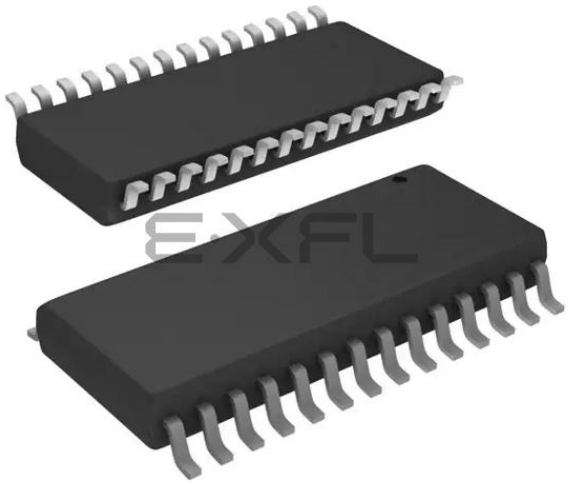
Microchip Technology - PIC16LF1902-I/SO Datasheet



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	3.5KB (2K x 14)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	128 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-SOIC (0.295", 7.50mm Width)
Supplier Device Package	28-SOIC
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic16lf1902-i-so

TABLE 1-2: PIC16LF1902/3 PINOUT DESCRIPTION

Name	Function	Input Type	Output Type	Description
RA0/AN0/SEG12	RA0	TTL	CMOS	General purpose I/O.
	AN0	AN	—	A/D Channel 0 input.
	SEG12	—	AN	LCD Analog output.
RA1/AN1/SEG7	RA1	TTL	CMOS	General purpose I/O.
	AN1	AN	—	A/D Channel 1 input.
	SEG7	—	AN	LCD Analog output.
RA2/AN2/COM2	RA2	TTL	CMOS	General purpose I/O.
	AN2	AN	—	A/D Channel 2 input.
	COM2	—	AN	LCD Analog output.
RA3/AN3/VREF+/COM3/SEG15	RA3	TTL	CMOS	General purpose I/O.
	AN3	AN	—	A/D Channel 3 input.
	VREF+	AN	—	A/D Voltage Reference input.
	COM3	—	AN	LCD Analog output.
	SEG15	—	AN	LCD Analog output.
RA4/T0CKI/SEG4	RA4	TTL	CMOS	General purpose I/O.
	T0CKI	ST	—	Timer0 clock input.
	SEG4	—	AN	LCD Analog output.
RA5/AN4/SEG5	RA5	TTL	CMOS	General purpose I/O.
	AN4	AN	—	A/D Channel 4 input.
	SEG5	—	AN	LCD Analog output.
RA6/CLKOUT/SEG1	RA6	TTL	CMOS	General purpose I/O.
	CLKOUT	—	CMOS	FOSC/4 output.
	SEG1	—	AN	LCD Analog output.
RA7/CLKIN/SEG2	RA7	TTL	CMOS	General purpose I/O.
	CLKIN	CMOS	—	External clock input (EC mode).
	SEG2	—	AN	LCD Analog output.
RB0/AN12/INT/SEG0	RB0	TTL	CMOS	General purpose I/O.
	AN12	AN	—	A/D Channel 12 input.
	INT	ST	—	External interrupt.
	SEG0	—	AN	LCD Analog output.
RB1(1)/AN10/SEG24/VLCD1	RB1	TTL	CMOS	General purpose I/O.
	AN10	AN	—	A/D Channel 10 input.
	SEG24	—	AN	LCD Analog output.
	VLCD1	AN	—	LCD analog input.
RB2(1)/AN8/SEG25/VLCD2	RB2	TTL	CMOS	General purpose I/O.
	AN8	AN	—	A/D Channel 8 input.
	SEG25	—	AN	LCD Analog output.
	VLCD2	AN	—	LCD analog input.
RB3(1)/AN9/SEG26/VLCD3	RB3	TTL	CMOS	General purpose I/O.
	AN9	AN	—	A/D Channel 9 input.
	SEG26	—	AN	LCD Analog output.
	VLCD3	AN	—	LCD analog input.

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.

3.0 MEMORY ORGANIZATION

These devices contain the following types of memory:

- Program Memory
  - Configuration Words
  - Device ID
  - User ID
  - Flash Program Memory
- Data Memory
  - Core Registers
  - Special Function Registers
  - General Purpose RAM
  - Common RAM

The following features are associated with access and control of program memory and data memory:

- PCL and PCLATH
- Stack
- Indirect Addressing

3.1 Program Memory Organization

The enhanced mid-range core has a 15-bit program counter capable of addressing 32K x 14 program memory space. Table 3-1 shows the memory sizes implemented for the PIC16LF1902/3 family. Accessing a location above these boundaries will cause a wrap-around within the implemented memory space. The Reset vector is at 0000h and the interrupt vector is at 0004h (see Figures 3-1, and 3-2).

