



十速

**TM57MA17/18**

***DATA SHEET***

***Rev 0.90***

**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

<b>Version</b>	<b>Date</b>	<b>Description</b>
V0.90	Jan, 2017	New release.

# CONTENTS

<b>AMENDMENT HISTORY .....</b>	<b>2</b>
<b>CONTENTS.....</b>	<b>3</b>
<b>FEATURES .....</b>	<b>5</b>
<b>BLOCK DIAGRAM .....</b>	<b>8</b>
<b>PIN ASSIGNMENT .....</b>	<b>9</b>
<b>PIN DESCRIPTIONS .....</b>	<b>10</b>
<b>PIN SUMMARY.....</b>	<b>11</b>
<b>FUNCTIONAL DESCRIPTION .....</b>	<b>12</b>
<b>1. CPU Core .....</b>	<b>12</b>
1.1 Clock Scheme and Instruction Cycle .....	12
1.2 Program ROM (PROM).....	13
1.3 Programming Counter (PC) and Stack.....	14
1.4 ALU and Working (W) Register.....	15
1.5 RAM Addressing Mode .....	16
1.6 STATUS Register (F-Plane 03H) .....	18
1.7 Interrupt.....	19
<b>2. Chip Operation Mode .....</b>	<b>22</b>
2.1 Reset (000H) .....	22
2.2 System Configuration Register (SYSCFG) .....	23
2.3 Power-Down Mode .....	23
2.4 Dual System Clock.....	24
2.5 Dual System Clock Modes Transition .....	26
2.6 Power.....	30
<b>3. Peripheral Functional Block .....</b>	<b>32</b>
3.1 Watchdog (WDT) /Wakeup (WKT) Timer.....	32
3.2 Timer0 .....	35
3.3 Timer1 .....	41
3.4 T2:15-bit Timer.....	44
3.5 PWM0: (8+2) bits PWM.....	47
3.6 PWM1 / PWM2 / PWM3 .....	53
3.7 Analog-to-Digital Converter .....	57
3.7 OPA: Operational Amplifier .....	60
3.8 DPDMV .....	62
3.9 System Clock Oscillator.....	64
<b>4 I/O Port.....</b>	<b>65</b>
4.1 PA0-6, PB0-4 .....	65
4.2 PA7.....	70

<b>MEMORY MAP</b> .....	<b>71</b>
<b>F-Plane</b> .....	<b>71</b>
<b>R-Plane</b> .....	<b>77</b>
<b>INSTRUCTION SET</b> .....	<b>80</b>
<b>ELECTRICAL CHARACTERISTICS</b> .....	<b>93</b>
<b>1. Absolute Maximum Ratings</b> .....	<b>93</b>
<b>2. DC Characteristics</b> .....	<b>93</b>
<b>3. Clock Timing</b> .....	<b>94</b>
<b>4. Reset Timing Characteristics</b> .....	<b>95</b>
<b>5. LVR Circuit Characteristics</b> .....	<b>95</b>
<b>6. ADC Electrical Characteristics</b> .....	<b>95</b>
<b>7. LDO Characteristics (LDO25SEL = 0)</b> .....	<b>95</b>
<b>8. OPA Circuit Characteristics</b> .....	<b>96</b>
<b>9. DPDMV Circuit Characteristics</b> .....	<b>96</b>
<b>10. Characteristic Graphs</b> .....	<b>97</b>
<b>PACKAGING INFORMATION</b> .....	<b>102</b>
<b>16-DIP Package Dimension</b> .....	<b>103</b>
<b>16-SOP Package Dimension</b> .....	<b>104</b>

## FEATURES

1. **ROM: 1K x 14 bits MTP (Multi Time Programmable ROM)**
2. **RAM: 96 x 8 bits**
3. **STACK: 6 Levels**
4. **System Oscillation Sources (Fsys)**
  - Fast-clock
    - FIRC (Fast Internal RC): 8MHz (can be trimmed)
  - Slow-clock
    - SIRC (Slow Internal RC): 128 KHz @VCC=3V
5. **System Clock Prescaler**
  - System Oscillation Sources can be divided by 16/4/2/1 as System Clock (Fsys)
6. **Dual System Clock**
  - FIRC+SIRC
7. **Power Saving Operation Mode**
  - FAST Mode: Slow-clock can be disabled or enabled, Fast-clock keeps CPU running
  - SLOW Mode: Fast-clock can be disabled or enabled, Slow-clock keeps CPU running
  - IDLE Mode: Fast-clock and CPU stop. Slow-clock, T2, or Wake-up Timer keep running
  - STOP Mode: All Clocks stop, T2 and Wake-up Timer stop
8. **3 Independent Timers**
  - Timer0
    - 8-bit timer with divided by 1~256 pre-scale option/auto-reload/counter/interrupt/stop function
  - Timer1
    - 8-bit timer with divided by 1~256 pre-scale option/auto-reload/interrupt/stop function
  - T2
    - 15-bit timer with 4 interrupt interval time options
    - IDLE mode wake-up timer or used as one simple 15-bit time base
    - Clock source: Slow-clock (SIRC), Fsys/128
9. **Interrupt**
  - Three External Interrupt pins
    - 1 pin is falling edge wake-up triggered & interrupts
    - 2 pins are rising or falling edge wake-up triggered & interrupt
  - Timer0/Timer1/T2/WKT (wake-up) Interrupts

**10. Wake-up Timer (WKT)**

- Clocked by built-in RC oscillator with 4 adjustable interrupt times  
16 ms/32 ms/64 ms/128 ms @VCC=3V, 15 ms/30 ms/60 ms/120 ms @VCC=5V

**11. Watchdog Timer (WDT)**

- Clocked by built-in RC oscillator with 4 adjustable reset times  
130ms/260ms/1040ms/2080ms @VCC=3V, 120ms/240ms/960ms/1920ms @VCC=5V

**12. 4 Independent PWMs**

- PWM0:
  - 8+2 bits, duty-adjustable, period-adjustable controlled PWM
  - PWM0 clock source: System Clock, FIRC/3 or FIRC\*2, with 1/2/4/64 pre-scale option
  - With differential output pair
  - Non-overlap durations adjustable
  - PWM0P and PWM0N are high drive/sink pins
- PWM1:
  - 8-bit PWM1 with pre-scale/period-adjustment/clear and hold function
- PWM2:
  - 8-bit PWM2 with pre-scale/period-adjustment/clear and hold function
- PWM3:
  - 8-bit PWM3 with pre-scale/period-adjustment/clear and hold function

**13. 12-bit ADC Converter with 8 input channels and 1 internal reference voltage**

- ADC reference voltage=Internal reference voltage LDO  $\pm 2\%$  @25°C, VCC=3V~5V
- 2 levels LDO 2.5V or 1.25V can be selected

**14. Operational Amplifier (OPA)**

- offset voltage  $\leq 2\text{mV}$  @Vo=1.5V, TA=25°C, VCC=5V, VSS=0V
- with offset calibration

**15. DP1/DM1, DP2/DM2 for USB charging control application**

- 5 levels voltage output: 3.3V/2.7V/2.0V/1.2V/0.6V @VCC=5V
- DP/DM with short option
- DM with pull-low resistor option

**16. Reset Sources**

- Power On Reset
- Watchdog Rese
- Low Voltage Reset
- External Pin Reset

**17. 3-Level Low Voltage Reset: 2.9V/2.3V/2.0V (can be disable in IDLE/STOP mode)**

**18. Operating Voltage: Low Voltage Reset Level to 5.5V**

**19. Operating Temperature Range: -40°C to +85°C**

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

**21. Instruction set: 39 Instructions**

**22. Instruction Execution Time**

- 2 oscillation clocks per instruction except branch

**23. I/O ports: Maximum 13 programmable I/O pins**

- Open-Drain Output
- CMOS Push-Pull Output
- Schmitt Trigger Input with pull-up resistor option

**24. High-Sink and High-Drive I/O: PA0, PB3**

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

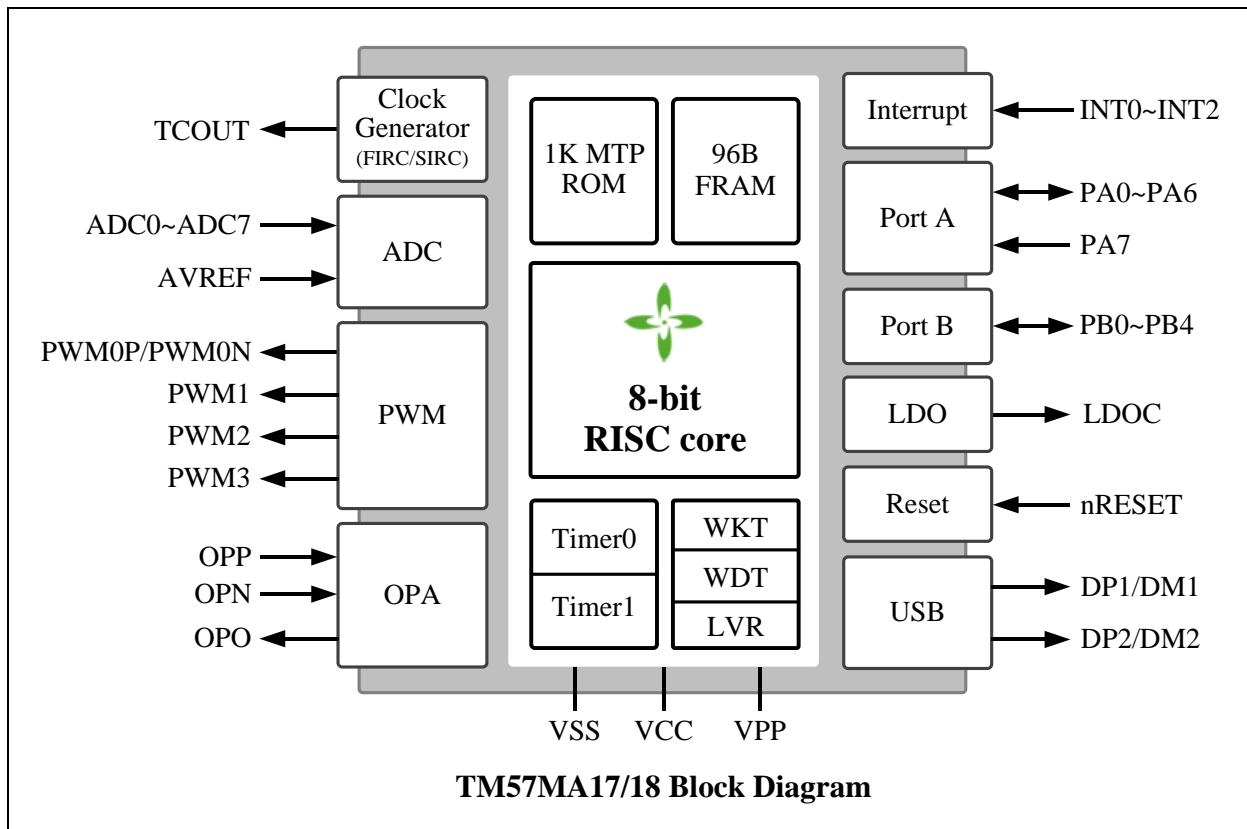
**26. Package Types:**

- SOP-16/DIP-16

**27. Supported EV board on ICE**

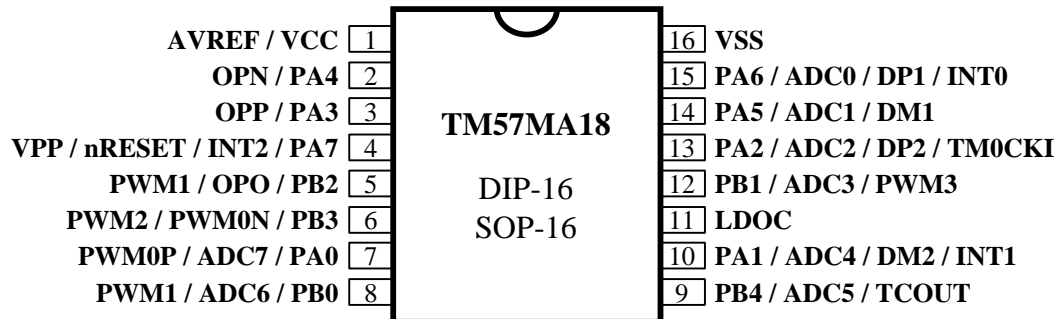
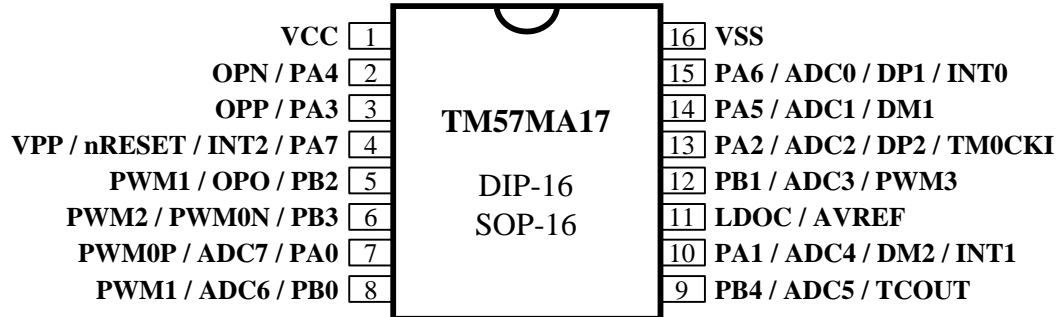
EV board: EV8215

### BLOCK DIAGRAM





## PIN ASSIGNMENT



## PIN DESCRIPTIONS

Name	In/Out	Pin Description
PA0-PA6	I/O	Bit-programmable I/O port for Schmitt-trigger input, CMOS push-pull output or open-drain output. Pull-up resistors are assignable by software.
PA7	I	Bit-programmable I/O port for Schmitt-trigger input or open-drain output. Pull-up resistor is always assignable.
PB0-PB4	I/O	Bit-programmable I/O port for Schmitt-trigger input, CMOS push-pull output or open-drain output. Pull-up resistors are assignable by software.
nRESET	I	External active low reset, internal pull-high
VCC, VSS	P	Power Voltage input pin and ground
VPP	I	PROM programming high voltage input
INT0-INT2	I	External interrupt input
TMOCKI	I	Timer0's input in counter mode
PWM0P	O	(8+2) bit PWM0 positive output
PWM0N	O	(8+2) bit PWM0 negative output
PWM1	O	PWM1 output
PWM2	O	PWM2 output
PWM3	O	PWM3 output
ADC7-ADC0	I	ADC channels input
AVREF	I	ADC reference voltage
OPP, OPN	I	OPA positive/negative input
OPO	O	OPA output
DP1, DM1	O	USB positive/negative data-channel 1 to external USB device.
DP2, DM2	O	USB positive/negative data-channel 2 to external USB device.
LDOC	O	Internal reference voltage 2.5V or 1.25V output (at least 1uF to ground)
TCOUT	O	Instruction Cycle ( $F_{sys}/2$ ) output

Programming pins:

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

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

**PIN SUMMARY**

MA17/18  16-SOP/DIP	Pin Name	Type	GPIO				Function After Reset	Alternate Function					
			Input		Output			PWM	OPA	ADC	USB	MISC	
			Weak Pull-up	Ext. Interrupt	O.D	P.P							
1	VCC (TM57MA17) VCC/AVREF (TM57MA18)	P											
2	OPN/PA4	I/O	○		○	○	PA4		○				
3	OPP/PA3	I/O	○		○	○	PA3		○				
4	VPP/nRESET/INT2/PA7	I/O	○	○	○		SYS						nRESET
5	PWM1/OPO/PB2	I/O	○		○	○	PB2	○	○				
6	PWM2/PWM0N/PB3	I/O	○		○	○	PB3	○					
7	PWM0P/ADC7/PA0	I/O	○		○	○	PA0	○		○			
8	ADC6/PB0	I/O	○		○	○	PB0			○			
9	PB4/ADC5/TCOUT	I/O	○		○	○	PB4			○			TCOUT
10	PA1/ADC4/DM2/INT1	I/O	○	○	○	○	PA1			○	○		
11	LDOC/AVREF (TM57MA17) LDOC (TM57MA18)	○											
12	PB1/ADC3/PWM3	I/O	○		○	○	PB1	○		○			
13	PA2/ADC2/DP2/TM0CKI	I/O	○		○	○	PA2			○	○		TM0CKI
14	PA5/ADC1/DM1	I/O	○		○	○	PA5			○	○		
15	PA6/ADC0/DP1/INT0	I/O	○	○	○	○	PA6			○	○		
16	VSS	P											

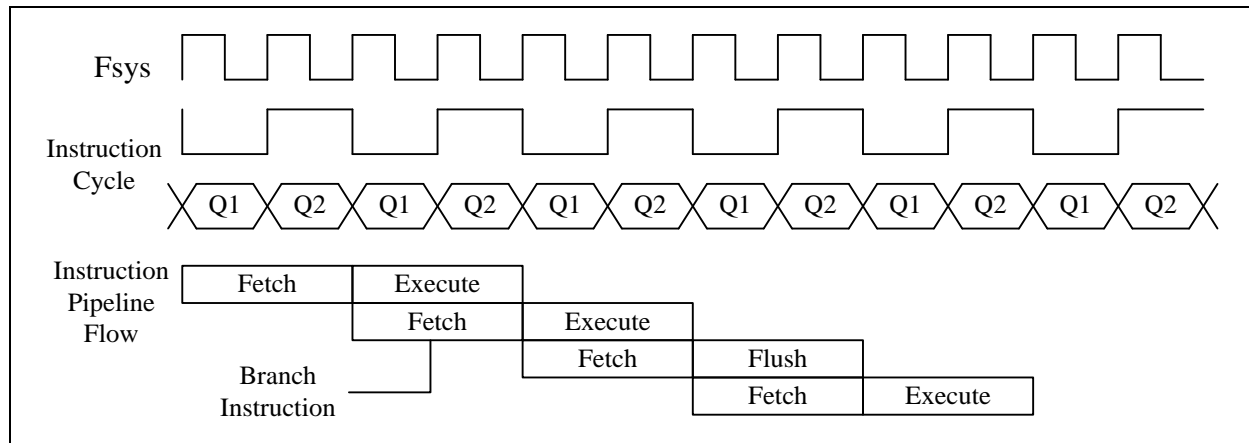
Symbol : P.P. = CMOS Push-Pull Output  
 O.D. = Open Drain Output  
 SYS = by SYSCFG bit

## FUNCTIONAL DESCRIPTION

### 1. CPU Core

#### 1.1 Clock Scheme and Instruction Cycle

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



Terminology definitions:

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

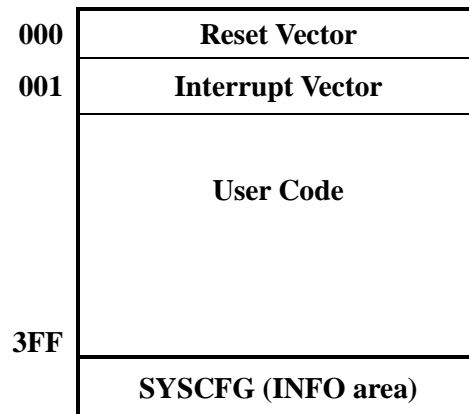
(2) **Instruction Cycle**=Fsys/2

FIRC: Fast Internal RC oscillator

SIRC: Slow Internal RC oscillator

## 1.2 Program ROM (PROM)

The MTP Program ROM of this device is 1K words, with an extra INFO area to store the SYSCFG. The ROM can be written multi-times and can be read as long as the PROTECT bit of SYSCFG is not set. The SYSCFG can be read no matter PROTECT is set or cleared, but can be written only when PROTECT is not set or ROM is erased. That is, unprotect the PROTECT bit needs the erased ROM.



### 1.3 Programming Counter (PC) and Stack

The Programming Counter is 10-bit wide capable of addressing a 1Kx14 MTP ROM. As a program instruction is executed, the PC will contain the address of the next program instruction to be executed. The PC value is normally increased by one except the followings. The Reset Vector (000h) and the Interrupt Vector (001h) are provided for PC initialization and Interrupt. For CALL/GOTO instructions, PC loads 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 6-level in depth. The CALL instruction and hardware interrupt will push STACK level in order. While the RET/RETI/RETLW instruction pops the STACK level in order.

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

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

```

ORG      000H                ; Reset Vector
GOTO     START

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

LOOP:
MOVFW   INDEX                ; Move index value to W register.
CALL    TABLE                ; To lookup data, W=55H.
.....
INCF    INDEX, 1              ; Increment the index address for next address
.....
GOTO    LOOP                  ; Go to LOOP label.
.....
MOVLW   (TABLE2>>8)&0xff
MOVWF   DPH                   ; DPH register (F1E.1~0)
MOVLW   (TABLE2)&0xff
MOVWF   DPL                   ; DPL register (F1D.7~0)
TABRL
TABRH
.....

TABLE:
ADDWF   PCL, 1                ; Add the W with PCL, the result back in PCL.
RETLW   55H                   ; W=55h when return
RETLW   56H                   ; W=56H when return
RETLW   58H                   ; W=58H when return
.....
ORG      368H

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

```

#### 1.4 ALU and Working (W) Register

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

Note: /Borrow represents inverted of Borrow register.

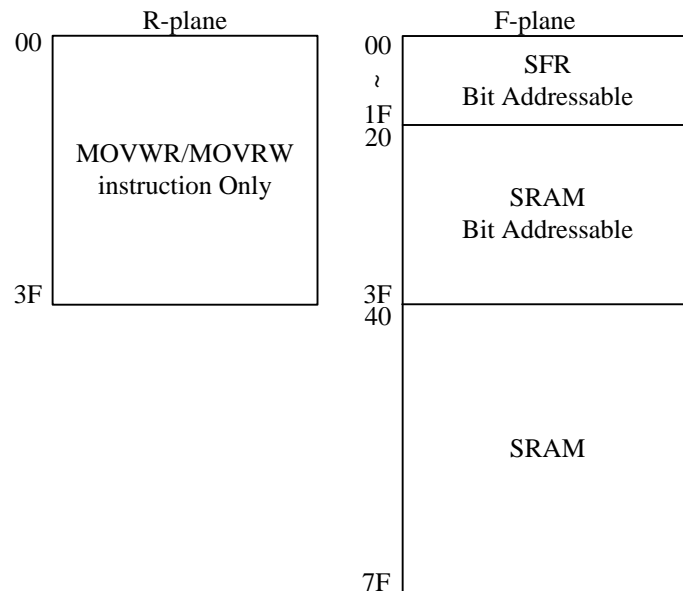
/Digit Borrow represents inverted of Digit Borrow register.

### 1.5 RAM Addressing Mode

There are two Data Memory Planes in CPU, R-Plane and F-Plane. The F-Plane supports rich instructions operation, such as ADDWF, INCF, MOVWF,..., while the R-Plane only supports MOVWR and MOVWRW instructions to exchange data between R-Plane and W-Register.

The R-Plane can be indirect accessed via RSR register (F1C.7~0) and INDR (R00). The INDR register is not a physical register. Addressing INDR actually addresses the register whose address is contained in the RSR register (RSR is a pointer).

The lower locations of F-Plane are reserved for the SFR. Above the SFR is General Purpose Data Memory, implemented as static RAM. F-Plane can be addressed directly or indirectly. Indirect Addressing is made by INDF register. The INDF register is not a physical register. Addressing INDF actually addresses the register whose address is contained in the FSR register (FSR is a pointer). The first half of F-Plane is bit-addressable, while the second half of F-Plane is not bit-addressable.



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

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

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

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

◇Example: Move R-Plane location 0BH SFR data into W register.

```
MOVRW    0BH           ; To get a content of R-Plane location 0BH and save in W.
```

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

```
MOVFW    20H           ; To get a content of F-Plane location 20H and save in W.
```





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

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

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

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

STATUS	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
<b>Reset Value</b>	0	0	–	0	0	0	0	0
<b>R/W</b>	R/W	R/W	–	R	R	R/W	R/W	R/W
Bit	Description							
7	<b>GB1:</b> General Purpose Bit 1							
6	<b>GB0:</b> General Purpose Bit 0							
5	<b>Reserved</b>							
4	<b>TO:</b> Time Out Flag 0: after Power On Reset, LVR Reset, or CLRWDT/SLEEP instruction 1: WDT time out occurs							
3	<b>PD:</b> Power Down Flag 0: after Power On Reset, LVR Reset, or CLRWDT instruction 1: after SLEEP instruction							
2	<b>Z:</b> Zero Flag 0: the result of a logic operation is not zero 1: the result of a logic operation is zero							
1	<b>DC:</b> 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	<b>C:</b> 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.

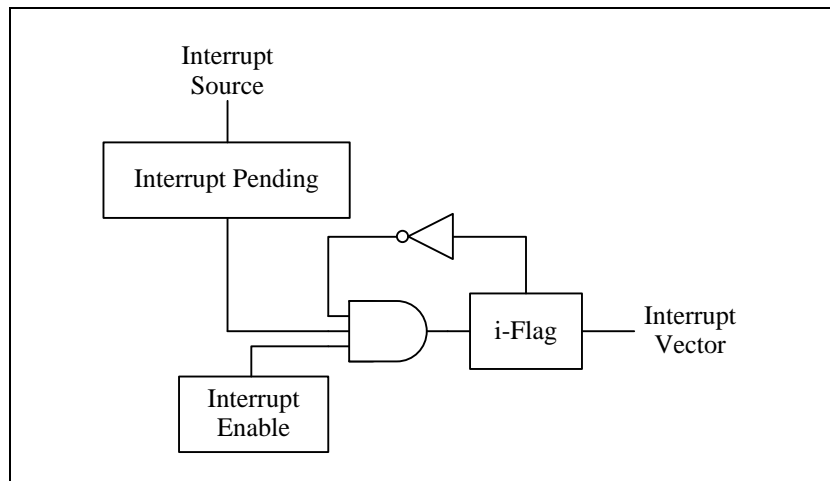
```
BTFSS    STATUS, 0    ; Check the carry flag
GOTO     LABEL_1     ; If C=0, goto label_1
GOTO     LABEL_2     ; If C=1, goto label_2
```

### 1.7 Interrupt

This device has 1 level, 1 vector and 7 interrupt sources. Each interrupt source has its own enable control bit. An interrupt event will set its individual pending flag; no matter its interrupt enable control bit is 0 or 1. Because device has only 1 vector, there is not an interrupt priority register. The interrupt priority is determined by F/W.

If the corresponding interrupt enable bit has been set (INTIE), it would trigger CPU to service the interrupt. CPU accepts interrupt in the end of current executed instruction cycle. In the mean while, a “CALL 001” instruction is inserted to CPU, and i-flag is set to prevent recursive interrupt nesting.

The i-flag is cleared in the instruction after the “RETI” instruction. That is, at least one instruction in main program is executed before service the pending interrupt. The interrupt event is level triggered. F/W must clear the interrupt event register while serving the interrupt routine.



