十速

TM56M152A

DATA SHEET

Rev 1.01

AMENDMENT HISTORY

Version	Date	Description
0.90	Nov, 2023	1. Modify POR Voltage
0.91	Jan, 2024	1. Add the table of OPTION into the chapter of Interrupt 2. Modify the description about OPTION in the table of MEMORY MAP 3. Add note for unbonded pads
0.92	Jan, 2024	1. Fix typo of IORWX as "(W) OR (f)" 2. Add note for suggested RDCTL below the graph of minimal operating voltage 3. Add the table of RDCTL into the section of Program ROM (PROM) 4. Change the suggested value of RDCTL to bold font. 5. Delete the operating voltage of RDCTL=8ns in the table of DC characteristics 6. Add the condition of PWMCKS=FIRC*1 into the table of operating voltage of Fsys=8MHz
0.93	Feb, 2024	1. Add measured operating current for the condition of "ATD Off" in the table of "Power Supply Current" 2. To propose the purpose of ATD in the chapter of "FEATURES"
0.94	Feb, 2024	1. Add ADC reference voltage for ADVREFS=11b into the table of ADC Electrical Characteristics 2. To inhibit LVR1.6V and LVR1.73V becomes default 3. Delete the column of TM56M1522 in the table of "FAMILY OVERVIEW"
0.95	Mar, 2024	1. Fix typo: default value of SYSCFG is 0000_0110_0000_0000 2. Replace "CMOS Output" with "CMOS Output (except PWMx)" in I/O Pin Function Table 1~4
0.96	May, 2024	1. Delete items of wafer and dice in Ordering Information 2. Add PSDA and PSCL into the chapters of PIN ASSIGNMENT DIAGRAM, PIN DESCRIPTION and PIN SUMMARY 3. Replace "non-overlap" with "dead-zone(non-overlap)" 4. Lowering standards of Vbg and 2.48V ADC Vref
0.97	Aug, 2024	1. Add semicolon into the example of "6.4 PWM: 16 bits PWM" 2. Add comment about ATD 3. Update the link of "MOVX" 4. Modify the method to clear interrupt flag 5. Delete the rows before version 0.9 in the table of "AMENDMENT HISTORY" 6. Add comments for PORSEL and ATDOFF 7. Replace "ATDOFF=0" with "ATD On" in the table of DC Characteristics 8. Change the specification of FIRC frequency from +-1% to +-1.2% 8. Add comment for "Clock Timing": The value of this parameter is based on the characteristics of tested samples. 9. Add comment for ELECTRICAL CHARACTERISTICS: All of the parameters are based on the characteristics of tested samples. 10. Add specification of SIRC frequency 11. Replace "1/2 bias" with "LCD 1/2 bias" in the table of "PIN SUMMARY" 12. Add LVCTL into the section of "Low Voltage Reset (LVR)" and the chapter of "Interrupt" 13. Fix typo: The reset value of LVDHYS is 0
0.98	Nov, 2024	1. Add the specification of ADC conversion current 2. Add condition into "BandGap Voltage Reference" and "ADC reference voltage": No power disturbance 3. Add comment: ATD off(recommend for EFT issue) 4. Add max and min limit of 2V VBG 5. Changed the description of BG2TRIM from "exact" to "slightly exact"
0.99	Dec, 2024	1. Add the figures of "PA7 Structure" and "Constraint on PA7", and update the figure of "General Pin Structure"
1.00	Apr, 2025	1. Delete the specification of FIRC Frequency @4V/25℃ in the chapter of ELECTRICAL CHARACTERISTICS 2. Modify the description about TM0IF below Timer0 Block Diagram 3. Fix typo for the value of ADCTL2 in the example of 6.5 Analog-to-Digital Converter 4. Fix typo: Replace "General Pin Structure" with "Constraint on PA7" 5. Add a simple explanation of pin-change wakeup 6. To change "pin change", "wake up" and "wakeup" as pin-change, wake-up and wake-up respectively in somewhere of this document. 7. Replace "I^2C SCL for program" and "I^2C SDA for program" with "clock for programmer" and "data for programmer" respectively. 8. Replace "All Pin Change" with "All Pin-change Wake-up" in the system block diagram. 9. Add supplementary explanation about pin-change wake-up into the table of family overview 10. Change Wake-up as "Wake-up Interrupt" in the table of "PIN SUMMARY" 11. Change the pins of external interrupt in the table of "PIN SUMMARY". 12. Change the maximum input voltage in the table of "ADC Electrical Characteristics" as V_{REF} 13. Add comments into the parameter column of the table of "ADC Electrical Characteristics" 14. Modify Timer0/1 block diagram 15. Fix something about ADVREFS 16. Modify the package dimension of DFN10

1.01	May, 2025	1. Delete the some "MTP" characters 2. Modify the system block diagram 3. Modify the table of "Family Overview" 4. Add SOT23-8

CONTENTS

FAMILY OVERVIEW

	TM56F1552 (TK) TM56F1522 (IO)	TM56M152A
EV board	On chip debug	TM56F1552 (TK) TM56F1522 (IO)
ROM	4k x 16	2k x 16
RAM	336	128
EEPROM	128	X
CTK	V	X
SIRC	84 KHz@5V/25℃	92.8 KHz@5V/25℃
WDT	96ms, 192ms, 768ms,1536ms @5V	91ms, 183ms, 732ms, 1463 ms @5V
WKT	12ms,24ms,48ms,96ms @5V	11ms,23ms,46ms,91ms @5V
SXT, FXT	V	X
SFR.RDCTL	X	V (suggest RDCTL=4ns)
OPA	V	X
SFR.OPOF (CMPP to OPO)	OPOF=0 (POR, CMPP <= OPO) OPOF=1 (CMPP <= CIPx)	X
SFR.ADVREFS	VCC / 2.48V	VCC /1.2V / 2V / 2.48V ADVREFS=1.2V/2V, could not be emulated
SFR.BG2TRIM	X	Read BG2TRIM and Write into BGTRIM, obtain ADVREFS=2.0V
SFR.SVRF (DAC VREF)	VCC / 1.2 / 2.48V	X
CMP+DAC	V	X
SFR.IRCFT	X	X
PAD.LDOC	X	X
High Sink	75mA@5V	63mA@5V for all pins except PA7 PA7 has no high sink
IO	PA7~0 PB7~0 PD1~0	PA7~0 PB6~4, PB2~0
Pull down & 1/2 bias	V	X
POR	1.95V No PORSEL	1.63V Has PORSEL
Minimal Operating Voltage	1.9V @16MHz	2.3V @16MHz
LVR_{th}	2.05V~4.15V	1.73V~3.5V

LVD_{th}	2.2V~4.15V	1.73V~3.5V
PWM	Two outputs PWM0CLK: CPUCLK or FIRC (16MHz) or FIRC*2 (32MHz)	One output PWM0CLK: CPUCLK or FIRC/256 or FIRC (16MHz) or FIRC*2 (32MHz)
T2	V	X
XINT0~2	Two input	One input
ATD	X	V
EFTCON	X	X
All Pin-change Wake-up Interrupt	has pin-change wake-up, but no relative interrupt and flag	V

FEATURES

1. **ROM: 2K x 16 bits MTP(TM56M152A)**
2. **RAM: 128 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, 93 KHz@V_{CC}=5V)
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. **2 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**
   - Three External Interrupt pins

     - 1 pin is falling edge wake-up triggered & Interrupts

     - 2 pins are rising or falling edge wake-up triggered & Interrupt
   - Timer0 / Timer1 / Wake-up Timer Interrupt
   - ADC Interrupt
   - PWM Interrupt
   - LVD Interrupt
   - All Port Pin-change Wake-up Interrupts
9. **Wake-up Timer (WKT)**
   - Clocked by built-in RC oscillator with 4 adjustable interrupt times

     - 11 ms / 23 ms / 46 ms / 91 ms @V_{CC}=5V
10. **Watchdog Timer (WDT)**
    - Clocked by built-in RC oscillator with 4 adjustable reset times

      - 91 ms / 183 ms / 732 ms / 1463 ms @V_{CC}=5V

- Watchdog timer can be disabled / enabled in STOP mode

11. Six 16 bits PWMs

- Six individual duty-adjustable, shared period-adjustable
- PWM clock source: System clock (Fsys), FIRC/256, FIRC (16 MHz), FIRC*2 (32 MHz)
- PWM0 supports complementary output (PWM0P, PWM0N)
- PWM0 output with dead-zone(non-overlap) time durations adjustable: (0~15)*(PWMCLK)
- PWM0N/0P/1/2/3/5 has only one output

12. 12-bit ADC with 13 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} (1.2V), V_{BG} (2.48V) and V_{BG} (2V)

13. Reset Sources

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

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

- 15-Level Low Voltage Reset: 1.73V ~ 3.5V, can be disabled
- 15-Level Low Voltage Detection: 1.73V ~ 3.5V, can be disabled

15. Operating Voltage

- Fsys= 16 MHz, LVR~5.5V. Suggest LVR ≥ 2.30V
- Fsys= 8 MHz, PWMCKS=FIRC*1, LVR~5.5V. Suggest LVR ≥ 1.55V

  Note: Refer to the "Electrical Characteristics Graphs".

16. Operating Temperature Range : -40°C to + 105°C

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

18. Integrated 16-bit Cyclic Redundancy Check (CRC) function

19. Instruction set: 39 Instructions

20. I/O ports:

- Maximum 14 programmable I/O pins
  - Open-Drain Output
  - CMOS Push-Pull Output
  - Schmitt Trigger Input with pull-up resistor option
  - All I/O with High-Sink except PA7
- All pin-change wake-up (falling edge and rising edge trigger) and interrupt

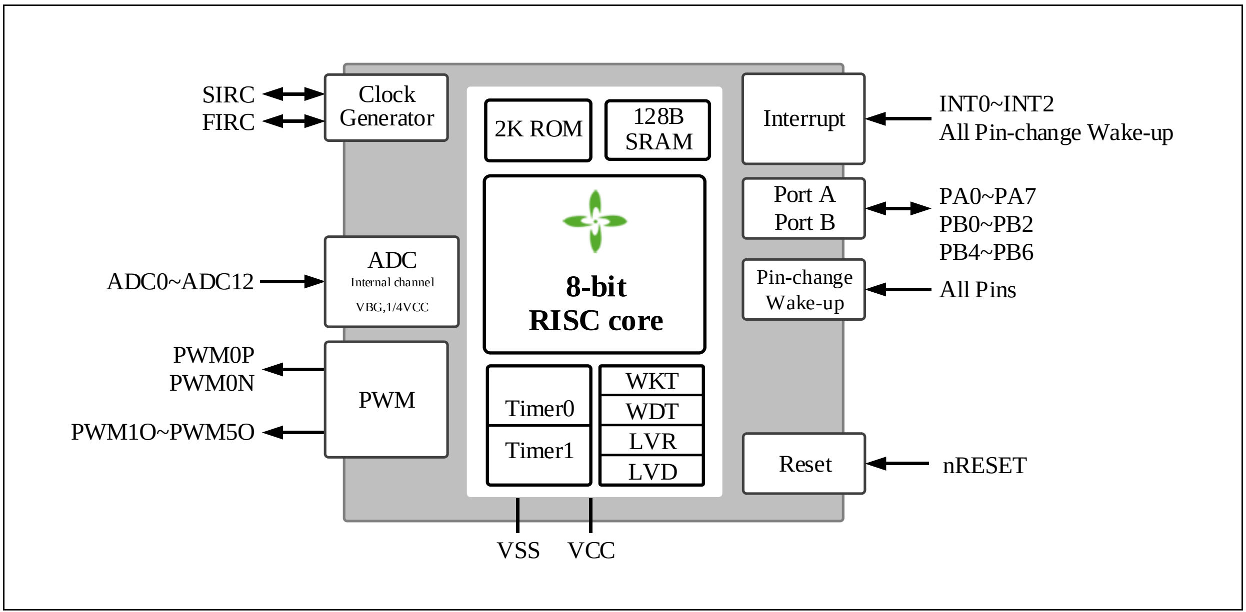
21. Programming connectivity support 5-wire (ICP) or 7-wire program

22. RDCTL: Read signal delay control for Program ROM

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

23. **Trimmed VBG1.2V/2V**
    - The users could move BG2TRIM to BGTRIM for slightly exact 2V VBG.
24. **ATD: Automatic transient detection(Read signal length control for Program ROM) to enhance the performance of power consumption at slow mode**
25. **Package Types:**
    - 16-pin SOP (150 mil)
    - 14-pin SOP (150 mil)
    - 10-pin MSOP (118 mil)
    - 8-pin SOP (150 mil)
    - 16-pin QFN (3*3*0.75 - 0.5mm)
    - 10-pin DFN (2*2*0.75 - 0.4mm)
    - 8-pin SOT23
26. **Supported EV board**
    - TM56F1552/22

SYSTEM BLOCK DIAGRAM

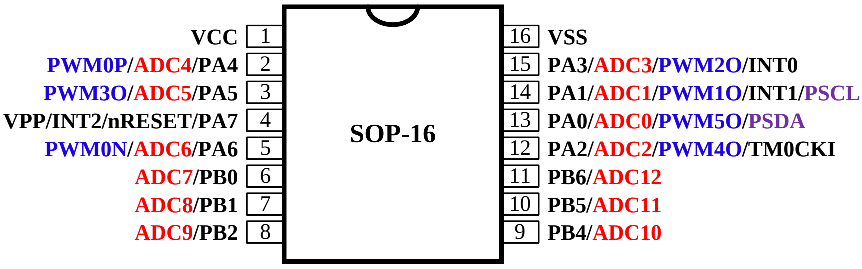


TM56M152A Block Diagram

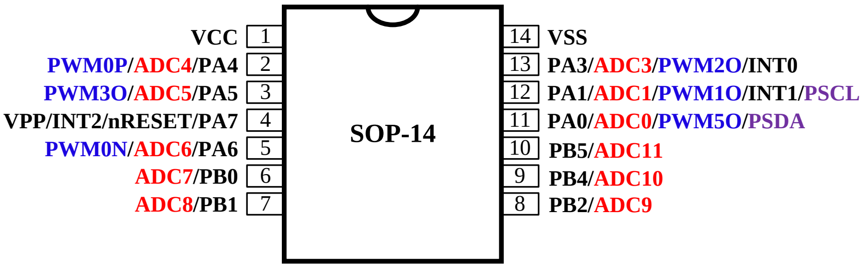
PIN ASSIGNMENT DIAGRAM

Software initialization is necessary for the pads that are not bonded.

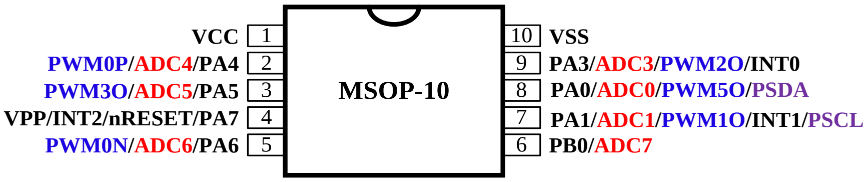
TM56M152A



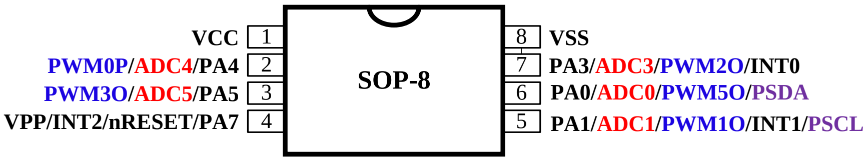
TM56M152A



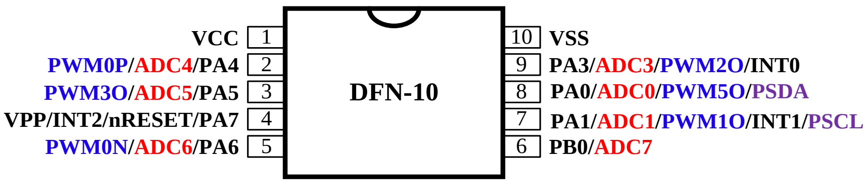
TM56M152A



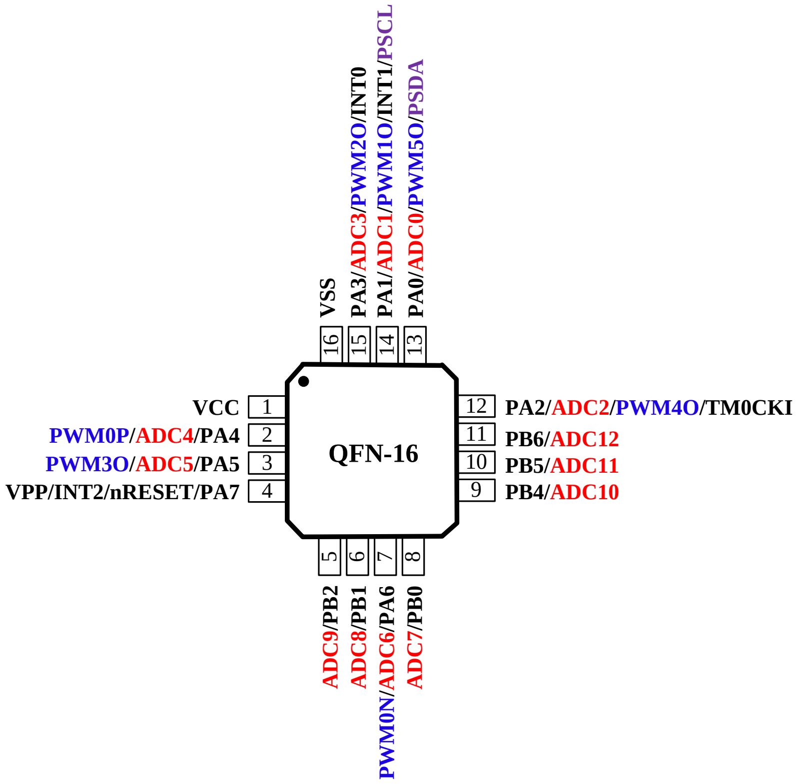
TM56M152A



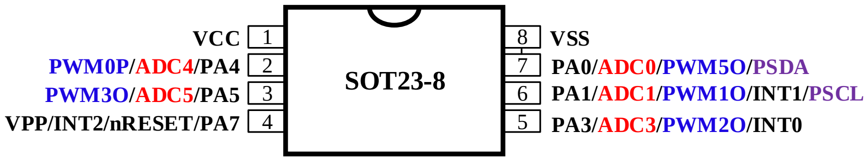
TM56M152A



TM56M152A



TM56M152A



PIN DESCRIPTIONS

Name	In/Out	Pin Description
PA0~PA7 PB0~PB2 PB4~PB6	I/O	Bit-programmable I/O port for Schmitt-trigger input, CMOS push-pull output or open-drain output. Pull-up resistors are assignable by software.
nRESET	I	External active low reset
VCC, VSS	P	Power Voltage input pin and ground
VPP	I	Programming high voltage(9.5V) input
INT0~INT2	I	External interrupt input
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	16 bits PWM1~PWM5 output
ADC0~ADC12	I	ADC channel input
PSCL	I	Clock for programmer
PSDA	I/O	Data for programmer

Programming pins:

Normal mode (7-wire): VCC / VSS / PA0(PSDA) / PA1(PSCL) / PA4 / PA5 / PA7(VPP)

ICP mode (5-wire): VCC / VSS / PA0(PSDA) / PA1(PSCL) / PA7(VPP) - When using ICP (In-Circuit Program) mode, the PCB needs to remove all components of PA0, PA1.

