



十速

TM57ME16/18

DATA SHEET

Rev 0.96

tenx reserves the right to change or discontinue the manual and online documentation to this product herein to improve reliability, function or design without further notice. **Tenx** does not assume any liability arising out of the application or use of any product or circuit described herein; neither does it convey any license under its patent rights nor the rights of others. **Tenx** products are not designed, intended, or authorized for use in life support appliances, devices, or systems. If Buyer purchases or uses tenx products for any such unintended or unauthorized application, Buyer shall indemnify and hold tenx and its officers, employees, subsidiaries, affiliates and distributors harmless against all claims, cost, damages, and expenses, and reasonable attorney fees arising out of, directly or indirectly, any claim of personal injury or death associated with such unintended or unauthorized use even if such claim alleges that tenx was negligent regarding the design or manufacture of the part.

AMENDMENT HISTORY

Version	Date	Description
V0.90	Jun, 2013	New release
V0.91	Aug, 2013	<ol style="list-style-type: none"> 1. Modify EV board 2. Modify Pin Assignment diagram 3. Modify Pin Summary 4. Modify program code
V0.92	Oct, 2014	Add packaging information
V0.93	Feb, 2015	Change the operating voltage
V0.94	May, 2016	Remark: DS- TM57ME16_E change Doc No. to DS- TM57ME16/18_E <ol style="list-style-type: none"> 1. SIRC frequency select 2. Add device TM57ME18
V0.95	Nov, 2017	Add Characteristic Graphs of Fsys vs. LVR
V0.96	May, 2018	Add package TSSOP-8

CONTENTS

AMENDMENT HISTORY	2
FEATURES	5
BLOCK DIAGRAM	7
PIN ASSIGNMENT	8
PIN DESCRIPTION	8
PIN SUMMARY.....	9
FUNCTIONAL DESCRIPTION	10
1. CPU Core	10
1.1 Clock Scheme and Instruction Cycle	10
1.2 RAM Addressing Mode	11
1.3 Programming Counter (PC) and Stack.....	12
1.4 ALU and Working (W) Register.....	14
1.5 STATUS Register (F-Plane 03H)	15
1.6 Interrupt.....	16
2. Chip Operation Mode	19
2.1 Reset.....	19
2.2 System Configuration Register (SYSCFG)	20
2.3 MTP ROM	21
2.4 Power-Down Mode	21
2.5 Dual System Clock.....	22
2.6 Dual System Clock Modes Transition	24
3. Peripheral Functional Block	28
3.1 Watchdog (WDT) / Wakeup (WKT) Timer.....	28
3.2 Timer0: 8-bit Timer / Counter with Pre-scale (PSC)	31
3.3 T2: 15-bit Timer.....	36
3.4 PWM0: 8-bit PWM.....	39
3.5 System Clock Oscillator.....	43
3.6 FIRC and SIRC Clock Frequency Selection	44
4. I/O Port.....	45
4.1 PA0-2	45
4.2 PA3-4	47
4.3 PA7.....	49
MEMORY MAP.....	51
F-Plane	51
R-Plane	53
INSTRUCTION SET	56

ELECTRICAL CHARACTERISTICS	69
1. Absolute Maximum Ratings.....	69
2. DC Characteristics	70
3. Clock Timing	71
4. Reset Timing Characteristics	71
5. Characteristic Graphs.....	72
PACKAGING INFORMATION	75

FEATURES

1. **ROM: 1K x 14 bits MTP**
2. **RAM: 48 x 8 bits**
3. **STACK: 5 Levels**
4. **I/O Ports: Two bit-programmable I/O ports (Max. 6 pins)**
5. **Two Independent Timers**
 - Timer0
 - 8-bit timer0 with divided by 1 ~ 256 pre-scale option / counter / interrupt / stop function
 - T2
 - 15-bit T2 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) or Fsys/128
6. **One 8-bit PWM with pre-scale / period-adjustment / buffer-reload / interrupt / clear and hold function**
7. **Min. Operating Voltage (power on) and Speed: VDD can be lowest to 1.5V when the Fsys is 4 MHz**
8. **PA1 ~ PA4 individual pin low level wake up**
9. **System Oscillation Sources (Fsys)**
 - Fast-clock
 - FIRC (Fast Internal RC): 1 MHz / 2 MHz / 4 MHz
 - Slow-clock
 - SIRC (Slow Internal RC)
 $V_{DD} = 3V$, SIRC = 110 KHz / 27.5 KHz / 6.88 KHz / 1.72 KHz
10. **Power Saving Operation Modes**
 - FAST Mode: Slow-clock can be disabled or enabled, Fast-clock keeps CPU running
 - SLOW Mode: Fast-clock stops, 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
11. **Dual System Clock**
 - FIRC + SIRC

12. Reset Sources

- Power On Reset
- Watchdog Reset
- Low Voltage Reset
- External pin Reset

13. 1-Level Low Voltage Reset

- TM57ME16: 1.2V (can be disabled)
- TM57ME18: 1.6V (can be disabled)

14. Operation Voltage: Low Voltage Reset Level to 4.2V**15. Operating Temperature Range: -40°C to +85°C****16. Interrupts**

- Two External Interrupt Pins
 - One pins are falling edge triggered
 - One pin is rising or falling edge triggered
- Timer0 / T2 / Wake-up Timer Interrupts
- PWM0 Interrupt

17. Watchdog (WDT) / Wake-up (WKT) Timer

- Clocked by built-in RC oscillator with 4 adjustable Reset / Interrupt time options
 $V_{DD} = 3V$, WDT/WKT = 150 ms / 75 ms / 37.5 ms / 18.75 ms
- Watchdog timer can be disabled/enabled in Power-down mode

18. I/O Port Modes

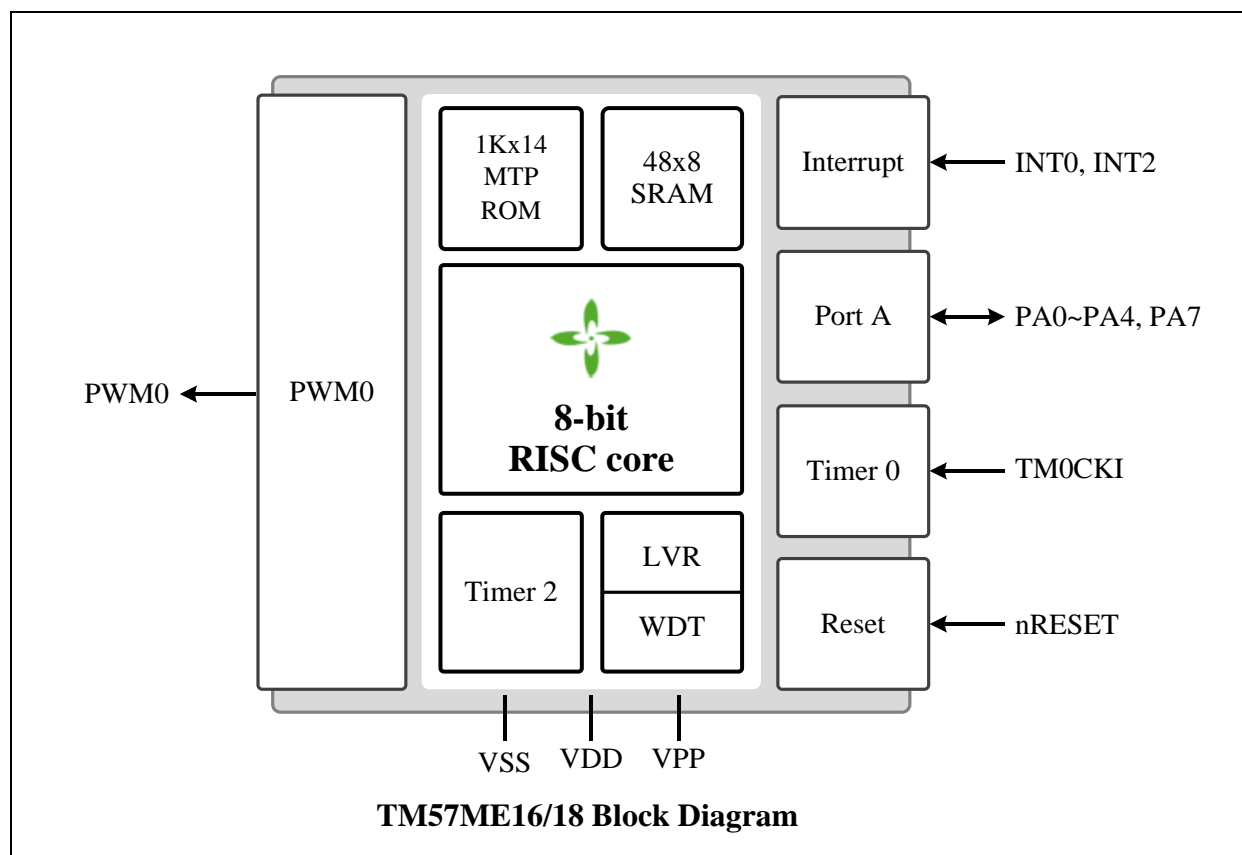
- Pseudo-Open-Drain Output (PA2 ~ PA0)
- Open-Drain Output
- CMOS Push-Pull Output
- Schmitt Trigger Input with pull-up resistor option

19. High-Sink IO (PA0)**20. Table Read Instruction: 14-bit ROM data lookup table****21. Instruction set: 39 Instructions****22. Package Types**

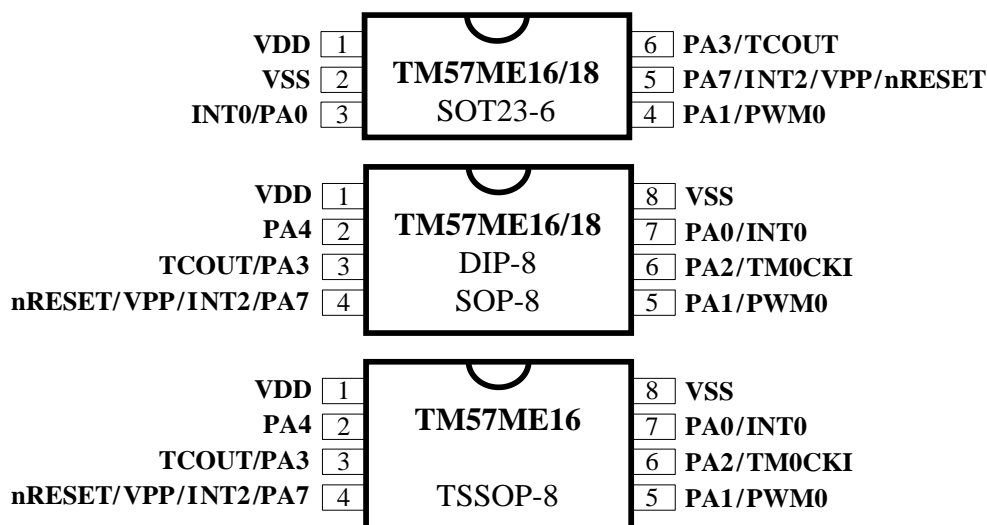
- SOT23-6
- 8-pin DIP (300 mil)
- 8-pin SOP (150 mil)
- TSSOP-8

23. Supported EV board on ICE: EV board EV2771

BLOCK DIAGRAM



PIN ASSIGNMENT



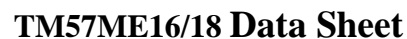
PIN DESCRIPTION

Name	In/Out	Pin Description
PA0–PA2	I/O	Bit-programmable I/O port for Schmitt-trigger input, CMOS push-pull output or “pseudo-open-drain” output. Pull-up resistors are assignable by software.
PA3–PA4	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.
PA7	I/O	Bit-programmable I/O port for Schmitt-trigger input or open-drain output. Pull-up resistor is assignable by software.
nRESET	I	External active low reset
TCOUT	O	Instruction cycle clock output. The instruction clock frequency is system clock frequency divided by two ($F_{sys}/2$)
VDD, VSS	P	Power Voltage input pin and ground
VPP	I	MTP programming high voltage input
INT0, INT2	I	External interrupt input
PWM0	O	PWM0 output
TM0CKI	I	Timer0’s input in counter mode

PIN SUMMARY

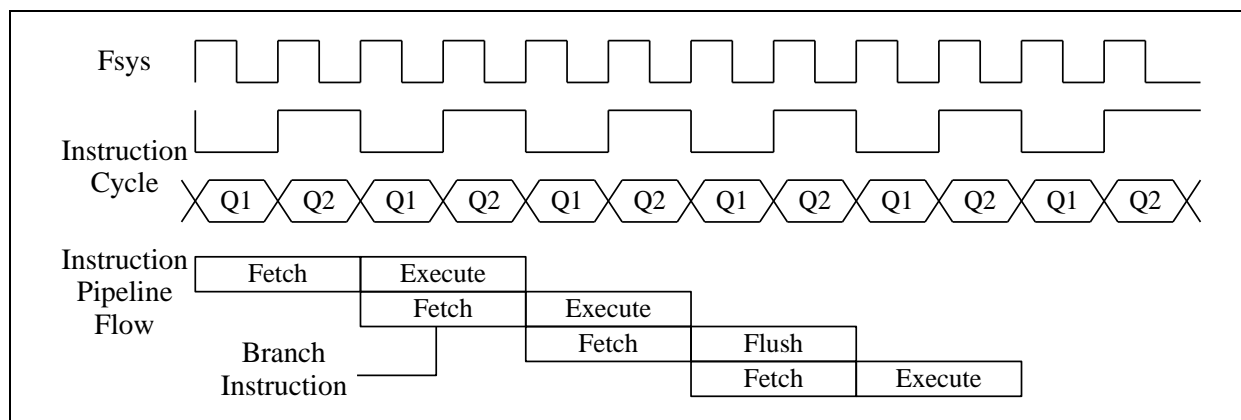
DIP-8/SOP-8/TSSOP-8	SOT23-6	Pin Name	Type	GPIO						Function After Reset	Alternate Function			
				Input		Output					PWM	Touch Key	ADC	MISC
				Weak Pull-up	Ext. Interrupt	O.D	P.O.D	P.P	High Sink					
1	1	VDD	P											
2		PA4	I/O	○		○		○		PA4				
3	6	PA3/TCOUT	I/O	○		○		○		PA3				TCOUT
4	5	PA7/INT2/ nRESET/VPP	I/O	○	○	○				SYS				nRESET
5	4	PA1/PWM0	I/O	○			○	○		PA1	○			
6		PA2/TM0CKI	I/O	○			○	○		PA2				TM0CKI
7	3	PA0/INT0	I/O	○	○		○	○	○	PA0				
8	2	VSS	P											

Symbol : P.P. = Push-Pull Output
 P.O.D. = Pseudo Open Drain
 O.D. = Open Drain
 SYS = by SYSCFG bit



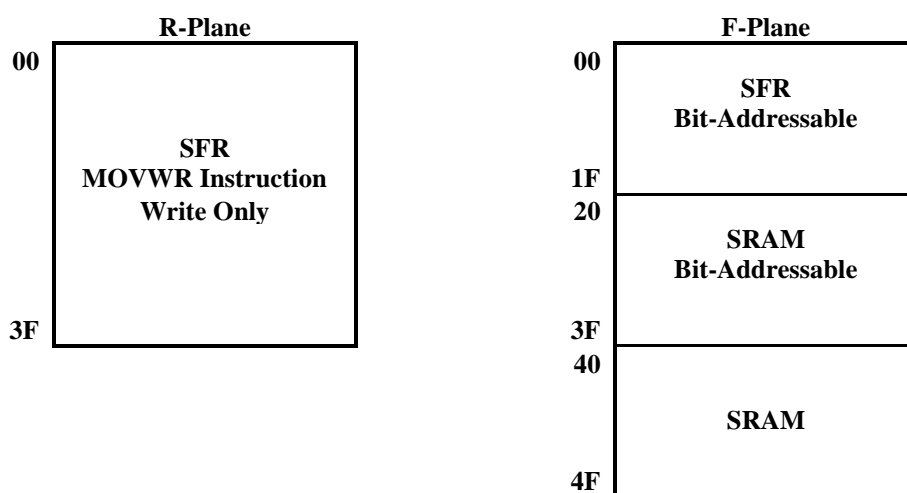
1. CPU Core

The system clock is internally divided by two to generate Q1 state and Q2 state for each instruction cycle. The Programming Counter (PC) is updated at Q1 and the instruction is fetched from program ROM and latched into the instruction register in Q2. It is then decoded and executed during the following Q1-Q2 cycle. Branch instructions take two cycles since the fetch instruction is ‘flushed’ from the pipeline, while the new instruction is being fetched and then executed.



1.2 RAM Addressing Mode

There are two Data Memory Planes in CPU, R-Plane and F-Plane. The registers in R-Plane are write-only. The “MOVWR” instruction copy the W-register’s content to R-Plane registers by direct addressing mode. The lower locations of F-Plane are reserved for the SFR. Above the SFR is General Purpose Data Memory, implemented as static RAM. F-Plane can be addressed directly or indirectly. Indirect Addressing is made by INDF register. The INDF register is not a physical register. Addressing INDF actually addresses the register whose address is contained in the FSR (F04.6~0) register (FSR is a pointer). The first half of F-Plane is bit-addressable, while the second half of F-Plane is not bit-addressable.



◇Example: Write immediate data into R-Plane register

```
MOVLW    AAH           ; Move immediate AAH into W register
MOVWR    05H           ; Move W value into R-Plane location 05H
```

◇Example: Write immediate data into F-Plane register

```
MOVLW    55H           ; Move immediate 55H into W register
MOVWF    20H           ; Move W value into F-Plane location 20H
```

◇Example: Move F-Plane location 20H data into W register

```
MOVFW    20H           ; To get a content of F-Plane location 20H to W
```

◇Example: Clear all user SRAM data by indirectly addressing mode

```

MOVWLW    20H           ; W = 20H (SRAM start address)
MOVWF     FSR           ; Set start address of user SRAM into FSR register
LOOP:
MOVLW     00H           ; Clear user SRAM data
MOVWF     INDF          ; Increment the FSR for next address
INCF      FSR, 1
MOVLW     50H           ; W = 50H (SRAM end address)
XORWF     FSR, 0        ; Check the FSR is end address of user SRAM?
BTFFS     STATUS, Z     ; Check the Z flag
```

```

GOTO      LOOP          ; If Z = 0, goto LOOP label
...
          ; If Z = 1, exit LOOP

```

1.3 Programming Counter (PC) and Stack

The Programming Counter is 10-bit wide capable of addressing a 1K x 14 MTP 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 the followings. The Reset Vector (000h) and the Interrupt Vector (001h) are provided for PC initialization and Interrupt. For CALL / GOTO instructions, PC loads 10 bits address from instruction word. For RET / RETI / RETLW instructions, PC retrieves its content from the top level STACK. For the other instructions updating PC [7:0], the PC [9:8] keeps unchanged. Therefore, the data of a lookup table must be located with the same PC [9:8]. The STACK is 10-bit wide and 5-level in depth. The CALL instruction and hardware interrupt will push STACK level in order. While the RET / RETI / RETLW instructions pop the STACK level in order.

For table lookup, the device offers the powerful table read instructions TABRL, TABRH to return the 14-bit ROM data into W register by setting the DPTR = {DPH, DPL} registers in F-Plane.

◇Example: To look up the MTP data located “TABLE”

```

          ORG      000H          ; Reset Vector
START:    GOTO     START        ; Goto user program address

          MOVLW    00H
          MOVWF    INDEX        ; Set lookup table's address (INDEX)
LOOP:     MOVFW    INDEX        ; Move INDEX value to W register
          CALL     TABLE      ; To Lookup data (W = 55H when INDEX = 00H)
          ...
          INCF     INDEX, 1     ; Increment the INDEX for next address
          ...
          GOTO     LOOP        ; Goto LOOP label

TABLE:    ORG      X00H          ; X = 1, 2, 3
          ADDWF    PCL, 1       ; (Addr = X00H) Add the W with PCL, the result
          ; back in PCL
          RETLW    55H          ; W = 55H when return
          RETLW    56H          ; W = 56H when return
          RETLW    58H          ; W = 58H when return

```

Note: TM57ME16/18 defines 256 ROM addresses as one page, so that TM57ME16/18 has four pages, 000H~0FFH, 100H~1FFH, 200H~2FFH, and 300H~3FFH. On the other words, PC[9:8] can be defined as page. A lookup table must be located at the same page to avoid getting wrong data. Thus, the lookup table has maximum 255 data for above example with starting a lookup table at X00H (X = 1, 2, 3). If a lookup table has fewer data, it needs not setting the starting address at X00H, but only confirms all lookup table data are located at the same page.

