十速

# TM57PE10

## *DATA SHEET*

## *Rev V1.8*

# AMENDMENT HISTORY

| Version | Date | Description |
|---|---|---|
| V1.0 | Oct, 2010 | New release |
| V1.1 | Jan, 2011 | 1. Add more description about /Borrow and /Digit Borrow in ALU and Working (W) Register section.<br>2. Add Internal RC mode description and figure in System Clock Oscillator section. |
| V1.2 | Aug, 2011 | Modify the operating voltage. |
| V1.3 | Dec, 2011 | 1. Add Ordering Information table in the Packaging Information section.<br>2. Add 16-pin DIP / SOP in Features section.<br>3. Add 16-pin Package Dimension. |
| V1.4 | Jan, 2012 | 1. Add the Electrical Characteristics specs in the Features section.<br>2. Add description in Reset section.<br>3. Merge the information about LVR Circuit Characteristics into DC Characteristics table. |
| V1.5 | Jul, 2012 | Modify document format. |
| V1.6 | May, 2013 | 1. Modify Block Diagram.<br>2. Modify Packaging Information. |
| V1.7 | Jul, 2013 | 1. Add supported EV board on ICE.<br>2. Add pin summary. |
| V1.8 | Aug, 2013 | 1. Modify pin assignment.<br>2. Modify Interrupt description.<br>3. Modify Ordering Information, omit “-X” |

# CONTENTS

# FEATURES

1. ROM: 1K x 14 bits OTP or 512 x 14 bits TTP™ (Two Time Programmable ROM)
2. RAM: 48 x 8 bits
3. STACK: 5 Levels
4. I/O ports: Two Bit programmable I/O ports (Max. 16 pins)
5. Two Independent Timers
   - 8-bit timer0 with divided by 1~256 pre-scale option, counter function, Stop counting
   - 15-bit timer2 with 4 interrupt interval option

     Timer2 is used to idle mode wake-up timer or one simple 15-bit time base
6. 8-bit PWM0 with prescale/period-adjustment/buffer-reload/clear and hold function
7. 8-bit PWM1 is a simple fixed frequency and duty cycle variable PWM generator
8. One analog voltage comparator
9. Min. Operating Voltage (power on) and Speed: VDD=1.5V, @4 MHz
10. PA1~PA6, PB1~PB6 individual pin low level wake up
11. Oscillation Sources
    - Fast Clock:
      - FXT (Fast Crystal): 1 MHz~24 MHz
      - FIRC (Fast Internal RC): 4/8 MHz
      - XRC (External R, External C): 10 KHz~3 MHz
    - Slow Clock:
      - SXT (Slow Crystal): 32768 Hz
      - XRC (External R, External C): 10 KHz~3 MHz
      - SIRC (Slow Internal RC): 151 KHz/37 KHz/9.4 KHz/2.4 KHz, @5V; 115 KHz/29 KHz/7.2 KHz/1.9 KHz, @3V
12. Power Saving Operation Mode
    - Fast Mode: Slow Clock can be disabled or enabled
    - Slow Mode: Fast Clock stops, CPU is running
    - Idle Mode: Slow Clock is running, CPU stops, Timer2 is running
    - Stop Mode: All Clocks stop, Wake-up Timer is disabled or enabled

**13.** Dual System Clock
- FIRC + SIRC
- FIRC + SXT
- FIRC + XRC
- FXT + SIRC
- XRC + SIRC

**14.** Reset
- Power On Reset
- Watchdog Reset
- Low Voltage Reset
- External pin Reset

**15.** 2-Level Low Voltage Reset: 1.5V/2.3V (Can be disabled)

**16.** Operation Voltage: Low Voltage Reset Level to 5.5V
- fosc = 4 MHz, 1.7V ~ 5.5V
- fosc = 8 MHz, 1.9V ~ 5.5V
- fosc = 12 MHz, 2.1V ~ 5.5V
- fosc = 16 MHz, 2.3V ~ 5.5V

**17.** Interrupt
- Three External Interrupt pins:
  - Two pins are falling edge triggered
  - One pin is rising or falling edge triggered
- Timer0, Timer2, Wake-up Timer Interrupt
- PWM0, CMP interrupt

**18.** Watchdog Timer
- Clocked by built-in RC oscillator with 4 adjustable Reset/Interrupt Time (108 ms/56 ms/28 ms/14 ms, @5V; 138 ms/72 ms/36 ms/18 ms, @3V)
- Watchdog timer can be disabled/enabled in STOP mode

**19.** I/O Ports
- CMOS Output
- Pseudo-Open-Drain or Open-Drain Output
- Schmitt Trigger Input with/without pull-up resistor

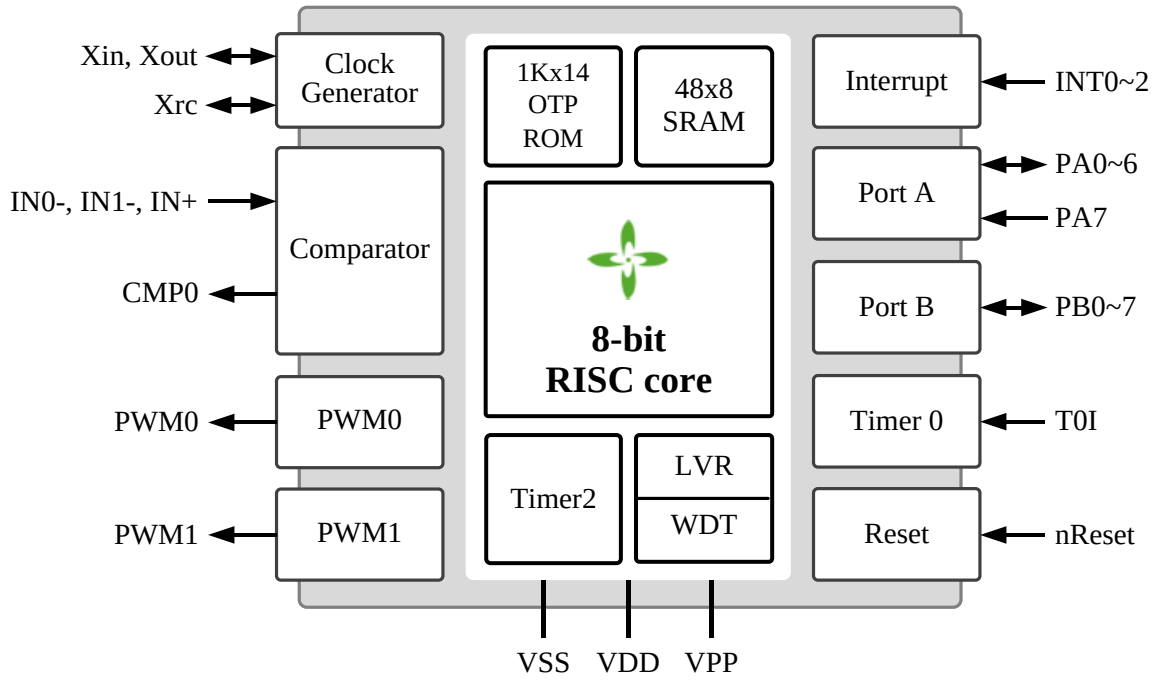
**20.** Instruction set: 36 Instructions

**21.** Package Types: 14-DIP/SOP, 16-DIP/SOP, 18-DIP/SOP

**22. Supported EV Board on ICE**

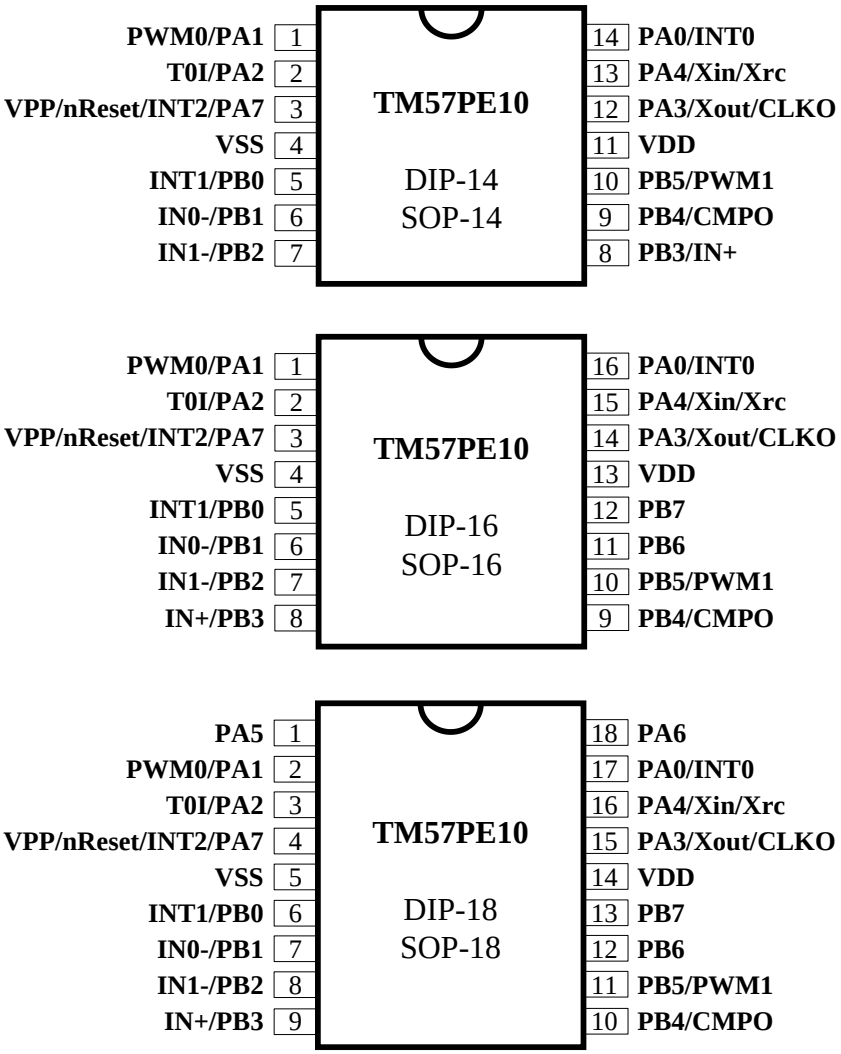
EV Board: EV2786

# BLOCK DIAGRAM




**TM57PE10 Block Diagram**

# PIN ASSIGNMENT



# PIN DESCRIPTION

| Name | In/Out | Pin Description |
|---|---|---|
| PA0–PA2 | I/O | Bit-programmable I/O port for Schmitt-trigger input, CMOS push-pull output or "pseudo-open-drain" output. Pull-up resistors are assignable by software. |
| PA3–PA6 | I/O | Bit-programmable I/O port for Schmitt-trigger input, CMOS push-pull output or open-drain output. Pull-up resistors are assignable by software. |
| PA7 | I | Schmitt-trigger input |
| PB0–PB7 | I/O | Bit-programmable I/O port for Schmitt-trigger input, CMOS push-pull output or open-drain output. Pull-up resistors are assignable by software. |
| nRESET | I | External active low reset |
| Xin, Xout | – | Crystal/Resonator oscillator connection for system clock |
| Xrc | – | External RC oscillator connection for system clock |
| CLKO | O | CPU Instruction clock output for external/internal RC mode |
| VDD, VSS | P | Power Voltage input pin and ground |
| VPP | I | PROM programming high voltage input |
| INT0–INT2 | I | External interrupt input |
| PWM0–PWM1 | O | PWM output |
| IN0-, IN1-, IN+ | I | Comparator voltage input |
| CMPO | O | Comparator output |
| T0I | I | Clock input to Timer0 |

# PIN SUMMARY

| Pin Number | | | Pin Name | Type | GPIO | | | | | Function After Reset | Alternate Function | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | Input | | Output | | | | | | | |
| 18-SOP/DIP | 16-SOP/DIP | 14-SOP/DIP | | | Weak Pull-up | Ext. Interrupt | O.D | P.O.D | P.P | | PWM | Touch Key | ADC | MISC |
| 1 | - | - | **PA5** | I/O | O | | O | | O | PA5 | | | | |
| 2 | 1 | 1 | **PA1/PWM0** | I/O | O | | | O | O | PA1 | O | | | |
| 3 | 2 | 2 | **PA2/T0I** | I/O | O | | | O | O | PA2 | | | | T0I |
| 4 | 3 | 3 | **PA7/INT2/nReset/VPP** | I | O | O | | | | PA7 | | | | nReset |
| 5 | 4 | 4 | **VSS** | P | | | | | | | | | | |
| 6 | 5 | 5 | **PB0/INT1** | I/O | O | O | O | | O | PB0 | | | | |
| 7 | 6 | 6 | **PB1/IN0-** | I/O | O | | O | | O | PB1 | | | | IN0- |
| 8 | 7 | 7 | **PB2/IN1-** | I/O | O | | O | | O | PB2 | | | | IN1- |
| 9 | 8 | 8 | **PB3/IN+** | I/O | O | | O | | O | PB3 | | | | IN1+ |
| 10 | 9 | 9 | **PB4/CMPO** | I/O | O | | O | | O | PB4 | | | | CMP0 |
| 11 | 10 | 10 | **PB5/PWM1** | I/O | O | | O | | O | PB5 | O | | | |
| 12 | 11 | - | **PB6** | I/O | O | | O | | O | PB6 | | | | |
| 13 | 12 | - | **PB7** | I/O | O | | O | | O | PB7 | | | | |
| 14 | 13 | 11 | **VDD** | P | | | | | | | | | | |
| 15 | 14 | 12 | **PA3/Xout/CLKO** | I/O | O | | O | | O | PA3 | | | | CLKO |
| 16 | 15 | 13 | **PA4/Xin/Xrc** | I/O | O | | O | | O | PA4 | | | | |
| 17 | 16 | 14 | **PA0/INT0** | I/O | O | O | | O | O | PA0 | | | | |
| 18 | - | - | **PA6** | I/O | O | | O | | O | PA6 | | | | |

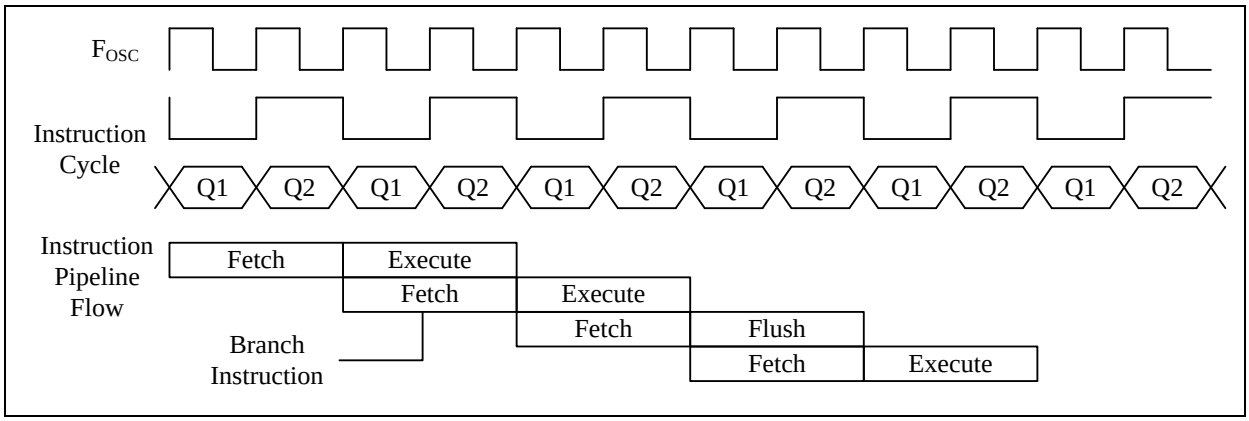
Symbol：P.P. = Push-Pull Output
P.O.D. = Pseudo Open Drain
O.D. = Open Drain

# FUNCTIONAL DESCRIPTION

## 1. CPU Core

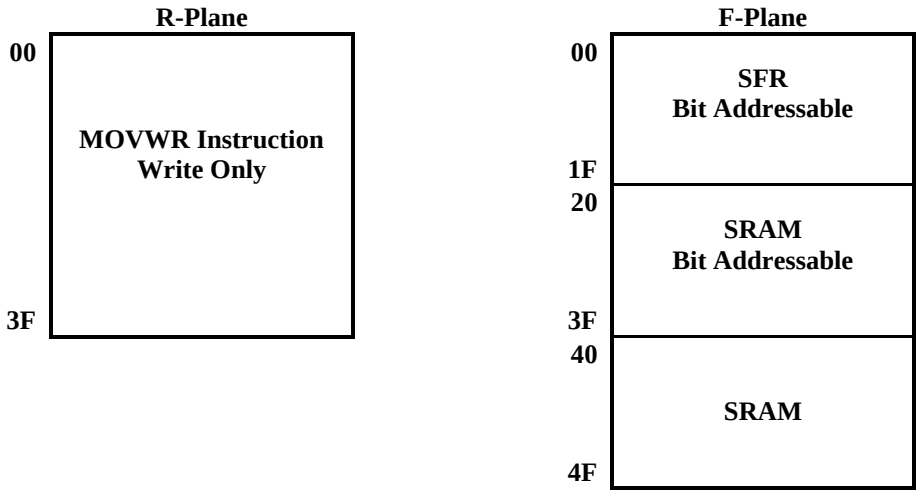
### 1.1 Clock Scheme and Instruction Cycle

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



## 1.2 Addressing Mode

There are two Data Memory Planes in CPU, R-Plane and F-Plane. The registers in R-Plane are write-only. The "MOVWR" instruction copy the W-register's content to R-Plane registers by direct addressing mode. The lower locations of F-Plane are reserved for the SFR. Above the SFR is General Purpose Data Memory, implemented as static RAM. F-Plane can be addressed directly or indirectly. Indirect Addressing is made by INDF register. The INDF register is not a physical register. Addressing INDF actually addresses the register whose address is contained in the FSR 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.