PIN SUMMARY

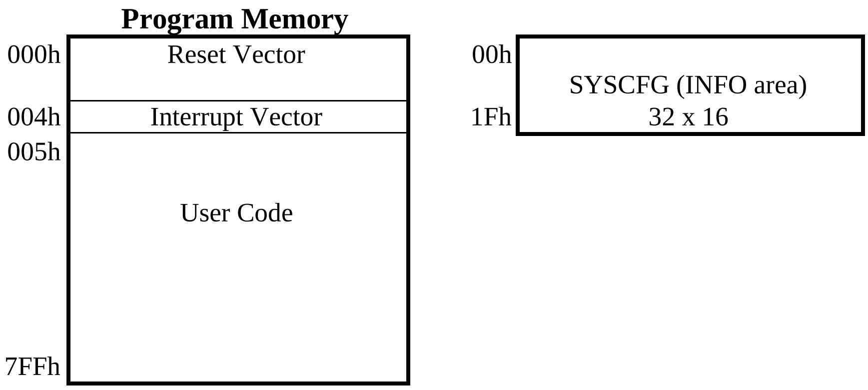
Pin Number						Pin Name	Type	GPIO							Alternate Function			
								Input				Output						
TM56M152A(SOP-16)	TM56M152A(QFN-16)	TM56M152A(SOP-14)	TM56M152A(MSOP-10) (DFN-10)	TM56M152A(SOT23-8)	TM56M152A(SOP-8)			Pull-up Control	Pull-down Control	Ext. Interrupt	Wake-up Interrupt	Open Drain	CMOS Push-Pull	1/2 V_{CC} (LCD 1/2 Bias)	PWM	ADC	Comparator	MISC
2	2	2	2	2	2	PA4/ADC4/PWM0P	I/O	●			●	●	●		●	●		
3	3	3	3	3	3	PA5/ADC5/PWM3O	I/O	●			●	●	●		●	●		
4	4	4	4	4	4	PA7/nRESET/INT2/VPP	I/O	●		●	●	●	●					nRESET/VPP
5	7	5	5		–	PA6/ADC6/PWM0N	I/O	●			●	●	●		●	●		
6	8	6	6		–	PB0/ADC7	I/O	●			●	●	●			●		
7	6	7	–		–	PB1/ADC8	I/O	●			●	●	●			●		
8	5	8	–		–	PB2/ADC9	I/O	●			●	●	●			●		
9	9	9	–		–	PB4/ADC10	I/O	●			●	●	●			●		
10	10	10	–		–	PB5/ADC11	I/O	●			●	●	●			●		
11	11	–	–		–	PB6/ADC12	I/O	●			●	●	●			●		
12	12	–	–		–	PA2/ADC2/PWM4O/TM0CKI	I/O	●			●	●	●		●	●		TM0CKI
13	13	11	8	7	6	PA0/ADC0/PWM5O/PSDA	I/O	●			●	●	●		●	●		Programming
14	14	12	7	6	5	PA1/ADC1/PWM1O/INT1/PSCL	I/O	●		●	●	●	●		●	●		Programming
15	15	13	9	5	7	PA3/ADC3/PWM2O/INT0	I/O	●		●	●	●	●		●	●		
16	16	14	10	8	8	VSS	P											
1	1	1	1	1	1	VCC	P											

FUNCTION DESCRIPTION

1 CPU Core

1.1 Program ROM (PROM)

The ROM of this device is 2K words, with an extra 32-Word INFO area to store the SYSCFG. 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℃ 。



113h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
RDCTL	–	–	–	–	–	–	RDCTL	
R/W	–	–	–	–	–	–	R/W	
Reset	–	–	–	–	–	–	1	0

113h.1~0 **RDCTL:** Read signal delay control for Program ROM
00: 16ns delay for read signal of Program ROM
01: 12ns delay for read signal of Program ROM
10: 8ns delay for read signal of Program ROM
11: 4ns delay for read signal of Program ROM
Change this register at slow clock for safety.
The user must switch this register to "4ns" to enhance the performance of minimal operating voltage.
This feature can't be emulated.

1.1.1 Reset Vector (000h)

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

1.1.2 Interrupt Vector (004h)

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

1.2 System Configuration Register (SYSCFG)

The System Configuration Register (SYSCFG) is located at 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.

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	**LVR**: Low Voltage Reset Mode	
		0001	LV Reset 1.73V
		0010	LV Reset 1.85V
		0011	LV Reset 1.98V
		0100	LV Reset 2.11V
		0101	LV Reset 2.23V
		0110	LV Reset 2.36V
		0111	LV Reset 2.49V
		1000	LV Reset 2.61V
		1001	LV Reset 2.74V
		1010	LV Reset 2.87V
		1011	LV Reset 2.99V
		1100	LV Reset 3.12V
		1101	LV Reset 3.25V
		1110	LV Reset 3.37V
		1111	LV Reset 3.50V
	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)
	2-0	tenx Reserved	

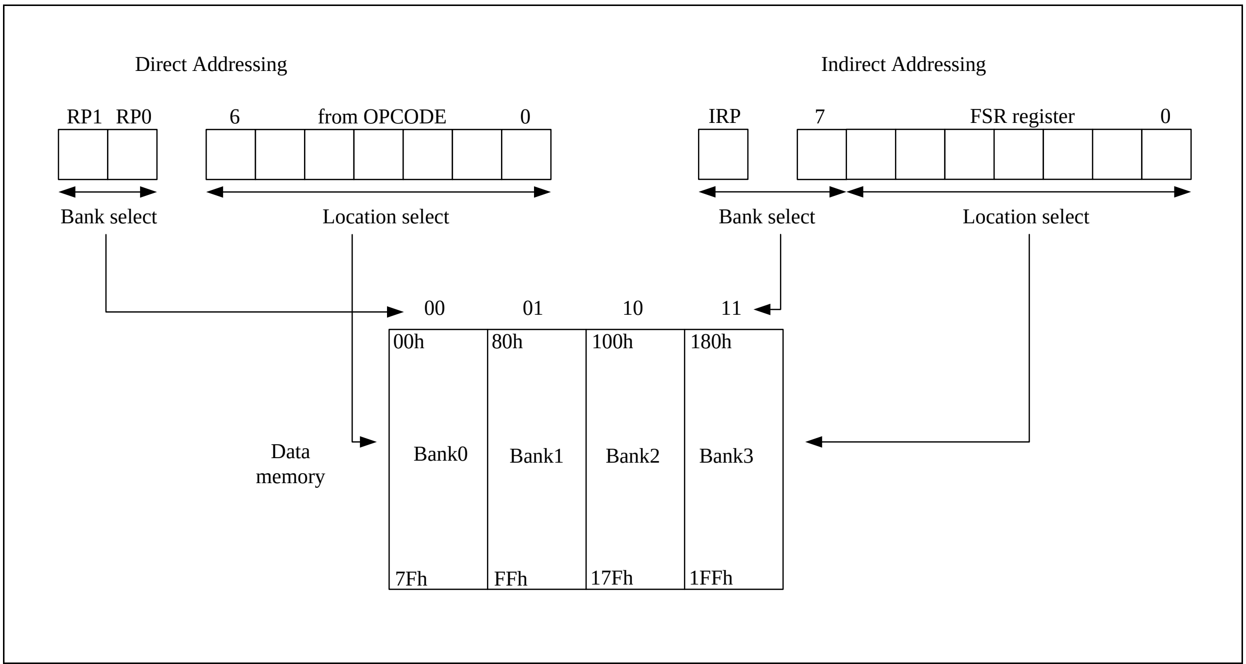
1.3 RAM Addressing Mode

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:

BCF	TM0IE	→	BCX	TM0IE
DECF	CNT, 1	→	DECX	CNT, 1
INCFSZ	RAM25, 0	→	INCXSZ	RAM25, 0
MOVWF	PAMOD10	→	MOVWX	PAMOD10
RLF	RAMA0, 0	→	RLX	RAMA0, 0
SWAPF	ADCTL, 0	→	SWAPX	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	PINMOD	185h	DPL
006h	PBD	086h	PAMOD32	106h		186h	DPH
007h		087h	PAMOD54	107h		187h	CRCDL
008h		088h	PAMOD76	108h		188h	CRCDH
009h		089h	PWMCTL	109h	LVRPD	189h	CRCIN
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	
00Eh	INTIF1	08Eh	PBMOD54	10Eh	BGTRIM	18Eh	
00Fh	CLKCTL	08Fh	PBMOD76	10Fh	IRCF	18Fh	
010h	TM0RLD	090h		110h		190h	
011h	TM0CTL	091h	OPTION2	111h	BG2TRIM	191h	
012h	TM1	092h	PWMPRDH	112h		192h	
013h	TM1RLD	093h	PWMPRDL	113h	RDCTL	193h	
014h	TM1CTL	094h	PWM0DH	114h		194h	
015h		095h	PWM0DL	115h		195h	
016h	LVCTL	096h	PWM1DH	116h		196h	
017h	ADCDH	097h	PWM1DL	117h		197h	
018h	ADCTL	098h	PWM2DH	118h		198h	
019h	ADCTL2	099h	PWM2DL	119h		199h	
01Ah		09Ah	PWM3DH	11Ah		19Ah	
01Bh		09Bh	PWM3DL	11Bh		19Bh	
01Ch		09Ch	PWM4DH	11Ch		19Ch	
01Dh		09Dh	PWM4DL	11Dh		19Dh	
01Eh		09Eh	PWM5DH	11Eh		19Eh	
01Fh		09Fh	PWM5DL	11Fh		19Fh	
020h	RAM Bank0 area (80 Bytes)	0A0h	RAM Bank1 area (32 Bytes)	120h	Don't Use	1A0h	Don't Use
		0C0h	Don't Use				
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
OPTION2   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     OPTION2        ; read OPTION2 (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
```

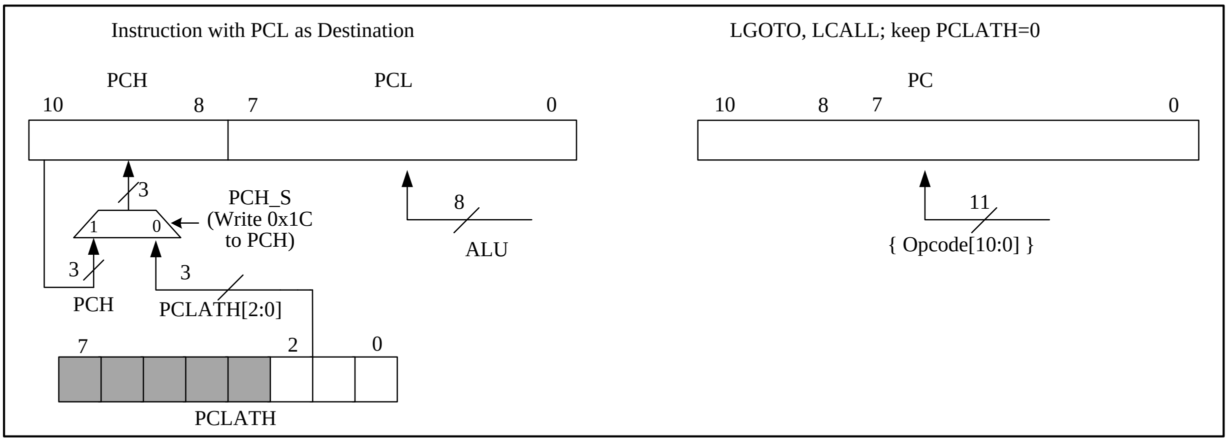
1.4 Programming Counter (PC) and Stack

The Programming Counter is 11-bit wide and capable of addressing a 2K x 16 ROM. 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. For RET/RETI/RETLW instruction, PC retrieves its content from the top level STACK.

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[10:8]) 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.2~0 **PCLATH:** Programming Counter high byte data when instruction with PCL as destination is executed, and PCH_S is 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
After reset, the PCH_S is cleared

10Ch.2~0 **PCH (R)**: Programming Counter data bit 10~8

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 and TABR register by setting DPTR={DPH, DPL} registers. It also offers another way to read the16-bit ROM data into W and TABR register by setting TABR (18Ch) for C language.

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

```
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
        …
        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 and TABR
                                          (W = TABR = 86h)
            TABRH                       ; Read PROM high byte data to W and TABR
                                          (W = TABR = 19h)
            …
; Table Read by SFR TABR
            MOVLW     01h               ; TABR write 01h = instruction TABRL
            MOVWX     TABR              ; Read PROM low byte data to W and TABR
                                          (W = TABR = 86h)
            MOVXW     TABR              ; Read TABR to W (W = 86h)
            MOVLW     02h               ; TABR write 02h = instruction TABRH
            MOVWX     TABR              ; read PROM high byte data to W and TABR
                                          (W = TABR = 19h)
            MOVXW     TABR              ; read TABR to W (W = 19h)
            …
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
            …
TABLE2:
            .DT       0x1986            ; 16-bit ROM data
            .DT       0x3719
            …
```

Note: The chip define 256 ROM address as one page, so that ROM has 8 pages, 000h~0FFh, 100h~1FFh, …, 700h~7FFh. On the other words, PC[10:8] 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.

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 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.

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.

STATUS	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Reset Value	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R	R	R/W	R/W	R/W

Bit	Description	
7	**IRP**: Register Bank Select bit (used for indirect addressing) 0 = Bank 0,1 (000h - 0FFh) 1 = Bank 2,3 (100h - 1FFh)	
6:5	**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	
4	**TO**: Time Out Flag 0: after Power On Reset or CLRWDT/SLEEP instruction 1: WDT time out occurs	
3	**PD**: Power Down Flag 0: after Power On Reset or CLRWDT instruction 1: after SLEEP instruction	
2	**Z**: Zero Flag 0: the result of a logic operation is not zero 1: the result of a logic operation is zero	
1	**DC**: Decimal Carry Flag or Decimal / Borrow Flag	
	ADD instruction	SUB instruction
	0: no carry 1: a carry from the low nibble bits of the result occurs	0: a borrow from the low nibble bits of the result occurs 1: no borrow
0	**C**: Carry Flag or /Borrow Flag	
	ADD instruction	SUB instruction
	0: no carry 1: a carry occurs from the MSB	0: a borrow occurs from the MSB 1: no borrow

◇ 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. See the following LVR Selection Table; user must also consider the lowest operating voltage of operating frequency.

LVR Selection Table:

LVR level	Operating voltage
LVR1.73	5.5V > V_{CC} > 1.73V
LVR1.85	5.5V > V_{CC} > 1.85V
LVR1.98	5.5V > V_{CC} > 1.98V
LVR2.11	5.5V > V_{CC} > 2.11V
LVR2.23	5.5V > V_{CC} > 2.23V
LVR2.36	5.5V > V_{CC} > 2.36V
LVR2.49	5.5V > V_{CC} > 2.49V
LVR2.61	5.5V > V_{CC} > 2.61V
LVR2.74	5.5V > V_{CC} > 2.74V
LVR2.87	5.5V > V_{CC} > 2.87V
LVR2.99	5.5V > V_{CC} > 2.99V
LVR3.12	5.5V > V_{CC} > 3.12V
LVR3.25	5.5V > V_{CC} > 3.25V
LVR3.37	5.5V > V_{CC} > 3.37V
LVR3.50	5.5V > V_{CC} > 3.50V

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

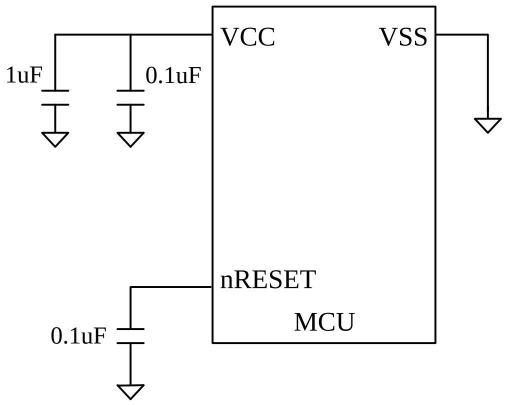
16h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
LVCTL	LVDF	LVDHYS	LVRSAV	LVDSAV	LVDS			
R/W	R	R/W	R/W	R/W	R/W			
Reset	0	0	1	1	0	0	0	0

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
```

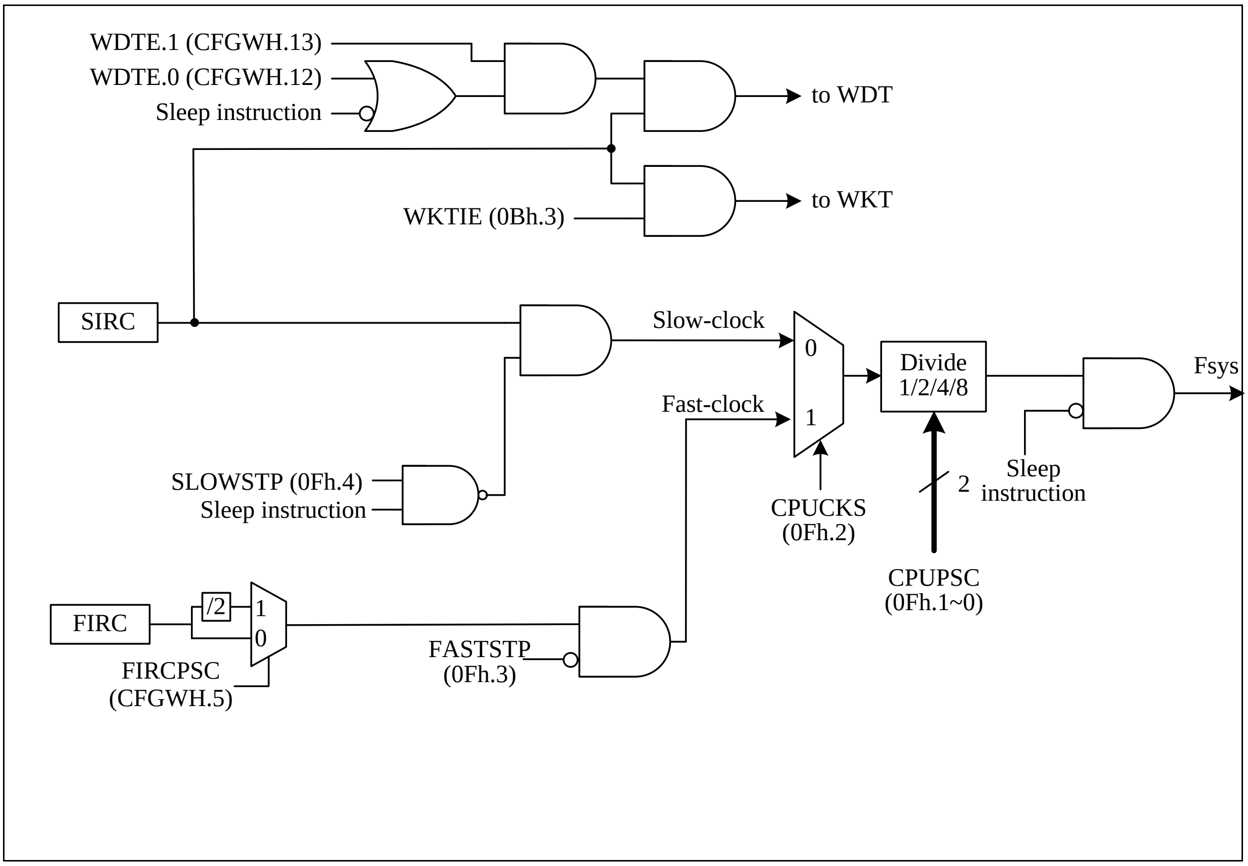
3 Clock Circuitry and Operation Mode

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 can be configured to keep oscillating to provide clock source to WKT/WDT block. Refer to the Figure as below.

After Reset, the device is running at SLOW mode with 93 KHz(@Vcc=5V) 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 PWMCKS (91h.5~4).

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 block, which can select other clock source. 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.