◇Example: To look up the MTP data located “TABLE” by TABRL and TABRH instructions

	ORG	000H	; Reset Vector
	GOTO	START	; Goto user program address
START:	MOVLW	(TABLE >>8) & 0xff	; Get high byte address of TABLE label
	MOVWF	DPH	; DPH (F17.1~0) = 02H
	MOVLW	(TABLE) & 0xff	; Get low byte address of TABLE label
	MOVWF	DPL	; DPL (F04.7~0) = 80H
LOOP:	TABRL		; W = 86H when DTPR = {DPH, DPL} = 0280H
	TABRH		; W = 19H when DTPR = {DPH, DPL} = 0280H
	...		
	INCF	DPL, 1	; Increment the DPL for next address
	...		
	GOTO	LOOP	; Goto LOOP label
TABLE:	ORG	280H	
	DT	0x1986	; 14-bit ROM data
	DT	0x3719	; 14-bit ROM data

1.4 ALU and Working (W) Register

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

Note: /Borrow represents inverted of Borrow register.

/Digit Borrow represents inverted of Digit Borrow register.

1.5 STATUS Register (F-Plane 03H)

This register contains the arithmetic status of ALU, the reset status, and the voltage 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 BCF, BSF and MOVWF instructions are used to alter the STATUS register because these instructions do not affect those bits.

STATUS	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Reset Value	–	0	–	0	0	0	0	0
R/W	–	R/W	–	R	R	R/W	R/W	R/W
Bit	Description							
7	Not Used							
6	GB0: General Purpose Bit 0							
5	Not Used							
4	TO: Time Out Flag 0: after Power On Reset, LVR Reset, or CLRWDT / SLEEP instructions 1: WDT time out occurs							
3	PD: Power Down Flag 0: after Power On Reset, LVR Reset, or CLRWDT instruction 1: after SLEEP instruction							
2	Z: Zero Flag 0: the result of a logic operation is not zero 1: the result of a logic operation is zero							
1	DC: Decimal Carry Flag or Decimal /Borrow Flag							
	ADD instruction				SUB instruction			
	0: no carry 1: a carry from the low nibble bits of the result occurs				0: a borrow from the low nibble bits of the result occurs 1: no borrow			
0	C: Carry Flag or / Borrow Flag							
	ADD instruction				SUB instruction			
	0: no carry 1: a carry occurs from the MSB				0: a borrow occurs from the MSB 1: no borrow			

◇Example: Write immediate data into STATUS register

```
MOVLW    00H
MOVWF    STATUS           ; Clear STATUS register
```

◇Example: Bit addressing set and clear STATUS register

```
BSF      STATUS, 0        ; Set C = 1
BCF      STATUS, 0        ; Clear C = 0
```

◇Example: Determine the C flag by BTFSS instruction

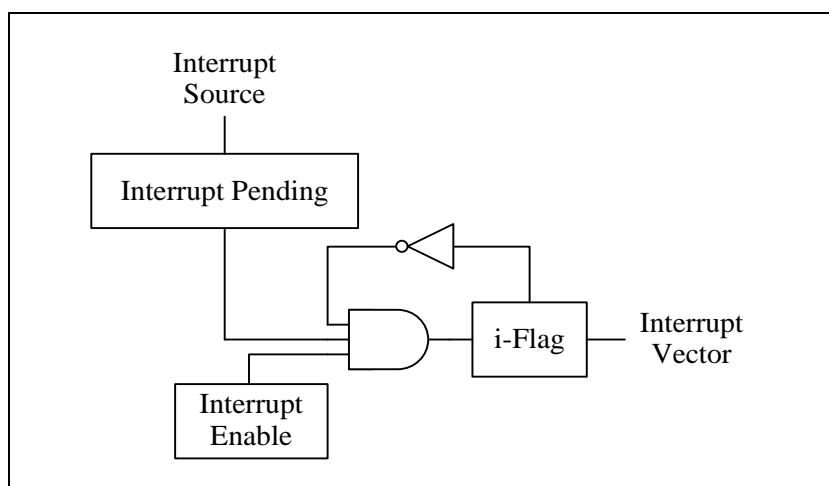
```
BTFSS    STATUS, 0        ; Check the C flag
GOTO     LABEL_1          ; If C = 0, goto LABEL_1 label
GOTO     LABEL_2          ; If C = 1, goto LABEL_2 label
```

1.6 Interrupt

The TM57ME16/18 has 1 level, 1 vector and 6 interrupt sources. Each interrupt source has its own enable control bit. An interrupt event will set its individual pending flag; no matter its interrupt enable control bit is 0 or 1. Because TM57ME16/18 has only 1 vector, there is no interrupt priority register. The interrupt priority is determined by F/W.

If the corresponding interrupt enable bit has been set (INTIE), it would trigger CPU to service the interrupt. CPU accepts interrupt in the end of current executed instruction cycle. In the mean while, a “CALL 001” 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 INT0 (PA0) interrupt request with rising edge trigger

```

ORG      000H          ; Reset Vector
GOTO     START         ; Goto user program address

ORG      001H          ; All interrupt vector
GOTO     INT           ; If INT0 (PA0) input occurred rising edge

START:
ORG      002H
MOVLW    xxxxxx0B
MOVWR    PAE           ; Disable INT0 (PA0) CMOS push-pull output
                        ; mode
MOVLW    xxxxxx0B
MOVWR    PAPUN         ; Enable INT0 (PA0) input pull-up resistor
MOVLW    xxxxxx1B
MOVWF    PAD           ; Release INT0 (PA0), it becomes Schmitt-trigger
                        ; input mode with input pull-up resistor
MOVLW    0001x0xxB
MOVWR    R0B           ; Set INT0 interrupt trigger as rising edge
MOVLW    1111110B
MOVWF    INTIF         ; Clear INT0 interrupt request flag
MOVLW    00000001B
MOVWF    INTIE         ; Enable INT0 interrupt

MAIN:
...
GOTO     MAIN

INT:
MOVWF    40H           ; Store W data to SRAM 40H
MOVFW    STATUS        ; Get STATUS data
MOVWF    41H           ; Store STATUS data to SRAM 41H

BTFSS    INT0IF        ; Check INT0IF bit
GOTO     EXIT_INT      ; INT0IF = 0, exit interrupt subroutine
                        ; INT0 interrupt service routine
...
MOVLW    1111110B
MOVWF    INTIF         ; Clear INT0 interrupt request flag

EXIT_INT:
MOVFW    41H           ; Get SRAM 41H data
MOVWF    STATUS        ; Restore STATUS data
MOVFW    40H           ; Restore W data
RETI              ; Return from interrupt

```

F08	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTIE	PWM0IE	T2IE	—	TM0IE	WKTIE	INT2IE	—	INT0IE
R/W	R/W	R/W	—	R/W	R/W	R/W	—	R/W
Reset	0	0	—	0	0	0	—	0

F08.7 **PWM0IE**: PWM0 interrupt enable

0: disable

1: enable

F08.6 **T2IE**: T2 interrupt enable

0: disable

1: enable

F08.4 **TM0IE**: Timer0 interrupt enable

0: disable

1: enable

F08.3 **WKTIE**: Wakeup Timer interrupt enable

0: disable

1: enable

F08.2 **INT2IE**: INT2 (PA7) pin interrupt enable

0: disable

1: enable

F08.0 **INT0IE**: INT0 (PA0) pin interrupt enable

0: disable

1: enable

F09	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTIF	PWM0IF	T2IF	—	TM0IF	WKTIF	INT2IF	—	INT0IF
R/W	R/W	R/W	—	R/W	R/W	R/W	—	R/W
Reset	0	0	—	0	0	0	—	0

F09.7 **PWM0IF**: PWM0 interrupt event pending flag

This bit is set by H/W while PWM0 overflows, write 0 to this bit will clear this flag

F09.6 **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

F09.4 **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

F09.3 **WKTIF**: WKT interrupt event pending flag

This bit is set by H/W while WKT time out, write 0 to this bit will clear this flag

F09.2 **INT2IF**: INT2 interrupt event pending flag

This bit is set by H/W at INT2 pin's falling edge, write 0 to this bit will clear this flag

F09.0 **INT0IF**: INT0 interrupt event 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

R0B	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
MR0B	—	—	—	INT0EDG	TCOE	—	WKTTPSC	
R/W	—	—	—	W	W	—	W	
Reset	—	—	—	0	0	—	0	0

R0B.4 **INT0EDG**: INT0 (PA0) trigger edge select

0: INT0 (PA0) pin falling edge to trigger interrupt event

1: INT0 (PA0) pin rising edge to trigger interrupt event

2. Chip Operation Mode

2.1 Reset

The TM57ME16/18 can be RESET in four ways.

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

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. The Low Voltage Reset features static reset when supply voltage is below a threshold level. There are one threshold levels can be selected. The LVR's operation mode is defined by the SYSCFG register.

The External Pin Reset and Watchdog Reset can be disabled or enabled by the SYSCFG register. These two resets also set all the control registers to their default reset value.

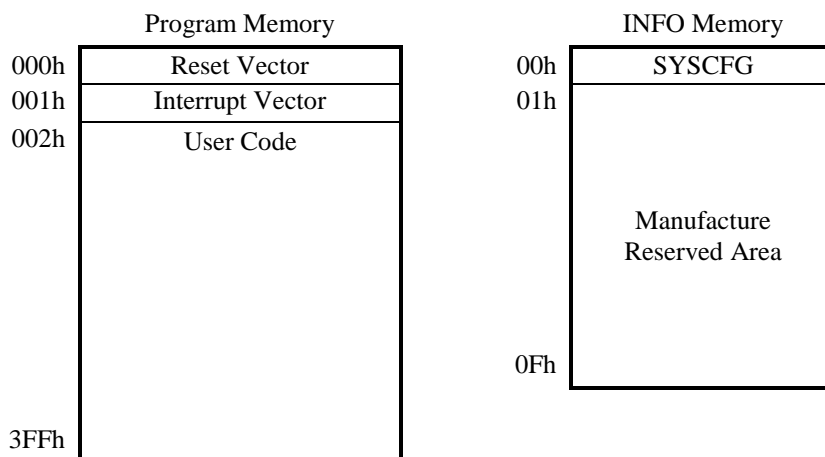
2.2 System Configuration Register (SYSCFG)

The System Configuration Register (SYSCFG) is located at MTP INFO area. The SYSCFG determines the option for initial condition of MCU. It is written by MTP Writer only. User can select clock source, LVR threshold voltage and chip operation mode by SYSCFG register. The default value of SYSCFG is 0000h. The 13th bit of SYSCFG is code protection selection bit. If this bit is 1, the data in MTP will be protected, when user reads MTP.

Bit	13~0	
Default Value	00_0000_0000_0000	
Bit	Description	
13	PROTECT : Code Protection Selection	
	0	Disable
	1	Enable
12-11	Reserved	
10	LVR : Low Voltage Reset Mode	
	0	LVR disable
	1	TM57ME16: LVR = 1.2V, always enable TM57ME18: LVR = 1.6V, always enable
9	Reserved	
8	CLKS : Power on Clock source Selection	
	0	CPU Clock runs Fast-clock after Power On or Reset
	1	CPU Clock runs Slow-clock after Power On or Reset
7	XRSTE : External Pin (PA7) Reset Enable	
	0	Disable, PA7 as IO pin
	1	Enable
6	WDTE : WDT Reset Enable	
	0	Disable
	1	Enable
5	FIRC : Fast Internal RC Frequency Clock Source Selection	
	0	FIRC clock = 4 MHz when FIRCKS (R19.7~6) = "10"
	1	FIRC clock = 2 MHz when FIRCKS (R19.7~6) = "10"
4-0	Reserved	

2.3 MTP ROM

The MTP Program ROM of this device is 1K words, with an extra INFO area to store the SYSCFG and manufacture data. The MTP ROM can be written multi-times and can be read as long as the PROTECT bit of SYSCFG is not set. The SYSCFG can be read no matter PROTECT is set or cleared, but can be written only when PROTECT is cleared or MTP ROM is blank. That is, unprotect the PROTECT bit can be done only if the Program ROM area is blank. The tenx certified writer can do the above actions with the sophisticated software.



2.4 Power-Down Mode

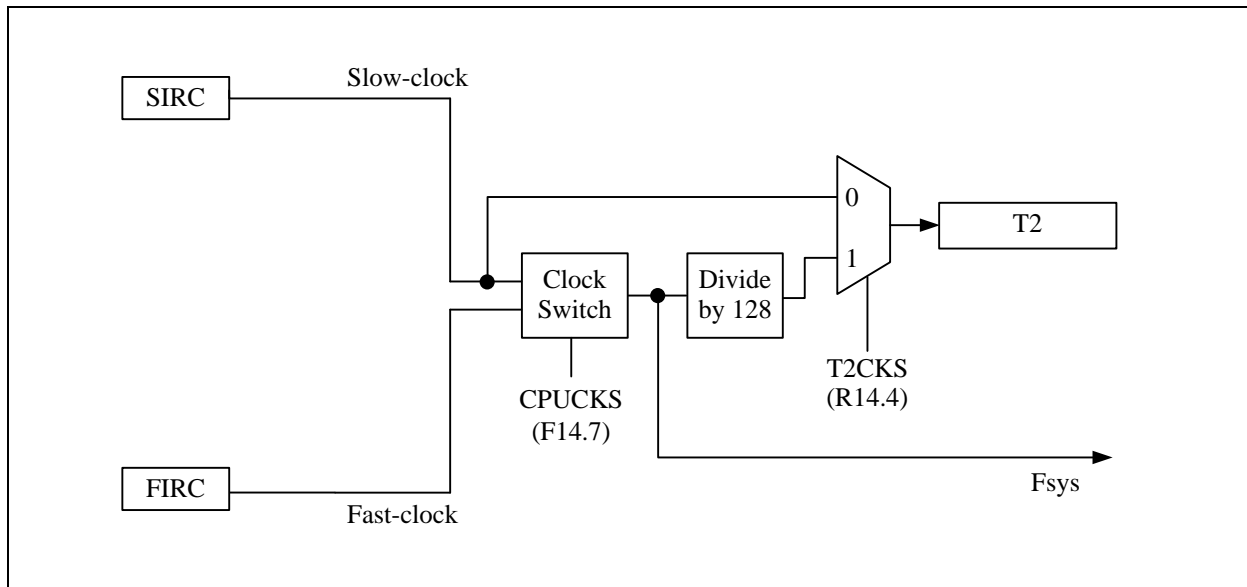
The Power-down mode includes IDLE Mode and STOP Mode. It is activated by SLEEP instruction. During the Power-down mode, the system clock and peripherals stop to minimize power consumption, whether the WDT / WKT / T2 Timer are working or not depend on F/W setting. The Power-down mode can be terminated by Reset, or enabled Interrupts (External pins and WKT / T2 interrupts) or PA1-4 pins low level wake up.

R03	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWRDN	PWRDN							
R/W	W							
Reset	—	—	—	—	—	—	—	—

R03.7~0 **PWRDN:** Write this register to enter Power-down (STOP / IDLE) Mode

2.5 Dual System Clock

TM57ME16/18 is designed with dual-clock system. There are two kinds of clock source, SIRC (Slow Internal RC) Clock and FIRC (Fast Internal RC) Clock. Each clock source can be applied to CPU kernel as system clock. When in IDLE mode, only Slow-clock can be configured to keep oscillating to provide clock source to T2 block. Refer to the Figure as below.



FAST Mode:

After power on or reset, if SYSCFG [8] is cleared, TM57ME16/18 enters FAST mode, otherwise enters SLOW mode. In FAST mode, firmware can also enable or disable the Slow-clock for the T2 system operating.

In this mode, the program is executed using Fast-clock as CPU clock (Fsys). The Timer0 block and PWM0 block are driven by Fast-clock. T2 can be driven by Slow-clock or Fast-clock by setting T2CKS (R14.4). If T2CKS is cleared and SLOWEN (F14.5) is set, T2 can be driven by Slow-clock in FAST mode.

SLOW Mode:

After power on or reset, if SYSCFG [8] is set, TM57ME16/18 enters SLOW mode. In this mode, Fast-clock is stopped and Slow-clock is enabled for power saving. All peripheral blocks (Timer0, PWM0, T2, etc...) clock sources are Slow-clock in the SLOW mode.

IDLE Mode:

When SLOWEN (F14.5) is set and T2CKS (R14.4) is cleared, the TM57ME16/18 will enter the “IDLE Mode” after executing the SLEEP instruction. In this mode, the Slow-clock will continue running to provide clock to T2 block. CPU stops fetching code and all blocks are stop except T2 related circuits.

Another way to keep clock oscillation in IDLE mode is setting WKTIE = 1 (F08.3) before executing the SLEEP instruction. In such condition, the WKT keeps working and wakes 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, which is useful for low power mode Touch Key detection.

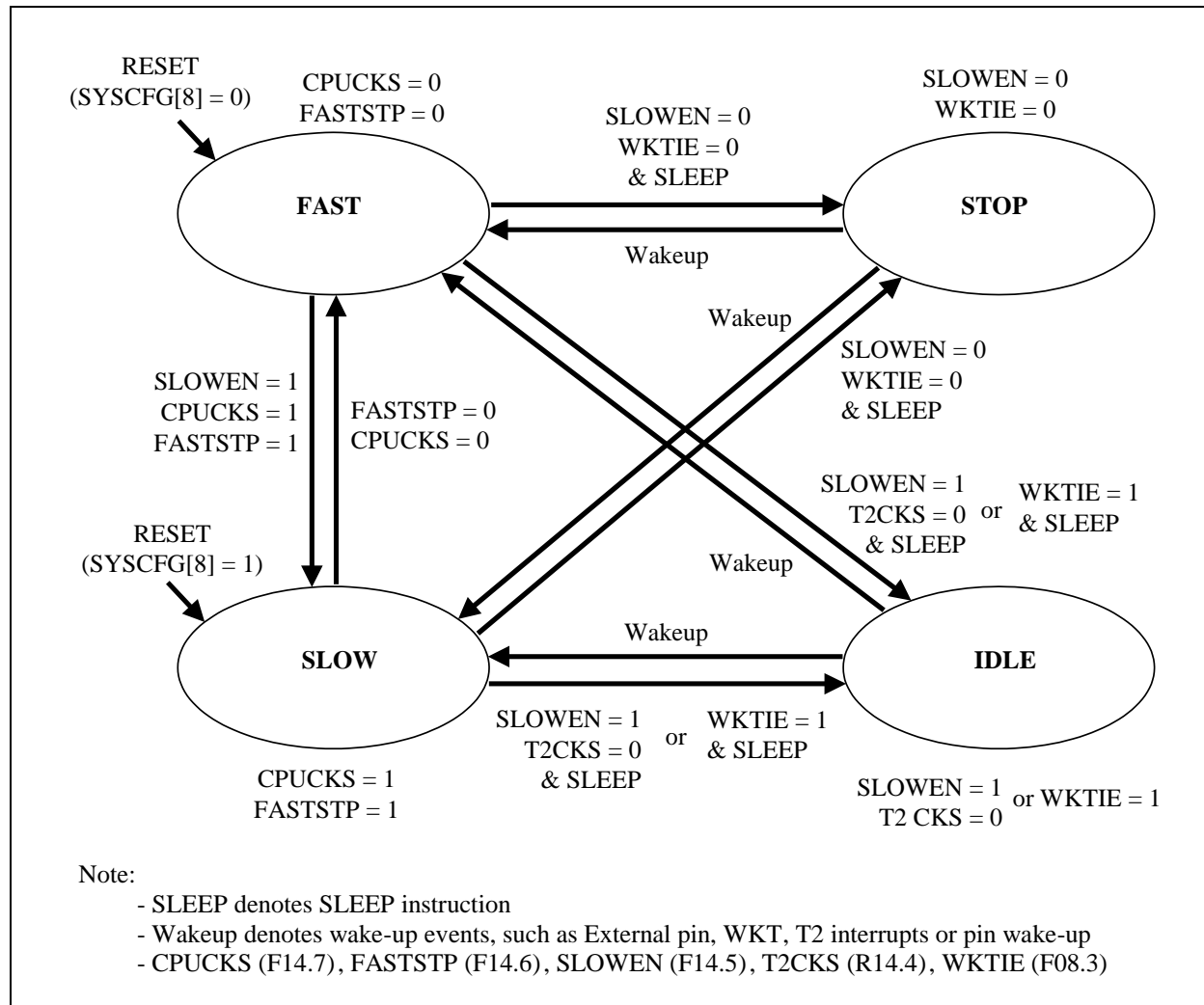
STOP Mode:

When SLOWEN (F14.5) and WKTIE (F08.3) are cleared, all blocks will be turned off and the TM57ME16/18 will enter the “STOP Mode” after executing the SLEEP instruction. STOP mode is similar to IDLE mode. The difference is all clock oscillators either Fast-clock or Slow-clock are stopped and no clocks are generated.

