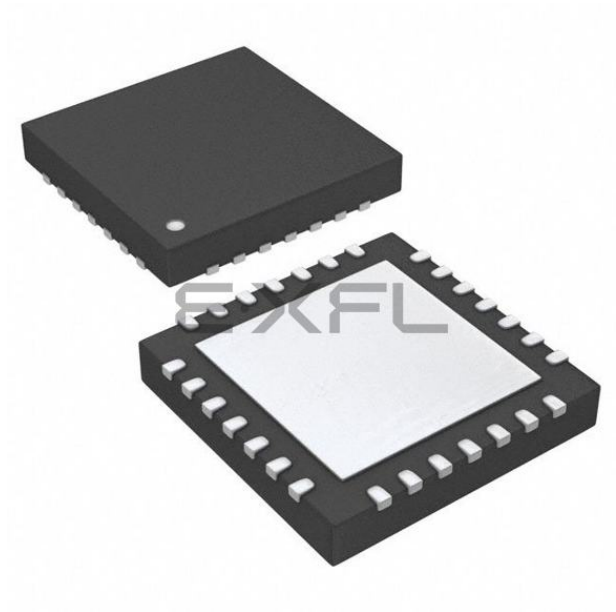


Welcome to E-XFL.COM

What is "Embedded - Microcontrollers"?

"Embedded - Microcontrollers" refer to small, integrated circuits designed to perform specific tasks within larger systems. These microcontrollers are essentially compact computers on a single chip, containing a processor core, memory, and programmable input/output peripherals. They are called "embedded" because they are embedded within electronic devices to control various functions, rather than serving as standalone computers. Microcontrollers are crucial in modern electronics, providing the intelligence and control needed for a wide range of applications.

Applications of "Embedded - Microcontrollers"

Details	
Product Status	Active
Core Processor	PIC
Core Size	8-Bit
Speed	20MHz
Connectivity	-
Peripherals	Brown-out Detect/Reset, LCD, POR, PWM, WDT
Number of I/O	25
Program Memory Size	7KB (4K x 14)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	256 x 8
Voltage - Supply (Vcc/Vdd)	1.8V ~ 3.6V
Data Converters	A/D 11x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	28-UFQFN Exposed Pad
Supplier Device Package	28-UQFN (4x4)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic16lf1903-i-mv

TABLE 1-2: PIC16LF1902/3 PINOUT DESCRIPTION (CONTINUED)

Name	Function	Input Type	Output Type	Description
RB4(1)/AN11/COM0	RB4	TTL	CMOS	General purpose I/O.
	AN11	AN	—	A/D Channel 11 input.
	COM0	—	AN	LCD Analog output.
RB5(1)/AN13/COM1	RB5	TTL	CMOS	General purpose I/O.
	AN13	AN	—	A/D Channel 13 input.
	COM1	—	AN	LCD Analog output.
RB6(1)/ICSPCLK/SEG14	RB6	TTL	CMOS	General purpose I/O.
	ICSPCLK	ST	—	Serial Programming Clock.
	SEG14	—	AN	LCD Analog output.
RB7(1)/ICSPDAT/SEG13	RB7	TTL	CMOS	General purpose I/O.
	ICSPDAT	ST	CMOS	ICSP™ Data I/O.
	SEG13	—	AN	LCD Analog output.
RC0/T1OSO/T1CKI	RC0	TTL	CMOS	General purpose I/O.
	T1OSO	XTAL	XTAL	Timer1 oscillator connection.
	T1CKI	ST	—	Timer1 clock input.
RC1/T1OSI	RC1	TTL	CMOS	General purpose I/O.
	T1OSI	XTAL	XTAL	Timer1 oscillator connection.
RC2/SEG3	RC2	TTL	CMOS	General purpose I/O.
	SEG3	—	AN	LCD Analog output.
RC3/SEG6	RC3	TTL	CMOS	General purpose I/O.
	SEG6	—	AN	LCD Analog output.
RC4/T1G/SEG11	RC4	TTL	CMOS	General purpose I/O.
	T1G	XTAL	XTAL	Timer1 oscillator connection.
	SEG11	—	AN	LCD Analog output.
RC5/SEG10	RC5	TTL	CMOS	General purpose I/O.
	SEG10	—	AN	LCD Analog output.
RC6/SEG9	RC6	ST	CMOS	General purpose I/O.
	SEG9	—	AN	LCD Analog output.
RC7/SEG8	RC7	ST	CMOS	General purpose I/O.
	SEG8	—	AN	LCD Analog output.
RE3/$\overline{MCLR}$/VPP	RE3	TTL	CMOS	General purpose I/O.
	$\overline{MCLR}$	ST	—	Master Clear with internal pull-up.
	VPP	HV	—	Programming voltage.
VDD	VDD	Power	—	Positive supply.
VSS	VSS	Power	—	Ground reference.

Legend: AN = Analog input or output; CMOS = CMOS compatible input or output; OD = Open-Drain
TTL = TTL compatible input; ST = Schmitt Trigger input with CMOS levels; I^2C = Schmitt Trigger input with I^2C levels
HV = High Voltage; XTAL = Crystal

Note 1: These pins have interrupt-on-change functionality.

3.2.6 CORE FUNCTION REGISTERS SUMMARY

The Core Function registers listed in Table 3-4 can be addressed from any Bank.

TABLE 3-4: CORE FUNCTION REGISTERS SUMMARY

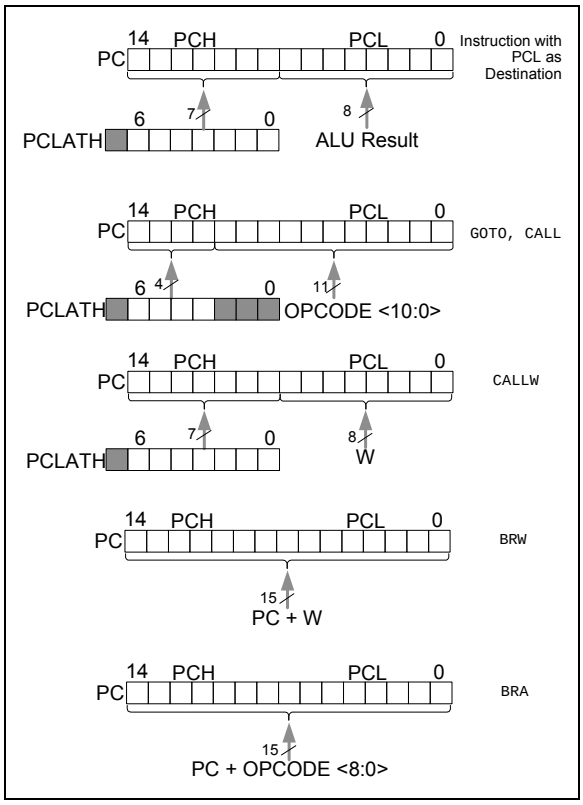
Addr	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR, BOR	Value on all other Resets
Bank 0-31											
x00h or x80h	INDF0	Addressing this location uses contents of FSR0H/FSR0L to address data memory (not a physical register)								xxxx xxxx	uuuu uuuu
x01h or x81h	INDF1	Addressing this location uses contents of FSR1H/FSR1L to address data memory (not a physical register)								xxxx xxxx	uuuu uuuu
x02h or x82h	PCL	Program Counter (PC) Least Significant Byte								0000 0000	0000 0000
x03h or x83h	STATUS	—	—	—	$\overline{TO}$	$\overline{PD}$	Z	DC	C	---1 1000	---q quuu
x04h or x84h	FSR0L	Indirect Data Memory Address 0 Low Pointer								0000 0000	uuuu uuuu
x05h or x85h	FSR0H	Indirect Data Memory Address 0 High Pointer								0000 0000	0000 0000
x06h or x86h	FSR1L	Indirect Data Memory Address 1 Low Pointer								0000 0000	uuuu uuuu
x07h or x87h	FSR1H	Indirect Data Memory Address 1 High Pointer								0000 0000	0000 0000
x08h or x88h	BSR	—	—	—	BSR4	BSR3	BSR2	BSR1	BSR0	---0 0000	---0 0000
x09h or x89h	WREG	Working Register								0000 0000	uuuu uuuu
x0Ah or x8Ah	PCLATH	—	Write Buffer for the upper 7 bits of the Program Counter							-000 0000	-000 0000
x0Bh or x8Bh	INTCON	GIE	PEIE	TMR0IE	INTE	IOCIE	TMR0IF	INTF	IOCIF	0000 0000	0000 0000

