十速

# TM57PE11B/C

## *DATA SHEET*

## *Rev V1.2*

# AMENDMENT HISTORY

| Version | Date | Description |
|---|---|---|
| V1.0 | Aug, 2012 | New release |
| V1.1 | Jan, 2013 | 1. Add Low Voltage Detect description in Features section.<br>2. Modify Block Diagram.<br>3. Modify PA7 circuit in IO Port section.<br>4. Modify Internal RC Frequency in Electrical Characteristics section.<br>5. Modify CPU start up time in Electrical Characteristics section.<br>6. Modify Characteristics Graphs in Electrical Characteristics section.<br>7. Add SFR description.<br>8. Add TM57PE11C. |
| V1.2 | Jun, 2013 | 1. Add supported EV board on ICE.<br>2. Add Pin Summary. |

# CONTENTS

# FEATURES

1. ROM: 1K x 14 bits OTP or 512 x 14 bits TTP™ (Two Time Programmable ROM)
2. RAM: 48 x 8 bits
3. STACK: 5 Levels
4. I/O 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. Watchdog Timer
   - Clocked by built-in RC oscillator with 4 adjustable Reset/Interrupt Time (96 ms/48 ms/24 ms/12 ms, @5V; 128 ms/64 ms/32 ms/16 ms, @3V)
   - Watchdog timer can be disabled/enabled in STOP mode
7. Reset
   - Power On Reset
   - Watchdog Reset
   - Low Voltage Reset
   - External pin Reset
8. System Clock Mode
   - Internal RC: 4/8 MHz.
     - TM57PE11B: When the IRC is 8 MHz, the LVR can only be set to 3.1V (cannot use 2.2V).
     - TM57PE11C: When the IRC is 8 MHz, the LVR can only be set to 2.9V (cannot use 2.0V).
   - External RC
9. 2-Level Low Voltage Reset
   - TM57PE11B: 2.2V/3.1V (Can be disabled)
   - TM57PE11C: 2.0V/2.9V (Can be disabled)

| LVR / Freq | 2.2V/2.0V | 3.1V/2.9V | Disable |
|---|---|---|---|
| 4 MHz | ☑ | ☑ | ☑ |
| 8 MHz | ☒ | ☑ | ☑ |

10. 2-Level Low Voltage Detect
    - TM57PE11B: 2.3V/3.2V (Can be disabled)
    - TM57PE11C: 2.1V/3.0V (Can be disabled)
11. Operation Voltage: Low Voltage Reset Level to 5.5V
    - Fsys = 4 MHz, 2.0V ~ 5.5V
    - Fsys = 8 MHz, 2.5V ~ 5.5V

**12.** I/O Port

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

**13.** Instruction set: 36 Instructions

**14.** Interrupt

- Two External Interrupt pins:
  - One pin is falling edge triggered
  - One pin is rising or falling edge triggered
- TM0, Wake-up Timer Interrupt

**15.** PA1~PA4 individual pin low level wake up

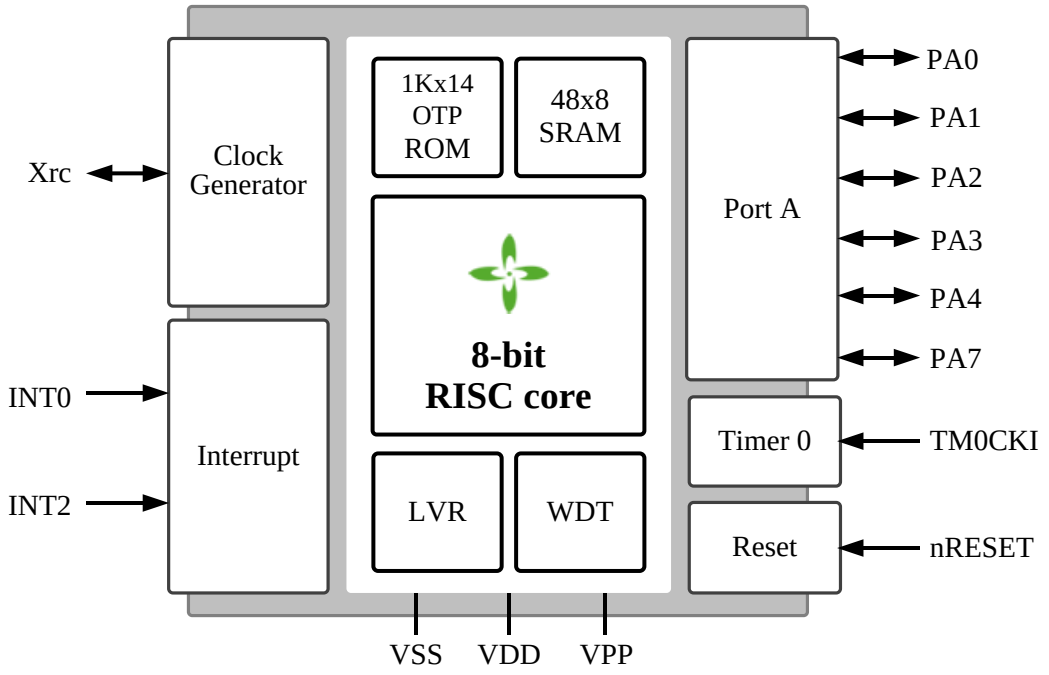
**16.** STOP Mode Support

**17.** Support 5-wire program

**18.** Supported EV Board on ICE

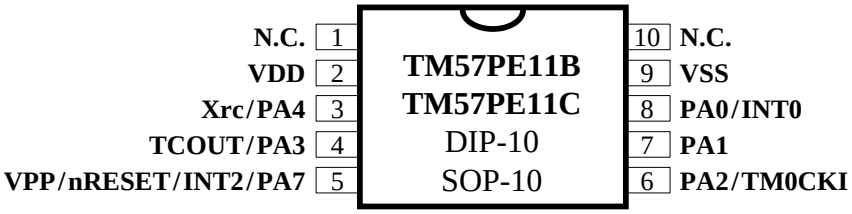
EV Board: EV2780

# BLOCK DIAGRAM




**TM57PE11B/TM57PE11C 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 |
| Xrc | – | External RC oscillator connection for system clock |
| TCOUT | O | Instruction cycle clock output. The instruction clock frequency is system clock frequency divided by two (Fsys/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 |

# PIN SUMMARY

| Pin Number | Pin Name | Type | GPIO | | | | | Function After Reset | Alternate Function | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | Input | | Output | | | | | | | |
| 10-SOP/DIP | | | Weak Pull-up | Ext. Interrupt | O.D | P.O.D | P.P | | PWM | Touch Key | ADC | MISC |
| 1 | VDD | P | | | | | | | | | | |
| 2 | PA4/Xrc | I/O | O | | O | | O | PA4 | | | | |
| 3 | PA3/TCOUT | I/O | O | | O | | O | PA3 | | | | TCOUT |
| 4 | PA7/INT2/nRESET/VPP | I/O | O | O | O | | | PA7 | | | | nRESET |
| 5 | NC | - | | | | | | | | | | |
| 6 | NC | - | | | | | | | | | | |
| 7 | PA2/TM0CKI | I/O | O | | O | | O | PA2 | | | | TM0CKI |
| 8 | PA1 | I/O | O | | | O | O | PA1 | | | | |
| 9 | PA0/INT0 | I/O | O | O | | O | O | PA0 | | | | |
| 10 | VSS | P | | | | | | | | | | |

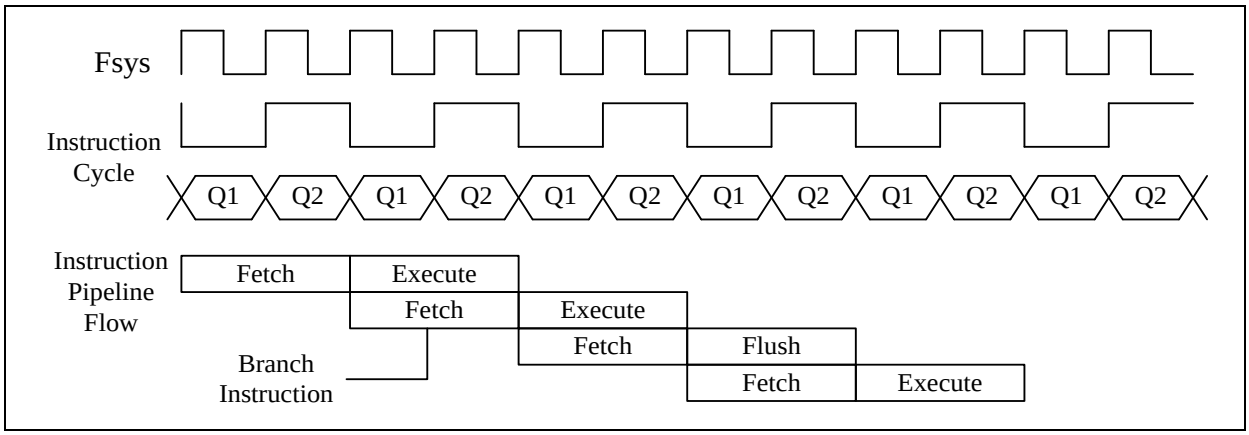
Symbol：P.P. = Push-Pull Output
P.O.D. = Pseudo Open Drain
O.D. = Open Drain

