cycle instruction.	
Cycle	2	
Example	LABEL1 LCALL SUB1	B : PC =LABEL1 A : PC =SUB1, TOS =LABEL1 + 1

LGOTO	Unconditional Branch	
Syntax	LGOTO k	
Operands	k : 0000h ~ 1FFFh	
Operation	PC.12~0 ← 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>. LGOTO is a two-cycle instruction.	
Cycle	2	
Example	LABEL1 LGOTO SUB1	B : PC =LABEL1 A : PC =SUB1

MOVX	**Move f**	
Syntax	MOVX f [,d]	
Operands	f : 000h ~ 1FFh, d : 0, 1	
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 : 000h ~ 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

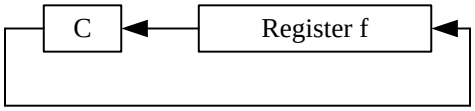
MOVWX	**Move W to "f"**	
Syntax	MOVWX f	
Operands	f : 000h ~ 1FFh	
Operation	(f) ← (W)	
Status Affected	-	
OP-Code	ff00 0000 1fff ffff	
Description	Move data from W register to register 'f'.	
Cycle	1	
Example	MOVWX REG1	B : REG1 =0xFF, W =0x4F A : REG1 =0x4F, W =0x4F

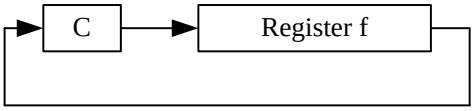
NOP	**No Operation**	
Syntax	NOP	
Operands	-	
Operation	No Operation	
Status Affected	-	
OP-Code	0000 0000 0000 0000	
Description	No Operation	
Cycle	1	
Example	NOP	-

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 B : W =0x07 : A : W =value of k8 TABLE ADDWX PCL, 1 RETLW k1 RETLW k2 : RETLW kn

RLX	**Rotate Left "f" through Carry**
Syntax	RLX f [,d]
Operands	f : 000h ~ 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 : 000h ~ 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 0x15 — B : W =0x25 A : W =0xF0

SUBWX	Subtract W from "f"
Syntax	SUBWX f [,d]
Operands	f : 000h ~ 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 : 000h ~ 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 REG1, 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, (TABR) ← ROM[DPTR] high byte content, Where DPTR = {DPH[max:8], DPL[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 (TAB1>>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, (TABR) ← ROM[DPTR] low byte content, Where DPTR = {DPH[max:8], DPL[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 is register MOVLW (TAB1>>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

TSTX	Test if "f" is zero	
Syntax	TSTX f	
Operands	f : 000h ~ 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 : 000h ~ 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 REG1, 1	B : REG1 =0xAF, W =0xB5 A : REG1 =0x1A, W =0xB5

ELECTRICAL CHARACTERISTICS

All of the parameters are based on the characteristics of tested samples.

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_{CC} = 5.0V, unless otherwise specified)

Parameter	Symbol	Conditions		Min.	Typ.	Max.	Unit
Operating Voltage	V_{cc}	Fsys = 16 MHz (FIRC)(RDCTL=4ns) (PWM0CKS=Fsys)(-40℃ ~ 105℃)		2.2	–	5.5	V
		Fsys = 8 MHz (FIRC/2) (RDCTL=4ns) (PWM0CKS=Fsys) (-40℃ ~ 105℃)		1.6	–	5.5	V
Input High Voltage	V_{IH}	All Input	V_{CC} = 3.0~5.0V	0.6V_{CC}	–	V_{CC}	V
Input Low Voltage	V_{IL}	All Input	V_{CC} = 3.0~5.0V	V_{SS}	–	0.2V_{CC}	V
I/O port Source Current	I_{OH}	IODR=3	V_{CC} = 5.0V, V_{OH} = 4.5V(*)		51		mA
			V_{CC} = 4.0V, V_{OH} = 2.8V(*)		36		
			V_{CC} = 5.0V, V_{OH} = 4.5V		20		
			V_{CC} = 3.0V, V_{OH} = 2.7V		8		
		IODR=2	V_{CC} = 5.0V, V_{OH} = 4.5V		15.5		
		IODR=1	V_{CC} = 5.0V, V_{OH} = 4.5V		10.5		
		IODR=0	V_{CC} = 5.0V, V_{OH} = 4.5V		5.5		
I/O port Sink Current	I_{OL}	HSINK=1	V_{CC} = 5.0V, V_{OL} = 1.5V(*)		160		mA
			V_{CC} = 4.0V, V_{OL} = 1.2V(*)		120		
			V_{CC} = 5.0V, V_{OL} = 0.5V		71		
			V_{CC} = 3.0V, V_{OL} = 0.3V		32		
		HSINK=0	V_{CC} = 5.0V, V_{OL} = 0.5V		40		mA
			V_{CC} = 3.0V, V_{OL} = 0.3V		18		
Input Leakage Current (pin high)	I_{ILH}	All Input	V_{IN} = V_{CC}	–	–	1	μA
Input Leakage Current (pin low)	I_{ILL}	All Input	V_{IN} = 0V	–	–	–1	μA