Legend: x = unknown, u = unchanged, q = value depends on condition, - = unimplemented, read as '0', r = reserved. Shaded locations are unimplemented, read as '0'.

3.3 PCL and PCLATH

The Program Counter (PC) is 15 bits wide. The low byte comes from the PCL register, which is a readable and writable register. The high byte (PC<14:8>) is not directly readable or writable and comes from PCLATH. On any Reset, the PC is cleared. Figure 3-4 shows the five situations for the loading of the PC.

FIGURE 3-4: LOADING OF PC IN DIFFERENT SITUATIONS



3.3.1 MODIFYING PCL

Executing any instruction with the PCL register as the destination simultaneously causes the Program Counter PC<14:8> bits (PCH) to be replaced by the contents of the PCLATH register. This allows the entire contents of the program counter to be changed by writing the desired upper seven bits to the PCLATH register. When the lower eight bits are written to the PCL register, all 15 bits of the program counter will change to the values contained in the PCLATH register and those being written to the PCL register.

3.3.2 COMPUTED GOTO

A computed GOTO is accomplished by adding an offset to the program counter (ADDWF PCL). When performing a table read using a computed GOTO method, care should be exercised if the table location crosses a PCL memory boundary (each 256-byte block). Refer to the Application Note AN556, *"Implementing a Table Read"* (DS00556).

3.3.3 COMPUTED FUNCTION CALLS

A computed function CALL allows programs to maintain tables of functions and provide another way to execute state machines or look-up tables. When performing a table read using a computed function CALL, care should be exercised if the table location crosses a PCL memory boundary (each 256-byte block).

If using the CALL instruction, the PCH<2:0> and PCL registers are loaded with the operand of the CALL instruction. PCH<6:3> is loaded with PCLATH<6:3>.

The CALLW instruction enables computed calls by combining PCLATH and W to form the destination address. A computed CALLW is accomplished by loading the W register with the desired address and executing CALLW. The PCL register is loaded with the value of W and PCH is loaded with PCLATH.

3.3.4 BRANCHING

The branching instructions add an offset to the PC. This allows relocatable code and code that crosses page boundaries. There are two forms of branching, BRW and BRA. The PC will have incremented to fetch the next instruction in both cases. When using either branching instruction, a PCL memory boundary may be crossed.

If using BRW, load the W register with the desired unsigned address and execute BRW. The entire PC will be loaded with the address PC + 1 + W.

If using BRA, the entire PC will be loaded with PC + 1 +, the signed value of the operand of the BRA instruction.

4.2 Code Protection

Code protection allows the device to be protected from unauthorized access. Program memory protection is controlled independently. Internal access to the program memory is unaffected by any code protection setting.

4.2.1 PROGRAM MEMORY PROTECTION

The entire program memory space is protected from external reads and writes by the $\overline{CP}$ bit in Configuration Word 1. When $\overline{CP}$ = 0, external reads and writes of program memory are inhibited and a read will return all '0's. The CPU can continue to read program memory, regardless of the protection bit settings. Writing the program memory is dependent upon the write protection setting. See **Section 4.3 "Write Protection"** for more information.

4.3 Write Protection

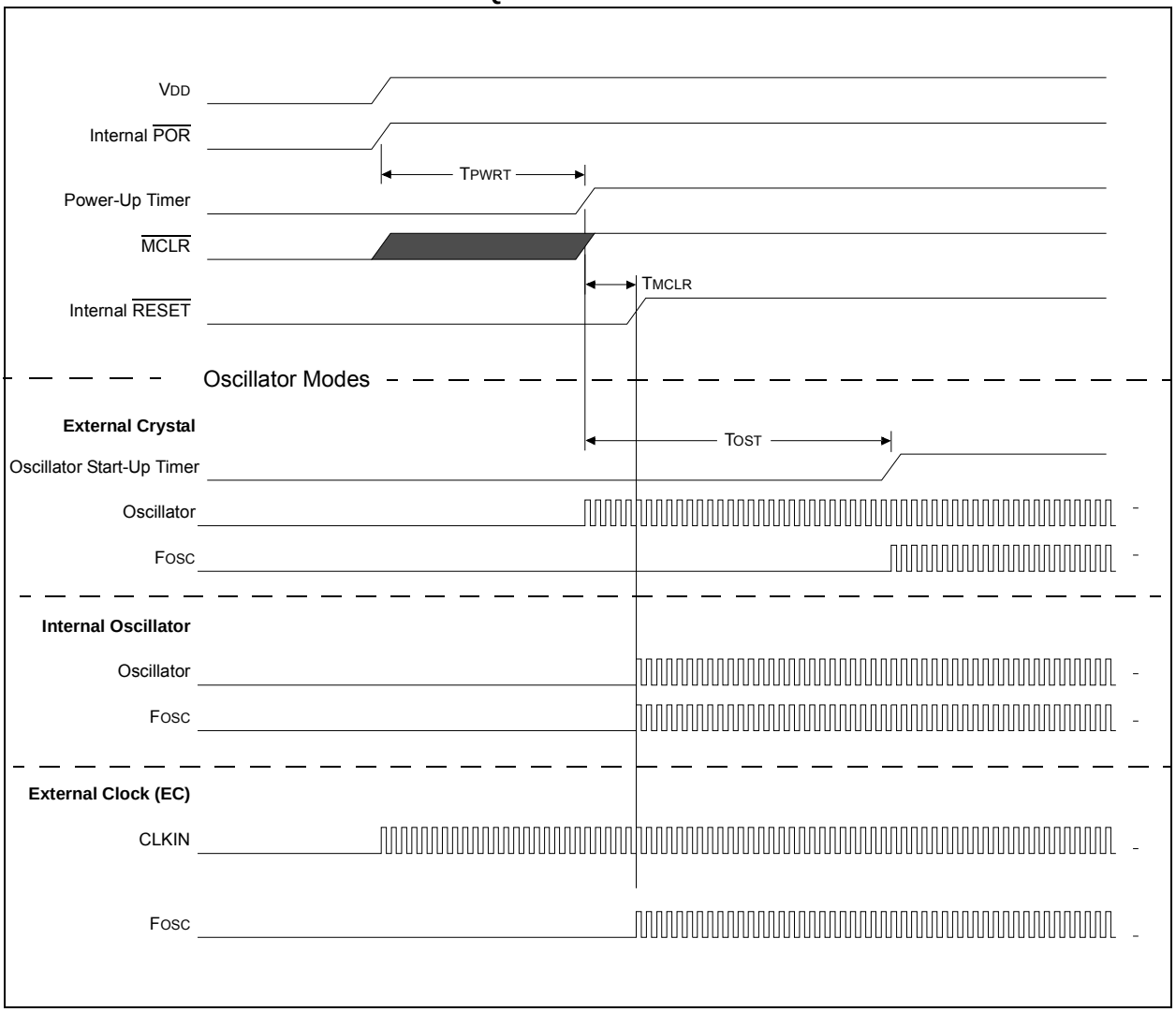
Write protection allows the device to be protected from unintended self-writes. Applications, such as boot loader software, can be protected while allowing other regions of the program memory to be modified.

