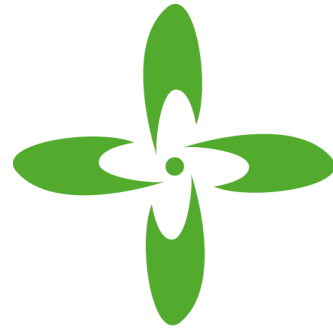




十速

TM57MA25

DATA SHEET

Rev 1.5

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

AMENDMENT HISTORY

Version	Date	Description
V1.4	Aug, 2016	New release. DS-TM57MA25_EV14 separated from DS-TM57MA21B_MA25_EV13
V1.5	Jan, 2017	1. Modify comparsion table (P8) 2. Modify typing error (P11 、 P54 、 P55…) 3. Modify/Add DC characteristics (P71 、 P74)

CONTENTS

AMENDMENT HISTORY	2
CONTENTS.....	3
FEATURES	5
BLOCK DIAGRAM	9
PIN ASSIGNMENT	10
PIN DESCRIPTIONS	11
PIN SUMMARYf	12
FUNCTIONAL DESCRIPTION	13
1. CPU Core	13
1.1 Clock Scheme and Instruction Cycle	13
1.2 Program ROM (PROM).....	14
1.3 Programming Counter (PC) and Stack.....	15
1.4 ALU and Working (W) Register.....	16
1.5 RAM Addressing Mode	17
1.6 STATUS Register (F-Plane 03H)	19
1.7 Interrupt.....	21
2. Chip Operation Mode	23
2.1 Reset (000H)	23
2.2 System Configuration Register (SYSCFG)	24
2.3 Dual System Clock.....	26
2.4 Dual System Clock Modes Transition	28
3. Peripheral Functional Block	31
3.1 Watchdog (WDT) /Wakeup (WKT) Timer.....	31
3.2 Timer0	33
3.3 Timer1	36
3.4 T2:15-bit Timer.....	38
3.5 PWM0: (8+2) bits PWM.....	40
3.6 PWM1: (8+2) bits PWM.....	43
3.7 Analog-to-Digital Converter	44
3.8 System Clock Oscillator.....	46
4. I/O Port.....	47
4.1 PA0-2	47
4.2 PA3-6, PB0-1, PD0-7.....	48
4.3 PA7.....	50
MEMORY MAP.....	51
F-Plane	51

R-Plane	54
INSTRUCTION SET	58
ELECTRICAL CHARACTERISTICS	70
1. Absolute Maximum Ratings	70
2. DC Characteristics	71
3. Clock Timing	72
4. Reset Timing Characteristics	72
5. LVR Circuit Characteristics	72
6. ADC Electrical Characteristics	72
7. Characteristic Graphs.....	73
PACKAGING INFORMATION	76
16-DIP (300 mil) Package Dimension	77
16-SOP (150 mil) Package Dimension	78
16-SSOP (150 mil) Package Dimension	79

FEATURES

1. **ROM: 2K x 14 bits MTP (Multi Time Programmable ROM)**
2. **RAM: 96 x 8bit**– Refer to the below comparsion table.(page 8)
3. **STACK: 6 Levels**
4. **System Oscillation Sources (Fsys)**
 - Fast-clock
 - FIRC (Fast Internal RC) : 1.5 / 4 / 6 /12 MHz
 - Slow-clock
 - SIRC (Slow Internal RC) : 140 KHz / 35 KHz / 8.75 KHz / 2.2 KHz @VCC=3V
5. **Dual System Clock**
 - FIRC + SIRC
6. **Power Saving Operation Mode**
 - FAST Mode: Slow-clock can be disabled or enabled, Fast-clock keeps CPU running
 - SLOW Mode: Fast-clock can be disabled or enabled, Slow-clock keeps CPU running
 - IDLE Mode: Fast-clock and CPU stop. Slow-clock, T2, or Wake-up Timer keep running
 - STOP Mode: All Clocks stop, T2 and Wake-up Timer stop
7. **3 Independent Timers**
 - Timer0
 - 8-bit timer divided by 1~256 pre-scaler option, Counter / Interrupt / Stop function
 - Timer1
 - 8-bit timer divided by 1~256 pre-scaler option, Reload / Interrupt / Stop function
 - Overflow and Toggle out
 - T2
 - 15-bit timer with 4 interrupt interval time options
 - IDLE mode wake-up timer or used as one simple 15-bit time base
 - Clock source: Slow-clock (SIRC) , Fsys/128
8. **Interrupt**
 - Three External Interrupt pins
 - 2 pins are falling edge wake-up triggered & interrupts
 - 1 pin is rising or falling edge wake-up triggered & interrupt
 - Timer0 / Timer1 / T2 / WKT (wake-up) Interrupts
9. **Wake-up (WKT) Timer**
 - Clocked by built-in RC oscillator with 4 adjustable interrupt times
15 ms / 30 ms / 60 ms / 120 ms @VCC=3V

10. Watchdog Timer

- Clocked by built-in RC oscillator with 4 adjustable reset times
 - 125 ms / 250 ms / 1000 ms / 2000 ms @VCC=5V
- Watchdog timer can be disabled/enabled in STOP mode (WDTSTP, R0E.5)

11. 2 Independent PWMs

- PWM0:
 - 8+2 bits, duty-adjustable, period-adjustable controlled PWM
 - PWM0 clock source: Fast-clock or Fast-clock-pre*, with 1~64 pre-scalers
- PWM1:
 - 8+2 bits, duty-adjustable controlled PWM
 - PWM1 clock source: Fast-clock or Fast-clock-pre* (up to 12 MHz)

Note: **Fast-clock-pre** is the original fast clock, Fast-clock is the divided clock of Fast-clock-pre.

12. 12-bit ADC Converter with 11 input channels.**13. Reset Sources**

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

14. Low Voltage Reset Option: LVR2.0V, LVR2.0 off in STOP mode, LVR2.3V, LVR2.9V**15. Enhanced Power Noise Rejection****16. Operating Voltage: Low Voltage Reset Level to 5.5V**

- Fsys=1 MHz, 2.0 ~5.5V
- Fsys=4 MHz, 2.0~5.5V
- Fsys=6 MHz, 2.2~5.5V
- Fsys=12 MHz, 2.6~5.5V

17. Operating Temperature Range: -40°C to +85°C**18. Table Read Instruction: 14-bit ROM data lookup table.****19. Instruction set: 38 Instructions****20. Instruction Execution Time**

- 2 oscillation clocks per instruction except branch

21. I/O ports: Maximum 18 programmable I/O pins

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

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

23. Package Types:

- 16-pin SSOP/SOP (150 mil), DIP (300mil); 8-pin SOP (150mil), DIP (300mil)

24. Supported EV board on ICE

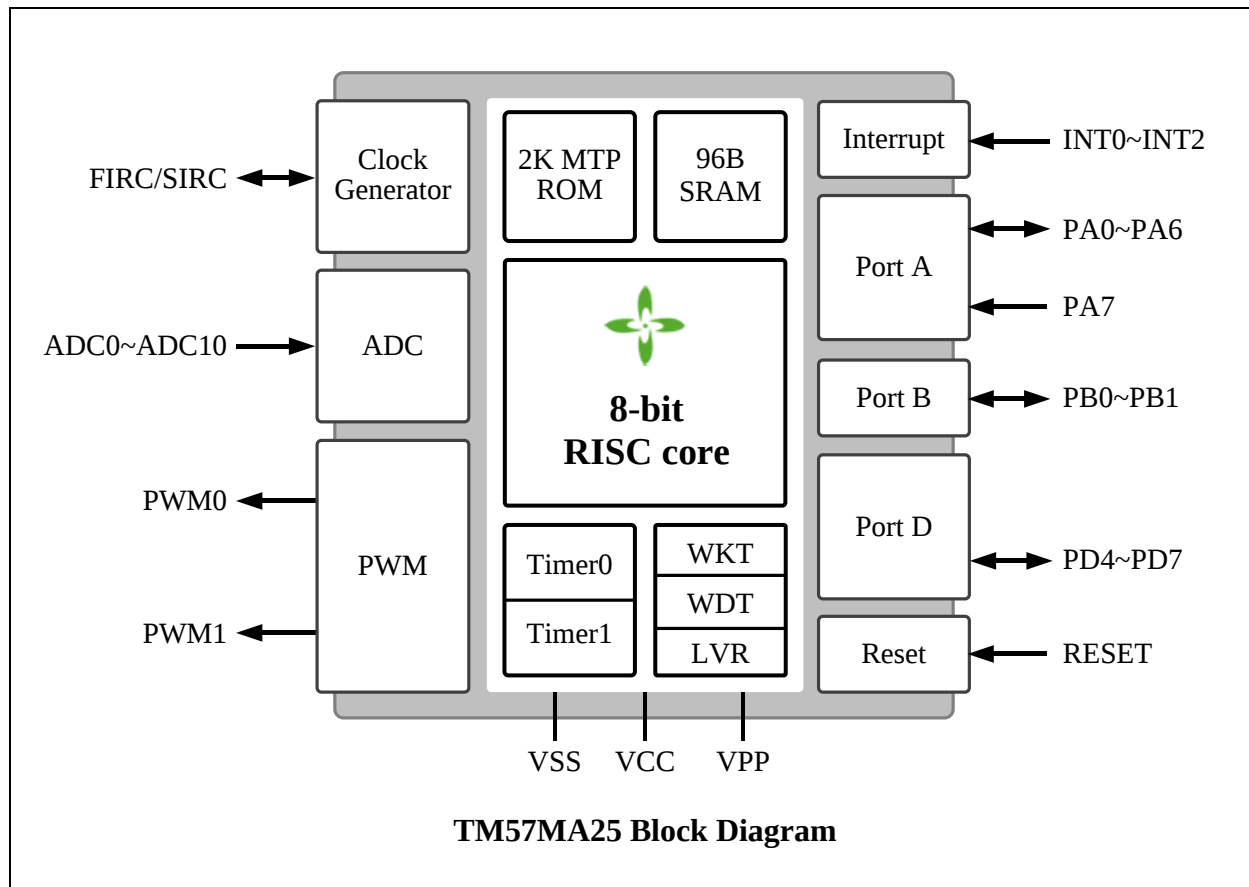
EV board: EV2781

TM57MA21B & TM57MA25 & TM57MA28 comparsion table:

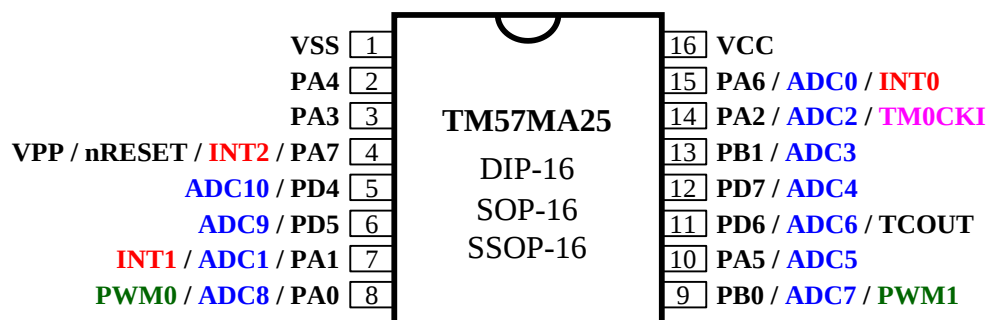
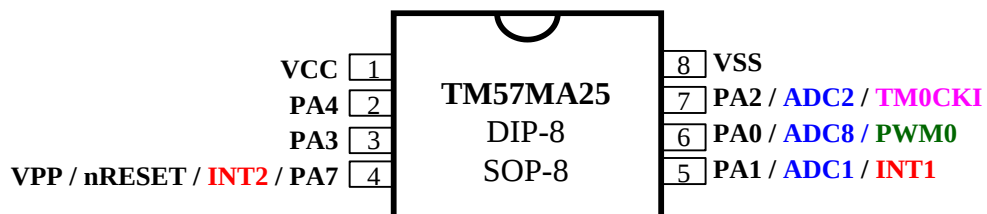
	EV2781	TM57MA21B	TM57MA25
EV board	-	EV2781	EV2781
SRAM	Common: 20~27 Bank0: 28~7F Bank1: 28~7F	Common: 20~27 Bank0: 28~7F Bank1: 28~7F	Common: 20~27 Bank0: 28~7F
Fast-clock	FXT / XRC / FIRC	FXT / XRC / FIRC	FIRC
Slow-clock	SXT / XRC / SIRC (140 KHz@5V)	SXT /XRC / SIRC (140 KHz@5V)	SIRC (140 KHz@5V)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

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

BLOCK DIAGRAM



PIN ASSIGNMENT



PIN DESCRIPTIONS

Name	In/Out	Pin Description
PA0–PA2	I/O	Bit-programmable I/O port for Schmitt-trigger input, CMOS “ push-pull ” output or “ pseudo-open-drain ” output. Pull-up resistors are assignable by software.
PA3–PA6 PB0–PB1 PD0–PD7	I/O	Bit-programmable I/O port for Schmitt-trigger input, CMOS “ push-pull ” output or “ open-drain ” output. Pull-up resistors are assignable by software.
PA7	I	Schmitt-trigger input with pull-high
nRESET	I	External active low reset, internal pull-high
VCC, VSS	P	Power Voltage input pin and ground
VPP	I	PROM programming high voltage input
INT0–INT2	I	External interrupt input
TM0CKI	I	Timer0’s input in counter mode
TM1OUT	O	(TM57MA25 don’t support)
TCOUT	O	Instruction cycle clock divided by N output. Where N is 1, 2, 4, 8. The instruction clock frequency is system clock frequency divided by two ($F_{sys}/2$).
PWM0/PWM1	O	(8+2) bit PWM output
ADC10~ADC0	I	A/D channels input

Programming pins:

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

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

PIN SUMMARYf

TM57MA25		Pin Name	Type	GPIO					Function After Reset	Alternate Function			
8- SOP/ DIP	16-SSOP/ SOP/ DIP			Input		Output				PWM		ADC	MISC
				Weak Pull-up	Ext. Interrupt	P.O.D	O.D	P.P					
1	1	VSS	P										
2	2	PA4	I/O	○			○	○	PA4				
3	3	PA3	I/O	○			○	○	PA3				
4	4	VPP/nRESET/INT2/PA7	I	○	○				SYS				nRESET
	5	ADC10/PD4	I/O	○			○	○	PD4			○	
	6	ADC9/PD5	I/O	○			○	○	PD5			○	
5	7	PA1/ADC1/INT1	I/O	○	○	○		○	PA1			○	
6	8	PA0/ADC8/PWM0	I/O	○		○		○	PA0	○		○	
	9	PB0/ADC7/PWM1	I/O	○			○	○	PB0	○		○	
	10	PA5/ADC5	I/O	○			○	○	PA5			○	
	11	PD6/ADC6/TCOUT	I/O	○			○	○	PD6			○	TCOUT
	12	PD7/ADC4	I/O	○			○	○	PD7			○	
	13	PB1/ADC3	I/O	○			○	○	PB1			○	
7	14	PA2/ADC2/TM0CKI	I/O	○		○		○	PA2			○	TM0CKI
	15	PA6/ADC0/INT0	I/O	○	○		○	○	PA6			○	
8	16	VCC	P										

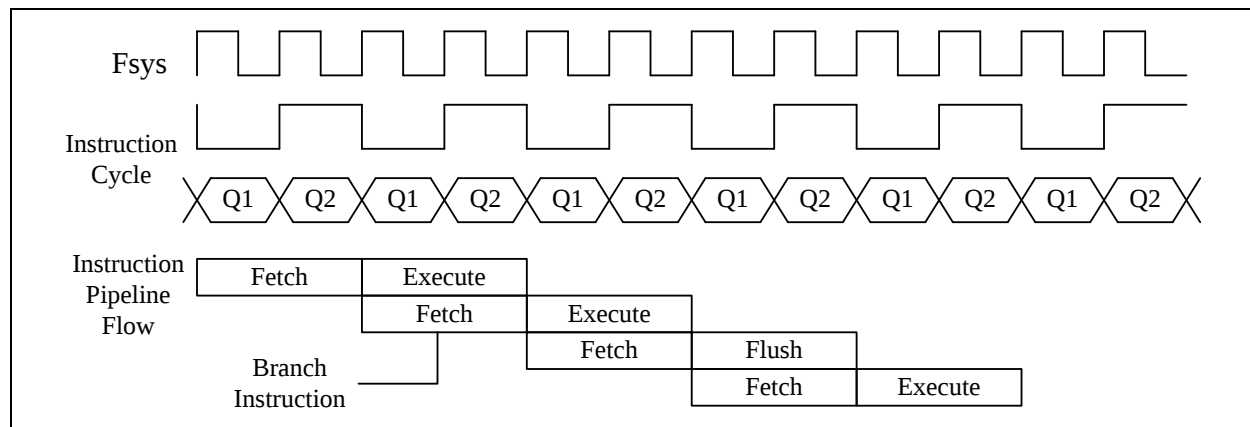
Symbol : P.P. = Push-Pull Output
 P.O.D. = Pseudo Open Drain
 O.D. = Open Drain
 SYS = by SYSCFG bit