2.6 Dual System Clock Modes Transition

TM57ME16/18 is operated in one of four modes: FAST Mode, SLOW Mode, IDLE Mode, and STOP Mode.

Modes Transition Diagram:



CPU Mode & Clock Functions Table:

Mode	Oscillator	Fsys	Fast-clock	Slow-clock	TM0/PWM0	WKT	T2	Wakeup event
FAST	FIRC	Fast-clock	Run	Run/Stop	Run	Run/Stop	Run	X
SLOW	SIRC	Slow-clock	Stop	Run	Run	Run/Stop	Run	X
IDLE	SIRC	Stop	Stop	Run/Stop	Stop	Run/Stop	Run	WKT/T2/IO
STOP	Stop	Stop	Stop	Stop	Stop	Stop	Stop	IO

FAST Mode transits to SLOW Mode:

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

- (1) Enable Slow-clock (SLOWEN = 1)
- (2) Switch system clock source to Slow-clock (CPUCKS = 1)
- (3) Stop Fast-clock (FASTSTP = 1)

Note: Stop Fast-clock (FASTSTP = 1) is optional. If R11.3 (PWMCKS) is set, FASTSTP must be cleared. Otherwise FIRC 8M will not oscillate.

◇Example: Switch operating mode from FAST mode to SLOW mode with SIRC

BSF	SLOWEN	; Enable Slow-clock
BSF	CPUCKS	; Switch system clock source to Slow-clock
BSF	FASTSTP	; Stop Fast-clock

SLOW Mode transits to FAST Mode:

Slow-clock can be enabled by the SLOWEN (F14.5) or the CPUCKS (F14.7) bits. The following steps are suggested to be executed by order when SLOW mode transits to FAST mode:

- (1) Enable Fast-clock (FASTSTP = 0)
- (2) Switch system clock source to Fast-clock (CPUCKS = 0)
- (3) Stop Slow-clock (SLOWEN = 0)

Note: Stop Slow-clock (SLOWEN = 0) is optional. Slow-clock can keep oscillating to provide T2 counter block in FAST mode.

◇Example: Switch operating mode from SLOW mode to FAST mode

BCF	FASTSTP	; Enable Fast-clock
BCF	CPUCKS	; Switch system clock source to Fast-clock
BCF	SLOWEN	; Stop Slow-clock

IDLE Mode Setting:

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

- (1) Switch T2 clock source to Slow-clock (T2CKS = 0)
- (2) Enable Slow-clock (SLOWEN = 1)
- (3) Execute SLEEP instruction

IDLE mode can be woken up by interrupts (XINT, WKT or T2) or PA1-4 pins low level wake up.

◇Example: Switch operating mode to IDLE mode (T2 clock source is Slow-clock divided by 32768)

```

MOVLW    00000000B    ; T2 clock source is Slow-clock divided by 32768
MOVWR    R14           ; Slow-clock source is SIRC
BSF       SLOWEN       ; Stop Fast-clock
SLEEP                    ; Enter IDLE mode

```

STOP Mode Setting:

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

- (1) Stop Slow-clock (SLOWEN = 0)
- (2) Disable WDT / WKT (WKTIE = 0)
- (3) Execute SLEEP instruction

STOP mode can be woken up by interrupt (XINT) or PA1-4 pins low level wake up.

◇Example: Switch operating mode to STOP mode

```

BCF       SLOWEN       ; Stop Slow-clock
BCF       WKTIE        ; Disable WKT / WDT
SLEEP                    ; Enter STOP mode

```

IO setting notes in STOP/IDLE mode:

F14	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
MF14	CPUCKS	FASTSTP	SLOWEN	—	HSNK	T2CLR	TM0STP	PWM0CLR
R/W	R/W	R/W	R/W	—	R/W	R/W	R/W	R/W
Reset	1/0	0	1	—	0	0	0	1

F14.7 **CPUCKS**: System clock (Fsys) selection, the reset value depends on SYSCFG [8].

0: Fast-clock

1: Slow-clock

If SLOWM (SYSCFG [8]) = 1, the reset value is 1, otherwise is 0

F14.6 **FASTSTP**: Fast-clock & FIRC 8M Enable / Disable

0: Fast-clock & FIRC 8M enable

1: Fast-clock & FIRC 8M disable

F14.5 **SLOWEN**: Slow-clock Enable / Disable

0: Slow-clock is disabled, except CPUCKS = 1

1: Slow-clock is enabled

R11	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWM0CTL	—	—	—	—	PWMCKS	PWM0OE	PWM0PSC	
R/W	—	—	—	—	W	W	W	
Reset	—	—	—	—	0	0	0	0

R11.3 **PWMCKS**: PWM Clock source select

0: Fsys

1: FIRC 8MHz

R14	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
MR14	–	–	–	T2CKS	T2PSC		–	–
R/W	–	–	–	W	W		–	–
Reset	–	–	–	0	0	0	–	–

R14.4 **T2CKS**: T2 clock source selection
 0: Slow-clock
 1: Fsys/128

R0B	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
MR0B	–	–	–	INT0EDG	TCOE	–	WKTTPSC	
R/W	–	–	–	W	W	–	W	
Reset	–	–	–	0	0	–	0	0

R0B.1~0 **WKTTPSC**: WDT / WKT pre-scale option or SIRC frequency select
 WDT / WKT pre-scale select @ 3V:
 00: 18.75 ms
 01: 37.5 ms
 10: 75 ms
 11: 150 ms

SIRC frequency select @ 3V:
 00: 110 KHz
 01: 27.5 KHz
 10: 6.88 KHz
 11: 1.72 KHz

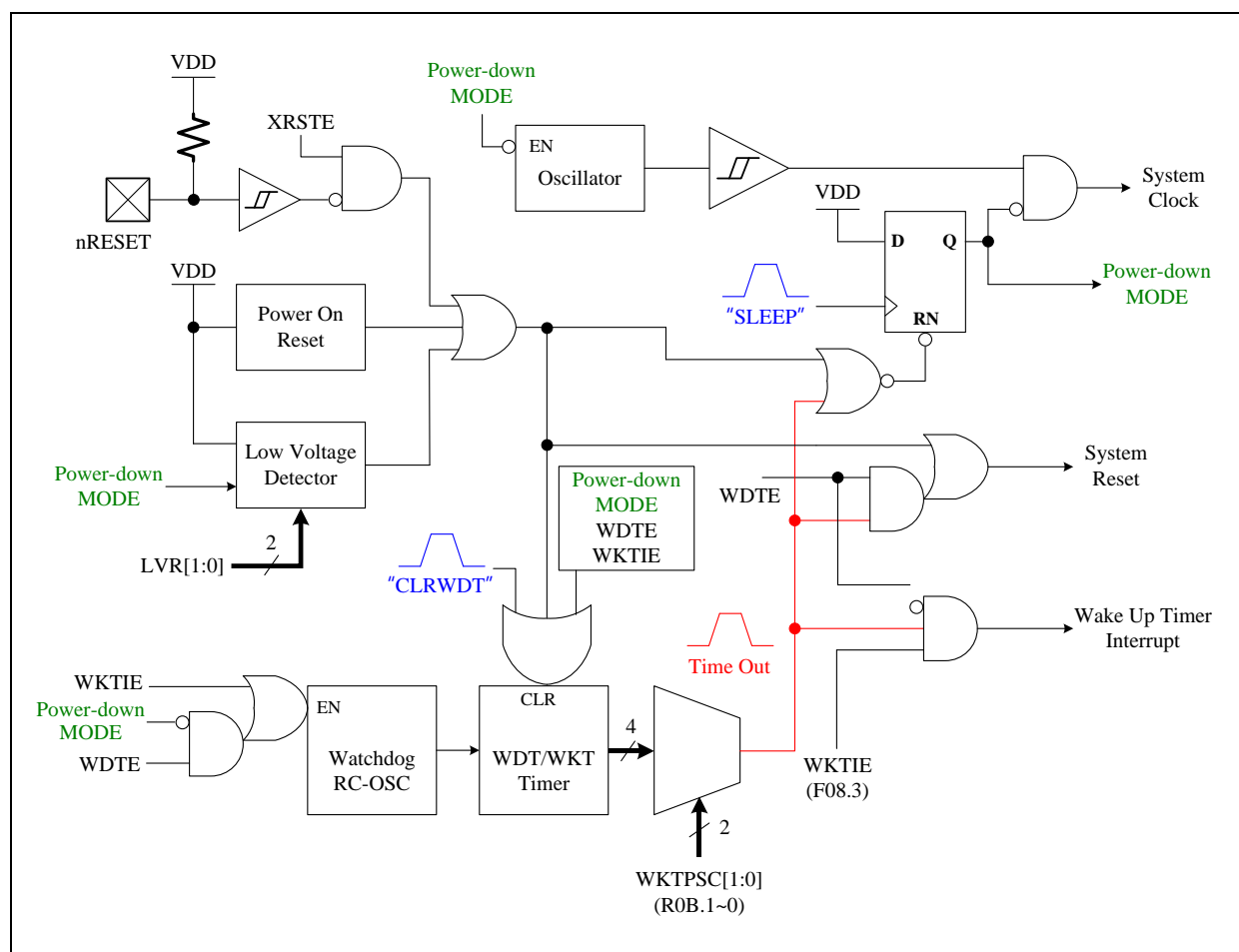
R19	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
FIRCKS	FIRCKS		–	–	–	–	–	–
R/W	W		–	–	–	–	–	–
Reset	1	0	–	–	–	–	–	–

R19.7~6 **FIRCKS**: FIRC clock source selection
 00: 1 MHz
 01: 2 MHz
 10: 4 or 2 MHz (4 MHz if SYSCFG [5] = 0, 2 MHz if SYSCFG [5] = 1)
 11: 4 MHz

3. Peripheral Functional Block

3.1 Watchdog (WDT) / Wakeup (WKT) Timer

The WDT and WKT share the same internal RC Timer which clock source is from SIRC. The overflow period of WDT / WKT can be selected from 18.75 ms to 150 ms. The WDT / WKT timer is cleared by the CLRWDT instruction. If the Watchdog Reset is enabled (SYSCFG [6], WDTE = 1), the WDT generates the chip reset signal, otherwise, the WKT only generates overflow time out interrupt. The WDT / WKT works in both normal mode and IDLE mode. During IDLE mode, user can further choose to enable or disable the WDT/WKT by "WKTIE" (F08.3). If WKTIE is cleared in IDLE mode (no matter WDTE is 1 or 0), the internal RC Timer stops for power saving. In other words, user keeps the WDT / WKT alive in IDLE mode by setting WKTIE = 1. If the WDTE is set and WKTIE is cleared, WDT / WKT timer will be cleared and stopped for power saving in IDLE mode. If the WDTE and WKTIE are set, WDT / WKT timer keeps counting in IDLE / normal mode. Refer to the following table and figure.



WDT / WKT Block Diagram

The WDT and WKT's behavior in different Mode are shown as below table.

Mode	WDTE	WKTIE	Watchdog RC Oscillator
Normal Mode	0	0	Stop
	0	1	Run
	1	0	
	1	1	
Power-down Mode	0	0	Stop
	0	1	Run
	1	0	Stop
	1	1	Run

F03	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
STATUS	–	GB0	–	TO	PD	Z	DC	C
R/W	–	R/W	–	R	R	R/W	R/W	R/W
Reset	–	0	–	0	0	0	0	0

F03.4 **TO:** WDT time out flag, read-only

0: after Power On Reset, LVR Reset, or CLRWDT / SLEEP instructions

1: WDT time out occurs

F08	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTIE	PWM0IE	T2IE	–	TM0IE	WKTIE	INT2IE	–	INT0IE
R/W	R/W	R/W	–	R/W	R/W	R/W	–	R/W
Reset	0	0	–	0	0	0	–	0

F08.3 **WKTIE:** Wakeup Timer interrupt enable

0: disable

1: enable

F09	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTIF	PWM0IF	T2IF	–	TM0IF	WKTIF	INT2IF	–	INT0IF
R/W	R/W	R/W	–	R/W	R/W	R/W	–	R/W
Reset	0	0	–	0	0	0	–	0

F09.3 **WKTIF:** WKT interrupt event pending flag

This bit is set by H/W while WKT time out, write 0 to this bit will clear this flag

R04	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
WDTCLR	WDTCLR							
R/W	W							
Reset	–	–	–	–	–	–	–	–

R04.7~0 **WDTCLR:** Write this register to clear WDT/WKT

R0B	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
MR0B	—	—	—	INT0EDG	TCOE	—	WKTTPSC	
R/W	—	—	—	W	W	—	W	
Reset	—	—	—	0	0	—	0	0

R0B.1~0 **WKTTPSC:** WDT / WKT pre-scale option or SIRC frequency select

WDT/WKT pre-scale select @ 3V:

00: 18.75 ms

01: 37.5 ms

10: 75 ms

11: 150 ms

SIRC frequency select @ 3V:

00: 110 KHz

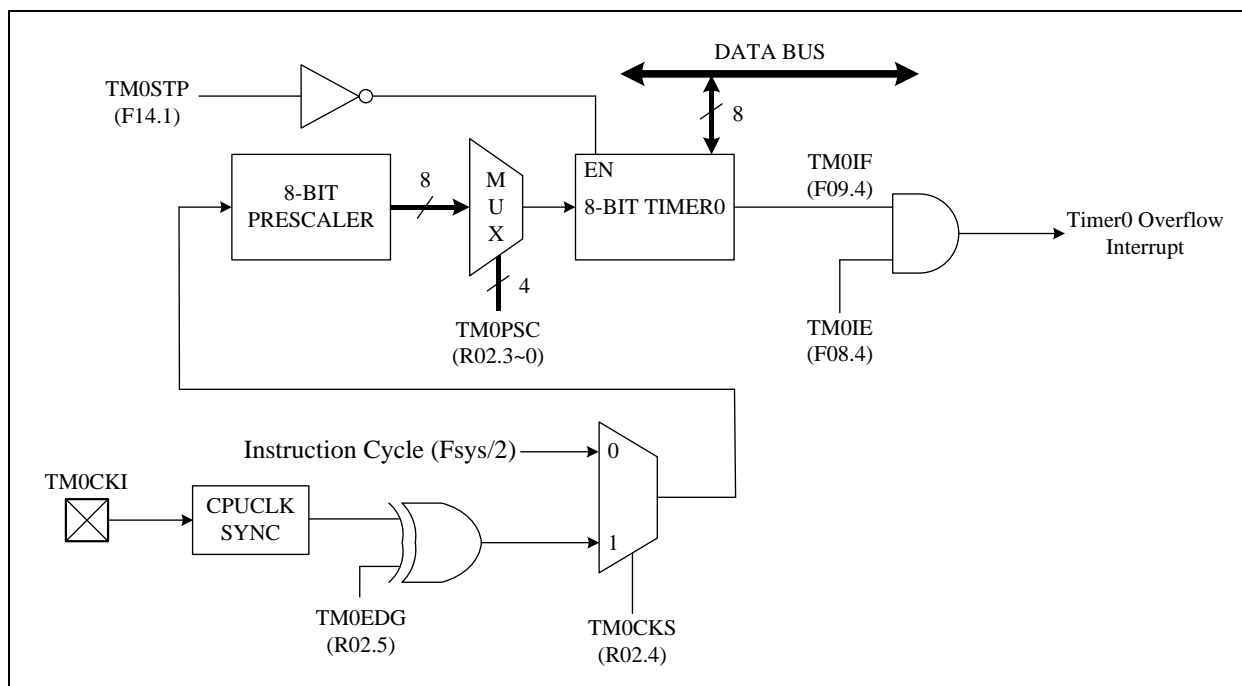
01: 27.5 KHz

10: 6.88 KHz

11: 1.72 KHz

3.2 Timer0: 8-bit Timer / Counter with Pre-scale (PSC)

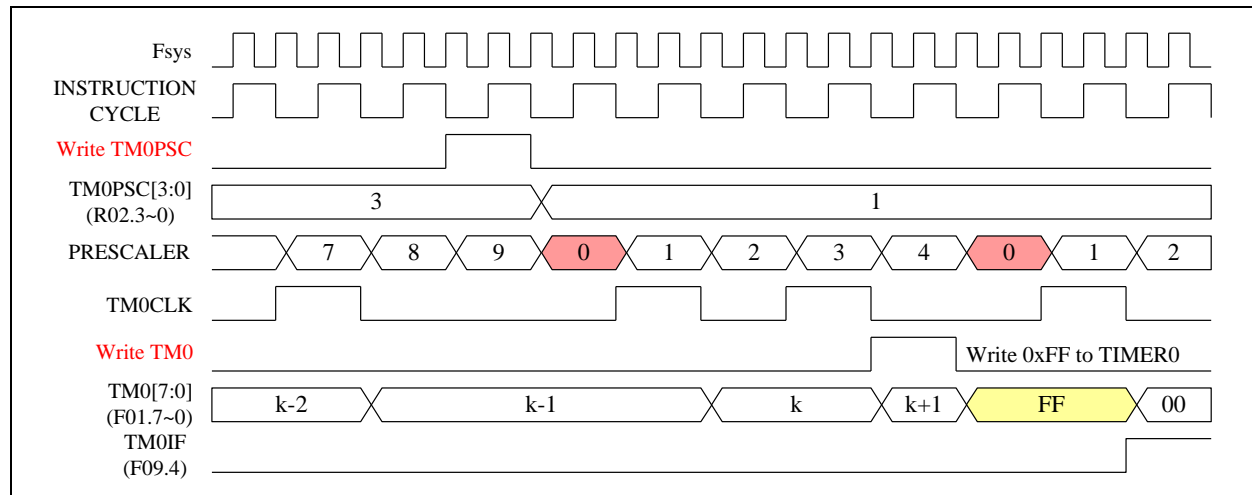
The Timer0 is an 8-bit wide register of F-Plane. It can be read or written as any other register of F-Plane. Besides, Timer0 increases itself periodically and automatically rolls over based on the pre-scaled clock source, which can be the instruction cycle or TM0CKI (PA2) rising / falling input. The Timer0's increasing rate is determined by the TM0PSC [3:0] (R02.3~0) bits in R-Plane. The Timer0 can generate interrupt flag TM0IF (F09.4) when it rolls over. It generates Timer0 interrupt if the TM0IE (F08.4) bit is set. Timer0 can be stopped counting if the TM0STP (F14.1) bit is set.



Timer0 Block Diagram

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 00h, TM0IF (Timer0 Interrupt Flag) will be set to 1 and generate interrupt if TM0IE (Timer0 Interrupt Enable) is set. The following timing diagram describes the Timer0 works in pure Timer mode.