The WRT<1:0> bits in Configuration Word 2 define the size of the program memory block that is protected.

4.4 User ID

Four memory locations (8000h-8003h) are designated as ID locations where the user can store checksum or other code identification numbers. These locations are readable and writable during normal execution. See **Section 10.4 "User ID, Device ID and Configuration Word Access"** for more information on accessing these memory locations. For more information on checksum calculation, see the **"***PIC16F193X/LF193X/PIC16F194X/LF194X/PIC16LF190X Memory Programming Specification***"** (DS41397).

FIGURE 5-3: RESET START-UP SEQUENCE



5.11 Determining the Cause of a Reset

Upon any Reset, multiple bits in the STATUS and PCON register are updated to indicate the cause of the Reset. Table 5-3 and Table 5-4 show the Reset conditions of these registers.

TABLE 5-3: RESET STATUS BITS AND THEIR SIGNIFICANCE

STKOVF	STKUNF	$\overline{\text{RWDT}}$	$\overline{\text{RMCLR}}$	$\overline{\text{RI}}$	$\overline{\text{POR}}$	$\overline{\text{BOR}}$	$\overline{\text{TO}}$	$\overline{\text{PD}}$	Condition
0	0	1	1	1	0	x	1	1	Power-on Reset
0	0	1	1	1	0	x	0	x	Illegal, $\overline{\text{TO}}$ is set on $\overline{\text{POR}}$
0	0	1	1	1	0	x	x	0	Illegal, $\overline{\text{PD}}$ is set on $\overline{\text{POR}}$
0	0	u	1	1	u	0	1	1	Brown-out Reset
u	u	0	u	u	u	u	0	u	WDT Reset
u	u	u	u	u	u	u	0	0	WDT Wake-up from Sleep
u	u	u	u	u	u	u	1	0	Interrupt Wake-up from Sleep
u	u	u	0	u	u	u	u	u	$\overline{\text{MCLR}}$ Reset during normal operation
u	u	u	0	u	u	u	1	0	$\overline{\text{MCLR}}$ Reset during Sleep
u	u	u	u	0	u	u	u	u	RESET Instruction Executed
1	u	u	u	u	u	u	u	u	Stack Overflow Reset (STVREN = 1)
u	1	u	u	u	u	u	u	u	Stack Underflow Reset (STVREN = 1)

TABLE 5-4: RESET CONDITION FOR SPECIAL REGISTERS(2)

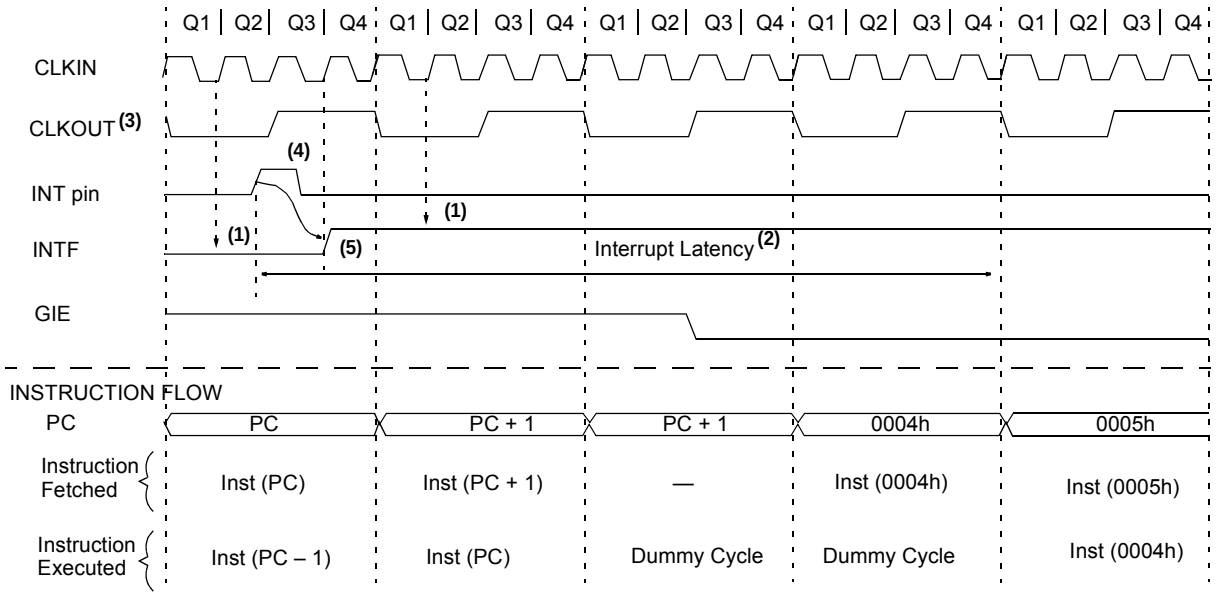
Condition	Program Counter	STATUS Register	PCON Register
Power-on Reset	0000h	---1 1000	00-1 110x
$\overline{\text{MCLR}}$ Reset during normal operation	0000h	---u uuuu	uu-u 0uuu
$\overline{\text{MCLR}}$ Reset during Sleep	0000h	---1 0uuu	uu-u 0uuu
WDT Reset	0000h	---0 uuuu	uu-0 uuuu
WDT Wake-up from Sleep	PC + 1	---0 0uuu	uu-u uuuu
Brown-out Reset	0000h	---1 1uuu	00-1 11u0
Interrupt Wake-up from Sleep	PC + 1(1)	---1 0uuu	uu-u uuuu
RESET Instruction Executed	0000h	---u uuuu	uu-u u0uu
Stack Overflow Reset (STVREN = 1)	0000h	---u uuuu	1u-u uuuu
Stack Underflow Reset (STVREN = 1)	0000h	---u uuuu	u1-u uuuu

Legend: u = unchanged, x = unknown, - = unimplemented bit, reads as '0'.