ps: (*)V_{OH}=0.7*V_{CC} , V_{OL}=0.3*V_{CC}

Parameter	Symbol	Conditions		Min.	Typ.	Max.	Unit
Power Supply Current (No Load) (ATD On)	I_{CC}	FAST mode FIRC 16 MHz	V_{CC} = 5.0V	–	4.5	–	mA
			V_{CC} = 3.0V	–	2.6	–	
		FAST mode FIRC 8 MHz	V_{CC} = 5.0V	–	3.0	–	
			V_{CC} = 3.0V	–	1.8	–	
		FAST mode FIRC 4 MHz	V_{CC} = 5.0V	–	2.0	–	
			V_{CC} = 3.0V	–	1.4	–	
		FAST mode FIRC 2 MHz	V_{CC} = 5.0V	–	1.4	–	
			V_{CC} = 3.0V	–	0.9	–	
		SLOW mode SIRC div1 FIRC STOP POR/LVR/LVD On ATD Off	V_{CC} = 5.0V	–	0.854	–	
			V_{CC} = 3.0V	–	0.636	–	
		SLOW mode SIRC div1 FIRC STOP POR/LVR/LVD On ATD On	V_{CC} = 5.0V	–	0.166	–	
			V_{CC} = 3.0V	–	0.127	–	
		SLOW mode SIRC div1 FIRC STOP POR/LVR/LVD Off ATD On	V_{CC} = 5.0V	–	0.035	–	
			V_{CC} = 3.0V	–	0.022	–	
		IDLE mode SIRC div1 POR/LVR/LVD Off	V_{CC} = 5.0V	–	8.1	–	µA
			V_{CC} = 3.0V	–	2.6	–	
		STOP mode POR/LVR Off	V_{CC} = 5.0V	–	–	1	µA
			V_{CC} = 3.0V	–	–	1	
Pull-up Resistor	R_{UP}	V_{IN} = 0 V Ports A, B, D	V_{CC} = 5.0V	–	36.5	–	KΩ
			V_{CC} = 3.0V	–	38.3	–	
Pull-down Resistor	R_{DOWN}	V_{IN} = 5 V Ports A, B, D	V_{CC} = 5.0V	–	37.0	–	
			V_{CC} = 3.0V	–	38.3	–	
POR voltage		T_A = 25℃		1.48	1.63	1.78	V

3. Clock Timing

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

Parameter	Condition		Min.	Typ.	Max.	Unit
FIRC Frequency (*)	T_A = -40℃ ~ 105℃	V_{CC} = 3.0 ~ 5.0V	-3.8%	16	+2.5%	MHz
	T_A = -40℃ ~ 105℃	V_{CC} = 4.0 V	-3.4%	16	+1%	
	T_A = 0℃ ~ 70℃	V_{CC} = 4.0 V	-2.6%	16	+0.9%	
	T_A = 25℃	V_{CC} = 3.5 ~ 5.0 V	-1.2%	16	+1%	
SIRC Frequency	T_A = 25℃	V_{CC} = 5.0 V		95.8		KHz

(*) 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} = 5.0 V ±10 %	–	21	–	µs
WDT time	V_{CC} = 5.0 V, WDTPSC = 11b	–	2761	–	ms
WKT time	V_{CC} = 5.0 V, WKTPSC = 11b	–	700	–	ms
CPU start up time	V_{CC} = 5.0 V	–	14.8	–	ms

5. LVR Circuit Characteristics

(T_A = 25℃)