## 1.3 Programming Counter (PC) and Stack

The Programming Counter is 10-bit wide capable of addressing a 1K x 14 OTP ROM. As a program instruction is executed, the PC will contain the address of the next program instruction to be executed. The PC value is normally increased by one except the followings. The Reset Vector (000h) and the Interrupt Vector (001h) are provided for PC initialization and Interrupt. For CALL/GOTO instructions, PC loads 10 bits address from instruction word. For RET/RETI/RETLW instructions, PC retrieves its content from the top level STACK. For the other instructions updating PC [7:0], the PC [9:8] keeps unchanged. The STACK is 10-bit wide and 5-level in depth. The CALL instruction and hardware interrupt will push STACK level in order. While the RET/RETI/RETLW instruction pops the STACK level in order.

### 1.4 ALU and Working (W) Register

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

Note: /Borrow represents inverted of Borrow register.

/Digit Borrow represents inverted of Digit Borrow register.

### 1.5 STATUS Register

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.

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

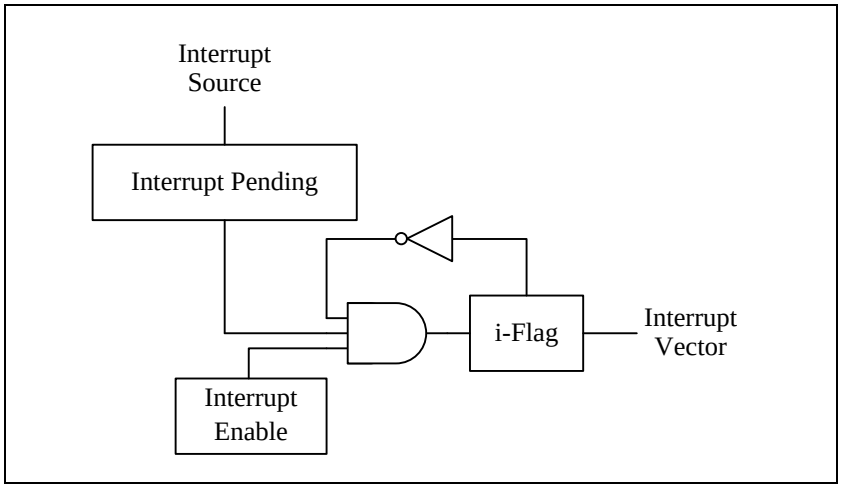
<table>
<tr><td>Bit</td><td colspan="2">Description</td></tr>
<tr><td>7</td><td colspan="2">Not Used</td></tr>
<tr><td>6</td><td colspan="2">General Purpose Bit</td></tr>
<tr><td>5</td><td colspan="2">Not Used</td></tr>
<tr><td>4</td><td colspan="2">TO: Time Out<br>0: after Power On Reset, LVR Reset, or CLRWDT/SLEEP instruction<br>1: WDT time out occurs</td></tr>
<tr><td>3</td><td colspan="2">PD: Power Down<br>0: after Power On Reset, LVR Reset, or CLRWDT instruction<br>1: after SLEEP instruction</td></tr>
<tr><td>2</td><td colspan="2">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="2">DC: Decimal Carry Flag or Decimal /Borrow Flag</td></tr>
<tr><td>ADD instruction</td><td>SUB instruction</td></tr>
<tr><td>1: a carry from the low nibble bits of the result occurs<br>0: no carry</td><td>1: no borrow<br>0: a borrow from the low nibble bits of the result occurs</td></tr>
<tr><td rowspan="3">0</td><td colspan="2">C: Carry Flag or /Borrow Flag</td></tr>
<tr><td>ADD instruction</td><td>SUB instruction</td></tr>
<tr><td>1: a carry occurs from the MSB<br>0: no carry</td><td>1: no borrow<br>0: a borrow occurs from the MSB</td></tr>
</table>

### 1.6 Interrupt

The TM57PE10 has 1 level, 1 vector and 8 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 TM57PE10 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 (F-Plane 08h), 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.



# 2. Chip Operation Mode

## 2.1 Reset

The TM57PE10 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 two threshold levels can be selected. The LVR's operation mode is defined by the SYSCFG register.

There are two voltage selections for the LVR threshold level, one is higher level which is suitable for application with VDD is more than 3.3V, while another one is suitable for application with VDD is less than 3.3V. See the following LVR Selection Table; user must also consider the lowest operating voltage of operating frequency.

LVR Selection Table:

| LVR Threshold Level | Consider the operating voltage to choose LVR |
|---|---|
| LVR2.3 | 5.5V > $V_{DD}$ > 3.3V |
| LVR1.5 | $V_{DD}$ is wide voltage range |

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 flag is not affected by these resets.

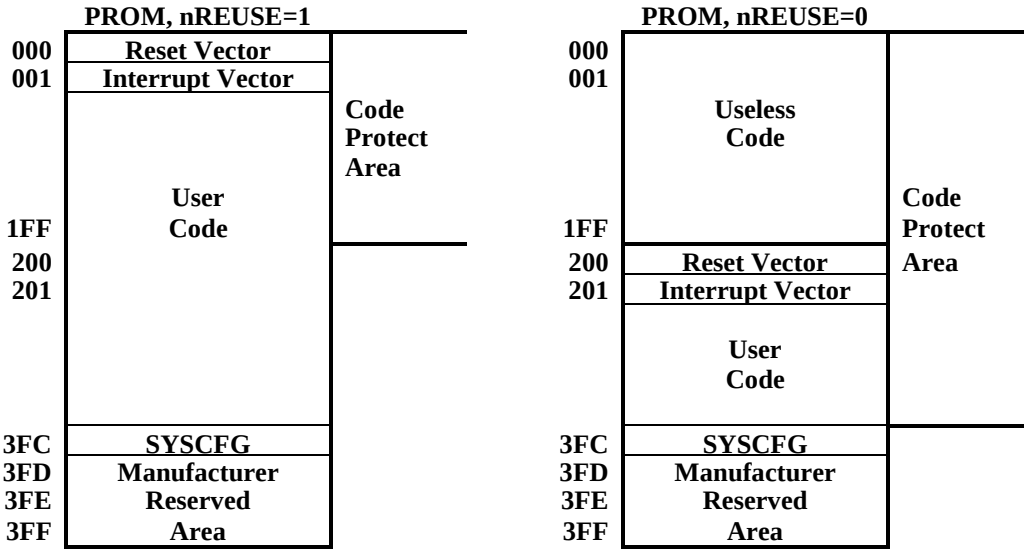
## 2.2 System Configuration Register (SYSCFG)

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 0, the data in PROM will be protected, when user reads PROM.

| **Bit** | **13~0** | |
|---|---|---|
| **Default Value** | **11111111111111** | |
| Bit | Description | |
| 13 | **nPROTECT**: Code protection selection | |
| | 1 | No protect |
| | 0 | Code protection |
| 12 | **nREUSE**: PROM Re-use control | |
| | 1 | Not Re-use |
| | 0 | Re-use |
| 11-10 | **LVR**: LV Reset Mode | |
| | 11 | LVR threshold is 1.5V, always enable |
| | 10 | LVR threshold is 1.5V, disable in sleep mode |
| | 01 | LVR threshold is 2.3V, always enable |
| | 00 | LVR disable |
| 9-8 | **CLKS**: Fast Clock Source Selection | |
| | 11 | Fast Crystal (1 MHz~24 MHz) |
| | 10 | Slow Crystal |
| | 01 | Fast Internal RC (4/8 MHz) |
| | 00 | External RC |
| 7 | **XRESETE**: External pin Reset Enable | |
| | 1 | Enable External pin Reset |
| | 0 | Disable External pin Reset to use as input pin |
| 6 | **WDTE**: WDT Reset Enable | |
| | 1 | Enable WDT Reset |
| | 0 | Disable WDT Reset |
| 5 | **FIRC**: 1:FIRCCLK=8 MHz, 0: FIRCCLK=4 MHz | |
| 4-0 | **FIRCF**: Fast Internal RC Frequency adjustment control | |

## 2.3 PROM Re-use ROM

The PROM of this device is 1K words. For some F/W program, the program size could be less than 512 words. To fully utilize the PROM, the device allows users to reuse the PROM. This feature is named as Two Time Programmable (TTP) ROM. While the first half of PROM is occupied by a useless program code and the second half of the PROM remains blank, users can re-write the PROM with the updated program code into the second half of the PROM. In the Re-use mode, the Reset Vector and Interrupt Vector are re-allocated at the beginning of the PROM's second half by the Assembly Compiler. Users simply choose the "REUSE" option in the ICE tool interface, and then the Compiler will move the object code to proper location. That is, the user's program still has reset vector at address 000h, but the compiled object code has reset vector at 200h. In the SYSCFG, if nPROTECT=0 and nREUSE=1, the Code protection area is first half of PROM. This allows the Writer tool to write then verify the Code during the Re-use Code programming. After the Re-use Code being written into the PROM's second half, user should write "nREUSE" control bit to "0". In the mean while, the Code protection area becomes the whole PROM except the Reserved Area.

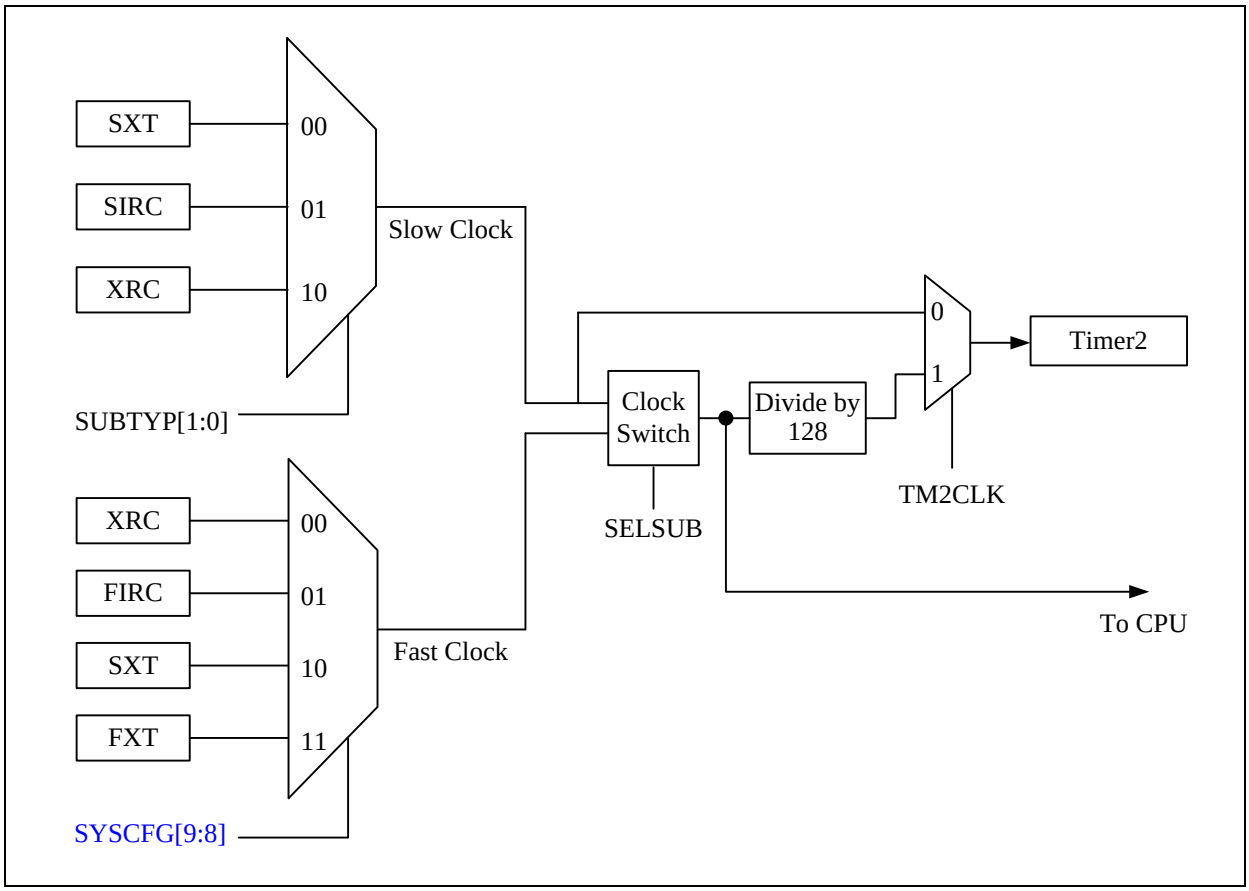



## 2.4 Power-Down Mode

The Power-down mode is activated by SLEEP instruction. During the Power-down mode, the system clock and peripherals stop to minimize power consumption, while the WDT/WKT Timer is working or not depends on F/W setting. The Power down mode can be terminated by Reset, or enabled Interrupts (External pins and WKT interrupt) or PA1-6 and PB1-6 pins low level wake up.

## 2.5 Dual System Clock

TM57PE10 is designed with dual-clock system. There are five kinds of clock source, FXT (Fast Crystal) Clock, SXT (Slow Crystal) Clock, XRC (External RC) Clock, SIRC (Slow Internal RC) Clock 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 Timer2. Refer to the Figure as below.



**Fast Mode:**

After power on or reset, TM57PE10 enters fast mode. In fast mode, TM57PE10 can select FXT, XRC or FIRC as its CPU clock by SYSCFG bit9 and bit8 setting. Besides, firmware can also enable or disable the slow clock for the Timer2 system operating. In this mode, the program is executed using fast clock as CPU clock. The Timer0, PWM0, PWM1 blocks are also driven by fast clock. Timer2 can also be driven by fast clock by setting TM2CLK to "1".

**Slow Mode:**

In slow mode, TM57PE10 can select SXT, XRC or SIRC as its CPU clock by R-Plane control register (SUBTYP). In this mode, the fast clock is stopped and slow clock is enabled for power saving. All peripheral blocks clock sources are slow clock in the slow mode.

**IDLE Mode:**

If slow clock is enabled and TM2CLK=0 before executing the SLEEP instruction, the TM57PE10 enters the "Idle Mode". In this mode, the slow clock will continue running to provide clock to Timer2 block. CPU stop fetching code and all blocks are stop except Timer2 related circuits.

