十速

TM57PA28

DATA SHEET

Rev 0.92

AMENDMENT HISTORY

Version	Date	Description
V0.90	Mar, 2014	New release
V0.91	Jan, 2015	1. Correct ADCKS (R0E)typing error (p40) 2. Correct PWMANOV=000, PWMACLK*0 (p53)
V0.92	Dec, 2015	1. Correct Touch Key to 6 channels (p6) 2. Add Instruction time (p12) 3. Add dead-band notice (p21) 4. Add LVR temperature curve (p73)

CONTENTS

FEATURES

1. ROM: 2K x 14 bits OTP
2. RAM: 176 x 8 bits
3. STACK: 5 Levels
4. System Oscillation Sources
   - Fast-clock: FIRC (Fast Internal RC): 1 / 2 / 4 / 8 MHz
   - Slow-clock: SIRC (Slow Internal RC): 110KHz@5V; 88KHz@3V
5. System Clock Prescaler: System Oscillation Sources can be divided by 16/4/2/1 as System Clock (Fsys)
6. Power Saving Operation Modes
   - FAST Mode: Slow-clock can be disabled or enabled.
   - SLOW Mode: Fast-clock stops, Slow-clock keeps CPU running
   - IDLE Mode: Fast-clock and CPU stop. T2 keeps running
   - STOP Mode: All Clocks stop, T2 stops
7. Dual System Clock
   - FIRC + SIRC
8. 3 Independent Timers
   - Timer0
     - 8-bit timer divided by 1 ~ 256 pre-scales option, Counter / Interrupt / Stop function
     - Overflow and Toggle out
   - Timer1
     - 8-bit timer with two pre-scalers options, Counter / Interrupt / Stop / Reload 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: SIRC, Fsys/128
9. PWM
   - (8+2) bits PWMA, with duty-adjustable / period-adjustment / buffer-reload
   - Clock Source (PWMACLK): Fsys / 16 MHz / 32 MHz
   - Complementary PWM output (PWMAP, PWMAN)
   - Non-overlap time durations adjustable: (0~10)*(1/$f_{PWMACLK}$)
   - PWMAP / PWMAN are high drive / sink pins

10. Touch Key: 6 channels

11. ADC

- 12-bit ADC with 11 channels and 1 internal reference voltage(VBG, 1.25±1%) on 11th channel
- ADC's AVREF power source can from internal LDO 2.5V or external AVREF pin
- ADC input channel voltage swing range is 0~AVREF.

12. Built-in LDO 2.5V±1% output to LDOC pin and also supply to ADC's AVREF.

13. Over Current Protection (OCP)

14. Over Voltage Protection (OVP)

15. Interrupt

- Three External Interrupt pins
  - 1 pin (PA7/INT2) is falling edge wake-up triggered & interrupts
  - 2 pins (PA6/INT0, PA1/INT1) are rising or falling edge wake-up triggered & interrupt
- Timer0 / Timer1 / T2 / OCP / OVP interrupts

16. PD1, PB1, PB0 individual pin low level wake up

17. Reset Sources

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

18. Low Voltage Reset Option: 2.2V / 3.1V / 2.2V, disable in STOP mode / Disable

19. Operation Voltage: Low Voltage Reset Level to 5.5V

- Fsys = 4 MHz, 2.0V ~ 5.5V
- Fsys = 8 MHz, 2.4V ~ 5.5V

20. Operating Temperature Range: -40℃ to +85℃

21. Interrupts

- Three External Interrupt Pins
  - Two pins are falling edge triggered
  - One pin is rising or falling edge triggered
- Timer0 / T2 / Comparator Interrupts

22. Watchdog Timer (WDT)

- Clocked by built-in RC oscillator with 4 adjustable reset time options

  140 ms / 280 ms / 1100 ms / 2280 ms @ V_{DD} = 5V
- Watchdog timer can be disabled/enabled in STOP/IDLE mode

23. Table Read Instruction: 14-bit ROM data lookup table

24. Instruction Set: 39 Instructions

25. Instruction Execution Time: 2 oscillation clocks per instruction except branch

26. 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

27. I/O Ports: Three bit-programmable I/O ports (Max. 18 pins)

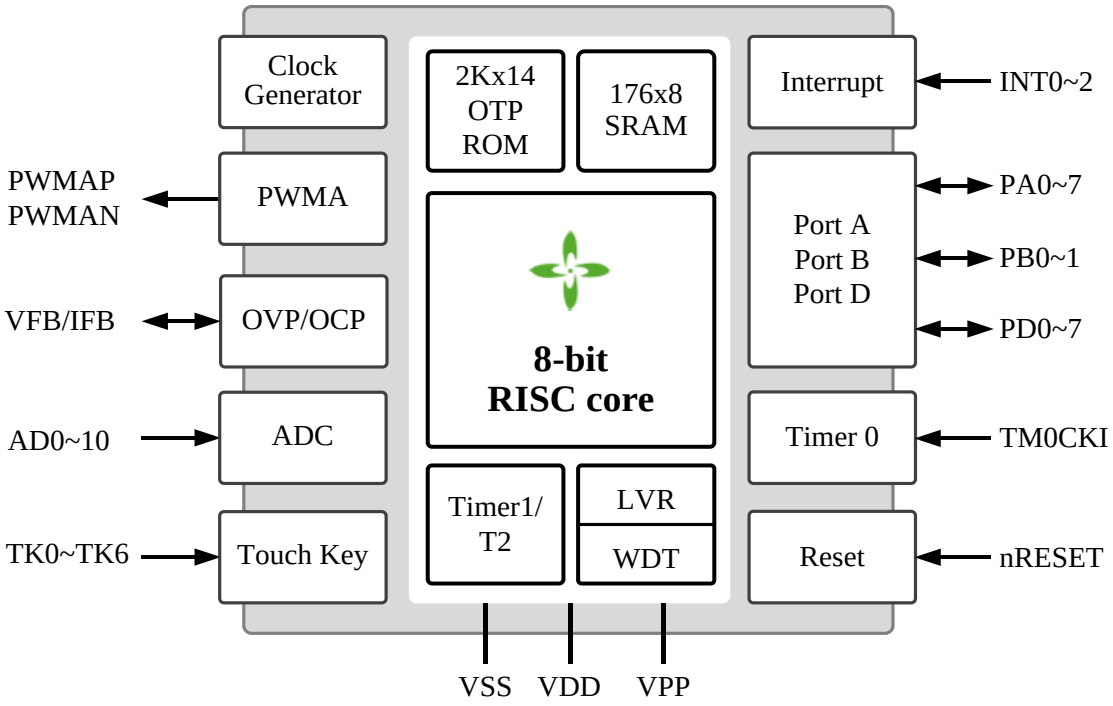
28. Package Types:

- 20-pin SSOP (209 mil)
- 20-pin DIP (300 mil)
- 20-pin SOP (300 mil)
- 16-pin DIP (300 mil)
- 16-pin SOP (150 mil)

29. Supported EV board on ICE

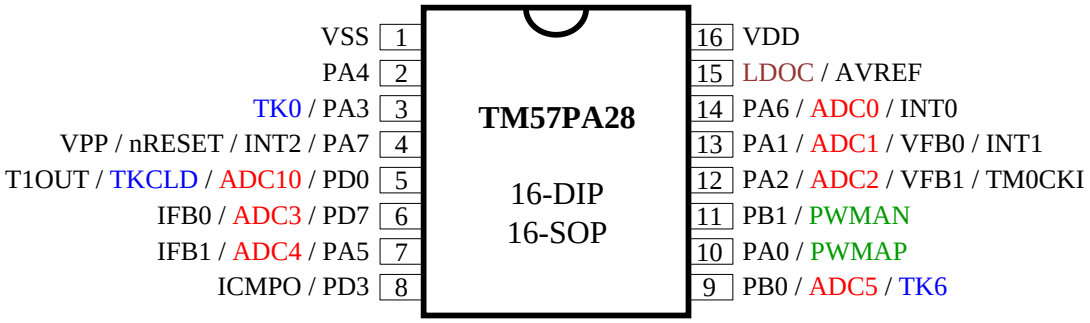
EV board: EV2767

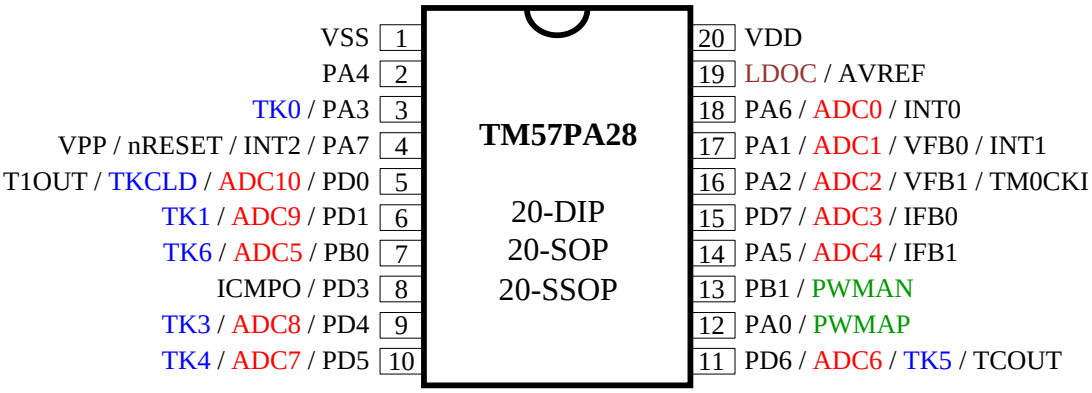
BLOCK DIAGRAM



TM57PA28 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~PA6	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.
PB0~PB1	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.
PD0~PD7	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. Pull-up resistor is assignable by software.
TCOUT	O	Instruction cycle clock output. can be Fsys/2, Fsys/4, 4 MHz or 4 MHz invert
TM0OUT	O	Timer0 overflow toggle output
TM1OUT	O	Timer1 overflow toggle output
ADC0~ADC11	I	A/D converter input
VDD, VSS	P	Power Voltage input pin and ground
VPP	I	PROM programming high voltage input
INT0~INT2	I	External interrupt input
PWMAP PWMAN	O	PWMA complement outputs
TM0CKI	I	Timer0's input in counter mode
TK0~TK6	I	Touch key input
TKCLD	I	Touch key capacitor input
LDOC(1)	O	Internal reference voltage 2.5V output, need connect 1 uF to ground
AVREF(1)	I	ADC reference voltage input
VFB0/VFB1	I	Voltage feedback input channel
IFB0/IFB1	I	Current feedback input channel
ICMPO	O	Current feedback compare output

Note:

(1) According to IVREFS(R17.1) to decide the LDOC or AVREF function. After power-on reset, this pin function is used to act AVREF (IVREFS=0).

Programming Pins:

- Normal Mode: VDD, VSS, PA0, PA1, PA3, PA4, VPP
- ISP Mode: VDD, VSS, PA0, PA1, VPP. (Note: When using ISP mode, the related pin's components must be restricted or removed.)

PIN SUMMARY

Pin Number		Pin Name	Type	GPIO					Function After Reset	Alternate Function			
				Input		Output							
20-SOP/DIP	16-SOP/DIP			Weak Pull-up	Ext. Interrupt	O.D	P.O.D	P.P		PWM	Touch Key	ADC	MISC
1	1	VSS	P										
2	2	PA4	I/O			O		O	PA4				
3	3	TK0/PA3	I/O					O	PA3		O		
4	4	VPP/nRESET/ INT2/PA7	I/O	O	O	O			PA7				nRESET
5	5	T1OUT/TKCLD/ ADC10/PD0	I/O			O		O	PD0		O	O	T1OUT
6		TK1/ADC9/PD1	I/O			O		O	PD1		O	O	
7	9	TK6/ADC5/PB0	I/O	O		O		O	PB0		O	O	
8	8	ICMPO/PD3	I/O			O		O	PD3				ICMPO
9		TK3/ADC8/PD4	I/O			O		O	PD4		O	O	
10		TK4/ADC7/PD5	I/O			O		O	PD5		O	O	
11		PD6/ADC6/ TK5/TCOUT	I/O			O		O	PD6		O	O	TCOUT
12	10	PA0/PWMAP	I/O				O	O	PA0	O			
13	11	PB1/PWMAN	I/O	O		O		O	PB1	O			
14	7	PA5/ADC4/IFB1	I/O			O		O	PA5			O	IFB1
15	6	PD7/ADC3/IFB0	I/O			O		O	PD7			O	IFB0
16	12	PA2/ADC2/VFB1/ TM0CKI	I/O				O	O	PA2			O	VFB1
17	13	PA1/ADC1/ VFB0/INT1	I/O	O	O		O	O	PA1			O	VFB0
18	14	PA6/ADC0/INT0	I/O	O	O	O		O	PA6			O	
19	15	LDOC/AVREF	P						AVREF			O	LDOC
20	16	VDD	P										

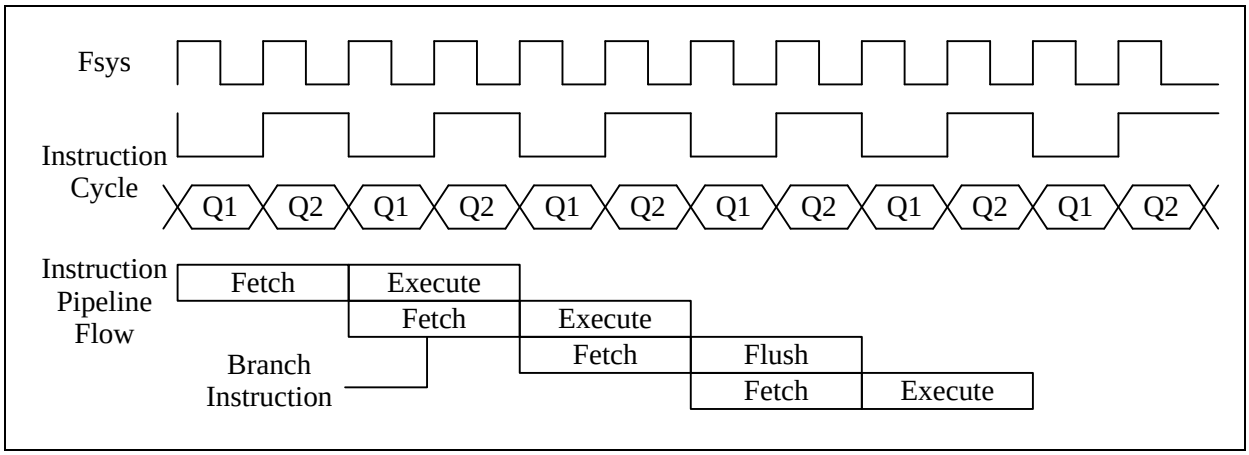
Symbol：P.P. = Push-Pull Output
P.O.D. = Pseudo Open Drain
O.D. = Open Drain

FUNCTIONAL DESCRIPTION

1. CPU Core

1.1 Clock Scheme and Instruction Cycle

The system clock (Fsys) 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.



Terminology definitions:

Fsys: System clock. The main clock that drives the core logic and all peripherals. The clock source can be either Fast-clock or Slow-clock which can be set by registers.

Fast-clock: The clock source is Fast Internal RC oscillator (FIRC).

Slow-clock: The clock source is Slow Internal RC oscillator (SIRC).

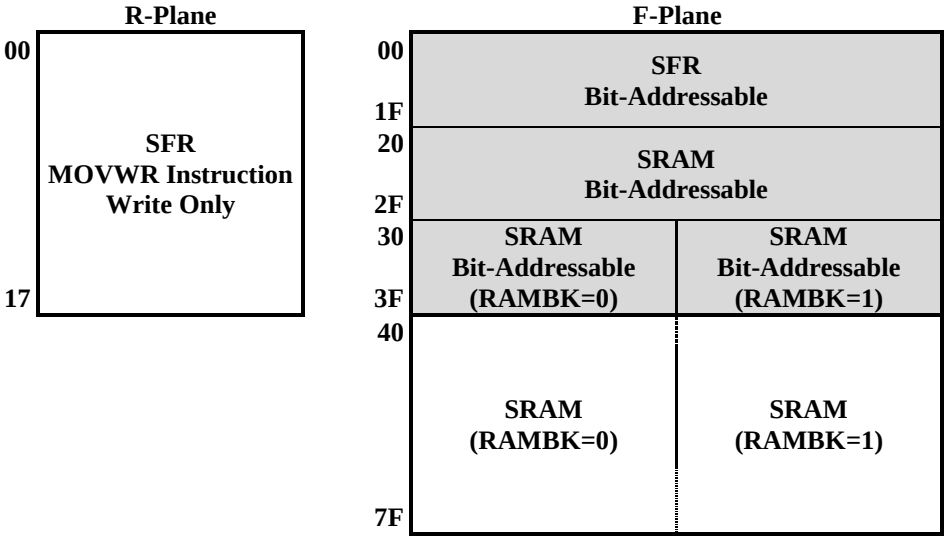
Instruction Cycle = Fsys / 2

Refer to the diagram "Clock Scheme Block Diagram" in the Chapter 2.

Fsys	Instruction time (if 2 cycles)	Instruction time (if 1 cycle)
4MHz	1uS	500nS
8MHz	500uS	250nS

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. And there are two RAM banks can be selected by RAMBK (F03.5).



◇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 SRAM Bank0 data by indirect addressing mode

```
        MOVLW   20H     ; W = 20H (SRAM start address)
        MOVWF   FSR     ; Set start address of user SRAM into FSR register
        BCF     STATUS,5 ; Set RAMBK = 0
LOOP:
        MOVLW   00H
        MOVFW   INDF    ; Clear user SRAM data
        INCF    FSR, 1  ; Increment the FSR for next address
        MOVLW   80H     ; W = 80H (SRAM end address)
        XORWF   FSR, 0  ; Check the FSR is end address of user SRAM?
        BTFSS   STATUS, 2 ; 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 11-bit wide capable of addressing a 2K x 14 OTP 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 11 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[10:8] keeps unchanged. Therefore, the data of a lookup table must be located with the same PC[10:8]. The STACK is 11-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 PROM data located "TABLE"

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

            ORG     X00H        ; X = 1, 2, 3, …, 6, 7
TABLE:
            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:

The device defines 256 ROM addresses as one page, so that it has eight pages, 000H~0FFH, 100H~1FFH, 200H~2FFH, …, and 700H~7FFH. On the other words, PC[10:8] 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, …, 6, 7). If a lookup table has fewer data, it needs not set the starting address at X00H, just only confirm all lookup table data are located at the same page.

◇Example: To look up the PROM data located in "TABLE" by TABRL and TABRH instructions

```
        ORG     000H            ; Reset Vector
        GOTO    START           ; Goto user program address
START:
        MOVLW   (TABLE>>3)&0xff ; Get high byte address of TABLE label
        MOVWF   F0D             ; DPH (F0D.7~5) = 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

        ORG     280H
TABLE:
        .DT     0x1986          ; 14-bit ROM data
        .DT     0x3719, 0x2983  ; 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 (F03H)

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. The RAMBK bit is used to the SRAM Bank selection.