Parameter	Symbol	Condition	LVRS	Min.	Typ.	Max.	Unit
LVR Voltage	LVR_{th}	T_A = 25℃	0000	Disable			V
			0001	–	1.73	–	
			0010	–	1.85	–	
			0011	–	1.97	–	
			0100	–	2.10	–	
			0101	–	2.22	–	
			0110	–	2.35	–	
			0111	–	2.47	–	
			1000	–	2.59	–	
			1001	–	2.72	–	
			1010	–	2.84	–	
			1011	–	2.97	–	
			1100	–	3.10	–	
			1101	–	3.22	–	
			1110	–	3.33	–	
			1111	–	3.45	–	
LVR Hysteresis Window	V_{HYS_LVR}	T_A = 25℃		–	0	–	mV
Low Voltage Detection time	T_{LVR}	T_A = 25℃		100	–	–	μs

6. LVD Circuit Characteristics

(T_A = 25℃)

Parameter	Symbol	Condition	LVDS	Min.	Typ.	Max.	Unit
LVD Voltage	LVD_{th}	T_A = 25℃	0000	Disable			V
			0001	–	1.85	–	
			0010	–	2.00	–	
			0011	–	2.20	–	
			0100	–	2.35	–	
			0101	–	2.50	–	
			0110	–	2.70	–	
			0111	–	2.85	–	
			1000	–	3.00	–	
			1001	–	3.20	–	
			1010	–	3.35	–	
			1011	–	3.50	–	
			1100	–	3.70	–	
			1101	–	3.85	–	
			1110	–	4.00	–	
			1111	–	4.20	–	
LVD Hysteresis Window	V_{HYS_LVD}	LVDHYS = 0		–	10	–	mV
		LVDHYS = 1, LVDS=8		–	15	–	
Low Voltage Detection time	T_{LVD}	T_A = 25℃		100	–	–	μs

7. ADC Electrical Characteristics

(T_A = 25℃, V_{CC} = 3.0V to 5.5V, V_{SS} = 0V)

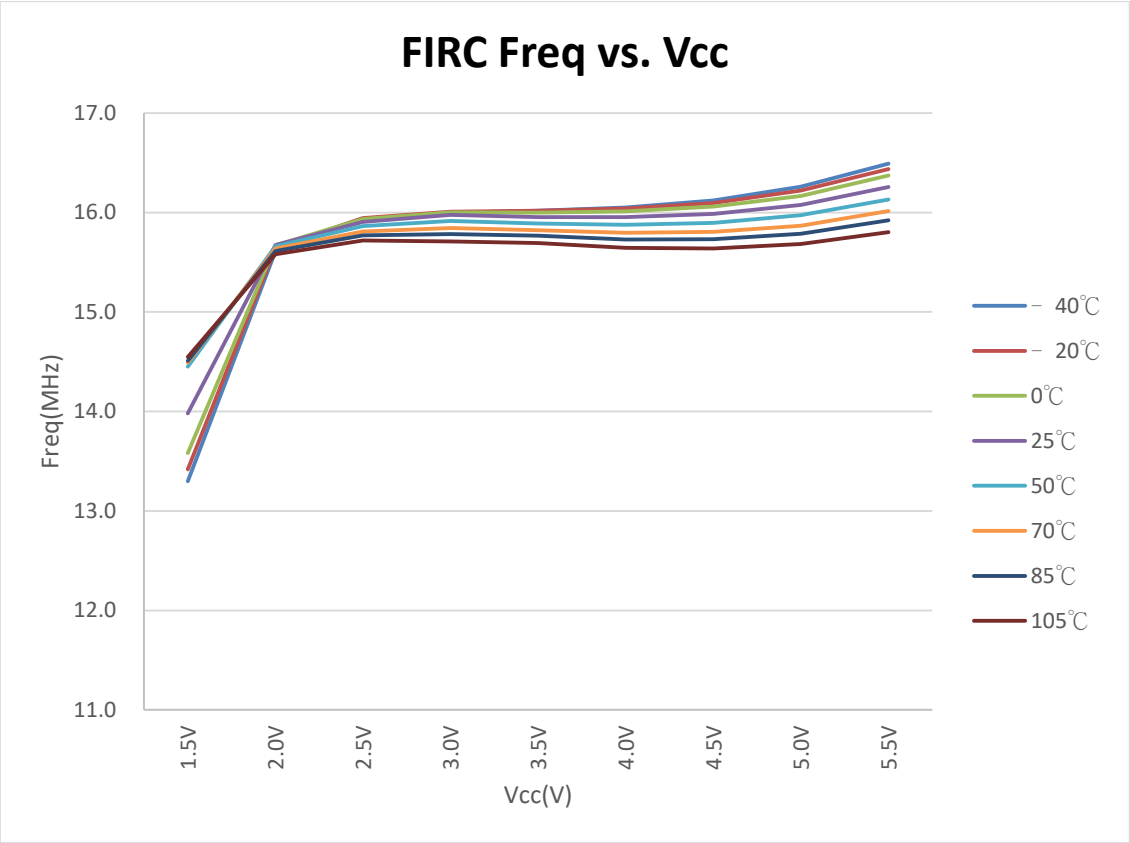