FUNCTIONAL DESCRIPTION

1. CPU Core

1.1 Clock Scheme and Instruction Cycle

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



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

Terminology definitions:

- (1) **Fast-clock-pre:** The clock source that contains one fast frequency oscillation: Fast Internal RC oscillator (FIRC).
- (2) **Fast-clock:** The words means the all set of fast clocks, its clock frequency can be Fast-clock-pre divided by 1, 2, 3, 8.
- (3) **Slow-clock:** The clock source that contains Slow Internal RC oscillator (SIRC).
- (4) **Fsys:** System clock. The main clock that drive the core logic and most peripherals. The clock source can be either Fast-clock or Slow-clock which can be set by registers.
- (5) **Instruction Cycle**=Fsys/2

FXT: Fast Crystal

SXT: Slow Crystal (32 KHz)

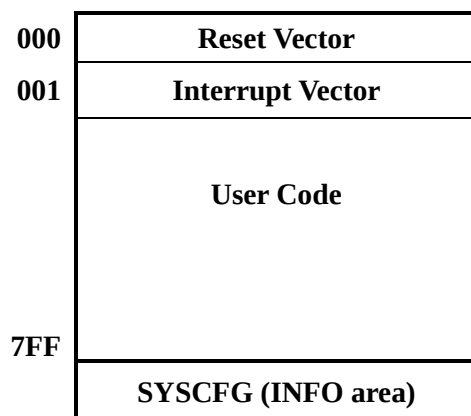
FIRC: Fast Internal RC oscillator

XRC: Fast or Slow External RC oscillator

SIRC: Slow Internal RC oscillator

1.2 Program ROM (PROM)

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



1.3 Programming Counter (PC) and Stack

The Programming Counter is 11-bit wide capable of addressing a 2Kx14 MTP ROM. As a program instruction is executed, the PC will contain the address of the next program instruction to be executed. The PC value is normally increased by one except the followings. The Reset Vector (000h) and the Interrupt Vector (001h) are provided for PC initialization and Interrupt. For CALL/GOTO instructions, PC loads 11 bits address from instruction word. For RET/RETI/RETLW instructions, PC retrieves its content from the top level STACK. For the other instructions updating PC [7:0], the PC [10:8] keeps unchanged. The STACK is 11-bit wide and 6-level in depth. The CALL instruction and hardware interrupt will push STACK level in order. While the RET/RETI/RETLW instruction pops the STACK level in order.

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

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

```

        ORG      000H                      ; Reset Vector
        GOTO     START

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

LOOP:
        MOVFW    INDEX                    ; Move index value to W register.
        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 (F0F.2~0)
        MOVLW    (TABLE2) &0xff
        MOVWF    DPL                       ; DPL register (F13.7~0)
        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 768H

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

```

1.4 ALU and Working (W) Register

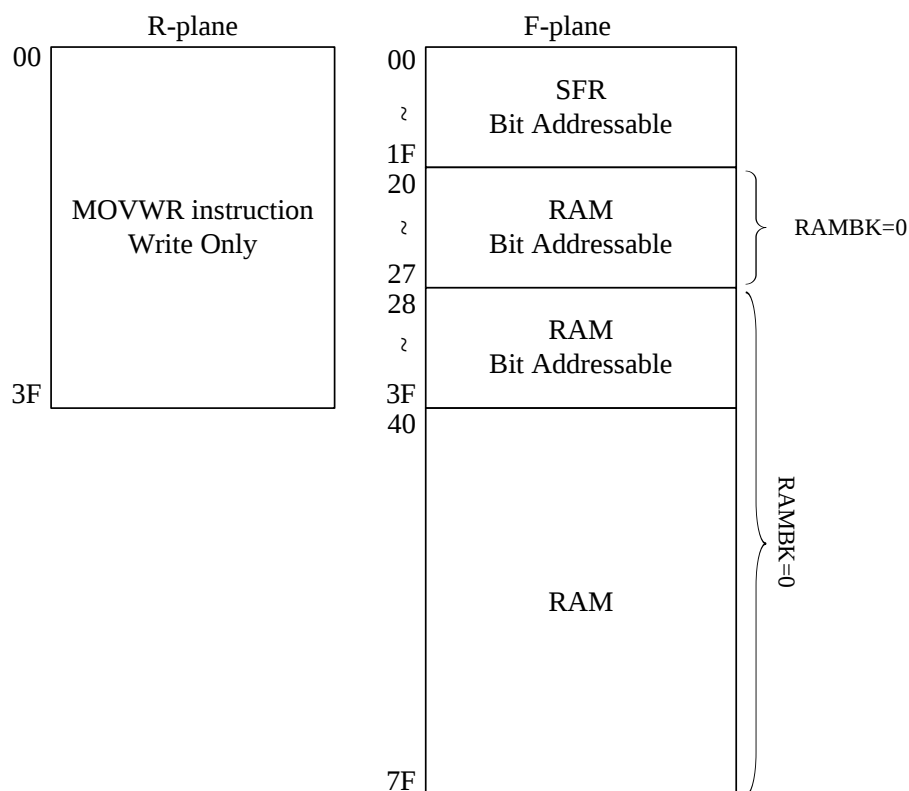
The ALU is 8-bit wide and capable of addition, subtraction, shift and logical operations. In two-operand instructions, typically one operand is the W register, which is an 8-bit non-addressable register used for ALU operations. The other operand is either a file register or an immediate constant. In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

Note: /Borrow represents inverted of Borrow register.

/Digit Borrow represents inverted of Digit Borrow register.

1.5 RAM Addressing Mode

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



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

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

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

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

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

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

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

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

1.6 STATUS Register (F-Plane 03H)

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

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

◇Example: Write immediate data into STATUS register.

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

◇Example: Bit addressing set and clear STATUS register.

```
BSF      STATUS, 0    ; Set C=1.
BSF      03H, 5       ; Selection RAM Bank1
BCF      STATUS, 0    ; Clear C=0.
BCF      03H, 5       ; Selection RAM Bank0
```

◇Example: Determine the C flag by BTFSS instruction.

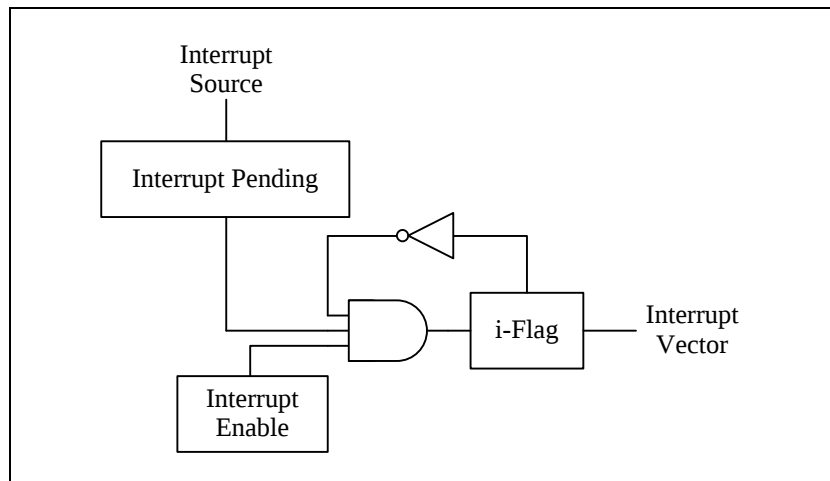
```
BTFSS    STATUS, 0    ; Check the carry flag
GOTO     LABEL_1      ; If C=0, goto label_1
GOTO     LABEL_2      ; If C=1, goto label_2
```

1.7 Interrupt

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

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

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



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

```

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

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

    ORG      02H

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

MAIN:
    ...
    GOTO     MAIN

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

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

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

```

2. Chip Operation Mode

2.1 Reset (000H)

This device can be RESET in four ways.

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

After Power-On-Reset, all system and peripheral control registers are then set to their default hardware Reset values. The clock source, LVR level and chip operation mode are selected by the SYSCFG register value.

The Low Voltage Reset features static reset when supply voltage is below a threshold level. There are three threshold levels can be selected. The LVR's operation mode is defined by the SYSCFG register. See the following LVR Selection Table; user must also consider the lowest operating voltage of operating frequency.

LVR Selection Table:

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

Different Fsys have different system minimum operating voltage, reference to Operating Voltage of DC characteristics, if current system voltage is low than minimum operating voltage and lower LVR is selected, then the system maybe enter dead-band and error occur.

The External Pin Reset and Watchdog Reset can be disabled or enabled by the SYSCFG register. These two resets also set all the control registers to their default reset value. The TO/PD flags are not affected by these resets.

◇Example: Defining Reset Vector

```

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

ORG      010H

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

```

2.2 System Configuration Register (SYSCFG)

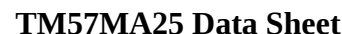
The System Configuration Register (SYSCFG) is located at ROM address 3FCh. The SYSCFG determines the option for initial condition of MCU. It is written by PROM Writer only. User can select clock source, LVR threshold voltage and chip operation mode by SYSCFG register. The default value of SYSCFG is 3FFFh. The 13th bit of SYSCFG is code protection selection bit. If this bit is 0, the data in PROM will be protected, when user reads PROM.

Bit	13~0	
Default Value	111111111111	
Bit	Description	
13	PROTECT : Code protection selection	
	1	Enable
	0	Disable
12	PWRSAPV : Power Saving Mode (affect power consumption for STOP/IDLE mode)	
	1	Enable (IVC* off)
	0	Disable
11-10	LVR : Low Voltage Reset Mode	
	11	2.0V, Always Enable
	10	2.0V, Disable in STOP Mode (if MODE3V =1, force LVR=2'b10)
	01	2.3V, Always Enable
	00	2.9V, Always Enable
9-8	INT0EDGE : INT0(PA6) trigger edge (Notice: EV2781 only support falling edge emulation)	
	x1	Rising edge
	x0	Falling edge
7	XRSTE : External Pin (PA7) Reset Enable	
	1	Enable
	0	Disable (PA7 as input I/O pin)
6	WDTE : WDT Reset Enable	
	1	Enable
	0	Disable
5	MODE3V : Operating Voltage Selection	
	1	Operating Voltage $\leq$ 3.6V, default Fsys=Slow-clock (SIRC 2KHz)
	0	Operating Voltage $>$ 3.6V, default Fsys=Fast-clock (FIRC 4MHz)
4-0	Tenx Reserved	

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

- SYSCFG setting & power consumption relationship:

MODE3V	PWRSAP	LVR[1:0]	FAST/SLOW mode	STOP/IDLE mode
1	x	xx	LVR2.0V on, IVC off	LVR2.0V off, IVC off
0	1	00	LVR2.9V on, IVC on	LVR2.0V on, IVC off
0	1	01	LVR2.3V on, IVC on	LVR2.0V on, IVC off
0	1	10	LVR2.0V on, IVC on	LVR2.0V off, IVC off
0	1	11	LVR2.0V on, IVC on	LVR2.0V on, IVC off
0	0	00	LVR2.9V on, IVC on	LVR2.9V on, IVC on (+160 uA at VCC=5V)
0	0	01	LVR2.3V on, IVC on	LVR2.3V on, IVC on (+160 uA at VCC=5V)



The device is designed with dual-clock system. There are five kinds of clock source, i.e. FXT (Fast Crystal) Clock, SXT (Slow Crystal) Clock, XRC (External RC) Clock, 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:

After power on or reset, if MODE3V=0, device enters FAST mode, otherwise enters SLOW mode. In FAST mode, device can select FXT, XRC or FIRC as its CPU clock by SYSCFG[9:8] setting. Besides, firmware can also enable or disable the Slow-clock for the T2 system operating.

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

SLOW Mode:

After power-on or reset, if MODE3V=1, device enters SLOW mode, the default Slow-clock is SIRC & Clock=2 KHz. If user want to configure another Slow-clock type (SXT, XRC or SIRC) , must first enter FAST mode to change SLOWCKS[1:0] , then back to SLOW mode. In this mode, the Fast-clock can stopped (by FASTSTP=1, for power saving) or running (by FASTSTP=0) , and Slow-clock is enabled. All peripheral blocks (Timer0, Timer1etc...) clock sources are Slow-clock in the SLOW mode.

IDLE Mode:

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

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

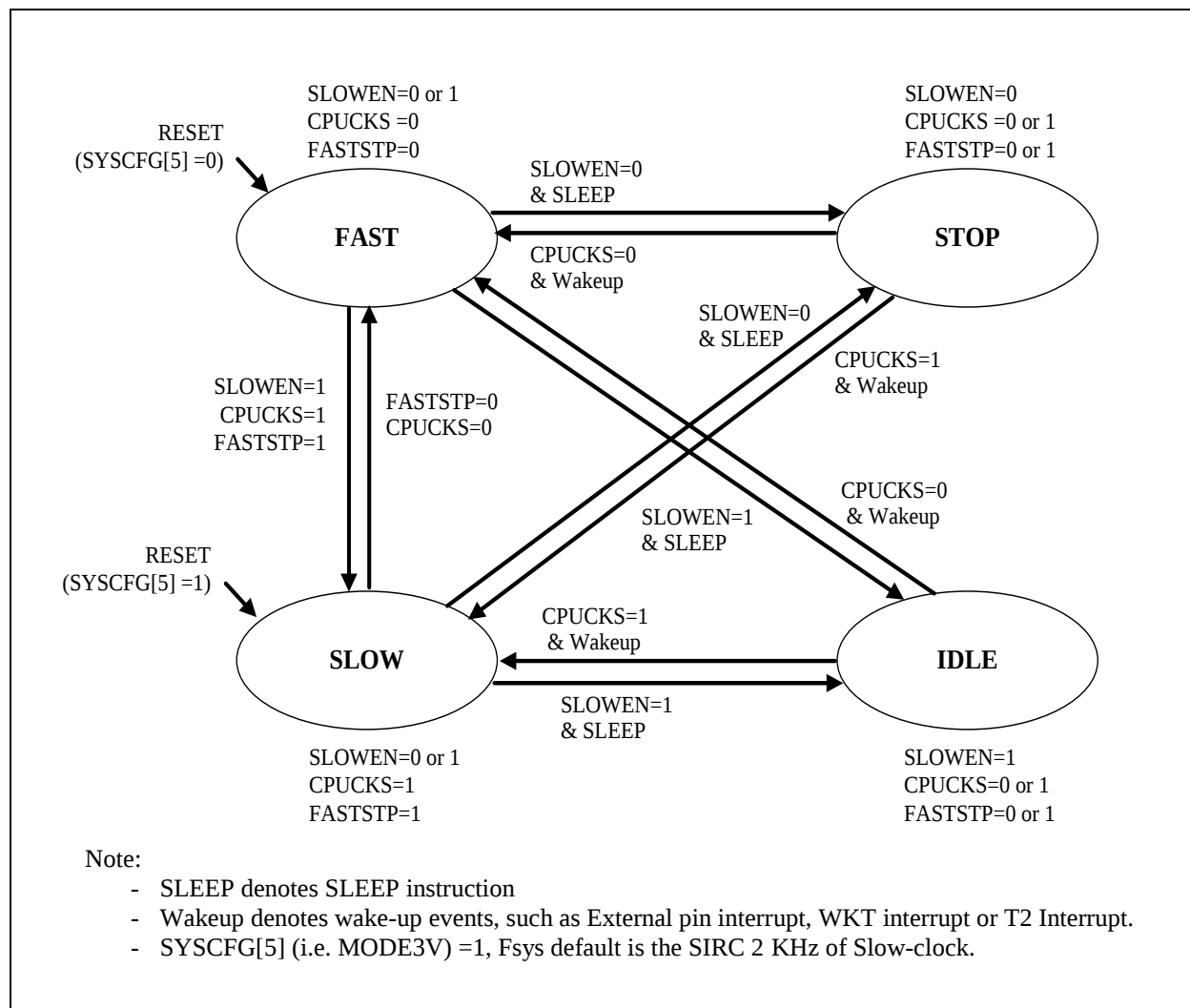
T2 and WKT/WDT are independent and have their own control registers. It is possible to keep both T2 and WKT working and wake-up in the IDLE mode, which is useful for low power mode Touch Key detection.

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.

