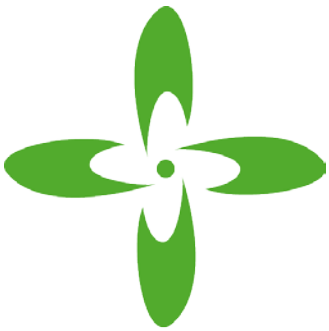


十速

# TM57MA28/28B

## *DATA SHEET*

## *Rev 0.97*

# AMENDMENT HISTORY

| Version | Date | Description |
|---|---|---|
| 0.90 | Dec, 2016 | New release. |
| 0.91 | Dec, 2016 | 1. Add MOS10 (P7、P10、P76、P77)<br>2. Modify typing error (P55、P56) |
| 0.92 | Mar, 2018 | 1. Add package type information (P7、P79)<br>2. Modify ADC max clock (P75) |
| 0.93 | July, 2018 | 1. Modify ordering information (P79) |
| 0.94 | Sep, 2018 | 1. Modify dual clock description (P27)<br>2. Correct TM0 example (P35)<br>3. Add PA7 zero crossing detection note (P53) |
| 0.95 | Dec, 2018 | Add another SOP16 pinout and ordering information (P10, P79) |
| 0.96 | May, 2019 | 1. MA28 don't support LVR2.3/2.9V (P6, P8)<br>2. MA28B can support LVR2.3/2.9V (P6, P8)<br>3. Add MA28B pkg and ordering info. (P10,P79)<br>4. Modify ADC example (P46) |
| 0.97 | July, 2019 | 1. MA28 don't support LVR2.3/2.9V (P6, P8, P25, P26)<br>2. Remove another SOP16 (P10, P79)<br>3. Add ADCKS=750KHz note (P46, P75) |

# CONTENTS

# FEATURES

1. **ROM: 2K x 14 bits MTP (Multi Time Programmable ROM)**
2. **RAM: 96 x 8 bits –** Refer to the below comparsion table.(page 8)
3. **STACK: 6 Levels**
4. **System Oscillation Sources (Fsys)**
   - Fast-clock
     - FIRC (Fast Internal RC): 1.5M/4M/6M/12MHz
   - Slow-clock
     - SIRC (Slow Internal RC): 140KHz/35KHz/8.75KHz/2.2KHz @VCC=3V
5. **Dual System Clock**
   - FIRC+SIRC
6. **Power Saving Operation Mode**
   - FAST Mode: Slow-clock can be disabled or enabled, Fast-clock keeps CPU running
   - SLOW Mode: Fast-clock can be disabled or enabled, Slow-clock keeps CPU running
   - IDLE Mode: Fast-clock and CPU stop. Slow-clock, T2, or Wake-up Timer keep running
   - STOP Mode: All Clocks stop, T2 and Wake-up Timer stop
7. **3 Independent Timers**
   - Timer0
     - 8-bit timer divided by 1~256 pre-scaler option, Counter/Interrupt/Stop function
   - Timer1
     - 8-bit timer divided by 1~256 pre-scaler option, Reload/Interrupt/Stop function
     - Overflow and Toggle out
   - T2
     - 15-bit timer with 4 interrupt interval time options
     - IDLE mode wake-up timer or used as one simple 15-bit time base
     - Clock source: Slow-clock (SIRC), Fsys/128
8. **Interrupt**
   - Three External Interrupt pins
     - 2 pins are falling edge wake-up triggered & interrupts
     - 1 pin is rising or falling edge wake-up triggered & interrupt
   - Timer0/Timer1/T2/WKT (wake-up) Interrupts

9. **Wake-up (WKT) Timer**
   - Clocked by built-in RC oscillator with 4 adjustable interrupt times
     15 ms/30 ms/60 ms/120 ms @VCC=3V
10. **Watchdog Timer**
   - Clocked by built-in RC oscillator with 4 adjustable reset times
     - 110 ms/210 ms/840 ms/1680 ms @VCC=5V
   - Watchdog timer can be disabled/enabled in STOP mode (WDTSTP, R0E.5)
11. **2 Independent PWMs**
   - PWM0:
     - 8+2 bits, duty-adjustable, period-adjustable controlled PWM
     - PWM0 clock source: Fast-clock or Fast-clock-pre*, with 1~64 pre-scalers
   - PWM1:
     - 8+2 bits, duty-adjustable controlled PWM
     - PWM1 clock source: Fast-clock or Fast-clock-pre* (up to 12 MHz)

   Note: **Fast-clock-pre** is the original fast clock, Fast-clock is the divided clock of Fast-clock-pre.
12. **12-bit ADC Converter with 11 input channels and 1 internal reference voltage on ch11**
   - Internal reference voltage (VBG) 1.25V ±2% on channel-11
13. **Built-in 1/2 bias VCC/2 voltage generator to PD3~PD0 maximum 4x13 dots for LCD display**
14. **Reset Sources**
   - Power On Reset/Watchdog Reset/Low Voltage Reset/External Pin Reset
15. **Low Voltage Reset Option:**
   - **TM57MA28**: LVR2.0V, LVR2.0 off in STOP mode
   - **TM57MA28B**: LVR2.0V, LVR2.0 off in STOP mode, LVR2.3V, LVR2.9V
16. **Enhanced Power Noise Rejection**
17. **Operating Voltage:** Low Voltage Reset Level to 5.5V
   - Fsys=1 MHz, 2.0 ~5.5V
   - Fsys=4 MHz, 2.0~5.5V
   - Fsys=6 MHz, 2.2~5.5V
   - Fsys=12 MHz, 2.6~5.5V
18. **Operating Temperature Range: -40°C to +85°C**
19. **Table Read Instruction: 14-bit ROM data lookup table.**
20. **Instruction set: 38 Instructions**
21. **Instruction Execution Time**
   - 2 oscillation clocks per instruction except branch
22. **I/O ports: Maximum 18 programmable I/O pins**

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

**23. Programming connectivity support 5-wire (ISP) or 8-wire program.**

**24. Package Types:**

- 20-pin DIP (300 mil), SOP (300 mil), QFN (3x3x0.75mm)
- 16-pin DIP 300 mil), SOP (150 mil) , SSOP (150mil)
- 10-pin MSOP (118 mil)

**25. Supported EV board on ICE**

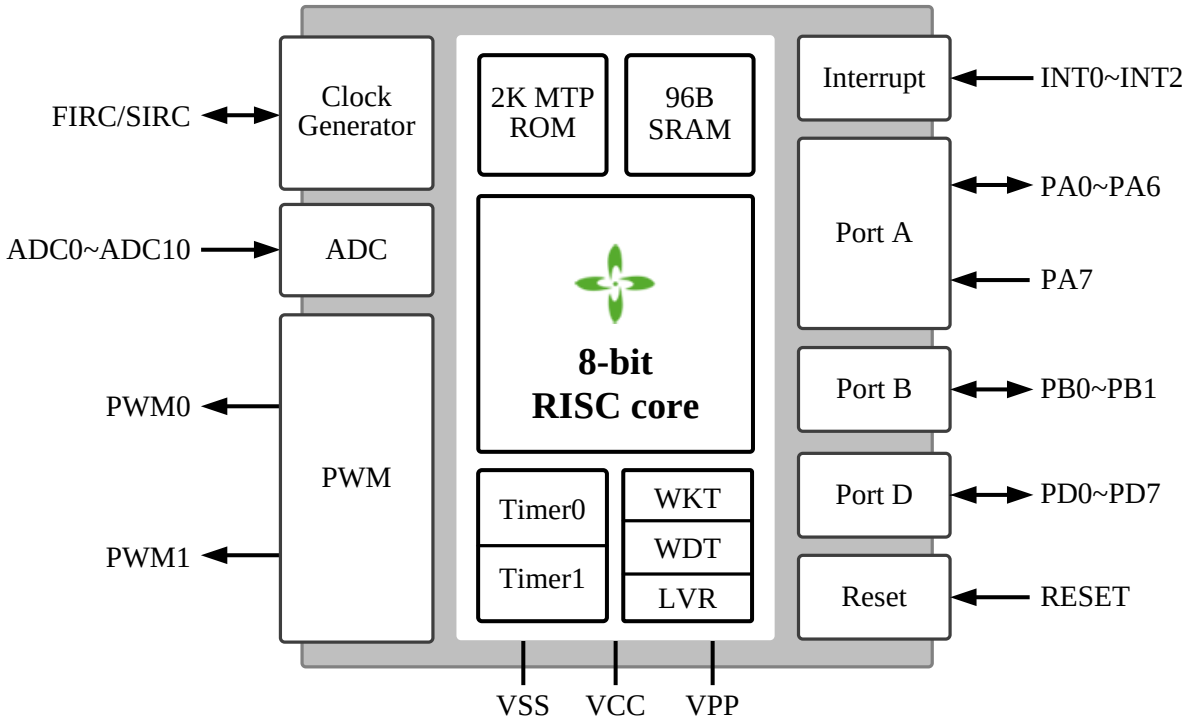
EV board: EV2781C

**TM57MA21B & TM57MA25 & TM57MA28 & TM57MA28B comparsion table:**

| | EV2781 | TM57MA21B | TM57MA25 |
|---|---|---|---|
| EV board | - | EV2781 | EV2781 |
| SRAM | Common: 20~27<br>Bank0: 28~7F<br>Bank1: 28~7F | Common: 20~27<br>Bank0: 28~7F<br>Bank1: 28~7F | Common: 20~27<br>Bank0: 28~7F |
| Fast-clock | FXT / XRC / FIRC | FXT / XRC / FIRC | FIRC |
| Slow-clock | SXT/XRC/<br>SIRC (140 KHz@5V) | SXT/XRC/<br>SIRC (140 KHz@5V) | SIRC (140 KHz@5V) |
| INT0 (PA6) edge interrupt event | Only support falling edge emulation | Only support falling edge | Falling or rising select by the INT0EDGE of SYSCFG |
| Internal VBG | 1.18V+/-8% | 1.18V+/-8% | X |
| TouchKey | Y | Y | X |
| Buzzer | Y | Y | X |
| WDT Timer | 110~1680 ms @5V | 110~1680 ms @5V | 125~2000 ms @5V |
| 1/2 bias LCD | Need ext. pull-low resistors | X | X |
| PA4 Vih (typ) @5V | 3.6V | 3.6V | 2.3V |
| PA4 Vil (typ) @5V | 0.8V | 0.8V | 1.5V |
| PA3 Vih (typ) @5V | 3.0V | 3.0V | 2.3V |
| IO $R_{UP}$ @5V | 140Kohm | 140Kohm | 110Kohm |
| I/O | 18 I/O + ICE I/F<br>LQFP128 | 18 I/O<br>PA7~0, PB1~0, PD7~0 | 14 I/O<br>PA7~0, PB1~0, PD7~4 |

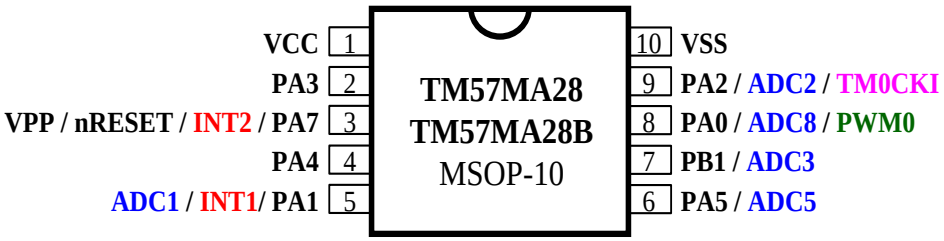
| | **TM57MA28** | **TM57MA28B** | |
|---|---|---|---|
| EV board | EV2781 | same as left side | |
| SRAM | Common: 20~27<br>Bank0: 28~7F | same as left side | |
| Fast-clock | FIRC | same as left side | |
| Slow-clock | SIRC (140 KHz@5V) | same as left side | |
| INT0 (PA6) edge interrupt event | Falling or rising select by the INT0EDGE of SYSCFG | same as left side | |
| Internal VBG | 1.25V+/-2% | same as left side | |
| TouchKey | X | same as left side | |
| Buzzer | X | same as left side | |
| WDT Timer | 125~2000 ms @5V | same as left side | |
| 1/2 bias LCD | Yes, SYSCFG[9] option | same as left side | |
| PA4 Vih (typ) @5V | 2.3V | same as left side | |
| PA4 Vil (typ) @5V | 1.5V | same as left side | |
| PA3 Vih (typ) @5V | 2.3V | same as left side | |
| IO $R_{UP}$ @5V | 110Kohm | same as left side | |
| I/O | 18 I/O<br>PA7~0, PB1~0, PD7~0 | same as left side | |
| LVR | 2.0V | 2.0 / 2.3 / 2.9V | |

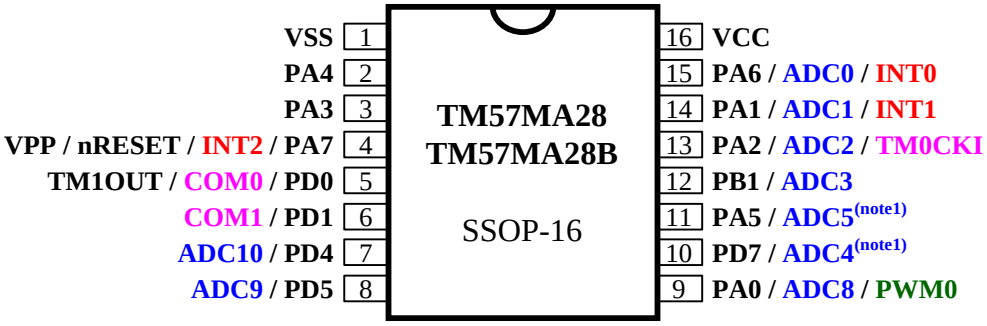
# BLOCK DIAGRAM



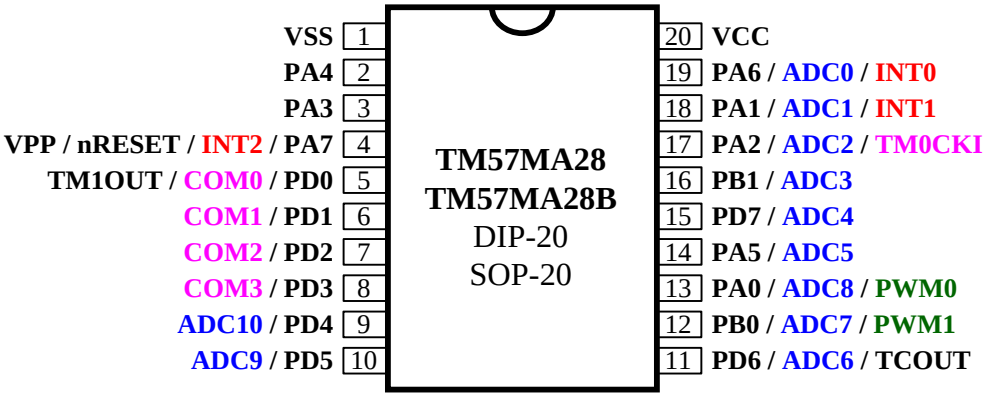

**TM57MA28 Block Diagram**

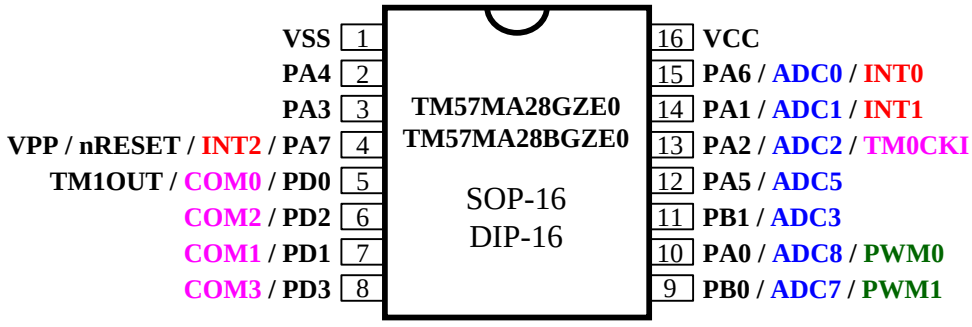
# PIN ASSIGNMENT

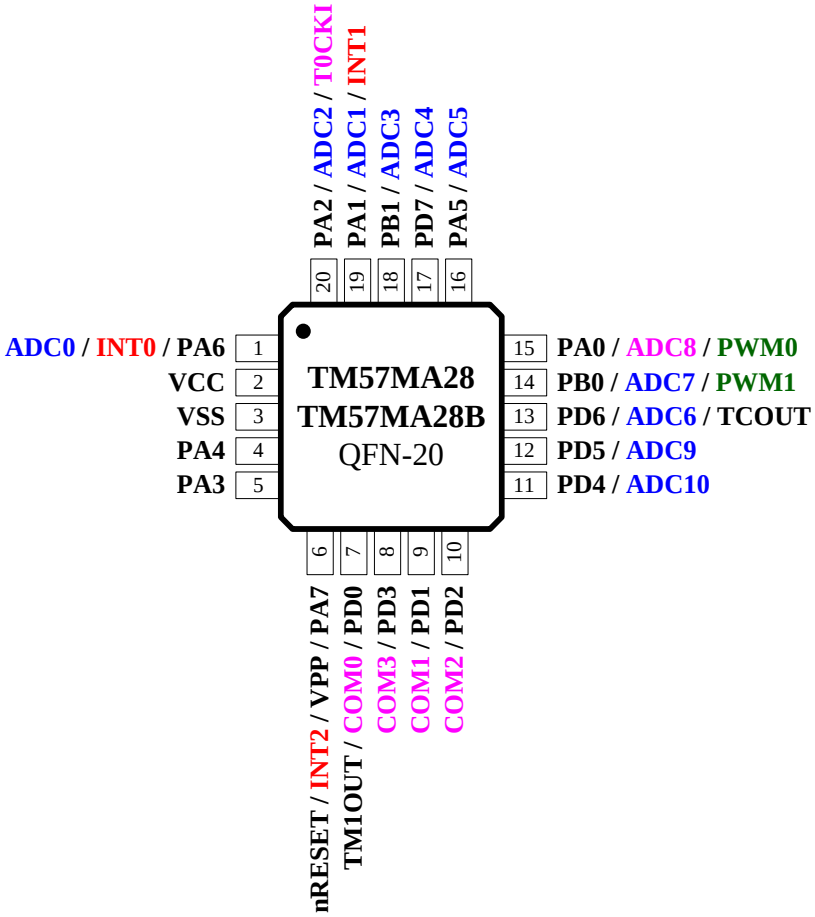



**note1: These two pins order are different with TM57MA21B.**




PA2 / ADC2 / T0CKI
PA1 / ADC1 / INT1
PB1 / ADC3
PD7 / ADC4
PA5 / ADC5
20
19
18
17
16
ADC0 / INT0 / PA6 1
VCC 2
VSS 3
PA4 4
PA3 5
TM57MA28
TM57MA28B
QFN-20
15 PA0 / ADC8 / PWM0
14 PB0 / ADC7 / PWM1
13 PD6 / ADC6 / TCOUT
12 PD5 / ADC9
11 PD4 / ADC10
6
7
8
9
10
nRESET / INT2 / VPP / PA7
TM1OUT / COM0 / PD0
COM3 / PD3
COM1 / PD1
COM2 / PD2

# PIN DESCRIPTIONS

| 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<br>PB0-PB1<br>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. |
| PA7 | I | Schmitt-trigger input with pull-high |
| nRESET | I | External active low reset, internal pull-high |
| VCC, VSS | P | Power Voltage input pin and ground |
| VPP | I | PROM programming high voltage input |
| INT0-INT2 | I | External interrupt input |
| TM0CKI | I | Timer0's input in counter mode |
| TM1OUT | O | Timer1 match output, TM1OUT toggles when Timer1 overflow occurs. |
| TCOUT | O | Instruction cycle clock divided by N output. Where N is 1,2,4,8. The instruction clock frequency is system clock frequency divided by two (Fsys/2). |
| PWM0/PWM1 | O | (8+2) bit PWM output |
| ADC10~ADC0 | I | A/D channels input |
| COM3~COM0 | O | 1/2 bias VCC/2 provide for LCD display |

Programming pins:

Normal mode: VCC/ VSS/ PA0/ PA1/ PA2/ PA3/ PA4/ PA7(VPP)

ISP mode: VCC/ VSS/ PA0/ PA1/ PA7(VPP) -When using ISP (In-system Program) mode, the PCB needs to remove all components of PA0, PA1, PA7.

# PIN SUMMARY

| TM57MA28(B) | | Pin Name | Type | GPIO | | | | | Function After Reset | Alternate Function | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | Input | | Output | | | | | | | |
| 20-SOP/DIP | 16-SOP | | | Weak Pull-up | Ext. Interrupt | P.O.D | O.D | P.P | | PWM | LCD | ADC | MISC |
| 1 | 1 | VSS | P | | | | | | | | | | |
| 2 | 2 | PA4 | I/O | O | | | O | O | | | | | |
| 3 | 3 | PA3 | I/O | O | | | O | O | | | | | |
| 4 | 4 | VPP/nRESET/INT2/PA7 | I | O | O | | | | SYS | | | | nRESET |
| 5 | 5 | TM1OUT/PD0 | I/O | O | | | O | O | PD0 | | O | | TM1OUT |
| 6 | 7 | PD1 | I/O | O | | | O | O | PD1 | | O | | |
| 7 | 6 | PD2 | I/O | O | | | O | O | PD2 | | O | | |
| 8 | 8 | PD3 | I/O | O | | | O | O | PD3 | | O | | |
| 9 | - | ADC10/PD4 | I/O | O | | | O | O | PD4 | | | O | |
| 10 | - | ADC9/PD5 | I/O | O | | | O | O | PD5 | | | O | |
| 11 | - | PD6/ADC6/TCOUT | I/O | O | | | O | O | PD6 | | | O | TCOUT |
| 12 | 9 | PB0/ADC7/PWM1 | I/O | O | | | O | O | PB0 | O | | O | |
| 13 | 10 | PA0/ADC8/PWM0 | I/O | O | | O | | O | PA0 | O | | O | |
| 14 | 12 | PA5/ADC5 | I/O | O | | | O | O | PA5 | | | O | |
| 15 | - | PD7/ADC4 | I/O | O | | | O | O | PD7 | | | O | |
| 16 | 11 | PB1/ADC3 | I/O | O | | | O | O | PB1 | | | O | |
| 17 | 13 | PA2/ADC2/TM0CKI | I/O | O | | O | | O | PA2 | | | O | TM0CKI |
| 18 | 14 | PA1/ADC1/INT1 | I/O | O | O | O | | O | PA1 | | | O | |
| 19 | 15 | PA6/ADC0/INT0 | I/O | O | O | | O | O | PA6 | | | O | |
| 20 | 16 | VCC | P | | | | | | | | | | |

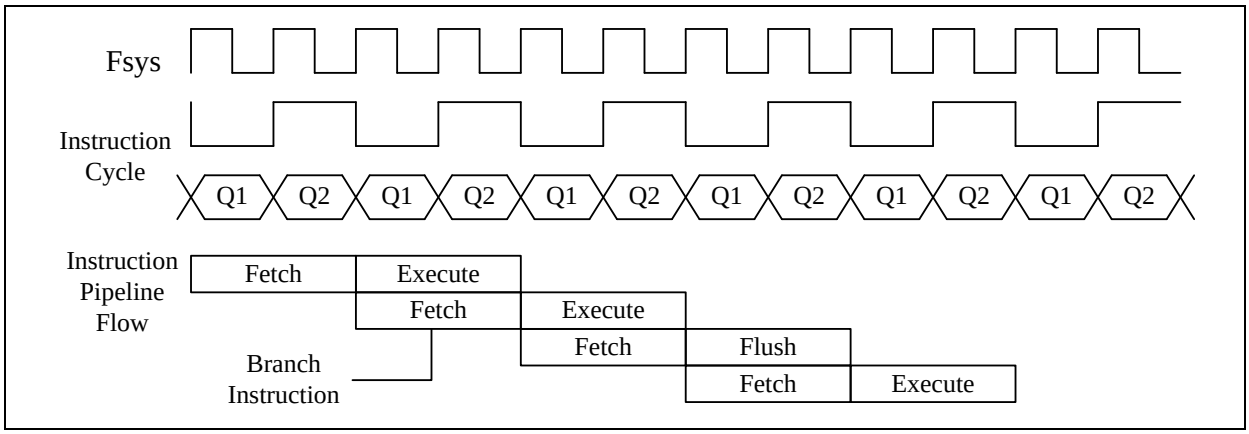
Symbol：P.P. = Push-Pull Output
P.O.D. = Pseudo Open Drain
O.D. = Open Drain
SYS = by SYSCFG bit

# FUNCTIONAL DESCRIPTION

## 1. CPU Core

### 1.1 Clock Scheme and Instruction Cycle

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




When Fsys=4 MHz, the two instruction cycles time (CALL, RET…) =1uS, single instruction cycle time (MOVWF…) =0.5uS

Terminology definitions:

(1) **Fast-clock-pre**: The clock source that contains one fast frequency oscillation: Fast Internal RC oscillator (FIRC).

