



十速

TM57PT16/PT16B/PA16

DATA SHEET

Rev 0.93

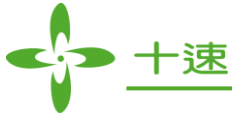
tenx reserves the right to change or discontinue the manual and online documentation to this product herein to improve reliability, function or design without further notice. **tenx** does not assume any liability arising out of the application or use of any product or circuit described herein; neither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

AMENDMENT HISTORY

| Version | Date | Description |
|---------|-----------|--|
| V0.90 | Jun, 2014 | New release |
| V0.91 | Jan, 2015 | Adding new product TM57PT16B and related description |
| V0.92 | Jan, 2016 | Ordering information update |
| V0.93 | Mar, 2018 | Adding DIP-8, SOP-8, MSOP-8 package type |

CONTENTS

| | |
|---|-----------|
| AMENDMENT HISTORY | 2 |
| CONTENTS..... | 3 |
| FEATURES | 5 |
| BLOCK DIAGRAM | 7 |
| PIN ASSIGNMENT | 8 |
| PIN DESCRIPTION | 9 |
| PIN SUMMARY..... | 10 |
| FUNCTIONAL DESCRIPTION | 11 |
| 1. CPU Core | 11 |
| 1.1 Clock Scheme and Instruction Cycle | 11 |
| 1.2 Addressing Mode | 12 |
| 1.3 Programming Counter (PC) and Stack..... | 14 |
| 1.4 ALU and Working (W) Register..... | 14 |
| 1.5 STATUS Register | 15 |
| 1.6 Interrupt..... | 16 |
| 2. Chip Operation Mode | 19 |
| 2.1 Reset..... | 19 |
| 2.2 External Reset Circuit | 20 |
| 2.3 System Configuration Register (SYSCFG) | 21 |
| 2.4 PROM Re-use ROM | 22 |
| 2.5 Power Down Mode..... | 23 |
| 3. Peripheral Functional Block | 24 |
| 3.1 Watchdog (WDT) / Wakeup (WKT) Timer..... | 24 |
| 3.2 Timer0: 8-bit Timer/Counter with Pre-scale (PSC)..... | 27 |
| 3.3 PWM0: 8-bit PWM..... | 31 |
| 3.4 Analog-to-Digital Converter | 34 |
| 3.5 Touch Key (TM57PT16/PT16B only)..... | 37 |
| 3.6 System Clock Oscillator..... | 44 |
| 4. I/O Port..... | 45 |
| 4.1 PA0-1 | 45 |
| 4.2 PA2-4 | 46 |
| 4.3 PA7..... | 47 |
| MEMORY MAP..... | 49 |
| F-Plane | 49 |
| R-Plane..... | 53 |



INSTRUCTION SET 55

ELECTRICAL CHARACTERISTICS 67

- 1. Absolute Maximum Ratings 67**
- 2. DC Characteristics 68**
- 3. Clock Timing 69**
- 4. Reset Timing Characteristics 69**
- 5. ADC Electrical Characteristics 69**
- 6. Characteristic Graphs..... 70**

ORDERING INFORMATION 73

- SOT23-6 Package Dimension 74**
- DIP-8 (300mil) Package Dimension 75**
- SOP-8 (150mil) Package Dimension 76**
- MSOP-8 (118mil) Package Dimension 77**

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 port: One Bit-programmable I/O port (Max. 6 pins)**
5. **Timer/Counter: One 8-bit timer/counter with divided by 1~256 pre-scale option**
6. **PWM: One 8-bit PWM0 with prescale/period-adjustment function**
7. **10-bit ADC with 5 channels input**
8. **4-channel Touch Key (TM57PT16/PT16B only)**
 - H/W auto scan (ATK), threshold adjustable for each key
 - Interrupt / Wake-up CPU while key pressed
 - Internal built-in reference capacitor (TM57PT16B only)
9. **Watchdog Timer**
 - Clocked by built-in RC oscillator with 4 adjustable Reset/Interrupt Time (176 ms/88 ms/44 ms/22 ms, @5V; 224 ms/112 ms/56 ms/28 ms, @3V)
 - Watchdog timer can be disabled/enabled in STOP mode
10. **Reset**
 - Power On Reset
 - Watchdog Reset
 - Low Voltage Reset
 - External pin Reset
11. **System Clock Mode**
 - Internal RC: 4/8 MHz.
 - When the IRC is 8 MHz, the LVR can only be set to 3.1V (cannot use 2.2V).
12. **2-Level Low Voltage Reset: 2.2V/3.1V (Can be disabled)**

| Freq | LVR | | |
|-------|------|------|---------|
| | 2.2V | 3.1V | Disable |
| 4 MHz | ☑ | ☑ | ☑ |
| 8 MHz | ☒ | ☑ | ☑ |

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

- F_{sys} = 4 MHz, 1.9V ~ 5.5V
- F_{sys} = 8 MHz, 2.3V ~ 5.5V

14. I/O Port

- CMOS Output
- Pseudo-Open-Drain or Open-Drain Output
- Schmitt Trigger Input with/without pull-up resistor

15. Instruction set: 36 Instructions
16. Interrupt

- Two External Interrupt pins:
 - One pin is falling edge triggered
 - One pin is rising or falling edge triggered
- TM0, Wake-up Timer, Auto Touch Key (TM57PT16/PT16B only) Interrupt

17. PA1~PA4 individual pin low level wake up
18. Power Saving Operation Modes

- Normal Mode: Internal RC keeps CPU running
- STOP Mode: All Clocks stop
- IDLE Mode: Internal RC and CPU stop, WKT or Auto Touch Key (TM57PT16/PT16B only) keep running

19. Support 5-wire program
20. Package Types:

- SOT23-6
- DIP-8 (300mil)
- SOP-8 (150mil)
- MSOP-8 (118mil)

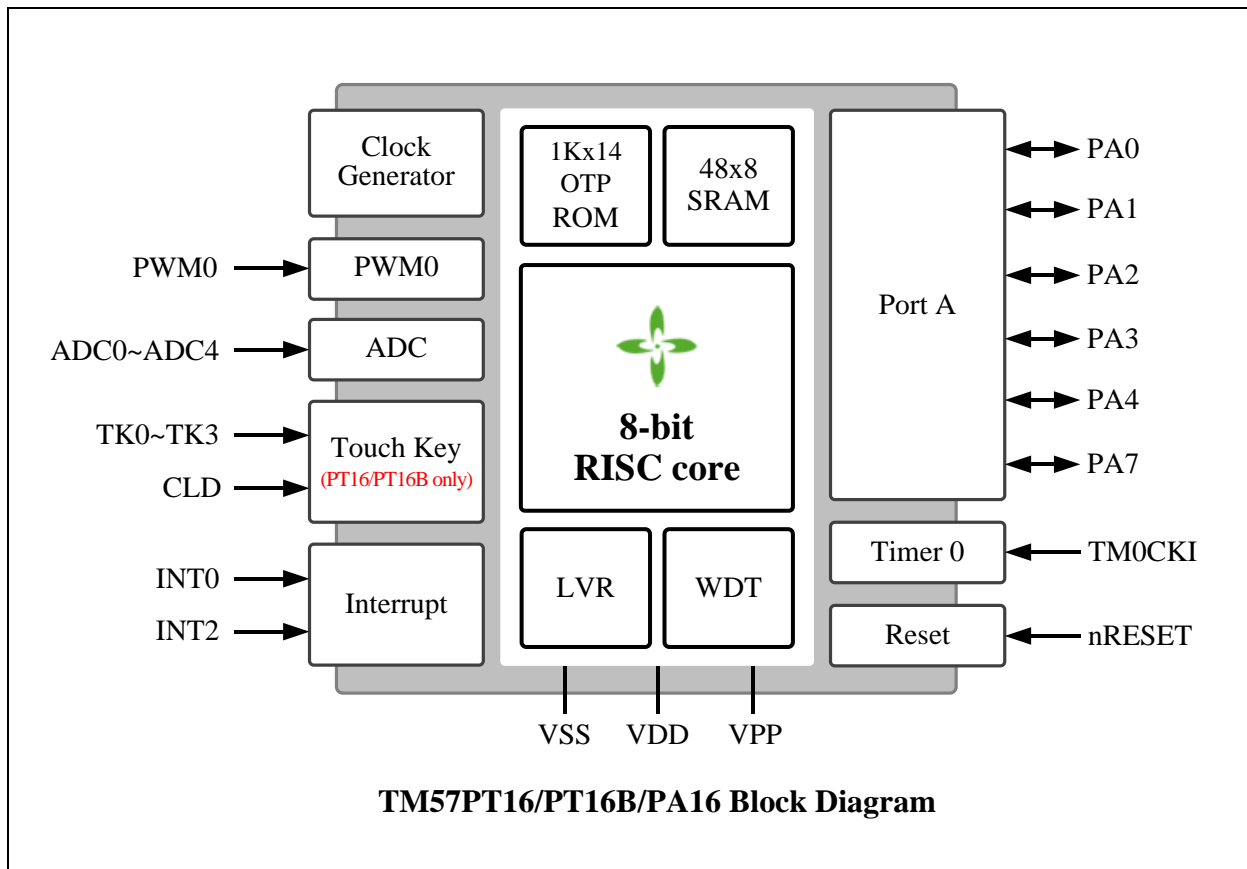
21. Supported EV Board on ICE

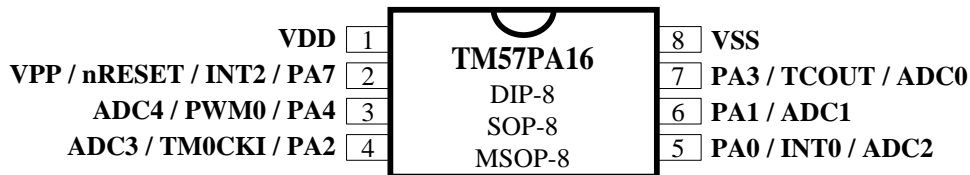
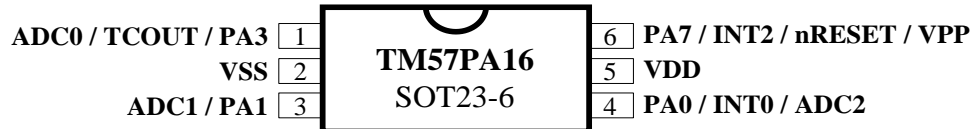
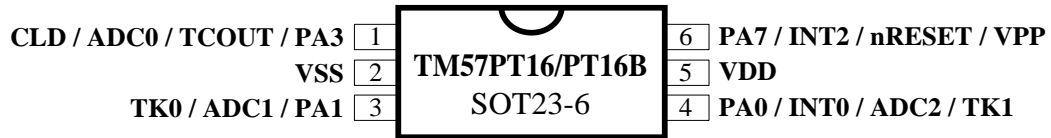
EV Board: EV8201

22. Comparison Table:

| | EV8201 | TM57PT16 | TM57PT16B |
|-----------------------|-----------------|-----------------|--------------------------------------|
| EV Board | – | EV8201 | EV8201 |
| Touch Key | 4-channel | 4-channel | 4-channel + Internal Reference |
| ATK | 4-channel | 4-channel | 4-channel (Must set TKCHS2=0) |
| SFR Difference | TKTMR (F14.6~4) | TKTMR (F14.6~4) | TKTMR (F14.6~5) TKCHS2 (F14.4) |

BLOCK DIAGRAM



PIN ASSIGNMENT


PIN DESCRIPTION

| Name | In/Out | Pin Description |
|------------|--------|---|
| PA0–PA1 | 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. |
| PA2–PA4 | I/O | Bit-programmable I/O port for Schmitt-trigger input, CMOS push-pull output or open-drain output. Pull-up resistors are assignable by software. |
| PA7 | I/O | Bit-programmable I/O port for Schmitt-trigger input or open-drain output. Pull-up resistors are assignable by software. |
| nRESET | I | External active low reset |
| TCOUT | O | Instruction cycle clock output. The instruction clock frequency is system clock frequency divided by two ($F_{sys}/2$) |
| VDD, VSS | P | Power Voltage input pin and ground |
| VPP | I | PROM programming high voltage input |
| INT0, INT2 | I | External interrupt input |
| TM0CKI | I | Timer0’s input in counter mode |
| NC | - | Not connected |
| PWM0 | O | PWM0 output |
| TK0–TK3 | I | Touch key input (TM57PT16/PT16B only) |
| CLD | I | Touch key capacitor input (TM57PT16/PT16B only) |
| ADC0–ADC4 | I | ADC channel input |

PIN SUMMARY

| Pin Number | | Pin Name | Type | GPIO | | | | | Function After Reset | Alternate Function | | | |
|----------------|---------|---------------------|------|--------------|----------------|--------|-------|-----|----------------------|--------------------|--------------------------------|-----|--------|
| 8-SOP/DIP/MSOP | SOT23-6 | | | Input | | Output | | | | PWM | Touch Key (PT16/PT16B only) | ADC | MISC |
| | | | | Weak Pull-up | Ext. Interrupt | O.D | P.O.D | P.P | | | | | |
| 1 | 5 | VDD | P | | | | | | | | | | |
| 2 | 6 | PA7/INT2/nRESET/VPP | I/O | ○ | ○ | ○ | | | PA7 | | | | nRESET |
| 3 | - | PA4/PWM0/ADC4/TK3 | I/O | ○ | | ○ | | ○ | PA4 | ○ | ○ | ○ | |
| 4 | - | PA2/TM0CKI/ADC3/TK2 | I/O | ○ | | ○ | | ○ | PA2 | | ○ | ○ | TM0CKI |
| 5 | 4 | PA0/INT0/ADC2/TK1 | I/O | ○ | ○ | | | ○ | PA0 | | ○ | ○ | |
| 6 | 3 | PA1/ADC1/TK0 | I/O | ○ | | | | ○ | PA1 | | ○ | ○ | |
| 7 | 1 | PA3/TCOUT/ADC0/CLD | I/O | ○ | | ○ | | ○ | PA3 | | ○ | ○ | TCOUT |
| 8 | 2 | VSS | P | | | | | | | | | | |

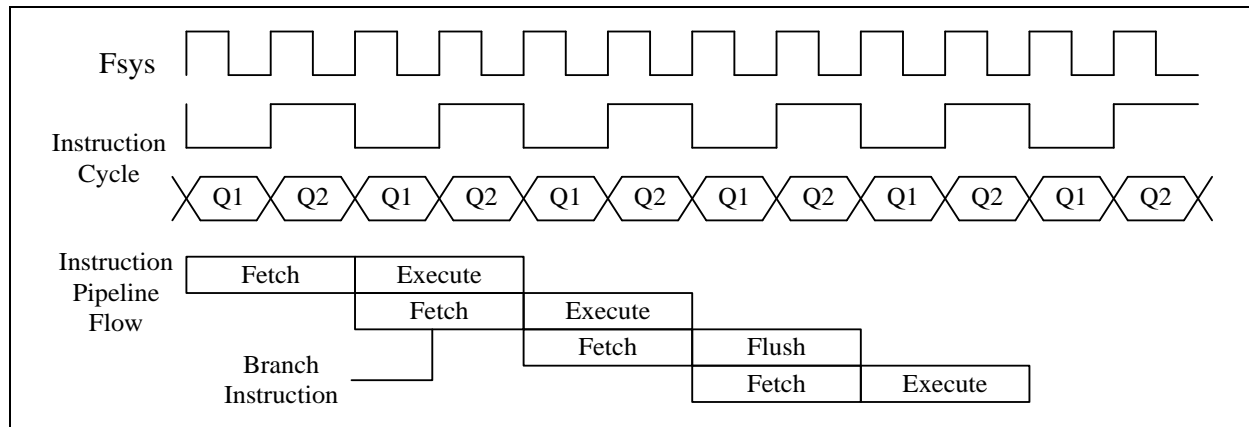
Symbol : P.P. = Push-Pull Output
P.O.D. = Pseudo Open Drain
O.D. = Open Drain

FUNCTIONAL DESCRIPTION

1. CPU Core

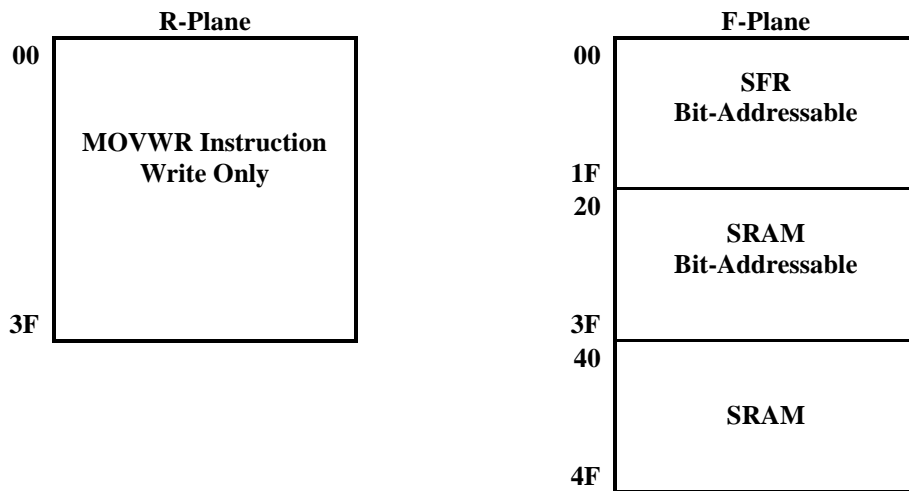
1.1 Clock Scheme and Instruction Cycle

The system clock 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 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
```

◇Example: Write immediate data into F-Plane register

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

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

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

◇Example: Clear SRAM Bank0 data by indirect addressing mode

```
MOVLW    20H                ; W = 20H (SRAM start address)
MOVWF    FSR                ; Set start address of user SRAM into FSR register

