十速

TM56F1542

DATA SHEET

Rev 0.92

AMENDMENT HISTORY

Version	Date	Description
0.90	July, 2023	New Release
0.91	Aug, 2023	Add Low Power Reading mode and related electrical characteristics.
0.92	Aug, 2023	Modify detail description in the datasheet.

CONTENTS

FEATURES

1. **ROM: 4K x 16 bits Flash Program Memory**
2. **EEPROM: 128 x 8 bits**
3. **RAM: 336 x 8 bits**
4. **STACK: 8 Levels**
5. **System Oscillation Sources (Fsys)**
   - Fast-clock: FIRC (Fast Internal RC, 18.432 MHz)
   - Slow-clock: SIRC (Slow Internal RC, 50 KHz @VCC=5V)
6. **System Clock Prescaler**
   - System Oscillation Sources can be divided by 1/2/4/8 as System Clock (Fsys)
7. **Dual System Clock:** FIRC+SIRC
8. **Power Saving Operation Mode**
   - FAST Mode: Fast-clock keeps CPU running , Slow-clock can be enabled/disabled
   - SLOW Mode: Fast-clock is disabled, Slow-clock keeps CPU running
   - IDLE Mode: Fast-clock and CPU stop. Slow-clock, T2, or Wake-up Timer keep running
   - STOP Mode: All clocks stop, T2 and Wake-up Timer stop
9. **3 Independent Timers**
   - Timer0: 8-bit timer divided by 1~256 pre-scaler option, Reload/Interrupt/Stop function
   - Timer1
     - 8-bit timer divided by 1~256 pre-scaler option, Reload/Interrupt/Stop function
     - Overflow and Toggle out
   - T2
     - 15-bit timer with 4 interrupt interval time options
     - IDLE mode wake-up timer or used as one simple 15-bit time base
     - Clock source: Slow-clock (SIRC), Fsys/128
10. **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/T2/WKT (wake-up) Interrupts
    - ADC Interrupt
    - Touch Key Interrupt
    - I2C Interrupt

- Pin Change Interrupt
- LVD Interrupt

11. Wake-up (WKT) Timer

- Clocked by built-in RC oscillator with 4 adjustable interrupt times
  20.5 ms/41 ms/82 ms/164 ms @ Vcc=5V

12. Watchdog Timer

- Clocked by built-in RC oscillator with 4 adjustable reset times
  164ms/328ms/655ms/1311ms @ Vcc=5V
- Watchdog timer can be disabled/enabled in STOP mode

13. PWM

- PWM0 :
  - 16 bits, duty-adjustable, period-adjustable
  - Clock source: System clock(Fsys), FIRC (18.432MHz) or FIRC*2 (36.864MHz)
  - Complementary output (PWM0P, PWM0N), 4 output modes in total
  - Non overlap time durations adjustable: (0~15) * PWM CLK
- PWM1~5:
  - 16 bits, duty-adjustable (Independent) , period shared with PWM0
  - Clock source shared with PWM0

14. Touch Key

- One 5-channel Touch Key module, and One 8-channel Touch Key module
- Each module include
  - 3-bit TK reference clock capacitor adjustment
  - 8-bit touch key clock frequency select(can be fixed frequency or auto change)
  - 14-bit TK scan length adjustment
- Interrupt/Wake-up CPU while key is pressed.

15. I2C Interface

- Specific purpose slave I2C interface with interrupt function

16. 12-bit ADC Converter with 10 input channels and 2 internal reference voltage

- Two Internal voltage channel: VBG or $^{1}/_{4}$ Vcc
- ADC reference voltage: Vcc or VBG(2.50V)

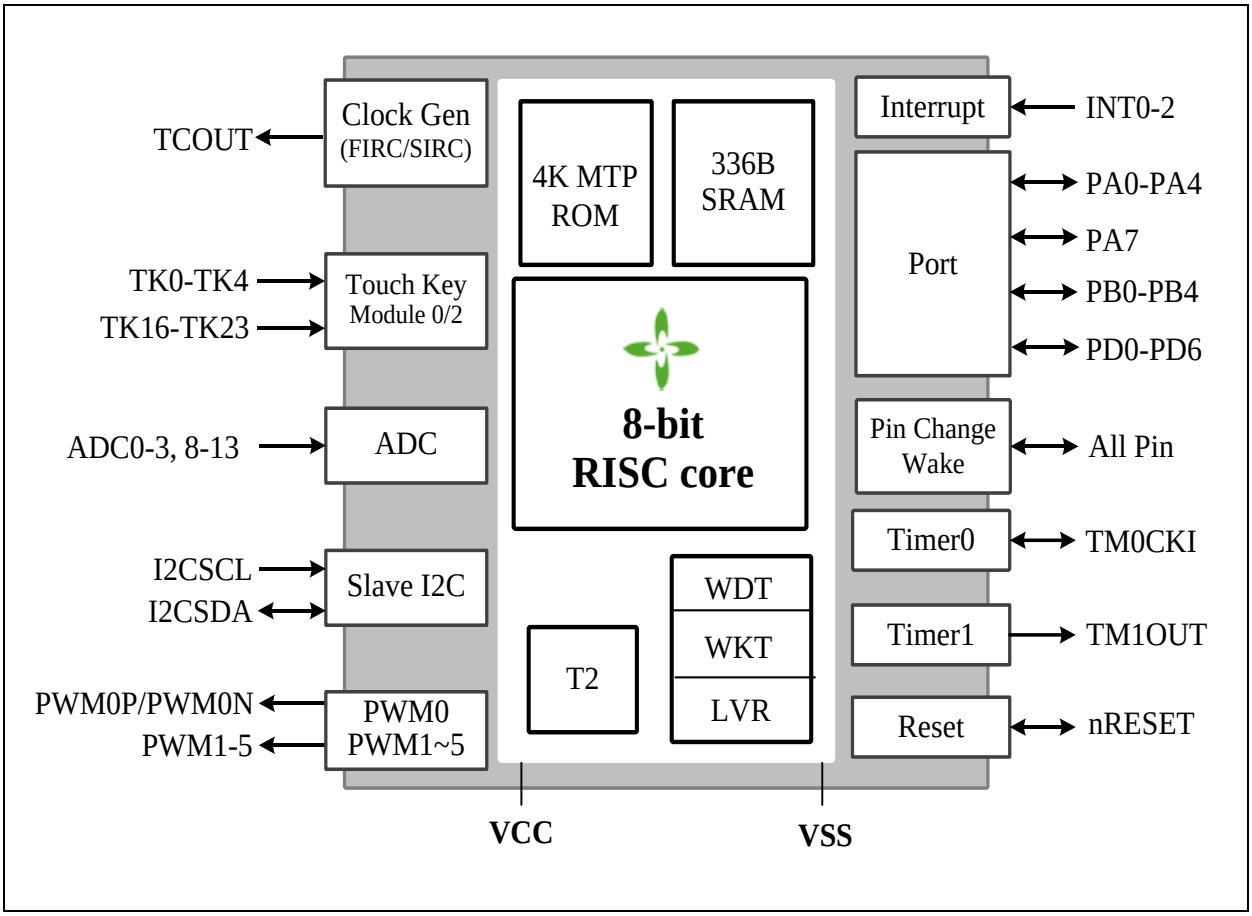
17. All pin change wake up (negedge and posedge trigger)

18. Reset Sources

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

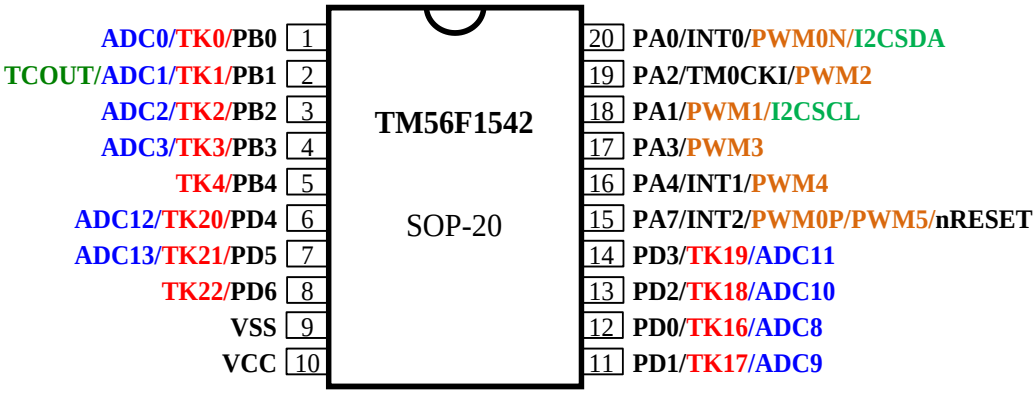
19. **Low Voltage Reset (LVR) /Low Voltage Detection Flag (LVD) Option:**
    - 16-Level Low Voltage Reset: 2.05 - 4.15 V, can be disabled
    - 15-Level Low Voltage Detection Flag 2.28V - 4.15V, can be disabled
20. **Operating Voltage**
    - Fsys= 1 MHz, LVR ~ 5.5V
    - Fsys=18.432 MHz, 2.5 ~ 5.5V
21. **Operating Temperature Range : -40℃ to + 85℃**
22. **Integrated 16-bit Cyclic Redundancy Check (CRC)**
23. **Table Read Instruction: 16-bit ROM data lookup table**
24. **Instruction set: 39 Instructions**
25. **Instruction Execution Time**
    - 2 system clocks (Fsys) per instruction except branch
26. **I/O ports: Maximum 26 programmable I/O pins**
    - Open-Drain Output
    - CMOS Push-Pull Output
    - Schmitt Trigger Input with pull-up resistor option
    - Support High sink mode and Constant current Drive mode
27. **Programming connectivity support 4-wire (ICP) or 5-wire program**
28. **Page Locker Size: 512W/640W/768W …./2304W by 128 words step**
29. **Package Types:**
    - SOP-20/16

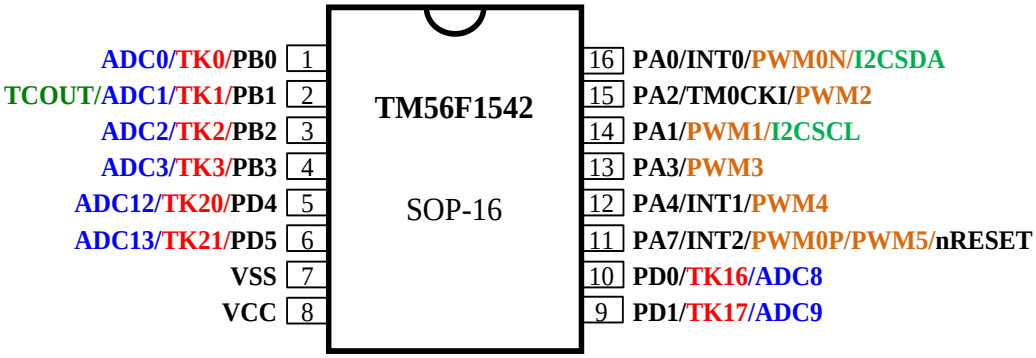
SYSTEM BLOCK DIAGRAM



TM56F1542 Block Diagram

PIN ASSIGNMENT DIAGRAM





PIN DESCRIPTIONS

Name	In/Out	Pin Description
PA7, PA4–PA0	I/O	Bit-programmable I/O port for Schmitt-trigger input, CMOS push-pull output or open-drain output. Pull-up resistors are assignable by software.
nRESET	I	External active low reset/ Schmitt-trigger input
VCC, VSS	P	Power input pin and ground
INT0-INT2	I	External interrupt input
PB4-PB0	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.
PD7-PD0	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.
PWM0P/PWM0N PWM1 PWM2 PWM3 PWM4 PWM5	O	PWM0/PWM1/PWM2/PWM3/PWM4/PWM5 outputs
ADC3-0, ADC8-13	I	Analog to Digital Convert input pin
TK4-TK0	I	Touch Key Module 0 input
TK23-TK16	I	Touch Key Module 2 input
TM0CKI	I	Timer0's input pin in counter mode
TCOUT	O	Fsys/2 clock output
TM1OUT	O	Timer1 overflow toggle output
I2CSCL	I	I2C Serial clock input
I2CSDA	I/O	I2C serial data pin

Programming pins:

Normal mode (5-wire): VCC / VSS / PA0 / PA1 / PA2

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

Pin Summary

Pin number	Pin Name	Type	GPIO				Function after reset	Alternate Function				
			Input		Output							
SOP20			Weak Pull-up	Ext. Interrupt	O.D.	P.P.		PWM	ADC	Touch Key	I2C	MISC
1	PB0/TM1OUT/TK0/ADC0	I/O	✓		✓	✓	PB0		✓	✓		TM1OUT
2	PB1/TCOUT/TK1/ADC1	I/O	✓		✓	✓	PB1		✓	✓		TCOUT
3	PB2/TK2/ADC2	I/O	✓		✓	✓	PB2		✓	✓		
4	PB3/TK3/ADC3	I/O	✓		✓	✓	PB3		✓	✓		
5	PB4/TK4	I/O	✓		✓	✓	PB4			✓		
6	PD0/TK16/ADC8	I/O	✓		✓	✓	PD0		✓	✓		
7	PD1/TK17/ADC9	I/O	✓		✓	✓	PD1		✓	✓		
8	PD2/TK18/ADC10	I/O	✓		✓	✓	PD2		✓	✓		
9	PD3/TK19/ADC11	I/O	✓		✓	✓	PD3		✓	✓		
10	PD4/TK20/ADC12	I/O	✓		✓	✓	PD4		✓	✓		
11	PD5/TK21/ADC13	I/O	✓		✓	✓	PD5		✓	✓		
12	PD6/TK22	I/O	✓		✓	✓	PD6			✓		
13	VSS	P										
14	PA7/PWM0P/INT2	I/O	✓	✓	✓	✓	PA7	✓				nRESET
15	PA4/PWM4/INT1	I/O	✓	✓	✓	✓	PA4	✓				
16	VCC	P										
17	PA3/PWM3	I/O	✓		✓	✓	PA3	✓				
18	PA1/PWM1/I2CSCL	I/O	✓		✓	✓	PA1	✓			✓	
19	PA0/PWM0N/INT0/I2CSDA	I/O	✓	✓	✓	✓	PA0	✓			✓	
20	PA2/TM0CKI/PWM2	I/O	✓		✓	✓	PA2	✓				TM0CKI

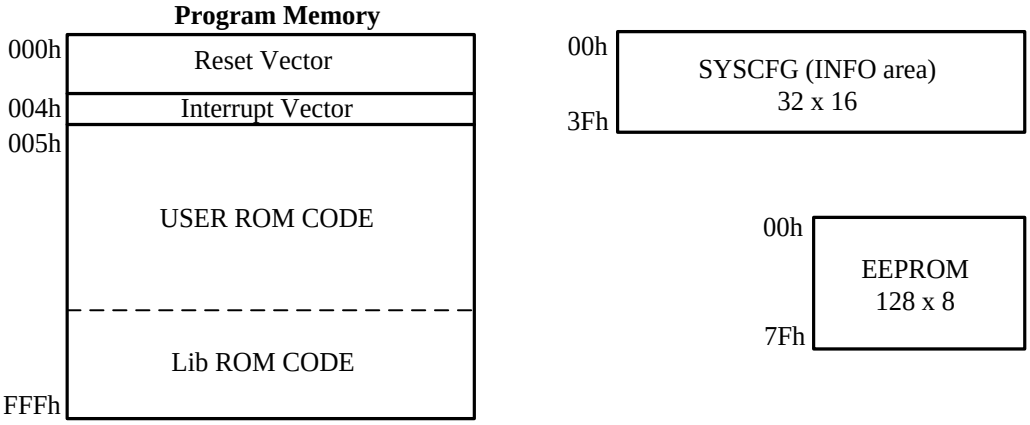
Symbol：O.D. = Open Drain
P.P. = Push-Pull Output

FUNCTION DESCRIPTION

1. CPU Core

1.1 Program ROM (PROM)

The 4Kx16 bits Flash Program ROM of this device is 4K words, with an extra 32-Word INFO area to store the SYSCFG and another extra 128-Byte EEPROM. The ROM can be written multi-times and can be read as long as the PROTECT and LPROT bit of SYSCFG are not set. The SYSCFG can be read no matter PROTECT or LPROT is set or cleared, but PROTECT bit can be cleared only when User ROM Code area is erased, and LPROT bit can be cleared only when the Lib ROM Code area is erased. That is, clearing the PROTECT or LPROT bit needs to erase the corresponding ROM area. If LPROT bit is set, the ROM can still be written multi-times in the User ROM Code area to update user ROM code again by writer, but the Lib ROM Code area will not be read or written again by writer until the LPROT bit is cleared. 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.
Furthermore, when the clock source of CPUCLK is SRC, TM56F1542 supports low power reading mode, which can effectively reduce the working current. Setting ISAVB (105h.0~1) to 2'b00 will make TM56F1542 enter this mode. **It is prohibited to leave ISAVB as 2'b00 when the clock source of CPUCLK is FIRC, and SYSREG bit7~2 should always be kept at 6'b000000, or unexpected problem may occur. So make sure ISAVB is 2'b11 before switch FAST mode.**



105h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
SYSREG	–	–	–	–	–	–	ISAVB	
R/W	–	–	–	–	–	–	R/W	
Reset	0	0	0	0	0	0	11	

105h.1~0 **ISAVB:** Power saving mode for Reading Operation of ROM at SLOW mode.
00: Enable
01/10: Reserved
11: Disable

◇ Example: Switch to Low Power Reading mode from FAST mode

```
BCX     SLOWSTP     ; Enable Slow-clock.
NOP
BCX     CPUCKS      ; Fsys=Slow-clock
MOVLW   00h
MOVWX   SYSREG      ; Enter low power reading mode and
                    ; SYSREG bit7~2 remain 6'b000000
```

◇ Example: Switch to Low Power Reading mode from SLOW mode

```
MOVLW   00h
MOVWX   SYSREG      ; Enter low power reading mode and SYSREG bit7~2
                    ; remain 6'b000000
```

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)

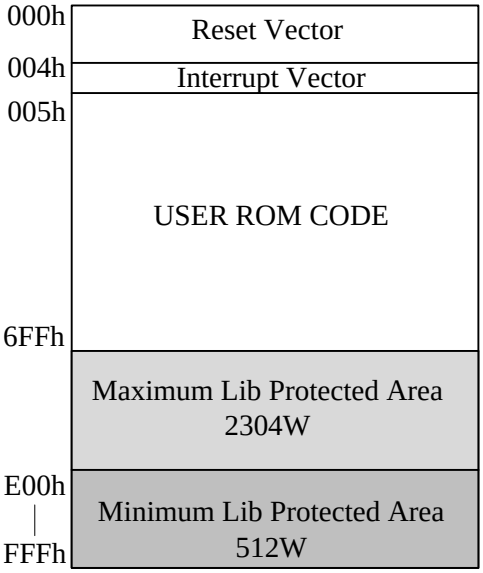
The System Configuration Register (SYSCFG) is located at Flash INFO area; it contains two 13bits registers (CFGWL/CFGWH). The SYSCFG determines the option for initial condition of CPU. It is written by PROM Writer only. User can select LVR operation Mode and chip operation mode by SYSCFG register. The 15th bit of CFGWH is code protect selection bit. If this bit is 1, the data in PROM will be protected, when user reads PROM.

Bit		**13-0**	
Default Value		**00000000000000**	
Bit		**Description**	
CFGWL	13	**LPROT**: Lib Code protection selection	
		1	Enable
		0	Disable
	12-9	**LSIZE**: Lib Size selection	
		1111	2304W
		…	…
		0001	512W
		0000	No use Page locker function
	8-0	Reserved	
CFGWH	15	**PROTECT**: Code protection selection	
		1	Enable
		0	Disable
	14	**XRSTE**: External Pin (PA7) Reset Enable	
		1	Enable
		0	Disable
	13-10	**LVR**: Low Voltage Reset Mode	
		1111	LV Reset 4.15V
		1110	LV Reset 4.01V
		…	…
		0001	LV Reset 2.19V
		0000	LV Reset 2.05V
	9-8	**WDTE**: WDT Reset Enable	
		11	Always Enable
		10	Enable in FAST/SLOW mode, Disable in IDLE/STOP mode
		0X	Disable
	7-0	Reserved	

1.3 Page Lock Function

TM56F1542 supports Page locker function. By setting LPROT (CFGWL.13).If library code provider turns this function on and selects different size (512~2304W) of lib protected area by LSZIE (CFGWL12~9), firmware developer can't read ROM code of Lib Protected area by TABRL/TABRH instruction or in any other way. However, firmware developer still can continue to complete the main code in the unprotected area.

4K Program ROM



LSIZE	Lib Protected Area
2304	700h-FFFh
2176	780h-FFFh
2048	800h-FFFh
1920	880h-FFFh
1792	900h-FFFh
1664	980h-FFFh
1536	A00h-FFFh
1408	A80h-FFFh
1280	B00h-FFFh
1152	B80h-FFFh
1024	C00h-FFFh
896	C80h-FFFh
768	D00h-FFFh
640	D80h-FFFh
512	E00h-FFFh

1.4 EEPROM

TM56F1542 contains 128 bytes of data EEPROM memory. It is organized as a separate data space, which single bytes can be read and written. According the physical characteristic, EEPROM needs longer access time than Program ROM. EEPROM has an endurance of at least 10K write/erase cycle.

EEPROM Read usage is the same as using Table Read instruction except EEPROM enable bit is set to high and DPH is always set to zero. Writing 0xE2 to register EEPEN (192h) can set the EEPROM enable bit, and writing other value to EEPEN (192h) will clear the EEPROM enable bit.

◇Example: read EEPROM data @address 23h

```
MOVLW       E2h         ;
MOVWX       EEPEN       ; set EEPROM enable bit
CLRX        DPH         ; set DPH=0 for EEPROM write/read
MOVLW       00h
MOVWX       DPH
MOVLW       23h
MOVWX       DPL         ; set DPTR=0023h
; read EEPROM @Address 23h data into W by using opcode TABRL
TABRL
….

; Another way to read EEPROM data into W
MOVLW       01h         ;
MOVWX       TABR        ; TABR=01h=opcode TABRL
MOVXW       TABR        ; Read EEPROM data to W
…
```

The EEPROM Write usage is similar to read EEPROM except LVRPD is set to 0x38 to disable LVR and stop the Interrupt service. When F/W writes data to the register EEPDT (193h), the data will also be written to EEPROM.

◇Example: write EEPROM data A5h to address 23h

```
CLRX        INTIE       ; disable INTIE
CLRX        INTIE1      ; disable INTIE1
MOVLW       E2h         ;
MOVWX       EEPEN       ; set EEPROM enable bit
CLRX        DPH         ; set DPH=0 for EEPROM write/read
MOVLW       23h
MOVWX       DPL         ; set DPTR=0023h
MOVLW       00000010b
MOVWX       EEPCTL      ; set EEPROM write with 10mS time out
MOVLW       038h        ; W=38h
MOVWX       LVRPD       ; LVRPD = 038h, force LVR disable
                        ; LVR must be disabled before EEP write operation
MOVLW       A5h
MOVWX       EEPDT       ; write data A5h EEPDT (193h)
                        ; The data also save to EEPROM @Address 23h
BTXSC       EEPTO       ; check EEPROM write time-out flag
LGOTO       TIMEOUT
CLRX        EEPEN       ; protect EEPROM from abnormal write
CLRX        LVRPD       ; LVRPD = 00h ,  LVR/POR enable
```

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

191h.7 **EEPTO:** EEPROM Write Time-Out flag
191h.1~0 **EEPTE:** Write Time-Out enable (Busy wait time)
00:Disable 01: 2.5ms 10:10ms 11:20ms

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

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

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

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

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

107h.7~0 **LVRPD:** LVR/POR power down register
Write 0x37 to force LVR+POR be disabled. (LVR must be disabled before EEPROM Write operation)
Write 0x38 to force LVR be disabled, POR still enable.
Write 0x39 to force POR be disabled, LVR still enable
Write other value to enable LVR+POR

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							

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

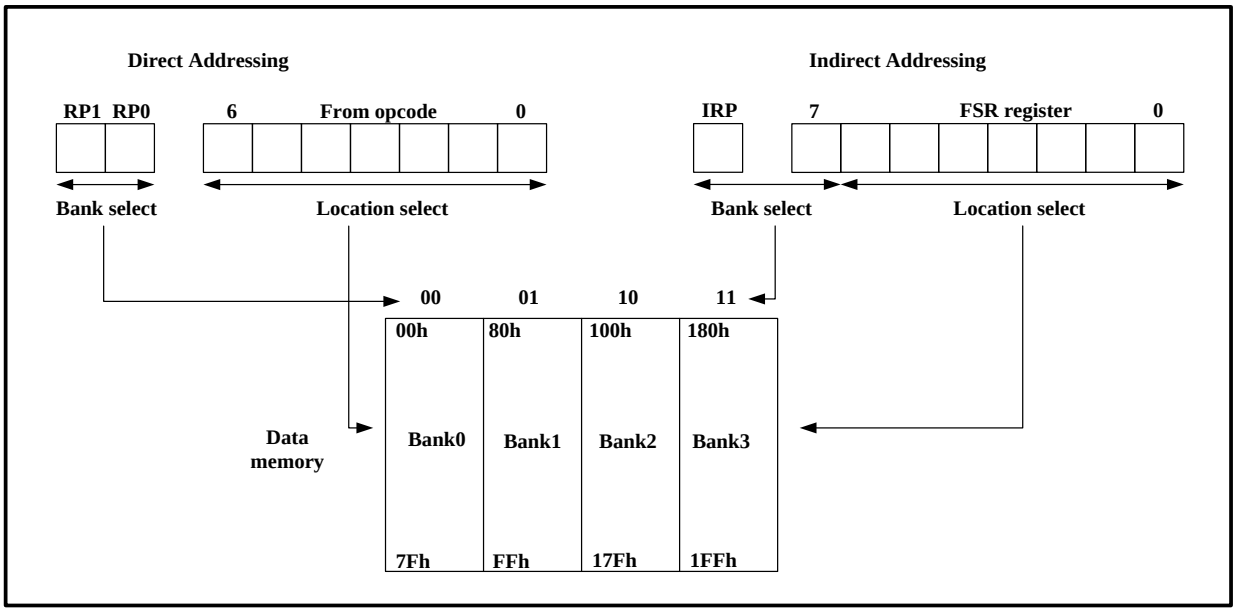
1.5 RAM Addressing Mode

There is a Data Memory Space in CPU. The 336-Byte memory is partitioned into four banks memory plane. 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, which is 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 bit for BANK 0/1/2/3.