TABLE 3-1: DEVICE SIZES AND ADDRESSES

Device	Program Memory Space (Words)	Last Program Memory Address	High-Endurance Flash Memory Address Range(1)
PIC16LF1902	2,048	07FFh	0780h-07FFh
PIC16LF1903	4,096	0FFFh	0F80h-0FFFh

Note 1: High-endurance Flash applies to low byte of each address in the range.

TABLE 3-5: SPECIAL FUNCTION REGISTER SUMMARY (CONTINUED)

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 2											
10Ch	LATA	PORTA Data Latch								xxxx xxxx	uuuu uuuu
10Dh	LATB	PORTB Data Latch								xxxx xxxx	uuuu uuuu
10Eh	LATC	PORTC Data Latch								xxxx xxxx	uuuu uuuu
10Fh to 115h	—	Unimplemented								—	—
116h	BORCON	SBOREN	BORFS	—	—	—	—	—	BORRDY	10-- ---q	uu-- ---u
117h	FVRCON	FVREN	FVRRDY	TSEN	TSRNG	—	—	ADFVR1	ADFVR0	0q00 --00	0q00 --00
118h to 11Fh	—	Unimplemented								—	—
Bank 3											
18Ch	ANSELA	—	—	ANSA5	—	ANSA3	ANSA2	ANSA1	ANSA0	--1- 1111	--11 1111
18Dh	ANSELB	—	—	ANSB5	ANSB4	ANSB3	ANSB2	ANSB1	ANSB0	--11 1111	--11 1111
18Eh	—	Unimplemented								—	—
18Fh	—	Unimplemented								—	—
190h	—	Unimplemented								—	—
191h	PMADRL	Program Memory Address Register Low Byte								0000 0000	0000 0000
192h	PMADRH	—(2)	Program Memory Address Register High Byte							1000 0000	1000 0000
193h	PMDATL	Program Memory Read Data Register Low Byte								xxxx xxxx	uuuu uuuu
194h	PMDATH	—	—	Program Memory Read Data Register High Byte						--xx xxxx	--uu uuuu
195h	PMCON1	—(2)	CFGS	LWLO	FREE	WRERR	WREN	WR	RD	1000 x000	1000 q000
196h	PMCON2	Program Memory Control Register 2								0000 0000	0000 0000
197h to 19Fh	—	Unimplemented								—	—
Bank 4											
20Ch	—	Unimplemented								—	—
20Dh	WPUB	WPUB7	WPUB6	WPUB5	WPUB4	WPUB3	WPUB2	WPUB1	WPUB0	1111 1111	1111 1111
20Eh	—	Unimplemented								—	—
20Fh	—	Unimplemented								—	—
210h	WPUE	—	—	—	—	WPUE3	—	—	—	---- 1---	---- 1---
211h to 21Fh	—	Unimplemented								—	—
Bank 5											
28Ch — 29Fh	—	Unimplemented								—	—
Bank 6											
30Ch — 31Fh	—	Unimplemented								—	—

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

Note 1: These registers can be addressed from any bank.

2: Unimplemented, read as '1'.

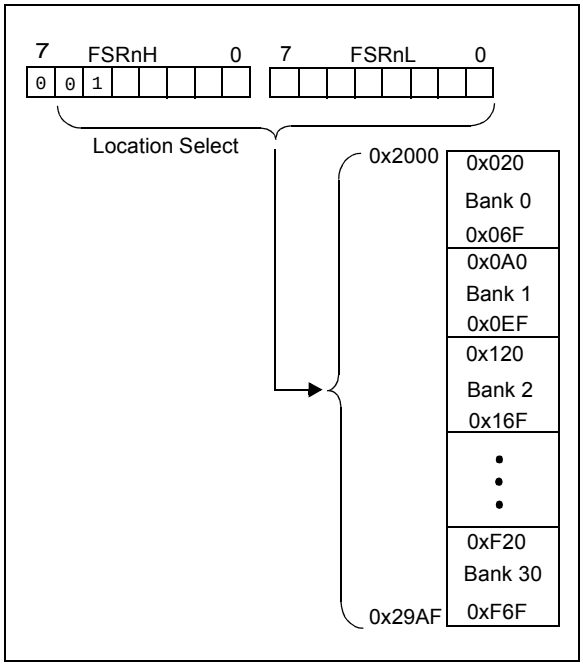
3.5.2 LINEAR DATA MEMORY

The linear data memory is the region from FSR address 0x2000 to FSR address 0x29AF. This region is a virtual region that points back to the 80-byte blocks of GPR memory in all the banks.

Unimplemented memory reads as 0x00. Use of the linear data memory region allows buffers to be larger than 80 bytes because incrementing the FSR beyond one bank will go directly to the GPR memory of the next bank.

The 16 bytes of common memory are not included in the linear data memory region.