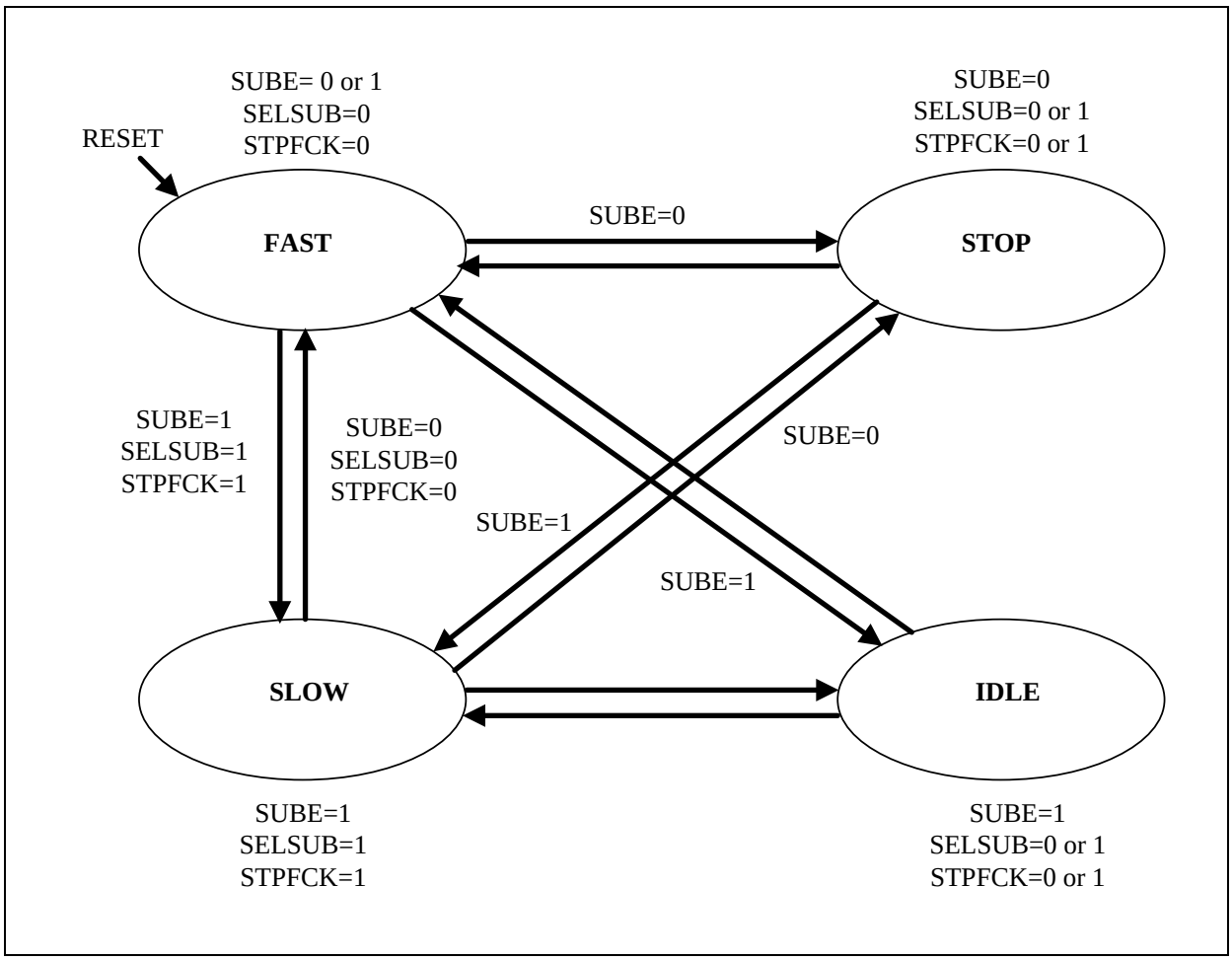
**Stop Mode:**

If slow clock is disabled before executing the SLEEP instruction, every block is turned off and the TM57PE10 enters the "Stop Mode". Stop mode is similar to idle mode. The difference is all clock oscillators either fast or slow is powered-down and no clock is generated.

## 2.6 Dual System Clock Modes Transition

TM57PE10 is operated in one of four modes: Fast Mode, Slow Mode, Idle Mode, and Stop Mode.

**Modes Transition Diagram:**




**Fast Mode transits to Slow Mode:**

Fast mode can be chosen by SYSCFG [9:8] when equals to 11 (Fast Crystal), 00 (External RC), or 01 (Fast Internal RC). The following steps are suggested to be executed by order when fast mode transits to slow mode:

(1) Enable slow clock (SUBE=1)

(2) Switch to slow clock (SELSUB=1)

(3) Stop fast clock (STPFCK=1)

**Slow Mode transits to Fast Mode:**

Slow mode can be enabled by SUBE bit and SELSUB bit in CLKCTRL register. The following steps are suggested to be executed by order when slow mode transits to fast mode:

(1) Enable fast clock (STPFCK=0)

(2) Switch to fast clock (SELSUB=0)

(3) Stop slow clock (SUBE=0)

Note: Stop slow clock (SUBE=0) is optional. Slow clock can keep oscillating to provide Timer2 Counter block in fast mode.

**Idle Mode Setting:**

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

(1) Enable slow clock (SUBE=1)

(2) Switch Timer2 clock source to slow clock (TM2CLK=0)

(3) Execute SLEEP instruction

Idle mode can be woken up by XINT, PAWKUP, PBWAKP, Wake-up Timer, and Timer2 interrupt.

**Stop Mode Setting:**

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

(1) Stop slow clock (SUBE=0)

(2) Execute SLEEP instruction

Stop mode can be woken up by XINT, PAWKUP, PBWAKP, and Wake-up Timer.

**IO setting note in dual clock mode：**

Note: In slow clock modes, PA3 and PA4 must be set as input pull-up mode when slow clock selects SXT or XRC mode. PA3 and PA4 IO setting list is as shown bellow.

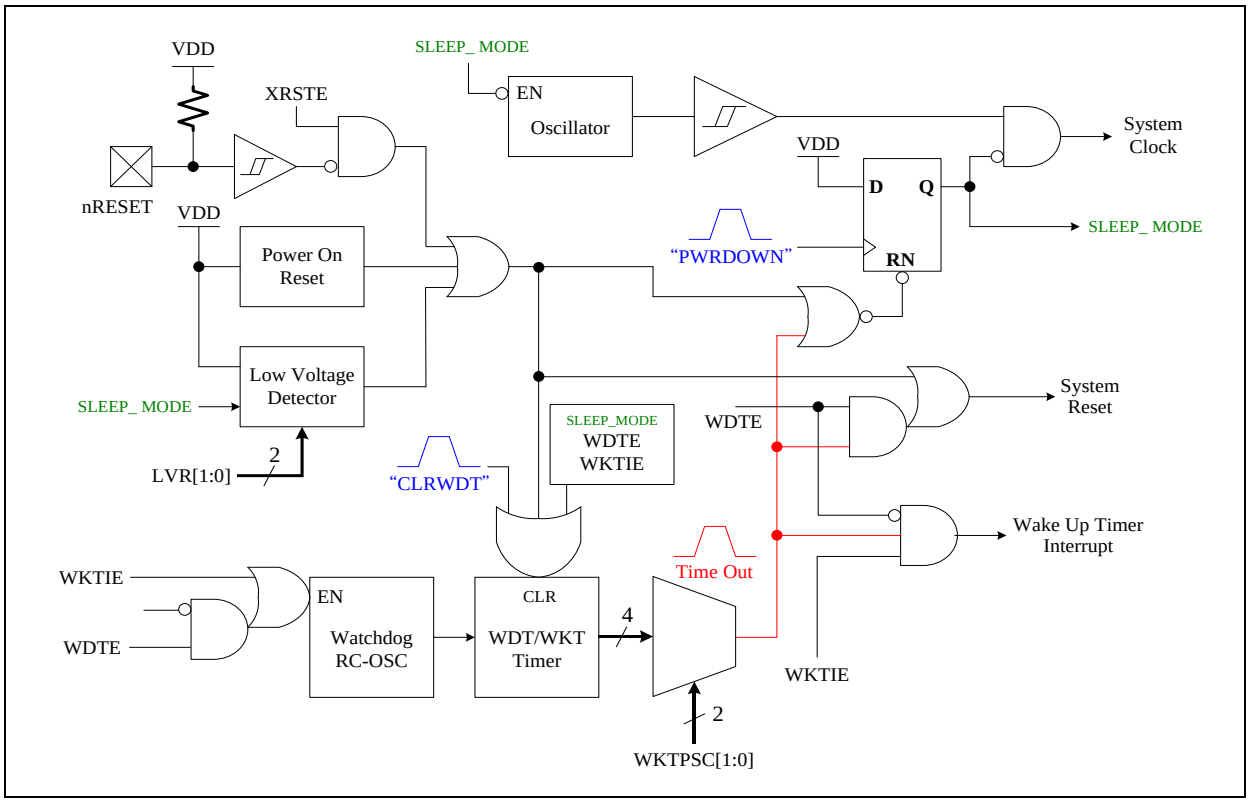
| | Fast Clock | Slow Clock | PAD3 | PAE3 | nPAPU3 | PAD4 | PAE4 | nPAPU4 |
|---|---|---|---|---|---|---|---|---|
| 1 | FIRC | SIRC | ※ | ※ | ※ | ※ | ※ | ※ |
| 2 | FIRC | SXT | 1 | 0 | 0 | 1 | 0 | 0 |
| 3 | FIRC | XRC | ※ | ※ | ※ | 1 | 0 | 0 |
| 4 | FXT | SIRC | ※ | ※ | ※ | ※ | ※ | ※ |
| 5 | XRC | SIRC | ※ | ※ | ※ | ※ | ※ | ※ |

※：Don’t care

# 3. Peripheral Functional Block

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

The WDT and WKT share the same internal RC Timer. The overflow period of WDT/WKT can be selected from 14 ms to 138 ms. The WDT/WKT is cleared by the CLRWDT instruction. If the Watchdog Reset is enabled (WDTE=1), the WDT generates the chip reset signal, otherwise, the WKT only generates overflow time out interrupt. The WDT/WKT works in both normal mode and sleep mode. During sleep mode, user can further choose to enable or disable the WDT/WKT by "WKTIE". If WKTIE=0 in sleep mode (no matter WDTE is 1 or 0), the internal RC Timer stops for power saving. In other words, user keeps the WDT/WKT alive in Sleep Mode by setting WKTIE=1. If the WDTE=1 and WKTIE=0, WDT/WKT timer will be cleared and stopped to power saving in sleep mode. If the WDTE=1 and WKTIE=1, WDT/WKT timer keeps counting in sleep/normal mode. Refer to the following table and figure.

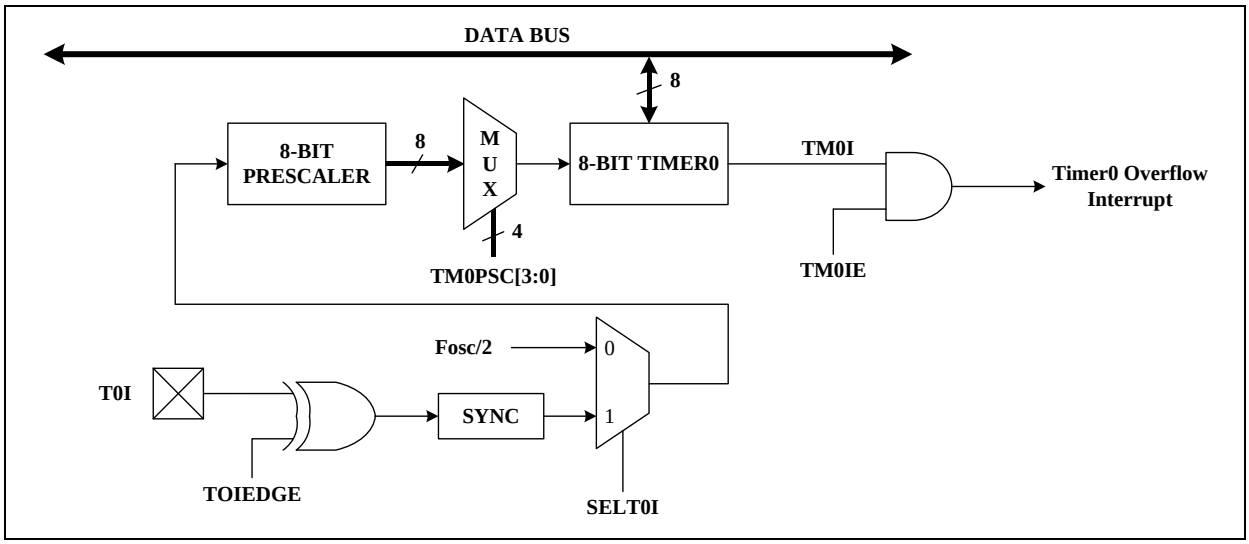


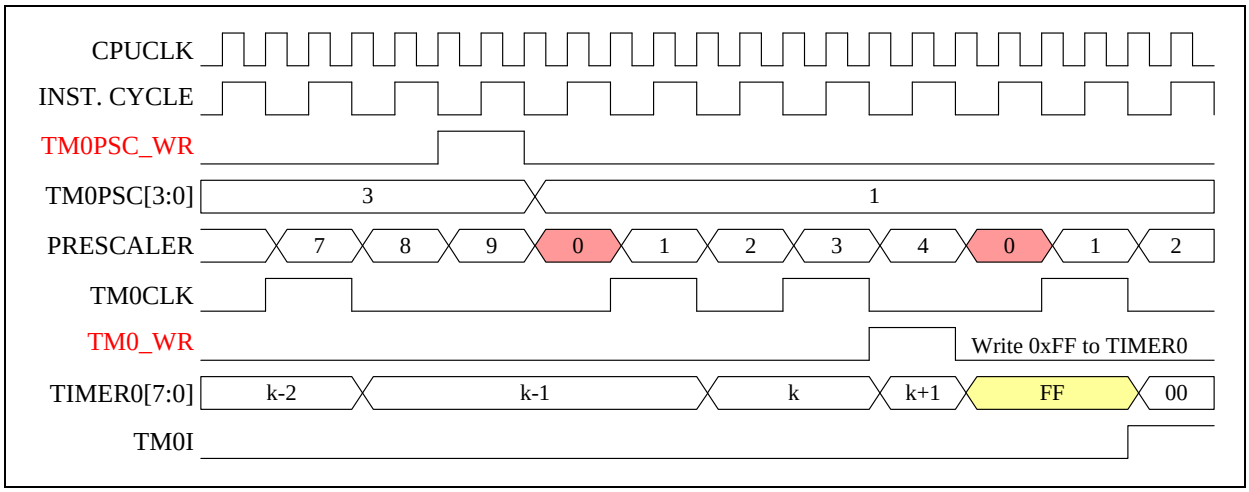
If the user program needs the MCU totally shuts down for power conservation in sleep mode, the following setting of control bits should be followed.

| Mode | WDTE | WKTIE | Watchdog RC Oscillator |
|---|---|---|---|
| Normal Mode | 0 | 0 | Stop |
| | 0 | 1 | Run |
| | 1 | 0 | |
| | 1 | 1 | |
| Sleep Mode | 0 | 0 | Stop |
| | 0 | 1 | Run |
| | 1 | 0 | Stop |
| | 1 | 1 | Run |

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

The Timer0 is an 8-bit wide register of F-Plane. It can be read or written as any other register of F-Plane. Besides, Timer0 increases itself periodically and automatically rolls over based on the pre-scaled clock source, which can be the instruction cycle or T0I input. The Timer0 increase rate is determined by "Timer0 Pre-Scale" (TM0PSC) register in R-Plane. The Timer0 can generate interrupt flag (TM0I) when it rolls over.




Timer0 interrupt frequency by instruction cycle: (Fosc / 2) / div / 256

**Note: The div variable represents the prescale factor by TM0PSC [3:0] select value (1, 2, ~ 128, 256)**

When Fosc = 4 MHz, div = TM0PSC [3:0] when select 4'b0000 = 1

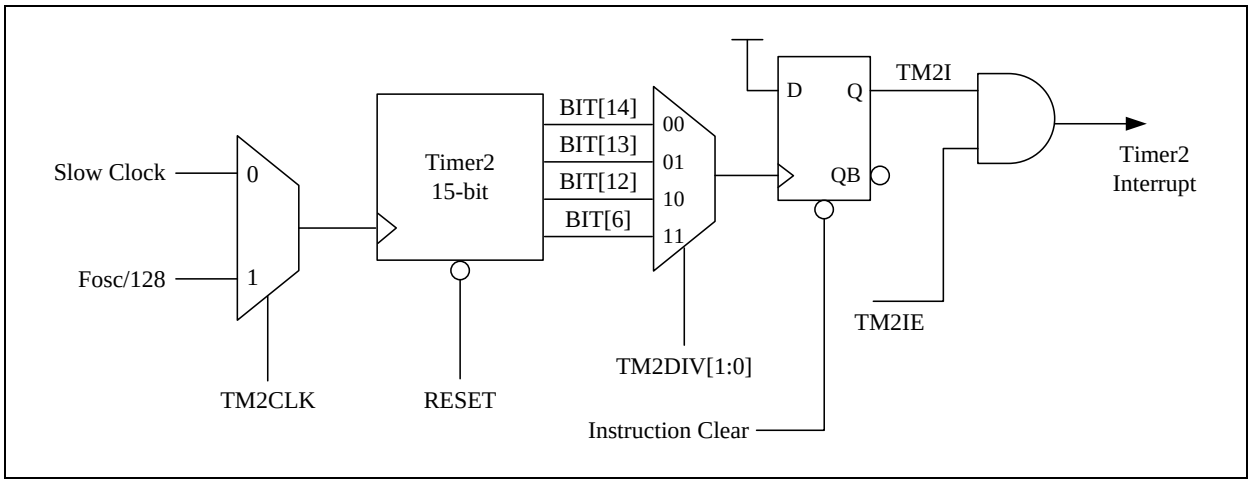
(4M/2)/1/256 = 2M/256 Hz = 7.8125 KHz

Timer0 interrupt frequency by T0I: (T0I) / div / 256

**Note: T0I frequency <= Fosc / 4**

### 3.3 Timer2: 15-bit Timer

The Timer2 is a 15-bit counter and the clock sources are from either Fosc/128 or slow clock. It is used to generate time base interrupt and Timer2 counter block clock. The Timer2 content cannot be read by instructions. It generates interrupt flag (TM2I) with the clock divided by 32768, 16384, 8192, and 128, depends on TM2DIV register bits. Figure shows the block diagram of Timer2.