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

Bit	Description	
7	**GB0:** General Purpose Bit 0	
6	**GB1**: General Purpose Bit 1	
5	**RAMBK**: SRAM Bank Selection 0: SRAM Bank1 1: SRAM Bank1	
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

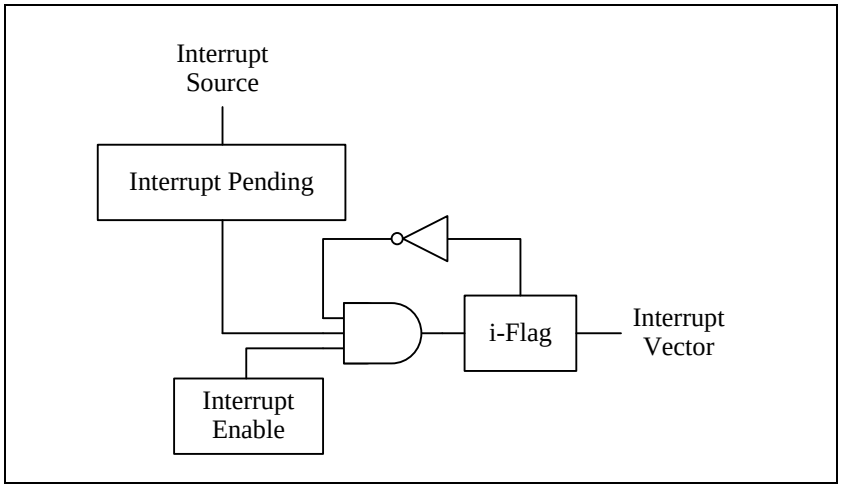
```
BTFSS   STATUS, 0       ; Check the C flag
GOTO    LABEL_1         ; If C = 0, goto LABEL_1 label
GOTO    LABEL_2         ; If C = 1, goto LABEL_2 label
```

1.6 Interrupt

The device 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 this device has only 1 vector, there is not an 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 INT1 (PA1) interrupt request with rising edge trigger

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

            ORG     001H            ; All interrupt vector
            GOTO    INT             ; If INT1 (PA1) input occurred rising edge

            ORG     002H
START:
            MOVLW   11111101B
            MOVWR   PAPUN           ; Enable INT1 (PA1) input pull up resistor.

            MOVLW   00001000B
            MOVWR   R0C             ; Set INT1 interrupt trigger as rising edge
            MOVLW   11111101B
            MOVWF   INTIF           ; Clear INT1 interrupt request flag
            MOVLW   00000010B
            MOVWF   INTIE           ; Enable INT1 interrupt
MAIN:
            …
            GOTO    MAIN

INT:
            MOVWF   20H             ; Store W data to SRAM 20H
            MOVFW   STATUS          ; Get STATUS data
            MOVWF   21H             ; Store STATUS data to SRAM 21H

            BTFSS   INT1IF          ; Check INT1IF bit
            GOTO    EXIT_INT        ; INT1IF = 0, exit interrupt subroutine
            …                       ; INT1 interrupt service routine
            MOVLW   11111101B
            MOVWF   INTIF           ; Clear INT1 interrupt request flag
EXIT_INT:
            MOVFW   21H             ; Get SRAM 21H data
            MOVWF   STATUS          ; Restore STATUS data
            MOVFW   20H             ; Restore W data
            RETI                    ; Return from interrupt
```

2 Chip Operation Mode

2.1 Reset

The device 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 three threshold levels can be selected. The LVR's operation mode is defined by the SYSCFG register.

There are two voltage selections for the LVR threshold level, one is higher level which is suitable for application with VDD is more than 3.5V or is fixed 5.0V, and the second one is suitable for application with VDD is wide voltage operating. See the following LVR Selection Table; user must also consider the lowest operating voltage of operating frequency.

LVR Selection Table:

LVR Threshold Level	Consider the operating voltage to choose LVR
LVR3.1	5.5V > V_{DD} > 3.5V or V_{DD}=5.0V
LVR2.2	V_{DD} is wide voltage range

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

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

2.2 System Configuration Register (SYSCFG)

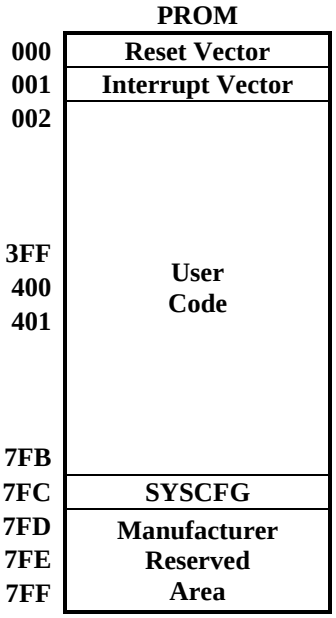
The System Configuration Register (SYSCFG) is located at ROM address 7FCh. The SYSCFG determines the option for initial condition of MCU. It is written by PROM Writer only. User can select LVR threshold voltage and chip operation mode by SYSCFG register. The default value of SYSCFG is 14'b11_111x_xxxx_xxxx. The 13th bit of SYSCFG is code protection selection bit. If this bit is 0, the data in PROM will be protected, when user reads PROM.

Bit	**13~0**	
Default Value	**11_1111_1111_1111**	
Bit	Description	
13	**PROTECT**: Code protection selection	
	0	Enable
	1	Disable
12-11	**LVR**: Low Voltage Reset Mode	
	00	LVR, Always Disable
	01	3.1V, Always Enable
	10	2.2V, Disable in STOP/IDLE mode for saving power
	11	2.2V, Always Enable
10	**XRSTE**: External Pin (PA7) Reset Enable	
	0	Disable (PA7 as IO pin)
	1	Enable
9	**WDTE**: WDT Reset Enable	
	0	Disable
	1	Enable
8-0	**Reserved**	

2.3 Program ROM (PROM)

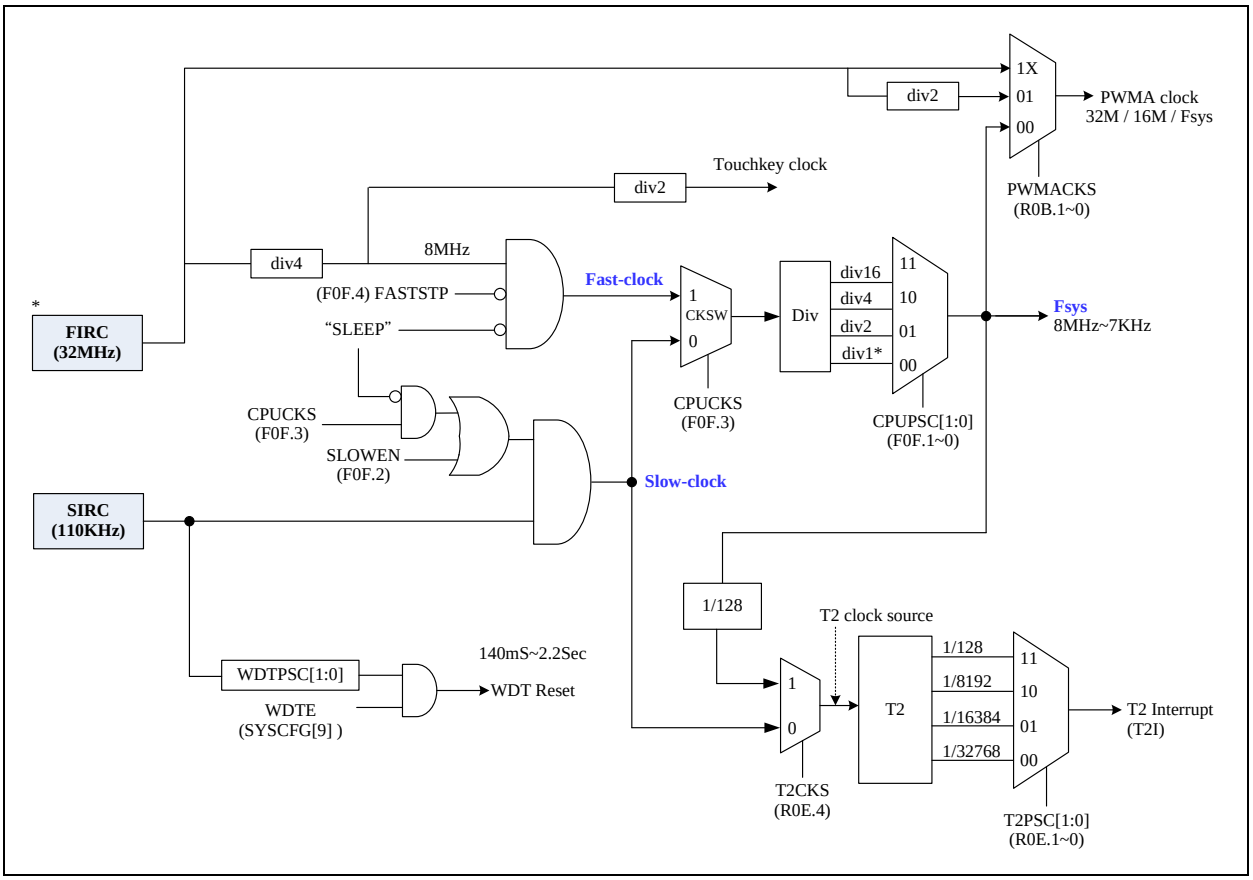
The PROM of this device is 2K words. The address from 7FC to 7FF is reserved by tenx. The address 7FC stores the system configuration bits (SYSCFG).

When PROTECT (SYSCFG.13) is 0, that mean protect is enabled. The address from 000 to 7FB will be protected.



2.4 Dual System Clock

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



Clock Scheme Block Diagram

FAST Mode:

The device enters FAST mode by setting the CPUCKS (F0F.3). In this mode, the program is executed using Fast-clock as system clock source. The Timer0, Timer1 block is driven by Fast-clock. The PWMA can be driven by Fsys, FIRC16MHz or FIRC 32MHz by setting PWMACKS (R0B.1~0).

SLOW Mode:

After power on or reset, the device enters SLOW mode first, the default Slow-clock is SIRC (~110 KHz).

IDLE Mode:

When SLOWEN (F0F.2) is set, the device 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.

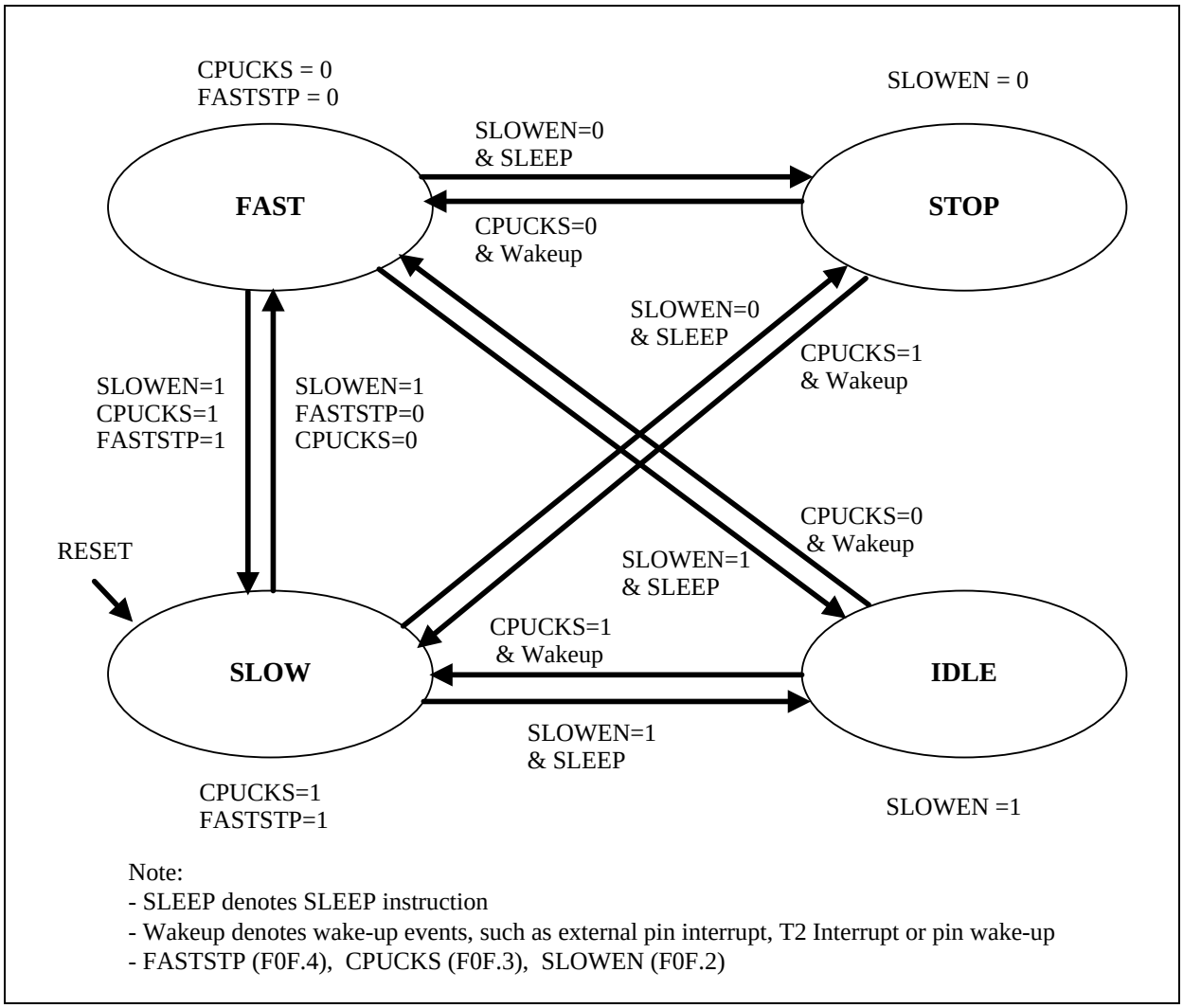
T2 is independent and has its own control registers. It is possible to keep T2 working and wake-up in the IDLE mode.

STOP Mode:

When SLOWEN (F0F.2) is set, all blocks will be turned off and the device will enter the STOP mode after executing the SLEEP instruction. The 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

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



CPU Operation Block Diagram

CPU Mode & Clock Functions Table:

Mode	Oscillator	Fsys	Fast-clock	Slow-clock	TM0	T2	PWMA	Wakeup event
FAST	FIRC	Fast-clock	Run	Set by SLOWEN bit	Run	Run	Run	X
SLOW	SIRC	Slow-clock	Set by FASTSTP bit	Run	Run	Run	Run	X
IDLE	SIRC	Stop	Stop	Run	Stop	Run	Stop	T2/IO
STOP	Stop	Stop	Stop	Stop	Stop	Stop	Stop	IO

- **FAST mode switches to SLOW mode**

If SLOWCKS is set, the source clock of Fsys is Slow-clock. The following steps are suggested to be executed by order when FAST mode switches to SLOW mode:

(1) Enable Slow-clock (SLOWEN=1)

(2) Switch system clock source (Fsys) to Slow-clock (CPUCKS=1)

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

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

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

- **SLOW mode switches to FAST mode**

If SLOWCKS is cleared, the source clock of Fsys is Fast-clock. The following steps are suggested to be executed by order when SLOW mode switches to FAST mode:

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

(2) Enable Slow-clock (SLOWEN=1)

(3) Switch system clock source (Fsys) to Fast-clock (CPUCKS=0)

(4) Stop Slow-clock (SLOWEN=0)

(5) Change to desired CPUPSC

Note: Item (4) 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
BSF     SLOWEN      ; Enable Slow-clock
NOP
BCF     CPUCKS      ; Switch Fsys to Fast-clock
BCF     SLOWEN      ; Disable Slow-clock
```

- **IDLE mode Setting**

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

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

IDLE mode can be waken up by external interrupts(INTx), T2, PD1 and PB[1:0] low level wakeup.

◇Example: Switch operating mode to IDLE mode

```
BSF     SLOWEN          ; Enable Slow-clock
MOVLW   01000001B       ;
MOVWR   R0E             ; T2 clock source = Slow-clock, T2PSC=1/16384
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) Stop WDT function (optional, if WDTE=1, set WDTSTP=1)
(3) Execute SLEEP instruction

STOP mode can be waken up by external interrupt (INTx), PD1 and PB[1:0] low level wakeup.

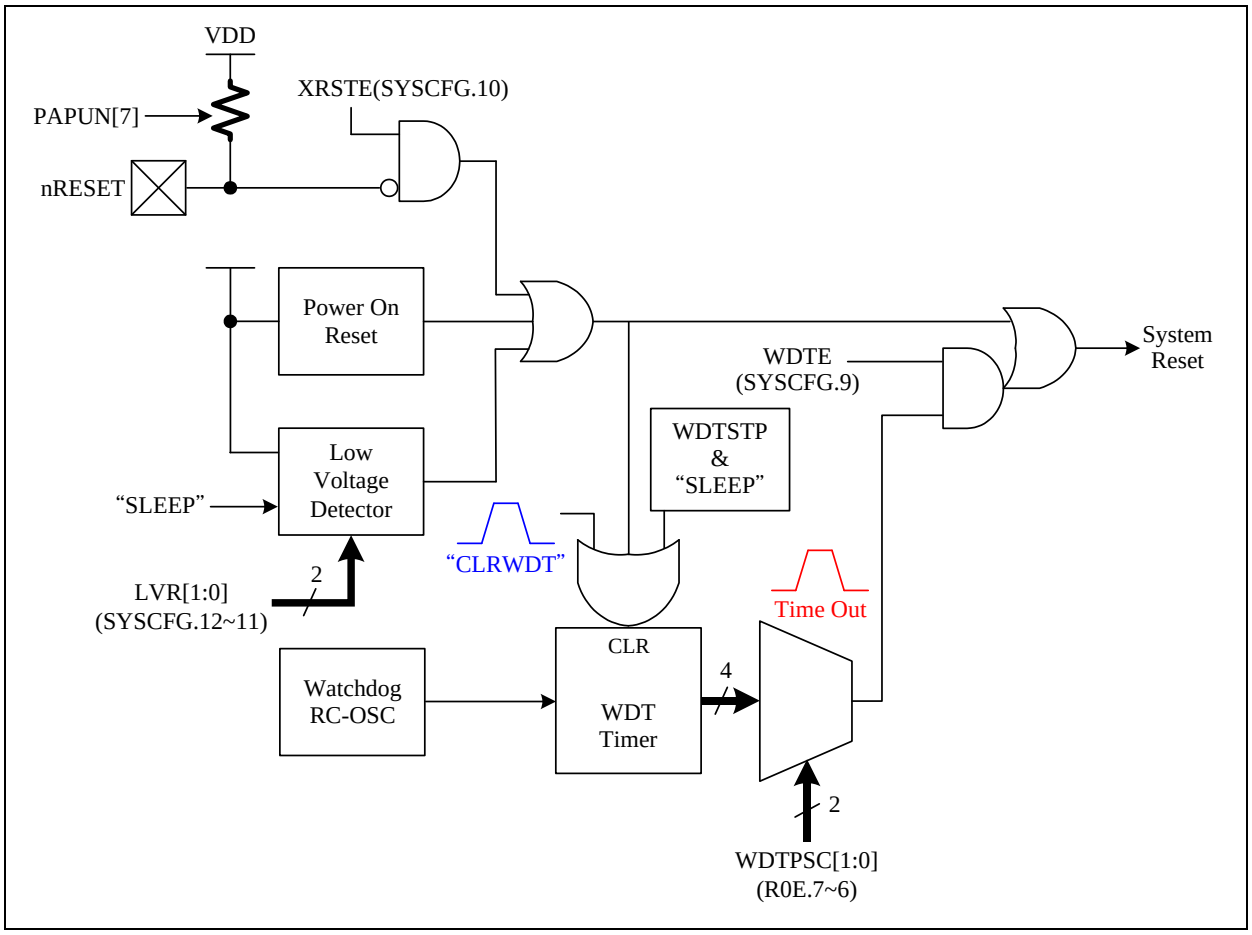
◇Example: Switch operating mode to STOP mode