Parameter	Conditions	Min.	Typ.	Max.	Units
Total Accuracy	V_{CC} = 5.0V, V_{SS} = 0V, F_{ADC} = 1 MHz	–	±3	–	LSB
Integral Non-Linearity		–	±3.2	–	
Differential Non-Linearity		–	±1	±4	
Max Input Clock freq. (F_{ADC})	Source impedance (Rs < 5KΩ)	–	–	4	MHz
	Source impedance (Rs < 10KΩ)	–	–	2	
	Source impedance (Rs < 25KΩ)	–	–	1	
	Source is VBG (ADCHS=01110b)	–	–	F_{SYSCLK}/4	
Conversion Time	F_{ADC} = 1 MHz	–	21	–	μs
Conversion Current	Vcc=5V, ADVREFS=000b	–	0.45	–	mA
	Vcc=4V, ADVREFS=01xb	–	0.6	–	
BandGap Voltage Reference (1.183V V_{BG}) ADC reference voltage (V_{REF}) (ADVREFS=001b) (No power disturbance)	25°C, V_{CC} = 3.0V~5.0V	-1.1%	1.183	+1.2%	V
	-20°C~105°C, V_{CC} = 3.0V~5.0V	-1.5%	1.183	+1.5%	V
BandGap Voltage Reference (2.53V V_{BG}) ADC reference voltage (V_{REF}) (ADVREFS=01xb) (No power disturbance)	-20°C~105°C, V_{CC} = 3.5V~5.0V	-1.6%	2.53	+1.6%	V
BandGap Voltage Reference (2V V_{BG}) ADC reference voltage (V_{REF}) (ADVREFS=11xb) (No power disturbance)	-20°C~105°C, V_{CC} = 3.5V~5.0V	-1.5%	2	+1.5%	V
V_{CC}/4 reference voltage	25°C, V_{CC} = 3.0V~5.0V	-1%	0.25V_{CC}	+1%	V
Input Voltage	ADVREFS=00xb, 01xb, 10xb, 11xb	V_{SS}	–	V_{REF}	V