Note 1: When the wake-up is due to an interrupt and Global Enable bit (GIE) is set, the return address is pushed on the stack and PC is loaded with the interrupt vector (0004h) after execution of PC + 1.

2: If a Status bit is not implemented, that bit will be read as '0'.

FIGURE 7-3: INT PIN INTERRUPT TIMING



Note 1: INTF flag is sampled here (every Q1).

2: Asynchronous interrupt latency = 3-5 TCY. Synchronous latency = 3-4 TCY, where TCY = instruction cycle time. Latency is the same whether Inst (PC) is a single cycle or a 2-cycle instruction.

3: CLKOUT not available in all Oscillator modes.

4: For minimum width of INT pulse, refer to AC specifications in **Section 21.0 "Electrical Specifications"**.

5: INTF is enabled to be set any time during the Q4-Q1 cycles.

9.6 Watchdog Control Register

REGISTER 9-1: WDTCON: WATCHDOG TIMER CONTROL REGISTER

U-0	U-0	R/W-0/0	R/W-1/1	R/W-0/0	R/W-1/1	R/W-1/1	R/W-0/0
—	—	WDTPS<4:0>					SWDTEN
bit 7							bit 0

Legend:		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'
u = Bit is unchanged	x = Bit is unknown	-m/n = Value at POR and BOR/Value at all other Resets
'1' = Bit is set	'0' = Bit is cleared	

bit 7-6 **Unimplemented:** Read as '0'

bit 5-1 **WDTPS<4:0>:** Watchdog Timer Period Select bits(1)

Bit Value = Prescale Rate

00000 = 1:32 (Interval 1 ms nominal)
00001 = 1:64 (Interval 2 ms nominal)
00010 = 1:128 (Interval 4 ms nominal)
00011 = 1:256 (Interval 8 ms nominal)
00100 = 1:512 (Interval 16 ms nominal)
00101 = 1:1024 (Interval 32 ms nominal)
00110 = 1:2048 (Interval 64 ms nominal)
00111 = 1:4096 (Interval 128 ms nominal)
01000 = 1:8192 (Interval 256 ms nominal)
01001 = 1:16384 (Interval 512 ms nominal)
01010 = 1:32768 (Interval 1s nominal)
01011 = 1:65536 (Interval 2s nominal) (Reset value)
01100 = 1:131072 (2^{17}) (Interval 4s nominal)
01101 = 1:262144 (2^{18}) (Interval 8s nominal)
01110 = 1:524288 (2^{19}) (Interval 16s nominal)
01111 = 1:1048576 (2^{20}) (Interval 32s nominal)
10000 = 1:2097152 (2^{21}) (Interval 64s nominal)
10001 = 1:4194304 (2^{22}) (Interval 128s nominal)
10010 = 1:8388608 (2^{23}) (Interval 256s nominal)

10011 = Reserved. Results in minimum interval (1:32)
•
•
•
11111 = Reserved. Results in minimum interval (1:32)

bit 0 **SWDTEN:** Software Enable/Disable for Watchdog Timer bit

If WDTE<1:0> = 00:
This bit is ignored.
If WDTE<1:0> = 01:
1 = WDT is turned on
0 = WDT is turned off
If WDTE<1:0> = 1x:
This bit is ignored.

Note 1: Times are approximate. WDT time is based on 31 kHz LFINTOSC.

10.2.2 FLASH MEMORY UNLOCK SEQUENCE

The unlock sequence is a mechanism that protects the Flash Program Memory from unintended self-write programming or erasing. The sequence must be executed and completed without interruption to successfully complete any of the following operations:

- Row Erase
- Load program memory write latches
- Write of program memory write latches to program memory
- Write of program memory write latches to User IDs

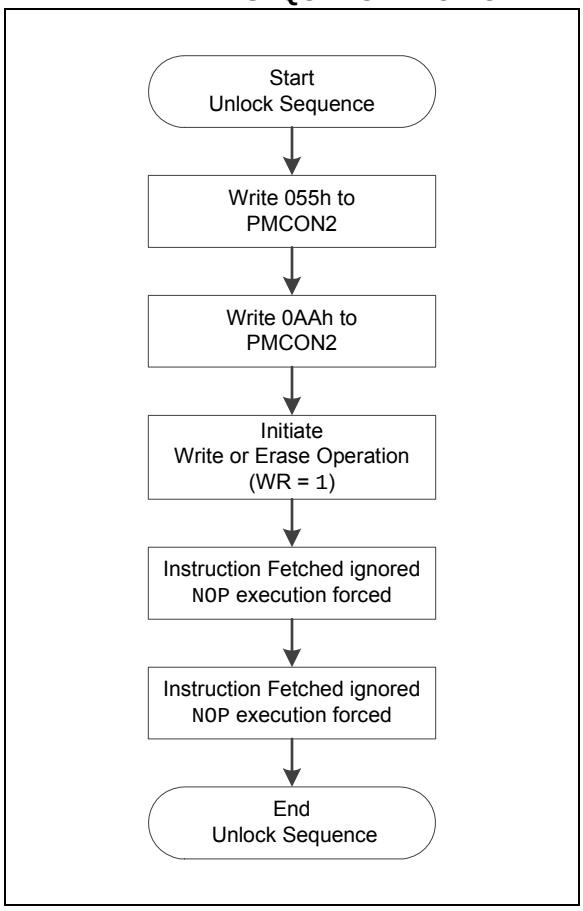
The unlock sequence consists of the following steps:

1. Write 55h to PMCON2
2. Write AAh to PMCON2
3. Set the WR bit in PMCON1
4. NOP instruction
5. NOP instruction

Once the WR bit is set, the processor will always force two NOP instructions. When an Erase Row or Program Row operation is being performed, the processor will stall internal operations (typical 2 ms), until the operation is complete and then resume with the next instruction. When the operation is loading the program memory write latches, the processor will always force the two NOP instructions and continue uninterrupted with the next instruction.

Since the unlock sequence must not be interrupted, global interrupts should be disabled prior to the unlock sequence and re-enabled after the unlock sequence is completed.

FIGURE 10-3: FLASH PROGRAM MEMORY UNLOCK SEQUENCE FLOWCHART



15.2.5 A/D CONVERSION PROCEDURE

This is an example procedure for using the ADC to perform an Analog-to-Digital conversion:

1. Configure Port:
   - Disable pin output driver (Refer to the TRIS register)
   - Configure pin as analog (Refer to the ANSEL register)
2. Configure the ADC module:
   - Select ADC conversion clock
   - Configure voltage reference
   - Select ADC input channel
   - Turn on ADC module
3. Configure ADC interrupt (optional):
   - Clear ADC interrupt flag
   - Enable ADC interrupt
   - Enable peripheral interrupt
   - Enable global interrupt**(1)**
4. Wait the required acquisition time**(2)**.
5. Start conversion by setting the GO/$\overline{\text{DONE}}$ bit.
6. Wait for ADC conversion to complete by one of the following:
   - Polling the GO/$\overline{\text{DONE}}$ bit
   - Waiting for the ADC interrupt (interrupts enabled)
7. Read ADC Result.
8. Clear the ADC interrupt flag (required if interrupt is enabled).

Note 1: The global interrupt can be disabled if the user is attempting to wake-up from Sleep and resume in-line code execution.