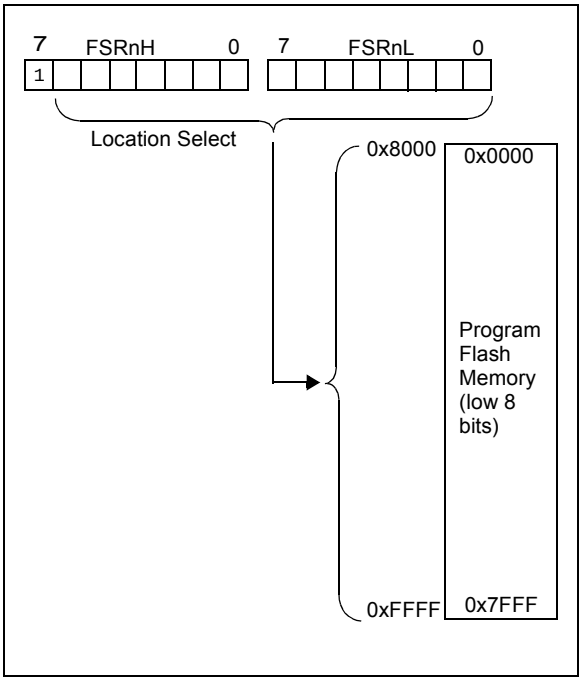
FIGURE 3-11: LINEAR DATA MEMORY MAP



3.5.3 PROGRAM FLASH MEMORY

To make constant data access easier, the entire program Flash memory is mapped to the upper half of the FSR address space. When the MSB of FSRnH is set, the lower 15 bits are the address in program memory which will be accessed through INDF. Only the lower eight bits of each memory location is accessible via INDF. Writing to the program Flash memory cannot be accomplished via the FSR/INDF interface. All instructions that access program Flash memory via the FSR/INDF interface will require one additional instruction cycle to complete.

FIGURE 3-12: PROGRAM FLASH MEMORY MAP



7.1 Operation

Interrupts are disabled upon any device Reset. They are enabled by setting the following bits:

- GIE bit of the INTCON register
- Interrupt Enable bit(s) for the specific interrupt event(s)
- PEIE bit of the INTCON register (if the Interrupt Enable bit of the interrupt event is contained in the PIE1 and PIE2 registers)

The INTCON, PIR1 and PIR2 registers record individual interrupts via interrupt flag bits. Interrupt flag bits will be set, regardless of the status of the GIE, PEIE and individual interrupt enable bits.

The following events happen when an interrupt event occurs while the GIE bit is set:

- Current prefetched instruction is flushed
- GIE bit is cleared
- Current Program Counter (PC) is pushed onto the stack
- Critical registers are automatically saved to the shadow registers (See **Section 7.5 "Automatic Context Saving"**)
- PC is loaded with the interrupt vector 0004h

The firmware within the Interrupt Service Routine (ISR) should determine the source of the interrupt by polling the interrupt flag bits. The interrupt flag bits must be cleared before exiting the ISR to avoid repeated interrupts. Because the GIE bit is cleared, any interrupt that occurs while executing the ISR will be recorded through its interrupt flag, but will not cause the processor to redirect to the interrupt vector.

The `RETFIE` instruction exits the ISR by popping the previous address from the stack, restoring the saved context from the shadow registers and setting the GIE bit.

For additional information on a specific interrupt's operation, refer to its peripheral chapter.

Note 1:	Individual interrupt flag bits are set, regardless of the state of any other enable bits.
2:	All interrupts will be ignored while the GIE bit is cleared. Any interrupt occurring while the GIE bit is clear will be serviced when the GIE bit is set again.

7.2 Interrupt Latency

Interrupt latency is defined as the time from when the interrupt event occurs to the time code execution at the interrupt vector begins. The latency for synchronous interrupts is three or four instruction cycles. For asynchronous interrupts, the latency is three to five instruction cycles, depending on when the interrupt occurs. See Figure 7-2 and Figure 7.3 for more details.

7.6 Interrupt Control Registers

7.6.1 INTCON REGISTER

The INTCON register is a readable and writable register, which contains the various enable and flag bits for TMR0 register overflow, interrupt-on-change and external INT pin interrupts.

Note: Interrupt flag bits are set when an interrupt condition occurs, regardless of the state of its corresponding enable bit or the Global Enable bit, GIE of the INTCON register. User software should ensure the appropriate interrupt flag bits are clear prior to enabling an interrupt.

REGISTER 7-1: INTCON: INTERRUPT CONTROL REGISTER

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R-0/0
GIE	PEIE	TMR0IE	INTE	IOCIE	TMR0IF	INTF	IOCIF
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 **GIE:** Global Interrupt Enable bit
1 = Enables all active interrupts
0 = Disables all interrupts

bit 6 **PEIE:** Peripheral Interrupt Enable bit
1 = Enables all active peripheral interrupts
0 = Disables all peripheral interrupts

bit 5 **TMR0IE:** Timer0 Overflow Interrupt Enable bit
1 = Enables the Timer0 interrupt
0 = Disables the Timer0 interrupt

bit 4 **INTE:** INT External Interrupt Enable bit
1 = Enables the INT external interrupt
0 = Disables the INT external interrupt

bit 3 **IOCIE:** Interrupt-on-Change Interrupt Enable bit
1 = Enables the interrupt-on-change interrupt
0 = Disables the interrupt-on-change interrupt

bit 2 **TMR0IF:** Timer0 Overflow Interrupt Flag bit
1 = TMR0 register has overflowed
0 = TMR0 register did not overflow

bit 1 **INTF:** INT External Interrupt Flag bit
1 = The INT external interrupt occurred
0 = The INT external interrupt did not occur

bit 0 **IOCIF:** Interrupt-on-Change Interrupt Flag bit
1 = When at least one of the interrupt-on-change pins changed state
0 = None of the interrupt-on-change pins have changed state

10.4 User ID, Device ID and Configuration Word Access

Instead of accessing program memory, the User ID's, Device ID/Revision ID and Configuration Words can be accessed when CFGS = 1 in the PMCON1 register. This is the region that would be pointed to by PC<15> = 1, but not all addresses are accessible. Different access may exist for reads and writes. Refer to Table 10-2.