Timer0 works in Timer mode (TM0CKS = 0)

The equation of Timer0 interrupt timer value is as following:

$$\text{Timer0 interrupt interval cycle time} = \text{Instruction cycle time} / \text{TM0PSC} / 256$$

◇Example: Setup Timer0 work in Timer mode, Fsys = Fast-clock = FIRC 4 MHz

; Setup Timer0 clock source and divider

```
MOVLW    00x00101B    ; TM0CKS = 0, Timer0 clock is instruction cycle
MOVWR    TM0CTL        ; TM0PSC = 0101b, divided by 32
```

; Setup Timer0

```
BSF      TM0STP        ; Timer0 stops counting
CLRFR    TM0           ; Clear Timer0 content
```

; Enable Timer0 and interrupt function

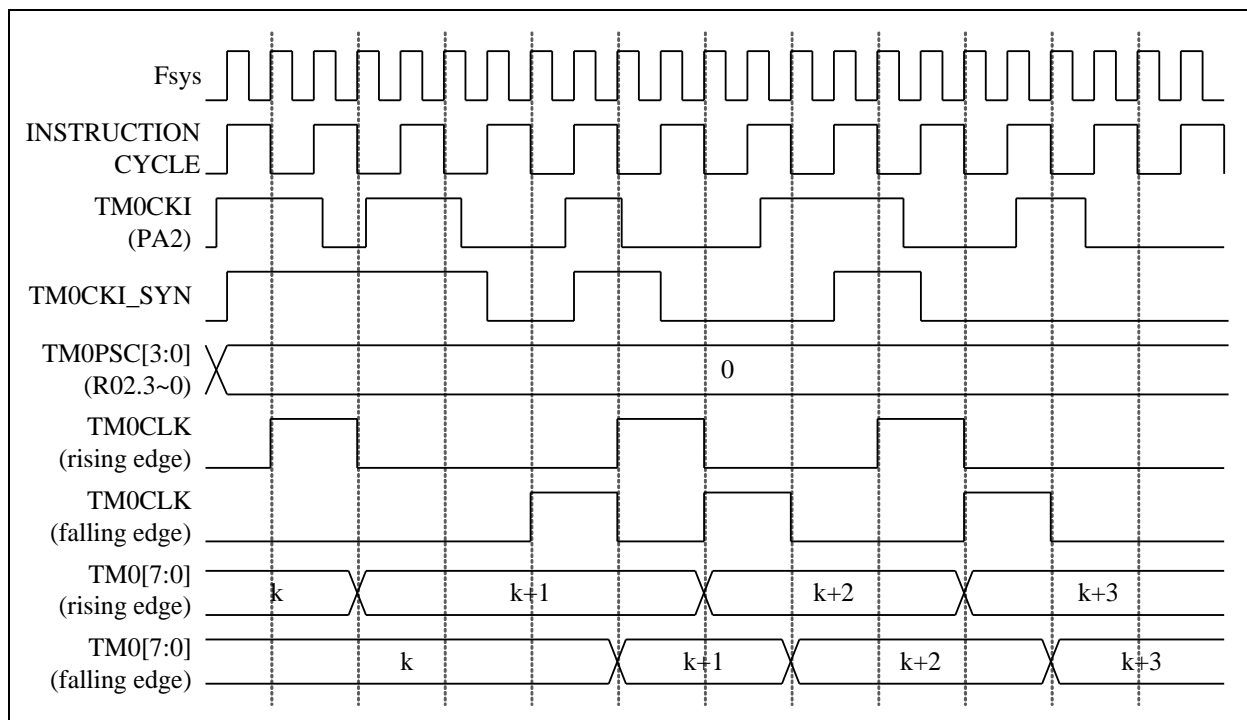
```
MOVLW    11101111B    ; Clear Timer0 request interrupt flag
MOVWF    INTIF         ; Clear Timer0 request interrupt flag
BSF      TM0IE         ; Enable Timer0 interrupt function
BCF      TM0STP        ; Enable Timer0 counting
```

Timer0 clock source is $F_{sys}/2 = 4 \text{ MHz} / 2 = 2 \text{ MHz}$, Timer0 divided by 32

$$\text{Timer0 interrupt frequency} = 2 \text{ MHz} / 32 / 256 = 244.14 \text{ Hz}$$

Counter Mode:

If $TM0CKS = 1$, then Timer0 counter source clock is from $TM0CKI$ (PA2) pin. $TM0CKI$ signal is synchronized by instruction cycle that means the high/low time durations of $TM0CKI$ must be longer than one instruction cycle time to guarantee each $TM0CKI$'s change will be detected correctly by the synchronizer. The following timing diagram describes the Timer0 works in Counter mode.



Timer0 works in Counter mode ($TM0CKS = 1$) for $TM0CKI$

◇Example: Setup Timer0 works in Counter mode

; Setup Timer0 clock source and divider

```
MOVLW    00110000B
MOVWR    TM0CTL
```

```
; TM0EDG = 1, counting edge is falling edge
; TM0CKS = 1, Timer0 clock is TM0CKI (PA2)
; TM0PSC = 0000b, divided by 1
```

; Setup Timer0

```
BSF      TM0STP
CLRFR    TM0
```

```
; Timer0 stops counting
; Clear Timer0 content
```

; Enable Timer0 and read Timer0 counter

```
BCF      TM0STP
...
BSF      TM0STP
MOVFR    TM0
```

```
; Enable Timer0 counting
; Timer0 stops counting
; Read Timer0 content
```

F01	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TM0	TM0							
R/W	R/W							
Reset	0	0	0	0	0	0	0	0

F01.7~0 **TM0**: Timer0 content

F08	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTIE	PWM0IE	T2IE	–	TM0IE	WKTIE	INT2IE	–	INT0IE
R/W	R/W	R/W	–	R/W	R/W	R/W	–	R/W
Reset	0	0	–	0	0	0	–	0

F08.4 **TM0IE**: Timer0 interrupt enable
 0: disable
 1: enable

F09	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTIF	PWM0IF	T2IF	–	TM0IF	WKTIF	INT2IF	–	INT0IF
R/W	R/W	R/W	–	R/W	R/W	R/W	–	R/W
Reset	0	0	–	0	0	0	–	0

F09.4 **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

F14	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
MF14	CPUCKS	FASTSTP	SLOWEN	–	HSNK	T2CLR	TM0STP	PWM0CLR
R/W	R/W	R/W	R/W	–	R/W	R/W	R/W	R/W
Reset	1/0	0	1	–	0	0	0	1

F14.1 **TM0STP**: Timer0 counter stop
 0: Timer0 is counting
 1: Timer0 stop counting

R02	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TM0CTL	–	–	TM0EDG	TM0CKS	TM0PSC			
R/W	–	–	W	W	W			
Reset	–	–	0	0	0	0	0	0

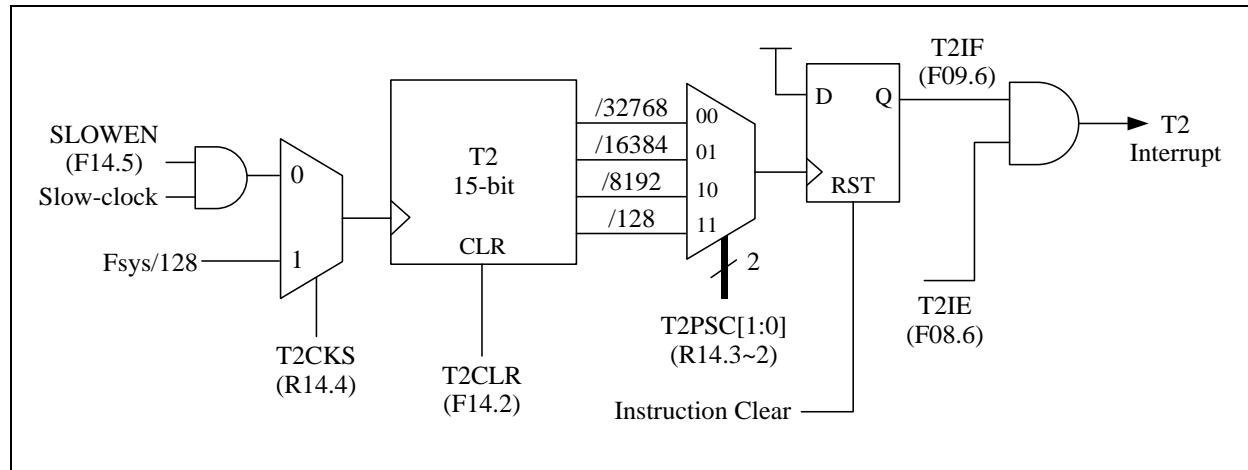
R02.5 **TM0EDG**: TM0CKI (PA2) edge selection for Timer0 Prescaler count
 0: TM0CKI (PA2) rising edge for Timer0 Prescaler count
 1: TM0CKI (PA2) falling edge for Timer0 Prescaler count

R02.4 **TM0CKS**: Timer0 Prescaler clock select
 0: Instruction Cycle as Timer0 Prescaler clock
 1: TM0CKI (PA2) as Timer0 Prescaler clock

R02.3~0 **TM0PSC**: Timer0 Prescale
 0000: divided by 1
 0001: divided by 2
 0010: divided by 4
 0011: divided by 8
 0100: divided by 16
 0101: divided by 32
 0110: divided by 64
 0111: divided by 128
 1xxx: divided by 256

3.3 T2: 15-bit Timer

The T2 is a 15-bit counter and the clock sources are from either $F_{sys}/128$ or Slow-clock. The clock source is used to generate time base interrupt and T2 counter block clock. It is selected by T2CKS (R14.4). The T2's 15-bit content cannot be read by instructions. It generates interrupt flag T2IF (F09.6) with the clock divided by 32768, 16384, 8192, or 128 depends on the T2PSC [1:0] (R14.3~2) bits. The following figure shows the block diagram of T2.



T2 Block Diagram

◇Example: CPU is running at FAST mode, F_{sys} = Fast-clock = FIRC 2 MHz,

T2 clock source is $F_{sys}/128$

; Setup FIRC frequency

```
MOVLW    00000001B
MOVWR    FIRCKS          ; FIRC is 2 MHz
```

; Setup T2 clock source and divider

```
MOVLW    000101xxB      ; T2CKS = 1, T2 clock source is Fsys/128
MOVWR    R14             ; T2PSC = 01b, divided by 16384
BSF      T2CLR           ; T2CLR = 1, clear T2 counter
```

; Enable T2 interrupt function

```
MOVLW    10111111B
MOVWF    INTIF           ; Clear T2 request interrupt flag
BSF      T2IE            ; Enable T2 interrupt function
```

T2 clock source is $F_{sys}/128 = 2 \text{ MHz} / 128 = 15625 \text{ Hz}$, T2 divided by 16384

T2 interrupt frequency = $15625 \text{ Hz} / 16384 = 0.95 \text{ Hz}$

T2 interrupt period = $1 / 0.95 \text{ Hz} = 1.05 \text{ s}$

◇Example: CPU is running at SLOW mode, Fsys = Slow-clock = SIRC, T2 clock source is SIRC

; Setup CPU runs at SLOW mode

```

MOV LW    00000000B
MOV W R0B           ; Slow-clock is 110K @3V
BSF SLOWEN          ; Enable Slow-clock
BSF CPUCKS          ; Switch system clock source to Slow-clock
BSF FASTSTP         ; Stop Fast-clock

```

; Setup T2 clock source and divider

```

MOV LW    00000000B           ; T2CKS = 0, T2 clock source is Slow-clock
MOV W R14           ; T2PSC = 00b, divided by 32768
BSF T2CLR           ; T2CLR = 1, clear T2 counter

```

; Enable T2 interrupt function

```

MOV LW    10111111B
MOV W INTIF           ; Clear T2 request interrupt flag
BSF T2IE           ; Enable T2 interrupt function

```

T2 clock source is Slow-clock = 110 KHz @3V, T2 divided by 32768

T2 interrupt frequency = 110K Hz / 32768 = 3.35 Hz

T2 interrupt period = 1 / 3.35 Hz = 298ms

F08	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTIE	PWM0IE	T2IE	—	TM0IE	WKTIE	INT2IE	—	INT0IE
R/W	R/W	R/W	—	R/W	R/W	R/W	—	R/W
Reset	0	0	—	0	0	0	—	0

F08.6 **T2IE**: T2 interrupt enable

0: disable

1: enable

F09	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTIF	PWM0IF	T2IF	—	TM0IF	WKTIF	INT2IF	—	INT0IF
R/W	R/W	R/W	—	R/W	R/W	R/W	—	R/W
Reset	0	0	—	0	0	0	—	0

F09.6 **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

F14	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
MF14	CPUCKS	FASTSTP	SLOWEN	–	HSNK	T2CLR	TM0STP	PWM0CLR
R/W	R/W	R/W	R/W	–	R/W	R/W	R/W	R/W
Reset	1/0	0	1	–	0	0	0	1

F14.5 **SLOWEN**: Slow-clock Enable / Disable

0: Slow-clock is disabled, except CPUCKS = 1

1: Slow-clock is enabled

F14.2 **T2CLR**: T2 counter clear

0: T2 is counting

1: T2 is cleared immediately, this bit is auto cleared by H/W

R0B	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
MR0B	–	–	–	INT0EDG	TCOE	–	WKTPSC	
R/W	–	–	–	W	W	–	W	
Reset	–	–	–	0	0	–	0	0

R0B.1~0 **WKTPSC**: WDT / WKT pre-scale option or SIRC frequency select

WDT / WKT pre-scale select @3V:

00: 18.75 ms

01: 37.5 ms

10: 75 ms

11: 150 ms

SIRC frequency select @3V:

00: 110 KHz

01: 27.5 KHz

10: 6.88 KHz

11: 1.72 KHz

R14	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
MR14	–	–	–	T2CKS	T2PSC		–	–
R/W	–	–	–	W	W		–	–
Reset	–	–	–	0	0		–	–

R14.4 **T2CKS**: T2 clock source selection

0: Slow-clock

1: Fsys/128

R14.3~2 **T2PSC**: T2 prescaler. T2 clock source

00: divided by 32768

01: divided by 16384

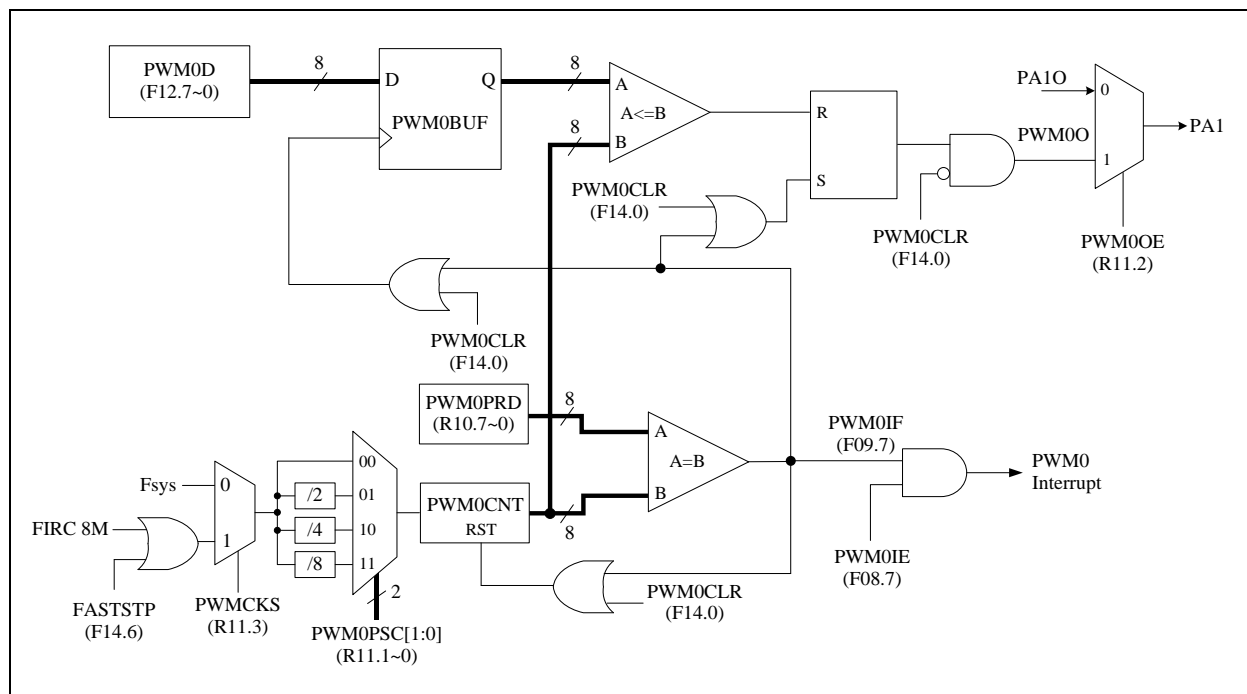
10: divided by 8192

11: divided by 128

3.4 PWM0: 8-bit PWM

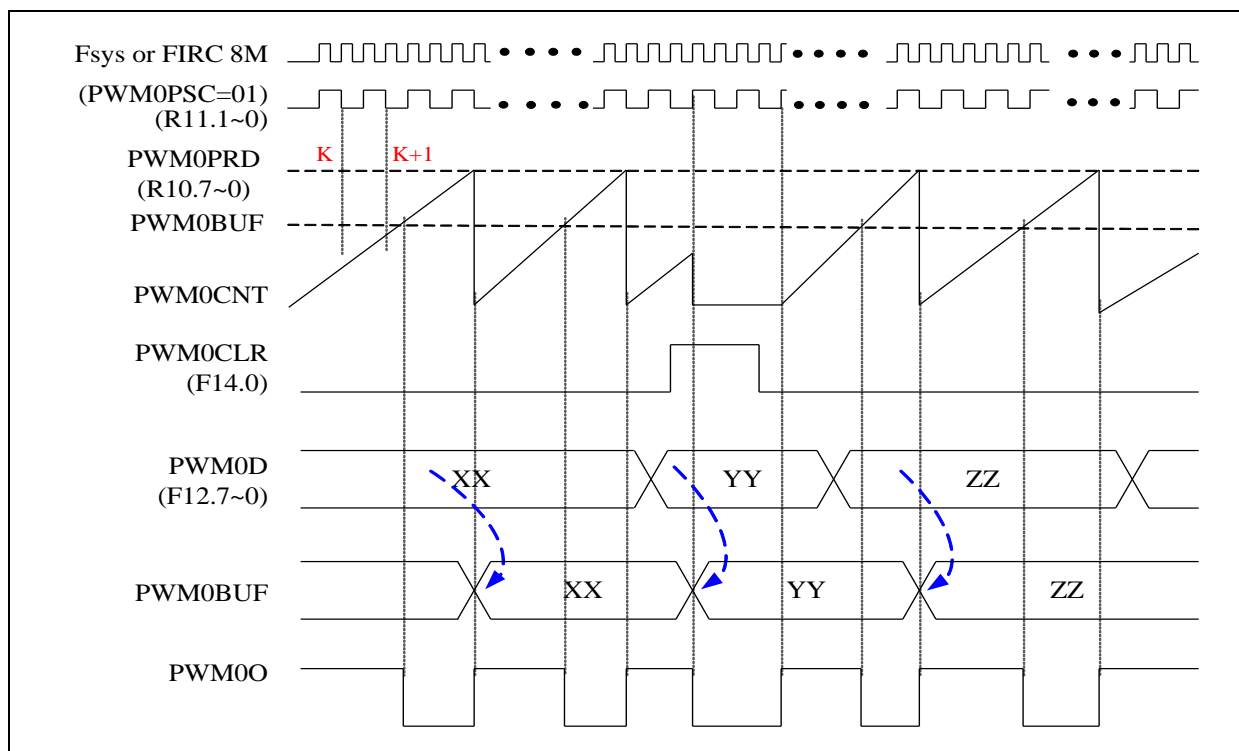
The chip has a built-in 8-bit PWM generator. The source clock comes from Fsys or FIRC 8 MHz divided by 1, 2, 4, and 8. If PWM source clock is select to FIRC 8 MHz, FASTSTP (F14.6) must be cleared. Otherwise FIRC 8 MHz will not oscillate at slow mode. The PWM0 duty cycle can be changed with writing to PWM0D (F12.7~0). Writing to PWM0D will not change the current PWM0 duty until the current PWM0 period completes. When finish current PWM0 period, the new value of PWM0D will be updated to the PWM0BUF.

The PWM0 will be output to PA1 if PWM0OE (R11.2) is set. With I/O mode setting, the PWM0 output can be set as CMOS push-pull or open-drain output mode. When PAE[1] (R05.1) is set, the output is CMOS push-pull output mode, otherwise is open-drain output mode. Also, the PWM0 period complete will generate an interrupt when PWM0IE (F08.7) is set. Setting the PWM0CLR (F14.0) bit will clear the PWM0 counter and load the PWM0D to PWM0BUF, PWM0CLR bit must be cleared so that the PWM0 counter can count. Figure shows the block diagram of PWM0.



PWM0 Block Diagram

Figure shows the PWM0 waveforms. When PWM0CLR (F14.0) bit is set or PWM0BUF equals to zero, the PWM0 output is cleared to '0' no matter what its current status is. Once the PWM0CLR bit is cleared and PWM0BUF is not zero, the PWM0 output is set to '1' to begin a new PWM cycle. PWM0 output will be '0' when PWM0CNT greater than or equals to PWM0BUF. PWM0CNT keeps counting up when equals to PWM0PRD (R10.7~0), the PWM0 output is set to '1' again.