```
BCF     SLOWEN          ; Disable Slow-clock
SLEEP                   ; Enter STOP mode
```

3 Peripheral Functional Block

3.1 Watchdog (WDT) Timer

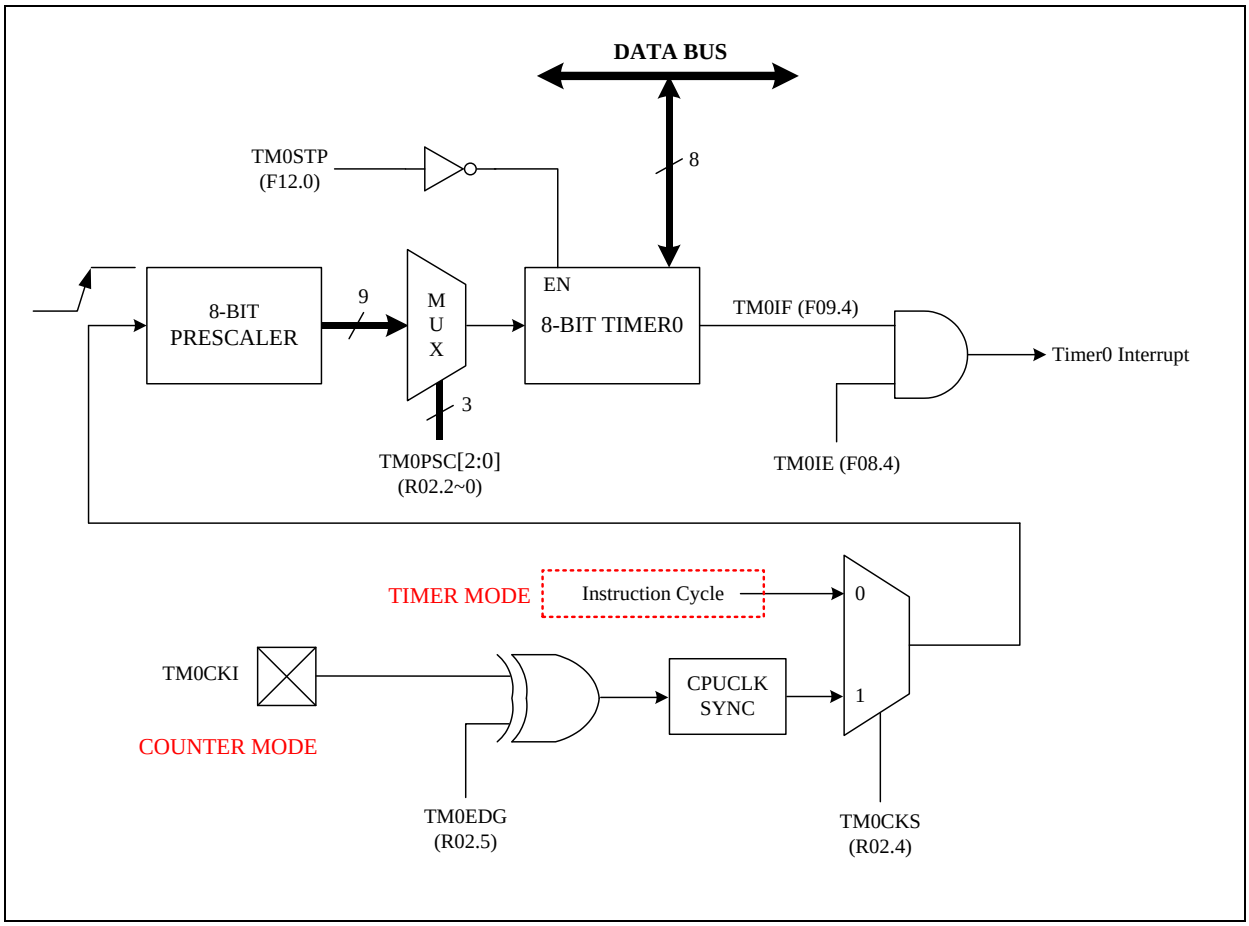
The WDT clock source is internal RC Timer. It is enabled by setting the WDTE (SYSCFG[9]). The overflow period of WDT can be selected from WDTPSC[1:0]. The WDT timer is cleared by the CLRWDT instruction. If the Watchdog is enabled (SYSCFG[9]=1) and WDT timer is overflow, the WDT will generates chip reset signal. Set WDTSTP (R0E.5)=1 can let WDT timer stop counting to saving power during IDLE/STOP mode. In the other words, WDTSTP=0 WDT timer is always keep counting even if the SLEEP instruction is executed.



WDT Block Diagram

3.2 Timer0

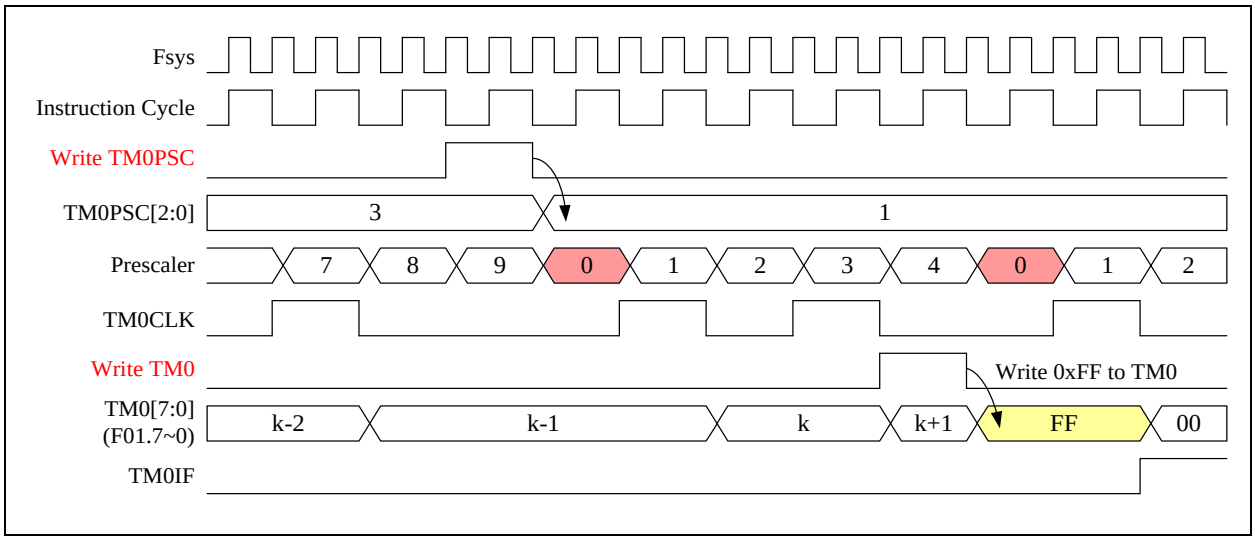
The Timer0 is an 8-bit wide register of F-Plane. It can be read or written as any other register of F-Plane. Besides, Timer0 increases itself periodically and automatically rolls over based on the pre-scaled clock source, which can be the instruction cycle or TM0CKI (PA2) rising/falling input. The Timer0's increasing rate is determined by the TM0PSC[2:0] (R02.2~0). 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 (F12.0) bit is set. TM0OUT is an output signal that toggles when Timer0 overflows.



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:

Timer0 interrupt interval cycle time = Instruction cycle time / TM0PSC / (256-TM0)

◇Example: Setup Timer0 work in Timer mode

```
; Setup TM0 clock source and divider
        MOVLW   00000101B       ; TM0CKS = 0, Timer0 clock is instruction cycle
        MOVWR   TM0CTL          ; TM0PSC = 101b, divided by 32

; Setup TM0 timer
        BSF     TM0STP          ; Stop Timer0 counting
        MOVLW   156
        MOVWF   TM0             ; Write 156 into TM0

; Enable Timer0 and interrupt function
        MOVLW   1110 1111B      ; Clear Timer0 request interrupt flag by byte
        MOVWF   INTIF           ; byte operation

        BSF     TM0IE           ; Enable Timer0 interrupt function
        BCF     TM0STP          ; Enable Timer0 counting
```

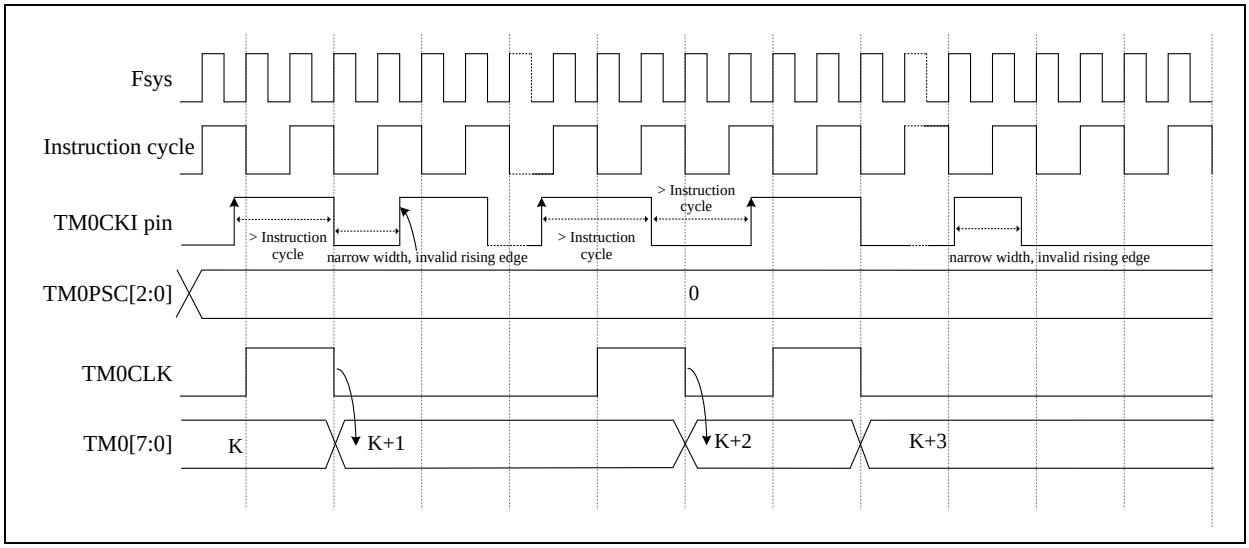
Fsys = 4MHz

Timer0 clock source is Fsys/2 = 4 MHz / 2 = 2 MHz, Timer0 divided by 32

Timer0 interrupt frequency = 2 MHz / 32 / (256-156) = 625Hz

● 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.



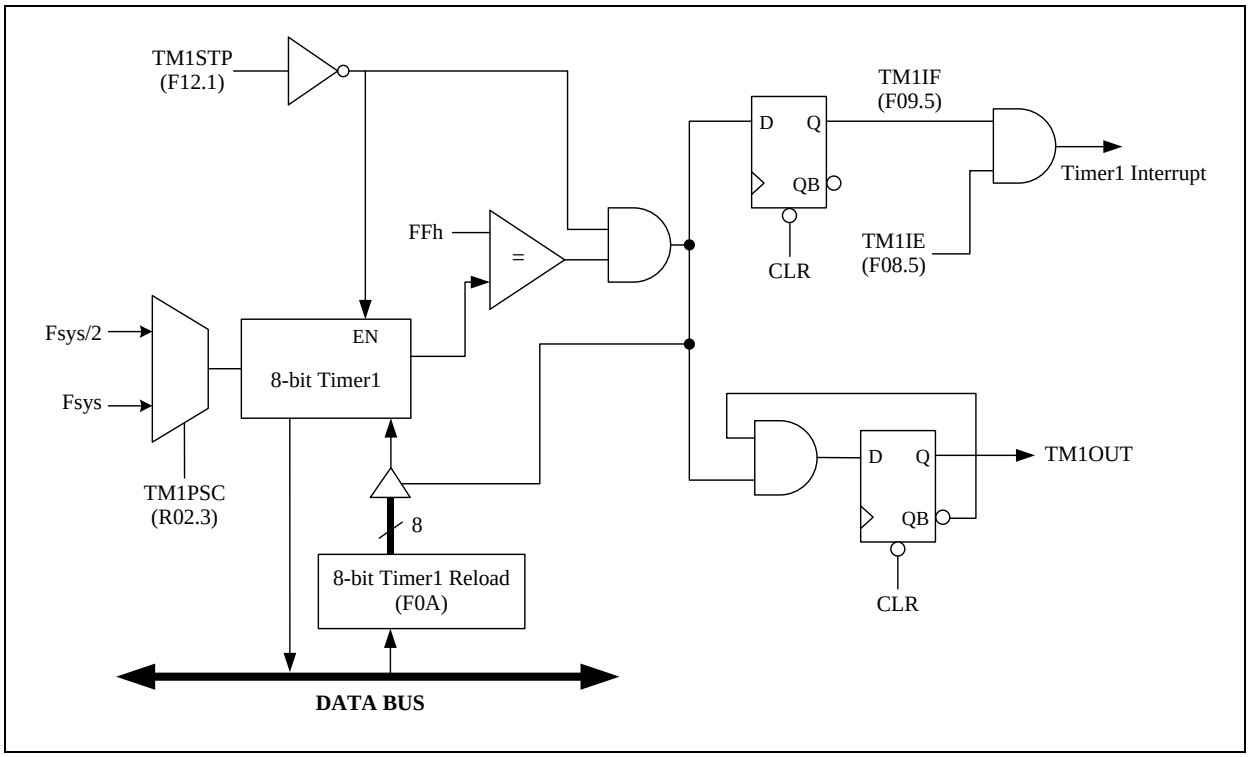
Timer0 works in Counter mode for TM0CKI. (TM0CKS=1, TM0EDG=0)

◇Example: Setup TM0 works in Counter mode and clock source from TM0CKI

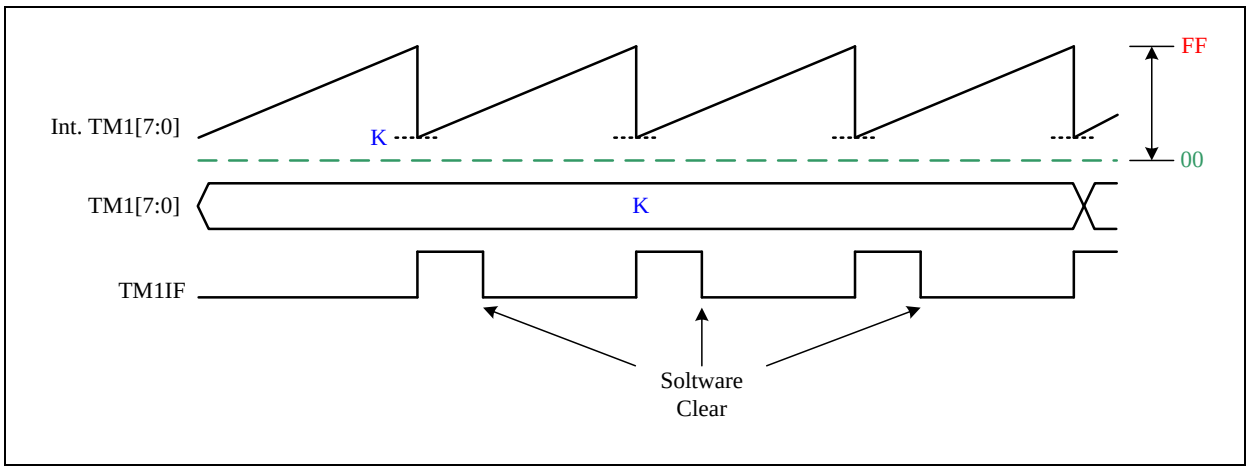
```
; Setup TM0 clock clock source and divider
        MOVLW   00110000B       ; TM0EDG=1, TM0 prescaler counting falling edge
        MOVWR   TM0CTL          ; TM0CKS=1, TM0 is TM0CKI (PA2)
                                ; TM0PSC=000b, divided by 1

; Setup TM0 timer and stop TM0 counting
        BSF     TM0STP          ; Stop TM0 counting
        MOVLW   00H
        MOVWF   TM0             ; Write 0 into TM0 register

; Enable TM0 counting and read TM0 counter
        BCF     TM0STP          ; Enable TM0 counting
        …
        BSF     TM0STP          ; Stop Timer0 counting
        MOVFW   TM0             ; Read Timer0 content
```

3.3 Timer1

The Timer1 is an 8-bit wide register of F-Plane. It can be read or written as any other register of F-Plane. Besides, Timer1 increases itself periodically and automatically reloads a new "offset value" (TM1) while it rolls over based on the pre-scaled instruction clock. The Timer1 increase rate is determined by TM1PSC (R02.3). Set the TM1STP (F12.1) will stop Timer1 counting. TM1OUT is an output signal that toggles when Timer1 overflow.



Timer1 Block Diagram



Timer1 Reload Diagram

◇Example: Setup TM0 work in Timer mode and counting overflow toggle out to TM1OUT(PD0) configuration.

```
; Setup TM1 clock source, divider and enable TM1OUT
        MOVLW   10000000B
        MOVWR   R02             ; R02.3=0 (TM1PSC), Select TM1 clock = Fsys/2.
                                ; R02.7=1, Enable TM1OUT function pin (PD0).