2.4 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	T2	Wakeup event
FAST	FIRC, FXT SXT, XRC	Fast-clock	Run	Set by SLOWEN bit	Run	Run	X
SLOW	SXT, XRC SIRC	Slow-clock	Set by FASTSTP bit	Run	Run	Run	X
IDLE	SXT, XRC SIRC	Stop	Stop	Run	Stop	Run	WKT/IO/T2
STOP	Stop	Stop	Stop	Stop	Stop	Stop	IO

● PA3/PA4 IO setting notes in STOP/IDLE mode

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

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

※: Don't care

● FAST mode switches to SLOW mode

FAST mode can be chosen by SYSCFG [9:8] when equals to 11(FXT) , 00(XRC) , or 01(FIRC) . The following steps are suggested to be executed by order when FAST mode switches to SLOW mode:

- (1) Enable Slow-clock (SLOWEN=1)
- (2) Switch to Slow-clock (CPUCKS=1)
- (3) Stop Fast-clock (FASTSTP=1)

◇Example: Switch FAST mode to SLOW mode.

```

MOVLW    01100001B
MOVWF    F0B          ; Slow-clock type=SIRC & 2 KHz.
BSF      SLOWEN       ; Enable Slow-clock.
NOP
BSF      CPUCKS       ; Fsys=Slow-clock.
BSF      FASTSTP      ; Disable Fast-clock.

```

● SLOW mode switches to FAST mode

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

- (1) Enable Fast-clock (FASTSTP=0)
- (2) Switch to Fast-clock (CPUCKS=0)
- (3) Stop Slow-clock (SLOWEN=0)

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

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

```
MOVLW    00001000B
MOVWR    R15                ; Fast-clock=Fast-clock-pre/2
BCF      FASTSTP            ; Enable Fast-clock.
NOP
BCF      CPUCKS             ; Fsys=Fast-clock
BCF      SLOWEN             ; Disable Slow-clock
```

● IDLE mode Setting

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

- (1) Enable Slow-clock (SLOWEN=1) or WKT(WKTIE=1)
- (2) Switch T2 clock source to Slow-clock (T2CKS=0)
- (3) Execute SLEEP instruction

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

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

```
BSF      SLOWEN             ; Enable Slow-clock.
MOVLW    00000001B
MOVWR    R15                ; T2 Clock source=Slow-clock. TM2PSC=div 16384
SLEEP    ; Enter IDLE mode.
```

● STOP Mode Setting

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

- (1) Stop Slow-clock (SLOWEN=0)
- (2) Stop WKT/WDT (WKTIE=0, WDTSTP=1)
- (3) Execute SLEEP instruction

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

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

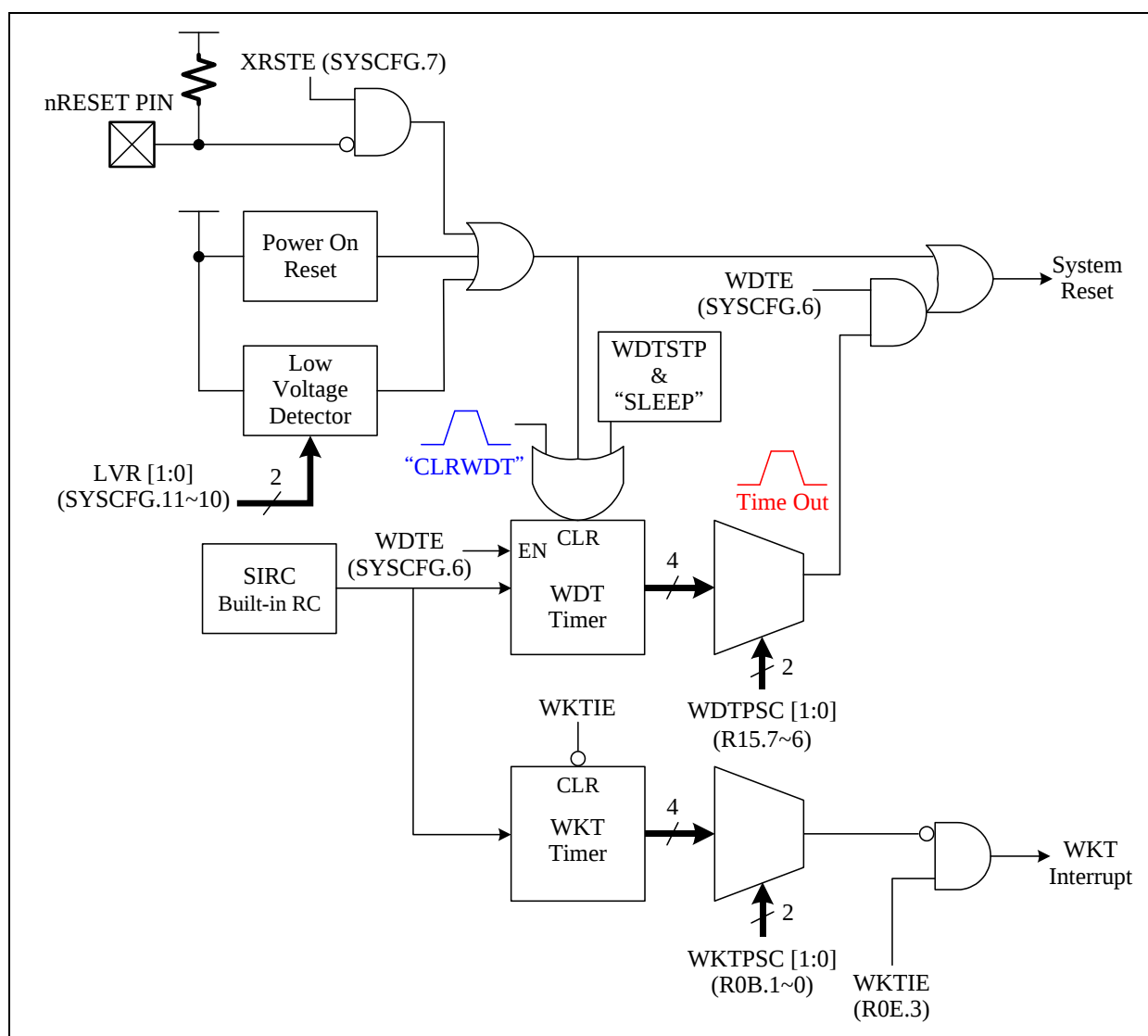
```
BCF      SLOWEN             ; Disable Slow-clock.
MOVLW    00000000B          ; Disable WKT counting
MOVWR    INTIE
MOVLW    01100100B         ; Stop WDT counting in STOP mode
MOVWR    R15
SLEEP    ; Enter STOP mode.
```

3. Peripheral Functional Block

3.1 Watchdog (WDT) /Wakeup (WKT) Timer

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

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



WDT/WKT Block Diagram

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

◇Example: Clear watchdog timer by CLRWDT instruction.

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

◇Example: Clear watchdog timer by write WDTCLR register.

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

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

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

SLEEP
```

◇Example: Set WKT period and interrupt function.

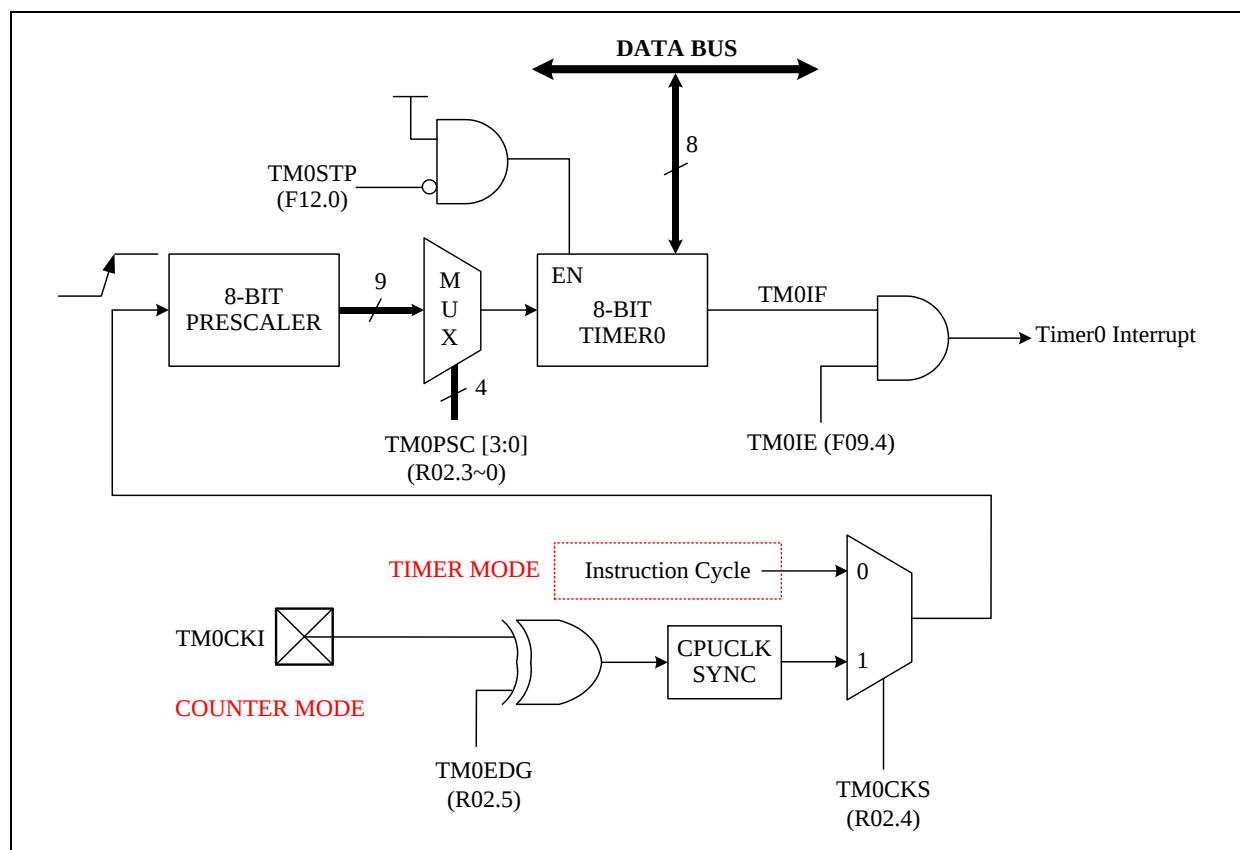
```
MOVLW    000000010B
MOVWR    R0B        ; Select WKT period=60 ms @3V (default 120 ms).

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

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

3.2 Timer0

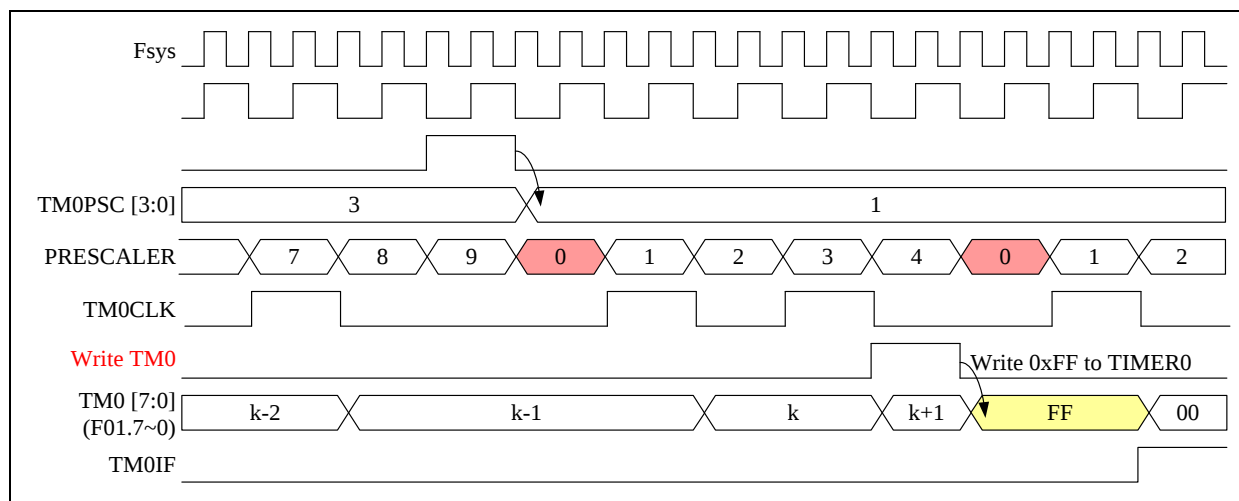
The Timer0 is an 8-bit wide register of F-Plane 01h (TM0). It can be read or written as any other register of F-Plane. Besides, Timer0 increases itself periodically and automatically rolls over based on the pre-scaled clock source, which can be the instruction cycle or TM0CKI (PA2) rising / falling input.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



Timer0 Block Diagram

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

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



Timer0 works in Timer mode (TM0CKS=0)

The equation of TM0 interrupt time value is as following:

$$\text{TM0 interrupt interval cycle time} = \text{Instruction cycle time} / \text{TM0PSC} / (256 - \text{TM0})$$

◇Example: Setup TM0 work in Timer mode

; Setup TM0 clock source and divider

```
MOVLW    00000101B
MOVWR    R02
```

; R02.4=0, Setup TM0 clock=Instruction cycle
; R02.3~0=5 (TM0PSC)
; TM0 clock prescaler = Instruction cycle divided by 32

; Set TM0 timer.

```
BSF      TM0STP
MOVLW    156
MOVWF    TM0
```

; Disable TM0 counting (Default " 0 ").
; Write 156 into TM0 register of F-Plane. (F01)

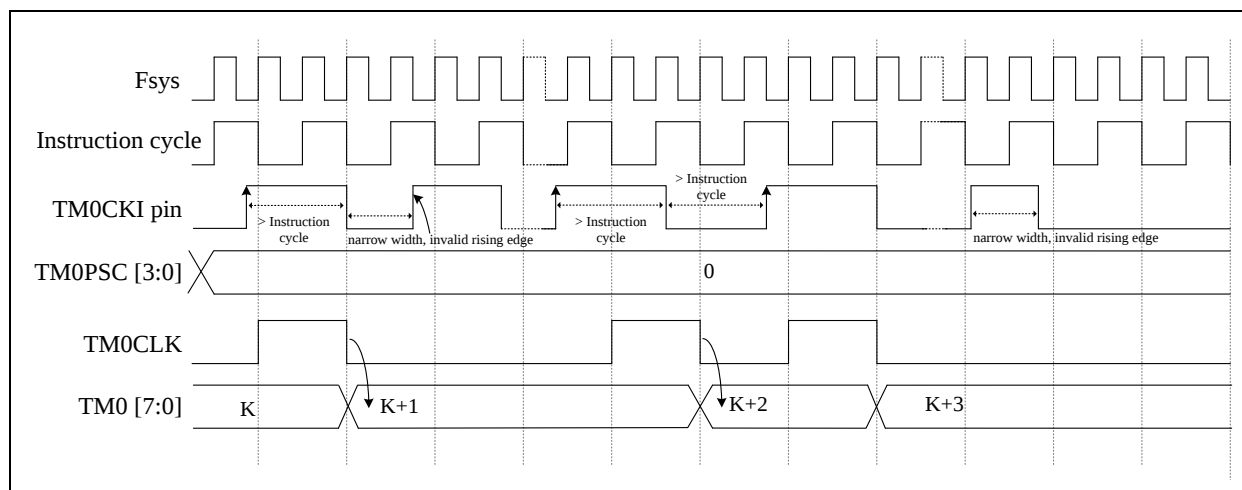
; Enable TM0 timer and interrupt function.

```
MOVLW    11101111B
MOVWF    INTIF
MOVLW    00010000B
MOVWR    INTIE
BCF      TM0STP
```

; Clear TM0 request interrupt flag by byte operation
; F-Plane 09H
; Enable TM0 interrupt function
; R0E
; Enable TM0 counting (Default " 0 ").

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

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



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

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

; Setup TM0 clock source from TM0CKI pin (PA2) and divider.

```
MOVLW    00110000B
```

```
MOVWR    R02
```

; R02.5=1, Select TM0 prescaler counting edge=falling edge.

; R02.4=1, Setup TM0 clock=TM0CKI pin (PA2)

; R02.3~0=0 (TM0PSC)

; TM0 clock prescale r=Instruction cycle divided by 1

; Set TM0 timer and stop TM0 counting.

```
BSF      TM0STP
```

; Disable TM0 counting (Default " 0 ").

```
MOVLW    00H
```

```
MOVWF    TM0
```

; Write 0 into TM0 register of F-Plane.