# FUNCTIONAL DESCRIPTION

## 1. CPU Core

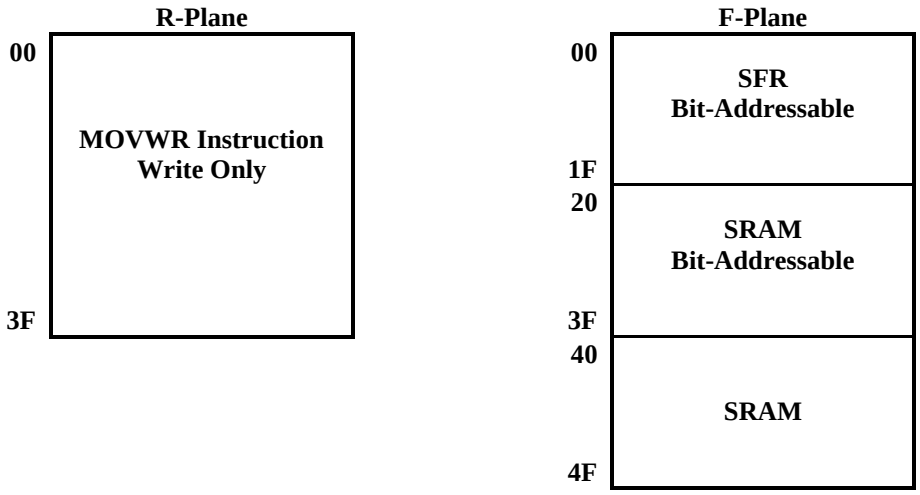
### 1.1 Clock Scheme and Instruction Cycle

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.




## 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.

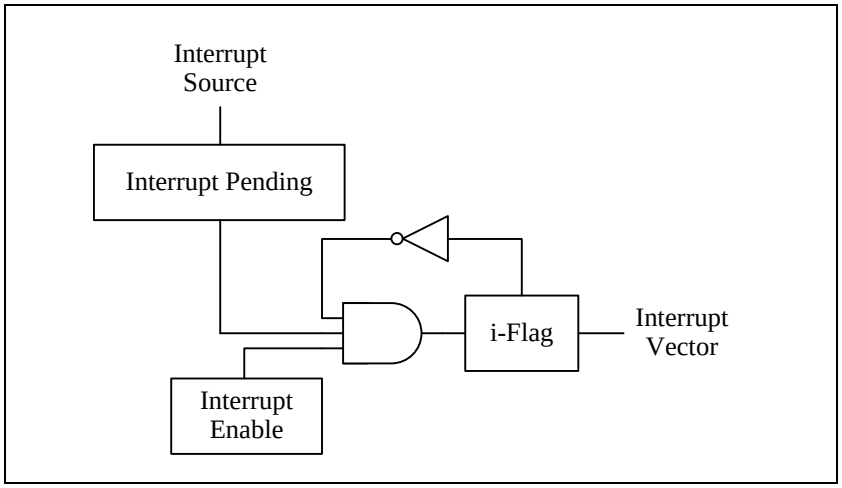
<table>
<tr><th>STATUS</th><th>Bit 7</th><th>Bit 6</th><th>Bit 5</th><th>Bit 4</th><th>Bit 3</th><th>Bit 2</th><th>Bit 1</th><th>Bit 0</th></tr>
<tr><th>Reset Value</th><td>0</td><td>–</td><td>–</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr>
<tr><th>R/W</th><td>R</td><td>–</td><td>–</td><td>R</td><td>R</td><td>R/W</td><td>R/W</td><td>R/W</td></tr>
<tr><td>Bit</td><td colspan="8">Description</td></tr>
<tr><td>7</td><td colspan="8">LVD: Low Voltage Detector Flag<br>TM57PE11B: LVD threshold is 2.3V/3.2V when LVR is 2.2V/3.1V.<br>TM57PE11C: LVD threshold is 2.1V/3.0V when LVR is 2.0V/2.9V.<br>0: $V_{DD}$ voltage is more than LVD threshold, or LVR is disabled<br>1: $V_{DD}$ voltage is less than LVD threshold</td></tr>
<tr><td>6~5</td><td colspan="8">Not Used</td></tr>
<tr><td>4</td><td colspan="8">TO: Time Out Flag<br>0: after Power On Reset, LVR Reset, or CLRWDT/SLEEP instruction<br>1: WDT time out occurs</td></tr>
<tr><td>3</td><td colspan="8">PD: STOP Flag<br>0: after Power On Reset, LVR Reset, or CLRWDT instruction<br>1: after SLEEP instruction</td></tr>
<tr><td>2</td><td colspan="8">Z: Zero Flag<br>0: the result of a logic operation is not zero<br>1: the result of a logic operation is zero</td></tr>
<tr><td rowspan="3">1</td><td colspan="8">DC: Decimal Carry Flag or Decimal /Borrow Flag</td></tr>
<tr><td colspan="4">ADD instruction</td><td colspan="4">SUB instruction</td></tr>
<tr><td colspan="4">0: no carry<br>1: a carry from the low nibble bits of the result occurs</td><td colspan="4">0: a borrow from the low nibble bits of the result occurs<br>1: no borrow</td></tr>
<tr><td rowspan="3">0</td><td colspan="8">C: Carry Flag or /Borrow Flag</td></tr>
<tr><td colspan="4">ADD instruction</td><td colspan="4">SUB instruction</td></tr>
<tr><td colspan="4">0: no carry<br>1: a carry occurs from the MSB</td><td colspan="4">0: a borrow occurs from the MSB<br>1: no borrow</td></tr>
</table>

## 1.6 Interrupt

The TM57PE11B/TM57PE11C has 1 level, 1 vector and 4 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 TM57PE11B/TM57PE11C 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.




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

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 | – | – | – | TM0IF | WKTIF | INT2IF | – | INT0IF |
| R/W | – | – | – | R/W | R/W | R/W | – | R/W |
| Reset | – | – | – | 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

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 | – | WKTPSC | |
| 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 TM57PE11B/TM57PE11C 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.

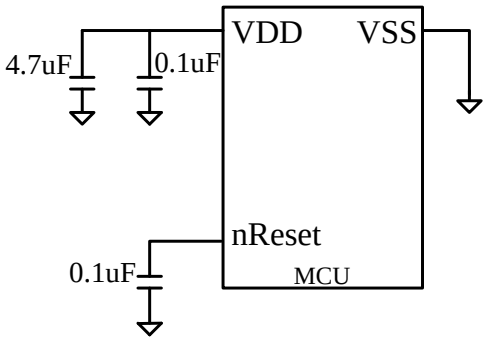
LVR Selection Table:

| LVR Threshold Level | Consider the operating voltage to choose LVR |
|---|---|
| LVR3.1 (TM57PE11B)<br>LVR2.9 (TM57PE11C) | 5.5V > $V_{DD}$ > 3.3V |
| LVR2.2 (TM57PE11B)<br>LVR2.0 (TM57PE11C) | $V_{DD}$ 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.



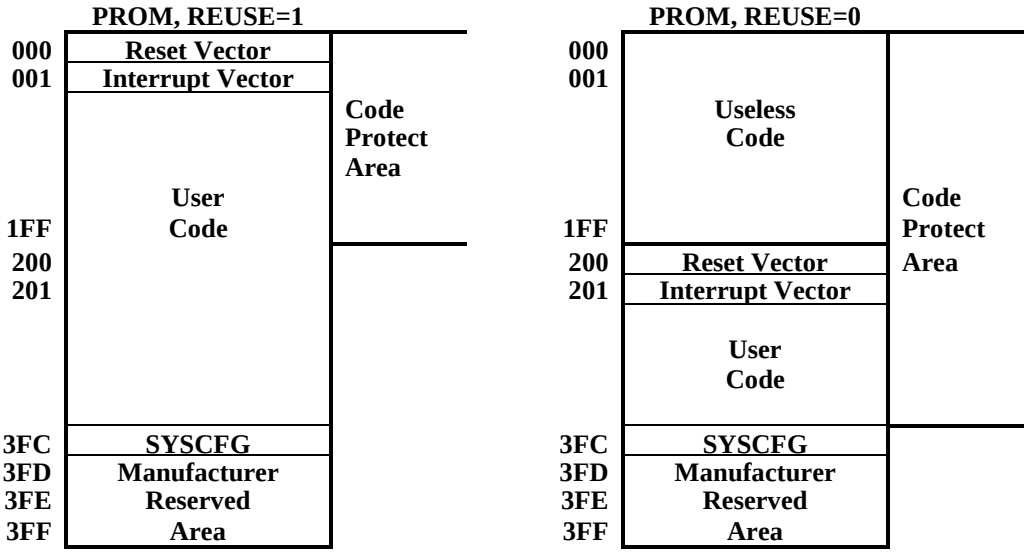
### 2.3 System Configuration Register (SYSCFG)

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** | **11111111111111** | |
| 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, LVD Disable |
| | 01 | TM57PE11B: LVR = 3.1V, LVD = 3.2V, always enable<br>TM57PE11C: LVR = 2.9V, LVD = 3.0V, always enable |
| | 10 | TM57PE11B: LVR = 2.2V, LVD = 2.3V, disable in STOP mode<br>TM57PE11C: LVR = 2.0V, LVD = 2.1V, disable in STOP mode |
| | 11 | TM57PE11B: LVR = 2.2V, LVD = 2.3V, always enable<br>TM57PE11C: LVR = 2.0V, LVD = 2.1V, always enable |
| 9-8 | **CLKT**: Clock Source Type | |
| | 00 | External RC |
| | 01 | Internal RC (4/8 MHz) |
| | 1X | Invalid |
| 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<br>TM57PE11B: When the IRC is 8 MHz, the LVR can only be set to 3.1V or disable (can't use 2.2V).<br>TM57PE11C: When the IRC is 8 MHz, the LVR can only be set to 2.9V or disable (can't use 2.0V). | |
| 4-0 | **Tenx 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=0 and REUSE=1, the Code protection area is first half of PROM. This allows the Writer tool to write then verify the Code during the Re-use Code programming. After the Re-use Code being written into the PROM's second half, user should write "REUSE" control bit to "0". In the mean while, the Code protection area becomes the whole PROM except the Reserved Area.