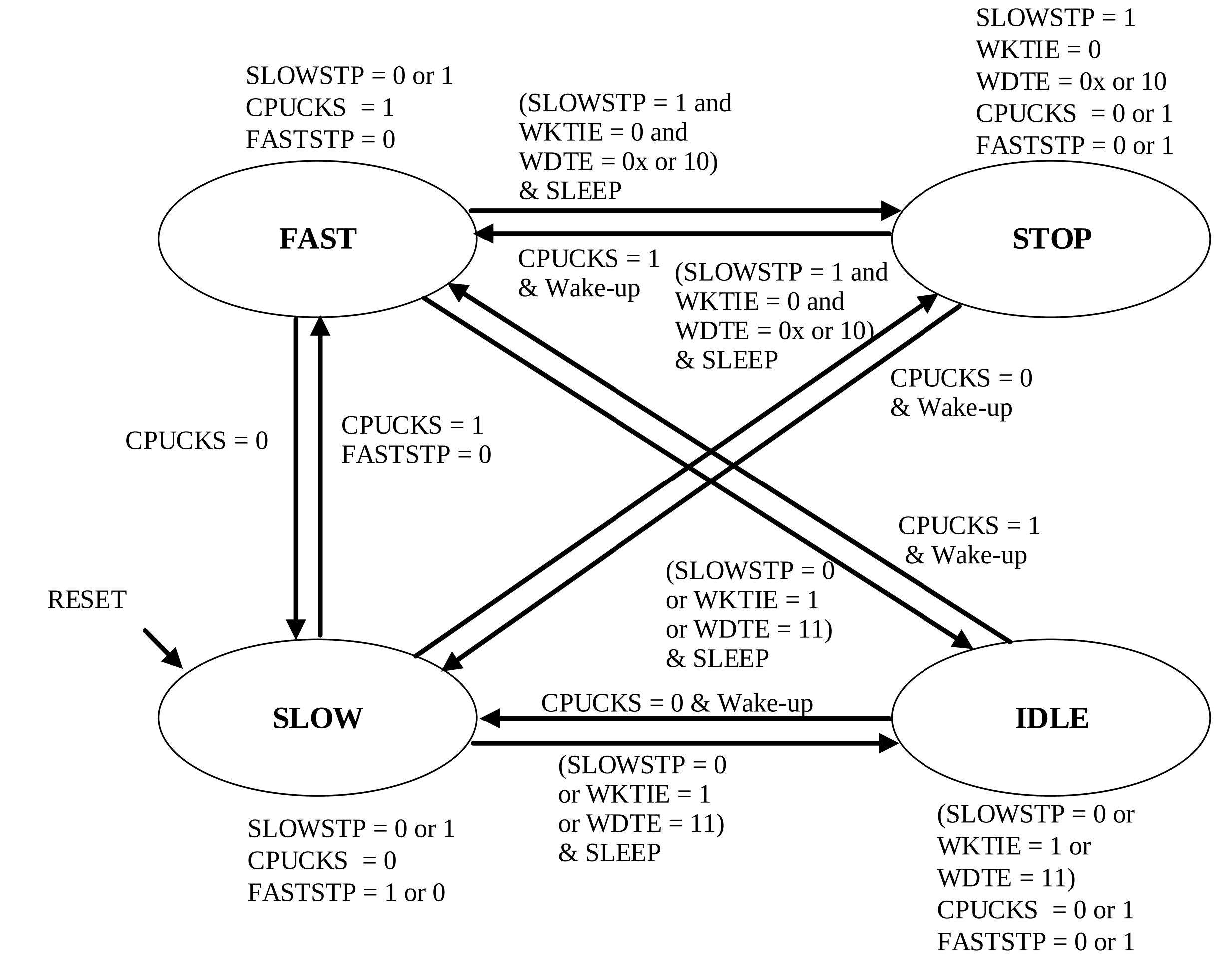
(2) Set WKTIE=1 or WDTE=11, before executing the SLEEP instruction, the SIRC can still oscillate to keep WKT/WDT operating in IDLE mode.

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 External pin interrupt or WKT 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 External interrupt and WKT 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 external pin interrupt and 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	INT2IE	INT1IE	INT0IE
R/W	R/W	–	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	–	0	0	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

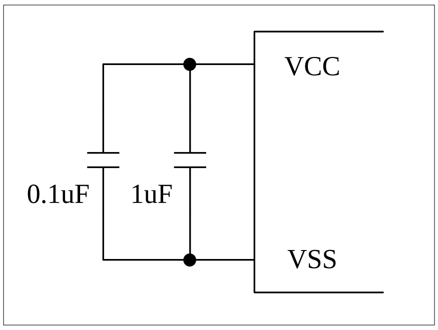
0Fh.3 **FASTSTP**: Fast-clock stop
0: Fast-clock is running
1: Fast-clock stops running

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.



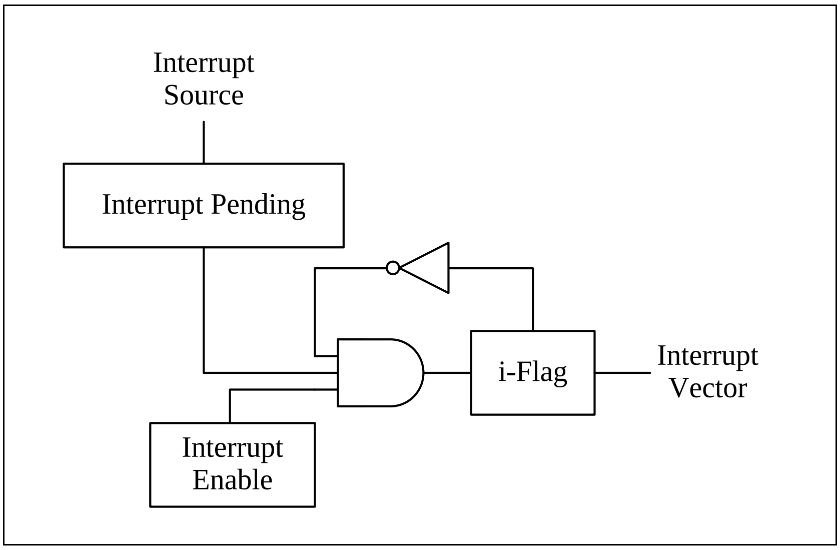
Internal RC Mode

4 Interrupt

The Chip has 1 level, 1 vector and 9 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[5:0], INTIE1[6], INTIE1[1:0]) 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 INT1 (PA1) interrupt request with rising edge trigger

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

            ORG       004h              ; All interrupt vector
            LGOTO     INT               ; If INT1 (PA1) input occurred rising edge

            ORG       005h
START:
            MOVLW     0000xxxxb
            MOVWX     PAMOD10           ; Select INT1 Pin Mode as mode 0000b
                                        ; Open drain output low or input with Pull-up
            MOVLW     xxxxxx1xb
            MOVWX     PAD               ; Release INT1, it becomes Schmitt-trigger
                                        ; input with input pull-up resistor
            MOVLW     xx1xxxxxb
            MOVWX     OPTION            ; Set INT1 interrupt trigger as rising edge
            MOVLW     11111101b
            MOVWX     INTIF             ; Clear INT1 interrupt request flag
            MOVLW     00000010b
            MOVWX     INTIE             ; Enable INT1 interrupt
MAIN:
            …
            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     INT1IF            ; Check INT1IF bit
            LCALL     INT1_SUB          ; INT1IF = 1, jump to INT1 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

INT1_SUB:                               ; INT1 interrupt service routine
            …
            MOVLW     11111101b
            MOVWX     INTIF             ; Clear INT1 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	INT2IE	INT1IE	INT0IE
R/W	R/W	–	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	–	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.2 **INT2IE:** INT2 interrupt 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	INT2IF	INT1IF	INT0IF
R/W	R/W	–	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	–	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.2 **INT2IF:** INT2 pin falling interrupt pending flag
This bit is set by H/W at INT2 pin’s falling edge, write FBh 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, write FDh to INTIF will 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, write FEh to INTIF will clear this flag

0Dh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTIE1	–	PCIE	–	–	–	–	PWMIE	LVDIE
R/W	–	R/W	–	–	–	–	R/W	R/W
Reset	–	0	–	–	–	–	0	0

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

0Dh.1 **PWMIE:** PWM 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	–	PCIF	–	–	–	–	PWMIF	LVDIF
R/W	–	R/W	–	–	–	–	R/W	R/W
Reset	–	0	–	–	–	–	0	0

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.1 **PWMIF:** PWM interrupt event pending flag
This bit is set by H/W after PWM 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 VCC < VLVD, 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	INT1EDG	–	WDTPSC		WKTPSC	
R/W	R/W	R/W	R/W	–	R/W		R/W	
Reset	0	0	0	–	1	1	1	1

81h.6 **INT0EDG:** INT0 pin interrupt edge selection
0: falling edge to trigger
1: rising edge to trigger

81h.5 **INT1EDG:** INT0 pin interrupt edge selection
0: falling edge to trigger
1: rising edge to 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

0000: Disable	0100 : 2.11V	1000: 2.61V	1100: 3.12V
0001: 1.73V	0101: 2.23V	1001: 2.74V	1101: 3.25V
0010: 1.85V	0110: 2.36V	1010: 2.87V	1110: 3.37V
0011: 1.98V	0111: 2.49V	1011: 2.99V	1111: 3.50V

5 I/O Port

5.1 PA0-PA7, PB0-PB2, PB4-PB6

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, pin-change wake-up, PWMO and so on. All IO except PA7 support two sink current options, which are defined by the HSINK (105h.2). **PA7 has no high-sink capability. All IOs have no 1/2 bias and pull-down capability.**

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

PAxMOD PBxMOD	PADx PBDx	PA0~PA7, PB0~PB2, PB4~PB6 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	Hi-Z	-	-	-

I/O Pin Function Table 1

PAxMOD PBxMOD	PADx PBDx	PA0~PA7, PB0~PB2, PB4~PB6 pin function	Pin State	Resistor Pull-down	Digital Input	Pin-changed Wake-up
0100b	0	Open Drain	Drive Low	-	-	-
	1	Input	Hi-Z	-	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	-	-	-	-

I/O Pin Function Table 2

PAxMOD PBxMOD	PADx PBDx	PA0~PA7, PB0~PB2, PB4~PB6 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				

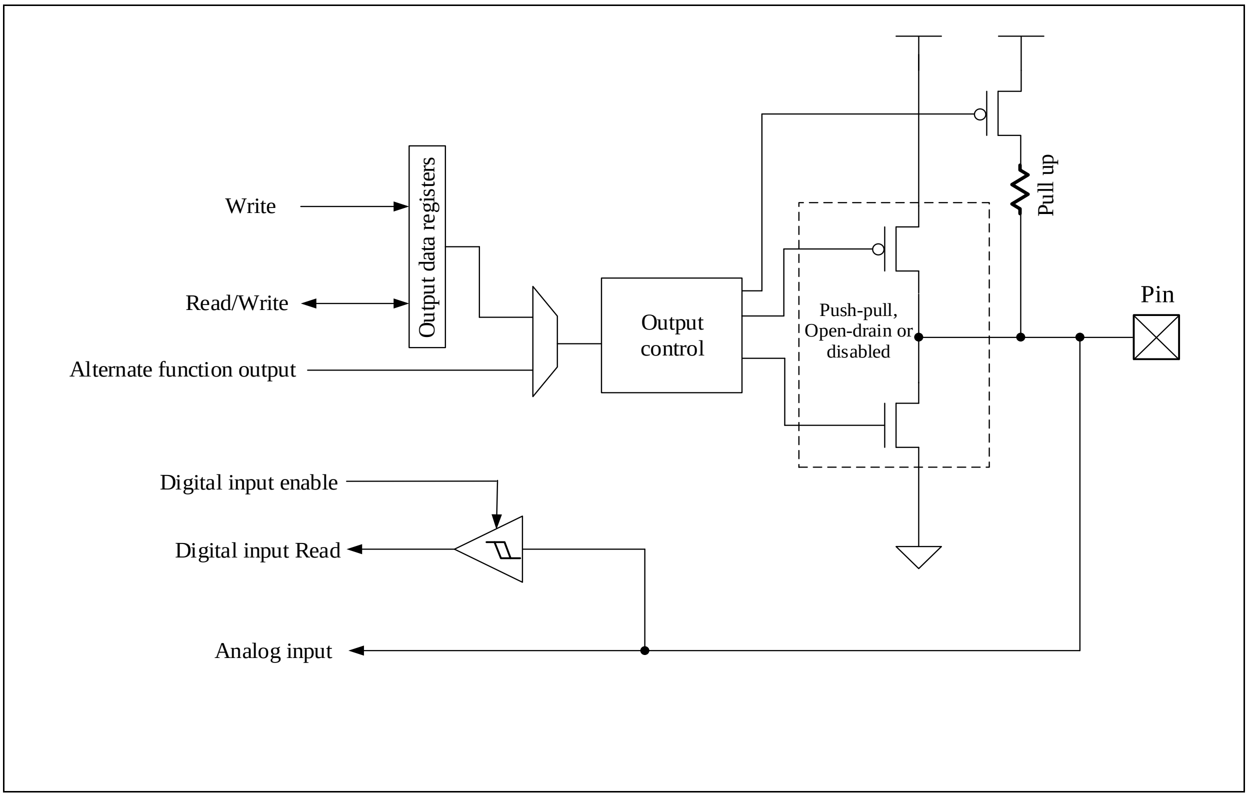
I/O Pin Function Table 3

PAxMOD PBxMOD	PADx PBDx	PA0~PA7, PB0~PB2, PB4~PB6 pin function	Pin State	Resistor Pull-down	Digital Input	Pin-changed Wake-up
1100b	0	Open Drain	Drive Low	-	-	-
	1	Input	Hi-Z	-	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		Reserved		-	-	-

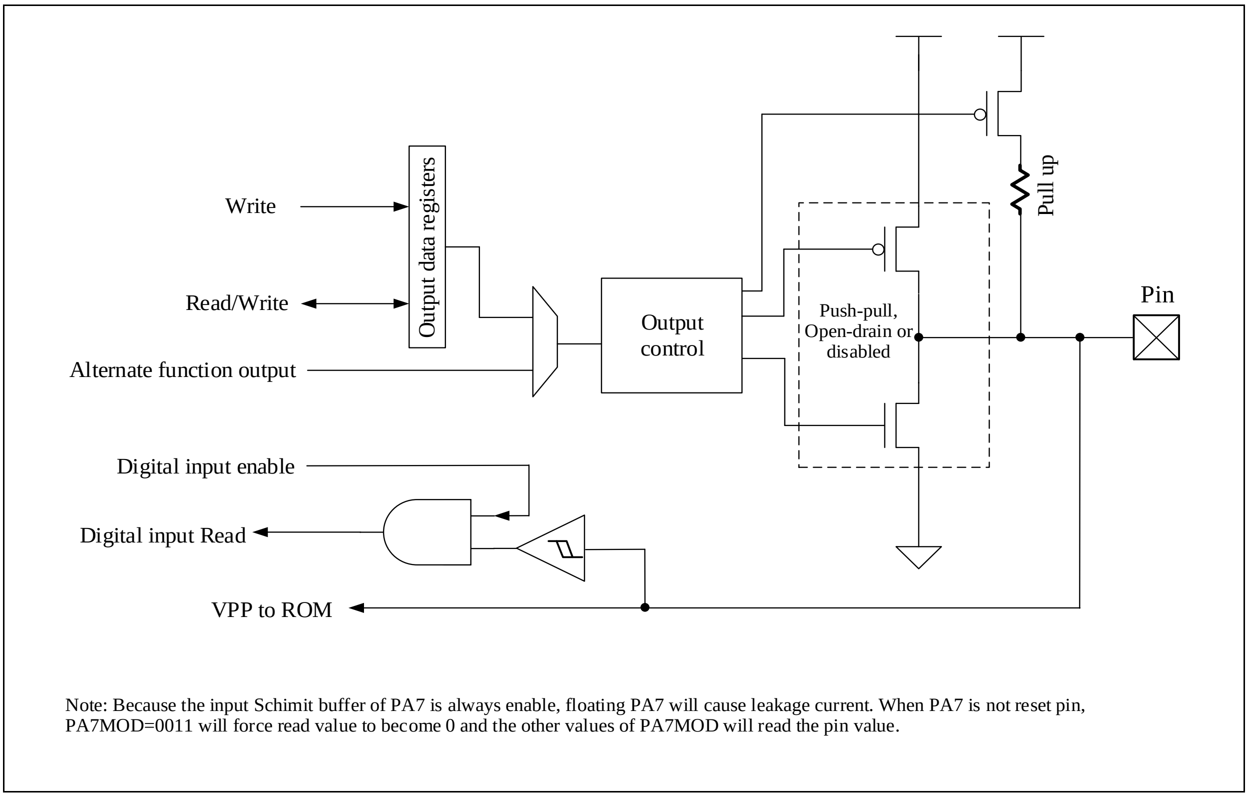
I/O Pin Function Table 4

Pin Name	PAxMOD / PBxMOD Setting	
	0011b (Analog in/out)	0111b (Digital output)
PA0	ADC0	PWM5O
PA1	ADC1	PWM1O
PA2	ADC2	PWM4O
PA3	ADC3	PWM2O
PA4	ADC4	PWM0P
PA5	ADC5	PWM3O
PA6	ADC6	PWM0N
PA7	-	-
PB0	ADC7	-
PB1	ADC8	-
PB2	ADC9	-
PB4	ADC10	-
PB5	ADC11	-
PB6	ADC12	-

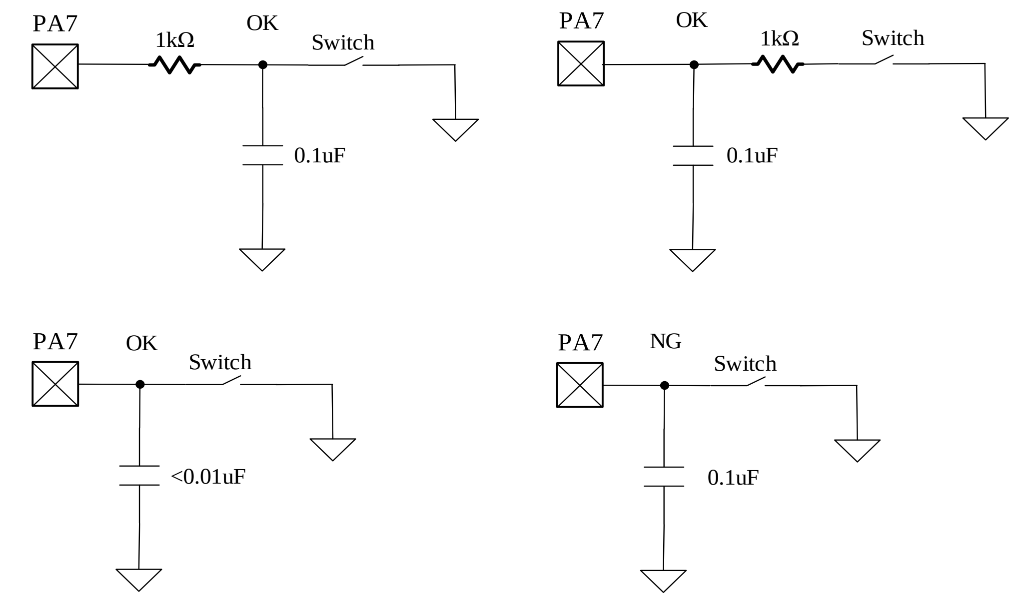
Special function for PxxMOD Table



General Pin Structure



PA7 Structure



Constraint on PA7

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	
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 and pin-changed wake-up
1101: Open drain or digital input and pin-changed wake-up
1110: CMOS Push-pull
1111: Reserved

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	-				PB2MOD			
R/W	-				R/W			
Reset	-				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	-				PB6MOD			
R/W	-				R/W			
Reset	-				0	0	0	1

8Fh.3~0 **PB6MOD ~ PB4MOD, PB2MOD ~ PB0MOD**: PB6~PB4 and PB2~PB0 Pin Mode Control
8Eh.7~4 0000: Open drain or digital input with pull-up
8Eh.3~0 0001: Open drain or digital input
8Dh.3~0 0010: CMOS Push-pull
8Ch.7~4 0011: Analog input
8Ch.3~0 0100: Open drain or digital input
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 and pin-changed wake-up
1101: Open drain or digital input and pin-changed wake-up
1110: CMOS Push-pull
1111: Reserved

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

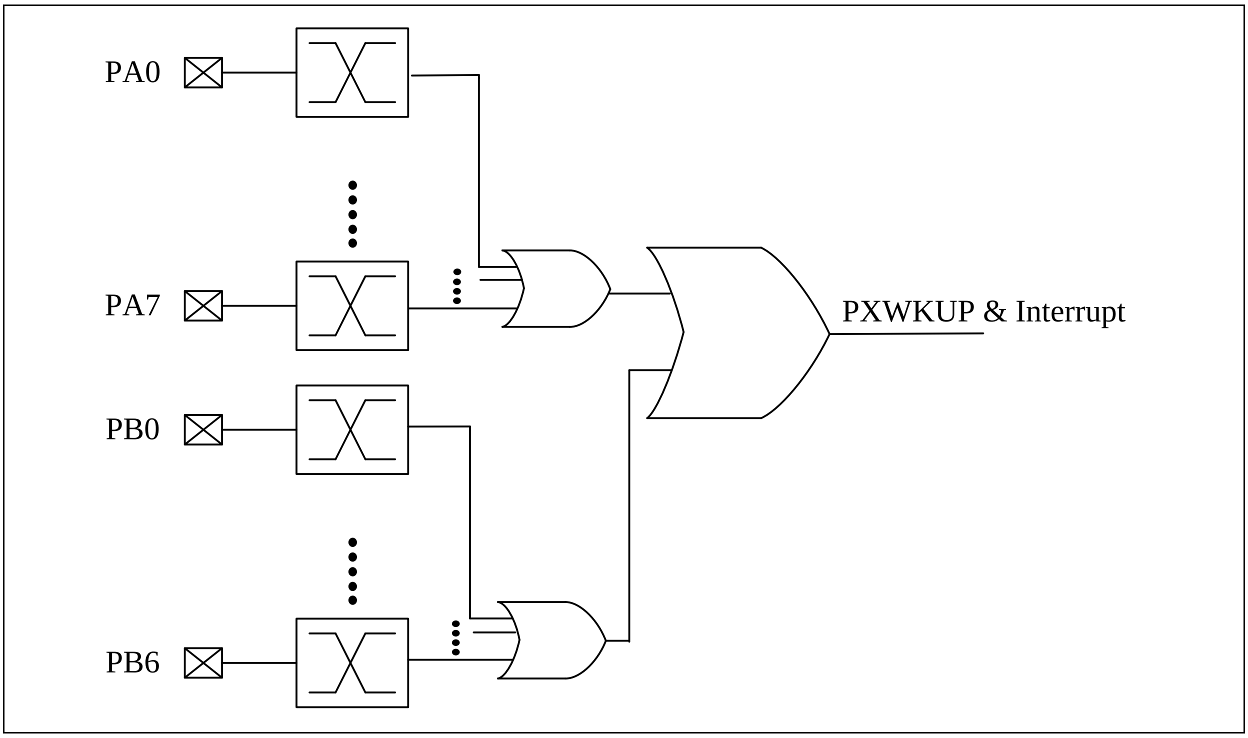
06h.7~0 **PBD**: PB7~PB0 data

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

105h.5 **Reserved**: read as unknown after reset
105h.2 **HSINK**: All IO ports high sink current enable
0: low sink current
1: high sink current. PA7 has no high-sink capability.
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.