; Start TM0 count and read TM0 counter.

```
BCF      TM0STP
```

; Enable TM0 counting.

```
NOP
```

```
NOP
```

```
NOP
```

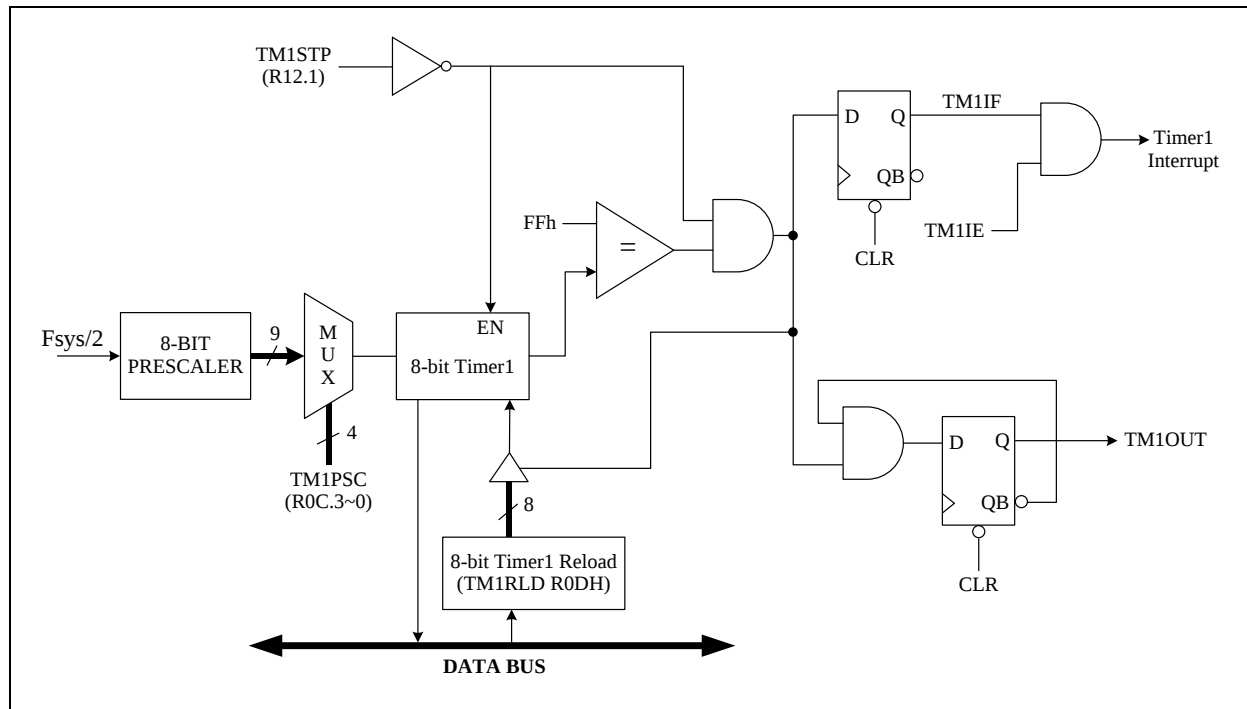
```
BSF      TM0STP
```

; Disable TM0 counting (Default " 0 ")

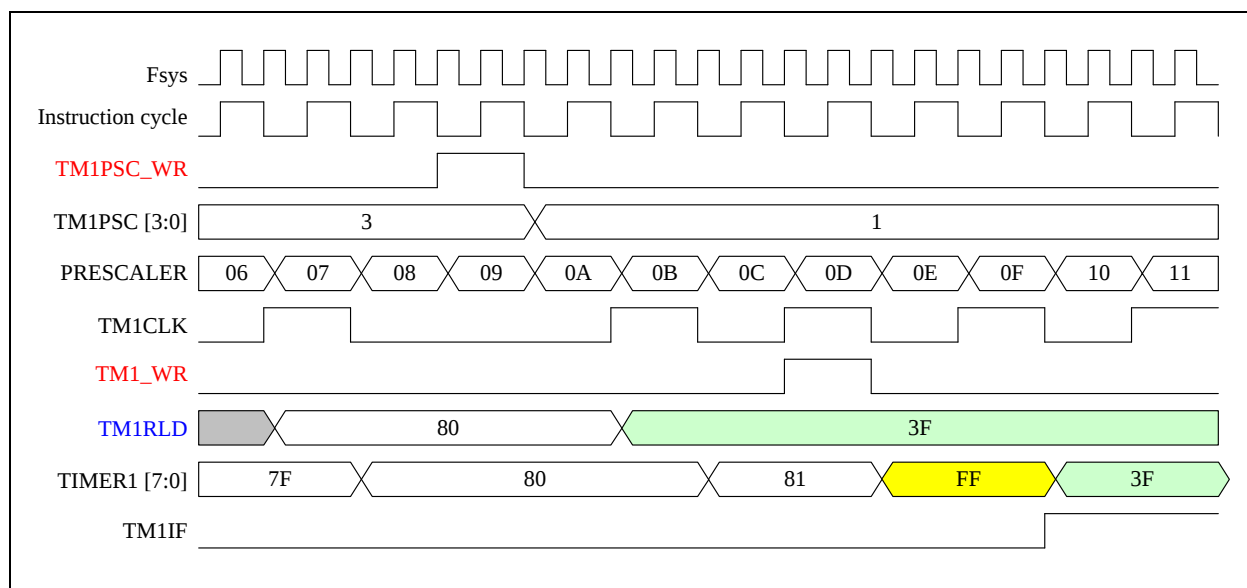
```
MOVWF    TM0
```

3.3 Timer1

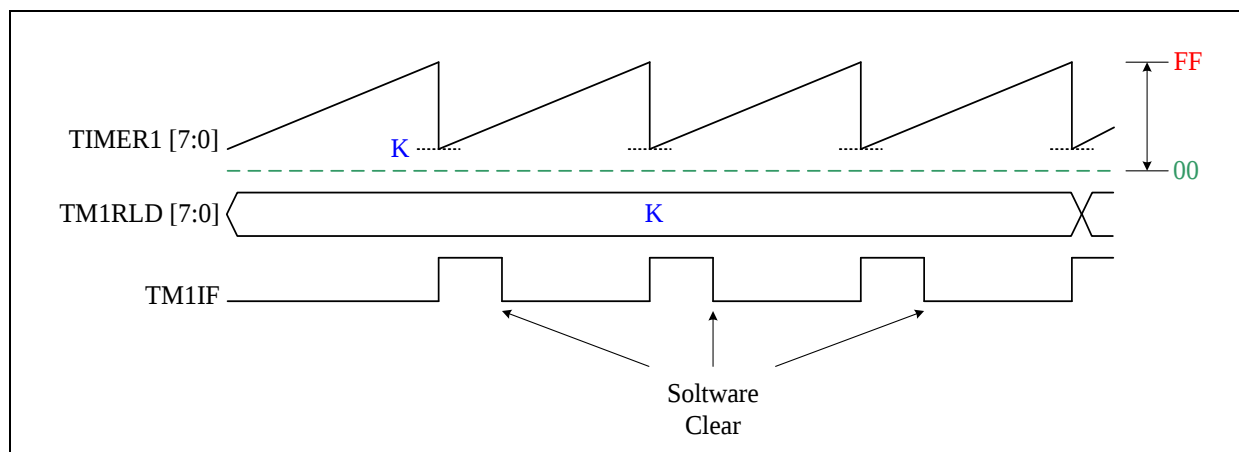
The Timer1 is an 8-bit wide register of F-Plane. It can be read or written as any other register of F-Plane. Besides, Timer1 increases itself periodically and automatically reloads a new " offset value " (TM1RLD) while it rolls over based on the pre-scaled instruction clock. The Timer1 increase rate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



Timer1 Block Diagram



Timer1 Timing Diagram



Timer1 Reload Diagram

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

; Setup TM1 clock source, divider and enable TM1OUT

```

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

; Set TM1 timer offset and stops TM1 counting

```

BSF      TM1STP        ; Stop TM1 counting (Default " 0 ").
MOVLW    F0H
MOVWF    TM1           ; Write F0H into TM1 counter (F0A, F-Plane)
    
```

; Enable TM0 timer and interrupt function.

```

MOVLW    11011111B    ; Clear TM1 request interrupt flag by byte operation
MOVWF    INTIF         ; F-Plane 09H

MOVLW    00100000B    ; Enable TM1 interrupt function.
MOVWR    INTIE         ;

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

Example:

Fsys=4 MHz , TM1PSC=1, TM1 clock source=Fsys/4=1 MHz

TM1RLD=0xF0,

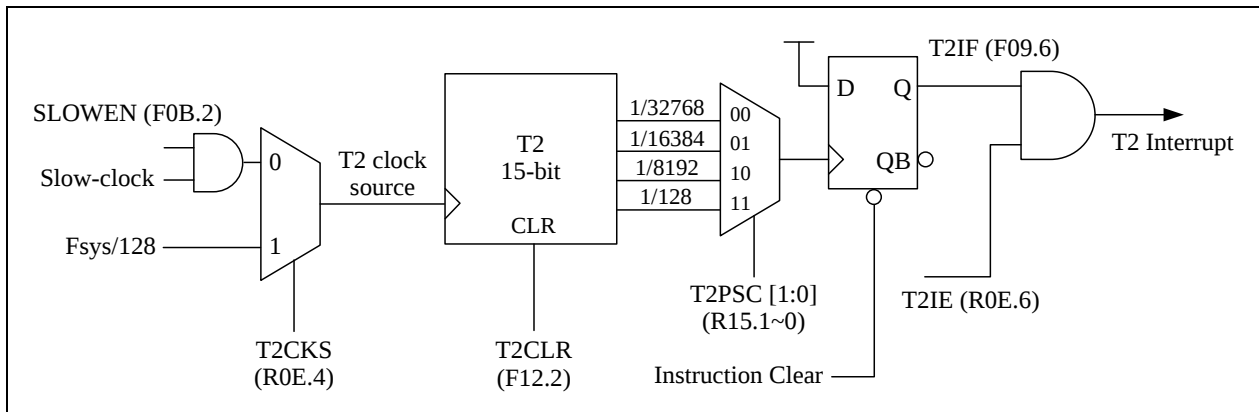
TM1 interrupt time= (1/1 MHz) * (0xFF-0xF0) =1 us*16=16 us

TM1OUT output time period=16 us *2=32 us.

TM1OUT output frequency=1/32 us=31.250 KHz.

3.4 T2:15-bit Timer

The T2 is a 15-bit counter and the clock sources are from either $F_{sys}/128$ or Slow-clock. It is used to generate time base interrupt and T2 counter block clock. The T2 content cannot be read by instructions. It generates interrupt flag T2IF (F09.6) with the clock divided by 32768/16384/8192/128 depends on T2PSC [1:0] (R15.1~0) register bits. The following figure shows the block diagram of T2.



T2 Block Diagram

Example:

[CPU running at FAST mode, F_{sys} =Fast-clock=FIRC 6 MHz]

◇Example:

; Setup T2 clock source and divider .

MOVLW	xxx11001B	;R15.4 (T2CKS) =1, T2 clock source = $F_{sys}/128$
MOVWR	R15	;R15.3~2 (FCLKDIV) =2, F_{sys} =Fast-clock=Fast-clock- ; pre/2
		; F_{sys} =Fast-clock=12 MHz/2=6 MHz
		;R15.1~0 (T2PSC) =1, Divided by 16384

BSF	T2CLR	;F12.2 (T2CLR) , Stop T2 counting.
-----	-------	------------------------------------

; Enable T2 timer and interrupt function.

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

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

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

T2 clock source is $F_{sys}/128=6\text{ MHz}/128 = 46875\text{ Hz}$, T2PSC =/16384

T2 frequency= $46875\text{ Hz}/16384=2.86\text{ Hz}$

Example:

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

◇Example:

; Setup CPU running at SLOW mode

```

MOV LW    00000000B    ;
MOV W F    F0B          ; Slow-clock type=SXT (Default " 01 ")
BSF        SLOWEN       ; Enable Slow-clock.
NOP
BSF        CPUCKS        ; Select Fsys=Slow-clock.
BSF        FASTSTP       ; Stop Fast-clock .

```

; Setup T2 clock source and divider

```

MOV LW    xxx0xx00B    ; R15.4 (T2CKS) =0, T2 clock source=Slow-clock
MOV W R    R15          ; Fsys=Slow-clock=32768 Hz
                          ; R15.1~0 (T2PSC) =0, Divided by 32768

BSF        T2CLR        ; Stop T2 counting.

```

; Enable T2 timer and interrupt function.

```

MOV LW    10111111B    ; Clear T2 request interrupt flag
MOV W F    INTIF        ; F-Plane 09H

MOV LW    01000000B    ; Enable T2 interrupt function.
MOV W R    INTIE        ; R0E

BCF T2CLR                          ; Enable T2 counting (Default " 0 ").

```

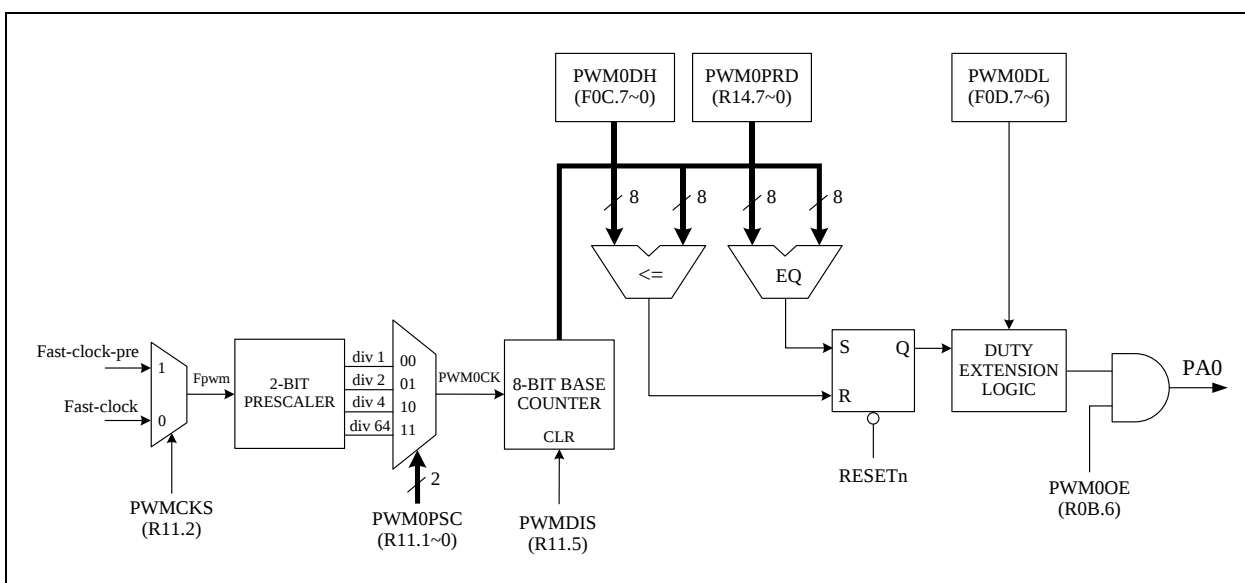
T2 clock source is Slow-clock=32768 Hz, T2PSC=/32768,