PWM0 Timing Diagram

◇Example: CPU is running at FAST mode, Fsys = Fast-clock = FIRC 4 MHz

; Setup PWM0 prescaler, period, and duty

BSF	PWM0CLR	; PWM0CLR = 1, PWM0 clear and hold
MOVLW	0000 <u>0101</u> B	; PWM0OE = 1, PWM0 output to PA1 pin
MOVWR	PWM0CTL	; PWM0PSC = 01b, divided by 2 (Fsys/2)

MOVLW	FFH	
MOVWR	PWM0PRD	; Set PWM0 period = FFH + 1 = 256

MOVLW	80H	
MOVWF	PWM0D	; Set PWM0 duty = 80H = 128
BCF	PWM0CLR	; PWM0CLR = 0, PWM0 is running

; Enable PWM0 interrupt function

MOVLW	<u>0</u> 1111111B	
MOVWF	INTIF	; Clear PWM0 request interrupt flag
BSF	PWM0IE	; Enable PWM0 interrupt function

PWM0 output duty = PWM0D / (PWM0PRD + 1) = 128 / (255 + 1) = 1 / 2

Fsys = 4 MHz, PWM0 divided by 2

PWM0 output/interrupt frequency = 4 MHz / 2 / (255 + 1) = 7812.5 Hz

F08	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTIE	PWM0IE	T2IE	–	TM0IE	WKTIE	INT2IE	–	INT0IE
R/W	R/W	R/W	–	R/W	R/W	R/W	–	R/W
Reset	0	0	–	0	0	0	–	0

F08.7 **PWM0IE**: PWM0 interrupt enable
 0: disable
 1: enable

F09	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTIF	PWM0IF	T2IF	–	TM0IF	WKTIF	INT2IF	–	INT0IF
R/W	R/W	R/W	–	R/W	R/W	R/W	–	R/W
Reset	0	0	–	0	0	0	–	0

F09.7 **PWM0IF**: PWM0 interrupt event pending flag
 This bit is set by H/W while PWM0 overflows, write 0 to this bit will clear this flag

F12	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWM0D	PWM0D							
R/W	R/W							
Reset	0	0	0	0	0	0	0	0

F12.7~0 **PWM0D**: PWM0 duty

F14	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
MF14	CPUCKS	FASTSTP	SLOWEN	–	HSNK	T2CLR	TM0STP	PWM0CLR
R/W	R/W	R/W	R/W	–	R/W	R/W	R/W	R/W
Reset	1/0	0	1	–	0	0	0	1

F14.6 **FASTSTP**: Fast-clock & FIRC 8M Enable / Disable
 0: Fast-clock & FIRC 8M enable
 1: Fast-clock & FIRC 8M disable

F14.0 **PWM0CLR**: PWM0 clear and hold
 0: PWM0 is running
 1: PWM0 is cleared and hold

R10	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWM0PRD	PWM0PRD							
R/W	W							
Reset	1	1	1	1	1	1	1	1

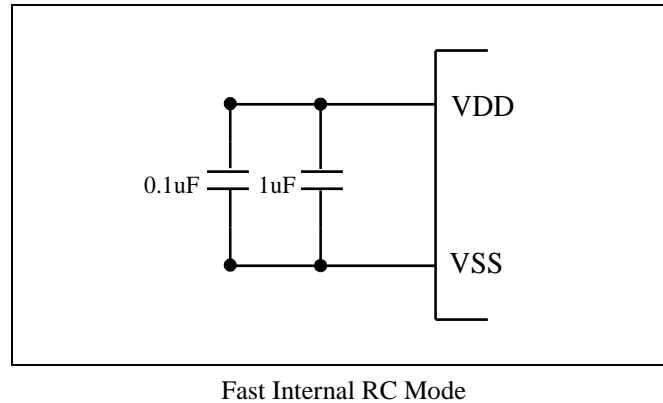
R10.7~0 **PWM0PRD**: PWM0 period data

R11	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWM0CTL	—	—	—	—	PWMCKS	PWM0OE	PWM0PSC	
R/W	—	—	—	—	W	W	W	
Reset	—	—	—	—	0	0	0	0

- R11.3 **PWMCKS: PWM Clock source select**
 0: Fsys
 1: FIRC 8 MHz
- R11.2 **PWM0OE: PWM0 positive output to PA1 pin**
 0: disable
 1: enable
- R11.1~0 **PWM0PSC: PWM0 prescaler, PWM0 clock select**
 00: Fsys divided by 1
 01: Fsys divided by 2
 10: Fsys divided by 4
 11: Fsys divided by 8

3.5 System Clock Oscillator

System clock can be operated in two different oscillation modes. Two oscillation modes are FIRC and SIRC respectively. In the Fast Internal RC mode (FIRC), the on-chip oscillator generates 4/2/1 MHz system clock. Since power noise degrades the performance of Fast Internal Clock Oscillator, placing power supply bypass capacitors 1 μ F and 0.1 μ F very close to VDD / VSS pins to improve the stability of clock and the overall system. In the Slow Internal RC mode (SIRC), it provides a lower speed and accuracy of the oscillator for power saving purpose.



3.6 FIRC and SIRC Clock Frequency Selection

The FIRC frequency is selected by FIRCKS (R19.7~6), while the SIRC frequency is selected by WKTPSC (R0B.1~0). Consider the Fsys safety and integrity, it is recommended to change FIRCKS in SLOW Mode and change WKTPSC in FAST Mode.

R0B	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
MR0B	—	—	—	INT0EDG	TCOE	—	WKTPSC	
R/W	—	—	—	W	W	—	W	
Reset	—	—	—	0	0	—	0	0

R0B.1~0 **WKTPSC:** WDT / WKT pre-scale option or SIRC frequency select

WDT / WKT pre-scale select @ 3V:

00: 18.75 ms

01: 37.5 ms

10: 75 ms

11: 150 ms

SIRC frequency select @ 3V:

00: 110 KHz

01: 27.5 KHz

10: 6.88 KHz

11: 1.72 KHz

R19	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
FIRCKS	FIRCKS		—	—	—	—	—	—
R/W	W		—	—	—	—	—	—
Reset	1	0	—	—	—	—	—	—

R19.7~6 **FIRCKS:** FIRC clock source selection

00: 1 MHz

01: 2 MHz

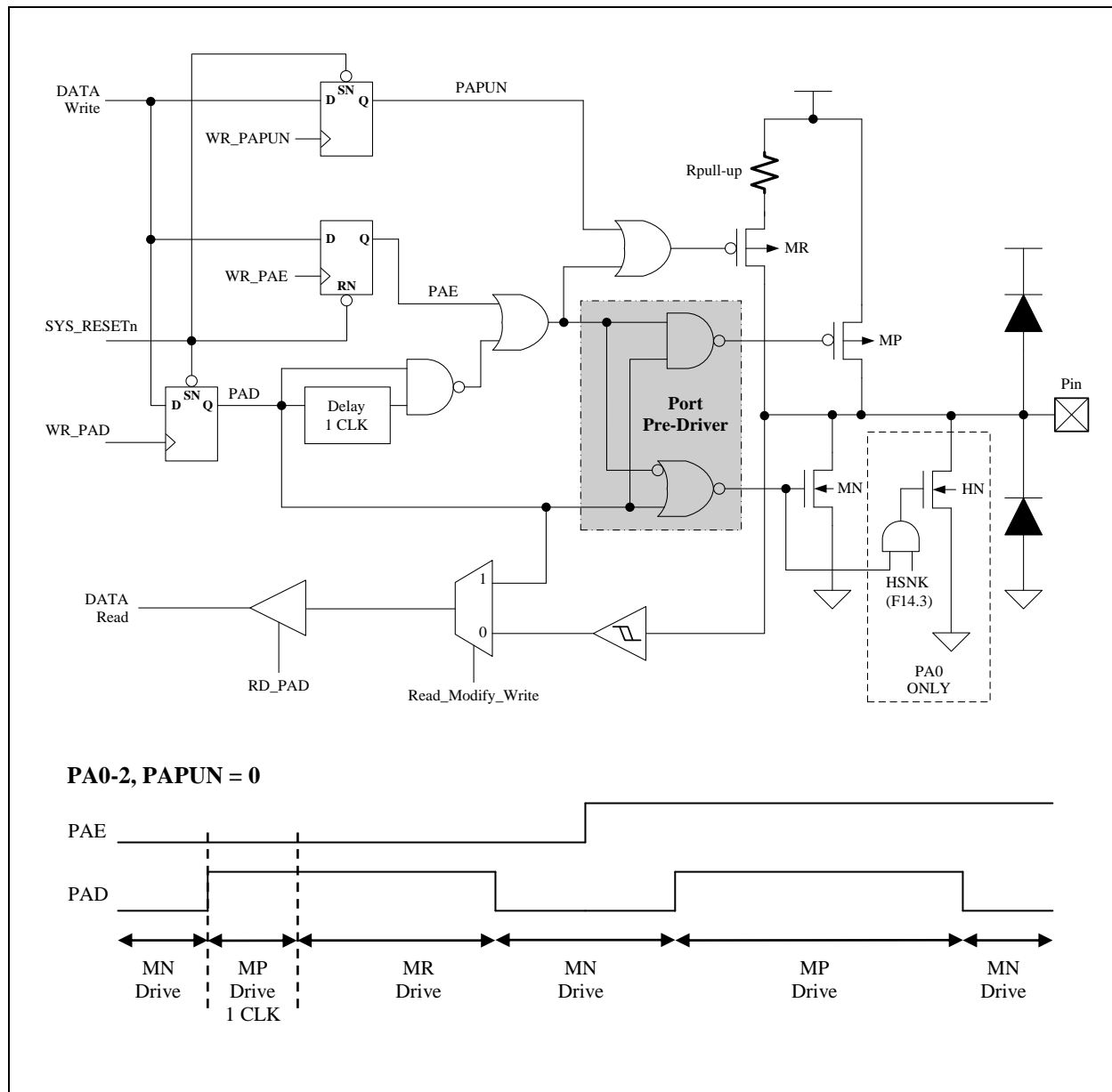
10: 4 or 2 MHz (4 MHz if SYSCFG [5] = 0, 2 MHz if SYSCFG [5] = 1)

11: 4 MHz

4. I/O Port

4.1 PA0-2

These pins can be used as Schmitt-trigger input, CMOS push-pull output or “pseudo-open-drain” output. PA0 have got extra High Sink output. When HSNK (F14.3) is cleared, PA0 is the same with the other PA1-2. When HSNK is set, PA0 is High Sink output enable. 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 PAE = 0 and PAD = 1. To use the pin in pseudo-open-drain mode, S/W sets the PAE = 0. The benefit of pseudo-open-drain structure is that the output rise time can be much faster than pure open-drain structure. S/W sets PAE = 1 to use the pin in CMOS push-pull output mode. Reading the pin data (PAD) has different meaning. In “Read-Modify-Write” instruction, CPU actually reads the output data register. In the other instructions, CPU reads the pin state. The so-called “Read-Modify-Write” instruction includes BSF, BCF and all instructions using F-Plane as destination.

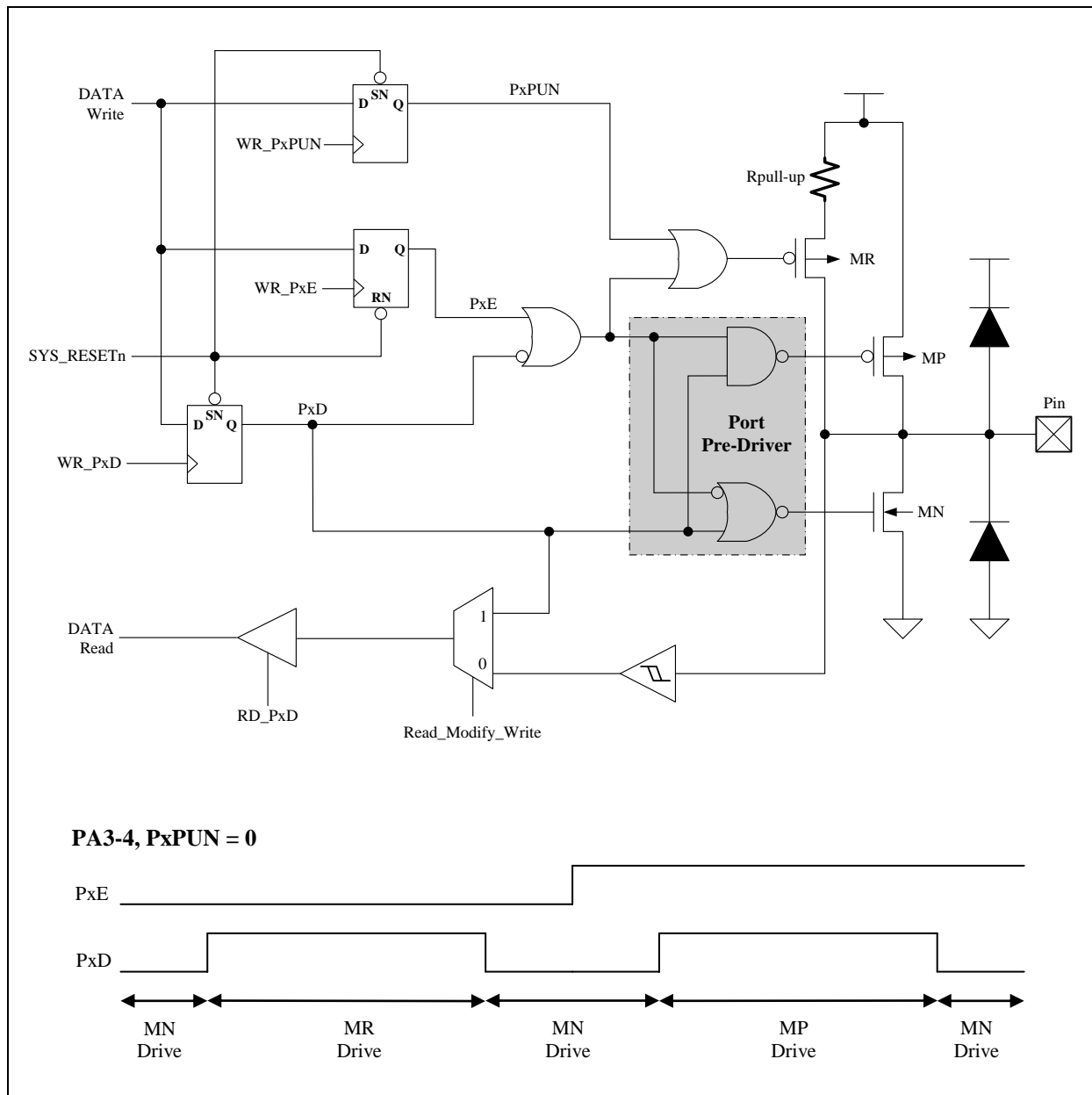


How to control PA0-2 status can be concluded as following list.

PAE0-2	PAD0-2	PAPUN0-2	PIN STATE	Pull-up	Mode
0	0	X	Low	No	pseudo-open-drain output
0	1	0	High	YES	pseudo-open-drain output or input with pull-high
0	1	1	Hi-Z	No	pseudo-open-drain output or input without pull-high
1	0	X	Low	No	CMOS push-pull output
1	1	X	High	No	CMOS push-pull output

4.2 PA3-4

These pins are almost the same as PA0-2, except they do not support pseudo-open-drain mode. They can be used in pure open-drain mode, instead.

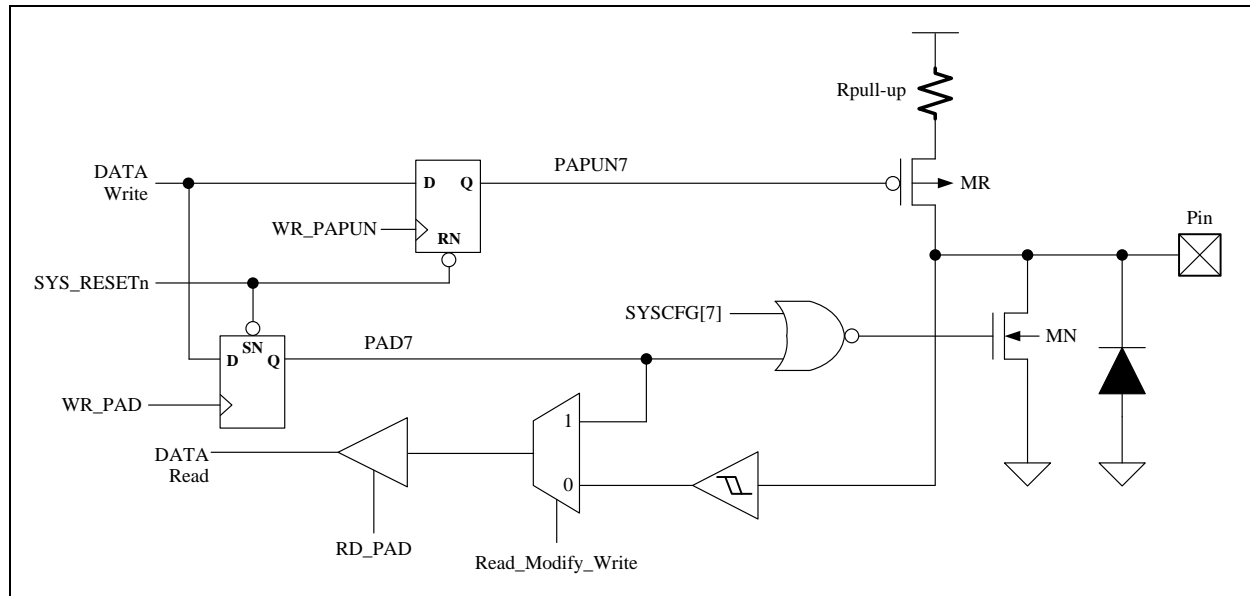


How to control PA3-4 status can be concluded as following list.

PAE3-4	PAD3-4	PAPUN3-4	PIN STATE	Pull-up	Mode
0	0	X	Low	No	open-drain output
0	1	0	High	Yes	open-drain output or input with pull-high
0	1	1	Hi-Z	No	open-drain output or input without pull-high
1	0	X	Low	No	CMOS push-pull output
1	1	X	High	No	CMOS push-pull output

4.3 PA7

PA7 can be used in Schmitt-trigger input or open-drain output which is set by the PAD [7] (F05.7) bit. When the PAD [7] bit is set, PA7 is assigned as Schmitt-trigger input mode, otherwise is assigned as open-drain output mode and output low. The pull-up resistor connected to this pin default, and can be disabled by S/W. In open-drain output mode, the pull-up resistor will not be disabled automatically. The pull-up resistor can be disabled by S/W in open-drain output mode for power saving.



How to control PA7 status can be concluded as following list.

SYSCFG [7]	PAD7	PAPUN7	PN STATE	Pull-up	MODE
0	0	0	Low	Yes	open-drain output with pull-high (not suggested to use this mode)
0	0	1	Low	No	open-drain output without pull-high
0	1	0	High	Yes	input with pull-high
0	1	1	Hi-Z	No	input without pull-high
1	X	0	High	Yes	reset input with pull-high
1	X	1	Hi-Z	No	reset input without pull-high

F05	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PAD	PAD7	—						
R/W	R/W	—						
Reset	1	—	—	1	1	1	1	1

- F05.7 **PAD7:** PA7 data or pin mode control
 0: PA7 is open-drain output mode and output low
 1: PA7 is Schmitt-trigger input mode
- F05.4~0 **PAD:** PA4~PA0 data
 0: output low
 1: output high or Schmitt-trigger input mode

R05	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PAE	—	—	—	PAE				
R/W	—	—	—	W				
Reset	—	—	—	0	0	0	0	0

- R05.4~3 **PAE:** PA4~PA3 pin mode control
0: the pin is open-drain output or Schmitt-trigger input
1: the pin is CMOS push-pull output
- R05.2~0 **PAE:** PA2~PA0 pin mode control
0: the pin is pseudo-open-drain output or Schmitt-trigger input
1: the pin is CMOS push-pull output

R08	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PAPUN	PAPUN7	—	—	PAPUN				
R/W	W	—	—	W				
Reset	0	—	—	1	1	1	1	1

- R08.7 **PAPUN7:** PA7 pull-up resistor enable
0: the pin pull-up resistor is enabled
1: the pin pull-up resistor is disabled
- R08.4~0 **PAPUN:** PA4~PA0 pull-up resistor enable
0: the pin pull-up resistor is enabled, except
a: the pin's output data register (PAD) is 0
b: the pin's CMOS push-pull mode is chosen (PAE = 1)
1: the pin pull-up resistor is disabled

R13	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PAWKEN	—	—	—	PAWKEN				—
R/W	—	—	—	W				—
Reset	—	—	—	0	0	0	0	—

- R13.4~1 **PAWKEN:** PA4~PA1 individual pin low level wake up control
0: disable
1: enable

F14	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
MF14	CPUCKS	FASTSTP	SLOWEN	—	HSNK	T2CLR	TM0STP	PWM0CLR
R/W	R/W	R/W	R/W	—	R/W	R/W	R/W	R/W
Reset	1/0	0	1	—	0	0	0	1

- F14.3 **HSNK:** PA0 output PIN High-Sink
0: disable
1: enable

MEMORY MAP

F-Plane

Name	Address	R/W	Rst	Description
(F00) INDF		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
(F01) TM0		Function related to: Timer0		
TM0	01.7~0	R/W	0	Timer0 content
(F02) PCL		Function related to : PROGRAM COUNT		
PCL	02.7~0	R/W	0	Programming Counter LSB [7~0]
(F03) STATUS		Function related to: STATUS		
-	03.7	-	-	Reserved
GB0	03.6	R/W	0	General purpose bit 0
-	03.5	-	-	Reserved
TO	03.4	R	0	WDT timeout flag
PD	03.3	R	0	Power-down mode flag
Z	03.2	R/W	0	Zero flag
DC	03.1	R/W	0	Decimal Carry flag or Decimal /Borrow flag
C	03.0	R/W	0	Carry flag or /Borrow flag
(F04) FSR		Function related to: RAM W/R / Table Read		
DPL	04.7~0	R/W	-	Table read low address, data ROM pointer (DPTR) low byte
FSR	04.6~0	R/W	-	File Select Register, indirect address mode pointer
(F05) PAD		Function related to: Port A		
PAD7	05.7	R	-	PA7 pin or “data register” state
		W	1	0: PA7 is open-drain output mode 1: PA7 is Schmitt-trigger input mode
PAD	05.4~0	R	-	Port A pin or “data register” state
		W	1F	Port A output data register
(F08) INTIE		Function related to: Interrupt Enable		
PWM0IE	08.7	R/W	0	PWM0 interrupt enable 0: disable 1: enable
T2IE	08.6	R/W	0	T2 interrupt enable 0: disable 1: enable
-	08.5	-	-	Reserved
TM0IE	08.4	R/W	0	Timer0 interrupt enable 0: disable 1: enable
WKTIE	08.3	R/W	0	Wakeup Timer interrupt enable 0: disable 1: enable
INT2IE	08.2	R/W	0	INT2 (PA7) pin interrupt enable 0: disable 1: enable
-	08.1	-	-	Reserved
INT0IE	08.0	R/W	0	INT0 (PA0) pin interrupt enable 0: disable 1: enable