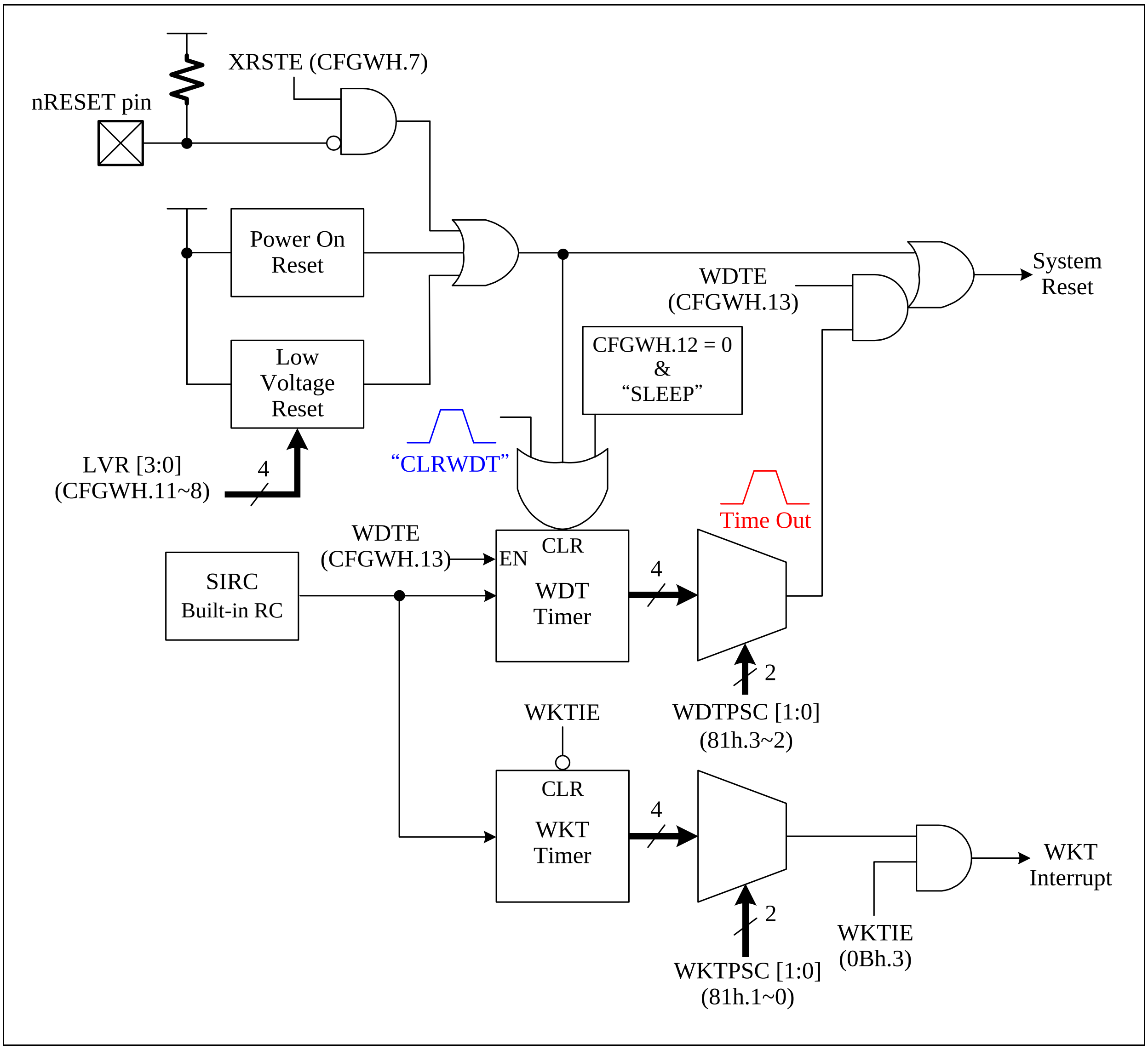


6 Peripheral Functional Block

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      00000111b
        MOVWX      OPTION              ; Select WDT Time out=168 ms @5V
        …
```

◇ Example: Set WKT period and interrupt function.

```
        MOVLW      00000110b
        MOVWX      OPTION              ; Select WKT period=42 ms @5V
        MOVLW      11110111b           ; 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	INT2IF	INT1IF	INT0IF
R/W	R/W	–	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	–	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	INT2IE	INT1IE	INT0IE
R/W	R/W	–	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	–	0	0	0	0	0	0

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

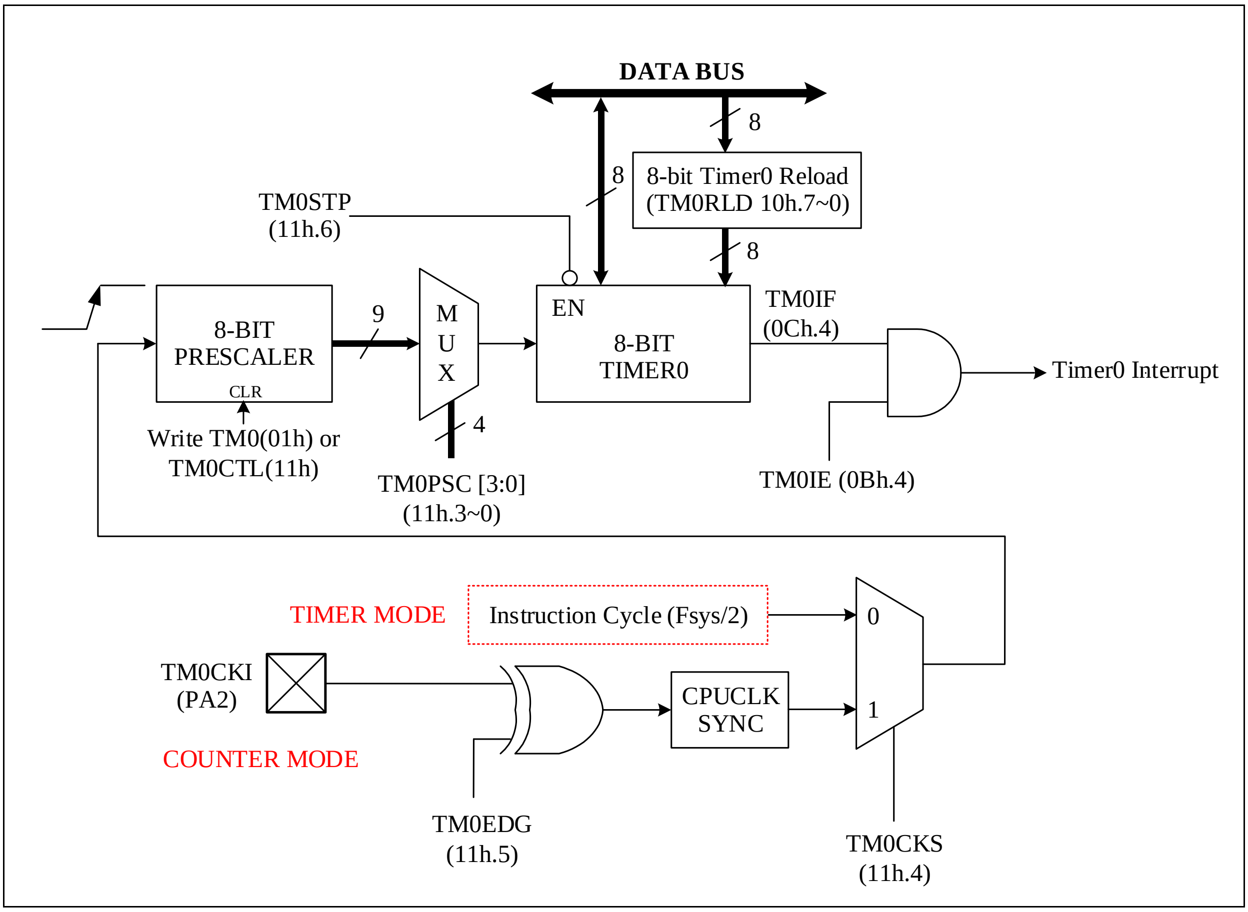
81h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
OPTION	HWAUTO	INT0EDG	INT1EDG	–	WDTPSC		WKTPSC	
R/W	R/W	R/W	R/W	–	R/W		R/W	
Reset	0	0	0	–	1	1	1	1

81h.3~2 **WDTPSC:** WDT period (@V_{CC}=5V)
00: 91 ms
01: 183 ms
10: 732 ms
11: 1463 ms

81h.1~0 **WKTPSC:** WKT period (@V_{CC}=5V)
00: 11 ms
01: 23 ms
10: 46 ms
11: 91 ms

6.2 Timer0

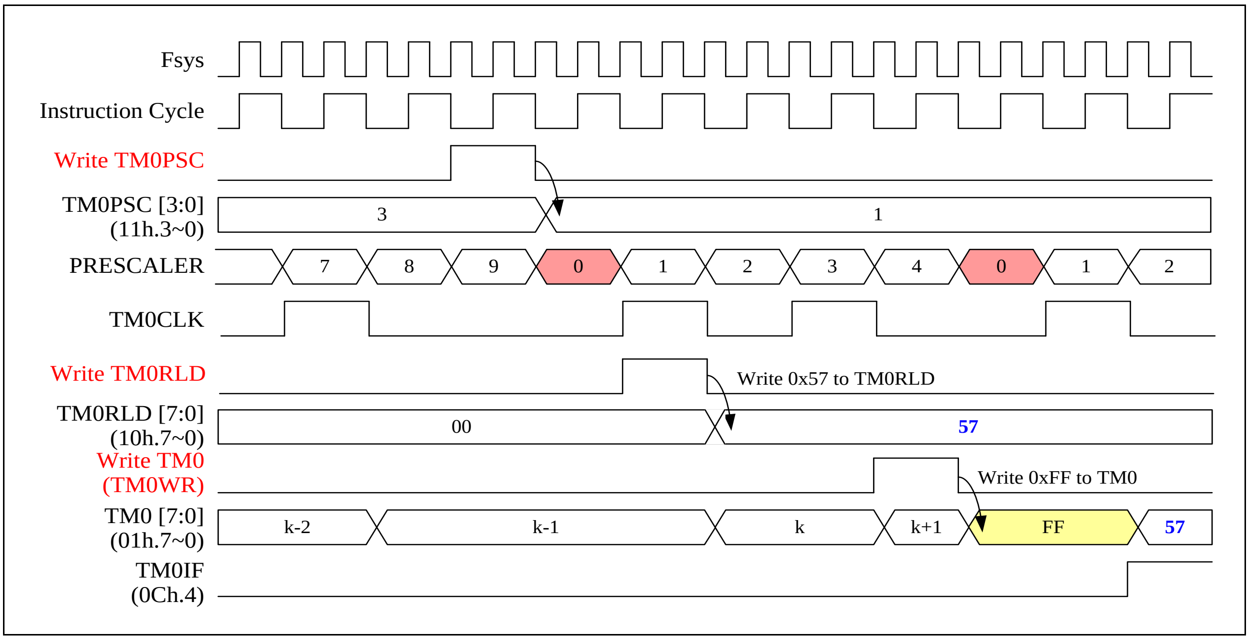
Timer0(TM0) (01h.7~0) is an 8-bit wide register. It can be read or written as any other register. Besides, Timer0 increases itself periodically and automatically rolls over a new "offset value" (TM0RLD) while it rolls over based on the pre-scaled clock source, which can be Fsys/2 or TM0CKI (PA2) rising/falling 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 frequency = 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     11101111b
        MOVWX     INTIF             ; Clear Timer0 request interrupt flag
        BSX       TM0IE             ; Enable Timer0 interrupt function
        BCX       TM0STP            ; Enable Timer0 counting
```

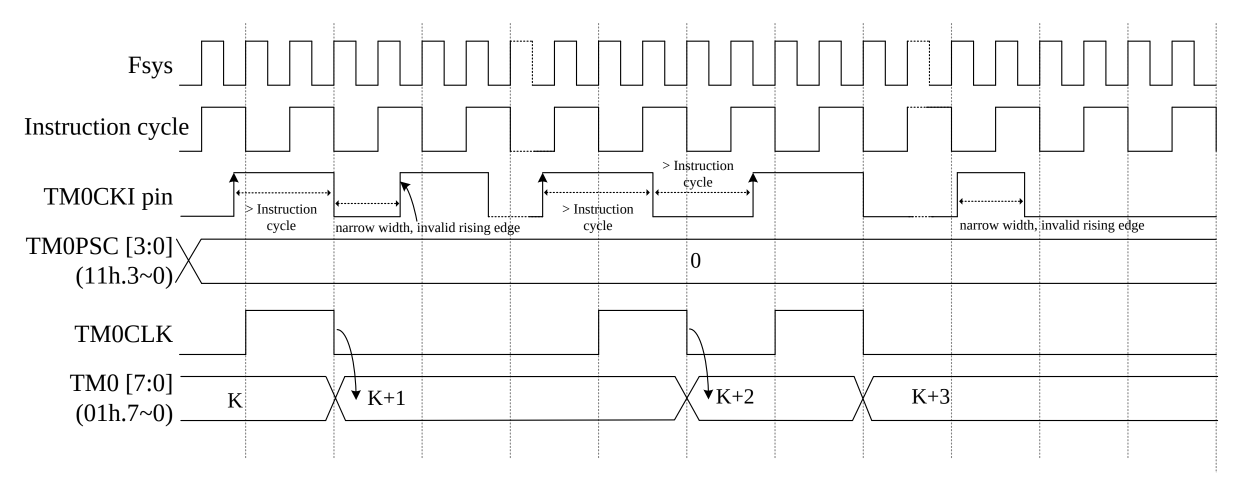
Timer0 interrupt frequency = Fsys / 2 / TM0PSC / (256-TM0RLD),

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	INT2IE	INT1IE	INT0IE
R/W	R/W	–	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	–	0	0	0	0	0	0

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

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

0Ch.4 **TM0IF:** Timer0 interrupt event pending flag
This bit is set by H/W while Timer0 overflows, 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

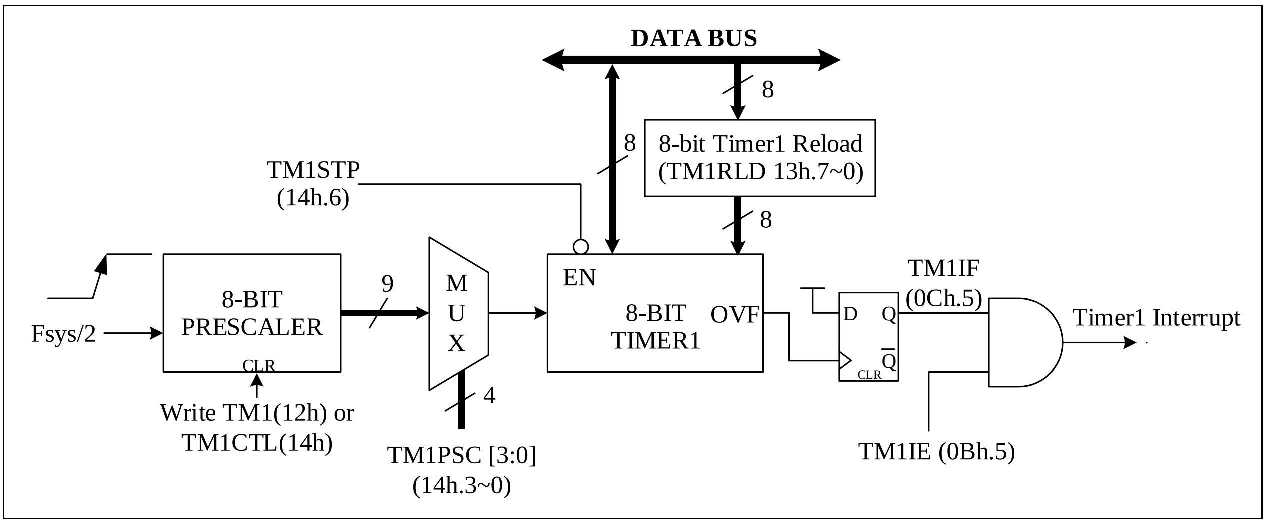
11h.5 **TM0EDG:** Timer0 prescaler counting edge for TM0CKI pin
0: rising edge
1: falling edge

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

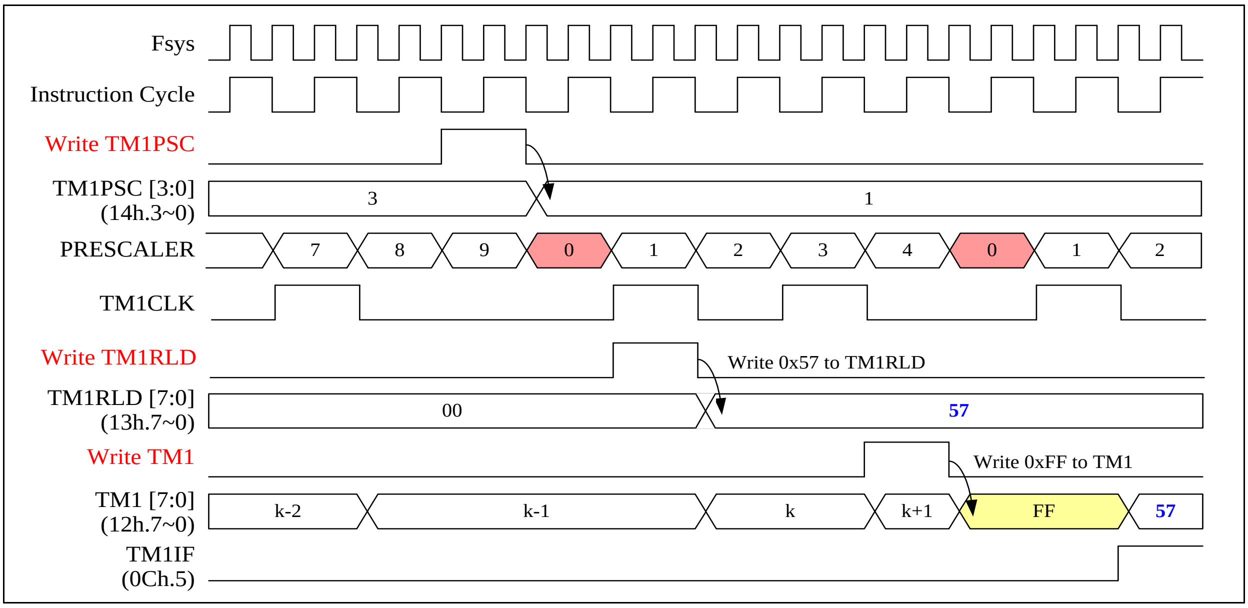
11h.3~0 **TM0PSC:** Timer0 prescaler. Timer0 prescaler clock source divided by
0000: 1 0001: 2 0010: 4 0011: 8
0100: 16 0101: 32 0110: 64 0111: 128
1xxx: 256

6.3 Timer1

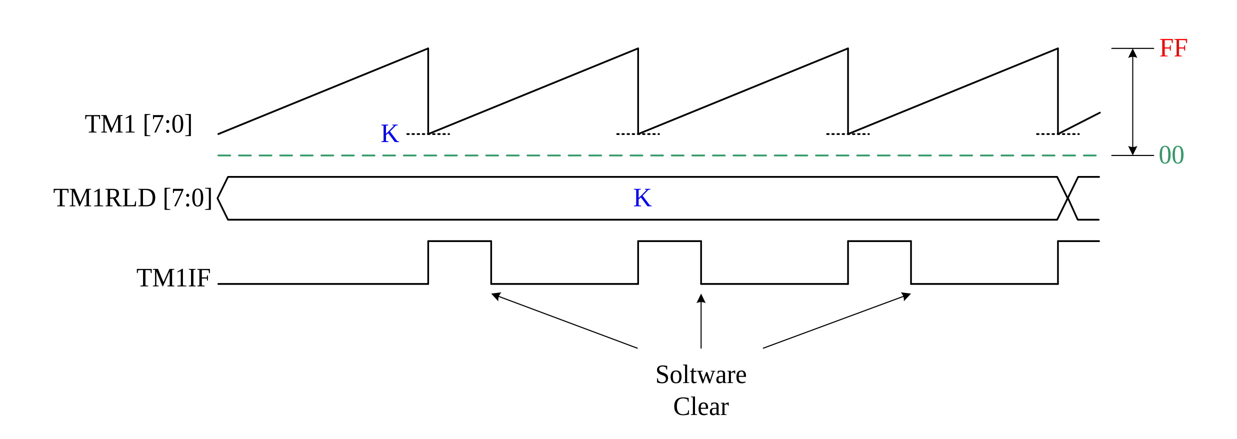
Timer1(TM1) (12h.7~0) is an 8-bit wide register. It can be read or written as any other register. Besides, Timer1 increases itself periodically and automatically reloads a new "offset value" (TM1RLD) while it rolls over based on the pre-scaled instruction clock (Fsys/2). The Timer1 increase rate is determined 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 = 93 KHz / 2 = 46.5 KHz