LOOP:
MOVLW    00H                ; Clear user SRAM data
MOVWF    INDF               ; Clear user SRAM data
INCF    FSR, 1              ; Increment the FSR for next address
MOVLW    50H                ; W = 50H (SRAM end address)
XORWF    FSR, 0             ; Check the FSR is end address of user SRAM?
BTFSS    STATUS, Z          ; Check the Z flag
GOTO     LOOP               ; If Z = 0, goto LOOP label
...                ; If Z = 1, exit LOOP
```

1.3 Programming Counter (PC) and Stack

The Programming Counter is 10-bit wide capable of addressing a 1K x 14 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 |
| R/W | – | – | – | R | R | R/W | R/W | R/W |
| Bit | Description | | | | | | | |
| 7~5 | Not Used | | | | | | | |
| 4 | TO: Time Out Flag 0: after Power On Reset, LVR Reset, or CLRWDT/SLEEP instruction 1: WDT time out occurs | | | | | | | |
| 3 | PD: STOP Flag 0: after Power On Reset, LVR Reset, or CLRWDT instruction 1: after SLEEP instruction | | | | | | | |
| 2 | Z: Zero Flag 0: the result of a logic operation is not zero 1: the result of a logic operation is zero | | | | | | | |
| 1 | DC: Decimal Carry Flag or Decimal /Borrow Flag | | | | | | | |
| | ADD instruction | | | | SUB instruction | | | |
| | 0: no carry 1: a carry from the low nibble bits of the result occurs | | | | 0: a borrow from the low nibble bits of the result occurs 1: no borrow | | | |
| 0 | C: Carry Flag or /Borrow Flag | | | | | | | |
| | ADD instruction | | | | SUB instruction | | | |
| | 0: no carry 1: a carry occurs from the MSB | | | | 0: a borrow occurs from the MSB 1: no borrow | | | |

◇Example: Write immediate data into STATUS register

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

◇Example: Bit addressing set and clear STATUS register

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

◇Example: Determine the C flag by BTFSS instruction

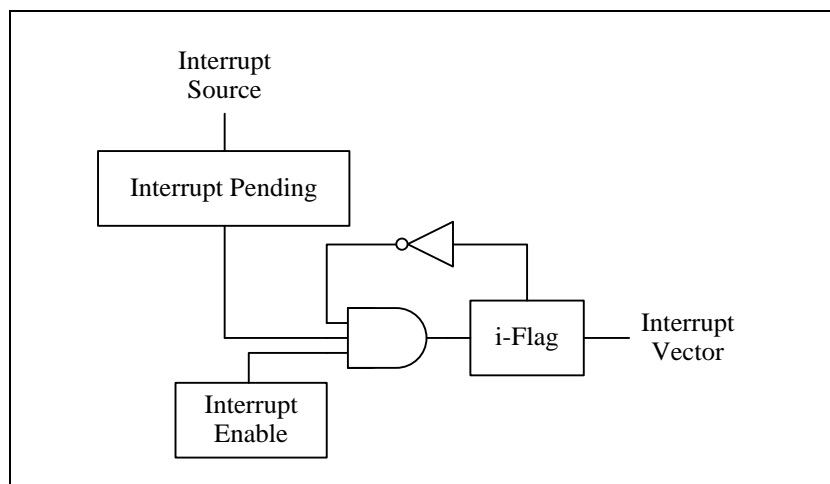
```
BTFSS    STATUS, 0       ; Check the C flag
GOTO     LABEL_1        ; If C = 0, goto LABEL_1 label
GOTO     LABEL_2        ; If C = 1, goto LABEL_2 label
```

1.6 Interrupt

The TM57PA16 has 1 level, 1 vector and 4 interrupt sources. TM57PT16/PT16B has 1 level, 1 vector and 5 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 TM57PT16/PT16B/PA16 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 INT0 (PA0) interrupt request with rising edge trigger

```

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

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

START:
    ORG      002H

    MOVLW   xxxxxxx0B
    MOVWR   PAE           ; Disable INT0 (PA0) CMOS push-pull output
                                ; mode

    MOVLW   xxxxxxx0B
    MOVWR   PAPUN        ; Enable INT0 (PA0) input pull-up resistor

    MOVLW   xxxxxxx1B
    MOVWF   PAD          ; Release INT0 (PA0), it becomes Schmitt-trigger
                                ; input mode with input pull-up resistor

    MOVLW   0001x0xxB
    MOVWR   R0B          ; Set INT0 interrupt trigger as rising edge

    MOVLW   1111110B
    MOVWF   INTIF        ; Clear INT0 interrupt request flag
    MOVLW   00000001B
    MOVWF   INTIE        ; Enable INT0 interrupt

MAIN:
    ...
    GOTO     MAIN

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

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

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

```

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

F08.5 **ATKIE:** Auto Touch Key interrupt enable (TM57PT16/PT16B only)

0: disable
1: enable

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

0: disable
1: enable

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

0: disable
1: enable

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

0: disable
1: enable

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

0: disable
1: enable

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

F09.5 **ATKIF:** Auto Touch Key interrupt event pending flag (TM57PT16/PT16B only)

This bit is set by H/W while ATK detected the key touched, write 0 to this bit will clear this flag

F09.4 **TM0IF:** Timer0 interrupt event pending flag

This bit is set by H/W while Timer0 overflows, write 0 to this bit will clear this flag

F09.3 **WKTIF:** Wakeup Timer interrupt event pending flag

This bit is set by H/W while Wakeup Timer is timeout, write 0 to this bit will clear this flag

F09.2 **INT2IF:** INT2 (PA7) pin falling interrupt pending flag

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

F09.0 **INT0IF:** INT0 (PA0) pin falling/rising interrupt pending flag

This bit is set by H/W at INT0 pin's falling/rising edge, write 0 to this bit will clear this flag

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

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

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

2. Chip Operation Mode

2.1 Reset

The TM57PT16/PT16B/PA16 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 V_{DD} is more than 3.3V, while another one is suitable for application with V_{DD} is less than 3.3V. See the following LVR Selection Table; user must also consider the lowest operating voltage of operating frequency.

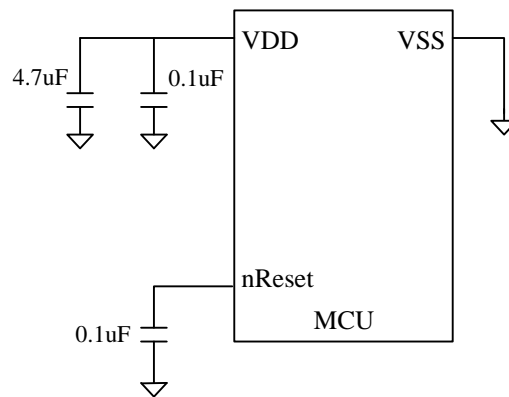
LVR Selection Table:

| LVR Threshold Level | Consider the operating voltage to choose LVR |
|---------------------|--|
| LVR3.1 | $5.5V > V_{DD} > 3.3V$ |
| LVR2.2 | 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 External Reset Circuit

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



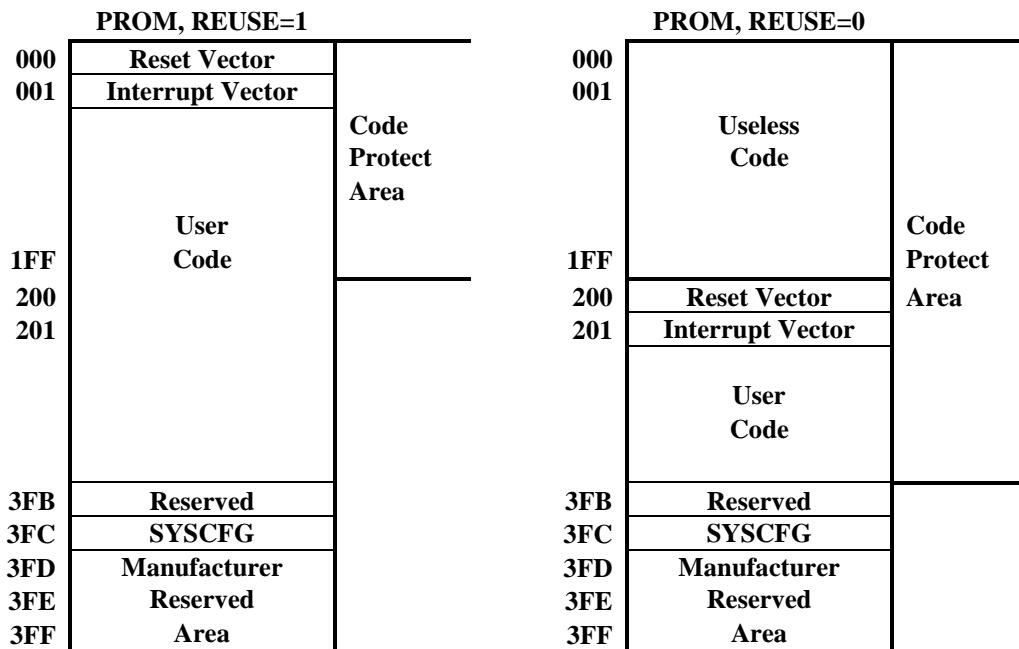
2.3 System Configuration Register (SYSCFG)

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 | 111111111111 | |
| Bit | Description | |
| 13 | PROTECT: Code protection selection | |
| | 0 | Enable |
| | 1 | Disable |
| 12 | REUSE: PROM Re-use control | |
| | 0 | Enable |
| | 1 | Disable |
| 11-10 | LVR: LV Reset Mode | |
| | 00 | LVR Disable |
| | 01 | LVR = 3.1V, always enable |
| | 10 | LVR = 2.2V, disable in STOP mode |
| | 11 | LVR = 2.2V, always enable |
| 9-8 | Reserved | |
| 7 | XRSTE: External pin Reset Enable | |
| | 0 | Disable, PA7 as IO pin |
| | 1 | Enable |
| 6 | WDTE: WDT Reset Enable | |
| | 0 | Disable |
| | 1 | Enable |
| 5 | IRC: 0: IRC=8 MHz, 1: IRC=4 MHz When the IRC is 8 MHz, the LVR can only be set to 3.1V or disable (can't use 2.2V). | |
| 4-0 | Reserved | |

2.4 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 PROTECT is enabled and not Re-use, 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 "REUSE" control bit to "0". In the mean while, the Code protection area becomes the whole PROM except the Reserved Area.



2.5 Power Down Mode

The Power Down mode includes STOP mode and IDLE mode. It is activated by SLEEP instruction.

STOP Mode:

When WKTIE (F08.3) is cleared, all blocks will be turned off and the TM57PT16/PT16B/PA16 will enter the “STOP Mode” after executing the SLEEP instruction. During the STOP mode, the system clock and peripherals stop to minimize power consumption. The STOP mode can be terminated by Reset, or enabled External Pins Interrupt or PA1-4 pins low level wake up.

IDLE Mode:

When WKTIE (F08.3) is set, the TM57PT16/PT16B/PA16 will enter the “IDLE Mode” after executing the SLEEP instruction. TM57PT16/PT16B has another way to enter the IDLE mode by setting TKAUTO (F15.2) before executing the SLEEP instruction. During the IDLE mode, the system clock and peripherals stops to minimize power consumption, but WKT/WDT and auto touch key (**TM57PT16/PT16B only**) can still run. The IDLE mode can be terminated by Reset, or enabled Interrupts (External pins, WKT and ATK interrupts) or PA1-4 pins low level wake up. (More details for Touch Key setting is in chapter 3.5)

| R03 | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|-------|-------|-------|-------|-------|-------|-------|-------|-------|
| PWRDN | PWRDN | | | | | | | |
| R/W | W | | | | | | | |
| Reset | – | – | – | – | – | – | – | – |

R03.7~0 **PWRDN**: Write this register to enter Power Down Mode

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

F08.3 **WKTIE**: Wakeup Timer interrupt enable
 0: disable
 1: enable

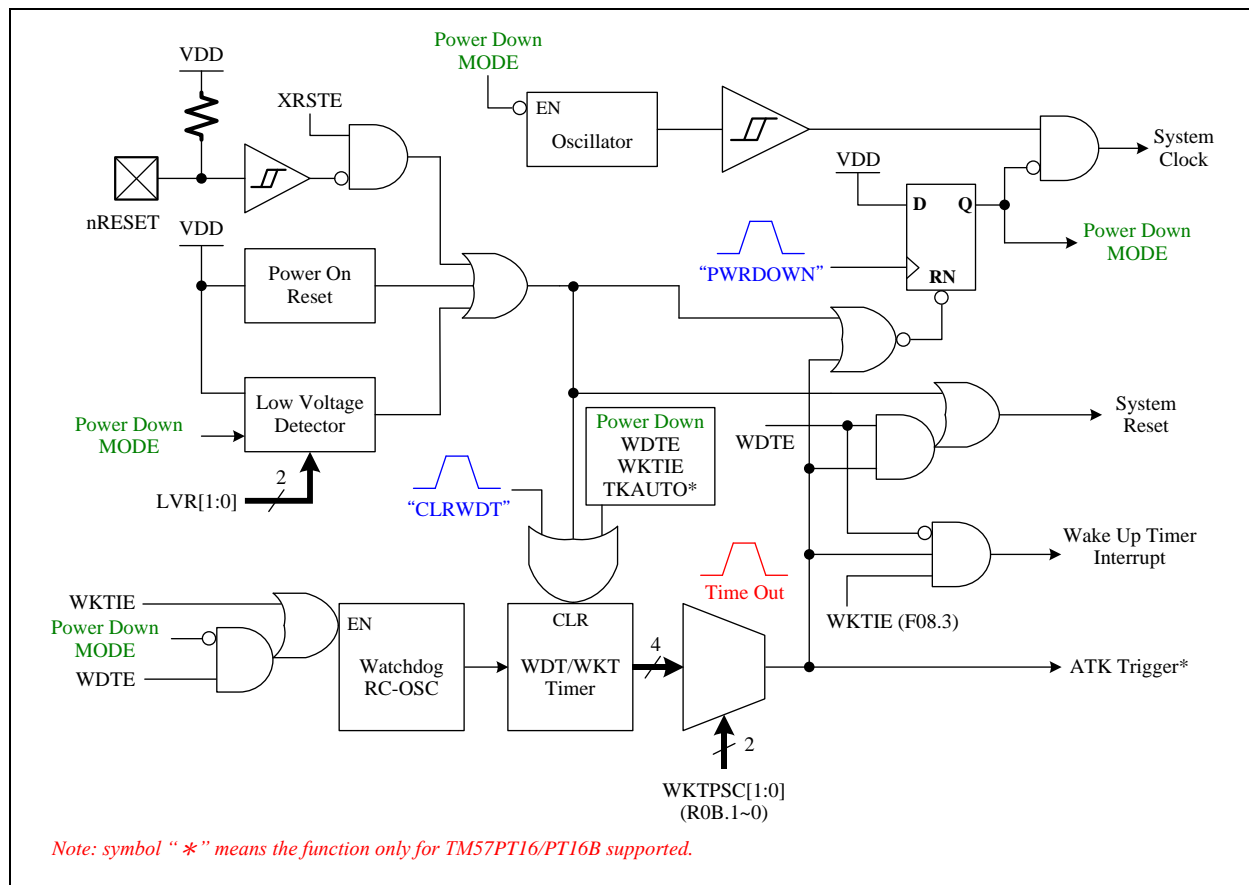
| F15 | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|--------|-------|-------|-------|-------|-------|--------|---------|-------|
| ATKCTL | – | – | – | – | – | TKAUTO | TKSCNUM | |
| R/W | – | – | – | – | – | R/W | R/W | |
| Reset | – | – | – | – | – | 0 | 0 | 0 |

F15.2 **TKAUTO**: Touch key auto scan mode enable (**TM57PT16/PT16B only**)
 0: disable H/W Auto Mode
 1: enable H/W Auto Mode

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 22 ms to 224 ms. The WDT/WKT is cleared by the CLRWDT instruction. If the Watchdog Reset is enabled (SYSCFG[6], WDTE=1), the WDT generates the chip reset signal, otherwise, the WKT only generates overflow time out interrupt. The WDT/WKT works in normal mode and IDLE mode. User can further choose to enable or disable the WDT/WKT by "WKTIE" (F08.3) to enter IDLE or STOP mode. If WKTIE is cleared (no matter WDTE is 1 or 0), the internal RC Timer stops for power saving. In other words, user keeps the WDT/WKT alive in IDLE Mode by setting WKTIE to "1". If the WDTE is set and WKTIE is cleared, WDT/WKT timer will be cleared and stopped for power saving in STOP mode. If the WDTE and WKTIE are set, WDT/WKT timer keeps counting in IDLE/normal mode. Refer to the following table and figure.



If the user program needs the MCU totally shut down for power conservation in STOP mode, the below setting of control bits should be followed.

| Mode | | WDTE | WKTIE | TKAUTO (PT16/PT16B only) | Watchdog RC Oscillator |
|-----------------|-----------|------|-------|-----------------------------|------------------------|
| Normal Mode | | 0 | 0 | 0 | Stop |
| | | 0 | 1 | | Run |
| | | 1 | 0 | | |
| | | 1 | 1 | | |
| | | x | x | 1 | Run |
| Power Down Mode | IDLE Mode | x | x | 1 | Run |
| | | x | 1 | x | |
| | STOP Mode | x | 0 | 0 | Stop |

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

F03.4 **TO:** WDT time out flag, read-only
 0: after Power On Reset, LVR Reset, or CLRWDT/SLEEP instructions
 1: WDT time out occurs

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

F08.3 **WKTIE:** Wakeup Timer interrupt enable
 0: disable
 1: enable

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

F09.3 **WKTIF:** Wakeup Timer interrupt event pending flag
 This bit is set by H/W while Wakeup Timer is timeout, write 0 to this bit will clear this flag

| F15 | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|--------|-------|-------|-------|-------|-------|--------|---------|-------|
| ATKCTL | – | – | – | – | – | TKAUTO | TKSCNUM | |
| R/W | – | – | – | – | – | R/W | R/W | |
| Reset | – | – | – | – | – | 0 | 0 | 0 |

F15.2 **TKAUTO:** Touch key auto scan mode enable (TM57PT16/PT16B only)
 0: disable H/W Auto Mode
 1: enable H/W Auto Mode

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

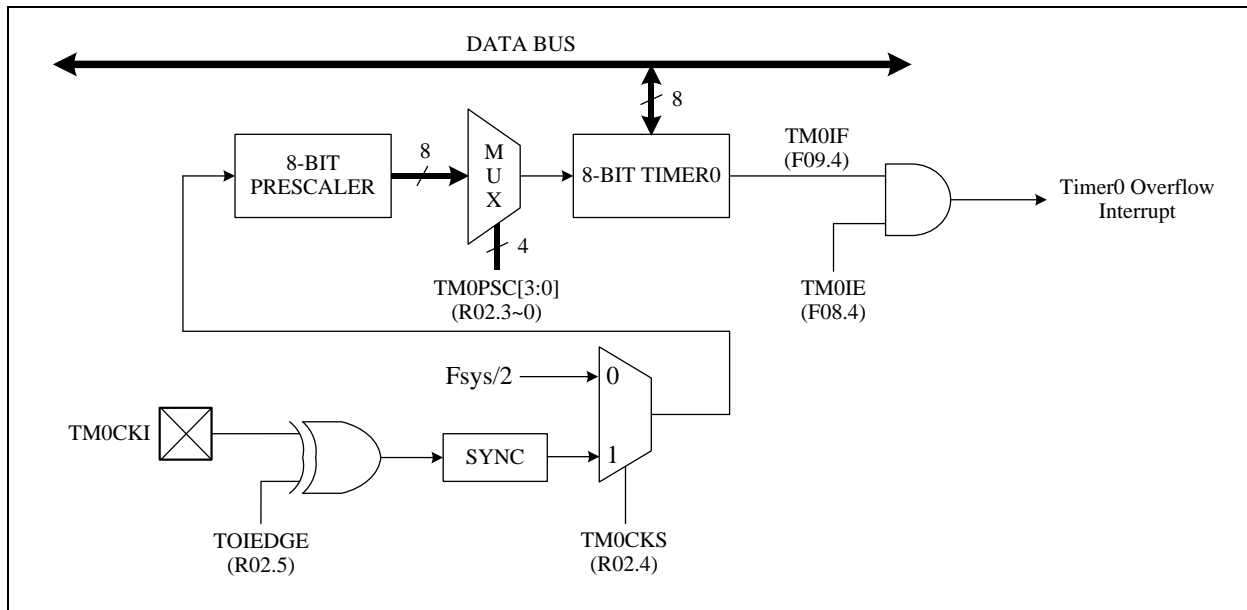
R04.7~0 **WDTCLR**: Write this register to clear WDT/WKT

| R0B | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|-------|-------|-------|-------|---------|-------|-------|--------|-------|
| MROB | – | – | – | INT0EDG | TCOE | – | WKTpsc | |
| R/W | – | – | – | W | W | – | W | |
| Reset | – | – | – | 0 | 0 | – | 1 | 1 |

R0B.1~0 **WKTpsc**: WDT/WKT pre-scale option select
 00: WDT/WKT period is 22 ms, @5V; 28 ms, @3V
 01: WDT/WKT period is 44 ms, @5V; 56 ms, @3V
 10: WDT/WKT period is 88 ms, @5V; 112 ms, @3V
 11: WDT/WKT period is 176 ms, @5V; 224 ms, @3V

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

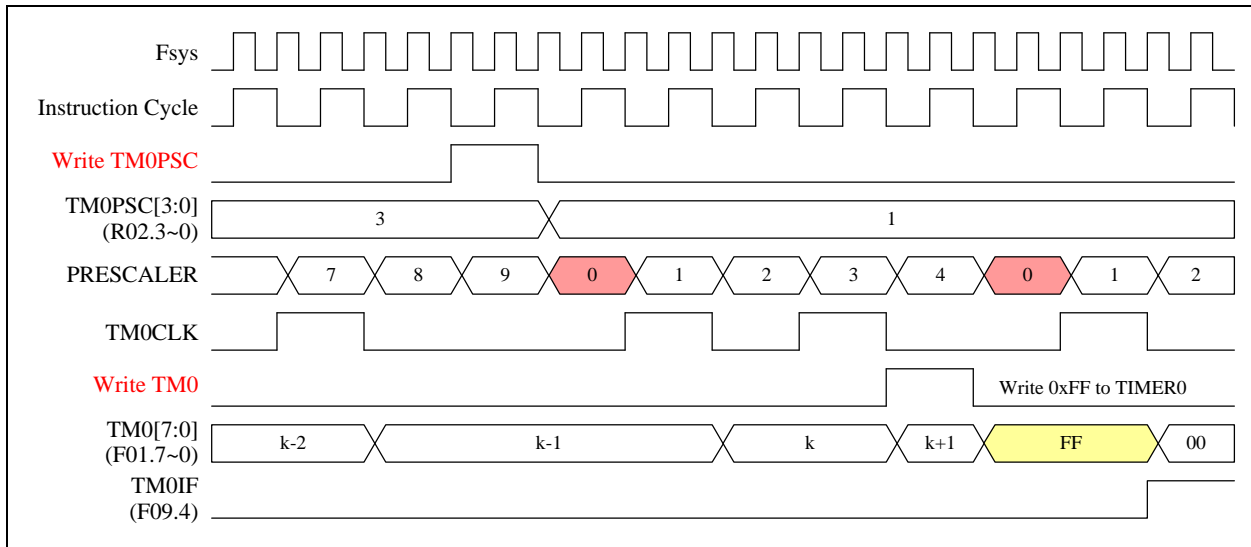
The Timer0 is an 8-bit wide register of F-Plane. It can be read or written as any other register of F-Plane. Besides, Timer0 increases itself periodically and automatically rolls over based on the pre-scaled clock source, which can be the instruction cycle or TMOCKI (PA2) rising/falling input. The Timer0's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



Timer0 Block Diagram

Timer Mode:

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



Timer0 works in Timer mode (TM0CKS = 0)

The equation of Timer0 interrupt timer value is as following:

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

◇Example: Setup Timer0 work in Timer mode, Fsys = IRC Clock = 4MHz

; Setup Timer0 clock source and divider