◇Example: Setup INT1 (PA1) interrupt request and rising edge trigger.

```

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

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

ORG      02H
START:
MOVLW   xxxx00xxB          ; Select INT1 (PA1) pin mode as Mode0
MOVWR   PAMODL             ; Open drain output low or input with Pull-up
BSF     PAD, 1              ; Release INT1, set PA1 as input with Pull-up resistor
MOVLW   10001111B          ; Set INT1 interrupt trigger as rising edge.
MOVWR   R0B
MOVLW   1111101B           ; Clear INT1 interrupt request flag
MOVWF   INTIF
MOVLW   x0000010B          ; Enable INT1 interrupt.
MOVWR   INTIE
  
```

MAIN:

```
...
GOTO    MAIN
```

INT\_SUBROUTINE:

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

BTFSS   INT1IF         ; Check INT1IF bit.
GOTO    EXIT_INT      ; INT1IF=0, exit interrupt vector.
...
MOVWF   INT1IF         ; INT1 interrupt service routine.
MOVLW   1111101B
MOVWF   INTIF          ; Clear INT1 interrupt request flag
GOTO    EXIT_INT
```

EXIT\_INT:

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

F08	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTIE	–	T2IE	TM1IE	TM0IE	WKTIE	INT2IE	INT1IE	INT0IE
R/W	–	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	–	0	0	0	0	0	0	0

- F08.6    **T2IE:** T2 interrupt enable  
0: disable  
1: enable
- F08.5    **TM1IE:** Timer1 interrupt enable  
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
- F08.1    **INT1IE:** INT1 (PA1) interrupt enable  
0: disable  
1: enable
- F08.0    **INT0IE:** INT0 (PA6) interrupt enable  
0: disable  
1: enable

F09	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTIF	–	T2IF	TM1IF	TM0IF	WKTIF	INT2IF	INT1IF	INT0IF
R/W	–	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	–	0	0	0	0	0	0	0

- F09.6 **T2IF:** T2 interrupt event pending flag  
This bit is set by H/W while T2 overflows, write 0 to this bit will clear this flag
- F09.5 **TM1IF:** Timer1 interrupt event pending flag  
This bit is set by H/W while Timer1 overflows, 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
- F09.1 **INT1IF:** INT1 (PA1) pin falling interrupt pending flag  
This bit is set by H/W at INT1 pin's falling edge, write 0 to this bit will clear this flag
- F09.0 **INT0IF:** INT0 (PA6) 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	INT1EDG	INT0EDG	T2PSC		WDTTPSC		WKTTPSC	
R/W	R/W	R/W	R/W		R/W		R/W	
Reset	0	0	0	0	1	1	1	1

- R0B.7 **INT1EDG:** INT1 (PA1) trigger edge select  
0: INT1 (PA1) pin falling edge to trigger interrupt event  
1: INT1 (PA1) pin rising edge to trigger interrupt event
- R0B.6 **INT0EDG:** INT0 (PA6) trigger edge select  
0: INT0 (PA6) pin falling edge to trigger interrupt event  
1: INT0 (PA6) pin rising edge to trigger interrupt event

## 2. Chip Operation Mode

### 2.1 Reset (000H)

This device can be RESET in four ways.

- Power-On-Reset
- Low Voltage Reset (LVR)
- External Pin Reset (PA7)
- Watchdog Reset (WDT)

After Power-On-Reset, all system and peripheral control registers are then set to their default hardware Reset values. The LVR level is selected by the SYSCFG register value. The Low Voltage Reset features static reset when supply voltage is below a threshold level. There are three threshold levels can be selected. The LVR's operation mode is defined by the SYSCFG register. See the following LVR Selection Table; user must also consider the lowest operating voltage of operating frequency.

LVR Selection Table:

LVR Level	Consider the operating voltage to choose LVR
LVR2.9	$5.5V > V_{CC} > 3.6V$
LVR2.3	$5.5V > V_{CC} > 3.0V$
LVR2.0	$V_{CC}$ 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 flags are not affected by these resets.

◇Example: Defining Reset Vector

```

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

ORG      010H

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

## 2.2 System Configuration Register (SYSCFG)

The System Configuration Register (SYSCFG) is located at MTP INFO area. The SYSCFG determines the option for initial condition of MCU. It is written by PROM Writer only. User can select LVR operation mode and chip operation mode by SYSCFG register. The 13th bit of SYSCFG is code protection selection bit. If this bit is 1, the data in PROM will be protected.

Bit	13~0	
Default Value	00_1100_xxxx_xxxx	
Bit	Description	
13	<b>PROTECT</b> : Code protection selection	
	1	Enable
	0	Disable
12	<b>XRSTE</b> : External pin (PA7) reset enable	
	1	Enable
	0	Disable (PA7 as input I/O pin)
11-10	<b>LVR</b> : Low voltage reset mode	
	11	LVR level=2.0V, always enable
	10	LVR level=2.0V, disable in IDLE/STOP mode
	01	LVR level=2.3V
	00	LVR level=2.9V
9-8	<b>WDTE</b> : WDT reset enable	
	11	WDT always enable
	10	WDT enable in FAST/SLOW mode, disable in IDLE/STOP mode
	0x	WDT disable
7-0	Tenx Reserved	

## 2.3 Power-Down Mode

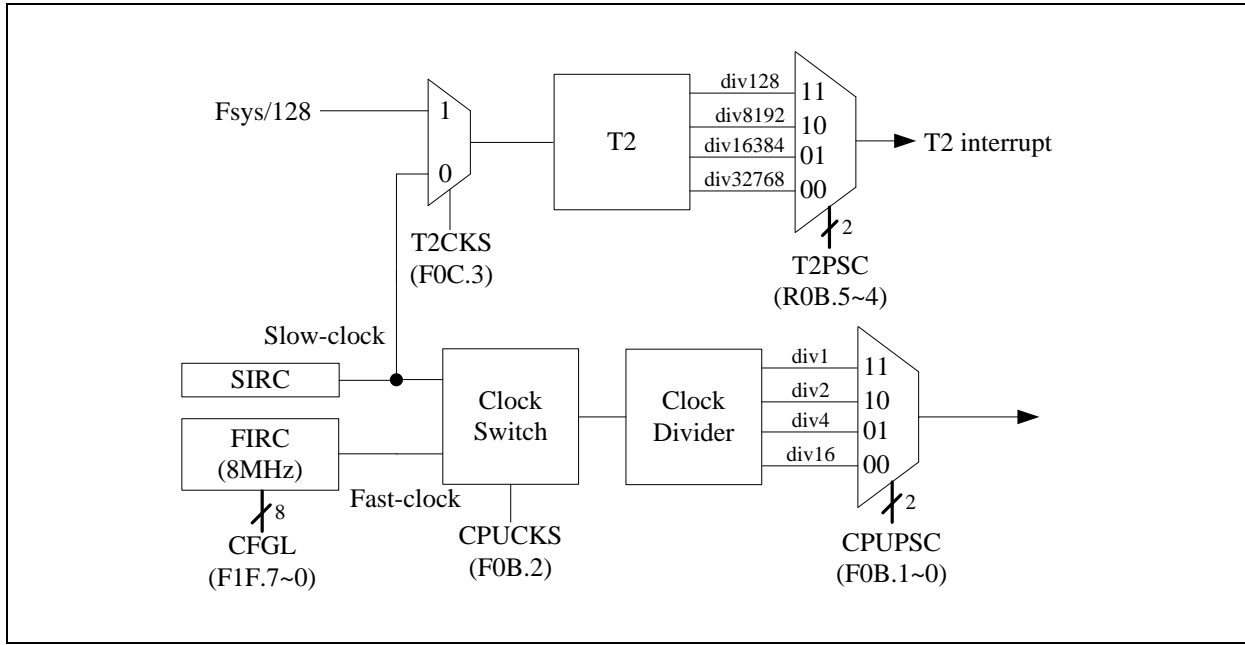
The Power-down mode includes IDLE mode and STOP mode. It is activated by SLEEP instruction. During the Power-down mode, the system clock and peripherals stop to minimize power consumption, whether the WDT/WKT Timer are working or not depend on F/W setting. The Power-down mode can be terminated by Reset, or enabled Interrupts (External pins and WKT interrupts).

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/IDLE) mode

### 2.4 Dual System Clock

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



**Clock Scheme Block Diagram**



**FAST Mode:**

The device enters FAST mode by setting the CPUCKS (F0B.2). In this mode, the system clock source is FIRC. The Timer0, Timer1 and ADC blocks are also driven by Fast-clock; The PWM0 and PWM1~3 blocks can be driven by FIRCx2, FIRC/3 or Fast-clock. T2 can also be driven by Fast-clock by setting T2CKS=1 and CPUCKS=1.

**SLOW Mode:**

After power on or reset, the device enters SLOW mode, the default system clock source is SIRC. In this mode, the Fast-clock can be stopped (by FASTSTP=1, for power saving) or run (by FASTSTP=0), and Slow-clock is enabled. All peripheral blocks (Timer0, Timer1, etc...) clock sources are driven by Slow-clock.

**IDLE Mode:**

When SLOWSTP (F0F.4) is cleared, the device will enter the IDLE mode after executing the SLEEP instruction. In this mode, the Slow-clock will continue running to provide clock to T2 block (T2CKS=0).

Another way to keep clock oscillation in IDLE mode is setting WKTIE=1 (F08.3) to keeping WKT running before executing the SLEEP instruction or WDTIE=11B (SYSCFG.9~8) to keeping WDT running. In such condition, the Slow-clock will also keep running for wakes up CPU periodically no matter SLOWSTP is set or cleared.

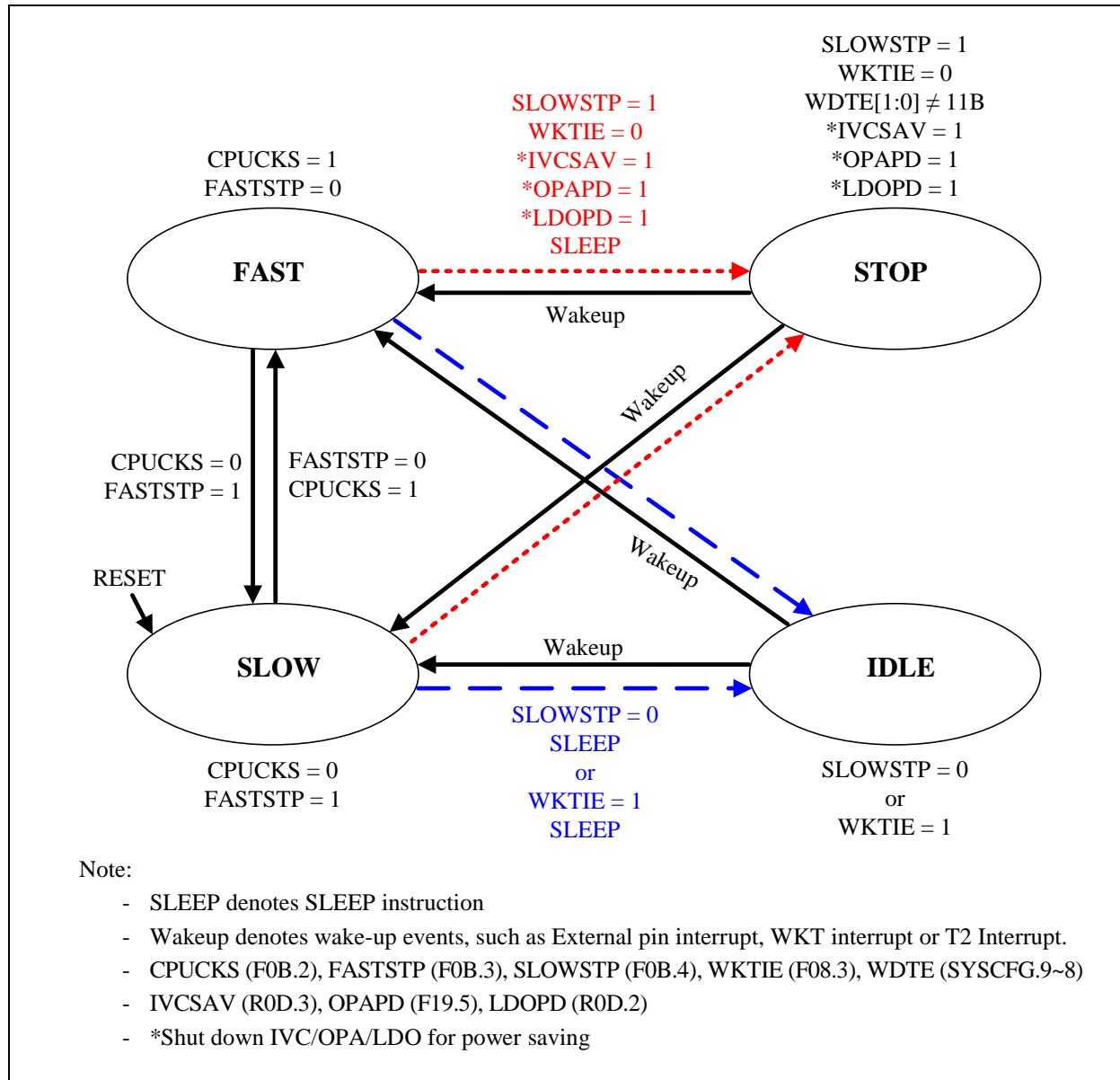
T2 and WKT/WDT are independent and have their own control registers. It is possible to keep T2 or WKT working and wake-up in the IDLE mode, which is useful for low power mode Touch Key detection.

**STOP Mode:**

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

### 2.5 Dual System Clock Modes Transition

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



CPU Operation Block Diagram

### CPU Mode & Clock Functions Table:

Mode	Oscillator	Fsys	Fast-clock	Slow-clock	TM0/TM1 ADC PWM0~3	T2	Wakeup event
<b>FAST</b>	FIRC	Fast-clock	Run	Run	Run	Run	-
<b>SLOW</b>	SIRC	Slow-clock	Set by FASTSTP bit	Run	Run	Run	-
<b>IDLE</b>	SIRC	Stop	Stop	Run	Stop	Run	WKT/IO/T2
<b>STOP</b>	Stop	Stop	Stop	Stop	Stop	Stop	IO

**● FAST mode switches to SLOW mode**

The source clock of Slow-clock is SIRC. The following steps are suggested to be executed by order when FAST mode switches to SLOW mode:

- (1) Switch system clock source to Slow-clock (CPUCKS=0)
- (2) Stop Fast-clock (FASTSTP=1)

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

```
BCF CPUCKS      ; Switch system clock source to Slow-clock
BSF FASTSTP     ; Stop Fast-clock
```

**● SLOW mode switches to FAST mode**

The source clock of Fast-clock is FIRC. The following steps are suggested to be executed by order when SLOW mode transits to FAST mode:

- (1) Enable Fast-clock (FASTSTP=0)
- (2) Switch system clock source to Fast-clock (CPUCKS=1)

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

```
BCF FASTSTP     ; Enable Fast-clock.
BSF CPUCKS      ; Switch system clock source to Fast-clock
```

**● IDLE mode Setting**

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

- (1) Enable Slow-clock (SLOWSTP=0) or WKT (WKTIE=1)
- (2) Execute SLEEP instruction

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

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

```
BCF      SLOWSTP      ; Enable Slow-clock.
SLEEP                                ; Enter IDLE mode.
```

**● STOP Mode Setting**

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

- (1) Stop Slow-clock (SLOWSTP=1)
- (2) Stop WKT/WDT (WKTIE=0)
- (3) Shut down IVC/OPA/LDO for power saving (IVCSAV=1, OPAPD=1, LDOPD=1)
- (4) Execute SLEEP instruction

STOP mode can be woken up only by External pins interrupt.

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

```
BSF      SLOWSTP      ; Disable Slow-clock.
BCF      WKTIE       ; Disable WDT/WKT
BSF      OPAPD       ; Shut down OPA in STOP mode
MOVLW   xxxx11xxB   ; R0D.3=1 (IVCSAV), Shut down IVC in STOP mode
MOVWR   PWRCTL      ; R0D.2=1 (LDOPD), shut down LDO in STOP mode
SLEEP
```

F0B	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
CLKCTL	–	–	–	SLOWSTP	FASTSTP	CPUCKS	CPUPSC	
R/W	–	–	–	R/W	R/W	R/W	R/W	
Reset	–	–	–	0	0	0	1	1

F0B.4 **SLOWSTP**: Slow-clock stop  
 0: Slow-clock is running  
 1: Slow-clock stops running in Power-down mode

F0B.3 **FASTSTP**: Fast-clock stop  
 0: Fast-clock is running  
 1: Fast-clock stops running

F0B.2 **CPUCKS**: System clock source select  
 0: Slow-clock  
 1: Fast-clock

F0B.1~0 **CPUPSC**: System clock source prescaler. System clock source  
 00: divided by 16  
 01: divided by 4  
 10: divided by 2  
 11: divided by 1

F19	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
OPACTL	OPAMOD	OPAOFSEL	OPAPD	OPAOFADJ				
R/W	R/W	R/W	R/W	R/W				
Reset	0	0	1	0	0	0	0	0

F19.5 **OPAPD**: OPA power down  
 0: OPA running  
 1: OPA power down

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

R0D	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWRCTL	CLKFLT	VCCFLT	–	–	IVCSAV	LDOPD	LDO25SEL	MODE3V
R/W	R/W	R/W	–	–	R/W	R/W	R/W	R/W
Reset	0	0	–	–	0	0	1	0

R0D.3 **IVCSAV**: IVC auto power saving in STOP/IDLE mode  
 0: disable IVC save function  
 1: enable IVC save function

R0D.2 **LDOPD**: Internal LDO (2.5V/1.25V) power down  
 0: LDO running  
 1: LDO power down

## 2.6 Power

The TM57MA17/18 has two built-in internal low dropout regulators, IVC and LDO. When MODE3V (R0D.0) =0, the voltage regulator outputs 3.3V power to the internal chip circuit. When MODE3V=1, the IVC is turned off, and the internal circuit receives a power supply directly from the VCC pin. Because the IVC consumes 150 $\mu$ A for operation, turning off IVC by setting MODE3V=1 can reduce the chip current consumption. However, setting MODE3V=1 is only valid for an operating condition of  $V_{CC}<3.6V$ . The IVCSAV (R0D.3) also controls the IVC. When MODE3V=0 and IVCSAV=1, the IVC is turned off in IDLE/STOP mode for saving power consumption. The LDO can output 2.5V or 1.25V power to LDOC pin by selecting LDO25SEL (R0D.1). The LDO also can be turned off by LDOPD (R0D.1) for saving power consumption.

### MODE3V=0

Operation Mode	SFR	CFGW	IVC	LVR	Function
	IVCSAV	LVRE			
FAST SLOW	X	00	ON	ON	LV Reset 2.9V
	X	01	ON	ON	LV Reset 2.3V
	X	10	ON	OFF	LV Reset Disable
	X	11	ON	ON	LV Reset 2.0V
IDLE STOP	0	00	ON	ON	LV Reset 2.9V
	0	01	ON	ON	LV Reset 2.3V
	0	10	ON	OFF	LV Reset Disable
	0	11	ON	ON	LV Reset 2.0V
	1	00	OFF	ON	LV Reset 2.9V
	1	01	OFF	ON	LV Reset 2.3V
	1	10	OFF	OFF	LV Reset Disable
	1	11	OFF	ON	LV Reset 2.0V

### MODE3V=1

Operation Mode	SFR	CFGW	IVC	LVR	Function
	IVCSAV	LVRE			
FAST SLOW	X	00	OFF	ON	LV Reset 2.9V
	X	01	OFF	ON	LV Reset 2.3V
	X	10	OFF	OFF	LV Reset Disable
	X	11	OFF	ON	LV Reset 2.0V
IDLE STOP	X	00	OFF	ON	LV Reset 2.9V
	X	01	OFF	ON	LV Reset 2.3V
	X	10	OFF	OFF	LV Reset Disable
	X	11	OFF	ON	LV Reset 2.0V

R0D	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWRCTL	CLKFLT	VCCFLT	–	–	IVCSAV	LDOPD	LDO25SEL	MODE3V
R/W	R/W	R/W	–	–	R/W	R/W	R/W	R/W
Reset	0	0	–	–	0	0	1	0



- R0D.3 **IVCSAV: IVC auto power saving in STOP/IDLE mode**
  - 0: disable IVC save function
  - 1: enable IVC save function
  
- R0D.2 **LDOPD: Internal LDO (2.5/1.25V) power down**
  - 0: LDO running
  - 1: LDO power down
  
- R0D.1 **LDO25SEL: Internal LDO voltage select**
  - 0: 1.25V
  - 1: 2.5V
  
- R0D.0 **MODE3V: 3V mode selection**
  - 0: system operate in 5V mode ( $V_{cc} > 3.6V$ )
  - 1: system operate in 3V mode ( $V_{cc} < 3.6V$ )

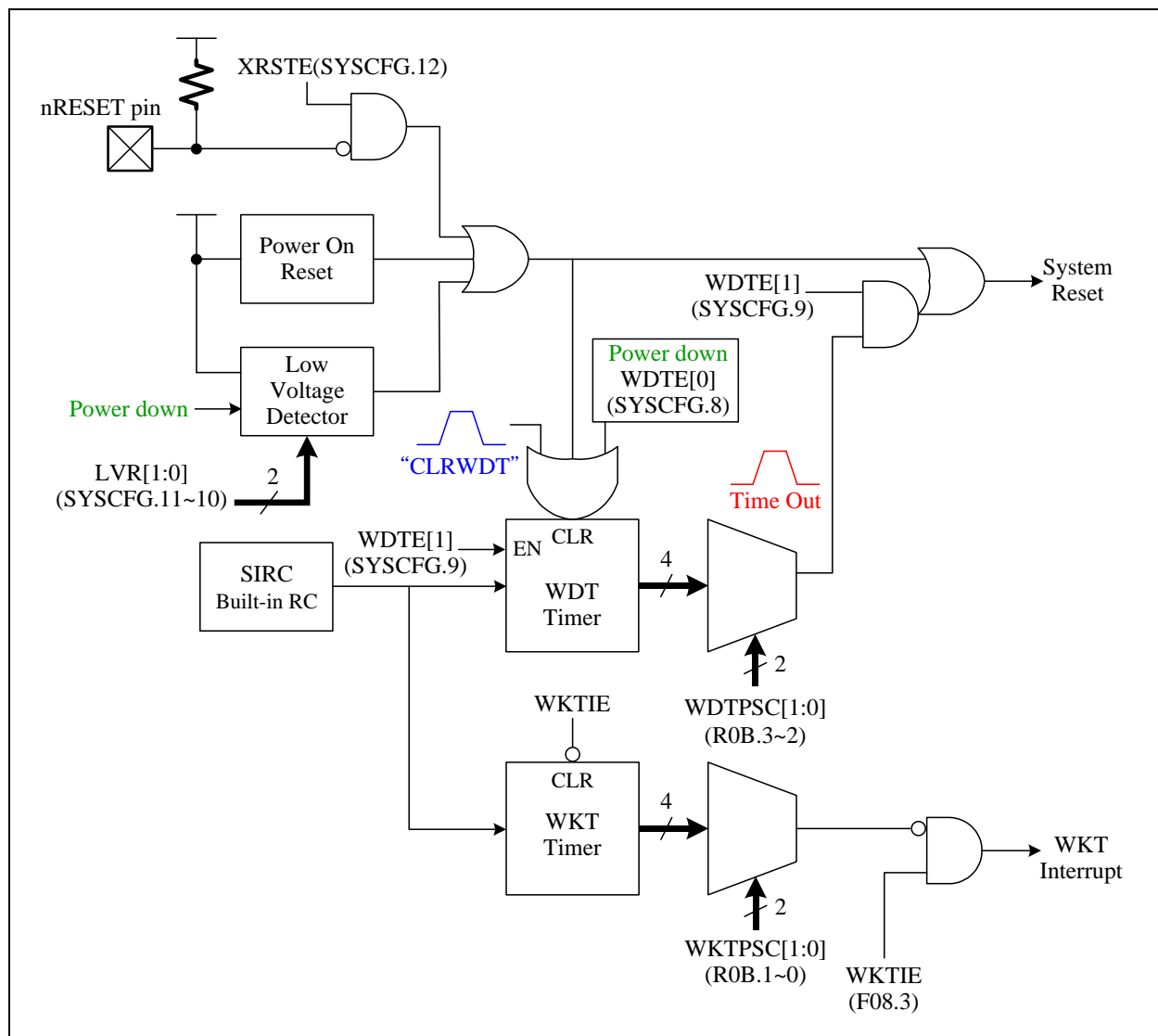
### 3. Peripheral Functional Block

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

The WDT and WKT share the same built-in internal RC Oscillator and have individual own counters. The overflow period of WDT, WKT can be selected by individual prescaler (WDTPSC[1:0], WKTTPSC[1:0]). The WDT timer is cleared by the CLRWDT instruction. If the Watchdog is enabled (SYSCFG[9], WDTE[1]=1), the WDT generates the chip reset signal. In IDLE mode, the WDT is only enabled when WDTE[1:0]=11B. Otherwise it will be disabled and stopped for power saving.

The WKT timer is an interval timer. When WKT overflow time out, it will generate overflow time out flag “WKTIF” (F09.3). The WKT timer is cleared/stopped by WKTIE=0. Set WKTIE=1, the WKT timer will generate WKT overflow time out interrupt and always count regardless at any CPU operating mode.

Watchdog clear is controlled by CLRWDT instruction and moving any value into WDTCLR (R04) is to clear watchdog timer.



WDT/WKT Block Diagram



◇Example: Clear watchdog timer by CLRWDT instruction.

```

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

◇Example: Clear watchdog timer by write WDTCLR register.