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

The 336-Byte memory plane can be addressed directly or indirectly for BANK 0/1/2/3. The INDF register is not a physical register. Addressing the INDF register will cause indirect addressing. Indirect addressing is available by using the INDF register. Any instruction using the INDF register actually accesses the register pointed to by the File Select Register, FSR. Reading the INDF register itself, indirectly (FSR = '0') will read 00h. Writing to the INDF register indirectly 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 refined. 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 bank.

For example:

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

【BANK0】 000~07Fh		【BANK1】 080h~0FFh		【BANK2】 100h~17Fh		【BANK3】 180h~1FFh	
000h	**INDF0**	080h	**INDF0**	100h	**INDF0**	180h	**INDF0**
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	PAMODH	105h	SYSREG	185h	DPL
006h	PBD	086h	PAMODL	106h	ChgRdMode	186h	DPH
007h		087h	PBMODH	107h	LVRPD	187h	
008h	PDD	088h	PBMODL	108h		188h	
009h		089h		109h		189h	
00Ah	**PCLATH**	08Ah	**PCLATH**	10Ah	**PCLATH**	18Ah	**PCLATH**
00Bh	**INTIE**	08Bh	**INTIE**	10Bh	**INTIE**	18Bh	**INTIE**
00Ch	INTIF	08Ch		10Ch	PCH	18Ch	TABR
00Dh	FSR1	08Dh	PDMODH	10Dh		18Dh	
00Eh	INTIE1	08Eh	PDMODL	10Eh	BGTRIM	18Eh	CRCDL
00Fh	CLKCTL	08Fh	OPTION2	10Fh	IRCF	18Fh	CRCDH
010h	TM0RLD	090h	PWMOE	110h	I2CTXD0	190H	CRCIN
011h	TM0CTL	091h	PWMCTL	111h	I2CTXD1	191h	EEPCTL
012h	TM1	092h	PWMPRDH	112h	I2CCTL	192h	EEPEN
013h	TM1RLD	093h	PWMPRDL	113h	I2CFLG	193h	EEPDT
014h	TM1CTL	094h	PWM0DH	114h	I2CRCD0	194h	TKM0TMRL
015h	T2CTL	095h	PWM0DL	115h	I2CRCD1	195h	TKM0TMRH
016h	LVCTL	096h	PWM1DH	116h	TKM0DL	196h	
017h	ADCDH	097h	PWM1DL	117h	TKM0DH	197h	
018h	ADCTL	098h	PWM2DH	118h		198h	TKM2TMRL
019h	ADCTL2	099h	PWM2DL	119h		199h	TKM2TMRH
01Ah	INTIF1	09Ah	PWM3DH	11Ah	TKM2DL	19Ah	TKM0REFC
01Bh	IOCCTL	09Bh	PWM3DL	11Bh	TKM2DH	19Bh	
01Ch	PAWKE	09Ch	PWM4DH	11Ch	TKMCON0	19Ch	TKM2REFC
01Dh	PBWKE	09Dh	PWM4DL	11Dh	TKMCON1	19Dh	TKMCHS0
01Eh		09Eh	PWM5DH	11Eh	TKMCTL0	19Eh	TKMCHS1
01Fh	PDWKE	09Fh	PWM5DL	11Fh	TKMCTL1	19Fh	CTRLRFK
020h 06Fh	General Purpose SRAM (80 Bytes)	0A0h 0EFh	General Purpose SRAM (80 Bytes)	120h 16Fh	General Purpose SRAM (80 Bytes)	1A0h 1EFh	General Purpose SRAM (80 Bytes)
070h 07Fh	**Common Area (16 Bytes)**	0F0h 0FFh	**accesses 070h~07Fh**	**170h** 17Fh	**accesses 070h~07Fh**	**1F0h** 1FFh	**accesses 070h~07Fh**

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

```
TM1         equ     012h    ;SFR in Bank0
PWM1DH      equ     096h    ;SFR in Bank1
I2CTXD0     equ     110h    ;SFR in Bank2
TKM0TMRL    equ     194h    ;SFR in Bank3
RAM20       equ     020h    ;RAM in Bank0
RAMA0       equ     0A0h    ;RAM in Bank1
RAM120      equ     120h    ;RAM in Bank2
RAM1A0      equ     1A0h    ;RAM in Bank3

MOVXW       TM1             ; read TM1 (Bank0) to W
MOVXW       PWM1DH          ; read PWM1PRD (Bank1) to W
MOVXW       I2CTXD0         ; read I2CTXD0 (Bank2) to W
MOVXW       TKM0TMRL        ; read TKM0TMRL (Bank4) to W

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

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

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

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

```
BCX         IRP             ; IRP=0 => Bank0/1
MOVLW       10h             ; W=10h
MOVWX       FSR             ; FSR = W =10h
MOVXW       INDF            ; read SFR TM0RLD (10h) to W

BCX         IRP             ; IRP=0 =>Bank0/1
MOVLW       10h             ; W=10h
MOVWX       FSR             ; FSR = W =10h
MOVLW       37h             ; W=37h
MOVWX       INDF            ; TM0RLD (10h) = W = 037h
```

000h/080h/100h/180h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INDF	INDF							
R/W	R/W							
Reset	0							

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

004h/084h/104h/184h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
FSR	FSR							
R/W	R/W							
Reset	0							

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

1.6 Programming Counter (PC) and Stack

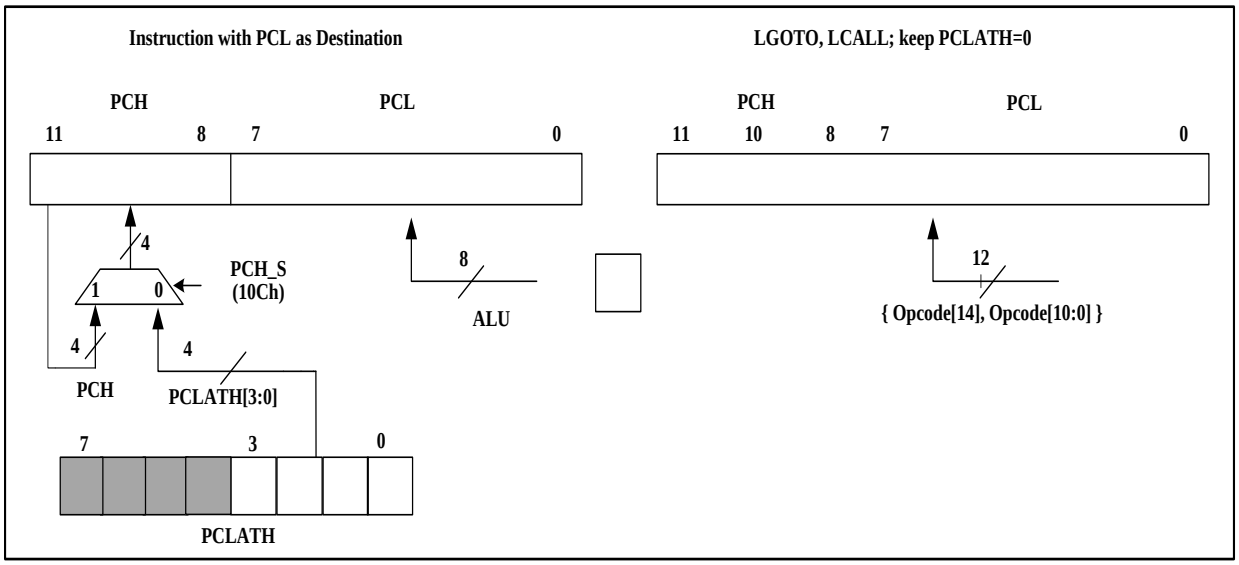
The Programming Counter is 12-bit wide capable of addressing a 4K x 16 Flash ROM. As a program instruction is executed, the PC will contain the address of the next program instruction to be executed. The PC value is normally increased by one except for the following cases. The Reset Vector (000h) and the Interrupt Vector (004h) are provided for PC initialization and interrupt. For LCALL/LGOTO instruction, PC loads 11 bit address from instruction word and upper 1 bit from PCLATH[3]. For RET/RETI/RETLW instruction, PC retrieves its content from the top level STACK.

Before CALL/GOTO instruction is executed, the PCLATH[3] must be set if the destination address more than 2K, otherwise the PCLATH[3] must be cleared. Similar as RAM Addressing Mode (refer section 1.5), the chip provides new instruction set LCALL/LGOTO to replace CALL/GOTO instruction set. When using LCALL/LGOTO, user don't need to care about the destination address, just only keep PCLATH[3] cleared.

The low byte data of the Programming Counter (PC[7:0]) can be read or written by PCL register (002h/082h/102h/182h). The high byte data of Programming Counter (PC[11:8]) can only be read by PCH register (10Ch). The internal flag PCH_S is used to select the source of PCH, when executing any instruction with the PCL register as the destination. Write 0x1C to PCH register can set PCH_S, and write others value to PCH register will clear PCH_S. After reset, the PCH_S will be 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 PCH will not change. **It is recommended to setting PCH_S to '1' when using any instruction with the PCL register as the destination**, but C language doesn't support this function.



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

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

00Ah	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PCLATH	GPR				PCLATH			
R/W	R/W				R/W			
Reset	0	0	0	0	0	0	0	0

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

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

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

10Ch.7~0 **PCH_S:** Program Counter Upper bits selection when instruction with PCL as destination is executed
write 0x1C to set PCH_S= 1: PCH keep the original value
write others to clear PCH_S=0: PCH is from PCLATH

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

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

```
ORG         00h                         ; Reset Vector
            LGOTO       START

ORG         04h                         ; Interrupt Entry Address
            MOVWX       W_TMP           ; if HWAUTO =1, can omit this line
            MOVXW       STATUS          ; if HWAUTO =1, can omit this line
            MOVWX       STATUS_TMP      ; if HWAUTO =1, can omit this line
            MOVXW       PCLATH
            MOVWX       PCLATH_TMP

            BTXSC       TM0IF
            LCALL       TM0INT_TASK
            …
            MOVXW       PCLATH_TMP
            MOVWX       PCLATH
            MOVXW       STATUS_TMP      ; if HWAUTO =1, can omit this line
            MOVWX       STATUS          ; if HWAUTO =1, can omit this line
            SWAPX       W_TMP, f        ; if HWAUTO =1, can omit this line
            SWAPX       W_TMP           ; if HWAUTO =1, can omit this line
            MOVXW       W_TMP           ; if HWAUTO =1, can omit this line
            RETI
START:
            MOVLW       00h
            MOVWX       INDEX           ; Set lookup table’s address
            MOVLW       1Ch
            MOVWX       PCH_S
            MOVLW       00h
            MOVWX       RAM20           ; RAM20 as PCL counter
LOOP:
            MOVLW       RAM20
            LCALL       TABLE1          ; To lookup data
            …
            INCX        RAM20, 1        ; Set PCL counter total value
            …
            LGOTO       LOOP            ; Go to LOOP label
            …
            MOVLW       (TABLE2>>8) & 0xff
            MOVWX       DPH
            MOVLW       (TABLE2) & 0xff
            MOVWX       DPL             ; DPTR = {DPH, DPL} = TABLE2
; Table Read by opcode TABRL / TABRH
            TABRL                       ; read PROM low byte data to W (W=86h)
            TABRH                       ; read PROM high byte data to W (W=19h)
            …
```

```
; Table Read by SFR TABR
            MOVLW   01h             ; TABR = 01h = instruction TABRL
            MOVWX   TABR            ; read PROM low byte data to TABR (TABR = 86h)
            MOVXW   TABR            ; read TABR to W (W=86h)
            MOVLW   02h             ; TABR=02h=instruction TABRH
            MOVWX   TABR            ; read PROM high byte data to TABR (TABR = 19h)
            MOVXW   TABR            ; read PROM high byte data to W (W=19h)
            …

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

ORG         368h
TABLE2:
            .DT     0x1986          ; 16-bit ROM data
          . .DT     0x3719
            …

ORG         100h
TM0INT_TASK:
            MOVLW 11101111B
            MOVWX INTIF
            …
            RET
```

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

81h.7 HWAUTO**:** Enter interrupt vector, HW auto save/restore WREG and STATUS w/o TO, PD (only work for ASM, not for C)
0: disable
1: enable; (only work for ASM not for C)

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
2. TABR write 02h = instruction TABRH
3. After step.1 or step.2, read TABR to get main ROM table read value
After step.1, read TABR to get EEPROM value (when EEPEN = E2h)
Table Read for ASM: instruction TABRL / TABRH or register TABR
Table Read for C: using register TABR

1.7 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 ©, 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

1.8 STATUS Register (003h/083h/103h/183h)

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

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

◇ Example: Write immediate data into STATUS register.

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

◇ Example: Bit addressing set and clear STATUS register.

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

◇ Example: Determine the C flag by BTXSS instruction.

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

2. Reset

This device can be reset in four ways.

- Power-On-Reset (POR)
- Low Voltage Reset (LVR)
- External Pin Reset (PA7)
- Watchdog Reset (WDT)

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 will return to 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

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

2.2 Low Voltage Reset

The chip provides Low Voltage Reset (LVR) and Low Voltage Detection (LVD) functions. There are 16-level LVR can be selected by CFGWH and 15-level LVD can be selected by SFR LVDS (16h.3~0).

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

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

0000: Set LVR at 2.05V
0001: Set LVR at 2.19V
0010: Set LVR at 2.33V
0011: Set LVR at 2.47V
0100: Set LVR at 2.62V
0101: Set LVR at 2.75V
0110: Set LVR at 2.89V
0111: Set LVR at 3.03V
1000: Set LVR at 3.17V
1001: Set LVR at 3.31V
1010: Set LVR at 3.45V
1011: Set LVR at 3.59V
1100: Set LVR at 3.73V
1101: Set LVR at 3.87V
1110: Set LVR at 4.01V
1111: Set LVR at 4.15V

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

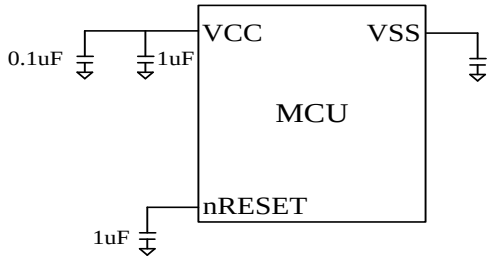
16h.7 **LVDF:** Low Voltage Detect output Flag
16h.5 **LVRSAV:** when set to 1, LVR auto power off in STOP/IDLE mode
16h.4 **LVDSAV:** when set to 1, LVD auto power off in STOP/IDLE mode
16h.3~0 **LVDS:** Low Voltage Detect select
0000: Disable LVD
0001: Set LVD at 2.19V
0010: Set LVD at 2.33V
0011: Set LVD at 2.47V
0100: Set LVD at 2.62V
0101: Set LVD at 2.75V
0110: Set LVD at 2.89V
0111: Set LVD at 3.03V
1000: Set LVD at 3.17V
1001: Set LVD at 3.31V
1010: Set LVD at 3.45V
1011: Set LVD at 3.59V
1100: Set LVD at 3.73V
1101: Set LVD at 3.87V
1110: Set LVD at 4.01V
1111: Set LVD at 4.15V

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

2.3 External Pin Reset

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

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



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

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

2.4 Watchdog Timer Reset

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

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

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

81h/181h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
OPTION	HWAUTO	INT0EDG	INT1EGE	-	WDTPSC		WKTPSC	
R/W	R/W	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 (@VCC=5V)
00: 164ms WDT overflow rate
01: 328ms WDT overflow rate
10: 655ms WDT overflow rate
11: 1311ms WDT overflow rate

◇ Example: Defining Reset Vector

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

        ORG     010h

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

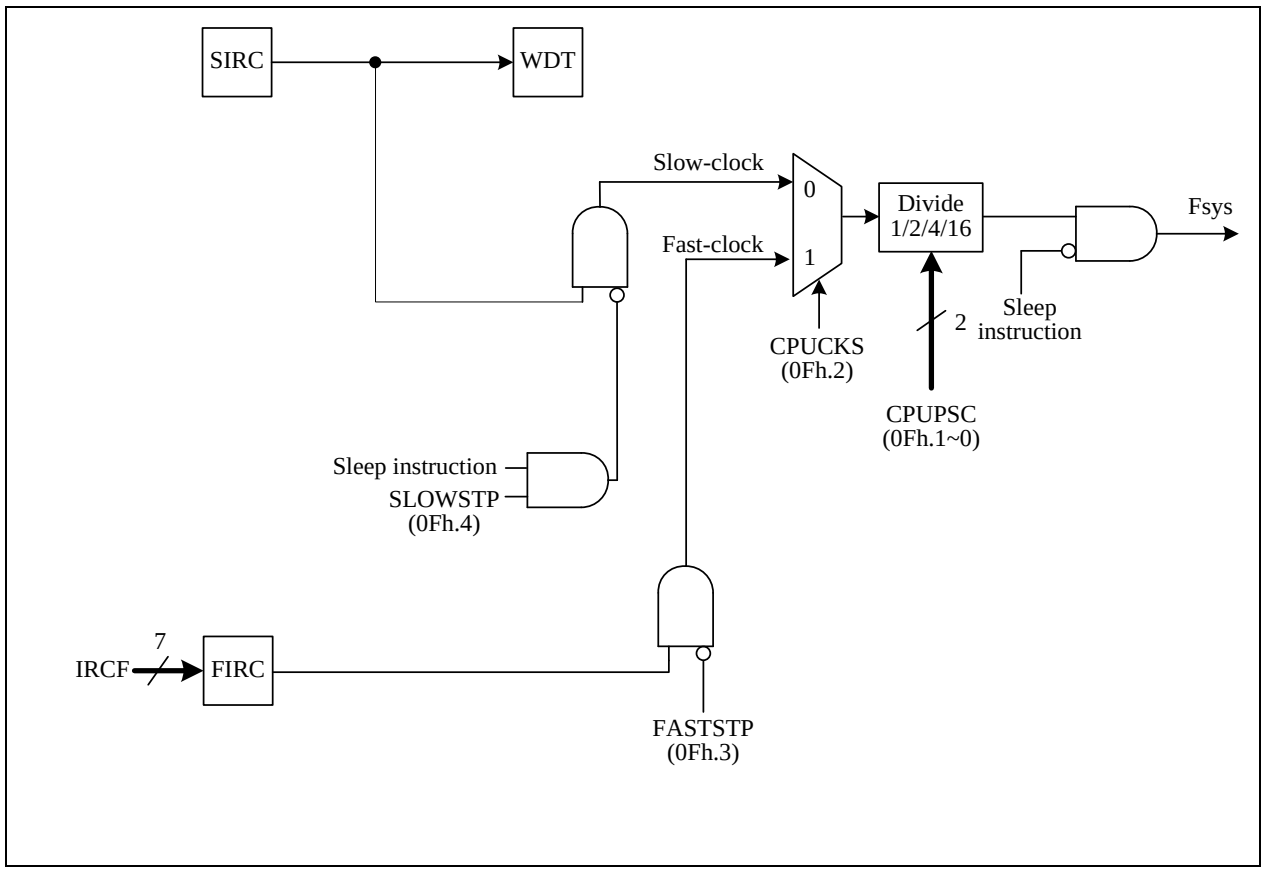
3. Clock Circuitry and Operation Mode

3.1 System Clock

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

After Reset, the device is running at SLOW mode with 50 KHz SIRC. S/W should select the proper clock rate for chip operation safety. The higher V_{CC} allows the chip to run at a higher System clock frequency. In a typical condition, an 18.432 MHz System clock rate requires V_{CC} > 2.8V.

The CLKCTL SFR controls the System clock operating. H/W automatically blocks the S/W abnormally setting for this register. 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 (Fast Internal RC) can be adjusted by IRCF (10Fh). And IRCF is trimmed to FIRC=18.432 MHz in chip manufacturing

FAST Mode:

In this mode, the program is executed using Fast-clock as CPU clock (Fsys). Timer0 and Timer1 blocks are driven by Fast-clock; The PWMx blocks can be driven by FIRC, FIRC*2 or Fsys. T2 can be driven by Slow-clock or Fsys/128 by setting T2CKS (15h.2).

SLOW Mode:

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

IDLE Mode:

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

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

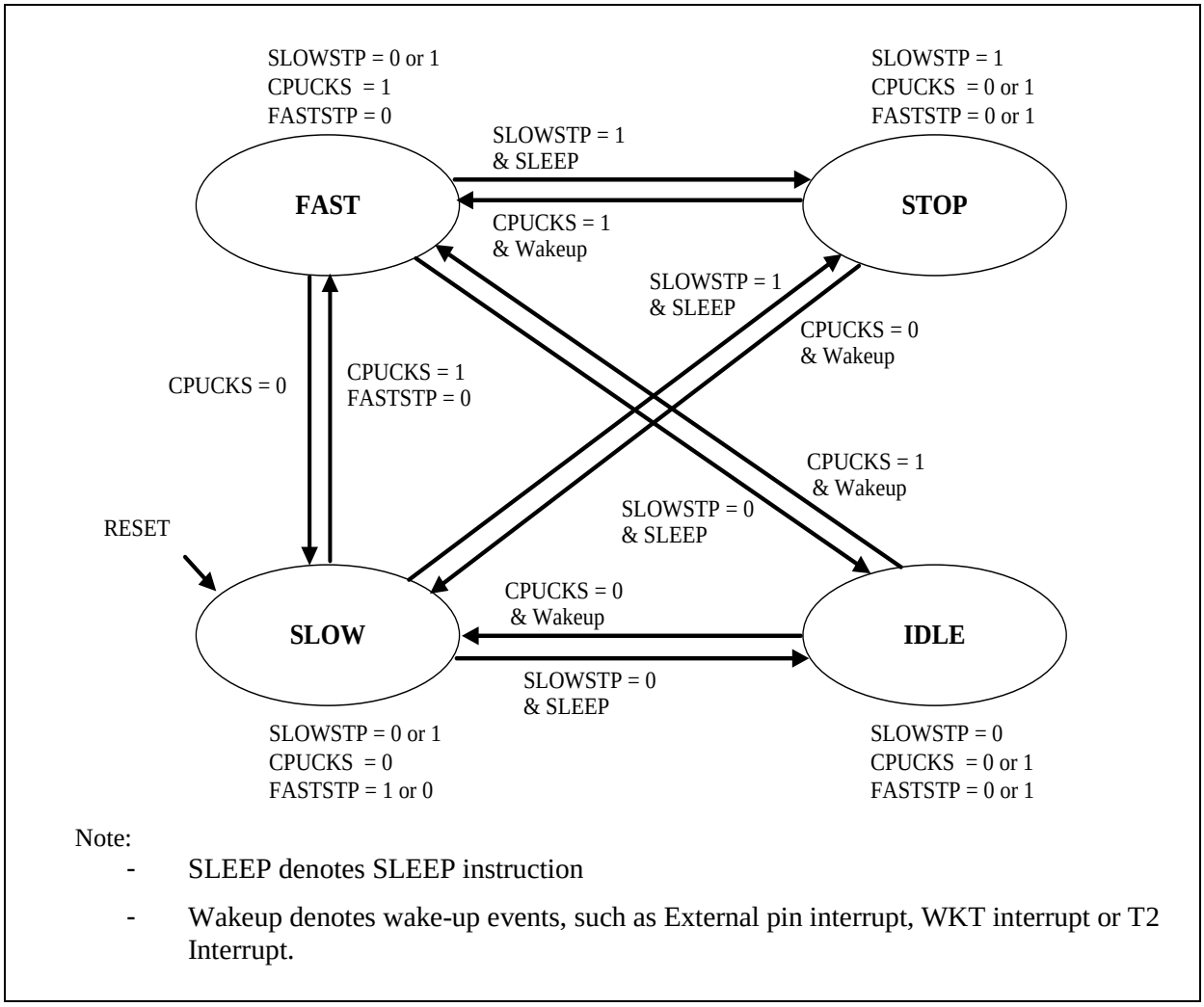
T2 and WKT/WDT are independent and have their own control registers. It is possible to keep both T2 and WKT working and wake-up in the IDLE mode.

STOP Mode:

If Slow-clock and WKT/WDT are disabled before executing SLEEP instruction, every block is turned off and the device enters STOP mode. STOP mode is similar to IDLE mode. The difference is all clock oscillators either Fast-clock or Slow-clock is power down and no clock is generated.

3.2 Dual System Clock Modes Transition

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



Note:

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

CPU Operation Block Diagram

CPU Mode & Clock Functions Table:

Mode	Oscillator	Fsys	Fast-clock	Slow-clock	TM0/TM1	T2	Wakeup event
FAST	FIRC	Fast-clock	Run	Set by SLOWSTP	Run	Run	X
SLOW	SIRC	Slow-clock	Set by FASTSTP	Run	Run	Run	X
IDLE	SIRC	Stop	Stop	Run	Stop	Run	WKT/IO/T2
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) Enable Slow-clock (SLOWSTP=0)

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

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

◇ Example: Switch FAST mode to SLOW mode.

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

- **SLOW mode switches to FAST mode**

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

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

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

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

```
BCX     FASTSTP     ; 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)

(2) Switch T2 clock source to Slow-clock (T2CKS=0)

(3) Execute SLEEP instruction

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

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

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

STOP Mode Setting

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

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

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

(3) Execute SLEEP instruction

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

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

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

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	R/W
Reset	–	–	–	0	1	0	1	1

0Fh.4 **SLOWSTP:** Stop Slow-clock in Stop Mode
0: Slow-clock keeps running
1: Slow-clock stops running in Power-down mode

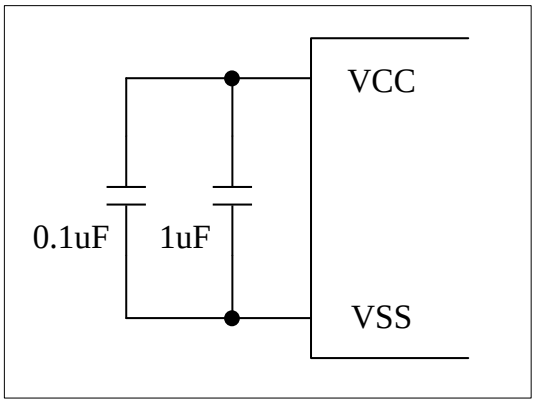
0Fh.3 **FASTSTP:** Stop Fast Clock
0: Fast Clock keeps running
1: Fast Clock stops running

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

0Fh.1~0 **CPUPSC:** System clock Prescaler
0: divided by 8
1: divided by 4
2: divided by 2
3: divided by 1

3.3 System Clock Oscillator

In the Fast Internal RC (FIRC) mode, the on-chip oscillator generates 18.432 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.