T2 frequency=32768 Hz/32768=1 Hz

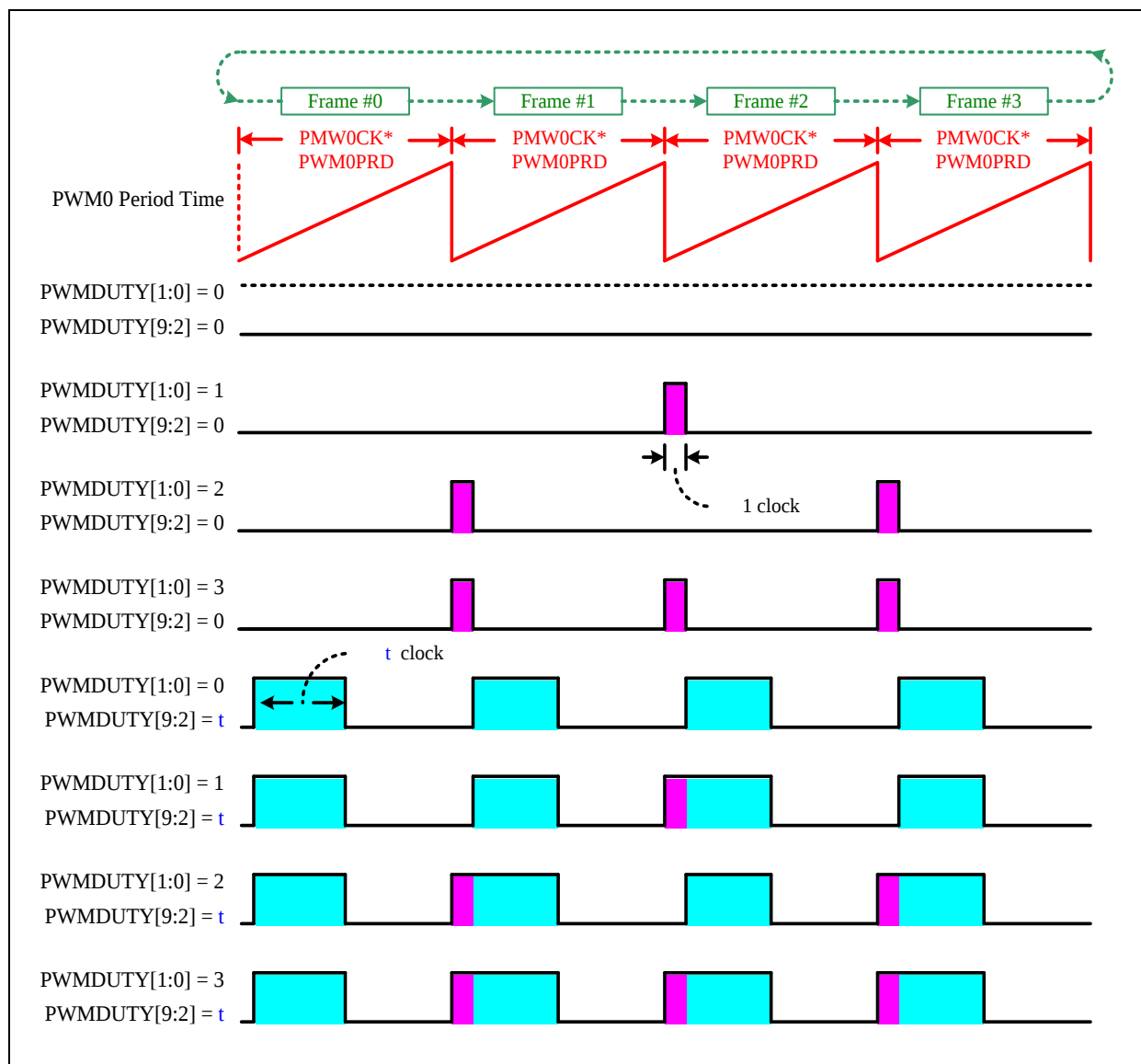
3.5 PWM0: (8+2) bits PWM

The PWM can generate fix frequency waveform with 1024 duty resolution based on Fpwm clock, the Fpwm can select Fast-clock or Fast-clock-pre by PWMCKS (R11.2). A spread LSB technique allows PWM to run its frequency at “ System Clock divided by 256 ” instead of “ System Clock divided by 1024 ”, which means the PWM is 4 times faster than normal. The advantage of higher PWM frequency is that the post RC filter can transform the PWM signal to more stable DC voltage level. The PWM output signal reset to low level whenever the 8-bit base counter matches the 8-bit MSB of PWM duty register PWM0DH (F0C.7~0). When the base counter rolls over, the 2-bit LSB of PWM duty register PWM0DL (F0D.7~6) decides whether to set the PWM output signal high immediately or set it high after one clock cycle delay.

The PWM0 period can be set by writing period value to PWM0PRD register (R14). Note that changing the PWM0PRD is immediately changing the PWM0PRD values, which are different from PWM0DH/PWM0DL which has buffer to update the duty at the end of current period. The Programmer must pay attention to the current time to change PWM0PRD by observing the following figure. There is a digital comparator that compares the PWM0 counter and PWM0PRD, if PWM0 counter is larger than PWM0PRD after setting the PWM0PRD, a fault long PWM cycle will be generated because PWM0 counter must count to overflow then keep counting PWM0PRD to finish the cycle.



PWM0 Block Diagram



PWM0 8+2 Timing Diagram

Example:

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

; Setup PWM0 clock source and prescaler .

```
MOVLW    00000101B    ; R11.5 (PWMDIS) =0, PWM0/PWM1 clock enable
MOVWR    R11           ; R11.2 (PWMCKS) =1, Fpwm=Fast-clock-pre(12 MHz)
                        ; R11.1~0 (PWM0PSC) =1,
                        PWM clock source=Fpwm/2=6 MHz
```

```
MOVLW    80H
MOVWR    PWM0PRD        ; Set PWM0 period[7:0] =80H.
```

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

```
MOVLW    20H
MOVWF    PWM0DH         ; Set PWM0 duty[9:2] =20H
```

```
MOVLW    01000000B
MOVWR    R0B           ; R0B.6 (PWM0OE) =1, Enable PWM0 OUT (PA0) .
```

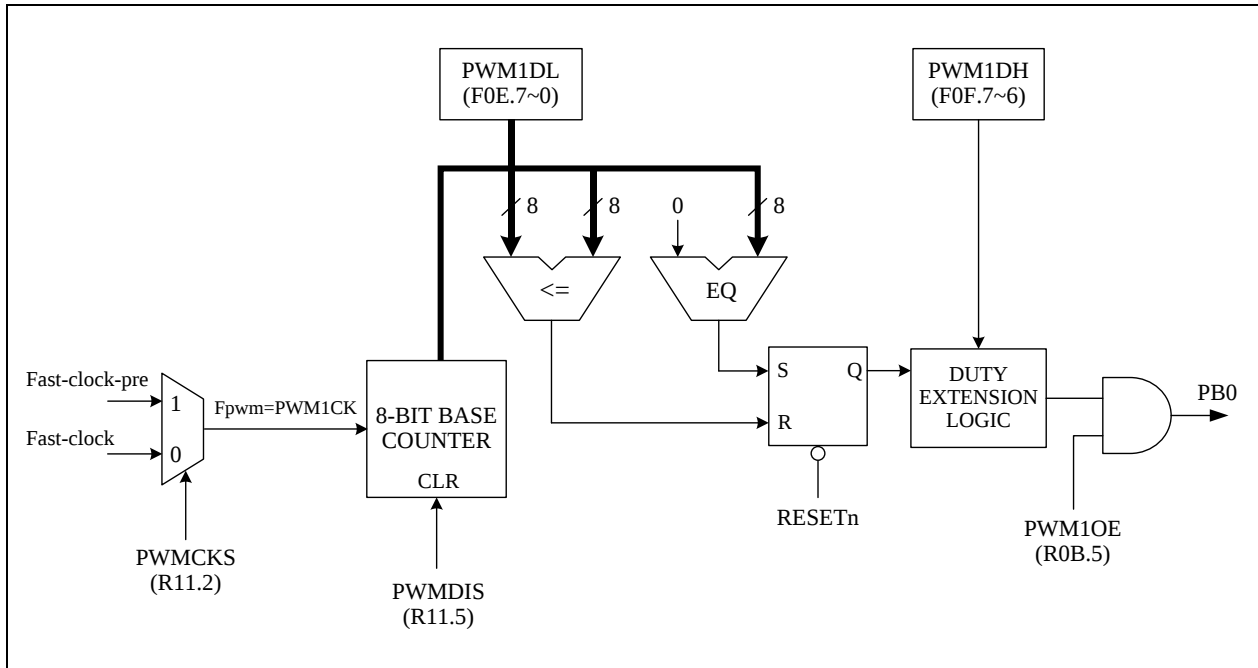
Fsys=4 MHz, PWM clock source=Fpwm/2=6 MHz, PWM0PRD=80H, PWM0DL=00H, PWM0DH = 20H

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

PWM0 output duty=32/128=25%.

3.6 PWM1: (8+2) bits PWM

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



PWM1 Block Diagram

Example:

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

; Setup PWM1 clock source.

```
MOVLW    0000000B    ; R11.5 (PWMDIS) =0, PWM0/PWM1 clock enable
MOVWR    R11          ; R11.2 (PWMCKS) =0, Fpwm=Fast-clock (4 MHz)
```

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

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

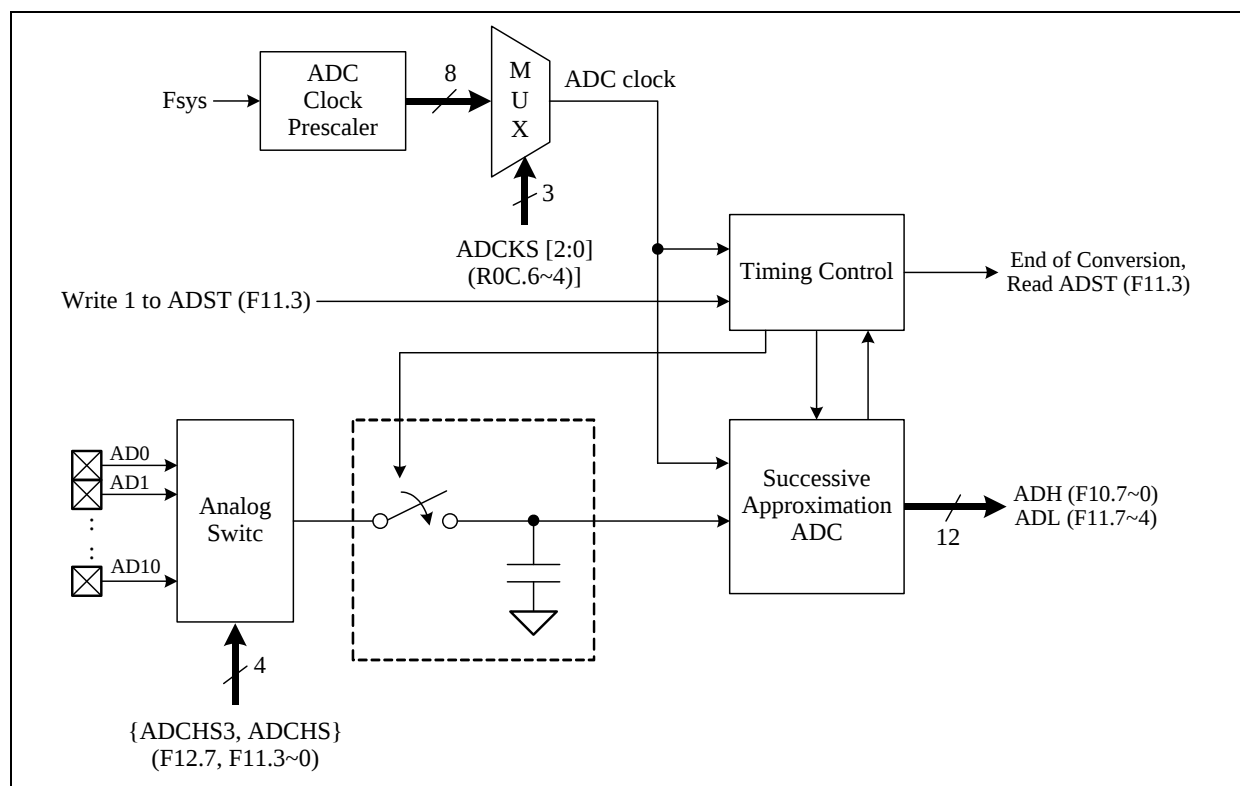
```
MOVLW    00100000B
MOVWR    R0B          ; R0B.5 (PWM1OE) =1, Enable PWM1 OUT (PB0)
```

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

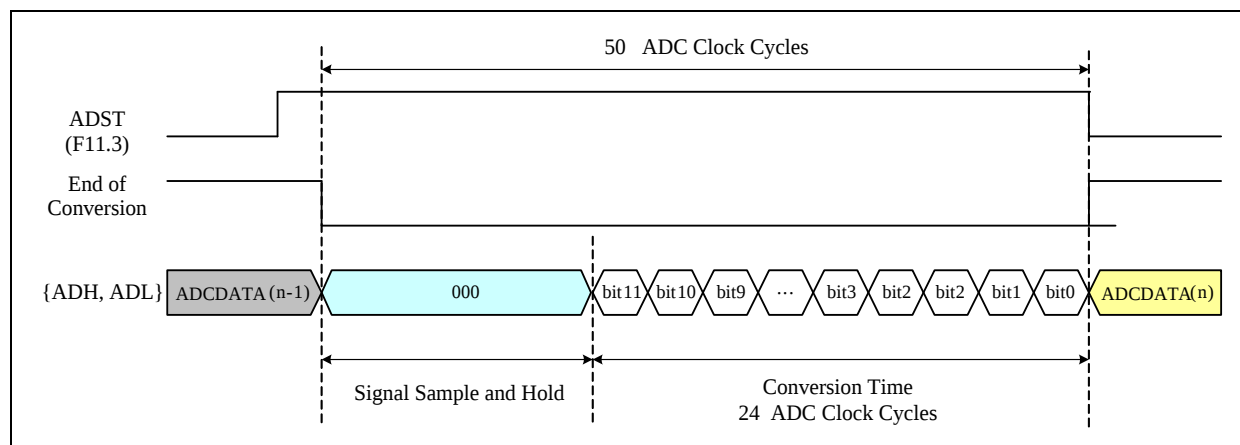
PWM1 output frequency=4 MHz/256=15.625 KHz

PWM1 output duty=128/256=50%.

3.7 Analog-to-Digital Converter



The 12-bit ADC (Analog to Digital Converter) consists of a 12-channel analog input multiplexer, control register, clock generator, 12-bit successive approximation register, and output data register. To use the ADC, user needs to set ADCKS (R0C.6~4) to choose a proper ADC clock frequency, which must be less than 1 MHz. User then launches the ADC conversion by setting the ADST (F11.3) control bit. After end of conversion, H/W automatic clears the ADST (F11.3) bit. User can poll this bit to know the conversion status. The ADPIE8 (R13.2~0), ADPIE (R12.7~0) control registers are used for ADC pin configuration, user can write the corresponding bit to “0” when the pin is used as an ADC input. The setting can disable the pin logical input path to save power consumption.



Example:

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

ADC clock frequency=1 MHz, ADC channel=ADC2 (PA2).

◇Example:

```

MOVLW    01100000B    ; Fsys=4 MHz
MOVWR    R0C           ; R0C.6~4 (ADCKS) =ADC clock=Fsys/4=1 MHz

MOVLW    00000100B    ; ADC2 (PA2) pull-high disable
MOVWR    PAPUN;

MOVLW    11111011B
MOVWR    ADPIE         ; R12, Disable ADC2 (PA2) digital input to saving power
                        ; in STOP/IDLE Mode

MOVLW    00000010B
MOVWF    ADCTL         ; F11.2~0 (ADCHS [2:0] ) =2, ADC select ADC2 (PA2 pin).
BCF      ADCHS3        ; F12.7 (ADCHS [3] ) =0

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

```

WAIT_ADC:

```

BTFSC    ADST          ; Wait ADC conversion finish.
GOTO     WAIT_ADC

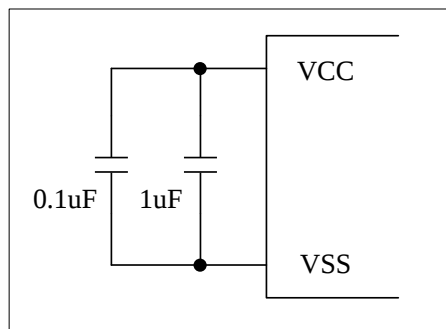
MOVWF    ADH           ; F10.7~0, Read ADC result [11:4] into W
MOVWF    ADCTL         ; F11.7~4, Read ADC result [3:0] into W

:
:

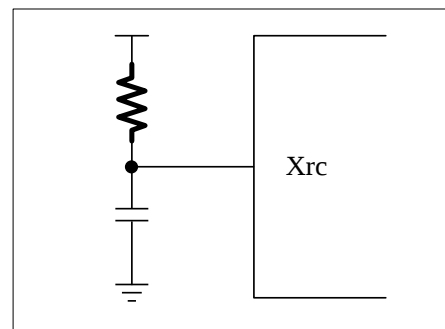
```

3.8 System Clock Oscillator

System clock can be operated in four different oscillation modes, which is selected by setting the CLK in the SYSCFG register. In Slow/Fast Crystal (SXT/FXT) mode, a crystal or ceramic resonator is connected to the Xin and Xout pins to establish oscillation. In external RC (XRC) mode, the external resistor and capacitor determine the oscillation frequency. In the Fast Internal RC (FIRC) mode, the on-chip oscillator generates 12/6/4/1.5 MHz system clock, which is controlled by register FCLKDIV [1:0] (R15.3~2) bits. Since power noise degrades the performance of Internal Clock Oscillator, placing power supply bypass capacitors 1 uF and 0.1 uF very close to VCC/VSS pins improves the stability of clock and the overall system.