### 3.4 PWM0: 8-bit PWM

The chip has a built-in 8-bit PWM generator. The source clock comes from Fosc divided by 1, 2, 4, and 8. The PWM0 duty cycle can be changed with writing to PWM0DUTY, writing to PWM0DUTY will not change the current PWM0 duty until the current PWM0 period completes. When current PWM0 period is finish, the new value of PWM0DUTY will be updated to the PWM0BUF.

The PWM0 will be output to PA1 if PWM0E is set to 1. Also, the PWM0 period complete will generate an interrupt when PWM0IE is set to 1. Setting the CLRPWM0 bit will clear the PWM0 counter and load the PWM0DUTY to PWM0BUF, CLRPWM0 bit must be cleared so that the PWM0 counter can count. Figure shows the block diagram of PWM0.

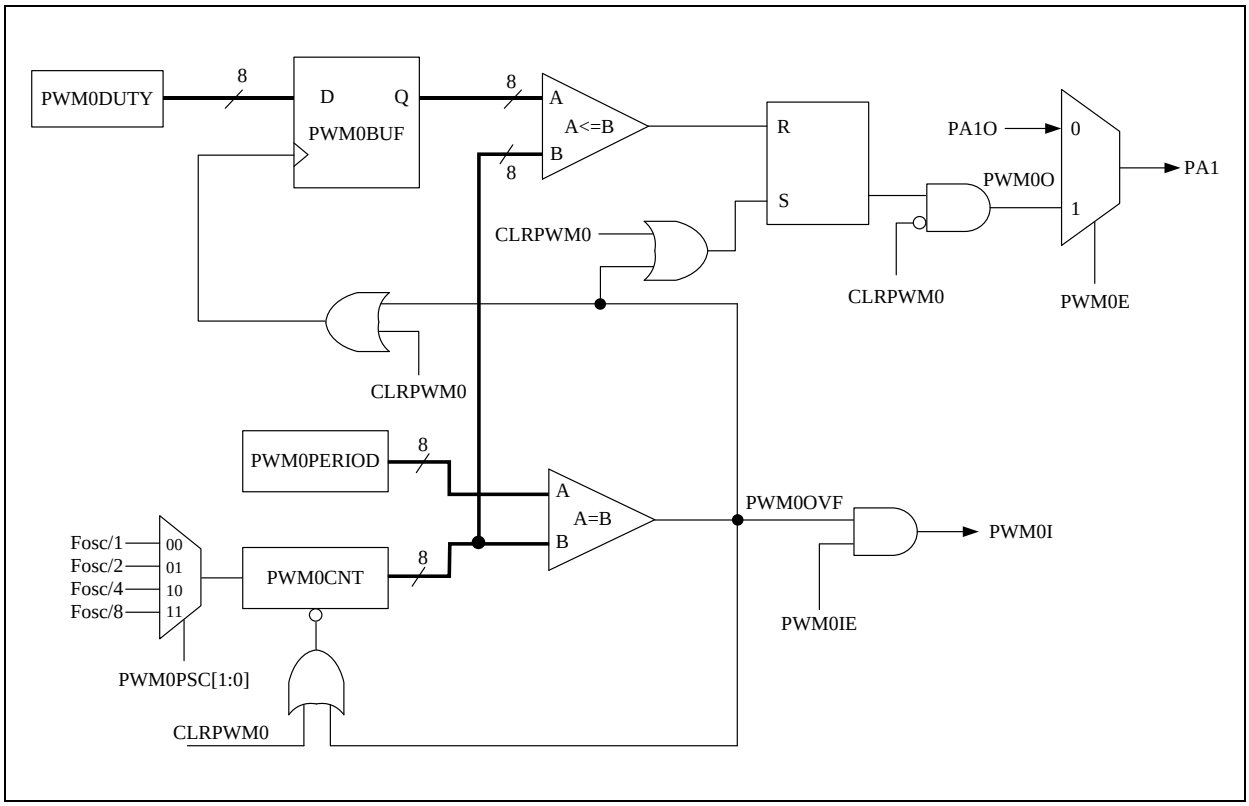
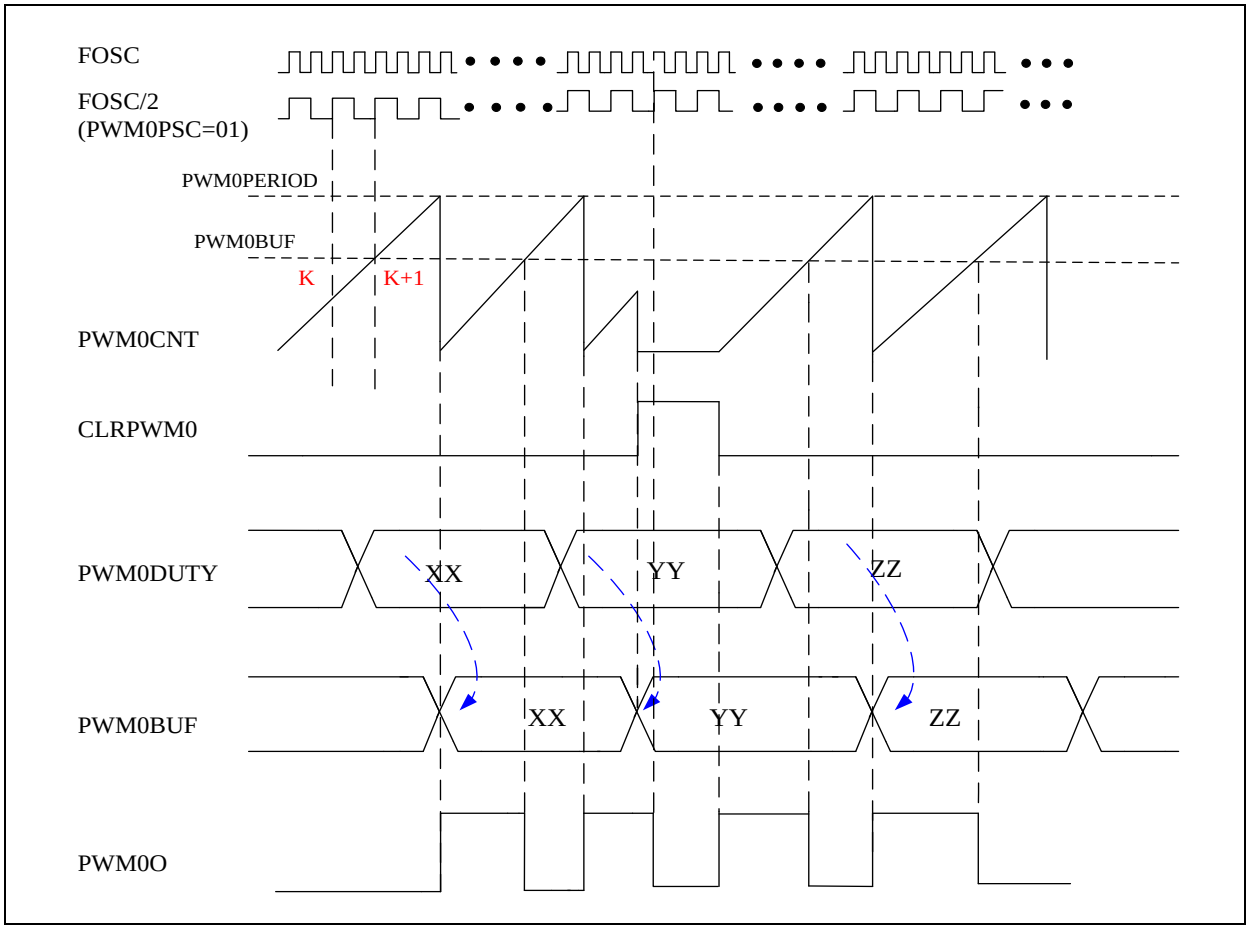


Figure shows the PWM0 waveforms. When CLRPWM0 bit is set to '1', the PWM0 output is cleared to '0' no matter what its current status is. Once the CLRPWM0 bit is cleared to '0', the PWM0 output is set to '1' to begin a new PWM cycle. PWM0 output will be '0' when PWM0CNT greater than or equals to PWM0BUF. PWM0CNT keeps counting up when equals to PWM0PERIOD, the PWM0 output is set to '1' again.




PWM0 output duty = [PWM0DUTY / (PWM0PERIOD + 1)]

When PWM0DUTY = 80H, PWM0PERIOD = FFH, PWM0 output duty will be 1/2

PWM0 output frequency = (Fosc) / div / (PWM0PERIOD + 1)

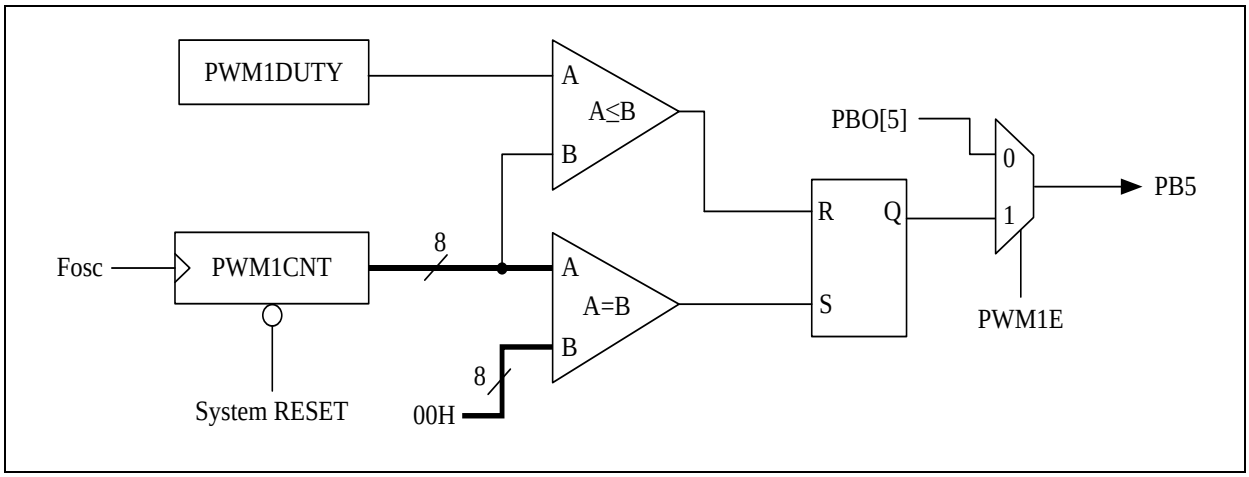
Note: The div variable represents the prescale factor by PWM0PSC [1:0] select value (1, 2, 4, 8)

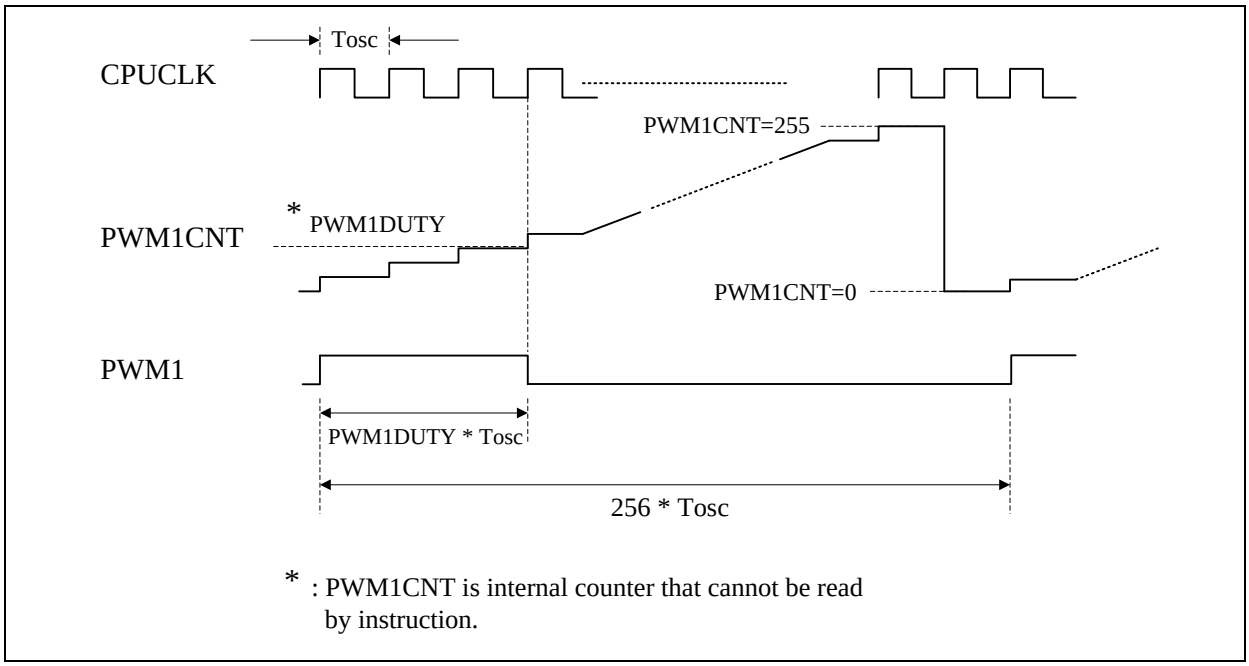
div = PWM0PSC [1:0] when select 2'b00 = 1

When PWM0PERIOD = FFH, Fosc = 4 MHz, PWM0 output frequency = (4M)/1/256 = 15.625 KHz

### 3.5 PWM1: 8-bit PWM

PWM1 is a simple fixed frequency and duty cycle variable PWM generator. The PWM frequency is fixed, the period is system clock counts from 0 to 255. The duty can be set via PWM1DUTY register. The output of PWM1 shares the pin PB5 that can be selected by PWM1E control bit. Figure is the block diagram of PWM1.