(2) **Fast-clock**: The words means the all set of fast clocks, its clock frequency can be Fast-clock-pre divided by 1, 2, 3, 8.

(3) **Slow-clock**: The clock source that contains Slow Internal RC oscillator (SIRC).

(4) **Fsys**: System clock. The main clock that drive the core logic and most peripherals. The clock source can be either Fast-clock or Slow-clock which can be set by registers.

(5) **Instruction Cycle**=Fsys/2

FXT: Fast Crystal

SXT: Slow Crystal (32 KHz)

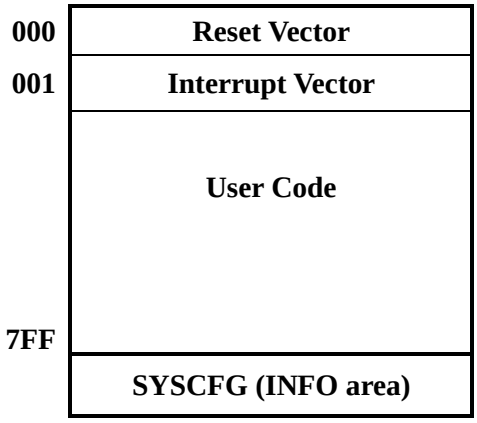
FIRC: Fast Internal RC oscillator

XRC: Fast or Slow External RC oscillator

SIRC: Slow Internal RC oscillator

### 1.2 Program ROM (PROM)

The MTP Program ROM of this device is 2K words, with an extra INFO area to store the SYSCFG. The ROM can be written multi-times and can be read as long as the PROTECT bit of SYSCFG is not set. The SYSCFG can be read no matter PROTECT is set or cleared, but can be written only when PROTECT is not set or ROM is erased. That is, unprotect the PROTECT bit needs the erased ROM.



### 1.3 Programming Counter (PC) and Stack

The Programming Counter is 11-bit wide capable of addressing a 2Kx14 MTP ROM. As a program instruction is executed, the PC will contain the address of the next program instruction to be executed. The PC value is normally increased by one except the followings. The Reset Vector (000h) and the Interrupt Vector (001h) are provided for PC initialization and Interrupt. For CALL/GOTO instructions, PC loads 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. The STACK is 11-bit wide and 6-level in depth. The CALL instruction and hardware interrupt will push STACK level in order. While the RET/RETI/RETLW instruction pops the STACK level in order.

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

◇Example: To look up the PROM data located “TABLE” & “TABLE2”.

```
            ORG         000H            ; Reset Vector
            GOTO        START

START:
            MOVLW       00H
            MOVWF       INDEX           ; Set lookup table’s address.

LOOP:
            MOVFW       INDEX           ; Move index value to W register.
            CALLTABLE                   ; To lookup data, W=55H.
            …..
            INCF INDEX, 1               ; Increment the index address for next address
            …..
            GOTO        LOOP            ; Go to LOOP label.
            …..
            MOVLW       (TABLE2>>8) &0xff
            MOVWF       DPH             ; DPH register (F0F.2~0)
            MOVLW       (TABLE2) &0xff
            MOVWF       DPL             ; DPL register (F13.7~0)
            TABRL                       ; W=86H
            TABRH                       ; W=19H
            …..

TABLE:
            ADDWF       PCL, 1          ; Add the W with PCL, the result back in PCL.
            RETLW       55H             ; W=55h when return
            RETLW       56H             ; W=56H when return
            RETLW       58H             ; W=58H when return
            …..
            ORG  768H

TABLE2:
            .DT    0x1986, 0x3719, 0x2983…  ; 14-bit ROM data
```

### 1.4 ALU and Working (W) Register

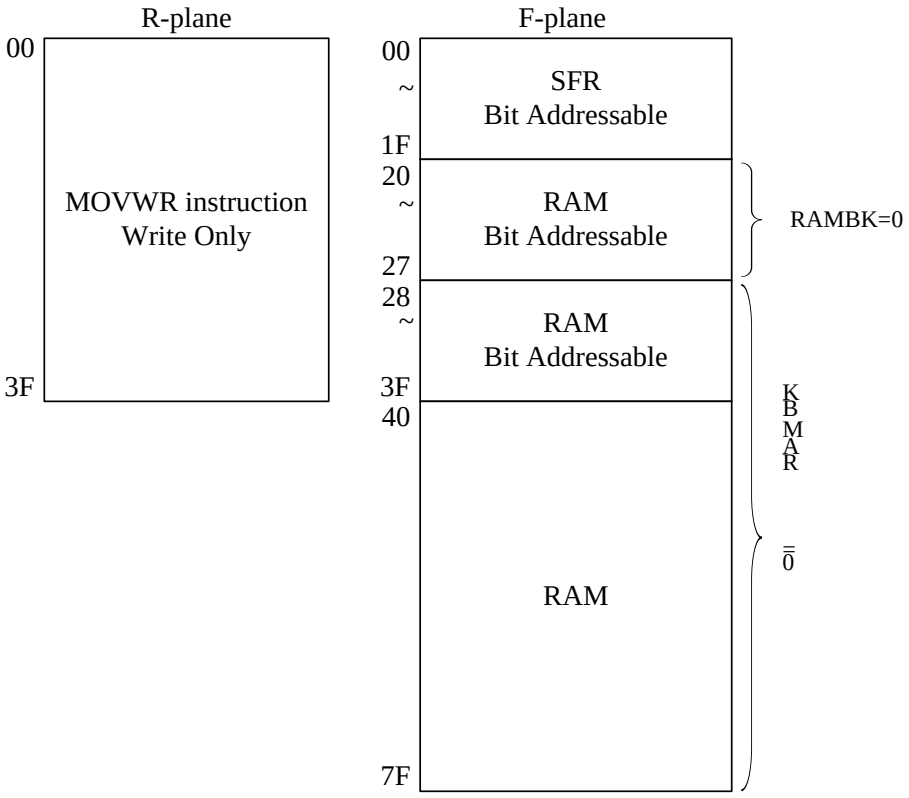
The ALU is 8-bit wide and capable of addition, subtraction, shift and logical operations. In two-operand instructions, typically one operand is the W register, which is an 8-bit non-addressable register used for ALU operations. The other operand is either a file register or an immediate constant. In 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 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 copies 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 register (FSR is a pointer). The first half of F-Plane is bit-addressable, while the second half of F-Plane is not bit-addressable.



◇Example: Write immediate data into R-Plane register.

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

◇Example: Move the immediate data 55H to W register and F-Plane location 20H.

```
MOVLW   55H     ; Move immediate 55H into W register.
MOVWF   20H     ; To get a content of W and save in 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 and save in W.
```

◇Example: Indirectly addressing mode with FSR/INDF register. (F-Plane 04H/00H)

```
MOVLW   20H
MOVWF   FSR     ; Move immediate 20H into FSR register.
MOVLW   55H
MOVWF   INDF    ; Use data pointer FSR write a data into F-Plane location
                ; 20H. 55H into F-plane 20H.
INCF    FSR, 1  ; Increment the index address for next address.
MOVFW   INDF    ; Use data pointer FSR read a data from F-Plane location
                ; 21H. W data from F-Plane 21H
```

### 1.6 STATUS Register (F-Plane 03H)

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

<table>
<tr><td>STATUS</td><td>Bit 7</td><td>Bit 6</td><td>Bit 5</td><td>Bit 4</td><td>Bit 3</td><td>Bit 2</td><td>Bit 1</td><td>Bit 0</td></tr>
<tr><td>Reset Value</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr>
<tr><td>R/W</td><td>R/W</td><td>R/W</td><td>R</td><td>R</td><td>R</td><td>R/W</td><td>R/W</td><td>R/W</td></tr>
<tr><td>Bit</td><td colspan="8">Description</td></tr>
<tr><td>7</td><td colspan="8">GB0: General Purpose Bit 0</td></tr>
<tr><td>6</td><td colspan="8">GB1: General Purpose Bit 1</td></tr>
<tr><td>5</td><td colspan="8">Reserved</td></tr>
<tr><td>4</td><td colspan="8">TO: Time Out Flag<br>0: after Power On Reset, LVR Reset, or CLRWDT/SLEEP instruction<br>1: WDT time out occurs</td></tr>
<tr><td>3</td><td colspan="8">PD: Power Down Flag<br>0: after Power On Reset, LVR Reset, or CLRWDT instruction<br>1: after SLEEP instruction</td></tr>
<tr><td>2</td><td colspan="8">Z: Zero Flag<br>0: the result of a logic operation is not zero<br>1: the result of a logic operation is zero</td></tr>
<tr><td rowspan="3">1</td><td colspan="8">DC: Decimal Carry Flag or Decimal /Borrow Flag</td></tr>
<tr><td colspan="4">ADD instruction</td><td colspan="4">SUB instruction</td></tr>
<tr><td colspan="4">0: no carry<br>1: a carry from the low nibble bits of the result occurs</td><td colspan="4">0: a borrow from the low nibble bits of the result occurs<br>1: no borrow</td></tr>
<tr><td rowspan="3">0</td><td colspan="8">C: Carry Flag or/Borrow Flag</td></tr>
<tr><td colspan="4">ADD instruction</td><td colspan="4">SUB instruction</td></tr>
<tr><td colspan="4">0: no carry<br>1: a carry occurs from the MSB</td><td colspan="4">0: a borrow occurs from the MSB<br>1: no borrow</td></tr>
</table>

◇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.
BSF     03H, 7      ; Set GB0=1
BCF     STATUS, 0   ; Clear C=0.
BCF     03H, 7      ; Clear GB0=0
```

◇Example: Determine the C flag by BTFSS instruction.

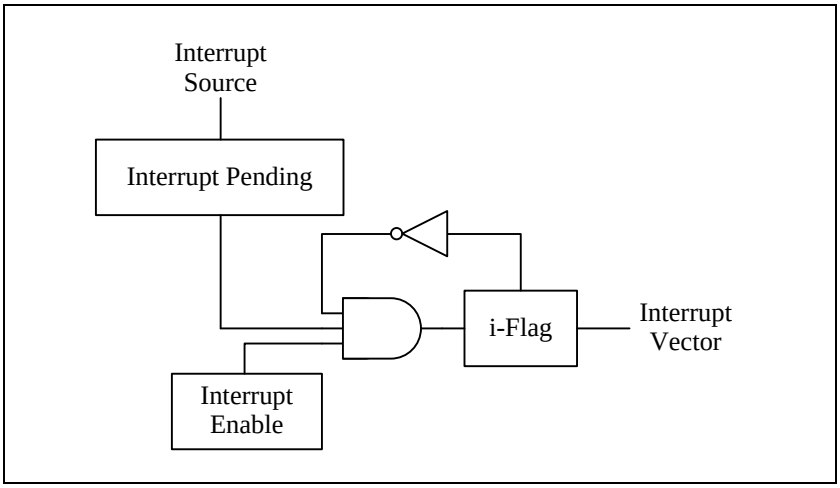
```
BTFSS   STATUS, 0   ; Check the carry flag
GOTO    LABEL_1     ; If C=0, goto label_1
GOTO    LABEL_2     ; If C=1, goto label_2
```

### 1.7 Interrupt

This device has 1 level, 1 vector and eight 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 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 and rising edge trigger.

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

        ORG     01H             ; All interrupt vector.
        GOTO    INT_SUBROUTINE  ; If INT1 (PA1) input occurred rising edge.

        ORG     02H

START:
        MOVLW   11111101B
        MOVWR   PAPUN           ; Enable INT1 (PA1) input pull up resistor.
        MOVLW   00010000B
        MOVWR   R0B             ; Set INT1 interrupt trigger as rising edge.
        MOVLW   11111101B
        MOVWF   INT1IF          ; Clear INT1 interrupt request flag
        MOVLW   00000010B
        MOVWR   INTIE           ; Enable INT1 interrupt.

MAIN:
        …
        GOTO    MAIN

INT_SUBROUTINE:
        MOVWF   GPR0            ; Push routine to Save W and STATUS data to buffers.
        MOVFW   STATUS          ; F-Plane 03H
        MOVWF   GPR1

        BTFSS   INT1IF          ; Check INT1IF bit.
        GOTO    EXIT_INT        ; INT1IF=0, exit interrupt vector.
        …                       ; INT1 interrupt service routine.
        MOVLW   11111101B
        MOVWF   INTIF           ; Clear INT1 interrupt request flag
        GOTO    EXIT_INT

EXIT_INT:
        MOVFW   GPR1            ; POP Routine W and STATUS data from buffers.
        MOVWF   STATUS
        MOVFW   GPR0
        RETI
```

# 2. Chip Operation Mode

## 2.1 Reset (000H)

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

LVR Selection Table:

| LVR level | Operating voltage |
|---|---|
| LVR2.9* | 5.5V > VCC > 3.3V or VCC is fixed at 5.0V |
| LVR2.3* | 5.5V > VCC > 2.7V |
| LVR2.0 | 5.5V > VCC > 2.2V or VCC is fixed at 3.6V |

Note: LVR2.9/LVR2.3 are only for TM57MA28B

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 also set all the control registers to their default reset value. The TO/PD flags are not affected by these resets.

◇Example: Defining Reset Vector

```
        ORG     000H
        GOTO    START           ; Jump to user program address.

        ORG     010H

START:
        …                       ; 010H, The head of user program
        …
        GOTO    START
```

## 2.2 System Configuration Register (SYSCFG)

The System Configuration Register (SYSCFG) is located at ROM address 3FCh. The SYSCFG determines the option for initial condition of MCU. It is written by PROM Writer only. User can select clock source, LVR threshold voltage and chip operation mode by SYSCFG register. The default value of SYSCFG is 3FFFh. The 13th bit of SYSCFG is code protection selection bit. If this bit is 1, the data in PROM will be protected, when user reads PROM.

| **Bit** | **13~0** | |
|---|---|---|
| **Default Value** | **11111111111111** | |
| Bit | Description | |
| 13 | **PROTECT**: Code protection selection | |
| | 1 | Enable |
| | 0 | Disable |
| 12 | **PWRSAV**: Power Saving Mode (affect power consumption for STOP/IDLE mode) | |
| | 1 | Enable (IVC* off) |
| | 0 | Disable (increase standby current 250uA@3V in STOP/IDLE mode) |
| 11-10 | **LVR**: Low Voltage Reset Mode | |
| | 11 | 2.0V, Always Enable |
| | 10 | 2.0V, Disable in STOP/IDLE Mode (if **MODE3V**=1, force LVR=2'b10) |
| | 01 | TM57MA28: 2.3V don't support<br>TM57MA28B: 2.3V always enable |
| | 00 | TM57MA28: 2.9V don't support<br>TM57MA28B: 2.9V always enable |
| 9 | **PD30IOE**: PD[3:0] I/O Mode Enable | |
| | 1 | PD [3:0] as I/O mode |
| | 0 | PD [3:0] as LCD mode (PDE [3:0] will be invalid) |
| 8 | **INT0EDGE**: INT0 (PA6) trigger edge (Notice: EV2781 only support falling edge emulation) | |
| | 1 | Rising edge |
| | 0 | Falling edge |
| 7 | **XRSTE**: External Pin (PA7) Reset Enable | |
| | 1 | Enable |
| | 0 | Disable (ie: PA7 as input pin) |
| 6 | **WDTE**: WDT Reset Enable | |
| | 1 | Enable |
| | 0 | Disable |
| 5 | **MODE3V**: Operating Voltage Selection | |
| | 1 | Operating Voltage≦3.6V, default Fsys=Slow-clock (SIRC 2 KHz) |
| | 0 | Operating Voltage＞3.6V, default Fsys=Fast-clock (FIRC 4 MHz) |
| 4-0 | Tenx Reserved | |

* IVC is the chip built-in 3.3V regulator for internal circuit.

- SYSCFG setting & power consumption relationship:

TM57MA28:

| MODE3V | PWRSAV | LVR[1:0] | FAST/SLOW mode | STOP/IDLE mode |
|---|---|---|---|---|
| 1 | X | XX | LVR2.0V on, IVC off | LVR2.0V off<br>IVC off |
| 0 | 0 | 2V, Always | LVR2.0V on, IVC on | LVR2.0V on<br>IVC on (+250 uA at VCC=3V) |
| 0 | 1 | 2V, Always | LVR2.0V on, IVC on | LVR2.0V on<br>IVC off |
| 0 | 0 | 2V, Disable | LVR2.0V on, IVC on | LVR2.0V off<br>IVC on (+250 uA at VCC=3V) |
| 0 | 1 | 2V, Disable | LVR2.0V on, IVC on | LVR2.0V off<br>IVC off |

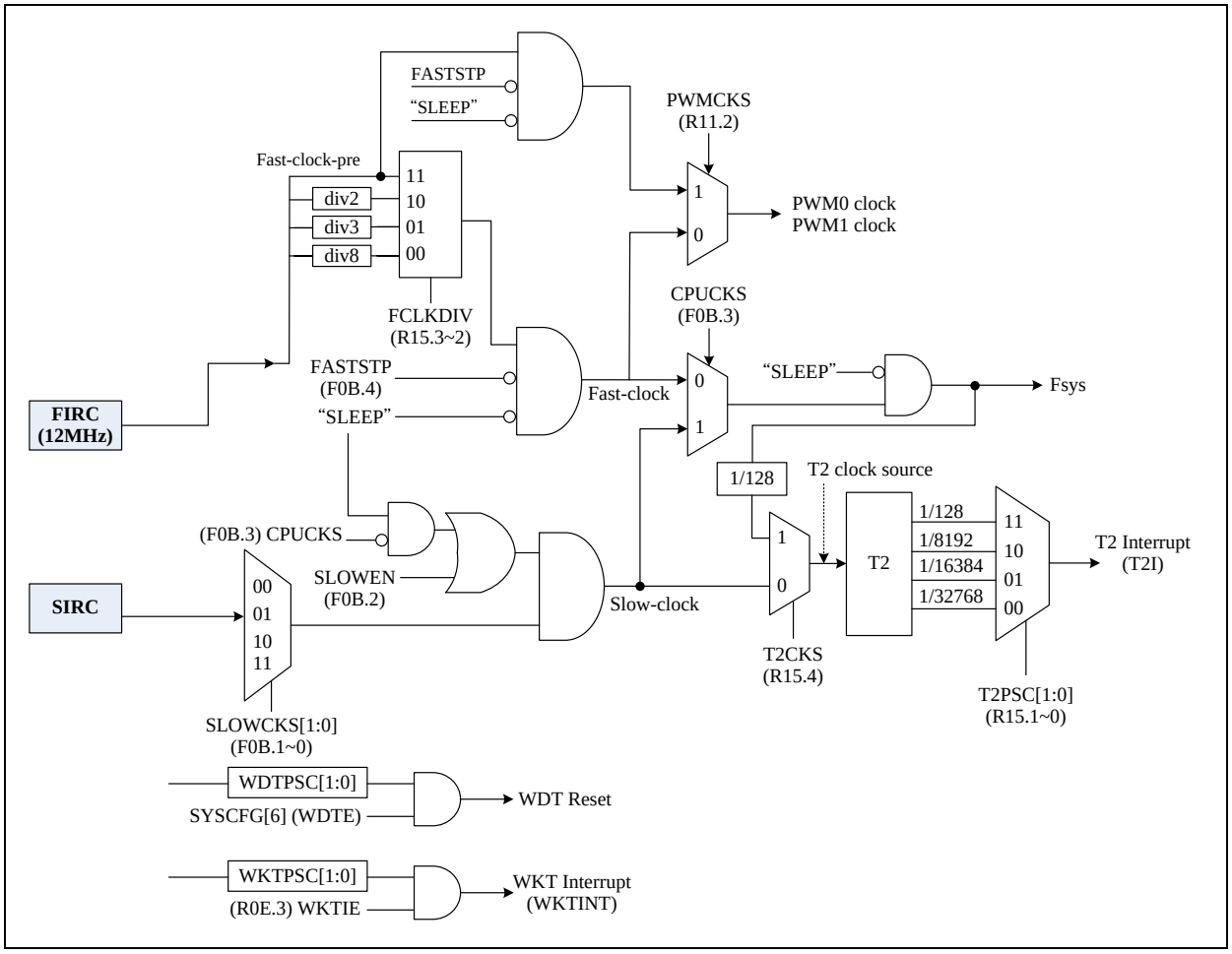
TM57MA28B:

| MODE3V | PWRSAV | LVR[1:0] | FAST/SLOW mode | STOP/IDLE mode |
|---|---|---|---|---|
| 1 | X | XX | LVR2.0V on, IVC off | LVR2.0V off<br>IVC off |
| 0 | 0 | 2V, Always | LVR2.0V on, IVC on | LVR2.0V on<br>IVC on (+250 uA at VCC=3V) |
| 0 | 1 | 2V, Always | LVR2.0V on, IVC on | LVR2.0V on<br>IVC off |
| 0 | 0 | 2V, Disable | LVR2.0V on, IVC on | LVR2.0V off<br>IVC on (+250 uA at VCC=3V) |
| 0 | 1 | 2V, Disable | LVR2.0V on, IVC on | LVR2.0V off<br>IVC off |
| 0 | 0 | 2.3V | LVR2.3V on, IVC on | LVR2.3V on<br>IVC on (+250 uA at VCC=3V) |
| 0 | 1 | 2.3V | LVR2.3V on, IVC on | LVR2.3V off<br>IVC off |
| 0 | 0 | 2.9V | LVR2.9V on, IVC on | LVR2.9V on<br>IVC on (+250 uA at VCC=3V) |
| 0 | 1 | 2.9V | LVR2.9V on, IVC on | LVR2.9V off<br>IVC off |

## 2.3 Dual System Clock

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

**TM57MA28/28B don't support FXT/SXT/XRC.**




**Clock Scheme Block Diagram**

**FAST Mode:**

After power on or reset, if MODE3V(SYSCFG[5])=0, device enters FAST mode, otherwise enters SLOW mode. User can set CPUCKS=1 to enter SLOW mode. Besides, firmware can also enable or disable the Slow-clock for the T2 system operating.