Internal RC Mode

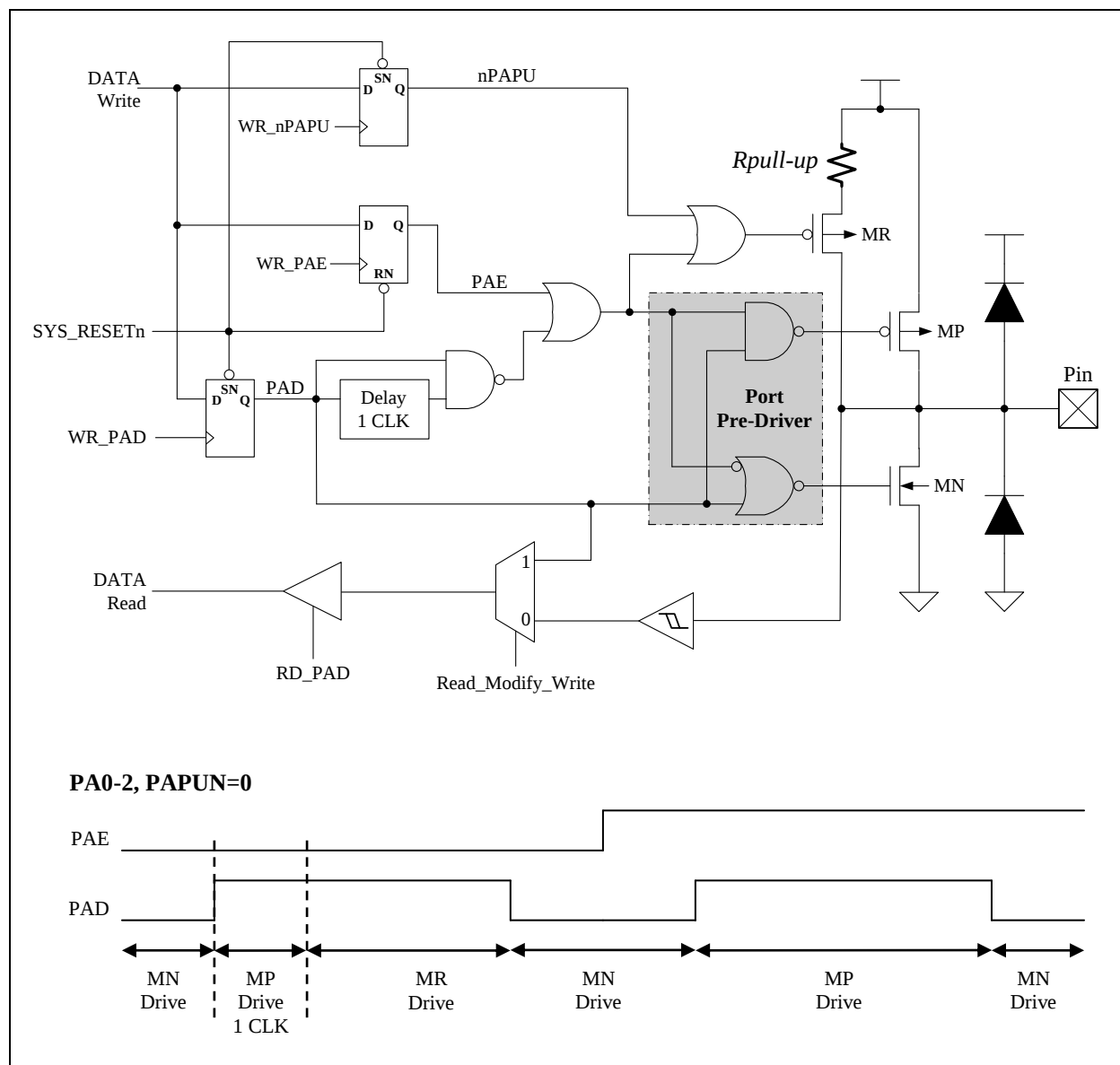


External RC Oscillator

4. I/O Port

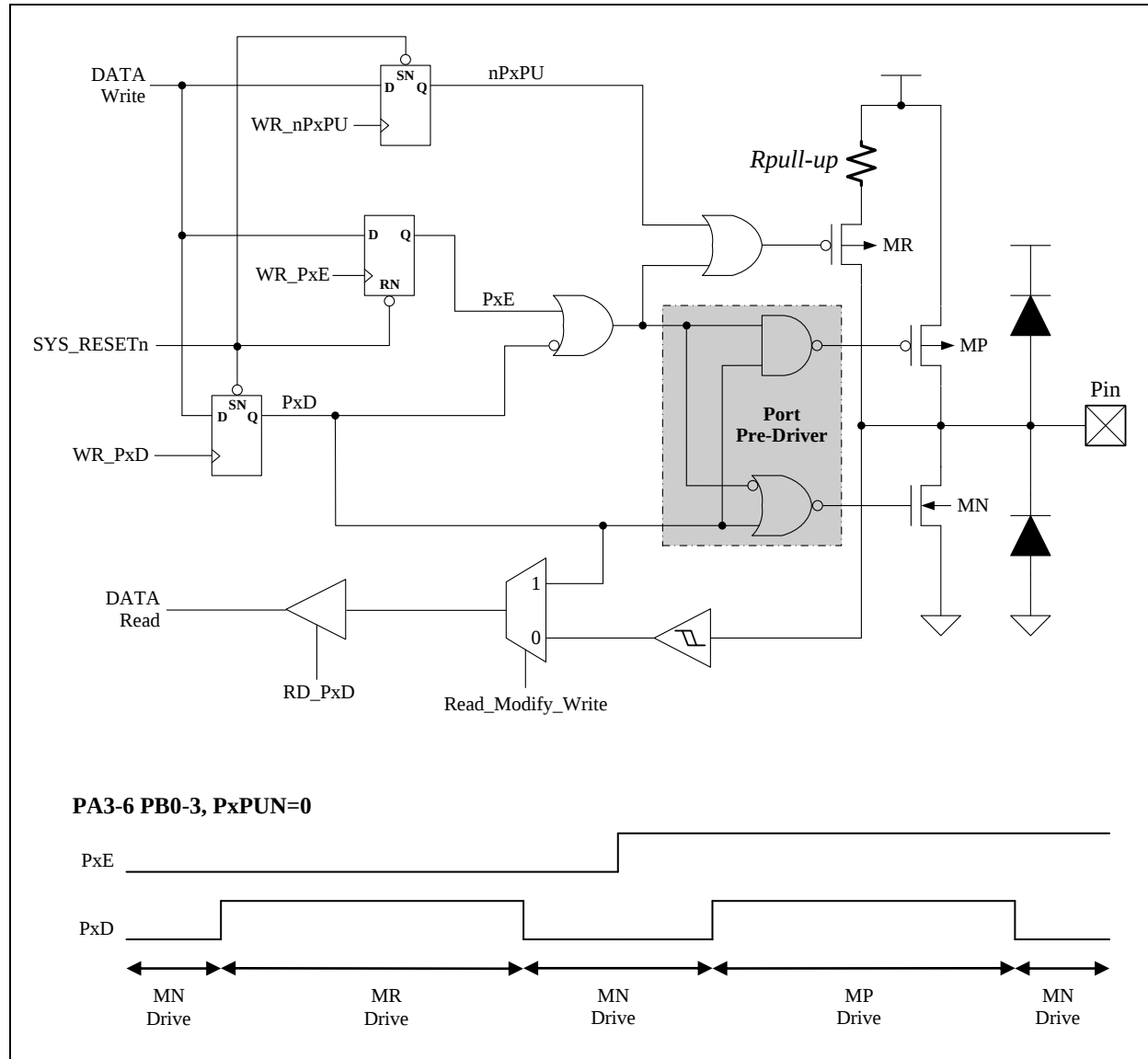
4.1 PA0-2

These pins can be used as Schmitt-trigger input, CMOS push-pull output or “pseudo-open-drain” output. The pull-up resistor is assignable to each pin by S/W setting. To use the pin in Schmitt-trigger input mode, S/W needs to set the PAE=0 and PAD=1. To use the pin in “pseudo-open-drain” mode, S/W sets the PAE=0. The benefit of pseudo-open-drain structure is that the output rise time can be much faster than pure open-drain structure. S/W sets PAE=1 to use the pin in CMOS push-pull output mode. Reading the pin data (PAD) has different meaning. In “Read-Modify-Write” instruction, CPU actually reads the output data register. In the others instructions, CPU reads the pin state. The so-called “Read-Modify-Write” instruction includes BSF, BCF and all instructions using F-Plane as destination.



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

These pins are almost the same as PA0-2, except they do not support pseudo-open-drain mode. They can be used in pure open-drain mode, instead.



◇Example: I/O mode selecting

```
MOVLW    FFH
MOVWF     PDD
MOVLW     00H
MOVWR     PDPUN    ; Set PD port pull-high enable
MOVLW     00H
MOVWR     PDE      ; Set all ports to be Schmitt-trigger input
```

◇Example: Set PA0-2 is pseudo-open-drain mode

```
MOVLW     xxxxx000B
MOVWR     PAE      ; Set PA2-PA0 as pseudo-open-drain mode

MOVLW     xxxxx000B
MOVWF     PAD      ; PA2~PA0 output low level.
```

◇Example: Set PA0-2 is CMOS push-pull output mode.

```
MOVLW     xxxxx111B
MOVWR     PAE      ; Set PA2-PA0 as CMOS push-pull output mode
```

◇Example: Read data from input port.

```
MOVLW     FFH      ; “pseudo-open-drain” I/O structure, port must output High first
MOVWF     PDD      ; before read pin
MOVWF     PDD      ; Read data from Port D.
```

◇Example: Write data to output port.

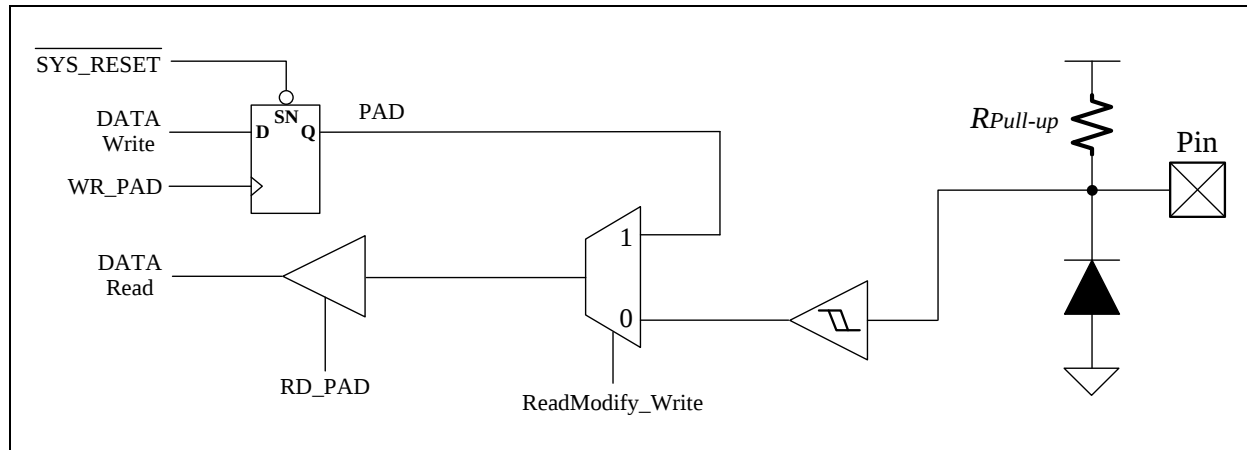
```
MOVLW     55H
MOVWF     PAD      ; Write data 55H to Port A.
MOVWF     PBD      ; Write data 55H to Port B.
```

◇Example: Write one bit data to output port.

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

4.3 PA7

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



◇Example: Read state from PA7.