```
; Setup Timer1 clock source and divider
        MOVLW     00000011b
        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 = 46.5 KHz, TM1PSC = div 8, TM1RLD = 255
Timer1 interrupt frequency = 46.5 KHz / 2 / 8 / (256-255) = 2.906 KHz

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

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

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

0Ch.5 **TM1IF:** Timer1 interrupt event pending flag
This bit is set by H/W while Timer1 overflows, 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

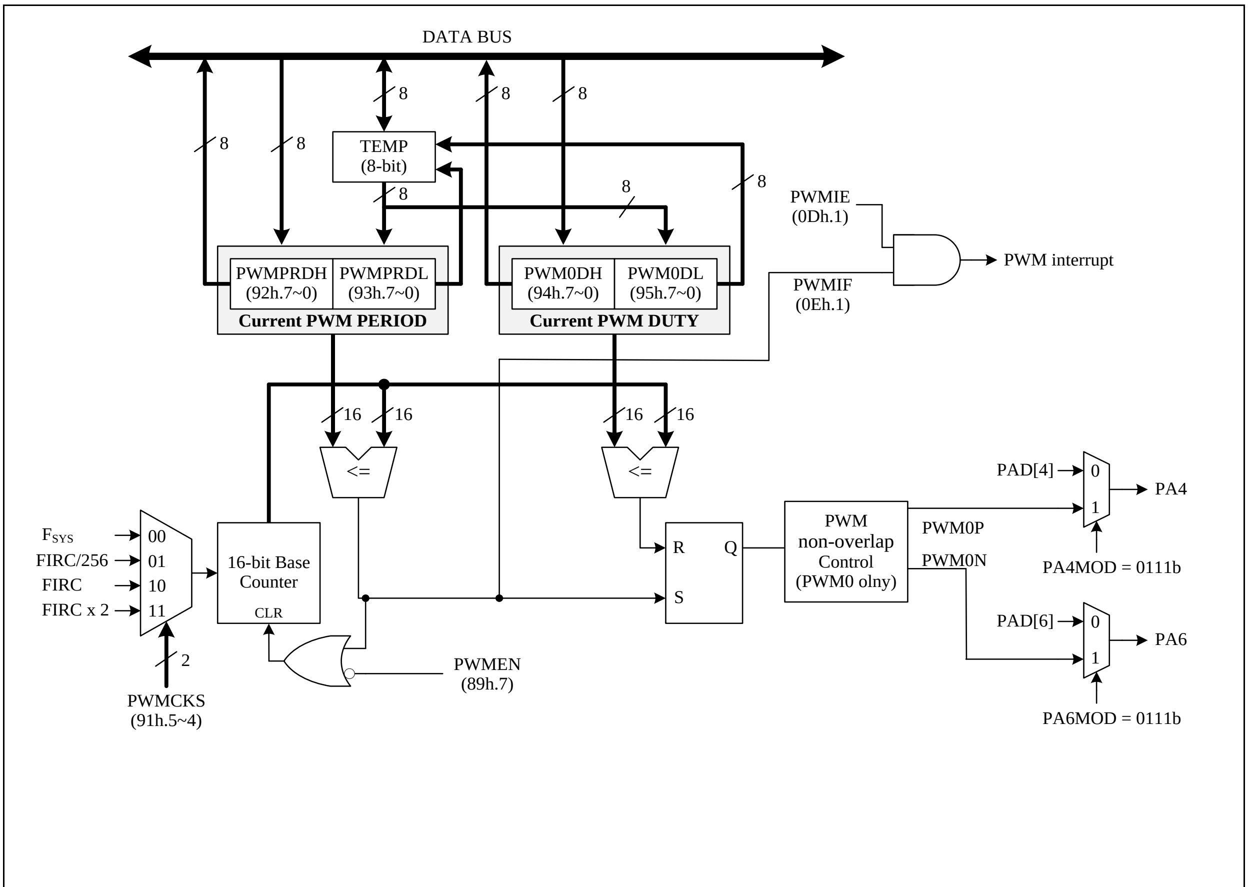
6.4 PWM: 16 bits PWM

There are six PWMs in this chip. PWM0~PWM5 have independent 16-bit duty control register, and share a set of 16-bit period register. The PWM can generate varies frequency waveform with 65536 duty resolution on the basis of the PWM clock. The PWM clock can select Fsys, FIRC/256, FIRC (16 MHz), or FIRC*2 (32 MHz) as its clock source. The following takes PWM0 as an example for description.

The 16-bit PWMPRD, PWM0D registers both have a low byte and high byte structure. The high bytes can be directly accessed, but the low bytes can only be accessed via an internal 8-bit buffer, reading or writing to 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 write or read 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***.

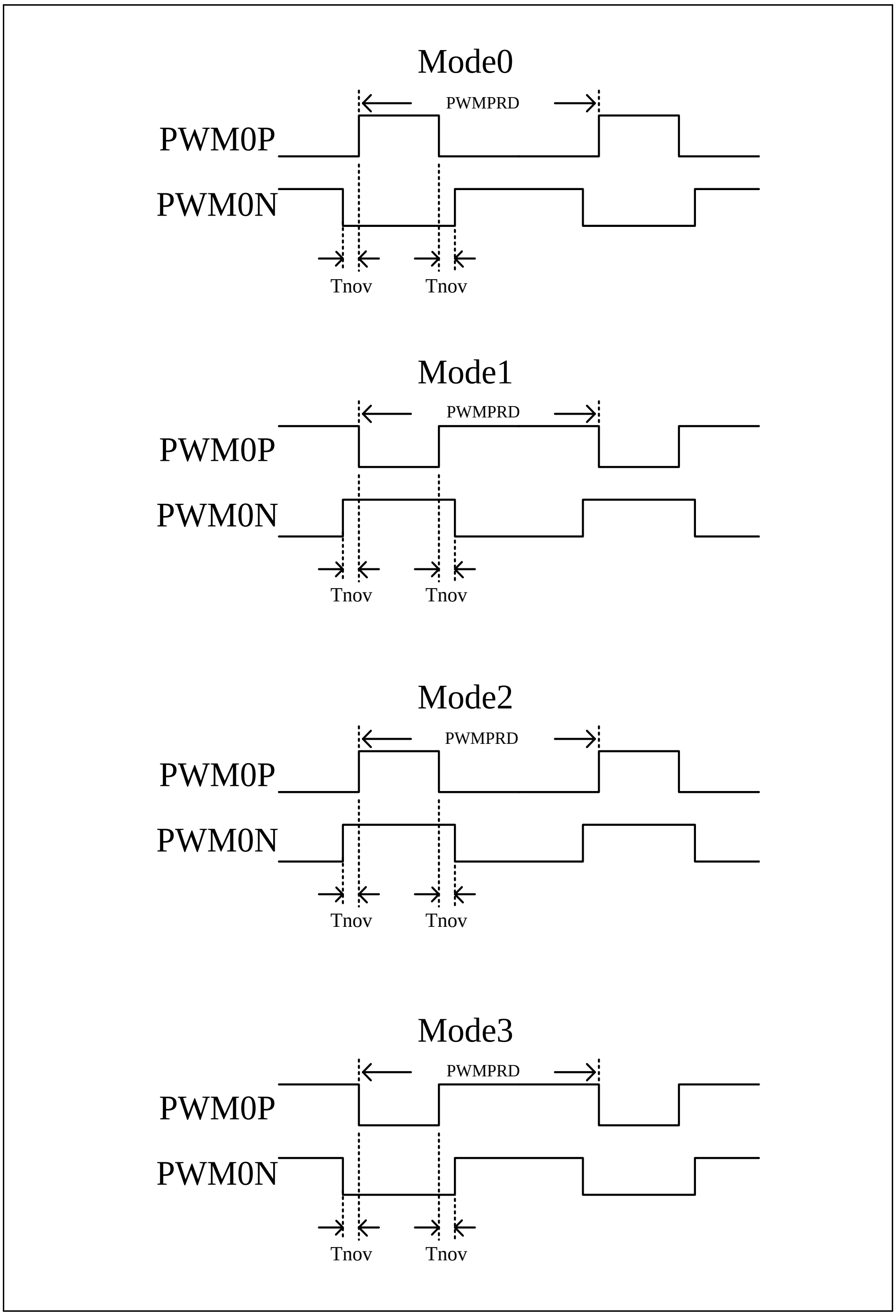
If PWMEN is cleared, the PWM0~5 will be cleared and stopped, otherwise the PWM0~5 remain running. The PWM0 structure is shown as follow. The PWM0 duty cycle can be changed by writing to PWM0DH and PWM0DL. The PWM0 output signal resets to a low level whenever the 16-bit base counter matches the 16-bit PWM0 duty register {PWM0DH, PWM0DL}. The PWM0 period can be set by writing the period value to the PWMPRDH and PWMPRDL registers. After writing the PWM0DH or PWMPRDH register, H/W will update PWM period and duty immediately. PWM0~5 share an interrupt flag, and an interrupt flag is generated at the end of the period.

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



PWM0 Block Diagram

Only PWM0 can be output via PWM0P and PWM0N with four different modes. The edges of the PWM pulse can be separated with 16 different dead-zone(non-overlap) clocks intervals (Tnov). The width of Tnov can be selected by PWM0DZ (89h.3~0) within 0~15 PWM clock. The default output form is Mode0. The waveforms of the four output modes are shown below.



PWM0 Waveform Modes

◇ Example:

```
; 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   xx10xxxxb       ;
        MOVWX   OPTION2         ; FIRC 16 MHz as PWM clock source

; Setup PWM0 period and duty setting
        MOVLW   FFh
        MOVWX   PWMPRDL         ; write sequence: PWMPRDL then PWMPRDH
        MOVLW   7Fh
        MOVWX   PWMPRDH         ; Set PWM 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   10000000b       ; 89h.7 = 1, PWM0 enable
        MOVWX   PWMCTL          ; 89h.5~4 = 0, PWM0 Mode0 output
                                ; 89h.3~0 = 0, PWM0 dead-zone(non-overlap) output
                                ; disable
```

Example:

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

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

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

0Dh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTIE1	–	PCIE	–	–	–	–	PWMIE	LVDIE
R/W	–	R/W	–	–	–	–	R/W	R/W
Reset	–	0	–	–	–	–	0	0

0Dh.1 **PWMIE:** PWM interrupt enable
0: disable
1: enable

0Eh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTIF1	–	PCIF	–	–	–	–	PWMIF	LVDIF
R/W	–	R/W	–	–	–	–	R/W	R/W
Reset	–	0	–	–	–	–	0	0

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

<table>
<tr><td>89h</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>PWMCTL</td><td>PWMEN</td><td>–</td><td colspan="2">PWM0OM</td><td colspan="4">PWM0DZ</td></tr>
<tr><td>R/W</td><td>R/W</td><td>–</td><td colspan="2">R/W</td><td colspan="4">R/W</td></tr>
<tr><td>Reset</td><td>0</td><td>–</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr>
</table>

89h.7 **PWMEN:** PWM0~5 enable
0: disable
1: enable

89h.5~4 **PWM0OM:** PWM0 output mode select
00: Mode0
01: Mode1
10: Mode2
11: Mode3

89h.3~0 **PWM0DZ:** PWM0 dead-zone(non-overlap) control
0000: no dead-zone(non-overlap)
0001: dead-zone(non-overlap) width are 1 PWM clock cycle
0010: dead-zone(non-overlap) width are 2 PWM clock cycles
…
1111: dead-zone(non-overlap) width are 15 PWM clock cycles

<table>
<tr><td>91h</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>OPTION2</td><td>–</td><td>–</td><td colspan="2">PWMCKS</td><td>–</td><td>–</td><td>–</td><td>–</td></tr>
<tr><td>R/W</td><td>–</td><td>–</td><td colspan="2">R/W</td><td>–</td><td>–</td><td>–</td><td>–</td></tr>
<tr><td>Reset</td><td>–</td><td>–</td><td>0</td><td>0</td><td>–</td><td>–</td><td>–</td><td>–</td></tr>
</table>

91h.5~4 **PWMCKS:** PWM clock source select
00: Fsys
01: FIRC/256
10: FIRC (16 MHz)
11: FIRC x 2 (32 MHz). Refer to the graph of minimal operating voltage for PWMCKS=FIRC x 2.

<table>
<tr><td>92h</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>PWMPRDH</td><td colspan="8">PWMPRDH</td></tr>
<tr><td>R/W</td><td colspan="8">R/W</td></tr>
<tr><td>Reset</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td></tr>
</table>

92h.7~0 **PWMPRDH:** PWM0~5 period high byte
write sequence: PWMPRDL then PWMPRDH
read sequence: PWMPRDH then PWMPRDL

<table>
<tr><td>93h</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>PWMPRDL</td><td colspan="8">PWMPRDL</td></tr>
<tr><td>R/W</td><td colspan="8">R/W</td></tr>
<tr><td>Reset</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td></tr>
</table>

93h.7~0 **PWMPRDL:** PWM0~5 period low byte
write sequence: PWMPRDL then PWMPRDH
read sequence: PWMPRDH then PWMPRDL

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
PWM1DH	PWM1DH							
R/W	R/W							
Reset	1	0	0	0	0	0	0	0

96h.7~0 **PWM1DH:** PWM1 duty high byte
write sequence: PWMxDL then PWMxDH
read sequence: PWMxDH then PWMxDL

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

97h.7~0 **PWM1DL:** PWM1 duty low byte
write sequence: PWMxDL then PWMxDH
read sequence: PWMxDH then PWMxDL

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

98h.7~0 **PWM2DH:** PWM2 duty high byte
write sequence: PWMxDL then PWMxDH
read sequence: PWMxDH then PWMxDL

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

99h.7~0 **PWM2DL:** PWM2 duty low byte
write sequence: PWMxDL then PWMxDH
read sequence: PWMxDH then PWMxDL

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

9Ah.7~0 **PWM3DH:** PWM3 duty high byte
write sequence: PWMxDL then PWMxDH
read sequence: PWMxDH then PWMxDL

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

9Bh.7~0 **PWM3DL:** PWM3 duty low byte
write sequence: PWMxDL then PWMxDH
read sequence: PWMxDH then PWMxDL

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

9Ch.7~0 **PWM4DH:** PWM4 duty high byte
write sequence: PWMxDL then PWMxDH
read sequence: PWMxDH then PWMxDL

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

9Dh.7~0 **PWM4DL:** PWM4 duty low byte
write sequence: PWMxDL then PWMxDH
read sequence: PWMxDH then PWMxDL

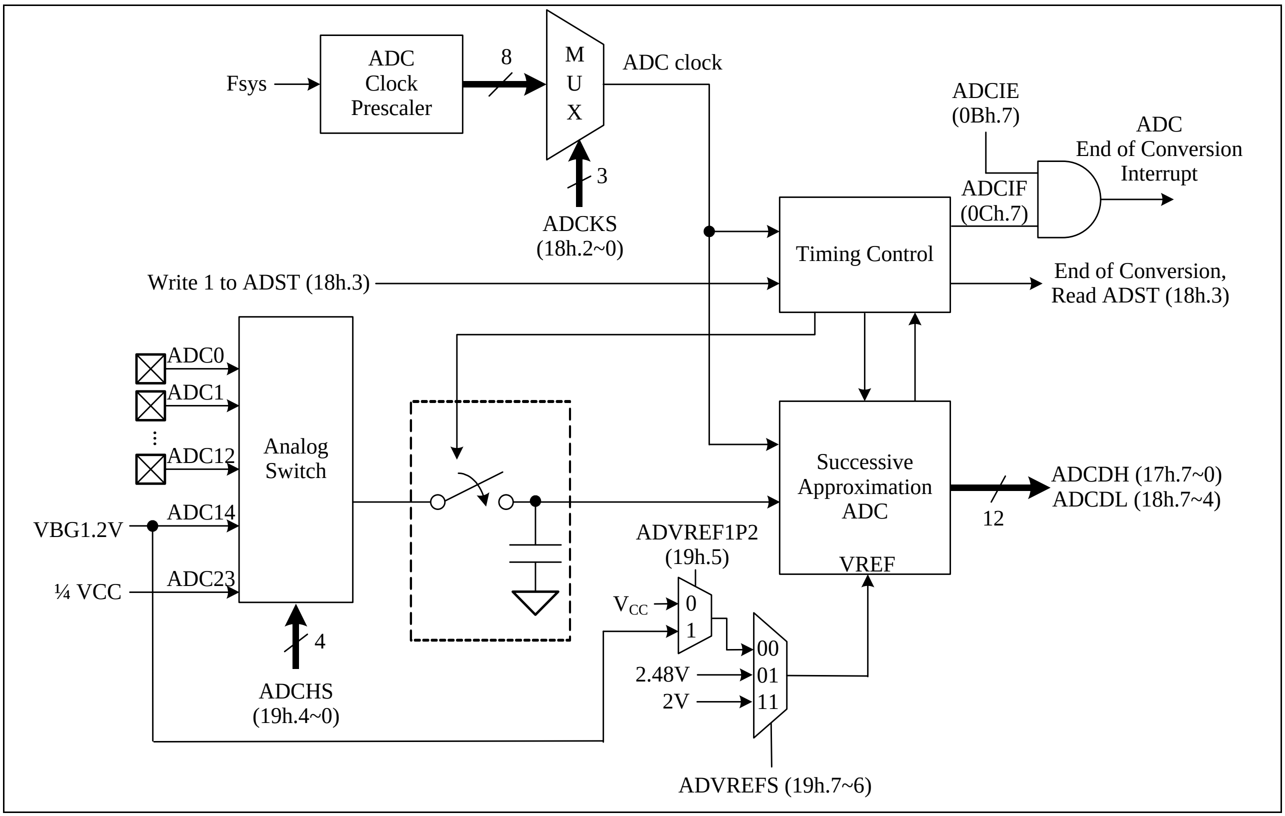
9Eh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWM5DH	PWM5DH							
R/W	R/W							
Reset	1	0	0	0	0	0	0	0

9Eh.7~0 **PWM5DH:** PWM5 duty high byte
write sequence: PWMxDL then PWMxDH
read sequence: PWMxDH then PWMxDL

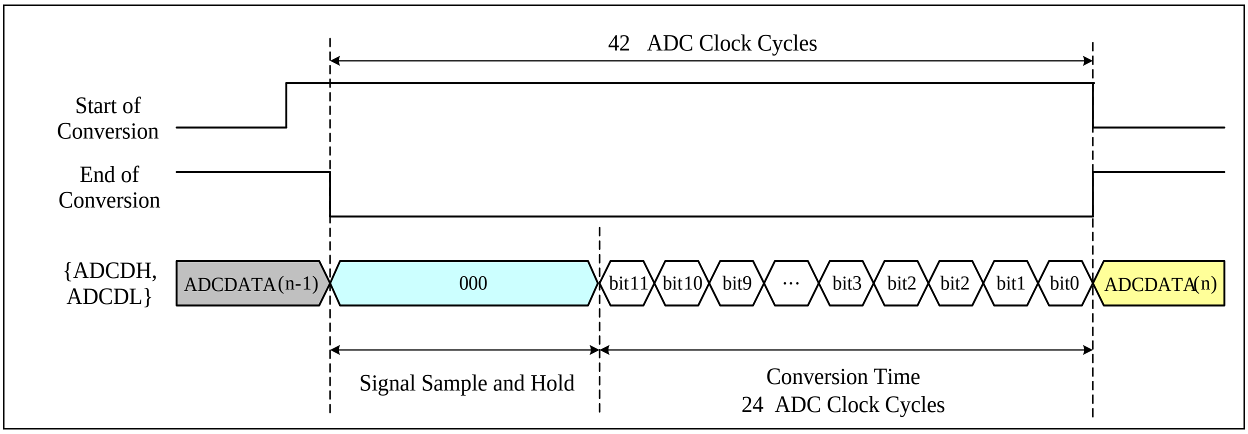
9Fh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWM5DL	PWM5DL							
R/W	R/W							
Reset	0	0	0	0	0	0	0	0

9Fh.7~0 **PWM5DL:** PWM5 duty low byte
write sequence: PWMxDL then PWMxDH
read sequence: PWMxDH then PWMxDL

6.5 Analog-to-Digital Converter



The 12-bit ADC (Analog to Digital Converter) consists of a 15-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 1 MHz. User then launches the ADC conversion by setting the ADST (18h.3) control bit. After end of conversion, H/W automatic clears the ADST (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 ADC23 is 1/4VCC for ADC. ADC reference voltage can be configured as V_{CC} or V_{BG} by ADVREFS (19h.7~6). Furthermore, if ADVREFS is changed to 2.48V 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.