## 2.5 STOP Mode

The STOP mode is activated by "SLEEP" instruction. During the STOP mode, the system clock and peripherals stops to minimize power consumption, while the WDT/WKT Timer is working or not depends on F/W setting. The STOP mode can be terminated by Reset, or enabled Interrupts (External pins and WKT interrupts) or PA1-4 pins low level wake up.

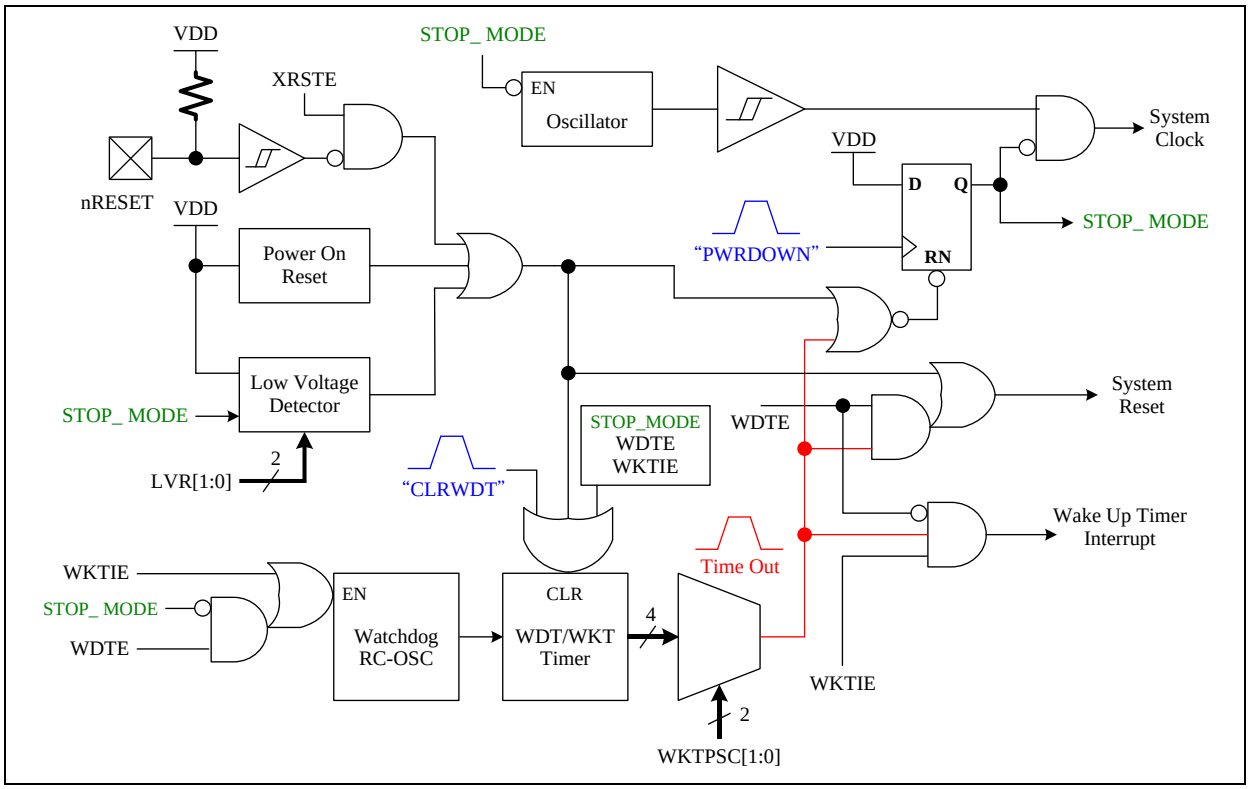
| R03 | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| PWRDN | PWRDN | | | | | | | |
| R/W | W | | | | | | | |
| Reset | – | – | – | – | – | – | – | – |

R03.7~0 **PWRDN:** Write this register to enter Power Down (STOP) 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 12 ms to 128 ms. The WDT/WKT is cleared by the CLRWDT instruction. If the Watchdog Reset is enabled (WDTE=1), the WDT generates the chip reset signal, otherwise, the WKT only generates overflow time out interrupt. The WDT/WKT works in both normal mode and STOP mode. During STOP mode, user can further choose to enable or disable the WDT/WKT by "WKTIE". If WKTIE=0 in STOP mode (no matter WDTE is 1 or 0), the internal RC Timer stops for power saving. In other words, user keeps the WDT/WKT alive in STOP Mode by setting WKTIE=1. If the WDTE=1 and WKTIE=0, WDT/WKT timer will be cleared and stopped to power saving in STOP mode. If the WDTE=1 and WKTIE=1, WDT/WKT timer keeps counting in STOP/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 | Watchdog_RC Oscillator |
|---|---|---|---|
| Normal Mode | 0 | 0 | Stop |
| | 0 | 1 | Run |
| | 1 | 0 | |
| | 1 | 1 | |
| STOP Mode | 0 | 0 | Stop |
| | 0 | 1 | Run |
| | 1 | 0 | Stop |
| | 1 | 1 | Run |

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

F03.4 **TO:** WDT time out flag

| F08 | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| INTIE | – | – | – | TM0IE | WKTIE | INT2IE | – | INT0IE |
| R/W | – | – | – | R/W | R/W | R/W | – | R/W |
| Reset | – | – | – | 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 | – | – | – | TM0IF | WKTIF | INT2IF | – | INT0IF |
| R/W | – | – | – | R/W | R/W | R/W | – | R/W |
| Reset | – | – | – | 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

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

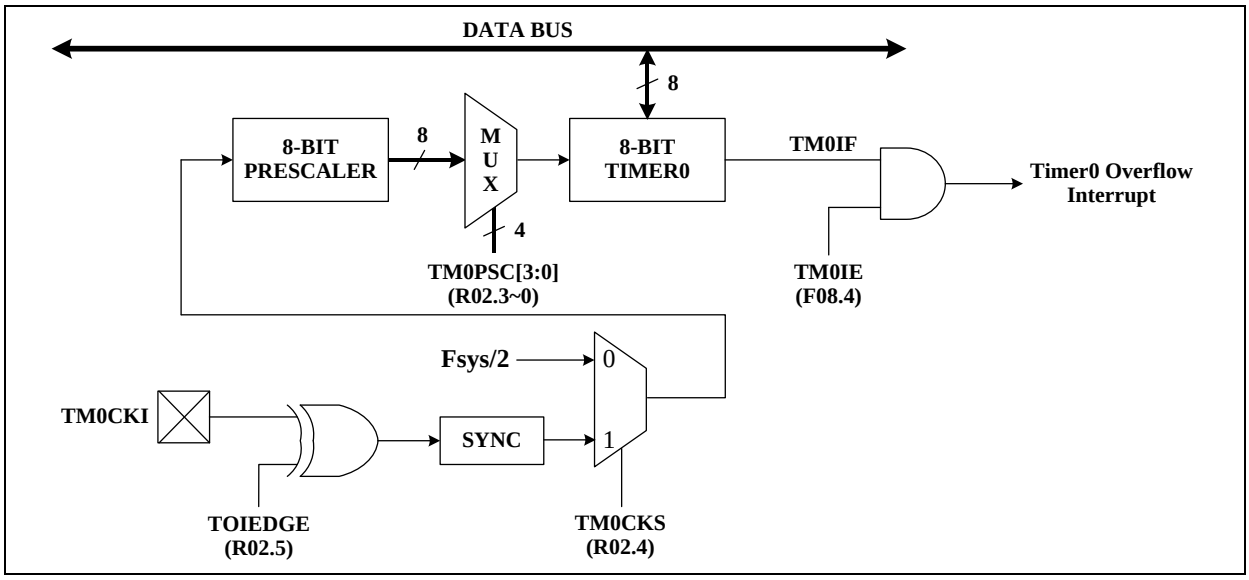
R04.7~0 **WDTCLR:** Write this register to clear WDT/WKT