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

```

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

MEMORY MAP

F-Plane

Name	Address	R/W	Rst	Description
(F00) INDF		Function related to: RAM W/R		
INDF	00.7~0	R/W	-	Not a physical register, addressing INDF actually point to the register whose address is contained in the FSR register
(F01) TM0		Function related to: Timer0		
TM0	01.7~0	R/W	0	Timer0 content
(F02) PCL		Function related to: PROGRAM COUNT		
PCL	02.7~0	R/W	0	Programming Counter LSB [7~0]
(F03) STATUS		Function related to: STATUS		
GB0	03.7	R/W	0	General purpose bit 0
GB1	03.6	R/W	0	General purpose bit 1
RAMBK	03.5	R/W	0	RAM Bank Selection (Fixed to 0)
TO	03.4	R	0	WDT timeout flag, cleared by PWRST, ‘SLEEP’ or ‘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
(F04) FSR		Function related to: RAM W/R		
GB2	04.7	R/W	0	General purpose bit 2
FSR	04.6~0	R/W	-	File Select Register, indirect address mode pointer
(F05) PAD		Function related to: Port A		
PAD7	05.7	R	-	PA7 pin state
PAD	05.6~0	R	-	Port A pin or “ data register ” state
		W	7F	Port A output data register
(F06) PBD		Function related to: Port B		
PBD	06.1~0	R	-	Port B pin or “ data register ” state
		W	3	Port B output data register
(F07) PDD		Function related to: Port D		
PDD	07.7~0	R	-	Port D pin or “ data register ” state
		W	FF	Port D output data register

Name	Address	R/W	Rst	Description
(F09) INTIF				

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

R-Plane

Name	Address	R/W	Rst	Description
(R02) TM0CTL				Function related to: TCOUT/TM0
TCOPSC	02.7~6	W	0	TCOUT prescaler 0: Fsys/2 1: Fsys/4 2: Fsys/8 3: Fsys/16
TM0EDG	02.5	W	0	Timer0 prescaler counting edge for TM0CKI (PA2) pin 0: rising edge 1: falling edge
TM0CKS	02.4	W	0	Timer0 prescaler clock source 0: Instruction cycle 1: TM0CKI (PA2) pin
TM0PSC	02.3~0	W	0	Timer0 prescaler. Timer0 prescaler clock source divided by 0000: Fsys/2 0101: Fsys/64 0001: Fsys/4 0110: Fsys/128 0010: Fsys/8 0111: Fsys/256 0011: Fsys/16 1xxx: Fsys/512 0100: Fsys/32
(R03) PWRDN				Function related to: Power Down
PWRDN	03	W	-	write this register to enter STOP/IDLE Mode (i.e. ' SLEEP ' instruction)
(R04) WDTCLR				Function related to: WDT
WDTCLR	04	W	-	write this register to clear WDT timer (i.e. ' CLRWDT ' instruction)
(R05) PAE				Function related to: Port A
PAE	05.6~3	W	0	Each bit controls its corresponding pin, if the bit is 0: the pin is open-drain output or Schmitt-trigger input 1: the pin is CMOS push-pull output
	05.2~0	W	0	Each bit controls its corresponding pin, if the bit is 0: the pin is pseudo-open-drain output or Schmitt-trigger input 1: the pin is CMOS push-pull output
(R06) PBE				Function related to: Port B
PBE	06.1~0	W	0	Each bit controls its corresponding pin, if the bit is 0: the pin is open-drain output or Schmitt-trigger input 1: the pin is CMOS push-pull output
(R07) PDE				Function related to: Port D
PDE	07.7~0	W	0	Each bit controls its corresponding pin, if the bit is 0: the pin is open-drain output or Schmitt-trigger input 1: the pin is CMOS push-pull output
(R08) PAPUN				Function related to: Port A
PAPUN	08.6~0	W	7F	Each bit controls its corresponding pin, if the bit is 0: the pin pull up resistor is enabled, except a. the pin's output data register (PAD) is 0 b. the pin's CMOS push-pull mode is chosen (PAE=1) c. the pin is working for PWM output 1: the pin pull up resistor is disabled

Name	Address	R/W	Rst	Description
(R09) PBPUN				Function related to: Port B
PBPUN	09.1~0	W	3	Each bit controls its corresponding pin, if the bit is 0: the pin pull up resistor is enabled, except a. the pin's output data register (PBD) is 0 b. the pin's CMOS push-pull mode is chosen (PBE=1) c. the pin is working for PWM output 1: the pin pull up resistor is disabled
(R0A) PDPUN				Function related to: Port D
PDPUN	0a.7~0	W	FF	Each bit controls its corresponding pin, if the bit is 0: the pin pull up resistor is enabled, except a. the pin's output data register (PDD) is 0 b. the pin's CMOS push-pull mode is chosen (PDE=1) c. pins are working for TM1OUT/TCOUT output 1: the pin pull up resistor is disabled
(R0B) MR0B				Function related to: PWM0/PWM1/INT1/TCOUT/TM1/WKT
	0b.7			Reserved
PWM0OE	0b.6	W	0	Enable PWM0 output to PA0 pin
PWM1OE	0b.5	W	0	Enable PWM1 output to PB0 pin
INT1EDG	0b.4	W	0	INT1 pin (PA1) edge interrupt event 0: falling edge to trigger 1: rising edge to trigger
TCOE	0b.3	W	0	Enable post-prescaler Instruction Cycle (Fsys/2) output to PD6 pin (TCOUT)
TM1OE	0b.2	W	0	(TM57MA25 don't support)
WKT PSC	0b.1~0	W	3	WKT period (@VCC=3V) 00: 15 ms 01: 30 ms 10: 60 ms 11: 120 ms
(R0C) MR0C				Function related to: ADC/TM1
	0c.7			Reserved
ADCKS	0c.6~4	W	0	ADC clock frequency selection: 000: Fsys/256 100: Fsys/16 001: Fsys/128 101: Fsys/8 010: Fsys/64 110: Fsys/4 011: Fsys/32 111: Fsys/2
TM1PSC	0c.3~0	W	0	Timer1 prescaler. Timer1 clock source divided by 0000: Fsys/2 0101: Fsys/64 0001: Fsys/4 0110: Fsys/128 0010: Fsys/8 0111: Fsys/256 0011: Fsys/16 1xxx: Fsys/512 0100: Fsys/32
(R0D) TM1RLD				Function related to: TM1
TM1RLD	0d.7~0	W	0	Timer1 reload offset value while it rolls over

Name	Address	R/W	Rst	Description
(R0E) INTIE				Function related to: Interrupt Enable
	0e.7			Reserved
T2IE	0e.6	W	0	T2 interrupt enable, 1=enable, 0=disable
TM1IE	0e.5	W	0	Timer1 interrupt enable, 1=enable, 0=disable
TM0IE	0e.4	W	0	Timer0 interrupt enable, 1=enable, 0=disable
WKTIE	0e.3	W	0	Wakeup Timer interrupt enable, 1=enable, 0=disable Set 0 to clear & disable WKT timer
INT2IE	0e.2	W	0	INT2 pin (PA7) interrupt enable, 1=enable, 0=disable
INT1IE	0e.1	W	0	INT1 pin (PA1) interrupt enable, 1=enable, 0=disable
INT0IE	0e.0	W	0	INT0 pin (PA6) interrupt enable, 1=enable, 0=disable
(R0F) TEST				
TSTREG	0f.7~0	W	0	Test mode register, user does not write it.
(R11) MR11				Function related to: EFT/PWM0/PWM1
	11.7	W	0	Reserved, keep write to 0
VCCFLT	11.6	W	0	Enable EFT enhance operation mode, 1=enable, 0=disable
PWMDIS	11.5	W	0	PWM0/PWM1 clock disable (PWMDIS=1) or enable (PWMDIS=0)
	11.4~3			Reserved
PWMCKS	11.2	W	0	PWM0/PWM1 clock source, Fpwm= 0: Fast-clock 1: Fast-clock-pre
PWM0PSC	11.1~0	W	0	PWM0 prescaler, PWM0 clock source divided by 00: Fpwm 01: Fpwm/2 10: Fpwm/4 11: Fpwm/64
(R12) ADPIE				Function related to: ADC
ADPIE	12.7	W	1	Each bit controls its corresponding port I/O enable pin, if the bit is 0: enable ADC7 channel input 1: enable PB0 I/O digital input
	12.6	W	1	0: enable ADC6 channel input 1: enable PD6 I/O digital input
	12.5	W	1	0: enable ADC5 channel input 1: enable PA5 I/O digital input
	12.4	W	1	0: enable ADC4 channel input 1: enable PD7 I/O digital input
	12.3	W	1	0: enable ADC3 channel input 1: enable PB1 I/O digital input
	12.2	W	1	0: enable ADC2 channel input 1: enable PA2 I/O digital input
	12.1	W	1	0: enable ADC1 channel input 1: enable PA1 I/O digital input
	12.0	W	1	0: enable ADC0 channel input 1: enable PA6 I/O digital input
(R13) ADPIE8				Function related to: ADC
ADPIE8	13.7			Reserved
	13.6	W	1	0: enable non-digital input 1: enable PD0 I/O digital input
	13.5	W	1	0: enable non-digital input 1: enable PD1 I/O digital input
	13.4	W	1	0: enable non-digital input 1: enable PD2 I/O digital input
	13.3	W	1	0: enable non-digital input 1: enable PD3 I/O digital input
	13.2	W	1	0: enable ADC10 channel input 1: enable PD4 I/O digital input
	13.1	W	1	0: enable ADC9 channel input 1: enable PD5 I/O digital input
	13.0	W	1	0: enable ADC8 channel input 1: enable PA0 I/O digital input

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

INSTRUCTION SET

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

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

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

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

Mnemonic		Op Code	Cycle	Flag Affect	Description
Byte-Oriented File Register Instruction					
ADDWF	f, d	00 0111 dfff ffff	1	C, DC, Z	Add W and " f "
ANDWF	f, d	00 0101 dfff ffff	1	Z	AND W with " f "
CLRF	f	00 0001 1fff ffff	1	Z	Clear " f "
CLRW		00 0001 0100 0000	1	Z	Clear W
COMF	f, d	00 1001 dfff ffff	1	Z	Complement " f "
DECF	f, d	00 0011 dfff ffff	1	Z	Decrement " f "
DECFSZ	f, d	00 1011 dfff ffff	1 or 2	-	Decrement " f ", skip if zero
INCF	f, d	00 1010 dfff ffff	1	Z	Increment " f "
INCFSZ	f, d	00 1111 dfff ffff	1 or 2	-	Increment " f ", skip if zero
IORWF	f, d	00 0100 dfff ffff	1	Z	OR W with " f "
MOVFW	f	00 1000 0fff ffff	1	-	Move " f " to W