In this mode, the program is executed using Fast-clock as CPU clock (Fsys). The Timer0, Timer1 blocks are also driven by Fast-clock, The PWM0, and PWM1 block can driven by Fast-clock or Fast-clock-pre. T2 can also be driven by Fast-clock by setting T2CKS=1 and CPUCKS=0.

**SLOW Mode:**

After power-on or reset, if MODE3V=1, device enters SLOW mode, the default Slow-clock is SIRC & Clock=2 KHz. In this mode, the Fast-clock can stopped (by FASTSTP=1, for power saving) or running (by FASTSTP=0), and Slow-clock is enabled. All peripheral blocks (Timer0, Timer1... etc) clock sources are Slow-clock in the SLOW mode.

**IDLE Mode:**

If Slow-clock is enabled and T2CKS=0 before executing the SLEEP instruction, the CPU enters the IDLE mode. In this mode, the Slow-clock will continue running to provide clock to T2 block. CPU stop fetching code and all blocks are stop except T2 related circuits.

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

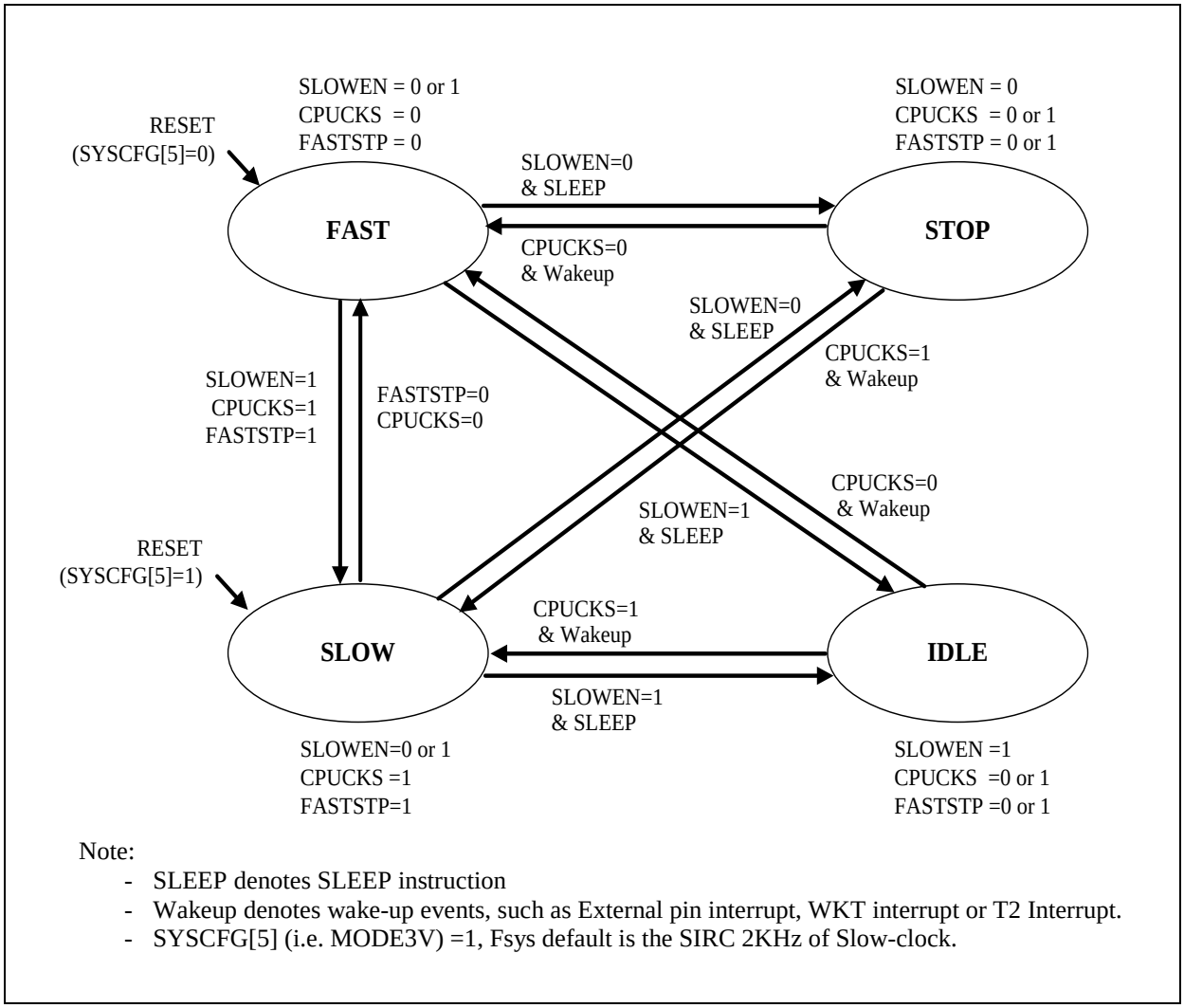
T2 and WKT/WDT are independent and have their own control registers. It is possible to keep both T2 and WKT working and wake-up in the IDLE mode.

**STOP Mode:**

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

## 2.4 Dual System Clock Modes Transition

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




Note:

- SLEEP denotes SLEEP instruction
- Wakeup denotes wake-up events, such as External pin interrupt, WKT interrupt or T2 Interrupt.
- SYSCFG[5] (i.e. MODE3V) =1, Fsys default is the SIRC 2KHz of Slow-clock.

**CPU Operation Block Diagram**

CPU Mode & Clock Functions Table:

| Mode | Oscillator | Fsys | Fast-clock | Slow-clock | TM0/TM1 | T2 | Wakeup event |
|---|---|---|---|---|---|---|---|
| FAST | FIRC | Fast-clock | Run | Set by SLOWEN bit | Run | Run | X |
| SLOW | SIRC | Slow-clock | Set by FASTSTP bit | Run | Run | Run | X |
| IDLE | SIRC | Stop | Stop | Run | Stop | Run | WKT/IO/T2 |
| STOP | Stop | Stop | Stop | Stop | Stop | Stop | IO |

- **PA3/PA4 IO setting notes in STOP/IDLE mode**

Note: In STOP/IDLE mode, PA3 or PA4 must enable internal pull-high to avoid floating state for different Slow-clock types. The PA3 and PA4 IO setting list as below:

| | Fast-clock | Slow-clock | PAD3 | PAE3 | PAPUN3 | PAD4 | PAE4 | PAPUN4 |
|---|---|---|---|---|---|---|---|---|
| | FIRC | SIRC | ※ | ※ | ※ | ※ | ※ | ※ |

**※: Don't care**

- **FAST mode switches to SLOW mode**

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

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

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

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

◇Example: Switch FAST mode to SLOW mode.

```
MOVLW   011000001B
MOVWF   F0B         ; Slow-clock type=SIRC & 2 KHz.
BSF     SLOWEN      ; Enable Slow-clock.
NOP
BSF     CPUCKS      ; Fsys=Slow-clock.
BSF     FASTSTP     ; Disable Fast-clock.
```

- **SLOW mode switches to FAST mode**

SLOW mode can be enabled by SLOWEN bit and CPUCKS bit in F0B register of F-plane. The following steps are suggested to be executed by order when SLOW mode switches to FAST mode:

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

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

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

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

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

```
MOVLW   00001000B
MOVWR   R15         ; Fast-clock = Fast-clock-pre/2
BCF     FASTSTP     ; Enable Fast-clock.
NOP
BCF     CPUCKS      ; Fsys = 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) or WKT(WKTIE=1)

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

(3) Execute SLEEP instruction

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

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

```
BSF     SLOWEN      ; Enable Slow-clock.
MOVLW   000000 01 B
MOVWR   R15         ; T2 Clock source=Slow-clock. TM2PSC=div 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 WKT/WDT (WKTIE=0, WDTSTP=1)

(3) Execute SLEEP instruction

STOP mode can be waken up only by External pin interrupt.

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

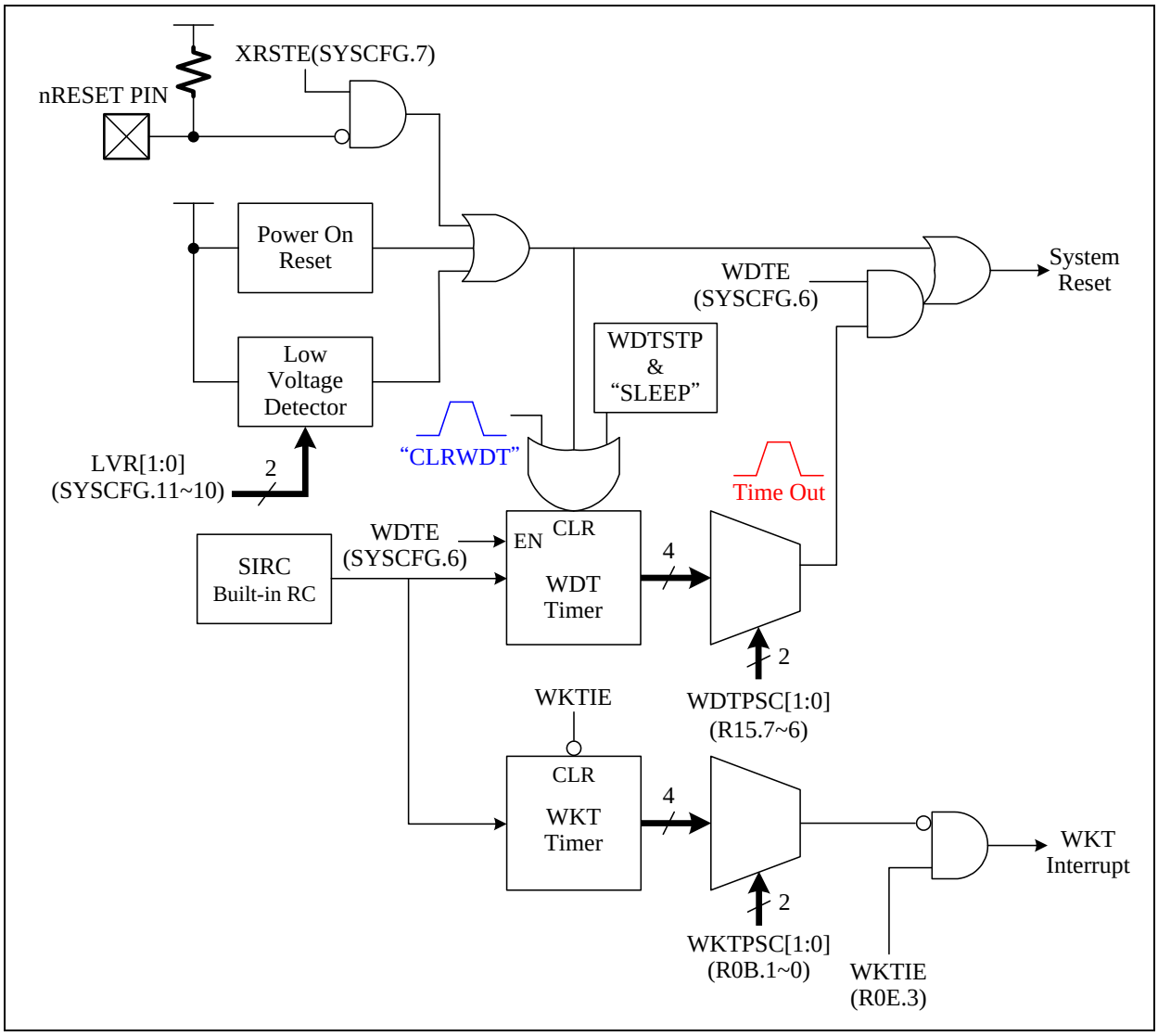
```
BCF     SLOWEN      ; Disable Slow-clock.
MOVLW   0000 0 000B ; Disable WKT counting
MOVWR   INTIE
MOVLW   01 1 00100B ; Stop WDT counting  in STOP mode
MOVWR   R15
SLEEP               ; Enter STOP mode.
```

# 3. Peripheral Functional Block

## 3.1 Watchdog (WDT) /Wakeup (WKT) Timer

The WDT and WKT share the same built-in internal RC Oscillator and have individual own counters. The overflow period of WDT, WKT can be selected by individual prescaler (WDTPSC[1:0], WKTPSC[1:0]). The WDT timer is cleared by the CLRWDT instruction. If the Watchdog is enabled (SYSCFG[7]=WDTE=1), the WDT generates the chip reset signal. Set WDTSTP (R15.5) to '1' can let WDT timer stop counting after executing SLEEP instruction, i.e. WDTSTP=0 WDT timer is always keep counting even if the SLEEP instruction is executed.

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




**WDT/WKT Block Diagram**

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

◇Example: Clear watchdog timer by CLRWDT instruction.

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

◇Example: Clear watchdog timer by write WDTCLR register.

```
MAIN:
        …                       ; Execute program.
        MOVWF       WDTCLR      ; Write any value into WDTCLR register.
        …
        GOTO        MAIN
```

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

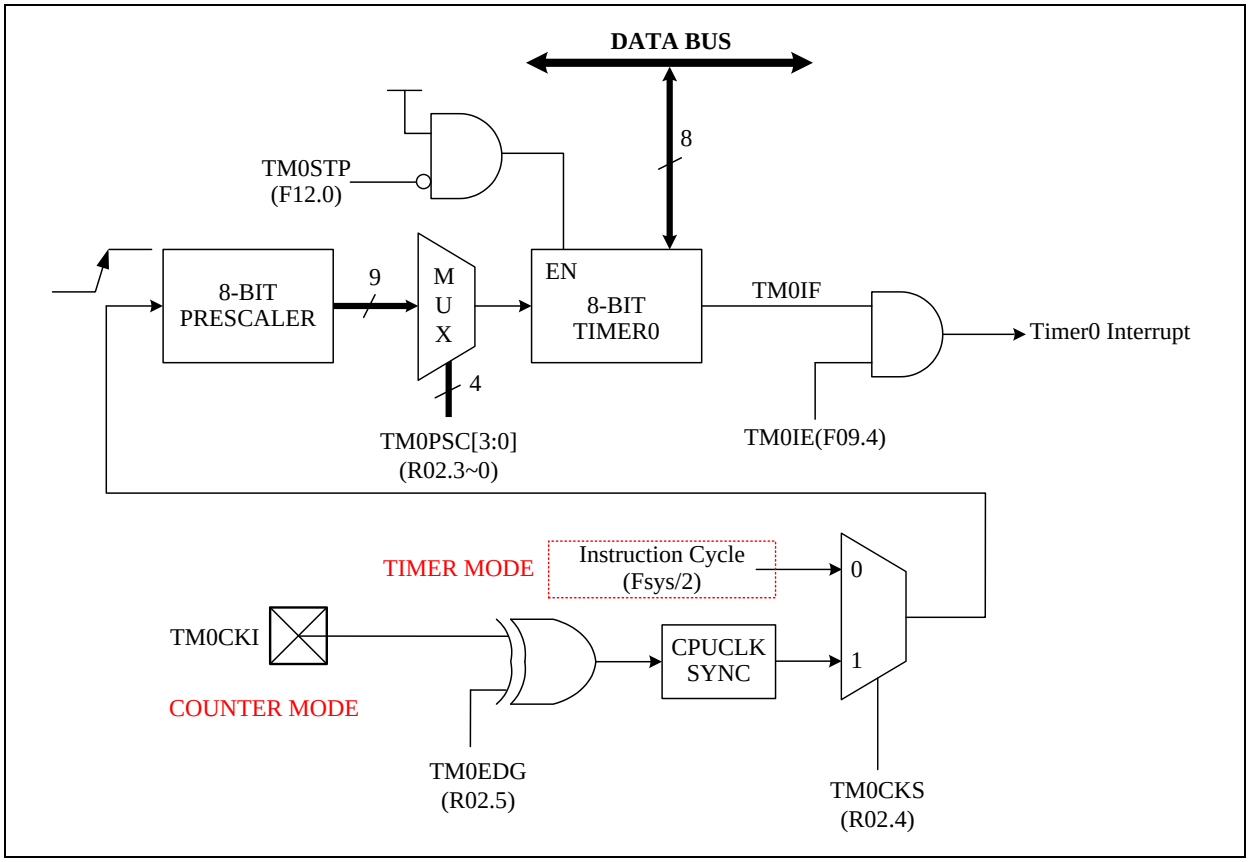
```
        MOVLW       00100100B
        MOVWR       R15         ; Select WDT Time out=120 ms @3V  (default 240 ms).
                                ; Setup WDT disable in IDLE/STOP mode (WDTSTP=0)
                                ; default WDT enable in IDLE/STOP.
        SLEEP
```

◇Example: Set WKT period and interrupt function.

```
        MOVLW       000000010B
        MOVWR       R0B         ; Select WKT period=60 ms @3V (default 120 ms).

        MOVLW       11110111B   ; Clear WKT interrupt request flag by using byte
operation
                                ; Don't use bit operation "BCF  WKTIF" clear interrupt flag
        MOVWF       INTIF       ; F-Plane 09H

        MOVLW       00001000B   ; Enable WKT interrupt function
        MOVWR       R0E
```

### 3.2 Timer0

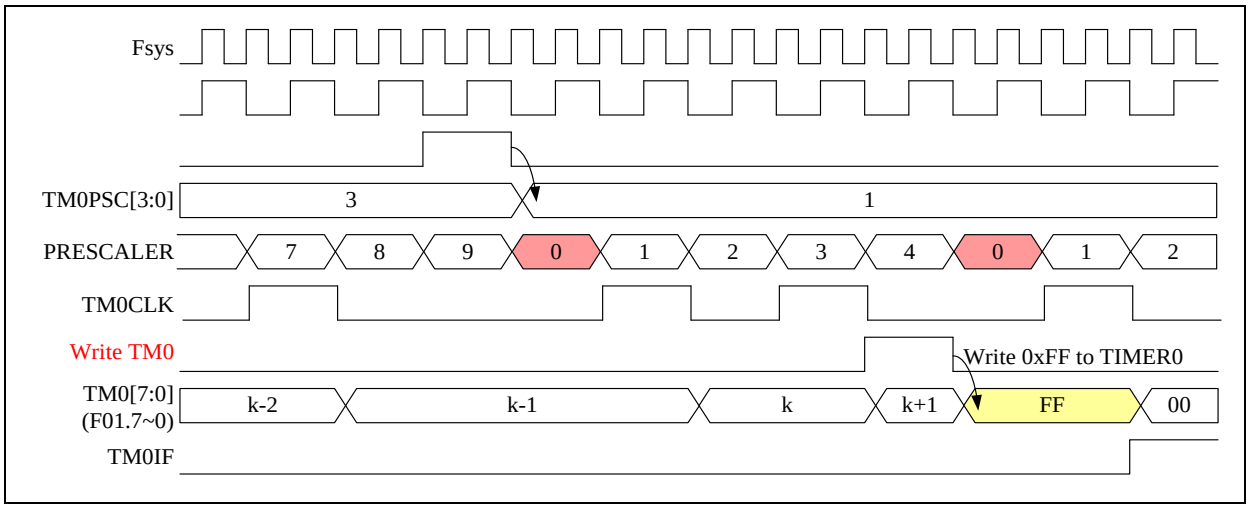
The Timer0 is an 8-bit wide register of F-Plane 01h (TM0). It can be read or written as any other register of F-Plane. Besides, Timer0 increases itself periodically and automatically rolls over based on the pre-scaled clock source, which can be the instruction cycle (Fsys/2) or TM0CKI (PA2) rising/falling input. The Timer0 increase rate is determined by "Timer0 Pre-Scale" (TM0PSC) register in R-Plane. The Timer0 always generates TM0IF when its count rolls over. It generates Timer0 Interrupt if (TM0IE) is set. Timer0 can be stopped counting if the TM0STP bit is set.




**Timer0 Block Diagram**

The following timing diagram describes the Timer0 works in pure Timer mode.

When the Timer0 prescaler (TM0PSC) is written, the internal 8-bit prescaler will be cleared to 0 to make the counting period correct at the first Timer0 count. TM0CLK is the internal signal that causes the Timer0 to increase by 1 at the end of TM0CLK. TM0WR is also the internal signal that indicates the Timer0 is directly written by instruction; meanwhile, the internal 8-bit prescaler will be cleared. When Timer0 counts from FFh to 00h, TM0IF (Timer0 Interrupt Flag) will be set to 1 and generate interrupt if TM0IE (Timer0 Interrupt Enable) is set.




**Timer0 works in Timer mode (TM0CKS=0)**

The equation of TM0 interrupt time value is as following:

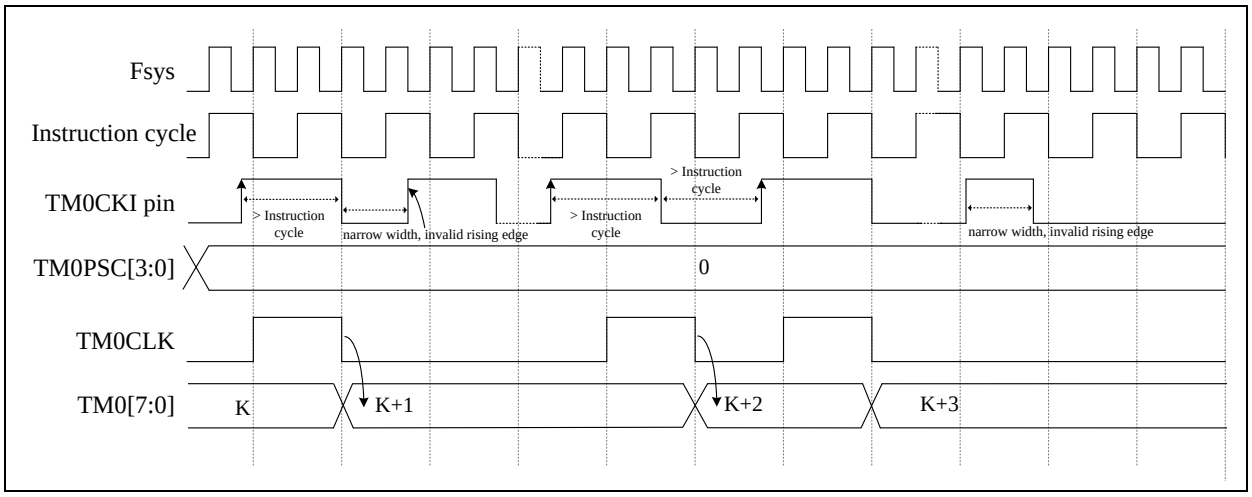
TM0 interrupt frequency = (Fsys/2)/TM0PSC/(256-TM0)

◇Example: Setup TM0 work in Timer mode

```
; Setup TM0 clock source and divider
        MOVLW   00000101B       ; R02.4=0, Setup TM0 clock = Fsys/2
        MOVWR   R02             ; R02.3~0=5 (TM0PSC)
                                ; TM0 clock prescaler = Fsys/64
; Set TM0 timer.
        BSF     TM0STP          ; Disable TM0 counting (Default "0").
        MOVLW   156
        MOVWF   TM0             ; Write 156 into TM0 register of F-Plane.(F01)

; Enable TM0 timer and interrupt function.
        MOVLW   11101111B       ; Clear TM0 request interrupt flag by byte operation
        MOVWF   INTIF           ; F-Plane 09H
        MOVLW   00010000B       ; Enable TM0 interrupt function
        MOVWR   INTIE           ; R0E
        BCF     TM0STP          ; Enable TM0 counting (Default “0”).
```

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

If TM0CKS=1 then Timer0 counter source clock is from TM0CKI pin. TM0CKI signal is synchronized by instruction cycle 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.