```

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

◇Example: Set WKT period and interrupt function.

```

MOVLW   0x000010B                ; R0B.1~0=2 (WKTPSC), WKT period=60 ms @5V
MOVWF   MR0B

MOVLW   0x11110111B              ; Clear WKT interrupt flag by using byte operation
MOVWF   INTIF                    ; Don't use bit operation "BCF WKTIF" to clear flag
                                           ; F-Plane 09H

MOVLW   0x00001000B              ; Enable WKT interrupt function
MOVWF   INTIE
    
```

F03	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
STATUS	GB1	GB0	RAMBK	TO	PD	Z	DC	C
R/W	R/W	R/W	R/W	R	R	R/W	R/W	R/W
Reset	0	0	0	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	–	T2IE	TM1IE	TM0IE	WKTIE	INT2IE	INT1IE	INT0IE
R/W	–	W	W	W	W	W	W	W
Reset	–	0	0	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	–	T2IF	TM1IF	TM0IF	WKTIF	INT2IF	INT1IF	INT0IF
R/W	–	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	–	0	0	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

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 timer

R0B	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
MROB	INT1EDG	INT0EDG	T2PSC		WDTPSC		WKTTPSC	
R/W	R/W	R/W	R/W		R/W		R/W	
Reset	0	0	0	0	1	1	1	1

R0B.3~2 **WDTPSC**: WDT period (@VCC=5V)

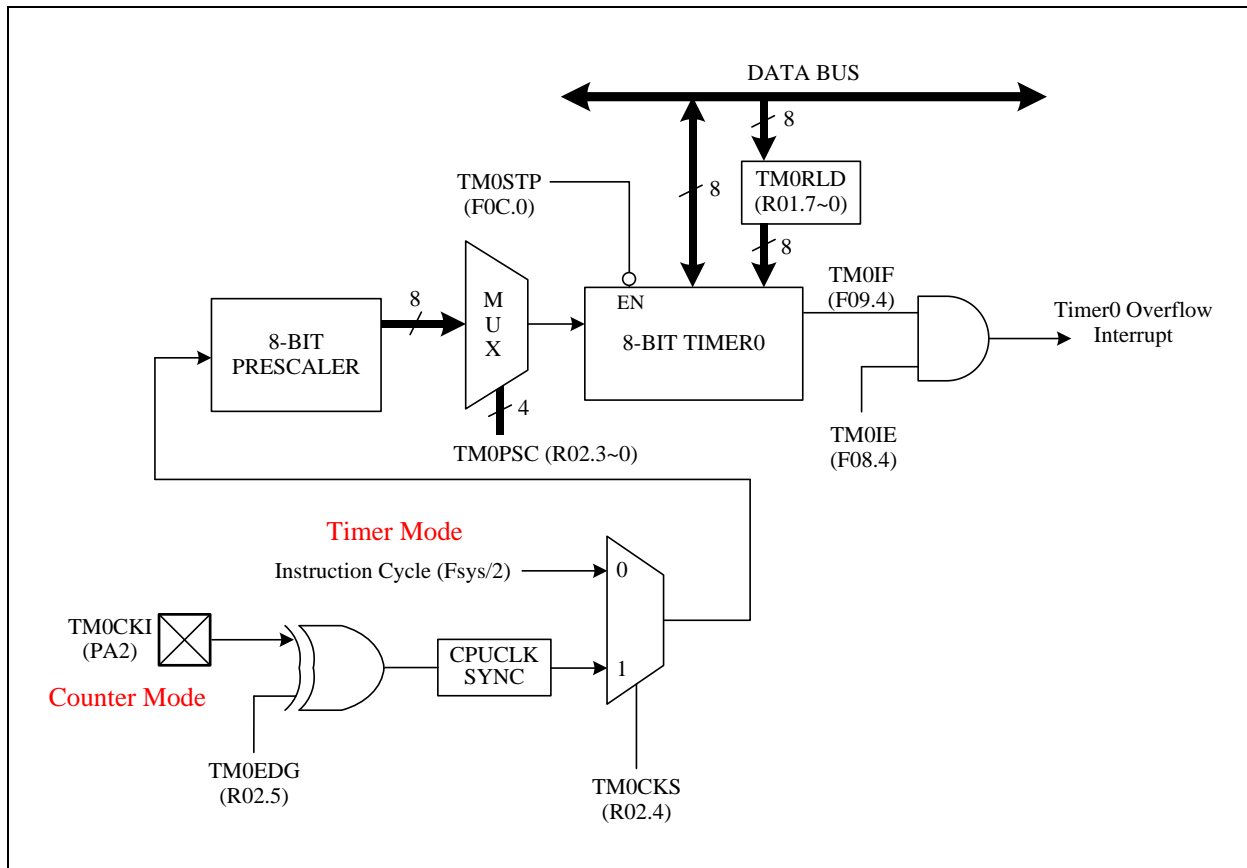
- 00: 120 ms
- 01: 240 ms
- 10: 960 ms
- 11: 1920 ms

R0B.1~0 **WKTTPSC**: WKT period (@VCC=5V)

- 00: 15 ms
- 01: 30 ms
- 10: 60 ms
- 11: 120 ms

### 3.2 Timer0

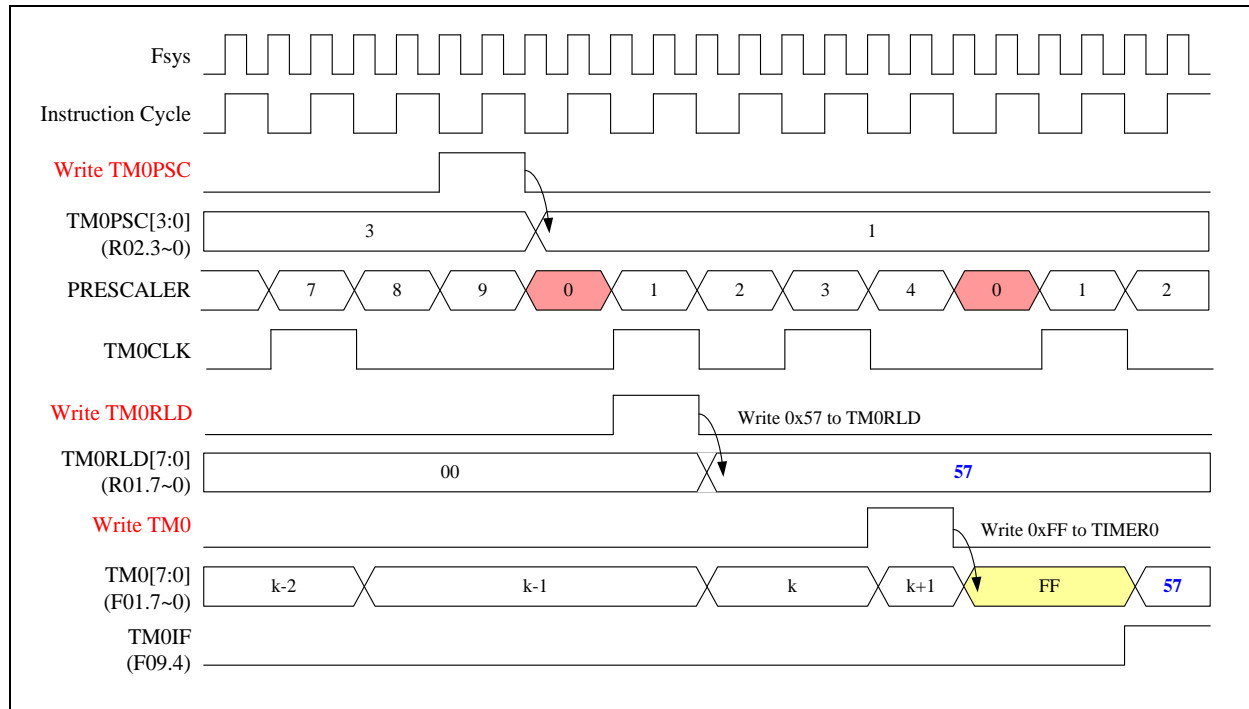
The Timer0 is an 8-bit wide register of F-Plane 01h (TM0). It can be read or written as any other register of F-Plane. Besides, Timer0 increases itself periodically and automatically rolls over based on the pre-scaled clock source, which can be the instruction cycle or TM0CKI (PA2) rising/falling input. The Timer0 increase rate is determined by the TM0PSC (R02.3~0). The Timer0 always generates interrupt flag TM0IF (F09.4) and also reload the new data from TM0RLD (R01.7~0) when its count rolls over. It generates Timer0 Interrupt if the TM0IE (F08.4) bit is set. Timer0 can be stopped counting if the TM0STP (F0C.0) 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 TM0RLD data, 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 time value is as following:

$$\text{Timer0 interrupt interval cycle time} = \text{Instruction cycle time} / \text{TM0PSC} / (256 - \text{TM0RLD})$$

◇Example: Setup TM0 work in Timer mode,  $F_{\text{sys}} = \text{Fast-clock} / \text{CPUPSC} = \text{FIRC } 8 \text{ MHz} / 1 = 8 \text{ MHz}$

; Setup Timer0 clock source and divider

```
BSF          CPUCKS          ; Set Fast-clock as system clock
MOVLW       xxx0 0101B      ; R02.4 = 0 (TM0CKS), TM0 clock = Instruction cycle
MOVWR       TM0CTL          ; R02.3~0 = 5 (TM0PSC), divided by 32
                                     ; Timer0 clock prescaler=Instruction cycle divided by 32
```

; Setup Timer0 reload data

```
MOVLW       80H
MOVWR       TM0RLD          ; Set Timer0 reload data=128
```

; Setup Timer0

```
BSF          TM0STP         ; Disable Timer0 counting (Default "0").
MOVLW       00H
MOVWF       TM0             ; Clear Timer0 content
```



; Enable Timer0 and interrupt function.

```
MOVLW    11101111B    ; Clear Timer0 request interrupt flag by byte operation
MOVWF    INTIF          ; F-Plane 09H
```

```
MOVLW    00010000B    ; Enable Timer0 interrupt function
MOVWF    INTIE         ; F-Plane 08H
```

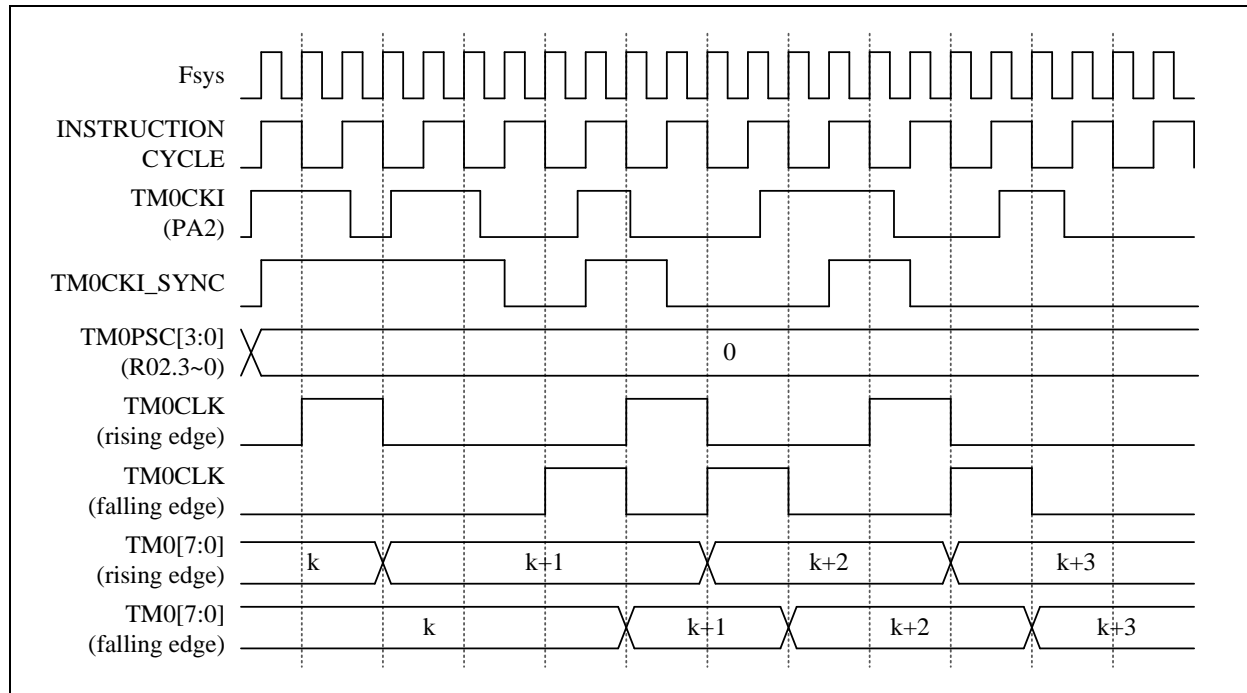
```
BCF      TM0STP        ; Enable Timer0 counting (Default "0").
```

Timer0 clock source is  $F_{sys}/2=8\text{ MHz}/2=4\text{ MHz}$ , Timer0 divided by 32

Timer0 interrupt frequency= $4\text{ MHz}/32/(256-128)=976.56\text{ Hz}$

**Counter Mode:**

If TM0CKS=1 then Timer0 counter source clock is from TM0CKI pin. TM0CKI signal is synchronized by instruction cycle that means the high/low time durations of TM0CKI must be longer than one instruction cycle time to guarantee each TM0CKI's change will be detected correctly by the synchronizer. The following timing diagram describes the Timer0 works in Counter mode.



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

◇Example: Setup Timer0 work in Counter mode and clock source from TM0CKI pin (PA2)  
; Setup Timer0 clock source from TM0CKI pin (PA2) and divider.

```

MOVLW    001 1 0000B
MOVWR    TM0CTL    ; R02.5=1, Select counting edge is falling edge
                    ; R02.4=1, Setup Timer0 clock = TM0CKI pin(PA2)
                    ; R02.3~0=0 (TM0PSC), divided by 1
                    ; TM0 clock prescaler = Instruction cycle divided by 1

```

; Set Timer0 timer and stop TM0 counting.

```

BSF      TM0STP    ; Disable Timer0 counting (Default "0").
MOVLW    00H
MOVWF    TM0       ; Write 0 into Timer0 register of F-Plane 01H.

```

; Start Timer0 count and read Timer0 counter.

```

BCF      TM0STP    ; Enable Timer0 counting.
NOP
NOP
NOP
BSF      TM0STP    ; Disable Timer0 counting (Default "0")
MOVWF    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	–	T2IE	TM1IE	TM0IE	WKTIE	INT2IE	INT1IE	INT0IE
R/W	–	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	–	0	0	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	–	T2IF	TM1IF	TM0IF	WKTIF	INT2IF	INT1IF	INT0IF
R/W	–	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	–	0	0	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

F0C	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
MF0C	–	TCOE	PWM123CLR	PWM0CLR	T2CKS	T2CLR	TM1STP	TM0STP
R/W	–	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	–	0	–	–	–	–	1	0

F0C.0 **TM0STP**: Timer0 counter stop  
 0: Release  
 1: Stop counting

R01	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TM0RLD	TM0RLD							
R/W	R/W							
Reset	0	0	0	0	0	0	0	0

R01.7~0 **TM0RLD**: Timer0 reload data

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

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

R02.4 **TM0CKS**: Timer0 clock source select  
 0: Instruction Cycle ( $F_{sys}/2$ ) as Timer0 prescaler clock  
 1: TM0CKI (PA2) as Timer0 prescaler clock

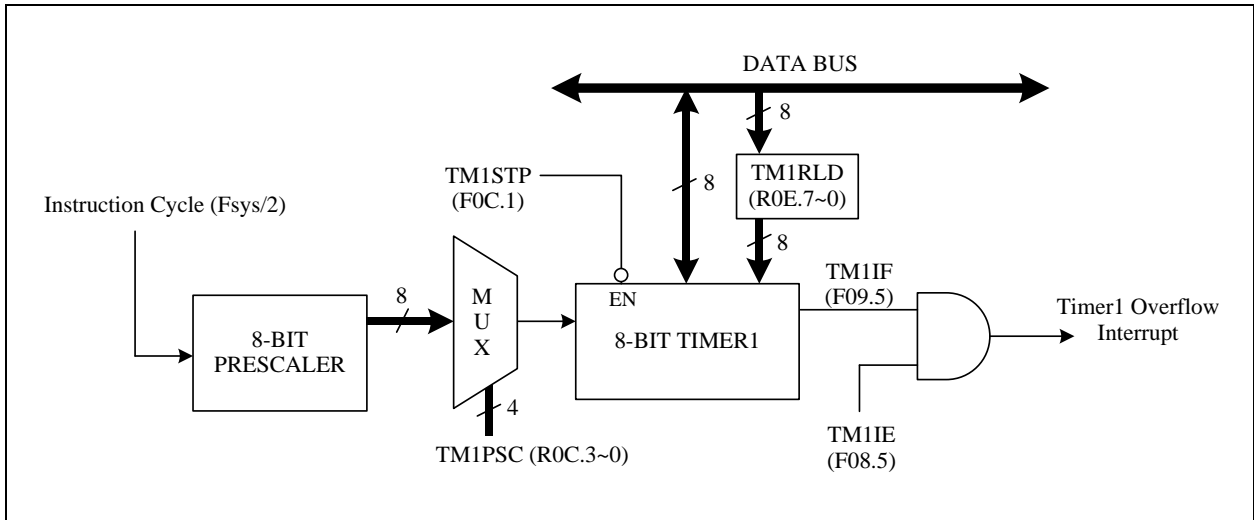


R02.3~0 **TM0PSC:** Timer0 prescaler. Timer0 clock source  
0000: divided by 1  
0001: divided by 2  
0010: divided by 4  
0011: divided by 8  
0100: divided by 16  
0101: divided by 32  
0110: divided by 64  
0111: divided by 128  
1xxx: divided by 256

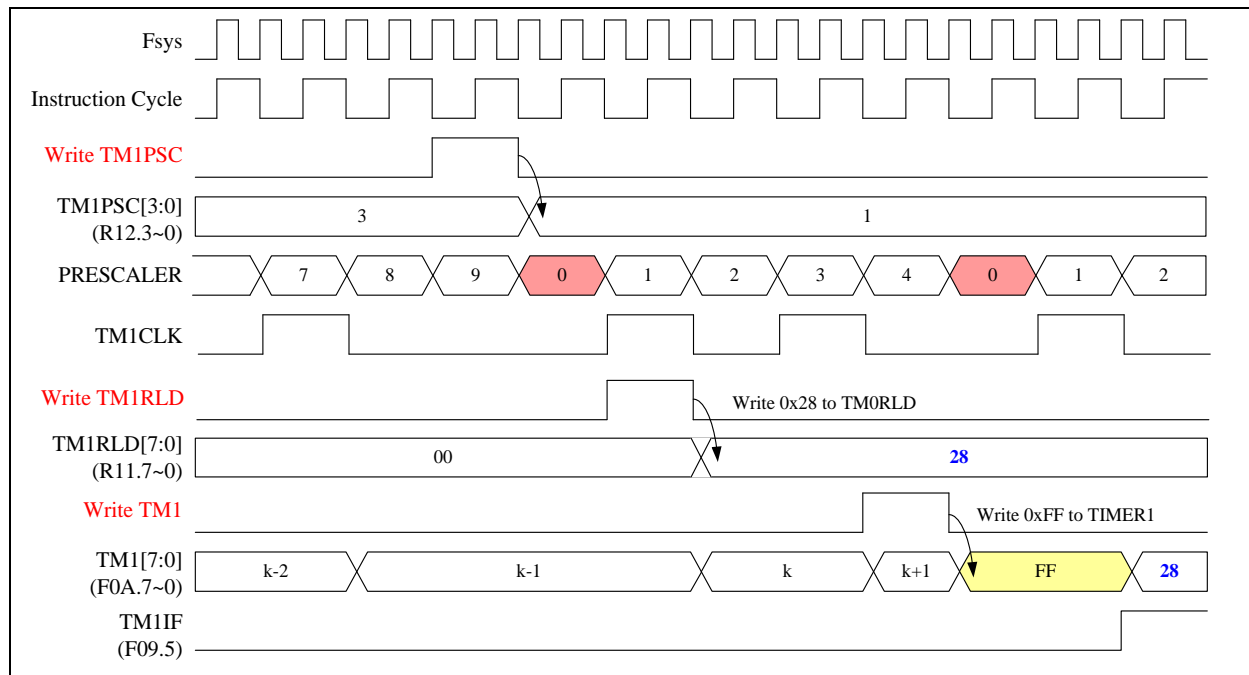


### 3.3 Timer1

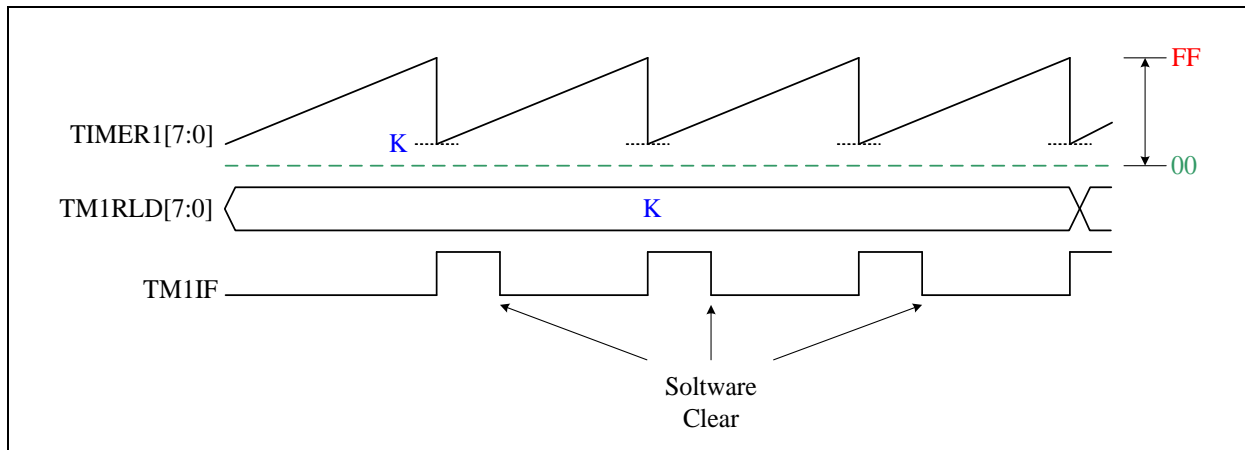
The Timer1 is an 8-bit wide register of F-Plane. It can be read or written as any other register of F-Plane. It is almost the same as Timer0, except Timer1 doesn't have Counter Mode. Timer1 increases itself periodically and automatically rolls over based on the pre-scaled instruction cycle. The Timer1's increasing rate is determined by the TM1PSC (R0C.3~0). The Timer1 can generate interrupt flag TM1IF (F09.5) and also reload the new data from TM1RLD (R0E.7~0) when it rolls over. It generates Timer1 interrupt if the TM1IE (F08.5) bit is set. Timer1 can be stopped counting if the TM1STP (F0C.1) bit is set.



Timer1 Block Diagram



Timer1 Timing Diagram


**Timer1 Reload Diagram**

When the Timer1 prescaler (TM1PSC) is written, the internal 8-bit prescaler will be cleared to 0 to make the counting period correct at the first Timer1 count. TM1CLK is the internal signal that causes the Timer1 to increase by 1 at the end of TM1CLK. TM1WR is also the internal signal that indicates the Timer1 is directly written by instruction; meanwhile, the internal 8-bit prescaler will be cleared. When Timer1 counts from FFh to TM1RLD data, TM1IF (Timer1 Interrupt Flag) will be set to 1 and generate interrupt if TM1IE (Timer1 Interrupt Enable) is set. The timing diagram describes the Timer1 works in pure Timer mode is shown in above.

The equation of Timer1 interrupt frequency is as following:

$$\text{Timer1 interrupt frequency} = \text{Instruction cycle frequency} / \text{TM1PSC} / (256 - \text{TM1RLD})$$

◇Example: Setup Timer1 work in Timer mode.

; Setup Timer1 clock source and divider

```

MOVLW    00000111B    ; F0B.2=1 (CPUCKS), Fsys=Fast-clock
MOVWF    CLKCTL          ; F0B.1~0=3 (CPUPSC), divided by 1
MOVLW    000000101B    ; R0C.3~0=5 (TM1PSC), divided by 32
MOVWR    TM1CTL          ; Select Timer1 clock= (Fsys/2)/32=Fsys/64
    
```

; Set Timer1 timer offset and stops TM1 counting

```

BSF      TM1STP          ; Stop Timer1 counting (Default "0").
MOVLW    F0H            ; Write F0H into Timer1 counter (F18.7~0)
MOVWF    TM1
MOVLW    F0H
MOVWR    TM1RLD         ; Write F0H into Timer1 Reload (R0E.7~0)
    
```

; Enable Timer1 timer and interrupt function.

```

MOVLW    11011111B    ; Clear Timer1 request interrupt flag by byte operation
MOVWF    INTIF          ; F-Plane 09H

MOVLW    00100000B    ; Enable Timer1 interrupt function.
MOVWF    INTIE

BCF      TM1STP          ; Enable Timer1 counting (Default "0").
    
```

F08	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTIE	–	T2IE	TM1IE	TM0IE	WKTIE	INT2IE	INT1IE	INT0IE
R/W	–	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	–	0	0	0	0	0	0	0

F08.5 **TM1IE**: Timer1 interrupt enable  
 0: disable  
 1: enable

F09	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTIF	–	T2IF	TM1IF	TM0IF	WKTIF	INT2IF	INT1IF	INT0IF
R/W	–	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	–	0	0	0	0	0	0	0

F09.5 **TM1IF**: Timer1 interrupt event pending flag  
 This bit is set by H/W while Timer1 overflows, write 0 to this bit will clear this flag

F0C	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
MF0C	–	TCOE	PWM123CLR	PWM0CLR	T2CKS	T2CLR	TM1STP	TM0STP
R/W	–	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	–	0	–	–	–	–	1	0

F0C.1 **TM1STP**: Timer1 counter stop  
 0: Release  
 1: Stop counting

F18	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TM1	TM1							
R/W	R/W							
Reset	0	0	0	0	0	0	0	0

F18 **TM1**: Timer1 content

R0E	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TM1RLD	TM1RLD							
R/W	R/W							
Reset	0	0	0	0	0	0	0	0

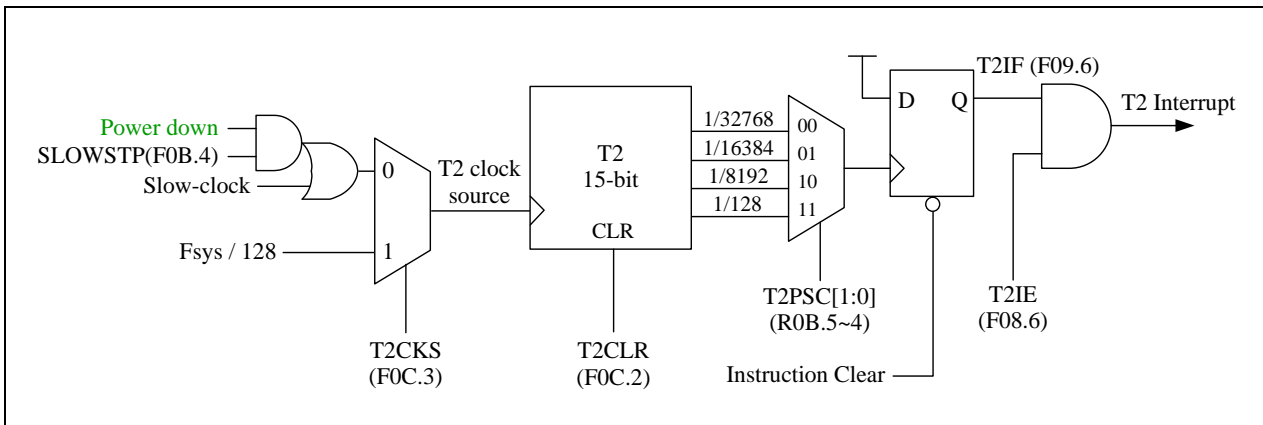
R0E.7~0 **TM1RLD**: Timer1 reload data

R0C	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
MROC	PWM0PSC		PWM123PSC		TM1PSC			
R/W	R/W		R/W		R/W			
Reset	0	0	0	0	0	0	0	0

R0C.3~0 **TM1PSC**: Timer1 prescaler. Timer1 clock source (Fsys/2)  
 0000: divided by 1 (Fsys/2)  
 0001: divided by 2 (Fsys/4)  
 0010: divided by 4 (Fsys/8)  
 0011: divided by 8 (Fsys/16)  
 0100: divided by 16 (Fsys/32)  
 0101: divided by 32 (Fsys/64)  
 0110: divided by 64 (Fsys/128)  
 0111: divided by 128 (Fsys/256)  
 1xxx: divided by 256 (Fsys/512)

### 3.4 T2:15-bit Timer

The T2 is a 15-bit counter and the clock sources are from either  $F_{sys}/128$  or Slow-clock. It is used to generate time base interrupt and T2 counter block clock. It is selected by T2CKS (F0C.3). The T2 content cannot be read by instructions. It generates interrupt flag T2IF (F09.6) with the clock divided by 32768/16384/8192/128 depends on T2PSC[1:0] (R0B.5~4) register bits. The following figure shows the block diagram of T2.



**T2 Block Diagram**

The equation of T2 interrupt frequency is as following:

$$\text{T2 interrupt frequency} = \text{T2 clock source frequency} / \text{T2PSC}$$

◇Example: CPU is running at FAST mode,  $F_{sys} = \text{Fast-clock} / 2 = \text{FIRC} \ 4 \text{ MHz}$

; Setup T2 clock source and divider.