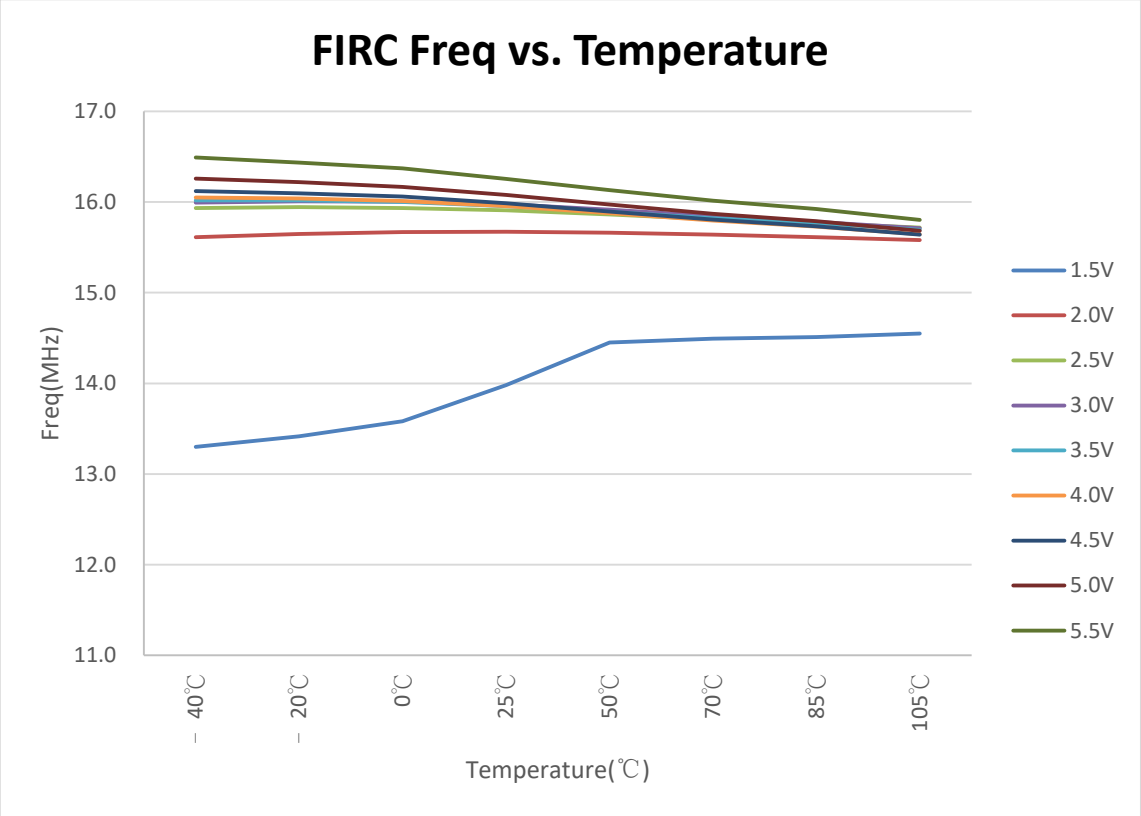
8. Comparator Characteristics

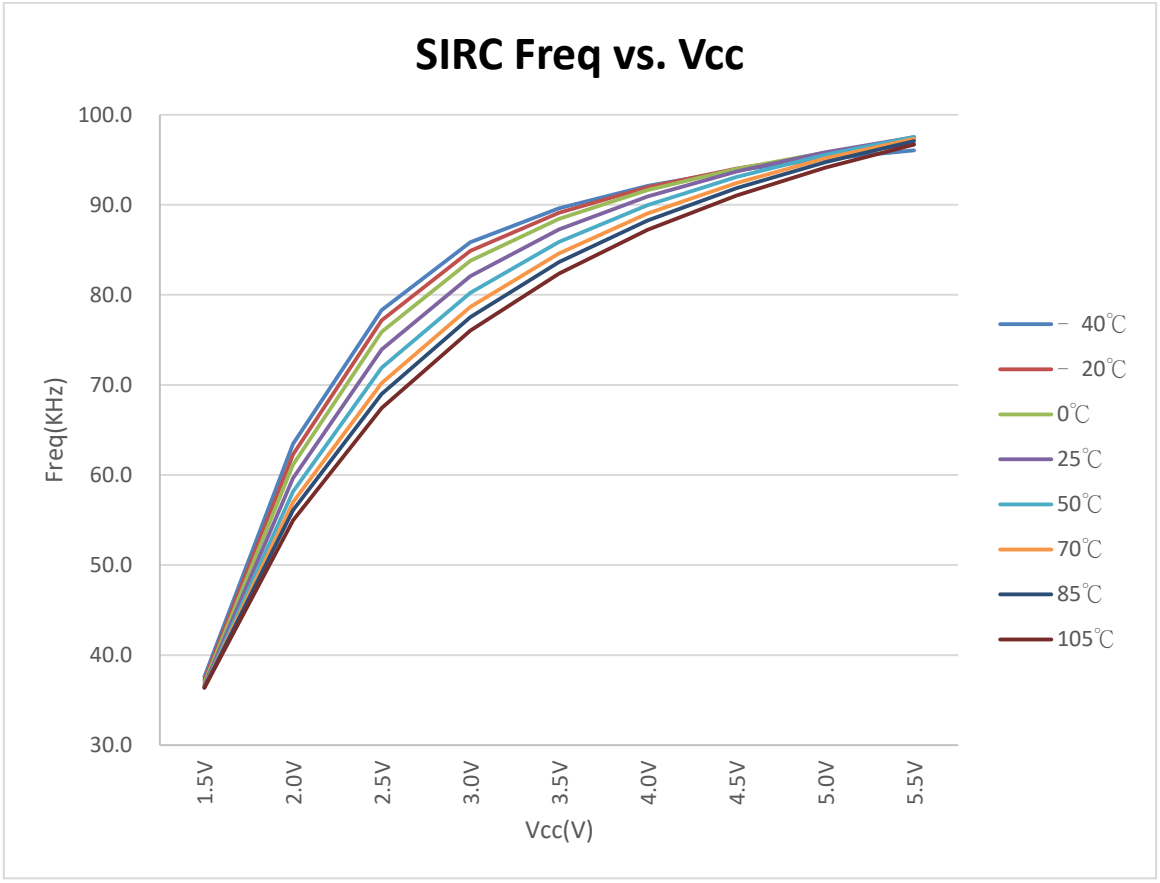
(T_A = 25℃, V_{CC} = 3.0V to 5.5V, V_{SS} = 0V)