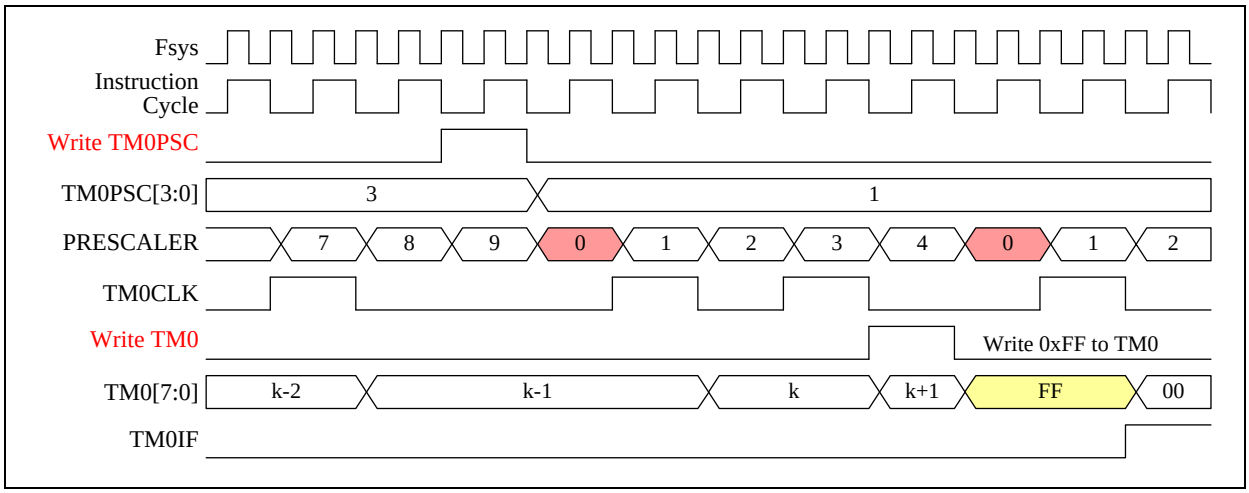
| R0B | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| MR0B | – | – | – | INT0EDG | TCOE | – | WKTPSC | |
| R/W | – | – | – | W | W | – | W | |
| Reset | – | – | – | 0 | 0 | – | 1 | 1 |

R0B.1~0 **WKTPSC:** WDT/WKT pre-scale option select
00: WDT/WKT period is 12 ms, @5V; 16 ms, @3V
01: WDT/WKT period is 24 ms, @5V; 32 ms, @3V
10: WDT/WKT period is 48 ms, @5V; 64 ms, @3V
11: WDT/WKT period is 96 ms, @5V; 128 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 TM0CKI (PA2) input. The Timer0 increase rate is determined by "Timer0 Prescale" (TM0PSC) register in R-Plane. The Timer0 can generate interrupt flag (TM0IF) when it rolls over.




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

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

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

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

Timer0 interrupt frequency by TM0CKI: (TM0CKI) / div / 256

**Note: TM0CKI frequency <= Fsys / 4**

| 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 | – | – | – | TM0IE | WKTIE | INT2IE | – | INT0IE |
| R/W | – | – | – | R/W | R/W | R/W | – | R/W |
| Reset | – | – | – | 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 | – | – | – | TM0IF | WKTIF | INT2IF | – | INT0IF |
| R/W | – | – | – | R/W | R/W | R/W | – | R/W |
| Reset | – | – | – | 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 |

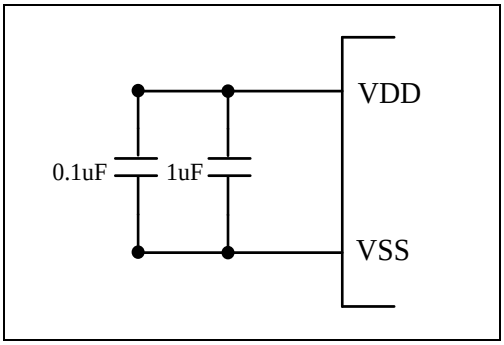
R02.5 **TM0EDG:** TM0CKI (PA2) edge selection for Timer0 Prescaler count
0: TM0CKI (PA2) rising edge for Timer0 Prescaler count
1: TM0CKI (PA2) falling edge for Timer0 Prescaler count

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

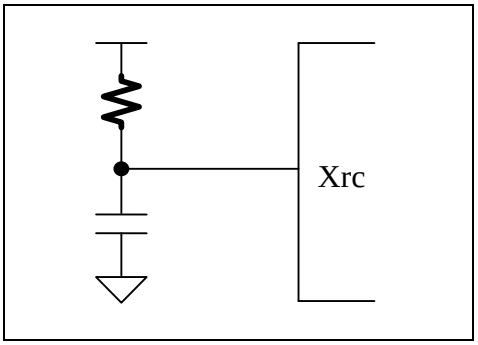
R02.3~0 **TM0PSC:** Timer0 Prescale
0000: divided by 1
0001: divided by 2
0010: divided by 4
0011: divided by 8
0100: divided by 16
0101: divided by 32
0110: divided by 64
0111: divided by 128
1xxx: divided by 256

### 3.3 System Clock Oscillator

System clock can be operated in two different oscillation modes, which is selected by setting the CLKT in the SYSCFG register. In external RC mode, the external resistor and capacitor determine the oscillation frequency. In the 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/2.9V (cannot use 2.2V/2.0V). 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