2: Refer to **Section 15.3 "A/D Acquisition Requirements"**.

EXAMPLE 15-1: A/D CONVERSION

```
;This code block configures the ADC
;for polling, Vdd and Vss references, Frc
;clock and AN0 input.
;
;Conversion start & polling for completion
; are included.
;
BANKSEL   ADCON1       ;
MOVLW     B'11110000'  ;Right justify, Frc
                       ;clock
MOVWF     ADCON1       ;Vdd and Vss Vref
BANKSEL   TRISA        ;
BSF       TRISA,0      ;Set RA0 to input
BANKSEL   ANSEL        ;
BSF       ANSEL,0      ;Set RA0 to analog
BANKSEL   ADCON0       ;
MOVLW     B'00000001'  ;Select channel AN0
MOVWF     ADCON0       ;Turn ADC On
CALL      SampleTime   ;Acquisiton delay
BSF       ADCON0,ADGO  ;Start conversion
BTFSC     ADCON0,ADGO  ;Is conversion done?
GOTO      $-1          ;No, test again
BANKSEL   ADRESH       ;
MOVF      ADRESH,W     ;Read upper 2 bits
MOVWF     RESULTHI     ;store in GPR space
BANKSEL   ADRESL       ;
MOVF      ADRESL,W     ;Read lower 8 bits
MOVWF     RESULTLO     ;Store in GPR space
```

16.2 Option and Timer0 Control Register

REGISTER 16-1: OPTION_REG: OPTION REGISTER

R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1
$\overline{\text{WPUEN}}$	INTEDG	TMR0CS	TMR0SE	PSA	PS<2:0>		
bit 7							bit 0

Legend:		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'
u = Bit is unchanged	x = Bit is unknown	-n/n = Value at POR and BOR/Value at all other Resets
'1' = Bit is set	'0' = Bit is cleared	

bit 7 **$\overline{\text{WPUEN}}$:** Weak Pull-up Enable bit
1 = All weak pull-ups are disabled (except $\overline{\text{MCLR}}$, if it is enabled)
0 = Weak pull-ups are enabled by individual WPUx latch values

bit 6 **INTEDG:** Interrupt Edge Select bit
1 = Interrupt on rising edge of INT pin
0 = Interrupt on falling edge of INT pin

bit 5 **TMR0CS:** Timer0 Clock Source Select bit
1 = Transition on T0CKI pin
0 = Internal instruction cycle clock (FOSC/4)

bit 4 **TMR0SE:** Timer0 Source Edge Select bit
1 = Increment on high-to-low transition on T0CKI pin
0 = Increment on low-to-high transition on T0CKI pin

bit 3 **PSA:** Prescaler Assignment bit
1 = Prescaler is not assigned to the Timer0 module
0 = Prescaler is assigned to the Timer0 module

bit 2-0 **PS<2:0>:** Prescaler Rate Select bits

Bit Value	Timer0 Rate
000	1 : 2
001	1 : 4
010	1 : 8
011	1 : 16
100	1 : 32
101	1 : 64
110	1 : 128
111	1 : 256

TABLE 16-1: SUMMARY OF REGISTERS ASSOCIATED WITH TIMER0

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Register on Page
INTCON	GIE	PEIE	TMR0IE	INTE	IOCIE	TMR0IF	INTF	IOCIF	60
OPTION_REG	$\overline{\text{WPUEN}}$	INTEDG	TMR0CS	TMR0SE	PSA	PS<2:0>			121
TMR0	Timer0 Module Register								119*
TRISA	TRISA7	TRISA6	TRISA5	TRISA4	TRISA3	TRISA2	TRISA1	TRISA0	90

Legend: — = Unimplemented location, read as '0'. Shaded cells are not used by the Timer0 module.
* Page provides register information.

REGISTER 18-7: LCDRL: LCD REFERENCE LADDER CONTROL REGISTERS

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	U-0	R/W-0/0	R/W-0/0	R/W-0/0
LRLAP<1:0>		LRLBP<1:0>		—	LRLAT<2:0>		
bit 7							bit 0

Legend:		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’
u = Bit is unchanged	x = Bit is unknown	-n/n = Value at POR and BOR/Value at all other Resets
‘1’ = Bit is set	‘0’ = Bit is cleared	

bit 7-6 **LRLAP<1:0>:** LCD Reference Ladder A Time Power Control bits
During Time interval A (Refer toFigure 18-4):
00 = Internal LCD Reference Ladder is powered down and unconnected
01 = Internal LCD Reference Ladder is powered in Low-Power mode
10 = Internal LCD Reference Ladder is powered in Medium-Power mode
11 = Internal LCD Reference Ladder is powered in High-Power mode

bit 5-4 **LRLBP<1:0>:** LCD Reference Ladder B Time Power Control bits
During Time interval B (Refer to Figure 18-4):
00 = Internal LCD Reference Ladder is powered down and unconnected
01 = Internal LCD Reference Ladder is powered in Low-Power mode
10 = Internal LCD Reference Ladder is powered in Medium-Power mode
11 = Internal LCD Reference Ladder is powered in High-Power mode

bit 3 **Unimplemented:** Read as ‘0’

bit 2-0 **LRLAT<2:0>:** LCD Reference Ladder A Time Interval Control bits
Sets the number of 32 kHz clocks that the A Time Interval Power mode is active