```

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

; Enable Timer0 and interrupt function

```

MOV LW    11101111B
MOV WRF  INTIF        ; Clear Timer0 request interrupt flag
BSF      TM0IE        ; Enable Timer0 interrupt function
    
```

; Setup Timer0

```

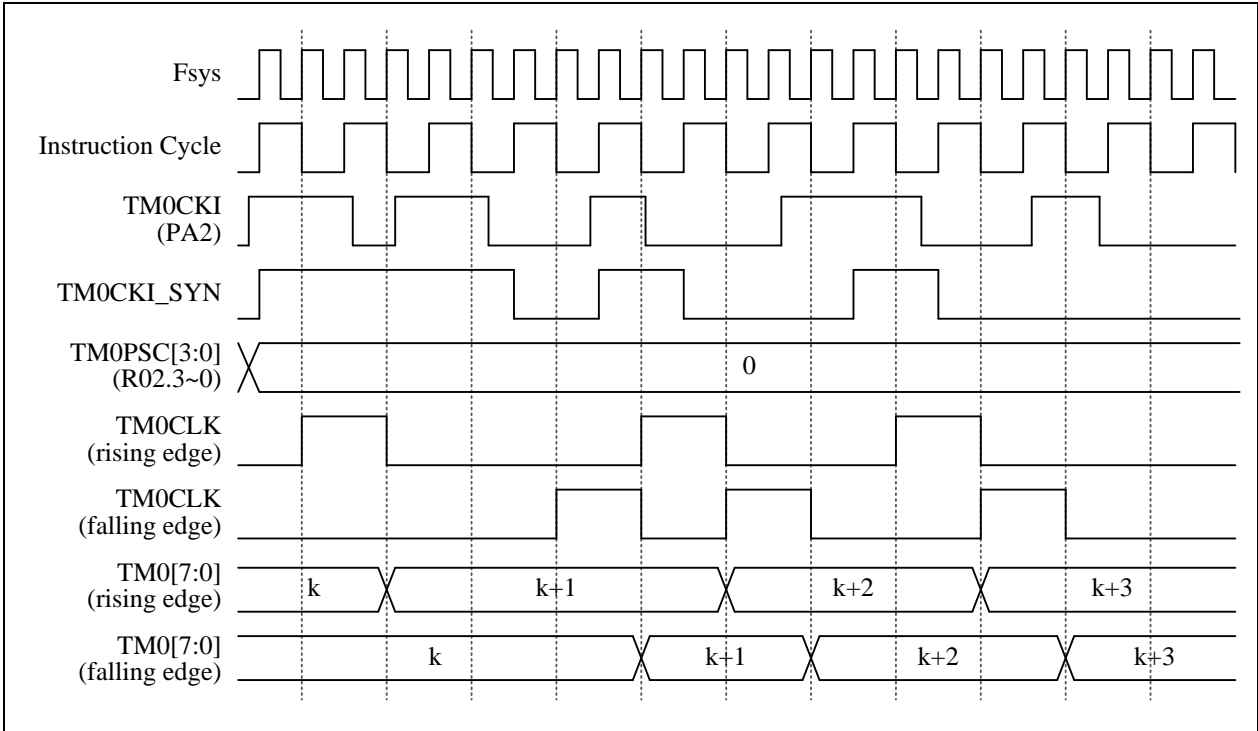
CLR F     TM0          ; Clear Timer0 content
    
```

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

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

Counter Mode:

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



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

◇Example: Setup Timer0 works in Counter mode

```

; Setup Timer0 clock source and divider
MOV LW    00110000B    ; TM0EDG = 1, counting edge is falling edge
MOV WREG TM0CTL       ; TM0CKS = 1, Timer0 clock is TM0CKI (PA2)
                                ; TM0PSC = 0000b, divided by 1

; Setup Timer0
CLR F      TM0         ; Clear Timer0 content
...

; Read Timer0 counter
MOV F     TM0         ; Read Timer0 content
    
```

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

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

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

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

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

F09.4 **TM0IF**: Timer0 interrupt event pending flag
 This bit is set by H/W while Timer0 overflows, write 0 to this bit will clear this flag

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

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

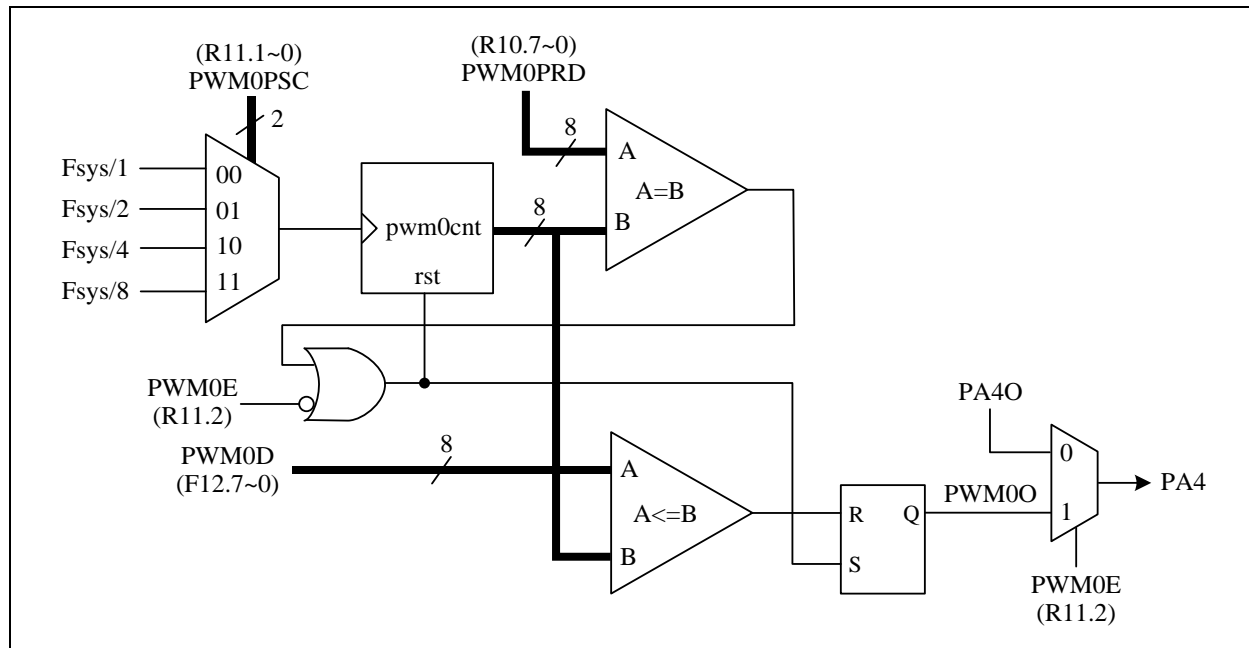
R02.4 **TM0CKS**: Timer0 Prescaler clock select
 0: Instruction Cycle as Timer0 Prescaler clock
 1: TM0CKI (PA2) as Timer0 Prescaler clock

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

3.3 PWM0: 8-bit PWM

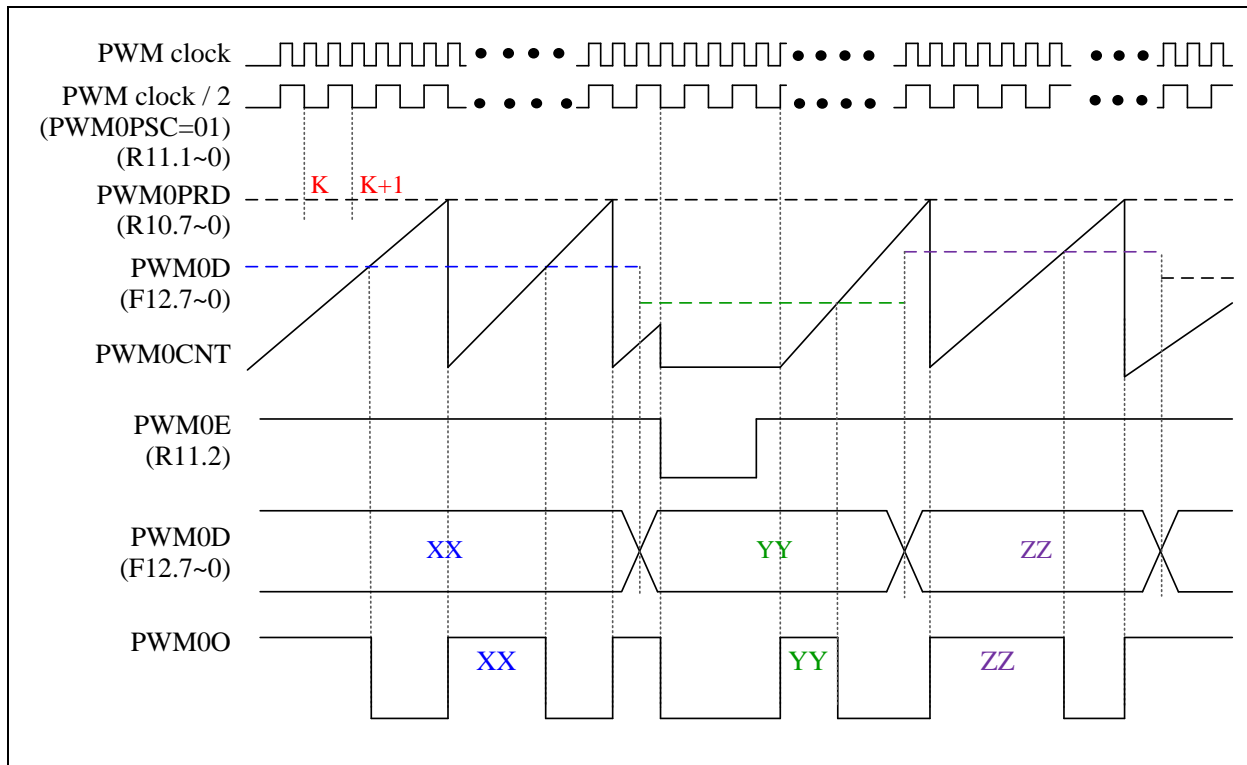
The chip has a built-in 8-bit PWM generator. The source clock comes from Fsys divided by 1, 2, 4, and 8 according to PWM0PSC (R11.1~0). The PWM0 duty cycle can be changed with writing to PWM0D (F12.7~0).

The PWM0 will be output to PA4 if PWM0E (R11.2) is set. With I/O mode setting, the PWM0 output can be set as CMOS push-pull or open-drain output mode. When PAE[4] (R05.4) is set, the output is CMOS push-pull output mode, otherwise is open-drain output mode. Figure shows the block diagram of PWM0.



PWM0 Block Diagram

Figure shows the PWM0 waveforms. When PWM0E bit is cleared or PWM0D equals to zero, the PWM0 output is cleared to '0' no matter what its current status is. Once the PWM0E bit is set and PWM0D is not zero, the PWM0 output is set to '1' to begin a new PWM cycle. PWM0 output will be '0' when PWM0CNT is greater than or equals to PWM0D. PWM0CNT keeps counting up when equals to PWM0PRD (R10.7~0), the PWM0 output is set to '1' again.



PWM0 Block Diagram

◇Example: PWM0 output and PWM0 clock is divided by 2, F_{sys} = 4 MHz

; Setup PWM0 period and duty

```

MOV LW    FFH
MOV WREG  PWM0PRD    ; Set PWM0 period = FFH + 1 = 256
    
```

```

MOV LW    80H
MOV WREG  PWM0D      ; Set PWM0 duty = 80H = 128
    
```

; Setup PWM0 prescaler and output enable

```

MOV LW    00000101B    ; PWM0E = 1, PWM0 output to PA4 pin
MOV WREG  PWM0CTL      ; PWM0PSC = 01b, divided by 2
    
```

$$\text{PWM0 output duty} = \text{PWM0D} / (\text{PWM0PRD} + 1) = 128 / (255 + 1) = 1 / 2$$

$$\text{PWM clock} = F_{\text{sys}} = 4 \text{ MHz, PWM clock divided by 2}$$

$$\text{PWM0 output frequency} = 4 \text{ MHz} / 2 / (255 + 1) = 7812.5 \text{ Hz}$$

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

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

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

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

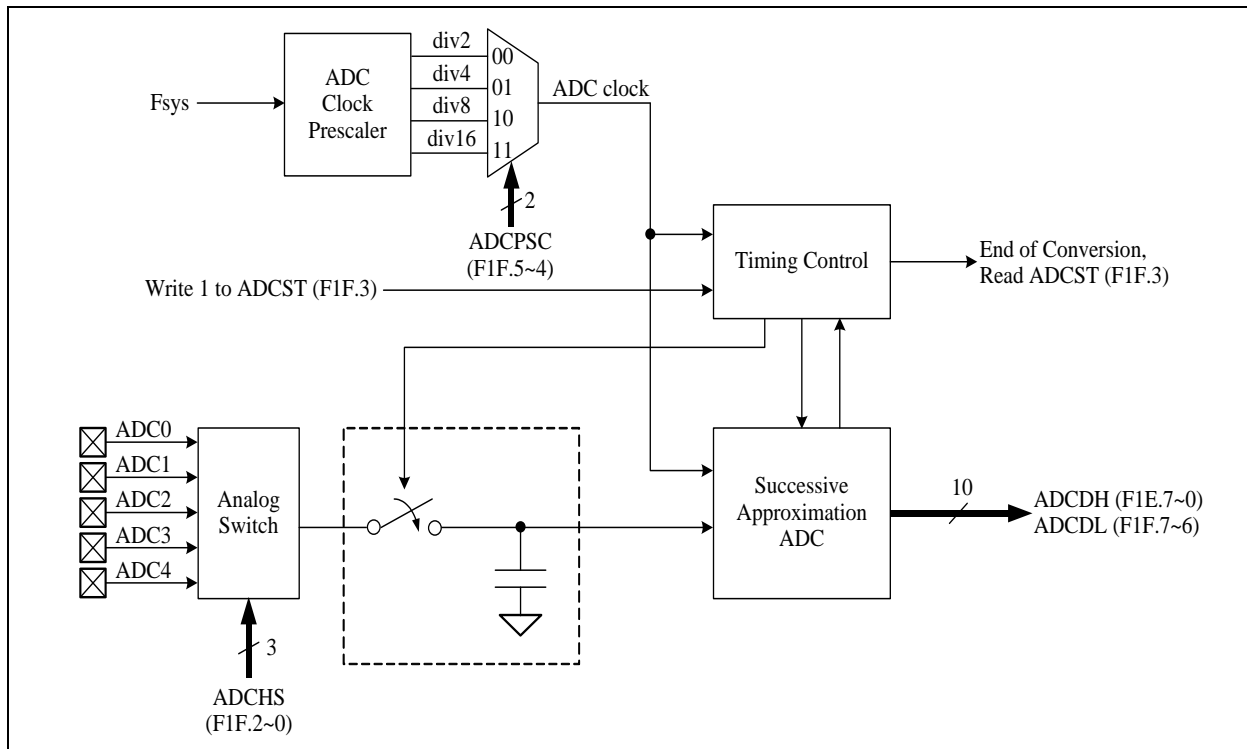
| R11 | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|--------|-------|-------|-------|-------|-------|-------|---------|-------|
| PWMCTL | – | – | – | – | – | PWM0E | PWM0PSC | |
| R/W | – | – | – | – | – | W | W | |
| Reset | – | – | – | – | – | 0 | 0 | 0 |

R11.2 **PWM0E**: PWM0 positive output to PA4 pin
 0: disable
 1: enable

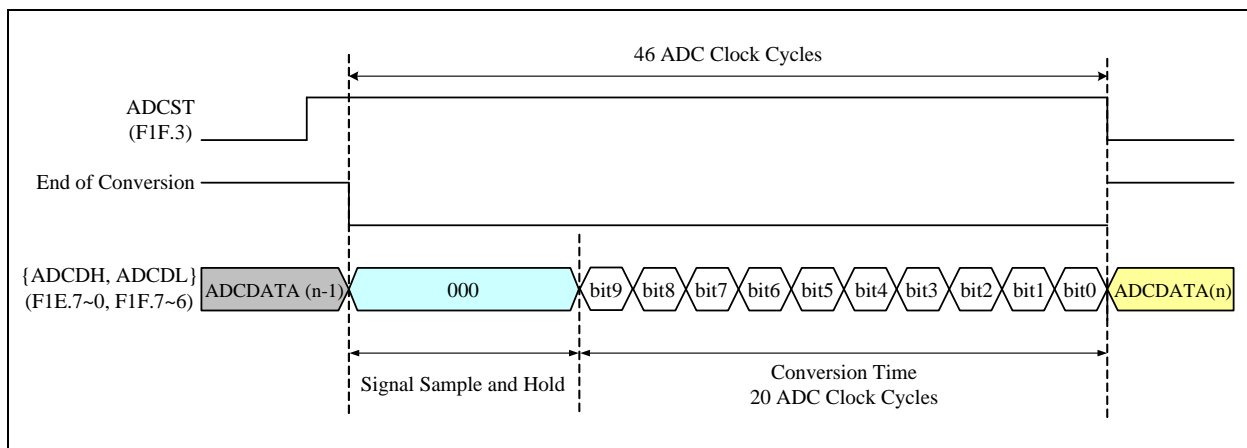
R11.1~0 **PWM0PSC**: PWM0 prescaler, PWM0 clock source
 00: divided by 1
 01: divided by 2
 10: divided by 4
 11: divided by 8

3.4 Analog-to-Digital Converter

The 10-bit ADC (Analog to Digital Converter) consists of a 5-channel analog input multiplexer, control register, clock generator, 10-bit successive approximation register, and output data register. To use the ADC, user needs to set ADCPSC (F1F.5~4) to choose a proper ADC clock frequency, which must be less than 2 MHz. User then launches the ADC conversion by setting the ADCST (F1F.3) control bit. After end of conversion, H/W automatic clears the ADCST (F1F.3) bit. User can poll this bit to know the conversion status. The PAIE (R14.4~0) 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.