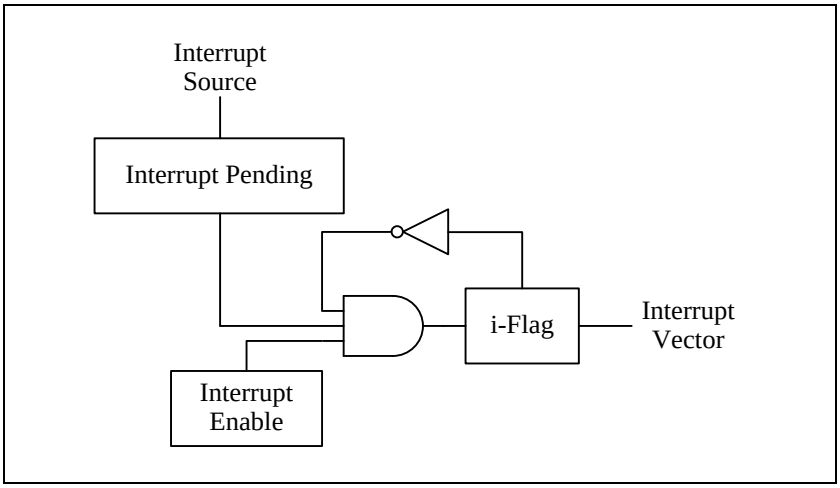
Internal RC Mode

4. Interrupt

TM56F1542 has 1 level, 1 vector and 12 interrupt sources. Each interrupt source has corresponding enable control bit. An interrupt event will set its individual pending flag no matter the enable control bit is 0 or 1.

If the corresponding interrupt enable bit (INTIE or INTIE1) has been set, it would trigger CPU to service the interrupt. CPU accepts interrupt in the end of current executed instruction cycle. 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 (PA4) interrupt request with rising edge trigger.

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

        ORG         004h            ; All interrupt vector
        LGOTO       INT             ; If INT1 (PA4) input occurred rising edge

        ORG         100h
START:
        MOVLW       xxxxxx00b
        MOVWX       PAMODH          ; select INT1 Pin Mode as Mode0

        MOVLW       xxx1xxxxb
        MOVWX       PAD             ; Release INT1, it becomes Schmitt-trigger
                                    ; input with pull-up resistor
        MOVLW       xx1xxxxxb
        MOVWX       OPTION          ; Set INT1 interrupt trigger as rising edge
        MOVLW       111111 0 1b
        MOVWX       INTIF           ; Clear INT1 interrupt request flag
        MOVLW       000000 1 0b
        MOVWX       INTIE           ; Enable INT1 interrupt
MAIN:
        ….
        LGOTO       MAIN
INT:
        MOVWX       RAM20h          ; Store W data to RAM 20h
        MOVXW       STATUS          ; Get STATUS data
        MOVWX       RAM21h          ; Store SATAUS data to RAM 21h
CHKI1:
        BTXSC       INT1IE          ;
        BTXSS       INT1IF          ;
        LGOTO       END_CHK         ; if INT1IE=0 or INT1IE=1 & INT1IF=0
        LCALL       INT1INT         ; if INT1IE=1 & INT1IF=1
        ….                          ; CALL INT1 interrupt service routine

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

INT1INT:
        MOVLW       11111101b
        MOVWX       INTIF           ; clear INT1IF
        …..
        RET
```

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

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

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

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

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

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

INTIE.2 **INT2IE:** INT2 (PA7) interrupt enable
0: disable
1: enable

INTIE.1 **INT1IE:** INT1 (PA4) interrupt enable
0: disable
1: enable

INTIE.0 **INT0IE:** INT0 (PA0) interrupt enable
0: disable
1: enable

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

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

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

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

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

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

0Ch.2 **INT2IF:** INT2 (PA7) pin falling interrupt pending flag
This bit is set by H/W at INT2 pin's falling edge, write 0 to this bit will clear this flag

0Ch.1 **INT1IF:** INT1 (PA4) pin falling/rising interrupt pending flag
This bit is set by H/W at INT1 pin's falling/rising edge, write 0 to this bit will clear this flag

0Ch.0 **INT0IF:** INT0 (PA0) pin falling/rising interrupt pending flag
This bit is set by H/W at INT0 pin's falling/rising edge, write 0 to this bit will clear this flag

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

81h.6 INT0EDG**:** INT0 interrupt trigger edge
0: falling edge trigger, 1: rising edge trigger
81h.5 INT1EDG**:** INT1 interrupt trigger edge
0: falling edge trigger, 1: rising edge trigger

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

INTIE1.7 **LVDIE:** LVD interrupt enable
0: disable
1: enable
INTIE1.3 **TKIE:** Touch Key interrupt enable
0: disable
1: enable
INTIE1.2 **I2CIE:** Slave I2C interrupt enable
0: disable
1: enable
INTIE1.0 **PXIE:** Pin Change Wakeup interrupt enable
0: disable
1: enable

1Ah	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTIF1	LVDIF	TKM2IF	-	TKM0IF	TKIF	I2CIF	-	PXIF
R/W	R/W	R/W	-	R/W	R/W	R/W	-	R/W
Reset	0	0	-	0	0	0	-	0

1Ah.7 **LVDIF:** Low Voltage Detect interrupt pending flag
This bit is set by H/W , write 0 to this bit will clear this flag
1Ah.6 **TKM2IF:** Touch Key module2 interrupt pending flag
This is set by H/W after end of TK2 conversion, write 0 to clear this bit or write 1 to TKM2SOC will clear this flag
1Ah.4 **TKM0IF:** Touch Key module0 interrupt pending flag
This is set by H/W after end of TK0 conversion, write 0 to clear this bit or write 1 to TKM0SOC will clear this flag
1Ah.3 **TKIF:** Touch Key interrupt pending flag,
set by H/W while TKM0 or TKM2 are end of conversion, write 0 to this bit will clear all of Touch Key interrupt flag
1Ah.2 **I2CIF:** Slave I2C interrupt pending flag
This bit is set by H/W while

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

Write 0 to this bit will clear this flag and slave I2C related flags.
1Ah.0 **PXIF:** Pin Change interrupt pending flag
This bit is set by H/W while the corresponding pin change, write 0 to this bit will clear this flag

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

113h.5 **TXD1F:** Slave I2C transmitting data register 1 flag
This bit is set by H/W while I2CTXD1 data transmitting finished, write 0 to this bit will clear this flag

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

113h.3 **RCD1OVF:** Slave I2C receiving data register 1 overflow
This bit is set by H/W while receiving I2CRCD1 overflow, write 0 to this bit will clear this flag

113h.2 **RCD1F:** Slave I2C receiving data register 1 flag
This bit is set by H/W while data receiving to I2CRCD1 finished, write 0 to this bit will clear this flag

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

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

5. I/O Port

5.1 PA0-4, PA7, PB0-4 and PD0-6

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

These pins can operate in four different modes as below.

Mode	PA0~PA4, PA7, PB0~PB4, PD0~PD6 pin function	PxD SFR data	Pin State	Resistor Pull-up	Digital Input
Mode 0	Open Drain	0	Drive Low	N	N
	Input	1	Pull-up	Y	Y
Mode 1	Open Drain	0	Drive Low	N	N
		1	Hi-Z	N	Y
Mode 2	CMOS Output	0	Drive Low	N	N
		1	Drive High	N	N
	Touch Key	0	TK	N	N
Mode 3	ADC	0	–	N	N
	Wakeup	1	–	Y	Y
		0	–	N	Y

I/O Pin Function Table

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

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

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

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

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

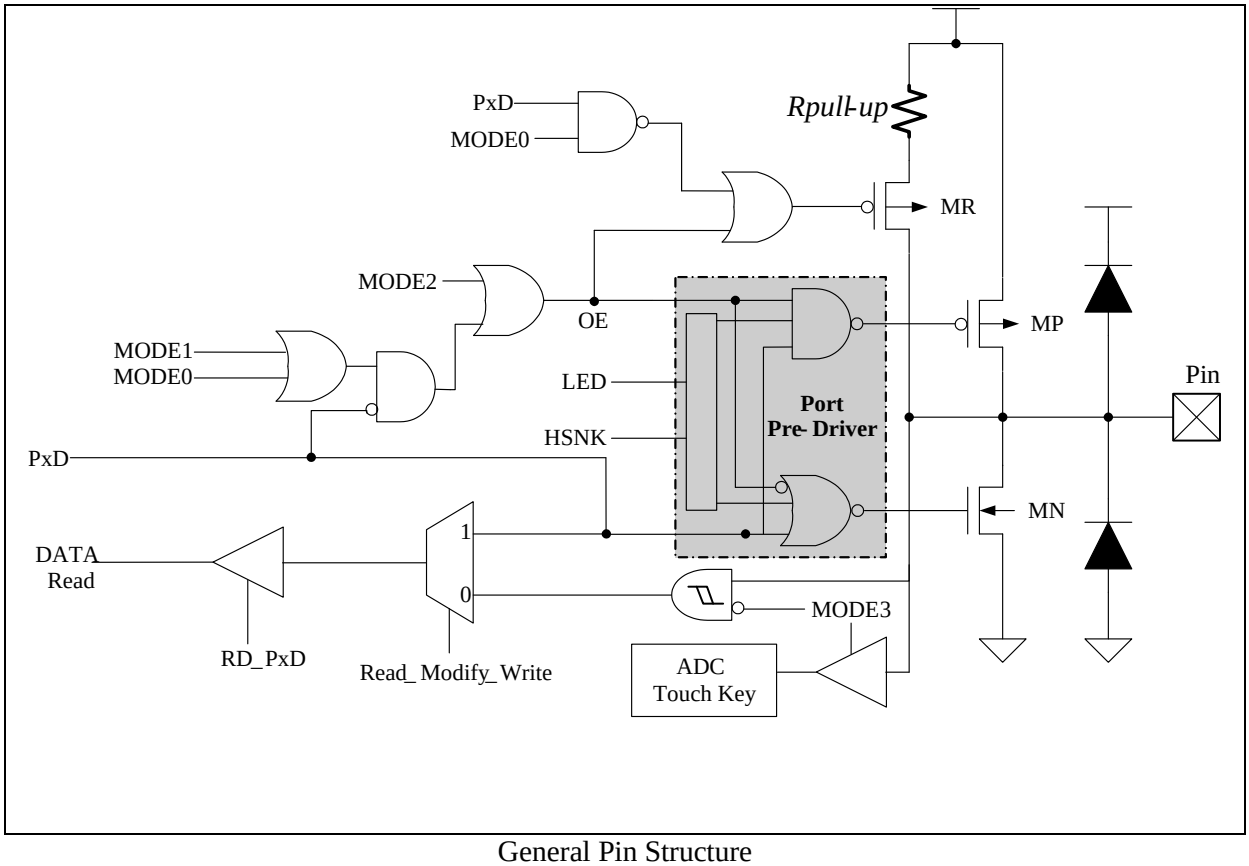
Pin Name	Wake-up	CKO	ADC/TK	others	Mode3
PA0	Wakeup / INT0			PWM0N / I2CSDA	
PA1	Wakeup			PWM1 / I2CSCL	
PA2	Wakeup			PWM2 / TM0CKI	
PA3	Wakeup			PWM3	
PA4	Wakeup / INT1			PWM4	
PA7	Wakeup / INT2			PWM0P / PWM5	
PB0	Wakeup	TM1OUT	ADC0 / TK0		ADC0
PB1	Wakeup	TCOUT	ADC1 / TK1		ADC1
PB2	Wakeup		ADC2 / TK2		ADC2
PB3	Wakeup		ADC3 / TK3		ADC3
PB4	Wakeup		TK4		
PD0	Wakeup		ADC8 / TK16		ADC8
PD1	Wakeup		ADC9 / TK17		ADC9
PD2	Wakeup		ADC10 / TK18		ADC10
PD3	Wakeup		ADC11 / TK19		ADC11
PD4	Wakeup		ADC12 / TK20		ADC12
PD5	Wakeup		ADC13 / TK21		ADC13
PD6	Wakeup		TK22		

PortA/B/D multi-function Table

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

Alternative Function	Mode	PxD SFR data	Pin State	Other necessary SFR setting
INT0, INT1 TM0CKI	**0**	1	Input with Pull-up	INTxIE TM0CTL
	1	1	Input	
TK0~TK3, TK16~TK22	**2**	0	Touch Key Idling, CMOS output Low	
			Touch Key Scanning	TKMxCHS
AD0~AD3, AD8~AD13	**3**	X	ADC Channel	ADCHS
PWM0N, PWM0P, PWM1/2/3/4/5	**1**	X	PWM Output (Open Drain)	PWM0N PWM0P PWM1/2/3/4/5
	2	X	PWM Output (COMS Output)	
I2CSCL	**0**	1	Input with Pull-up	I2CCTL
	1	1	Input	
I2CSDA	**0**	X	Input with Pull-up/ Open Drain Output	
	1	X	Input / Open Drain Output	
Wake-up	**3**	0	Input	
		1	Input with Pull-up	

Mode Setting for Port Alternative Function



General Pin Structure

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

05h.7 **PAD:** PA7 data
05h.4~0 **PAD:** PA4~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

06h.3~0 **PBD:** PB3~PB0 data

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

08h.6~0 **PDD:** PD6~PD0 data

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

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

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

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

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

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

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

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

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

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

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

88h.7~6 **PB3MOD:** PB3 Pin Mode Control
00: Mode0
01: Mode1
10: Mode2
11: Mode3, PB3 as ADC3 channel input

88h.5~4 **PB2MOD:** PB2 Pin Mode Control
00: Mode0
01: Mode1
10: Mode2
11: Mode3, PB2 as ADC2 channel input

88h.3~2 **PB1MOD:** PB1 Pin Mode Control
00: Mode0
01: Mode1
10: Mode2
11: Mode3, PB1 as ADC1 channel input

88h.1~0 **PB0MOD:** PB0 Pin Mode Control
00: Mode0
01: Mode1
10: Mode2
11: Mode3, PB0 as ADC0 channel input

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

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

8Dh.3~2 **PD5MOD:** PD5 Pin Mode Control
00: Mode0
01: Mode1
10: Mode2
11: Mode3, PD5 as ADC13 channel input

8Dh.1~0 **PD4MOD:** PD4 Pin Mode Control
00: Mode0
01: Mode1
10: Mode2
11: Mode3, PD4 as ADC12 channel input

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

8Eh.7~6 **PD3MOD:** PD3 Pin Mode Control
00: Mode0
01: Mode1
10: Mode2
11: Mode3, PD3 as ADC11 channel input

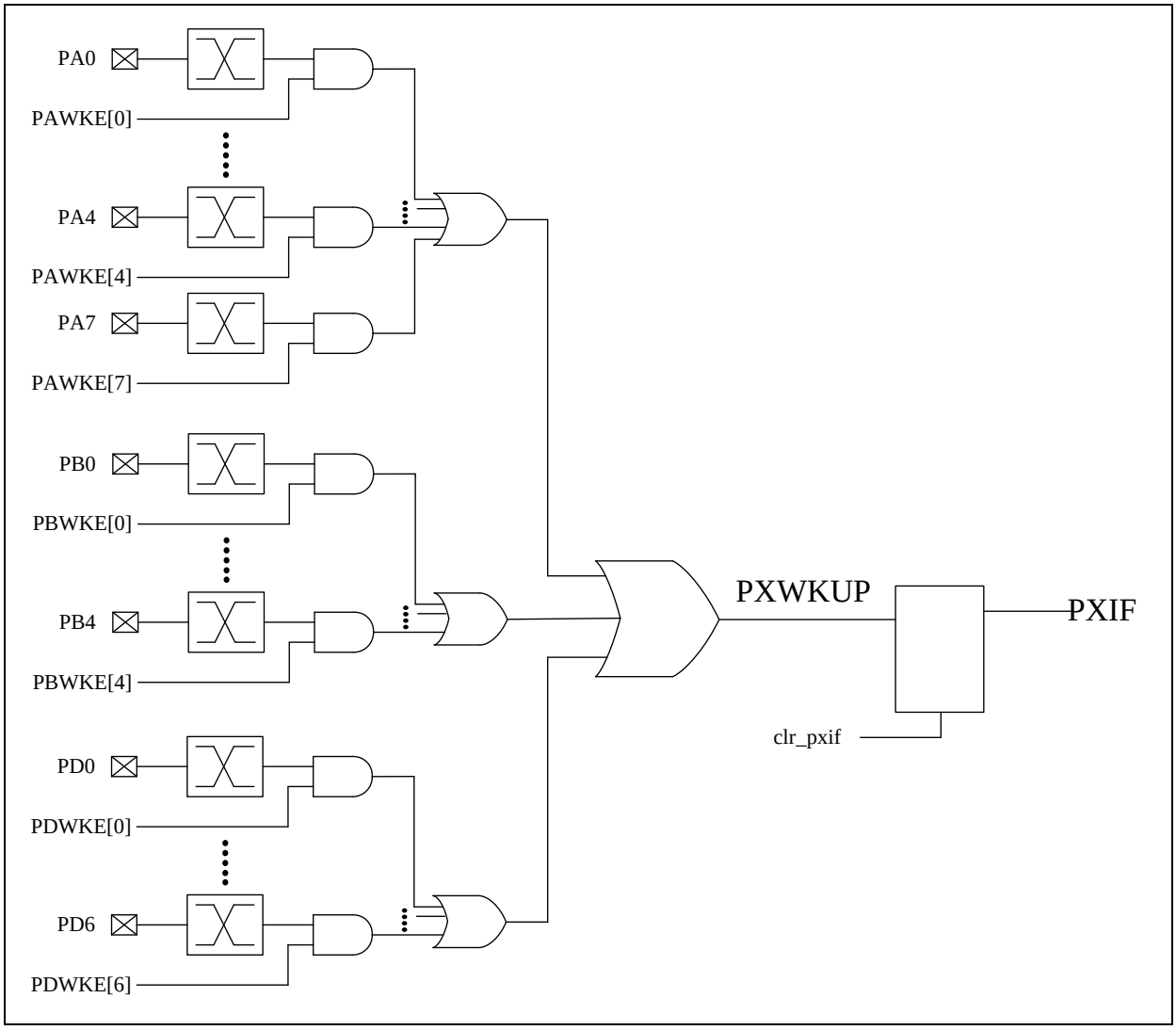
8Eh.5~4 **PD2MOD:** PD2 Pin Mode Control
00: Mode0
01: Mode1
10: Mode2
11: Mode3, PD2 as ADC10 channel input

8Eh.3~2 **PD1MOD:** PD1 Pin Mode Control
00: Mode0
01: Mode1
10: Mode2
11: Mode3, PD1 as ADC9 channel input

8Eh.1~0 **PD0MOD:** PD0 Pin Mode Control
00: Mode0
01: Mode1
10: Mode2
11: Mode3, PD0 as ADC8 channel input

5.2 Pin Interrupt

Pin interrupts include INT0, INT1,INT2 and PA0-4, PA7, PB0-4, and PD0-6 Change Interrupt. These pins are also able to wake up system from STOP mode. INT0 and INT1 are falling edge or low level triggered, INT2 is falling edge triggered. IO pins Change Interrupt will be triggered by any pin state change when the pin wakeup register PxWKE (x=A, B or D) and corresponding Pin Change Interrupt Enable bit (PXIE) is set to enable the function.



◇ Example: Setup Port B as Pin Change interrupts.

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

        ORG     004h        ; All interrupt vector
        LGOTO   INT         ; If PXINT (PB) pin change occurred

        ORG     100h
START:
        MOVLW   00000000b
        MOVWX   PBMODH      ; set PB Pin Mode as Mode0
```

```
        MOVWX     PBMODL

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

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

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

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

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

1Ch.7 **PAWKE:** PA7 individual wakeup enable
1Ch.4~0 **PAWKE:** PA4~PA0 individual wakeup enable

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

1Dh.4~0 **PBWKE:** PB4~PB0 individual wakeup enable

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

1Fh.6~0 **PDWKE:** PD6~PD0 individual wakeup enable

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

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

1Ah	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTIF1	LVDIF	TKM2IF	-	TKM0IF	TKIF	I2CIF	-	PXIF
R/W	R/W	R/W	-	R/W	R/W	R/W	-	R/W
Reset	0	0	-	0	0	0	-	0

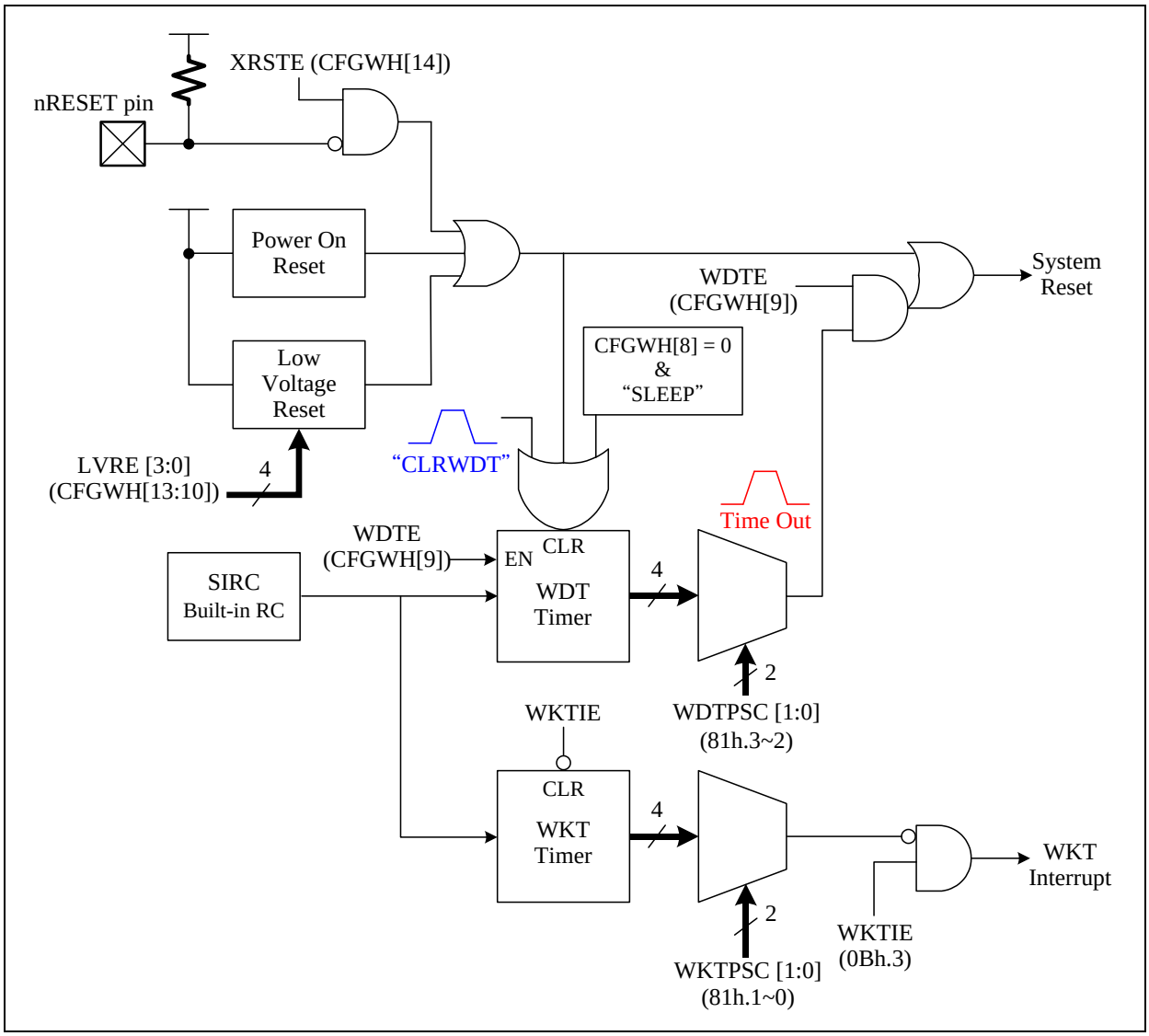
1Ah.0 **PXIF:** Pin Change interrupt pending flag
This bit is set by H/W while the corresponding pin change, write 0 to this bit will clear this flag

6. Peripheral Functional Block

6.1 Watchdog (WDT) /Wakeup (WKT) Timer

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

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



WDT/WKT Block Diagram

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

◇ Example: Clear watchdog timer by CLRWDT instruction.

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

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

```
        MOVLW       0000**01**11b
        MOVWX       OPTION      ; Select WDT Time out=256 ms @5V

        SLEEP
```

◇ Example: Set WKT period and interrupt function.