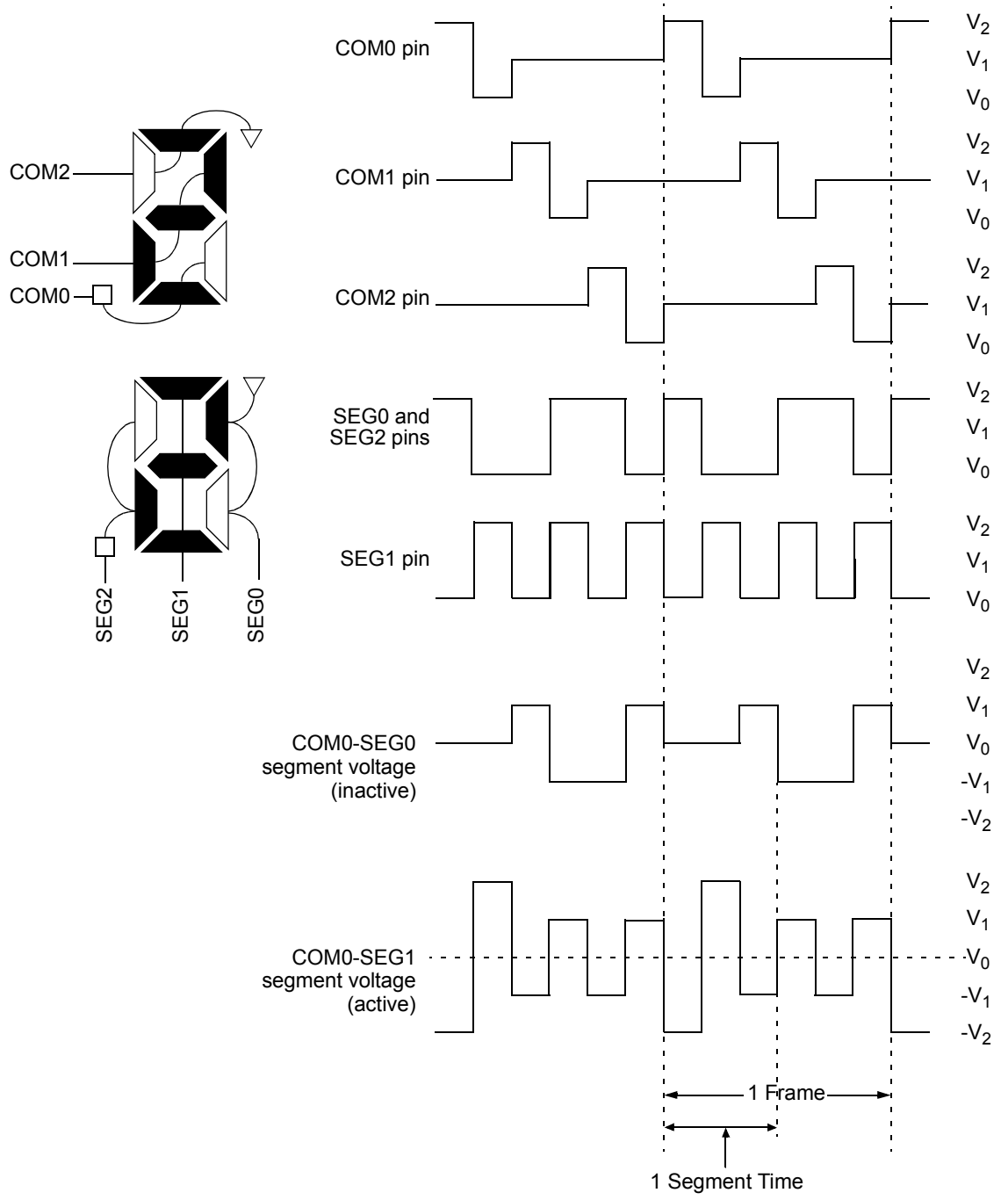
For type A waveforms (WFT = 0):

000 = Internal LCD Reference Ladder is always in ‘B’ Power mode
001 = Internal LCD Reference Ladder is in ‘A’ Power mode for 1 clock and ‘B’ Power mode for 15 clocks
010 = Internal LCD Reference Ladder is in ‘A’ Power mode for 2 clocks and ‘B’ Power mode for 14 clocks
011 = Internal LCD Reference Ladder is in ‘A’ Power mode for 3 clocks and ‘B’ Power mode for 13 clocks
100 = Internal LCD Reference Ladder is in ‘A’ Power mode for 4 clocks and ‘B’ Power mode for 12 clocks
101 = Internal LCD Reference Ladder is in ‘A’ Power mode for 5 clocks and ‘B’ Power mode for 11 clocks
110 = Internal LCD Reference Ladder is in ‘A’ Power mode for 6 clocks and ‘B’ Power mode for 10 clocks
111 = Internal LCD Reference Ladder is in ‘A’ Power mode for 7 clocks and ‘B’ Power mode for 9 clocks

For type B waveforms (WFT = 1):

000 = Internal LCD Reference Ladder is always in ‘B’ Power mode.
001 = Internal LCD Reference Ladder is in ‘A’ Power mode for 1 clock and ‘B’ Power mode for 31 clocks
010 = Internal LCD Reference Ladder is in ‘A’ Power mode for 2 clocks and ‘B’ Power mode for 30 clocks
011 = Internal LCD Reference Ladder is in ‘A’ Power mode for 3 clocks and ‘B’ Power mode for 29 clocks
100 = Internal LCD Reference Ladder is in ‘A’ Power mode for 4 clocks and ‘B’ Power mode for 28 clocks
101 = Internal LCD Reference Ladder is in ‘A’ Power mode for 5 clocks and ‘B’ Power mode for 27 clocks
110 = Internal LCD Reference Ladder is in ‘A’ Power mode for 6 clocks and ‘B’ Power mode for 26 clocks
111 = Internal LCD Reference Ladder is in ‘A’ Power mode for 7 clocks and ‘B’ Power mode for 25 clocks

FIGURE 18-13: TYPE-A WAVEFORMS IN 1/3 MUX, 1/2 BIAS DRIVE



Note: 1 Frame = 2 single segment times.

18.11 Operation During Sleep

The LCD module can operate during Sleep. The selection is controlled by bit SLPEN of the LCDCON register. Setting the SLPEN bit allows the LCD module to go to Sleep. Clearing the SLPEN bit allows the module to continue to operate during Sleep.

If a SLEEP instruction is executed and SLPEN = 1, the LCD module will cease all functions and go into a very low-current Consumption mode. The module will stop operation immediately and drive the minimum LCD voltage on both segment and common lines. Figure 18-20 shows this operation.

The LCD module can be configured to operate during Sleep. The selection is controlled by bit SLPEN of the LCDCON register. Clearing SLPEN and correctly configuring the LCD module clock will allow the LCD module to operate during Sleep. Setting SLPEN and correctly executing the LCD module shutdown will disable the LCD module during Sleep and save power.

If a SLEEP instruction is executed and SLPEN = 1, the LCD module will immediately cease all functions, drive the outputs to Vss and go into a very low-current mode. The SLEEP instruction should only be executed after the LCD module has been disabled and the current cycle completed, thus ensuring that there are no DC voltages on the glass. To disable the LCD module, clear the LCDEN bit. The LCD module will complete the disabling process after the current frame, clear the LCDA bit and optionally cause an interrupt.

The steps required to properly enter Sleep with the LCD disabled are:

- Clear LCDEN
- Wait for LCDA = 0 either by polling or by interrupt
- Execute SLEEP

If SLPEN = 0 and SLEEP is executed while the LCD module clock source is FOSC/4, then the LCD module will halt with the pin driving the last LCD voltage pattern. Prolonged exposure to a fixed LCD voltage pattern will cause damage to the LCD glass. To prevent LCD glass damage, either perform the proper LCD module shutdown prior to Sleep, or change the LCD module clock to allow the LCD module to continue operation during Sleep.

If a SLEEP instruction is executed and SLPEN = 0 and the LCD module clock is either T1OSC or LFINTOSC, the module will continue to display the current contents of the LCDDATA registers. While in Sleep, the LCD data cannot be changed. If the LCDIE bit is set, the device will wake from Sleep on the next LCD frame boundary. The LCD module current consumption will not decrease in this mode; however, the overall device power consumption will be lower due to the shutdown of the CPU and other peripherals.

Table 18-7 shows the status of the LCD module during a Sleep while using each of the three available clock sources.

Note:	When the LCDEN bit is cleared, the LCD module will be disabled at the completion of frame. At this time, the port pins will revert to digital functionality. To minimize power consumption due to floating digital inputs, the LCD pins should be driven low using the PORT and TRIS registers.

If a SLEEP instruction is executed and SLPEN = 0, the module will continue to display the current contents of the LCDDATA registers. To allow the module to continue operation while in Sleep, the clock source must be either the LFINTOSC or T1OSC external oscillator. While in Sleep, the LCD data cannot be changed. The LCD module current consumption will not decrease in this mode; however, the overall consumption of the device will be lower due to shut down of the core and other peripheral functions.