```

MOVLW    xxx00110B    ; F0B.2=1 (CPUCKS), Fsys=Fast-clock
MOVWF    CLKCTL        ; F0B.1~0=2 (CPUPSC), divided by 2
                                ; Fsys=8 MHz/2=4 MHz

MOVLW    xxxx1xxxB
MOVWF    MF0C          ; F0C.3=1 (T2CKS), T2 clock source=Fsys/128

MOVLW    xx01xxxxB
MOVWF    MR0B          ; R0B.5~4=1 (T2PSC), divided by 16384

BSF      T2CLR         ; F0C.2=1, clear T2 counter
    
```

; Enable T2 interrupt function.

```

MOVLW    10111111B    ; Clear T2 request interrupt flag by byte operation
MOVWF    INTIF        ; F-Plane 09H

MOVLW    01000000B    ; Enable T2 interrupt function.
MOVWF    INTIE        ; F-Plane 08H
    
```

T2 clock source is  $F_{sys}/128 = 4 \text{ MHz} / 128 = 31250 \text{ Hz}$ ,  $T2PSC = 16384$

T2 frequency =  $31250 \text{ Hz} / 16384 = 1.907 \text{ Hz}$

## ◇Example:

```

; Setup T2 clock source and divider
    MOVLW    xxxx0xxxB
    MOVWF    MF0C           ; F0C.3=0 (T2CKS), T2 clock source=Slow-clock

    MOVLW    xx00xxxxB
    MOVWF    MR0B           ; R0B.5~4=1 (T2PSC), divided by 32768

    BSF      T2CLR         ; F0C.2=1, clear T2 counter

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

    MOVLW    01000000B    ; Enable T2 interrupt function.
    MOVWF    INTIE         ; F-Plane 08H
    
```

T2 clock source is Slow-clock = 128 KHz, T2PSC=/32768,

T2 frequency=128000 Hz/32768 ≈ 3.91Hz

F08	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTIE	–	T2IE	TM1IE	TM0IE	WKTIE	INT2IE	INT1IE	INT0IE
R/W	–	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	–	0	0	0	0	0	0	0

F08.6 **T2IE**: T2 interrupt enable  
 0: disable  
 1: enable

F09	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTIF	–	T2IF	TM1IF	TM0IF	WKTIF	INT2IF	INT1IF	INT0IF
R/W	–	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	–	0	0	0	0	0	0	0

F09.6 **T2IF**: T2 interrupt event pending flag  
 This bit is set by H/W while T2 overflows, write 0 to this bit will clear this flag

F0B	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
CLKCTL	–	–	–	SLOWSTP	FASTSTP	CPUCKS	CPUPSC	
R/W	–	–	–	R/W	R/W	R/W	R/W	
Reset	–	–	–	0	0	0	1	1

F0B.4 **SLOWSTP**: Slow-clock stop  
 0: Slow-clock is running  
 1: Slow-clock stops running in Power-down mode

F0C	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
MF0C	–	TCOE	PWM123CLR	PWM0CLR	T2CKS	T2CLR	TM1STP	TM0STP
R/W	–	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	–	0	0	0	0	0	1	0

F0C.3 **T2CKS:** T2 clock source select  
 0: Slow-clock  
 1: Fsys/128

F0C.2 **T2CLR:** T2 counter clear  
 0: T2 is counting  
 1: T2 is cleared immediately, this bit is auto cleared by H/W

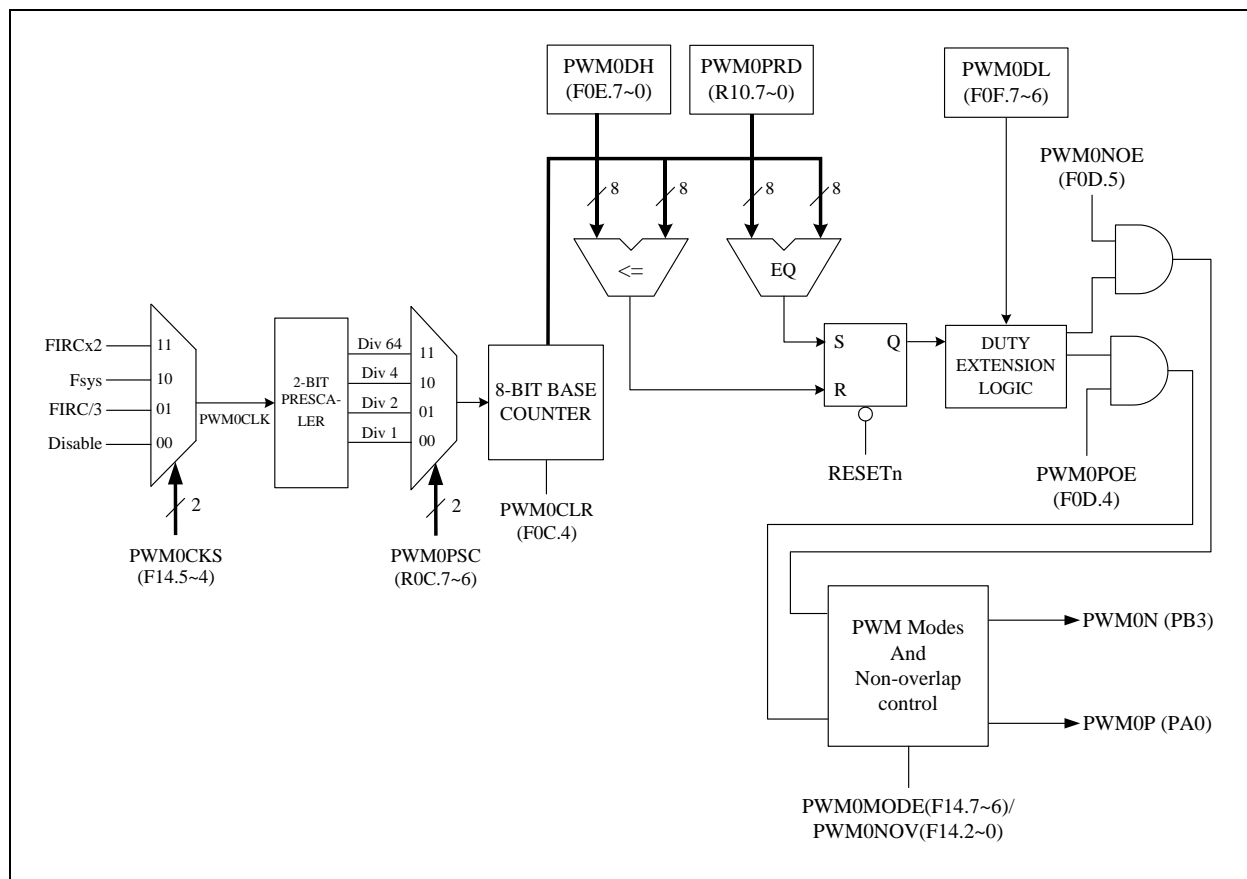
R0B	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
MR0B	INT1EDG	INT0EDG	T2PSC		WDTPSC		WKTPSC	
R/W	R/W	R/W	R/W		R/W		R/W	
Reset	0	0	0	0	1	1	1	1

R0B.5~6 **T2PSC:** T2 prescaler. T2 clock source  
 00: divided by 32768  
 01: divided by 16384  
 10: divided by 8192  
 11: divided by 128

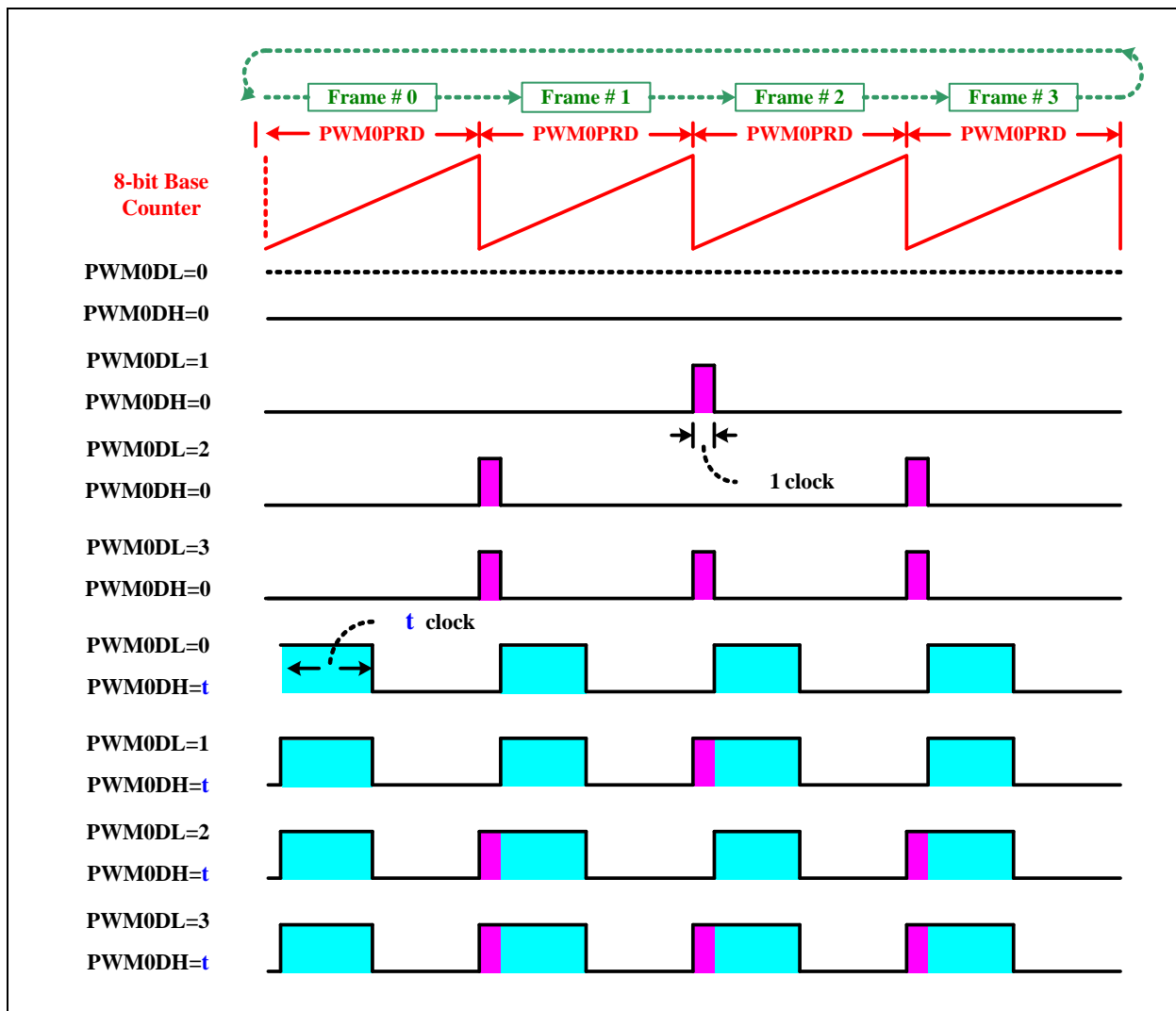
### 3.5 PWM0: (8+2) bits PWM

The PWM can generate various frequency waveform with 1024 duty resolution based on PWM0CLK, which can select Fsys, FIRC 16 MHz, FIRC/3 or disable, decided by PWM0CK[S][1:0] (F14.5~4). A spread LSB technique allows PWM0 to run its frequency at “PWM0CLK divided by 256” instead of “PWM0CLK divided by 1024”, which means the PWM is 4 times faster than normal. The advantage of higher PWM frequency is that the post RC filter can transform the PWM signal to more stable DC voltage level. The PWM output signal reset to low level whenever the 8-bit base counter matches the 8-bit MSB of PWM duty register PWM0DH (F0E.7~0). When the base counter rolls over, the 2-bit LSB of PWM duty register PWM0DL (F0F.7~6) decides whether to set the PWM output signal high immediately or set it high after one clock cycle delay.

The PWM0 period can be set by writing period value to PWM0PRD register (R10.7~0). Note that changing the PWM0PRD will immediately change the PWM0PRD values, which are different from PWM0DH / PWM0DL which has buffer to update the duty at the end of current period. The Programmer must pay attention to the current time to change PWM0PRD by observing the following figure. There is a digital comparator that compares the PWM0 counter and PWM0PRD, if PWM0 counter is larger than PWM0PRD after setting the PWM0PRD, a fault long PWM cycle will be generated because PWM0 counter must count to overflow then keep counting to PWM0PRD to finish the cycle.



PWM0 Block Diagram



PWM0 8+2 Timing Diagram



◇Example: CPU running at Fast mode, Fsys=FIRC 8 MHz

```

; Setup Pin mode
MOVLW    xxxxxx10B    ; R06.1~0=2 (PA0MOD), PA0 Pin mode=Mode2
MOVWR    PAMODL        ; Mode2: CMOS output

MOVLW    10xxxxxxB    ; R08.7~6=2 (PB3MOD), PB3 Pin mode=Mode2
MOVWR    PBMODL        ; Mode2: CMOS output

; Setup PWM0 clock prescaler
BSF      PWM0CLR        ; F0C.4=1, PWM0 clear and hold

MOVLW    xx11xxxxB    ; F14.5~4=3 (PWM0CKS)
MOVWF    PWM0MD        ; PWM0 clock source = FIRCx2
MOVLW    11xxxxxxB    ; R0C.7~6=3(PWM0PSC), PWM0 prescaler / 64
MOVWR    MR0C

MOVOLW   0011x000B    ; F14.7~6=0 (PWM0MODE), PWM0 mode=Mode0
MOVWR    PWM0MD        ; F14.2~0=0 (PWM0NOV), PWM0 non-overlap time=0
                        ; Setup PWM0 mode & Non-overlap control

MOVLW    7FH
MOVWR    PWM0PRD        ; Set PWM0 period=7FH

MOVLW    00xxxxxxB    ; Set PWM0DL duty=00H
MOVWF    PWM0DL

MOVLW    20H
MOVWF    PWM0DH        ; Set PWM0DH duty=20H

MOVLW    xx11xx0xB    ; F0D.5=1 (PWM0NOE), Enable PWM0N output to PB3
MOVWF    PWMCTL        ; F0D.4=1 (PWM0NOE), Enable PWM0P output to PA0
                        ; PWM2 is also output to PB3, so need to set PWM2OE=0
BCF      PWM0CLR        ; F0C.4=0, release PWM0 clear and hold

```

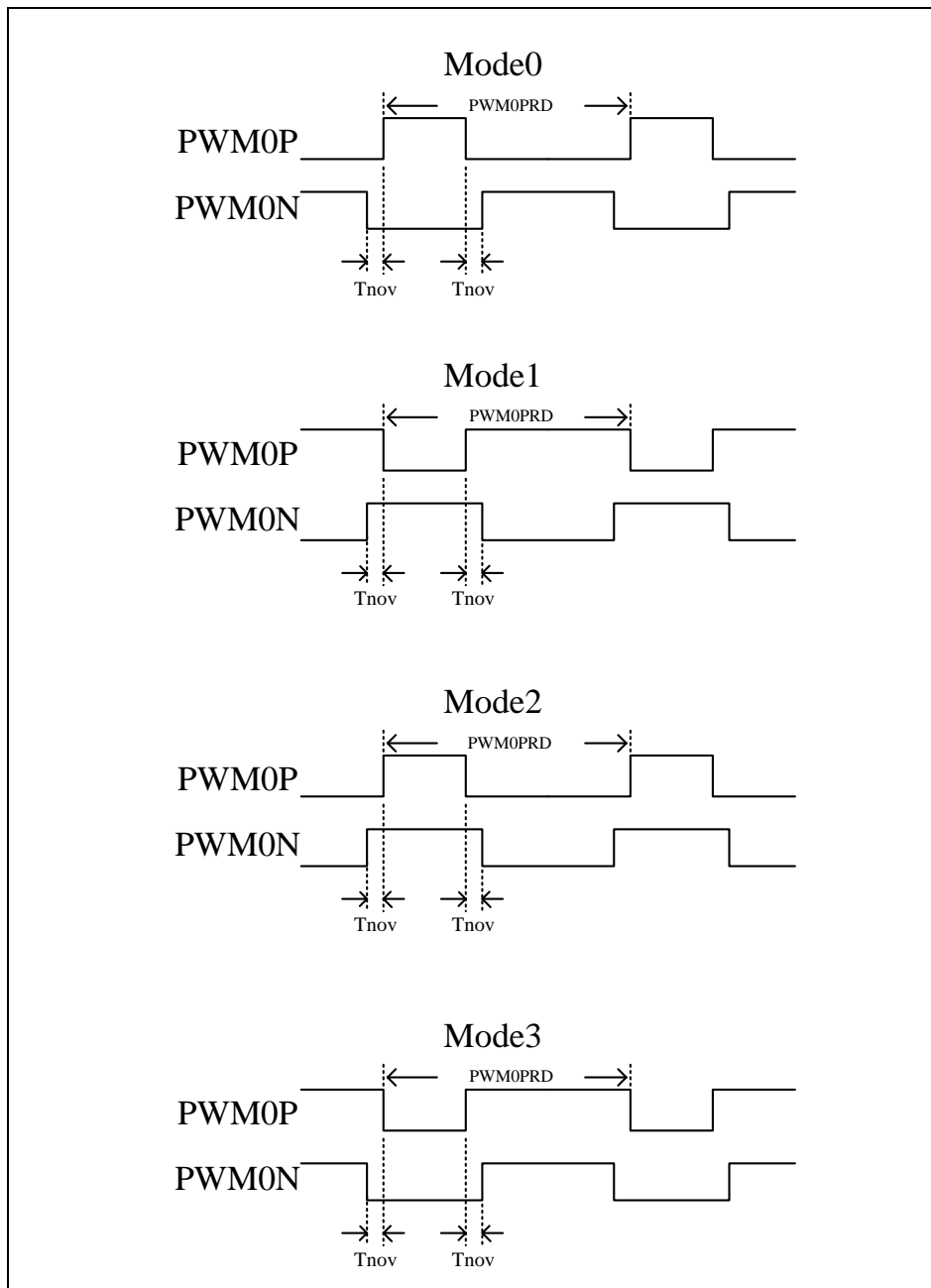
Example:

```

PWM0 clock source=FIRCx2, PWM0PSC=/64, PWM0PRD=7FH,
PWM0DL=00H, PWM0DH=20H
PWM0 output frequency=16 MHz/64/ (PWM0PRD+1) =16 MHz/64/128=1953 Hz.
PWM0P output duty=32:128=25 %.

```

PWM0 can be output via PWM0P and PWM0N with four different modes. The edges of the PWM pulse can be separated with 6 different time non-overlap clocks intervals ( $T_{nov}$ ), 0s, 4 PWM0CLKs, 5 PWM0CLKs, 6 PWM0CLKs, 7 PWM0CLKs, and 8 PWM0CLKs which are selected by PWM0NOV (F14.2~0). The default output form is Mode0. The waveforms of the four output modes are shown below.



PWM0 Waveform Modes

F0C	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
MF0C	–	TCOE	PWM123CLR	PWM0CLR	T2CKS	T2CLR	TM1STP	TM0STP
R/W	–	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	–	0	0	0	0	0	1	0

F0C.4 **PWM0CLR**: PWM0 clear and hold  
 0: PWM0 is running  
 1: PWM0 is clear and hold

F0D	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWMCTL	PWM123CKS		PWM0NOE	PWM0POE	PWM1AOE	PWM1BOE	PWM2OE	PWM3OE
R/W	R/W		R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	0	0	0	0	0	0	0

F0D.5 **PWM0NOE**: Enable PWM0N output to PB3 pin  
 0: disable  
 1: enable

F0D.4 **PWM0POE**: Enable PWM0P output to PA0 pin  
 0: disable  
 1: enable

F0E	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWM0DH	PWM0DH							
R/W	R/W							
Reset	0	0	0	0	0	0	0	0

F0E.7~0 **PWM0DH**: PWM0 duty 8-bit MSB

F0F	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWM0DL	PWM0DL		–	–	–	–	–	–
R/W	R/W		–	–	–	–	–	–
Reset	0	0	–	–	–	–	–	–

F0F.7~6 **PWM0DL**: PWM0 duty 2-bit LSB

F14	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWM0MD	PWM0MODE		PWM0CKS		–	PWM0NOV		
R/W	R/W		R/W		–	R/W		
Reset	0	0	1	0	–	0	0	0

F11.7~6 **PWM0MODE**: PWM0 differential output mode  
 00: Mode 0  
 01: Mode 1  
 10: Mode 2  
 11: Mode 3

F11.5~4 **PWM0CKS**: PWM0 clock source (Fpwm0) select  
 00: Disable  
 01: FIRC/3  
 10: Fsys  
 11: FIRCx2

**F11.2~0 PWM0NOV: PWM0 non-overlap control**

000: original PWM0  
 001: non-overlap 4 PWM0CLKs  
 010: non-overlap 5 PWM0CLKs  
 011: non-overlap 6 PWM0CLKs  
 100: non-overlap 7 PWM0CLKs  
 101: non-overlap 8 PWM0CLKs

R0C	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
MR0C	PWM0PSC		PWM123PSC		TM1PSC			
R/W	R/W		R/W		R/W			
Reset	0	0	0	0	0	0	0	0

**R0C.7~6 PWM0PSC: PWM0 prescaler. PWM0 clock source (Fpwm0)**

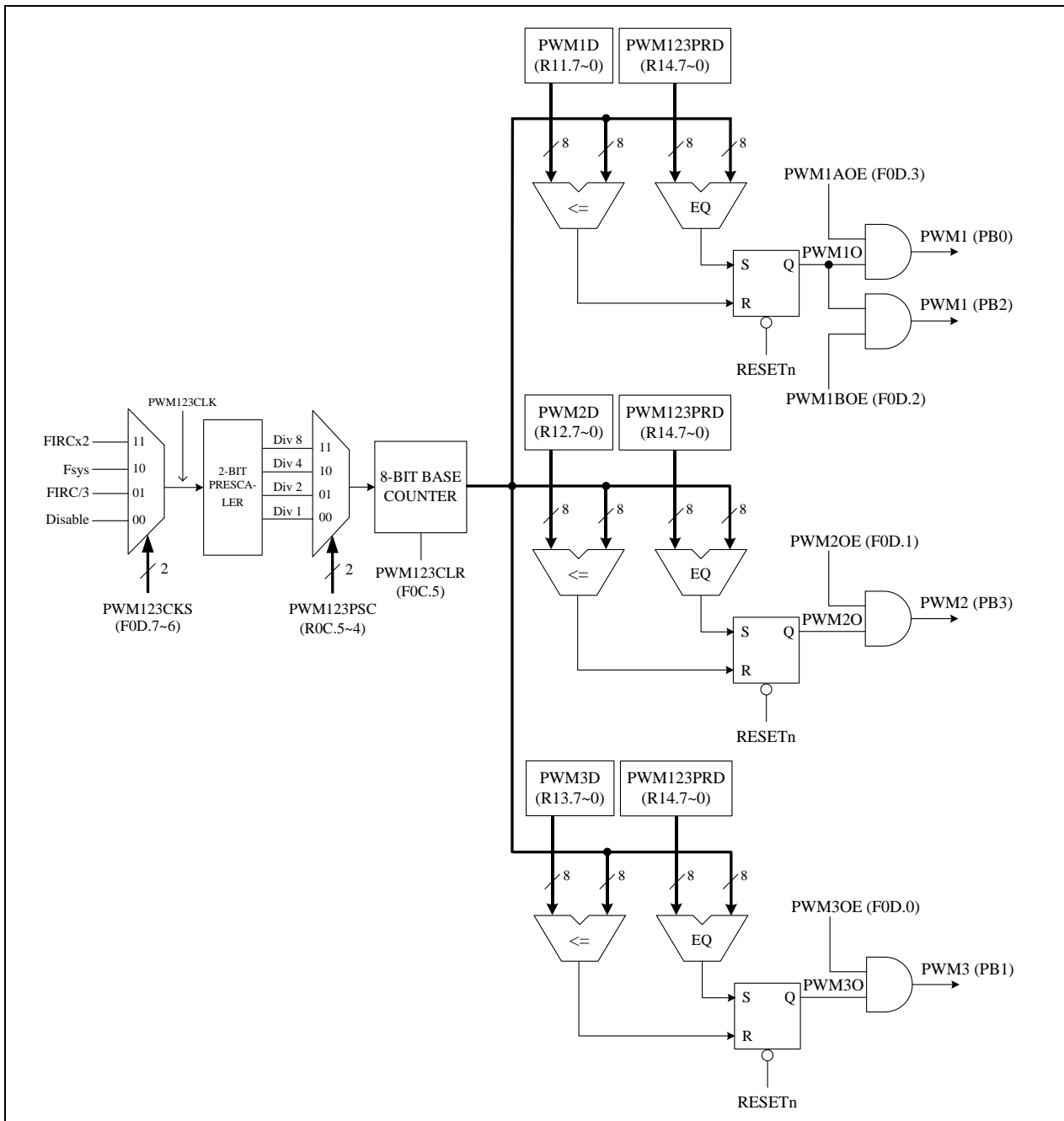
00: divided by 1  
 01: divided by 2  
 10: divided by 4  
 11: divided by 64

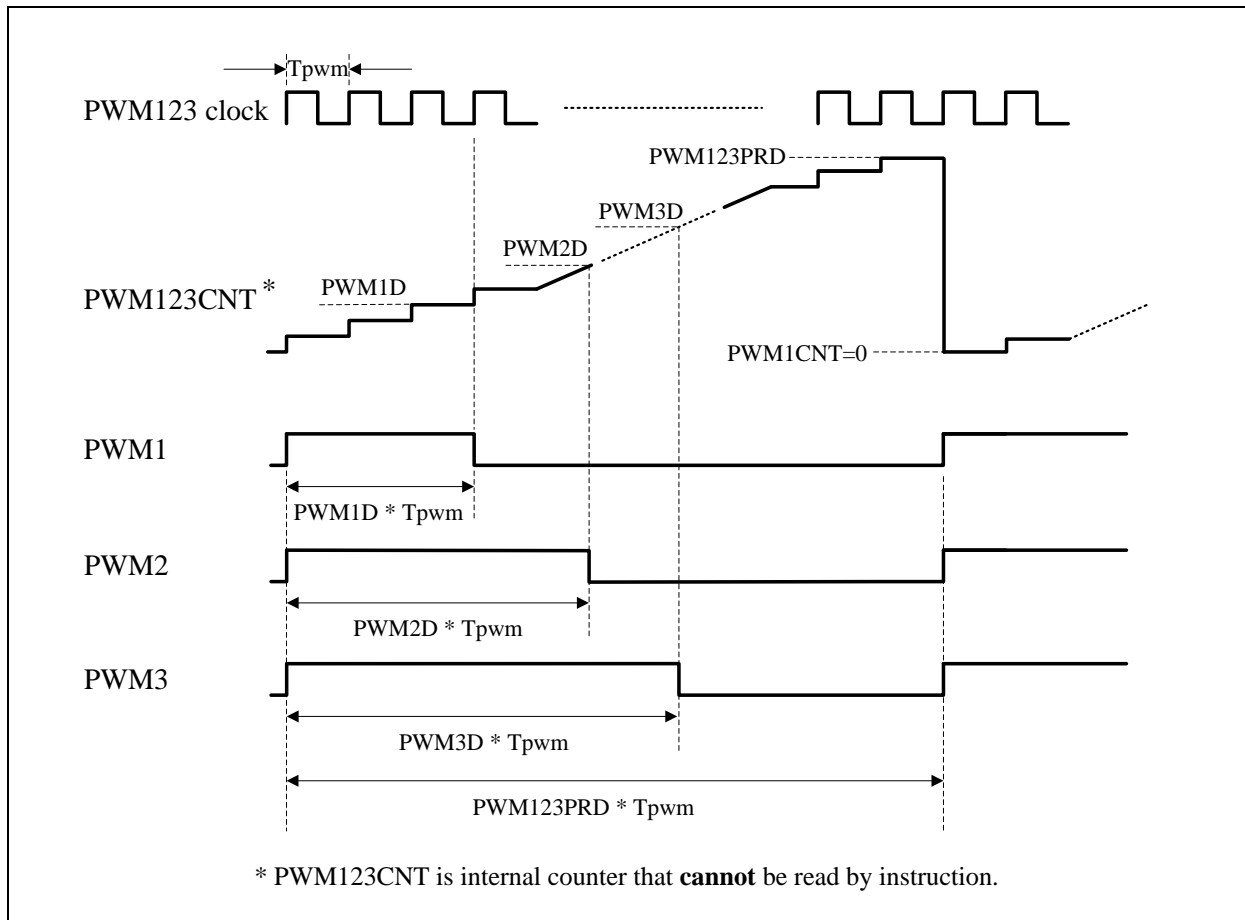
R10	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWM0PRD	PWM0PRD							
R/W	R/W							
Reset	1	1	1	1	1	1	1	1

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

### 3.6 PWM1 / PWM2 / PWM3

This device doesn't only have one 8+2 bits PWM0 but also has three built-in 8-bit PWM generators. There are PWM1, PWM2 and PWM3. All of them use the same clock source. The PWM1~PWM3 clock source can select Fsys, FIRC 16 MHz, FIRC/3 or disable, decided by PWM123CK S[1:0] (F0D.7~6). And it also can be divided by 1, 2, 4, and 8 according to PWM123PSC (R0C.5~4). The PWM1~PWM3 shared one period register PWM123PRD (R14.7~0), and their duty cycle can be changed with writing to their independent duty register PWM1D (F11.7~0), PWM2D (F12.7~0) and PWM3D (F13.7~0). The PWM1~3 will output to I/O by setting PWM1AOE (F0D.3), PWM1BOE (F0D.2), PWM2OE (F0D.1), and PWM3OE (F0D.0). Setting the PWM123CLR (F0C.5) bit will clear the PWM counter. Figure shows the block diagram of PWM1~PWM3.





◇Example: CPU running at Fast mode, F<sub>sys</sub>=FIRC 8 MHz

; Setup Pin mode