```
        MOVLW       000001**10b**
        MOVWX       OPTION      ; Select WKT period=64 ms @5V.
        MOVLW       1111**0**111b   ; Clear WKT interrupt request flag by using byte operation
        MOVWX       INTIF       ;Don't use bit operation "BCX WKTIF" clear INTIF

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

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

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

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

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

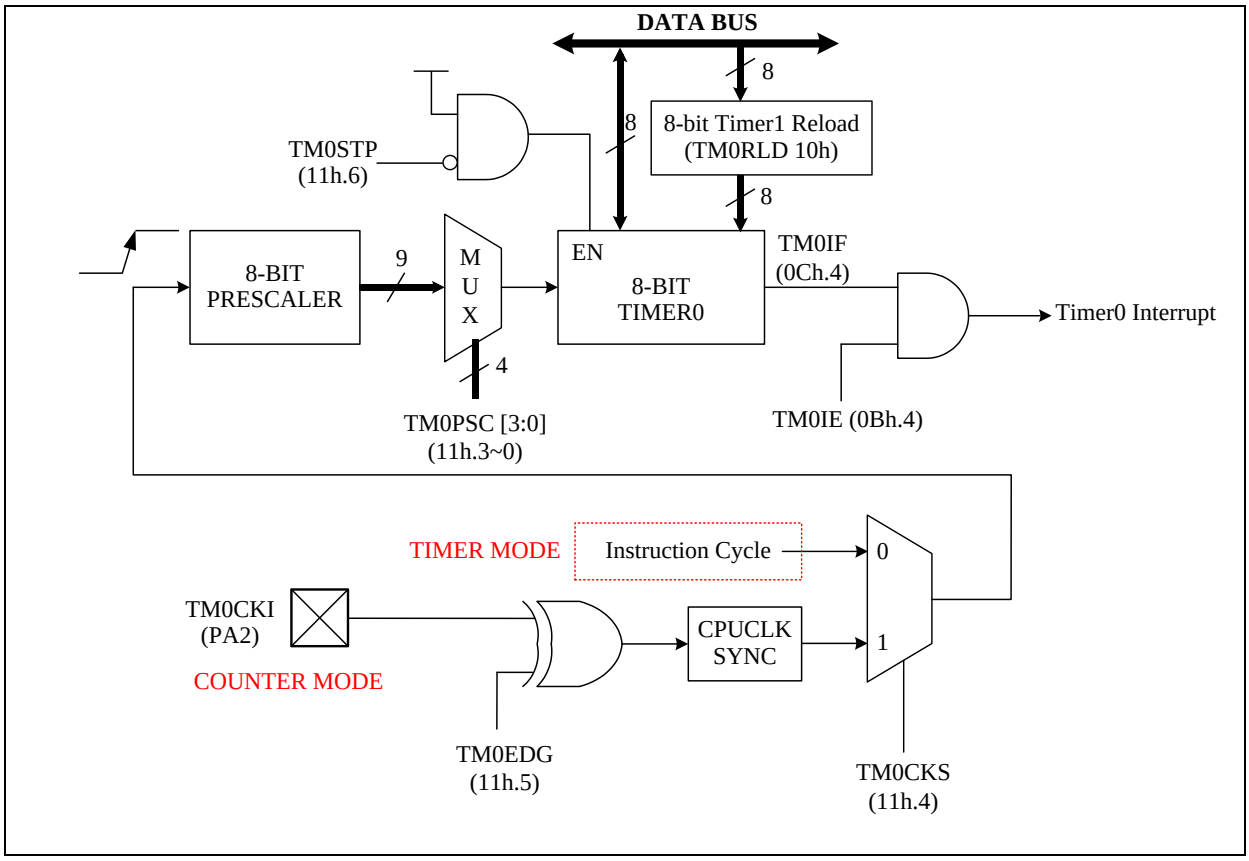
81h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
OPTON	HWAUTO	INT0EDG	INT1EGE	-	WDTPSC		WKTPSC	
R/W	R/W	R/W	R/W	-	R/W	R/W	R/W	R/W
Reset	0	0	0	-	1	1	1	1

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

81h.1~0 **WKTPSC:** WKT period (@VCC=5V)
00: 21ms
01: 41 ms
10: 82 ms
11: 164 ms

6.2 Timer0

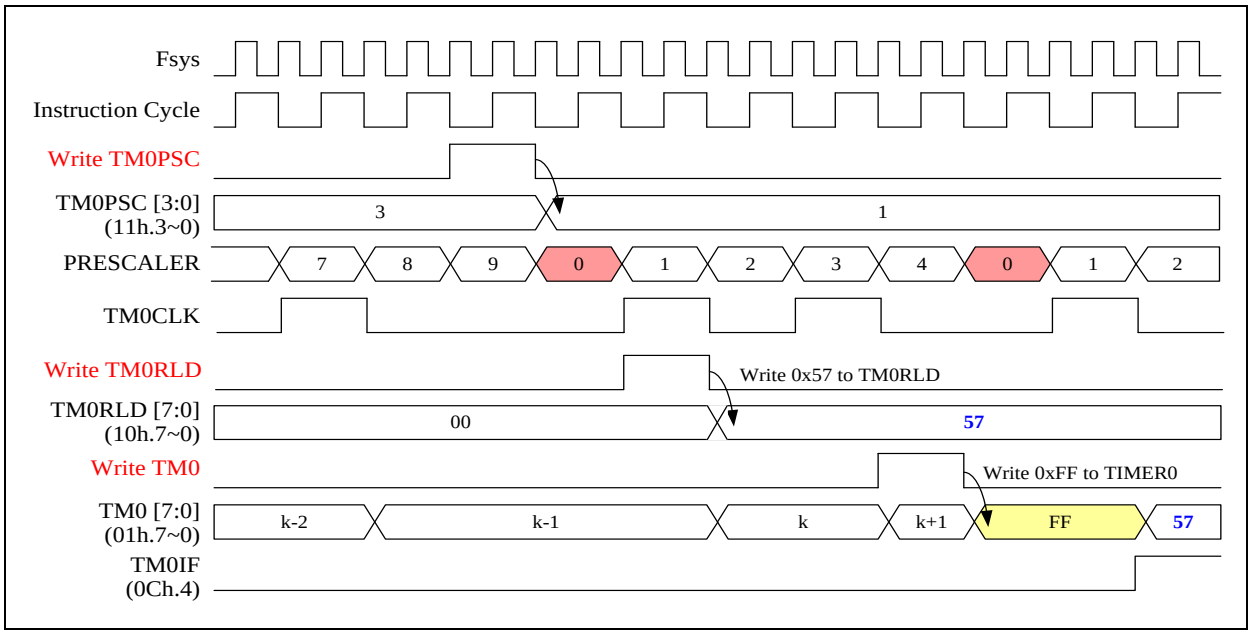
TM0 (01h.7~0) is an 8-bit wide register. It can be read or written as any other register. Besides, Timer0 increases itself periodically and automatically rolls over a new "offset value" (TM0RLD) while it rolls over based on the pre-scaled clock source, which can be Fsys/2 when TM0CKS=0, or TM0CKI (PA2) rising/falling input when TM0CKS=1. The Timer0 increase rate is determined by “Timer0 Pre-Scale” (TM0PSC) register. The Timer0 always generates TM0IF when its count rolls over. It generates Timer0 Interrupt if (TM0IE) is set. Timer0 can be stopped counting if the TM0STP bit is set.



Timer0 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 TM0 interrupt time value is as following:

TM0 interrupt interval cycle time=Fsys / 2 / TM0PSC / (256-TM0RLD)

◇ Example: Setup TM0 work in Timer mode

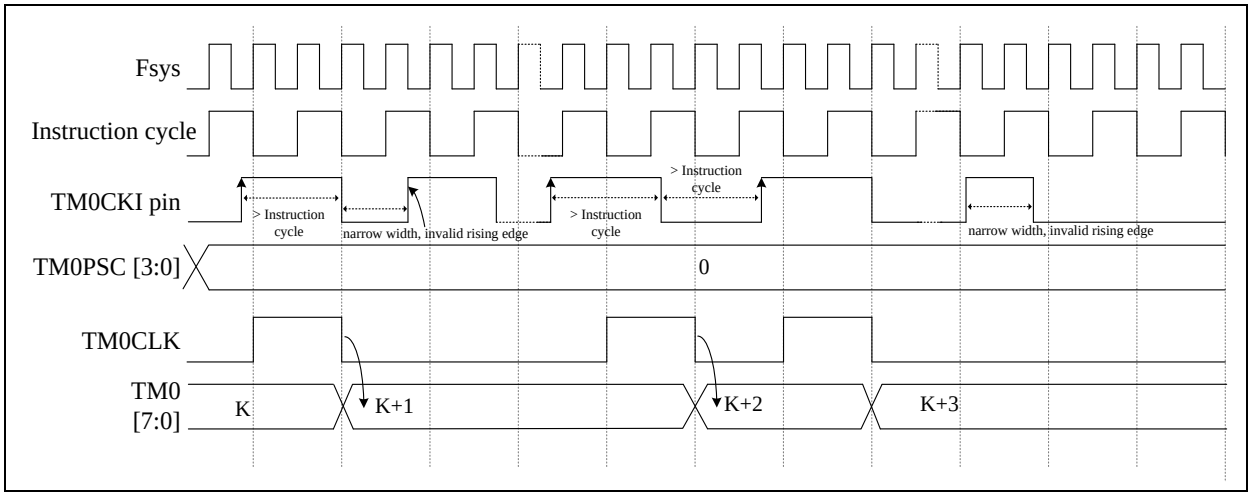
```
; Setup TM0 clock source and divider
        MOVLW   0000 0101B      ; TM0CKS=0, Setup TM0 clock= Fsys/2
        MOVWX   TM0CTL          ; TM0PSC=5, TM0PSC= Fsys/64

; Set TM0 timer.
        BSX     TM0STP          ; Disable TM0 counting (Default "0").
        MOVLW   03h
        MOVWX   TM0             ; Write 0x03 value to TM0 register
        MOVLW   80h
        MOVWX   TM0RLD          ; Set Timer0 reloado data = 128

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

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

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



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

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

```
; Setup TM0 clock source from TM0CKI pin (PA2) and divider.
        MOVLW   00110000b
        MOVWX   TM0CTL      ; TM0EDG=1
                            ; Select TM0 prescaler counting edge=falling edge.
                            ; TM0CKS=1, Setup TM0 clock=TM0CKI pin (PA2)
                            ; TM0PSC=0
                            ; TM0 clock prescaler= TM0CKI divided by 1

; Set TM0 timer and stop TM0 counting.
        BSX     TM0STP      ; Disable TM0 counting (Default "0").
        MOVLW   00h
        MOVWX   TM0         ; Clear Timer0 content.

; Start TM0 count and read TM0 counter.
        BCX     TM0STP      ; Enable TM0 counting.
        NOP
        NOP
        NOP
        BSX     TM0STP      ; Disable TM0 counting (Default "0")

        MOVXW   TM0
```

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

01h **TM0:** Timer0 content

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

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

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

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

10h **TM0RLD:** Timer0 Reload Data

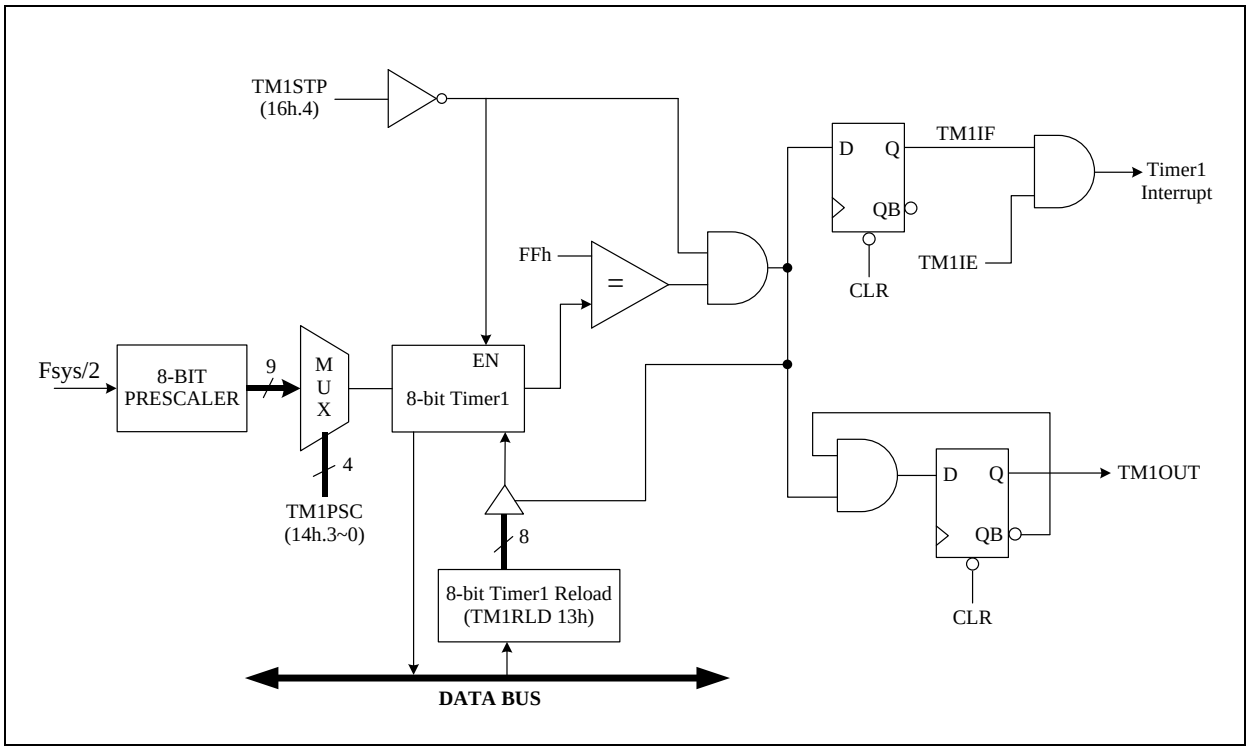
11h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TM0CTL	-	TM0STP	TM0EDG	TM0CKS	TM0PSC			
R/W	-	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	-	0	0	0	0	0	0	0

11h.6 **TM0STP:** Timer0 counter stop
0: Release 1: Stop counting
11h.5 **TM0EDG:** Timer0 prescaler counting edge for TM0CKI pin
0: rising edge 1: falling edge
11h.4 **TM0CKS**: Timer0 prescaler clock source
0:Fsys/2 1: TM0CKI pin (PA2 pin)
11h.3~0 **TM0PSC**: Timer0 prescaler. Timer0 prescaler clock source divided by

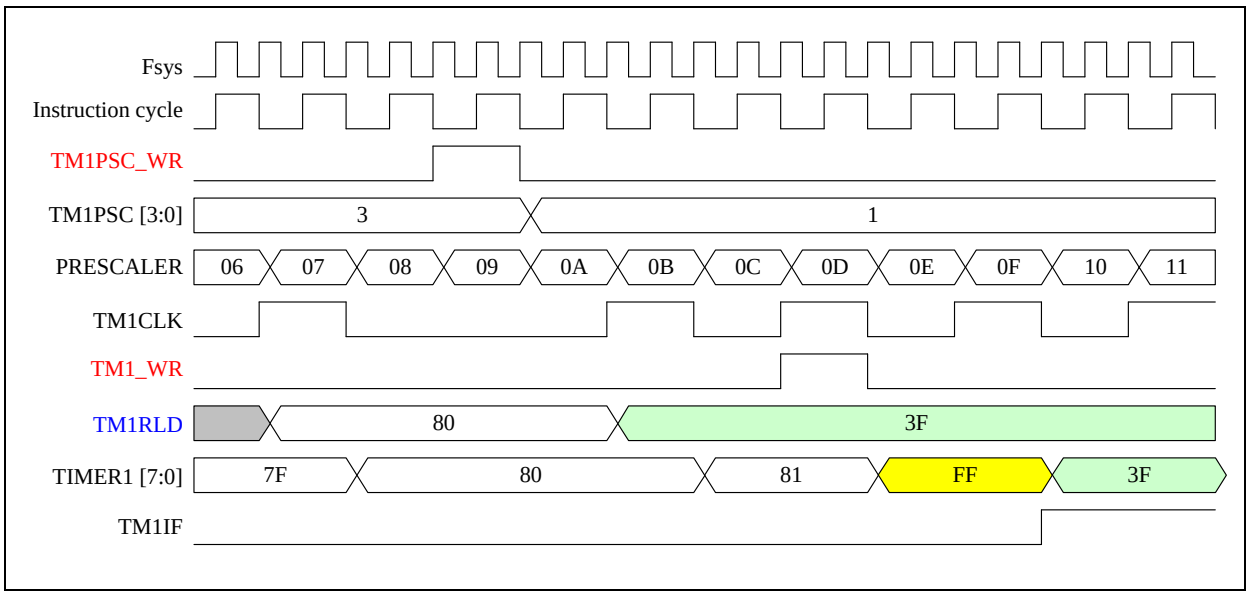
0000: /1	0001: /2	0010: /4	0011: /8
0100: /16	0101: /32	0110: /64	0111: /128
1000: /256	1001: /512	1010: /1024	1011: /2048
1100: /4096	1101: /8192	1110: /16384	1111: /32768

6.3 Timer1

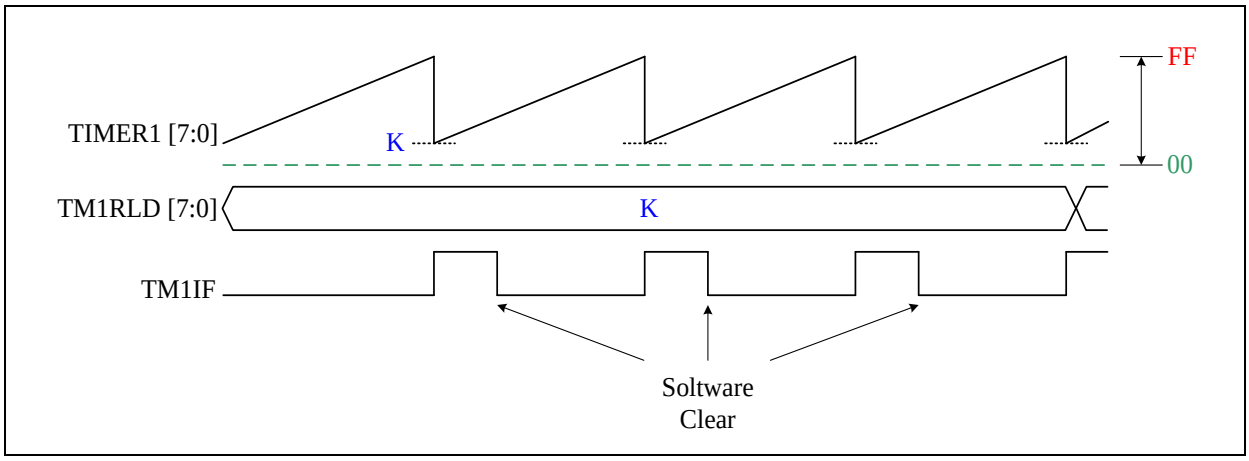
The Timer1 is an 8-bit wide register. It can be read or written as any other register. Besides, Timer1 increases itself periodically and automatically reloads a new "offset value" (TM1RLD) while it rolls over based on the pre-scaled instruction clock (Fsys/2). The Timer1 increase rate is determined by TM1PSC register. Set the TM1STP bit will stop Timer1 counting. TM1OUT is an output signal that toggles when Timer1 overflow.



Timer1 Block Diagram



Timer1 Timing Diagram



Timer1 Reload Diagram

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

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

; Set TM1 timer offset and stops TM1 counting
        BSX     TM1STP      ; Stop TM1 counting (Default "0").
        MOVLW   F0h
        MOVWX   TM1         ; Write F0h into TM1 counter

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

        MOVLW   00100000b   ; Enable TM1 interrupt function.
        MOVWX   INTIE

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

Example:

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

TM1RLD=0xF0,

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

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

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

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

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

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

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

12h **TM1:** Timer1 content

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

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

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

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

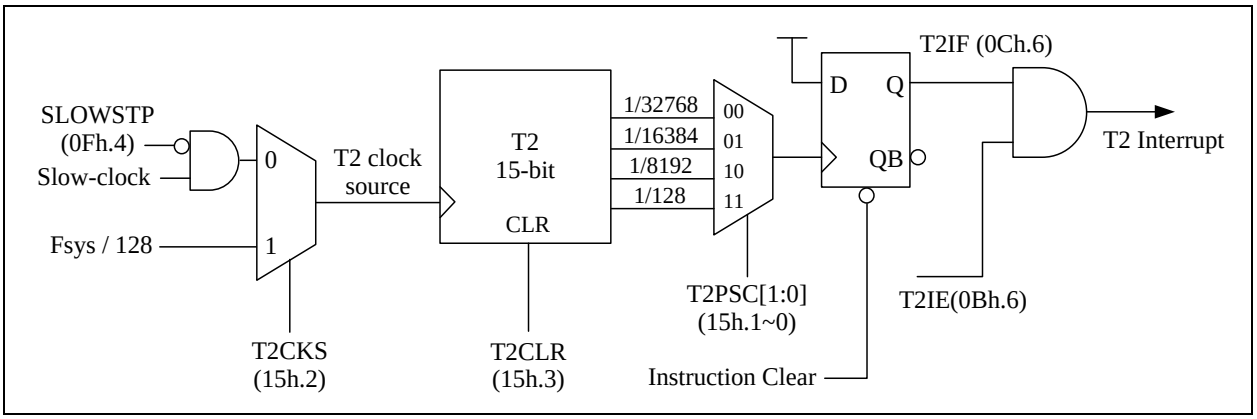
14h.3~0 **TM1PSC:** Timer1 prescaler. Timer1 clock source divided by
0000: Fsys/2 0001: Fsys/4 0010: Fsys/8 0011: Fsys/16
0100: Fsys/32 0101: Fsys/64 0110: Fsys/128 0111: Fsys/256
1xxx: Fsys/512

98h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
MF098	TCOE	TM1OE	PWMCKS		-	INT1SEL	INT0SEL	-
R/W	R/W	R/W	R/W		-	R/W	R/W	-
Reset	0	0	0	0	-	0	0	-

8Fh.6 **TM1OE:** Enable Timer1 overflow toggle output to PB0 pin (TM1OUT)

6.4 T2:15-bit Timer

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



T2 Block Diagram

Example:

[CPU running at FAST mode, Fsys=Fast-clock= FIRC/4= 4.6 MHz]

◇ Example:

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

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

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

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

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

T2 frequency = 36000 Hz / 16384 ≒ 2.197 Hz

◇Example:

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

◇ Example:

```
; Setup T2 clock source and divider
        MOVLW     00001 000b    ; 15h.3 (T2CLR)=1, Stop T2 counting
        MOVWX     T2CTL         ; 15h.2 (T2CKS) = 0, T2 clock source = Slow-clock
                                ; 15.1~0 (T2PSC) =0, Divided by 32768

; Enable T2 timer and interrupt function.
        MOVLW   10111111b       ; Clear T2 request interrupt flag
        MOVWX   INTIF

        MOVLW   01000000b       ; Enable T2 interrupt function.
        MOVWX   INTIE

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

T2 clock source is Slow-clock = 50KHz, T2PSC = /32768,

T2 frequency = 50000Hz / 32768 ≒ 1.53Hz

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

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

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

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

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	R/W
Reset	-	-	-	0	1	0	1	1

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

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

15h.3 **T2CLR:** T2 counter clear
0: Release 1: Stop counting

15h.2 **T2CKS:** “T2 clock source” selection.
0: Slow-clock ; 1: Fsys/128

15h.1~0 **T2PSC:** T2 prescaler. “T2 clock source” divided by -
00: 32768 01: 16384 10: 8192 11: 128