; Set TM1 timer offset and stops TM1 counting
        BSF     TM1STP          ; Stop TM1 counting
        MOVLW   F0H
        MOVWF   TM1             ; Write F0H into TM1 counter (F0A, F-Plane)

; Enable TM0 timer and interrupt function.
        MOVLW   11011111B       ; Clear TM1 request interrupt flag by byte operation
        MOVWF   INTIF           ; F-Plane 09H

        MOVLW   00100000B       ; Enable TM1 interrupt function.
        MOVWF   INTIE

        BCF     TM1STP          ; Enable TM1 counting
```

Example:

Fsys=2 MHz, TM1PSC=0, TM1 clock source = Fsys/2 = 1 MHz

TM1=0xF0,

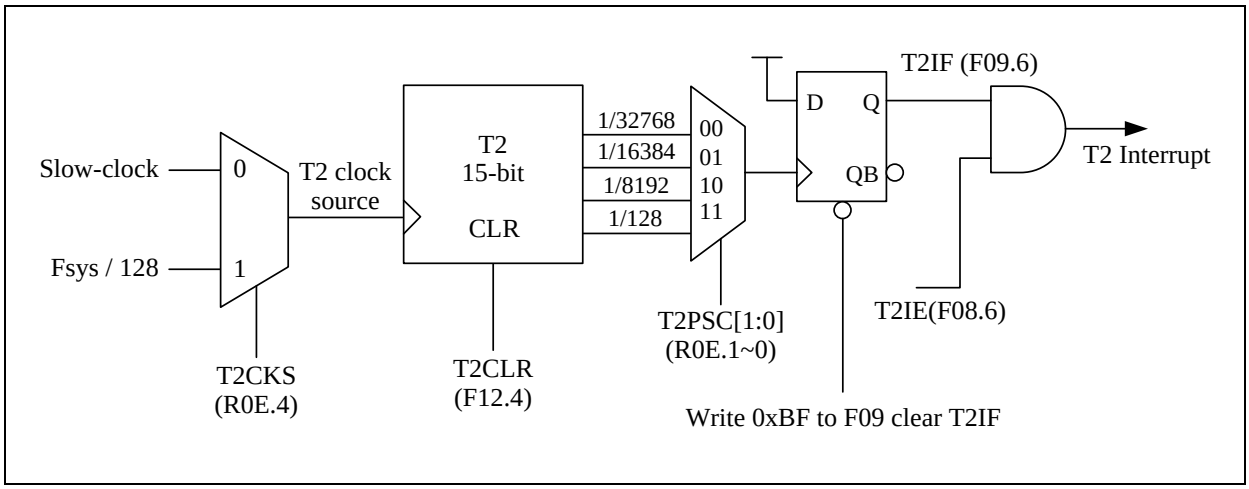
TM1 interrupt time = (1/1 MHz) * (0xFF – 0xF0) = 1 us*16 = 16 us

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

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

3.4 T2: 15-bit Timer

The T2 is a 15-bit counter and the clock sources are from either Fsys/128 or Slow-clock. The clock source is used to generate time base interrupt and T2 counter block clock. 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] (R0E.1~0) bits. The following figure shows the block diagram of T2.



T2 Block Diagram

◇Example: If CPU is running at FAST mode, Fsys=Fast-clock=FIRC4MHz. Set T2 clock source is Fsys/128 and divided by 16384

```
; Setup T2 clock source and divider,
        MOVLW   01010001B       ; T2 clock source = Fsys/128
        MOVWR   R0E             ; T2PSC = 01b, divided by 16384
        BSF     T2CLR           ; T2CLR = 1, clear and stop T2 counting

; Enable T2 interrupt function
        MOVLW   10111111B       ; Clear T2 interrupt flag by byte operation
        MOVWF   INTIF           ;

        BSF     T2IE            ; Enable T2 interrupt function

        BCF     T2CLR           ; T2CLR = 0, enable T2 counting
```

T2 clock source is Fsys/128= 4MHz/128= 31250 Hz, T2PSC = 1/16384

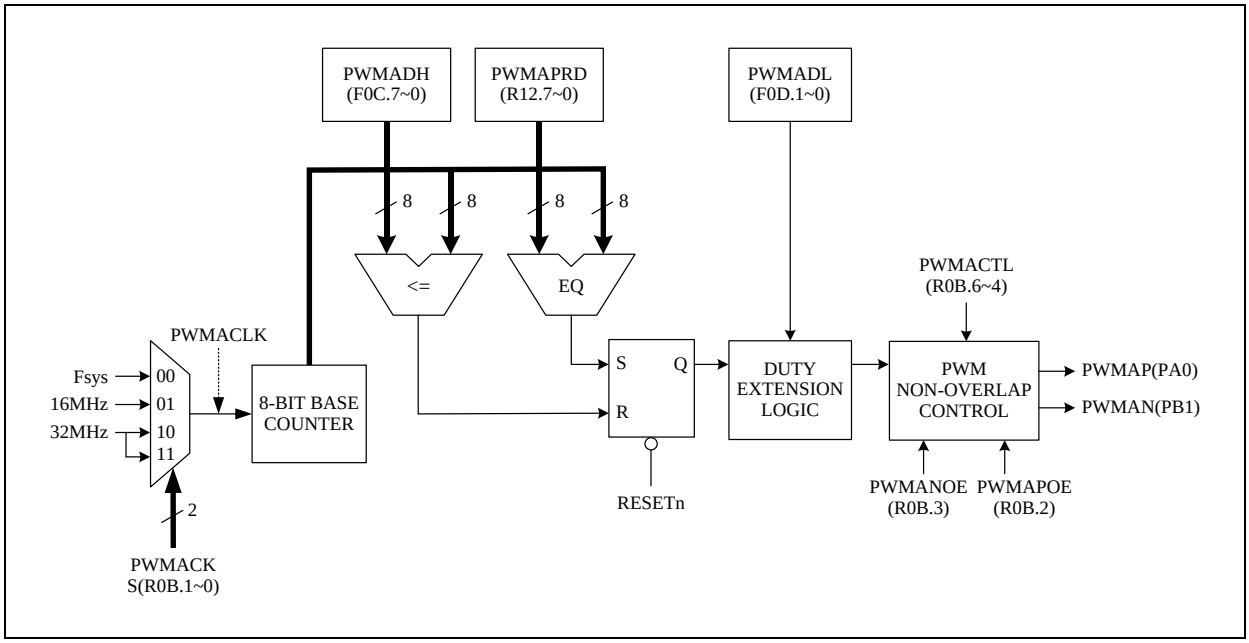
T2 interrupt frequency = 31250 / 16384 = 1.9 Hz

T2 interrupt period = 1 / 1.9 Hz = 0.524s

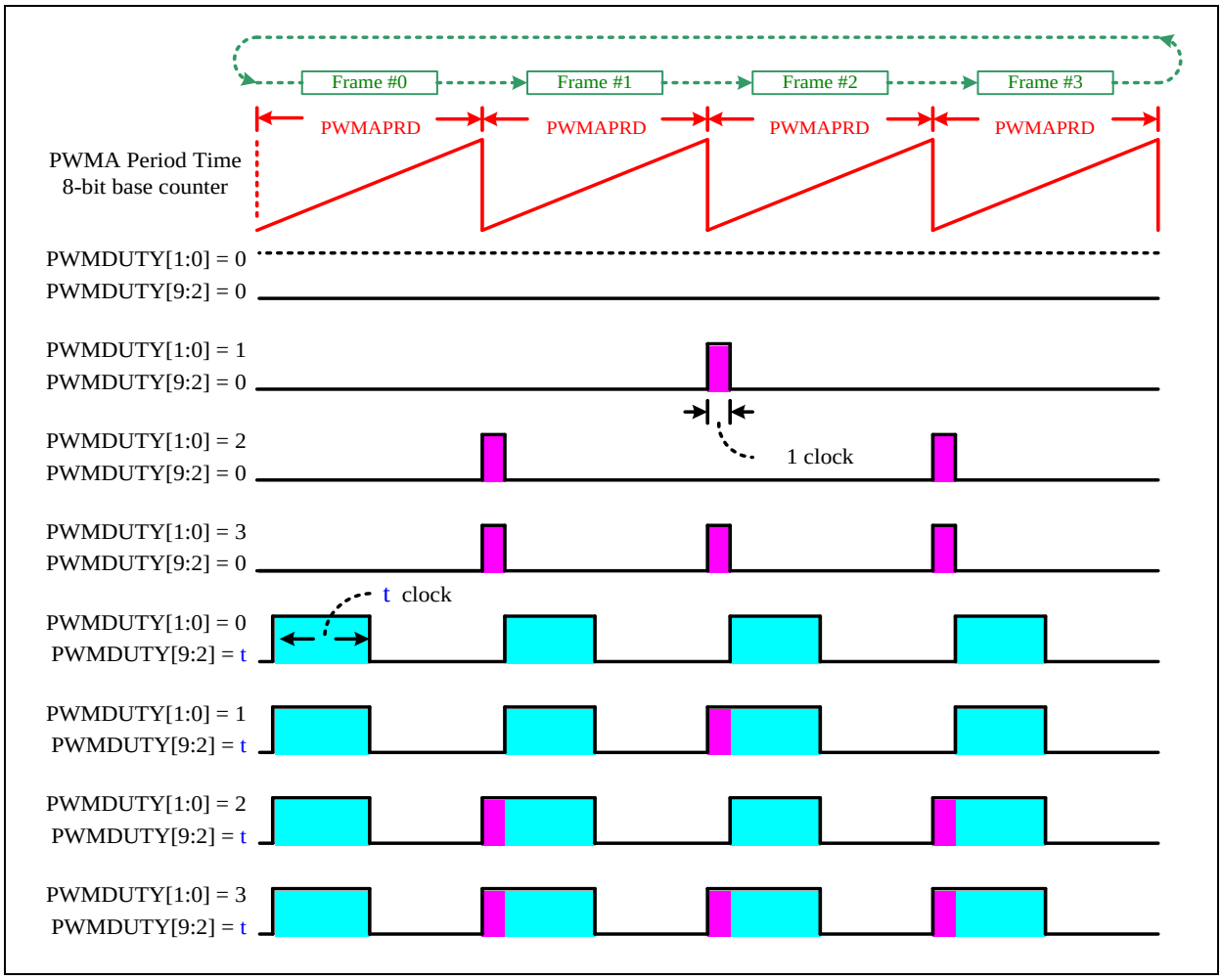
3.5 PWMA: (8+2) bits PWM

The PWM clock source (PWMACLK) can be chosen by PWMCKS (R0B.1~0) bits. A spread LSB technique allows PWM to run its frequency at "System Clock divided by 256" instead of "System Clock divided by 1024", which means the PWM is 4 times faster than normal. The advantage of higher PWM frequency is that the post RC filter can transform the PWM signal to more stable DC voltage level. The PWM output signal reset to low level whenever the 8-bit base counter matches the 8-bit MSB of PWM duty register PWMADH (F0C.7~0). When the base counter rolls over, the 2-bit LSB of PWM duty register PWMADL(F0D.1~0) decides whether to set the PWM output signal high immediately or set it high after one clock cycle delay.

The PWM period can be set by writing period value to PWMAPRD register (R12.7~0). Note that changing the PWMAPRD is immediately changing the PWMAPRD values, which are different from PWM0DH /PWM0DL which has buffer to update the duty at the end of current period. The programmer must pay attention to the current time to change PWMAPRD by observing the following figure. There is a digital comparator that compares the PWM0 counter and PWMAPRD, if PWM counter is larger than PWMAPRD after setting the PWMAPRD, a fault long PWM cycle will be generated because PWM counter must count to overflow then keep counting PWMAPRD to finish the cycle.

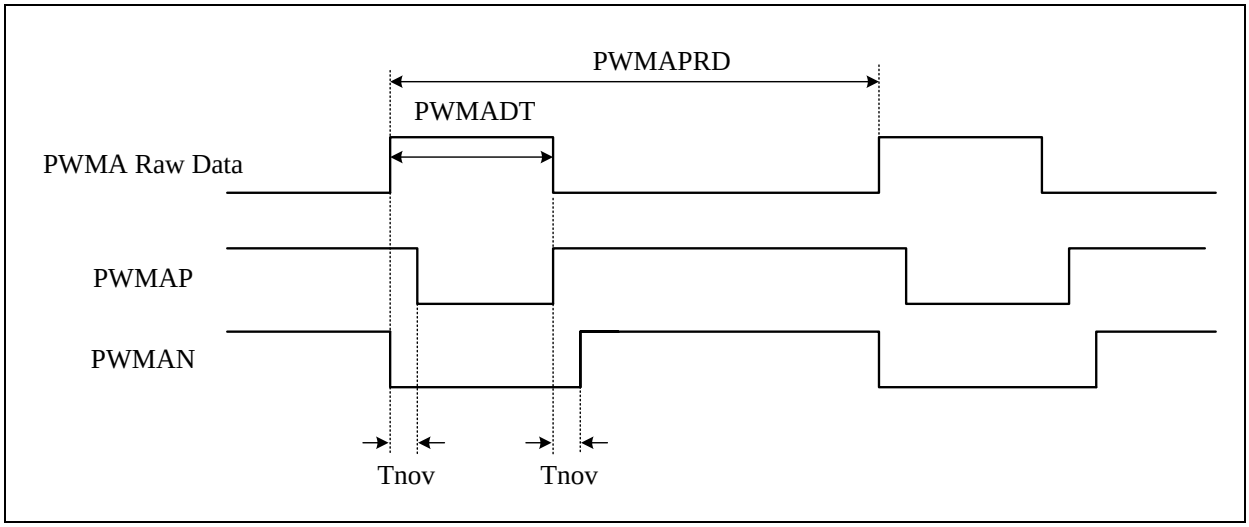


PWM Block Diagram



PWM Timing Diagram

PWM can be output via PWMAP and PWMAN with complementary mode. The edges of the PWMAP/PWMAN pulse can be non-overlap clocks intervals (Tnov), 0s ~ 10*PWMACLK clocks which are select by PWMANOV (R0B.6~4). The waveform is shown below.



PWM Complementary Output

◇Example: CPU is running at FAST mode, Fsys = 4MHz

```
MOVLW   00101101B       ; Tnov=4* (1/FPWMACLK), FPWMACLK = 16MHz
MOVWR   PWMACTL         ; PWMANOE enable , PWMAPOE enable

MOVLW   63H
MOVWR   PWMAPRD         ; Set PWMA period = 63H + 1 = 100

MOVLW   14H
MOVWF   PWMADH          ; Set PWMA duty hi byte = 14H = 20

MOVLW   00H
MOVWF   F0D             ; Set PWMA duty  2bit LSB =00H
```

PWMADL=00H, PWMADH=14H, PWMAPRD=63H, PWMACKS=01 ($F_{PWMACLK}$ = 16 MHz)

Tnov = 4/16 MHz = 250 ns (PWMANOV=010)

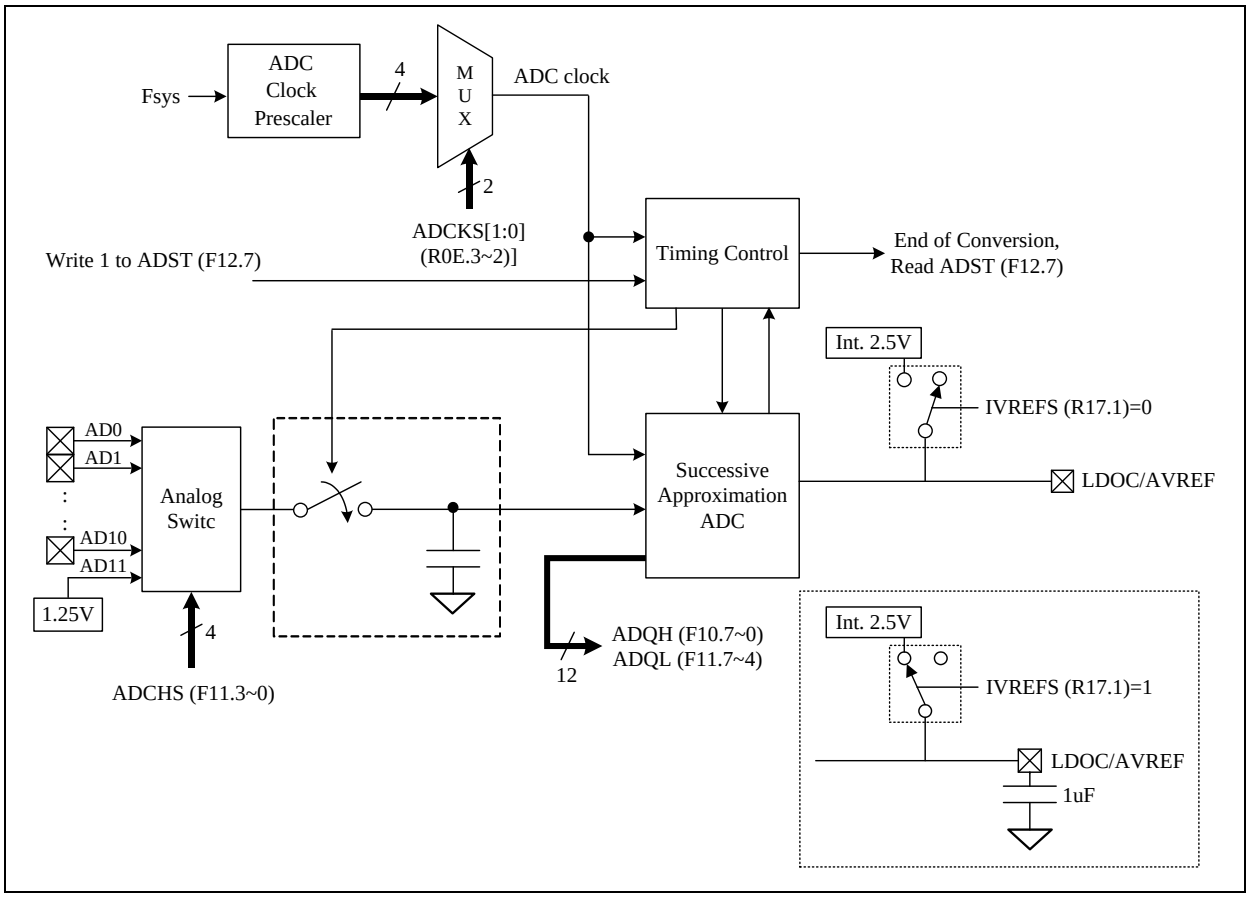
PWMA Raw Data output period = 16 MHz / (PWMAPRD+1) = 160 KHz

Raw Data output duty = PWMADT / (PWMAPRD + 1) = 20 / (99 + 1) = 20%

PWMAP low duty = PWMADT time – (Tnov)

PWMAN high duty = PWMAPRD time – PWMADT time – (Tnov)

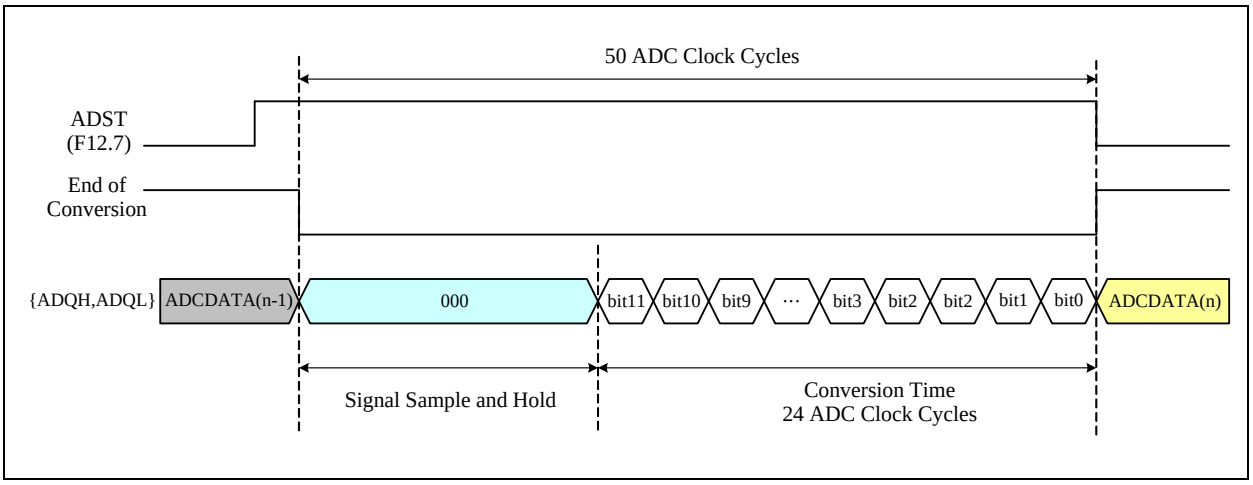
3.6 Analog-to-Digital Converter



ADC block diagram

The 12-bit ADC (Analog to Digital Converter) consists of a 12-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(R0E.3~4) to choose a proper ADC clock frequency, which must be less than 1 MHz. User then launches the ADC conversion by setting the ADST(F12.7) control bit. After end of conversion, H/W automatic clears the ADST(F12.7) bit. User can poll this bit to know the conversion status. The ADPIE8(R14.4~0),ADPIE(R13.7~0) control registers are used for ADC pin configuration, user can write the corresponding bit to "0" when the pin is used as an ADC input. The setting can disable the pin logical input path to save power consumption.

The LDOC/AVREF pin is multi-function, one is as LDOC pin, another is as AVREF pin. Afer power-on reset the default is AVREF function. Set IVREFS (R17.1)=1 to use internal voltage 2.5V as ADC reference voltage and LDOC pin should connect 1uF or above to ground, ADC input swing should be limited in 0~2.5V. Set IVREFS (R17.1)=0 to use external voltage as ADC reference voltage, in this case, AVREF pin usually connect to external referenced voltage(TL431) or VDD power.



Example:

[CPU running at FAST mode , Fsys = FIRC 4MHz]

ADC clock frequency = 1MHz, ADC channel = ADC2 (PA2).

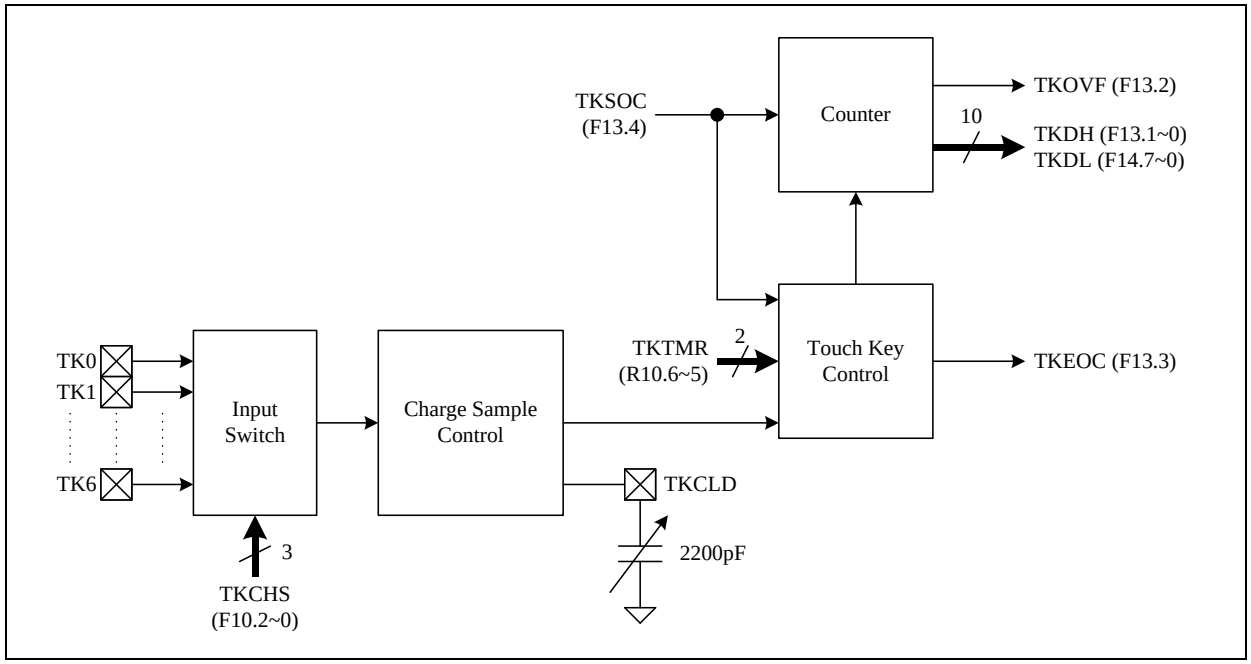
◇Example:

```
MOVLW   01001100B   ; Fsys = 4 MHz
MOVWR   R0E         ; R0E.3~2(ADCKS) = ADC clock = Fsys/4 = 1 MHz

MOVLW   00000100B   ; ADC2(PA2) pull-high disable
MOVWR   PAPUN;

MOVLW   11111011B   ; R13, Disable ADC2(PA2) digital input to saving power in
MOVWR   ADPIE       ; STOP/IDLE Mode

MOVLW   00000010B
MOVWF   ADCTL       ; F11.3~0 (ADCHS[3:0])= 2, ADC select ADC2 (PA2 pin).

BSF     ADST        ; F12.7 (ADST), ADC starts conversion.

WAIT_ADC:

BTFSC   ADST        ; Wait ADC conversion finish.
GOTO    WAIT_ADC

MOVFW   ADQH        ; F10.7~0, Read ADCQ[11:4] into W
MOVFW   ADCTL       ; F11.7~4, Read ADCQ[3:0] into W
```

3.8 Touch Key

The Touch Key offers an easy, simple and reliable method to implement finger touch applications. For most applications, only requires an external capacitor component on TKCLD pin.

Setting the TKSOC (F13.4) bit to start touch key conversion, the TKSOC bit will be cleared by H/W while end of conversion. “TKEOC=0” means conversion is in process, while “TKEOC=1” means the conversion is finish. After TKEOC’s (F13.3) edge rising, user must wait at least 10 us for next conversion. The touch key counting value is stored into TKDATA[9:0] (TKDH, TKDL). If TKOVF=1, it means the conversion has exceeded in period time, reduce TKTMR (R10.6~5) to fit the range of TKDATA[9:0]. On the other hand, if TKOVF=0, but TKDATA[9:0] is too small, increase TKTMR to adapting the system board circumstances. The more detailed information, refer to touch key application note.



Touch Key Block Diagram

◇Example: CPU running at FAST mode, Touch key channel = TK0 (PA3)

```
        MOVLW   11111110B       ; PD0/TKCLD  for touch key input
        MOVWR   PDE
        MOVLW   11110111B       ; PA3/TK0  for touch key input
        MOVWR   PAE

        MOVLW   00000001B       ; PD0 pull-high disable, also need an external
        MOVWR   PDPUN           ; Cap to ground
        MOVLW   00001000B
        MOVWR   PAPUN           ; PA3 pull-high disable

        MOVLW   11100111B       ; Disable PD0/PA3 digital input to saving power
        MOVWR   ADPIE8          ; in STOP/IDLE mode
                                ;
        MOVLW   01000000B       ; TKPD=0, TKTMR=2, TKCHS=0 (PA3/TK0)
        MOVWR   TKCTL           ;
        :
        BSF     TKSOC           ; Touch key start conversion
        NOP
        NOP
        NOP
        BCF     TKSOC

WAIT_TK:
        BTFSS   TKEOC           ; Wait touch key conversion finish
        GOTO    WAIT_TK

        MOVFW   TKDH            ; Read TKDATA[9:8]
        MOVFW   TKDL            ; Read TKDATA[7:0]
```

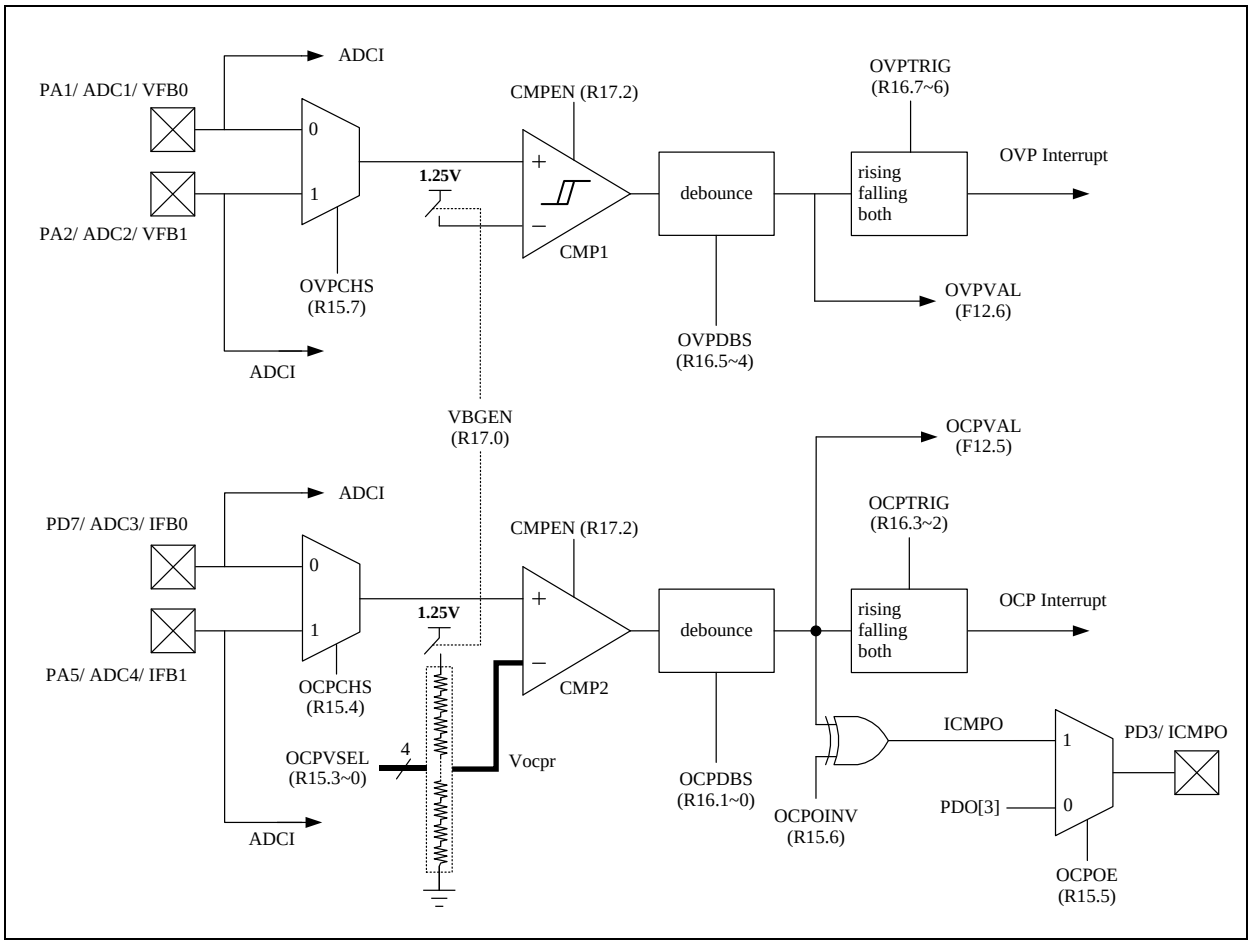
3.9 Over Current and Voltage Protection

The device supports the over voltage protection (OVP) and over current protection (OCP) mechanism.

Over Voltage Protection: CMP1 negative input (CMP1-) is fixed at 1.25V, CMP1 positive input (CMP1+) can be selected as VFB0 or VFB1 by OVPCHS (R15.7), when CMP1+ is greater than 1.25V, OVPVAL (F12.6) will generate the high state, set OVPDBS (R16.5~4)=3 can improve the system power noise interference. The OVPTRIG (R16.7~6) can monitor OVPVAL changed state and trigger interrupt events from high to low (falling) or low to high (rising) or level change (both).

Over Current Protection: The CMP2+ voltage (Vocpr) is 1.25V voltage divider, write OCPVSEL(R15.3) to select different voltage as CMP2- input voltge, set OCPDBS (R16.1~0)=3 is the best configuration to reduce system noise interference. PD3/ICMPO can directly output CMP2 compared result state by set OCPOE (R15.5)=1, the OCPOINV (R15.6) bit controls the output value need invert or not.

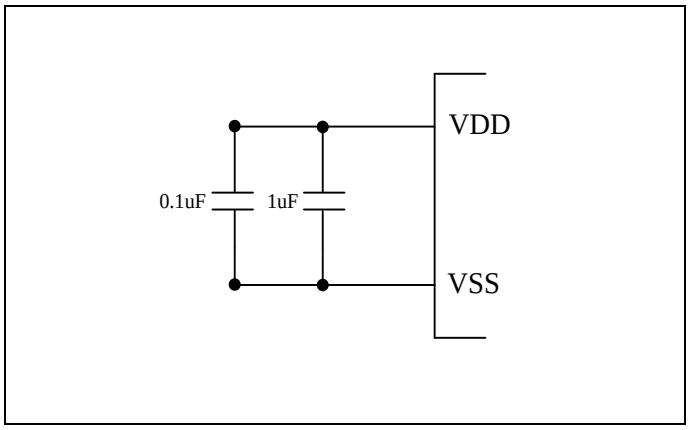
Set CMPEN (R17.2)=1 will turn on CMP1 and CMP2 function, the internal reference voltage also will automatically be enabled.



OVP / OCP block diagram

3.10 System Clock Osci

System clock can be operated in FIRC or SIRC. In the FIRC mode, the on-chip oscillator generates 8/4/2/0.5MHz system clock, which is configured by CPUPSC[1:0] (F0F.1~0). Since power noise degrades the performance of Fast Internal Clock Oscillator, placing power supply bypass capacitors 1 uF and 0.1 uF very close to VDD/VSS pins to improve the stability of clock and the overall system.

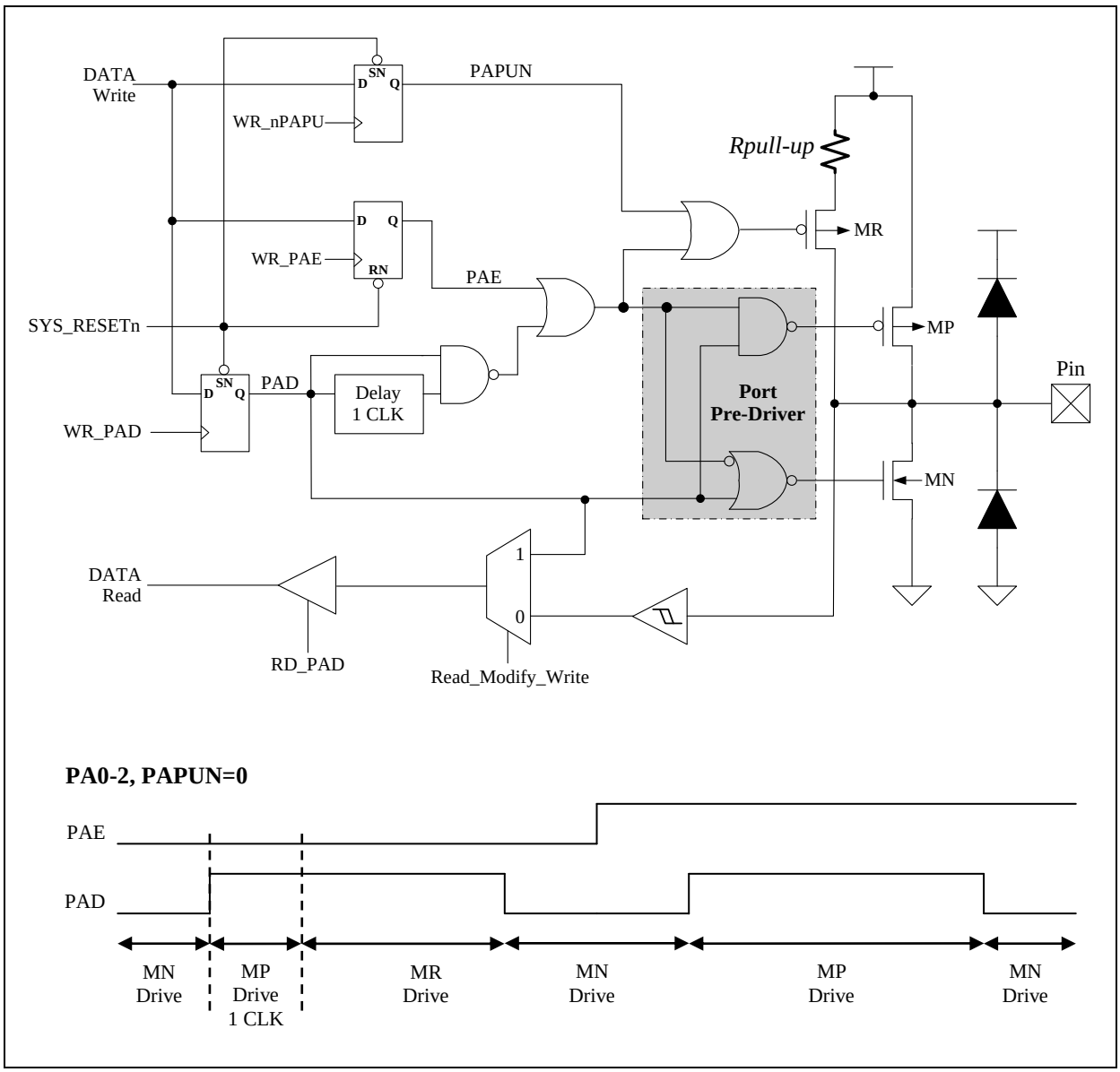


Fast Internal RC Mode

4 I/O Port

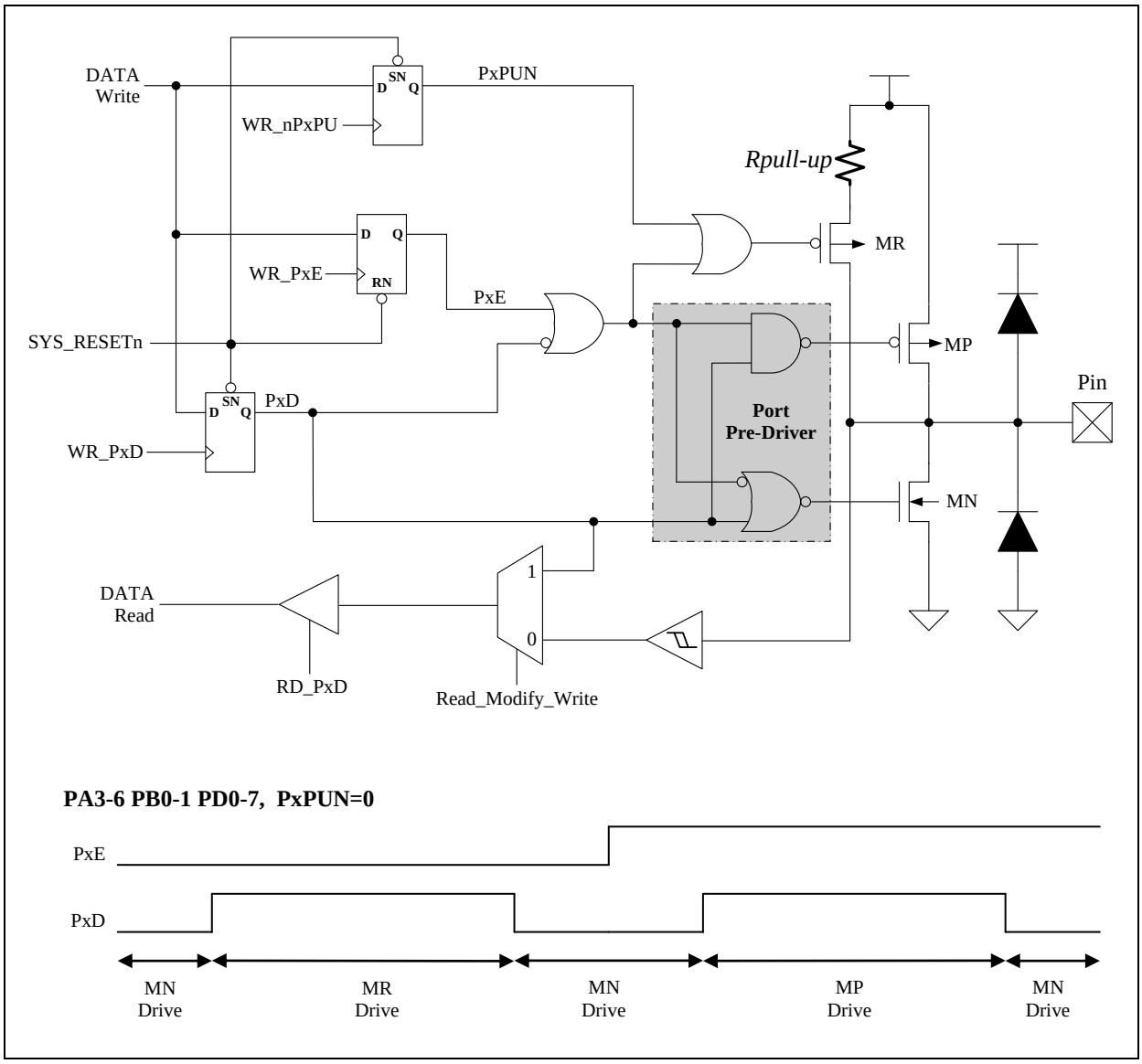
4.1 PA0-2

These pins can be used as Schmitt-trigger input, CMOS push-pull output or "pseudo-open-drain" output. 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 others instructions, CPU reads the pin state. The so-called "Read-Modify-Write" instruction includes BSF, BCF and all instructions using F-Plane as destination.



4.2 PA3-6, PD0-7, PB0-1

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.



◇Example: I/O mode selecting

```
MOVLW   FFH
MOVWF   PDD
MOVLW   00H
MOVWR   PDPUN   ; Set PD port pull-high enable
MOVLW   00H
MOVWR   PDE     ; Set all ports to be Schmitt-trigger input
```

◇Example: Set PA0-2 is pseudo-open-drain mode

```
MOVLW   xxxxx000B
MOVWR   PAE     ; Set PA2-PA0 as pseudo-open-drain mode

MOVLW   xxxxx000B
MOVWF   AD      ; PA2~PA0 output low level.
```

◇Example: Set PA0-2 is CMOS push-pull output mode.