When read access is initiated on an address outside the parameters listed in Table 10-2, the PMDATH:PMDATL register pair is cleared, reading back '0's.

TABLE 10-2: USER ID, DEVICE ID AND CONFIGURATION WORD ACCESS (CFGS = 1)

Address	Function	Read Access	Write Access
8000h-8003h	User IDs	Yes	Yes
8006h	Device ID/Revision ID	Yes	No
8007h-8008h	Configuration Words 1 and 2	Yes	No

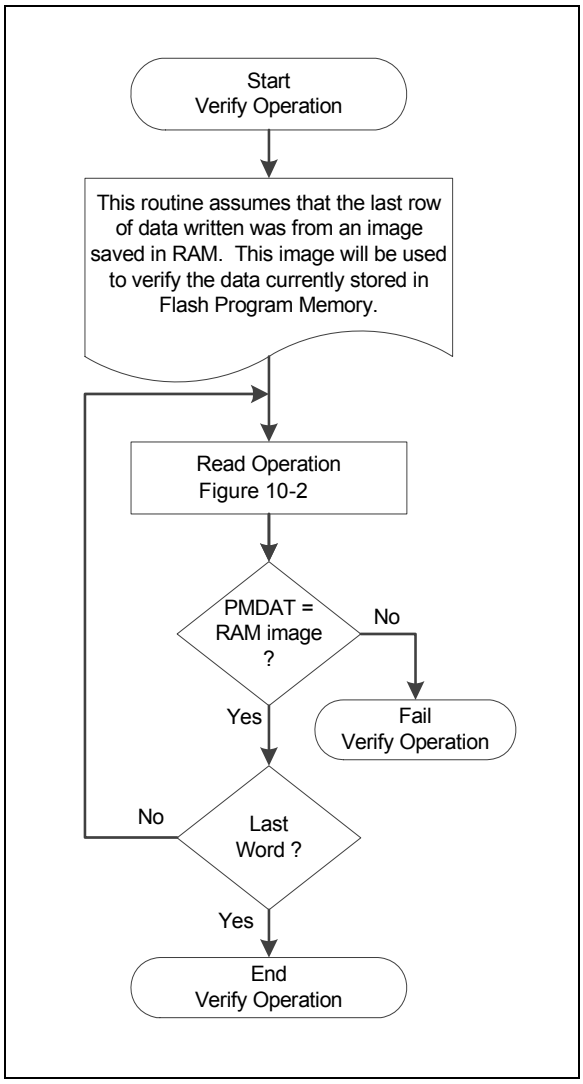
EXAMPLE 10-4: CONFIGURATION WORD AND DEVICE ID ACCESS

```
* This code block will read 1 word of program memory at the memory address:
*   PROG_ADDR_LO (must be 00h-08h) data will be returned in the variables;
*   PROG_DATA_HI, PROG_DATA_LO

    BANKSEL  PMADRL           ; Select correct Bank
    MOVLW    PROG_ADDR_LO     ;
    MOVWF    PMADRL           ; Store LSB of address
    CLRF     PMADRH           ; Clear MSB of address

    BSF      PMCON1,CFGS      ; Select Configuration Space
    BCF      INTCON,GIE       ; Disable interrupts
    BSF      PMCON1,RD        ; Initiate read
    NOP                       ; Executed (See Figure 10-2)
    NOP                       ; Ignored (See Figure 10-2)
    BSF      INTCON,GIE       ; Restore interrupts

    MOVF     PMDATL,W         ; Get LSB of word
    MOVWF    PROG_DATA_LO     ; Store in user location
    MOVF     PMDATH,W         ; Get MSB of word
    MOVWF    PROG_DATA_HI     ; Store in user location
```

10.5 Write Verify

It is considered good programming practice to verify that program memory writes agree with the intended value. Since program memory is stored as a full page then the stored program memory contents are compared with the intended data stored in RAM after the last write is complete.

FIGURE 10-8: FLASH PROGRAM MEMORY VERIFY FLOWCHART



11.0 I/O PORTS

In general, when a peripheral is enabled on a port pin, that pin cannot be used as a general purpose output. However, the pin can still be read.