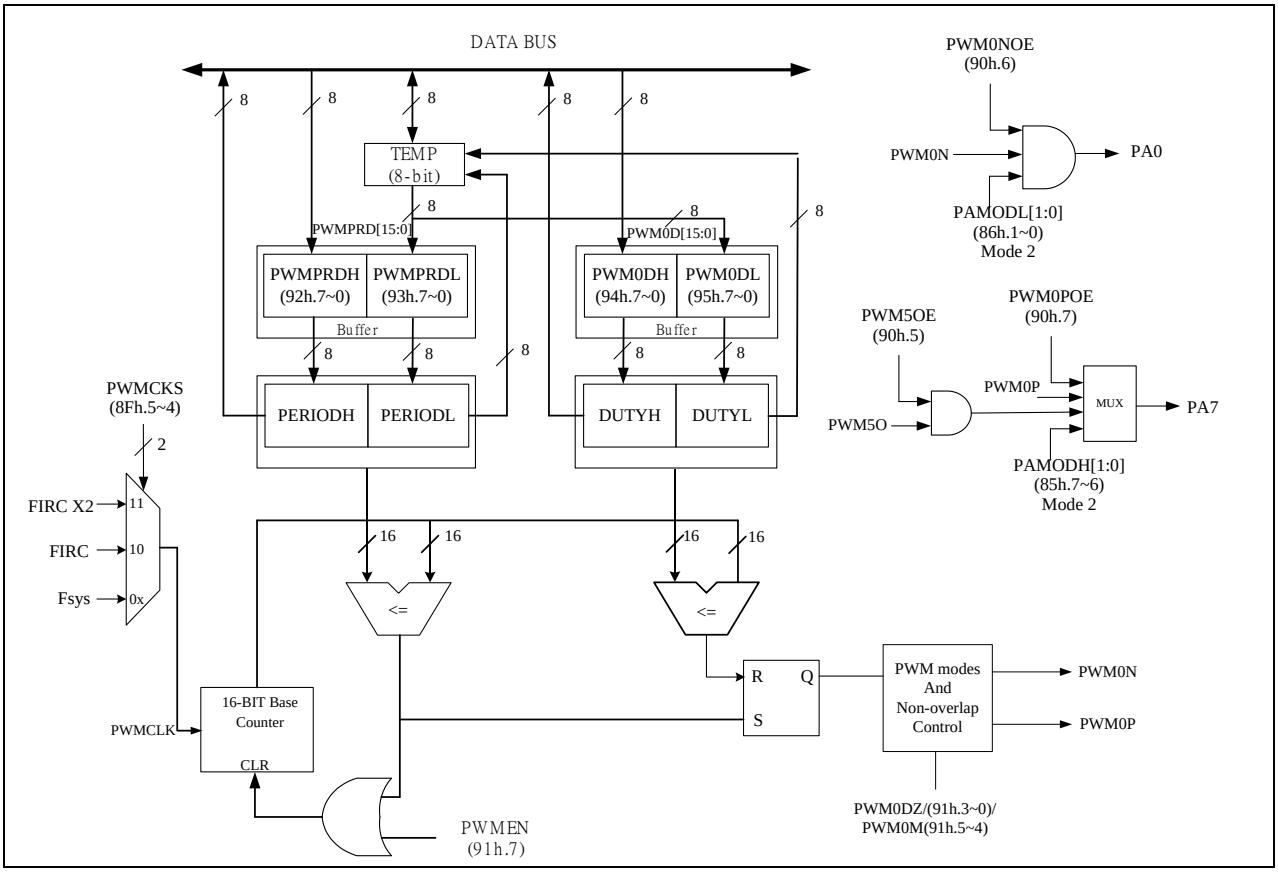
6.5 PWM: Six 16-bit PWM Module

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

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

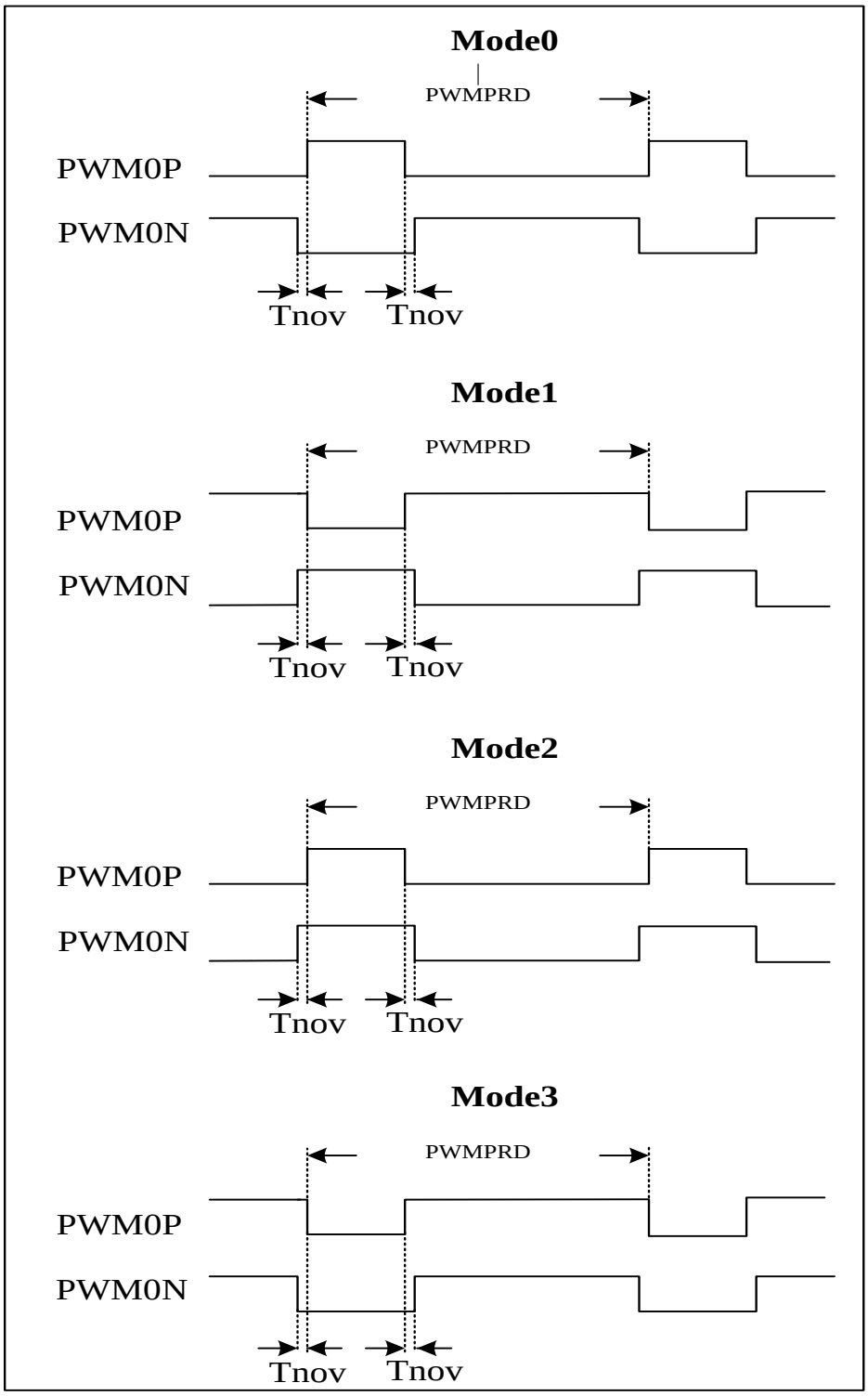
If PWMEN is cleared, the PWM0~5 will be cleared and stopped, otherwise the PWM0~5 remain running. The PWM0 structure is shown as follow. The PWM0 duty cycle can be changed by writing to PWM0DL and PWM0DH. The PWM0 period can be set by writing the period value to PWMPRDL and PWMPRDH register. There is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

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



PWM0 Block Diagram

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



PWM0 Waveform Modes

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

; Setup PWM0 clock prescaler

```
        MOVLW   00000000B   ;
        MOVWX   OPTION2     ; PWMCKS=00B, PWM clk source = Fsys=FIRC 18.432MHz
                            ; Tpwmclk=(1/18.432M)
        MOVLW   20h
        MOVWX   PWMPRDL     ; set PWM period Low byte =20h
                            ; low byte data is stored in TEMP, not in PWMPRD register
        MOVLW   03h
        MOVWX   PWMPRDH     ; set PWM period High byte =03h
                            ; PWMPRD = 0320h (TEMP data is also saved to PWMPRD)
        MOVLW   90h
        MOVWX   PWM0DL      ; set PWM0 Duty Low byte    =90h
                            ; low byte data is stored in TEMP, not in PWM0D register
        MOVLW   01h         ;
        MOVWX   PWM0DH      ; set PWM0 Duty High byte    =01h
                            ; PWM0D = 0190h (TEMP data is also saved to PWM0D)
        MOVLW   10010100B
        MOVWX   PWMCTL      ; set PWMEN=1, PWM0OM=10B=Mode 2,
                            ; PMW0DZ=0100B= non-overlap 4*Tpwmclk
        MOVLW   10xxxx10B   ;
        MOVWX   PAMODL      ; set PA7, PA0 as CMOS output
        MOVLW   11000000B
        MOVWX   PWMOE       ; enable PWM0P output to PA7, enable PWM0N output to PA0
```

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

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

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

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

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

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

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

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

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

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

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

91h.3~0 **PWM0DZ:**PWM0 Dead Zone (non-overlap) control
0000: non-overlap 0*Tpwmclk; Original PWM0
0001: non-overlap 1*Tpwmclk
0010: non-overlap 2*Tpwmclk
…
1101: non-overlap 13*Tpwmclk
1110: non-overlap 14*Tpwmclk
1111: non-overlap 16*Tpwmclk

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

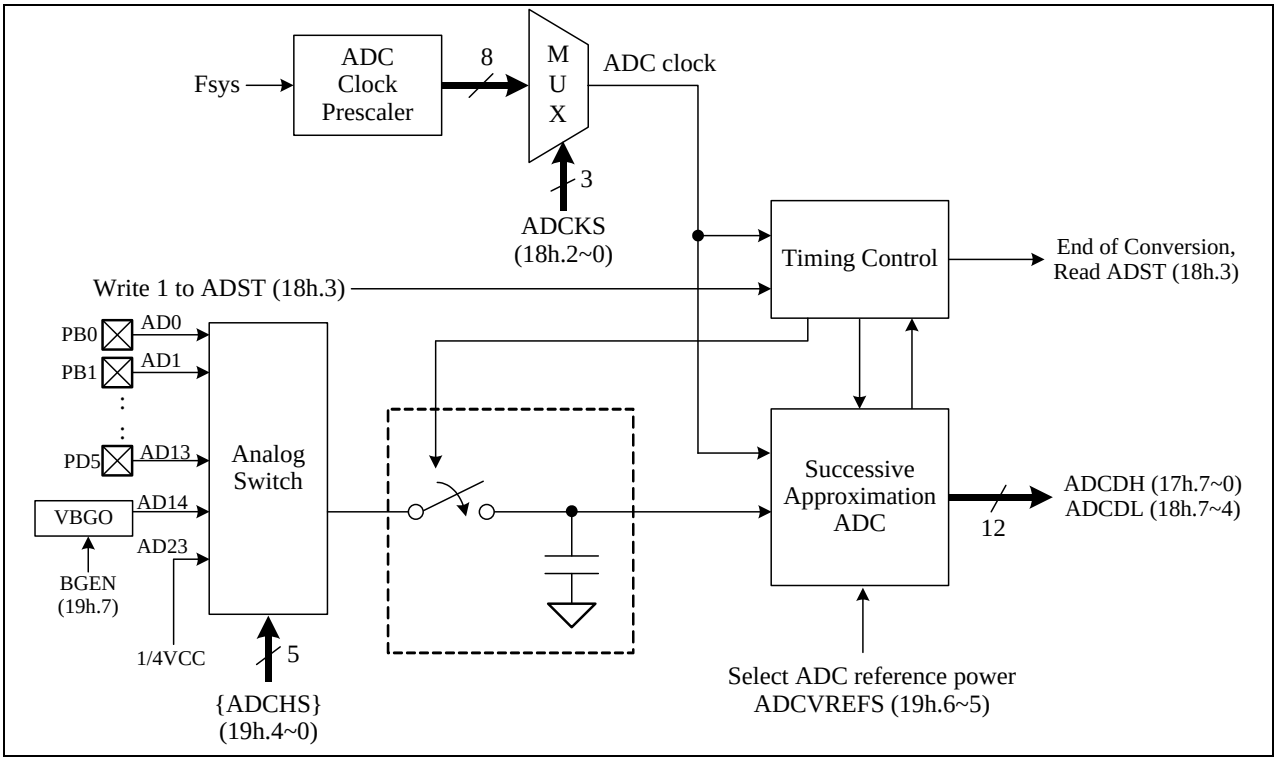
9Eh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWM5DH	PWM5DH							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	0	0	0	0	0	0	0

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

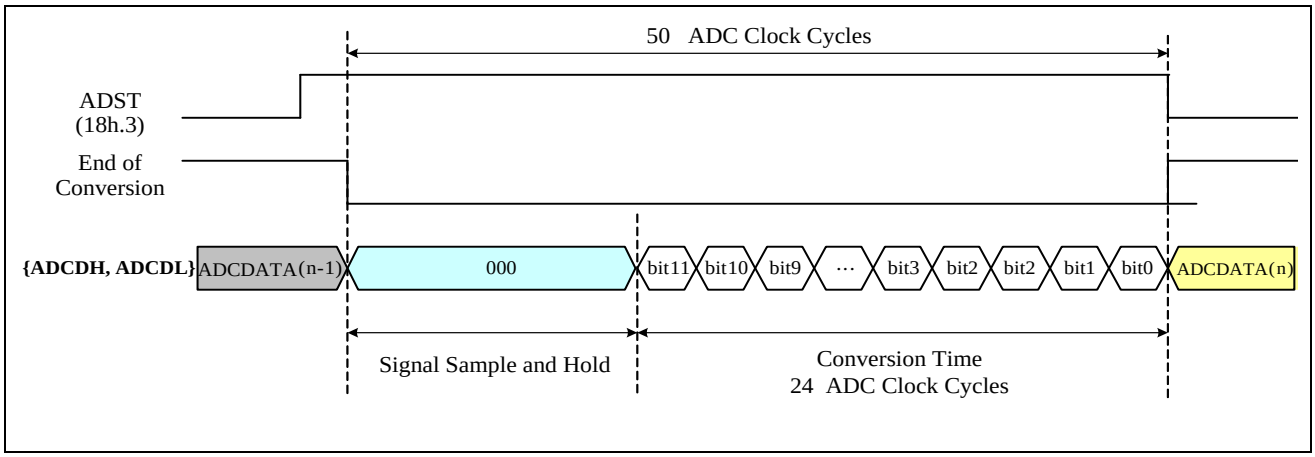
9Fh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWM5DL	PWM5DL							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

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

6.6 Analog-to-Digital Converter



The 12-bit ADC (Analog to Digital Converter) consists of a 10-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. The PxMODx control registers are used for ADC pin configuration, user must set the Pin Mode=3 when the pin is used as an ADC input. The setting can disable the pin logical input path to save power consumption. User needs to set ADCHS (19h.4~0) to choose the input channel of ADC. Furthermore, AD14 is VBG input for ADC and AD23 is 1/4VCC input for ADC. Besides, VBG is controlled by BGEN (19h.7). ADC reference voltage is selected by ADCVREFS (19h.6~5).



Example:

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

ADC clock frequency=FIRC/32, ADC channel=ADC10 (PD2).

◇ Example:

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

MOVLW     0111 0101b    ; ADC10 (PD2) Pin Mode=3=ADC input
MOVWX     PDMODL

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

MOVLW     1 00 01010b   ; 19h.7=1, enable VBG; 19h.6~5=0, VREF=VCC
MOVWX     ADCTL2        ; 19h.4~0 (ADCHS [4:0]) =10, select ADC10 (PD2 pin).

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

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

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

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

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

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

18h.7~4 **ADCDL:** ADC output data LSB, ADQ [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	BGEN	ADCVREFS		ADCHS				
R/W	R/W	R/W		R/W				
Reset	1	0	0		0	0	0	0

F19.7 **BGEN:** Band Gap BG1.25V enable
0: Disable
1: Enable and Auto disable in STOP/IDLE mode

F19.6~5 **ADCVREFS:** ADC VREF select,
00: VCC, 01: 2.5V, 10: 3V, 11: 4V

F19.4~0 **ADCHS:** ADC channel select

00000: ADC0 (PB0)	00100: Reserved	01000: ADC8 (PD0)	01100: ADC12 (PD4)
00001: ADC1 (PB1)	00101: Reserved	01001: ADC9 (PD1)	01101: ADC13 (PD5)
00010: ADC2 (PB2)	00110: Reserved	01010: ADC10 (PD2)	01110: VBGO
00011: ADC3 (PB3)	00111: Reserved	01011: ADC11 (PD3)	10111: 1/4 VCC

6.7 Touch Key

The Touch Key offers an easy, simple and reliable method to implement finger touch detection. In most applications, it doesn't require any external component. The device support 2 modules, 13 channels touch key detection.
To use Touch Key, user must setup Pin Mode (***see Section 5***) correctly as below table. Setting Mode2 for an idle Touch Key pin as CMOS output Low can reduce the mutual interference between the adjacent keys.

PxMODx setting for Touch Key	TK0~TK4, TK16~TK23
Pin is Touch Key, Idling	CMOS output Low (Mode2)
Pin is Touch Key, Scanning	

There are two Touch Key Modules (Module0 and Module2) in the TM56F1542. Each module can work independently. In Touch Key Module, there are two oscillators: Reference Clock (RCKx) and Touch Clock (TCKx), which are connected to the Reference Counter and Data Counter respectively. The frequency of RCKx can be adjusted by setting TKMxREFC. Reference Counter is used to control conversion time. TKMxTCP is to select touch key clock frequency (only available in TKMxJMP=0). When TKMxJMP =1, the touch key clock frequency will automatically change.

From starting touch key conversion to end, it will take 0 to 4095 RCK oscillation cycles by setting TKMxTMR. After end of conversion, user can get TK Data (TKMxDH, TKMxDL) from Data counter. TK Data is affected by finger touching. As finger touching TCK is getting slower, the value of TK Data is smaller than the no finger touching. According to the difference of TKMxDATA, user can check if it is touched of not. A suitable TKMxTMR and TKMxREFC setting can adjust TK Data to adapt the system board circumstances. To get the best TKMxREFC setting, user can try different TKMxREFC value, and then find the one which makes the TK Data and TKMxTMR as close as possible. In the other hand, user can adjust the overall operating frequency of the TK system (including TCKx & RCKx) by setting TKMxFSL (frequency select). For all Touch Key Module, there is a control signal TKMHSENSE (11Fh.7) can be set to enhance the Touch key performance.

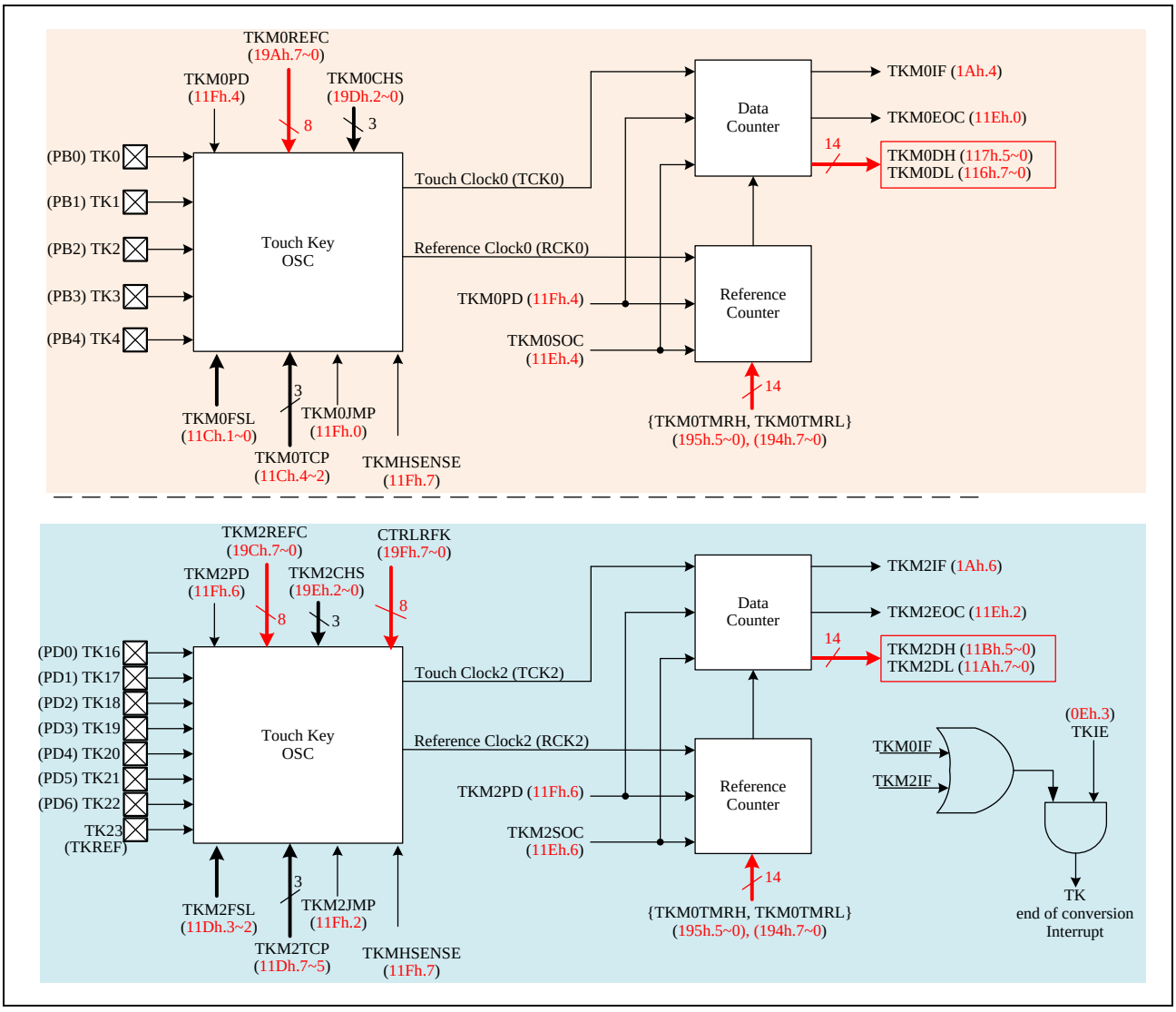
To start the Scanning, user assigns TKMxPD=0, then set the TKMxSOC bit to start touch key conversion, the TKMxSOC bit can be automatically cleared while end of conversion. However, if the SYSCLK is too slow, H/W might fail to clear TKMxSOC due to clock sampling rate. TKMxEOC=0 means conversion is in process. TKMxEOC=1 means the conversion is finish, and the touch key counting result is stored into the 12 bits TK Data Counter TKMxDH and TKMxDL.

TKIF (sum of TKM0IF and TKM2IF) will active at the first time enable Touch Key function (TKMxPD=0), user should clear TKIF after TKMxPD cleared.

Touch Key Channel 23 is reference key. The capacitor of the reference key is control by CTRLRFK (19F.7~0). (00: minimum capacitor on TK23, FF: maximum capacitor on TK23)

TKM0IF	TKM2IF	TKIF	STATE
0	0	0	IDLE
1	0	1	TK Module0 is end of conversion
0	1	1	TK Module2 is end of conversion
1	1	1	TK Module0 and Module2 are both end of conversion

Touch Key Interrupt Flag Description



Touch Key Structure

◇ Example: Use TK module0 and Module2.Touch key channel = TK2 (PB2) and TK20 (PD4).

```
.ORG        000h
            LGOTO  START
.ORG        004h
INT:
            BTXSC       TKIF        ; check TKIF
            LCALL       INT_TK
            RETI
INT_TK:
            BTXSC       TKM0IF      ; check TKM0IF
            LCALL       INT_TKM0
            BTXSC       TKM2IF      ; check TKM2IF
            LCALL       INT_TKM2
            RET
INT_TKM0:
            MOVLW       11101111b   ; clear TKM0IF
            MOVWX       TKMFLG
            MOVXW       TKM0DH      ; read TK0 DATA[13:8] into W register
            …
            MOVXW       TKM0DL      ; read TK0 DATA[7:0] into W register
            …
            RET
INT_TKM2:
            MOVLW       11101111b   ; clear TKM2IF
            MOVWX       TKMFLG
            MOVXW       TKM2DH      ; read TK2 DATA[13:8] into W register
            …
            MOVXW       TKM2DL      ; read TK2 DATA[7:0] into W register
            …
            RET

START:
              …..
SET_MODE:
            MOVLW       xx10xxxxb   ; PBMODL[5:4] = 10b
            MOVWX       PBMODL      ; set PB2 as Mode 2 for touch key input
            BCX         PBD,2       ; clear PB2 as CMOS output low

            MOVLW       xxxxxx10b   ; PDMODH[1:0] = 10b
            MOVWX       PDMODH      ; set PD4 as Mode 2 for touch key input
            BCX         PDD,4       ; clear PD4 as CMOS output low

TK_INIT:
            MOVLW       000 001 10b ;
            MOVWX       TKMCON0     ; set TKM0TCP=1, TKM0FSL=2
            MOVLW       110 0 00 00b ;
            MOVWX       TKMCON1     ; set TKM2TCP=6, TKM2FSL=0, TKM1FSL=1

            MOVLW       00h
            MOVWX       TKM0TMRH
            MOVLW       64h
```

```
            MOVWX     TKM0TMRL   ; TKM0TMR=64h
            MOVLW     01h
            MOVWX     TKM2TMRH
            MOVLW     26h
            MOVWX     TKM2TMRL   ; TKM2TMR =126h

            MOVLW     4Dh
            MOVWX     TKM0REFC   ; TKM0REFC=64h
            MOVLW     8Ah
            MOVWX     TKM2REFC   ; TKM2REFC=8Ah

            MOVLW     00000010b  ; set TKM0CHS=2 (TK2)
            MOVWX     TKMCHS
            MOVLW     00000100b
            MOVWX     TKM2CHS    ; set TKM2CHS=4 (TK20)

            MOVLW     10000100b
            MOVWX     TKMCTL1    ; TKMHSEN = 1, TKM0/TKM2 high sensitivity
                                 ; TMK2PD = TKM0PD=0,
                                 ; enable TMM2 / TKM0 activity
                                 ; TKM2JMP = 1, TKM2 clock mode auto-change
                                 ; TKM0JMP = 0
                                 ;  clock mode fixed(refer to TKM1TCP / TKM0TCP)
            MOVLW     11110111b
            MOVWX     INTIF1     ; clear TKIF
            BSX       TKIE       ; enable TKIE,  enable TK interrupt
TK_START:
             BSX      TKM0SOC    ; start TKM0 conversion
             BSX      TKM2SOC    ; start TKM2 conversion
             …
```

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

INTIE1.3 **TKIE:** Touch Key interrupt enable
0: disable
1: enable

1Ah	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTIF1	LVDIF	TKM2IF	-	TKM0IF	TKIF	I2CIF	-	PXIF
R/W	R/W	R/W	-	R/W	R/W	R/W	-	R/W
Reset	0	0	-	0	0	0	-	0

1Ah.6 **TKM2IF:** Touch Key module2 interrupt pending flag
This is set by H/W after end of TK2 conversion, write 0 to clear this bit or write 1 to TKM2SOC will clear this flag

1Ah.4 **TKM0IF:** Touch Key module0 interrupt pending flag
This is set by H/W after end of TK0 conversion, write 0 to clear this bit or write 1 to TKM0SOC will clear this flag

1Ah.3 **TKIF:** ouch Key interrupt pending flag,
set by H/W while TKM0 or TKM2 are end of conversion, write 0 to this bit will clear all of Touch Key interrupt flag

116h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TKM0DL	TKM0DL							
R/W	R	R	R	R	R	R	R	R
Reset	-	-	-	-	-	-	-	-

116h.7~0 **TKM0DL:** Touch Key Module0 data LSB[7:0]

117h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TKM0DH	-	-	TKM0DH					
R/W	-	-	R	R	R	R	R	R
Reset	-	-	-	-	-	-	-	-

117h.5~0 **TKM0DH:** Touch Key Module0 data MSB[13:8]

11Ah	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TKM2DL	TKM2DL							
R/W	R	R	R	R	R	R	R	R
Reset	-	-	-	-	-	-	-	-

11Ah.7~0 **TKM2DL:** Touch Key Module2 data LSB[7:0]

11Bh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TKM2DH	-	-	TKM2DH					
R/W	-	-	R	R	R	R	R	R
Reset	-	-	-	-	-	-	-	-

11Bh.5~0 **TKM2DH:** Touch Key Module2 data MSB[13:8]

11Ch	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TKMCON0	-	-	-	TKM0TCP			TKM0FSL	
R/W	-	-	-	R/W	R/W	R/W	R/W	R/W
Reset	-	-	-	0	0	1	0	0

11Ch.4~2 **TKM0TCP:** TK module0 touch key clock frequency select; (only available in TKM0JMP=0)

11Ch.1~0 **TKM0FSL:** TK module0 clock(RCK0/TCK0) frequency selection;
00: slowest, …, 11: fastest

11Dh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TKMCON1	TKM2TCP			-	TKM2FSL		-	-
R/W	R/W	R/W	R/W	-	R/W	R/W	-	-
Reset	0	0	0	-	0	0	-	-

11Dh.7~5 **TKM2TCP:** TK module2 touch key clock frequency select; (only available in TKM2JMP=0)

11Dh.3~2 **TKM2FSL:** TK module2 clock(RCK2/TCK2) frequency selection;
00: slowest, …, 11: fastest

11Eh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TKMCTL0	-	TKM2SOC	-	TKM0SOC	-	TKM2EOC	-	TKM0EOC
R/W	-	R/W	-	R/W	-	R/W	-	R/W
Reset	-	0	-	0	-	-	-	-

11Eh.6 **TKM2SOC:** Start Touch Key Module2 conversion
Set 1 to start Touch Key Module2 conversion. If SYSCLK is fast enough, this bit will be cleared by H/W at the end of conversion. S/W can also write 0 to clear this flag

11Eh.4 **TKM0SOC:** Start Touch Key Module0 conversion
Set 1 to start Touch Key Module0 conversion. If SYSCLK is fast enough, this bit will be cleared by H/W at the end of conversion. S/W can also write 0 to clear this flag

11Eh.2 **TKM2EOC:** Touch Key Module2 end of conversion flag, TKM2EOC may have 3uS delay after TKM2SOC=1, so F/W must wait enough time before polling this Flag.
0: Indicates conversion is in progress
1: Indicates conversion is finished

11Eh.0 **TKM0EOC:** Touch Key Module0 end of conversion flag, TKM0EOC may have 3uS delay after TKM0SOC=1, so F/W must wait enough time before polling this Flag.
0: Indicates conversion is in progress
1: Indicates conversion is finished

11Fh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TKMCTL1	TKMHSEN	TKM2PD	-	TKM0PD	-	TKM2JMP	-	TKM0JMP
R/W	R/W	R/W	-	R/W	-	R/W	-	R/W
Reset	0	1	-	1	-	0	-	0

11Fh.7 **TKMHSEN:** TK Module Sensitivity, 1=higher sensitivity; 0=normal.