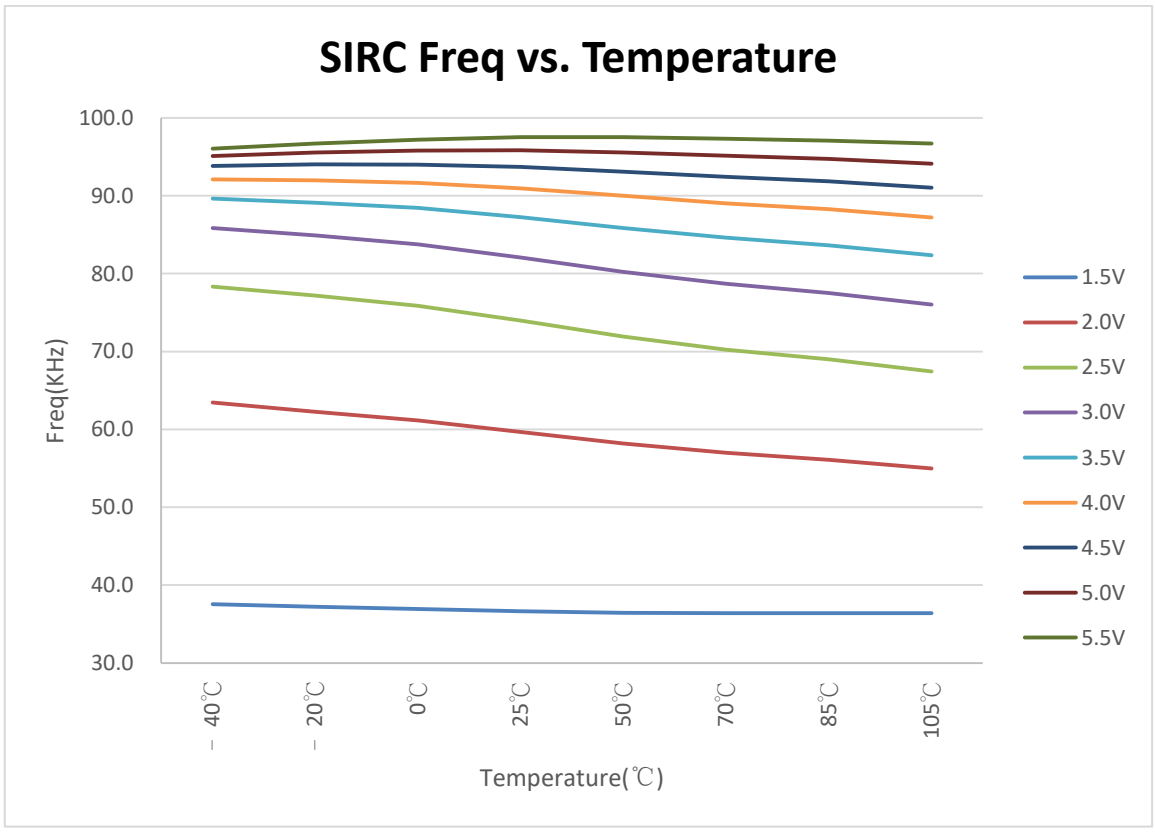
Parameter	Conditions	Min.	Typ.	Max.	Units
Power supply	–	2.2	–	5.5	V
Quiescent Current	V_{CC} = 5.0V	–	100	–	μA
DAC Current	V_{CC} = 5.0V	60	–	220	μA
V_{OS_CMP}	V_{CC} = 5.0V	-15	–	15	mV
V_{CM_CMP}	V_{CC} = 5.0V	0	–	V_{CC}-0.5	V
V_{HYS_CMP}	V_{CC} = 5.0V	-	1	-	mV