```

MOV LW    10 10 10 10B    ; R08.7~6=2 (PB3MOD), PB3 Pin mode=Mode2
                                ; R08.5~4=2 (PB2MOD), PB2 Pin mode=Mode2
                                ; R08.3~2=2 (PB1MOD), PB1 Pin mode=Mode2
                                ; R08.1~0=2 (PB0MOD), PB0 Pin mode=Mode2
                                ; Mode2: CMOS output

MOV WR    PBMODL
    
```

; Setup PWM123 clock prescaler

```

BSF       PWM123CLR    ; F0C.5=1, PWM123 clear and hold

MOV LW    01xxxxxxB    ; F0D.7~6=1 (PWM123CKS)
MOV WF    PWMCTL        ; PWM123 clock source = FIRC/3
MOV LW    xx01xxxxB    ; R0C.5~4=1 (PWM123PSC), PWM123 prescaler/2
MOV WR    MR0C

MOV LW    C7H
MOV WR    PWM123PRD    ; Set PWM123 period=C7H

MOV LW    32H
MOV WR    PWM1D        ; Set PWM1 duty=32H
    
```

```

MOVLW    64H
MOVWR    PWM2D          ; Set PWM2 duty = 64H

MOVLW    96H
MOVWR    PWM3D          ; Set PWM3 duty=96H

MOVLW    01xx1111B    ; F0D.3=1 (PWM1AOE), Enable PWM1 output to PB0
MOVWF    PWMCTL          ; F0D.2=1 (PWM1BOE), Enable PWM1 output to PB2
                                ; F0D.1=1 (PWM2OE), Enable PWM2 output to PB3
                                ; F0D.0=1 (PWM3OE), Enable PWM3 output to PB1

BCF      PWM123CLR      ; F0C.5=0, release PWM123clear and hold
    
```

Example:

PWM1 output duty= $PWM1D / (PWM123PRD + 1) = 50 / (199 + 1) = 1/4$

PWM2 output duty= $PWM2D / (PWM123PRD + 1) = 100 / (199 + 1) = 1/2$

PWM3 output duty= $PWM3D / (PWM123PRD + 1) = 150 / (199 + 1) = 3/4$

PWM123 clock= $FIRC/3 = 8 \text{ MHz} / 3 = 2.66 \text{ MHz}$ , PWM123 clock divided by 2

PWM123 output frequency= $2.66 \text{ MHz} / 2 / (199 + 1) = 6650 \text{ Hz}$

F0C	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
MF0C	–	TCOE	PWM123CLR	PWM0CLR	T2CKS	T2CLR	TM1STP	TM0STP
R/W	–	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	–	0	0	0	0	0	1	0

F0C.5 **PWM123CLR**: PWM1, PWM2 and PWM3 clear and hold

0: PWM1, PWM2 and PWM3 are running

1: PWM1, PWM2 and PWM3 are clear and hold

F0D	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWMCTL	PWM123CKS		PWM0NOE	PWM0POE	PWM1AOE	PWM1BOE	PWM2OE	PWM3OE
R/W	R/W		R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	0	0	0	0	0	0	0

F0D.7~6 **PWM123CKS**: PWM1, PWM2 and PWM3 clock source (Fpwm123) select

00: Disable

01: FIRC/3

10: Fsys

11: FIRCx2

F0D.3 **PWM1AOE**: Enable PWM1 output to PB0 pin

0: disable

1: enable

F0D.2 **PWM1BOE**: Enable PWM1 output to PB2 pin

0: disable

1: enable

F0D.1 **PWM2OE:** Enable PWM2 output to PB3 pin  
 0: disable  
 1: enable

F0D.0 **PWM3OE:** Enable PWM3 output to PB1 pin  
 0: disable  
 1: enable

R0C	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
MR0C	PWM0PSC		PWM123PSC		TM1PSC			
R/W	R/W		R/W		R/W			
Reset	0	0	0	0	0	0	0	0

R0C.5~4 **PWM123PSC:** PWM1~PWM3 prescaler. PWM1, PWM2 and PWM3 clock source (Fpwm123)  
 00: divided by 1  
 01: divided by 2  
 10: divided by 4  
 11: divided by 8

R11	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWM1D	PWM1D							
R/W	R/W							
Reset	0	0	0	0	0	0	0	0

R11.7~0 **PWM1D:** PWM1 duty

R12	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWM2D	PWM2D							
R/W	R/W							
Reset	0	0	0	0	0	0	0	0

R12.7~0 **PWM2D:** PWM2 duty

R13	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWM3D	PWM3D							
R/W	R/W							
Reset	0	0	0	0	0	0	0	0

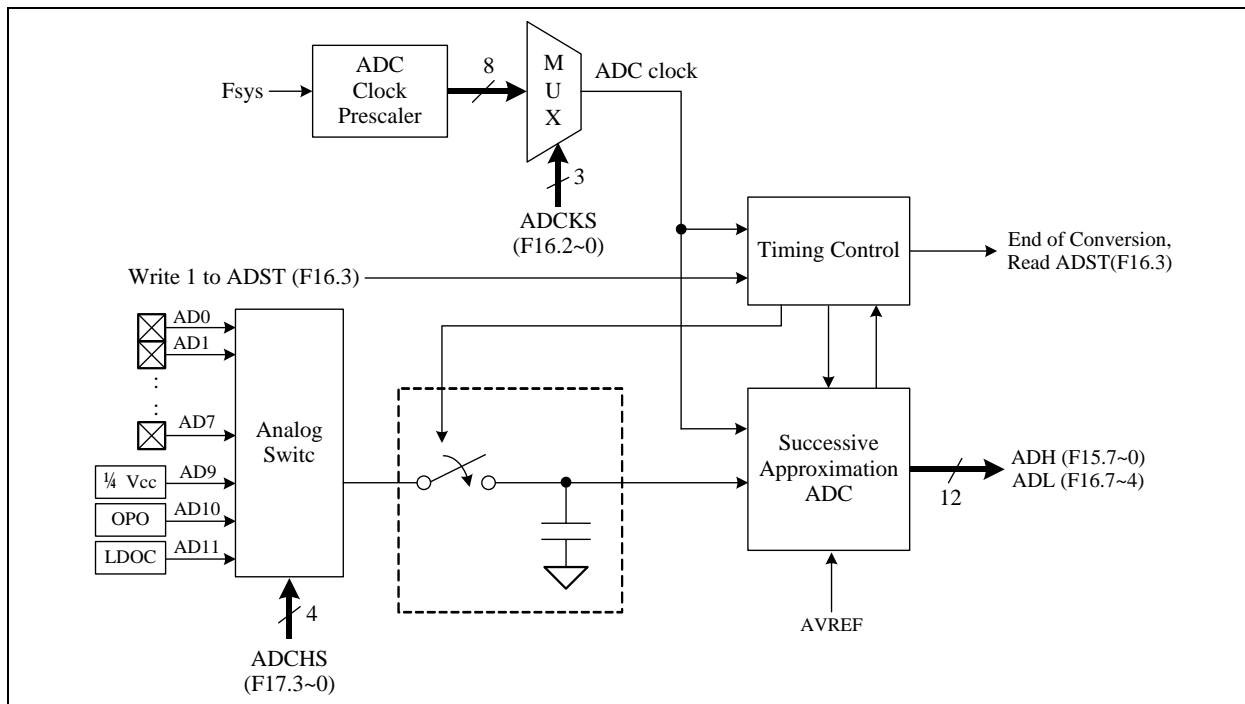
R13.7~0 **PWM3D:** PWM3 duty

R14	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWM123PRD	PWM123PRD							
R/W	R/W							
Reset	1	1	1	1	1	1	1	1

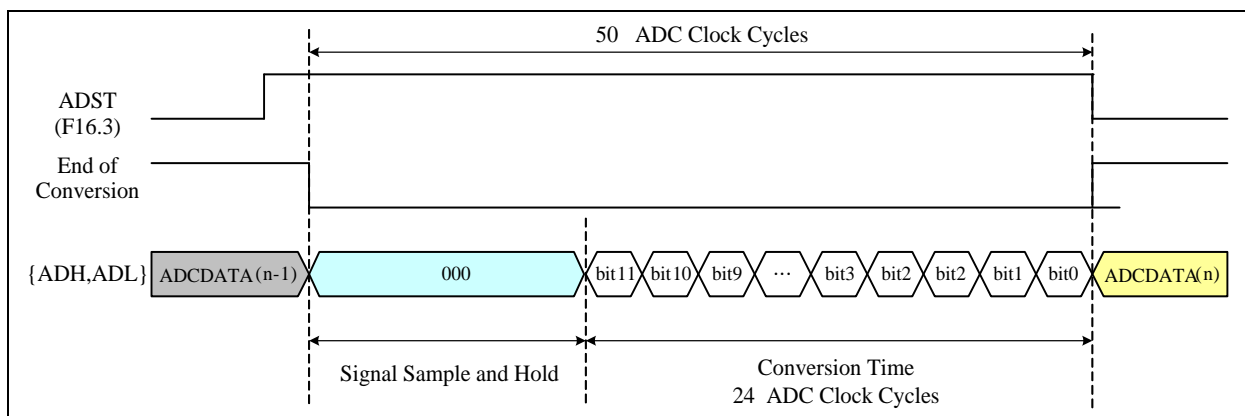
R14.7~0 **PWM123PRD:** PWM1, PWM2 and PWM3 shared period data



### 3.7 Analog-to-Digital Converter



The 12-bit ADC (Analog to Digital Converter) consists of a 11-channel analog input multiplexer, control register, clock generator, 12-bit successive approximation register, and output data register. To use the ADC, user needs to set ADCKS (F16.2~0) to choose a proper ADC clock frequency, which must be less than 1 MHz. User then launches the ADC conversion by setting the ADST (F16.3) control bit. After end of conversion, H/W automatic clears the ADST (F16.3) bit. User can poll this bit to know the conversion status. About pin mode configuration, user needs to set the I/O mode as Mode3 when the pin is used as an ADC input. The setting can disable the pin logical input path to save power consumption. User needs to set ADCHS (F17.3~0) to choose the input channel of ADC. One of them, AD9 is 1/4 Vcc input for ADC. But, for better results, user needs delay 30 uS after setting the ADC input channel=AD9, then begin to use ADC again. In TM57MA18, ADC reference voltage is VCC. It should be noted that the voltage of ADC input channel can't exceed 0.95\*VCC. In TM57MA17, ADC reference voltage is LDOC (1.25V or 2.5V). The reference voltage can be selected by LDO25SEL (R0D.1). On the other hand the voltage of ADC input channel can't exceed 0.95\*2.5V (2.375V), and we must keep LDOPD (R0D.2) = 0 to remain enable LDOC (1.25V or 2.5V) when using the ADC.



Example:

[CPU running at FAST mode , Fsys=FIRC 4 MHz]  
 ADC clock frequency=1 MHz, ADC channel=ADC2 (PA2).

◇Example:

; Setup ADC clock

```

MOV LW   xxx00110B   ; F0B.2=1 (CPUCKS), Fsys=Fast-clock
MOV WF   CLKCTL       ; F0B.1~0=2 (CPUPSC), divided by 2
                        ; Fsys=8 MHz/2=4 MHz

MOV LW   xxxx0110B   ; F16.2~0 (ADCKS), ADC clock=Fsys/4=1 MHz
MOV WF   ADCTL

```

; Setup Pin mode

```

MOV LW   xx11xxxxB   ; R06.5~4=3 (PA2MOD), PA2 Pin mode=Mode3
MOV WR   PAMODL

MOV LW   00000010B   ; F17.3~0=2 (ADCHS), ADC select ADC2 (PA2 pin)
MOV WF   ADCHS

BSF      ADST         ; F16.3 (ADST), ADC start conversion

```

WAIT\_ADC:

```

BTFSC   ADST         ; Wait ADC conversion finish
GOTO    WAIT_ADC

MOV FW   ADH         ; F15.7~0, Read ADC result [11:4] into W
MOV FW   ADCTL       ; F16.7~4, Read ADC result [3:0] into W

:
:

```

F15	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
ADH	ADH							
R/W	R							
Reset	-	-	-	-	-	-	-	-

F15.7~0 **ADH:** ADC output data MSB[11~4]

F16	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
ADCTL	ADL				ADST	ADCKS		
R/W	R				R/W	R/W		
Reset	-	-	-	-	0	0	0	0

F16.7~4 **ADL:** ADC output data LSB[3~0]

F16.3 **ADST:** ADC start bit.  
 0: H/W clear after end of conversion  
 1: ADC start conversion

F16.2~0 **ADCKS:** ADC clock frequency (Fadc) select  
 000: Fsys/256  
 001: Fsys/128  
 010: Fsys/64  
 011: Fsys/32  
 100: Fsys/16  
 101: Fsys/8  
 110: Fsys/4  
 111: Fsys/2

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

F17.3~0 **ADCHS:** ADC channel select  
 0000: ADC0 (PA6)    0100: ADC4 (PA1)    1000: Reserved  
 0001: ADC1 (PA5)    0101: ADC5 (PB4)    1001: 1/4 Vcc  
 0010: ADC2 (PA2)    0110: ADC6 (PB0)    1010: OPO (PB2)  
 0011: ADC3 (PB1)    0111: ADC7 (PA0)    1011: LDOC (1.25V or 2.5V)

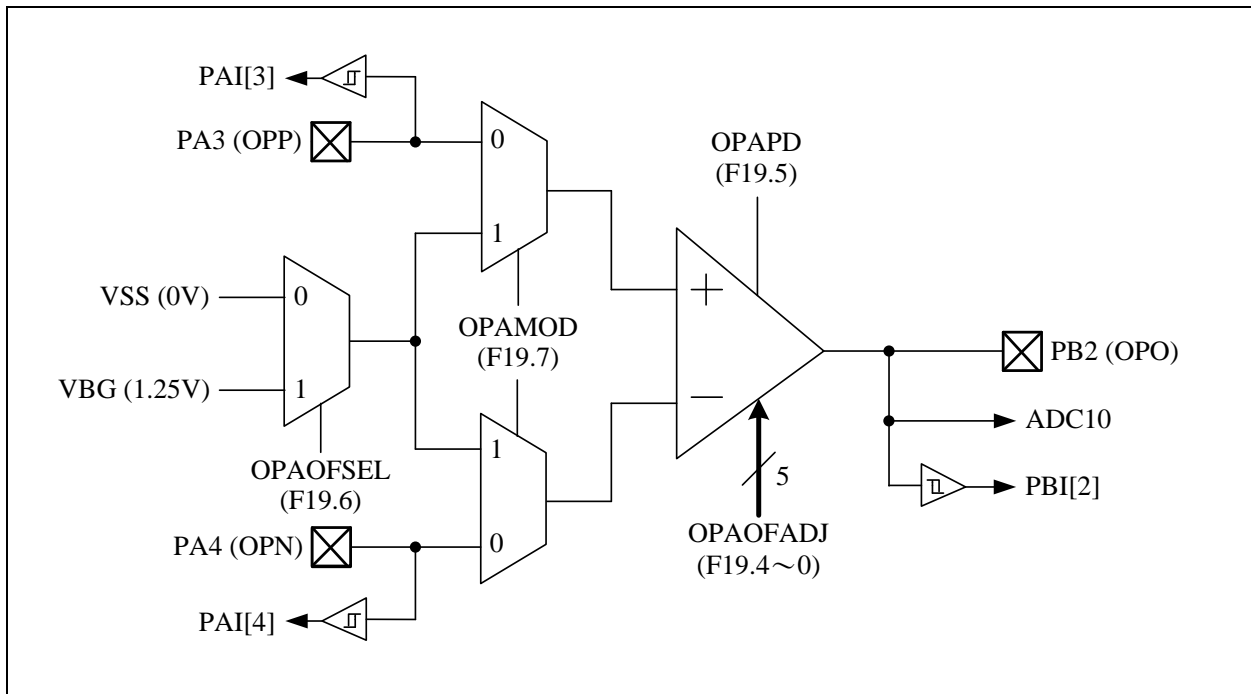
R0D	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWRCTL	CLKFLT	VCCFLT	-	-	IVCSAV	LDOPD	LDO25SEL	MODE3V
R/W	R/W	R/W	-	-	R/W	R/W	R/W	R/W
Reset	0	0	-	-	0	0	1	0

R0D.2 **LDOPD:** Internal LDO (2.5V/1.25V) power down  
 0: LDO running  
 1: LDO power down

R0D.1 **LDO25SEL:** Internal LDO voltage select  
 0: 1.25V  
 1: 2.5V

### 3.7 OPA: Operational Amplifier

There is an operational amplifier (OPA) in this device. The OPA is a CMOS amplifier featuring high input impedance, extremely low offset voltage, high gain and high stability. It allows common mode input voltage range which extends 0V to  $V_{CC}-1.2V$ . This cost-effective device is suitable for high gain, low frequency and low offset voltage application. The OPA is off after IC reset. We could clear OPAPD (F19.5) to turn on the feature of OPA. In addition, the OPA support the build-in offset calibration mode by setting OPAMOD (F19.7). There are two offset voltages (0V or 1.25V) can be selected by OPAOFSEL (F19.6). In the OPA offset calibration mode, user has to change the OPAOFADJ (F19.4~0) value from 00H to 1FH in turn and record the OPAOFADJ value when OPO output low into high. Similarly, changing the OPAOFADJ from 1FH to 00H in turn, we can get another value when OPO output high into low. According to the two values, the suitable OPAOFADJ can be choice for calibration. With I/O mode setting, the corresponding pins have to set as analog mode (Mode3). It can disable the pin logical input path for save power consumption. The OPA block diagram is shown as below. See detailed specifications section on page 96 in the OPA Circuit Characteristics.

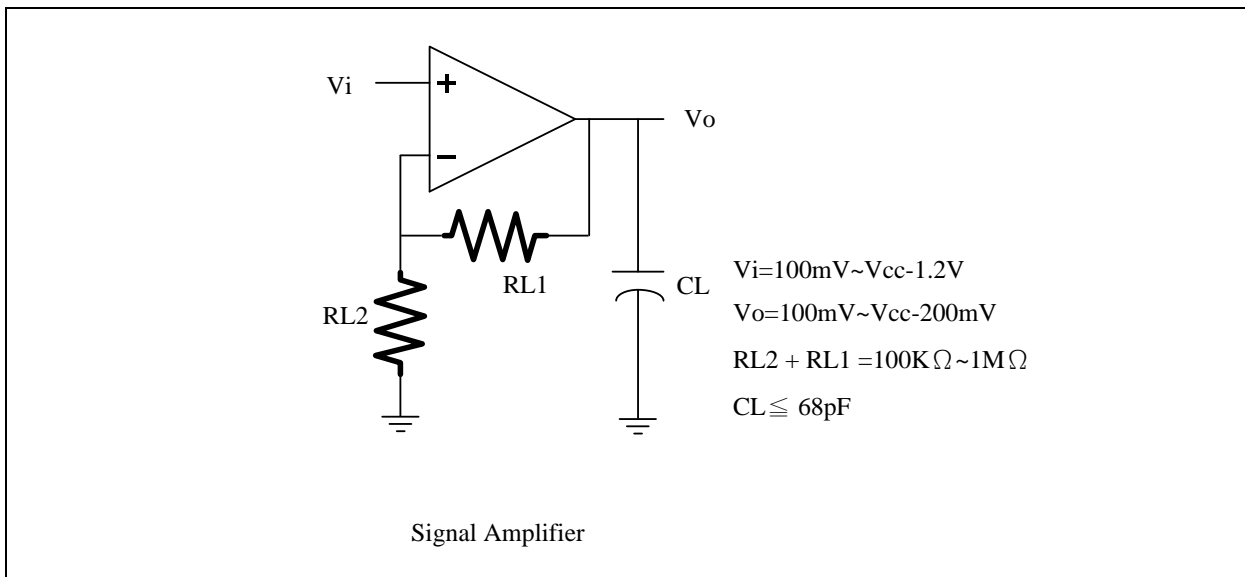
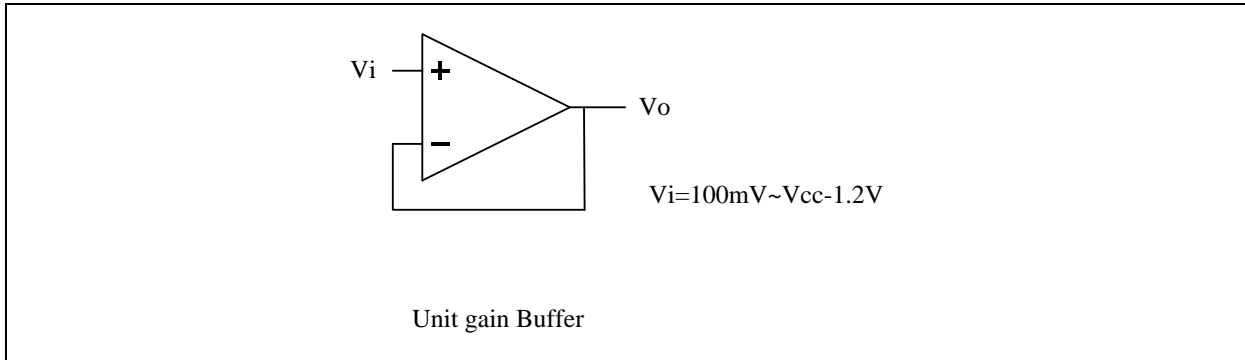


**OPA Block Diagram**

#### Feature:

- Low offset voltage :  $\leq 2$  mV
- Wide Unity Gain Bandwidth : 2.1 MHz
- Open Loop Gain : 120 dB
- Slew Rate : 1.4 V/ $\mu$ s

Example:

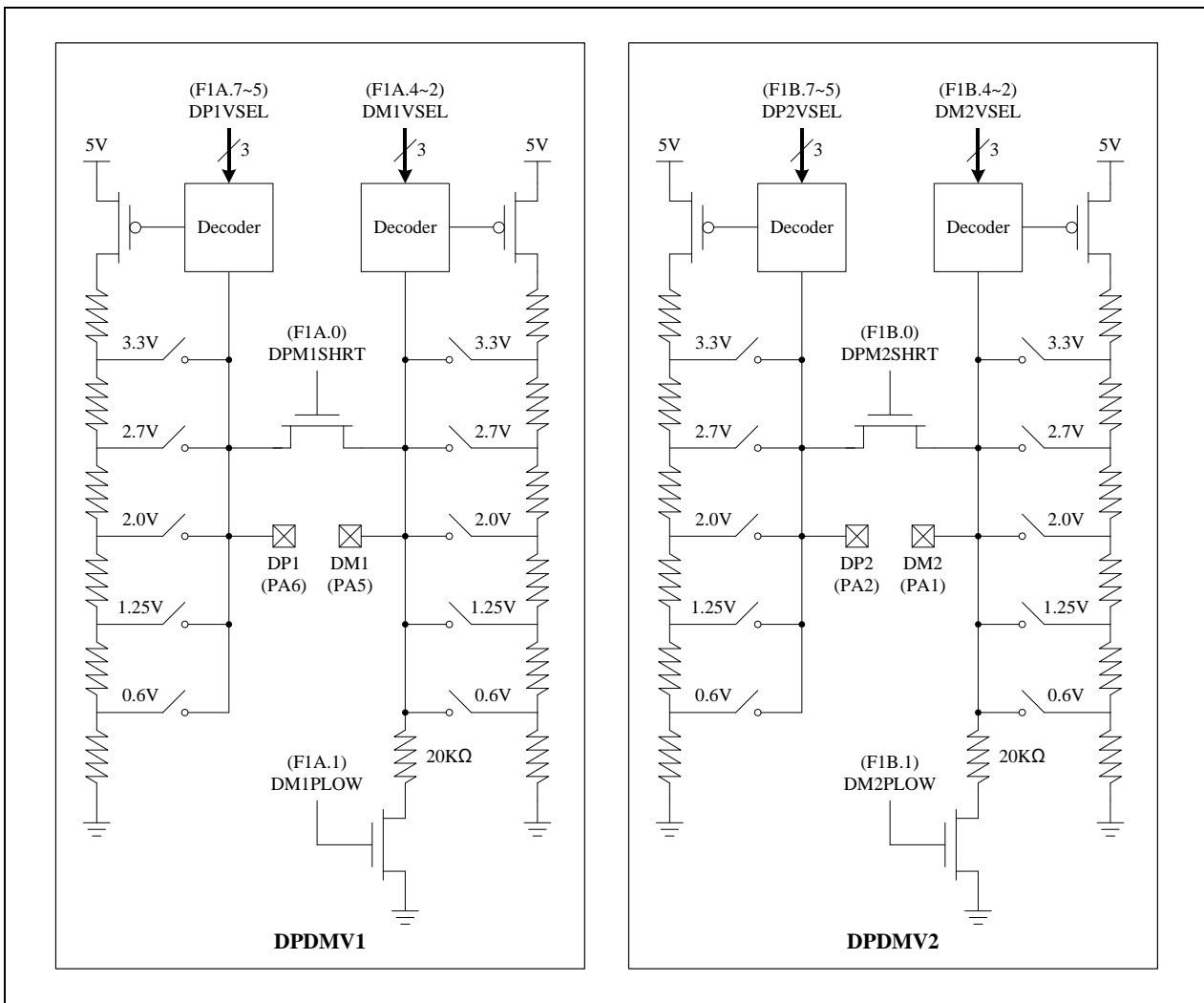


F19	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
OPACTL	OPAMOD	OPAOFSEL	OPAPD	OPAOFADJ				
R/W	R/W	R/W	R/W	R/W				
Reset	0	0	1	0	0	0	0	0

- F19.7    **OPAMOD:** OPA mode select  
           0: Normal mode  
           1: Calibration mode
  
- F19.6    **OPAOFSEL:** OPA offset voltage select  
           0: VSS (0V)  
           1: VBG (1.25V)
  
- F19.5    **OPAPD:** OPA power down  
           0: OPA on  
           1: OPA off
  
- F19.4~0    **OPAOFADJ:** OPA offset adjustment value  
           00000: Lowest voltage  
           ...  
           11111: Highest voltage

### 3.8 DPDMV

There are two sets DPDMV (DPDMV1 and DPDMV2) in this device. The DPDMV is a voltage switch that can output different voltage 3.3V/2.7V/2.0V/1.25V/0.6V to DP and DM by setting DPVSEL and DMVSEL. Besides, DP and DM can be as internal short circuit by setting DPMSHRT. The DM can assign a pull-low resistor by setting DMPLOW register. According the above-mentioned features, this device can support a high performance fast-charging solution following Quick Charge 2.0 High Voltage Dedicated Charging Port (HVDCP) Class A and Class B specification. It can support not only USB BC compliant devices application but also Apple/Samsung devices. With I/O mode setting, the corresponding pins have to set as analog mode (Mode3). It can disable the pin logical input path for save power consumption. The DPDMV block diagram is shown as below. See detailed specifications section on page 96 in the DPDMV Circuit Characteristics.



DPDMV Block Diagram

F1A	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
DPM1CTL	DP1VSEL			DM1VSEL			DM1PLOW	DPM1SHRT
R/W	R/W			R/W			R/W	R/W
Reset	0	0	0	0	0	0	0	0

F1A.7~5 **DP1VSEL**: DP1 (PA6) output voltage select

000: disable  
 001: 0.6V  
 010: 1.2V  
 011: 2.0V  
 100: 2.7V  
 101: 3.3V

F1A.4~2 **DM1VSEL**: DM1 (PA5) output voltage select

000: disable  
 001: 0.6V  
 010: 1.2V  
 011: 2.0V  
 100: 2.7V  
 101: 3.3V

F1A.1 **DM1PLOW**: DM1 pull-low

0: disable  
 1: enable

F1A.0 **DPM1SHRT**: DP1/DM1 short inside

0: DP1/DM1 no short  
 1: DP1/DM1 short together

F1B	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
DPM2CTL	DP2VSEL			DM2VSEL			DM2PLOW	DPM2SHRT
R/W	R/W			R/W			R/W	R/W
Reset	0	0	0	0	0	0	0	0

F1B.7~5 **DP2VSEL**: DP2 (PA2) output voltage select

000: disable  
 001: 0.6V  
 010: 1.2V  
 011: 2.0V  
 100: 2.7V  
 101: 3.3V

F1B.4~2 **DM2VSEL**: DM2 (PA1) output voltage select

000: disable  
 001: 0.6V  
 010: 1.2V  
 011: 2.0V  
 100: 2.7V  
 101: 3.3V

F1B.1 **DM2PLOW**: DM2 pull-low

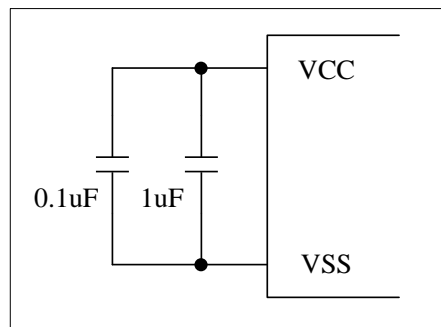
0: disable  
 1: enable

F1B.0 **DPM2SHRT**: DP2/DM2 short inside

0: DP2/DM2 no short  
 1: DP2/DM2 short together

### 3.9 System Clock Oscillator

System clock can be operated in two different oscillation modes. The two oscillation modes are FIRC and SIRC. In the Fast Internal RC mode (FIRC), the on-chip oscillator generates 8 MHz system clock that can be trimmed by IRCF (F1F.7~0). It can separate the frequency into 256 steps. In the Slow Internal RC mode (SIRC), the on-chip oscillator generates 128 KHz system clock. Since power noise degrades the performance of Internal Clock Oscillator, placing power supply bypass capacitors 1  $\mu$ F and 0.1  $\mu$ F very close to VDD/VSS pins to improve the stability of clock and the overall system.



Internal RC Mode

F1F	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
IRCF	IRCF							
R/W	R/W							
Reset	by SYSTEM setting							

F1F.7~0 **IRCF**: FIRC frequency adjustment  
 00H: Lowest frequency  
 .  
 .  
 .  
 FFH: Highest frequency



## 4 I/O Port

### 4.1 PA0-6, PB0-4

These pins can be used as Schmitt-trigger input, CMOS push-pull output or Open-drain output. The pull-up resistor is assignable to each pin by S/W setting. User can set each pin by their pin mode register. There are 4 kinds of pin modes Mode0, Mode1, Mode2 and Mode3 for each pin can be selected.

Mode	PA0-6, PB0-4 pin function	PXn SFR data	Pin State	Resistor Pull-up	Digital Input
<b>Mode 0</b>	Open Drain	0	Drive Low	N	N
		1	Pull-up	Y	Y
<b>Mode 1</b>	Open Drain	0	Drive Low	N	N
		1	Hi-Z	N	Y
<b>Mode 2</b>	CMOS Output	0	Drive Low	N	N
		1	Drive High	N	N
<b>Mode 3</b>	Alternative Function, such as ADC, OPA and DP/DM	X (don't care)	–	N	N

**PA0-6, PB0-4 I/O Pin Function Table**

If a PA0-6, PB0-4 pin is used for Schmitt-trigger input, S/W must set the I/O pin to Mode0 or Mode1 and set the corresponding Port Data SFR to 1 to disable the pin's output driving circuitry. Beside I/O port function, each PA0-6, PB0-4 pin has one or more alternative functions, ADC, OPA or DP/DM. Most of the functions are activated by setting the individual pin mode control SFR to Mode3. Reading the pin data (PA0-6, PB0-4) has different meaning. In "Read-Modify-Write" instruction, CPU actually reads the output data register. In the other instructions, CPU reads the pin state. The so-called "Read-Modify-Write" instruction includes BSF, BCF and all instructions using F-Plane as destination.

Pin Name	Function	INT	ADC	OPA	DP/DM	Mode3
PA0	PWM0P		ADC7			ADC7
PA1		INT1	ADC4		DM2	ADC4/DM2
PA2	TM0CKI		ADC2		DP2	ADC2/ DP2
PA3				OPP		OPP
PA4				OPN		OPN
PA5			ADC1		DM1	ADC1/ DM1
PA6		INT0	ADC0		DP1	ADC0/ DP1
PB0	PWM1		ADC6			ADC6
PB1	PWM3		ADC3			ADC3
PB2	PWM1			OPO		OPO
PB3	PWM2/PWM0N					–
PB4	TCOUT		ADC5			ADC5

**PA0-6, PB0-4 multi-function Table**

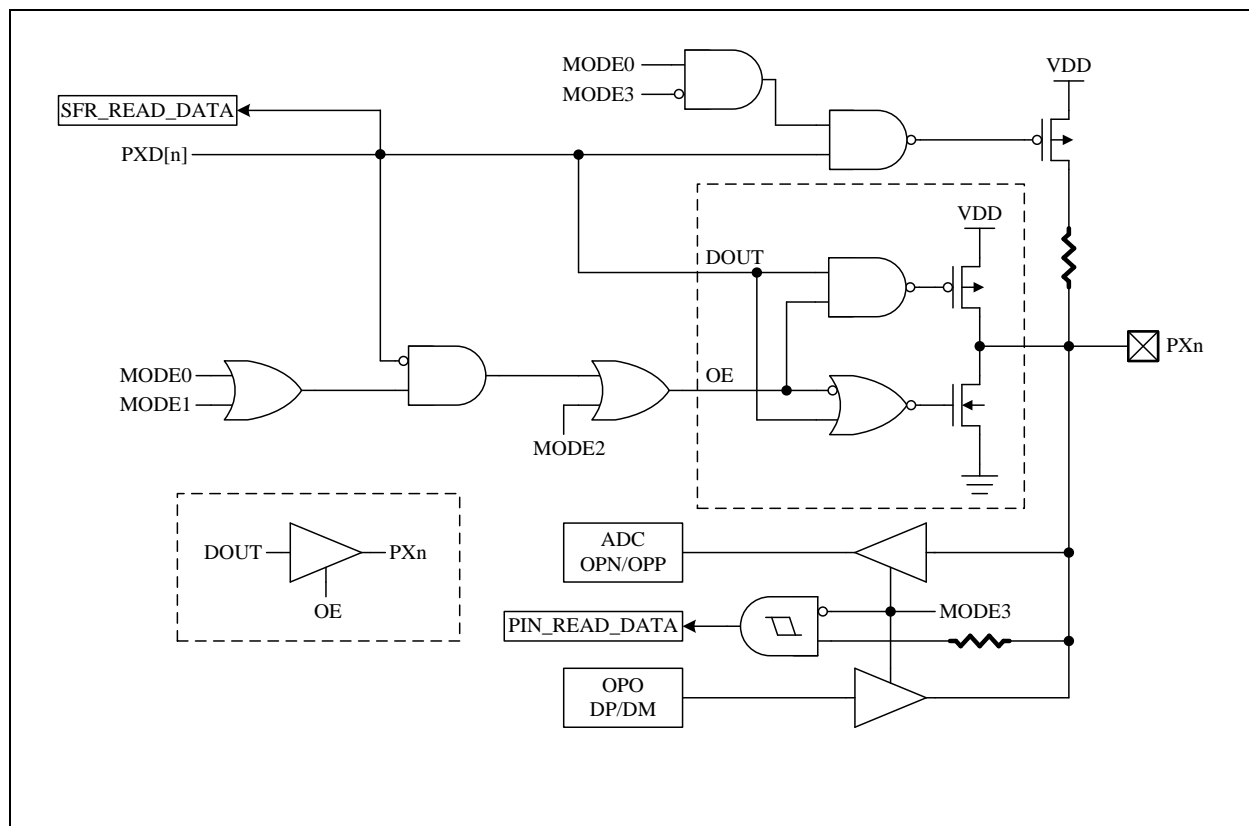
The necessary SFR setting for PA0-6, PB0-4 pin's alternative function is list below.

Alternative Function	Mode	PXn SFR data	Pin State	Other necessary SFR setting
TM0CKI, INT0, INT1	0	1	Input with Pull-up	TM0CTL INTIE
	1	1	Input	
TCOUT	0	X	Clock Open Drain Output with Pull-up	MF0C
	1	X	Clock Open Drain Output	
	2	X	Clock Output (CMOS Push-Pull)	
PWM0, PWM1~ PWM3	0	X	PWM Open Drain Output with Pull-up	PWMCTL
	1	X	PWM Open Drain Output	
	2	X	PWM Output (CMOS Push-Pull)	
ADC0~ADC7	3	X	ADC Channel	ADCHS
OPP, OPN, OPO	3	X	OPA analog I/O	OPACTL
DP1/DM1, DP2/DM2	3	X	USB positive/negative data channel	DPM1CTL DPM2CTL

Mode Setting for PA0-6, PB0-4 Alternative Function

For tables above, a “CMOS Output” pin means it can sink and drive at least 4mA current. It is not recommended to use such pin as input function.

An “Open Drain” pin means it can sink at least 4mA current but only drive a small current (<20μA). It can be used as input or output function and typically needs an external pull up resistor.



PA0-6, PB0-4 Pin Structure

◇Example: Set PA0 as Schmitt-trigger input with pull-up (Mode0)