11Fh.6 **TKM2PD:** Touch Key Module2 power down
0: Touch Key Module2 running
1: Touch Key Module2 power down

11Fh.4 **TKM0PD:** Touch Key Module0 power down
0: Touch Key Module0 running
1: Touch Key Module0 power down

11Fh.2 **TKM2JMP:** Touch Key Module2 clock mode
0: Fix frequency (refer to TKM2TCP**)**
1: Auto-change

11Fh.0 **TKM0JMP:** Touch Key Module0 clock mode
0: Fix frequency (refer to TKM0TCP)
1: Auto-change

194h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TKM0TMRL	TKM0TMRL							
R/W	R/W							
Reset	0	0	0	0	0	0	0	0

194h.7~0 **TKM0TMRL** Touch Key Module0 reference counter LSB[7~0]

195h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TKM0TMRH	-	-	TKM0TMRH					
R/W	-	-	R/W					
Reset	-	-	0	0	0	0	0	0

195h.3~0 **TKM0TMRH**: Touch Key Module0 reference counter MSB[13~8]

198h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TKM2TMRL	TKM2TMRL							
R/W	R/W							
Reset	0	0	0	0	0	0	0	0

198h.7~0 **TKM2TMRL** Touch Key Module2 reference counter LSB[7~0]

199h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TKM2TMRH	-	-	TKM2TMRH					
R/W	-	-	R/W					
Reset	-	-	0	0	0	0	0	0

199h.3~0 **TKM2TMRH**: Touch Key Module2 reference counter MSB[13~8]

19Ah	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TKM0REFC	TKM0REFC							
R/W	R/W							
Reset	0	0	0	0	0	0	0	0

19Ah.7~0 **TKM0REFC** TK module0 Reference clock capacitor select

19Ch	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TKM2REFC	TKM2REFC							
R/W	R/W							
Reset	0	0	0	0	0	0	0	0

19Ch.7~0 **TKM2REFC** TK module2 Reference clock capacitor select

19Dh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TKMCHS0	-	-	-	-	-	TKM0CHS		
R/W	-	-	-	-	-	R/W		
Reset	-	-	-	-	-	0	0	0

19Dh.2~0 **TKM0CHS**: TK module0 Channel Select

000: TK0 (PB0) 001: TK1 (PB1)
010: TK2 (PB2) 011: TK3 (PB3)
100: TK4 (PB4) 101: Reserved
110: Reserved 111: Reserved

19Eh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TKMCHS1	-	-	-	-	-	TKM2CHS		
R/W	-	-	-	-	-	R/W		
Reset	-	-	-	-	-	0	0	0

19Eh.2~0 **TKM2CHS**: TK module2 Channel Select

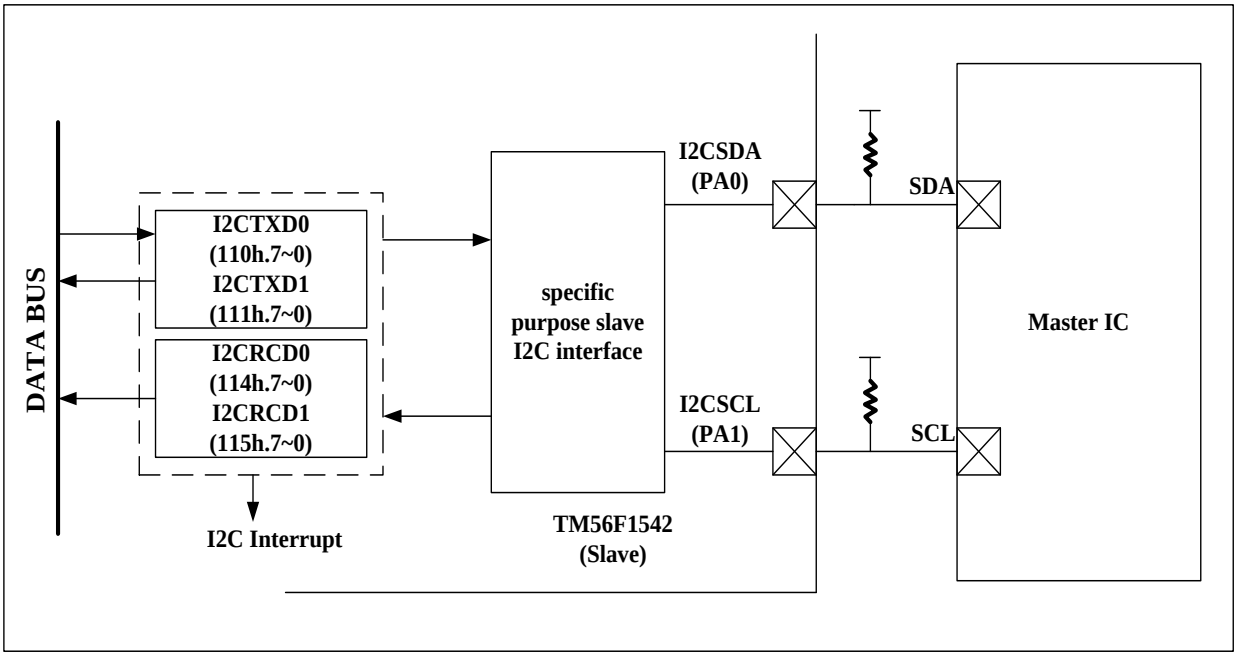
000: TK16 (PD0) 001: TK17 (PD1)
010: TK18 (PD2) 011: TK19 (PD3)
100: TK20 (PD4) 101: TK21 (PD5)
110: TK22 (PD6) 111: TK23 (TKREF)

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

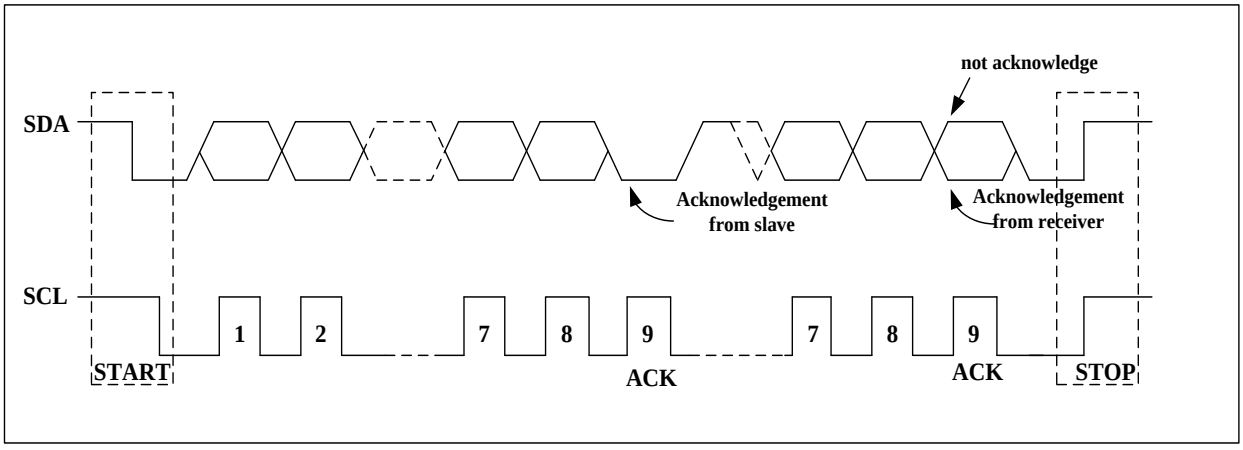
19Fh.7~0 **CTRLRFK**: TK module2 Reference Key(TK23) Capacitor Control

6.8 Specific Purpose Slave I2C Interface

Specific purpose slave I2C interface in TM56F1542 could be used for data transmission. This interface is based on a standard I2C (Inter-Integrated Circuit), and TM56F1542 is always as a slave mode. When the master node (another IC or device) sends the correct ID through I2C, it can read data from the register I2CTXD0 (110h.7~0) and I2CTXD1 (111h.7~0) of TM56F1542 or write data to the register I2CRCD0 (114h.7~0) and I2CRCD1 (115h.7~0) of TM56F1542.

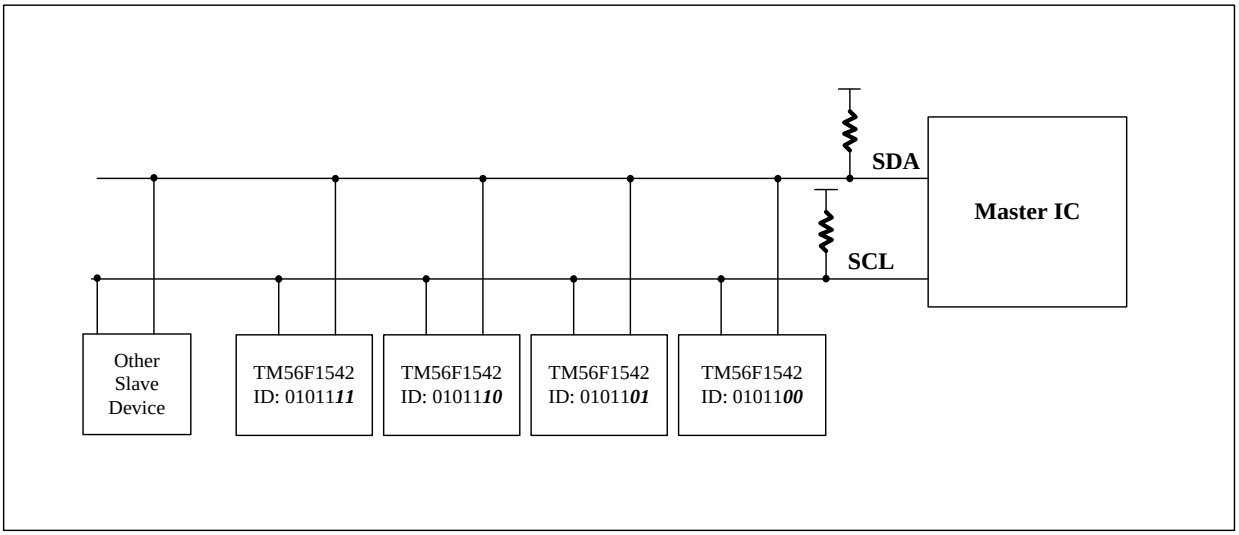


Slave I2C Interface Block Diagram

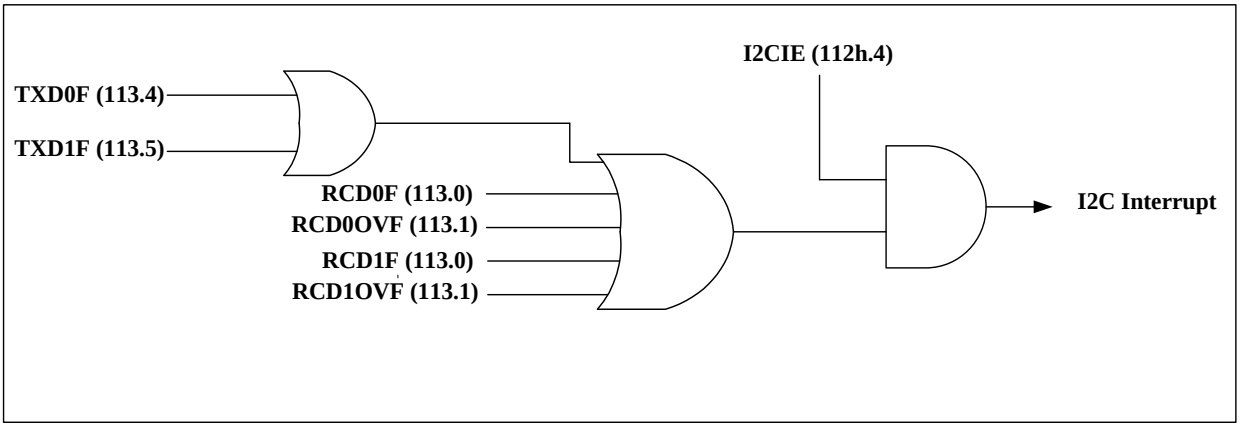


I2C Protocol

To use the slave I2C interface, the I2CEN (112h.3) bit has to be set. TM56F1542 supports 4 slave device IDs by setting I2CID (112.1~0). TM56F1542 can generate the transmitting flag TXD0F (113h.4) and TXD1F (113h.5) when data transmitting finished. It generates the receiving flag RCD0F (113h.0) and RCD1F (113h.2) when data receiving finished. It can also generates the receiving overflow flag RCD0OVF (113h.1) and RCD1OVF (113h.3) when data receiving finished but the receiving flag is not cleared. If one of those I2C flags is set, the I2C interrupt flag I2CIF (113h.6) will be generated. It generates I2C interrupt if the I2CIE (112h.4) bit is set. Refer to following table and figure.

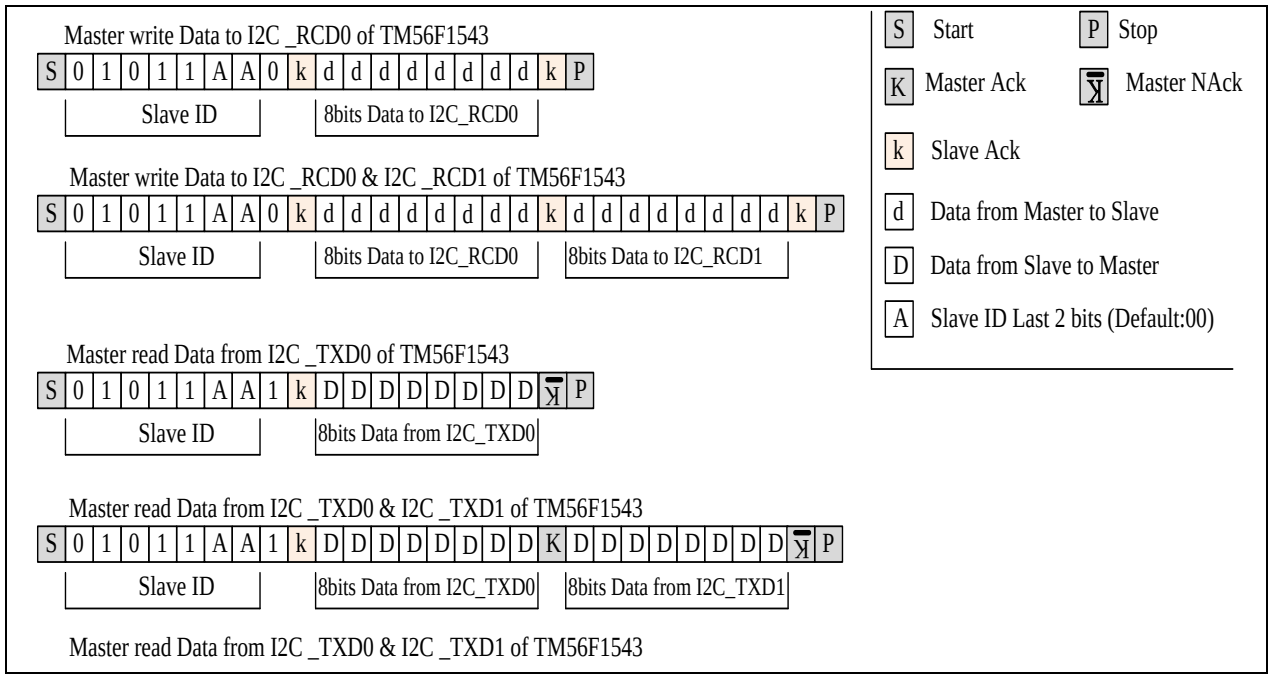


I2C Parallel Connection Application Circuit



Slave I2C Interrupt Block Diagram

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



TM56F1542 I2C Commands

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

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

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

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

112h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
I2CCTL	-	-	-	I2CIE	I2CEN	-	I2CID	
R/W	-	-	-	R/W	R/W	-	R/W	
Reset	-	-	-	0	0	-	0	0

112h.4 **I2CIE:** Slave I2C interrupt enable
0: disable
1: enable

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

112h.1~0 **I2CIE:** Slave I2C ID last 2 bits

113h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
I2CFLG	-	I2CIF	TXD1F	TXD0F	RCD1OVF	RCD1F	RCD1OVF	RCD0F
R/W	-	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	-	0	0	0	0	0	0	0

113h.6 **I2CIF:** I2C interrupt event pending flag, This bit is set by H/W while
a: I2CRCD0 or I2CRCD1 receive data finished
b: I2CRCD0 or I2CRCD1 data overflow occured
c: I2CTXD0 or I2CTXD1 data transmit finished

113h.5 **TXD1F:** Slave I2C transmitting data register 1 flag
This bit is set by H/W while I2CTXD1 data transmitting finished, write 0 to this bit will clear this flag

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

113h.3 **RCD1OVF:** Slave I2C receiving data register 1 overflow
This bit is set by H/W while receiving data to I2CRCD1 overflow, write 0 to this bit will clear this flag

113h.2 **RCD1F:** Slave I2C receiving data register 1 flag
This bit is set by H/W while data receiving to I2CRCD1 finished, write 0 to this bit will clear this flag

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

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

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

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

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

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

MEMORY MAP

Name	Address	R/W	Rst	Description
INDF (00h/80h/100h/180h)				**Function related to: RAM W/R**
INDF	00.7~0	R/W	-	Not a physical register, addressing INDF actually point to the register whose address is contained in the FSR register
TM0 (01h/101h)				**Function related to: Timer0**
TM0	01.7~0	R/W	0	Timer0 content
PCL (02h/82h/105h/182h)				**Function related to: PROGRAM COUNT**
PCL	02.7~0	R/W	0	Programming Counter LSB [7~0]
STATUS (03h/83h/103h/183h)				**Function related to: STATUS**
IRP	03.7	R/W	0	Register Bank Select bit (used for indirect addressing)
RP1	03.6	R/W	0	Register Bank Select bit 1 (used for direct addressing)
RP0	03.5	R/W	0	Register Bank Select bit 0 (used for direct addressing)
TO	03.4	R	0	WDT timeout flag, cleared by PWRST, 'SLEEP' or 'CLRWDT' instruction
PD	03.3	R	0	Power down flag, set by 'SLEEP', cleared by 'CLRWDT' instruction
Z	03.2	R/W	0	Zero flag
DC	03.1	R/W	0	Decimal Carry flag
C	03.0	R/W	0	Carry flag
FSR (04h/84h/104h/184h)				**Function related to: RAM W/R**
FSR	04.7~0	R/W	-	File Select Register, indirect address mode pointer
PAD (05h)				**Function related to: Port A**
PAD	05.7~0	R	-	Port A pin or "data register" state
		W	FF	Port A output data register
PBD (06h)				**Function related to: Port B**
PBD	06.4~0	R	-	Port B pin or "data register" state
		W	1F	Port B output data register
PDD (08h)				**Function related to: Port D**
PDD	08.6~0	R	-	Port D pin or "data register" state
		W	7F	Port D output data register
PCLATH (0Ah/8Ah/10Ah/18Ah)				**Function related to: PROGRAM COUNT**
PCLATH	0A.4~0	R/W	0	Write Buffer for the upper 5 bits of the Program Counter
INTIE (0Bh/8Bh/10Bh/18Bh)				**Function related to: Interrupt Enable**
ADCIE	0B.7	R/W	0	ADC interrupt enable, 1=enable, 0=disable
T2IE	0B.6	R/W	0	T2 interrupt enable, 1=enable, 0=disable
TM1IE	0B.5	R/W	0	Timer1 interrupt enable, 1=enable, 0=disable
TM0IE	0B.4	R/W	0	Timer0 interrupt enable, 1=enable, 0=disable
WKTIE	0B.3	R/W	0	Wakeup Timer interrupt enable, 1=enable, 0=disable Set 0 to clear & disable WKT timer
INT2IE	0B.2	R/W	0	INT2 pin (PA7) interrupt enable, 1=enable, 0=disable
INT1IE	0B.1	R/W	0	INT1 pin (PA4) interrupt enable, 1=enable, 0=disable
INT0IE	0B.0	R/W	0	INT0 pin (PA0) interrupt enable, 1=enable, 0=disable

Name	Address	R/W	Rst	Description
INTIF (0Ch)				**Function related to: Interrupt Flag**
ADCIF	0C.7	R	-	ADC interrupt flag, set by H/W after end of ADC conversion
		W	0	write 0: clear this flag; write 1: no action
T2IF	0C.6	R	-	T2 interrupt event pending flag, set by H/W while T2 overflows
		W	0	write 0: clear this flag; write 1: no action
TM1IF	0C.5	R	-	Timer1 interrupt event pending flag, set by H/W while Timer1 overflows
		W	0	write 0: clear this flag; write 1: no action
TM0IF	0C.4	R	-	Timer0 interrupt event pending flag, set by H/W while Timer0 overflows
		W	0	write 0: clear this flag; write 1: no action
WKTIF	0C.3	R	-	WKT interrupt event pending flag, set by H/W while WKT time out
		W	0	write 0: clear this flag; write 1: no action
INT2IF	0C.2	R	-	INT2 (PA7) interrupt event pending flag, set by H/W at INT2 pin's falling edge
		W	0	write 0: clear this flag; write 1: no action
INT1IF	0C.1	R	-	INT1 (PA4) interrupt event pending flag, set by H/W at INT1 pin's falling/rising edge
		W	0	write 0: clear this flag; write 1: no action
INT0IF	0C.0	R	-	INT0 (PA0) interrupt event pending flag, set by H/W at INT0 pin's falling/rising edge
		W	0	write 0: clear this flag; write 1: no action
INTE1 (0Eh)				**Function related to: Interrupt Enable Group 1**
LVDIE	0E.7	R/W	0	LVD interrupt enable, 0=disable , 1=enable
TKIE	0E.3	R/W	0	TK interrupt enable, 0=disable , 1=enable
I2CIE	0E.2	R/W	0	I2C interrupt enable, 0=disable , 1=enable
PXIE	0E.0	R/W	0	Pin change interrupt enable, 0=disable , 1=enable
CLKCTL (0Fh)				**Function related to: Fsys**
SLOWSTP	0F.4	R/W	0	Stop Slow-clock in Stop Mode 0: no Stop 1: Stop
FASTSTP	0F.3	R/W	1	Stop Fast-clock 0:no Stop 1:Stop
CPUCKS	0F.2	R/W	0	Select Fast-clock 0: Fsys=Slow-clock 1: Fsys=Fast-clock
CPUPSC	0F.1~0	R/W	11	Fsys Prescaler, 0: div 8, 1: div 4, 2: div 2, 3: div 1
TM0RLD (10h)				**Function related to: TM0**
TM0RLD	10.7~0	R/W	0	Timer0 reload Data