Name	Address	R/W	Rst	Description
(F09) INTIF		Function related to: Interrupt Flag		
PWM0IF	09.7	R	-	PWM0 interrupt event pending flag, set by H/W while PWM0 overflows
		W	0	0: clear this flag 1: no action
T2IF	09.6	R	-	T2 interrupt event pending flag, set by H/W while T2 overflows
		W	0	0: clear this flag 1: no action
-	09.5	-	-	Reserved
TM0IF	09.4	R	-	Timer0 interrupt event pending flag, set by H/W while Timer0 overflows
		W	0	0: clear this flag 1: no action
WKTIF	09.3	R	-	WKT interrupt event pending flag, set by H/W while WKT time out
		W	0	0: clear this flag 1: no action
INT2IF	09.2	R	-	INT2 interrupt event pending flag, set by H/W at INT2 pin's falling edge
		W	0	0: clear this flag 1: no action
-	09.1	-	-	Reserved
INT0IF	09.0	R	-	INT0 interrupt event pending flag, set by H/W at INT0 pin's falling / rising edge
		W	0	0: clear this flag 1: no action
(F12) PWM0D		Function related to: PWM0		
PWM0D	12.7~0	R/W	0	PWM0 duty
(F14) MF14		Function related to: CPUCLK / T2 / TM0 / PWM0		
CPUCKS	14.7	R/W	1/0	System clock (Fsys) selection, the reset value depends on SYSCFG [8]. 0: Fast-clock 1: Slow-clock If CLKS (SYSCFG [8]) = 1, the reset value is 1, otherwise is 0
FASTSTP	14.6	R/W	0	Fast-clock & FIRC 8M Enable / Disable 0: Fast-clock & FIRC 8M enable 1: Fast-clock & FIRC 8M disable
SLOWEN	14.5	R/W	1	Slow-clock Enable / Disable 0: Slow-clock is disabled, except CPUCKS = 1 1: Slow-clock is enabled
-	14.4	-	-	Reserved
HSNK	14.3	R/W	0	PA0 output PIN High-Sink 0: disable 1: enable
T2CLR	14.2	R/W	0	T2 counter clear 0: T2 is counting 1: T2 is cleared immediately, this bit is auto cleared by H/W
TM0STP	14.1	R/W	0	Timer0 counter stop 0: Timer0 is counting 1: Timer0 stops counting
PWM0CLR	14.0	R/W	1	PWM0 clear and hold 0: PWM0 is running 1: PWM0 is cleared and hold
(F17) DPH		Function related to: Table Read		
DPH	17.1~0	R/W	0	Table read high address, data ROM pointer (DPTR) high byte
User Data Memory				
SRAM	20~4F	R/W	-	Internal RAM

R-Plane

Name	Address	R/W	Rst	Description
(R02) TM0CTL				Function related to: Timer0
TM0EDG	02.5	W	0	TM0CKI (PA2) edge selection for Timer0 Prescaler count 0: TM0CKI (PA2) rising edge for Timer0 Prescaler count 1: TM0CKI (PA2) falling edge for Timer0 Prescaler count
TM0CKS	02.4	W	0	Timer0 Prescaler clock select 0: Instruction Cycle (Fsys/2) as Timer0 Prescaler clock 1: TM0CKI (PA2) as Timer0 Prescaler clock
TM0PSC	02.3~0	W	0	Timer0 Prescale 0000: divided by 1 0001: divided by 2 0010: divided by 4 0011: divided by 8 0100: divided by 16 0101: divided by 32 0110: divided by 64 0111: divided by 128 1xxx: divided by 256
(R03) PWRDN				Function related to: POWER DOWN
PWRDN	03	W	-	Write this register to enter Power-down (STOP / IDLE) Mode
(R04) WDTCLR				Function related to: WDT
WDTCLR	04	W	-	Write this register to clear WDT/WKT timer
(R05) PAE				Function related to: Port A
PAE	05.4~3	W	0	PA4~PA3 I/O mode control Each bit controls its corresponding pin, if the bit is 0: the pin is open-drain output or Schmitt-trigger input 1: the pin is CMOS push-pull output
	05.2~0	W	0	PA2~PA0 I/O mode control Each bit controls its corresponding pin, if the bit is 0: the pin is pseudo-open-drain output or Schmitt-trigger input 1: the pin is CMOS push-pull output
(R08) PAPUN				Function related to: Port A
PAPUN7	08.7	W	0	PA7 pull-up control, if the bit is 0: the pin pull-up resistor is enabled 1: the pin pull-up resistor is disabled
PAPUN	08.4~0	W	7F	PA4~PA0 pull-up control Each bit controls its corresponding pin, if the bit is 0: the pin pull-up resistor is enabled, except a. the pin's output data register (PAD) is 0 b. the pin's CMOS push-pull mode is chosen (PAE = 1) 1: the pin pull-up resistor is disabled

Name	Address	R/W	Rst	Description
(R0B) MR0B		Function related to: INT0 / TCOU / WKT		
INT0EDG	0b.4	W	0	INT0 pin (PA0) edge interrupt event 0: falling edge to trigger 1: rising edge to trigger
TCOE	0b.3	W	0	Enable Instruction Cycle (Fsys/2) output to PA3 pin (TCOUT) 0: disable 1: enable
-	0b.2	-	-	Reserved
WKTOSC	0b.1~0	W	00	WDT / WKT pre-scale option or SIRC frequency selections WDT/WKT pre-scale select @3V 00: 18.75 ms 01: 37.5 ms 10: 75 ms 11: 150 ms SIRC frequency select @3V: 00: 110 KHz 01: 27.5 KHz 10: 6.88 KHz 11: 1.72 KHz
(R10) PMW0PRD		Function related to: PWM0		
PMW0PRD	10.7~0	W	FF	PWM0 period data
(R11) PWM0CTL		Function related to: PWM0		
PWMCKS	11.3	W	0	PWM Clock source select, 0: Fsys 1: FIRC 8 MHz
PWM0OE	11.2	W	0	PWM0 positive output to PA1 pin 0: disable 1: enable
PWM0PSC	11.1~0	W	0	PWM0 prescaler, PWM0 clock select 00: Fsys divided by 1 01: Fsys divided by 2 10: Fsys divided by 4 11: Fsys divided by 8

Name	Address	R/W	Rst	Description
(R13) PAWKEN		Function related to: Port A / WAKE UP		
PAWKEN	13.4~1	W	0	PA4~PA1 individual pin low level wake up control Each bit controls its corresponding pin, if the bit is 0: disable 1: enable
(R14) MR14		Function related to: T2 / CPUCLK		
T2CKS	14.4	W	0	T2 clock source selection 0: Slow-clock 1: Fsys/128
T2PSC	14.3~2	W	0	T2 prescaler. T2 clock source 00: divided by 32768 01: divided by 16384 10: divided by 8192 11: divided by 128
(R19) FIRCKS		Function related to: CPUCLK		
FIRCKS	19.7~6	W	10	FIRC clock source selection 00: 1 MHz 01: 2 MHz 10: 4 or 2 MHz (4 MHz if SYSCFG [5] = 0, 2 MHz if SYSCFG [5] = 1) 11: 4 MHz

INSTRUCTION SET

Each instruction is a 14-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” or “r” 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	F-Plane Register File Address
r	R-Plane 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					
<u>ADDWF</u>	f,d	00 0111 dfff ffff	1	C, DC, Z	Add W and "f"
<u>ANDWF</u>	f,d	00 0101 dfff ffff	1	Z	AND W with "f"
<u>CLRF</u>	f	00 0001 1fff ffff	1	Z	Clear "f"
<u>CLRWF</u>		00 0001 0100 0000	1	Z	Clear W
<u>COMF</u>	f,d	00 1001 dfff ffff	1	Z	Complement "f"
<u>DECF</u>	f,d	00 0011 dfff ffff	1	Z	Decrement "f"
<u>DECFSZ</u>	f,d	00 1011 dfff ffff	1 or 2	-	Decrement "f", skip if zero
<u>INCF</u>	f,d	00 1010 dfff ffff	1	Z	Increment "f"
<u>INCFSZ</u>	f,d	00 1111 dfff ffff	1 or 2	-	Increment "f", skip if zero
<u>IORWF</u>	f,d	00 0100 dfff ffff	1	Z	OR W with "f"
<u>MOVFW</u>	f	00 1000 0fff ffff	1	-	Move "f" to W
<u>MOVWF</u>	f	00 0000 1fff ffff	1	-	Move W to "f"
<u>MOVWR</u>	r	00 0000 00rr rrrr	1	-	Move W to "r"
<u>RLF</u>	f,d	00 1101 dfff ffff	1	C	Rotate left "f" through carry
<u>RRF</u>	f,d	00 1100 dfff ffff	1	C	Rotate right "f" through carry
<u>SUBWF</u>	f,d	00 0010 dfff ffff	1	C, DC, Z	Subtract W from "f"
<u>SWAPF</u>	f,d	00 1110 dfff ffff	1	-	Swap nibbles in "f"
<u>TESTZ</u>	f	00 1000 1fff ffff	1	Z	Test if "f" is zero
<u>XORWF</u>	f,d	00 0110 dfff ffff	1	Z	XOR W with "f"
Bit-Oriented File Register Instruction					
<u>BCF</u>	f,b	01 000b bbff ffff	1	-	Clear "b" bit of "f"
<u>BSF</u>	f,b	01 001b bbff ffff	1	-	Set "b" bit of "f"
<u>BTFSC</u>	f,b	01 010b bbff ffff	1 or 2	-	Test "b" bit of "f", skip if clear
<u>BTFSS</u>	f,b	01 011b bbff ffff	1 or 2	-	Test "b" bit of "f", skip if set
Literal and Control Instruction					
<u>ADDLW</u>	k	01 1100 kkkk kkkk	1	C, DC, Z	Add Literal "k" and W
<u>SUBLW</u>	k	01 1101 kkkk kkkk	1	C, DC, Z	Subtract W from Literal "k"
<u>ANDLW</u>	k	01 1011 kkkk kkkk	1	Z	AND Literal "k" with W
<u>CALL</u>	k	10 kkkk kkkk kkkk	2	-	Call subroutine "k"
<u>CLRWDI</u>		00 0000 0000 0100	1	TO, PD	Clear Watch Dog Timer
<u>GOTO</u>	k	11 kkkk kkkk kkkk	2	-	Jump to branch "k"
<u>IORLW</u>	k	01 1010 kkkk kkkk	1	Z	OR Literal "k" with W
<u>MOVLW</u>	k	01 1001 kkkk kkkk	1	-	Move Literal "k" to W
<u>NOP</u>		00 0000 0000 0000	1	-	No operation
<u>RET</u>		00 0000 0100 0000	2	-	Return from subroutine
<u>RETI</u>		00 0000 0110 0000	2	-	Return from interrupt
<u>RETLW</u>	k	01 1000 kkkk kkkk	2	-	Return with Literal in W
<u>SLEEP</u>		00 0000 0000 0011	1	TO, PD	Go into Power-down mode, Clock oscillation stops
<u>TABRH</u>		00 0000 0101 1000	2	-	Lookup ROM high data to W
<u>TABRL</u>		00 0000 0101 0000	2	-	Lookup ROM low data to W
<u>XORLW</u>	k	01 1111 kkkk kkkk	1	Z	XOR Literal "k" with W



ADDLW	Add Literal "k" and W
ADDLW 0x00000000	W ← W + 0x00000000
ADDLW 0x00000001	W ← W + 0x00000001
ADDLW 0x00000002	W ← W + 0x00000002
ADDLW 0x00000003	W ← W + 0x00000003
ADDLW 0x00000004	W ← W + 0x00000004
ADDLW 0x00000005	W ← W + 0x00000005
ADDLW 0x00000006	W ← W + 0x00000006
ADDLW 0x00000007	W ← W + 0x00000007
ADDLW 0x00000008	W ← W + 0x00000008
ADDLW 0x00000009	W ← W + 0x00000009
ADDLW 0x0000000A	W ← W + 0x0000000A
ADDLW 0x0000000B	W ← W + 0x0000000B
ADDLW 0x0000000C	W ← W + 0x0000000C
ADDLW 0x0000000D	W ← W + 0x0000000D
ADDLW 0x0000000E	W ← W + 0x0000000E
ADDLW 0x0000000F	W ← W + 0x0000000F
ADDLW 0x00000010	W ← W + 0x00000010
ADDLW 0x00000011	W ← W + 0x00000011
ADDLW 0x00000012	W ← W + 0x00000012
ADDLW 0x00000013	W ← W + 0x00000013
ADDLW 0x00000014	W ← W + 0x00000014
ADDLW 0x00000015	W ← W + 0x00000015
ADDLW 0x00000016	W ← W + 0x00000016
ADDLW 0x00000017	W ← W + 0x00000017
ADDLW 0x00000018	W ← W + 0x00000018
ADDLW 0x00000019	W ← W + 0x00000019
ADDLW 0x0000001A	W ← W + 0x0000001A
ADDLW 0x0000001B	W ← W + 0x0000001B
ADDLW 0x0000001C	W ← W + 0x0000001C
ADDLW 0x0000001D	W ← W + 0x0000001D
ADDLW 0x0000001E	W ← W + 0x0000001E
ADDLW 0x0000001F	W ← W + 0x0000001F
ADDLW 0x00000020	W ← W + 0x00000020
ADDLW 0x00000021	W ← W + 0x00000021
ADDLW 0x00000022	W ← W + 0x00000022
ADDLW 0x00000023	W ← W + 0x00000023
ADDLW 0x00000024	W ← W + 0x00000024
ADDLW 0x00000025	W ← W + 0x00000025
ADDLW 0x00000026	W ← W + 0x00000026
ADDLW 0x00000027	W ← W + 0x00000027
ADDLW 0x00000028	W ← W + 0x00000028
ADDLW 0x00000029	W ← W + 0x00000029
ADDLW 0x0000002A	W ← W + 0x0000002A
ADDLW 0x0000002B	W ← W + 0x0000002B
ADDLW 0x0000002C	W ← W + 0x0000002C
ADDLW 0x0000002D	W ← W + 0x0000002D
ADDLW 0x0000002E	W ← W + 0x0000002E
ADDLW 0x0000002F	W ← W + 0x0000002F
ADDLW 0x00000030	W ← W + 0x00000030
ADDLW 0x00000031	W ← W + 0x00000031
ADDLW 0x00000032	W ← W + 0x00000032
ADDLW 0x00000033	W ← W + 0x00000033
ADDLW 0x00000034	W ← W + 0x00000034
ADDLW 0x00000035	W ← W + 0x00000035
ADDLW 0x00000036	W ← W + 0x00000036
ADDLW 0x00000037	W ← W + 0x00000037
ADDLW 0x00000038	W ← W + 0x00000038
ADDLW 0x00000039	W ← W + 0x00000039
ADDLW 0x0000003A	W ← W + 0x0000003A
ADDLW 0x0000003B	W ← W + 0x0000003B
ADDLW 0x0000003C	W ← W + 0x0000003C
ADDLW 0x0000003D	W ← W + 0x0000003D
ADDLW 0x0000003E	W ← W + 0x0000003E
ADDLW 0x0000003F	W ← W + 0x0000003F
ADDLW 0x00000040	W ← W + 0x00000040
ADDLW 0x00000041	W ← W + 0x00000041
ADDLW 0x00000042	W ← W + 0x00000042
ADDLW 0x00000043	W ← W + 0x00000043
ADDLW 0x00000044	W ← W + 0x00000044
ADDLW 0x00000045	W ← W + 0x00000045
ADDLW 0x00000046	W ← W + 0x00000046
ADDLW 0x00000047	W ← W + 0x00000047
ADDLW 0x00000048	W ← W + 0x00000048
ADDLW 0x00000049	W ← W + 0x00000049
ADDLW 0x0000004A	W ← W + 0x0000004A
ADDLW 0x0000004B	W ← W + 0x0000004B
ADDLW 0x0000004C	W ← W + 0x0000004C
ADDLW 0x0000004D	W ← W + 0x0000004D
ADDLW 0x0000004E	W ← W + 0x0000004E
ADDLW 0x0000004F	W ← W + 0x0000004F
ADDLW 0x00000050	W ← W + 0x00000050
ADDLW 0x00000051	W ← W + 0x00000051
ADDLW 0x00000052	W ← W + 0x00000052
ADDLW 0x00000053	W ← W + 0x00000053
ADDLW 0x00000054</	

Syntax	ADDLW k		
Operands	k : 00h ~ FFh		
Operation	$(W) \leftarrow (W) + k$		
Status Affected	C, DC, Z		
OP-Code	01 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	

ADDWF **Add W and "f"**

Syntax	ADDWF f [,d]
Operands	f : 00h ~ 7Fh, d : 0, 1
Operation	(destination) \leftarrow (W) + (f)
Status Affected	C, DC, Z
OP-Code	00 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	ADDWF 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) \leftarrow (W) AND k		
Status Affected	Z		
OP-Code	01 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	

ANDWF **AND W with "f"**

Syntax	ANDWF f[,d]
Operands	f : 00h ~ 7Fh, d : 0, 1
Operation	(destination) \leftarrow (W) AND (f)
Status Affected	Z
OP-Code	00 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	ANDWF FSR, 1 B : W = 0x17, FSR = 0xC2 A : W = 0x17, FSR = 0x02

BCF Clear "b" bit of "f"

Syntax	BCF f [,b]	
Operands	f : 00h ~ 3Fh, b : 0 ~ 7	
Operation	(f.b) ← 0	
Status Affected	-	
OP-Code	01 000b bbff ffff	
Description	Bit 'b' in register 'f' is cleared.	
Cycle	1	
Example	BCF FLAG_REG, 7	B : FLAG_REG = 0xC7 A : FLAG_REG = 0x47

BSF Set "b" bit of "f"

Syntax	BSF f [,b]	
Operands	f : 00h ~ 3Fh, b : 0 ~ 7	
Operation	(f.b) ← 1	
Status Affected	-	
OP-Code	01 001b bbff ffff	
Description	Bit 'b' in register 'f' is set.	
Cycle	1	
Example	BSF FLAG_REG, 7	B : FLAG_REG = 0x0A A : FLAG_REG = 0x8A

BTFSC Test "b" bit of "f", skip if clear(0)

Syntax	BTFSC f [,b]	
Operands	f : 00h ~ 3Fh, b : 0 ~ 7	
Operation	Skip next instruction if (f.b) = 0	
Status Affected	-	
OP-Code	01 010b bbff 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 BTFSC FLAG, 1 TRUE GOTO SUB1 FALSE ...	B : PC = LABEL1 A : if FLAG.1 = 0, PC = FALSE if FLAG.1 = 1, PC = TRUE

BTFSS Test "b" bit of "f", skip if set(1)

Syntax	BTFSS f [,b]	
Operands	f : 00h ~ 3Fh, b : 0 ~ 7	
Operation	Skip next instruction if (f.b) = 1	
Status Affected	-	
OP-Code	01 011b bbff 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 BTFSS FLAG, 1 TRUE GOTO SUB1 FALSE ...	B : PC = LABEL1 A : if FLAG.1 = 0, PC = TRUE if FLAG.1 = 1, PC = FALSE

CALL Call subroutine "k"

Syntax	CALL k
Operands	k : 000h ~ FFFh
Operation	Operation: TOS \leftarrow (PC) + 1, PC.11~0 \leftarrow k
Status Affected	-
OP-Code	10 kkkk kkkk kkkk
Description	Call Subroutine. First, return address (PC+1) is pushed onto the stack. The 12-bit immediate address is loaded into PC bits <11:0>. CALL is a two-cycle instruction.
Cycle	2
Example	LABEL1 CALL SUB1 B : PC = LABEL1 A : PC = SUB1, TOS = LABEL1 + 1

CLRF Clear "f"