Each port has three standard registers for its operation. These registers are:

- TRISx registers (data direction)
- PORTx registers (reads the levels on the pins of the device)
- LATx registers (output latch)

Some ports may have one or more of the following additional registers. These registers are:

- ANSELx (analog select)
- WPUx (weak pull-up)

TABLE 11-1: PORT AVAILABILITY PER DEVICE

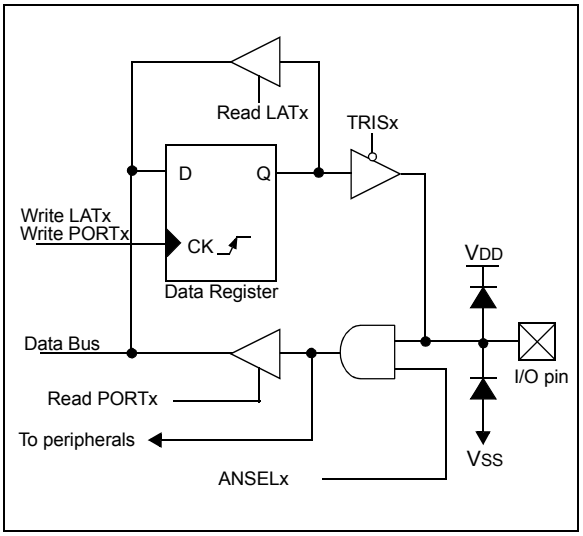
Device	PORTA	PORTB	PORTC	PORTE
PIC16LF1902/3	•	•	•	•

The Data Latch (LATA register) is useful for read-modify-write operations on the value that the I/O pins are driving.

A write operation to the LATA register has the same effect as a write to the corresponding PORTA register. A read of the LATA register reads of the values held in the I/O PORT latches, while a read of the PORTA register reads the actual I/O pin value.

Ports that support analog inputs have an associated ANSELx register. When an ANSEL bit is set, the digital input buffer associated with that bit is disabled. Disabling the input buffer prevents analog signal levels on the pin between a logic high and low from causing excessive current in the logic input circuitry. A simplified model of a generic I/O port, without the interfaces to other peripherals, is shown in Figure 11-1.

FIGURE 11-1: GENERIC I/O PORT OPERATION



EXAMPLE 11-1: INITIALIZING PORTA

```
; This code example illustrates
; initializing the PORTA register. The
; other ports are initialized in the same
; manner.

BANKSEL  PORTA       ;
CLRF     PORTA       ;Init PORTA
BANKSEL  LATA        ;Data Latch
CLRF     LATA        ;
BANKSEL  ANSELA      ;
CLRF     ANSELA      ;digital I/O
BANKSEL  TRISA       ;
MOVLW    B'00111000' ;Set RA<5:3> as inputs
MOVWF    TRISA       ;and set RA<2:0> as
                     ;outputs
```

TABLE 11-8: SUMMARY OF REGISTERS ASSOCIATED WITH PORTC

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Register on Page
LATC	LATC7	LATC6	LATC5	LATC4	LATC3	LATC2	LATC1	LATC0	93
PORTC	RC7	RC6	RC5	RC4	RC3	RC2	RC1	RC0	93
TRISC	TRISC7	TRISC6	TRISC5	TRISC4	TRISC3	TRISC2	TRISC1	TRISC0	93

Legend: x = unknown, u = unchanged, - = unimplemented locations read as '0'. Shaded cells are not used by PORTC.