```
MOVLW   xxxxx111B
MOVWR   PAE     ; Set PA2-PA0 as CMOS push-pull output mode
```

◇Example: Read data from input port.

```
MOVLW   FFH     ; “pseudo-open-drain” I/O structure, port must output High first
MOVWF   PDD     ; before read pin
MOVFW   PDD     ; Read data from Port D.
```

◇Example: Write data to output port.

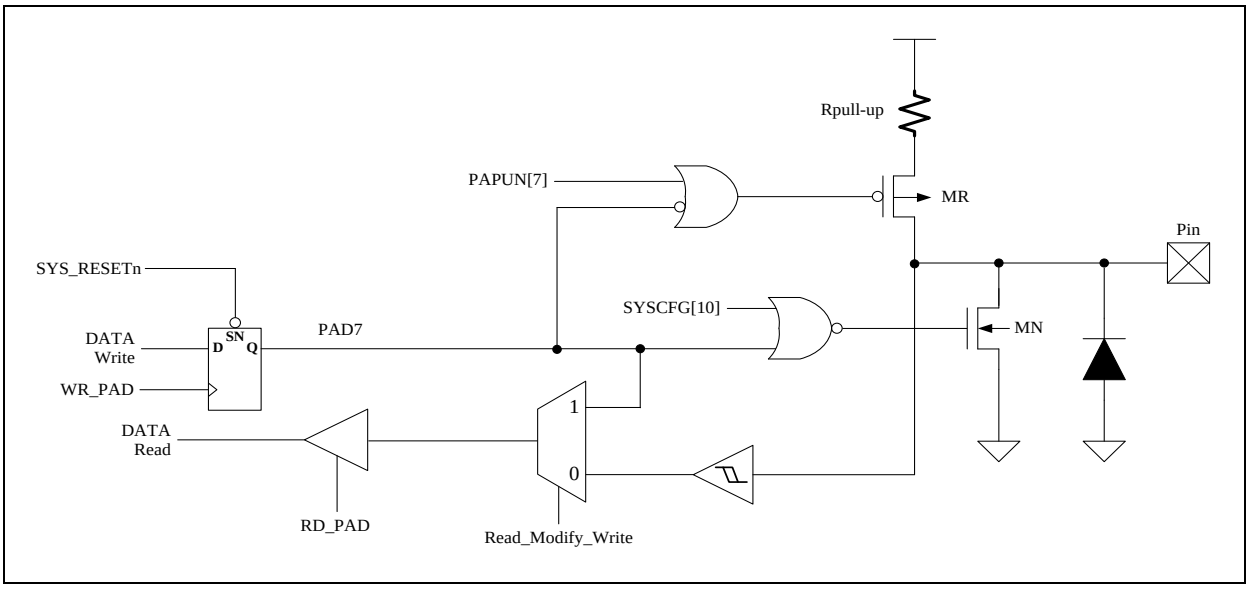
```
MOVLW   55H
MOVWF   PAD     ; Write data 55H to Port A.
MOVWF   PBD     ; Write data 55H to Port B.
```

◇Example: Write one bit data to output port.

```
BCF     PAD,0
BCF     PBD,1
BCF     PDD,2   ; Set PA0, PB1 and PD2 to be “0”.
BSF     PAD,3
BSF     PBD,4
BSF     PDD,7   ; Set PA3, PB4 and PD7 to be “1”.
```

4.3 PA7

PA7 can be used in Schmitt-trigger input or open-drain output which is setting 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 and PAD[7]=0, the pin state is output low and the pull-up resistor will be disabled automatically for power saving. When SYSCFG[10] is set, PA7 is only used in Schmitt-trigger input for external active low reset.



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

SYSCFG[10] (XRSTE)	Register Setting		Rpull-up	Pin State	Function
	PAPUN[7]	PAD[7]			
0	0	0	No	Low	Open-drain output low
		1	Yes	High	Open-drain output pull-high (default)
	1	0	No	Low	Open-drain output low
		1	No	Hi-Z	Open-drain output Hi-Z
1	0	0	No	Hi-Z	Reset input without pull-high
		1	Yes	High	Reset input with pull-high (default)
	1	0	No	Hi-Z	Reset input without pull-high
		1	No	Hi-Z	Reset input without pull-high

Note:

1) After power-on reset, the default state is PAPUN[7]=0, PAD[7]=1

2) If set SYSCFG[10]=1, pin state must avoid setting to Hi-Z state.

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 : TM0**	
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**	
GB0	03.7	R/W	0	General purpose bit 0
GB1	03.6	R/W	0	General purpose bit 1
RAMBK	03.5	R/W	0	SRAM Bank selection, 0: Bank0, 1: Bank1
TO	03.4	R	0	WDT time out flag, cleared by POR, 'SLEEP' or 'CLRWDT' instruction
PD	03.3	R	0	Sleep mode 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 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	04.7~0	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	05.6~0	R	-	Port A pin or “data register” state
		W	7F	Port A output data register
(F06) PBD			**Function related to : Port B**	
PBD	06.1~0	R	-	Port B pin or “data register” state
		W	3	Port B output data register
(F07) PDD			**Function related to : Port D**	
PDD	07.7~0	R	-	Port D pin or “data register” state
		W	FF	Port D output data register
(F08) INTIE			**Function related to : Interrupt Enable**	
OCPIE	08.7	R/W	0	OCP interrupt enable 0: disable 1: enable
T2IE	08.6	R/W	0	T2 interrupt enable 0: disable 1: enable
TM1IE	08.5	R/W	0	Timer1 interrupt enable 0: disable 1: enable
TM0IE	08.4	R/W	0	Timer0 interrupt enable 0: disable 1: enable
OVPIE	08.3	R/W	0	OVP interrupt enable 0: disable 1: enable
INT2IE	08.2	R/W	0	INT2 (PA7) pin interrupt enable 0: disable 1: enable
INT1IE	08.1	R/W	0	INT1 (PA1) pin interrupt enable 0: disable 1: enable
INT0IE	08.0	R/W	0	INT0 (PA6) pin interrupt enable 0: disable 1: enable

Name	Address	R/W	Rst	Description
(F09) INTIF				**Function related to : Interrupt Flag**
OCPIF	09.7	R	-	OCP interrupt event pending flag, set by H/W while OCP happens
		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
TM1IF	09.5	R	-	Timer1 interrupt event pending flag, set by H/W while Timer1 overflows
		W	0	0: clear this flag 1: no action
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
OVPIF	09.3	R	-	OVP interrupt event pending flag, set by H/W while OVP happens
		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
INT1IF	09.1	R	-	INT1 interrupt event pending flag, set by H/W at INT1 pin's falling/rising edge
		W	0	0: clear this flag 1: no action
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
(F0A) TM1				**Function related to : Timer1**
TM1	0a.7~0	R/W	0	Timer1 content
(F0C) PWMADH				**Function related to : PWMA**
PWMADH	0c.7~0	R/W	0	PWMA duty 8-bit MSB
(F0D) MF0D				**Function related to : PWMA / Table Read**
DPH	0d.7~5	R/W	0	Table read high address, data ROM pointer (DPTR) high byte
	0d.4~2			Reserved
PWMADL	0d.1~0	R/W	0	PWMA duty 2-bit LSB
(F0F) CLKS				**Function related to : CPUCLK**
	0f.7~5			Reserved
FASTSTP	0f.4	R/W	0	Fast-clock Enable/Disable 0: Enable 1: Disable
CPUCKS	0f.3	R/W	1	System clock (Fsys) selection 0: Fast-clock 1: Slow-clock
SLOWEN	0f.2	R/W	1	If CPUCKS=1, this bit configuration can be ignored. Slow-clock Enable/Disable 0: Disable 1: Enable
CPUPSC	0f.1~0	R/W	0	Fsys prescaler (Slow-clock/Fast-clock) 00: div1(110KHz/8MHz) 01: div2(50KHz/4MHz) 10: div4(28KHz/2MHz) 11: div16(7KHz/500KHz)
(F10) ADQH				**Function related to : ADC**
ADQH	10.7~0	R	-	ADC output data MSB, ADCQ[11:4]
(F11) ADCTL				**Function related to : ADC**
ADQL	11.7~4	R	-	ADC output data LSB, ADCQ[3:0]
ADCHS	11.3~0	R/W	F	ADC channel selection 0000: ADC0 (PA6) 0100: ADC4 (PA5) 1000: ADC8 (PD4) 0001: ADC1 (PA1) 0101: ADC5 (PB0) 1001: ADC9 (PD1) 0010: ADC2 (PA2) 0110: ADC6 (PD6) 1010: ADC10 (PD0) 0011: ADC3 (PD7) 0111: ADC7 (PD5) 1011~1111: 1.25V

Name	Address	R/W	Rst	Description
(F12) MF12				**Function related to : ADC / OVP / OCP / T2 / TM1 / TM0**
ADST	12.7	R/W	0	ADC start bit. 0: H/W clear after end of conversion 1: ADC start conversion
OVPVAL	12.6	R	-	OVP output value state
OCPVAL	12.5	R	-	OCP output value state
T2CLR	12.4	R/W	1	T2 counter clear 0: Counting 1: Clear and hold
	12.3~2			Reserved
TM1STP	12.1	R/W	0	Timer1 counter stop 0: Counting 1: Stop counting
TM0STP	12.0	R/W	0	Timer0 counter stop 0: Counting 1: Stop counting
(F13) TKDH				**Function related to : Touch key**
	13.7~5			Reserved
TKSOC	13.4	R/W	0	TK start of conversion, rising edge to start
TKEOC	13.3	R	1	TK end of conversion 1: end of conversion 0: converting in process
TKOVF	13.2	R	0	TK counter overflow 1: overflow 0: not overflow
TKDH	13.1~0	R	-	TK counter high byte TKDATA[9:8]
(F14) TKDL				**Function related to : Touch key**
TKDL	14.7~0	R	-	TK counter low byte TKDATA[7:0]
User Data Memory				
SRAM	20~2f	R/W	-	SRAM Common Area (16 Bytes)
	30~7f	R/W	-	SRAM Bank0 Area (RAMBK=0, 80 Bytes)
	38~7f	R/W	-	SRAM Bank1 Area (RAMBK=1, 80 Bytes)

R-Plane

Name	Address	R/W	Rst	Description
(R02) TM0CTL			**Function related to : Timer0**	
TM1OE	02.7	W	0	Enable Timer1 overflow toggle output to PD0 pin (TM1OUT)
TM0OE	02.6	W	0	Enable Timer0 overflow toggle output to PA5 pin (TM0OUT)
TM0EDG	02.5	W	0	Timer0 prescaler counting edge for TM0CKI pin 0: rising edge 1: falling edge
TM0CKS	02.4	W	0	Timer0 prescaler clock source selection 0: Instruction cycle (Fsys/2) 1: TM0CKI (PA2)
TM1PSC	02.3	W	0	Timer1 prescaler. Timer1 clock source is 0: Fsys/2 1: Fsys
TM0PSC	02.2~0	W	0	Timer0 prescaler. Timer0 prescaler clock source is divided by 000: 1/1 100: 1/16 001: 1/2 101: 1/32 010: 1/4 110: 1/64 011: 1/8 111: 1/128
(R03) PWRDN			**Function related to : POWER DOWN**	
PWRDN	03	W	-	Write this register to enter STOP/IDLE Mode (i.e. “SLEEP” instruction)
(R04) WDTCLR			**Function related to : WDT**	
WDTCLR	04	W	-	Write this register to clear WDT timer (i.e. “CLRWDT” instruction)
(R05) PAE			**Function related to : Port A**	
PAE	05.6~3	W	0	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	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
(R06) PBE			**Function related to : Port B**	
PBE	06.1~0	W	0	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
(R07) PDE			**Function related to : Port D**	
PDE	07.7~0	W	0	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
(R08) PAPUN			**Function related to : Port A**	
PAPUN	08.7~0	W	7F	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) c. the pin is working for PWM output 1: the pin pull up resistor is disabled
(R09) PBPUN			**Function related to : Port B**	
PBPUN	09.1~0	W	3	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 (PBD) is 0 b. the pin’s CMOS push-pull mode is chosen (PBE=1) c. the pin is working for PWM output 1: the pin pull up resistor is disabled

Name	Address	R/W	Rst	Description
(R0A) PDAPUN			**Function related to : Port D**	
PDPUN	0a.7~0	W	FF	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 (PDD) is 0 b. the pin's CMOS push-pull mode is chosen (PDE=1) c. pins are working for TM1OUT/TCOUT output 1: the pin pull up resistor is disabled
(R0B) PWMACTL			**Function related to : PWMA**	
	0b.7			Reserved
PWMANOV	0b.6~4	W	0	PWMA non-overlap clock 000: PWMACLK * 0 100:PWMACLK * 7 001: PWMACLK * 2 101:PWMACLK * 8 010: PWMACLK * 4 110:PWMACLK * 9 011: PWMACLK * 6 111:PWMACLK * 10 (620 ns @16 MHz)
PWMANOE	0b.3	W	0	1: enable PMWAN output to PB1
PWMAPOE	0b.2	W	0	1: enable PMWAP output to PA0
PWMACKS	0b.1~0	W	00	PMWA clock source (PWMACLK) 00: Fsys 01: 16 MHz 1x: 32 MHz
(R0C) MR0C			**Function related to : TCOUT / INT0 / INT1**	
TCOPSC	0c.7~6	W	0	TCOUT prescaler or TKCLK output 00: Fsys/2 01: Fsys/4 10: TKCLK 11: ~TKCLK
TCOE	oc.5	W	0	Enable post-prescaler TCOUT output to PD6 pin (TCOUT)
INT0EDG	0c.4	W	0	0: INT0(PA6) falling edge to trigger interrupt 1: rising edge
INT1EDG	0c.3	W	0	0: INT1(PA1) falling edge to trigger interrupt 1: rising edge
PDWKEN	0c.2	W	0	0: Disable PD1 low level to wakeup 1:enable
PBWKEN	0c.1~0	W	00	0: Disable PB[1:0] individual low level to wakeup 1:enable
(R0E) MR0E			**Function related to : TCOUT / INT0 / INT1**	
WDTPSC	0e.7~6	W	01	WDT period (not precision), @VDD=5V 00: 140 ms 01: 280 ms 10: 1100 ms 11: 2280 ms
WDTSTP	oe.5	W	0	WDT disable in IDLE / STOP mode, If WDTE=0, this bit is invalid 0: always counting 1: clear & stop counting
T2CKS	0e.4	W	0	"T2 clock source" selection 0: "Slow-clock" 1:Fsys/128
ADCKS	0e.3~2	W	00	ADC clock frequency selection: 00: Fsys/128 01: Fsys/64 10:Fsys/8 11:Fsys/4
T2PSC	0e.1~0	W	00	T2 prescaler. "T2 clock source" divided by - 00: 32768 01: 16384 10:8192 11:128
(R10) TKCTL			**Function related to : Touch key**	
TKPD	10.7	W	1	Touch key power down
TKTMR	10.6~5	W	10	Touch key conversion time. 0=shortest, 3=longest
	10.4~3			Reserved
TKCHS	10.2~0	W	000	Touch key channel select, TKCHS[2:0]= 000: TK0 (PA3) 100: TK4 (PD5/AD7) 001: TK1 (PD1/AD9) 101: TK5 (PD6/AD6) 010: TK2 (PD2) 110: TK6 (PB0/AD5) 011: TK3 (PD4/AD8) 111: Dummy Key
(R12) PMWAPRD			**Function related to : PWMA**	
PWMAPRD	12.7~0	W	FF	PWMA period data

Name	Address	R/W	Rst	Description
(R13) ADPIE				**Function related to : ADC / Touch key**
ADPIE	13.7	W	1	Each bit controls its corresponding port I/O enable pin, if the bit is 0: enable ADC7/ TK4 channel input 1: enable PD5 I/O digital input
	13.6	W	1	0: enable ADC6/ TK5 channel input 1: enable PD6 I/O digital input
	13.5	W	1	0: enable ADC5/ TK6 channel input 1: enable PB0 I/O digital input
	13.4	W	1	0: enable ADC4/ IFB1 channel input 1: enable PA5 I/O digital input
	13.3	W	1	0: enable ADC3/ IFB0 channel input 1: enable PD7 I/O digital input
	13.2	W	1	0: enable ADC2/ VFB1 channel input 1: enable PA2 I/O digital input
	13.1	W	1	0: enable ADC1/ VFB0 channel input 1: enable PA1 I/O digital input
	13.0	W	1	0: enable ADC0 channel input 1: enable PA6 I/O digital input
(R14) ADPIE8				**Function related to : ADC / Touch key**
ADPIE8	14.7~5			Reserved
	14.4	W	1	0: enable TK0 channel input 1: enable PA3 I/O digital input
	14.3	W	1	0: enable ADC10/ CLD channel input 1: enable PD0 I/O digital input
	14.2	W	1	0: enable ADC9/ TK1 channel input 1: enable PD1 I/O digital input
	14.1	W	1	0: enable TK2 channel input 1: enable PD2 I/O digital input
	14.0	W	1	0: enable ADC8/ TK3 channel input 1: enable PD4 I/O digital input
(R15) MR15				**Function related to : OVP / OCP**
OVPCHS	15.7	W	0	OVP positive input channel select 0: PA1/VFB0 1: PA2/VFB1
OCPOINV	15.6	W	0	ICMPO output inverter 0: Disable 1: Enable
OCPOE	15.5	W	0	OCP compare output to PD3 0: Disable 1: Enable
OCPCHS	15.4	W	0	OCP positive input channel selec 0: PD7/IFB0 1: PA5/IFB1
OCPVSEL	15.3~0	W	0	OCP voltage divider (Vocpr) 0000: 18 mV 0100: 83 mV 1000: 146 mV 1100: 208 mV 0001: 32 mV 0101: 98 mV 1001: 162 mV 1101: 226 mV 0010: 49 mV 0110: 113 mV 1010: 180 mV 1110: 242 mV 0011: 66 mV 0111: 130 mV 1011: 194 mV 1111: 255 mV
(R16) MR16				**Function related to : OVP / OCP**
OVPTRIG	16.7~6	W	1	OVP comparator interrupt trigger direction 00: rising 01: falling 10: both 11:N/A
OVPDBS	16.5~4	W	0	OVP comparator output debounce time 00: none 01: 2*Fsys 10: 4*Fsys 11: 8*Fsys(2uS@Fsys=4MHz)
OCPTRIG	16.3~2	W	1	OCP comparator interrupt trigger direction 00: rising 01: falling 10: both 11:N/A
OCPDBS	16.1~0	W	0	OCP comparator output debounce time 00: none 01: 4*Fsys 10: 8*Fsys 11: 16*Fsys(4uS@Fsys=4MHz)
(R17) MR17				**Function related to IVREF / Comparator / VBG**
	17.7~3			Reserved
CMPEN	17.2	W	0	OVP / OCP comparators enable 0: Disable 1: Enable
IVREFS	17.1	W	0	ADC AVREF power select 0: AVREF 1: Internal LDO 2.5V
VBGEN	17.0	W	0	VBG reference enable 0: Disable 1: Enable

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

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

COMF — Complement "f"

Syntax	COMF f [,d]
Operands	f : 00h ~ 7Fh, d : 0, 1
Operation	(destination) ← ($\bar{f}$)
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) ← (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) ← (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 ≠ 0, PC = LABEL1 + 1

GOTO — Unconditional Branch

Syntax	GOTO k
Operands	k : 000h ~ FFFh
Operation	PC.11~0 ← 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) ← (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) ← (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	LABEL1 INCFSZ 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	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

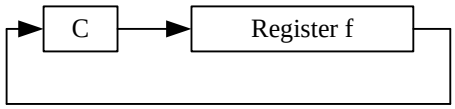
RETI	Return from Interrupt	
Syntax	RETI	
Operands	-	
Operation	PC ← TOS, GIE ← 1	
Status Affected	-	
OP-Code	00 0000 0110 0000	
Description	Return from Interrupt. Stack is POPed and Top-of-Stack (TOS) is loaded in to the PC. Interrupts are enabled. This is a two-cycle instruction.	
Cycle	2	
Example	RETI	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) ← (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	(destination) ← (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	(destination,7~4) ← (f.3~0), (destination.3~0) ← (f.7~4)	
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	