```

MOVLW    xxxxxxx1B
MOVWF    PAD
MOVLW    xxxxxx00B
MOVWR    PAMODL           ; Set PA0 as Schmitt-trigger input with pull-up
    
```

◇Example: Set PA0 as Schmitt-trigger input without pull-up (Mode1)

```

MOVLW    xxxxxxx1B
MOVWF    PAD
MOVLW    xxxxxx01B
MOVWR    PAMODL           ; Set PA0 as Schmitt-trigger input without pull-up
    
```

◇Example: Set PA0 as CMOS push-pull output mode and drive low (Mode2)

```

MOVLW    xxxxxxx0B
MOVWF    PAD
MOVLW    xxxxxx10B
MOVWR    PAMODL
    
```

◇Example: Set PA0 as CMOS push-pull output mode and PWM0P output (Mode2)

```

MOVLW    xxxxxx10B
MOVWR    PAMODL
MOVLW    xxx1xxxxB
MOVWF    PWMCTL           ; Set PWM0POE=1
    
```

◇Example: Set PA0 as ADC7 input (Mode3)

```

MOVLW    xxxxxx11B           ; Set PA0 as ADC analog input
MOVWR    PAMODL
MOVLW    xxxx0111B           ; Select channel ADC7
MOVWF    ADCHS
    
```

F05	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PAD	PAD							
R/W	R/W							
Reset	1	1	1	1	1	1	1	1

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

F05.6~0 **PAD[6:0]:** PA6~PA0 data  
 0: output low  
 1: output high or Schmitt-trigger input mode

F06	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PBD	–	–	–	PBD				
R/W	–	–	–	R/W				
Reset	–	–	–	1	1	1	1	1

F06.4~0 **PBD:** PB4~PB0 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
PAMODH	–	–	PA6MOD		PA5MOD		PA4MOD	
R/W	–	–	R/W		R/W		R/W	
Reset	–	–	0	1	0	1	0	1

R05.5~4 **PA6MOD**: PA6 Pin Mode Control  
 00: Mode0  
 01: Mode1  
 10: Mode2  
 11: Mode3, ADC0 / DP1

R05.3~2 **PA5MOD**: PA5 Pin Mode Control  
 00: Mode0  
 01: Mode1  
 10: Mode2  
 11: Mode3, ADC1 / DM1

R05.1~0 **PA4MOD**: PA4 Pin Mode Control  
 00: Mode0  
 01: Mode1  
 10: Mode2  
 11: Mode3, OPN

R06	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PAMODL	PA3MOD		PA2MOD		PA1MOD		PA0MOD	
R/W	R/W		R/W		R/W		R/W	
Reset	0	1	0	1	0	1	0	1

R06.7~6 **PA3MOD**: PA3 Pin Mode Control  
 00: Mode0  
 01: Mode1  
 10: Mode2  
 11: Mode3, OPP

R06.5~4 **PA2MOD**: PA2 Pin Mode Control  
 00: Mode0  
 01: Mode1  
 10: Mode2  
 11: Mode3, ADC2/DP2

R06.3~2 **PA1MOD**: PA1 Pin Mode Control  
 00: Mode0  
 01: Mode1  
 10: Mode2  
 11: Mode3, ADC4/DM2

R06.1~0 **PA0MOD**: PA0 Pin Mode Control  
 00: Mode0  
 01: Mode1  
 10: Mode2  
 11: Mode3, ADC7

R07	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PBMODH	–	–	–	–	–	–	PB4MOD	
R/W	–	–	–	–	–	–	R/W	
Reset	–	–	–	–	–	–	0	1

R07.1~0 **PB4MOD**: PB4 Pin Mode Control

- 00: Mode0
- 01: Mode1
- 10: Mode2
- 11: Mode3, ADC5

R08	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PBMODL	PB3MOD		PB2MOD		PB1MOD		PB0MOD	
R/W	R/W		R/W		R/W		R/W	
Reset	0	1	0	1	0	1	0	1

R08.7~6 **PB3MOD**: PB3 Pin Mode Control

- 00: Mode0
- 01: Mode1
- 10: Mode2
- 11: Reserved

R08.5~4 **PB2MOD**: PB2 Pin Mode Control

- 00: Mode0
- 01: Mode1
- 10: Mode2
- 11: Mode3, OPO

R08.3~2 **PB1MOD**: PB1 Pin Mode Control

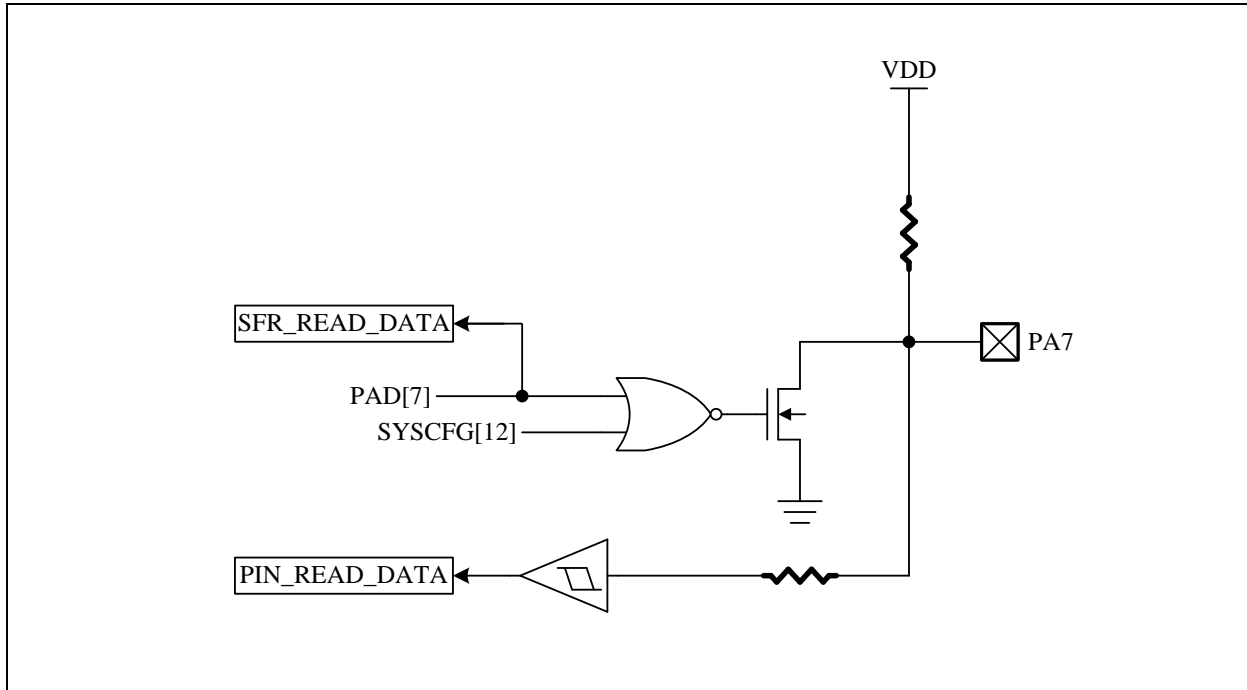
- 00: Mode0
- 01: Mode1
- 10: Mode2
- 11: Mode3, ADC3

R08.1~0 **PB0MOD**: PB0 Pin Mode Control

- 00: Mode0
- 01: Mode1
- 10: Mode2
- 11: Mode3, ADC6

### 4.2 PA7

PA7 can be used in Schmitt-trigger input or open-drain output which is setting by the PAD[7] (F05.7) bit. When the PAD[7] bit is set, PA7 is assigned as Schmitt-trigger input mode, otherwise is assigned as open-drain output mode and output low. The pull-up resistor is always connected to this pin. When SYSCFG[12] is set, PA7 is only used in Schmitt-trigger input for external active low reset.



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

SYSCFG[12]	PAD7	PN STATE	Pull-up	MODE
0	0	Low	Yes	open-drain output with pull-high (not suggest to use this mode)
0	1	High	Yes	input with pull-high
1	X	High	Yes	reset input with pull-high

◇Example: Read state from PA7.

Condition: SYSCFG[12] is set to “0”. If SYSCFG[12] = “1”, then PA7 pin is external reset pin function.

```

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

## MEMORY MAP

### F-Plane

Name	Address	R/W	Rst	Description
<b>(F00) INDF</b> <b>Function related to: RAM W/R</b>				
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
<b>(F01) TM0</b> <b>Function related to: Timer0</b>				
TM0	01.7~0	R/W	0	Timer0 content
<b>(F02) PCL</b> <b>Function related to: PROGRAM COUNT</b>				
PCL	02.7~0	R/W	0	Programming Counter LSB[7~0]
<b>(F03) STATUS</b> <b>Function related to: STATUS</b>				
GB1	03.7	R/W	0	General purpose bit 1
GB0	03.6	R/W	0	General purpose bit 0
TO	03.4	R	0	WDT timeout flag, cleared by PWRST, 'SLEEP' or 'CLRWDI' instruction
PD	03.3	R	0	Sleep mode flag, set by 'SLEEP', cleared by 'CLRWDI' instruction
Z	03.2	R/W	0	Zero flag
DC	03.1	R/W	0	Decimal Carry flag
C	03.0	R/W	0	Carry flag
<b>(F04) FSR</b> <b>Function related to: RAM W/R</b>				
GB2	04.7	R/W	0	General purpose bit 2
FSR	04.6~0	R/W	-	File select register, indirect address mode pointer
<b>(F05) PAD</b> <b>Function related to: Port A</b>				
PAD	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
	05.6~0	R	-	Port A pin or "data register" state
		W	7F	Port A output data register
<b>(F06) PBD</b> <b>Function related to: Port B</b>				
PBD	06.4~0	R	-	Port B pin or "data register" state
		W	FF	Port B output data register
<b>(F08) INTIE</b> <b>Function related to: Interrupt Enable</b>				
T2IE	08.6	R/W	0	T2 interrupt enable 0: disable 1: enable
TM1IE	08.5	R/W	0	Timer1 interrupt enable 0: disable 1: enable
TM0IE	08.4	R/W	0	Timer0 interrupt enable 0: disable 1: enable
WKTIE	08.3	R/W	0	Wakeup Timer interrupt enable 0: disable 1: enable

Name	Address	R/W	Rst	Description
INT2IE	08.2	R/W	0	INT2 (PA7) pin interrupt enable 0: disable 1: enable
INT1IE	08.1	R/W	0	INT1 (PA1) pin interrupt enable 0: disable 1: enable
INT0IE	08.0	R/W	0	INT0 (PA6) pin interrupt enable 0: disable 1: enable
<b>(F09) INTIF Function related to: Interrupt Flag</b>				
T2IF	09.6	R	-	T2 interrupt event pending flag, set by H/W while T2 overflows
		W	0	write 0: clear this flag; write 1: no action
TM1IF	09.5	R	-	Timer1 interrupt event pending flag, set by H/W while Timer1 overflows
		W	0	write 0: clear this flag; write 1: no action
TM0IF	09.4	R	-	Timer0 interrupt event pending flag, set by H/W while Timer0 overflows
		W	0	write 0: clear this flag; write 1: no action
WKTIF	09.3	R	-	WKT interrupt event pending flag, set by H/W while WKT time out
		W	0	write 0: clear this flag; write 1: no action
INT2IF	09.2	R	-	INT2(PA7) interrupt event pending flag, set by H/W at INT2 pin's falling edge
		W	0	write 0: clear this flag; write 1: no action
INT1IF	09.1	R	-	INT1(PA1) interrupt event pending flag, set by H/W at INT1 pin's falling / rising edge
		W	0	write 0: clear this flag; write 1: no action
INT0IF	09.0	R	-	INT0(PA6) interrupt event pending flag, set by H/W at INT0 pin's falling / rising edge
		W	0	write 0: clear this flag; write 1: no action
<b>(F0A) PCH Function related to: PROGRAM COUNT</b>				
PCH	0a.1~0	R/W	0	Programming Counter MSB[9~8]
<b>(F0B) CLKCTL Function related to: Fsys</b>				
SLOWSTP	0b.4	R/W	0	Slow-clock stop 0: Slow-clock is running 1: Slow-clock stops running in Power-down mode
FASTSTP	0b.3	R/W	0	Fast-clock stop 0: Fast-clock is running 1: Fast-clock stops running
CPUCKS	0b.2	R/W	0	System clock source select 0: Slow-clock 1: Fast-clock
CPUPSC	0b.1~0	R/W	3	System clock source prescaler. System clock source 00: divided by 16 01: divided by 4 10: divided by 2 11: divided by 1



Name	Address	R/W	Rst	Description
<b>(F0C) MF0C</b>				<b>Function related to: Fsys/PWM0~3/Timer0/Timer1/T2</b>
TCOE	0c.6	R/W	0	Enable Instruction Cycle (Fsys/2) output to PB4 pin (TCOUT) 0: disable 1: enable
PWM123CLR	0c.5	R/W	0	PWM1/PWM2/PWM3 clear and hold 0: PWM1/PWM2/PWM3 are running 1: PWM1/PWM2/PWM3 are clear and hold
PWM0CLR	0c.4	R/W	0	PWM0 clear and hold 0: PWM0 is running 1: PWM0 is clear and hold
T2CKS	0c.3	R/W	0	T2 clock select 0: Slow-clock 1: Fsys/128
T2CLR	0c.2	R/W	0	T2 counter clear 0: T2 is counting 1: T2 is cleared immediately, this bit is auto cleared by H/W
TM1STP	0c.1	R/W	0	Timer1 counter stop 0: Release 1: Stop counting
TM0STP	0c.0	R/W	0	Timer0 counter stop 0: Release 1: Stop counting
<b>(F0D) PWMCTL</b>				<b>Function related to: PWM0/PWM1/PWM2/PWM3</b>
PWM123CKS	0d.7~6	R/W	2	PWM1, PWM2 and PWM3 clock source (Fpwm123) select 00: Disable 01: FIRC/3 10: Fsys 11: FIRCx2
PWM0NOE	0d.5	R/W	0	Enable PWM0N output to PB3 pin 0: disable 1: enable
PWM0POE	0d.4	R/W	0	Enable PWM0P output to PA0 pin 0: disable 1: enable
PWM1AOE	0d.3	R/W	0	Enable PWM1 output to PB0 pin 0: disable 1: enable
PWM1BOE	0d.2	R/W	0	Enable PWM1 output to PB2 pin 0: disable 1: enable
PWM2OE	0d.1	R/W	0	Enable PWM2 output to PB3 pin 0: disable 1: enable
PWM3OE	0d.0	R/W	0	Enable PWM3 output to PB1 pin 0: disable 1: enable
<b>(F0E) PWM0DH</b>				<b>Function related to: PWM0</b>
PWM0DH	0e.7~0	R/W	0	PWM0 duty 8-bit MSB
<b>(F0F) PWM0DL</b>				<b>Function related to: PWM0</b>
PWM0DL	0f.7~6	R/W	0	PWM0 duty 2-bit LSB