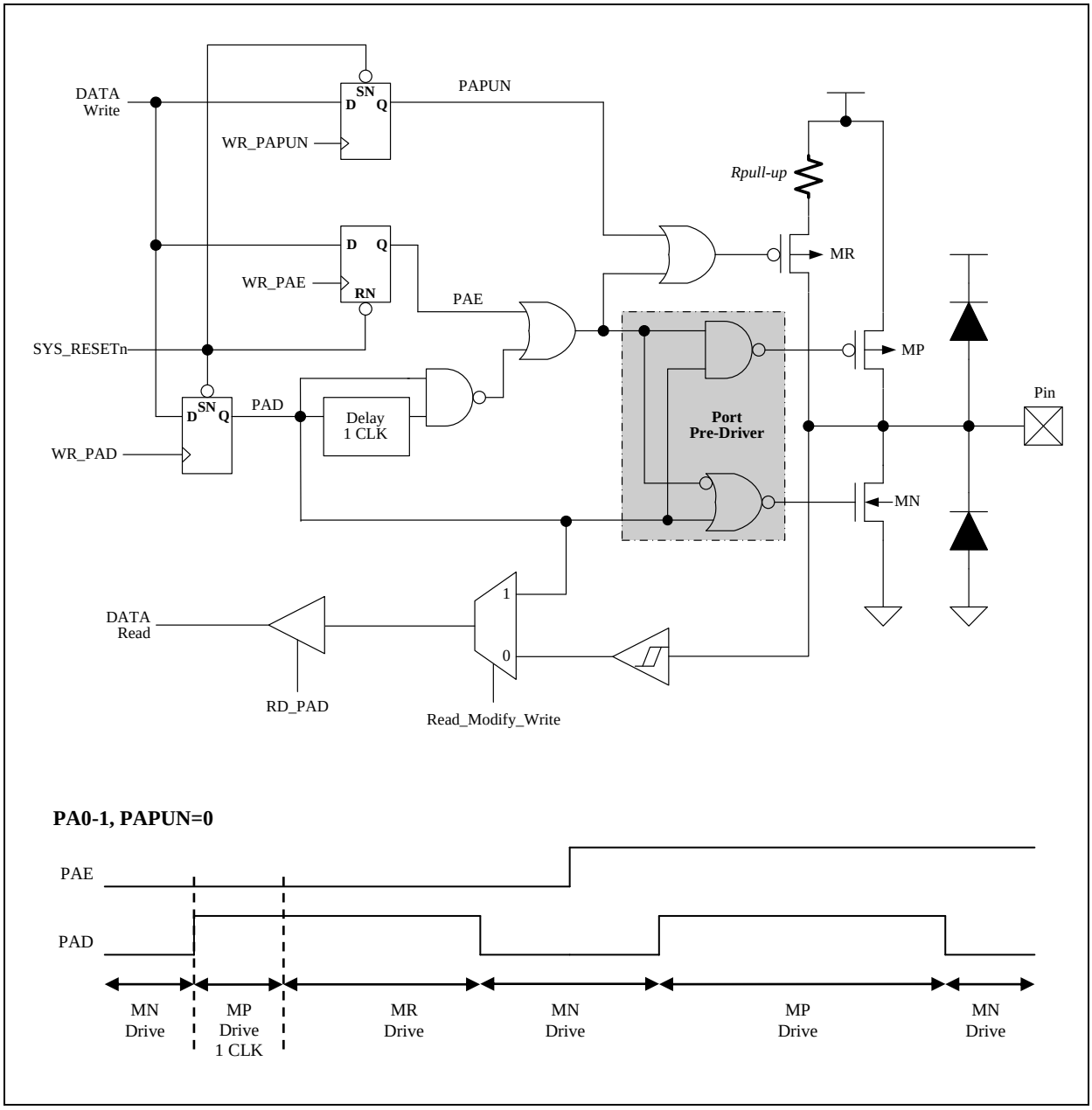



External RC Oscillator

# 4. I/O Port

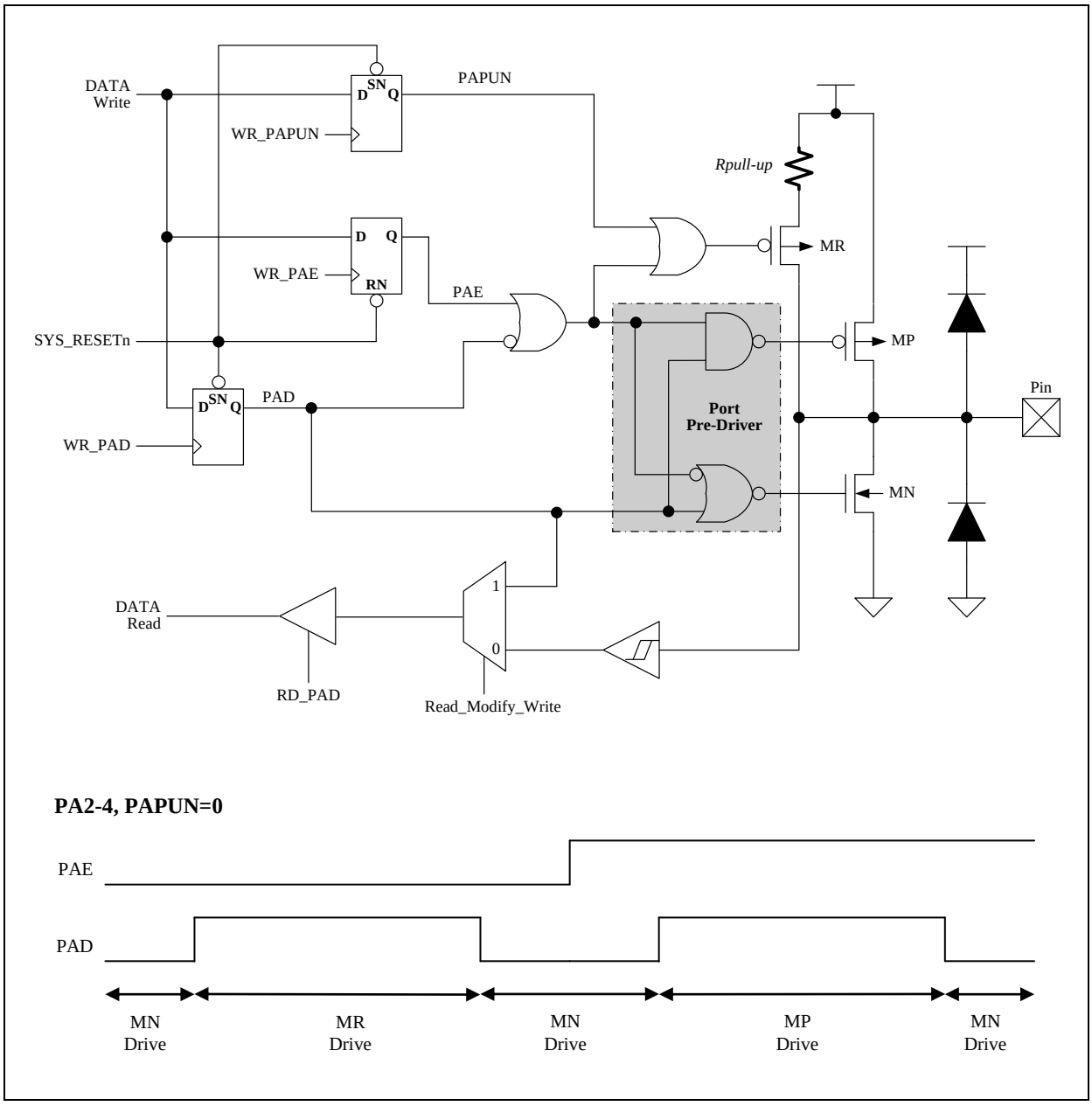
## 4.1 PA0-1

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

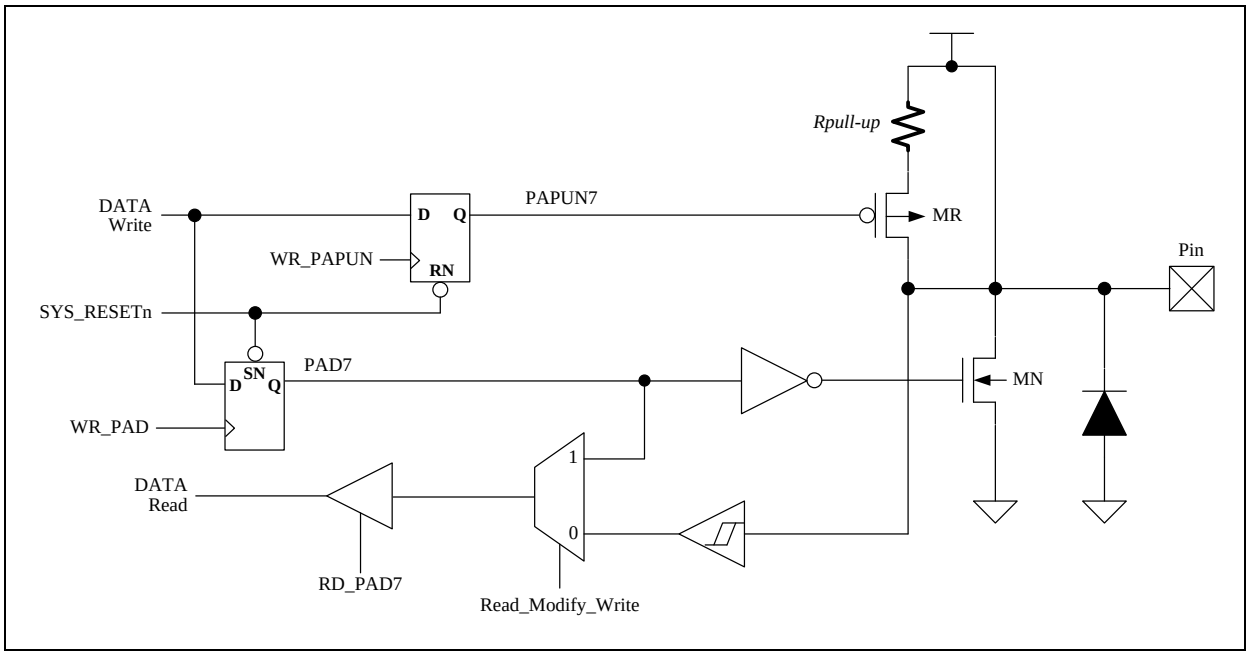
These pins are almost the same as PA0-1, except they do not support pseudo-open-drain mode. They can be used in pure open-drain mode, instead.




**PA2-4, PAPUN=0**

### 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)
c: the pin is working for external RC oscillation
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 | 0 |

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

# MEMORY MAP

## F-Plane

| Name | Address | R/W | Rst | Description |
|---|---|---|---|---|
| **(F00) INDF** | | | | Function related to: RAM W/R |
| **INDF** | **00.7~0** | R/W | - | Not a physical register, addressing INDF actually point to the register whose address is contained in the FSR register |
| **(F01) TM0** | | | | Function related to: Timer0 |
| **TM0** | **01.7~0** | R/W | 0 | Timer0 content |
| **(F02) PCL** | | | | Function related to: Programming Counter |
| **PCL** | **02.7~0** | R/W | 0 | Programming Counter LSB[7~0] |
| **(F03) STATUS** | | | | Function related to: Status |
| **LVD** | **03.7** | R | 0 | Low voltage detector flag |
| **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 |
| **C** | **03.0** | R/W | 0 | Carry 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<br>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 |
| **TM0IE** | **08.4** | R/W | 0 | Timer0 interrupt enable<br>0: disable<br>1: enable |
| **WKTIE** | **08.3** | R/W | 0 | Wakeup Timer interrupt enable<br>0: disable<br>1: enable |
| **INT2IE** | **08.2** | R/W | 0 | INT2 (PA7) falling interrupt enable<br>0: disable<br>1: enable |
| **INT0IE** | **08.0** | R/W | 0 | INT0 (PA0) falling/rising interrupt enable<br>0: disable<br>1: enable |

| Name | Address | R/W | Rst | Description |
|---|---|---|---|---|
| **(F09) INTIF** | | | | Function related to: Interrupt Flag |
| **TM0IF** | 09.4 | R | - | Timer0 interrupt event pending flag, set by H/W while Timer0 overflows |
| | | W | 0 | 0: clear this flag<br>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<br>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<br>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<br>1: no action |
| **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 | 0: TM0CKI (PA2) rising edge for Timer0 Prescaler count<br>1: TM0CKI (PA2) falling edge for Timer0 Prescaler count |
| **TM0CKS** | 02.4 | W | 0 | 0: Instruction Cycle as Timer0 Prescaler clock<br>1: TM0CKI as Timer0 Prescaler clock |
| **TM0PSC** | 02.3~0 | W | 0 | Timer0 Pre-Scale<br>0000: divided by 1<br>0001: divided by 2<br>0010: divided by 4<br>0011: divided by 8<br>0100: divided by 16<br>0101: divided by 32<br>0110: divided by 64<br>0111: divided by 128<br>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 | Each bit controls its corresponding pin, if the bit is<br>0: the pin is open-drain output or Schmitt-trigger input<br>1: the pin is CMOS push-pull output |
| | 05.1~0 | W | 0 | Each bit controls its corresponding pin, if the bit is<br>0: the pin is pseudo-open-drain output or Schmitt-trigger input<br>1: the pin is CMOS push-pull output |
| **(R08) PAPUN** | | | | Function related to: Port A |
| **PAPUN7** | 08.7 | W | 0 | 0: the pin pull-up resistor is enable<br>1: the pin pull-up resistor is disable |
| **PAPUN** | 08.4~0 | W | 1F | 0: the pin pull-up resistor is enable, except<br>a. the pin's output data register (PAD) is 0<br>b. the pin's CMOS push-pull mode is chosen (PAE=1)<br>c. the pin is working for external RC oscillation<br>1: the pin pull-up resistor is disable |
| **(R0B) MR0B** | | | | Function related to: INT0 / TCOUT / WKT / WDT |
| **INT0EDG** | 0b.4 | W | 0 | 0: INT0 (PA0) pin falling edge to trigger interrupt event<br>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) |
| **WKTPSC** | 0b.1~0 | W | 11 | WDT/WKT pre-scale option select<br>00: WDT/WKT period is 12 ms, @5V; 16 ms, @3V<br>01: WDT/WKT period is 24 ms, @5V; 32 ms, @3V<br>10: WDT/WKT period is 48 ms, @5V; 64 ms, @3V<br>11: WDT/WKT period is 96 ms, @5V; 128 ms, @3V |
| **(R13) PAWKEN** | | | | Function related to: Port A |
| **PAWKEN** | 13.4~1 | W | 0 | PA4~PA1 pin low level wake up enable<br>0: disable<br>1: enable |