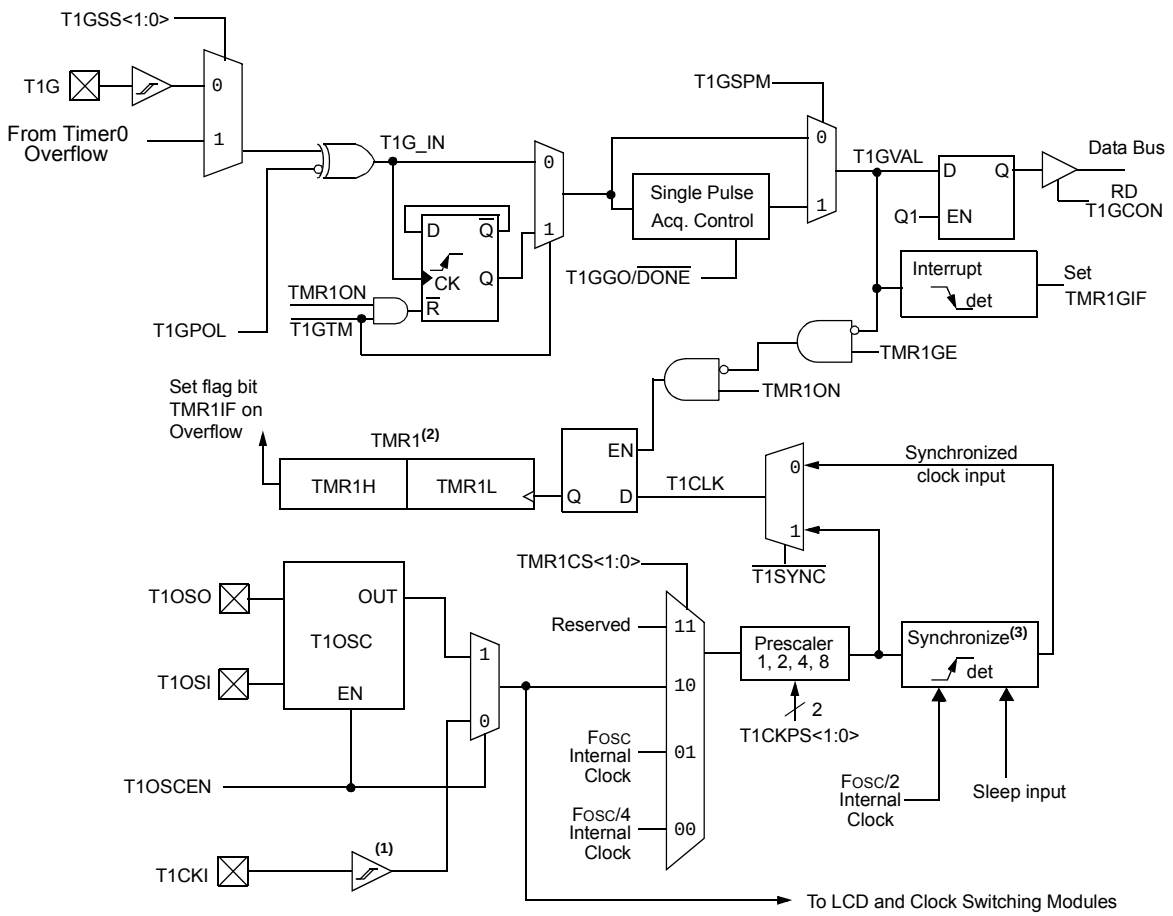
17.0 TIMER1 MODULE WITH GATE CONTROL

The Timer1 module is a 16-bit timer/counter with the following features:

- 16-bit timer/counter register pair (TMR1H:TMR1L)
- Programmable internal or external clock source
- 2-bit prescaler
- Dedicated 32 kHz oscillator circuit
- Multiple Timer1 gate (count enable) sources
- Interrupt on overflow
- Wake-up on overflow (external clock, Asynchronous mode only)
- Selectable Gate Source Polarity
- Gate Toggle mode
- Gate Single-pulse mode
- Gate Value Status
- Gate Event Interrupt

Figure 17-1 is a block diagram of the Timer1 module.

FIGURE 17-1: TIMER1 BLOCK DIAGRAM



Note 1: ST Buffer is high speed type when using T1CKI.
2: Timer1 register increments on rising edge.
3: Synchronize does not operate while in Sleep.

17.1 Timer1 Operation

The Timer1 module is a 16-bit incrementing counter which is accessed through the TMR1H:TMR1L register pair. Writes to TMR1H or TMR1L directly update the counter.

When used with an internal clock source, the module is a timer and increments on every instruction cycle. When used with an external clock source, the module can be used as either a timer or counter and increments on every selected edge of the external source.

Timer1 is enabled by configuring the TMR1ON and TMR1GE bits in the T1CON and T1GCON registers, respectively. Table 17-1 displays the Timer1 enable selections.

TABLE 17-1: TIMER1 ENABLE SELECTIONS

TMR1ON	TMR1GE	Timer1 Operation
0	0	Off
0	1	Off
1	0	Always On
1	1	Count Enabled

17.2 Clock Source Selection

The TMR1CS<1:0> and T1OSCEN bits of the T1CON register are used to select the clock source for Timer1. Table 17-2 displays the clock source selections.

17.2.1 INTERNAL CLOCK SOURCE

When the internal clock source is selected the TMR1H:TMR1L register pair will increment on multiples of FOSC as determined by the Timer1 prescaler.

When the FOSC internal clock source is selected, the Timer1 register value will increment by four counts every instruction clock cycle. Due to this condition, a 2 LSB error in resolution will occur when reading the Timer1 value. To utilize the full resolution of Timer1, an asynchronous input signal must be used to gate the Timer1 clock input.

The following asynchronous source may be used:

- Asynchronous event on the T1G pin to Timer1 gate

17.2.2 EXTERNAL CLOCK SOURCE

When the external clock source is selected, the Timer1 module may work as a timer or a counter.

When enabled to count, Timer1 is incremented on the rising edge of the external clock input T1CKI or the capacitive sensing oscillator signal. Either of these external clock sources can be synchronized to the microcontroller system clock or they can run asynchronously.

When used as a timer with a clock oscillator, an external 32.768 kHz crystal can be used in conjunction with the dedicated internal oscillator circuit.

Note: In Counter mode, a falling edge must be registered by the counter prior to the first incrementing rising edge after any one or more of the following conditions:

- Timer1 enabled after POR
- Write to TMR1H or TMR1L
- Timer1 is disabled
- Timer1 is disabled (TMR1ON = 0) when T1CKI is high then Timer1 is enabled (TMR1ON=1) when T1CKI is low.