ADC Block Diagram



◇Example: ADC clock frequency = 1MHz, ADC channel = ADC2 (PA0), Fsys = 4 MHz

```

MOV LW  xxxxxxxx1B      ; Set PA0 = 1
MOV W  PAD

MOV LW  xxxxxxxx1B      ; ADC2 (PA0) pull-high disable
MOV W  PAPUN

MOV LW  xxxxxxxx0B      ; Select PA0 input type as analog input
MOV W  PAIE

MOV LW  xxxxxxxx0B      ; PA0 CMOS push-pull output disable
MOV W  PAE

MOV LW  00010010B      ; ADCPSC = 01B, ADC clock = Fsys/4 = 1MHz
MOV W  ADCTL             ; ADCHS = 010B, select channel ADC2

BSF    ADCTL, 3          ; ADST = 1, ADC start conversion.

```

Conversing:

```

BTFS   ADCTL, 3          ; Polling ADCST
GOTO   Conversing       ; ADCST = 0, when ADC ended of conversion

MOV FW ADCDH
MOV W  20H               ; Store ADC MSB data into address 20H

MOV FW ADCTL
MOV W  21H               ; Store ADC LSB data into address 21H
:
:
:

```

| F1E | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|-------|-------|-------|-------|-------|-------|-------|-------|-------|
| ADCDH | ADCDH | | | | | | | |
| R/W | R | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

F1E.7~0 **ADCDH**: ADC output data MSB[9:2]

| F1F | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|-------|-------|-------|--------|-------|-------|-------|-------|-------|
| ADCTL | ADCDL | | ADCPSC | | ADCST | ADCHS | | |
| R/W | R | | R/W | | R/W | R/W | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

F1F.7~6 **ADCDL**: ADC output data LSB[1:0]

F1F.5~4 **ADCPSC**: ADC clock source prescaler. ADC clock source (Fsys)

- 00: divided by 2
- 01: divided by 4
- 10: divided by 8
- 11: divided by 16

F1A.3 **ADCST**: ADC start bit
 0: H/W clear after end of conversion
 1: ADC start conversion

F1A.2~0 **ADCHS**: ADC channel select

- 000: ADC0 (PA3)
- 001: ADC1 (PA1)
- 010: ADC2 (PA0)
- 011: ADC3 (PA2)
- 100: ADC4 (PA4)

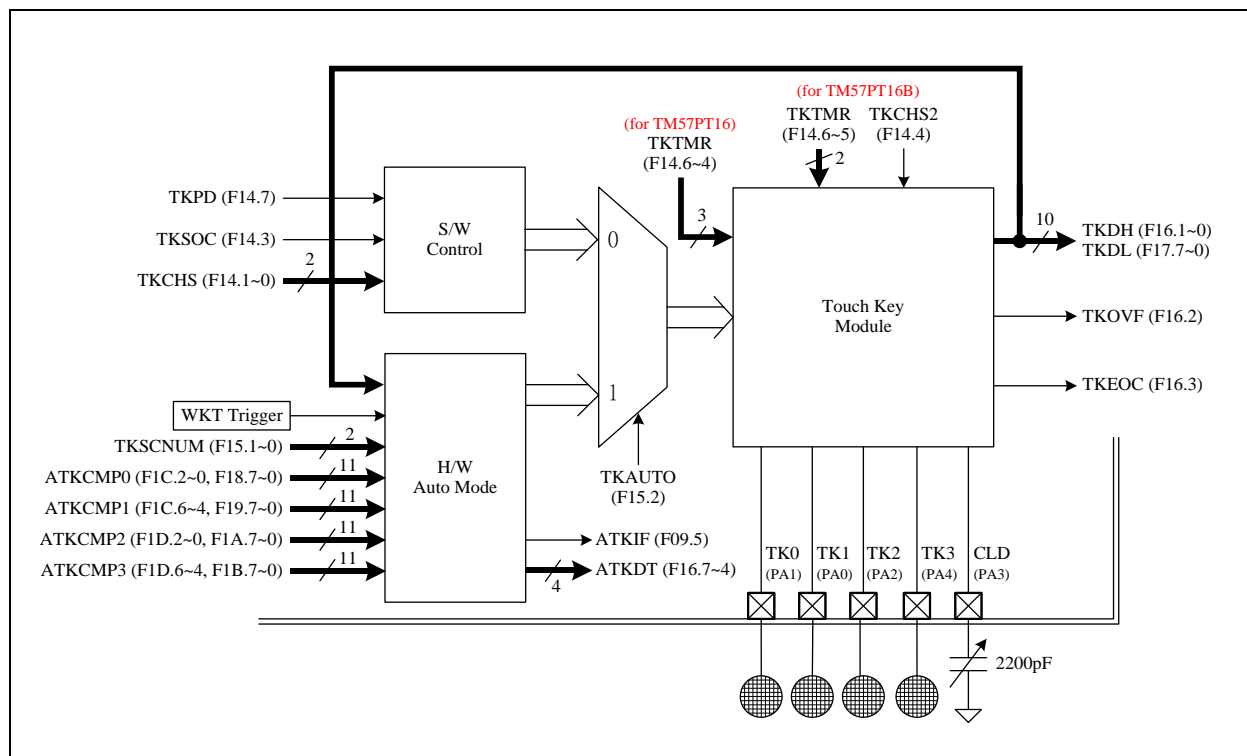
| R14 | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|-------|-------|-------|-------|-------|-------|-------|-------|-------|
| PAIE | – | – | – | PAIE | | | | |
| R/W | – | – | – | W | | | | |
| Reset | – | – | – | 1 | 1 | 1 | 1 | 1 |

R14.4~0 **PAIE**: PA4~0 input type select
 0: analog input
 1: digital input

3.5 Touch Key (TM57PT16/PT16B only)

Only TM57PT16/PT16B supported the Touch Key function. The Touch Key offers an easy, simple and reliable method to implement finger touch detection. For most applications, it only requires an external capacitor component on CLD pin. The device support 4 channels touch key detection with S/W manual mode and H/W auto (ATK) mode. Only one mode can be active at a time.

With I/O mode setting, The PAIE (R14.4~0) control registers are used for touch key pin configuration, user can write the corresponding bit to “0” when the pin is used as touch key input. The setting can disable the pin logical input path to save power consumption.



Touch Key Block Diagram

S/W Manual Mode Touch Key Detection

All touch key (TK0~TK3) can be used for S/W mode, it can be select by TKCHS (F14.1~0) bits. To start the S/W mode, TKAUTO (F15.2) and TKPD (F14.7) have to be cleared. After setting the TKSOC (F14.3) bit, the touch key starts conversion. The TKSOC bit can be automatically cleared while end of conversion. However, if the system clock is too slow, H/W might lose the auto clear TKSOC capability. “TKEOC=0” means conversion is in process, while “TKEOC=1” means the conversion is finish. After TKEOC’s rising edge, user must wait at least 10 us for next conversion. The touch key counting values is stored into the 10 bits touch key data count “TKDH (F16.1~0), TKDL (F17.7~0)”. If TKOVF (F16.2) is set, it means the conversion transaction exceeds period time. Reduce/Increase TKTMR (TM57PT16: F14.6~4; TM57PT16B: F14.6~5), can reduce/increase touch key data count to adapt the system board circumstances.

In TM57PT16B, The Touch Key unit has an internal built-in reference capacitor to simulate the Key behavior. Set {TKCHS2(F14.4), TKCHS}=111B and start the Touch Key can get the TKDATA of this reference capacitor. Since the internal capacitor never affected by water or mobile phone, it is useful for comparing the environment background noise.

H/W Auto Mode Touch Key (ATK) Detection

TK0~TK3 are all eligible for H/W auto mode by setting TKSCNUM (F15.1~0). This function can work in normal/IDLE mode and save the S/W effort as well as minimize the chip current consumption. To use this function, TKAUTO has to be set. The WKT will generate an overflow flag after WKT time out to trigger the touch key H/W auto mode starting. That can enable H/W control the touch key module fully. H/W then automatically detects the TK0~TK3's touch key data count at every 22 ms or 224 ms rate by WKTPSC (R0B.1~0). If those keys' data count are less than each pre-set compare thresholds ATKCMP0 (F1C.2~0, F18.7~0), ATKCMP1 (F1C.6~4, F19.7~0), ATKCMP2 (F1D.2~0, F1A.7~0) or ATKCMP3 (F1D.6~4, F1B.7~0), H/W will record the compare result in the ATKDT (F16.7~4). If ATKDT bits are not equal to "0", H/W will also generate interrupt flag ATKIF (F9.5) after touch key module ends conversion. It generates auto touch key interrupt and wake up CPU if the ATKIE (F8.5) bit is set. User can switch the touch key module to S/W manual mode after the touch key interrupt and identify/confirm the Key touch event. About I/O setting, the ATK will control the TK channel's PAPUN and PAIE automatically when this channel is being scanned. To use ATK in TM57PT16B, the TKCHS2 has to be cleared. Otherwise the wrong channel will be scanned. It will cause the function is invalid.

◇Example: S/W Mode, Touch key channel = TK1 (PA0).

```

MOVLW    xxxx1xx1B    ; Set PA0=1, PA3=1
MOVWF    PAD

MOVLW    xxxxx1xx1B    ; TK1 (PA0), CLD (PA3) pull-high disable
MOVWF    PAPUN

MOVLW    xxxxx0xx0B    ; Set PA0 as analog input for touch key input
MOVWF    PAIE          ; Set PA3 as analog input for connecting capacitor

MOVLW    xxxxx0xx0B    ; PA0, PA3 CMOS push-pull output disable
MOVWF    PAE

MOVLW    1100x01B
MOVWF    TKCTL          ; TKTMR=Level 4, TKCHS=01B (TK1)
BCF     TKCTL,7        ; TKPD=0
BSF     TKCTL,3        ; TKSOC=1, touch key start conversion
    
```

WAIT_TK:

```

BTSS    ATKDT,3        ; Polling TKEOC
GOTO    WAIT_TK        ; Waiting touch key conversion finish

MOVWF   TKDT           ; Read TKDH[1:0]
MOVWF   23H            ; Store W data to SRAM 23H

MOVWF   TKDL           ; Read TKDL[7:0]
MOVWF   24H            ; Store W data to SRAM 24H
    
```

◇Example: H/W Auto Mode, Touch key auto scan number = 4

```

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

ORG      001H          ; All interrupt vector
GOTO     INT

START:
ORG      002H

MOVLW   xxx1111B      ; Set PA0=1, PA1=1, PA2=1, PA3=1, PA4=1
MOVWF   PAD

MOVLW   xxxx1xxxB    ; Set PA3 pull-high disable
MOVWF   PAPUN        ; ATK will control the TK channels automatically

MOVLW   xxxx0xxxB    ; Set PA3 as analog input for touch key input
MOVWF   PAIE        ; ATK will control the TK channels automatically

MOVLW   xxx00000B    ; Set PA0~4 CMOS push-pull output disable
MOVWF   PAE

MOVLW   1 100xxxB    ; ATK will control the TKPD automatically
MOVWF   TKCTL        ; TKTMR= Level 4
                        ; Must set TKCHS2=0 in TM57PT16B
MOVLW   xxxx0 11B    ; TKSCNUM=11B (TK auto scan number=4)
MOVWF   ATKCTL

MOVLW   00H          ; Set auto TK compare thresholds
MOVWF   ATKCMP10H
MOVLW   01H
MOVWF   ATKCMP32H

MOVLW   C8H
MOVWF   ATKCMP0      ;Set ATKCMP0=200

MOVLW   C4H
MOVWF   ATKCMP1      ;Set ATKCMP1=196

MOVLW   0EH
MOVWF   ATKCMP2      ;Set ATKCMP2=270

MOVLW   BCH
MOVWF   ATKCMP3      ;Set ATKCMP3=188

MOVLW   0000011B
MOVWF   R0B          ; WKTPSC=11B, WKT period = 176ms,@5V

MOVLW   11011111B
MOVWF   INTIF        ; Clear ATK interrupt request flag

```

```

MOV LW    00100000B
MOV W    INTIE           ; Enable ATK interrupt

MAIN:
BSF      TKAUTO         ; TKAUTO=1, enable TK H/W auto Mode
(SLEEP)
:         ; Set system into IDLE mode
:         ; to reduce power consumption (dispensable)
:
GOTO     MAIN

INT:
MOV W    20H           ; Store W data to SRAM 20H
MOV F    STATUS       ; Get STATUS data
MOV W    21H           ; Store STATUS data to SRAM 21H

BTFSC   ATKIF         ; Check ATKIF bit
CALL    INT_ATK       ; ATKIF=0, exit interrupt subroutine
                        ; ATKIF interrupt service routine

...
GOTO    EXIT_INT

INT_ATK:
MOV LW   11011111B
MOV W    INTIF        ; Clear ATK interrupt request flag
MOV F    TKDT
MOV W    22H         ; Store ATK scan result to SRAM 22H
RET

EXIT_INT:
MOV F    21H         ; Get SRAM 21H data
MOV W    STATUS     ; Restore STATUS data
MOV F    20H         ; Restore W data
RETI      ; Return from interrupt

```

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

F08.5 **ATKIE:** Auto touch key interrupt enable
0: disable
1: enable

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

F09.5 **ATKIF:** Auto touch key interrupt event pending flag
This bit is set by H/W while ATK detected the key touched, write 0 to this bit will clear this flag

| F14 | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|-------|-------|-------|-------|--------|-------|-------|-------|-------|
| TKCTL | TKPD | TKTMR | | | TKSOC | - | TKCHS | |
| | | TKTMR | | TKCHS2 | | | | |
| R/W | R/W | R/W | | R/W | R/W | - | R/W | |
| Reset | 1 | 1 | 0 | 0 | 0 | - | 0 | 0 |

F14.7 **TKPD**: Touch key power down
 0: Touch key running
 1: Touch key power down

F14.6~4 **TKTMR**: Touch key conversion time select (For TM57PT16)
 000: Level 0 (shortest)
 001: Level 1
 010: Level 2
 011: Level 3
 100: Level 4
 101: Level 5
 110: Level 6
 111: Level 7 (longest)

F14.6~5 **TKTMR**: Touch key conversion time select (For TM57PT16B)
 00: Level 0 (shortest)
 01: Level 2
 10: Level 4
 11: Level 6 (longest)

F14.4 **TKCHS2**: Touch key channel (bit 2) select (For TM57PT16B)
 0: TK0~TK3 by TKCHS
 1: if TKCHS=11B, Touch key channel select internal reference capacitor
 if TKCHS≠11B, Touch key channel is invalid

F14.3 **TKSOC**: Touch key start of conversion, rising edge to start
 H/W auto cleared while end of conversion

F14.1~0 **TKCHS**: Touch key channel select
 00: TK0 (PA1)
 01: TK1 (PA0)
 10: TK2 (PA2)
 11: TK3 (PA4)

| F15 | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|--------|-------|-------|-------|-------|-------|--------|---------|-------|
| ATKCTL | - | - | - | - | - | TKAUTO | TKSCNUM | |
| R/W | - | - | - | - | - | R/W | R/W | |
| Reset | - | - | - | - | - | 0 | 0 | 0 |

F15.2 **TKAUTO**: Touch key auto scan mode enable
 0: disable H/W Auto Mode
 1: enable H/W Auto Mode

F15.1~0 **TKSCNUM**: Touch key auto scan channel number
 00: only detect TK0
 01: detect TK0 and TK1
 10: detect TK0, TK1 and TK2
 11: detect TK0~TK3

| F16 | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|-------|-------|-------|-------|-------|-------|-------|-------|-------|
| TKDT | ATKDT | | | | TKEOC | TKOVF | TKDH | |
| R/W | R | | | | R | R | R | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

F16.7~4 **ATKDT**: Touch key auto scan result
 xxx1: TK0 has a touch event
 xx1x: TK1 has a touch event
 x1xx: TK2 has a touch event
 1xxx: TK3 has a touch event

F16.3 **TKEOC**: Touch key end of conversion
 0: conversion is in process
 1: end of conversion

F16.2 **TKOVF**: Touch key counter overflow flag
 0: not overflow
 1: overflow

F16.1~0 **TKDH**: Touch key data MSB[9:8]

| F17 | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|-------|-------|-------|-------|-------|-------|-------|-------|-------|
| TKDL | TKDL | | | | | | | |
| R/W | R | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

F17.7~0 **TKDL**: Touch key data LSB[7:0]

| F18 | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---------|---------|-------|-------|-------|-------|-------|-------|-------|
| ATKCMP0 | ATKCMP0 | | | | | | | |
| R/W | R/W | | | | | | | |
| Reset | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |

F18.7~0 **ATKCMP0**: Touch key auto scan TK0 compare data threshold LSB[7:0]

| F19 | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---------|---------|-------|-------|-------|-------|-------|-------|-------|
| ATKCMP1 | ATKCMP1 | | | | | | | |
| R/W | R/W | | | | | | | |
| Reset | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |

F19.7~0 **ATKCMP1**: Touch key auto scan TK1 compare data threshold LSB[7:0]

| F1A | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---------|---------|-------|-------|-------|-------|-------|-------|-------|
| ATKCMP2 | ATKCMP2 | | | | | | | |
| R/W | R/W | | | | | | | |
| Reset | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |

F1A.7~0 **ATKCMP2**: Touch key auto scan TK2 compare data threshold LSB[7:0]

| F1B | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---------|---------|-------|-------|-------|-------|-------|-------|-------|
| ATKCMP3 | ATKCMP3 | | | | | | | |
| R/W | R/W | | | | | | | |
| Reset | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |

F1B.7~0 **ATKCMP3**: Touch key auto scan TK3 compare data threshold LSB[7:0]

| F1C | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|-----------|-------|----------|-------|-------|-------|----------|-------|-------|
| ATKCMP10H | – | ATKCMP1H | | | – | ATKCMP0H | | |
| R/W | – | R/W | | | – | R/W | | |
| Reset | – | 0 | 0 | 0 | – | 0 | 0 | 0 |

F1C.6~4 **ATKCMP1H**: Touch key auto scan TK1 compare data threshold MSB[10:8]

F1C.2~0 **ATKCMP0H**: Touch key auto scan TK0 compare data threshold MSB[10:8]

| F1D | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|-----------|-------|----------|-------|-------|-------|----------|-------|-------|
| ATKCMP32H | – | ATKCMP3H | | | – | ATKCMP2H | | |
| R/W | – | R/W | | | – | R/W | | |
| Reset | – | 0 | 0 | 0 | – | 0 | 0 | 0 |

F1D.6~4 **ATKCMP3H**: Touch key auto scan TK3 compare data threshold MSB[10:8]

F1D.2~0 **ATKCMP2H**: Touch key auto scan TK2 compare data threshold MSB[10:8]

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

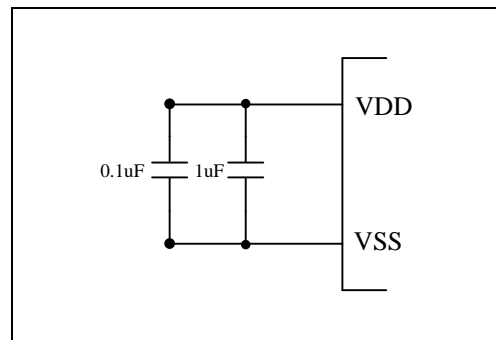
R0B.1~0 **WKTPSC**: WDT/WKT pre-scale option select
 00: WDT/WKT period is 22 ms, @5V; 28 ms, @3V
 01: WDT/WKT period is 44 ms, @5V; 56 ms, @3V
 10: WDT/WKT period is 88 ms, @5V; 112 ms, @3V
 11: WDT/WKT period is 176 ms, @5V; 224 ms, @3V

| R14 | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|-------|-------|-------|-------|-------|-------|-------|-------|-------|
| PAIE | – | – | – | PAIE | | | | |
| R/W | – | – | – | W | | | | |
| Reset | – | – | – | 1 | 1 | 1 | 1 | 1 |

R14.4~0 **PAIE**: PA4~0 input type select
 0: analog input
 1: digital input

3.6 System Clock Oscillator

System clock can only be operated in internal RC mode. The on chip oscillator generates 4/8 MHz system clock. When the IRC is 8 MHz, the LVR can only be set to 3.1V (cannot use 2.2V). 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.

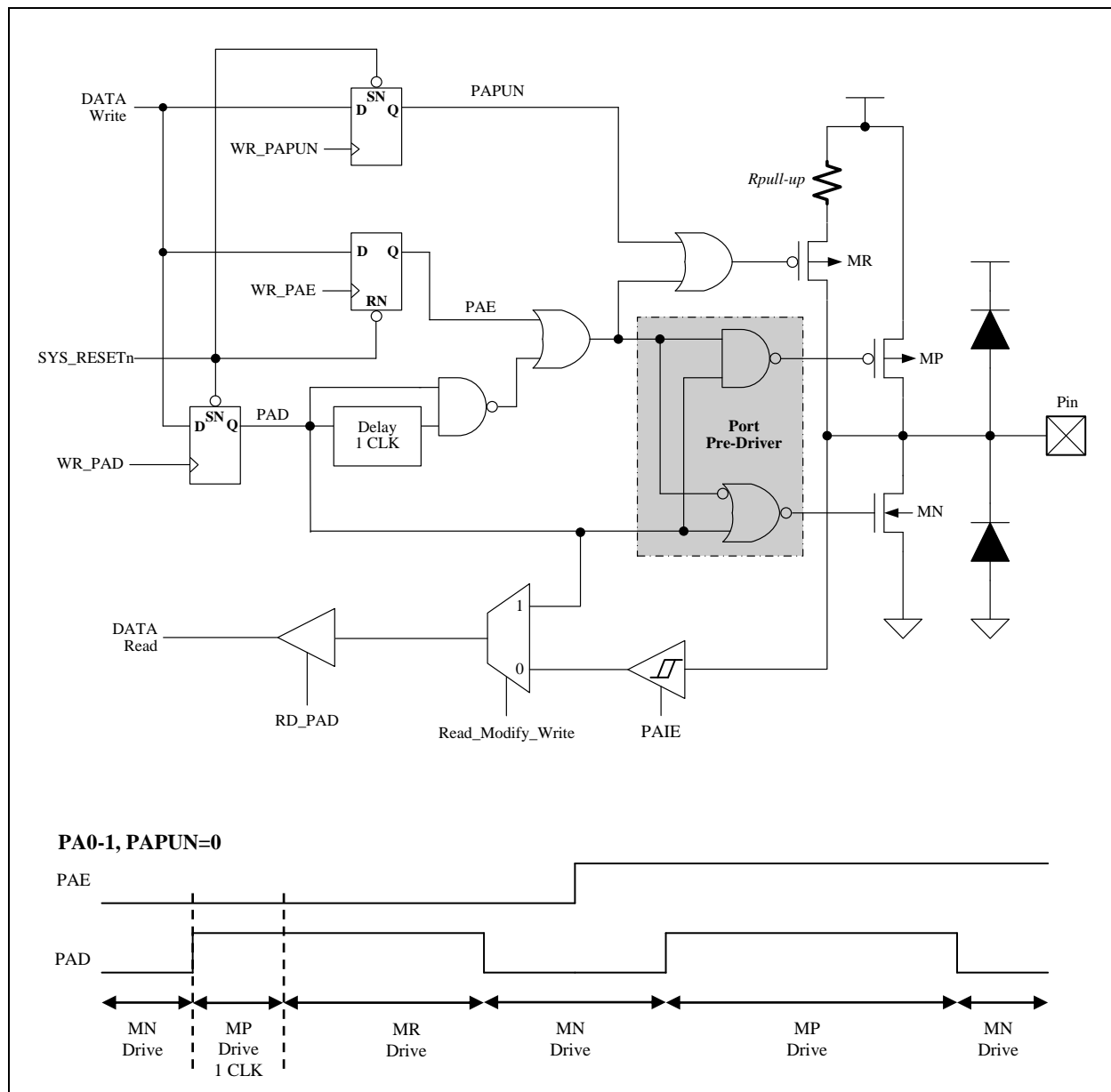


Internal RC Mode

4. I/O Port

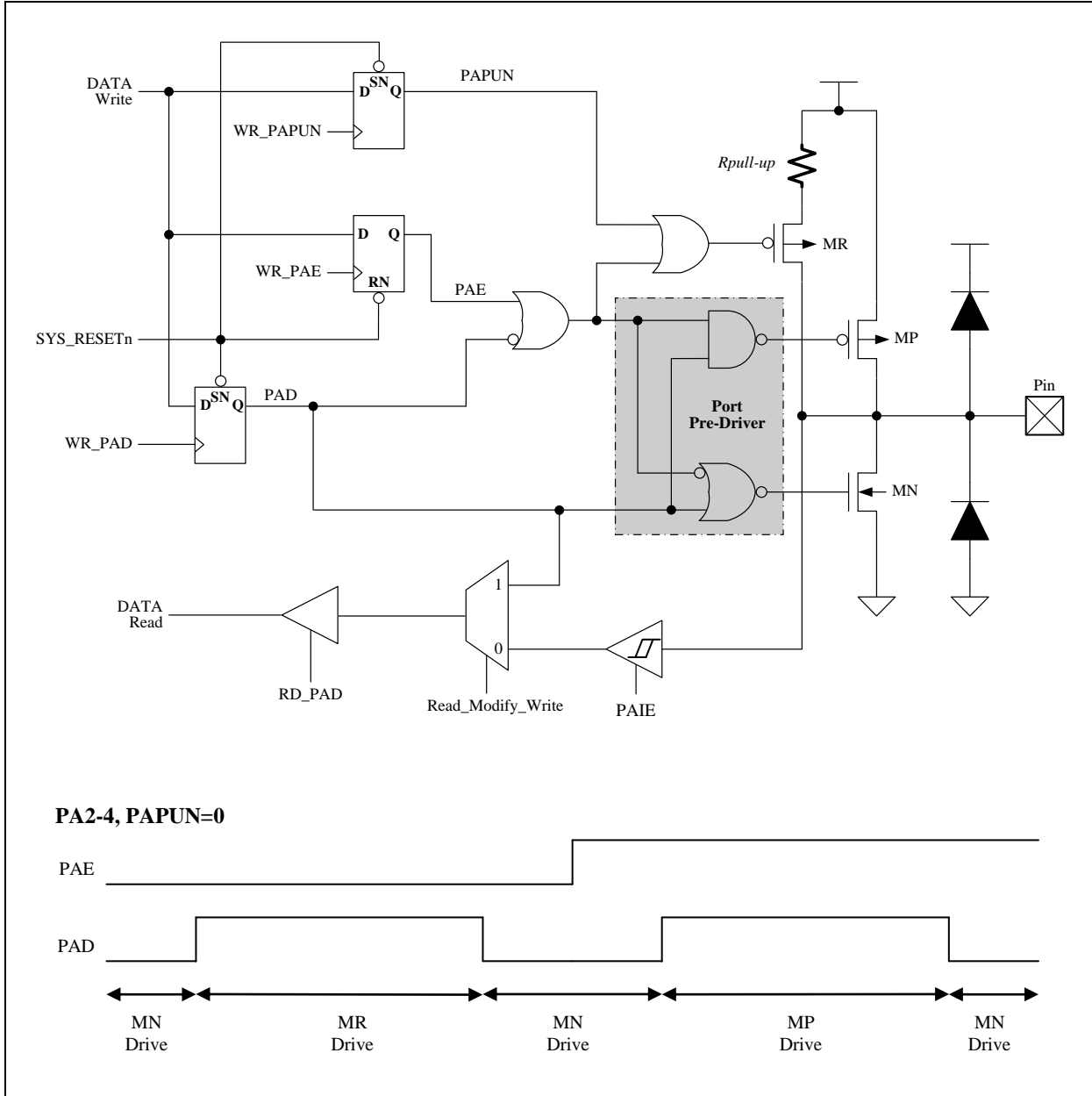
4.1 PA0-1

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.



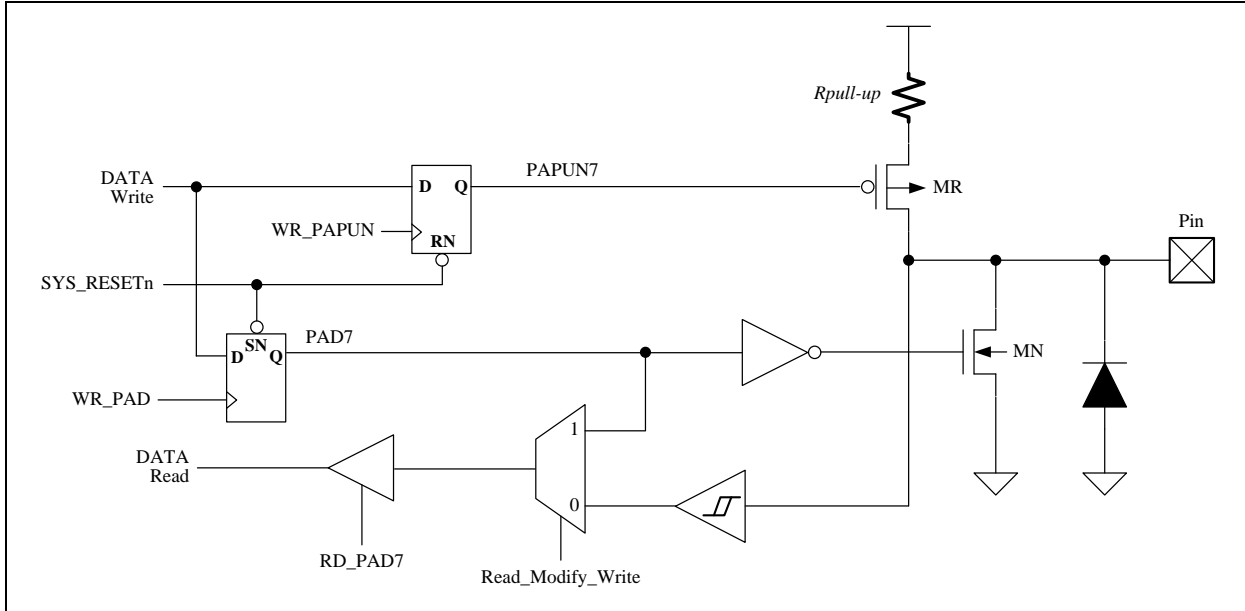
4.2 PA2-4

These pins are almost the same as PA0-1, 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 used in Schmitt-trigger input or pure open-drain output. The pull-up resistor connected to this pin default, and can be disabled by S/W.



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

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

F05.4~0 **PAD:** PA4~PA0 data
 0: output low
 1: output high or Schmitt-trigger input mode

| | | | | | | | | |
|-------|-------|-------|-------|-------|-------|-------|-------|-------|
| R05 | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
| PAE | – | – | – | PAE | | | | |
| R/W | – | – | – | W | | | | |
| Reset | – | – | – | 0 | 0 | 0 | 0 | 0 |

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

R05.1~0 **PAE:** PA1~PA0 pin mode control
 0: the pin is pseudo-open-drain output or Schmitt-trigger input
 1: the pin is CMOS push-pull output

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

R08.7 **PAPUN7**: PA7 pull-up resistor enable
 0: enable
 1: disable

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

| R13 | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|--------|-------|-------|-------|--------|-------|-------|-------|-------|
| PAWKEN | – | – | – | PAWKEN | | | | – |
| R/W | – | – | – | W | | | | – |
| Reset | – | – | – | 0 | 0 | 0 | 0 | – |

R13.4~1 **PAWKEN**: PA4~PA1 pin low level wakeup enable
 0: disable
 1: enable

| R14 | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|-------|-------|-------|-------|-------|-------|-------|-------|-------|
| PAIE | – | – | – | PAIE | | | | |
| R/W | – | – | – | W | | | | |
| Reset | – | – | – | 1 | 1 | 1 | 1 | 1 |

R14.4~0 **PAIE**: PA4~0 input type select
 0: analog input
 1: digital input

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: Programming Counter | | | | |
| PCL | 02.7~0 | R/W | 0 | Programming counter LSB[7:0] |
| (F03) STATUS Function related to: Status | | | | |
| TO | 03.4 | R | 0 | WDT time out flag |
| PD | 03.3 | R | 0 | STOP mode flag |
| Z | 03.2 | R/W | 0 | Zero flag |
| DC | 03.1 | R/W | 0 | Decimal Carry flag or Decimal /Borrow flag |
| C | 03.0 | R/W | 0 | Carry flag or /Borrow flag |
| (F04) FSR Function related to: RAM W/R | | | | |
| FSR | 04.6~0 | R/W | - | File select register, indirect address mode pointer |
| (F05) PAD Function related to: Port A | | | | |
| PAD7 | 05.7 | R | - | PA7 pin or “data register” state |
| | | W | 1 | 0: PA7 is open-drain output mode 1: PA7 is Schmitt-trigger input mode |
| PAD | 05.4~0 | R | - | Port A pin or “data register” state |
| | | W | 1F | Port A output data register |
| (F08) INTIE Function related to: Interrupt Enable | | | | |
| ATKIE (PT16/PT16B only) | 08.5 | R/W | 0 | Auto touch key interrupt enable 0: disable 1: enable |
| TM0IE | 08.4 | R/W | 0 | Timer0 interrupt enable 0: disable 1: enable |
| WKTIE | 08.3 | R/W | 0 | Wakeup timer interrupt enable 0: disable 1: enable |
| INT2IE | 08.2 | R/W | 0 | INT2 (PA7) falling interrupt enable 0: disable 1: enable |
| INT0IE | 08.0 | R/W | 0 | INT0 (PA0) falling/rising interrupt enable 0: disable 1: enable |

| Name | Address | R/W | Rst | Description |
|--|---------|-----|-----|---|
| (F09) INTIF | | | | Function related to: Interrupt Flag |
| ATKIF (PT16/PT16B only) | 09.5 | R | - | Auto touch key Interrupt event pending flag, set by H/W while auto touch key detected the key touched |
| | | W | 0 | 0: clear this flag 1: no action |
| TM0IF | 09.4 | R | - | Timer0 interrupt event pending flag, set by H/W while Timer0 overflows |
| | | W | 0 | 0: clear this flag 1: no action |
| WKTIF | 09.3 | R | - | WKT interrupt event pending flag, set by H/W while WKT is timeout |
| | | W | 0 | 0: clear this flag 1: no action |
| INT2IF | 09.2 | R | - | INT2 (PA7) pin falling interrupt pending flag, set by H/W at INT2 pin's falling edge |
| | | W | 0 | 0: clear this flag 1: no action |
| INT0IF | 09.0 | R | - | INT0 (PA0) pin falling/rising interrupt pending flag, set by H/W at INT0 pin's falling/rising edge |
| | | W | 0 | 0: clear this flag 1: no action |
| (F12) PWM0D | | | | Function related to: PWM0 |
| PWM0D | 12.7~0 | R/W | 0 | PWM0 duty |
| (F14) TKCTL (PT16/PT16B only) | | | | Function related to: Touch Key |
| TKPD | 14.7 | R/W | 1 | Touch key power down 0: Touch key running 1: Touch key power down |
| TKTMR (for TM57PT16) | 14.6~4 | R/W | 4 | Touch key conversion time select 000: Level 0 (shortest) 001: Level 1 010: Level 2 011: Level 3 100: Level 4 101: Level 5 110: Level 6 111: Level 7 (longest) |
| TKTMR (for TM57PT16B) | 14.6~5 | R/W | 2 | Touch key conversion time select 00: Level 0 (shortest) 01: Level 2 10: Level 4 11: Level 6 (longest) |
| TKCHS2 (for TM57PT16B) | 14.4 | R/W | 0 | Touch key channel (bit 2) select 0: TK0~TK3 by TKCHS 1: if TKCHS=11B, Touch key channel select internal reference capacitor if TKCHS≠11B, Touch key channel is invalid |
| TKSOC | 14.3 | R/W | 0 | Touch key start of conversion, rising edge to start H/W auto cleared while end of conversion |
| TKCHS | 14.1~0 | R/W | 0 | Touch key channel select 00: TK0 (PA1) 01: TK1 (PA0) 10: TK2 (PA2) 11: TK3 (PA4) |