Syntax	CLRF f
Operands	f : 00h ~ 7Fh
Operation	(f) \leftarrow 00h, Z \leftarrow 1
Status Affected	Z
OP-Code	00 0001 1fff ffff
Description	The contents of register 'f' are cleared and the Z bit is set.
Cycle	1
Example	CLRF FLAG_REG B : FLAG_REG = 0x5A A : FLAG_REG = 0x00, Z = 1

CLRW Clear W

Syntax	CLRW
Operands	-
Operation	(W) \leftarrow 00h, Z \leftarrow 1
Status Affected	Z
OP-Code	00 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

CLRWD T Clear Watchdog Timer

Syntax	CLRWD T
Operands	-
Operation	WDT/WKT Timer \leftarrow 00h
Status Affected	TO, PD
OP-Code	00 0000 0000 0100
Description	CLRWD T instruction clears the Watchdog/Wakeup Timer
Cycle	1
Example	CLRWD T B : WDT counter = ? A : WDT counter = 0x00

COMF
Complement "f"

Syntax	COMF f [,d]
Operands	f : 00h ~ 7Fh, d : 0, 1
Operation	(destination) \leftarrow (\bar{f})
Status Affected	Z
OP-Code	00 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	COMF REG1, 0 B : REG1 = 0x13 A : REG1 = 0x13, W = 0xEC

DECF
Decrement "f"

Syntax	DECF f [,d]
Operands	f : 00h ~ 7Fh, d : 0, 1
Operation	(destination) \leftarrow (f) - 1
Status Affected	Z
OP-Code	00 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	DECF CNT, 1 B : CNT = 0x01, Z = 0 A : CNT = 0x00, Z = 1

DECFSZ
Decrement "f", Skip if 0

Syntax	DECFSZ f [,d]
Operands	f : 00h ~ 7Fh, d : 0, 1
Operation	(destination) \leftarrow (f) - 1, skip next instruction if result is 0
Status Affected	-
OP-Code	00 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 DECFSZ CNT, 1 GOTO LOOP CONTINUE B : PC = LABEL1 A : CNT = CNT - 1 if CNT = 0, PC = CONTINUE if CNT \neq 0, PC = LABEL1 + 1

GOTO
Unconditional Branch

Syntax	GOTO k
Operands	k : 000h ~ FFFh
Operation	PC.11~0 \leftarrow k
Status Affected	-
OP-Code	11 kkkk kkkk kkkk
Description	GOTO is an unconditional branch. The 12-bit immediate value is loaded into PC bits <11:0>. GOTO is a two-cycle instruction.
Cycle	2
Example	LABEL1 GOTO SUB1 B : PC = LABEL1 A : PC = SUB1



INCF	Increment "f"
Syntax	INCF f [,d]
Operands	f : 00h ~ 7Fh
Operation	(destination) \leftarrow (f) + 1
Status Affected	Z
OP-Code	00 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	INCF CNT, 1 B : CNT = 0xFF, Z = 0 A : CNT = 0x00, Z = 1

INCFSZ	Increment "f", Skip if 0
Syntax	INCFSZ f[,d]
Operands	f : 00h ~ 7Fh, d : 0, 1
Operation	(destination) \leftarrow (f) + 1, skip next instruction if result is 0
Status Affected	-
OP-Code	00 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	<div style="display: flex; justify-content: space-between;"> <div> <p>LABEL1 INCFSZ CNT, 1</p> <p>GOTO LOOP</p> <p>CONTINUE</p> </div> <div> <p>B : PC = LABEL1</p> <p>A : CNT = CNT + 1</p> <p>if CNT = 0, PC = CONTINUE</p> <p>if CNT \neq 0, PC = LABEL1 + 1</p> </div> </div>

IORLW	Inclusive OR Literal with W	
Syntax	IORLW k	
Operands	k : 00h ~ FFh	
Operation	(W) ← (W) OR k	
Status Affected	Z	
OP-Code	01 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

IORWF	Inclusive OR W with "f"
Syntax	IORWF f [,d]
Operands	f : 00h ~ 7Fh, d : 0, 1
Operation	(destination) ← (W) OR k
Status Affected	Z
OP-Code	00 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	IORWF RESULT, 0 B : RESULT = 0x13, W = 0x91 A : RESULT = 0x13, W = 0x93, Z = 0

MOVFW Move "f" to W

Syntax	MOVFW f	
Operands	f : 00h ~ 7Fh	
Operation	(W) ← (f)	
Status Affected	-	
OP-Code	00 1000 0fff ffff	
Description	The contents of register 'f' are moved to W register.	
Cycle	1	
Example	MOVFW 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	01 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

MOVWF Move W to "f"

Syntax	MOVWF f	
Operands	f : 00h ~ 7Fh	
Operation	(f) ← (W)	
Status Affected	-	
OP-Code	00 0000 1fff ffff	
Description	Move data from W register to register 'f'.	
Cycle	1	
Example	MOVWF REG1	B : REG1 = 0xFF, W = 0x4F A : REG1 = 0x4F, W = 0x4F

MOVWR Move W to "r"

Syntax	MOVWR r	
Operands	r : 00h ~ 3Fh	
Operation	(r) ← (W)	
Status Affected	-	
OP-Code	00 0000 00rr rrrr	
Description	Move data from W register to register 'r'.	
Cycle	1	
Example	MOVWR REG1	B : REG1 = 0xFF, W = 0x4F A : REG1 = 0x4F, W = 0x4F



NOP	No Operation
------------	---------------------

Syntax	NOP
Operands	-
Operation	No Operation
Status Affected	-
OP-Code	00 0000 0000 0000
Description	No Operation
Cycle	1
Example	NOP

RET	Return from Subroutine
------------	-------------------------------

Syntax	RET
Operands	-
Operation	PC ← TOS
Status Affected	-
OP-Code	00 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	00 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	01 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	CALL TABLE B : W = 0x07 : A : W = value of k8 TABLE ADDWF PCL, 1 RETLW k1 RETLW k2 : RETLW kn

RLF Rotate Left "f" through Carry

Syntax	RLF f [,d]
Operands	f : 00h ~ 7Fh, d : 0, 1
Operation	
Status Affected	C
OP-Code	00 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	RLF REG1, 0 B : REG1 = 1110 0110, C = 0 A : REG1 = 1110 0110 W = 1100 1100, C = 1

RRF Rotate Right "f" through Carry

Syntax	RRF f [,d]
Operands	f : 00h ~ 7Fh, d : 0, 1
Operation	
Status Affected	C
OP-Code	00 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	RRF 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	00 0000 0000 0011
Description	Go into Power-down mode with the oscillator stops.
Cycle	1
Example	SLEEP -

SUBLW Subtract W from Literal "k"

Syntax	SUBLW k	
Operands	k : 00h ~ FFh	
Operation	$(W) \leftarrow (k) - (W)$	
Status Affected	C, DC, Z	
OP-Code	01 1101 kkkk kkkk	
Description	The contents of the W register are subtracted (2's complement method) from the eight-bit literal 'k' and the result is placed in the W register.	
Cycle	1	
Example	SUBLW 0x15	B : W = 0x10, C = ?, Z = ? A : W = 0x05, C = 1, Z = 0
	SUBLW 0x10	B : W = 0x10, C = ?, Z = ? A : W = 0x00, C = 1, Z = 1
	SUBLW 0x05	B : W = 0x10, C = ?, Z = ? A : W = 0xF5, C = 0, Z = 0

SUBWF Subtract W from "f"

Syntax	SUBWF f [,d]	
Operands	f : 00h ~ 7Fh, d : 0, 1	
Operation	$(\text{destination}) \leftarrow (f) - (W)$	
Status Affected	C, DC, Z	
OP-Code	00 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	SUBWF REG1, 1	B : REG1 = 0x03, W = 0x02, C = ?, Z = ? A : REG1 = 0x01, W = 0x02, C = 1, Z = 0
	SUBWF REG1, 1	B : REG1 = 0x02, W = 0x02, C = ?, Z = ? A : REG1 = 0x00, W = 0x02, C = 1, Z = 1
	SUBWF REG1, 1	B : REG1 = 0x01, W = 0x02, C = ?, Z = ? A : REG1 = 0xFF, W = 0x02, C = 0, Z = 0

SWAPF Swap Nibbles in "f"

Syntax	SWAPF f [,d]	
Operands	f : 00h ~ 7Fh, d : 0, 1	
Operation	$(\text{destination}, 7 \sim 4) \leftarrow (f, 3 \sim 0), (\text{destination}, 3 \sim 0) \leftarrow (f, 7 \sim 4)$	
Status Affected	-	
OP-Code	00 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	SWAPF 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	00 0000 0101 1000
Description	The W register is loaded with high byte of ROM[DPTR]. This is a two-cycle instruction.
Cycle	2
Example	<pre> MOVLW (TAB1&0xFF) MOVWF FSR ;Where FSR is F-Plane register MOVLW (TAB1>>8)&0xFF MOVWF DPH ;Where DPH is F-Plane register TABRL TABRH ;W = 0x89 ;W = 0x37 ORG 0234H TAB1: DT 0x3789, 0x2277 ;ROM data 14bits </pre>

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	00 0000 0101 0000
Description	The W register is loaded with low byte of ROM[DPTR]. This is a two-cycle instruction.
Cycle	2
Example	<pre> MOVLW (TAB1&0xFF) MOVWF FSR ;Where FSR is F-Plane register MOVLW (TAB1>>8)&0xFF MOVWF DPH ;Where DPH is F-Plane register TABRL TABRH ;W = 0x89 ;W = 0x37 ORG 0234H TAB1: DT 0x3789, 0x2277 ;ROM data 14bits </pre>

TESTZ Test if 'f' is zero

Syntax	TESTZ f	
Operands	f : 00h ~ 7Fh	
Operation	Set Z flag if (f) is 0	
Status Affected	Z	
OP-Code	00 1000 1fff ffff	
Description	If the content of register 'f' is 0, Zero flag is set to 1.	
Cycle	1	
Example	TESTZ 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	01 1111 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

XORWF Exclusive OR W with 'f'

Syntax	XORWF f [,d]	
Operands	f : 00h ~ 7Fh, d : 0, 1	
Operation	(destination) ← (W) XOR (f)	
Status Affected	Z	
OP-Code	00 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	XORWF REG, 1	B : REG = 0xAF, W = 0xB5 A : REG = 0x1A, W = 0xB5

ELECTRICAL CHARACTERISTICS

1. Absolute Maximum Ratings ($T_A = 25^\circ\text{C}$)

Parameter	Rating	Unit
Supply voltage	$V_{SS} - 0.3$ to $V_{SS} + 4.2$	V
Input voltage	$V_{SS} - 0.3$ to $V_{DD} + 0.3$	
Output voltage	$V_{SS} - 0.3$ to $V_{DD} + 0.3$	
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	4.2	V
Operating temperature	-40 to +85	$^\circ\text{C}$
Storage temperature	-65 to +150	

2. DC Characteristics ($T_A = 25^\circ\text{C}$, $V_{DD} = 1.2\text{V to } 4.2\text{V}$)

Parameter	Symbol	Conditions		Min	Typ	Max	Unit
Operating Voltage	V_{DD}	FAST mode, 25°C , $F_{\text{sys}} = 4\text{ MHz}$		1.5	–	4.2	V
		SLOW mode, 25°C , SIRC		1.2	–	4.2	
Input High Voltage	V_{IH}	All Input	$V_{DD} = 3\text{V}$	$0.6V_{DD}$	–	–	V
Input Low Voltage	V_{IL}	All Input	$V_{DD} = 3\text{V}$	–	–	$0.2V_{DD}$	V
I/O Port Source Current	I_{OH}	All Output except PA7	$V_{DD} = 3\text{V}$, $V_{OH} = 0.9V_{DD}$	2	4	–	mA
I/O Port Sink Current	I_{OL}	All Output, except PA0、PA7	$V_{DD} = 3\text{V}$, $V_{OL} = 0.1V_{DD}$	7	14	–	mA
		PA0 HSNK disable	$V_{DD} = 3\text{V}$, $V_{OL} = 0.1V_{DD}$	7	14	–	
		PA0 HSNK enable	$V_{DD} = 3\text{V}$, $V_{OL} = 0.1V_{DD}$	–	240	–	
			$V_{DD} = 3\text{V}$, $V_{OL} = 0.2V_{DD}$	–	420	–	
Supply Current	I_{DD}	PA7	$V_{DD} = 3\text{V}$, $V_{OL} = 0.1V_{DD}$	5	10	–	μA
		FAST mode, LVR enable, WDT enable	$V_{DD} = 3\text{V}$, $F_{\text{IRC}} = 4\text{ MHz}$	–	0.5	–	
		SLOW mode LVR enable	$V_{DD} = 3\text{V}$, SIRC, WKTPSC = 11	–	4	–	
		IDLE mode, LVR enable	$V_{DD} = 3\text{V}$, SIRC, WKTPSC = 11	–	3.7	–	
		IDLE mode, LVR disable	$V_{DD} = 3\text{V}$, SIRC, WKTPSC = 11	–	2.3	–	
		STOP mode, LVR enable	$V_{DD} = 3\text{V}$	–	1.3	–	
LVR Reference Voltage	V_{LVR}	$T_A = 25^\circ\text{C}$	TM57ME16	–	1.2	–	V
			TM57ME18	–	1.6	–	
LVR Hysteresis Voltage	V_{HYST}	$T_A = 25^\circ\text{C}$		–	± 0.1	–	V
Low Voltage Detection time	t_{LVR}	$T_A = 25^\circ\text{C}$		100	–	–	μs
Pull-Up Resistor	R_P	$V_{IN} = 0\text{V}$ Port A	$V_{DD} = 3\text{V}$	–	230	–	$\text{K}\Omega$

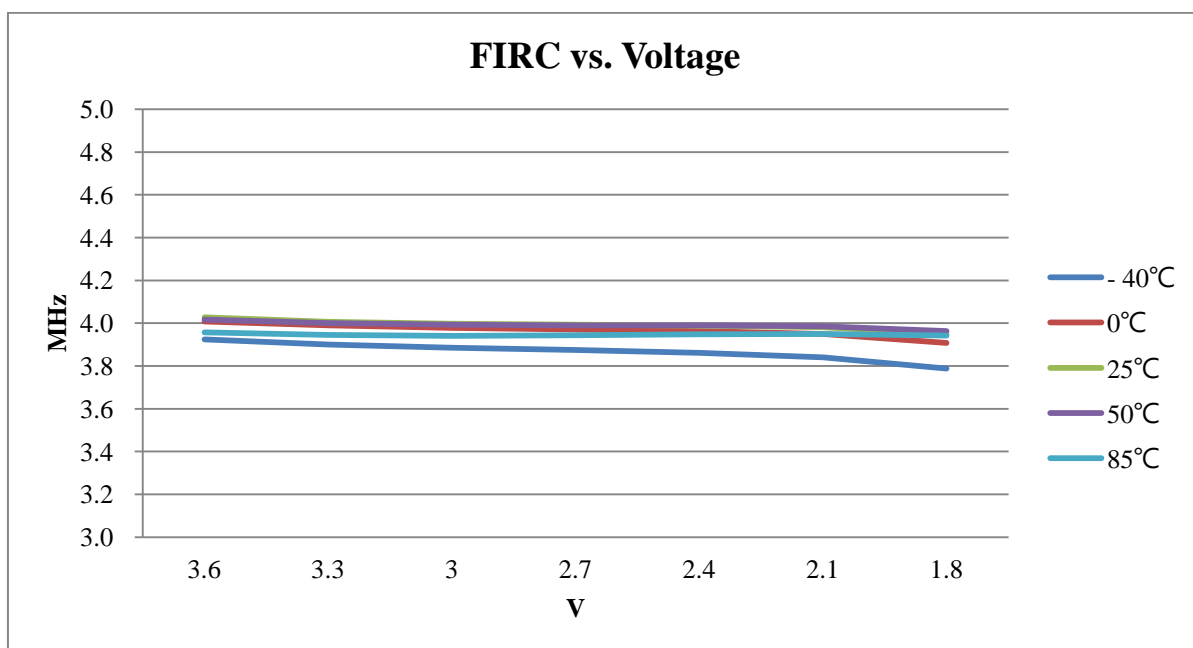
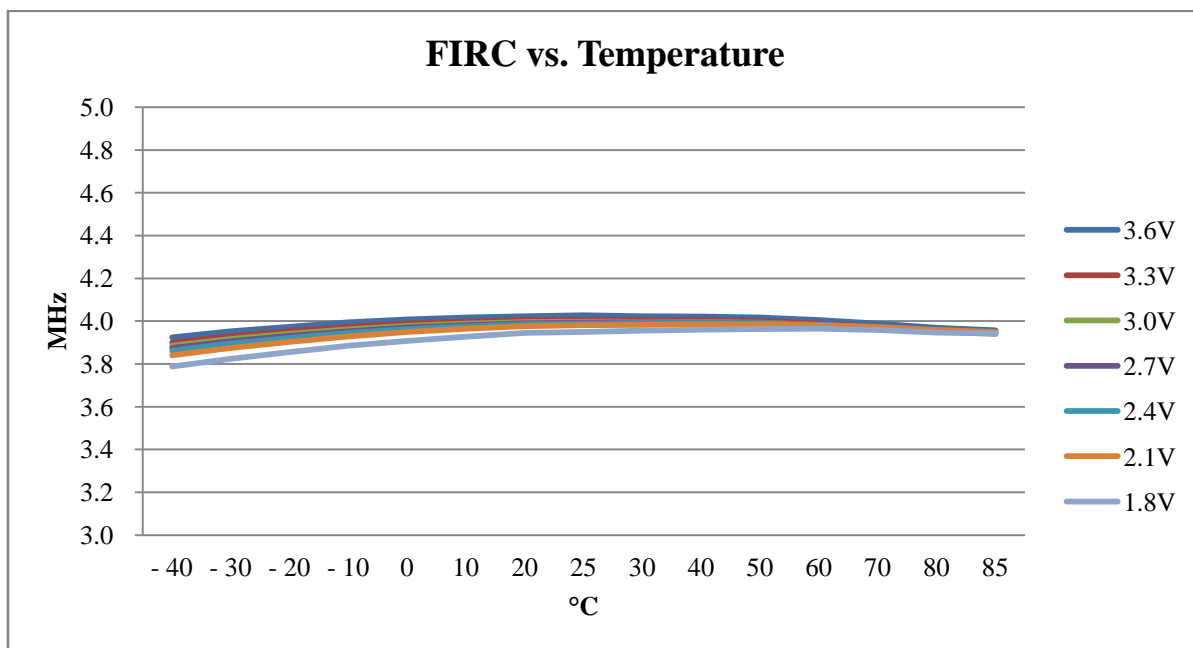
3. Clock Timing ($T_A = -40^{\circ}\text{C}$ to $+85^{\circ}\text{C}$)