PWM1 output duty = [PWM1DUTY / 256]

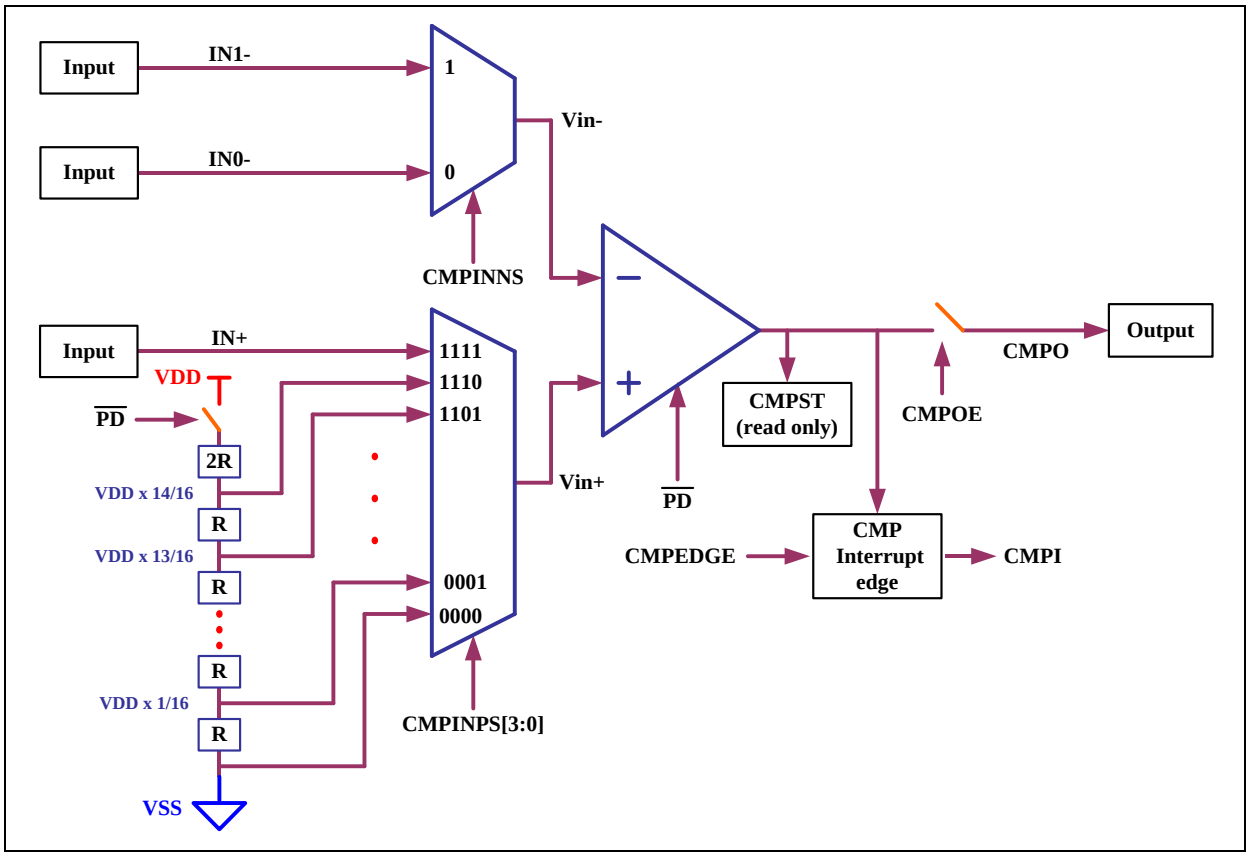
When PWM1DUTY = 80H, PWM1 output duty will be 1/2

PWM1 output frequency = (Fosc) / 256

When Fosc = 4 MHz, PWM1 output frequency = (4M)/256 = 15.625 KHz

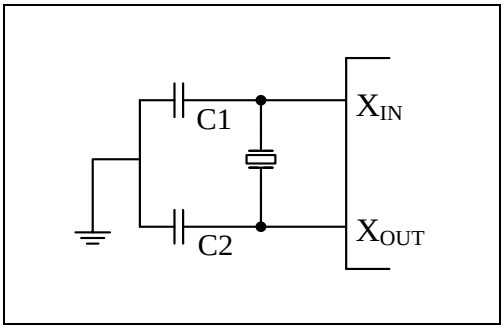
### 3.6 Analog Comparator

TM57PE10 includes an analog comparator. It can be enabled by CMPEN which is in R-Plane 17H Bit7. The analog comparator compares the input values on the positive pin Vin+ and negative pin Vin-. When the voltage on positive pin is higher than the voltage on the negative pin, the analog comparator out (CMPO) is set. The output status CMPST can be read from F-Plane 14H Bit3, or output to pin by setting CMPOE which in R-plane 17H Bit5. The analog comparator can generate interrupt (CMPI) when the output status changes. The user can select interrupt triggering on comparator output rise or fall. The input source of negative pin can be selected from IN0- or IN0+ by CMPINNS. The analog comparator supports internal reference voltage. The internal reference voltage provides the range of output voltage with 15 distinct levels. The range can be selected by CMPINPS. A block diagram of the analog comparator is shown below.

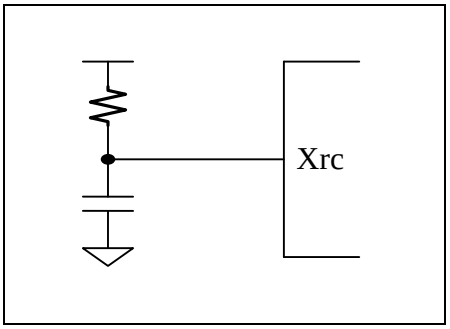


### 3.7 System Clock Oscillator

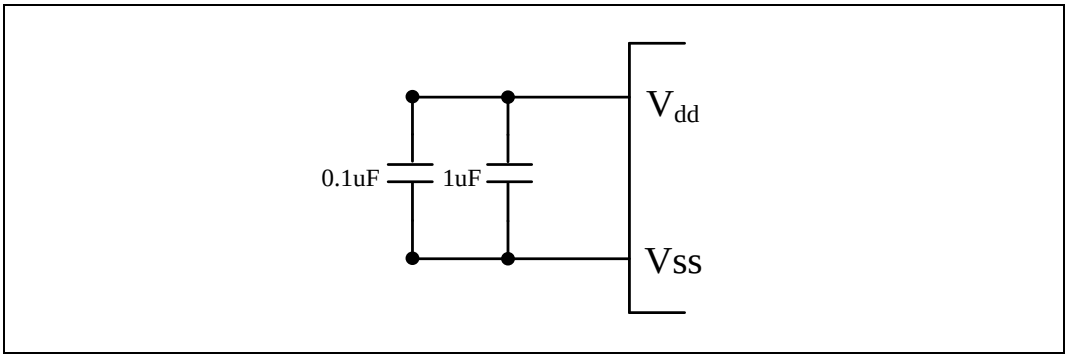
System clock can be operated in four different oscillation modes, which is selected by setting the CLKS in the SYSCFG register. In Slow/Fast Crystal mode, a crystal or ceramic resonator is connected to the Xin and Xout pins to establish oscillation. In external RC mode, the external resistor and capacitor determine the oscillation frequency. In the fast internal RC mode, the on chip oscillator generates 4/8 MHz system clock. In this mode, PCB Layout may have strong effect on the stability of Internal Clock Oscillator. Since power noise degrades the performance of Internal Clock Oscillator, placing power supply bypass capacitors 1 uF and 0.1 uF very close to VDD/VSS pins improves the stability of clock and the overall system.




External Oscillator Circuit
(Crystal or Ceramic)




External RC Oscillator

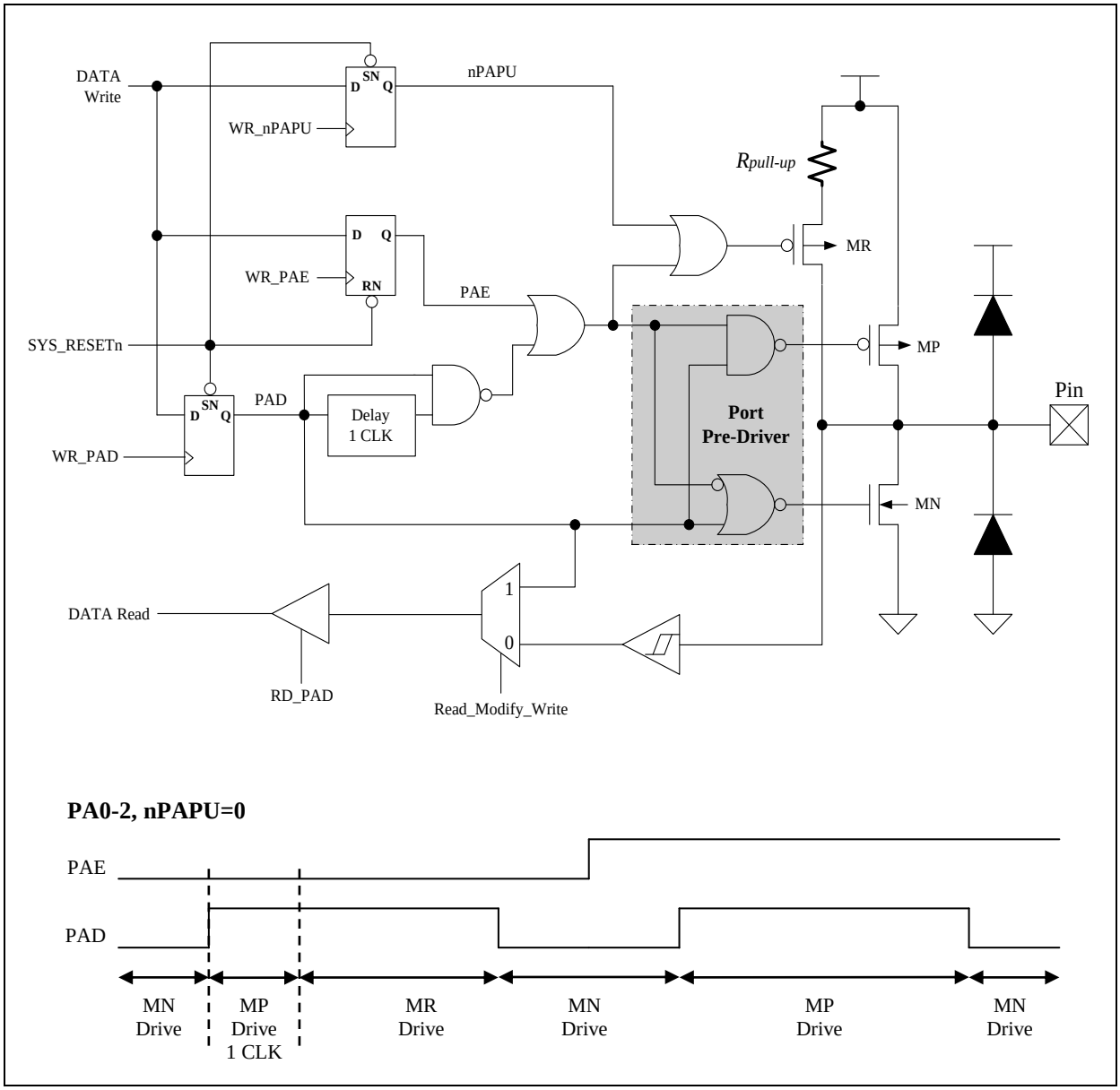



Internal RC Mode

# 4. I/O Port

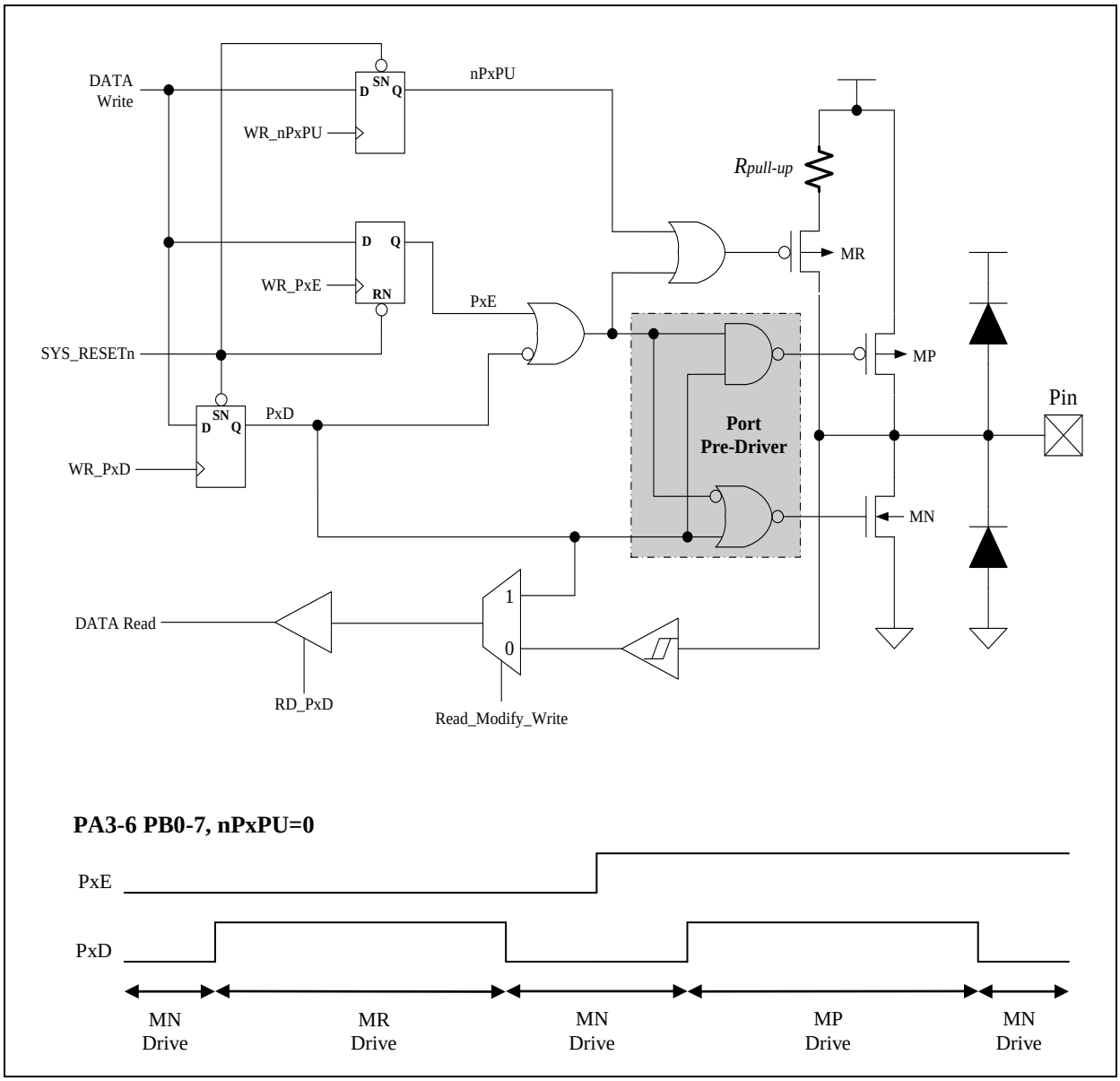
## 4.1 PA0-2

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 other instructions, CPU reads the pin state. The so-called "Read-Modify-Write" instruction includes BSF, BCF and all instructions using F-Plane as destination.



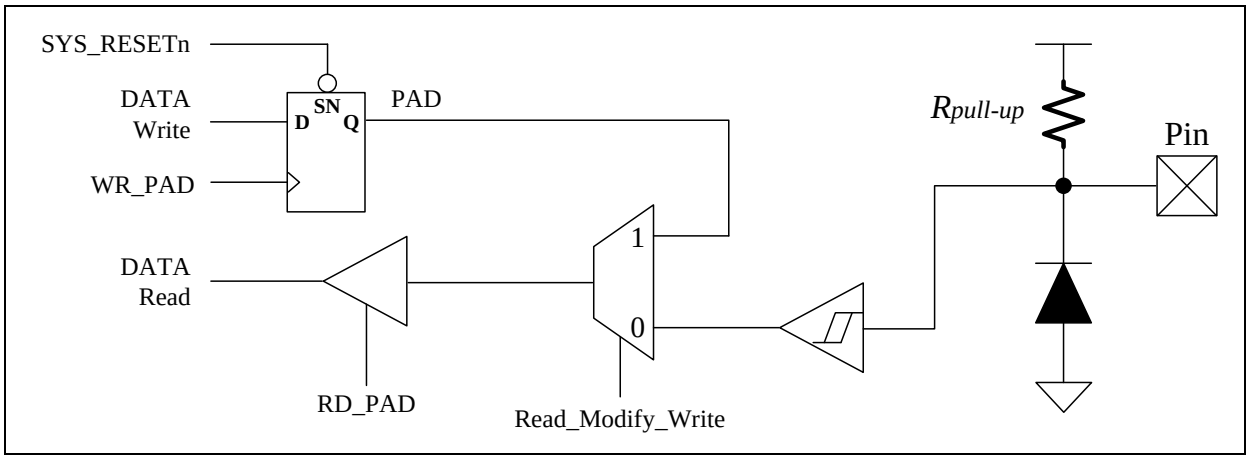
### 4.2 PA3-6, PB0-7

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.



### 4.3 PA7

PA7 can be only used in Schmitt-trigger input mode. The pull-up resistor is always connected to this pin.