| Name | Address | R/W | Rst | Description |
|--|---------|-----|-----|--|
| (F15) ATKCTL (PT16/PT16B only) | | | | Function related to: Touch Key |
| TKAUTO | 15.2 | R/W | 0 | Touch key auto scan mode enable 0: disable H/W auto mode 1: enable H/W auto mode |
| TKSCNUM | 15.1~0 | R/W | 3 | Touch key auto scan channel number 00: only detect TK0 01: detect TK0 and TK1 10: detect TK0, TK1 and TK2 11: detect TK0~TK3 |
| (F16) TKDT (PT16/PT16B only) | | | | Function related to: Touch Key |
| ATKDT | 16.7~4 | R | 0 | Touch key auto scan result xxx1: TK0 has a touch event xx1x: TK1 has a touch event x1xx: TK2 has a touch event 1xxx: TK3 has a touch event |
| TKEOC | 16.3 | R | 0 | Touch key end of conversion 0: conversion is in process 1: end of conversion |
| TKOVF | 16.2 | R | 0 | Touch key counter overflow flag 0: not overflow 1: overflow |
| TKDH | 16.1~0 | R | 0 | Touch key data MSB[9:8] |
| (F17) TKDL (PT16/PT16B only) | | | | Function related to: Touch Key |
| TKDL | 17.7~0 | R | 0 | Touch key data LSB[7:0] |
| (F18) ATKCMP0 (PT16/PT16B only) | | | | Function related to: Touch Key |
| ATKCMP0 | 18.7~0 | R/W | 40 | Touch key auto scan TK0 compare data threshold LSB[7:0] |
| (F19) ATKCMP1 (PT16/PT16B only) | | | | Function related to: Touch Key |
| ATKCMP1 | 19.7~0 | R/W | 40 | Touch key auto scan TK1 compare data threshold LSB[7:0] |
| (F1A) ATKCMP2 (PT16/PT16B only) | | | | Function related to: Touch Key |
| ATKCMP2 | 1A.7~0 | R/W | 40 | Touch key auto scan TK2 compare data threshold LSB[7:0] |
| (F1B) ATKCMP3 (PT16/PT16B only) | | | | Function related to: Touch Key |
| ATKCMP3 | 1B.7~0 | R/W | 40 | Touch key auto scan TK3 compare data threshold LSB[7:0] |
| (F1C) ATKCMP10H (PT16/PT16B only) | | | | Function related to: Touch Key |
| ATKCMP1H | 1C.6~4 | R/W | 0 | Touch key auto scan TK1 compare data threshold MSB[10:8] |
| ATKCMP0H | 1C.2~0 | R/W | 0 | Touch key auto scan TK0 compare data threshold MSB[10:8] |
| (F1D) ATKCMP32H (PT16/PT16B only) | | | | Function related to: Touch Key |
| ATKCMP3H | 1D.6~4 | R/W | 0 | Touch key auto scan TK3 compare data threshold MSB[10:8] |
| ATKCMP2H | 1D.2~0 | R/W | 0 | Touch key auto scan TK2 compare data threshold MSB[10:8] |
| (F1E) ADH | | | | Function related to: ADC |
| ADCDH | 1E.7~0 | R | 0 | ADC output data MSB[9:2] |

| Name | Address | R/W | Rst | Description |
|-------------------------|---------|-----|-----|--|
| (F1F) ADCTL | | | | Function related to: ADC |
| ADCDL | 1F.7~6 | R | 0 | ADC output data LSB[1:0] |
| ADCPSC | 1F.5~4 | R/W | 0 | ADC clock source prescaler. ADC clock source (Fsys) 00: divided by 2 01: divided by 4 10: divided by 8 11: divided by 16 |
| ADCST | 1F.3 | R | - | H/W clear this bit after ADC end of conversion |
| | | W | 0 | 0: no action 1: start ADC conversion |
| ADCHS | 1F.2~0 | R/W | 0 | ADC channel select 000: ADC0 (PA3) 001: ADC1 (PA1) 010: ADC2 (PA0) 011: ADC3 (PA2) 100: ADC4 (PA4) |
| User Data Memory | | | | |
| SRAM | 20~4F | R/W | - | Internal RAM |

R-Plane

| Name | Address | R/W | Rst | Description |
|--|---------|-----|-----|--|
| (R02) TM0CTL Function related to: Timer0 | | | | |
| TM0EDG | 02.5 | W | 0 | TM0CKI (PA2) edge selection for Timer0 prescaler count 0: TM0CKI (PA2) rising edge for Timer0 Prescaler count 1: TM0CKI (PA2) falling edge for Timer0 Prescaler count |
| TM0CKS | 02.4 | W | 0 | Timer0 clock source select 0: Instruction Cycle as Timer0 Prescaler clock 1: TM0CKI as Timer0 Prescaler clock |
| TM0PSC | 02.3~0 | W | 0 | Timer0 prescaler. Timer0 clock source 0000: divided by 1 0001: divided by 2 0010: divided by 4 0011: divided by 8 0100: divided by 16 0101: divided by 32 0110: divided by 64 0111: divided by 128 1xxx: divided by 256 |
| (R03) PWRDN Function related to: Power Down | | | | |
| PWRDN | 03 | W | - | Write this register to enter Power Down Mode |
| (R04) WDTCLR Function related to: WDT | | | | |
| WDTCLR | 04 | W | - | Write this register to clear WDT/WKT |
| (R05) PAE Function related to: Port A | | | | |
| PAE | 05.4~2 | W | 0 | PA4~PA2 I/O mode control Each bit controls its corresponding pin, if the bit is 0: the pin is open-drain output or Schmitt-trigger input 1: the pin is CMOS push-pull output |
| | 05.1~0 | W | 0 | PA1~PA0 I/O mode control Each bit controls its corresponding pin, if the bit is 0: the pin is pseudo-open-drain output or Schmitt-trigger input 1: the pin is CMOS push-pull output |
| (R08) PAPUN Function related to: Port A | | | | |
| PAPUN7 | 08.7 | W | 0 | PA7 pull-up control, if the bit is 0: the pin pull-up resistor is enable 1: the pin pull-up resistor is disable |
| PAPUN | 08.4~0 | W | 1F | PA4~PA0 pull-up control Each bit controls its corresponding pin, if the bit is 0: the pin pull-up resistor is enable, except a. the pin's output data register (PAD) is 0 b. the pin's CMOS push-pull mode is chosen (PAE=1) 1: the pin pull-up resistor is disable |

| Name | Address | R/W | Rst | Description |
|----------------------|---------|-----|-----|--|
| (R0B) MR0B | | | | Function related to: INT0 / TCOUT / WKT / WDT |
| INT0EDG | 0b.4 | W | 0 | INT0 pin (PA0) edge interrupt event 0: INT0 (PA0) pin falling edge to trigger interrupt event 1: INT0 (PA0) pin rising edge to trigger interrupt event |
| TCOE | 0b.3 | W | 0 | Enable Instruction Cycle (Fsys/2) output to PA3 pin (TCOUT) 0: disable 1: enable |
| WKT PSC | 0b.1~0 | W | 3 | WDT/WKT pre-scale option select 00: WDT/WKT period is 22 ms, @5V; 28 ms, @3V 01: WDT/WKT period is 44 ms, @5V; 56 ms, @3V 10: WDT/WKT period is 88 ms, @5V; 112 ms, @3V 11: WDT/WKT period is 176 ms, @5V; 224 ms, @3V |
| (R10) PWM0PRD | | | | Function related to: PWM0 |
| PWM0PRD | 10.7~0 | W | FF | PWM0 period data |
| (R11) PWMCTL | | | | Function related to: PWM0 |
| PWM0E | 11.2 | W | 0 | PWM0 positive output to PA4 pin 0: disable 1: enable |
| PWM0PSC | 11.1~0 | W | 0 | PWM0 prescaler, PWM0 clock source 00: divided by 1 01: divided by 2 10: divided by 4 11: divided by 8 |
| (R13) PAWKEN | | | | Function related to: Port A / Wake up |
| PAWKEN | 13.4~1 | W | 0 | PA4~PA1 individual pin low level wake up control Each bit controls its corresponding pin, if the bit is 0: disable 1: enable |
| (R14) PAIE | | | | Function related to: Port A / Touch Key / ADC |
| PAIE | 14.4~0 | W | 1F | PA input type selection Each bit controls its corresponding pin, if the bit is 0: analog input 1: digital input |

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 used by the instruction. The destination designator specifies where the result of the operation is 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 | | | | | |
| <u>ADDWF</u> | f,d | 00 0111 dfff ffff | 1 | C,DC,Z | Add W and "f" |
| <u>ANDWF</u> | f,d | 00 0101 dfff ffff | 1 | Z | AND W with "f" |
| <u>CLRF</u> | f | 00 0001 1fff ffff | 1 | Z | Clear "f" |
| <u>CLRW</u> | | 00 0001 0100 0000 | 1 | Z | Clear W |
| <u>COMF</u> | f,d | 00 1001 dfff ffff | 1 | Z | Complement "f" |
| <u>DECF</u> | f,d | 00 0011 dfff ffff | 1 | Z | Decrement "f" |
| <u>DECFSZ</u> | f,d | 00 1011 dfff ffff | 1 or 2 | - | Decrement "f", skip if zero |
| <u>INCF</u> | f,d | 00 1010 dfff ffff | 1 | Z | Increment "f" |
| <u>INCFSZ</u> | f,d | 00 1111 dfff ffff | 1 or 2 | - | Increment "f", skip if zero |
| <u>IORWF</u> | f,d | 00 0100 dfff ffff | 1 | Z | OR W with "f" |
| <u>MOVFW</u> | f | 00 1000 0fff ffff | 1 | - | Move "f" to W |
| <u>MOVWF</u> | f | 00 0000 1fff ffff | 1 | - | Move W to "f" |
| <u>MOVWR</u> | r | 00 0000 00rr rrrr | 1 | - | Move W to "r" |
| <u>RLF</u> | f,d | 00 1101 dfff ffff | 1 | C | Rotate left "f" through carry |
| <u>RRF</u> | f,d | 00 1100 dfff ffff | 1 | C | Rotate right "f" through carry |
| <u>SUBWF</u> | f,d | 00 0010 dfff ffff | 1 | C,DC,Z | Subtract W from "f" |
| <u>SWAPF</u> | f,d | 00 1110 dfff ffff | 1 | - | Swap nibbles in "f" |
| <u>TESTZ</u> | f | 00 1000 1fff ffff | 1 | Z | Test if "f" is zero |
| <u>XORWF</u> | f,d | 00 0110 dfff ffff | 1 | Z | XOR W with "f" |
| Bit-Oriented File Register Instruction | | | | | |
| <u>BCF</u> | f,b | 01 000b bbff ffff | 1 | - | Clear "b" bit of "f" |
| <u>BSF</u> | f,b | 01 001b bbff ffff | 1 | - | Set "b" bit of "f" |
| <u>BTFSC</u> | f,b | 01 010b bbff ffff | 1 or 2 | - | Test "b" bit of "f", skip if clear |
| <u>BTFSS</u> | f,b | 01 011b bbff ffff | 1 or 2 | - | Test "b" bit of "f", skip if set |
| Literal and Control Instruction | | | | | |
| <u>ADDLW</u> | k | 01 1100 kkkk kkkk | 1 | C,DC,Z | Add Literal "k" and W |
| <u>ANDLW</u> | k | 01 1011 kkkk kkkk | 1 | Z | AND Literal "k" with W |
| <u>CALL</u> | k | 10 kkkk kkkk kkkk | 2 | - | Call subroutine "k" |
| <u>CLRWDI</u> | | 00 0000 0000 0100 | 1 | TO,PD | Clear Watch Dog Timer |
| <u>GOTO</u> | k | 11 kkkk kkkk kkkk | 2 | - | Jump to branch "k" |
| <u>IORLW</u> | k | 01 1010 kkkk kkkk | 1 | Z | OR Literal "k" with W |
| <u>MOVLW</u> | k | 01 1001 kkkk kkkk | 1 | - | Move Literal "k" to W |
| <u>NOP</u> | | 00 0000 0000 0000 | 1 | - | No operation |
| <u>RET</u> | | 00 0000 0100 0000 | 2 | - | Return from subroutine |
| <u>RETI</u> | | 00 0000 0110 0000 | 2 | - | Return from interrupt |
| <u>RETLW</u> | k | 01 1000 kkkk kkkk | 2 | - | Return with Literal in W |
| <u>SLEEP</u> | | 00 0000 0000 0011 | 1 | TO,PD | Go into STOP mode, Clock oscillation stops |
| <u>XORLW</u> | 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 | $(W) \leftarrow (W) + k$ | |
| Status Affected | C, DC, Z | |
| OP-Code | 01 1100 kkkk kkkk | |
| Description | The contents of the W register are added to the eight-bit literal 'k' and the result is placed in the W register. | |
| Cycle | 1 | |
| Example | ADDLW 0x15 | B : W = 0x10 A : W = 0x25 |

ADDWF
Add W and "f"

| | | |
|-----------------|--|--|
| Syntax | ADDWF f [,d] | |
| Operands | f : 00h ~ 7Fh, d : 0, 1 | |
| Operation | $(\text{destination}) \leftarrow (W) + (f)$ | |
| Status Affected | C, DC, Z | |
| OP-Code | 00 0111 dfff ffff | |
| Description | Add the contents of the W register with register 'f'. If 'd' is 0, the result is stored in the W register. If 'd' is 1, the result is stored back in register 'f'. | |
| Cycle | 1 | |
| Example | ADDWF FSR, 0 | B : W = 0x17, FSR = 0xC2 A : W = 0xD9, FSR = 0xC2 |

ANDLW
Logical AND Literal "k" with W

| | | |
|-----------------|---|------------------------------|
| Syntax | ANDLW k | |
| Operands | k : 00h ~ FFh | |
| Operation | $(W) \leftarrow (W) \text{ AND } k$ | |
| Status Affected | Z | |
| OP-Code | 01 1011 kkkk kkkk | |
| Description | The contents of W register are AND'ed with the eight-bit literal 'k'. The result is placed in the W register. | |
| Cycle | 1 | |
| Example | ANDLW 0x5F | B : W = 0xA3 A : W = 0x03 |

ANDWF
AND W with "f"

| | | |
|-----------------|--|--|
| Syntax | ANDWF f [,d] | |
| Operands | f : 00h ~ 7Fh, d : 0, 1 | |
| Operation | $(\text{destination}) \leftarrow (W) \text{ AND } (f)$ | |
| Status Affected | Z | |
| OP-Code | 00 0101 dfff ffff | |
| Description | AND the W register with register 'f'. If 'd' is 0, the result is stored in the W register. If 'd' is 1, the result is stored back in register 'f'. | |
| Cycle | 1 | |
| Example | ANDWF FSR, 1 | B : W = 0x17, FSR = 0xC2 A : W = 0x17, FSR = 0x02 |

BCF Clear "b" bit of "f"

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

BSF Set "b" bit of "f"

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

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

| | | |
|-----------------|--|--|
| Syntax | BTFSC f [,b] | |
| Operands | f : 00h ~ 3Fh, b : 0 ~ 7 | |
| Operation | Skip next instruction if (f.b) = 0 | |
| Status Affected | - | |
| OP-Code | 01 010b bbff ffff | |
| Description | If bit 'b' in register 'f' is 1, then the next instruction is executed. If bit 'b' in register 'f' is 0, then the next instruction is discarded, and a NOP is executed instead, making this a 2nd cycle instruction. | |
| Cycle | 1 or 2 | |
| Example | LABEL1 BTFSC FLAG, 1 TRUE GOTO SUB1 FALSE ... | B : PC = LABEL1 A : if FLAG.1 = 0, PC = FALSE if FLAG.1 = 1, PC = TRUE |

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

| | | |
|-----------------|--|--|
| Syntax | BTFSS f [,b] | |
| Operands | f : 00h ~ 3Fh, b : 0 ~ 7 | |
| Operation | Skip next instruction if (f.b) = 1 | |
| Status Affected | - | |
| OP-Code | 01 011b bbff ffff | |
| Description | If bit 'b' in register 'f' is 0, then the next instruction is executed. If bit 'b' in register 'f' is 1, then the next instruction is discarded, and a NOP is executed instead, making this a 2nd cycle instruction. | |
| Cycle | 1 or 2 | |
| Example | LABEL1 BTFSS FLAG, 1 TRUE GOTO SUB1 FALSE ... | B : PC = LABEL1 A : if FLAG.1 = 0, PC = TRUE if FLAG.1 = 1, PC = FALSE |

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

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

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

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

| COMF | Complement "f" | |
|-----------------|--|--|
| Syntax | COMF f [,d] | |
| Operands | f : 00h ~ 7Fh, d : 0, 1 | |
| Operation | (destination) ← (\bar{f}) | |
| Status Affected | Z | |
| OP-Code | 00 1001 dfff ffff | |
| Description | The contents of register 'f' are complemented. If 'd' is 0, the result is stored in W. If 'd' is 1, the result is stored back in register 'f'. | |
| Cycle | 1 | |
| Example | COMF REG1, 0 | B : REG1 = 0x13 A : REG1 = 0x13, W = 0xEC |

| DECF | Decrement "f" | |
|-----------------|--|--|
| Syntax | DECF f [,d] | |
| Operands | f : 00h ~ 7Fh, d : 0, 1 | |
| Operation | (destination) ← (f) - 1 | |
| Status Affected | Z | |
| OP-Code | 00 0011 dfff ffff | |
| Description | Decrement register 'f'. If 'd' is 0, the result is stored in the W register. If 'd' is 1, the result is stored back in register 'f'. | |
| Cycle | 1 | |
| Example | DECF CNT, 1 | B : CNT = 0x01, Z = 0 A : CNT = 0x00, Z = 1 |

| DECFSZ | Decrement "f", Skip if 0 | |
|-----------------|---|--|
| Syntax | DECFSZ f [,d] | |
| Operands | f : 00h ~ 7Fh, d : 0, 1 | |
| Operation | (destination) ← (f) - 1, skip next instruction if result is 0 | |
| Status Affected | - | |
| OP-Code | 00 1011 dfff ffff | |
| Description | The contents of register 'f' are decremented. If 'd' is 0, the result is placed in the W register. If 'd' is 1, the result is placed back in register 'f'. If the result is 1, the next instruction is executed. If the result is 0, then a NOP is executed instead, making it a 2 cycle instruction. | |
| Cycle | 1 or 2 | |
| Example | LABEL1 DECFSZ CNT, 1 GOTO LOOP CONTINUE | B : PC = LABEL1 A : CNT = CNT - 1 if CNT = 0, PC = CONTINUE if CNT ≠ 0, PC = LABEL1 + 1 |