Name	Address	R/W	Rst	Description
TM0CTL (11h)				**Function related to: TM0**
TM0STP	11.6	R/W	0	Timer0 counter stop 0: Release 1: Stop counting
TM0EDG	11.5	R/W	0	Timer0 prescaler counting edge for TM0CKI pin 0: rising edge 1: falling edge
TM0CKS	11.4	R/W	0	Timer0 prescaler clock source 0: Fsys/2 1: TM0CKI pin (PA2 pin)
TM0PSC	11.3~0	R/W	0	Timer0 prescaler. Timer0 prescaler clock source divided by 0000: /1 0101: /32 1010: /1024 1111: /32768 0001: /2 0110: /64 1011: /2048 0010: /4 0111: /128 1100: /4096 0011: /8 1000: /256 1101: /8192 0100: /16 1001: /512 1110: /16384
TM1 (12h)				**Function related to: Timer1**
TM1	12.7~0	R/W	0	Timer1 content
TM1RLD (13h)				**Function related to: Timer1**
TM1RLD	13.7~0	R/W	0	Timer1 reload Data
TM1CTL (14h)				**Function related to: Timer1**
TM1STP	14.4	R/W	0	Timer1 counter stop 0: Release 1: Stop counting
TM1PSC	14.3~0	R/W	0	Timer1 prescaler. Timer1 clock source 0000: Fsys/2 0001: Fsys/4 0010: Fsys/8 0011: Fsys/16 0100: Fsys/32 0101: Fsys/64 0110: Fsys/128 0111: Fsys/256 1xxx: Fsys/512
T2CTL (15h)				**Function related to: T2**
T2CLR	15.3	R/W	1	T2 counter clear 0: Release 1: Stop counting
T2CKS	15.2	R/W	0	T2 clock source selection. 1: Fsys/128 0: Slow-clock
T2PSC	15.1~0	R/W	0	T2 prescaler. T2 clock source divided by - 00: 32768 01: 16384 10: 8192 11: 128
LVCTL (16h)				**Function related to: LVR/LVD**
LVDF	16.7	R	-	Low voltage detection flag, set by H/W while Vcc ≦ LVD
LVRSAV	16.5	R/W	1	LVR/LVD power save 1: LVR/LVD auto power off in STOP/IDLE mode 0: LVR/LVD enable in STOP/IDLE mode
LVDSAV	16.4	R/W	1	LVD auto power off in STOP/IDLE mode 0= LVD enable 1= LVD enable in slow/Fast mode; disable in STOP/IDLE mode
LVDS	16.3~0	R/W	01	LVD select; 0000: Disable, 0001: 2.19V, …. 1111: 4.15V
ADCDH (17h)				**Function related to: ADC**
ADCDH	17.7~0	R	-	ADC output data MSB, ADQ [11:4]

Name	Address	R/W	Rst	Description
ADCTL (18h)				**Function related to: ADC**
ADCDL	18.7~4	R	-	ADC output data LSB, ADQ [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: 000: Fsys/256 100: Fsys/16 001: Fsys/128 101: Fsys/8 010: Fsys/64 110: Fsys/4 011: Fsys/32 111: Fsys/2 (TC)
ADCTL2 (19h)				**Function related to: ADC**
BGEN	19.7	R/W	1	Band Gap BG1.25V enable 0: Disable 1: Enable and Auto disable in STOP/IDLE mode
ADCVREFS	19.6~5	R/W	0	ADC VREF select, 00: VCC, 01: 2.5V, 10: 3V, 11: 4V
ADCHS	19.4~0	R/W	0	ADC channel select 00000: ADC0 (PB0) 01010: ADC10(PD2) others: reserved 00001: ADC1 (PB1) 01011: ADC11(PD3) 00010: ADC2 (PB2) 01100: ADC12(PD4) 00011: ADC3 (PB3) 01101: ADC13(PD5) 01000: ADC8 (PD0) 01110: VBGO 01001: ADC9 (PD1) 10111: 1/4 VCC
INTIF1 (1Ah)				**Function related to: Interrupt Flag**
LVDIF	1A.7	R/W	0	LVD interrupt event pending flag, set by H/W while LV Detected write 0: clear this flag; write 1: no action
TKM2IF	1A.6	R/W	0	STK Module2 Interrupt event pending flag set by H/W at the STK module2 end of conversion S/W writes 0 to TKM2IF or sets the TKM2SOC bit to clear this flag.
TKM0IF	1A.4	R/W	0	STK Module0 Interrupt event pending flag set by H/W at the STK module0 end of conversion S/W writes 0 to TKM0IF or sets the TKM0SOC bit to clear this flag.
TKIF	1A.3	R/W	0	Touch Key interrupt event pending flag, set by H/W at the end of All Touch Key conversion S/W writes 0 to TKIF or sets the TKMxSOC bit to clear this flag.
I2CIF	1A.2	R/W	0	Slave I2C interrupt pending flag This bit is set by H/W while - I2CRCD0 or I2CRCD1 receive data finished - I2CRCD0 or I2CRCD1 data overflow occurred - I2CTXD0 or I2CTXD1 data transmit finished. Write 0 to clear this flag
PXIF	1A.0	R/W	0	Port A/B/D Pin Change Interrupt Flag F/W write 0 to clear this flag;

Name	Address	R/W	Rst	Description
IOCCTRL (1Bh)				**Function related to: IO Port**
LED	1B.7~4	R/W	0	IO current Control for LED usage enable LED[3]: PD port LED[1]: PB port LED[0]: PA port 1: enable, 0: disable
HSNK	1B.3~0	R/W	0	IO current Control for High Sink usage enable HSNK[3]: PD port HSNK[1]: PB port HSNK[0]: PA port 1: enable, 0: disable
PAWKE (1Ch)				**Function related to: Port A**
PAWKE	1C.7~0	R/W	0	PA pin Individual Wake up Enable 0: disable 1: enable
PBWKE (1Dh)				**Function related to: Port B**
PBWKE	1D.4~0	R/W	0	PB pin Individual Wake up Enable 0: disable 1: enable
PDWKE (1Fh)				**Function related to: Port D**
PDWKE	1F.7~0	R/W	0	PD pin Individual Wake up Enable 0: disable 1: enable
User Data Memory				
RAM	20~6F	R/W	-	RAM Bank0 area (80 Bytes)
RAM	70~7F	R/W	-	RAM common area (16 Bytes)

Name	Address	R/W	Rst	Description
OPTION (81h/181h)				**Function related to: STATUS/INT0/INT1/WDT/WKT**
HWAUTO	81.7	R/W	0	Enter interrupt vector, HW auto save/restore WREG and STATUS w/o TO, PD 0:disable 1: Enable
INT0EDG	81.6	R/W	0	INT0 pin edge interrupt event 0: falling edge to trigger 1: rising edge to trigger
INT1EDG	81.5	R/W	0	INT1 pin edge interrupt event 0: falling edge to trigger 1: rising edge to trigger
FREQL	81.4	R/W	1	SIRC Frequency selection 0: 60KHz 1: 50KHz
WDTPSC	81.3~2	R/W	11	WDT pre-scale selections: 00: 164mS 01: 328mS 10: 655mS 11: 1311mS
WKTPSC	81.1~0	R/W	11	WKT pre-scale selections: 00: 21mS 01: 41mS 10: 82mS 11: 164mS
PAMODH (85h)				**Function related to: Port A**
PA7MOD	85.7~6	R/W	01	PA7, PA4 I/O mode control 00: Mode0 01: Mode1 10: Mode2 11: Mode3
PA4MOD	85.1~0	R/W	01	
PAMODL (86h)				**Function related to: Port A**
PA3MOD	86.7~6	R/W	01	PA3~PA0 I/O mode control 00: Mode0 01: Mode1 10: Mode2 11: Mode3
PA2MOD	86.5~4	R/W	01	
PA1MOD	86.3~2	R/W	01	
PA0MOD	86.1~0	R/W	01	
PBMODH (87h)				**Function related to: Port B**
PB4MOD	87.1~0	R/W	01	PB4 I/O mode control 00: Mode0 01: Mode1 10: Mode2 11: Mode3
PBMODL (88h)				**Function related to: Port B**
PB3MOD	88.7~6	R/W	01	PB3~PB0 I/O mode control 00: Mode0 01: Mode1 10: Mode2 11: Mode3
PB2MOD	88.5~4	R/W	01	
PB1MOD	88.3~2	R/W	01	
PB0MOD	88.1~0	R/W	01	

Name	Address	R/W	Rst	Description
PDMODH (8Dh)				**Function related to: Port D**
PD6MOD	8D.5~4	R/W	01	PD6~PD4 I/O mode control 00: Mode0 01: Mode1 10: Mode2 11: Mode3
PD5MOD	8D.3~2	R/W	01	
PD4MOD	8D.1~0	R/W	01	
PDMODL (8Eh)				**Function related to: Port D**
PD3MOD	8E.7~6	R/W	01	PD3~PD0 I/O mode control 00: Mode0 01: Mode1 10: Mode2 11: Mode3
PD2MOD	8E.5~4	R/W	01	
PD1MOD	8E.3~2	R/W	01	
PD0MOD	8E.1~0	R/W	01	
OPTION2 (8Fh)				**Function related to: PWM0/PWM1**
TCOE	8F.7	R/W	0	TCOUT Output Enable 0: Disable 1: Enable, output to PB1
TM1OE	8F.6	R/W	0	Timer1 overflow toggle Output Enable 0: Disable 1:Enable, output to PB0
PWMCKS	8F.5~4	R/W	0	PWM0 Clock Source 0x: Fsys 10:FIRC 11:FIRC*2
INT1SEL	8F.1	R/W	0	INT1 Pin Select 0: PA4 1: PB4
INT0SEL	8F.0	R/W	0	INT0 Pin Select 0: PA0 1: PB0
PWMOE (90h)				**Function related to: PWM0/PWM1**
PWM0POE	90.7	R/W	0	PWM0P Output Enable to PA7 0: disable 1: enable
PWM0NOE	90.6	R/W	1	PWM0N Output Enable to PA0 0: disable 1: enable
PWM5OE	90.5	R/W	1	PWM5 Output Enable to PA7 0: disable 1: enable (PWM0POE has higher priority)
PWM4OE	90.4	R/W	1	PWM4 Output Enable to PA4 0: disable 1: enable
PWM3OE	90.3	R/W	1	PWM3 Output Enable to PA3 0: disable 1: enable
PWM2OE	90.2	R/W	0	PWM2 Output Enable to PA2 0: disable 1: enable
PWM1OE	90.1	R/W	0	PWM1 Output Enable to PA1 0: disable 1: enable

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

Name	Address	R/W	Rst	Description
SYSREG (105h)			**Function related to: TEST**	
SYSREG	105.7~2	R/W	0	Test bits, keep 00
ISAVB	105.1~0	R/W	11	Power saving mode for Reading operation of ROM. 00: Enable 01/10: Reserved 11: Disable
ChgRdMode **(106h)**			**Function related to: TEST**	
ChgRdMode	106.7~0	W	0	Test bits, Keep 00
LVRPD (107h)			**Function related to: LVR**	
LVRPD	107	W	0	LVR power down control Write 0x37 to force LVR+POR disable Write 0x38 to force LVR disable, POR still enable Write 0x39 to force POR disable, LVR still enable Write other value to enable LVR/POR
PORPDF	107.1	R	0	1: when Reg. 107h write 0x37 or 0x39; 0: when Reg. 107h write other value
LVRPDF	107.0	R	0	1: when Reg. 107h write 0x37 or 0x38; 0: when Reg. 107h write other value
PCH (10Ch)			**Function related to: PC**	
PCH	10C.4~0	R	0	Program Counter [12:8]
		W	0	Write 1Ch to this register, then it's not necessary to write PCLATCH for table read lookup
BGTRIM (10Eh)			**Function related to: VBG**	
BGTRIM	10E.3~0	R/W		VBG voltage adjustment 00h: Lowest voltage Fh: Highest voltage
I2CTXD0 (110h)			**Function related to: Slave I2C**	
I2CTXD0	110.7~0	R/W	0	The transmitting register0 of slave I2C
I2CTXD1 (111h)			**Function related to: Slave I2C**	
I2CTXD1	111.7~0	R/W	0	The transmitting register1 of slave I2C
I2CCTL (112h)			**Function related to: Slave I2C**	
I2CEN	112.3	R/W	0	Slave I2C interface enable 0: Disable 1: Enable
I2CID	112.1~0	R/W	0	Slave I2C ID last 2 bits

Name	Address	R/W	Rst	Description
I2CFLG (113h)			**Function related to: Slave I2C**	
TXD1F	113.5	R/W	0	Slave I2C transmitting data register 1 flag, set by H/W while I2CTXD1 data transmitting finished Write 0 to clear this flag
TXD0F	113.4	R/W	0	Slave I2C transmitting data register 0 flag, set by H/W while I2CTXD0 data transmitting finished Write 0 to clear this flag
RCD1OVF	113.3	R/W	0	Slave I2C receiving data register 1 overflow, set by H/W while receiving data to I2CRCD1 overflow. Write 0 to clear this flag
RCD1F	113.2	R/W	0	Slave I2C receiving data register 1 flag, set by H/W while I2CRCD1 data receiving finished. Write 0 to clear this flag
RCD0OVF	113.1	R/W	0	Slave I2C receiving data register 0 overflows, set by H/W while receiving data to I2CRCD0 overflow. Write 0 to clear this flag
RCD0F	113.0	R/W	0	Slave I2C receiving data register 0 flag, set by H/W while I2CRCD0 data receiving finished Write 0 to clear this flag
I2CRCD0 (114h)			**Function related to: Slave I2C**	
I2CRCD0	114.7~0	R	0	The receiving register0 of slave I2C
I2CRCD1 (115h)			**Function related to: Slave I2C**	
I2CRCD1	115.7~0	R	0	The receiving register1 of slave I2C
TKM0DL (116h)			**Function related to: Touch Key Module0**	
TKM0DL	116.7~0	R	-	STK Module0 Data LSB [7~0]
TKM0DH (117h)			**Function related to: Touch Key Module0**	
TKM0DH	117.5~0	R	-	STK Module0 Data MSB [13~8]
TKM2DL (11Ah)			**Function related to: Touch Key Module2**	
TKM2DL	11A.7~0	R	-	STK Module2 Data LSB [7~0]
TKM2DH (11Bh)			**Function related to: Touch Key Module2**	
TKM2DH	11B.5~0	R	-	STK Module2 Data MSB [13~8]
TKMCON0 (11Ch)			**Function related to: Touch Key Module**	
TKM0TCP	11C.4~2	R/W	1	TK module0 touch key clock frequency select; 000:slowest, …, 111:fastest
TKM0FSL	11C.1~0	R/W	0	TK module0 clock(RCK0/TCK0) frequency selection; 00: slowest, …, 11: fastest

Name	Address	R/W	Rst	Description
TKMCON1 (11Dh)				**Function related to: Touch Key Module**
TKM2TCP	11D.7~5	R/W	1	TK module2 touch key clock frequency select; 000:slowest, …, 111:fastest
TKM2FSL	11D.3~2	R/W	0	TK module2 clock(RCK2/TCK2) frequency selection; 00: slowest, …, 11: fastest
TKMCTL0 (11Eh)				**Function related to: Touch Key Module**
TKM2SOC	11E.6	R/W	0	STK Module2 Start of Conversion, HW clear it while end of conversion
TKM0SOC	11E.4	R/W	0	STK Module0 Start of Conversion, HW clear it while end of conversion
TKM2EOC	11E.2	R	-	STK Module2 End of Conversion
TKM0EOC	11E.0	R	-	STK Module0 End of Conversion
TKMCTL1 (11Fh)				**Function related to: Touch Key Module**
TKMHSEN	11F.7	R/W	0	All STK Module Sensitivity 0: lower sensitivity 1: higher sensitivity
TKM2PD	11F.6	R/W	1	STK Module2 Power Down 0: Touch Key disable 1: Touch Key enable
TKM0PD	11F.4	R/W	1	STK Module0 Power Down 0: Touch Key disable 1: Touch Key enable
TKM2JMP	11F.2	R/W	0	STK Module2 touch key clock mode 0: fixed frequency 1: auto-change frequency
TKM0JMP	11F.0	R/W	0	STK Module0 touch key clock mode 0: fixed frequency 1: auto-change frequency
User Data Memory				
RAM	120~16F	R/W	-	RAM Bank2 area (80 Bytes)

Name	Address	R/W	Rst	Description
DPL (185h)			**Function related to: Table Read**	
DPL	185.7~0	R/W	0	Table read low address, data ROM pointer (DPTR) low byte
DPH (186h)			**Function related to: Table Read**	
DPH	186.4~0	R/W	0	Table read high address, data ROM pointer (DPTR) high byte
TABR (18Ch)			**Function related to: Table Read**	
TABR	18C.7~0	R/W	0	1: TABR write 1 = opcode TABRL 2: TABR write 2 = opcode TABRH 3: after step1 or step2, read TABR to get main ROM table read value after step1, read TABR to get EEPROM value (when EEPEN=E2h) Table Read for ASM: TABRL/TABRH or TABR Table Read for C: TABR
CRCDL (18Eh)			**Function related to: CRC16**	
CRCDL	18E.7~0	R/W	0	CRC16 Data 7~0
CRCDH(18Fh)			**Function related to: CRC16**	
CRCDH	18F.7~0	R/W	0	CRC16 Data 15~8
CRCIN(190h)			**Function related to: CRC16**	
CRCIN	190.7~0	W	0	CRC input data
EEPCTL (191h)			**Function related to: EEPROM**	
EEPTO	191.7	R	0	EEPROM Write Time-out flag
PWRDEC	191.5	R/W	0	YMC IP 105 ℃ Detect enable (used for Test); 0: disable, 1: enable
HVS	191.4	R/W	1	YMC IP write High Voltage (used for Test); 0: disable, 1: enable (gate with PRG_CTRL[2], PRG_CTRL[2] default=1)
EEPTE	191.1~0	R/W	11	EEPROM Write Time-out enable; 00: Disable; 01:2.5ms; 10:10ms; 11:20.5ms
EEPEN (192h)			**Function related to: EEPROM**	
EEPEN	192.7~0	W	0	EEPROM Enable 0xE2: Enable Others: Disable
EEPDT (193h)			**Function related to: EEPROM**	
EEPDT	193.7~0	W	0	EEPROM Data to write in
TKM0TMRL (194h)			**Function related to: Touch Key Module0**	
TKM0TMRL	194.7~0	R/W	FF	STK Module0 reference counter LSB [7~0]
TKM0TMRH (195h)			**Function related to: Touch Key Module0**	
TKM0TMRH	195.5~0	R/W	0	STK Module0 reference counter MSB [13~8]
TKM2TMRL (198h)			**Function related to: Touch Key Module2**	
TKM2TMRL	198.7~0	R/W	FF	STK Module2 reference counter LSB [7~0]
TKM2TMRH (199h)			**Function related to: Touch Key Module2**	
TKM2TMRH	199.5~0	R/W	0	STK Module2 reference counter MSB [11~8]
TKM0REFC (19Ah)			**Function related to: Touch Key Module0**	
TKM0REFC	19A.7~0	R/W	-	STK Module0 Reference clock capacitor select
TKM2REFC (19Ch)			**Function related to: Touch Key Module2**	
TKM2REFC	19C.7~0	R/W	-	STK Module2 Reference clock capacitor select

Name	Address	R/W	Rst	Description
TKMCHS0 (19Dh)				**Function related to: Touch Key Module0**
TKM0CHS	19D.2~0	R/W	0	STK Module0 Channel Select: 0:TK0 1:TK1 2:TK2 3:TK3 4:TK4
TKMCHS1 (19Eh)				**Function related to: Touch Key Module2**
TSTREG2	19E.7~6	R/W	0	Test bits, Keep 0
TKM2CHS	19E.2~0	R/W	0	STK Module2 Channel Select: 0:TK16 1:TK17 2:TK18 3:TK19 4:TK20 5:TK21 6:TK22 7:TK23(REF)
CTRLRFK (19Fh)				**Function related to: Touch Key Module2**
CTRLRFK	19F.7~0	R/W	0	STK module2 Reference Key(TK23) Capacitor Control 0 : minimum capacitor on TK23 FF: maximum capacitor on TK23
User Data Memory				
RAM	1A0~1EF	R/W	-	RAM Bank3 area (80 Bytes)

INSTRUCTION SET

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

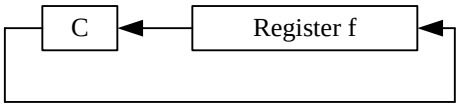
MOVWX	Move W to "f"	
Syntax	MOVWX f	
Operands	f : 00h ~ 1FFh	
Operation	(f) ← (W)	
Status Affected	-	
OP-Code	ff00 0000 1fff ffff	
Description	Move data from W register to register 'f'.	
Cycle	1	
Example	MOVWX REG1	B : REG1 =0xFF, W =0x4F A : REG1 =0x4F, W =0x4F

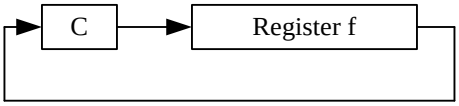
NOP	No Operation
Syntax	NOP
Operands	-
Operation	No Operation
Status Affected	-
OP-Code	0000 0000 0000 0000
Description	No Operation
Cycle	1
Example	NOP -

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

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

RETLW	Return with Literal in W
Syntax	RETLW k
Operands	k : 00h ~ FFh
Operation	PC ← TOS, (W) ← k
Status Affected	-
OP-Code	0001 1000 kkkk kkkk
Description	The W register is loaded with the eight-bit literal 'k'. The program counter is loaded from the top of the stack (the return address). This is a two-cycle instruction.
Cycle	2
Example	LCALL TABLE 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 : 00h ~ 1FFh, d : 0, 1
Operation	
Status Affected	C
OP-Code	ff00 1101 dfff ffff
Description	The contents of register 'f' are rotated one bit to the left through the Carry Flag. If 'd' is 0, the result is placed in the W register. If 'd' is 1, the result is stored back in register 'f'.
Cycle	1
Example	RLX REG1, 0 B : REG1 =1110 0110, C =0 A : REG1 =1110 0110 W =1100 1100, C =1

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

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

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

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

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

TABRH	**Return DPTR high byte to W**
Syntax	TABRH
Operands	-
Operation	(W) ← ROM[DPTR] high byte content, Where DPTR = {DPH[max:8], FSR[7:0]}
Status Affected	-
OP-Code	0000 0000 0101 1000
Description	The W register is loaded with high byte of ROM[DPTR]. This is a two-cycle instruction.
Cycle	2

Example

```
MOVLW   (TAB1&0xFF)
MOVWX   DPL                 ;Where DPL is register
MOVLW   (TBA1>>8)&0xFF
MOVWX   DPH                 ;Where DPH is register

TABRL                       ;W =0x89
TABRH                       ;W =0x37

        ORG 0234h
TAB1:
DT      0x3789, 0x2277      ;ROM data 16 bits
```

TABRL	**Return DPTR low byte to W**
Syntax	TABRL
Operands	-
Operation	(W) ← ROM[DPTR] low byte content, Where DPTR = {DPH[max:8], FSR[7:0]}
Status Affected	-
OP-Code	0000 0000 0101 0000
Description	The W register is loaded with low byte of ROM[DPTR]. This is a two-cycle instruction.
Cycle	2

Example

```
MOVLW   (TAB1&0xFF)
MOVWX   DPL                 ;Where DPL register
MOVLW   (TBA1>>8)&0xFF
MOVWX   DPH                 ;Where DPH register

TABRL                       ;W =0x89
TABRH                       ;W =0x37

        ORG 0234h
TAB1:
DT      0x3789, 0x2277      ;ROM data 16 bits
```

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

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

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

ELECTRICAL CHARACTERISTICS

1. Absolute Maximum Ratings (T_A =25℃)

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

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

Parameter	Sym	Conditions		Min	Typ	Max	Unit
Operating Voltage	V_{cc}	Fsys=18.432Mhz		2.5		5.5	
		Fsys= 9.2Mhz		LVR		5.5	
Input High Voltage	V_{IH}	All Input	V_{CC} = 3~5V	0.6Vcc	–	Vcc	V
Input Low Voltage	V_{IL}	All Input	V_{CC} = 3~5V	Vss	–	0.2Vcc	V
Output High Current	I_{OH}	All Output	V_{CC} = 5V,V_{OH} = 4.5V LED[x]=0	10	12	–	mA
			V_{CC} = 3V,V_{OH} = 2.7V LED[x]=0	4	5	–	
Output Low Current	I_{OL}	All Output (HSNK[x]=1)	V_{CC} = 5V, V_{OL} = 0.5V	60	74	–	mA
			V_{CC} = 3V, V_{OL} = 0.3V	28	34	–	
		All Output (HSNK[x]=0)	V_{CC} = 5V, V_{OL} = 0.5V	33	42	–	
			V_{CC} = 3V, V_{OL} = 0.3V	15	19	–	
Input Leakage Current (pin high)	I_{ILH}	All Input	V_{IN} = V_{CC}	–	–	1	uA
Input Leakage Current (pin low)	I_{ILL}	All Input	V_{IN} = 0V	–	–	–1	uA

Parameter	Sym	Conditions			Min	Typ	Max	Unit
Power Supply Current (No Load)	I_{CC}	FAST mode FIRC 18.4 MHz		V_{CC} = 5V	–	6.7	–	mA
				V_{CC} = 3V	–	4	–	
		FAST mode FIRC 9.2 MHz		V_{CC} = 5V	–	5.1	–	
				V_{CC} = 3V	–	3.2	–	
		FAST mode FIRC 4.6 MHz		V_{CC} = 5V	–	4.2	–	
				V_{CC} = 3V	–	2.8	–	
		FAST mode FIRC 2.3 MHz		V_{CC} = 5V	–	3.8	–	
				V_{CC} = 3V	–	2.5	–	
		SLOW mode SIRC 50KHz FIRC STOP	ISAVB = 11 (default)	V_{CC} = 5.0V	–	2.2	–	mA
				V_{CC} = 3.0V	–	1.6	–	
			ISAVB = 00	V_{CC} = 5.0V	–	1.3	–	mA
				V_{CC} = 3.0V	–	0.9	–	
		IDLE mode SIRC 50 KHz		V_{CC} = 5.0V	–	6.8	–	uA
				V_{CC} = 3.0V	–	2.2	–	
		STOP mode LVR/POR Off		V_{CC} = 5.0V	–	–	1	uA
				V_{CC} = 3.0V	–	–	1	
Pull-up Resistor	R_{UP}	V_{IN} = 0 V Ports A/B/D		V_{CC} = 5.0V	–	32	–	KΩ
				V_{CC} = 3.0V	–	56	–	

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

Parameter	Condition	Min	Typ	Max	Unit
FIRC Frequency (*)	25℃, V_{CC} =5.0 V	-1%	18.432	+1%	MHz
	-40℃ ~ 105℃, V_{CC} = 5.0 V	-2.5%	18.432	+2.5%	
	-40℃ ~ 105℃, V_{CC} = 3.0 ~ 5.0 V	-4.0%	18.432	+2.5%	

(*) FIRC frequency can be divided by 1/2/4/16.