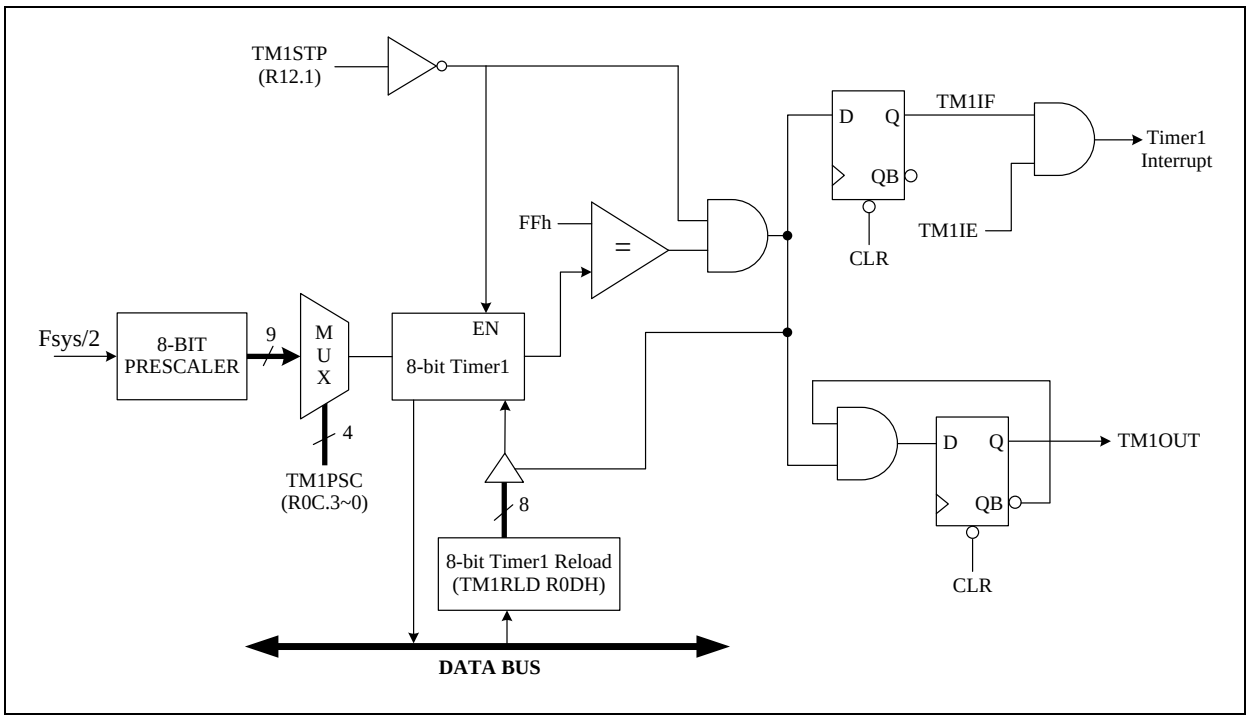

**Timer0 works in Counter mode for TM0CKI (TM0EDG=0), TM0CKS=1**

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

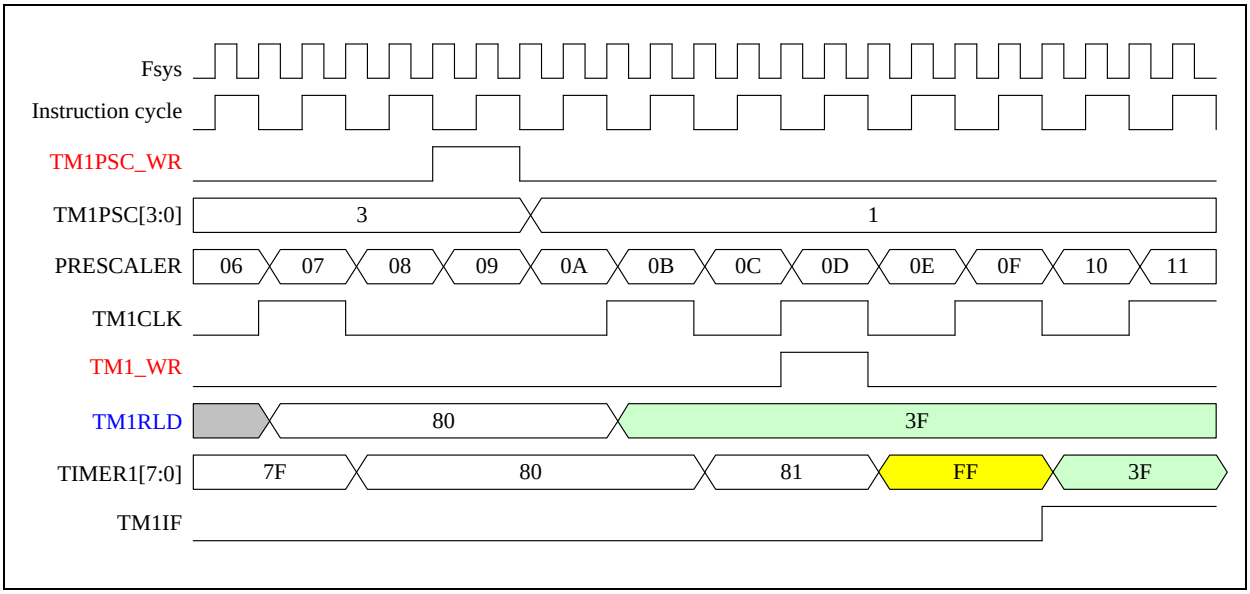
```
; Setup TM0 clock source from TM0CKI pin (PA2) and divider.
        MOVLW   00110000B
        MOVWR   R02             ; R02.5=1, Select TM0 prescaler counting edge=falling
                                  edge.
                                ; R02.4=1, Setup TM0 clock=TM0CKI pin(PA2)
                                ; R02.3~0=0 (TM0PSC)
                                ; TM0 clock prescaler=Fsys/2

; Set TM0 timer and stop TM0 counting.
        BSF     TM0STP          ; Disable TM0 counting (Default "0").
        MOVLW   00H
        MOVWF   TM0             ; Write 0 into TM0 register of F-Plane.

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

        MOVFW   TM0
```

### 3.3 Timer1

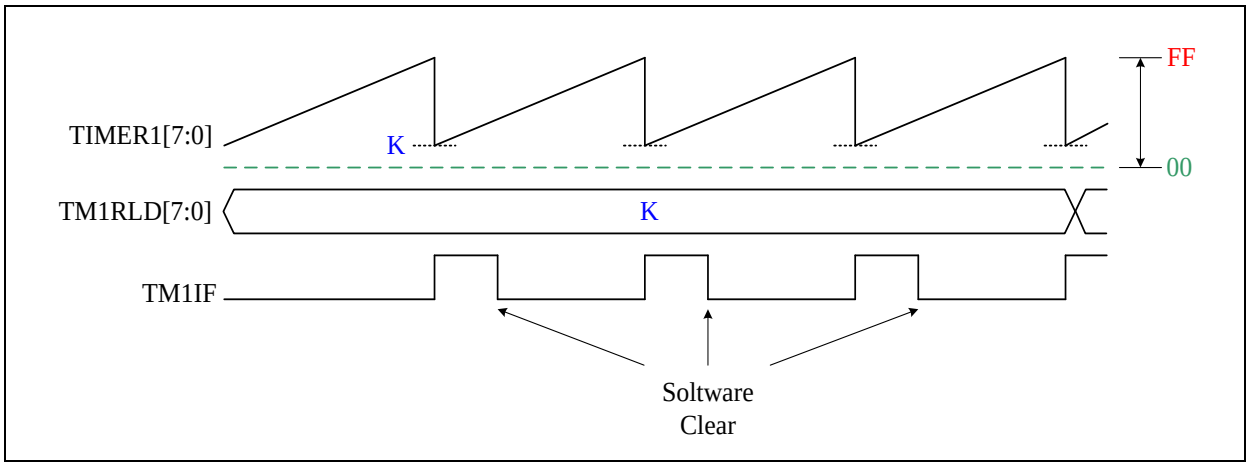
The Timer1 is an 8-bit wide register of F-Plane. It can be read or written as any other register of F-Plane. Besides, Timer1 increases itself periodically and automatically reloads a new "offset value" (TM1RLD) while it rolls over based on the pre-scaled instruction clock. The Timer1 increase rate is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




**Timer1 Block Diagram**




**Timer1 Timing Diagram**




**Timer1 Reload Diagram**

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

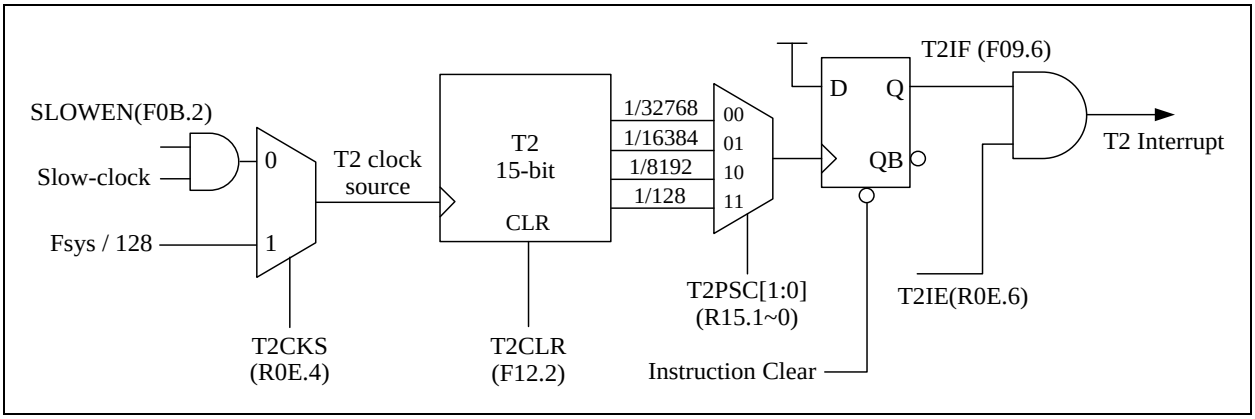
```
; Setup TM1 clock source, divider and enable TM1OUT
        MOVLW   00000101B
        MOVWR   R0C             ; R0C.3~0=5 (TM1PSC), Select TM1 clock=Fsys/64.
        MOVLW   00000100B
        MOVWR   R0B             ; R0B.2=1, Enable TM1OUT function pin (PD0).

; Set TM1 timer offset and stops TM1 counting
        BSF     TM1STP          ; Stop TM1 counting (Default "0").
        MOVLW   F0H
        MOVWF   TM1             ; Write F0H into TM1 counter (F0A, F-Plane)

; Enable TM1 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.
        MOVWR   INTIE           ;

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

Example:

Fsys=4 MHz, TM1PSC=1, TM1 clock source=Fsys/4=1 MHz
TM1RLD=0xF0,
TM1 interrupt time= (1/1 MHz) * (0xFF – 0xF0 ) =1 us*16=16 us
TM1OUT output time period=16 us *2=32 us.
TM1OUT output frequency=1/32 us=31.250 KHz.

### 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. It is used to generate time base interrupt and T2 counter block clock. The T2 content cannot be read by instructions. It generates interrupt flag T2IF (F09.6) with the clock divided by 32768/16384/8192/128 depends on T2PSC[1:0] (R15.1~0) register bits. The following figure shows the block diagram of T2.




**T2 Block Diagram**

Example:

[CPU running at FAST mode, Fsys=Fast-clock=FIRC 6 MHz]

◇Example:

```
; Setup T2 clock source and divider.
        MOVLW   xxx11001B       ; R15.4 (T2CKS) =1, T2 clock source=Fsys/128
        MOVWR   R15             ; R15.3~2 (FCLKDIV) =2, Fsys=Fast-clock=Fast-clock-;
                                  pre/2
                                ; Fsys=Fast-clock=12 MHz/2=6 MHz
                                ; R15.1~0 (T2PSC) =1, Divided by 16384
        BSF     T2CLR           ; F12.2 (T2CLR), Stop T2 counting.

; Enable T2 timer and interrupt function.
        MOVLW   10111111B       ; Clear T2 request interrupt flag by byte operation
        MOVWF   INTIF           ; F-Plane 09H

        MOVLW   01000000B       ; Enable T2 interrupt function.
        MOVWR   INTIE           ; R0E

        BCF     T2CLR           ; Enable T2 counting (Default “0”).
```

T2 clock source is Fsys/128=6 MHz/128=46875 Hz, T2PSC=/16384

T2 frequency=46875 Hz/16384=2.86 Hz

Example:

[CPU running at SLOW mode, Fsys=Slow-clock=SXT 32768 Hz]

◇Example:

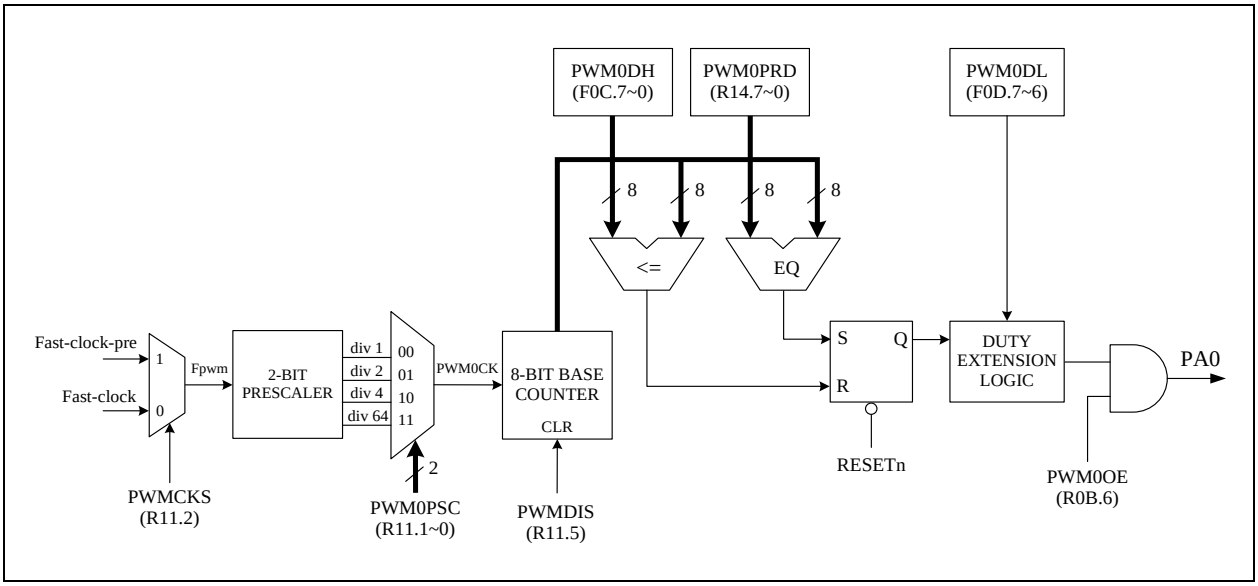
```
; Setup CPU running at SLOW mode
        MOVLW   00000000B       ;
        MOVWF   F0B             ; Slow-clock type=SXT (Default “01”)
        BSF     SLOWEN          ; Enable Slow-clock.
        NOP
        BSF     CPUCKS          ; Select Fsys=Slow-clock.
        BSF     FASTSTP         ; Stop Fast-clock.

; Setup T2 clock source and divider
        MOVLW   xxx0xx00B       ; R15.4 (T2CKS) =0, T2 clock source=Slow-clock
        MOVWR   R15             ; Fsys=Slow-clock=32768 Hz
                                ; R15.1~0 (T2PSC) =0, Divided by 32768

        BSF     T2CLR           ; Stop T2 counting.

; Enable T2 timer and interrupt function.
        MOVLW   10111111B       ; Clear T2 request interrupt flag
        MOVWF   INTIF           ; F-Plane 09H

        MOVLW   01000000B       ; Enable T2 interrupt function.
        MOVWR   INTIE           ; R0E

        BCF   T2CLR             ; Enable T2 counting (Default “0”).
```

T2 clock source is Slow-clock=32768 Hz, T2PSC=/32768,
T2 frequency=32768 Hz/32768=1 Hz

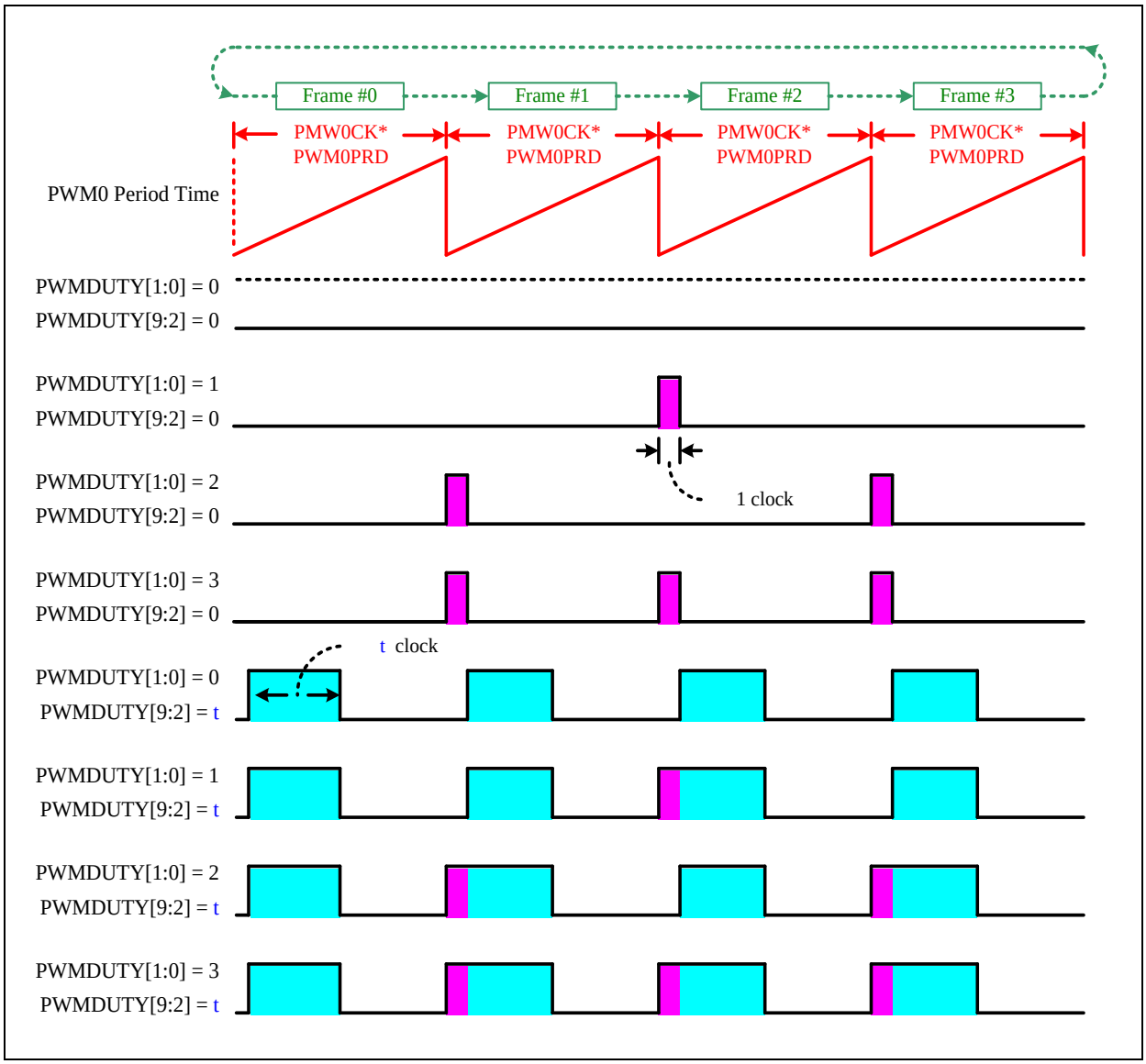
### 3.5 PWM0: (8+2) bits PWM

The PWM can generate fix frequency waveform with 1024 duty resolution based on Fpwm clock, the Fpwm can select Fast-clock or Fast-clock-pre by PWMCKS (R11.2). 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 PWM0DH (F0C.7~0). When the base counter rolls over, the 2-bit LSB of PWM duty register PWM0DL (F0D.7~6) decides whether to set the PWM output signal high immediately or set it high after one clock cycle delay.

The PWM0 period can be set by writing period value to PWM0PRD register (R14). Note that changing the PWM0PRD is immediately changing the PWM0PRD 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 PWM0PRD by observing the following figure. There is a digital comparator that compares the PWM0 counter and PWM0PRD, if PWM0 counter is larger than PWM0PRD after setting the PWM0PRD, a fault long PWM cycle will be generated because PWM0 counter must count to overflow then keep counting PWM0PRD to finish the cycle.




**PWM0 Block Diagram**




**PWM0 8+2 Timing Diagram**

Example:

[CPU running at FAST mode, Fsys=FIRC4M (FCLKDIV=1) ]

```
; Setup PWM0 clock source and prescaler .
        MOVLW   00000101B       ; R11.5 (PWMDIS)= 0, PWM0/PWM1 clock enable
        MOVWR   R11             ; R11.2 (PWMCKS)= 1, Fpwm=Fast-clock-pre(12 MHz)
                                ; R11.1~0 (PWM0PSC) =1,
                                  PWM clock source=Fpwm/2=6 MHz
        MOVLW   80H
        MOVWR   PWM0PRD         ; Set PWM0 period[7:0] =80H.