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

WAIT_ADC:
        BTXSC     ADST               ; 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
        …
```

0Bh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTIE	ADCIE	–	TM1IE	TM0IE	WKTIE	INT2IE	INT1IE	INT0IE
R/W	R/W	–	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	–	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	INT2IF	INT1IF	INT0IF
R/W	R/W	–	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	–	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				ADST	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 **ADST:** ADC start bit.
0: H/W clear after end of conversion
1: ADC start conversion
18h.2~0 **ADCKS:** ADC clock frequency selection:

000: Fsys/256	100: Fsys/16
001: Fsys/128	101: Fsys/8
010: Fsys/64	110: Fsys/4
011: Fsys/32	111: Fsys/2

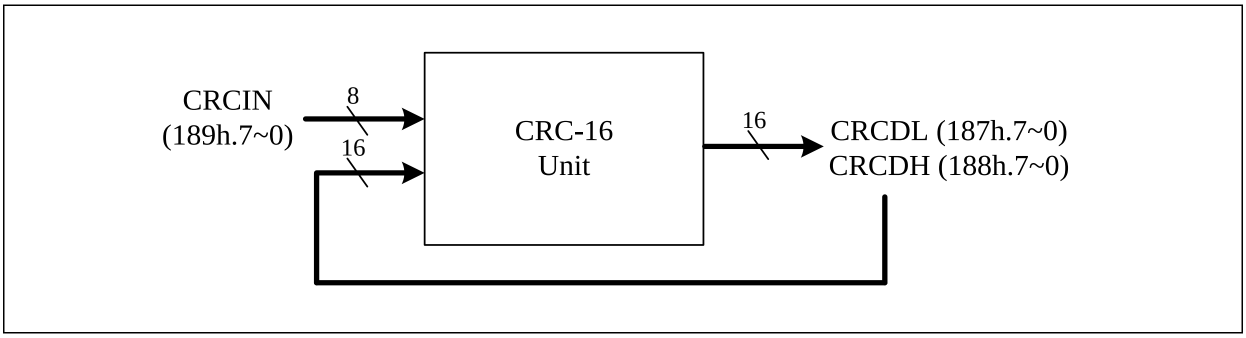
19h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
ADCTL2	ADVREFS		ADVREF1P2	ADCHS				
R/W	R/W		R/W	R/W				
Reset	0	0	0	1	1	1	1	1

19h.7~6 **ADVREFS:** ADC reference voltage and V_{BG} output voltage select
00: ADC reference voltage is V_{CC} or 1.2V V_{BG} according to the value of ADVREF1P2. V_{BG} is 1.20V
01: ADC reference voltage is V_{BG}, V_{BG} is 2.48V
10: Reserved
11: ADC reference voltage is V_{BG}, V_{BG} is 2.00V(This feature can't not be emulated)(Don't use for the selection of DAC's VREF)
19h.5 **ADVREF1P2:** ADC 1.2V reference voltage select
0: ADC reference voltage is V_{CC} when ADVREFS=00. V_{BG} is 1.2V
1: ADC reference voltage is 1.2V V_{BG} when ADVREFS=00. V_{BG} is 1.2V(This feature can't not be emulated)
19h.4~0 **ADCHS:** ADC channel select

00000: ADC0 (PA0)	01000: ADC8 (PB1)
00001: ADC1 (PA1)	01001: ADC9 (PB2)
00010: ADC2 (PA2)	01010: ADC10 (PB4)
00011: ADC3 (PA3)	01011: ADC11 (PB5)
00100: ADC4 (PA4)	01100: ADC12 (PB6)
00101: ADC5 (PA5)	01110: VBG
00110: ADC6 (PA6)	10111: 1/4 VCC
00111: ADC7 (PB0)	others: Reserved

6.6 Cyclic Redundancy Check (CRC)

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



CRC16 Block Diagram

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

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

187h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
CRCDL	CRCDL							
R/W	R/W							
Reset	1	1	1	1	1	1	1	1

187h.7~0 **CRCDL:** 16-bit CRC checksum data bit 7~0

188h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
CRCDH	CRCDH							
R/W	R/W							
Reset	1	1	1	1	1	1	1	1

188h.7~0 **CRCDH:** 16-bit CRC checksum data bit 15~8

189h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
CRCIN	CRCIN							
W	W							
Reset	–	–	–	–	–	–	–	–

189h.7~0 **CRCIN:** CRC data input, write this register to start CRC calculation

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	06.6~4	R	-	Port B pin or "data register" state
		W	7	Port B output data register
	06.2~0	R	-	Port B pin or "data register" state
		W	7	Port B output data register
PCLATH (0Ah/8Ah/10Ah/18Ah)				**Function related to: PROGRAM COUNT**
GPR	0A.7~3	R/W	0	General Purpose Register
PCLATH	0A.2~0	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
INT2IE	0B.2	R/W	0	INT2 pin (PA7) interrupt enable 0: disable 1: enable
INT1IE	0B.1	R/W	0	INT1 pin (PA1) interrupt enable 0: disable 1: enable
INT0IE	0B.0	R/W	0	INT0 pin (PA3) interrupt enable 0: disable 1: enable
INTIF (0Ch)				**Function related to: Interrupt Flag**

Name	Address	R/W	Rst	Description
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
		W	0	write F7h to INTIF will clear this flag
INT2IF	0C.2	R	-	INT2 (PA7) interrupt event pending flag, set by H/W at INT2 pin's falling edge
		W	0	write FBh to INTIF will clear this flag
INT1IF	0C.1	R	-	INT1 (PA1) interrupt 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) interrupt event pending flag, set by H/W at INT0 pin's falling/rising edge
		W	0	write FEh to INTIF will clear this flag
INTIE1 (0Dh)				**Function related to: Interrupt Enable**
PCIE	0D.6	R/W	0	All port pin-change wake-up interrupt enable 0: disable 1: enable
PWMIE	0D.1	R/W	0	PWM 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**
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
PWMIF	0E.1	R	-	PWM interrupt event pending flag, set by H/W after PWM 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
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

Name	Address	R/W	Rst	Description
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 0000: Disable 0100 : 2.11V 1000: 2.61V 1100: 3.12V 0001: 1.73V 0101: 2.23V 1001: 2.74V 1101: 3.25V 0010: 1.85V 0110: 2.36V 1010: 2.87V 1110: 3.37V 0011: 1.98V 0111: 2.49V 1011: 2.99V 1111: 3.50V
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
ADST	18.3	R/W	0	ADC start bit. 0: H/W clear after end of conversion 1: ADC start conversion
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	19.7~6	R/W	00	ADC reference voltage and V_{BG} output voltage select 00: ADC reference voltage is V_{CC} or 1.2V V_{BG} according to the value of ADVREF1P2. V_{BG} is 1.20V 01: ADC reference voltage is V_{BG}, V_{BG} is 2.48V

Name	Address	R/W	Rst	Description
				10: Reserved 11: ADC reference voltage is V_{BG}, V_{BG} is 2.00V(This feature can't not be emulated) (Don't use for the selection of DAC's VREF)
ADVREF1P2	19.5	R/W	0	ADC 1.2V reference voltage select 0: ADC reference voltage is V_{CC} when ADVREFS=00. V_{BG} is 1.2V 1: ADC reference voltage is 1.2V V_{BG} when ADVREFS=00. V_{BG} is 1.2V(This feature can't not be emulated)
ADCHS	19.4~0	R/W	1F	ADC channel select 00000: ADC0 (PA0) 01000: ADC8 (PB1) 00001: ADC1 (PA1) 01001: ADC9 (PB2) 00010: ADC2 (PA2) 01010: ADC10 (PB4) 00011: ADC3 (PA3) 01011: ADC11 (PB5) 00100: ADC4 (PA4) 01100: ADC12 (PB6) 00101: ADC5 (PA5) 01110: VBG 00110: ADC6 (PA6) 10111: 1/4 VCC 00111: ADC7 (PB0) 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 trigger 1: rising edge trigger
INT1EDG	81.5	R/W	0	INT1 pin interrupt edge selection 0: falling edge trigger 1: rising edge trigger
WDTPSC	81.3~2	R/W	3	WDT period selections: 00: 91ms 01: 183ms 10: 732ms 11: 1463ms @5V
WKTPSC	81.1~0	R/W	3	WKT period selections: 00: 11ms 01: 23ms 10: 46ms 11: 91ms @5V
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
PA6MOD	88.3~0	R/W	1	PA6 I/O mode control
PWMCTL (89h)				**Function related to: PWM0**
PWMEN	89.7	R/W	0	PWM Clock Enable 0: Disable 1: Enable
PWM0OM	89.5~4	R/W	0	PWM0 output mode 00: Mode0 01: Mode1 10: Mode2

Name	Address	R/W	Rst	Description
				11: Mode3
PWM0DZ	89.3~0	R/W	0	PWM0 dead-zone(non-overlap) control 0000: no dead-zone(non-overlap) 0001: dead-zone(non-overlap) width are 1 PWM clock cycle 0010: dead-zone(non-overlap) width are 2 PWM clock cycles … 1111: dead-zone(non-overlap) width are 15 PWM clock cycles
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**
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**
PB6MOD	8F.3~0	R/W	1	PB6 I/O mode control
OPTION2 (91h)				**Function related to: PWM0/INT2/INT1/INT0**
PWMCKS	91.5~4	R/W	00	PWM Clock Source 00: Fsys 01: FIRC/256 10: FIRC (16 MHz) 11: FIRC*2 (32 MHz). Refer to the graph of minimal operating voltage for PWMCKS=FIRC x 2.
PWMPRDH (92h)				**Function related to: PWM**
PWMPRDH	92.7~0	R/W	FF	PWM Period bit 15~8
PWMPRDL (93h)				**Function related to: PWM**
PWMPRDL	93.7~0	R/W	FF	PWM Period bit 7~0
PWM0DH (94h)				**Function related to: PWM0**
PWM0DH	94.7~0	R/W	80	PWM0 Duty bit 15~8
PWM0DL (95h)				**Function related to: PWM0**
PWM0DL	95.7~0	R/W	00	PWM0 Duty bit 7~0
PWM1DH (96h)				**Function related to: PWM1**
PWM1DH	96.7~0	R/W	80	PWM1 Duty bit 15~8
PWM1DL (97h)				**Function related to: PWM1**
PWM1DL	97.7~0	R/W	00	PWM1 Duty bit 7~0
PWM2DH (98h)				**Function related to: PWM2**
PWM2DH	98.7~0	R/W	80	PWM2 Duty bit 15~8
PWM2DL (99h)				**Function related to: PWM2**
PWM2DL	99.7~0	R/W	00	PWM2 Duty bit 7~0
PWM3DH (9Ah)				**Function related to: PWM3**
PWM3DH	9A.7~0	R/W	80	PWM3 Duty bit 15~8
PWM3DL (9Bh)				**Function related to: PWM3**
PWM3DL	9B.7~0	R/W	00	PWM3 Duty bit 7~0
PWM4DH (9Ch)				**Function related to: PWM4**
PWM4DH	9C.7~0	R/W	80	PWM4 Duty bit 15~8
PWM4DL (9Dh)				**Function related to: PWM4**
PWM4DL	9D.7~0	R/W	00	PWM4 Duty bit 7~0

Name	Address	R/W	Rst	Description
PWM5DH (9Eh)				**Function related to: PWM5**
PWM5DH	9E.7~0	R/W	80	PWM5 Duty bit 15~8
PWM5DL (9Fh)				**Function related to: PWM5**
PWM5DL	9F.7~0	R/W	00	PWM5 Duty bit 7~0
User Data Memory				
RAM	A0~BF	R/W	-	RAM Bank1 area (32 Bytes)
PINMOD (105h)				**Function related to: IO Port**
Reserved	105.5	R	x	read as unknown after reset
HSINK	105.2	R/W	1	All IO port high sink current enable 0: low sink current 1: high sink current. PA7 has no high-sink capability.
Reserved	105.1	R/W	0	must be kept at 0
Reserved	105.0	R/W	0	must be kept at 0
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.2~0	R	0	Program Counter data bit 10~8
BGTRIM (10Eh)				**Function related to: Bandgap**
BGTRIM	10E.4~0	R/W	CFG	VBG 1.2V trim value
IRCF (10Fh)				**Function related to: Internal RC**
IRCF	10F.6~0	R/W	CFG	FIRC trim value
BG2TRIM (111h)				**Function related to: Bandgap**
BG2TRIM	111.7~0	R	CFG	VBG 2V trim value. The users could move this register to BGTRIM for slightly exact 2V VBG. This feature can't be emulated.
RDCTL (113h)				**Function related to: Program ROM**
RDCTL	113.1~0	R/W	02	Read signal delay control for Program ROM 00: 16ns delay for read signal of Program ROM 01: 12ns delay for read signal of Program ROM 10: 8ns delay for read signal of Program ROM 11: 4ns delay for read signal of Program ROM Change this register at slow clock for safety. **The user must switch this register to "4ns" to enhance the performance of minimal operating voltage.** This feature can't be emulated.
User Data Memory				
RAM	120~16F	R/W	-	Don't Use

Name	Address	R/W	Rst	Description
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
CRCDL (187h)				**Function related to: CRC16**
CRCDL	187.7~0	R/W	FF	16-bit CRC checksum data bit 7~0
CRCDH (188h)				**Function related to: CRC16**
CRCDH	188.7~0	R/W	FF	16-bit CRC checksum data bit 15~8
CRCIN (189h)				**Function related to: CRC16**
CRCIN	189.7~0	W	0	CRC data input, write this register to start CRC calculation
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.*

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
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 and TABR
TABRL		0000 0000 0101 0000	2	-	Lookup ROM low data to W and TABR
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

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 — B : PC =LABEL1 LGOTO LOOP — A : CNT =CNT + 1 CONTINUE: — 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

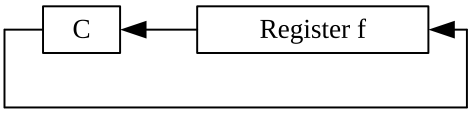
MOVWX	**Move W to "f"**	
Syntax	MOVWX f	
Operands	f : 000h ~ 1FFh	
Operation	(f) ← (W)	
Status Affected	-	
OP-Code	ff00 0000 1fff ffff	
Description	Move data from W register to register 'f'.	
Cycle	1	
Example	MOVWX REG1	B : REG1 =0xFF, W =0x4F A : REG1 =0x4F, W =0x4F

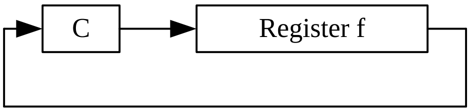
NOP	**No Operation**	
Syntax	NOP	
Operands	-	
Operation	No Operation	
Status Affected	-	
OP-Code	0000 0000 0000 0000	
Description	No Operation	
Cycle	1	
Example	NOP	-

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

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	(see below)

```
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°C)

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	°C
Storage temperature	-65 to +150	

2. DC Characteristics (T_A = 25°C, V_{CC} = 5.0V, unless otherwise specified)

Parameter	Symbol	Conditions		Min.	Typ.	Max.	Unit
Operating Voltage	V_{cc}	Fsys = 16 MHz (FIRC)(RDCTL=4ns) (PWMCKS=FIRC*1)(-40°C ~ 105°C)		2.3	–	5.5	V
		Fsys = 8 MHz (FIRC/2) (RDCTL=4ns) (PWMCKS=FIRC*1) (-40°C ~ 105°C)		1.55	–	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}	All I/O pin	V_{CC} = 5.0V, V_{OH} = 4.5V	6	12.7	–	mA
			V_{CC} = 3.0V, V_{OH} = 2.7V	2.5	5.3	–	
I/O port Sink Current	I_{OL}	All I/O pin except PA7 (HSINK=1)	V_{CC} = 5.0V, V_{OL} = 0.5V	32	63	–	mA
			V_{CC} = 3.0V, V_{OL} = 0.3V	15	29	–	
		All I/O pin (HSINK=0)	V_{CC} = 5.0V, V_{OL} = 0.5V	18	36	–	mA
			V_{CC} = 3.0V, V_{OL} = 0.3V	8	16	–	
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

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	–	3.3	–	mA
			V_{CC} = 3.0V	–	1.9	–	
		FAST mode FIRC 8 MHz	V_{CC} = 5.0V	–	2.3	–	
			V_{CC} = 3.0V	–	1.3	–	
		FAST mode FIRC 4 MHz	V_{CC} = 5.0V	–	1.6	–	
			V_{CC} = 3.0V	–	1.0	–	
		FAST mode FIRC 2 MHz	V_{CC} = 5.0V	–	1.1	–	
			V_{CC} = 3.0V	–	0.69	–	
		SLOW mode SIRC div1 FIRC STOP POR/LVR On	V_{CC} = 5.0V	–	0.058	–	
			V_{CC} = 3.0V	–	0.032	–	
		SLOW mode SIRC div1 FIRC STOP POR/LVR Off	V_{CC} = 5.0V	–	0.028	–	
			V_{CC} = 3.0V	–	0.017	–	
		SLOW mode SIRC div1 FIRC STOP POR/LVR Off ATD Off	V_{CC} = 5.0V	–	0.62	–	
			V_{CC} = 3.0V	–	0.45	–	
		IDLE mode SIRC div1 POR/LVR 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	V_{CC} = 5.0V	–	37.5	–	KΩ
			V_{CC} = 3.0V	–	38.7	–	
POR Voltage	V_{POR}	T_A = 25°C		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°C ~ 105°C	V_{CC} = 3.0 ~ 5.0V	-5%	16	+2%	MHz
	T_A = -40°C ~ 105°C	V_{CC} = 4.0 V	-3%	16	+1.5%	
	T_A = 0°C ~ 70°C	V_{CC} = 4.0 V	-2%	16	+1.5%	
	T_A = 25°C	V_{CC} = 3.0 ~ 5.0 V	-1.2%	16	+1.2%	
SIRC Frequency	T_A = 25°C	V_{CC} = 5.0 V		92.8		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 %	–	11	–	µs
WDT time	V_{CC} = 5.0 V, WDTPSC = 11b	–	1463	–	ms
WKT time	V_{CC} = 5.0 V, WKTPSC = 11b	–	91	–	ms
CPU start up time	V_{CC} = 5.0 V	–	21	–	ms

5. LVR Circuit Characteristics (T_A = 25°C)

Parameter	Symbol	Condition	Min.	Typ.	Max.	Unit
LVR Voltage	LVR_{th}	T_A = 25°C	–	1.73	–	V
			–	1.85	–	
			–	1.98	–	
			–	2.11	–	
			–	2.23	–	
			–	2.36	–	
			–	2.49	–	
			–	2.61	–	
			–	2.74	–	
			–	2.87	–	
			–	2.99	–	
			–	3.12	–	
			–	3.25	–	
			–	3.37	–	
			–	3.50	–	
LVR Hysteresis Window	V_{HYS_LVR}	T_A = 25°C	–	0	–	mV
Low Voltage Detection time	T_{LVR}	T_A = 25°C	100	–	–	µs