# MEMORY MAP

## F-Plane

| Name | Address | R/W | Rst | Description |
|---|---|---|---|---|
| **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 |
| **TIMER0** | 01.7~0 | R/W | 0 | Timer0 content |
| **PC** | 02.7~0 | R/W | 0 | Programming Counter [7~0] |
| **-** | 03.7 | - | - | Reserved |
| **GBIT1** | 03.6 | R/W | 0 | General purpose bit 1 |
| **-** | 03.5 | - | - | Reserved |
| **TO** | 03.4 | R | 0 | WDT time out flag |
| **PD** | 03.3 | R | 0 | Sleep mode flag |
| **ZFLAG** | 03.2 | R/W | 0 | Zero flag |
| **DCFLAG** | 03.1 | R/W | 0 | Decimal Carry flag |
| **CFLAG** | 03.0 | R/W | 0 | Carry flag |
| **GBIT2** | 04.7 | R/W | 0 | General purpose bit 2 |
| **FSR** | 04.6~0 | R/W | - | File Select Register, indirect address mode pointer |
| **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 |
| **PBD** | 06.7~0 | R | - | Port B pin or “data register” state |
| | | W | FF | Port B output data register |
| **PWM0IE** | 08.7 | R/W | 0 | PWM0 interrupt enable, 1=enable, 0=disable |
| **TM2IE** | 08.6 | R/W | 0 | Timer2 interrupt enable, 1=enable, 0=disable |
| **CMPIE** | 08.5 | R/W | 0 | Comparator interrupt enable, 1=enable, 0=disable |
| **TM0IE** | 08.4 | R/W | 0 | Timer0 interrupt enable, 1=enable, 0=disable |
| **WKTIE** | 08.3 | R/W | 0 | Wakeup Timer interrupt enable, 1=enable, 0=disable |
| **XINT2E** | 08.2 | R/W | 0 | INT2 pin interrupt enable, 1=enable, 0=disable |
| **XINT1E** | 08.1 | R/W | 0 | INT1 pin interrupt enable, 1=enable, 0=disable |
| **XINT0E** | 08.0 | R/W | 0 | INT0 pin interrupt enable, 1=enable, 0=disable |
| **PWM0I** | 09.7 | R | - | PWM0 interrupt event pending flag, set by H/W while PWM0 overflows |
| | | W | 0 | write 0: clear this flag; write 1: no action |
| **TM2I** | 09.6 | R | - | Timer2 interrupt event pending flag, set by H/W while Timer2 overflows |
| | | W | 0 | write 0: clear this flag; write 1: no action |
| **CMPI** | 09.5 | R | - | Comparator interrupt event pending flag |
| | | W | 0 | write 0: clear this flag; write 1: no action |
| **TM0I** | 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 |
| **WKTI** | 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 |
| **XINT2** | 09.2 | R | - | INT2 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 |

| Name | Address | R/W | Rst | Description |
|---|---|---|---|---|
| **XINT2** | 09.2 | R | - | INT2 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 |
| **XINT1** | 09.1 | R | - | INT1 interrupt event pending flag, set by H/W at INT1 pin's falling edge |
| | | W | 0 | write 0: clear this flag; write 1: no action |
| **XINT0** | 09.0 | R | - | INT0 interrupt event pending flag, set by H/W at INT0 pin's f/r edge |
| | | W | 0 | write 0: clear this flag; write 1: no action |
| **PWM0DUTY** | 12.7~0 | R/W | 0 | PWM0 duty |
| **PWM1DUTY** | 13.7~0 | R/W | 0 | PWM1 duty |
| **SELSUB** | 14.7 | R/W | 0 | Select slow clock as CPUCLK |
| **STPFCK** | 14.6 | R/W | 0 | Stop fast clock |
| **SUBE** | 14.5 | R/W | 0 | Slow clock enable |
| **-** | 14.4 | - | - | Reserved |
| **CMPST** | 14.3 | R | - | Comparator output status |
| **CLRTM2** | 14.2 | R/W | 0 | Write 1 to clear Timer2, auto cleared by H/W |
| **STOPTM0** | 14.1 | R/W | 0 | Stop Timer0 counting |
| **CLRPWM0** | 14.0 | R/W | 1 | PWM0 clear and hold |
| **SRAM** | 20~4F | R/W | - | Internal RAM |

## R-Plane

| Name | Address | R/W | Rst | Description |
|---|---|---|---|---|
| **T0IEDGE** | 02.5 | W | 0 | 0: T0I (PA2) rising edge to increase Timer0/PSC count<br>1: T0I (PA2) falling edge to increase Timer0/PSC count |
| **SELT0I** | 02.4 | W | 0 | 0: Timer0/PSC clock source is “Instruction Cycle”<br>1: Timer0/PSC clock source is T0I pin |
| **TM0PSC** | 02.3~0 | W | 0 | 0000: Timer0 input clock divided by 1<br>0001: Timer0 input clock divided by 2<br>~<br>0111: Timer0 input clock divided by 128<br>1000: Timer0 input clock divided by 256 |
| **PWRDOWN** | 03 | W | - | write this register to enter Power-Down Mode |
| **CLRWDT** | 04 | W | - | write this register to clear WDT/WKT |
| **PAE** | 05.6~3 | W | 0 | 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 | 0: the pin is pseudo-open-drain output or Schmitt-trigger input<br>1: the pin is CMOS push-pull output |
| **PBE** | 06.7~0 | W | 0 | 0: the pin is open-drain output or Schmitt-trigger input<br>1: the pin is CMOS push-pull output |
| **nPAPU** | 08.6~0 | W | 7F | 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 Crystal or external RC oscillation<br>1: the pin pull up resistor is disabled |
| **nPBPU** | 09.7~0 | W | FF | 0: the pin pull up resistor is enabled<br>1: the pin pull up resistor is disabled |
| **INT0EDGE** | 0b.4 | W | 0 | 0: INT0 pin falling edge to trigger interrupt event<br>1: INT0 pin rising edge to trigger interrupt event |
| **CLK2PIN** | 0b.3 | W | 0 | 0: No Instruction Clock output to PA3 pin<br>1: Instruction Clock output to PA3 pin for external/internal RC mode |
| - | 0b.2 | - | - | Reserved |
| **WKTPSC** | 0b.1~0 | W | 11 | WDT/WKT pre-scale option or SIRC frequency select<br>WDT/WKT pre-scale option<br>00: WDT/WKT period is 14 ms, @5V; 18 ms, @3V<br>01: WDT/WKT period is 28 ms, @5V; 36 ms, @3V<br>10: WDT/WKT period is 56 ms, @5V; 72 ms, @3V<br>11: WDT/WKT period is 108 ms, @5V; 138 ms, @3V<br>SIRC frequency select<br>00: SIRC Frequency is 151 KHz, @5V; 115 KHz, @3V<br>01: SIRC Frequency is 37 KHz, @5V; 29 KHz, @3V<br>10: SIRC Frequency is 9.4 KHz, @5V; 7.2 KHz, @3V<br>11: SIRC Frequency is 2.4 KHz, @5V; 1.9 KHz, @3V |

| Name | Address | R/W | Rst | Description |
|---|---|---|---|---|
| **PWM0PERIOD** | 10.7~0 | W | FF | PWM0 period |
| **PWM1E** | 11.3 | W | 0 | PWM1 positive output to PB5 pin |
| **PWM0E** | 11.2 | W | 0 | PWM0 positive output to PA1 pin |
| **PWM0PSC** | 11.1~0 | W | 0 | PWM0 Pre-Scale, 0: div1, 1: div2, 2: div4, 3: div8 |
| **PAWKUP** | 13.6~1 | W | 0 | Enable PA6~PA1 pin low level wake up |
| **TM2CLK** | 14.4 | W | 0 | Timer2 clock source<br>0: slow clock<br>1: CPUCLK/128 |
| **TM2DIV** | 14.3~2 | W | 0 | Timer2 interrupt is Timer2 clock divide by –<br>0: 32768, 1: 16384 , 2: 8192 ,3 :128 |
| **SUBTYP** | 14.1~0 | W | 0 | Slow clock type 0: SXT, 1: SIRC, 2: XRC |
| **CMPEN** | 17.7 | W | 0 | Comparator enable |
| **CMPEDGE** | 17.6 | W | 0 | 0: Comparator falling edge to trigger interrupt event<br>1: Comparator rising edge to trigger interrupt event |
| **CMPOE** | 17.5 | W | 0 | Comparator output to pin enable |
| **CMPINNS** | 17.4 | W | 0 | Comparator Vin- input select, 0: IN0-, 1: IN1- |
| **CMPINPS** | 17.3~0 | W | 0 | Comparator Vin+ input select<br>0000: $V_{SS}$<br>0001: $V_{DD}$ * 1/16<br>~<br>1110: $V_{DD}$ * 14/16<br>1111: IN+ |
| **PBWKUP** | 18.6~1 | W | 0 | Enable PB6~PB1 pin low level wake up |

# INSTRUCTION SET

Each instruction is a 14-bit word divided into an Op Code, which specifies the instruction type, and one or more operands, which further specify the operation of the instruction. The instructions can be categorized as byte-oriented, bit-oriented and literal operations list in the following table.

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

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

| Field / Legend | Description |
|---|---|
| f | F-Plane Register File Address |
| r | R-Plane Register File Address |
| b | Bit address |
| k | Literal. Constant data or label |
| d | Destination selection field, 0: Working register, 1: Register file |
| W | Working Register |
| Z | Zero Flag |
| C | Carry Flag |
| 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 |
| SLEEP | | 00 0000 0000 0011 | 1 | TO,PD | Go into standby 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 — 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 — 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 — B : PC = LABEL1<br>GOTO LOOP — A : CNT = CNT – 1<br>CONTINUE — 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 — 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 — 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 — B : PC = LABEL1<br>GOTO LOOP — A : CNT = CNT + 1<br>CONTINUE — 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 — 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 — 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 |

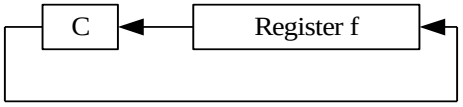
| NOP | No Operation |
|---|---|
| Syntax | NOP |
| Operands | - |
| Operation | No Operation |
| Status Affected | - |
| OP-Code | 00 0000 0000 0000 |
| Description | No Operation |
| Cycle | 1 |
| Example | NOP - |

| 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 |

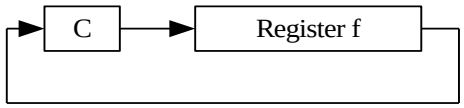
| RETI | Return from Interrupt |
|---|---|
| Syntax | RETI |
| Operands | - |
| Operation | PC ← TOS, GIE ← 1 |
| Status Affected | - |
| OP-Code | 00 0000 0110 0000 |
| Description | Return from Interrupt. Stack is POPed and Top-of-Stack (TOS) is loaded in to the PC. Interrupts are enabled. This is 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 B : W = 0x07<br>: A : W = value of k8<br>TABLE ADDWF PCL, 1<br>RETLW k1<br>RETLW k2<br>:<br>RETLW kn |

## RLF — Rotate Left "f" through Carry

| | |
|---|---|
| Syntax | RLF f [,d] |
| Operands | f : 00h ~ 7Fh, d : 0, 1 |
| Operation |   |
| Status Affected | C |
| OP-Code | 00 1101 dfff ffff |
| Description | The contents of register 'f' are rotated one bit to the left through the Carry Flag. If 'd' is 0, the result is placed in the W register. If 'd' is 1, the result is stored back in register 'f'. |
| Cycle | 1 |
| Example | RLF REG1, 0 — B : REG1 = 1110 0110, C = 0<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 SLEEP mode with the oscillator stopped. |
| 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℃)

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

## 2. DC Characteristics ($T_A$ = 25℃, $V_{DD}$ = 2.0V to 5.5V)

| Parameter | Sym | Conditions | | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|---|
| Operating Voltage | $V_{DD}$ | Fast mode, 25℃, $F_{OSC}$ = 24 MHz | | 2.8 | – | 5.5 | V |
| | | Fast mode, 25℃, $F_{OSC}$ = 16 MHz | | 2.1 | – | 5.5 | |
| | | Fast mode, 25℃, $F_{OSC}$ = 8 MHz | | 1.7 | – | 5.5 | |
| | | Fast mode, 25℃, $F_{OSC}$ = 4 MHz | | 1.5 | – | 5.5 | |
| | | Slow mode, 25℃, SIRC | | 1.5 | – | 5.5 | |
| Input High Voltage | $V_{IH}$ | All Input, except PA7 | $V_{DD}$ = 5V | 0.7$V_{DD}$ | – | – | V |
| | | | $V_{DD}$ = 3V | 0.7$V_{DD}$ | – | – | V |
| | | PA7 | $V_{DD}$ = 5V | 0.8$V_{DD}$ | – | – | V |
| | | | $V_{DD}$ = 3V | 0.8$V_{DD}$ | – | – | V |
| Input Low Voltage | $V_{IL}$ | All Input, except PA7 | $V_{DD}$ = 5V | – | – | 0.2$V_{DD}$ | V |
| | | | $V_{DD}$ = 3V | – | – | 0.2$V_{DD}$ | V |
| | | PA7 | $V_{DD}$ = 5V | – | – | 0.2$V_{DD}$ | V |
| | | | $V_{DD}$ = 3V | – | – | 0.2$V_{DD}$ | V |
| Output High Voltage | $V_{OH}$ | All Output | $V_{DD}$ = 5V, $I_{OH}$ = 8 mA | 4.4 | – | – | V |
| | | | $V_{DD}$ = 3V, $I_{OH}$ = 4 mA | 2.6 | – | – | V |
| Output Low Voltage | $V_{OL}$ | All Output | $V_{DD}$ = 5V, $I_{OL}$ = 20 mA | – | – | 0.5 | V |
| | | | $V_{DD}$ = 3V, $I_{OL}$ = 10 mA | – | – | 0.3 | V |
| Input Leakage Current (pin high) | $I_{ILH}$ | All Input | $V_{IN}$ = $V_{DD}$ | – | – | 1 | μA |
| Input Leakage Current (pin low) | $I_{ILL}$ | All Input | $V_{IN}$ = 0 V | – | – | –1 | μA |
| Supply Current | $I_{DD}$ | Fast mode, LVR enable, WDT enable | $V_{DD}$ = 5V, $F_{OSC}$ = 16 MHz | – | 5.2 | – | mA |
| | | | $V_{DD}$ = 3V, $F_{OSC}$ = 16 MHz | – | 2.3 | – | |
| | | | $V_{DD}$ = 5V, $F_{OSC}$ = 8 MHz | – | 3 | – | |
| | | | $V_{DD}$ = 3V, $F_{OSC}$ = 8 MHz | – | 1.4 | – | |
| | | | $V_{DD}$ = 5V, $F_{OSC}$ = 4 MHz | – | 1.7 | – | |
| | | | $V_{DD}$ = 3V, $F_{OSC}$ = 4 MHz | – | 0.8 | – | |
| | | | $V_{DD}$ = 5V, FIRC = 8 MHz | – | 2.3 | – | |
| | | | $V_{DD}$ = 3V, FIRC = 8 MHz | – | 1.2 | – | |
| | | | $V_{DD}$ = 5V, FIRC = 4 MHz | – | 1.3 | – | |
| | | | $V_{DD}$ = 3V, FIRC = 4 MHz | – | 0.7 | – | |

| Parameter | Sym | Conditions | | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|---|
| Supply Current | $I_{DD}$ | Slow mode, LVR enable, WKTPSC=11 | $V_{DD}$ = 5V, SXT = 32 KHz | – | 30 | – | μA |
| | | | $V_{DD}$ = 3V, SXT = 32 KHz | – | 10 | – | |
| | | | $V_{DD}$ = 5V, SIRC | – | 19 | – | |
| | | | $V_{DD}$ = 3V, SIRC | – | 5 | – | |
| | | Idle mode, LVR enable | $V_{DD}$ = 5V, SXT = 32 KHz | – | 10.5 | – | |
| | | | $V_{DD}$ = 3V, SXT = 32 KHz | – | 2.9 | – | |
| | | | $V_{DD}$ = 5V, SIRC | – | 9.2 | – | |
| | | | $V_{DD}$ = 3V, SIRC | – | 2.5 | – | |
| | | Idle mode, LVR disable | $V_{DD}$ = 5V, SXT = 32 KHz | – | 9.5 | – | |
| | | | $V_{DD}$ = 3V, SXT = 32 KHz | – | 2.2 | – | |
| | | | $V_{DD}$ = 5V, SIRC | – | 8.2 | – | |
| | | | $V_{DD}$ = 3V, SIRC | – | 1.8 | – | |
| | | Stop mode, LVR enable | $V_{DD}$ = 5V | – | 0.9 | – | |
| | | | $V_{DD}$ = 3V | – | 0.5 | – | |
| | | Stop mode, LVR disable | $V_{DD}$ = 5V | – | – | 0.1 | |
| | | | $V_{DD}$ = 3V | – | – | 0.1 | |
| System Clock Frequency | $F_{OSC}$ | $V_{DD} > LVR_{th}$ | $V_{DD}$ = 5V | – | – | 24 | MHz |
| | | | $V_{DD}$ = 3V | – | – | 24 | |
| | | | $V_{DD}$ = 2.4V | – | – | 20 | |
| | | | $V_{DD}$ = 2V | – | – | 12 | |
| | | | $V_{DD}$ = 1.5V | – | – | 6 | |
| LVR Reference Voltage | | $V_{LVR}$ | | – | 1.5 | – | V |
| | | | | – | 2.3 | – | |
| LVR Hysteresis Voltage | | $V_{HYST}$ | | – | ±0.1 | – | V |
| Low Voltage Detection time | | $t_{LVR}$ | | 100 | – | – | μs |
| Pull-Up Resistor | $R_P$ | $V_{IN}$ = 0 V Ports A/B | $V_{DD}$ = 5V<br>$V_{DD}$ = 3V | – | 66<br>126 | – | KΩ |
| | | $V_{IN}$ = 0 V PA7 | $V_{DD}$ = 5V<br>$V_{DD}$ = 3V | – | 56<br>56 | – | KΩ |