        MOVLW   00H
        MOVWF   PWM0DL          ; Set PWM0 duty[1:0] =0

        MOVLW   20H
        MOVWF   PWM0DH          ; Set PWM0 duty[9:2] =20H
        MOVLW   01000000B
        MOVWR   R0B             ; R0B.6 (PWM0OE) =1, Enable PWM0 OUT (PA0) .
```

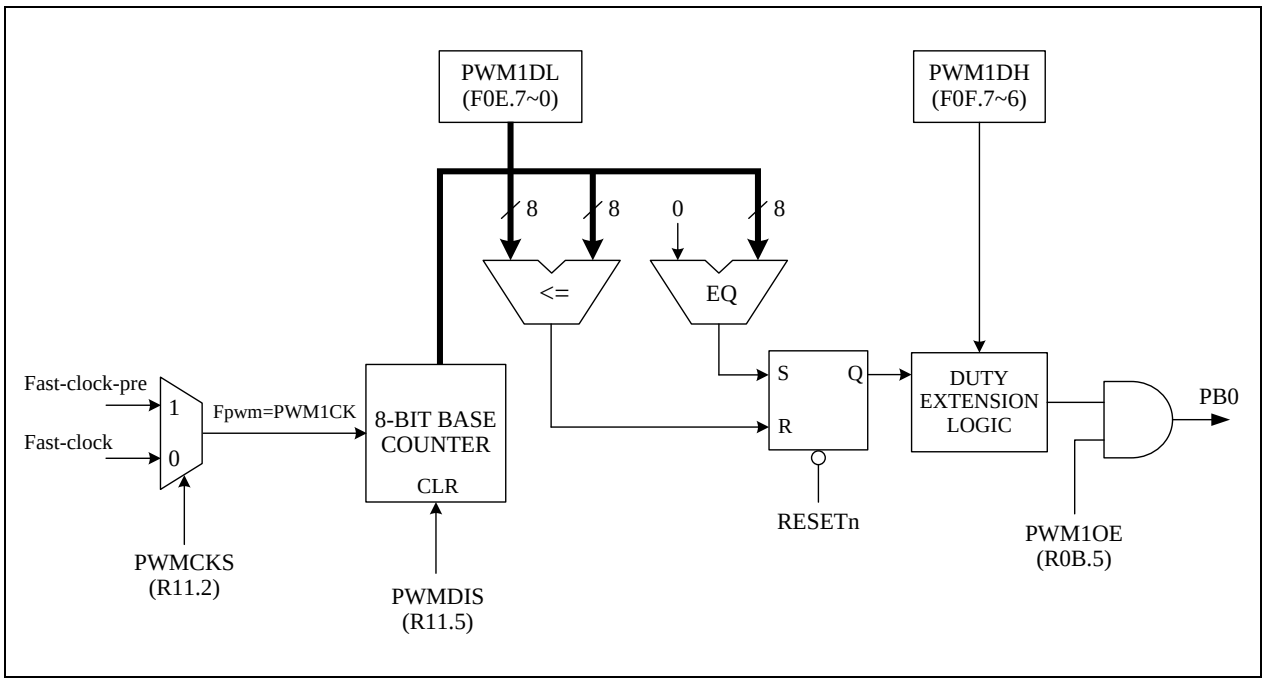
Fsys=4 MHz, PWM clock source=Fpwm/2=6 MHz, PWM0PRD=80H, PWM0DL=00H, PWM0DH =20H

PWM0 output frequency=6 MHz /(PWM0PRD+1) =6 MHz/129=46512 KHz

PWM0 output duty=32 /128=25%.

### 3.6 PWM1: (8+2) bits PWM

The PWM1 is similar with PWM0. The differences are: PWM1 period is fixed and PWM1 has not PWM1PSC.




**PWM1 Block Diagram**

Example:

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

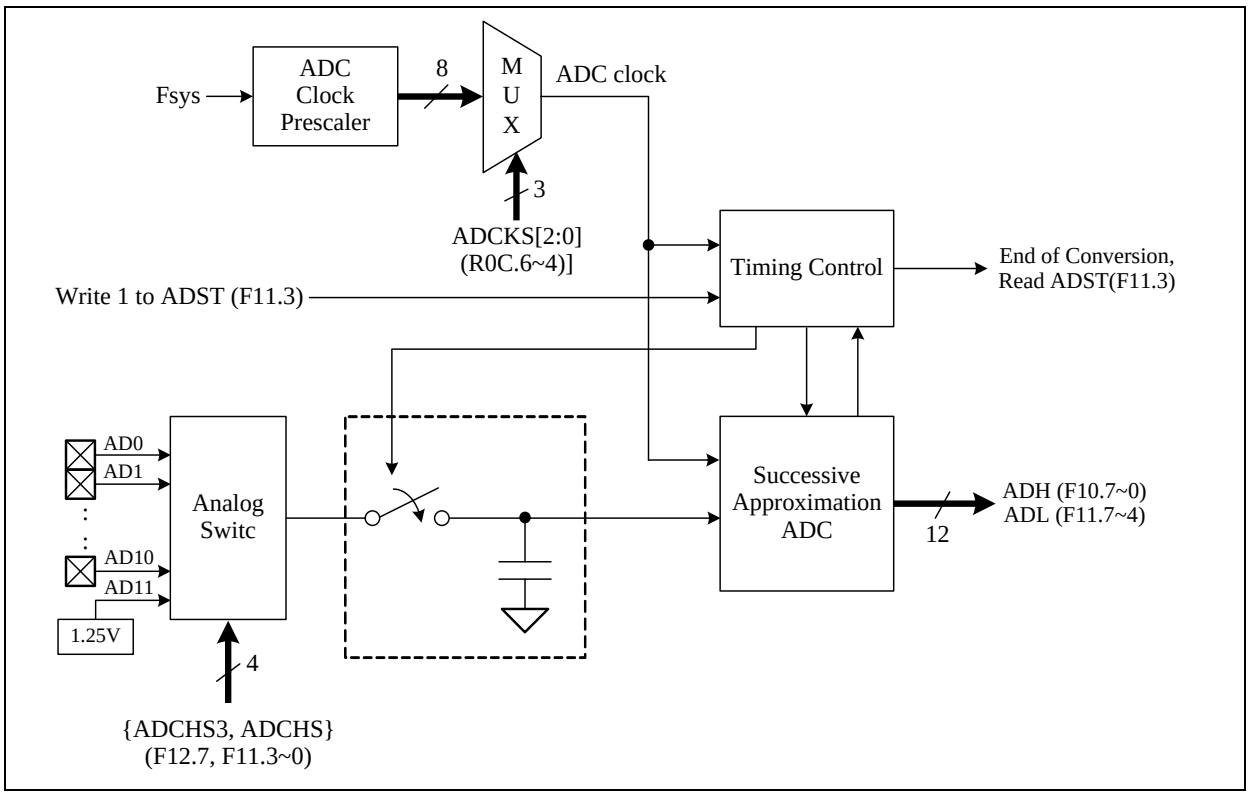
```
; Setup PWM1 clock source.
        MOVLW   00000000B       ; R11.5 (PWMDIS)= 0, PWM0/PWM1 clock enable
        MOVWR   R11             ; R11.2 (PWMCKS)= 0, Fpwm=Fast-clock(4 MHz)

        MOVLW   C0H
        MOVWF   F0F             ; Set PWM1 duty[1:0] =3

        MOVLW   80H
        MOVWF   PWM1DH          ; Set PWM1 duty[9:2] =80H

        MOVLW   00100000B
        MOVWR   R0B             ; R0B.5 (PWM1OE) =1, Enable PWM1 OUT (PB0)
```

Fsys=4 MHz, PWM clock source=Fast-clock = 4 MHz, PWM1DL=3, PWM1DH=80H

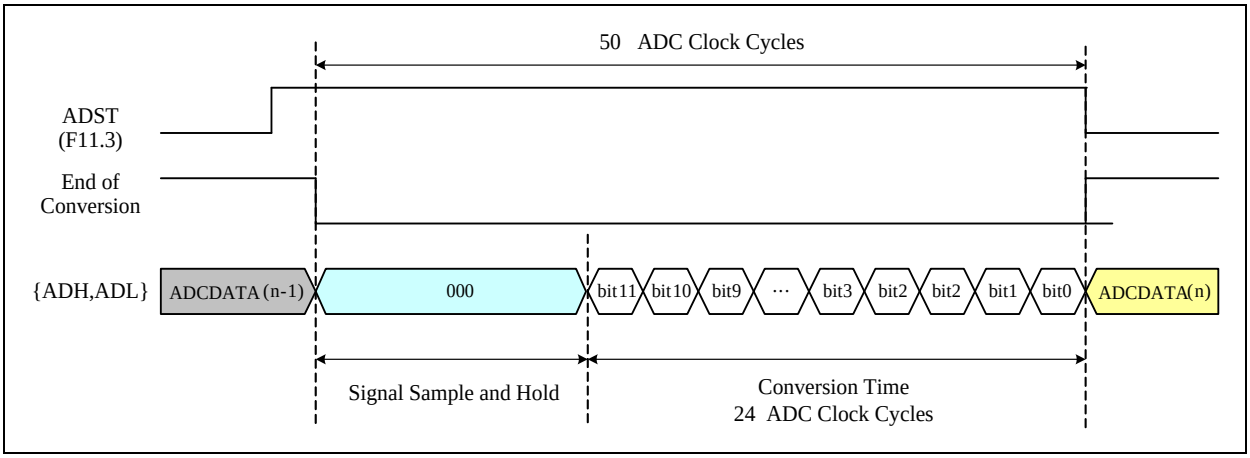
PWM1 output frequency=4 MHz/256=15.625 KHz

PWM1 output duty=128/256=50%.

### 3.7 Analog-to-Digital Converter




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 (R0C.6~4) to choose a proper ADC clock frequency, which must be less than 500KHz. User then launches the ADC conversion by setting the ADST (F11.3) control bit. After end of conversion, H/W automatic clears the ADST (F11.3) bit. User can poll this bit to know the conversion status. The ADPIE8 (R13.2~0), ADPIE (R12.7~0) 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.



Example1:

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

ADC clock frequency=500 KHz, ADC channel=ADC0 (PA6).

```
MOVLW   01010000B   ; Fsys=4 MHz
MOVWR   R0C         ; R0C.6~4 (ADCKS) =ADC clock=Fsys/8=500KHz

MOVLW   01000000B   ; ADC0 (PA6) pull-high disable
MOVWR   PAPUN;

MOVLW   11111110B
MOVWR   ADPIE       ; R12, Disable ADC0 (PA6) digital input to saving power
                    ; in STOP/IDLE Mode

MOVLW   00000000B
MOVWF   ADCTL       ; F11.2~0 (ADCHS[2:0])= 0, ADC select ADC0 (PA6 pin).
BCF     ADCHS3      ; F12.7 (ADCHS[3]) =0

BSF     ADST        ; F11.3 (ADST), ADC start conversion.

WAIT_ADC:
        BTFSC   ADST        ; Wait ADC conversion finish.
        GOTO    WAIT_ADC

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

Example2:

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

ADC clock frequency=750KHz, ADC channel=VBG

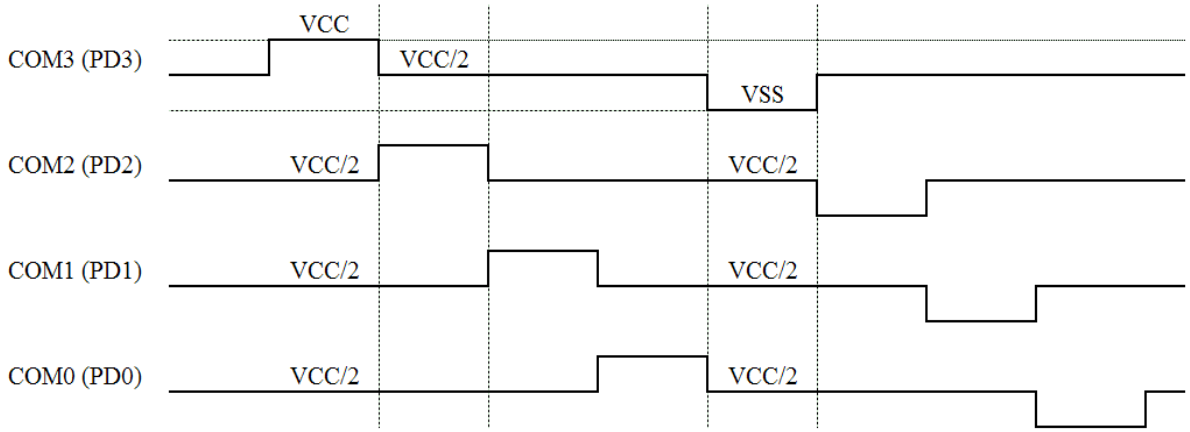
Choose any one AD ch (ex:ADC1) operating dummy ADC Low converting and then converts VBG ch

1. ADC1 (PA1) CMOS push-pull output LOW
2. ADCHS=ADC1(PA1)
3. ADST=1, and wait ADST=0 to finish conversion, throw out ADC result.
4. ADCHS=VBG
5. ADST=1, and wait ADST=0 to finish conversion

### 3.8 1/2 bias VCC/2 Voltage Generator

By setting SYSCFG (PD30IOE) option to be LCD mode, can enable entire PD[3:0] as LCD COM3~COM0 and will also always disable anyone PD[3:0] I/O function, meanwhile PDE[3:0] (R07.3~0) are invalid. LCD mode can output VCC、VCC/2、VSS three levels voltage to PD[3:0] by setting corresponding PDUN[3:0], PDD[3:0] shown as below pins state table. If enter STOP/IDLE mode, these PD[3:0] will automatically pull to VCC level.

| | **PD30IOE** (SYSCFG[9]) | **PDPUN**[3:0] (R0A.3~0) | **PDE**[3:0] (R07.3~0) | **PDD**[3:0] (F07.3~0) | Pins State |
|---|---|---|---|---|---|
| I/O mode | 1 | 1/0 | 1/0 | 1/0 | Normal IO |
| LCD mode | 0 | 0 | x | x | VCC/2 |
| | 0 | 1 | x | 0 | VSS |
| | 0 | 1 | x | 1 | VCC |




| PDPUN[3:0] | 4'b0000 | 4'b1000 | 4'b0100 | 4'b0010 | 4'b0001 | 4'b1000 | 4'b0100 | 4'b0010 | 4'b0001 |
|---|---|---|---|---|---|---|---|---|---|
| PDD[3:0] | 4'b1111 | 4'b1111 | 4'b1111 | 4'b1111 | 4'b1111 | 4'b0111 | 4'b1011 | 4'b1101 | 4'b1110 |
| PDE[3:0] | 4'bxxxx | 4'bxxxx | 4'bxxxx | 4'bxxxx | 4'bxxxx | 4'bxxxx | 4'bxxxx | 4'bxxxx | 4'bxxxx |

x: don't care

**1/2 bias VCC/2 voltage generator by S/W control (PD30IOE=0)**

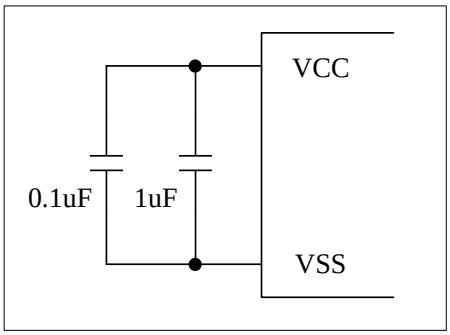
**Note: EV board (EV2781) don't support to emulate the PD3~PD0 LCD function, but *tenx* supports an AP note document to use #define, .IF, .ELSE, .ENDIF pseudo instructions to achieve functional check.**

◇Example:

```
MOVLW   x0000xxxB   ;
MOVWR   ADPIE8      ; PD0~PD3 set to non-digital input mode
MOVLW   xxxx1000B   ;
MOVWR   PDPUN       ; PD2~PD0 output VCC/2, PD3 depend on PDD[3]

MOVLW   xxxx0111B   ;
MOVWF   PDD         ; PD3 output VSS
```

### 3.9 System Clock Oscillator

System clock can be operated in two different oscillation modes. In the Fast Internal RC (FIRC) mode, the on-chip oscillator generates 12/6/4/1.5 MHz system clock, which is controlled by register FCLKDIV[1:0] (R15.3~2) bits. Since power noise degrades the performance of Internal Clock Oscillator, placing power supply bypass capacitors 1 uF and 0.1 uF very close to VCC/VSS pins improves the stability of clock and the overall system.

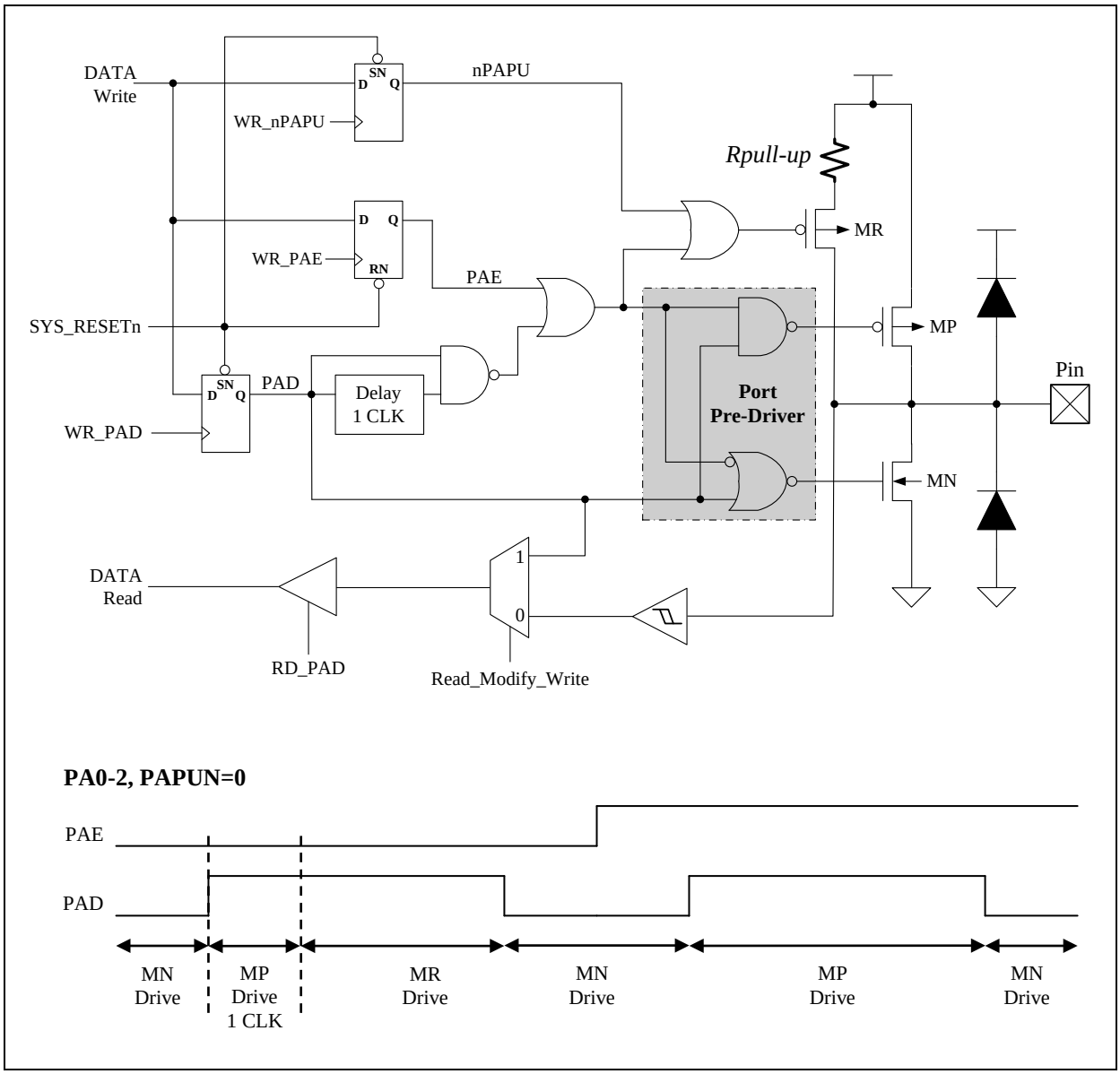



Internal RC Mode

# 4. I/O Port

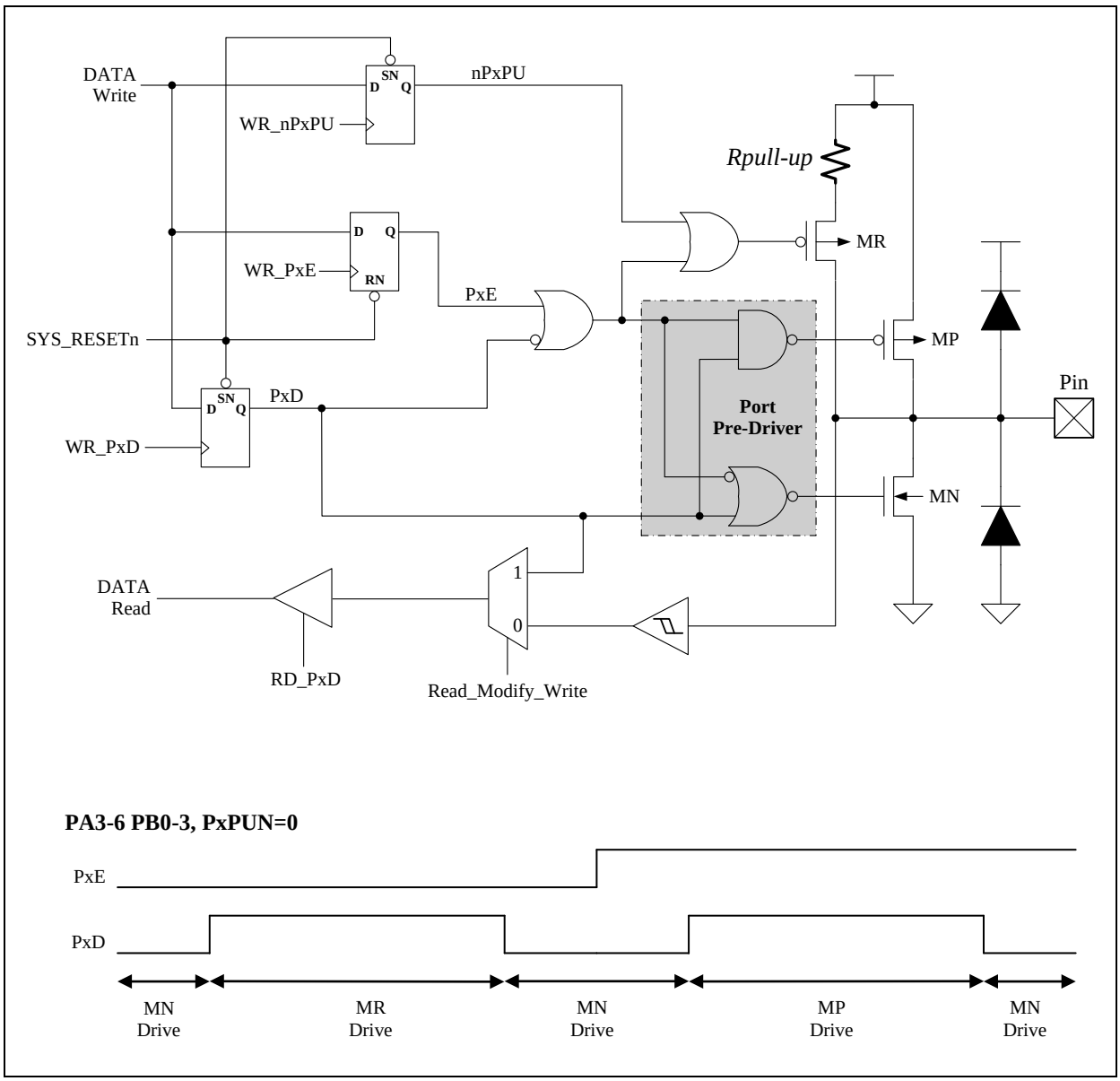
## 4.1 PA0-2

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. ◦



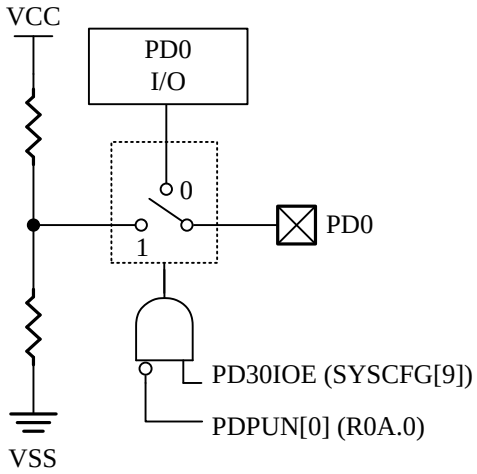
### 4.2 PA3-6, PB0-1, PD0-7

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.