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

INCF Increment "f"

| | | |
|-----------------|--|--|
| Syntax | INCF f [,d] | |
| Operands | f : 00h ~ 7Fh | |
| Operation | (destination) ← (f) + 1 | |
| Status Affected | Z | |
| OP-Code | 00 1010 dfff ffff | |
| Description | The contents of register 'f' are incremented. If 'd' is 0, the result is placed in the W register. If 'd' is 1, the result is placed back in register 'f'. | |
| Cycle | 1 | |
| Example | INCF CNT, 1 | B : CNT = 0xFF, Z = 0 A : CNT = 0x00, Z = 1 |

INCFSZ Increment "f", Skip if 0

| | | |
|-----------------|--|--|
| Syntax | INCFSZ f [,d] | |
| Operands | f : 00h ~ 7Fh, d : 0, 1 | |
| Operation | (destination) ← (f) + 1, skip next instruction if result is 0 | |
| Status Affected | - | |
| OP-Code | 00 1111 dfff ffff | |
| Description | The contents of register 'f' are incremented. If 'd' is 0, the result is placed in the W register. If 'd' is 1, the result is placed back in register 'f'. If the result is 1, the next instruction is executed. If the result is 0, a NOP is executed instead, making it a 2 cycle instruction. | |
| Cycle | 1 or 2 | |
| Example | LABEL1 INCFSZ CNT, 1 GOTO LOOP CONTINUE | B : PC = LABEL1 A : CNT = CNT + 1 if CNT = 0, PC = CONTINUE if CNT ≠ 0, PC = LABEL1 + 1 |

IORLW Inclusive OR Literal with W

| | | |
|-----------------|--|-------------------------------------|
| Syntax | IORLW k | |
| Operands | k : 00h ~ FFh | |
| Operation | (W) ← (W) OR k | |
| Status Affected | Z | |
| OP-Code | 01 1010 kkkk kkkk | |
| Description | The contents of the W register are OR'ed with the eight-bit literal 'k'. The result is placed in the W register. | |
| Cycle | 1 | |
| Example | IORLW 0x35 | B : W = 0x9A A : W = 0xBF, Z = 0 |

IORWF Inclusive OR W with "f"

| | | |
|-----------------|---|---|
| Syntax | IORWF f [,d] | |
| Operands | f : 00h ~ 7Fh, d : 0, 1 | |
| Operation | (destination) ← (W) OR k | |
| Status Affected | Z | |
| OP-Code | 00 0100 dfff ffff | |
| Description | Inclusive OR the W register with register 'f'. If 'd' is 0, the result is placed in the W register. If 'd' is 1, the result is placed back in register 'f'. | |
| Cycle | 1 | |
| Example | IORWF RESULT, 0 | B : RESULT = 0x13, W = 0x91 A : RESULT = 0x13, W = 0x93, Z = 0 |

MOVFW Move "f" to W

| | | |
|-----------------|---|---|
| Syntax | MOVFW f | |
| Operands | f : 00h ~ 7Fh | |
| Operation | (W) ← (f) | |
| Status Affected | - | |
| OP-Code | 00 1000 0fff ffff | |
| Description | The contents of register 'f' are moved to W register. | |
| Cycle | 1 | |
| Example | MOVFW FSR | B : FSR = 0xC2, W = ? A : FSR = 0xC2, W = 0xC2 |

MOVLW Move Literal to W

| | | |
|-----------------|--|---------------------------|
| Syntax | MOVLW k | |
| Operands | k : 00h ~ FFh | |
| Operation | (W) ← k | |
| Status Affected | - | |
| OP-Code | 01 1001 kkkk kkkk | |
| Description | The eight-bit literal 'k' is loaded into W register. The don't cares will assemble as 0's. | |
| Cycle | 1 | |
| Example | MOVLW 0x5A | B : W = ? A : W = 0x5A |

MOVWF Move W to "f"

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

MOVWR Move W to "r"

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

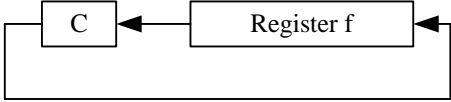
| NOP | No Operation |
|-----------------|---------------------|
| Syntax | NOP |
| Operands | - |
| Operation | No Operation |
| Status Affected | - |
| OP-Code | 00 0000 0000 0000 |
| Description | No Operation |
| Cycle | 1 |
| Example | NOP |

| RET | Return from Subroutine |
|-----------------|--|
| Syntax | RET |
| Operands | - |
| Operation | PC ← TOS |
| Status Affected | - |
| OP-Code | 00 0000 0100 0000 |
| Description | Return from subroutine. The stack is POPed and the top of the stack (TOS) is loaded into the program counter. This is a two-cycle instruction. |
| Cycle | 2 |
| Example | RET A : PC = TOS |


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

| RETLW | Return with Literal in W |
|-----------------|---|
| Syntax | RETLW k |
| Operands | k : 00h ~ FFh |
| Operation | PC ← TOS, (W) ← k |
| Status Affected | - |
| OP-Code | 01 1000 kkkk kkkk |
| Description | The W register is loaded with the eight-bit literal 'k'. The program counter is loaded from the top of the stack (the return address). This is a two-cycle instruction. |
| Cycle | 2 |
| Example | CALL TABLE B : W = 0x07 : A : W = value of k8 TABLE ADDWF PCL, 1 RETLW k1 RETLW k2 : RETLW kn |

RLF Rotate Left "f" through Carry

| | | |
|-----------------|---|---|
| Syntax | RLF f [,d] | |
| Operands | f : 00h ~ 7Fh, d : 0, 1 | |
| Operation |  | |
| Status Affected | C | |
| OP-Code | 00 1101 dfff ffff | |
| Description | The contents of register 'f' are rotated one bit to the left through the Carry Flag. If 'd' is 0, the result is placed in the W register. If 'd' is 1, the result is stored back in register 'f'. | |
| Cycle | 1 | |
| Example | RLF REG1, 0 | B : REG1 = 1110 0110, C = 0 A : REG1 = 1110 0110 W = 1100 1100, C = 1 |

RRF Rotate Right "f" through Carry

| | | |
|-----------------|--|---|
| Syntax | RRF f [,d] | |
| Operands | f : 00h ~ 7Fh, d : 0, 1 | |
| Operation |  | |
| Status Affected | C | |
| OP-Code | 00 1100 dfff ffff | |
| Description | The contents of register 'f' are rotated one bit to the right through the Carry Flag. If 'd' is 0, the result is placed in the W register. If 'd' is 1, the result is placed back in register 'f'. | |
| Cycle | 1 | |
| Example | RRF REG1, 0 | B : REG1 = 1110 0110, C = 0 A : REG1 = 1110 0110 W = 0111 0011, C = 0 |

SLEEP Go into 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 = ? A : REG1 = 0x01, W = 0x02, C = 1, Z = 0 |
| | SUBWF REG1, 1 | B : REG1 = 0x02, W = 0x02, C = ?, Z = ? A : REG1 = 0x00, W = 0x02, C = 1, Z = 1 |
| | SUBWF REG1, 1 | B : REG1 = 0x01, W = 0x02, C = ?, Z = ? A : REG1 = 0xFF, W = 0x02, C = 0, Z = 0 |

SWAPF Swap Nibbles in "f"

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

TESTZ Test if "f" is zero

| | | |
|-----------------|---|--|
| Syntax | TESTZ f | |
| Operands | f : 00h ~ 7Fh | |
| Operation | Set Z flag if (f) is 0 | |
| Status Affected | Z | |
| OP-Code | 00 1000 1fff ffff | |
| Description | If the content of register 'f' is 0, Zero flag is set to 1. | |
| Cycle | 1 | |
| Example | TESTZ REG1 | B : REG1 = 0, Z = ? A : REG1 = 0, Z = 1 |

XORLW Exclusive OR Literal with W

| | | |
|-----------------|---|------------------------------|
| Syntax | XORLW k | |
| Operands | k : 00h ~ FFh | |
| Operation | (W) ← (W) XOR k | |
| Status Affected | Z | |
| OP-Code | 01 1111 kkkk kkkk | |
| Description | The contents of the W register are XOR'ed with the eight-bit literal 'k'. The result is placed in the W register. | |
| Cycle | 1 | |
| Example | XORLW 0xAF | B : W = 0xB5 A : W = 0x1A |

XORWF Exclusive OR W with 'f'

| | | |
|-----------------|---|--|
| Syntax | XORWF f [,d] | |
| Operands | f : 00h ~ 7Fh, d : 0, 1 | |
| Operation | (destination) ← (W) XOR (f) | |
| Status Affected | Z | |
| OP-Code | 00 0110 dfff ffff | |
| Description | Exclusive OR the contents of the W register with register 'f'. If 'd' is 0, the result is stored in the W register. If 'd' is 1, the result is stored back in register 'f'. | |
| Cycle | 1 | |
| Example | XORWF REG, 1 | B : REG = 0xAF, W = 0xB5 A : REG = 0x1A, W = 0xB5 |

ELECTRICAL CHARACTERISTICS

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

| Parameter | Rating | Unit |
|---------------------------------|----------------------------------|------|
| Supply voltage | $V_{SS} - 0.3$ to $V_{SS} + 6.5$ | 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 | °C |
| Storage temperature | - 65 to + 150 | |

2. DC Characteristics ($T_A = 25^\circ\text{C}$, $V_{DD} = 1.9\text{V}$ to 5.5V)

| Parameter | Sym | Conditions | | Min | Typ | Max | Unit |
|----------------------------------|------------------|---------------------------------------|---|--------------|-----------|--------------|------------------|
| Operating Voltage | V_{DD} | 25°C, $F_{\text{sys}} = 8\text{ MHz}$ | | 2.3 | – | 5.5 | V |
| | | 25°C, $F_{\text{sys}} = 4\text{ MHz}$ | | 1.9 | – | 5.5 | |
| Input High Voltage | V_{IH} | All Input, except PA7 | $V_{DD} = 5\text{V}$ | $0.7 V_{DD}$ | – | – | V |
| | | | $V_{DD} = 3\text{V}$ | $0.7 V_{DD}$ | – | – | V |
| | | PA7 | $V_{DD} = 5\text{V}$ | $0.8 V_{DD}$ | – | – | V |
| | | | $V_{DD} = 3\text{V}$ | $0.8 V_{DD}$ | – | – | V |
| Input Low Voltage | V_{IL} | All Input | $V_{DD} = 5\text{V}$ | – | – | $0.2 V_{DD}$ | V |
| | | | $V_{DD} = 3\text{V}$ | – | – | $0.2 V_{DD}$ | V |
| I/O Port Source Current | I_{OH} | All Output | $V_{DD} = 5\text{V}$, $V_{OH} = 0.9 V_{DD}$ | 4 | 8 | – | mA |
| | | | $V_{DD} = 3\text{V}$, $V_{OH} = 0.9 V_{DD}$ | 1.5 | 3 | – | |
| I/O Port Sink Current | I_{OL} | All Output | $V_{DD} = 5\text{V}$, $V_{OL} = 0.1 V_{DD}$ | 10 | 20 | – | mA |
| | | | $V_{DD} = 3\text{V}$, $V_{OL} = 0.1 V_{DD}$ | 5 | 10 | – | |
| Input Leakage Current (pin high) | I_{ILH} | All Input | $V_{IN} = V_{DD}$ | – | – | 1 | μA |
| Input Leakage Current (pin low) | I_{ILL} | All Input | $V_{IN} = 0\text{V}$ | – | – | -1 | μA |
| Power Supply Current | I_{DD} | Run 8 MHz, LVR enable | $V_{DD} = 5\text{V}$ | – | 1.8 | – | mA |
| | | | $V_{DD} = 3\text{V}$ | – | 0.8 | – | |
| | | Run 4 MHz, LVR enable | $V_{DD} = 5\text{V}$ | – | 1.4 | – | mA |
| | | | $V_{DD} = 3\text{V}$ | – | 0.6 | – | |
| | | Stop mode, LVR enable | $V_{DD} = 5\text{V}$ | – | 1.8 | – | μA |
| | | | $V_{DD} = 3\text{V}$ | – | 0.5 | – | |
| | | Stop mode, LVR disable | $V_{DD} = 5\text{V}$ | – | – | 0.1 | μA |
| | | | $V_{DD} = 3\text{V}$ | – | – | 0.1 | |
| System Clock Frequency | F_{sys} | $V_{DD} > \text{LVR}_{th}$ | $V_{DD} = 3.1\text{V}$ | – | – | 8 | MHz |
| | | | $V_{DD} = 2.2\text{V}$ | – | – | 4 | |
| LVR Reference Voltage | V_{LVR} | $T_A = 25^\circ\text{C}$ | | – | 2.2 | – | V |
| | | | | – | 3.1 | – | V |
| LVR Hysteresis Voltage | V_{HYST} | $T_A = 25^\circ\text{C}$ | | – | ± 0.1 | – | V |
| Low Voltage Detection time | t_{LVR} | $T_A = 25^\circ\text{C}$ | | 100 | – | – | μs |
| Pull-Up Resistor | R_P | $V_{IN} = 0\text{ V}$ Port A | $V_{DD} = 5\text{V}$ | – | 68 | – | $\text{K}\Omega$ |
| | | | $V_{DD} = 3\text{V}$ | – | 133 | – | |
| | | $V_{IN} = 0\text{ V}$ PA7 | $V_{DD} = 5\text{V}$ | – | 60 | – | $\text{K}\Omega$ |
| | | | $V_{DD} = 3\text{V}$ | – | 133 | – | |

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

| Parameter | Condition | Min | Typ | Max | Unit |
|-----------------------|---|------|-----|------|------|
| Internal RC Frequency | 25°C , $V_{DD} = 3.0 \sim 5.5\text{V}$ | 3.88 | 4 | 4.12 | MHz |
| | 25°C , $V_{DD} = 2.6 \sim 3.0\text{V}$ | 3.8 | 4 | 4.2 | |
| | $-40^{\circ}\text{C} \sim 85^{\circ}\text{C}$, $V_{DD} = 2.6 \sim 5.5\text{V}$ | 3.72 | 4 | 4.28 | |

Note: The IRC frequency is trimmed in wafer type. After packaging or COB, the frequency will deviation range is about 10~20%.

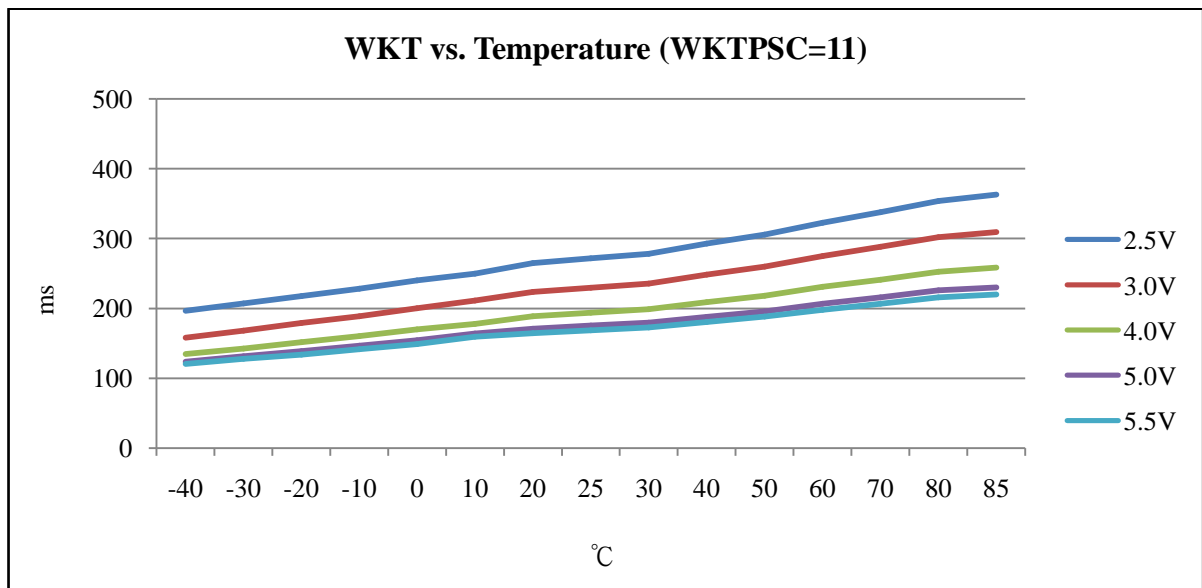
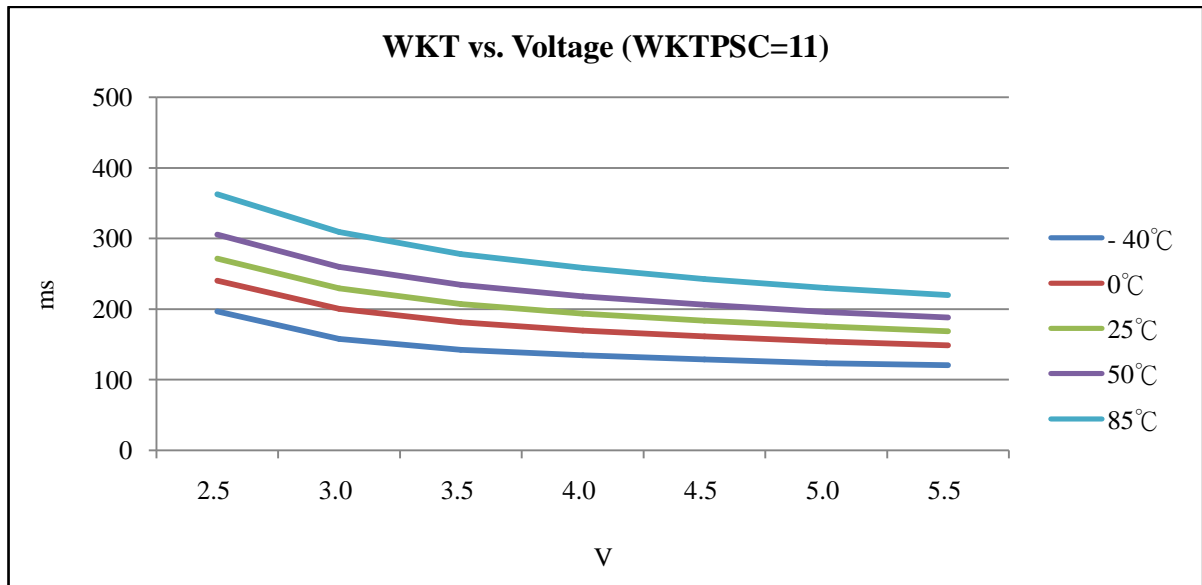
4. Reset Timing Characteristics ($T_A = -40^{\circ}\text{C}$ to $+85^{\circ}\text{C}$)