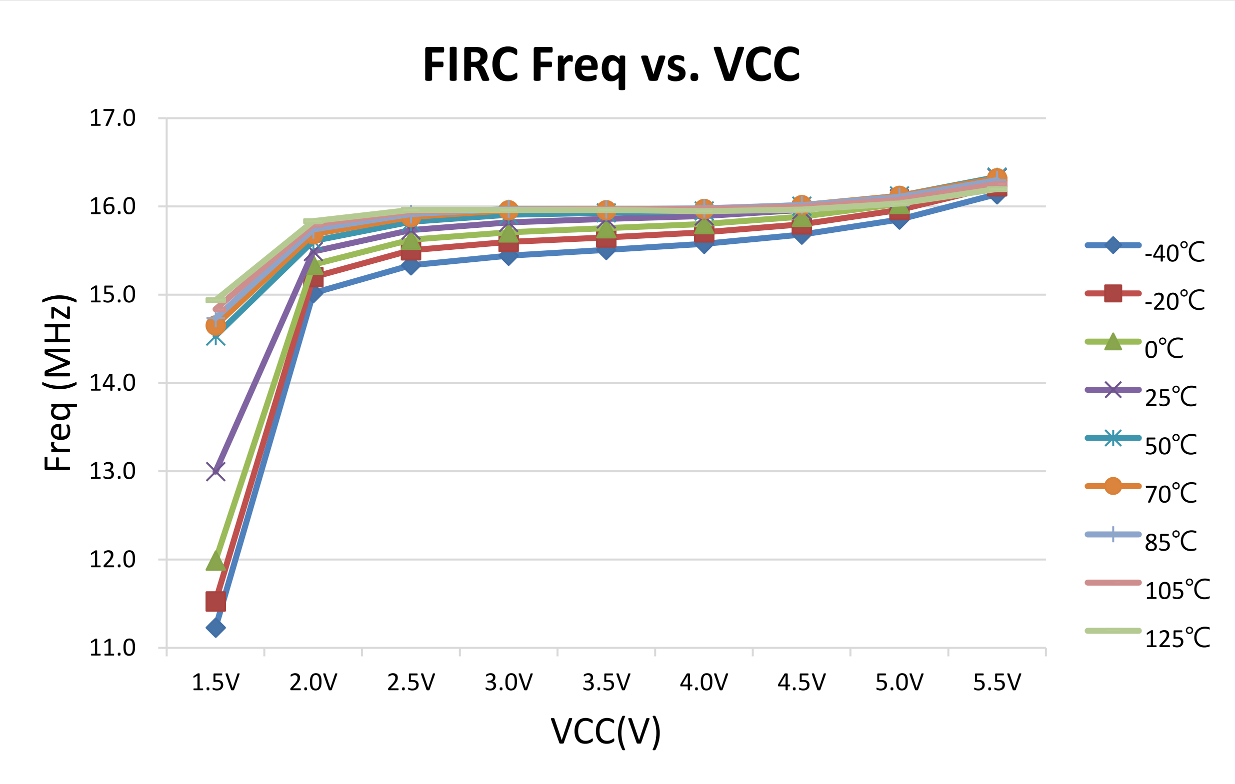
6. LVD Circuit Characteristics (T_A = 25°C)

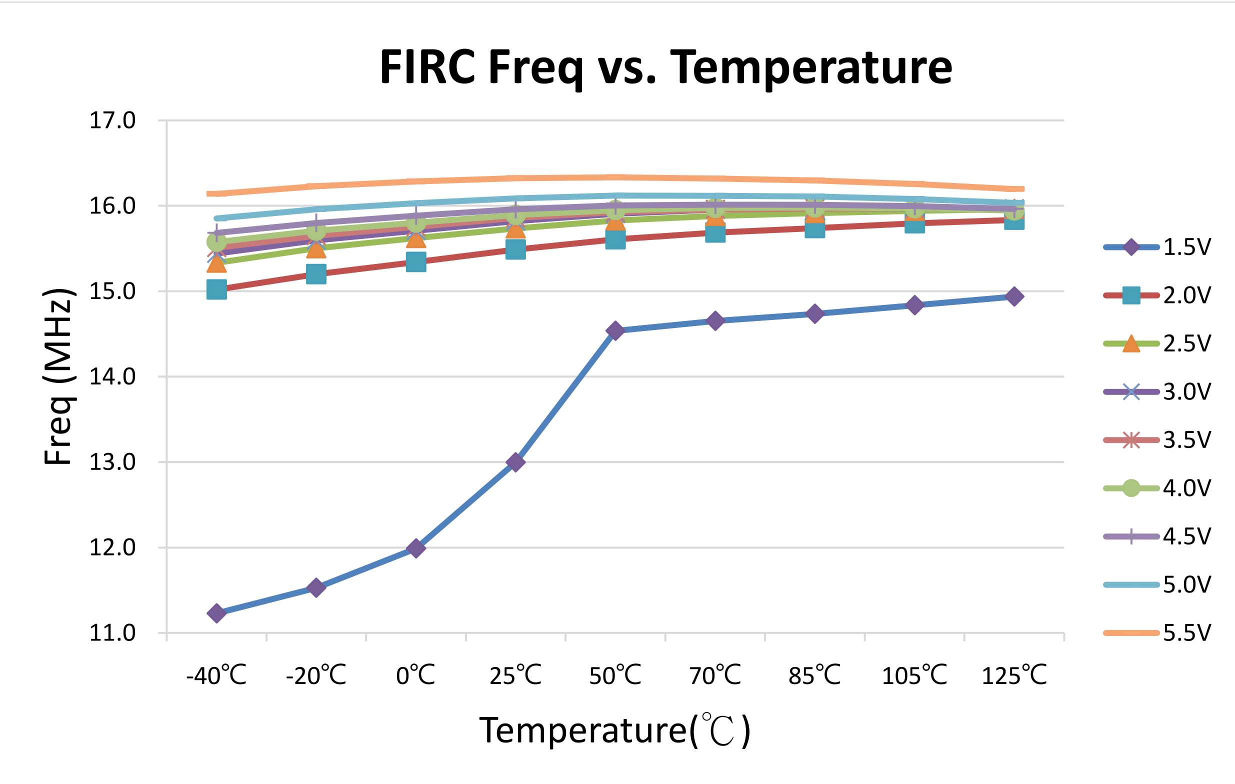
Parameter	Symbol	Condition	Min.	Typ.	Max.	Unit
LVD Voltage	LVD_{th}	T_A = 25°C	–	1.73	–	V
			–	1.85	–	
			–	1.98	–	
			–	2.11	–	
			–	2.23	–	
			–	2.36	–	
			–	2.49	–	
			–	2.61	–	
			–	2.74	–	
			–	2.87	–	
			–	2.99	–	
			–	3.12	–	
			–	3.25	–	
			–	3.37	–	
			–	3.50	–	
LVD Hysteresis Window	V_{HYS_LVD}	LVDHYS = 0	–	0	–	mV
		LVDHYS = 1	–	100	–	
Low Voltage Detection time	T_{LVD}	T_A = 25°C	100	–	–	µs

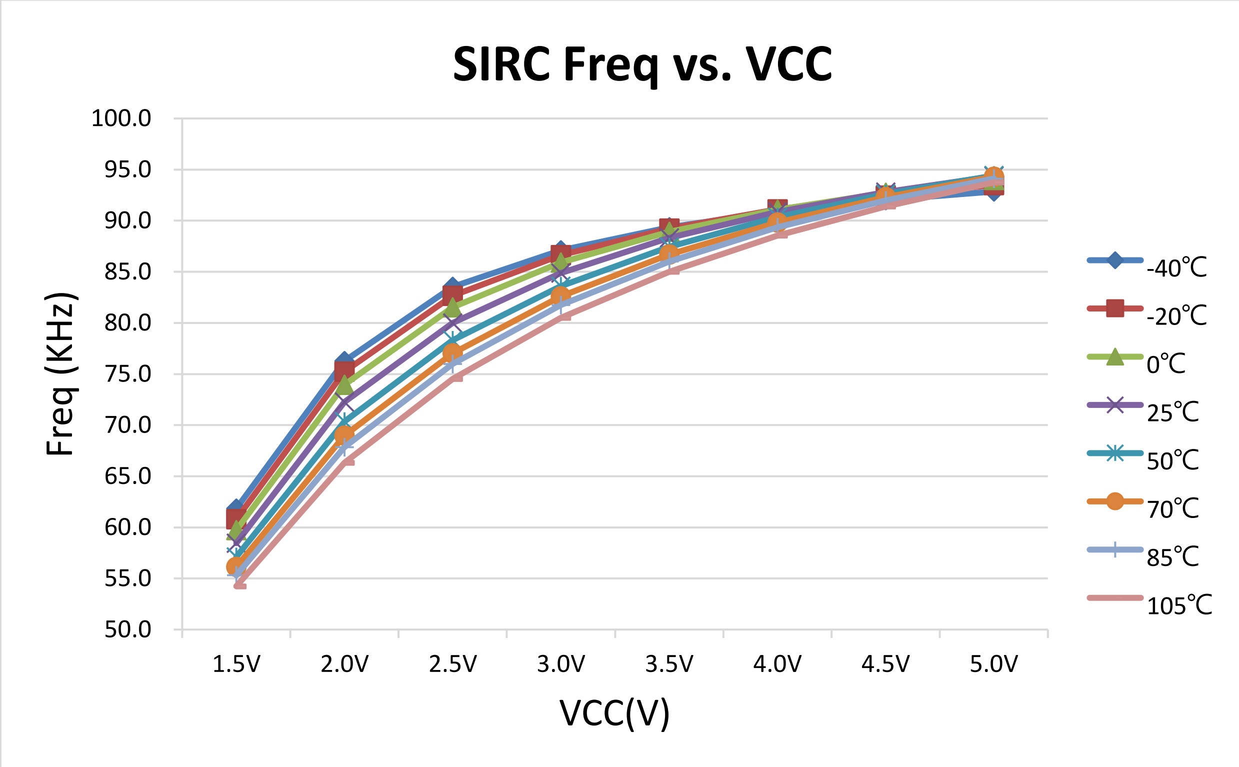
7. ADC Electrical Characteristics (T_A = 25°C, V_{CC} = 3.0V to 5.5V, V_{SS} = 0V)

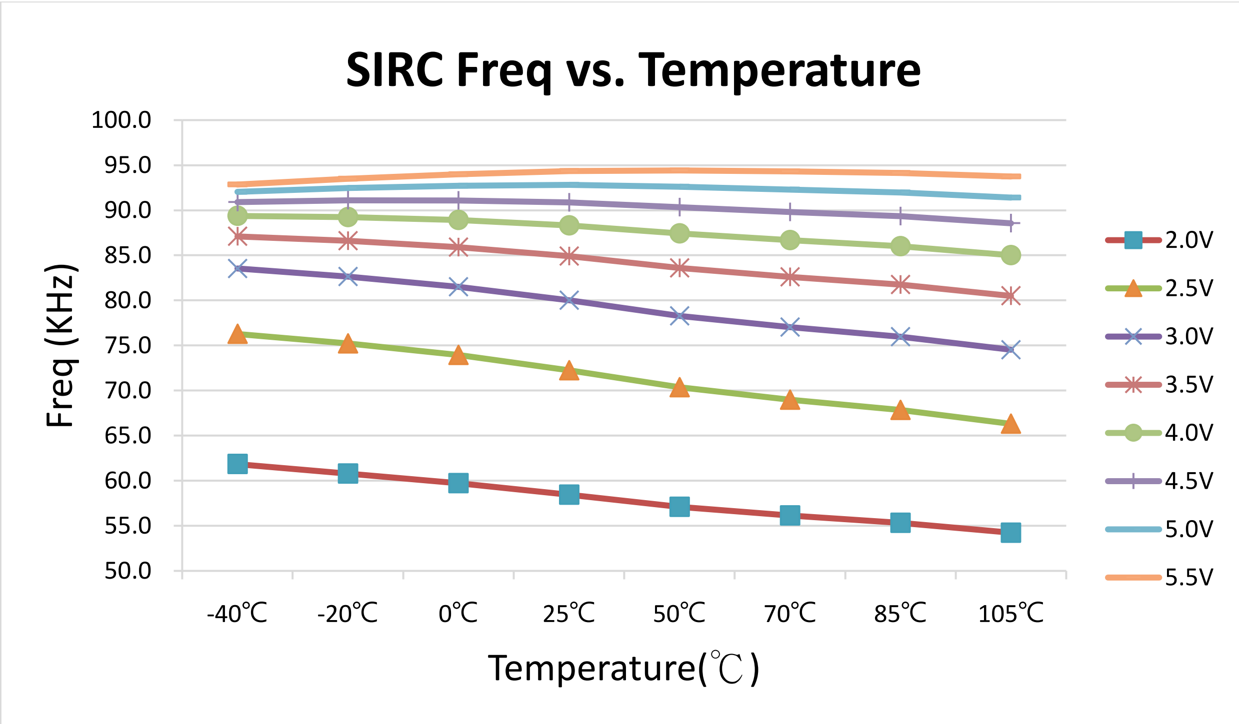
Parameter	Conditions	Min.	Typ.	Max.	Units
Total Accuracy	V_{CC} = 5.0V, V_{SS} = 0V, F_{ADC} = 1 MHz	–	±3	–	LSB
Integral Non-Linearity		–	±3.2	–	
Differential Non-Linearity		–	±1	±4	
Max Input Clock freq. (F_{ADC})	Source impedance (Rs<10K ohm)	–	–	2	MHz
	Source impedance (Rs<20K ohm)	–	–	1	
	Source impedance (Rs<50K ohm)	–	–	0.5	
	Source is VBG (ADCHS=01110b)	–	–	2	
Conversion Time	F_{ADC} = 1 MHz (Include sample and hold time)	–	42	–	μs
Conversion Current	Vcc=5V, ADVREFS=00b, ADVREF1P2=0	–	0.45	–	mA
	Vcc=4V, ADVREFS=01b	–	0.6	–	
BandGap Voltage Reference (1.2V V_{BG}) ADC reference voltage (V_{REF}) (ADVREFS=00b, ADVREF1P2=1b) (No power disturbance)	25ºC, V_{CC} = 3.0V~5.0V	-1.5%	1.20	+1.5%	V
	25ºC~105ºC, V_{CC} = 3.0V~5.0V	-2%	1.20	+2%	V
	-20ºC~105ºC, V_{CC} = 3.0V~5.0V	-2.5%	1.20	+2.5%	V
BandGap Voltage Reference (2.48V V_{BG}) ADC reference voltage (V_{REF}) (ADVREFS=01b) (No power disturbance)	25ºC, V_{CC} = 3.0V~5.2V	-2%	2.48	+2%	V
	-20ºC~105ºC, V_{CC} = 3.0V~5.2V	-2.5%	2.48	+2.5%	V
BandGap Voltage Reference (2V V_{BG}) ADC reference voltage (V_{REF}) (ADVREFS=11b) (No power disturbance)	25ºC, V_{CC} = 3.0V~5.2V	-2%	2	+2%	V
	-20ºC~105ºC, V_{CC} = 3.0V~5.2V	-2.5%	2	+2.5%	V
V_{CC}/4 reference voltage	25ºC, V_{CC} = 3.0V~5.5V	-1%	0.25V_{CC}	+1%	V
Input Voltage	–	V_{SS}	–	V_{REF}	V

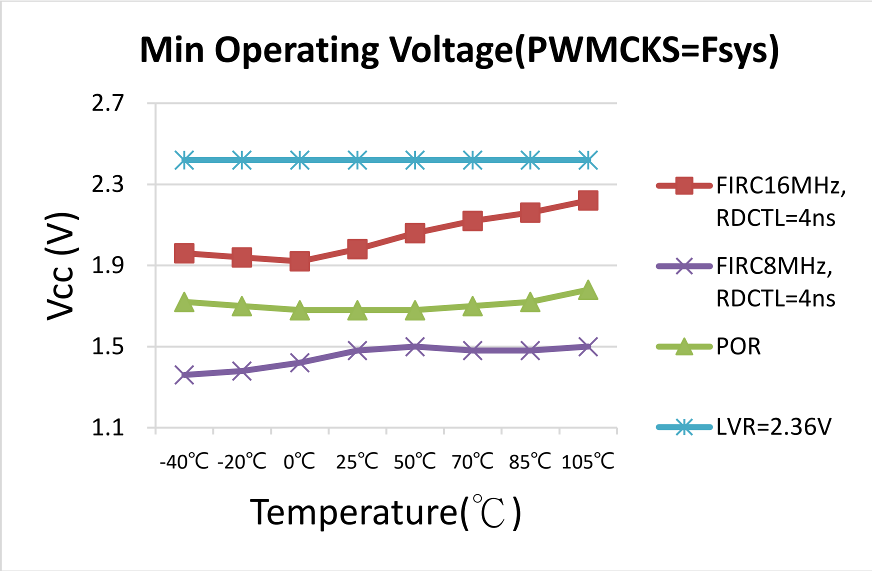
8. Characteristics Graphs



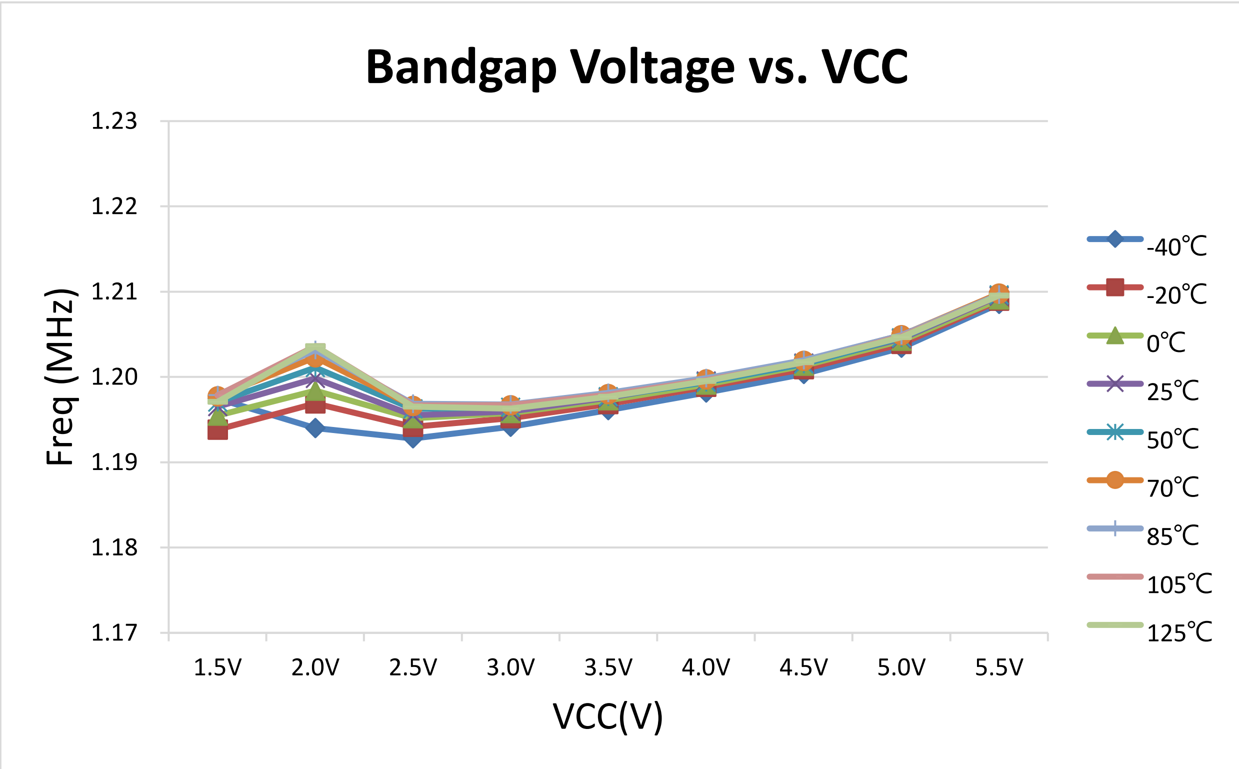



SIRC Freq vs. VCC
Freq (KHz)
100.0
95.0
90.0
85.0
80.0
75.0
70.0
65.0
60.0
55.0
50.0
1.5V
2.0V
2.5V
3.0V
3.5V
4.0V
4.5V
5.0V
VCC(V)
-40℃
-20℃
0℃
25℃
50℃
70℃
85℃
105℃


SIRC Freq vs. Temperature
Freq (KHz)
100.0
95.0
90.0
85.0
80.0
75.0
70.0
65.0
60.0
55.0
50.0
-40℃
-20℃
0℃
25℃
50℃
70℃
85℃
105℃
Temperature(℃)
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.**