**PA3-6 PB0-3, PxPUN=0**

PD0-PD3 pins architecture as below, more controls refer to the section “"3.8 1/2 bias VCC/2 Voltage Generator"



◇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   PAD    ; 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.

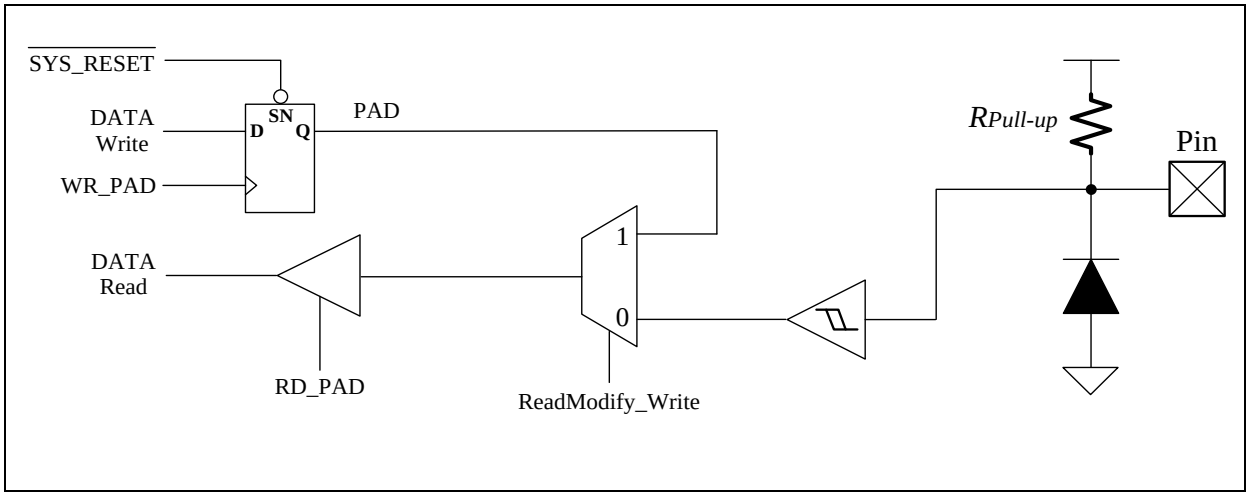
```
MOVLW   55H
MOVWF   PAD    ; Write data 55H to Port A.
MOVWF   PBD    ; Write data 55H to Port B.
```

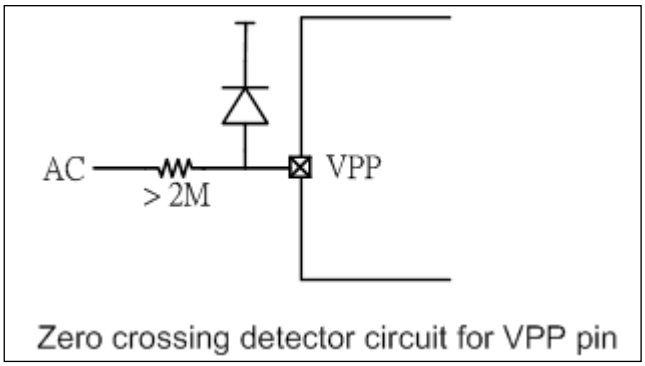
◇Example: Write one bit data to output port.

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

### 4.3 PA7

PA7 can be only used in Schmitt-trigger input mode. The pull-up resistor is always connected to this pin. PA7/VPP has no high voltage protection diode, need an external diode and resistor to achieve AC zero crossing detection




◇Example: Read state from PA7.

Condition: SYSCFG[7] is set to “0”. (If SYSCFG[7] = “1”, then PA7 pin to be external reset pin function.)

```
BTFSS    PAD,7
GOTO     LOOP_A      ; If PA7=0.
GOTO     LOOP_B      ; If PA7=1.
```

# MEMORY MAP

## F-Plane

| Name | Address | R/W | Rst | Description |
|---|---|---|---|---|
| **(F00) INDF** | | | | **Function related to: RAM W/R** |
| INDF | 00.7~0 | R/W | - | Not a physical register, addressing INDF actually point to the register whose address is contained in the FSR register |
| **(F01) TM0** | | | | **Function related to : Timer0** |
| TM0 | 01.7~0 | R/W | 0 | Timer0 content |
| **(F02) PCL** | | | | **Function related to: PROGRAM COUNT** |
| PCL | 02.7~0 | R/W | 0 | Programming Counter LSB[7~0] |
| **(F03) STATUS** | | | | **Function related to: STATUS** |
| 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 | RAM Bank Selection (Fixed to 0) |
| TO | 03.4 | R | 0 | WDT timeout flag, cleared by PWRST, ‘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 |
| C | 03.0 | R/W | 0 | Carry flag |
| **(F04) FSR** | | | | **Function related to: RAM W/R** |
| GB2 | 04.7 | R/W | 0 | General purpose bit 2 |
| FSR | 04.6~0 | R/W | - | File Select Register, indirect address mode pointer |
| **(F05) PAD** | | | | **Function related to: Port A** |
| PAD7 | 05.7 | R | - | PA7 pin state |
| 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 |

| Name | Address | R/W | Rst | Description |
|---|---|---|---|---|
| **(F09) INTIF** | | | | **Function related to: Interrupt Flag** |
| | 09.7 | | | Reserved |
| T2IF | 09.6 | R | - | T2 interrupt event pending flag, set by H/W while T2 overflows |
| | | W | 0 | write 0: clear this flag; write 1: no action |
| TM1IF | 09.5 | R | - | Timer1 interrupt event pending flag, set by H/W while Timer1 overflows |
| | | W | 0 | write 0: clear this flag; write 1: no action |
| TM0IF | 09.4 | R | - | Timer0 interrupt event pending flag, set by H/W while Timer0 overflows |
| | | W | 0 | write 0: clear this flag; write 1: no action |
| WKTIF | 09.3 | R | - | WKT interrupt event pending flag, set by H/W while WKT time out |
| | | W | 0 | write 0: clear this flag; write 1: no action |
| INT2IF | 09.2 | R | - | INT2(PA7) interrupt event pending flag, set by H/W at INT2 pin's falling edge |
| | | W | 0 | write 0: clear this flag; write 1: no action |
| INT1IF | 09.1 | R | - | INT1(PA1) interrupt event pending flag, set by H/W at INT1 pin's f/r edge |
| | | W | 0 | write 0: clear this flag; write 1: no action |
| INT0IF | 09.0 | R | - | INT0(PA6) interrupt event pending flag, set by H/W at INT0 pin's falling edge |
| | | W | 0 | write 0: clear this flag; write 1: no action |
| **(F0A) TM1** | | | | **Function related to: Timer1** |
| TM1 | 0a.7~0 | R/W | 0 | Timer1 content |
| **(F0B) CLKS** | | | | **Function related to: CPUCLK** |
| | 0b.7 | | | Reserved |
| SIRCKS | 0b.6~5 | R/W | 3 | SIRC clock selection (@VCC=3V)<br>00:140KHz 01:35KHz 10:8.75KHz 11:2.2KHz (not precisely) |
| FASTSTP | 0b.4 | R/W | 0 | Fast-clock Enable / Disable<br>0:Enable 1:Disable |
| CPUCKS | 0b.3 | R/W | 0/1<1> | System clock (Fsys) selection, the reset value depends on SYSCFG[5].<br>0: Fast-clock 1: Slow-clock<br><1> If MODE3V (SYSCFG[5]) =1, the value is 1, otherwise is 0 |
| SLOWEN | 0b.2 | R/W | 1 | If CPUCKS=1, this SLOWEN bit is invalid, Slow-clock keep oscillating<br>If CPUCKS=0, Set 1 to enable Slow-clock oscillate, Clear 0 to stop Slow-clock oscillating<br>Set SLOWEN=0 to save power before enter STOP mode. |
| SLOWCKS | 0b.1~0 | R/W | 1 | Slow-clock type<br>01: SIRC (see SIRCKS), should fixed to 1 |
| **(F0C) PWM0DH** | | | | **Function related to: PWM0** |
| PWM0DH | 0c.7~0 | R/W | 0 | PWM0 duty 8-bit MSB |
| **(F0D) PWM0DL** | | | | **Function related to: PWM0** |
| PWM0DL | 0d.7~6 | R/W | 0 | PWM0 duty 2-bit LSB |
| | 0d.5~0 | | | Reserved |

| Name | Address | R/W | Rst | Description |
|---|---|---|---|---|
| **(F0E) PWM1DH** | | | | **Function related to: PWM1** |
| PWM1DH | 0e.7~0 | R/W | 0 | PWM1 duty 8-bit MSB |
| **(F0F) MF0F** | | | | **Function related to : PWM1 / Table Read** |
| PWM1DL | 0f.7~6 | R/W | 0 | PWM1 duty 2-bit LSB |
| | 0f.5~3 | | | Reserved |
| DPH | 0f.2~0 | R/W | 0 | Table read high address, data rom pointer (DPTR) high byte |
| **(F10) ADH** | | | | **Function related to: ADC** |
| ADH | 10.7~0 | R | - | ADC output data MSB, ADQ[11:4] |
| **(F11) ADCTL** | | | | **Function related to: ADC** |
| ADL | 11.7~4 | R | - | ADC output data LSB, ADQ[3:0] |
| ADST | 11.3 | R/W | 0 | ADC start bit. 0: H/W clear after end of conversion 1: ADC start conversion, |
| ADCHS | 11.2~0 | R/W | 0 | ADC channel select bit2~bit0. {ADCHS3(F12.7), ADCHS} =<br>0000: ADC0 (PA6) 0100: ADC4 (PD7) 1000: ADC8 (PA0)<br>0001: ADC1 (PA1) 0101: ADC5 (PA5) 1001: ADC9 (PD5)<br>0010: ADC2 (PA2) 0110: ADC6 (PD6) 1010: ADC10 (PD4)<br>0011: ADC3 (PB1) 0111: ADC7 (PB0) 1011: 1.25v(VBG) |
| **(F12) MF12** | | | | **Function related to: ADC/T2/TM1/TM0** |
| ADCHS3 | 12.7 | R/W | 0 | ADC channel select bit3 |
| | 12.6~3 | | | Reserved |
| T2CLR | 12.2 | R/W | 1 | T2 counter clear 0: Release 1: Clear and hold |
| TM1STP | 12.1 | R/W | 0 | Timer1 counter stop 0: Release 1: Stop counting |
| TM0STP | 12.0 | R/W | 0 | Timer0 counter stop 0: Release 1: Stop counting |
| **(F13) DPL** | | | | **Function related to: Table Read** |
| DPL | 13.7~0 | R/W | 0 | Table read low address, data ROM pointer (DPTR) low byte |
| **User Data Memory** | | | | |
| SRAM | 20~27 | R/W | - | RAM common area (8 Bytes) |
| | 28~7F | R/W | - | RAM BANK0 area (RAMBK=0, 88 Bytes) |

## R-Plane

| Name | Address | R/W | Rst | Description |
|---|---|---|---|---|
| **(R02) TM0CTL** | | | | **Function related to: TCOUT/TM0** |
| TCOPSC | 02.7~6 | W | 0 | TCOUT prescaler<br>0: Fsys/2 1: Fsys/4 2: Fsys/8 3: Fsys/16 |
| TM0EDG | 02.5 | W | 0 | Timer0 prescaler counting edge for TM0CKI (PA2) pin<br>0: rising edge 1: falling edge |
| TM0CKS | 02.4 | W | 0 | Timer0 prescaler clock source<br>0:Fsys/2 1: TM0CKI (PA2) pin |
| TM0PSC | 02.3~0 | W | 0 | Timer0 prescaler. Timer0 prescaler clock source divided by<br>0000: Fsys/2 0101: Fsys/64<br>0001: Fsys/4 0110: Fsys/128<br>0010: Fsys/8 0111: Fsys/256<br>0011: Fsys/16 1xxx: Fsys/512<br>0100: Fsys/32 |
| **(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<br>0: the pin is open-drain output or Schmitt-trigger input<br>1: the pin is CMOS push-pull output |
| | 05.2~0 | W | 0 | Each bit controls its corresponding pin, if the bit is<br>0: the pin is pseudo-open-drain output or Schmitt-trigger input<br>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<br>0: the pin is open-drain output or Schmitt-trigger input<br>1: the pin is CMOS push-pull output |
| **(R07) PDE** | | | | **Function related to: Port D** |
| PDE | 07.7~0 | W | 0 | **When PD30IOE=1**, each bit controls its corresponding pin, if the bit is<br>0: the pin is open-drain output or Schmitt-trigger input<br>1: the pin is CMOS push-pull output<br>**When PD30IOE=0**, PDE[3:0] will be invaild. |
| **(R08) PAPUN** | | | | **Function related to: Port A** |
| PAPUN | 08.6~0 | W | 7F | Each bit controls its corresponding pin, if the bit is<br>0: the pin pull up resistor is enabled, except<br>a. the pin’s output data register (PAD) is 0<br>b. the pin’s CMOS push-pull mode is chosen (PAE=1)<br>c. the pin is working for PWM output<br>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<br>0: the pin pull up resistor is enabled, except<br>a. the pin’s output data register (PBD) is 0<br>b. the pin’s CMOS push-pull mode is chosen (PBE=1)<br>c. the pin is working for PWM output<br>1: the pin pull up resistor is disabled |

| Name | Address | R/W | Rst | Description |
|---|---|---|---|---|
| **(R0A) PDPUN** | | | | **Function related to: Port D** |
| PDPUN | 0a.7~0 | W | FF | **When PD30IOE=1**, each bit controls its corresponding pin, if the bit is<br>0: the pin pull up resistor is enabled, except<br>a. the pin's output data register (PDD) is 0<br>b. the pin's CMOS push-pull mode is chosen (PDE=1)<br>c. pins are working for TM1OUT/TCOUT output<br>1: the pin pull up resistor is disabled<br>**When PD30IOE=0**, each PDPUN[3:0] corresponding bit is<br>0: the pin can output VCC/2<br>1: the pin can output VCC or VSS, depend on PDD value |
| **(R0B) MR0B** | | | | **Function related to: PWM0/PWM1/INT1/TCOUT/TM1/WKT** |
| | 0b.7 | | | Reserved |
| PWM0OE | 0b.6 | W | 0 | Enable PWM0 output to PA0 pin |
| PWM1OE | 0b.5 | W | 0 | Enable PWM1 output to PB0 pin |
| INT1EDG | 0b.4 | W | 0 | INT1 pin (PA1) edge interrupt event<br>0: falling edge to trigger 1: rising edge to trigger |
| TCOE | 0b.3 | W | 0 | Enable post-prescaler Instruction Cycle(Fsys/2) output to PD6 pin (TCOUT) |
| TM1OE | 0b.2 | W | 0 | Enable Timer1 overflow toggle output to PD0 pin (TM1OUT) |
| WKTPSC | 0b.1~0 | W | 3 | WKT period (@VCC=3V)<br>00: 15 ms 01: 30 ms 10: 60 ms 11: 120 ms |
| **(R0C) MR0C** | | | | **Function related to: ADC/TM1** |
| | 0c.7 | | | Reserved |
| ADCKS | 0c.6~4 | W | 0 | ADC clock frequency selection:<br>000: Fsys/256 100: Fsys/16<br>001: Fsys/128 101: Fsys/8<br>010: Fsys/64 110: Fsys/4<br>011: Fsys/32 111: Fsys/2 |
| TM1PSC | 0c.3~0 | W | 0 | Timer1 prescaler. Timer1 clock source divided by<br>0000: Fsys/2 0101: Fsys/64<br>0001: Fsys/4 0110: Fsys/128<br>0010: Fsys/8 0111: Fsys/256<br>0011: Fsys/16 1xxx: Fsys/512<br>0100: Fsys/32 |
| **(R0D) TM1RLD** | | | | **Function related to: TM1** |
| TM1RLD | 0d.7~0 | W | 0 | Timer1 reload offset value while it rolls over |
| **(R0E) INTIE** | | | | **Function related to: Interrupt Enable** |
| | 0e.7 | | | Reserved |
| T2IE | 0e.6 | W | 0 | T2 interrupt enable, 1=enable, 0=disable |
| TM1IE | 0e.5 | W | 0 | Timer1 interrupt enable, 1=enable, 0=disable |
| TM0IE | 0e.4 | W | 0 | Timer0 interrupt enable, 1=enable, 0=disable |
| WKTIE | 0e.3 | W | 0 | Wakeup Timer interrupt enable, 1=enable, 0=disable<br>Set 0 to clear & disable WKT timer |
| INT2IE | 0e.2 | W | 0 | INT2 pin(PA7) interrupt enable, 1=enable, 0=disable |
| INT1IE | 0e.1 | W | 0 | INT1 pin(PA1) interrupt enable, 1=enable, 0=disable |
| INT0IE | 0e.0 | W | 0 | INT0 pin(PA6) interrupt enable, 1=enable, 0=disable |
| **(R0F) TEST** | | | | |
| TSTREG | 0f.7~0 | W | 0 | Test mode register, user does not write it. |

| Name | Address | R/W | Rst | Description |
|---|---|---|---|---|
| **(R11) MR11** | | | | **Function related to: EFT/PWM0/PWM1** |
| | 11.7 | W | 0 | Reserved, keep write to 0 |
| VCCFLT | 11.6 | W | 0 | Enable EFT enhance operation mode, 1=enable, 0=disable |
| PWMDIS | 11.5 | W | 0 | PWM0 / PWM1 clock disable(PWMDIS=1) or enable(PWMDIS=0) |
| | 11.4~3 | | | Reserved |
| PWMCKS | 11.2 | W | 0 | PWM0 / PWM1 clock source, Fpwm=<br>0: Fast-clock 1: Fast-clock-pre |
| PWM0PSC | 11.1~0 | W | 0 | PWM0 prescaler, PWM0 clock source divided by<br>00: Fpwm 01: Fpwm/2 10: Fpwm/4 11: Fpwm/64 |
| **(R12) ADPIE** | | | | **Function related to: ADC** |
| ADPIE | 12.7 | W | 1 | Each bit controls its corresponding port I/O enable pin, if the bit is<br>0: enable ADC7 channel input 1: enable PB0 I/O digital input |
| | 12.6 | W | 1 | 0: enable ADC6 channel input 1: enable PD6 I/O digital input |
| | 12.5 | W | 1 | 0: enable ADC5 channel input 1: enable PA5 I/O digital input |
| | 12.4 | W | 1 | 0: enable ADC4 channel input 1: enable PD7 I/O digital input |
| | 12.3 | W | 1 | 0: enable ADC3 channel input 1: enable PB1 I/O digital input |
| | 12.2 | W | 1 | 0: enable ADC2 channel input 1: enable PA2 I/O digital input |
| | 12.1 | W | 1 | 0: enable ADC1 channel input 1: enable PA1 I/O digital input |
| | 12.0 | W | 1 | 0: enable ADC0 channel input 1: enable PA6 I/O digital input |
| **(R13) ADPIE8** | | | | **Function related to: ADC** |
| ADPIE8 | 13.7 | | | Reserved |
| | 13.6 | W | 1 | 0: enable non-digital input 1: enable PD0 I/O digital input |
| | 13.5 | W | 1 | 0: enable non-digital input 1: enable PD1 I/O digital input |
| | 13.4 | W | 1 | 0: enable non-digital input 1: enable PD2 I/O digital input |
| | 13.3 | W | 1 | 0: enable non-digital input 1: enable PD3 I/O digital input |
| | 13.2 | W | 1 | 0: enable ADC10 channel input 1: enable PD4 I/O digital input |
| | 13.1 | W | 1 | 0: enable ADC9 channel input 1: enable PD5 I/O digital input |
| | 13.0 | W | 1 | 0: enable ADC8 channel input 1: enable PA0 I/O digital input |

| Name | Address | R/W | Rst | Description |
|---|---|---|---|---|
| **(R14) PMW0PRD** | | | | **Function related to: PWM0** |
| PWM0PRD | 14.7~0 | W | FF | PWM0 period data |
| **(R15) MR15** | | | | **Function related to: WDT/T2/CPUCLK** |
| WDTPSC | 15.7~6 | W | 1 | WDT period (@VCC=5V)<br>00: 110 ms 01: 210 ms 10: 840 ms 11: 1680 ms |
| WDTSTP | 15.5 | W | 0 | WDT disable in IDLE/STOP mode, If WDTE=0, this bit is invalid<br>1: clear & stop counting 0: always counting |
| T2CKS | 15.4 | W | 0 | "T2 clock source" selection. 1: Fsys/128 0: Slow-clock |
| FCLKDIV | 15.3~2 | W | 1 | Fast-clock divider, Fast-clock is<br>00: Fast-clock-pre/8 01: Fast-clock-pre/3 10: Fast-clock-pre/2 11: Fast-clock-pre |
| T2PSC | 15.1~0 | W | 0 | T2 prescaler. "T2 clock source" divided by -<br>00: 32768 01: 16384 10: 8192 11: 128 |

# 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 |
| TO | WDT Time Out Flag |
| PD | Power Down Flag |
| W | Working Register |
| Z | Zero Flag |
| C | Carry Flag |
| DC | Decimal Carry 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 |
| 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 |
| TABRL | | 00 0000 0101 0000 | 2 | - | Lookup ROM low data to W |
| TABRH | | 00 0000 0101 1000 | 2 | - | Lookup ROM high data to W |
| SLEEP | | 00 0000 0000 0011 | 1 | TO, PD | Go into Power-down mode, Clock oscillation stops |
| 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<br>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<br>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<br>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<br>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<br>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<br>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<br>TRUE GOTO SUB1<br>FALSE ... | B : PC = LABEL1<br>A : if FLAG.1 = 0, PC = FALSE<br>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<br>TRUE GOTO SUB1<br>FALSE ... | B : PC = LABEL1<br>A : if FLAG.1 = 0, PC = TRUE<br>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<br>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<br>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<br>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 = ?<br>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<br>B : REG1 = 0x13<br>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<br>B : CNT = 0x01, Z = 0<br>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<br>GOTO LOOP<br>CONTINUE<br>B : PC = LABEL1<br>A : CNT = CNT – 1<br>if CNT = 0, PC = CONTINUE<br>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<br>B : PC = LABEL1<br>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<br>B : CNT = 0xFF, Z = 0<br>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<br>GOTO LOOP<br>CONTINUE<br>B : PC = LABEL1<br>A : CNT = CNT + 1<br>if CNT = 0, PC = CONTINUE<br>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<br>B : W = 0x9A<br>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<br>B : RESULT = 0x13, W = 0x91<br>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 = ?<br>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 = ?<br>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<br>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<br>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<br>:<br>TABLE ADDWF PCL, 1<br>RETLW k1<br>RETLW k2<br>:<br>RETLW kn | B : W = 0x07<br>A : W = value of k8 |