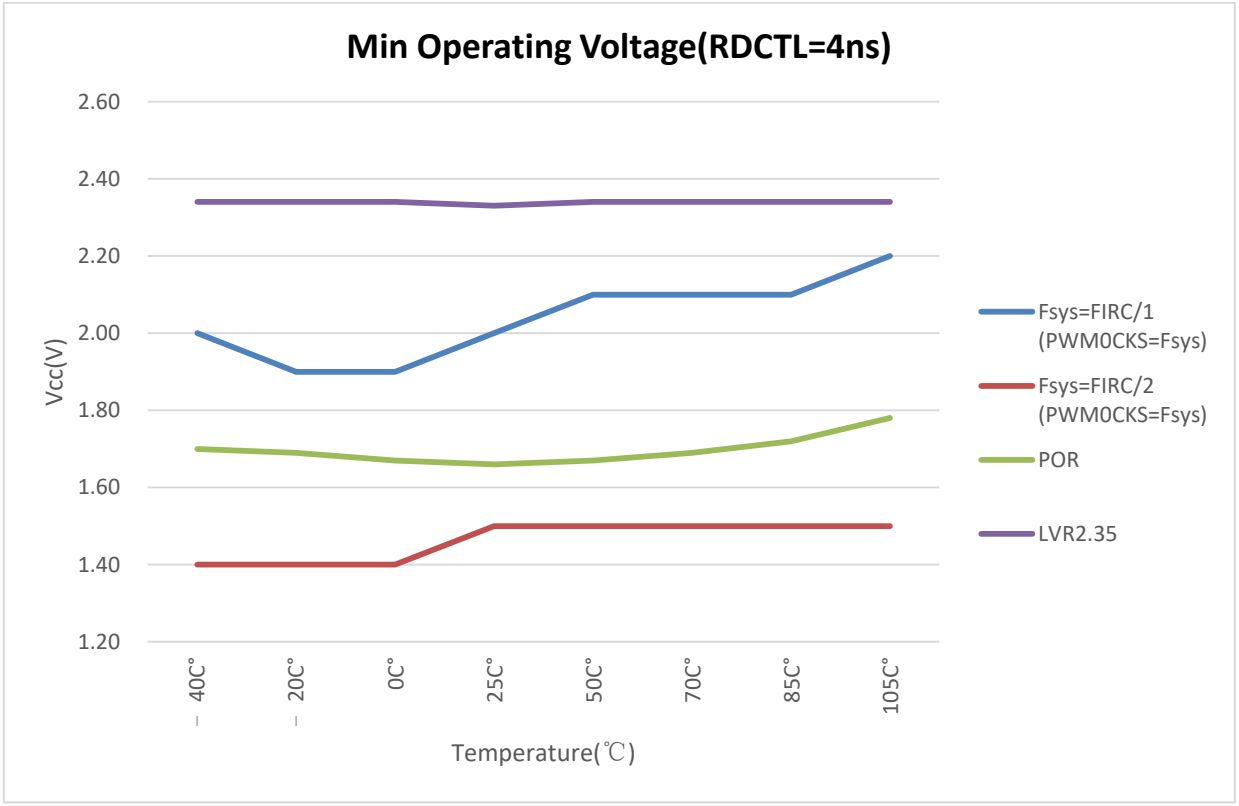
9. Characteristics Graphs



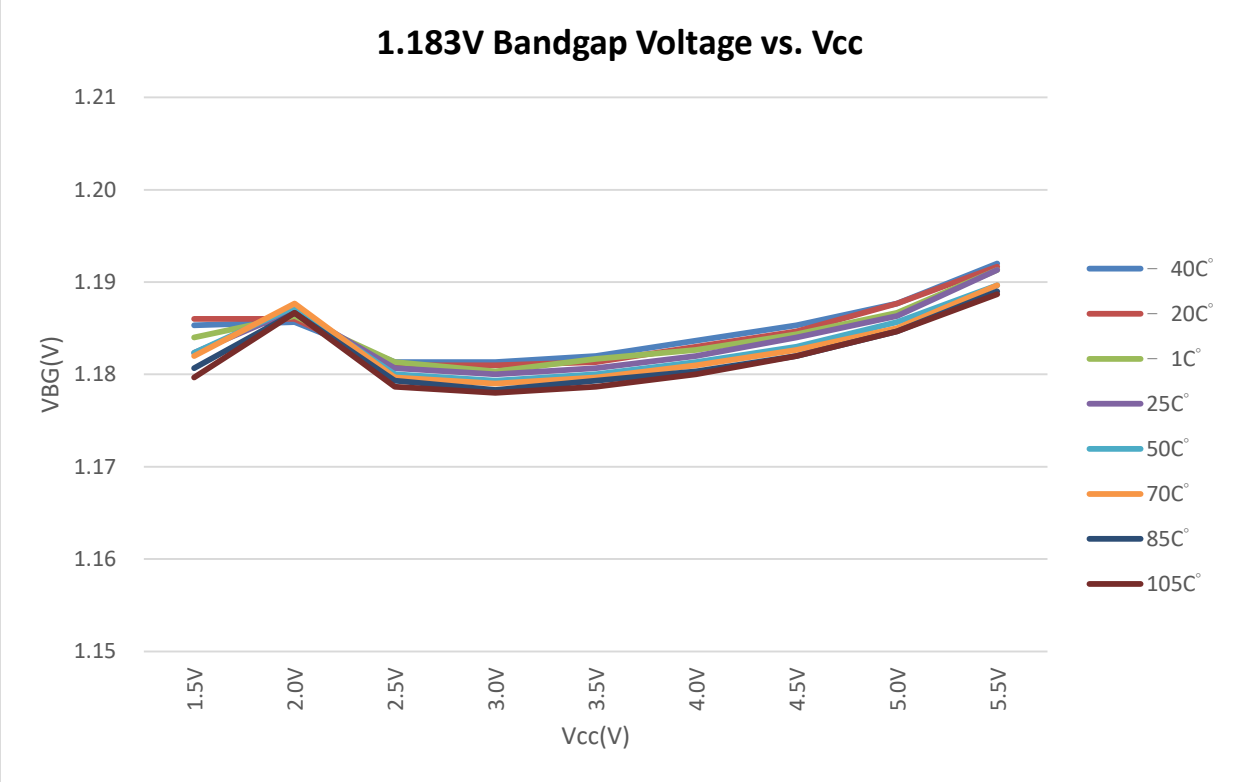



SIRC Freq vs. Vcc
100.0
90.0
80.0
70.0
60.0
50.0
40.0
30.0
Freq(KHz)
1.5V
2.0V
2.5V
3.0V
3.5V
4.0V
4.5V
5.0V
5.5V
Vcc(V)
– 40℃
– 20℃
0℃
25℃
50℃
70℃
85℃
105℃


SIRC Freq vs. Temperature
100.0
90.0
80.0
70.0
60.0
50.0
40.0
30.0
Freq(KHz)
– 40℃
– 20℃
0℃
25℃
50℃
70℃
85℃
105℃
Temperature(℃)
1.5V
2.0V
2.5V
3.0V
3.5V
4.0V
4.5V
5.0V
5.5V



Note: **The user must switch RDCTL to "4ns" to enhance the performance of minimal operating voltage**



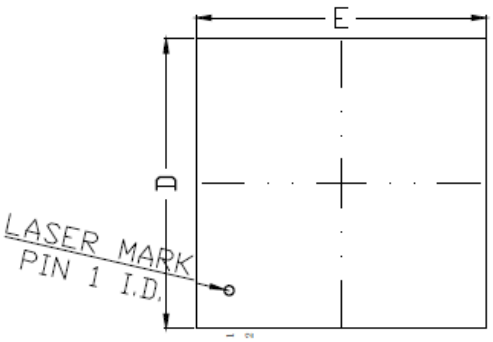