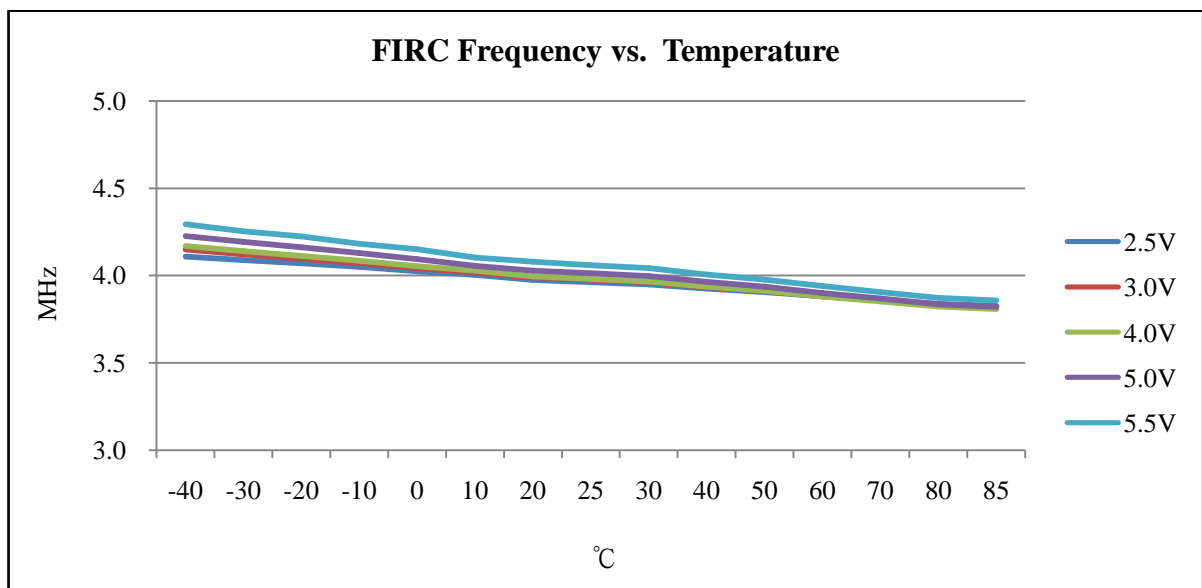
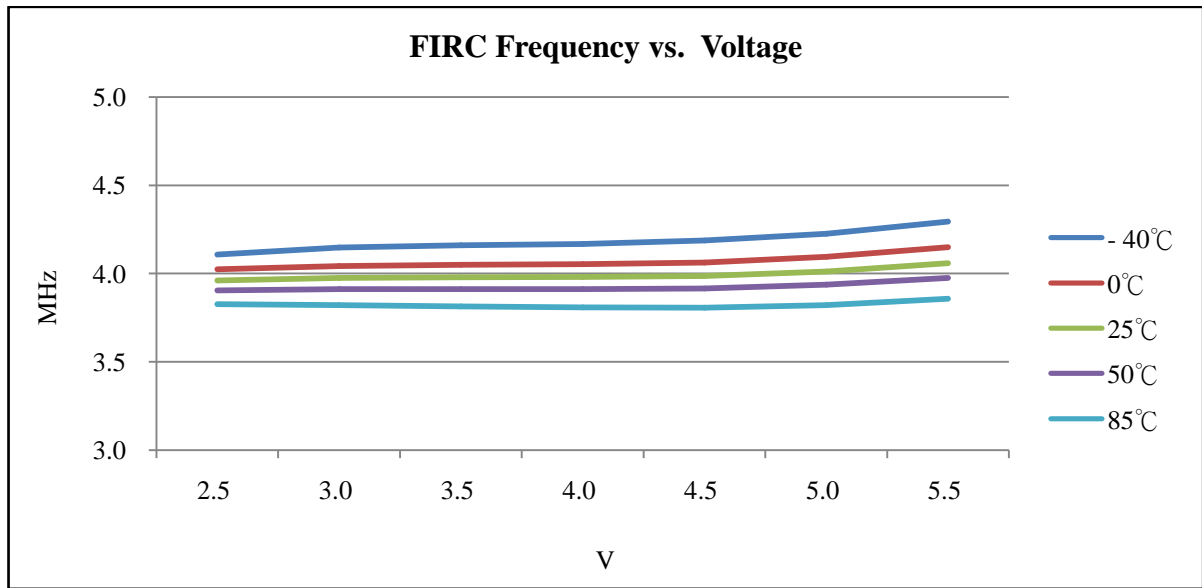
| Parameter | Conditions | Min | Typ | Max | Unit |
|-----------------------|-------------------------------------|-----|------|-----|---------------|
| RESET Input Low width | Input $V_{DD} = 5\text{V} \pm 10\%$ | 3 | – | – | μs |
| WDT wakeup time | $V_{DD} = 5\text{V}$, WKTPSC = 11 | – | 176 | – | ms |
| | $V_{DD} = 3\text{V}$, WKTPSC = 11 | – | 224 | – | |
| CPU start up time | $V_{DD} = 5\text{V}$ | – | 22.5 | – | ms |
| | $V_{DD} = 3\text{V}$ | – | 30.5 | – | |

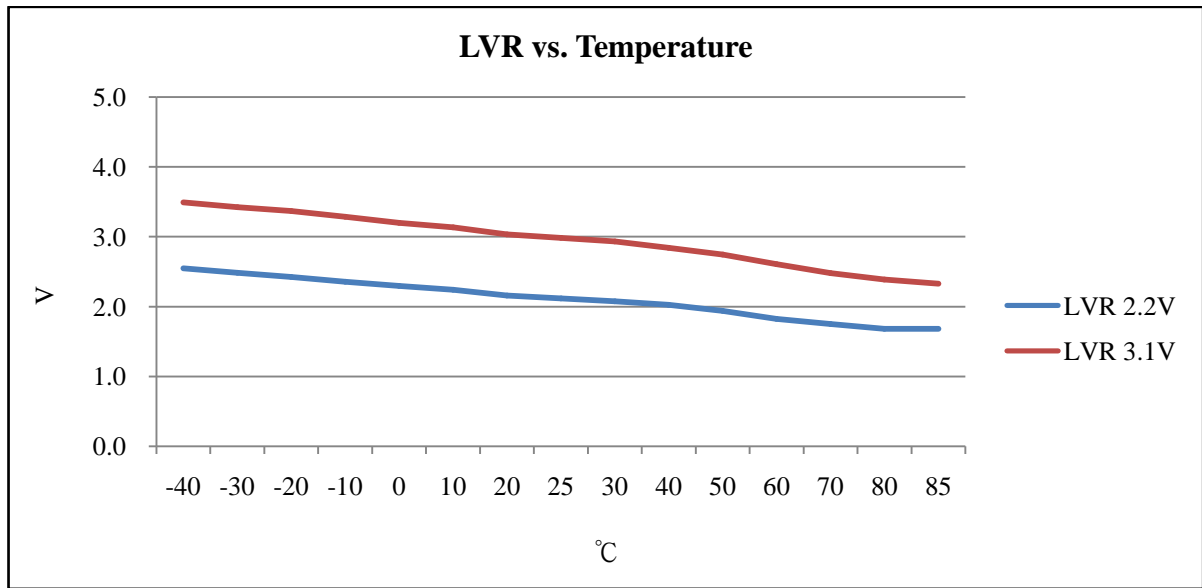
5. ADC Electrical Characteristics ($T_A = 25^{\circ}\text{C}$)

| Parameter | Conditions | Min | Typ | Max | Unit |
|--------------------------------------|---|----------|-----------|----------|---------------|
| Total Accuracy | $V_{DD} = 5\text{V}$, $V_{SS} = 0\text{V}$ | – | ± 2.5 | ± 5 | LSB |
| Internal Non-Linearity | | – | ± 3.2 | ± 6 | |
| Max Input Clock (f_{ADC}) | – | – | – | 2 | MHz |
| Conversion Time | $f_{\text{ADC}(\text{max})} = 2\text{MHz}$ | – | 25 | – | μs |
| Input Voltage | – | V_{SS} | – | V_{DD} | V |

6. Characteristic Graphs



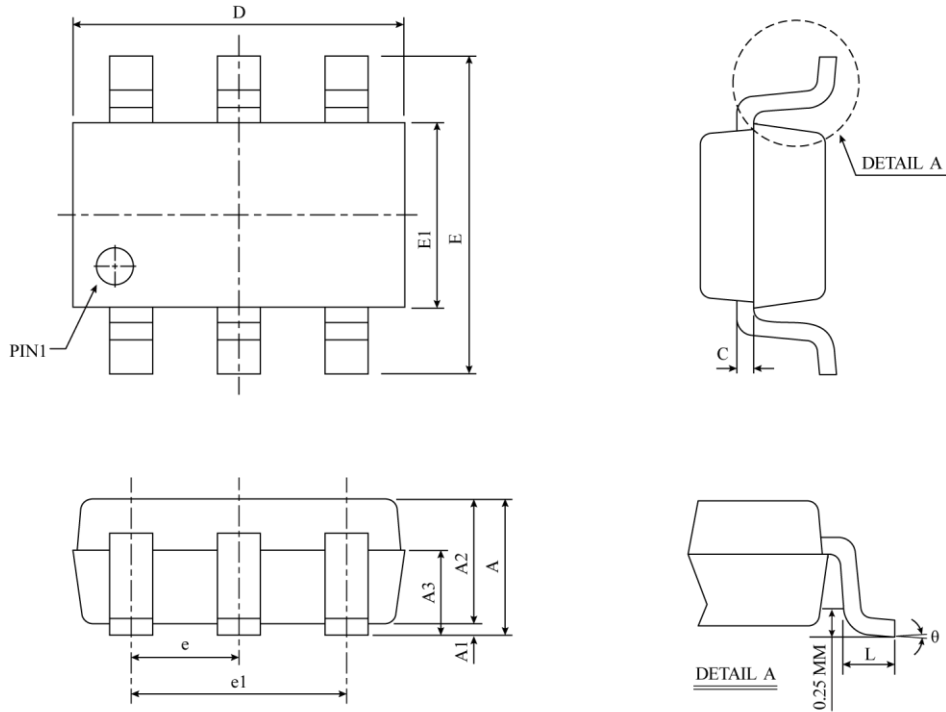




ORDERING INFORMATION

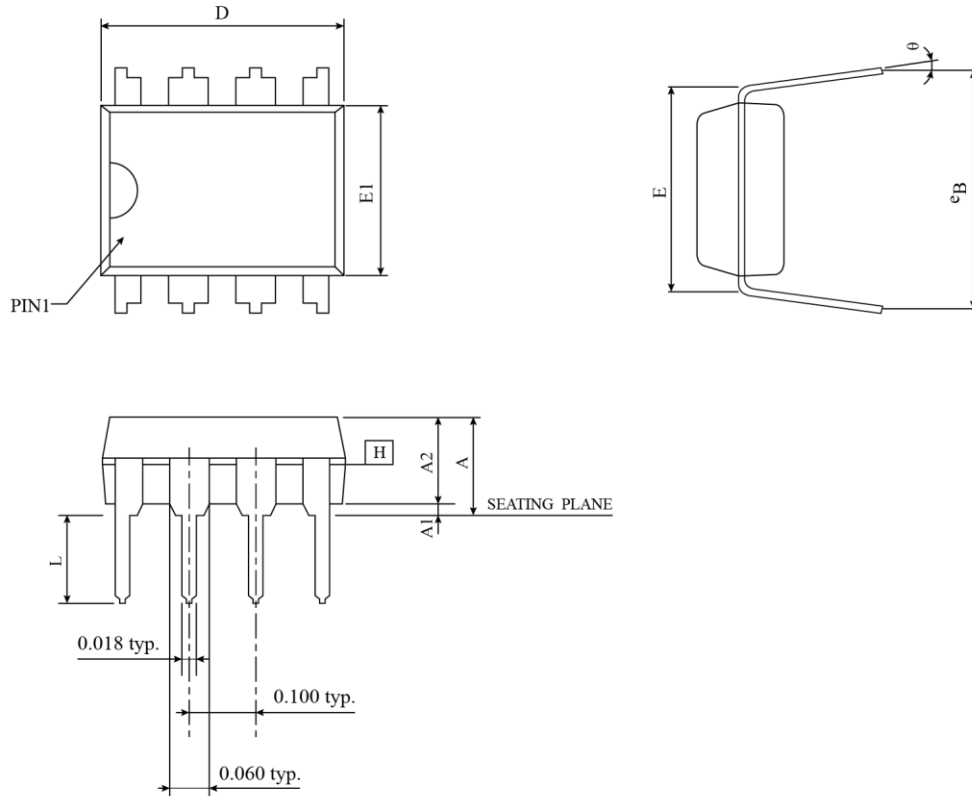
The ordering information:

| Ordering number | Package |
|-------------------|-------------------------|
| TM57PT16-OTP | Wafer / Dice blank chip |
| TM57PT16-COD | Wafer / Dice with code |
| TM57PT16AS-OTP-A8 | SOT23-6 |
| TM57PT16-OTP-01 | DIP-8 (300mil) |
| TM57PT16-OTP-14 | SOP-8 (150mil) |
| TM57PT16-OTP-52 | MSOP-8 (118mil) |
| TM57PT16B-OTP | Wafer / Dice blank chip |
| TM57PT16B-COD | Wafer / Dice with code |
| TM57PT16BS-OTP-A8 | SOT23-6 |
| TM57PT16B-OTP-01 | DIP-8 (300mil) |
| TM57PT16B-OTP-14 | SOP-8 (150mil) |
| TM57PT16B-OTP-52 | MSOP-8 (118mil) |
| TM57PA16-OTP | Wafer / Dice blank chip |
| TM57PA16-COD | Wafer / Dice with code |
| TM57PA16AS-OTP-A8 | SOT23-6 |
| TM57PA16-OTP-01 | DIP-8 (300mil) |
| TM57PA16-OTP-14 | SOP-8 (150mil) |
| TM57PA16-OTP-52 | MSOP-8 (118mil) |

SOT23-6 Package Dimension


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

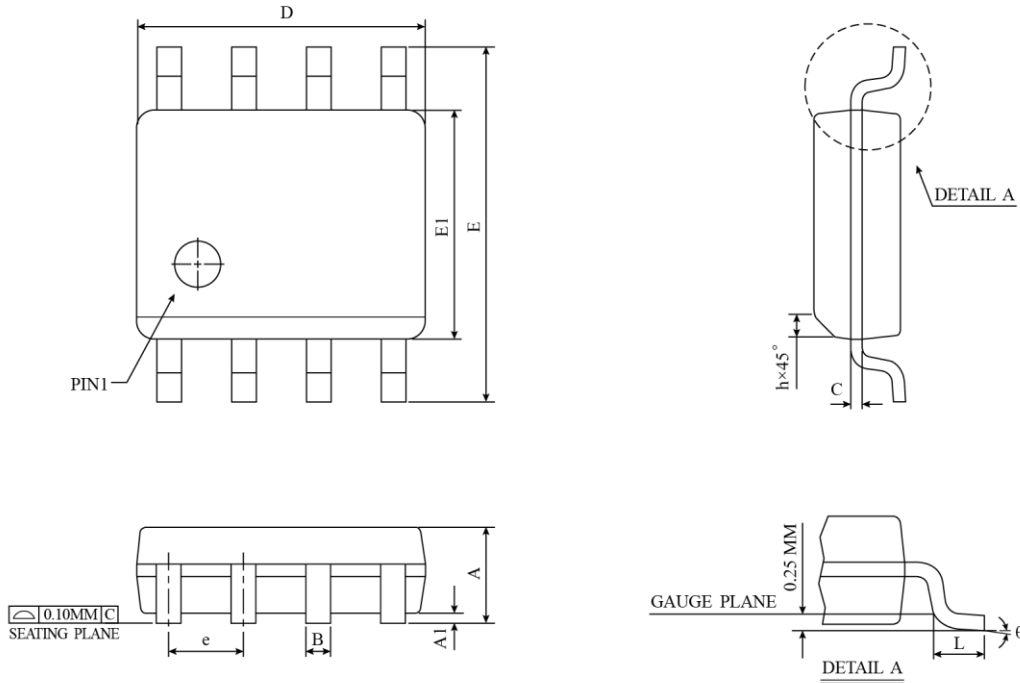
△ * NOTES : ALL DIMENSIONS REFER TO JEDEC STANDARD MO-178 AB
DO NOT INCLUDE MOLD FLASH OR PROTRUSIONS.

DIP-8 (300mil) Package Dimension


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

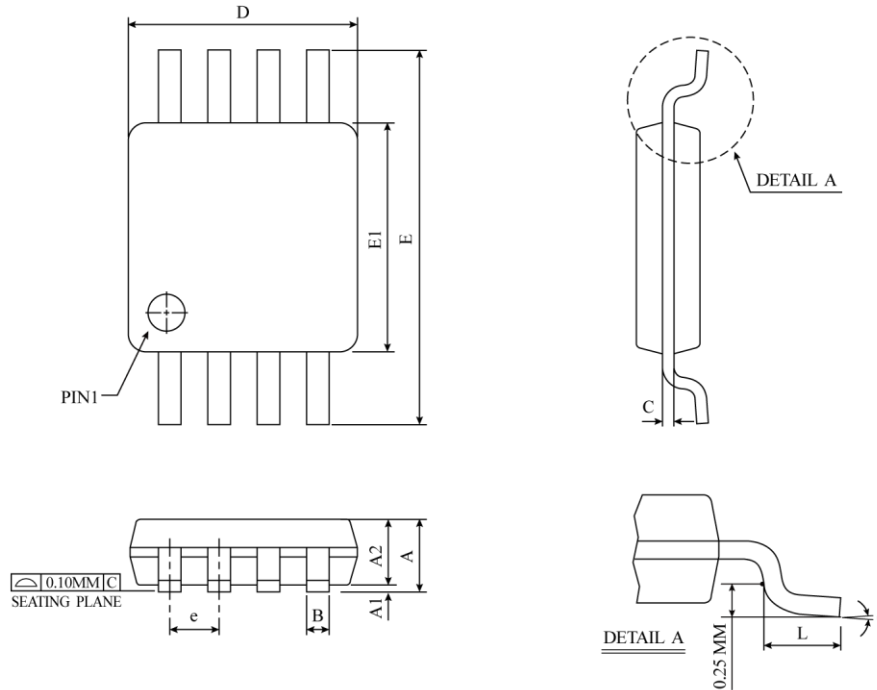
NOTES :

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

SOP-8 (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 | 0.004 | 0.0098 |
| B | 0.33 | 0.51 | 0.013 | 0.020 |
| C | 0.19 | 0.25 | 0.0075 | 0.0098 |
| D | 4.80 | 5.00 | 0.1890 | 0.1988 |
| 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 (AA) | | | |

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

MSOP-8 (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 | 0.76 | 0.86 | 0.95 | 0.030 | 0.034 | 0.037 |
| B | 0.28 | 0.33 | 0.38 | 0.011 | 0.013 | 0.015 |
| 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 | 0.65 BSC | | | 0.026 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 0.15 MM (0.006 INCH) PER SIDE.
 DIMENSION "E1" DOES NOT INCLUDE MOLD PROTRUSIONS
 MOLD PROTRUSIONS SHALL NOT EXCEED 0.25 MM (0.010 INCH) PER SIDE.