## TABRL Return DPTR low byte to W

| | |
|---|---|
| Syntax | TABRL |
| Operands | - |
| Operation | (W) ← ROM[DPTR] low byte content, Where DPTR={DPH[max:8],DPL[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   DPL             ; Where DPL 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 14 bits
```

## TABRH Return DPTR high byte to W

| | |
|---|---|
| Syntax | TABRH |
| Operands | - |
| Operation | (W) ← ROM[DPTR] high byte content, Where DPTR={DPH[max:8],DPL[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 |

## 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<br>A : REG1 = 1110 0110<br>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<br>A : REG1 = 1110 0110<br>W = 0111 0011, C = 0 |

## SLEEP — Go into standby mode, Clock oscillation stops

| | |
|---|---|
| Syntax | SLEEP |
| Operands | - |
| Operation | - |
| Status Affected | TO, PD |
| OP-Code | 00 0000 0000 0011 |
| Description | Go into STOP mode with the oscillator stops. |
| Cycle | 1 |
| Example | SLEEP — - |

## 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 = ?<br>A : REG1 = 0x01, W = 0x02, C = 1, Z = 0 |
| | SUBWF REG1, 1 — B : REG1 = 0x02, W = 0x02, C = ?, Z = ?<br>A : REG1 = 0x00, W = 0x02, C = 1, Z = 1 |
| | SUBWF REG1, 1 — B : REG1 = 0x01, W = 0x02, C = ?, Z = ?<br>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<br>A : REG1 = 0xA5, W = 0x5A |

| 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 = ?<br>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<br>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<br>A : REG = 0x1A, W = 0xB5 |

# ELECTRICAL CHARACTERISTICS

## 1. Absolute Maximum Ratings ($T_A$=25°C)

| Parameter | Rating | Unit |
|---|---|---|
| Supply voltage | $V_{SS}$ -0.3 to $V_{SS}$ +6.5 | V |
| Input voltage | $V_{SS}$ -0.3 to $V_{CC}$ +0.3 | |
| Output voltage | $V_{SS}$ -0.3 to $V_{CC}$ +0.3 | |
| Output current high per 1 PIN | -25 | mA |
| Output current high per all PIN | -80 | |
| Output current low per 1 PIN | +30 | |
| Output current low per all PIN | +150 | |
| Maximum operating voltage | 5.5 | V |
| Operating temperature | -40 to +85 | °C |
| Storage temperature | -65 to +150 | |

## 2. DC Characteristics ($T_A$=25°C, $V_{CC}$=2.0V to 5.5V)

| Parameter | Sym | Conditions | | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|---|
| Input High Voltage | $V_{IH}$ | All Input except PA7 | $V_{CC}$=3~5V | 0.6Vcc | | Vcc | V |
| | | PA7 | $V_{CC}$=3~5V | 0.7Vcc | | Vcc | V |
| Input Low Voltage | $V_{IL}$ | All Input except PA7 | $V_{CC}$=3~5V | Vss | | 0.2Vcc | V |
| | | PA7 | $V_{CC}$=3~5V | Vss | | 0.2Vcc | V |
| Output High Current | $I_{OH}$ | All Output | $V_{CC}$=5V,$V_{OH}$=4.5V | 4 | 8 | | mA |
| | | | $V_{CC}$=3V,$V_{OH}$=2.7V | 2 | 4 | | |
| Output Low Current | $I_{OL}$ | All Output | $V_{CC}$=5V,$V_{OL}$=0.5V | 9 | 18 | | mA |
| | | | $V_{CC}$=3V,$V_{OL}$=0.3V | 4 | 8 | | |
| Input Leakage Current (pin high) | $I_{ILH}$ | All Input | $V_{IN}$=$V_{CC}$ | – | – | 1 | uA |
| Input Leakage Current (pin low) | $I_{ILL}$ | All Input | $V_{IN}$=0V | – | – | –1 | uA |
| Power Supply Current (No Load) | $I_{CC}$ | FAST mode FIRC12 MHz, | $V_{CC}$=4.5 to 5.5V | – | 3 | | mA |
| | | FAST mode FIRC4 MHz | $V_{CC}$=4.5 to 5.5V | – | 1 | | |
| | | SLOW mode SIRC2KHz MODE3V=1 | $V_{CC}$=3.0V | | 6 | | uA |
| | | IDLE mode SIRC2KHz MODE3V=1 T2PSC=0 | $V_{CC}$=3.0V | | 3 | | uA |
| | | STOP mode MODE3V=0 PWRSAV=1 | $V_{CC}$=5.0V LVR disable in STOP | – | 0.1 | 1 | uA |
| | | | $V_{CC}$=5.0V, LVR enable | – | 4 | 8 | |
| | | STOP mode MODE3V=0 PWRSAV=0 | $V_{CC}$=5.0V LVR=2.3/2.9V | – | 300 | – | uA |
| | | STOP mode MODE3V=0 PWRSAV=1 | $V_{CC}$=5.0V LVR=2.3/2.9V | – | 4 | 8 | uA |
| | | STOP mode MODE3V=1 | $V_{CC}$=3.0V, LVR disable in STOP | – | 0.1 | 1 | uA |
| System Operating Voltage | $V_{SYS}$ | MODE3V=0 | Fsys=1MHz | 2.0 | – | 5.5 | V |
| | | | Fsys=4MHz | 2.0 | – | 5.5 | |
| | | | Fsys=6MHz | 2.2 | – | 5.5 | |
| | | | Fsys=12MHz | 2.6 | – | 5.5 | |
| | | MODE3V=1 LVRth force at 2.0V | Fsys=1MHz | 2.0 | – | 3.6 | |
| | | | Fsys=4MHz | 2.0 | – | 3.6 | |
| | | | Fsys=6MHz | 2.2 | – | 3.6 | |
| Pull-up Resistor | $R_{UP}$ | VIN=0 V Ports A/B/D | $V_{CC}$=5.0V | | 110 | | KΩ |
| | | | $V_{CC}$=3.0V | | 200 | | |
| | | VIN=0 V PA7 | $V_{CC}$=5.0V | | 60 | | KΩ |
| | | | $V_{CC}$=3.0V | | 60 | | |
| Bandgap Reference Voltage | $V_{BG}$ | | $V_{CC}$=5.0V | -2% | 1.25 | +2% | V |

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

| Parameter | Condition | Min | Typ | Max | Unit |
|---|---|---|---|---|---|
| FIRC Frequency (**) | 0°C ~ 85°C, $V_{CC}$=5.0 V | -2.5% | 12 | +2.5% | MHz |
| | 25°C, $V_{CC}$=3.0 ~ 5.0 V | -3% | 12 | +3% | |
| | 25°C, $V_{CC}$=2.5 ~ 5.0 V | -5% | 12 | +5% | |

(**) Fast-clock frequency can be configured to 1.5MHz, 4MHz, 6MHz, and 12MHz.

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

| Parameter | Conditions | Min | Typ | Max | Unit |
|---|---|---|---|---|---|
| RESET Input Low width | Input $V_{CC}$=5 V ±10 % | 3 | – | – | μs |
| WDT time | $V_{CC}$=5 V, WDTPSC=11 | -20% | 1680 | +20% | ms |
| WKT time | $V_{CC}$=3 V, WKTPSC=11 | -20% | 120 | +20% | ms |
| | $V_{CC}$=5 V, WKTPSC=11 | | 113 | | |
| CPU start up time | $V_{CC}$=5 V | – | 14 | – | ms |

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

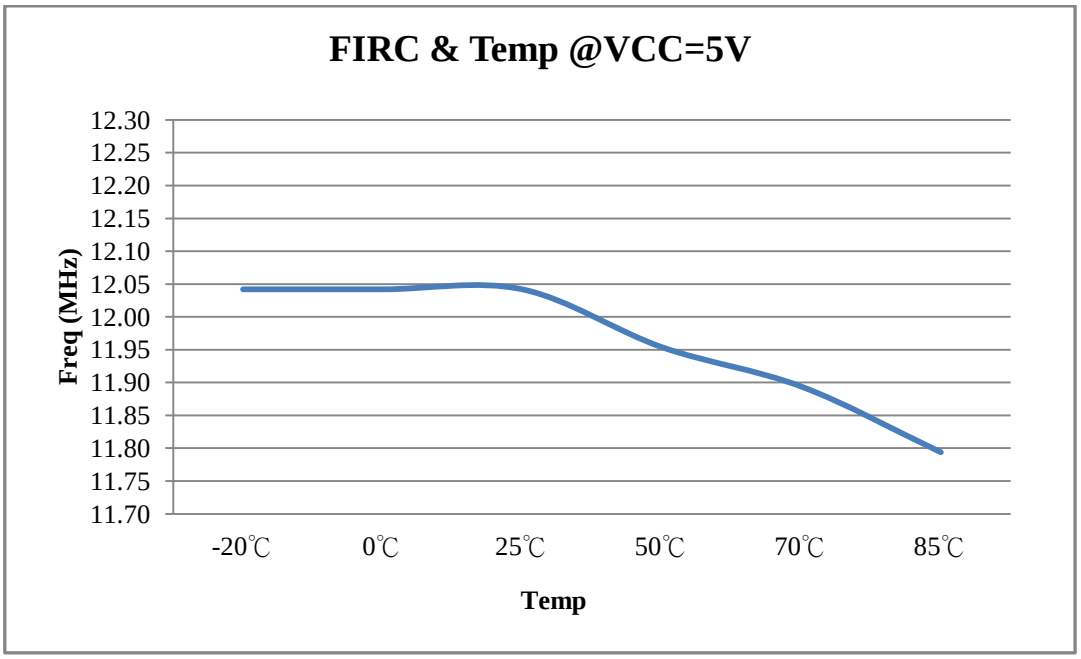
| Parameter | Symbol | Min | Typ | Max | Unit |
|---|---|---|---|---|---|
| LVR Reference Voltage | $LVR_{th}$ | -10% | 2.0 | +10% | V |
| | | -3% | 2.3 | +3% | |
| | | -3%- | 2.9 | +3% | |
| LVR Hysteresis Voltage | $V_{HYST}$ | – | ±0.1 | – | V |
| Low Voltage Detection time | $t_{LVR}$ | 100 | – | – | μs |