4. Reset Timing Characteristics (T_A = -40°C to +85°C)

Parameter	Conditions	Min	Typ	Max	Unit
RESET Input Low width	Input V_{CC} = 5 V ±10 %	–	30	–	μs
WDT time	V_{CC} = 3 V, WDTPSC = 11	–	1420	–	ms
	V_{CC} = 5 V, WDTPSC = 11		1280		
WKT time	V_{CC} = 3 V, WKTPSC = 11	–	160	–	ms
	V_{CC} = 5 V, WKTPSC = 11		180		
CPU start up time	V_{CC} = 5 V	–	38	–	ms

5. EEPROM Characteristics

Parameter	Conditions	Min	Typ	Max	Units
Write Voltage, Vcc	-20℃ ~ 105℃	3.0	5	5.5	V
	-40℃ ~ 105℃	3.5	5	5.5	V
Write Endurance(1)	Vcc=3.0 ~ 5.5V, -40 ~ 105℃	30K			cycles
	Vcc=3.0 ~ 5.5V, -20 ~ 105℃	40K			cycles
	Vcc=3.0 ~ 5.5V, -10 ~ 105℃	50K			cycles

Note:1 The value of this parameter is based on the characteristics of tested samples

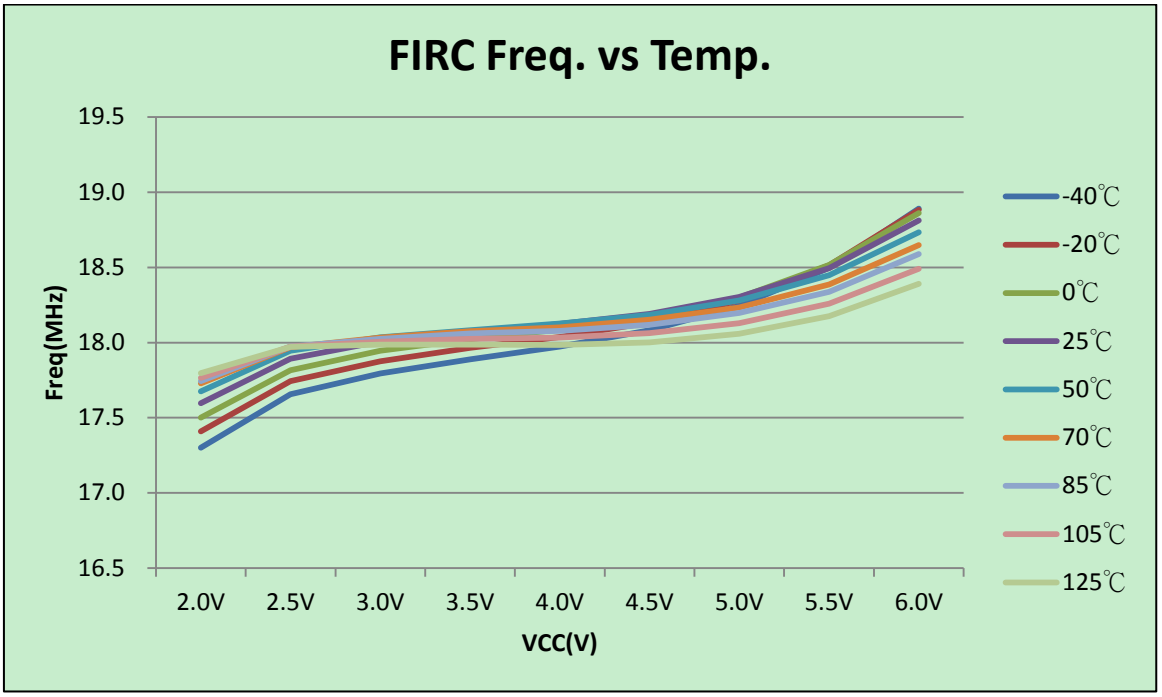
6. LVR Circuit Characteristics (T_A = 25℃)

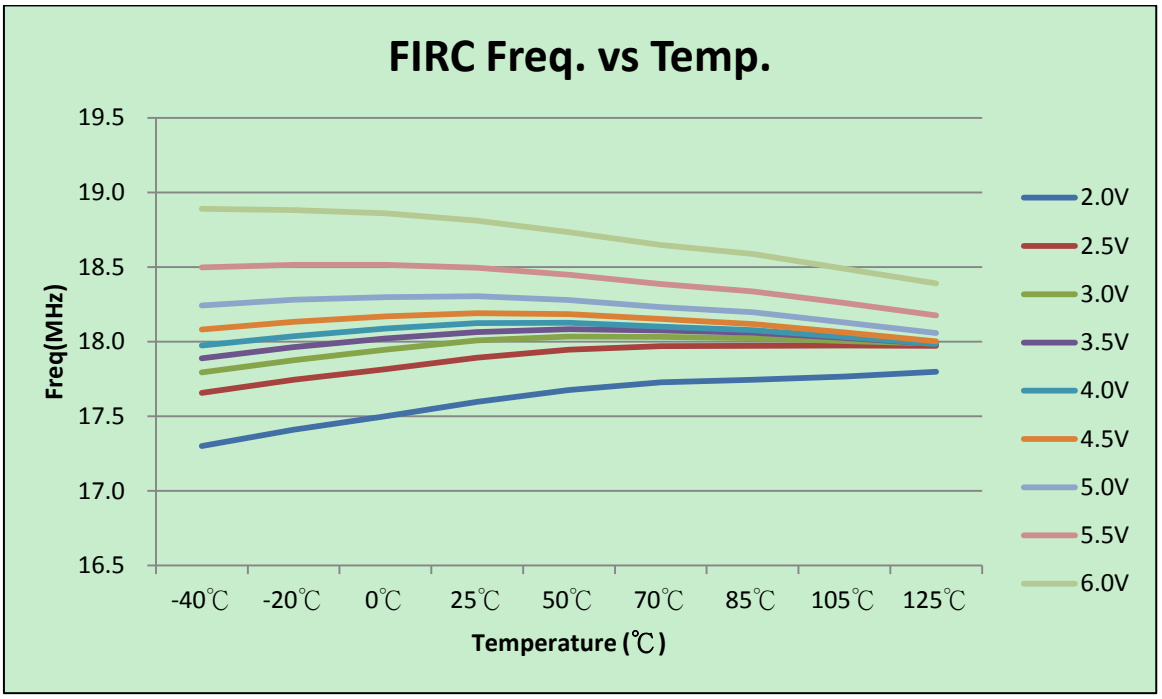
Parameter	Symbol	Conditions	Min	Typ	Max	Unit
LVR Voltage	LVR_{th}	T_A = 25℃	–	4.15	–	V
			–	4.01	–	
			–	3.87	–	
			–	3.73	–	
			–	3.59	–	
			–	3.45	–	
			–	3.31	–	
			–	3.17	–	
			–	3.03	–	
			–	2.89	–	
			–	2.75	–	
			–	2.62	–	
			–	2.47	–	
			–	2.33	–	
			–	2.19	–	
			–	2.05	–	
LVD Voltage	LVD_{th}	T_A = 25℃	–	4.15	–	V
			–	4.01	–	
			–	3.87	–	
			–	3.73	–	
			–	3.59	–	
			–	3.45	–	
			–	3.31	–	
			–	3.17	–	
			–	3.03	–	
			–	2.89	–	
			–	2.75	–	
			–	2.62	–	
			–	2.47	–	
			–	2.33	–	
			–	2.19	–	
			–	2.05	–	
LVR Hysteresis Voltage	V_{HYST}	T_A = 25℃	–	±0.1	–	V
Low Voltage Detection time	t_{LVR}	T_A = 25℃	100	–	–	μs

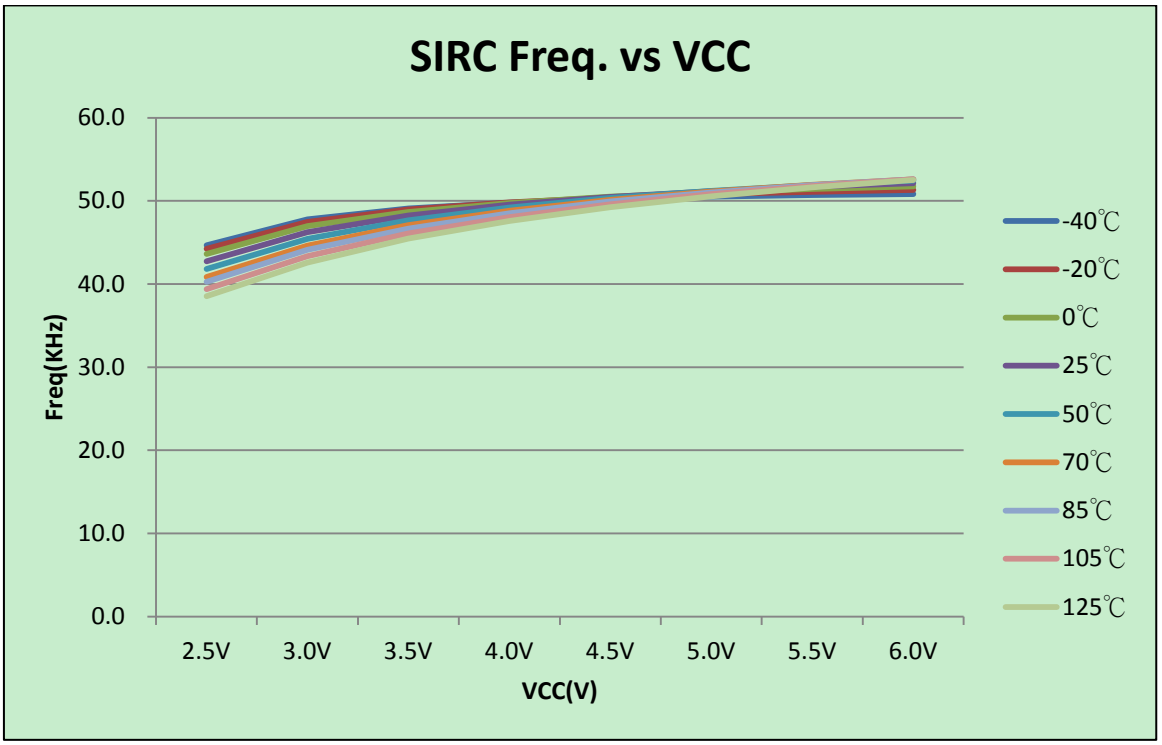
7. ADC Electrical Characteristics (T_A = 25℃, VCC = 3.0V to 5.5V, VSS = 0V)

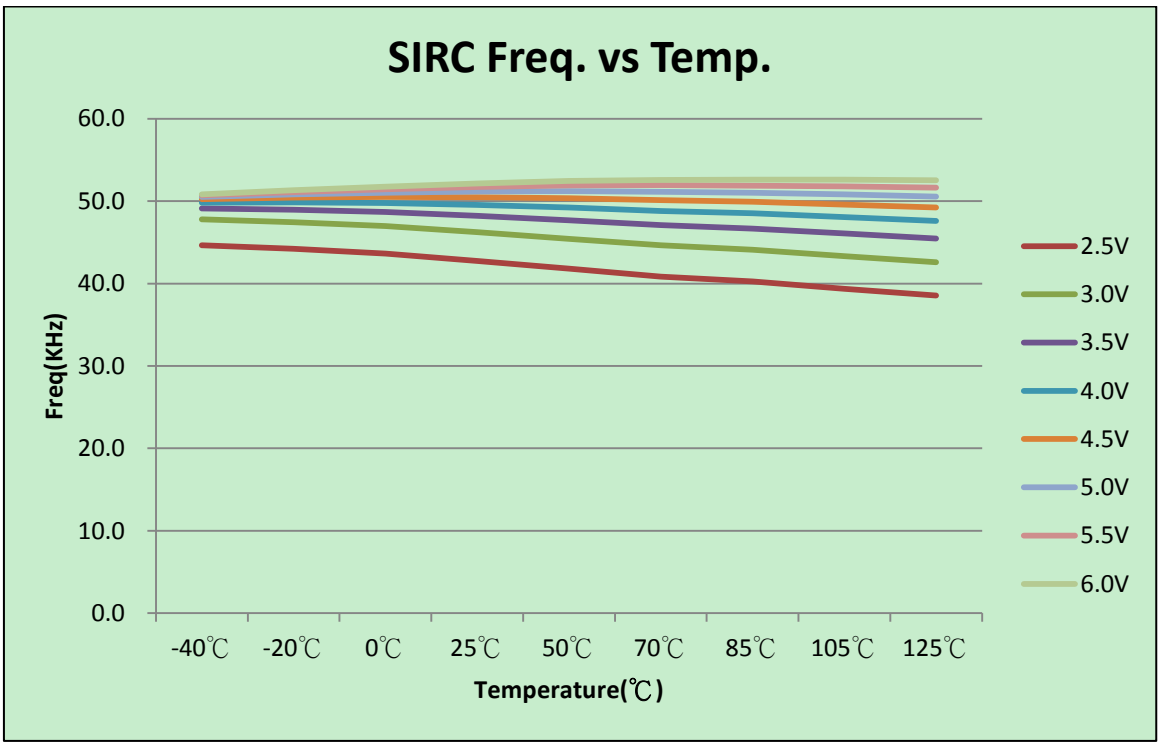
Parameter	Conditions	Min	Typ	Max	Units
Total Accuracy	V_{CC} = 5V, V_{SS} = 0V, f_{ADC} = 1 MHz	–	±3	±13	LSB
Integral Non-Linearity		–	±3.2	±15	
Differential Non-linearity		–	±1	±4	
Max Input Clock freq. (f_{ADC})	Source impedance (Rs<10K omh)	–	–	2	MHz
	Source impedance (Rs<20K omh)	–	–	1	
	Source impedance (Rs<50K omh)	–	–	0.5	
	Source is VBG (ADCHS=1100)	–	–	2	
Conversion Time	f_{ADC} = 1 MHz	–	50	–	μs
BandGap Voltage Reference (VBG)	-20℃~105℃, VCC = 3.0~5.0V	-2%	1.2	+1.5%	V
ADC reference voltage(V_{ref}) (ADCVREFS=1)	-20℃~105℃, VCC = 3.0~5.0V	-2.5%	2.5	+2%	
Vcc/4 reference voltage	25℃, VCC = 3.0~5.0V	-1%	0.25*Vcc	1%	
Input Voltage	–	V_{SS}	–	Vcc	

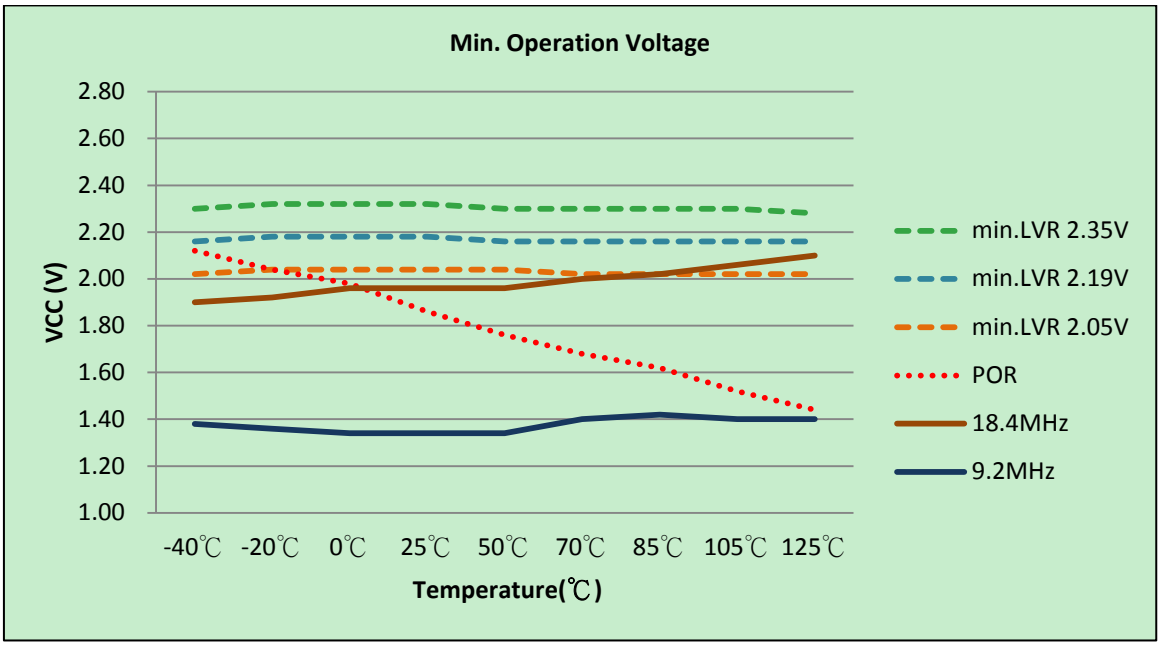
8. Characteristic Graphs

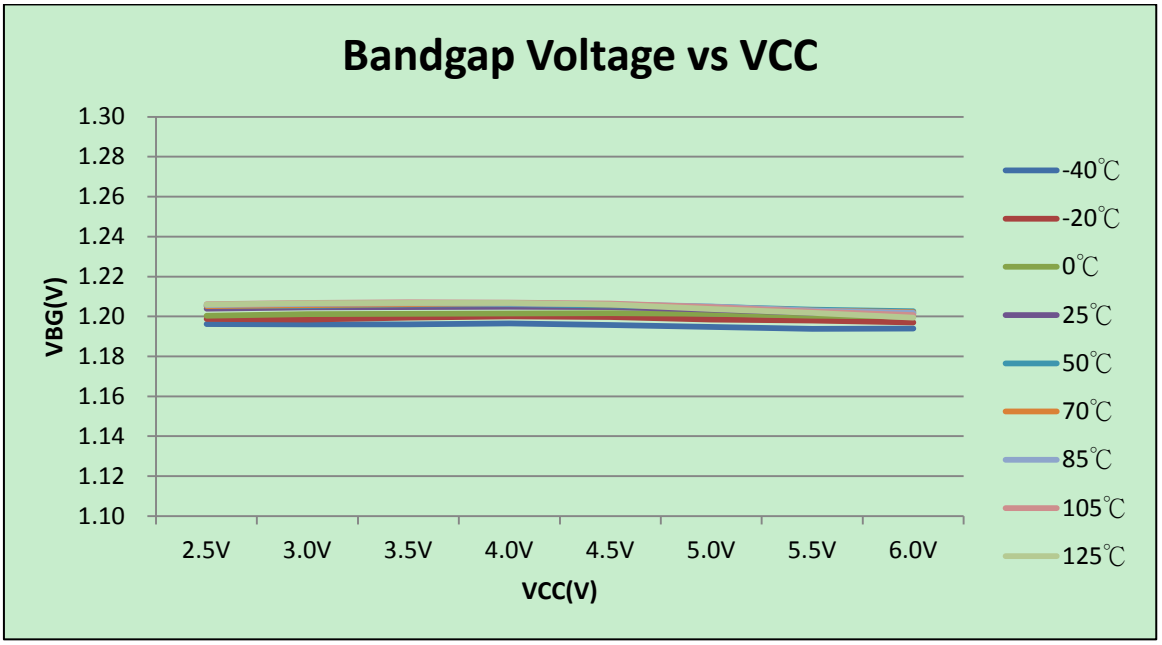





SIRC Freq. vs VCC
60.0
50.0
40.0
30.0
20.0
10.0
0.0
Freq(KHz)
2.5V
3.0V
3.5V
4.0V
4.5V
5.0V
5.5V
6.0V
VCC(V)
-40℃
-20℃
0℃
25℃
50℃
70℃
85℃
105℃
125℃


SIRC Freq. vs Temp.
60.0
50.0
40.0
30.0
20.0
10.0
0.0
Freq(KHz)
-40℃
-20℃
0℃
25℃
50℃
70℃
85℃
105℃
125℃
Temperature(℃)
2.5V
3.0V
3.5V
4.0V
4.5V
5.0V
5.5V
6.0V


Min. Operation Voltage
2.80
2.60
2.40
2.20
2.00
1.80
1.60
1.40
1.20
1.00
VCC (V)
-40℃ -20℃ 0℃ 25℃ 50℃ 70℃ 85℃ 105℃ 125℃
Temperature(℃)
min.LVR 2.35V
min.LVR 2.19V
min.LVR 2.05V
POR
18.4MHz
9.2MHz


Bandgap Voltage vs VCC
1.30
1.28
1.26
1.24
1.22
1.20
1.18
1.16
1.14
1.12
1.10
VBG(V)
2.5V 3.0V 3.5V 4.0V 4.5V 5.0V 5.5V 6.0V
VCC(V)
-40℃
-20℃
0℃
25℃
50℃
70℃
85℃
105℃
125℃

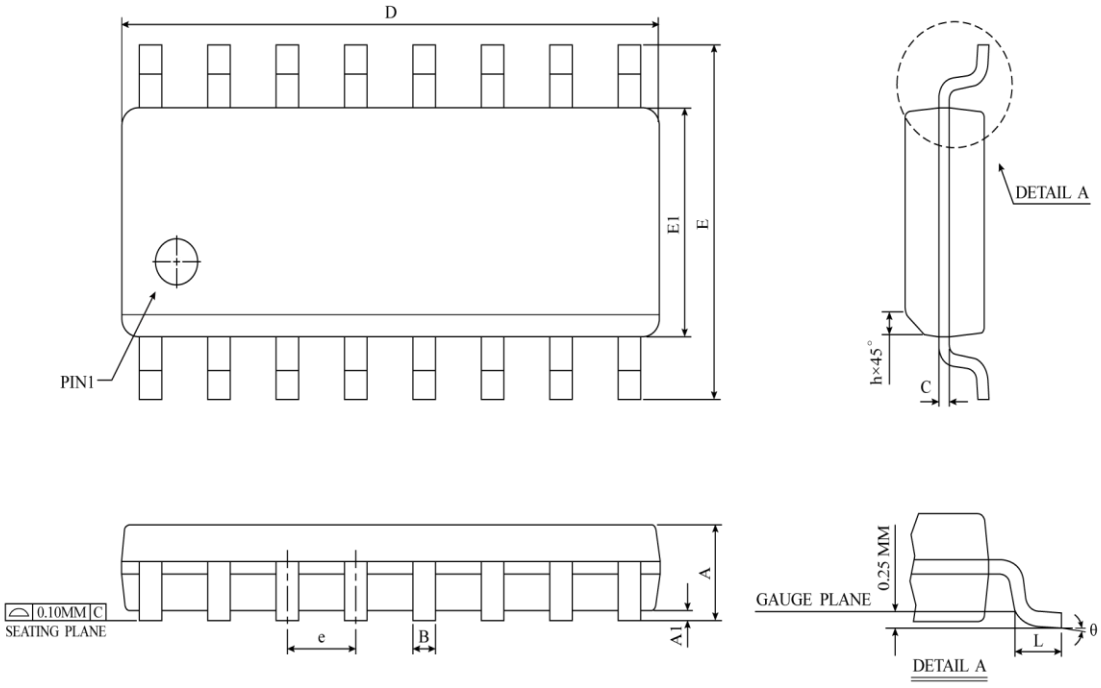
PACKAGING INFORMATION

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
TM56F1542-MTP	Wafer/Dice blank chip
TM56F1542-COD	Wafer/Dice with code
TM56F1542-MTP-21	SOP 20-pin (300 mil)
TM56F1542-MTP-16	SOP 16-pin (150 mil)

- SOP-16 (150mil) Package Dimension

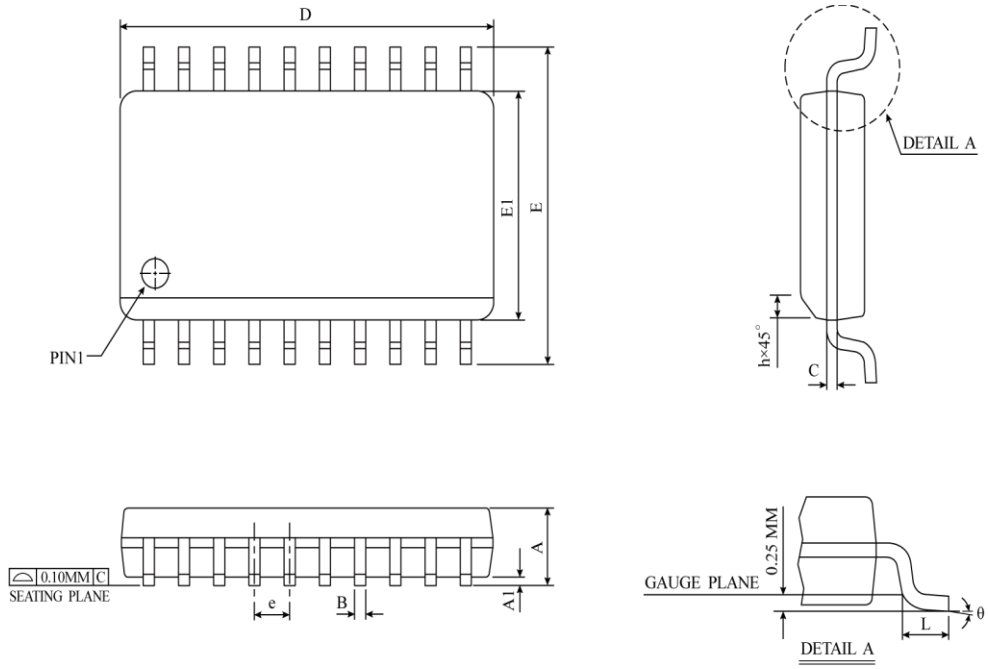


SYMBOL	DIMENSION IN MM			DIMENSION IN INCH		
	MIN	NOM	MAX	MIN	NOM	MAX
A	1.35	1.55	1.75	0.0532	0.0610	0.0688
A1	0.10	0.18	0.25	0.0040	0.0069	0.0098
B	0.33	0.42	0.51	0.0130	0.0165	0.0200
C	0.19	0.22	0.25	0.0075	0.0087	0.0098
D	9.80	9.90	10.00	0.3859	0.3898	0.3937
E	5.80	6.00	6.20	0.2284	0.2362	0.2440
E1	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-20 (300mil) Package Dimension



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

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

-