MOVWF	f	00 0000 1fff ffff	1	-	Move W to " f "
MOVWR	r	00 0000 00rr rrrr	1	-	Move W to " r "
RLF	f, d	00 1101 dfff ffff	1	C	Rotate left " f " through carry
RRF	f, d	00 1100 dfff ffff	1	C	Rotate right " f " through carry
SUBWF	f, d	00 0010 dfff ffff	1	C, DC, Z	Subtract W from " f "
SWAPF	f, d	00 1110 dfff ffff	1	-	Swap nibbles in " f "
TESTZ	f	00 1000 1fff ffff	1	Z	Test if " f " is zero
XORWF	f, d	00 0110 dfff ffff	1	Z	XOR W with " f "
Bit-Oriented File Register Instruction					
BCF	f, b	01 000b bbff ffff	1	-	Clear " b " bit of " f "
BSF	f, b	01 001b bbff ffff	1	-	Set " b " bit of " f "
BTFSC	f, b	01 010b bbff ffff	1 or 2	-	Test " b " bit of " f ", skip if clear
BTFSS	f, b	01 011b bbff ffff	1 or 2	-	Test " b " bit of " f ", skip if set
Literal and Control Instruction					
ADDLW	k	01 1100 kkkk kkkk	1	C, DC, Z	Add Literal " k " and W
ANDLW	k	01 1011 kkkk kkkk	1	Z	AND Literal " k " with W
CALL	k	10 kkkk kkkk kkkk	2	-	Call subroutine " k "
CLRWDI		00 0000 0000 0100	1	TO, PD	Clear Watch Dog Timer
GOTO	k	11 kkkk kkkk kkkk	2	-	Jump to branch " k "
IORLW	k	01 1010 kkkk kkkk	1	Z	OR Literal " k " with W
MOVLW	k	01 1001 kkkk kkkk	1	-	Move Literal " k " to W
NOP		00 0000 0000 0000	1	-	No operation
RET		00 0000 0100 0000	2	-	Return from subroutine
RETI		00 0000 0110 0000	2	-	Return from interrupt
RETLW	k	01 1000 kkkk kkkk	2	-	Return with Literal in W
TABRL		00 0000 0101 0000	2	-	Lookup ROM low data to W
TABRH		00 0000 0101 1000	2	-	Lookup ROM high data to W
SLEEP		00 0000 0000 0011	1	TO, PD	Go into Power-down mode, Clock oscillation stops
XORLW	k	01 1111 kkkk kkkk	1	Z	XOR Literal " k " with W

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

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

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

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

BCF Clear "b" bit of "f"

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

BSF Set "b" bit of "f"

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

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

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

CALL Call subroutine "k"

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

CLRF Clear "f"

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

CLRW Clear W

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

CLRWDW Clear Watchdog Timer

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

COMF

Complement "f"

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

DECF

Decrement "f"

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

DECFSZ

Decrement "f", Skip if 0

Syntax	DECFSZ f [,d]
Operands	f : 00h ~ 7Fh, d : 0, 1
Operation	(destination) $\leftarrow$ (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 $\neq$ 0, PC = LABEL1 + 1

GOTO

Unconditional Branch

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



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

INCFSZ	Increment "f", Skip if 0
Syntax	INCFSZ f [,d]
Operands	f : 00h ~ 7Fh, d : 0, 1
Operation	(destination) $\leftarrow$ (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 $\neq$ 0, PC = LABEL1 + 1

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

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

MOVFW Move "f" to W

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

MOVLW Move Literal to W

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

MOVWF Move W to "f"

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

MOVWR Move W to "r"

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



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

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

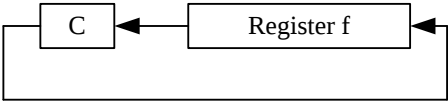
TABRL Return DPTR low byte to W

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

TABRH Return DPTR high byte to W

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

RLF Rotate Left "f" through Carry

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

RRF Rotate Right "f" through Carry

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

SLEEP Go into standby mode, Clock oscillation stops

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

SUBWF Subtract W from "f"

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

SWAPF
Swap Nibbles in "f"

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

TESTZ
Test if "f" is zero

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

XORLW
Exclusive OR Literal with W

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

XORWF
Exclusive OR W with "f"

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

ELECTRICAL CHARACTERISTICS

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

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

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

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	$V_{CC}=5\text{V}, V_{OH}=4.5\text{V}$	4	8		mA
			$V_{CC}=3\text{V}, V_{OH}=2.7\text{V}$	2	4		
Output Low Current	I_{OL}	All Output	$V_{CC}=5\text{V}, V_{OL}=0.5\text{V}$	9	18		mA
			$V_{CC}=3\text{V}, V_{OL}=0.3\text{V}$	4	8		
Input Leakage Current (pin high)	I_{ILH}	All Input	$V_{IN}=V_{CC}$	–	–	1	μA
Input Leakage Current (pin low)	I_{ILL}	All Input	$V_{IN}=0\text{V}$	–	–	–1	μA
Power Supply Current (No Load)	I_{CC}	FAST mode FIRC 12 MHz,	$V_{CC}=4.5$ to 5.5V	–	3		mA
		FAST mode FIRC 4 MHz	$V_{CC}=4.5$ to 5.5V	–	1		
		FAST mode FIRC 1.5 MHz MODE3V=1	$V_{CC}=3.0\text{V}$	–	0.5		mA
		SLOW mode SIRC2 KHz MODE3V=1	$V_{CC}=3.0\text{V}$		6		μA
		IDLE mode SIRC2 KHz MODE3V=1 T2PSC=0	$V_{CC}=3.0\text{V}$		3		μA
			$V_{CC}=5.0\text{V}$ LVR disable in STOP	–	0.1	1	μA
		STOP mode MODE3V=0	$V_{CC}=5.0\text{V}$, LVR enable	–	4	6	
System Operating Voltage	V_{SYS}	MODE3V=0	$F_{sys}=1\text{MHz}$	2.0	–	5.5	V
			$F_{sys}=4\text{MHz}$	2.0	–	5.5	
			$F_{sys}=6\text{MHz}$	2.2	–	5.5	
			$F_{sys}=12\text{MHz}$	2.6	–	5.5	
		MODE3V=1 LVR _{th} force at 2.0V	$F_{sys}=1\text{MHz}$	2.0	–	3.6	
			$F_{sys}=4\text{MHz}$	2.0	–	3.6	
			$F_{sy}=8\text{MHz}$	2.2	–	3.6	
Pull-up Resistor	R_{UP}	$V_{IN}=0\text{ V}$ Ports A/B/D	$V_{CC}=5.0\text{V}$		110		$\text{K}\Omega$
			$V_{CC}=3.0\text{V}$		200		
		$V_{IN}=0\text{ V}$ PA7	$V_{CC}=5.0\text{V}$		60		$\text{K}\Omega$
			$V_{CC}=3.0\text{V}$		60		

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

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

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

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	—	—	μs
WDT time	$V_{CC}=5\text{ V}$, $\text{WDTPSC}=11$	-20%	2000	+20%	ms
WKT time	$V_{CC}=3\text{ V}$, $\text{WKT PSC}=11$	-20%	120	+20%	ms
	$V_{CC}=5\text{ V}$, $\text{WKT PSC}=11$		113		
CPU start up time	$V_{CC}=5\text{ V}$	—	14	—	ms

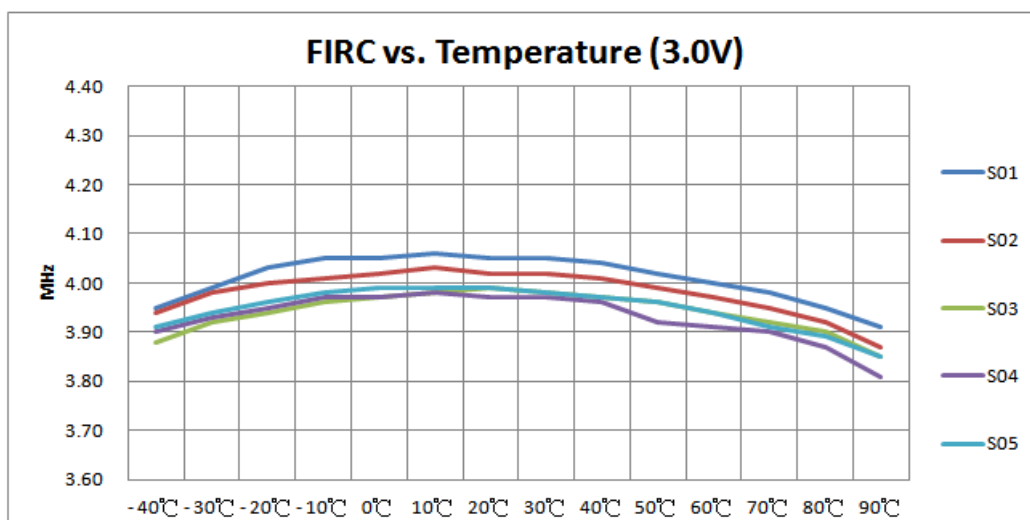
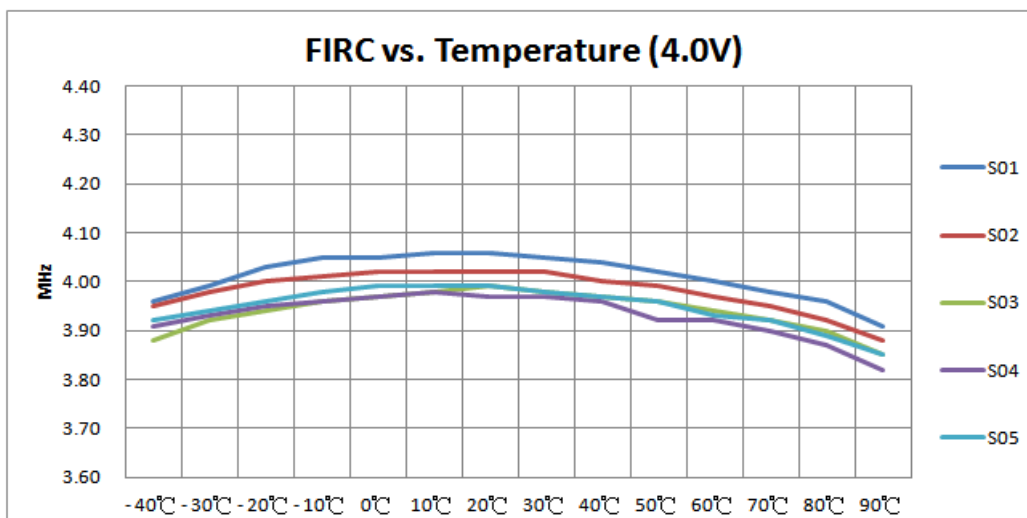
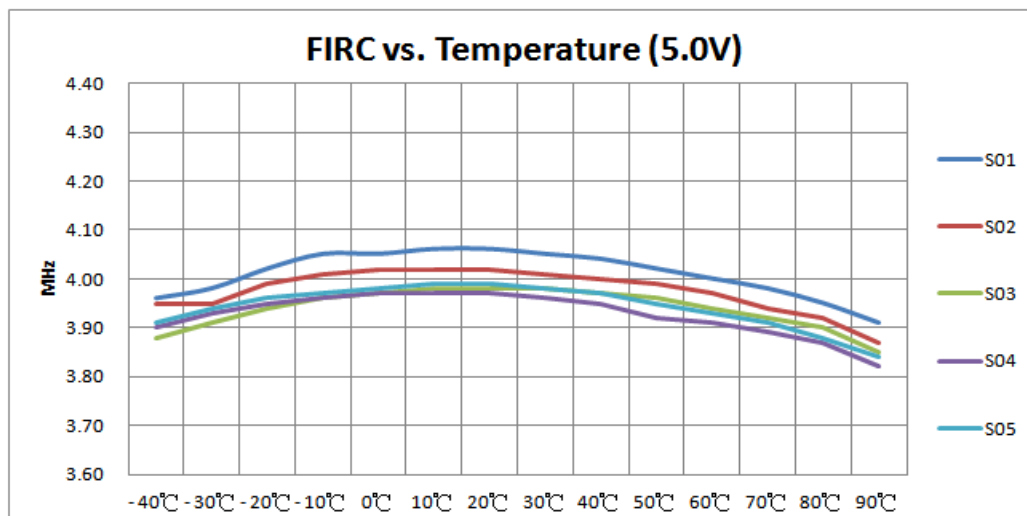
5. LVR Circuit Characteristics ($T_A=25^{\circ}\text{C}$)

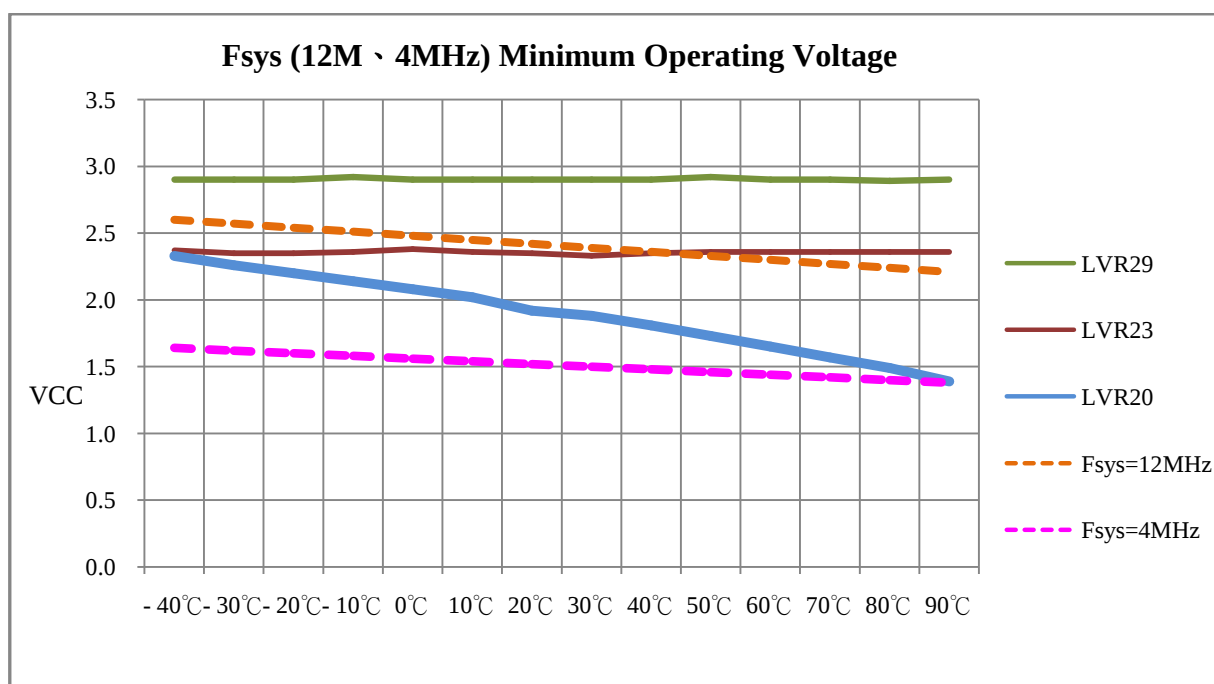
Parameter	Symbol	Min	Typ	Max	Unit
LVR Reference Voltage	LVR_{th}	-8%	2.0	+8%	V
		-3%	2.3	+3%	
		-3%-	2.9	+3%	
LVR Hysteresis Voltage	V_{HYST}	—	± 0.1	—	V
Low Voltage Detection time	t_{LVR}	100	—	—	μs

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

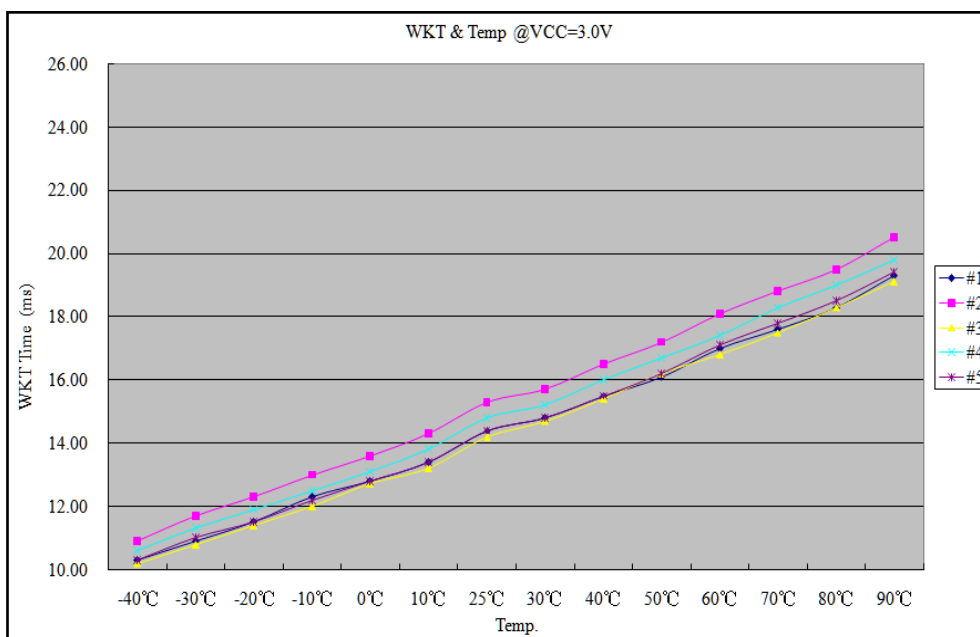
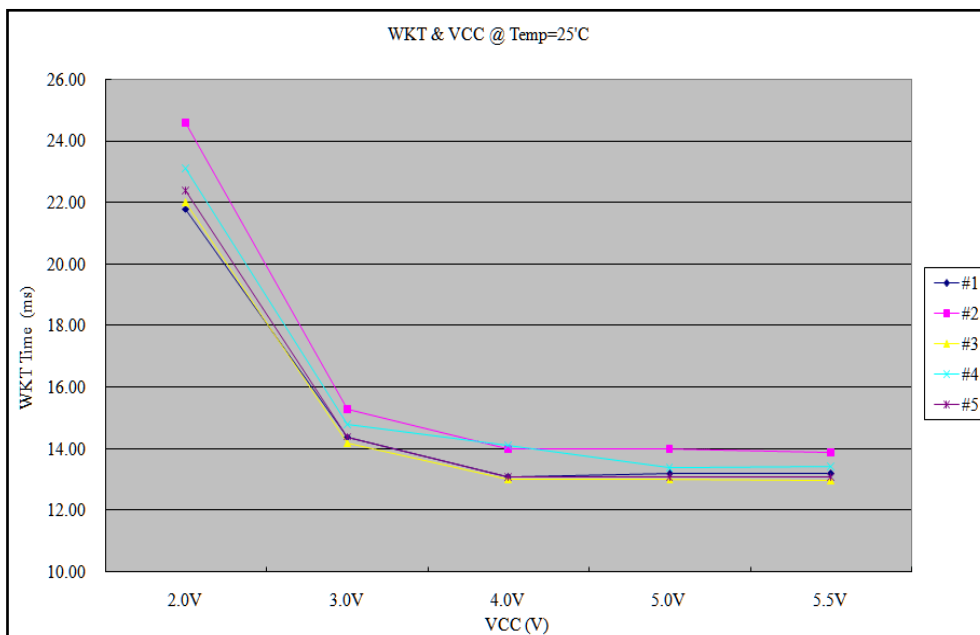
Parameter	Conditions	Min	Typ	Max	Units
Total Accuracy	$V_{CC}=5.12\text{ V}$, $V_{SS}=0\text{ V}$	—	± 2.5	± 4	LSB
Integral Non-Linearity		—	± 3.2	± 5	
Max Input Clock (f_{ADC})	—	—	—	1	MHz
Conversion Time	$f_{ADC}=1\text{ MHz}$	—	50	—	μs
Input Voltage	—	V_{SS}	—	V_{CC}	V

7. Characteristic Graphs





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

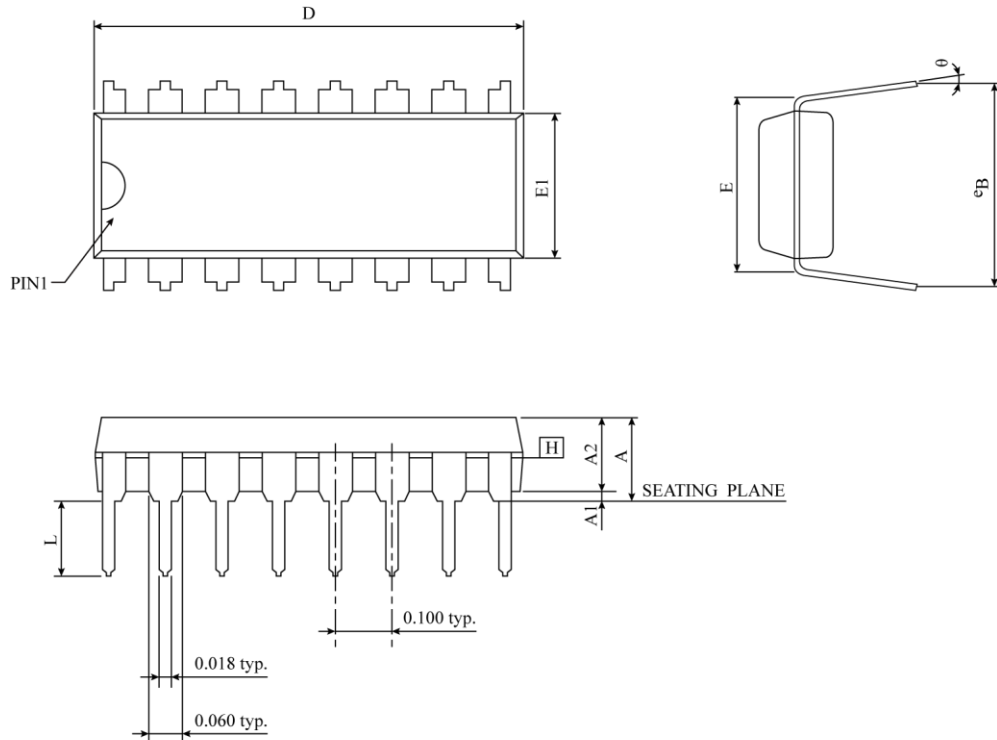


PACKAGING INFORMATION

The ordering information:

Ordering number	Package
TM57MA25-MTP-03	DIP 16-pin (300 mil)
TM57MA25-MTP-16	SOP 16-pin (150 mil)
TM57MA25-MTP-26	SSOP 16-pin (150 mil)
TM57MA25-MTP-01	DIP 8-pin (300 mil)
TM57MA25-MTP-14	SOP 8-pin (150 mil)

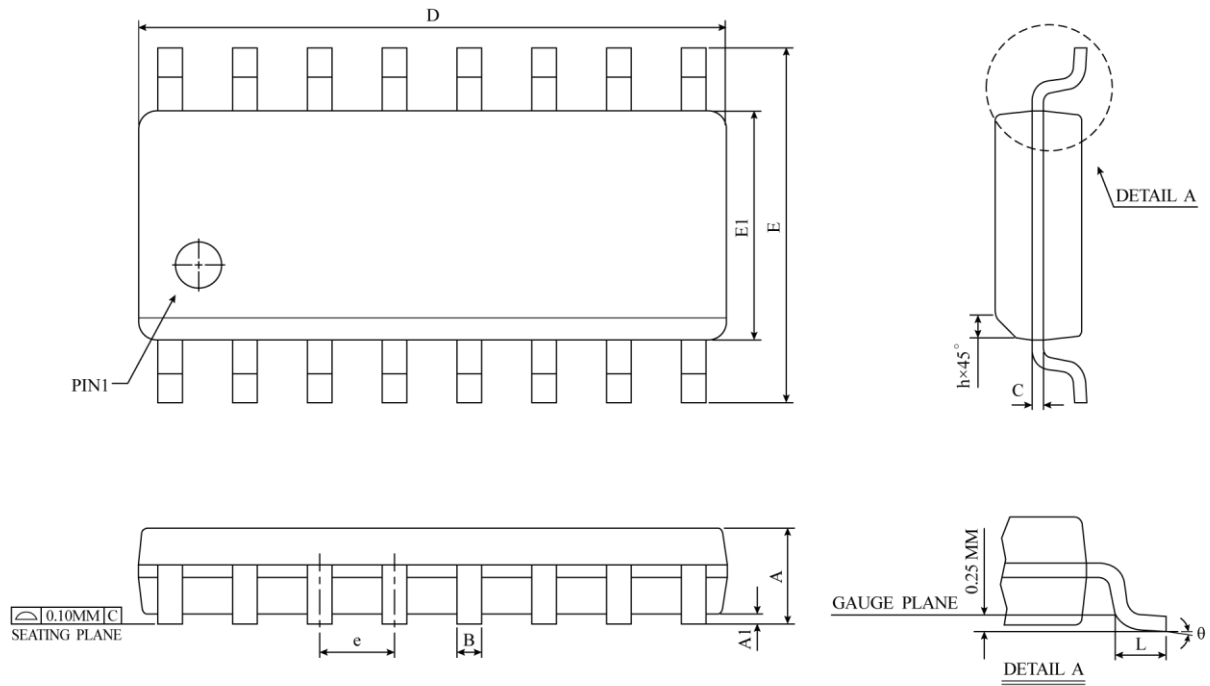
16-DIP (300 mil) Package Dimension



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)					

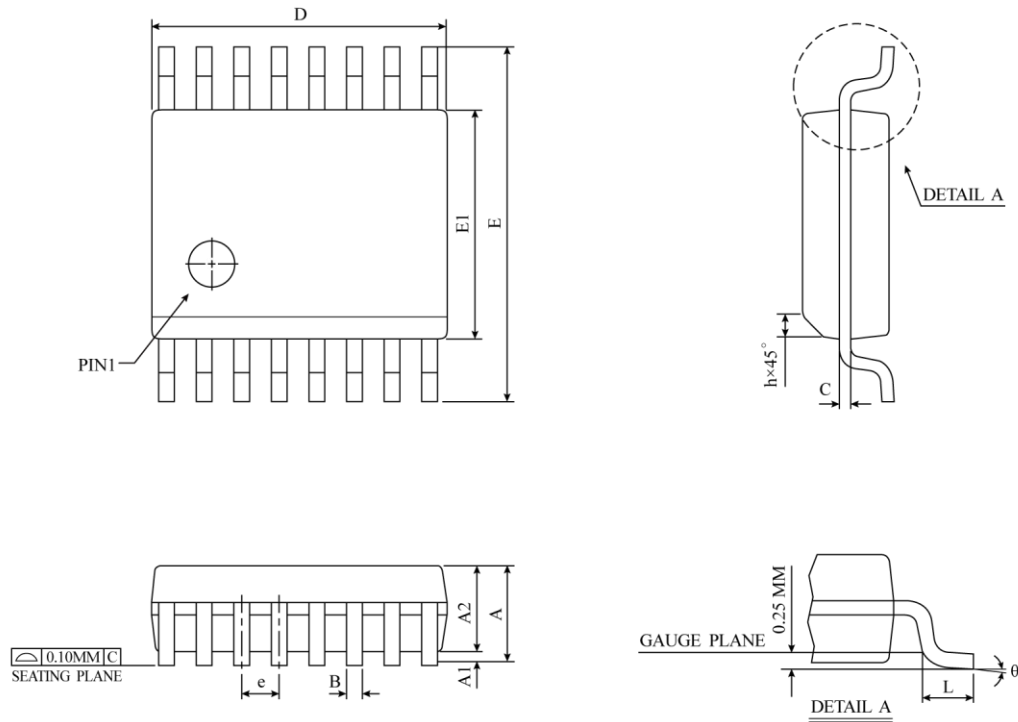
NOTES :

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 (150 mil) Package Dimension


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

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

16-SSOP (150 mil) Package Dimension


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

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