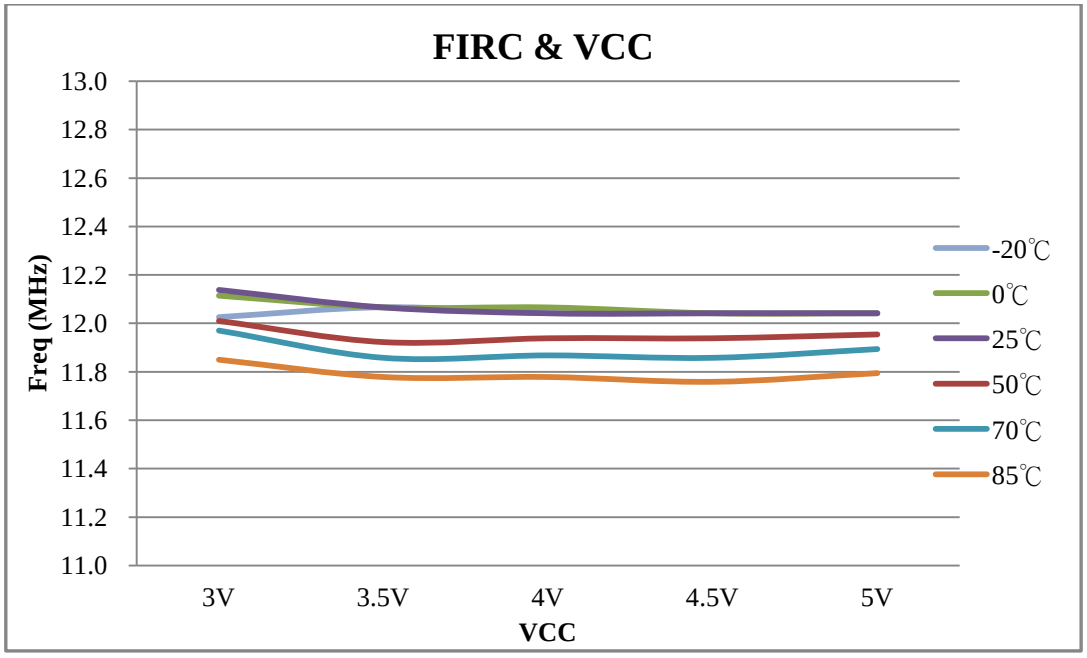
## 6. ADC Electrical Characteristics (TA=25°C, VCC=2.2V to 5.5V, VSS=0V)

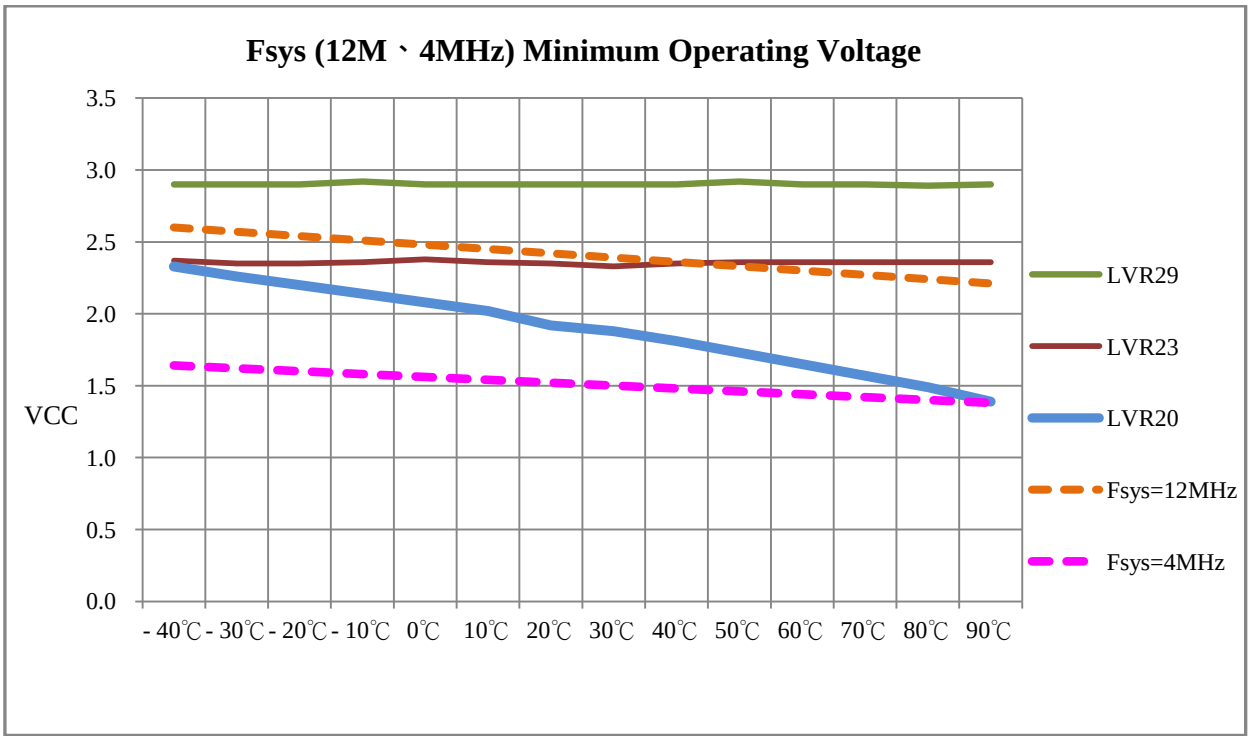
| Parameter | Conditions | Min | Typ | Max | Units |
|---|---|---|---|---|---|
| Total Accuracy | $V_{CC}$ =5.12V, $V_{SS}$ =0V | – | ±2.5 | ±4 | LSB |
| Integral Non-Linearity | | – | ±3.2 | ±5 | |
| Max Input Clock (ADCKS) | All ADC input, except VBG channel Source impedance (Rs<20K ohm) | – | – | 750 | KHz |
| | All ADC input, except VBG channel Source impedance (Rs<40K ohm) | – | – | 500 | |
| | ADCHS=VBG | – | 500 | 750* | |
| Conversion Time | ADCKS =500KHz | – | 100 | – | μs |
| Input Voltage | – | $V_{SS}$ | – | $V_{CC}$ | V |

Note: ADCKS=750KHz, must refer to recommended ADC example code in Section 3.7.

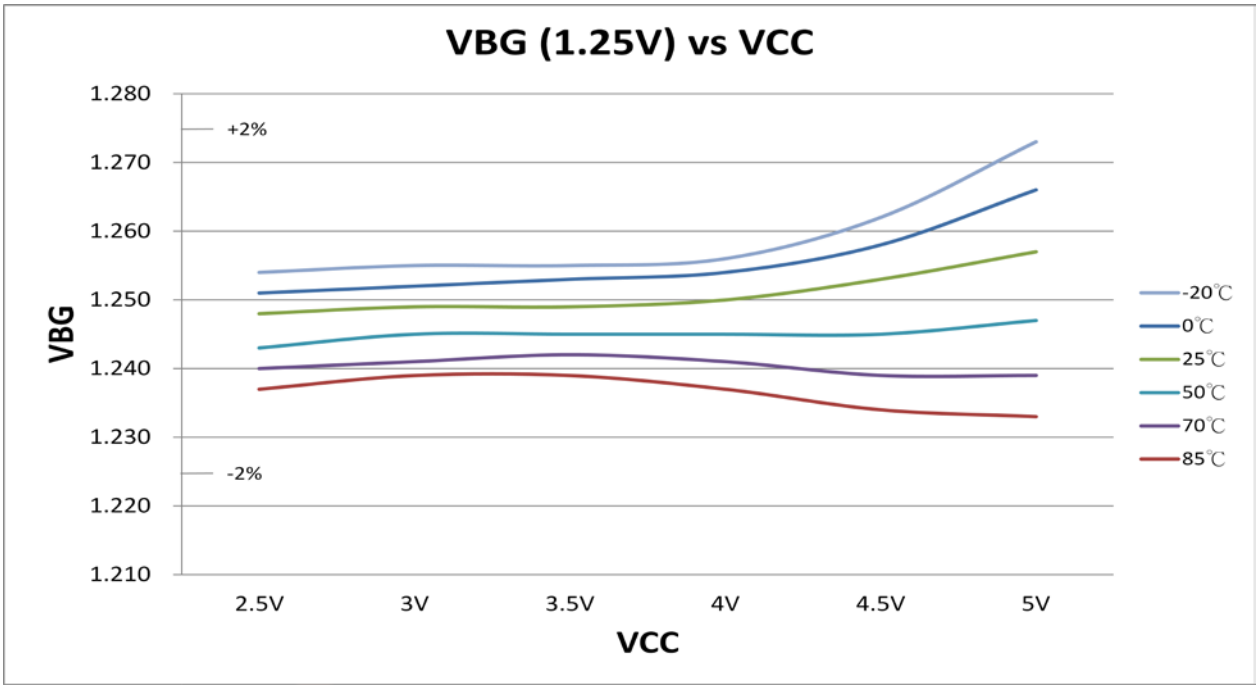
## 7. Characteristic Graphs

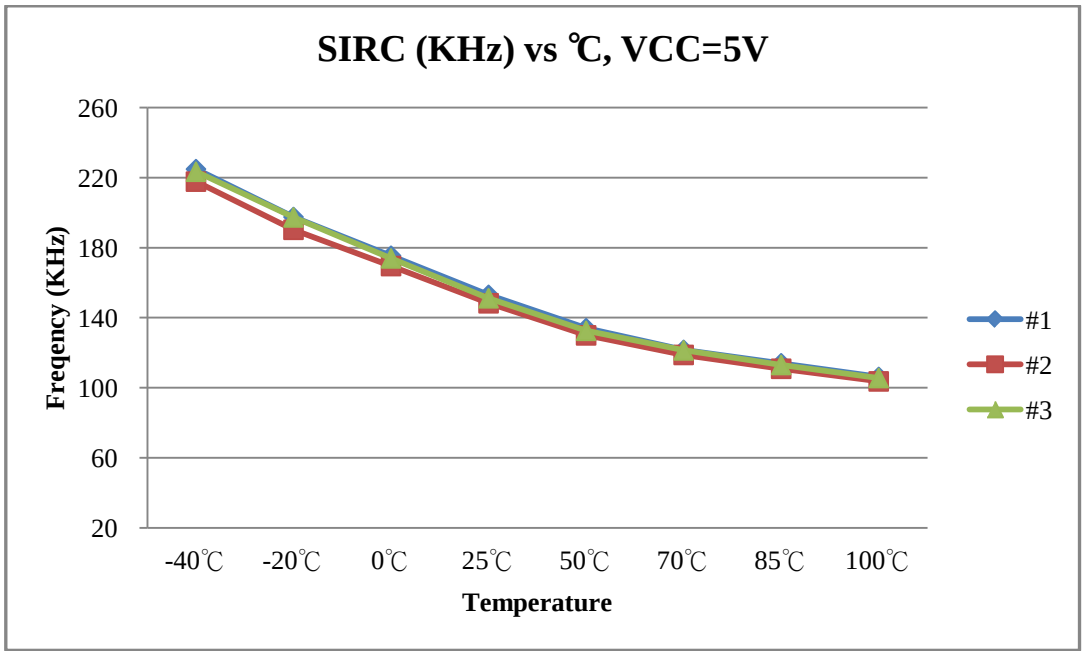


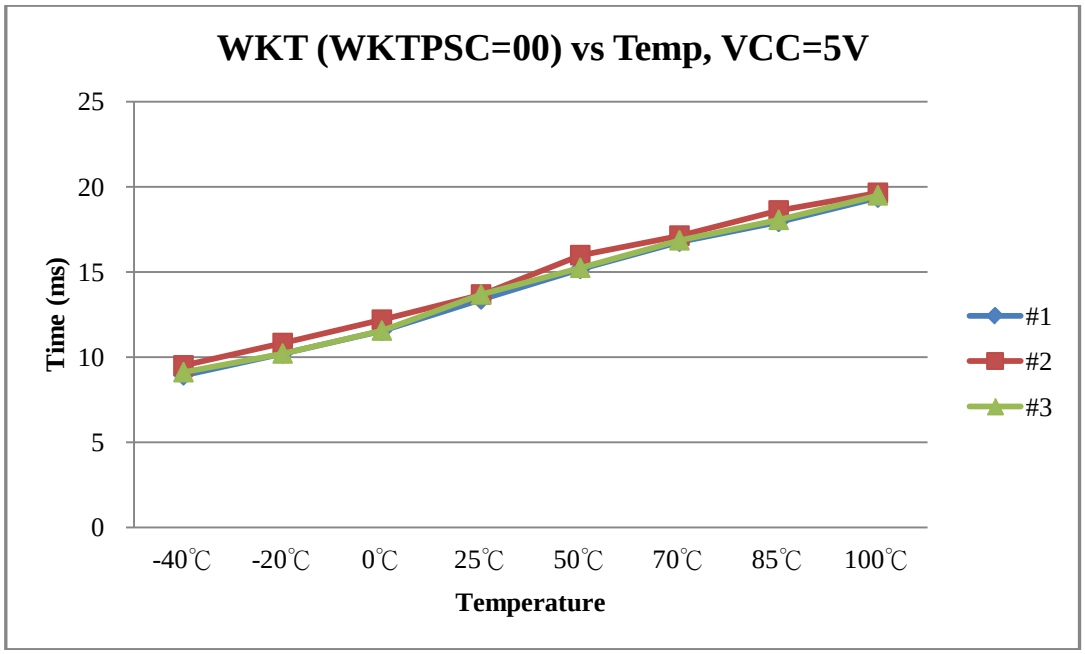






Note: Due to the variation of manufacturing process, this LVR20 will slightly vary between different chips.




SIRC (KHz) vs ℃, VCC=5V
Freqency (KHz)
260
220
180
140
100
60
20
-40℃
-20℃
0℃
25℃
50℃
70℃
85℃
100℃
Temperature
#1
#2
#3


WKT (WKTPSC=00) vs Temp, VCC=5V
Time (ms)
25
20
15
10
5
0
-40℃
-20℃
0℃
25℃
50℃
70℃
85℃
100℃
Temperature
#1
#2
#3

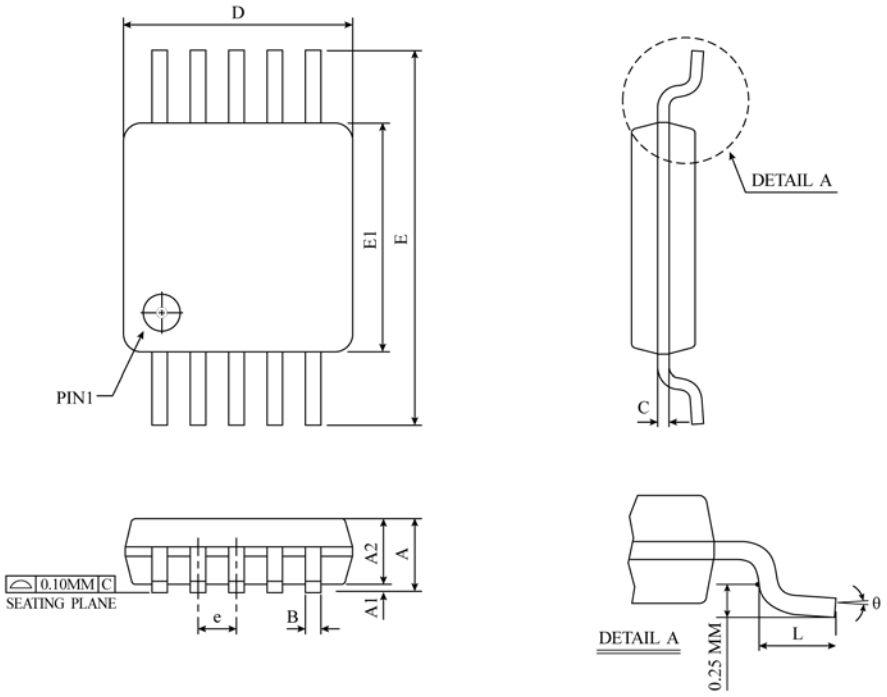
# PACKAGING INFORMATION

The ordering information:

| Ordering number | Package |
|---|---|
| TM57MA28-MTP | Wafer/Dice blank chip |
| TM57MA28-COD | Wafer/Dice with code |
| TM57MA28-MTP-53 | MSOP 10-pin (118mil) |
| TM57MA28GZE0-MTP-16 | SOP 16-pin (150 mil) |
| TM57MA28GZE0-MTP-03 | DIP 16-pin (300mil) |
| TM57MA28-MTP-26 | SSOP 16-pin (150 mil) |
| TM57MA28-MTP-05 | DIP 20-pin (300 mil) |
| TM57MA28-MTP-21 | SOP 20-pin (300 mil) |
| TM57MA28-MTP-B6 | QFN 20-pin (3x3x0.75-0.4mm) |

| Ordering number | Package |
|---|---|
| TM57MA28B-MTP | Wafer/Dice blank chip |
| TM57MA28B-COD | Wafer/Dice with code |
| TM57MA28B-MTP-53 | MSOP 10-pin (118mil) |
| TM57MA28BGZE0-MTP-16 | SOP 16-pin (150 mil) |
| TM57MA28BGZE0-MTP-03 | DIP 16-pin (300mil) |
| TM57MA28B-MTP-26 | SSOP 16-pin (150 mil) |
| TM57MA28B-MTP-05 | DIP 20-pin (300 mil) |
| TM57MA28B-MTP-21 | SOP 20-pin (300 mil) |
| TM57MA28B-MTP-B6 | QFN 20-pin (3x3x0.75-0.4mm) |

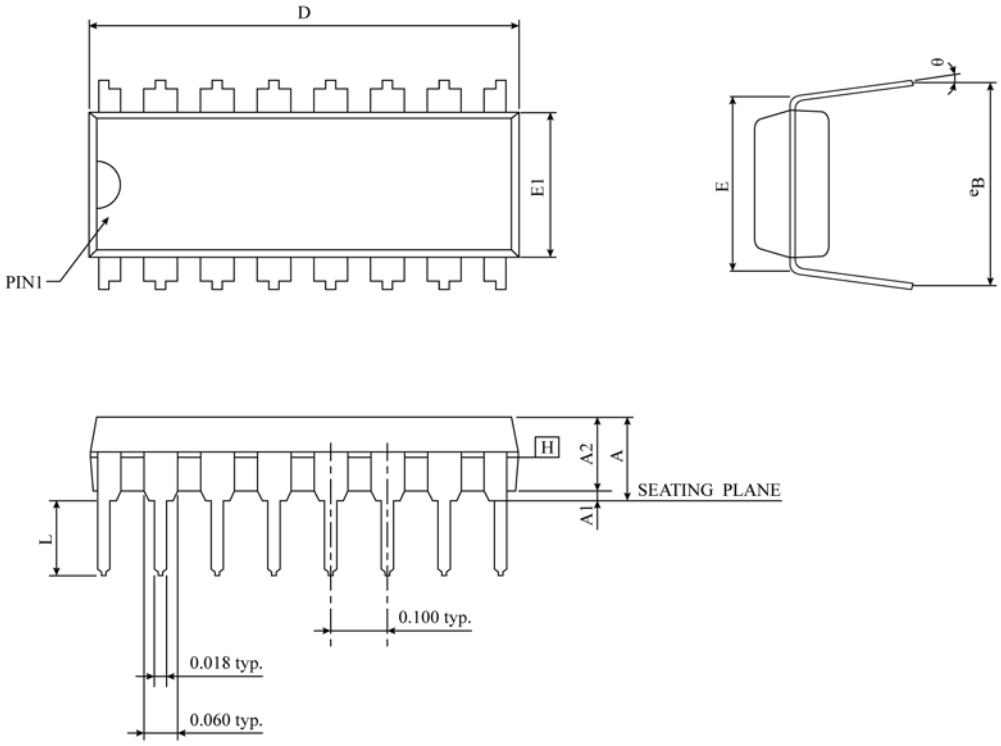
## 10-MSOP (118mil) Package Dimension




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

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

## 16-DIP (300mil) Package Dimension

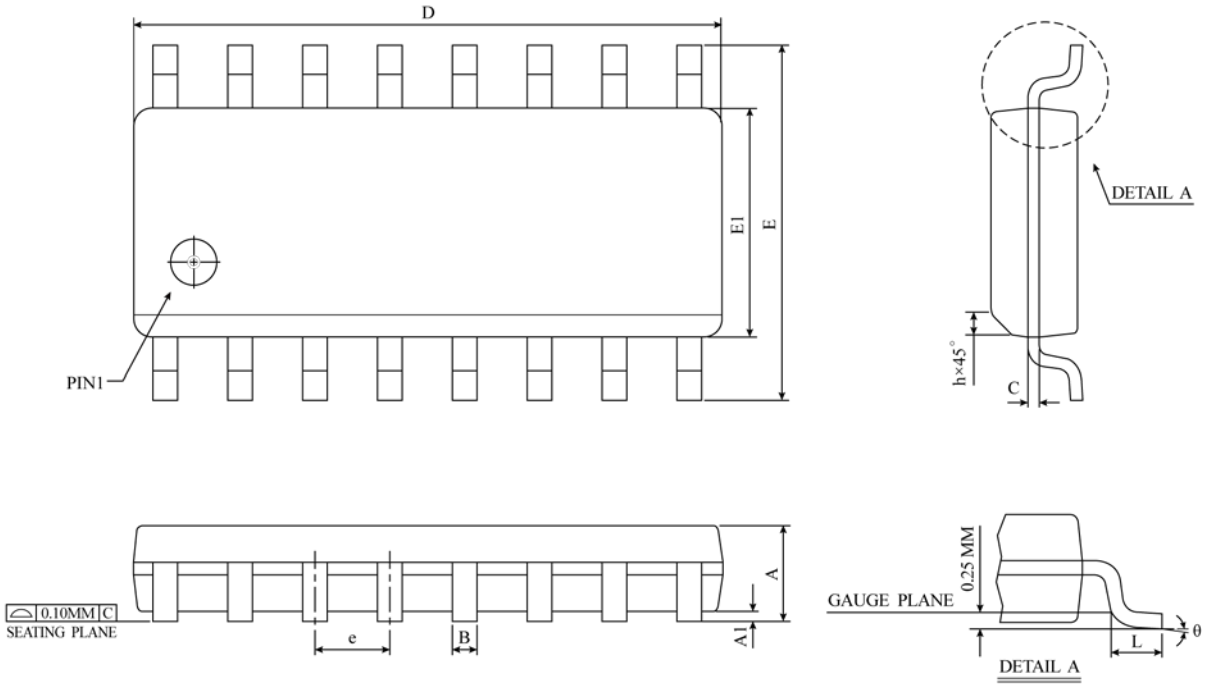



| SYMBOL | DIMENSION IN MM | | | DIMENSION IN INCH | | |
|---|---|---|---|---|---|---|
| | MIN | NOM | MAX | MIN | NOM | MAX |
| A | - | - | 4.369 | - | - | 0.172 |
| A1 | 0.381 | 0.673 | 0.965 | 0.015 | 0.027 | 0.038 |
| A2 | 3.175 | 3.302 | 3.429 | 0.125 | 0.130 | 0.135 |
| D | 18.669 | 19.177 | 19.685 | 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.

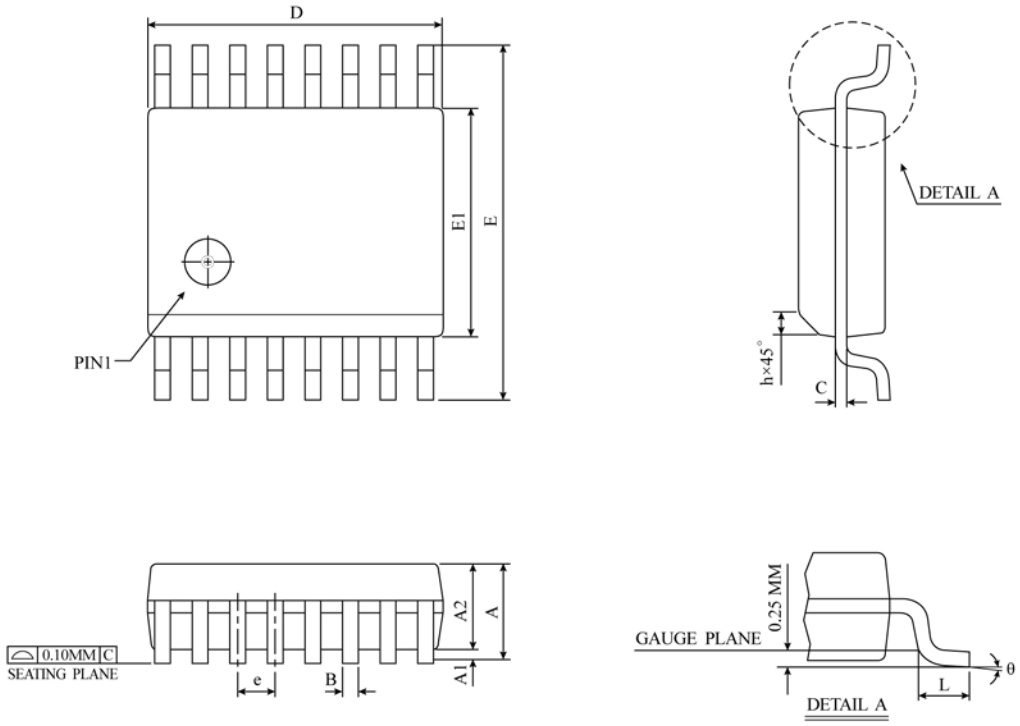
## 16-SOP (150 mil) Package Dimension




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

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

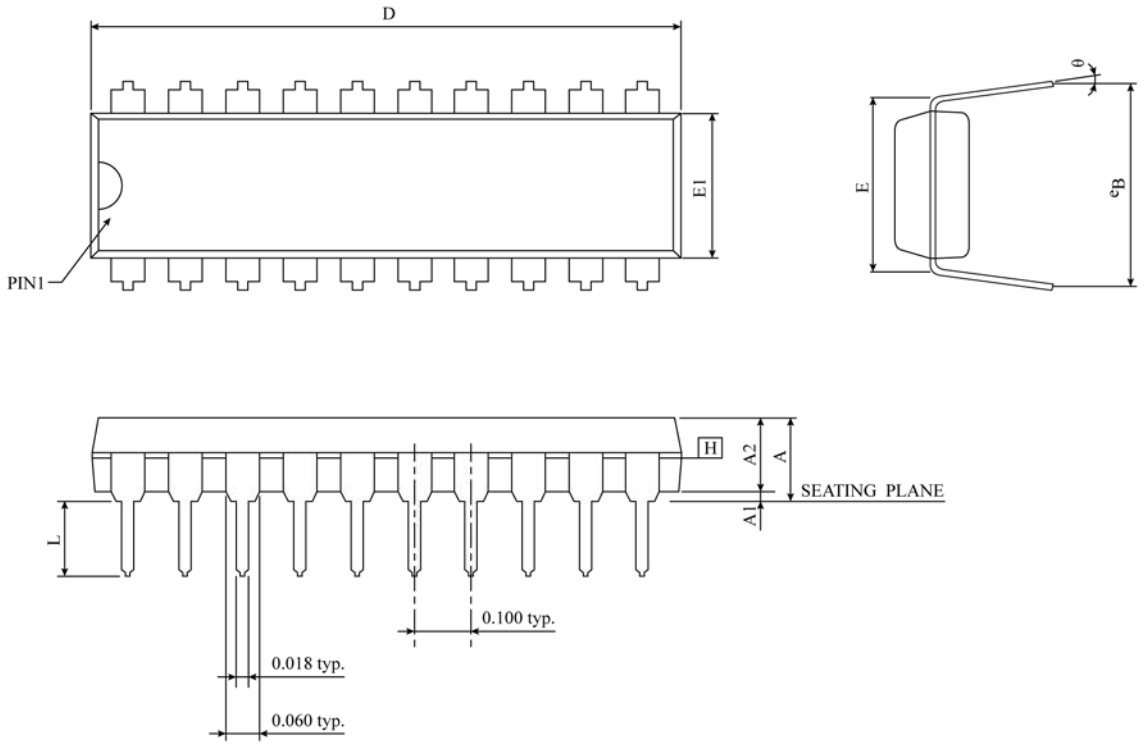
## 16-SSOP (150mil) Package Dimension




| SYMBOL | DIMENSION IN MM | | | DIMENSION IN INCH | | |
|---|---|---|---|---|---|---|
| | MIN | NOM | MAX | MIN | NOM | MAX |
| A | 1.35 | 1.55 | 1.75 | 0.053 | 0.061 | 0.069 |
| A1 | 0.10 | 0.18 | 0.25 | 0.004 | 0.007 | 0.010 |
| A2 | - | - | 1.50 | - | - | 0.059 |
| B | 0.20 | 0.25 | 0.30 | 0.008 | 0.010 | 0.012 |
| C | 0.18 | 0.22 | 0.25 | 0.007 | 0.009 | 0.010 |
| D | 4.80 | 4.90 | 5.00 | 0.189 | 0.193 | 0.197 |
| E | 5.79 | 6.00 | 6.20 | 0.228 | 0.236 | 0.244 |
| E1 | 3.81 | 3.90 | 3.99 | 0.150 | 0.154 | 0.157 |
| e | 0.635 BSC | | | 0.025 BSC | | |
| L | 0.41 | 0.84 | 1.27 | 0.016 | 0.033 | 0.050 |
| θ | 0° | 4° | 8° | 0° | 4° | 8° |
| JEDEC | M0-137 (AB) | | | | | |

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

## 20-DIP (300mil) Package Dimension

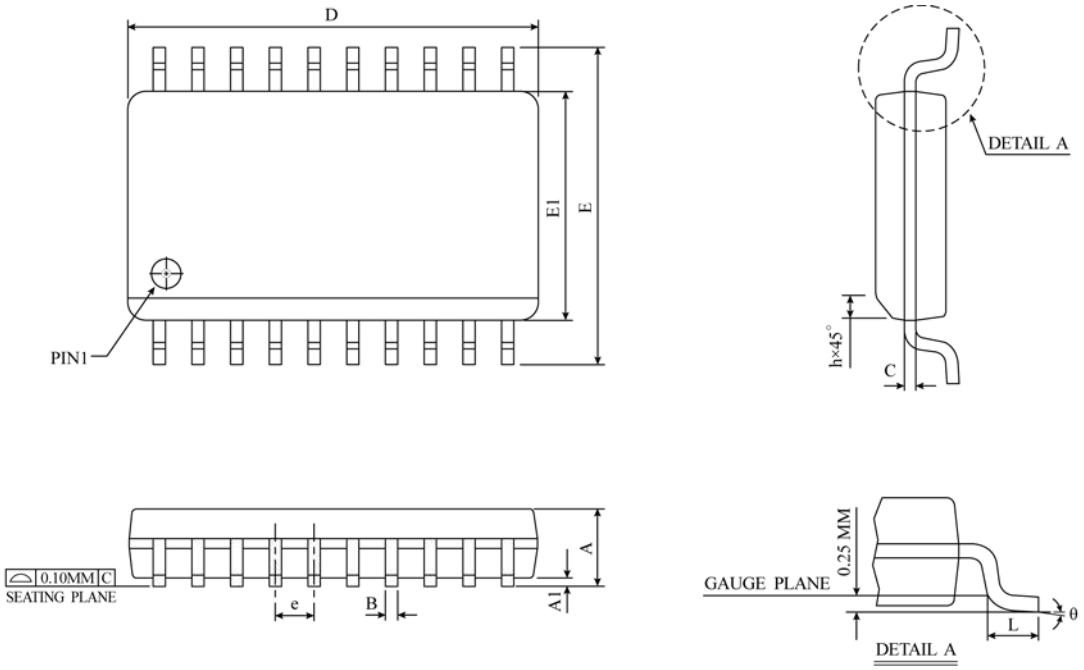



| SYMBOL | DIMENSION IN MM | | | DIMENSION IN INCH | | |
|---|---|---|---|---|---|---|
| | MIN | NOM | MAX | MIN | NOM | MAX |
| A | - | - | 4.445 | - | - | 0.175 |
| A1 | 0.381 | - | - | 0.015 | - | - |
| A2 | 3.175 | 3.302 | 3.429 | 0.125 | 0.130 | 0.135 |
| D | 25.705 | 26.061 | 26.416 | 1.012 | 1.026 | 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.

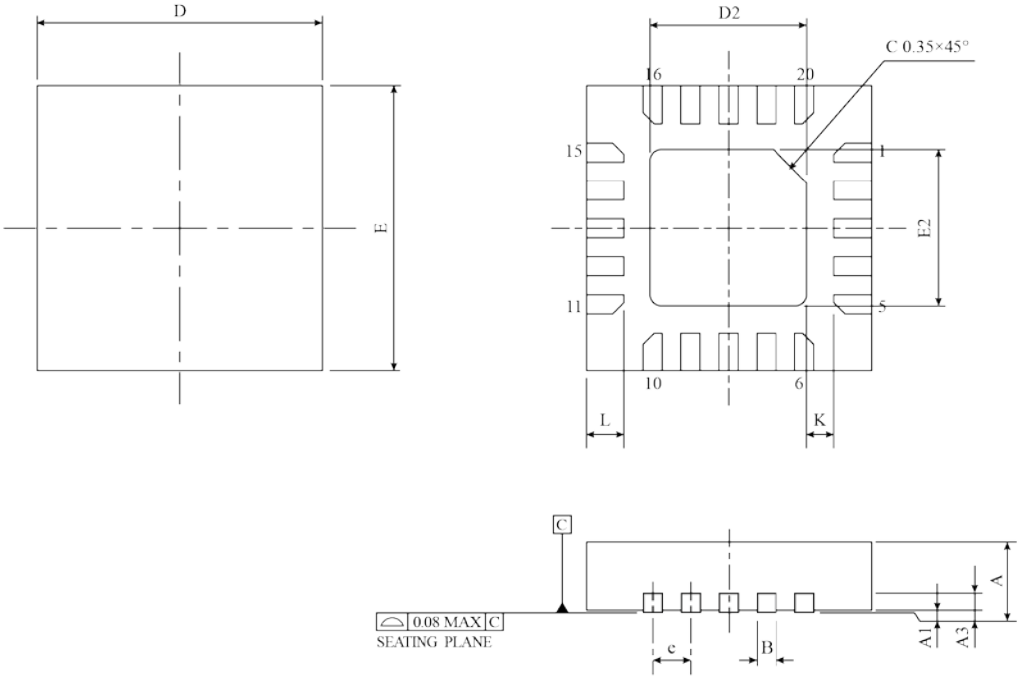
## 20-SOP (300mil) Package Dimension




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

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

## 20-QFN (3x3x0.75-0.4mm) Package Dimension




| SYMBOL | DIMENSION IN MM | | | DIMENSION IN INCH | | |
|---|---|---|---|---|---|---|
| | MIN | NOM | MAX | MIN | NOM | MAX |
| A | 0.70 | 0.75 | 0.80 | 0.028 | 0.030 | 0.031 |
| A1 | 0.00 | 0.02 | 0.05 | 0.000 | 0.001 | 0.002 |
| A3 | | 0.203 REF. | | | 0.008 REF. | |
| B | 0.15 | 0.20 | 0.25 | 0.006 | 0.008 | 0.010 |
| D | | 3.00 BSC | | | 0.118 BSC | |
| E | | 3.00 BSC | | | 0.118 BSC | |
| e | | 0.40 BSC | | | 0.016 BSC | |
| K | 0.20 | - | - | 0.008 | - | - |
| E2 | 1.60 | 1.65 | 1.70 | 0.063 | 0.065 | 0.067 |
| D2 | 1.60 | 1.65 | 1.70 | 0.063 | 0.065 | 0.067 |
| L | 0.30 | 0.40 | 0.50 | 0.012 | 0.016 | 0.020 |
| JEDEC | | | | | | |

⚠ * NOTES : 1. ALL DIMENSION ARE IN MILLIMETRS.
2. DIMENSION B APPLIES TO METALLLIZED TERMINAL AND IS MEASURED BETWEEN 0.15mm AND 0.30mm FROM THE TERMINAL TIP. IF THE TERMINAL HAS THE OPTIONAL RADIUS ON THE OTHER END OF THE TERMINAL, THE DIMENSION B SHOULD NOT BE MEASURED IN THAT RADIUS AREA.
3. BILATERAL COPLANARITY ZONE APPLIES TO THE EXPOSED HEAT SINK SLUG AS WELL AS THE TERMINALS.