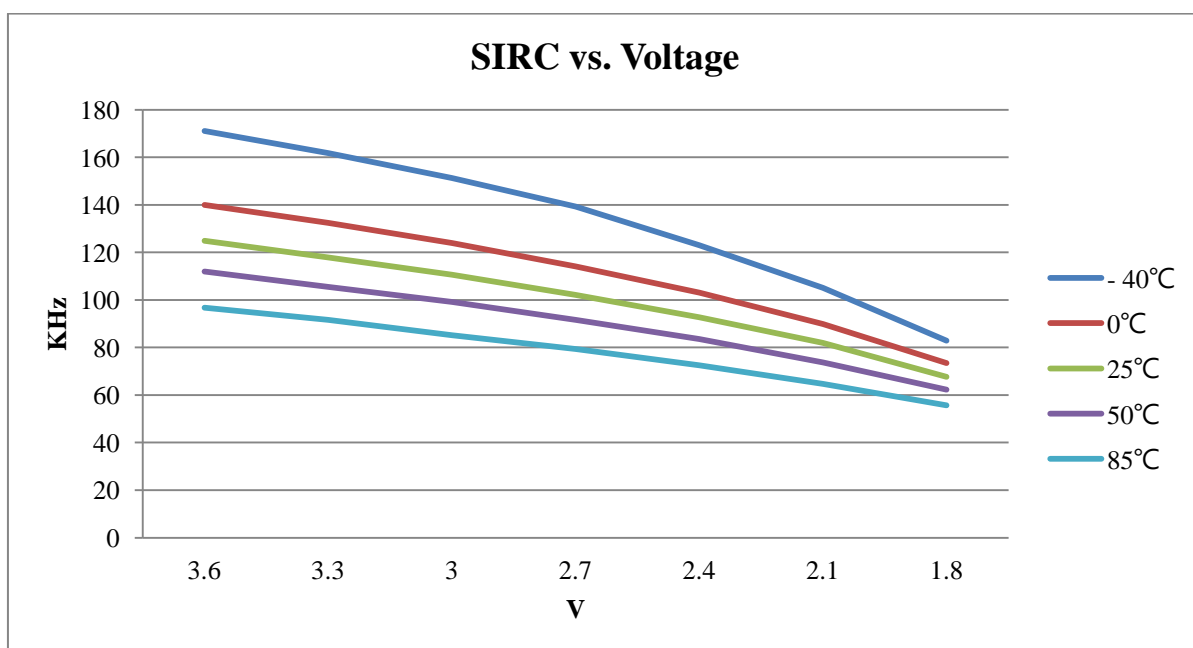
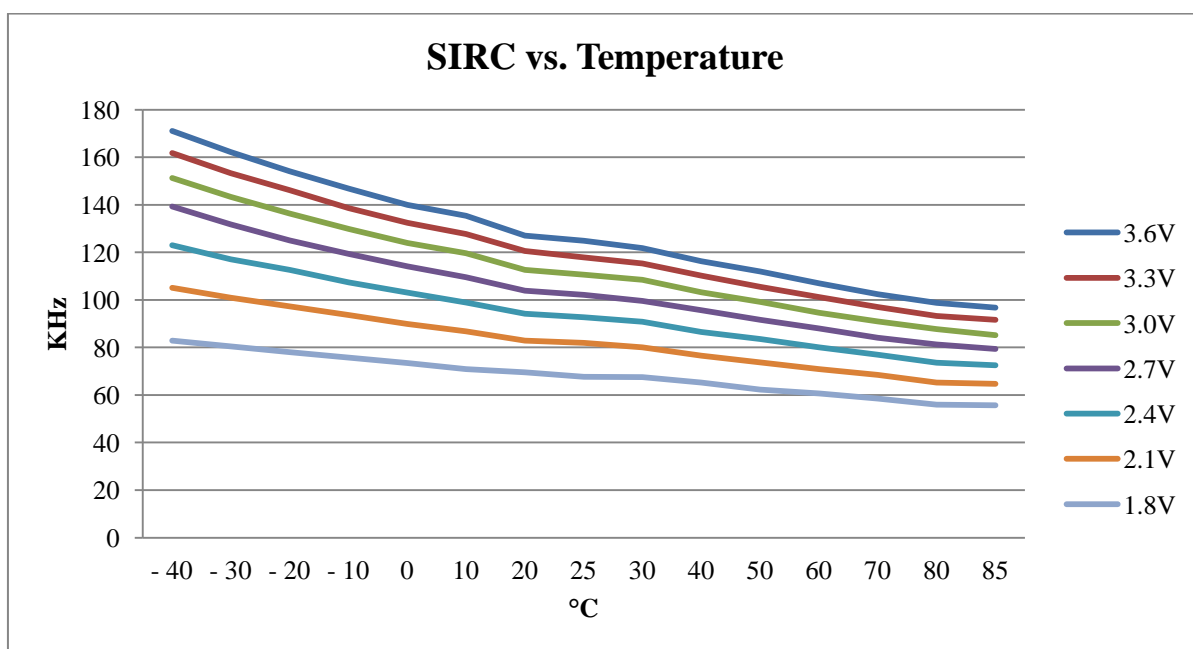
Parameter	Condition	Min	Typ	Max	Unit
Internal RC Frequency	25°C, $V_{DD} = 2.0 \sim 3.6\text{V}$	3.9	4	4.1	MHz
	-40°C ~ 85°C, $V_{DD} = 2.0 \sim 3.6\text{V}$	3.8	4	4.2	

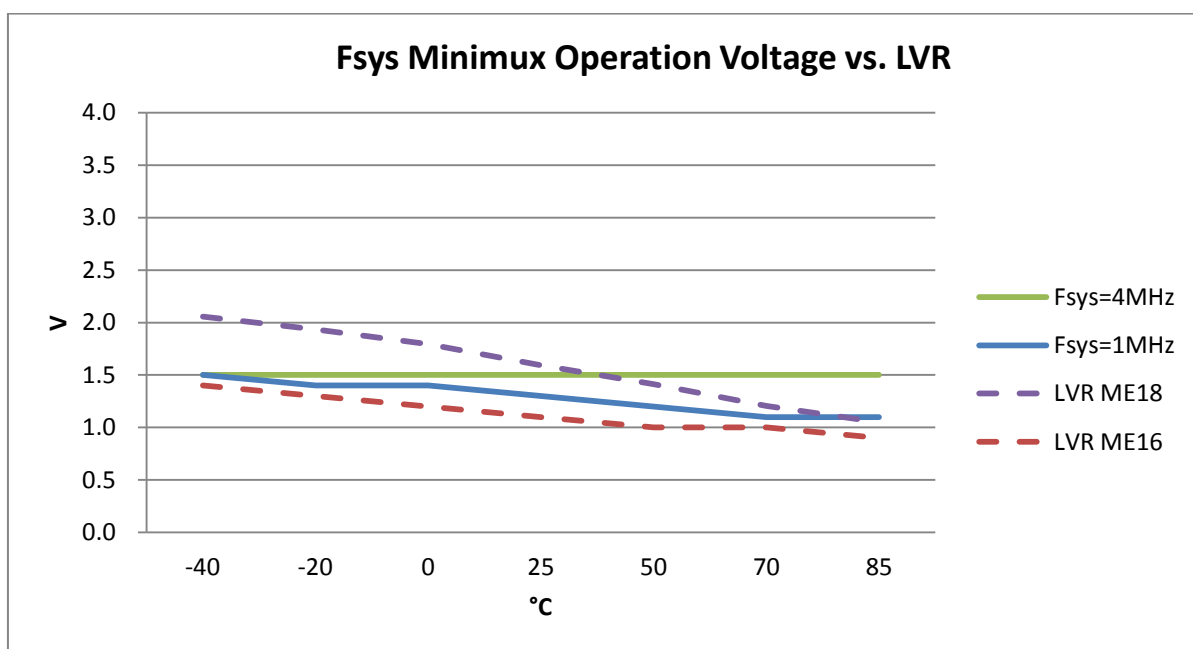
4. Reset Timing Characteristics ($T_A = -40^{\circ}\text{C}$ to $+85^{\circ}\text{C}$)

Parameter	Conditions	Min	Typ	Max	Unit
RESET Input Low width	Input $V_{DD} = 3\text{V} \pm 10\%$	3	—	—	μs
WDT wakeup time	$V_{DD} = 3\text{V}$, WKTPSC = 00	—	18	—	ms
CPU start up time	$V_{DD} = 3\text{V}$, SLOWM = 1	—	4.7	—	ms
	$V_{DD} = 3\text{V}$, SLOWM = 0	—	5	—	

5. Characteristic Graphs







Note: Due to the variation of manufacturing process, this LVR will slightly vary between different chips.

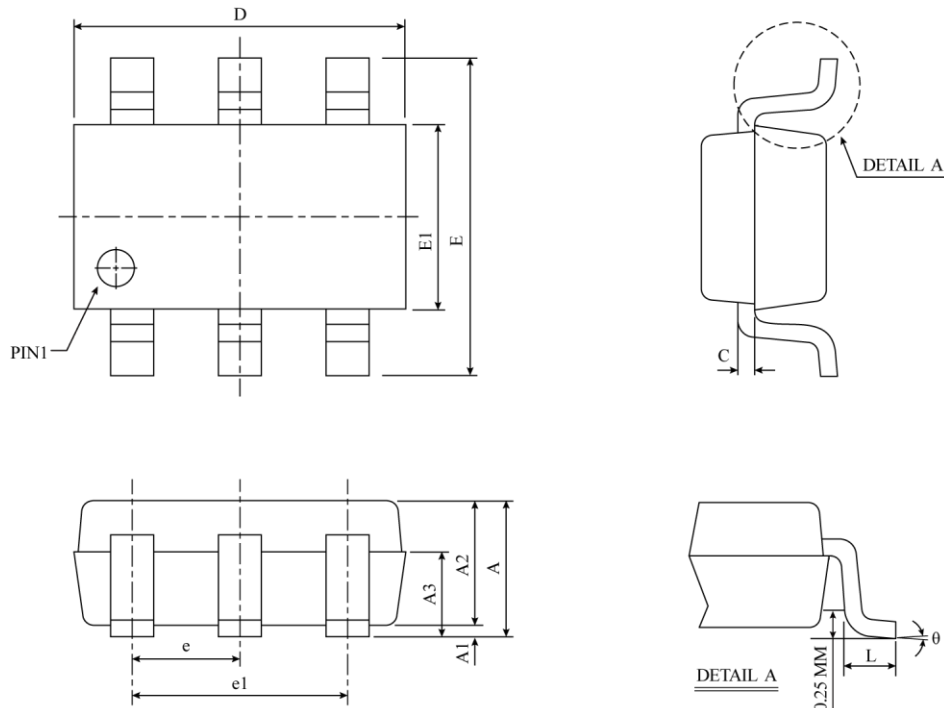
PACKAGING INFORMATION

The ordering information:


Ordering number	Package
TM57ME16-MTP	Wafer / Dice blank chip
TM57ME16-COD	Wafer / Dice with code
TM57ME16AS-MTP-A8	SOT23-6
TM57ME16-MTP-01	DIP 8-pin (300 mil)
TM57ME16-MTP-14	SOP 8-pin (150 mil)
TM57ME16-MTP-43	TSSOP-8
TM57ME18-MTP	Wafer / Dice blank chip
TM57ME18-COD	Wafer / Dice with code
TM57ME18CS-MTP-A8	SOT23-6
TM57ME18-MTP-01	DIP 8-pin (300 mil)
TM57ME18-MTP-14	SOP 8-pin (150 mil)

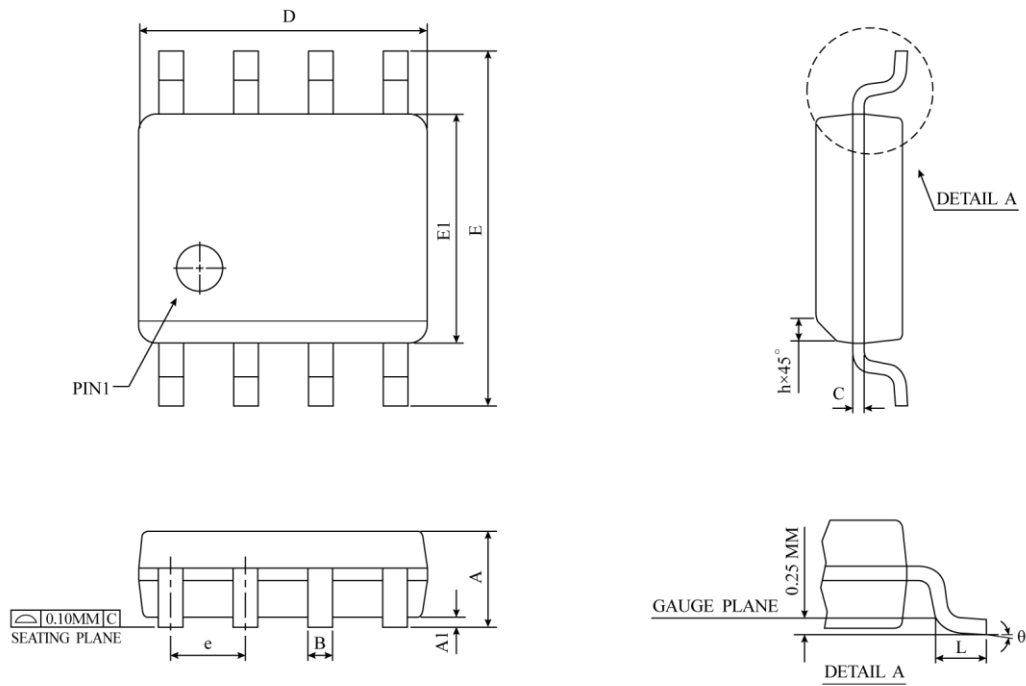
Package Information

SOT23-6 Package Dimension



SYMBOL	DIMENSION IN MM			DIMENSION IN INCH		
	MIN	NOM	MAX	MIN	NOM	MAX
A	-	-	1.45	-	-	0.057
A1	0	0.08	0.15	0	0.003	0.006
A2	0.90	1.10	1.30	0.035	0.043	0.051
A3	0.60	0.65	0.70	0.024	0.026	0.028
c	0.12	0.16	0.19	0.005	0.006	0.007
D	2.82	2.92	3.02	0.111	0.115	0.119
E	2.70	2.90	3.10	0.106	0.114	0.122
E1	1.52	1.62	1.72	0.060	0.064	0.068
e	0.85	0.95	1.05	0.033	0.037	0.041
e1	1.80	1.90	2.00	0.071	0.075	0.079
L	0.35	0.48	0.60	0.014	0.019	0.024
θ	0°	4°	8°	0°	4°	8°
JEDEC	M0-178 (AB)					

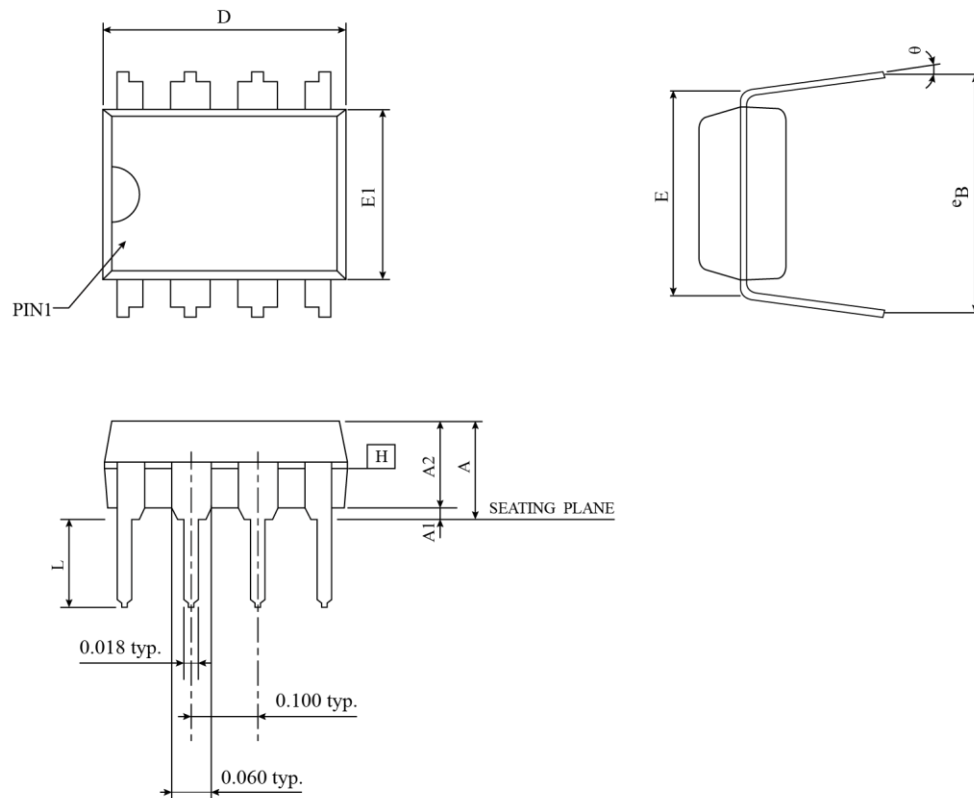
 * NOTES : ALL DIMENSIONS REFER TO JEDEC STANDARD MO-178 AB
DO NOT INCLUDE MOLD FLASH OR PROTRUSIONS.

SOP-8 (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	4.80	4.90	5.00	0.1890	0.1939	0.1988
E	5.80	6.00	6.20	0.2284	0.2362	0.2440
E1	3.80	3.90	4.00	0.1497	0.1536	0.1574
e	1.27 BSC			0.050 BSC		
h	0.25	0.38	0.50	0.0099	0.0148	0.0196
L	0.40	0.84	1.27	0.0160	0.0330	0.0500
θ	0°	4°	8°	0°	4°	8°
JEDEC	MS-012 (AA)					

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

DIP-8 (300mil) Package Dimension

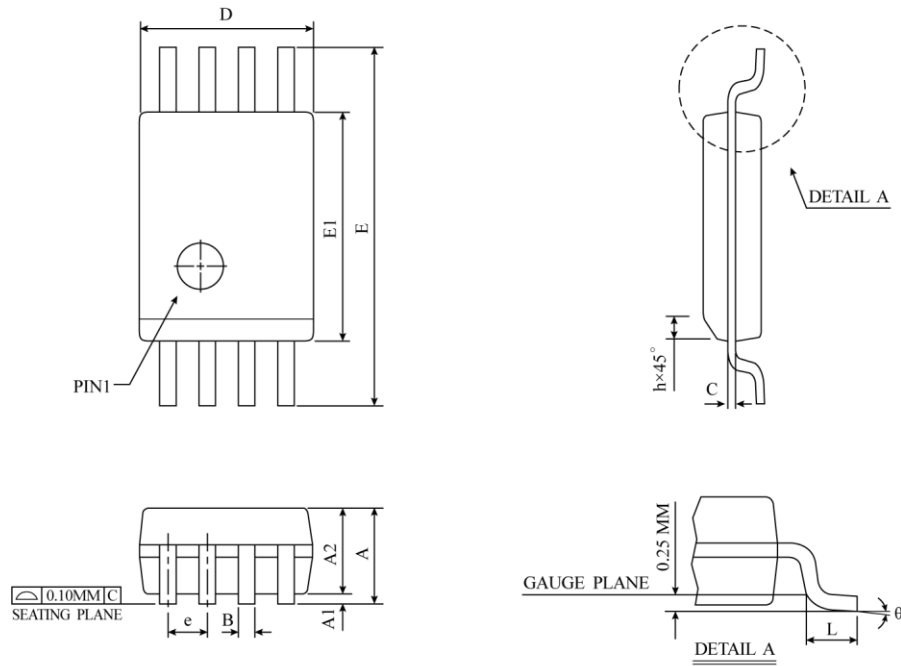


SYMBOL	DIMENSION IN MM		DIMENSION IN INCH	
	MIN	MAX	MIN	MAX
A	-	5.334	-	0.210
A1	0.381	-	0.015	-
A2	3.175	3.429	0.125	0.135
D	9.017	10.160	0.355	0.400
E	7.620 BSC		0.300 BSC	
E1	6.223	6.477	0.245	0.255
L	2.921	3.810	0.115	0.150
eB	8.509	9.525	0.335	0.375
θ	0°	15°	0°	15°
JEDEC	MS-001 (BA)			

NOTES :

1. "D", "E1" DIMENSIONS DO NOT INCLUDE MOLD FLASH OR PROTRUSIONS. MOLD FLASH OR PROTRUSIONS SHALL NOT EXCEED .010 INCH.
2. eB IS MEASURED AT THE LEAD TIPS WITH THE LEADS UNCONSTRAINED.
3. POINTED OR ROUNDED LEAD TIPS ARE PREFERRED TO EASE INSERTION.
4. DISTANCE BETWEEN LEADS INCLUDING DAM BAR PROTRUSIONS TO BE .005 INCH MINIMUM.
5. DATUM PLANE \square COINCIDENT WITH THE BOTTOM OF LEAD, WHERE LEAD EXITS BODY.

TSSOP-8 (173mil) Package Dimension



SYMBOL	DIMENSION IN MM			DIMENSION IN INCH		
	MIN	NOM	MAX	MIN	NOM	MAX
A	-	-	1.20	-	-	0.047
A1	0.05	0.10	0.15	0.002	0.004	0.006
A2	0.80	0.93	1.05	0.031	0.036	0.041
B	0.22 TYP			0.009 TYP		
D	2.90	3.00	3.10	0.114	0.118	0.122
E	6.40 BSC			0.252 BSC		
E1	4.30	4.40	4.50	0.169	0.173	0.177
e	0.65 TYP			0.026 TYP		
L	0.45	0.60	0.75	0.018	0.024	0.030
θ	0°	4°	8°	0°	4°	8°
JEDEC	MS-153 (AA)					

- △ *NOTES : DIMENSION " D " DOES NOT INCLUDE MOLD FLASH, PROTRUSIONS OR GATE BURRS.
MOLD FLASH, PROTRUSIONS OR GATE BURRS SHALL NOT EXCEED 0.15 PER SIDE.
DIMENSION " E1 " DOES NOT INCLUDE INTERLEAD FLASH OR PROTRUSION.
INTERLEAD FLASH OR PROTRUSIONS SHALL NOT EXCEED 0.25 PER SIDE.
DIMENSION " B " DOES NOT INCLUDE DAMBAR PROTRUSION.
ALLOWABLE DAMBAR PROTRUSION SHALL BE 0.08 MM TOTAL IN EXCESS OF
THE " B " DIMENSION AT MAXIMUM LOWER RADIUS OF THE LOWER RADIUS OF THE FOOT.
MINIMUM SPACE BETWEEN PROTRUSION AND ADJACENT LEAD IS 0.07 MM.