# INSTRUCTION SET

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

For byte-oriented instructions, “f” or “r” represents the address designator and “d” represents the destination designator. The address designator is used to specify which address in Program memory is 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** | | | | | |
| ADDWF | f,d | 00 0111 dfff ffff | 1 | C,DC,Z | Add W and "f" |
| ANDWF | f,d | 00 0101 dfff ffff | 1 | Z | AND W with "f" |
| CLRF | f | 00 0001 1fff ffff | 1 | Z | Clear "f" |
| CLRW | | 00 0001 0100 0000 | 1 | Z | Clear W |
| COMF | f,d | 00 1001 dfff ffff | 1 | Z | Complement "f" |
| DECF | f,d | 00 0011 dfff ffff | 1 | Z | Decrement "f" |
| DECFSZ | f,d | 00 1011 dfff ffff | 1 or 2 | - | Decrement "f", skip if zero |
| INCF | f,d | 00 1010 dfff ffff | 1 | Z | Increment "f" |
| INCFSZ | f,d | 00 1111 dfff ffff | 1 or 2 | - | Increment "f", skip if zero |
| IORWF | f,d | 00 0100 dfff ffff | 1 | Z | OR W with "f" |
| MOVFW | f | 00 1000 0fff ffff | 1 | - | Move "f" to W |
| MOVWF | f | 00 0000 1fff ffff | 1 | - | Move W to "f" |
| MOVWR | r | 00 0000 00rr rrrr | 1 | - | Move W to "r" |
| RLF | f,d | 00 1101 dfff ffff | 1 | C | Rotate left "f" through carry |
| RRF | f,d | 00 1100 dfff ffff | 1 | C | Rotate right "f" through carry |
| SUBWF | f,d | 00 0010 dfff ffff | 1 | C,DC,Z | Subtract W from "f" |
| SWAPF | f,d | 00 1110 dfff ffff | 1 | - | Swap nibbles in "f" |
| TESTZ | f | 00 1000 1fff ffff | 1 | Z | Test if "f" is zero |
| XORWF | f,d | 00 0110 dfff ffff | 1 | Z | XOR W with "f" |
| **Bit-Oriented File Register Instruction** | | | | | |
| BCF | f,b | 01 000b bbff ffff | 1 | - | Clear "b" bit of "f" |
| BSF | f,b | 01 001b bbff ffff | 1 | - | Set "b" bit of "f" |
| BTFSC | f,b | 01 010b bbff ffff | 1 or 2 | - | Test "b" bit of "f", skip if clear |
| BTFSS | f,b | 01 011b bbff ffff | 1 or 2 | - | Test "b" bit of "f", skip if set |
| **Literal and Control Instruction** | | | | | |
| ADDLW | k | 01 1100 kkkk kkkk | 1 | C,DC,Z | Add Literal "k" and W |
| ANDLW | k | 01 1011 kkkk kkkk | 1 | Z | AND Literal "k" with W |
| CALL | k | 10 kkkk kkkk kkkk | 2 | - | Call subroutine "k" |
| CLRWDT | | 00 0000 0000 0100 | 1 | TO,PD | Clear Watch Dog Timer |
| GOTO | k | 11 kkkk kkkk kkkk | 2 | - | Jump to branch "k" |
| IORLW | k | 01 1010 kkkk kkkk | 1 | Z | OR Literal "k" with W |
| MOVLW | k | 01 1001 kkkk kkkk | 1 | - | Move Literal "k" to W |
| NOP | | 00 0000 0000 0000 | 1 | - | No operation |
| RET | | 00 0000 0100 0000 | 2 | - | Return from subroutine |
| RETI | | 00 0000 0110 0000 | 2 | - | Return from interrupt |
| RETLW | k | 01 1000 kkkk kkkk | 2 | - | Return with Literal in W |
| SLEEP | | 00 0000 0000 0011 | 1 | TO,PD | Go into STOP mode, Clock oscillation stops |
| XORLW | k | 01 1111 kkkk kkkk | 1 | Z | XOR Literal "k" with W |

## ADDLW — Add Literal "k" and W

| | |
|---|---|
| Syntax | ADDLW k |
| Operands | k : 00h ~ FFh |
| Operation | (W) ← (W) + k |
| Status Affected | C, DC, Z |
| OP-Code | 01 1100 kkkk kkkk |
| Description | The contents of the W register are added to the eight-bit literal 'k' and the result is placed in the W register. |
| Cycle | 1 |
| Example | ADDLW 0x15 — B : W = 0x10; A : W = 0x25 |

## ADDWF — Add W and "f"

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

## ANDLW — Logical AND Literal "k" with W

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

## ANDWF — AND W with "f"

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

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

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

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

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

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

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

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

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

## COMF — Complement "f"

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

## DECF — Decrement "f"

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

## DECFSZ — Decrement "f", Skip if 0

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

## GOTO — Unconditional Branch

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

## INCF — Increment "f"

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

## INCFSZ — Increment "f", Skip if 0

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

## IORLW — Inclusive OR Literal with W

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

## IORWF — Inclusive OR W with "f"

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

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

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

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

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

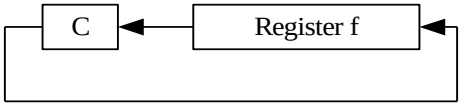
| NOP | No Operation | |
|---|---|---|
| Syntax | NOP | |
| Operands | - | |
| Operation | No Operation | |
| Status Affected | - | |
| OP-Code | 00 0000 0000 0000 | |
| Description | No Operation | |
| Cycle | 1 | |
| Example | NOP | - |

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

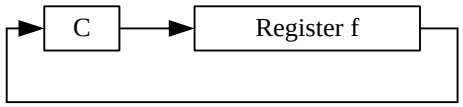
| RETI | Return from Interrupt | |
|---|---|---|
| Syntax | RETI | |
| Operands | - | |
| Operation | PC ← TOS, GIE ← 1 | |
| Status Affected | - | |
| OP-Code | 00 0000 0110 0000 | |
| Description | Return from Interrupt. Stack is POPed and Top-of-Stack (TOS) is loaded in to the PC. Interrupts are enabled.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

| RETLW | Return with Literal in W | |
|---|---|---|
| Syntax | RETLW k | |
| Operands | k : 00h ~ FFh | |
| Operation | PC ← TOS, (W) ← k | |
| Status Affected | - | |
| OP-Code | 01 1000 kkkk kkkk | |
| Description | The W register is loaded with the eight-bit literal 'k'. The program counter is loaded from the top of the stack (the return address). This is a two-cycle instruction. | |
| Cycle | 2 | |
| Example | CALL TABLE | 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<br>A : REG1 = 1110 0110<br>W = 1100 1100, C = 1 |

## RRF — Rotate Right "f" through Carry

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

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

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

| SUBWF | Subtract W from "f" | |
|---|---|---|
| Syntax | SUBWF f [,d] | |
| Operands | f : 00h ~ 7Fh, d : 0, 1 | |
| Operation | (destination) ← (f) – (W) | |
| Status Affected | C, DC, Z | |
| OP-Code | 00 0010 dfff ffff | |
| Description | Subtract (2's complement method) W register from register 'f'. If 'd' is 0, the result is stored in the W register. If 'd' is 1, the result is stored back in register 'f'. | |
| Cycle | 1 | |
| Example | SUBWF REG1, 1 | B : REG1 = 0x03, W = 0x02, C = ?, Z = ?<br>A : REG1 = 0x01, W = 0x02, C = 1, Z = 0 |
| | SUBWF REG1, 1 | B : REG1 = 0x02, W = 0x02, C = ?, Z = ?<br>A : REG1 = 0x00, W = 0x02, C = 1, Z = 1 |
| | SUBWF REG1, 1 | B : REG1 = 0x01, W = 0x02, C = ?, Z = ?<br>A : REG1 = 0xFF, W = 0x02, C = 0, Z = 0 |