TABLE 17-2: CLOCK SOURCE SELECTIONS

TMR1CS1	TMR1CS0	T1OSCEN	Clock Source
0	0	x	Instruction Clock (FOSC/4)
0	1	x	System Clock (FOSC)
1	0	0	External Clocking on T1CKI Pin
1	0	1	Osc. Circuit on T1OSI/T1OSO Pins
1	1	x	Reserved

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.

18.12 Configuring the LCD Module

The following is the sequence of steps to configure the LCD module.

1. Select the frame clock prescale using bits LP<3:0> of the LCDPS register.
2. Configure the appropriate pins to function as segment drivers using the LCDSEn registers.
3. Configure the LCD module for the following using the LCDCON register:
   - Multiplex and Bias mode, bits LMUX<1:0>
   - Timing source, bits CS<1:0>
   - Sleep mode, bit SLPEN
4. Write initial values to pixel data registers, LCDDATA0 through LCDDATA21.
5. Clear LCD Interrupt Flag, LCDIF bit of the PIR2 register and if desired, enable the interrupt by setting bit LCDIE of the PIE2 register.
6. Configure bias voltages by setting the LCDRL, LCDREF and the associated ANSELx registers as needed.
7. Enable the LCD module by setting bit LCDEN of the LCDCON register.

18.13 Disabling the LCD Module

To disable the LCD module, write all '0's to the LCDCON register.

18.14 LCD Current Consumption

When using the LCD module the current consumption consists of the following three factors:

- Oscillator Selection
- LCD Bias Source
- Capacitance of the LCD segments

The current consumption of just the LCD module can be considered negligible compared to these other factors.

18.14.1 OSCILLATOR SELECTION

The current consumed by the clock source selected must be considered when using the LCD module. See **Section 21.0 "Electrical Specifications"** for oscillator current consumption information.

18.14.2 LCD BIAS SOURCE

The LCD bias source, internal or external, can contribute significantly to the current consumption. Use the highest possible resistor values while maintaining contrast to minimize current.

18.14.3 CAPACITANCE OF THE LCD SEGMENTS

The LCD segments which can be modeled as capacitors which must be both charged and discharged every frame. The size of the LCD segment and its technology determines the segment's capacitance.

20.0 INSTRUCTION SET SUMMARY

Each PIC16 instruction is a 14-bit word containing the operation code (opcode) and all required operands. The opcodes are broken into three broad categories.

- Byte Oriented
- Bit Oriented
- Literal and Control

The literal and control category contains the most varied instruction word format.

Table 20-3 lists the instructions recognized by the MPASM™ assembler.

All instructions are executed within a single instruction cycle, with the following exceptions, which may take two or three cycles:

- Subroutine takes two cycles (CALL, CALLW)
- Returns from interrupts or subroutines take two cycles (RETURN, RETLW, RETFIE)
- Program branching takes two cycles (GOTO, BRA, BRW, BTFSS, BTFSC, DECFSZ, INCSFZ)
- One additional instruction cycle will be used when any instruction references an indirect file register and the file select register is pointing to program memory.

One instruction cycle consists of four oscillator cycles; for an oscillator frequency of 4 MHz, this gives a nominal instruction execution rate of 1 MHz.

All instruction examples use the format '0xhh' to represent a hexadecimal number, where 'h' signifies a hexadecimal digit.

20.1 Read-Modify-Write Operations

Any instruction that specifies a file register as part of the instruction performs a Read-Modify-Write (R-M-W) operation. The register is read, the data is modified, and the result is stored according to either the instruction, or the destination designator 'd'. A read operation is performed on a register even if the instruction writes to that register.

TABLE 20-1: OPCODE FIELD DESCRIPTIONS

Field	Description
f	Register file address (0x00 to 0x7F)
W	Working register (accumulator)
b	Bit address within an 8-bit file register
k	Literal field, constant data or label
x	Don't care location (= 0 or 1). The assembler will generate code with x = 0. It is the recommended form of use for compatibility with all Microchip software tools.
d	Destination select; d = 0: store result in W, d = 1: store result in file register f. Default is d = 1.
n	FSR or INDF number. (0-1)
mm	Pre-post increment-decrement mode selection

TABLE 20-2: ABBREVIATION DESCRIPTIONS

Field	Description
PC	Program Counter
$\overline{TO}$	Time-out bit
C	Carry bit
DC	Digit carry bit
Z	Zero bit
$\overline{PD}$	Power-down bit

21.3 AC Characteristics

The timing parameter symbols have been created with one of the following formats:

1. TppS2ppS
2. TppS

T			
F	Frequency	T	Time

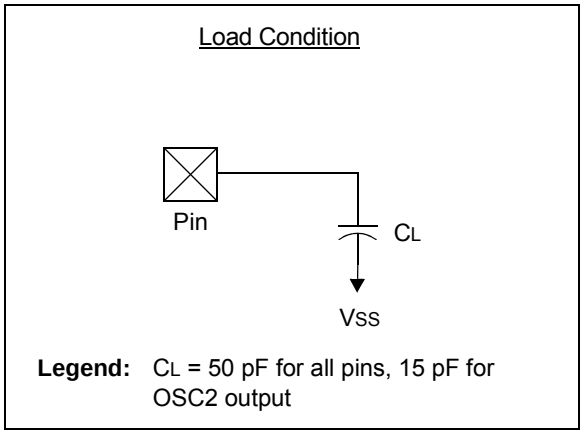
Lowercase letters (pp) and their meanings:

pp			
cc	CCP1	osc	OSC1
ck	CLKOUT	rd	$\overline{RD}$
cs	$\overline{CS}$	rw	$\overline{RD}$ or $\overline{WR}$
di	SDI	sc	SCK
do	SDO	ss	$\overline{SS}$
dt	Data in	t0	T0CKI
io	I/O PORT	t1	T1CKI
mc	$\overline{MCLR}$	wr	$\overline{WR}$