Table 18-7 shows the status of the LCD module during Sleep while using each of the three available clock sources:

TABLE 18-7: LCD MODULE STATUS DURING SLEEP

Clock Source	SLPEN	Operational During Sleep
T1OSC	0	Yes
	1	No
LFINTOSC	0	Yes
	1	No
FOSC/4	0	No
	1	No

Note:	The LFINTOSC or external T1OSC oscillator must be used to operate the LCD module during Sleep.

If LCD interrupts are being generated (Type-B waveform with a multiplex mode not static) and LCDIE = 1, the device will awaken from Sleep on the next frame boundary.

DECFSZ **Decrement f, Skip if 0**

Syntax:	[*label*] DECFSZ f,d
Operands:	$0 \le f \le 127$ $d \in [0,1]$
Operation:	(f) - 1 → (destination); skip if result = 0
Status Affected:	None
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.

GOTO **Unconditional Branch**

Syntax:	[*label*] GOTO k
Operands:	$0 \le k \le 2047$
Operation:	k → PC<10:0> PCLATH<6:3> → PC<14:11>
Status Affected:	None
Description:	GOTO is an unconditional branch. The 11-bit immediate value is loaded into PC bits <10:0>. The upper bits of PC are loaded from PCLATH<4:3>. GOTO is a 2-cycle instruction.

INCF **Increment f**

Syntax:	[*label*] INCF f,d
Operands:	$0 \le f \le 127$ $d \in [0,1]$
Operation:	(f) + 1 → (destination)
Status Affected:	Z
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'.

INCFSZ **Increment f, Skip if 0**

Syntax:	[*label*] INCFSZ f,d
Operands:	$0 \le f \le 127$ $d \in [0,1]$
Operation:	(f) + 1 → (destination), skip if result = 0
Status Affected:	None
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.

IORLW **Inclusive OR literal with W**

Syntax:	[*label*] IORLW k
Operands:	$0 \le k \le 255$
Operation:	(W) .OR. k → (W)
Status Affected:	Z
Description:	The contents of the W register are OR'ed with the 8-bit literal 'k'. The result is placed in the W register.

IORWF **Inclusive OR W with f**

Syntax:	[*label*] IORWF f,d
Operands:	$0 \le f \le 127$ $d \in [0,1]$
Operation:	(W) .OR. (f) → (destination)
Status Affected:	Z
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'.

RETFIE **Return from Interrupt**

Syntax: [*label*] RETFIE k

Operands: None

Operation: TOS → PC,
1 → GIE

Status Affected: None

Description: Return from Interrupt. Stack is POPed and Top-of-Stack (TOS) is loaded in the PC. Interrupts are enabled by setting Global Interrupt Enable bit, GIE (INTCON<7>). This is a 2-cycle instruction.

Words: 1

Cycles: 2

Example:

```
RETFIE
```

After Interrupt
PC = TOS
GIE = 1

RETLW **Return with literal in W**

Syntax: [*label*] RETLW k

Operands: 0 ≤ k ≤ 255

Operation: k → (W);
TOS → PC

Status Affected: None

Description: The W register is loaded with the 8-bit literal 'k'. The program counter is loaded from the top of the stack (the return address). This is a 2-cycle instruction.

Words: 1

Cycles: 2

Example:

```
CALL TABLE;W contains table
    ;offset value
•   ;W now has table value
•
•
ADDWF PC  ;W = offset
RETLW k1  ;Begin table
RETLW k2  ;
•
•
•
RETLW kn  ; End of table
```

Before Instruction
W = 0x07
After Instruction
W = value of k8

RETURN **Return from Subroutine**

Syntax: [*label*] RETURN

Operands: None

Operation: TOS → PC

Status Affected: None

Description: Return from subroutine. The stack is POPed and the top of the stack (TOS) is loaded into the program counter. This is a 2-cycle instruction.

RLF **Rotate Left f through Carry**

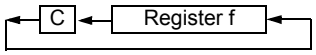
Syntax: [*label*] RLF f,d

Operands: 0 ≤ f ≤ 127
d ∈ [0,1]

Operation: See description below

Status Affected: C

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'.



Words: 1

Cycles: 1

Example:

```
RLF     REG1,0
```

Before Instruction
REG1 = 1110 0110
C = 0
After Instruction
REG1 = 1110 0110
W = 1100 1100
C = 1

TABLE 21-12: PIC16LF1902/3 CONVERTER (ADC) CHARACTERISTICS(1,2,3)

Operating Conditions (unless otherwise stated) VDD = 3.0V, TA = 25°C							
Param. No.	**Sym.**	**Characteristic**	**Min.**	**Typ†**	**Max.**	**Units**	**Conditions**
AD01	NR	Resolution	—	—	10	bit	
AD02	EIL	Integral Error	—	±1	±1.7	LSb	VREF = 3.0V
AD03	EDL	Differential Error	—	±1	±1	LSb	No missing codes VREF = 3.0V
AD04	EOFF	Offset Error	—	±1	±2.5	LSb	VREF = 3.0V
AD05	EGN	Gain Error	—	±1	±2.0	LSb	VREF = 3.0V
AD06	VREF	Reference Voltage	1.8	—	VDD	V	VREF = (VRPOS - VRNEG)
AD07	VAIN	Full-Scale Range	VSS	—	VREF	V	
AD08	ZAIN	Recommended Impedance of Analog Voltage Source	—	—	10	kΩ	Can go higher if external 0.01μF capacitor is present on input pin.

* These parameters are characterized but not tested.

† Data in "Typ" column is at 3.0V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

Note 1: Total Absolute Error includes integral, differential, offset and gain errors.

2: The ADC conversion result never decreases with an increase in the input voltage and has no missing codes.

3: See **Section 22.0 "DC and AC Characteristics Graphs and Charts"** for operating characterization.

TABLE 21-13: PIC16LF1902/3 A/D CONVERSION REQUIREMENTS

Standard Operating Conditions (unless otherwise stated)							
Param. No.	**Sym.**	**Characteristic**	**Min.**	**Typ†**	**Max.**	**Units**	**Conditions**
AD130*	TAD	ADC Clock Period (TADC)	1.0	—	6.0	μs	FOSC-based
		ADC Internal FRC Oscillator Period (TFRC)	1.0	2.0	6.0	μs	ADCS<2:0> = x11 (ADC FRC mode)
AD131	TCNV	Conversion Time (not including Acquisition Time)(1)	—	11	—	TAD	Set GO/$\overline{\text{DONE}}$ bit to conversion complete
AD132*	TACQ	Acquisition Time	—	5.0	—	μs	
AD133*	THCD	Holding Capacitor Disconnect Time	— —	1/2 TAD 1/2 TAD + 1TCY	— —		FOSC-based ADCS<2:0> = x11 (ADC FRC mode)

* These parameters are characterized but not tested.

† Data in "Typ" column is at 3.0V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

Note 1: The ADRES register may be read on the following TCY cycle.

23.6 MPLAB X SIM Software Simulator

The MPLAB X SIM Software Simulator allows code development in a PC-hosted environment by simulating the PIC MCUs and dsPIC DSCs on an instruction level. On any given instruction, the data areas can be examined or modified and stimuli can be applied from a comprehensive stimulus controller. Registers can be logged to files for further run-time analysis. The trace buffer and logic analyzer display extend the power of the simulator to record and track program execution, actions on I/O, most peripherals and internal registers.