PACKAGING INFORMATION

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.

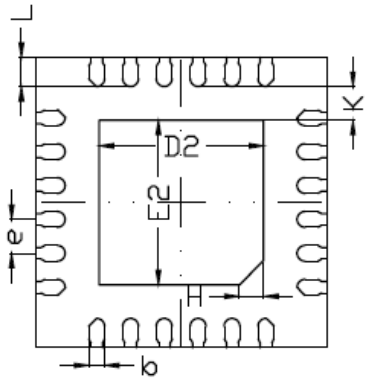
The ordering information:

Ordering number	Package
TM56F8152S-MTP-F6	QFN 24-pin (3x3x0.75-0.35 mm)

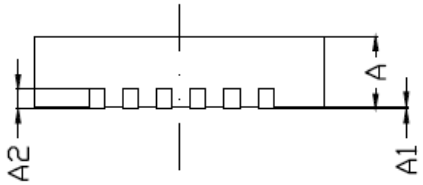
QFN 24 (3*3*0.75-0.35mm) Package Dimension



TOP VIEW



BOTTOM VIEW



SIDE VIEW

SYMBOL	MILLIMETER		
	MIN	NOM	MAX
A	0.70	0.75	0.80
A1	--	0.02	0.05
A2	0.20REF		
b	0.11	0.16	0.21
D	2.90	3.00	3.10
E	2.90	3.00	3.10
D2	1.60	1.70	1.80
E2	1.60	1.70	1.80
e	0.35BSC		
H	0.20	0.25	0.30
K	0.35REF		
L	0.25	0.30	0.35