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

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

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

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

# ELECTRICAL CHARACTERISTICS

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

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

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

| Parameter | Sym | Conditions | | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|---|
| Input High Voltage | $V_{IH}$ | All Input, except PA7 | $V_{DD}$ = 5V | 0.7 $V_{DD}$ | – | – | V |
| | | | $V_{DD}$ = 3V | 0.7 $V_{DD}$ | – | – | V |
| | | PA7 | $V_{DD}$ = 5V | 0.8 $V_{DD}$ | – | – | V |
| | | | $V_{DD}$ = 3V | 0.8 $V_{DD}$ | – | – | V |
| Input Low Voltage | $V_{IL}$ | All Input, except PA7 | $V_{DD}$ = 5V | – | – | 0.2 $V_{DD}$ | V |
| | | | $V_{DD}$ = 3V | – | – | 0.2 $V_{DD}$ | V |
| | | PA7 | $V_{DD}$ = 5V | – | – | 0.2 $V_{DD}$ | V |
| | | | $V_{DD}$ = 3V | – | – | 0.2 $V_{DD}$ | V |
| I/O Port Source Current | $I_{OH}$ | All Output | $V_{DD}$ = 5V, $V_{OH}$ = 0.9 $V_{DD}$ | 4 | 8 | – | mA |
| | | | $V_{DD}$ = 3V, $V_{OH}$ = 0.9 $V_{DD}$ | 2 | 4 | – | |
| I/O Port Sink Current | $I_{OL}$ | All Output | $V_{DD}$ = 5V, $V_{OL}$ = 0.1 $V_{DD}$ | 10 | 20 | – | mA |
| | | | $V_{DD}$ = 3V, $V_{OL}$ = 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 | $V_{IN}$ = 0V | – | – | –1 | µA |
| Power Supply Current | $I_{DD}$ | Run 8 MHz, No Load | $V_{DD}$ = 5V | – | 1.8 | – | mA |
| | | Run 4 MHz, No Load | $V_{DD}$ = 3V | – | 0.6 | – | |
| | | Stop mode, LVR disable | $V_{DD}$ = 5V | – | 0.1 | – | µA |
| | | | $V_{DD}$ = 3V | | 0.1 | | |
| | | Stop mode, LVR enable | $V_{DD}$ = 5V (TM57PE11B) | – | 4.4 | – | |
| | | | $V_{DD}$ = 5V (TM57PE11C) | | 3.0 | | |
| | | | $V_{DD}$ = 3V (TM57PE11B) | – | 1.2 | – | |
| | | | $V_{DD}$ = 3V (TM57PE11C) | | 0.8 | | |
| System Clock Frequency | Fsys | $V_{DD}$ > $LVR_{th}$ | $V_{DD}$ = 3.1V (TM57PE11B) $V_{DD}$ = 2.9V (TM57PE11C) | – | – | 8 | MHz |
| | | | $V_{DD}$ = 2.2V (TM57PE11B) $V_{DD}$ = 2.0V (TM57PE11C) | – | – | 4 | |
| | | | $V_{DD}$ = 3V | | 135 | | |

| Parameter | Sym | Conditions | | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|---|
| LVR Reference Voltage | $V_{LVR}$ | $T_A$ = 25℃ (TM57PE11B) | | – | 2.2 | – | V |
| | | | | – | 3.1 | – | V |
| | | $T_A$ = 25℃ (TM57PE11C) | | – | 2.0 | – | V |
| | | | | – | 2.9 | – | V |
| LVR Hysteresis Voltage | $V_{HYST}$ | $T_A$ = 25℃ | | – | ±0.1 | – | V |
| LVD Reference Voltage | $V_{LVD}$ | $T_A$ = 25℃ (TM57PE11B) | | – | 2.3 | – | V |
| | | | | – | 3.2 | – | V |
| | | $T_A$ = 25℃ (TM57PE11C) | | – | 2.1 | – | V |
| | | | | – | 3.0 | – | V |
| Low Voltage Detection time | $t_{LVR}$ | $T_A$ = 25℃ | | 100 | – | – | μs |
| Pull-Up Resistor | $R_P$ | $V_{IN}$ = 0 V Port A | $V_{DD}$ = 5V | – | 65 | – | KΩ |
| | | | $V_{DD}$ = 3V | | 120 | | |
| | | $V_{IN}$ = 0 V PA7 | $V_{DD}$ = 5V | – | 60 | – | KΩ |
| | | | $V_{DD}$ = 3V | | 135 | | |

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

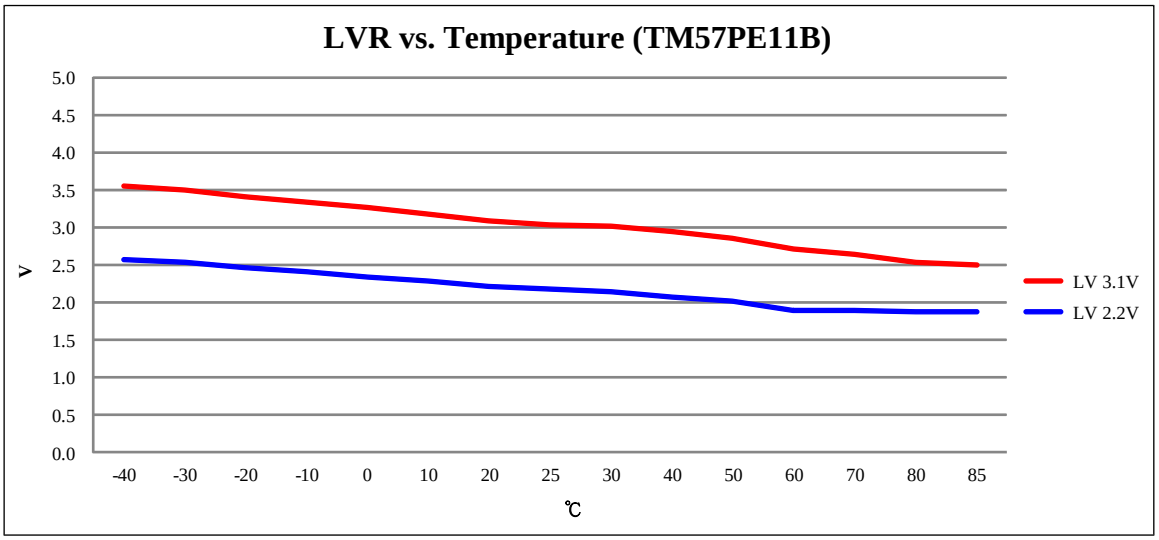
| Parameter | Condition | | | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|---|
| External RC Frequency | $V_{DD}$ = 3V | R = 4.7K | C = 20 pF | – | 3.0 | – | MHz |
| | | R = 10K | C = 100 pF | – | 0.8 | – | |
| | $V_{DD}$ = 5V | R = 4.7K | C = 20 pF | – | 3.8 | – | |
| | | R = 10K | C = 100 pF | – | 0.7 | – | |
| Internal RC Frequency | 25°C, $V_{DD}$ = 3.0 ~ 5.5V | | | 3.88 | 4 | 4.12 | |
| | 25°C, $V_{DD}$ = 2.6 ~ 3.0V | | | 3.8 | 4 | 4.2 | |
| | -40°C ~ 85°C, $V_{DD}$ = 2.6 ~ 5.5V | | | 3.72 | 4 | 4.28 | |

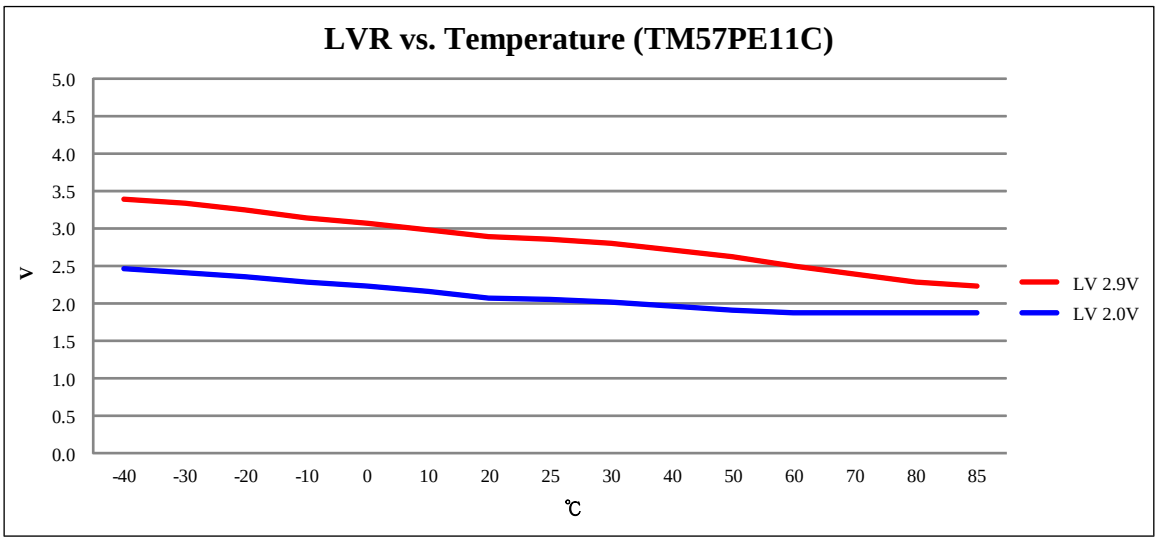
Note: The IRC frequency is trimmed in wafer type. After packaging or COB, the frequency will deviation range is about 10~20%.