The MPLAB X SIM Software Simulator fully supports symbolic debugging using the MPLAB XC Compilers, and the MPASM and MPLAB Assemblers. The software simulator offers the flexibility to develop and debug code outside of the hardware laboratory environment, making it an excellent, economical software development tool.

23.7 MPLAB REAL ICE In-Circuit Emulator System

The MPLAB REAL ICE In-Circuit Emulator System is Microchip's next generation high-speed emulator for Microchip Flash DSC and MCU devices. It debugs and programs all 8, 16 and 32-bit MCU, and DSC devices with the easy-to-use, powerful graphical user interface of the MPLAB X IDE.

The emulator is connected to the design engineer's PC using a high-speed USB 2.0 interface and is connected to the target with either a connector compatible with in-circuit debugger systems (RJ-11) or with the new high-speed, noise tolerant, Low-Voltage Differential Signal (LVDS) interconnection (CAT5).

The emulator is field upgradable through future firmware downloads in MPLAB X IDE. MPLAB REAL ICE offers significant advantages over competitive emulators including full-speed emulation, run-time variable watches, trace analysis, complex breakpoints, logic probes, a ruggedized probe interface and long (up to three meters) interconnection cables.

23.8 MPLAB ICD 3 In-Circuit Debugger System

The MPLAB ICD 3 In-Circuit Debugger System is Microchip's most cost-effective, high-speed hardware debugger/programmer for Microchip Flash DSC and MCU devices. It debugs and programs PIC Flash microcontrollers and dsPIC DSCs with the powerful, yet easy-to-use graphical user interface of the MPLAB IDE.

The MPLAB ICD 3 In-Circuit Debugger probe is connected to the design engineer's PC using a high-speed USB 2.0 interface and is connected to the target with a connector compatible with the MPLAB ICD 2 or MPLAB REAL ICE systems (RJ-11). MPLAB ICD 3 supports all MPLAB ICD 2 headers.

23.9 PICkit 3 In-Circuit Debugger/ Programmer

The MPLAB PICkit 3 allows debugging and programming of PIC and dsPIC Flash microcontrollers at a most affordable price point using the powerful graphical user interface of the MPLAB IDE. The MPLAB PICkit 3 is connected to the design engineer's PC using a full-speed USB interface and can be connected to the target via a Microchip debug (RJ-11) connector (compatible with MPLAB ICD 3 and MPLAB REAL ICE). The connector uses two device I/O pins and the Reset line to implement in-circuit debugging and In-Circuit Serial Programming™ (ICSP™).

23.10 MPLAB PM3 Device Programmer

The MPLAB PM3 Device Programmer is a universal, CE compliant device programmer with programmable voltage verification at V_{DDMIN} and V_{DDMAX} for maximum reliability. It features a large LCD display (128 x 64) for menus and error messages, and a modular, detachable socket assembly to support various package types. The ICSP cable assembly is included as a standard item. In Stand-Alone mode, the MPLAB PM3 Device Programmer can read, verify and program PIC devices without a PC connection. It can also set code protection in this mode. The MPLAB PM3 connects to the host PC via an RS-232 or USB cable. The MPLAB PM3 has high-speed communications and optimized algorithms for quick programming of large memory devices, and incorporates an MMC card for file storage and data applications.

23.11 Demonstration/Development Boards, Evaluation Kits, and Starter Kits

A wide variety of demonstration, development and evaluation boards for various PIC MCUs and dsPIC DSCs allows quick application development on fully functional systems. Most boards include prototyping areas for adding custom circuitry and provide application firmware and source code for examination and modification.

The boards support a variety of features, including LEDs, temperature sensors, switches, speakers, RS-232 interfaces, LCD displays, potentiometers and additional EEPROM memory.

The demonstration and development boards can be used in teaching environments, for prototyping custom circuits and for learning about various microcontroller applications.

In addition to the PICDEM™ and dsPICDEM™ demonstration/development board series of circuits, Microchip has a line of evaluation kits and demonstration software for analog filter design, KEELOQ® security ICs, CAN, IrDA®, PowerSmart battery management, SEEVAL® evaluation system, Sigma-Delta ADC, flow rate sensing, plus many more.

Also available are starter kits that contain everything needed to experience the specified device. This usually includes a single application and debug capability, all on one board.

Check the Microchip web page (www.microchip.com) for the complete list of demonstration, development and evaluation kits.

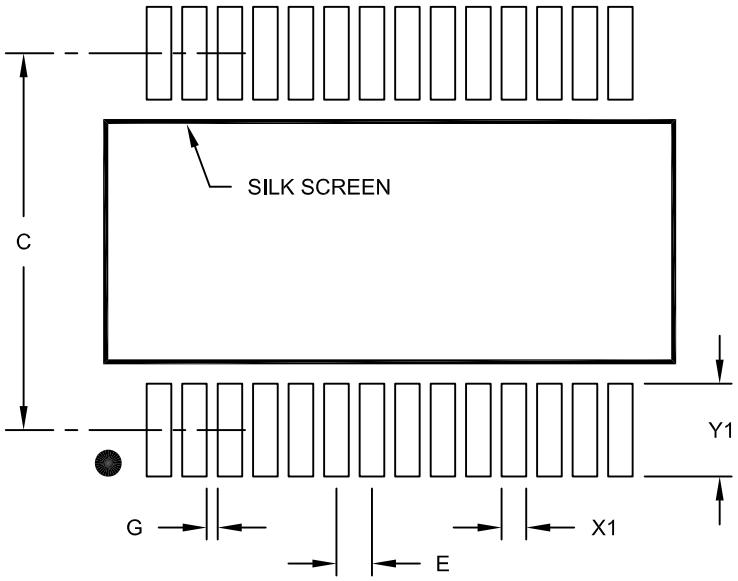
23.12 Third-Party Development Tools

Microchip also offers a great collection of tools from third-party vendors. These tools are carefully selected to offer good value and unique functionality.

- Device Programmers and Gang Programmers from companies, such as SoftLog and CCS
- Software Tools from companies, such as Gimpel and Trace Systems
- Protocol Analyzers from companies, such as Saleae and Total Phase
- Demonstration Boards from companies, such as MikroElektronika, Digilent® and Olimex
- Embedded Ethernet Solutions from companies, such as EZ Web Lynx, WIZnet and IPLogika®

28-Lead Plastic Shrink Small Outline (SS) - 5.30 mm Body [SSOP]

Note: For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging



RECOMMENDED LAND PATTERN

Units		MILLIMETERS		
Dimension Limits		MIN	NOM	MAX
Contact Pitch	E		0.65 BSC	
Contact Pad Spacing	C		7.20	
Contact Pad Width (X28)	X1			0.45
Contact Pad Length (X28)	Y1			1.75
Distance Between Pads	G	0.20		

Notes:

1. Dimensioning and tolerancing per ASME Y14.5M

BSC: Basic Dimension. Theoretically exact value shown without tolerances.

Microchip Technology Drawing No. C04-2073A