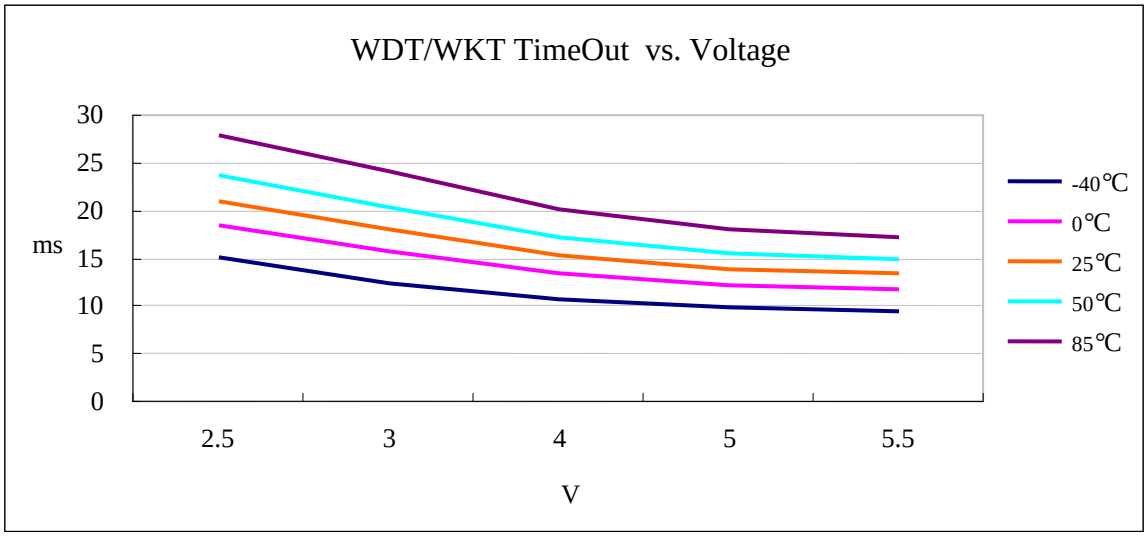
## 3. Clock Timing ($T_A$ = -40℃ to +85℃)

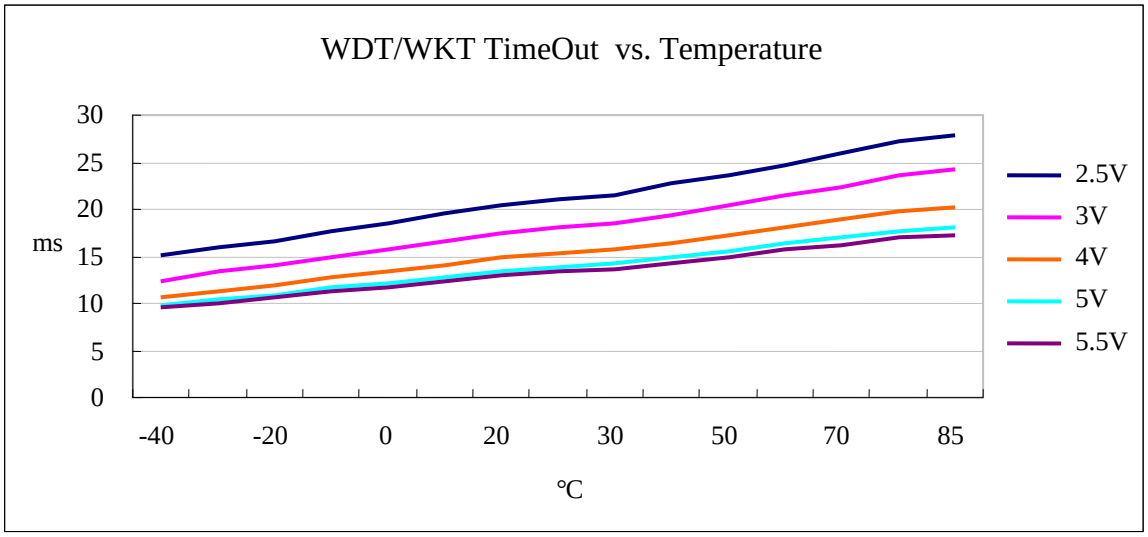
| Parameter | Condition | | | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|---|
| External RC Frequency | $V_{DD}$ = 3V | R = 4.7K | C = 20 pF | – | 2.7 | – | MHz |
| | | R = 10K | C = 100 pF | – | 0.8 | – | |
| | | R = 100K | C = 100 pF | – | 0.1 | – | |
| | $V_{DD}$ = 5V | R = 4.7K | C = 20 pF | – | 3.6 | – | |
| | | R = 10K | C = 100 pF | – | 0.7 | – | |
| | | R = 100K | C = 100 pF | – | 0.08 | – | |
| Fast Internal RC Frequency | 25℃, $V_{DD}$ = 3 ~ 5.5V | | | 7.75 | 8 | 8.25 | |
| | 25℃, $V_{DD}$ = 2.6 ~ 3V | | | 7.6 | 8 | 8.4 | |
| | -40℃ ~ 85℃, $V_{DD}$ = 2.6 ~ 5.5V | | | 7.5 | 8 | 8.5 | |

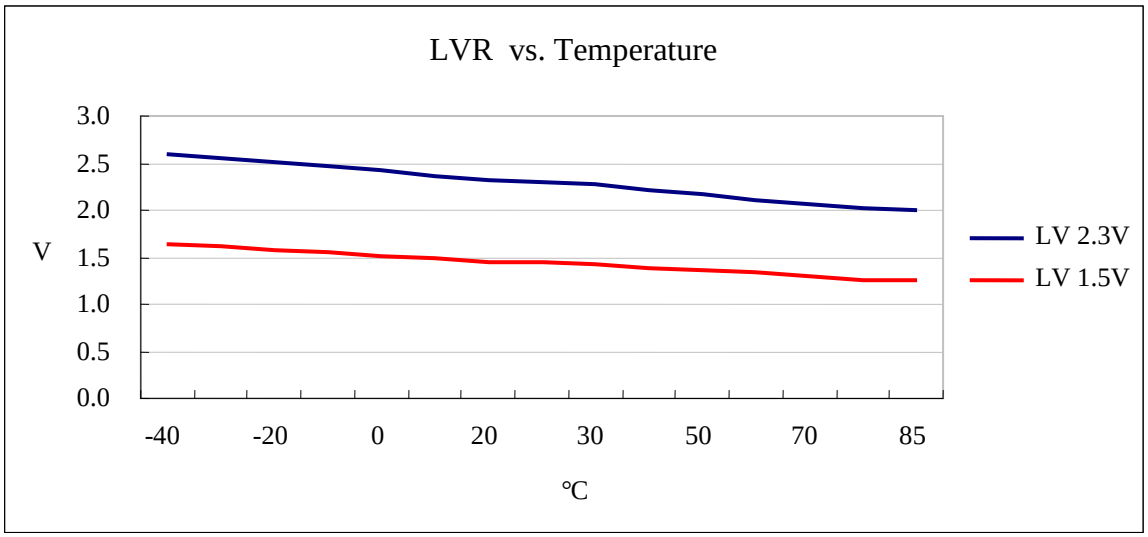
## 4. Reset Timing Characteristics ($T_A$ = -40℃ to +85℃, $V_{DD}$ = 5V)

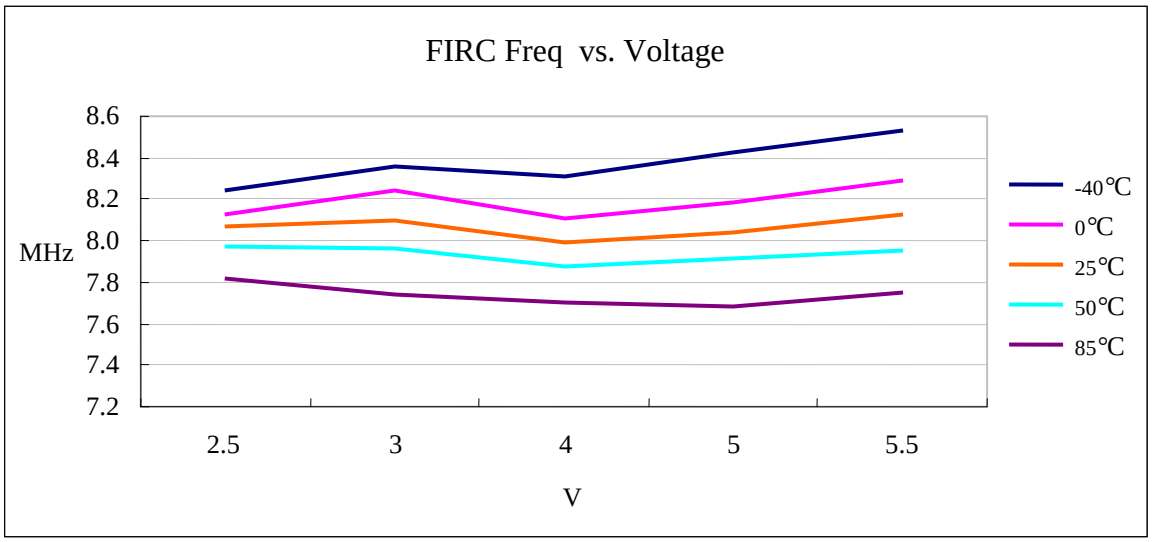
| Parameter | Conditions | Min | Typ | Max | Unit |
|---|---|---|---|---|---|
| RESET Input Low width | Input $V_{DD}$ = 5V ± 10 % | 3 | – | – | μs |
| WDT wakeup time | $V_{DD}$ = 5V, WKTPSC = 00 | – | 14 | – | ms |
| | $V_{DD}$ = 3V, WKTPSC = 00 | – | 18 | – | |
| CPU start up time | $V_{DD}$ = 5V | – | 3.5 | – | ms |

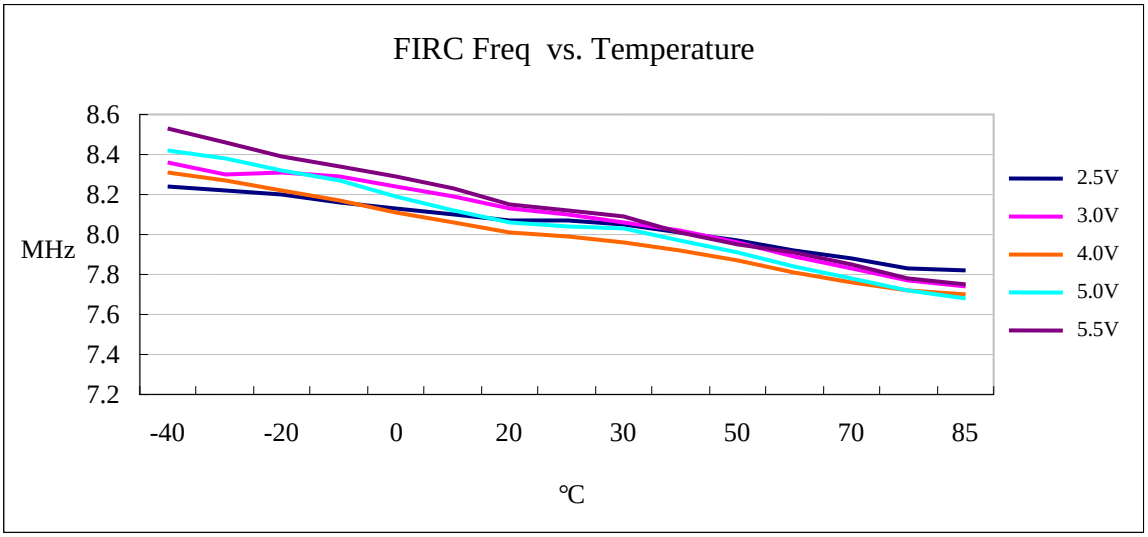
## 5. Characteristic Graphs




LVR vs. Temperature
3.0
2.5
2.0
V
1.5
1.0
0.5
0.0
-40
-20
0
20
30
50
70
85
℃
LV 2.3V
LV 1.5V


FIRC Freq vs. Voltage
8.6
8.4
8.2
MHz
8.0
7.8
7.6
7.4
7.2
2.5
3
4
5
5.5
V
-40℃
0℃
25℃
50℃
85℃

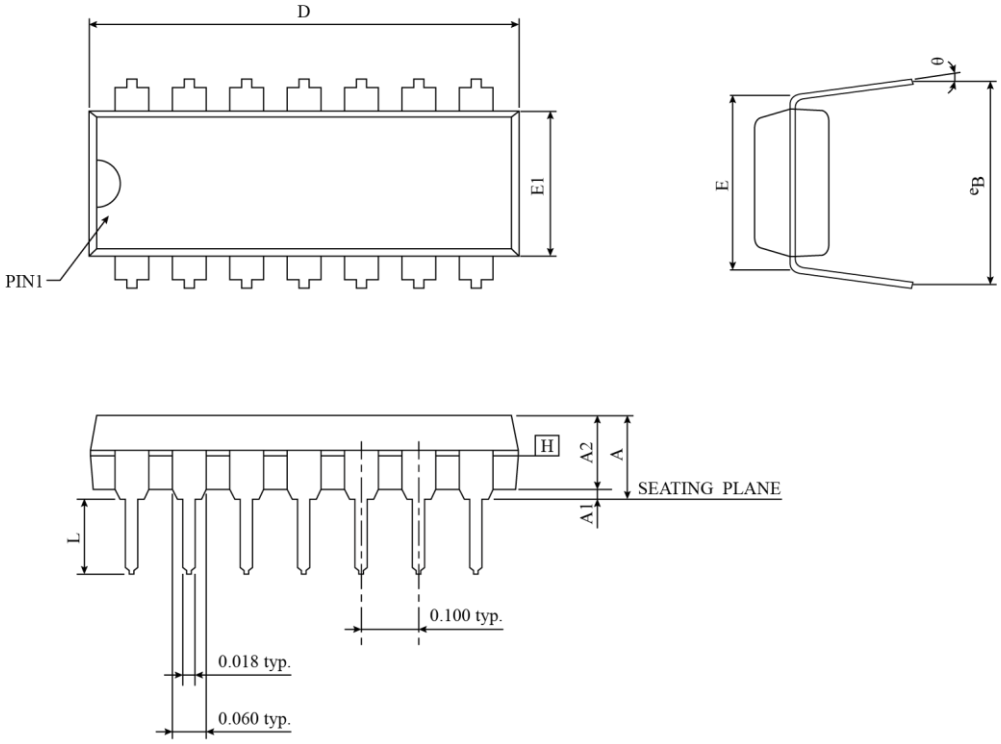

FIRC Freq vs. Temperature
8.6
8.4
8.2
MHz
8.0
7.8
7.6
7.4
7.2
-40
-20
0
20
30
50
70
85
℃
2.5V
3.0V
4.0V
5.0V
5.5V

# PACKAGING INFORMATION

The ordering information:

| Ordering number | Package |
|---|---|
| TM57PE10-OTP | Wafer / Dice blank chip |
| TM57PE10-COD | Wafer / Dice with code |
| TM57PE10-OTP-02 | DIP 14-pin (300 mil) |
| TM57PE10-OTP-15 | SOP 14-pin (150 mil) |
| TM57PE10-OTP-03 | DIP 16-pin (300 mil) |
| TM57PE10-OTP-16 | SOP 16-pin (150 mil) |
| TM57PE10-OTP-04 | DIP 18-pin (300 mil) |
| TM57PE10-OTP-20 | SOP 18-pin (300 mil) |

## DIP-14 ( 300mil ) Package Dimension

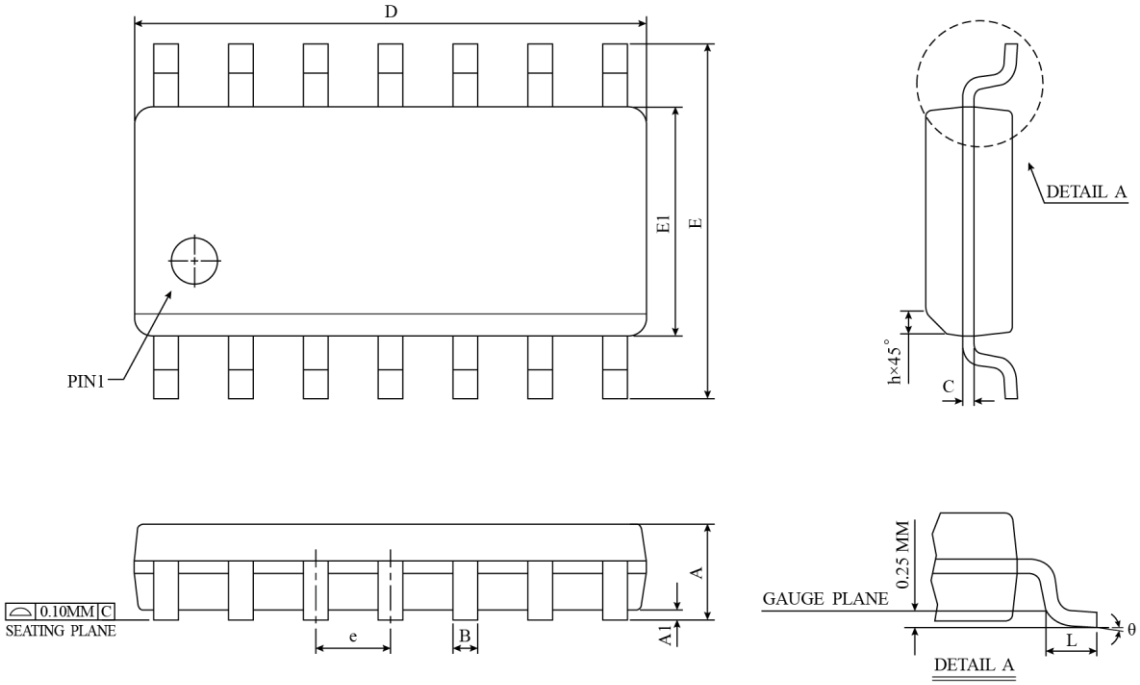



| SYMBOL | DIMENSION IN MM | | DIMENSION IN INCH | |
|---|---|---|---|---|
| | MIN | MAX | MIN | MAX |
| A | - | 5.334 | - | 0.210 |
| A1 | 0.381 | - | 0.015 | - |
| A2 | 3.175 | 3.429 | 0.125 | 0.135 |
| D | 18.669 | 19.685 | 0.735 | 0.775 |
| E | 7.620 BSC | | 0.300 BSC | |
| E1 | 6.223 | 6.477 | 0.245 | 0.255 |
| L | 2.921 | 3.810 | 0.115 | 0.150 |
| eB | 8.509 | 9.525 | 0.335 | 0.375 |
| θ | 0° | 15° | 0° | 15° |
| JEDEC | MS-001 (AA) | | | |

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.