Name	Address	R/W	Rst	Description
<b>(F14) PWM0MD</b>				<b>Function related to: PWM0</b>
PWM0MODE	14.7~6	R/W	0	PWM0 differential output mode 00: Mode0 01: Mode1 10: Mode2 11: Mode3
PWM0CKS	14.5~4	R/W	2	PWM0 clock source (Fpwm0) select 00: Disable 01: FIRC/3 10: Fsys 11: FIRCx2
PWM0NOV	14.2~0	R/W	0	PWM0 non-overlap control 000: original PWM0 001: non-overlap 4 PWM0CLKs 010: non-overlap 5 PWM0CLKs 011: non-overlap 6 PWM0CLKs 100: non-overlap 7 PWM0CLKs 101: non-overlap 8 PWM0CLKs
<b>(F15) ADH</b>				<b>Function related to: ADC</b>
ADH	15.7~0	R	-	ADC output data MSB[11~4]
<b>(F16) ADCTL</b>				<b>Function related to: ADC</b>
ADL	16.7~4	R	-	ADC output data LSB[3~0]
ADST	16.3	R/W	0	ADC start bit. 0: H/W clear after end of conversion 1: ADC start conversion
ADCKS	16.2~0	R/W	0	ADC clock frequency (Fadc) select 000: Fsys/256 001: Fsys/128 010: Fsys/64 011: Fsys/32 100: Fsys/16 101: Fsys/8 110: Fsys/4 111: Fsys/2
<b>(F17) ADCHS</b>				<b>Function related to: ADC</b>
ADCHS	17.3~0	R/W	0	ADC channel select 0000: ADC0 (PA6) 0001: ADC1 (PA5) 0010: ADC2 (PA2) 0011: ADC3 (PB1) 0100: ADC4 (PA1) 0101: ADC5 (PB4) 0110: ADC6 (PB0) 0111: ADC7 (PA0) 1000: Reserved 1001: 1/4 Vcc 1010: OPO (PB2) 1011: LDOC (1.25V or 2.5V, select by LDO25SEL (R0D.1))
<b>(F18) TM1</b>				<b>Function related to: Timer1</b>
TM1	18.7~0	R/W	0	Timer1 content

Name	Address	R/W	Rst	Description
<b>(F19) OPACTL</b>				<b>Function related to: OPA</b>
OPAMOD	19.7	R/W	0	OPA mode select 0: Normal mode 1: Calibration mode
OPAOFSEL	19.6	R/W	1	OPA offset voltage select 0: VSS (0V) 1: VBG (1.25V)
OPAPD	19.5	R/W	0	OPA power down 0: OPA on 1: OPA off
OPAOFADJ	19.4~0	R/W	0	OPA offset adjustment value 00000: Lowest voltage . . . 11111: Highest voltage
<b>(F1A) DPM1CTL</b>				<b>Function related to: DP1/DM1</b>
DP1VSEL	1a.7~5	R/W	0	DP1 (PA6) output voltage select 000: disable 001: 0.6V 010: 1.2V 011: 2.0V 100: 2.7V 101: 3.3V
DM1VSEL	1a.4~2	R/W	0	DM1 (PA5) output voltage select 000: disable 001: 0.6V 010: 1.2V 011: 2.0V 100: 2.7V 101: 3.3V
DM1PLOW	1a.1	R/W	0	DM1 pull-low 0: disable 1: enable
DPM1SHRT	1a.0	R/W	0	DP1/DM1 short inside 0: DP1/DM1 no short 1: DP1/DM1 short together
<b>(F1B) DPM2CTL</b>				<b>Function related to: DP2/DM2</b>
DP2VSEL	1b.7~5	R/W	0	DP2 (PA2) output voltage select 000: disable 001: 0.6V 010: 1.2V 011: 2.0V 100: 2.7V 101: 3.3V
DM2VSEL	1b.4~2	R/W	0	DM2 (PA1) output voltage select 000: disable 001: 0.6V 010: 1.2V 011: 2.0V 100: 2.7V 101: 3.3V

Name	Address	R/W	Rst	Description
DM2PLOW	1b.1	R/W	0	DM2 pull-low 0: disable 1: enable
DPM2SHRT	1b.0	R/W	0	DP2/DM2 short inside 0: DP2/DM2 no short 1: DP2/DM2 short together
<b>(F1C) RSR</b>		<b>Function related to: RAM W/R</b>		
RSR	1c.7~0	R/W	-	R-Plane file select register, indirect address mode pointer
<b>(F1D) DPL</b>		<b>Function related to: Table Read</b>		
DPL	1d.7~0	R/W	0	Table read low address, data ROM pointer (DPTR) low byte[7~0]
<b>(F1E) DPH</b>		<b>Function related to: Table Read</b>		
DPH	1e.1~0	R/W	0	Table read high address, data ROM pointer (DPTR) high byte[9~8]
<b>(F1F) IRCF</b>		<b>Function related to: Internal RC</b>		
IRCF	1f.7~0	R/W	by SYS.	FIRC frequency adjustment 00H: Lowest frequency . . . FFH: Highest frequency
<b>User Data Memory</b>				
FRAM	20~3F	R/W	-	FRAM bit-addressable area (32 Bytes)
	40~7F	R/W	-	FRAM common area (64 Bytes)

**R-Plane**

Name	Address	R/W	Rst	Description
<b>(R00) INDR</b>				<b>Function related to: RAM W/R</b>
INDR	00.7~0	R/W	-	Not a physical register, addressing INDR actually point to the register whose address is contained in the RSR register
<b>(R01) TM0RLD</b>				<b>Function related to: Timer0</b>
TM0RLD	01.7~0	R/W	0	Timer0 reload data
<b>(R02) TM0CTL</b>				<b>Function related to: Timer0</b>
TM0EDG	02.5	R/W	0	TM0CKI (PA2) edge selection for Timer0 prescaler count 0: TM0CKI rising edge for Timer0 prescaler count 1: TM0CKI falling edge for Timer0 prescaler count
TM0CKS	02.4	R/W	0	Timer0 clock source select 0: Instruction Cycle (Fsys/2) as Timer0 prescaler clock 1: TM0CKI (PA2) as Timer0 prescaler clock
TM0PSC	02.3~0	R/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
<b>(R03) PWRDN</b>				<b>Function related to: Power Down</b>
PWRDN	03	W	-	Write this register to enter STOP/IDLE Mode (i.e. 'SLEEP' instruction)
<b>(R04) WDTCLR</b>				<b>Function related to: WDT</b>
WDTCLR	04	W	-	Write this register to clear WDT timer (i.e. 'CLRWDT' instruction)
<b>(R05) PAMODH</b>				<b>Function related to: Port A</b>
PA6MOD	05.5~4	R/W	1	PA6~PA4 I/O mode control 00: Mode0 01: Mode1 10: Mode2 11: Mode3, used as analog I/O (ADC, OPA, DP/DM)
PA5MOD	05.3~2	R/W	1	
PA4MOD	05.1~0	R/W	1	
<b>(R06) PAMODL</b>				<b>Function related to: Port A</b>
PA3MOD	06.7~6	R/W	1	PA3~PA0 I/O mode control 00: Mode0 01: Mode1 10: Mode2 11: Mode3, used as analog I/O (ADC, OPA, DP/DM)
PA2MOD	06.5~4	R/W	1	
PA1MOD	06.3~2	R/W	1	
PA0MOD	06.1~0	R/W	1	
<b>(R07) PBMODH</b>				<b>Function related to: Port B</b>
PB4MOD	07.1~0	R/W	1	PB4 I/O mode control 00: Mode0 01: Mode1 10: Mode2 11: Mode3, used as ADC5

Name	Address	R/W	Rst	Description
<b>(R08) PBMODL</b>				<b>Function related to: Port B</b>
PB3MOD	08.7~6	R/W	1	PB3~PB0 I/O mode control 00: Mode0 01: Mode1 10: Mode2 11: Mode3, used as analog I/O (ADC, OPA)
PB2MOD	08.5~4	R/W	1	
PB1MOD	08.3~2	R/W	1	
PB0MOD	08.1~0	R/W	1	
<b>(R0B) MR0B</b>				<b>Function related to: INT0/INT1/T2/WKT/WDT</b>
INT1EDG	0b.7	R/W	0	INT1 (PA1) trigger edge select 0: INT1 (PA1) pin falling edge to trigger interrupt event 1: INT1 (PA1) pin rising edge to trigger interrupt event
INT0EDG	0b.6	R/W	0	INT0 (PA6) trigger edge select 0: INT0 (PA6) pin falling edge to trigger interrupt event 1: INT0 (PA6) pin rising edge to trigger interrupt event
T2PSC	0b.5~4	R/W	0	T2 prescaler. T2 clock source 00: divided by 32768 01: divided by 16384 10: divided by 8192 11: divided by 128
WDT PSC	0b.3~2	R/W	3	WDT period (@VCC=5V) 00: 120 ms 01: 240 ms 10: 960 ms 11: 1920 ms
WKT PSC	0b.1~0	R/W	3	WKT period (@VCC=5V) 00: 15 ms 01: 30 ms 10: 60 ms 11: 120 ms
<b>(R0C) MR0C</b>				<b>Function related to: PWM0/PWM1/PWM2/PWM3/Timer1</b>
PWM0PSC	0c.7~6	R/W	0	PWM0 prescaler. PWM0 clock source (Fpwm0) 00: divided by 1 01: divided by 2 10: divided by 4 11: divided by 64
PWM123PSC	0c.5~4	R/W	0	PWM1~PWM3 prescaler. PWM1~PWM3 clock source (Fpwm123) 00: divided by 1 01: divided by 2 10: divided by 4 11: divided by 8
TM1PSC	0c.3~0	R/W	0	Timer1 prescaler. Timer1 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

Name	Address	R/W	Rst	Description
<b>(R0D) PWRCTL</b>				<b>Function related to: EFT/Power Saving/LDO/System Voltage</b>
CLKFLT	0d.7	R/W	0	Fsys clock filter for noise defending 0: Higher Fsys 1: Lower Fsys
VCCFLT	0d.6	R/W	0	VCC filter, enhance the chip's power noise immunity 0: disable 1: enable
IVCSAV	0d.3	R/W	0	IVC auto power saving in STOP/IDLE mode 0: disable IVC save function 1: enable IVC save function
LDOPD	0d.2	R/W	0	Internal LDO (2.5V/1.25V) power down 0: LDO running 1: LDO power down
LDO25SEL	0d.1	R/W	1	Internal LDO voltage select 0: 1.25V 1: 2.5V
MODE3V	0d.0	R/W	0	3V mode selection 0: system operate in 5V mode (Vcc>3.6V) 1: system operate in 3V mode (Vcc<3.6V)
<b>(R0E) TM1RLD</b>				<b>Function related to: Timer1</b>
TM1RLD	0e.7~0	R/W	0	Timer1 reload data
<b>(R10) PWM0PRD</b>				<b>Function related to: PWM0</b>
PWM0PRD	10.7~0	R/W	FF	PWM0 period data
<b>(R11) PWM1D</b>				<b>Function related to: PWM1</b>
PWM1D	11.7~0	R/W	0	PWM1 duty
<b>(R12) PWM2D</b>				<b>Function related to: PWM2</b>
PWM2D	12.7~0	R/W	0	PWM2 duty
<b>(R13) PWM3D</b>				<b>Function related to: PWM3</b>
PWM3D	13.7~0	R/W	0	PWM3 duty
<b>(R14) PWM123PRD</b>				<b>Function related to: PWM1/PWM2/PWM3</b>
PWM123PRD	14.7~0	R/W	FF	PWM1, PWM2 and PWM3 shared period data

## INSTRUCTION SET

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

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

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

Field / Legend	Description
f	F-Plane Register File Address
r	R-Plane Register File Address
b	Bit address
k	Literal. Constant data or label
d	Destination selection field. 0 : Working register 1 : Register file
TO	WDT Time Out Flag
PD	Power Down Flag
W	Working Register
Z	Zero Flag
C	Carry Flag
DC	Decimal Carry Flag
PC	Program Counter
TOS	Top Of Stack
GIE	Global Interrupt Enable Flag (i-Flag)
[]	Option Field
()	Contents
.	Bit Field
B	Before
A	After
←	Assign direction



Mnemonic		Op Code	Cycle	Flag Affect	Description
<b>Byte-Oriented File Register Instruction</b>					
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
MOVRW	r	01 1111 00rr rrrr	1	-	Move "r" to W
MOVWF	f	00 0000 1fff ffff	1	-	Move W to "f"
MOVWR	r	01 1110 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"
<b>Bit-Oriented File Register Instruction</b>					
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
<b>Literal and Control Instruction</b>					
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		01 1110 0000 0100	1	TO, PD	Clear Watch Dog Timer
GOTO	k	11 kkkk kkkk kkkk	2	-	Jump to branch "k"
IORLW	k	01 1010 kkkk kkkk	1	Z	OR Literal "k" with W
MOVLW	k	01 1001 kkkk kkkk	1	-	Move Literal "k" to W
NOP		00 0000 0000 0000	1	-	No operation
RET		00 0000 0100 0000	2	-	Return from subroutine
RETI		00 0000 0110 0000	2	-	Return from interrupt
RETLW	k	01 1000 kkkk kkkk	2	-	Return with Literal in W
TABRL		00 0000 0101 0000	2	-	Lookup ROM low data to W
TABRH		00 0000 0101 1000	2	-	Lookup ROM high data to W
SLEEP		01 1110 0000 0011	1	TO, PD	Go into Power-down mode, Clock oscillation stops
XORLW	k	01 1101 kkkk kkkk	1	Z	XOR Literal "k" with W

<b>ADDLW</b>	<b>Add Literal "k" and W</b>	
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

<b>ADDWF</b>	<b>Add W and "f"</b>	
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

<b>ANDLW</b>	<b>Logical AND Literal "k" with W</b>	
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

<b>ANDWF</b>	<b>AND W with "f"</b>	
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

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

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

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

<b>CLRWD</b>	<b>Clear Watchdog Timer</b>
Syntax	CLRWD
Operands	-
Operation	WDT/WKT Timer ← 00h
Status Affected	TO, PD
OP-Code	01 1110 0000 0100
Description	CLRWD instruction clears the Watchdog/Wakeup Timer
Cycle	1
Example	CLRWD                                  B : WDT counter = ? A : WDT counter = 0x00

<b>COMF</b>	<b>Complement "f"</b>
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

<b>DECF</b>	<b>Decrement "f"</b>
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

<b>DECFSZ</b>	<b>Decrement "f", Skip if 0</b>
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

<b>GOTO</b>	<b>Unconditional Branch</b>
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

<b>INCF</b>	<b>Increment "f"</b>	
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

<b>INCFSZ</b>	<b>Increment "f", Skip if 0</b>	
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

<b>IORLW</b>	<b>Inclusive OR Literal with W</b>	
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

<b>IORWF</b>	<b>Inclusive OR W with "f"</b>	
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

---

**MOVRW                      Move "r" to W**


---

Syntax	MOVRW r	
Operands	r : 00h ~ 3Fh	
Operation	(W) ← (r)	
Status Affected	-	
OP-Code	01 1111 00rr rrrr	
Description	The contents of register 'r' are moved to W register.	
Cycle	1	
Example	MOVRW MR0B	B : MR0B = 0x0F, W = ? A : MR0B = 0x0F, W = 0x0F

---

**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	01 1110 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	<pre>CALL TABLE          B : W = 0x07                     :          A : W = value of k8 TABLE ADDWF PCL, 1       RETLW k1       RETLW k2       :       RETLW kn</pre>

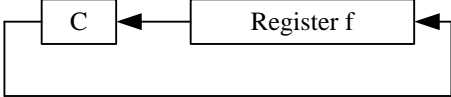
**TABRL**
**Return DPTR low byte to W**

Syntax	TABRL
Operands	-
Operation	(W) ← ROM[DPTR] low byte content, Where DPTR={DPH[max:8],DPL[7:0]}
Status Affected	-
OP-Code	00 0000 0101 0000
Description	The W register is loaded with low byte of ROM[DPTR]. This is a two-cycle instruction.
Cycle	2
Example	<pre>: : MOVLW  (TAB1&amp;0xFF) MOVWF  DPL          ; Where DPL is F-plane register MOVLW  (TAB1&gt;&gt;8)&amp;0xFF MOVWF  DPH          ; Where DPH is F-plane register  TABRL          ; W=0x89 TABRH         ; W=0x37  ORG  0234H TAB1: .DT   0x3789, 0x2277      ;ROM data 14 bits</pre>


**TABRH**
**Return DPTR high byte to W**

Syntax	TABRH
Operands	-
Operation	(W) ← ROM[DPTR] high byte content, Where DPTR={DPH[max:8],DPL[7:0]}
Status Affected	-
OP-Code	00 0000 0101 1000
Description	The W register is loaded with high byte of ROM[DPTR]. This is a two-cycle instruction.
Cycle	2

**RLF Rotate Left "f" through Carry**

Syntax	RLF f [,d]	
Operands	f : 00h ~ 7Fh, d : 0, 1	
Operation		
Status Affected	C	
OP-Code	00 1101 dfff ffff	
Description	The contents of register 'f' are rotated one bit to the left through the Carry Flag. If 'd' is 0, the result is placed in the W register. If 'd' is 1, the result is stored back in register 'f'.	
Cycle	1	
Example	RLF REG1, 0	B : REG1 = 1110 0110, C = 0 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	01 1110 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 1101 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

<b>XORWF</b>	<b>Exclusive OR W with 'f'</b>
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 $V_{CC}+0.3$	
Output voltage	$V_{SS}-0.3$ to $V_{CC}+0.3$	
Output current high per 1 PIN	-25	mA
Output current high per all PIN	-80	
Output current low per 1 PIN	+30	
Output current low per all PIN	+150	
Maximum operating voltage	5.5	V
Operating temperature	-40 to +85	°C
Storage temperature	-65 to +150	

### 2. DC Characteristics ( $T_A = 25^\circ\text{C}$ , $V_{DD} = 2.0\text{V}$ to $5.5\text{V}$ )

Parameter	Sym	Conditions	Min	Typ	Max	Unit	
Input High Voltage	$V_{IH}$	All Input, except PA7	$V_{CC} = 3\sim 5\text{V}$	$0.6V_{CC}$	–	$V_{CC}$	V
		PA7	$V_{CC} = 3\sim 5\text{V}$	$0.7V_{CC}$	–	$V_{CC}$	V
Input Low Voltage	$V_{IL}$	All Input, except PA7	$V_{CC} = 3\sim 5\text{V}$	$V_{SS}$	–	$0.2V_{CC}$	V
		PA7	$V_{CC} = 3\sim 5\text{V}$	$V_{SS}$	–	$0.2V_{CC}$	V
Output High Current	$I_{OH}$	All Output, except PA0, PB3	$V_{CC} = 5\text{V}, V_{OH}=4.5\text{V}$	4	9	–	mA
			$V_{CC} = 3\text{V}, V_{OH}=2.7\text{V}$	2	4	–	
		PA0, PB3	$V_{CC} = 5\text{V}, V_{OH}=4.5\text{V}$	15	30	–	
			$V_{CC} = 3\text{V}, V_{OH}=2.7\text{V}$	7	14	–	
Output Low Current	$I_{OL}$	All Output, except PA0, PB3	$V_{CC} = 5\text{V}, V_{OL}=0.5\text{V}$	7	15	–	mA
			$V_{CC} = 3\text{V}, V_{OL}=0.3\text{V}$	4	8	–	
		PA0, PB3	$V_{CC} = 5\text{V}, V_{OL}=0.5\text{V}$	32	65	–	
			$V_{CC} = 3\text{V}, V_{OL}=0.3\text{V}$	17	35	–	
Input Leakage Current (pin high)	$I_{ILH}$	All Input	$V_{IN} = V_{CC}$	–	–	1	uA
Input Leakage Current (pin low)	$I_{ILL}$	All Input	$V_{IN} = 0\text{V}$	–	–	-1	uA
Power Supply Current (No Load)	$I_{CC}$	FAST mode FIRC8 MHz, MODE3V = 0	$V_{CC} = 4.5$ to $5.5\text{V}$	–	1.6	–	mA
		FAST mode FIRC4 MHz MODE3V = 0	$V_{CC} = 4.5$ to $5.5\text{V}$	–	1.1	–	
		SLOW mode SIRC128 KHz LDOPD = 0 OPPD = 1 MODE3V = 1	$V_{CC} = 3.0\text{V}$	–	200	–	uA

Parameter	Sym	Conditions	Min	Typ	Max	Unit	
Power Supply Current (No Load)	I <sub>CC</sub>	SLOW mode SIRC128 KHz LDOPD = 1 OPPD = 1 MODE3V = 1	V <sub>CC</sub> = 3.0V	–	130	–	uA
		IDLE mode SIRC128 KHz MODE3V = 1 T2PSC = 0 LDOPD = 1 OPPD = 1	V <sub>CC</sub> = 3.0V LVR enable	–	5	–	uA
			V <sub>CC</sub> = 3.0V LVR disable in IDLE	–	3.8	–	
		STOP mode MODE3V = 1	V <sub>CC</sub> = 5.0V LVR disable in STOP	–	0.1	1	uA
			V <sub>CC</sub> = 5.0V, LVR enable	–	4	6	
STOP mode MODE3V = 1	V <sub>CC</sub> = 3.0V, LVR disable in STOP	–	0.1	1	uA		
System Operating Voltage	V <sub>SYS</sub>	MODE3V = 0	F <sub>sys</sub> = 2MHz	LVR <sub>th</sub>	–	5.5	V
			F <sub>sys</sub> = 4MHz	LVR <sub>th</sub>	–	5.5	
			F <sub>sys</sub> = 8MHz	LVR <sub>th</sub>	–	5.5	
		MODE3V = 1	F <sub>sys</sub> = 2MHz	LVR <sub>th</sub>	–	3.6	
			F <sub>sys</sub> = 4MHz	LVR <sub>th</sub>	–	3.6	
			F <sub>sys</sub> = 8MHz	LVR <sub>th</sub>	–	3.6	
Pull-up Resistor	R <sub>UP</sub>	VIN = 0 V Ports A/B	V <sub>CC</sub> = 5.0V	–	110	–	KΩ
			V <sub>CC</sub> = 3.0V	–	220	–	
		VIN = 0 V PA7	V <sub>CC</sub> = 5.0V	–	70	–	KΩ
			V <sub>CC</sub> = 3.0V	–	70	–	

### 3. Clock Timing (T<sub>A</sub> = -40°C to +85°C)

Parameter	Condition	Min	Typ	Max	Unit
FIRC Frequency	0°C ~ 85°C, V <sub>CC</sub> = 5.0 V	-2.5%	8	+2.5%	MHz
	25°C, V <sub>CC</sub> = 3.0 ~ 5.0 V	-3%	8	+3%	
	25°C, V <sub>CC</sub> = 2.5 ~ 5.0 V	-5%	8	+5%	
FIRC adjust step	IRCF increase or decrease by1	–	–	1%	

**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 $V_{CC} = 5\text{ V} \pm 10\%$	3	–	–	$\mu\text{s}$
WDT time	$V_{CC} = 3\text{ V}$ , WDTPSC = 11	-20%	2080	+20%	ms
	$V_{CC} = 5\text{ V}$ , WDTPSC = 11		1920		
WKT time	$V_{CC} = 3\text{ V}$ , WKTPSC = 11	-20%	128	+20%	ms
	$V_{CC} = 5\text{ V}$ , WKTPSC = 11		120		
CPU start up time	$V_{CC} = 3\text{ V}$	–	11	–	ms
	$V_{CC} = 5\text{ V}$	–	15	–	

**5. LVR Circuit Characteristics** ( $T_A = 25^{\circ}\text{C}$ )

Parameter	Symbol	Min	Typ	Max	Unit
LVR Reference Voltage	$LVR_{th}$	–	2.9	–	V
		–	2.3	–	
		–	2.0	–	
LVR Hysteresis Voltage	$V_{HYST}$	–	$\pm 0.1$	–	V
Low Voltage Detection time	$t_{LVR}$	100	–	–	$\mu\text{s}$

**6. ADC Electrical Characteristics** ( $T_A = 25^{\circ}\text{C}$ ,  $V_{CC} = 2.2\text{V}$  to  $5.5\text{V}$ ,  $V_{SS} = 0\text{V}$ )

Parameter	Conditions	Min	Typ	Max	Units
Total Accuracy	$V_{CC} = 5\text{V}$ , $V_{SS} = 0\text{V}$	–	$\pm 2.5$	$\pm 4$	LSB
Integral Non-Linearity		–	$\pm 3.2$	$\pm 5$	
Max Input Clock ( $f_{ADC}$ )	–	–	–	1	MHz
Conversion Time	$f_{ADC} = 1\text{ MHz}$	–	50	–	$\mu\text{s}$
Input Voltage	TM57MA17	$V_{SS}$	–	<b>0.95LDOC</b>	V
	TM57MA18	$V_{SS}$	–	<b>0.95VCC</b>	

**7. LDO Characteristics** ( $LDO25SEL = 0$ )

Parameter	Conditions	Min	Typ	Max	Units
LDOC pin	$T_A = 25^{\circ}\text{C}$ , $V_{CC} = 5.0\text{V}$	-1%	1.25	+1%	V
	$T_A = 25^{\circ}\text{C}$ , $V_{CC} = 2.0\sim 5.5\text{V}$	-2%	1.25	+2%	V
	$T_A = -40^{\circ}\text{C}$ to $+85^{\circ}\text{C}$ , $V_{CC} = 2.0\sim 5.5\text{V}$	-3%	1.25	+3%	V

**8. OPA Circuit Characteristics** ( $T_A = 25^\circ\text{C}$ ,  $V_{CC} = 5\text{V}$ ,  $V_{SS} = 0\text{V}$ ,  $C_{\text{Load}} = 100\text{ pF}$ ,  $R_{\text{Load}} = 1\text{ M}\Omega$ )