```
MOVLW   (TAB1&0xFF)
MOVWF   FSR                 ;Where FSR is F-Plane register
MOVLW   (TAB1>>8)&0xFF
MOVWF   DPH                 ;Were DPH is F-Plane register

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

        ORG 0234H
TAB1:
.DT     0x3789, 0x2277      ;ROM data 14bits
```

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	

```
MOVLW   (TAB1&0xFF)
MOVWF   FSR                 ;Where FSR is F-Plane register
MOVLW   (TAB1>>8)&0xFF
MOVWF   DPH                 ;Where DPH is F-Plane register

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

        ORG 0234H
TAB1:
.DT     0x3789, 0x2277      ;ROM data 14bits
```

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℃)

Parameter	Rating	Unit
Supply voltage	V_{SS} - 0.3 to V_{SS} + 6.5	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	5.5	V
Operating temperature	-40 to +85	℃
Storage temperature	-65 to +150	

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

Parameter	Symbol	Conditions		Min	Typ	Max	Unit
Operating Voltage	V_{DD}	FAST mode, Fsys = 8 MHz		2.4	–	5.5	V
		FAST mode, Fsys = 4 MHz		2.0	–	5.5	
		FAST mode, Fsys = 2 MHz		1.8	–	5.5	
		FAST mode, Fsys = 500KHz		1.6	–	5.5	
		SLOW mode, SIRC		1.6	–	5.5	
Input High Voltage	V_{IH}	All Input, except PA7	V_{DD} = 3~5V	$0.6V_{DD}$	–	–	V
		PA7	V_{DD} = 3~5V	$0.7V_{DD}$	–	–	V
Input Low Voltage	V_{IL}	All Input	V_{DD} = 3~5V	–	–	$0.2V_{DD}$	V
I/O Port Source Current	I_{OH}	All Output except below	V_{DD} = 5V, V_{OH} = 4.5V	4	8	–	mA
			V_{DD} = 3V, V_{OH} = 2.7V	2	4	–	
		PA0/PB1	V_{DD} = 5V, V_{OH} = 4.5V	20	24	–	
			V_{DD} = 3V, V_{OH} = 2.7V	8	12	–	
I/O Port Sink Current	I_{OL}	All Output, except below	V_{DD} = 5V, V_{OL} = 0.5V	10	20	–	mA
			V_{DD} = 3V, V_{OL} = 0.3V	5	10	–	
		PA7	V_{DD} = 5V, V_{OL} = 0.5V	15	30	–	
			V_{DD} = 3V, V_{OL} = 0.3V	6	12	–	
		PA0/PB1	V_{DD} = 5V, V_{OL} = 0.5V	30	38	–	
			V_{DD} = 3V, V_{OL} = 0.3V	14	18	–	
Input Leakage Current (pin high)	I_{ILH}	All Input	V_{IN} = V_{DD}	–	–	1	µA
Input Leakage Current (pin low)	I_{ILL}	All Input	V_{IN} = 0V	–	–	-1	
Pull-up Resistor	R_{UP}	All pins	VDD = 5V	–	60	–	KΩ
			VDD = 3V	–	120	–	
Supply Current (No Load)	I_{DD}	FAST mode	V_{DD} = 5V, Fsys = 8 MHz	–	3.8	–	mA
			V_{DD} = 5V, Fsys = 4 MHz	–	3.0	–	
			V_{DD} = 5V, Fsys = 2 MHz	–	2.8	–	
			V_{DD} = 5V, Fsys = 500K Hz	–	2.0	–	
		SLOW mode	V_{DD} = 5V, Fsys = 110 KHz	–	1	–	mA
		IDLE mode LVR enable	V_{DD} = 5V, Fsys = 110 KHz	–	12	–	µA
			V_{DD} = 3V, Fsys = 110 KHz	–	4	–	
		STOP mode LVR enable	V_{DD} = 5V	–	–	6	µA
			V_{DD} = 3V	–	–	2	
		STOP mode LVR disable	V_{DD} = 5V	–	–	1	
			V_{DD} = 3V	–	–	1	
LVR Reference Voltage	V_{LVR}	T_A = 25°C		2.0	2.2	2.4	V
				2.9	3.1	3.3	V
LVR Hysteresis Voltage	V_{HYST}	T_A = 25°C		–	± 0.1	–	V
Low Voltage Detection time	t_{LVR}	T_A = 25°C		100	–	–	µs

3. Clock Timing

Parameter	Condition	Min	Typ	Max	Unit
FIRC Frequency	V_{DD} = 3.0~5.0V, 25°C	Typ-3%	8	Typ+3%	MHz
	V_{DD} = 3.0~5.0V, -10°C ~ 70°C	Typ-5%	8	Typ+5%	
PWMAP/PWMAN Frequency	V_{DD} = 3.0~5.0V, 25°C	Typ-3%	32	Typ+3%	MHz
	V_{DD} = 3.0~5.0V, -10°C ~ 70°C	Typ-5%	32	Typ+5%	
SIRC Frequency	V_{DD} = 3.5~5.0V, -40°C ~ 85°C	Typ-20%	110	Typ+20%	KHz

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

Parameter	Conditions	Min	Typ	Max	Unit
RESET Input Low width	Input V_{DD} = 5 V ± 10 %	3	–	–	µs
WDT wakeup time	V_{DD} = 5V, WDTPSC = 2'b10	-20%	1100	+20%	ms
	V_{DD} = 3V, WDTPSC = 2'b10	-20%	1400	+20%	
CPU start up time	V_{DD} = 5V	–	19	–	ms
	V_{DD} = 3V	–	24	–	

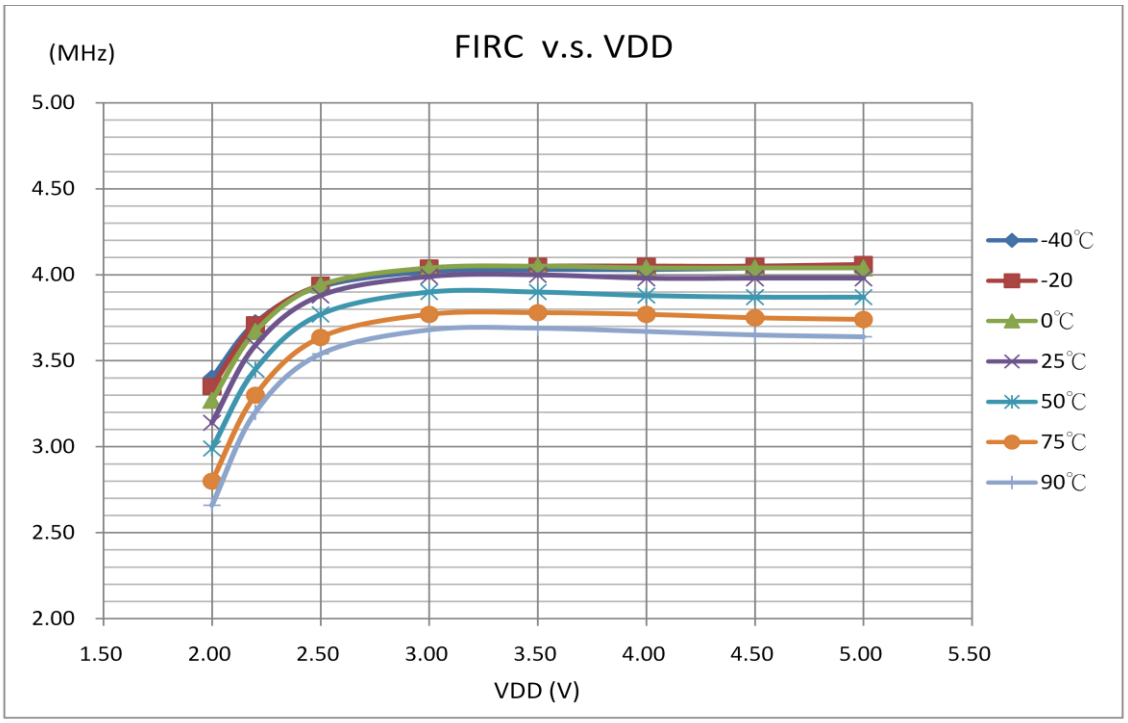
5. ADC Electrical Characteristics (T_A = 25℃, V_{DD} = 2.0V to 5.5V, V_{SS} = 0V)

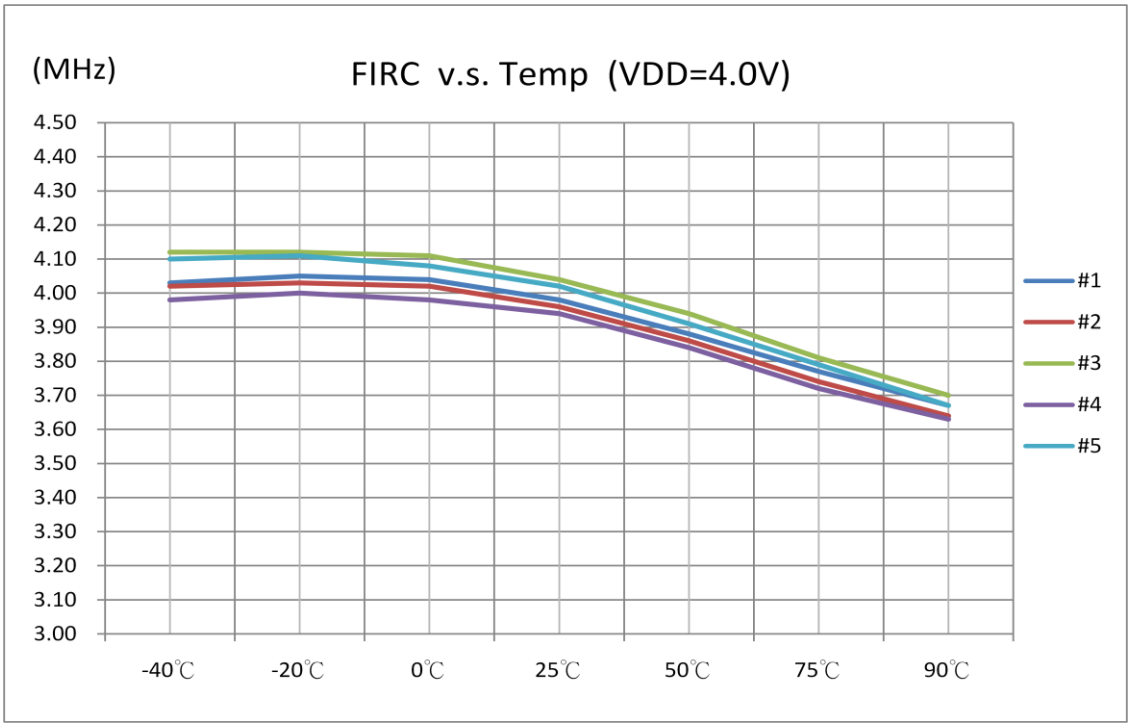
Parameter	Conditions	Min	Typ	Max	Units
Total Accuracy	V_{DD} = 5.12V, V_{SS} = 0V	–	± 2.5	± 4	LSB
Integral Non-Linearity		–	± 3.2	± 5	
Max Input Clock (f_{ADC})	–	–	–	1	MHz
Conversion Time	f_{ADC} = 1 MHz	–	50	–	µs
ADC channel Input Voltage	IVREFS=0	V_{SS}	–	AVREF	V