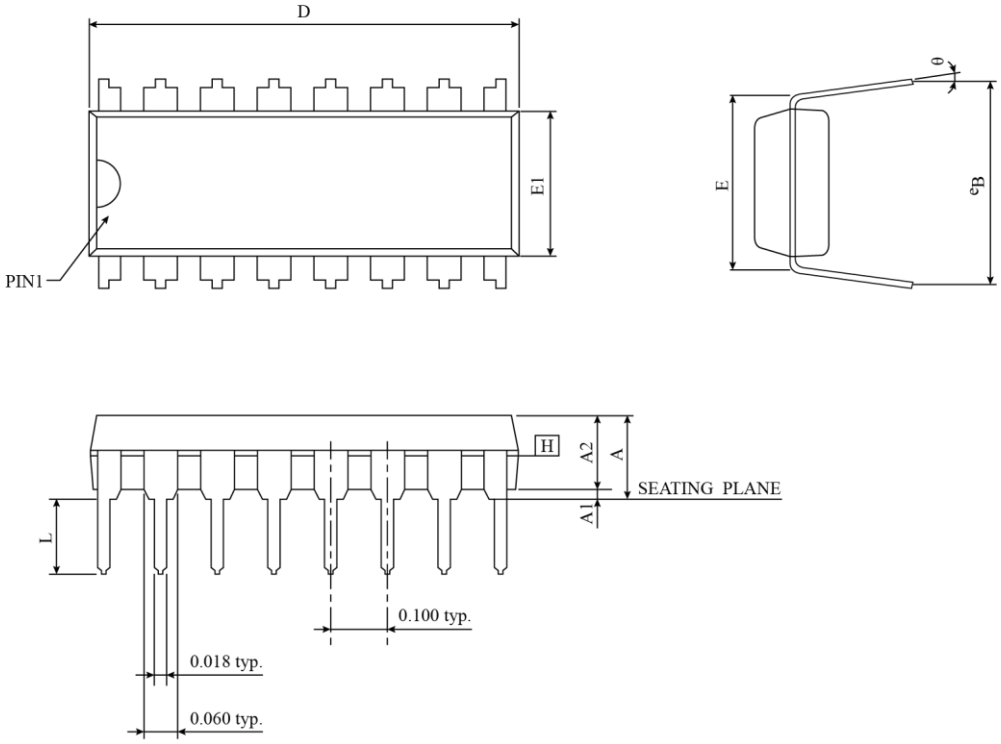
## SOP-14 ( 150mil ) Package Dimension




| SYMBOL | DIMENSION IN MM | | DIMENSION IN INCH | |
|---|---|---|---|---|
| | MIN | MAX | MIN | MAX |
| A | 1.35 | 1.75 | 0.0532 | 0.0688 |
| A1 | 0.10 | 0.25 | 0.0040 | 0.0098 |
| B | 0.33 | 0.51 | 0.013 | 0.020 |
| C | 0.19 | 0.25 | 0.0075 | 0.0098 |
| D | 8.55 | 8.75 | 0.3367 | 0.3444 |
| E | 5.80 | 6.20 | 0.2284 | 0.2440 |
| E1 | 3.80 | 4.00 | 0.1497 | 0.1574 |
| e | 1.27 BSC | | 0.050 BSC | |
| h | 0.25 | 0.50 | 0.0099 | 0.0196 |
| L | 0.40 | 1.27 | 0.016 | 0.050 |
| θ | 0° | 8° | 0° | 8° |
| JEDEC | MS-012 (AB) | | | |

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

## DIP-16 ( 300mil ) Package Dimension

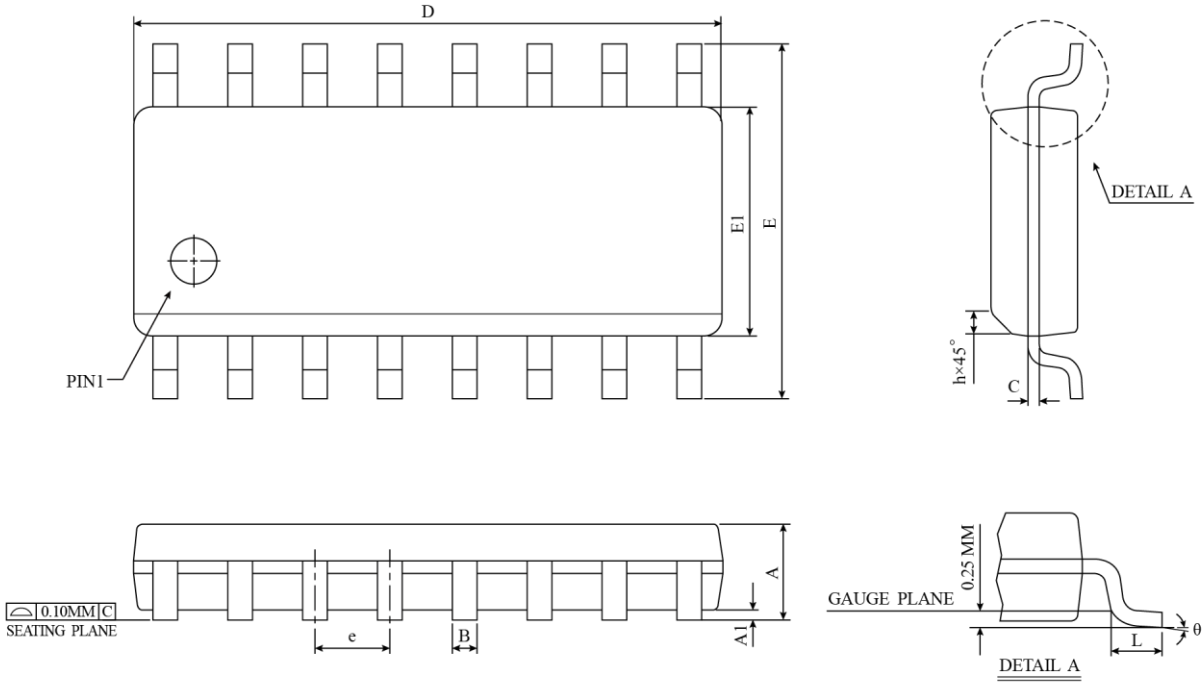



| SYMBOL | DIMENSION IN MM | | DIMENSION IN INCH | |
|---|---|---|---|---|
| | MIN | MAX | MIN | MAX |
| A | - | 4.369 | - | 0.172 |
| A1 | 0.381 | 0.965 | 0.015 | 0.038 |
| A2 | 3.175 | 3.429 | 0.125 | 0.135 |
| D | 18.669 | 19.685 | 0.735 | 0.775 |
| E | 7.620 BSC | | 0.300 BSC | |
| E1 | 6.223 | 6.477 | 0.245 | 0.255 |
| L | 2.921 | 3.810 | 0.115 | 0.150 |
| eB | 8.509 | 9.525 | 0.335 | 0.375 |
| θ | 0° | 15° | 0° | 15° |
| JEDEC | MS-001 (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.

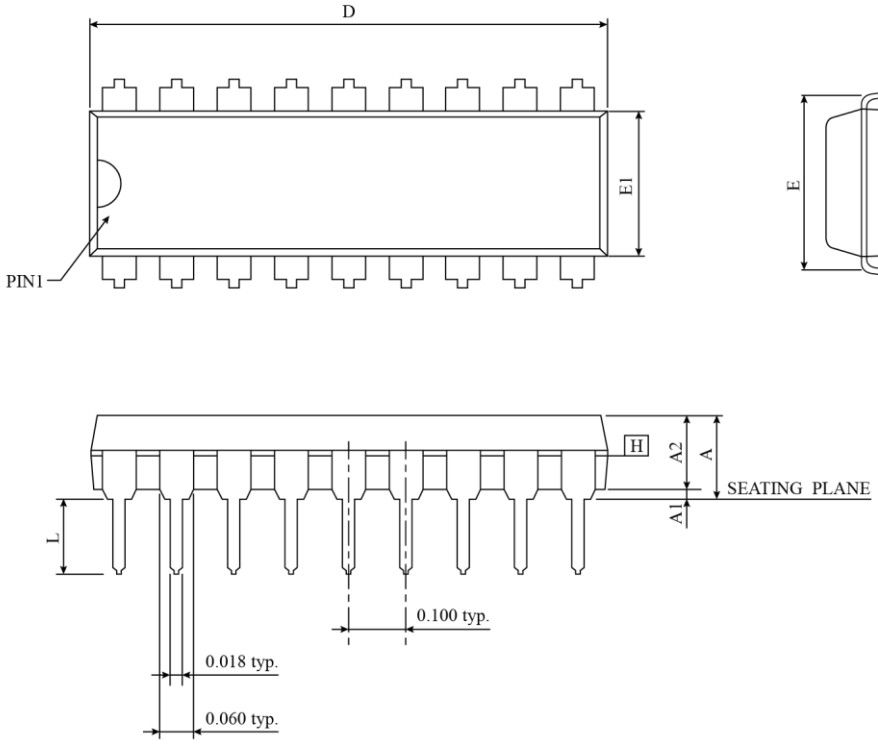
## SOP-16 ( 150mil ) Package Dimension




| SYMBOL | DIMENSION IN MM | | DIMENSION IN INCH | |
|---|---|---|---|---|
| | MIN | MAX | MIN | MAX |
| A | 1.35 | 1.75 | 0.0532 | 0.0688 |
| A1 | 0.10 | 0.25 | 0.0040 | 0.0098 |
| B | 0.33 | 0.51 | 0.013 | 0.020 |
| C | 0.19 | 0.25 | 0.0075 | 0.0098 |
| D | 9.80 | 10.00 | 0.3859 | 0.3937 |
| E | 5.80 | 6.20 | 0.2284 | 0.2440 |
| E1 | 3.80 | 4.00 | 0.1497 | 0.1574 |
| e | 1.27 BSC | | 0.050 BSC | |
| h | 0.25 | 0.50 | 0.0099 | 0.0196 |
| L | 0.40 | 1.27 | 0.016 | 0.050 |
| θ | 0° | 8° | 0° | 8° |
| JEDEC | MS-012 (AC) | | | |

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

## DIP-18 ( 300mil ) Package Dimension

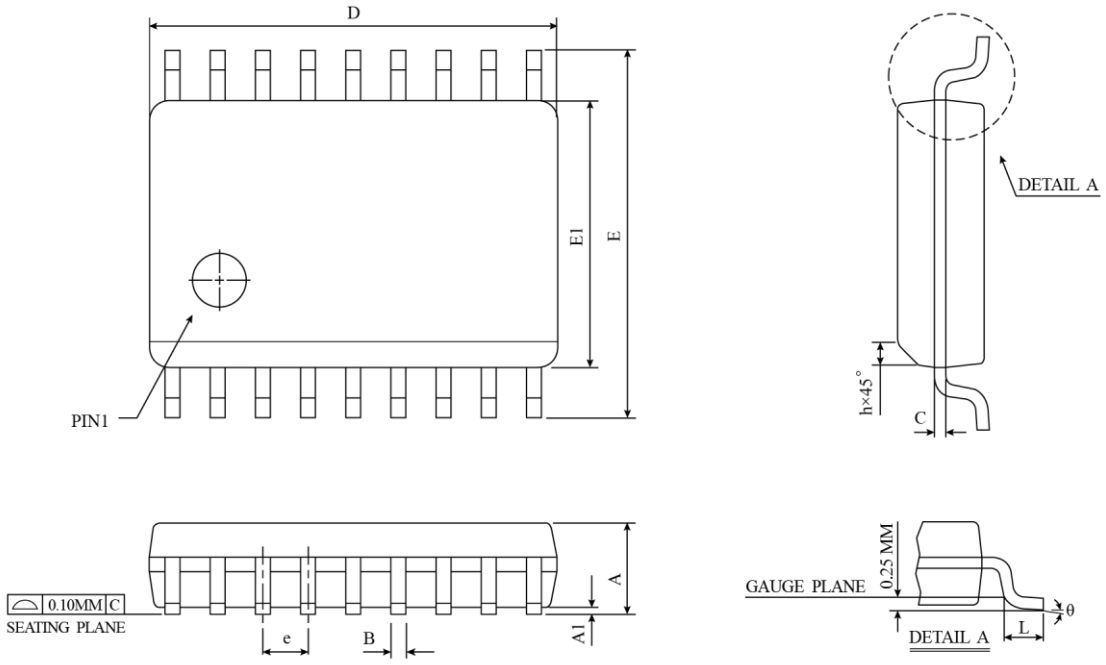



| SYMBOL | DIMENSION IN MM | | DIMENSION IN INCH | |
|---|---|---|---|---|
| | MIN | MAX | MIN | MAX |
| A | - | 5.334 | - | 0.210 |
| A1 | 0.381 | - | 0.015 | - |
| A2 | 3.175 | 3.429 | 0.125 | 0.135 |
| D | 22.352 | 23.368 | 0.880 | 0.920 |
| E | 7.620 BSC | | 0.300 BSC | |
| E1 | 6.223 | 6.477 | 0.245 | 0.255 |
| L | 2.921 | 3.810 | 0.115 | 0.150 |
| eB | 8.509 | 9.525 | 0.335 | 0.375 |
| θ | 0° | 15° | 0° | 15° |
| JEDEC | MS-001 (AC) | | | |

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.

## SOP-18 ( 300mil ) Package Dimension




| SYMBOL | DIMENSION IN MM | | DIMENSION IN INCH | |
|---|---|---|---|---|
| | MIN | MAX | MIN | MAX |
| A | 2.362 | 2.642 | 0.093 | 0.104 |
| A1 | 0.102 | 0.305 | 0.004 | 0.012 |
| B | 0.406 typ | | 0.016 typ | |
| C | 0.254 typ | | 0.010 typ | |
| D | 11.354 | 11.760 | 0.447 | 0.463 |
| E | 10.008 | 10.643 | 0.394 | 0.419 |
| E1 | 7.391 | 7.595 | 0.291 | 0.299 |
| e | 1.27 typ | | 0.050 typ | |
| h | 0.508 typ | | 0.020 typ | |
| L | 0.406 | 1.270 | 0.016 | 0.050 |
| θ | 0° | 8° | 0° | 8° |
| JEDEC | MS-012 (AB) | | | |

△ * NOTES：1. 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.

2. DIMENSION "E1" DOES NOT INCLUDE INTER-LEAD FLASH, OR PROTRUSIONS. INTER-LEAD FLASH AND PROTRUSIONS SHALL NOT EXCEED 0.25 MM ( 0.010 INCH ) PER SIDE.