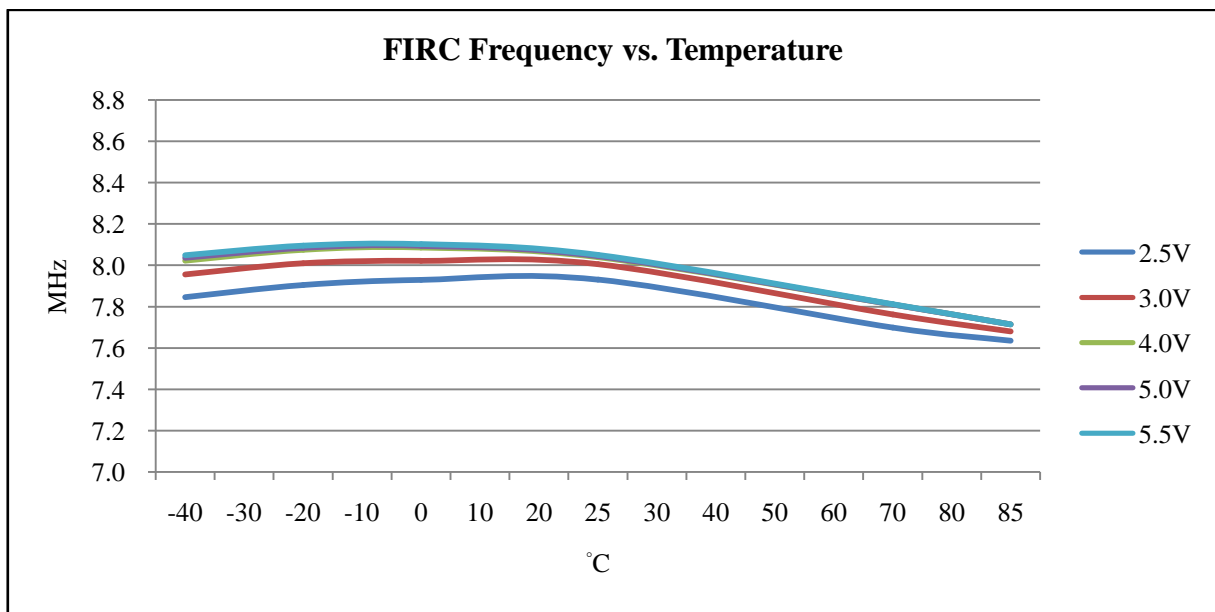
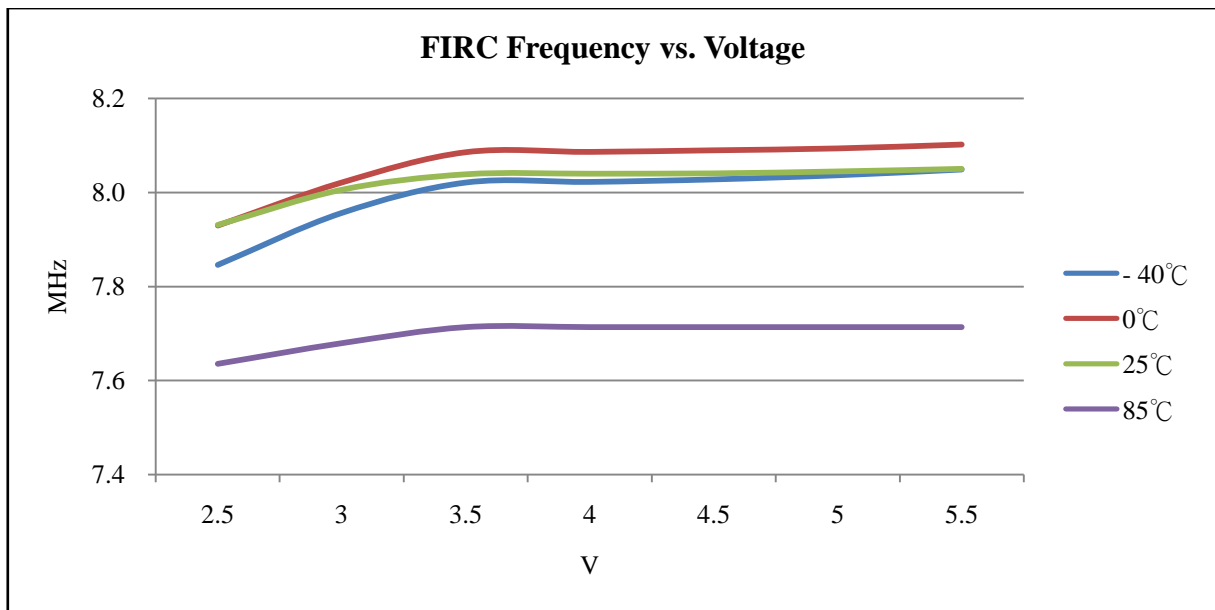
Symbol	Parameter	Test Conditions	Min	Typ	Max	Units
$V_{CC}$	Power supply		2.2	-	5.5	V
$V_{\text{icm}}$	Input Common voltage		0.1	-	$V_{CC}-1.2$	V
$V_{OS}$	Input Offset Voltage	$V_o = 2\text{V}$ , after trim	-	-	2	mV
$\Delta V_{OS} / \Delta T$	Temperature Coefficient of VOS	$V_o = 2\text{V}$	-	-	5	$\mu\text{V}/^\circ\text{C}$
$A_{VOL}$	Large Signal Voltage Gain	$R_L = 1\text{ M}\Omega$ $C_L = 100\text{ pF}$ $V_i = 0.1\text{ to }4\text{V}$ $V_o = 1\text{ to }4\text{V}$	-	120	-	dB
GBW	Gain Band Width Product	$R_L = 1\text{ M}\Omega$ $C_L = 100\text{ pF}$	-	2.1	-	MHz
CMRR	Common Mode Rejection Ratio	$V_o = 2\text{V}$	-	80	-	dB
PSRR	Power Supply Rejection Ratio	$V_o = 2\text{V}$	-	80	-	dB
$I_{CC}$	Supply Current Per Single Amplifier	$A_v = 1$ $V_o = 2\text{V}$ No load	-	300	-	$\mu\text{A}$
SR	Slew Rate at Unity Gain	No load	-	1.4	-	$\text{V}/\mu\text{s}$
$\Phi_m$	Phase Margin at Unity Gain	$R_L = 1\text{ M}\Omega$ $C_L = 100\text{ pF}$	-	60	-	Degree
$I_{OH}$	Output Source Current	$V_{i+} - V_{i-} \geq 10\text{mV}$	-	18	-	$\mu\text{A}$
$I_{OL}$	Output Sink Current	$V_{i-} - V_{i+} \geq 10\text{mV}$	-	20	-	mA

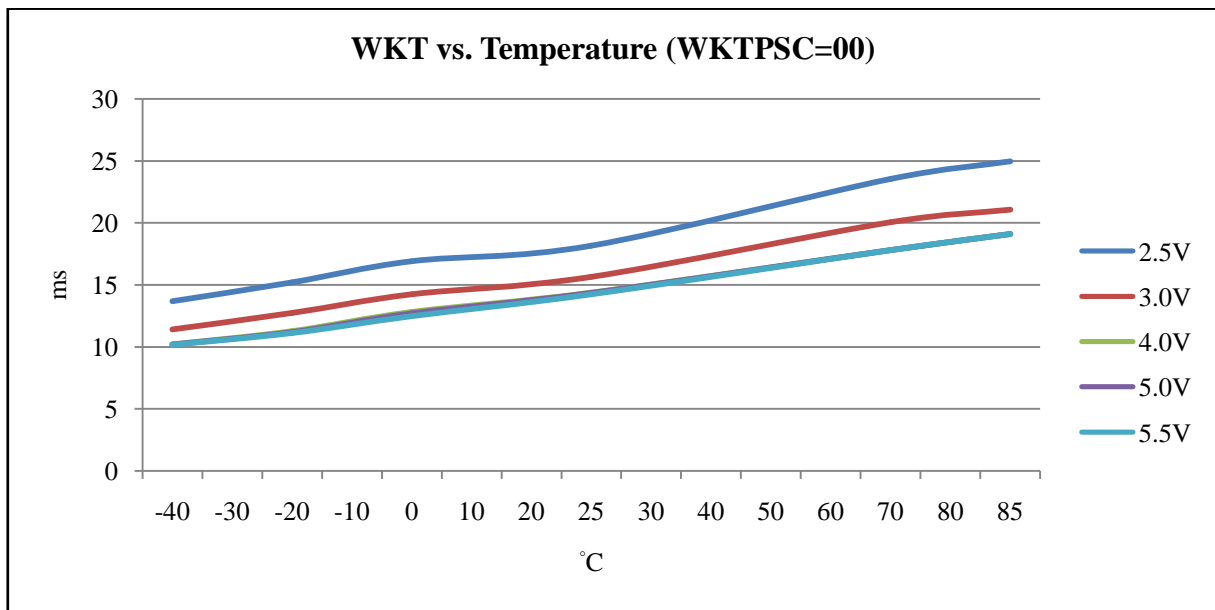
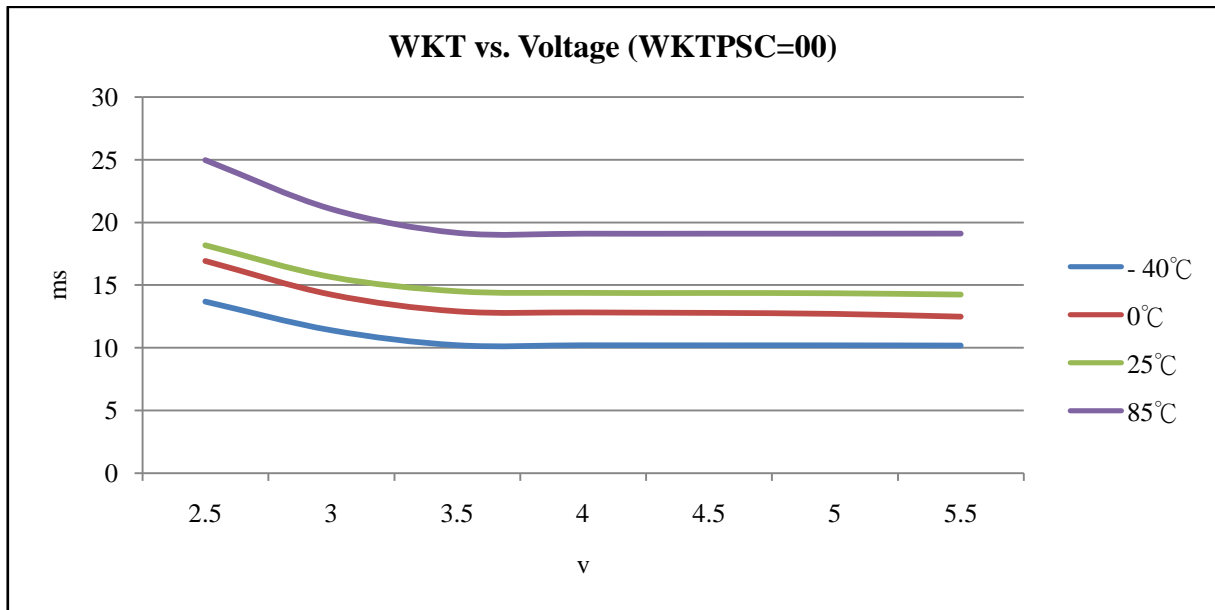
**9. DPDMV Circuit Characteristics** ( $T_A = 25^\circ\text{C}$ ,  $V_{CC} = 5\text{V}$ ,  $V_{SS} = 0\text{V}$ )

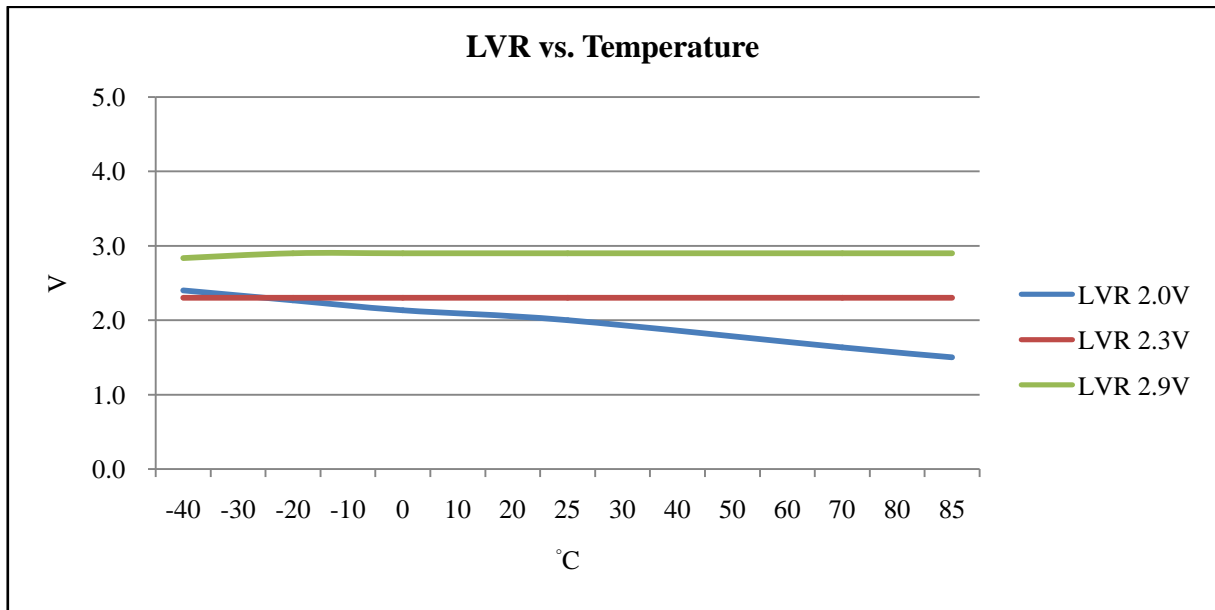
Parameter	Symbol	Test Conditions	Min	Typ	Max	Units
DP Current	$I_{DP}$		-	100	-	$\mu\text{A}$
DM Current	$I_{DM}$		-	100	-	$\mu\text{A}$
DP and DM short switch resistance	$R_{\text{Short}}$	$I_{\text{Short}} = 200\text{ }\mu\text{A}$	-	-	40	$\Omega$
DP Voltage variation	$\Delta V_{DP}$	$I_{DP} = 100\text{ }\mu\text{A}$	-2	-	-3	mV
DM Voltage variation	$\Delta V_{DM}$	$I_{DM} = 100\text{ }\mu\text{A}$	-2	-	-3	mV
DP and DM voltage level switch resistance	$R_{\text{SW}}$	$I_{\text{SW}} = 200\text{ }\mu\text{A}$	-	-	6	$\text{K}\Omega$



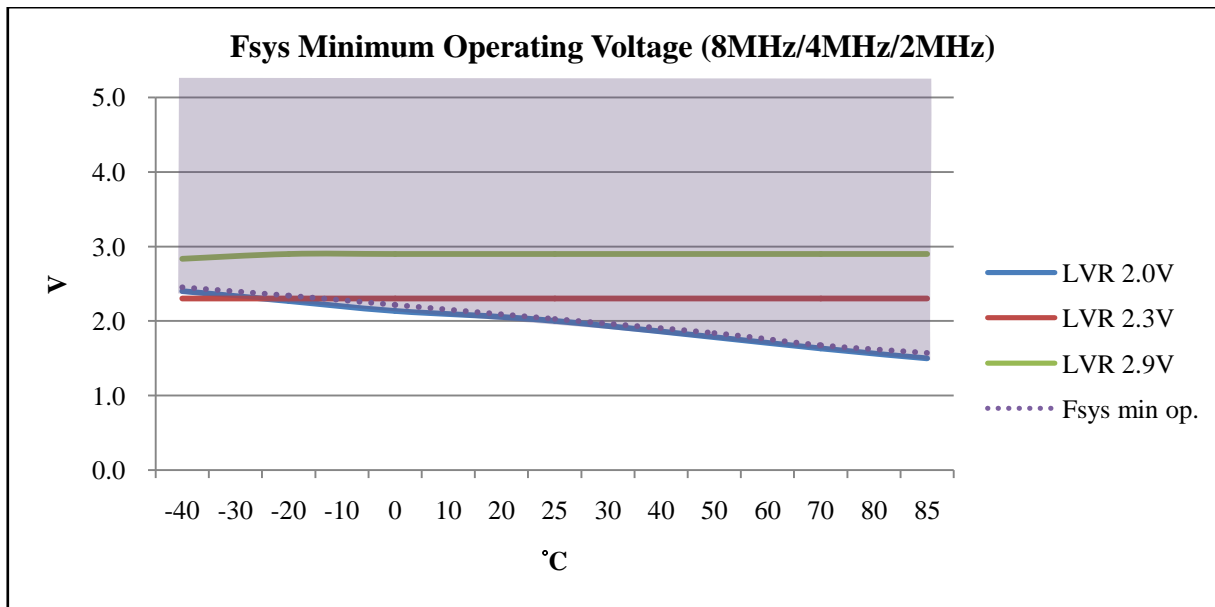
10. Characteristic Graphs



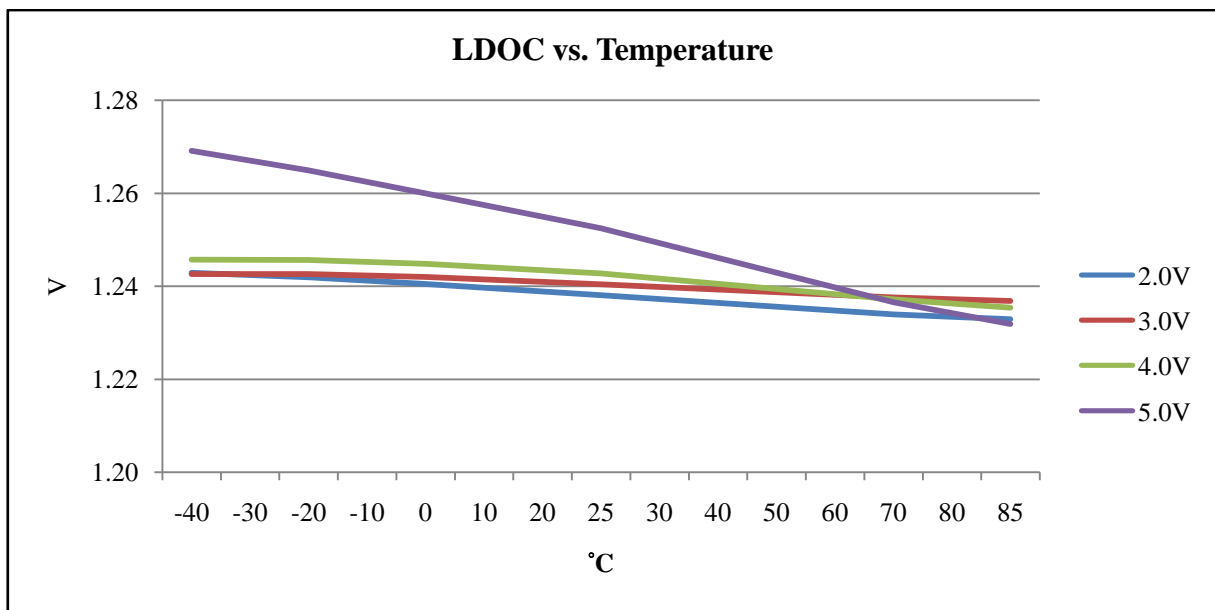
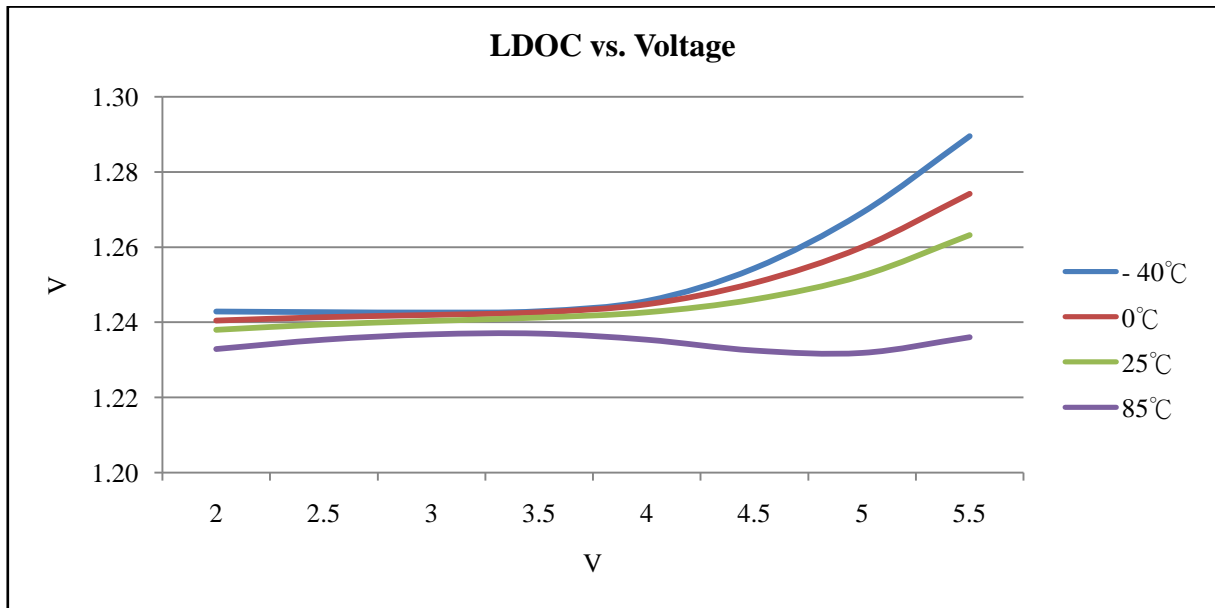


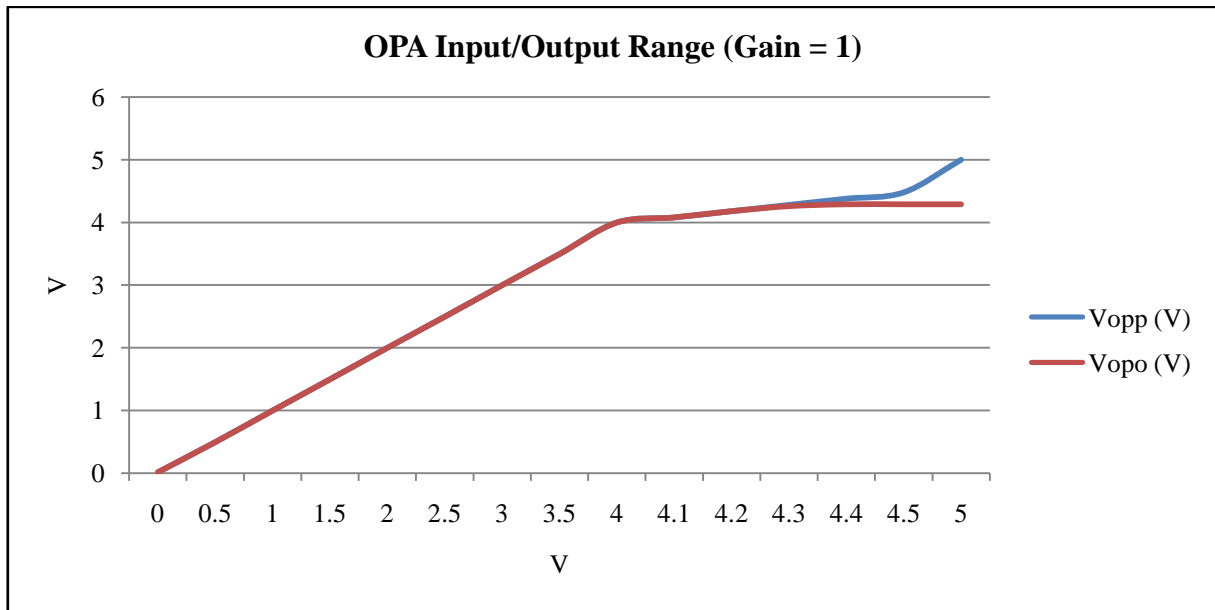


Note. Due to the variation of manufacturing process, this LVR will slightly vary between different chips.



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

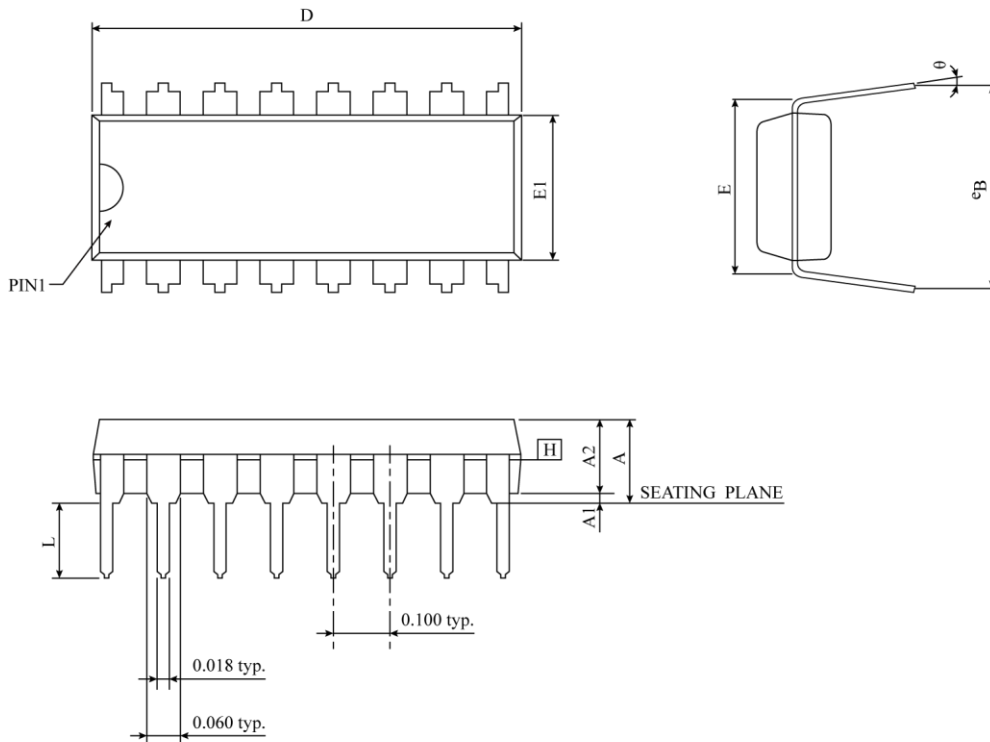




## PACKAGING INFORMATION

The ordering information:

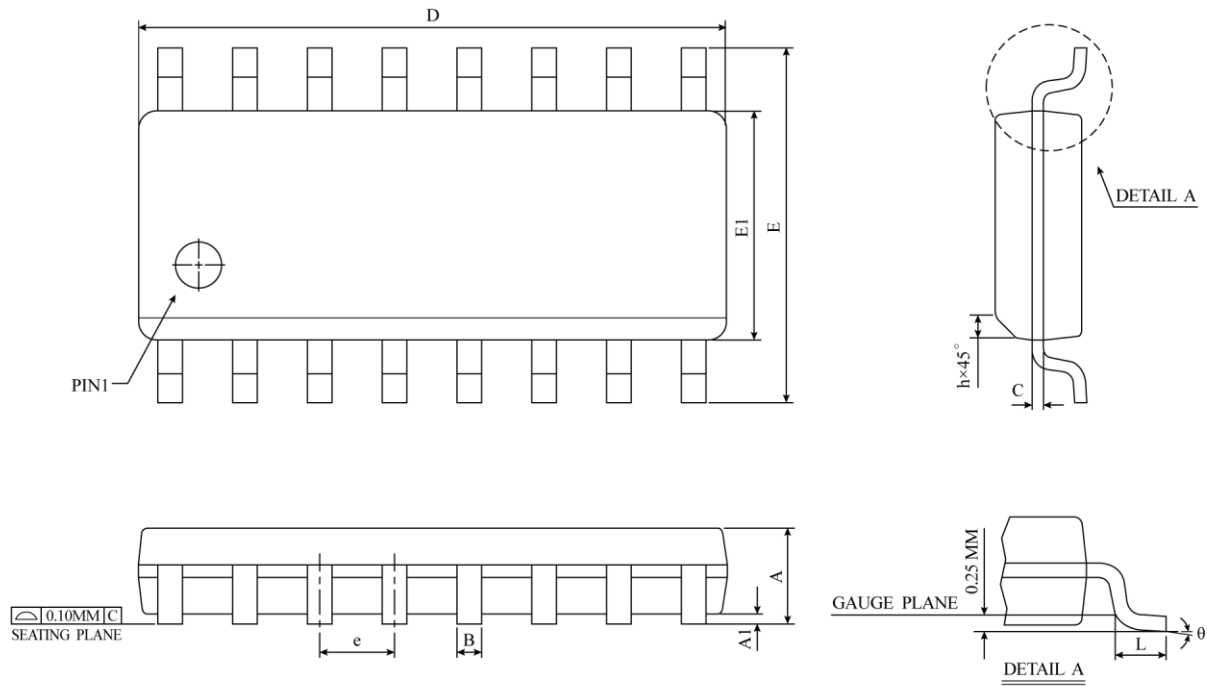
Ordering number	Package
TM57MA17-MTP	Wafer/Dice blank chip
TM57MA17-COD	Wafer/Dice with code
TM57MA17-MTP-03	DIP 16-pin (300 mil)
TM57MA17-MTP-16	SOP 16-pin (150 mil)
TM57MA18-MTP	Wafer/Dice blank chip
TM57MA18-COD	Wafer/Dice with code
TM57MA18-MTP-03	DIP 16-pin (300 mil)
TM57MA18-MTP-16	SOP 16-pin (150 mil)

**16-DIP Package Dimension**


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

**NOTES :**

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

**16-SOP Package Dimension**


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

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