	IVREFS=1, LDOC with 1uF Cap	V_{SS}	–	2.5	V

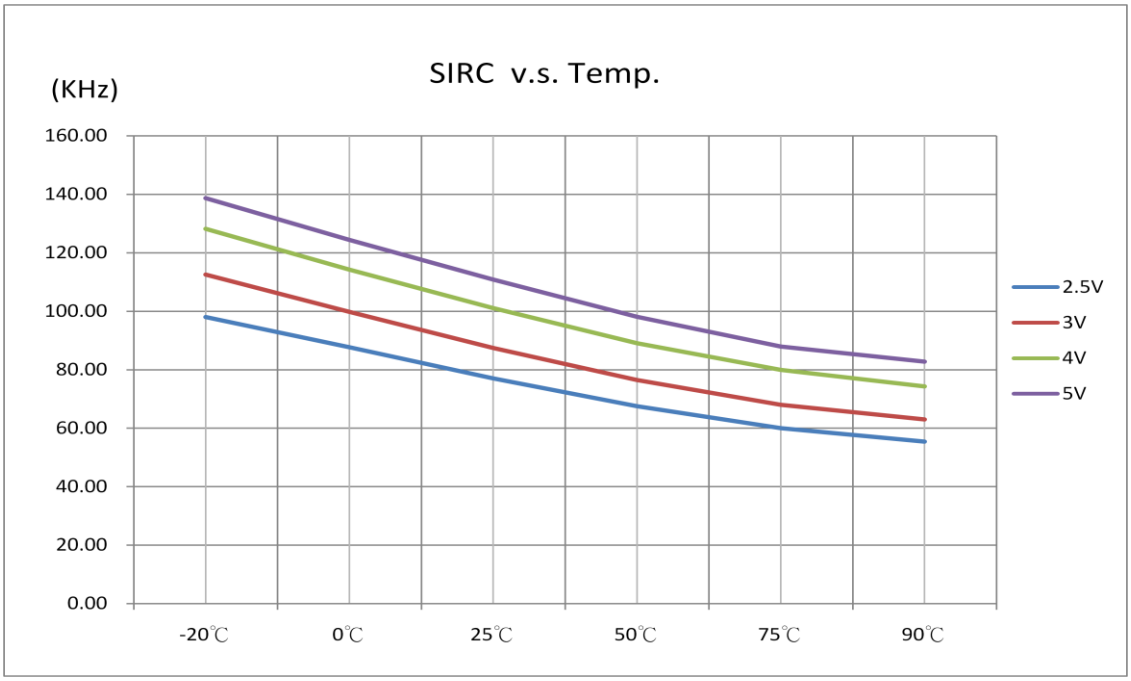
6. VBG/LDO/OVP/OCP Electrical Characteristics (T_A = 25℃, V_{DD} = 2.6V to 5.5V, V_{SS} = 0V)

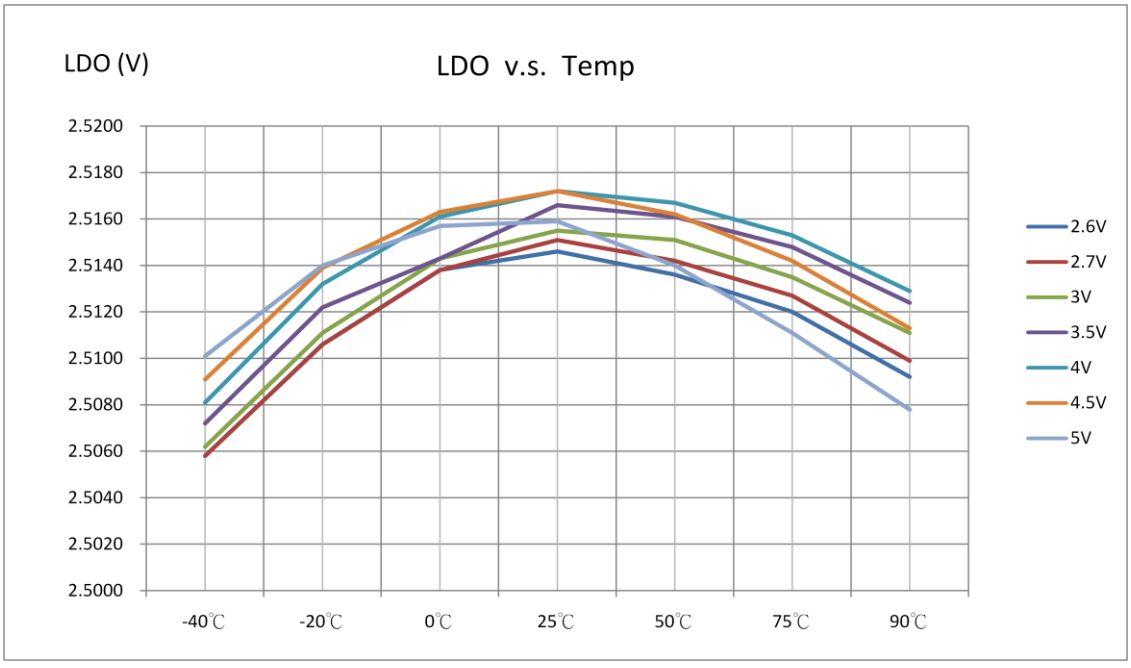
Parameter	Conditions	Min	Typ	Max	Units
VBG/CMP1/CMP2 Current	CMPEN=1, V_{DD} =4.0V	–	400	600	uA
CMP1/CMP2 input Offset Voltage	V_{DD} = 3~5V	-15	–	15	mV
OVP Hysteresis Voltage	V_{DD} = 3~5V	20	40	60	mV
OCP Hysteresis Voltage	V_{DD} = 3~5V	–	0	–	mV
VBG Voltage	VBGEN=1, T_A = -20~80°C	-1.0%	1.25	+1.0%	V
LDO Voltage	IVREFS=1, LDOC with 1uF Cap T_A = -20~80°C	-1.0%	2.5	+1.0%	V

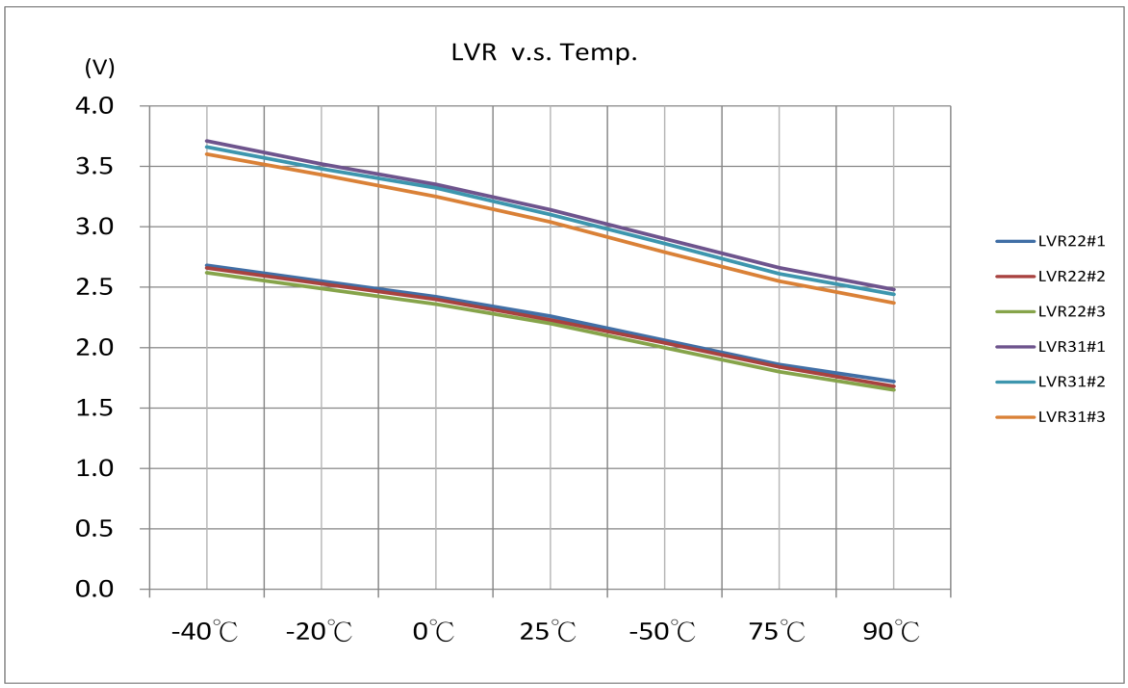
7. Characteristic Graphs






SIRC v.s. Temp.
(KHz)
160.00
140.00
120.00
100.00
80.00
60.00
40.00
20.00
0.00
-20℃
0℃
25℃
50℃
75℃
90℃
2.5V
3V
4V
5V


LDO (V)
LDO v.s. Temp
2.5200
2.5180
2.5160
2.5140
2.5120
2.5100
2.5080
2.5060
2.5040
2.5020
2.5000
-40℃
-20℃
0℃
25℃
50℃
75℃
90℃
2.6V
2.7V
3V
3.5V
4V
4.5V
5V

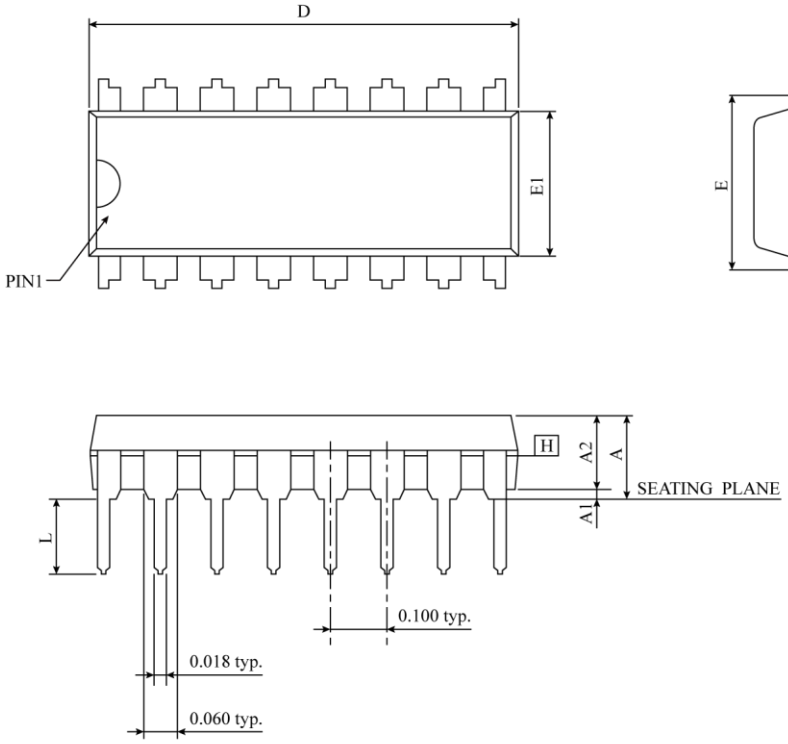

LVR v.s. Temp.
(V)
4.0
3.5
3.0
2.5
2.0
1.5
1.0
0.5
0.0
-40℃
-20℃
0℃
25℃
-50℃
75℃
90℃
LVR22#1
LVR22#2
LVR22#3
LVR31#1
LVR31#2
LVR31#3

PACKAGING INFORMATION

The ordering information:

Ordering number	Package
TM57PA28-OTP	Wafer / Dice blank chip
TM57PA28-COD	Wafer / Dice with code
TM57PA28-OTP-31	SSOP 20-pin (209 mil)
TM57PA28-OTP-05	DIP 20-pin (300 mil)
TM57PA28-OTP-21	SOP 20-pin (300 mil)
TM57PA28-OTP-03	DIP 16-pin (300 mil)
TM57PA28-OTP-16	SOP 16-pin (150 mil)

16-DIP Package Dimension (300 mil)

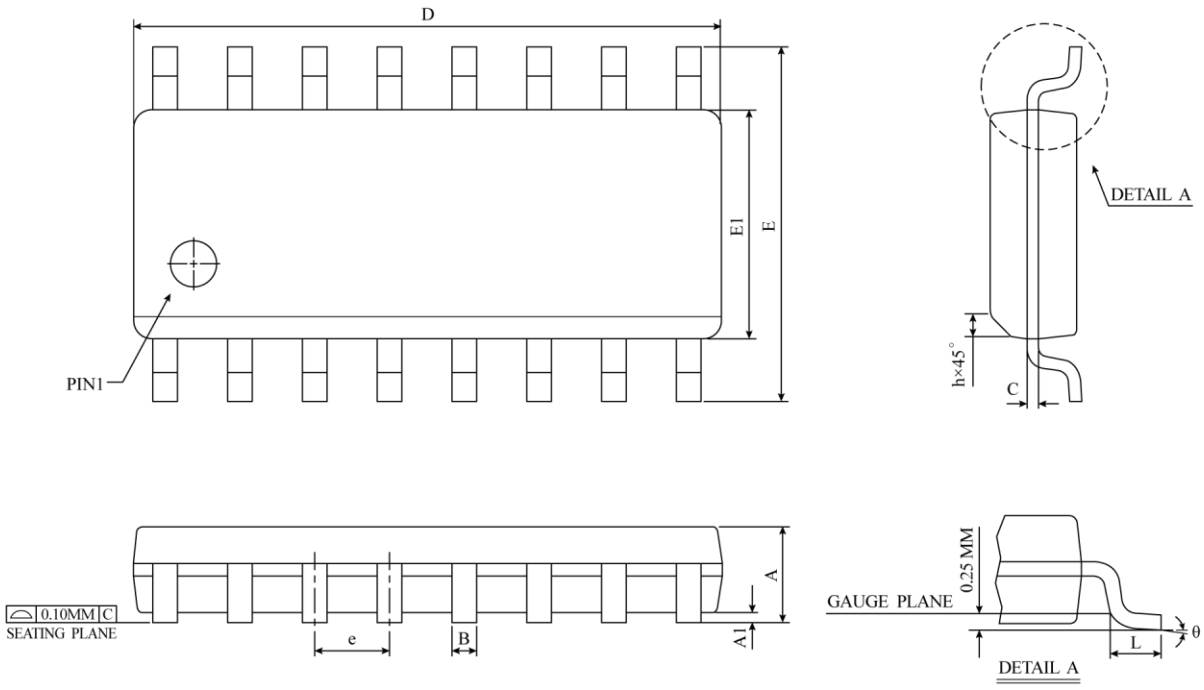


SYMBOL	DIMENSION IN MM			DIMENSION IN INCH		
	MIN	NOM	MAX	MIN	NOM	MAX
A	-	-	4.369	-	-	0.172
A1	0.381	0.673	0.965	0.015	0.027	0.038
A2	3.175	3.302	3.429	0.125	0.130	0.135
D	18.669	19.177	19.685	0.735	0.755	0.775
E		7.620 BSC			0.300 BSC	
E1	6.223	6.350	6.477	0.245	0.250	0.255
L	2.921	3.366	3.810	0.115	0.133	0.150
^{e}B	8.509	9.017	9.525	0.335	0.355	0.375
θ	0°	7.5°	15°	0°	7.5°	15°
JEDEC	MS-001 (BB)					

NOTES：

1. "D" , "E1" DIMENSIONS DO NOT INCLUDE MOLD FLASH OR PROTRUSIONS. MOLD FLASH OR PROTRUSIONS SHALL NOTEXCEED .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 MININUM.
5. DATUM PLANE H COINCIDENT WITH THE BOTTOM OF LEAD, WHERE LEAD EXITS BODY.

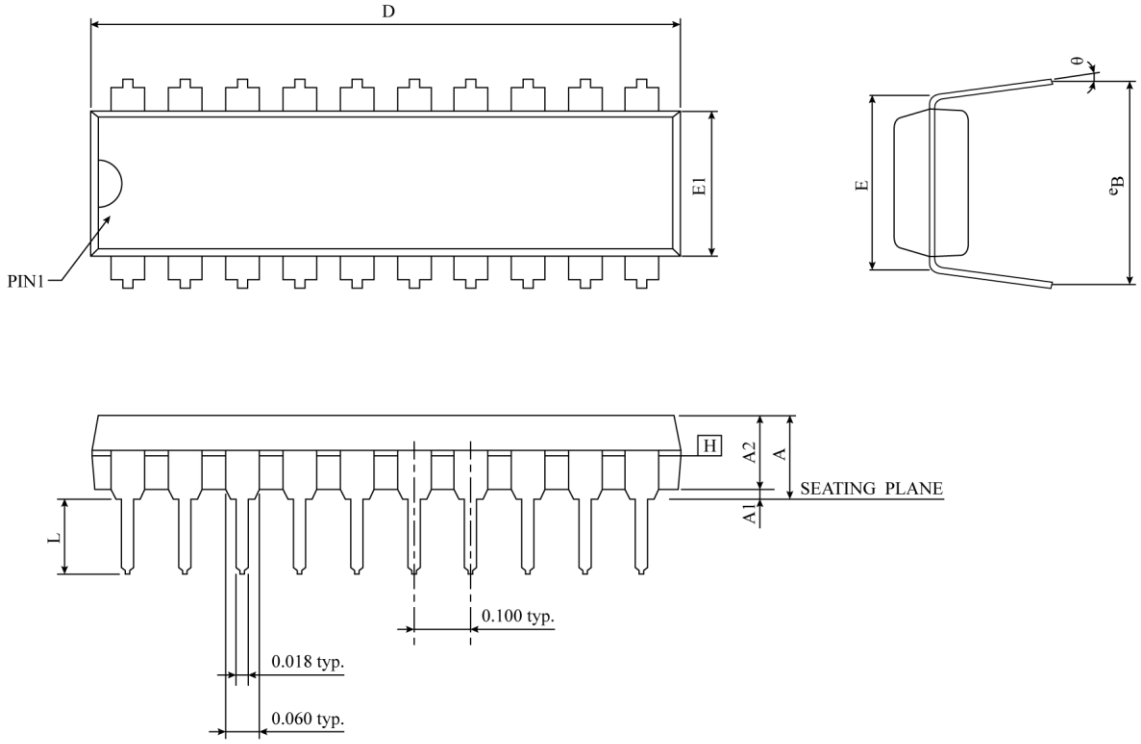
16-SOP Package Dimension (150 mil)



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.

20-DIP Package Dimension (300mil)

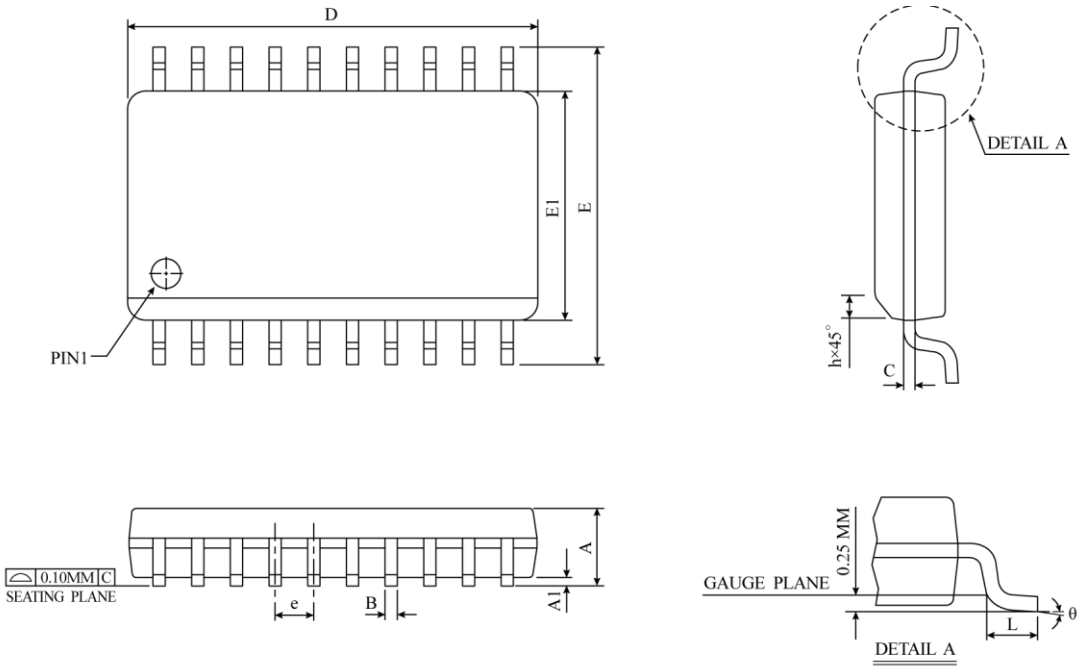


SYMBOL	DIMENSION IN MM			DIMENSION IN INCH		
	MIN	NOM	MAX	MIN	NOM	MAX
A	-	-	4.445	-	-	0.175
A1	0.381	-	-	0.015	-	-
A2	3.175	3.302	3.429	0.125	0.130	0.135
D	25.705	26.061	26.416	1.012	1.026	1.040
E	7.620	7.747	7.874	0.300	0.305	0.310
E1	6.223	6.350	6.477	0.245	0.250	0.255
L	3.048	3.302	3.556	0.120	0.130	0.140
e_B	8.509	9.017	9.525	0.335	0.355	0.375
θ	0°	7.5°	15°	0°	7.5°	15°
JEDEC	MS-001 (AD)					

NOTES：

1. "D" , "E1" DIMENSIONS DO NOT INCLUDE MOLD FLASH OR PROTRUSIONS. MOLD FLASH OR PROTRUSIONS SHALL NOTEXCEED .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 MININUM.
5. DATUM PLANE H COINCIDENT WITH THE BOTTOM OF LEAD, WHERE LEAD EXITS BODY.

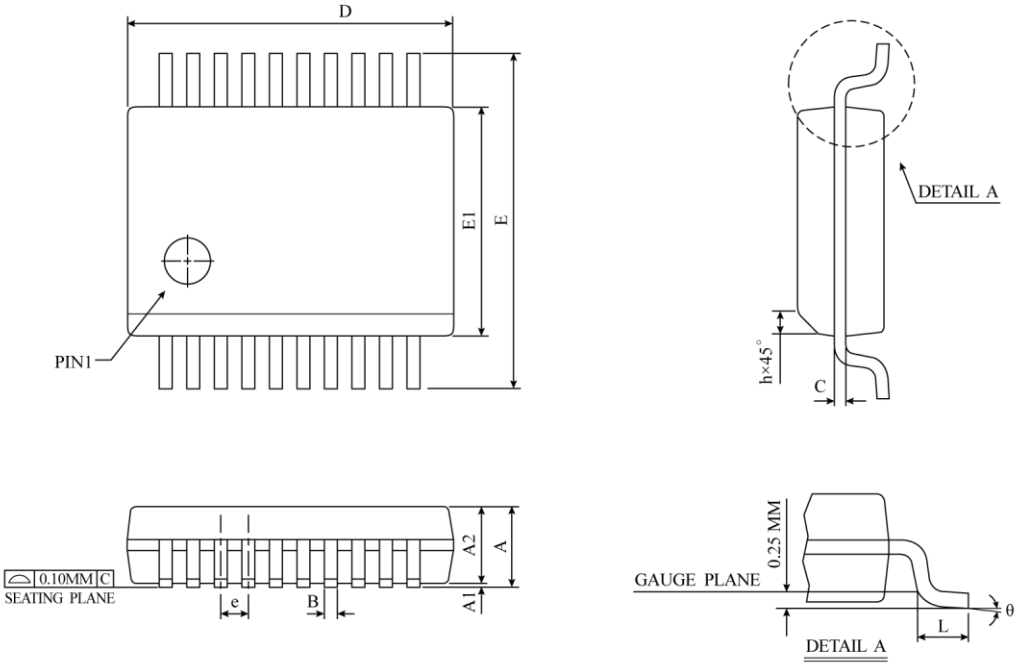
20-SOP Package Dimension (300 mil)



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.

20-SSOP Package Dimension (209mil)



SYMBOL	DIMENSION IN MM			DIMENSION IN INCH		
	MIN	NOM	MAX	MIN	NOM	MAX
A	-	-	2.0	-	-	0.079
A1	0.05	-	-	0.002	-	-
A2	1.65	1.75	1.85	0.065	0.069	0.073
B	0.22	0.28	0.33	0.009	0.011	0.013
C	0.09	0.15	0.21	0.004	0.006	0.008
D	6.90	7.20	7.50	0.272	0.284	0.295
E	7.40	7.80	8.20	0.291	0.307	0.323
E1	5.00	5.30	5.60	0.197	0.209	0.220
e	0.65 BSC			0.026 BSC		
L	0.55	0.75	0.95	0.022	0.030	0.038
θ	0°	4°	8°	0°	4°	8°
JEDEC	M0-150 (AE)					

△ * NOTES : DIMENSION " D " DO NOT INCLUDE MOLD FLASH OR PROTRUSIONS,
BUT DO INCLUDE MOLD MISMATCH MOLD FLASH OR PROTRUSION SHALL NOT EXCEED
0.20 MM (0.008 INCH) PER SIDE.