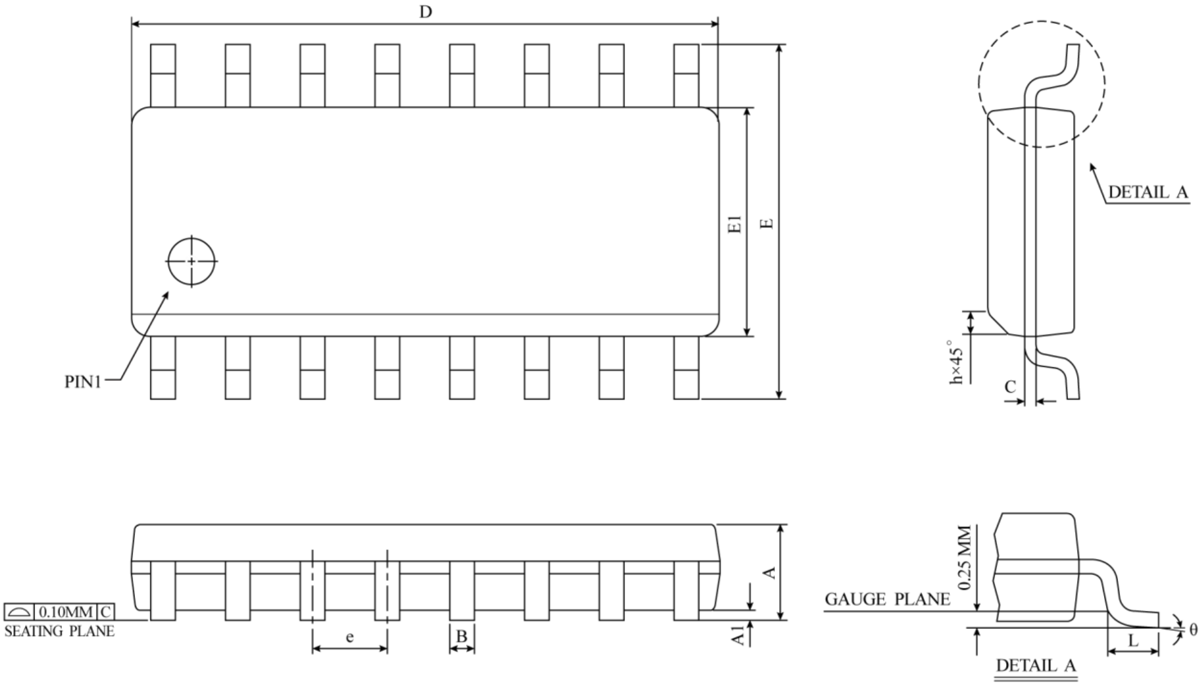
PACKAGING INFORMATION

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
TM56M152A-MTP-16	SOP 16-pin (150 mil)
TM56M152A-MTP-15	SOP 14-pin (150 mil)
TM56M152A-MTP-53	MSOP 10-pin (118 mil)
TM56M152A-MTP-14	SOP 8-pin (150 mil)
TM56M152A-MTP-96	QFN 16-pin (3*3*0.75 - 0.5mm)
TM56M152A-MTP-G3	DFN 10-pin (2*2*0.75 - 0.4mm)
TM56M152A-MTP-F7	SOT23 8-pin

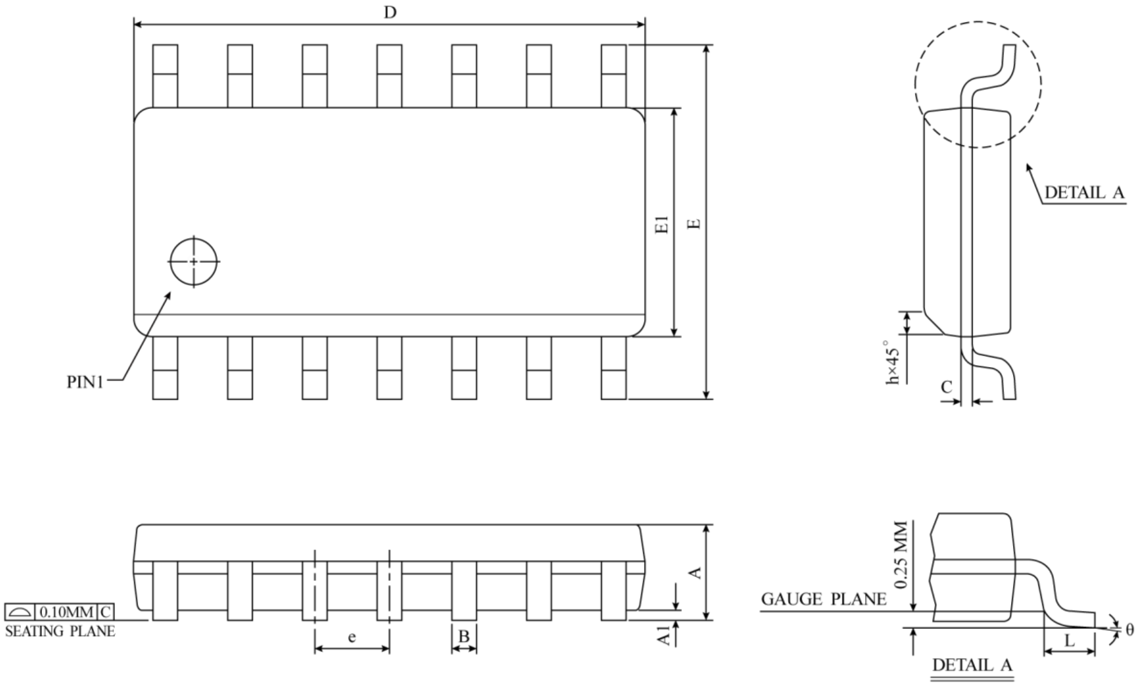
SOP-16 (150 mil) Package Dimension



SYMBOL	DIMENSION IN MM			DIMENSION IN INCH		
	MIN	NOM	MAX	MIN	NOM	MAX
A	1.35	1.55	1.75	0.0532	0.0610	0.0688
A1	0.10	0.18	0.25	0.0040	0.0069	0.0098
B	0.33	0.42	0.51	0.0130	0.0165	0.0200
C	0.19	0.22	0.25	0.0075	0.0087	0.0098
D	9.80	9.90	10.00	0.3859	0.3898	0.3937
E	5.80	6.00	6.20	0.2284	0.2362	0.2440
E1	3.80	3.90	4.00	0.1497	0.1536	0.1574
e	1.27 BSC			0.050 BSC		
h	0.25	0.38	0.50	0.0099	0.0148	0.0196
L	0.40	0.84	1.27	0.0160	0.0330	0.0500
θ	0°	4°	8°	0°	4°	8°
JEDEC	MS-012 (AC)					

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

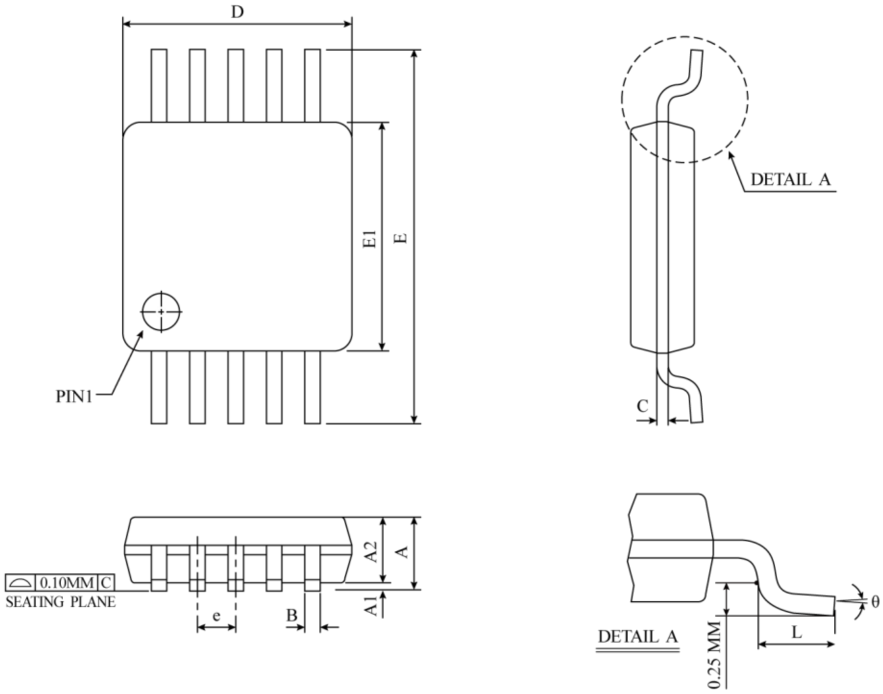
SOP-14 (150 mil) Package Dimension



SYMBOL	DIMENSION IN MM			DIMENSION IN INCH		
	MIN	NOM	MAX	MIN	NOM	MAX
A	1.35	1.55	1.75	0.0532	0.0610	0.0688
A1	0.10	0.18	0.25	0.0040	0.0069	0.0098
B	0.33	0.42	0.51	0.0130	0.0165	0.0200
C	0.19	0.22	0.25	0.0075	0.0087	0.0098
D	8.55	8.65	8.75	0.3367	0.3410	0.3444
E	5.80	6.00	6.20	0.2284	0.2362	0.2440
E1	3.80	3.90	4.00	0.1497	0.1536	0.1574
e	1.27 BSC			0.050 BSC		
h	0.25	0.38	0.50	0.0099	0.0148	0.0196
L	0.40	0.84	1.27	0.0160	0.0330	0.0500
θ	0°	4°	8°	0°	4°	8°
JEDEC	MS-012 (AB)					

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

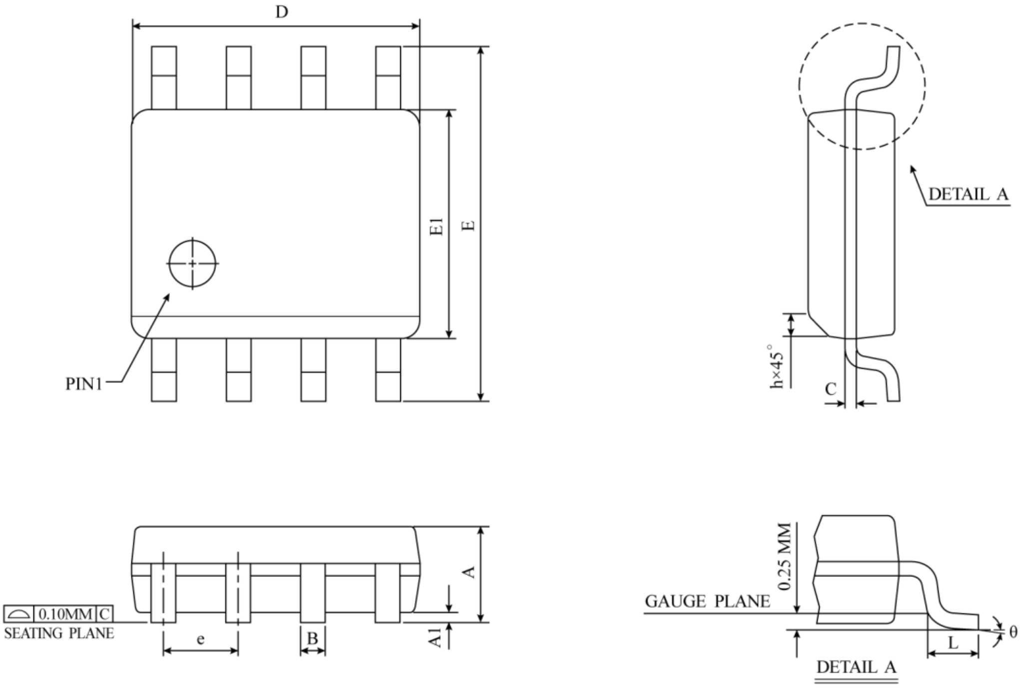
MSOP-10 (118 mil) Package Dimension



SYMBOL	DIMENSION IN MM			DIMENSION IN INCH		
	MIN	NOM	MAX	MIN	NOM	MAX
A	0.81	0.96	1.10	0.032	0.038	0.043
A1	0.05	0.10	0.15	0.002	0.004	0.006
A2	0.75	0.85	0.95	0.030	0.034	0.037
B	0.17	0.22	0.27	0.007	0.009	0.011
C	0.13	0.18	0.23	0.005	0.007	0.009
D	2.90	3.00	3.10	0.114	0.118	0.122
E	4.75	4.90	5.05	0.187	0.193	0.199
E1	2.90	3.00	3.10	0.114	0.118	0.122
e	0.50 BSC			0.020 BSC		
L	0.40	0.55	0.70	0.016	0.022	0.028
θ	0°	3°	6°	0°	3°	6°
JEDEC						

⚠ * NOTES : DIMENSION " D " DOES NOT INCLUDE MOLD PROTRUSIONS OR GATE BURRS.
MOLD PROTRUSIONS AND GATE BURRS SHALL NOT EXCEED 0.12 MM (0.005 INCH) PER SIDE.
DIMENSION " E1 " DOES NOT INCLUDE MOLD PROTRUSIONS
MOLD PROTRUSIONS SHALL NOT EXCEED 0.25 MM (0.010 INCH) PER SIDE.

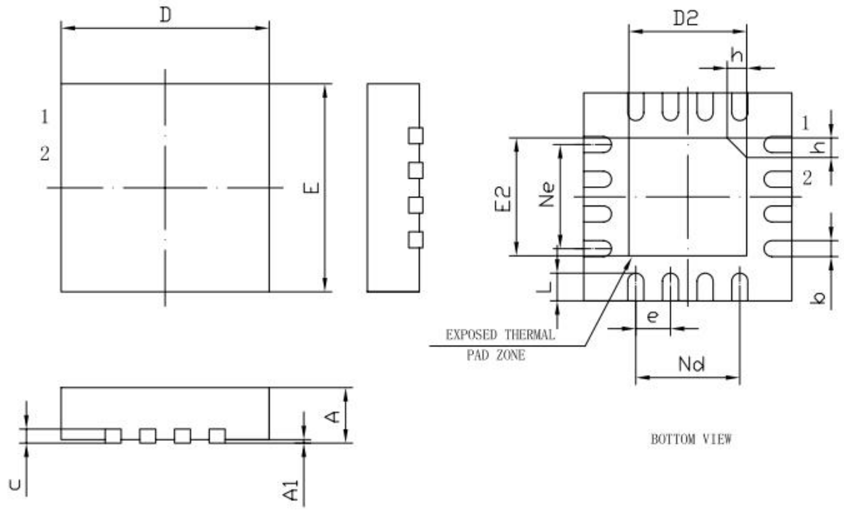
SOP-8 (150 mil) Package Dimension



SYMBOL	DIMENSION IN MM			DIMENSION IN INCH		
	MIN	NOM	MAX	MIN	NOM	MAX
A	1.35	1.55	1.75	0.0532	0.0610	0.0688
A1	0.10	0.18	0.25	0.0040	0.0069	0.0098
B	0.33	0.42	0.51	0.0130	0.0165	0.0200
C	0.19	0.22	0.25	0.0075	0.0087	0.0098
D	4.80	4.90	5.00	0.1890	0.1939	0.1988
E	5.80	6.00	6.20	0.2284	0.2362	0.2440
E1	3.80	3.90	4.00	0.1497	0.1536	0.1574
e	1.27 BSC			0.050 BSC		
h	0.25	0.38	0.50	0.0099	0.0148	0.0196
L	0.40	0.84	1.27	0.0160	0.0330	0.0500
θ	0°	4°	8°	0°	4°	8°
JEDEC	MS-012 (AA)					

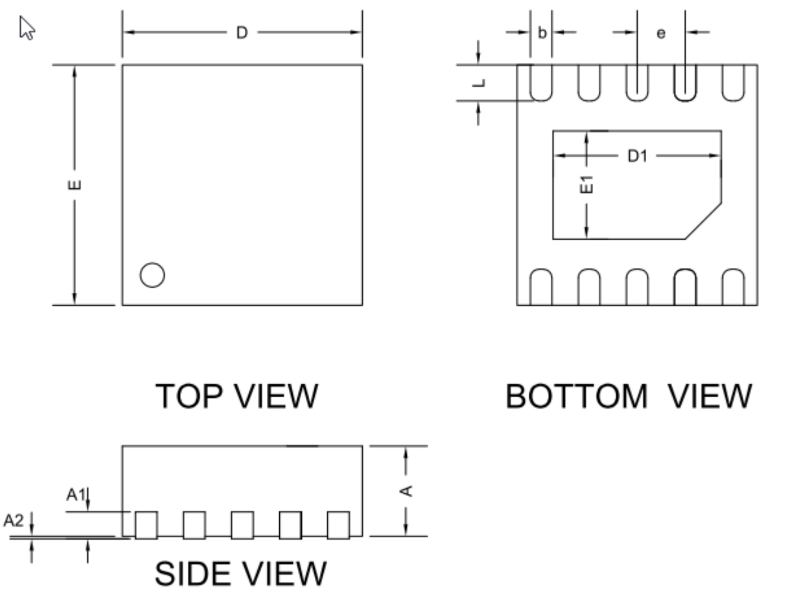
* NOTES : DIMENSION " D " DOES NOT INCLUDE MOLD FLASH, PROTRUSIONS OR GATE BURRS. MOLD FLASH, PROTRUSIONS AND GATE BURRS SHALL NOT EXCEED 0.15 MM (0.006 INCH) PER SIDE.

QFN-16 (3*3*0.75-0.5mm) Package Dimension



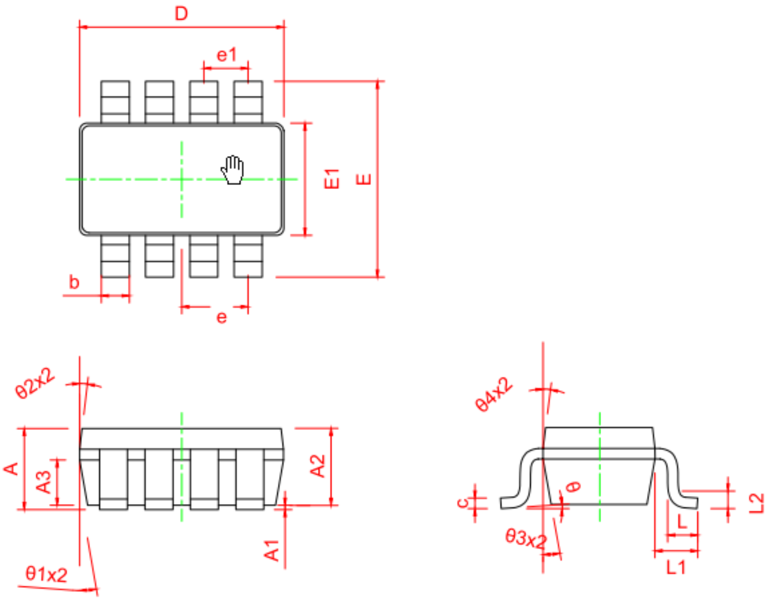
SYMBOL	MILLIMETER		
	MIN	NOM	MAX
A	0.70	0.75	0.80
A1	—	0.02	0.05
b	0.18	0.25	0.30
c	0.18	0.20	0.25
D	2.90	3.00	3.10
D2	1.55	1.65	1.75
e	0.50BSC		
Ne	1.50BSC		
Nd	1.50BSC		
E	2.90	3.00	3.10
E2	1.55	1.65	1.75
L	0.35	0.40	0.45
h	0.20	0.25	0.30
L/F载体尺寸 (mil)	75x75		

DFN-10 (2*2*0.75-0.4mm) Package Dimension



COMMON DIMENSIONS (UNITS OF MEASURE=MILLIMETER)			
SYMBOL	MILLMETER		
	MIN	NOM	MAX
D	1.95	2.00	2.05
E	1.95	2.00	2.05
D1	1.375	1.40	1.425
E1	0.875	0.90	0.925
b	0.155	0.18	0.205
L	0.275	0.30	0.325
e	0.40BSC		
A	0.70	0.75	0.80
A1	0.203REF		
A2	0.00	0.02	0.05
DIE PAD SIZE	1.8 X 1.1		

SOT23-8 Package Dimension



项目	MIN	NOM	MAX
A	--	--	1.25
A1	0.02	0.05	0.10
A2	1.05	1.10	1.15
A3	0.60	0.65	0.70
D	2.82	2.92	3.02
E	2.60	2.80	3.00
E1	1.50	1.60	1.70
L	0.30	0.45	0.60
L1	0.60 REF		
L2	0.25 BSC		
θ	0°	—	8°
b	0.28	0.35	0.42
c	0.10	0.15	0.20
e	0.950 BSC		
e1	0.633 BSC		
θ1	12°BSC		
θ2	10°BSC		
θ3	12°BSC		
θ4	10°BSC		