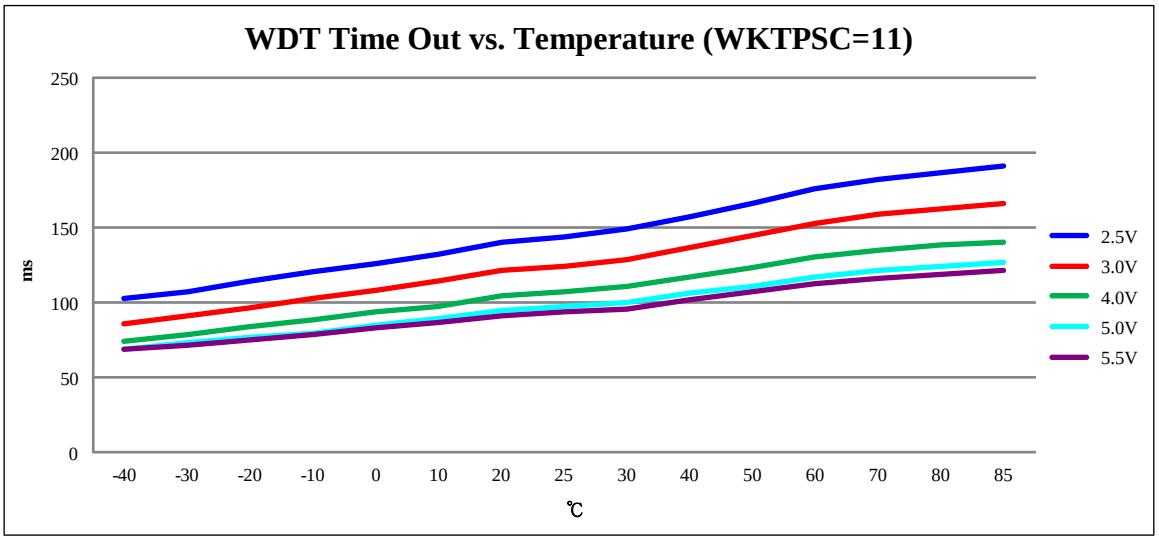
## 4. Reset Timing Characteristics ($T_A$ = -40℃ to +85℃)

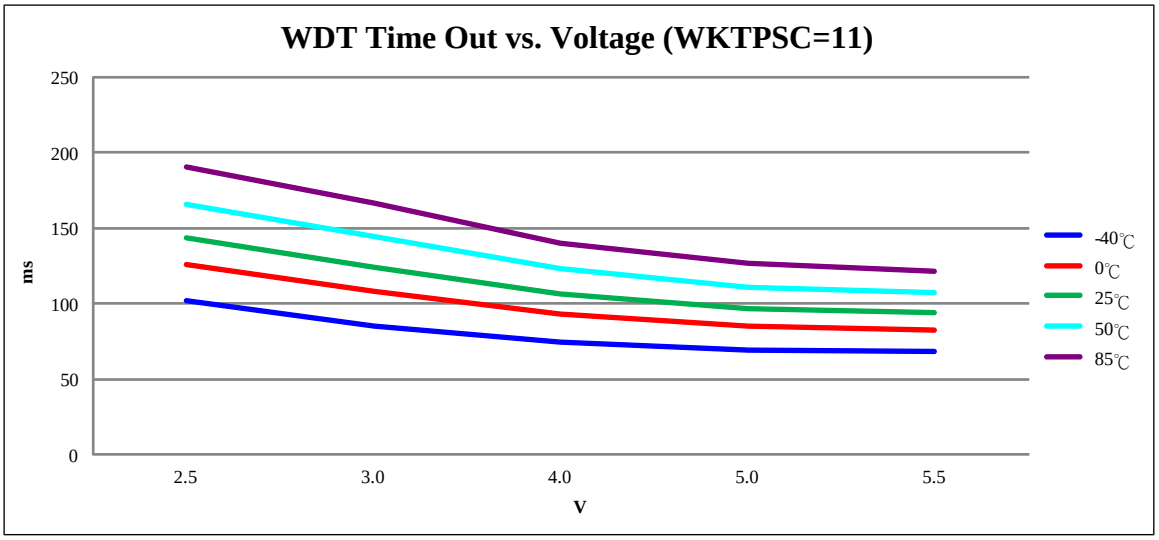
| Parameter | Conditions | Min | Typ | Max | Unit |
|---|---|---|---|---|---|
| RESET Input Low width | Input $V_{DD}$ = 5V ± 10 % | 3 | – | – | μs |
| WDT wakeup time | $V_{DD}$ = 5V, WKTPSC = 11 | – | 96 | – | ms |
| | $V_{DD}$ = 3V, WKTPSC = 11 | – | 128 | – | |
| CPU start up time | $V_{DD}$ = 5V | – | 12 | – | ms |
| | $V_{DD}$ = 3V | – | 16 | – | |

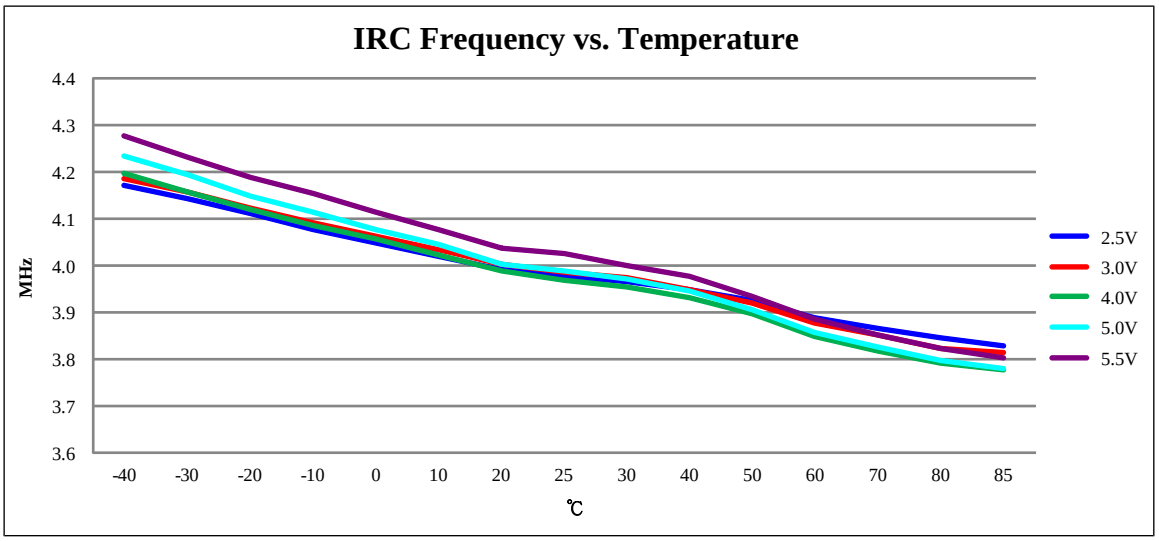
## 5. Characteristic Graphs

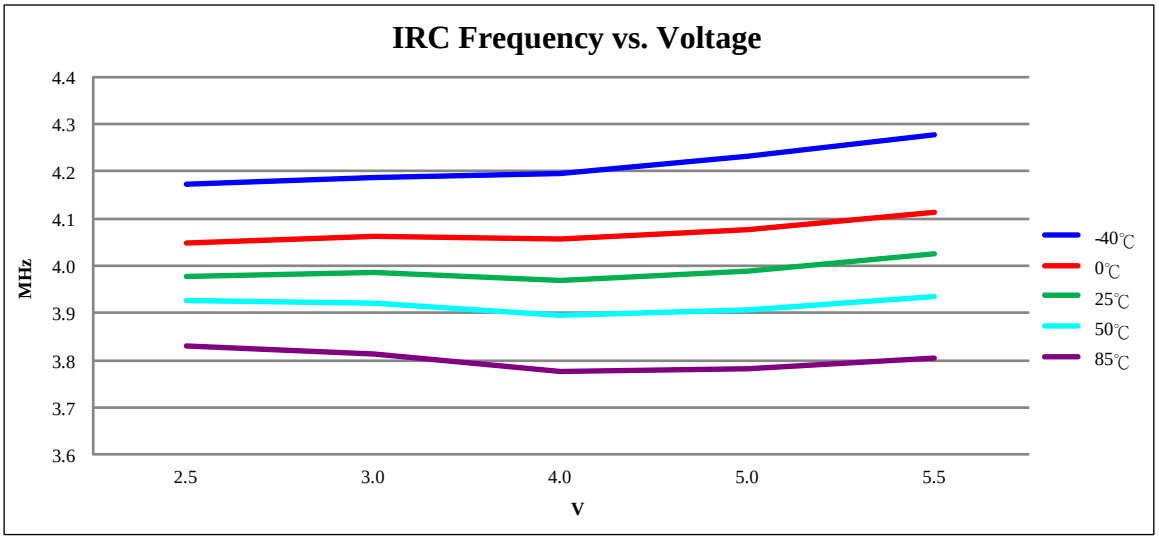



WDT Time Out vs. Voltage (WKTPSC=11)
250
200
150
100
50
0
ms
2.5
3.0
4.0
5.0
5.5
V
-40℃
0℃
25℃
50℃
85℃


IRC Frequency vs. Temperature
4.4
4.3
4.2
4.1
4.0
3.9
3.8
3.7
3.6
MHz
-40
-30
-20
-10
0
10
20
25
30
40
50
60
70
80
85
℃
2.5V
3.0V
4.0V
5.0V
5.5V


IRC Frequency vs. Voltage
4.4
4.3
4.2
4.1
4.0
3.9
3.8
3.7
3.6
MHz
2.5
3.0
4.0
5.0
5.5
V
-40℃
0℃
25℃
50℃
85℃

# ORDERING INFORMATION

The ordering information:

| Ordering number | Package |
|---|---|
| TM57PE11B-OTP | Wafer / Dice blank chip |
| TM57PE11B-COD | Wafer / Dice with code |
| TM57PE11C-OTP | Wafer / Dice blank chip |
| TM57PE11C-COD | Wafer / Dice with code |