Uppercase letters and their meanings:

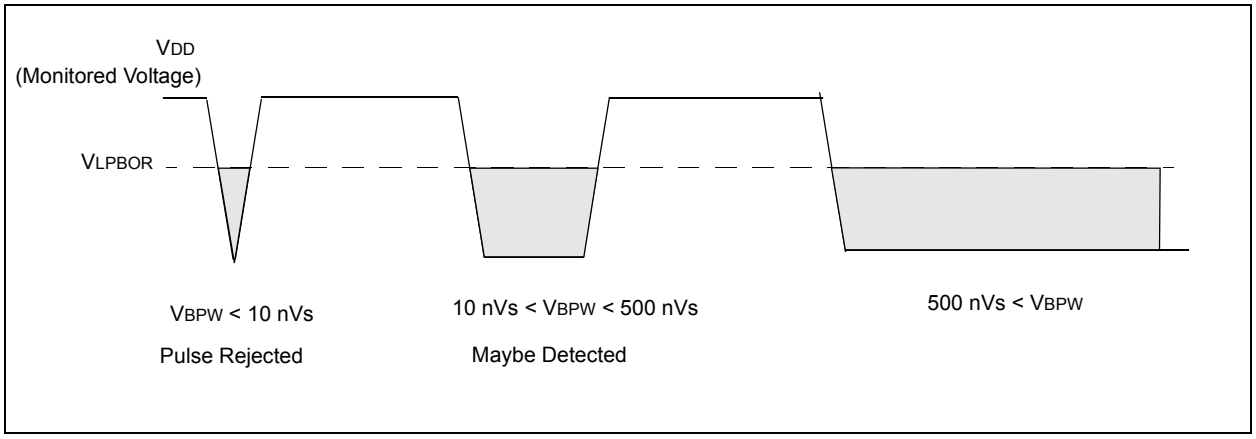
S			
F	Fall	P	Period
H	High	R	Rise
I	Invalid (High-Impedance)	V	Valid
L	Low	Z	High-Impedance

FIGURE 21-4: LOAD CONDITIONS



Legend: C_L = 50 pF for all pins, 15 pF for OSC2 output

FIGURE 21-8: MINIMUM PULSE WIDTH FOR LPBOR DETECTION



23.2 MPLAB XC Compilers

The MPLAB XC Compilers are complete ANSI C compilers for all of Microchip's 8, 16, and 32-bit MCU and DSC devices. These compilers provide powerful integration capabilities, superior code optimization and ease of use. MPLAB XC Compilers run on Windows, Linux or MAC OS X.

For easy source level debugging, the compilers provide debug information that is optimized to the MPLAB X IDE.

The free MPLAB XC Compiler editions support all devices and commands, with no time or memory restrictions, and offer sufficient code optimization for most applications.

MPLAB XC Compilers include an assembler, linker and utilities. The assembler generates relocatable object files that can then be archived or linked with other relocatable object files and archives to create an executable file. MPLAB XC Compiler uses the assembler to produce its object file. Notable features of the assembler include:

- Support for the entire device instruction set
- Support for fixed-point and floating-point data
- Command-line interface
- Rich directive set
- Flexible macro language
- MPLAB X IDE compatibility

23.3 MPASM Assembler

The MPASM Assembler is a full-featured, universal macro assembler for PIC10/12/16/18 MCUs.

The MPASM Assembler generates relocatable object files for the MPLINK Object Linker, Intel® standard HEX files, MAP files to detail memory usage and symbol reference, absolute LST files that contain source lines and generated machine code, and COFF files for debugging.

The MPASM Assembler features include:

- Integration into MPLAB X IDE projects
- User-defined macros to streamline assembly code
- Conditional assembly for multipurpose source files
- Directives that allow complete control over the assembly process

23.4 MPLINK Object Linker/ MPLIB Object Librarian

The MPLINK Object Linker combines relocatable objects created by the MPASM Assembler. It can link relocatable objects from precompiled libraries, using directives from a linker script.

The MPLIB Object Librarian manages the creation and modification of library files of precompiled code. When a routine from a library is called from a source file, only the modules that contain that routine will be linked in with the application. This allows large libraries to be used efficiently in many different applications.

The object linker/library features include:

- Efficient linking of single libraries instead of many smaller files
- Enhanced code maintainability by grouping related modules together
- Flexible creation of libraries with easy module listing, replacement, deletion and extraction

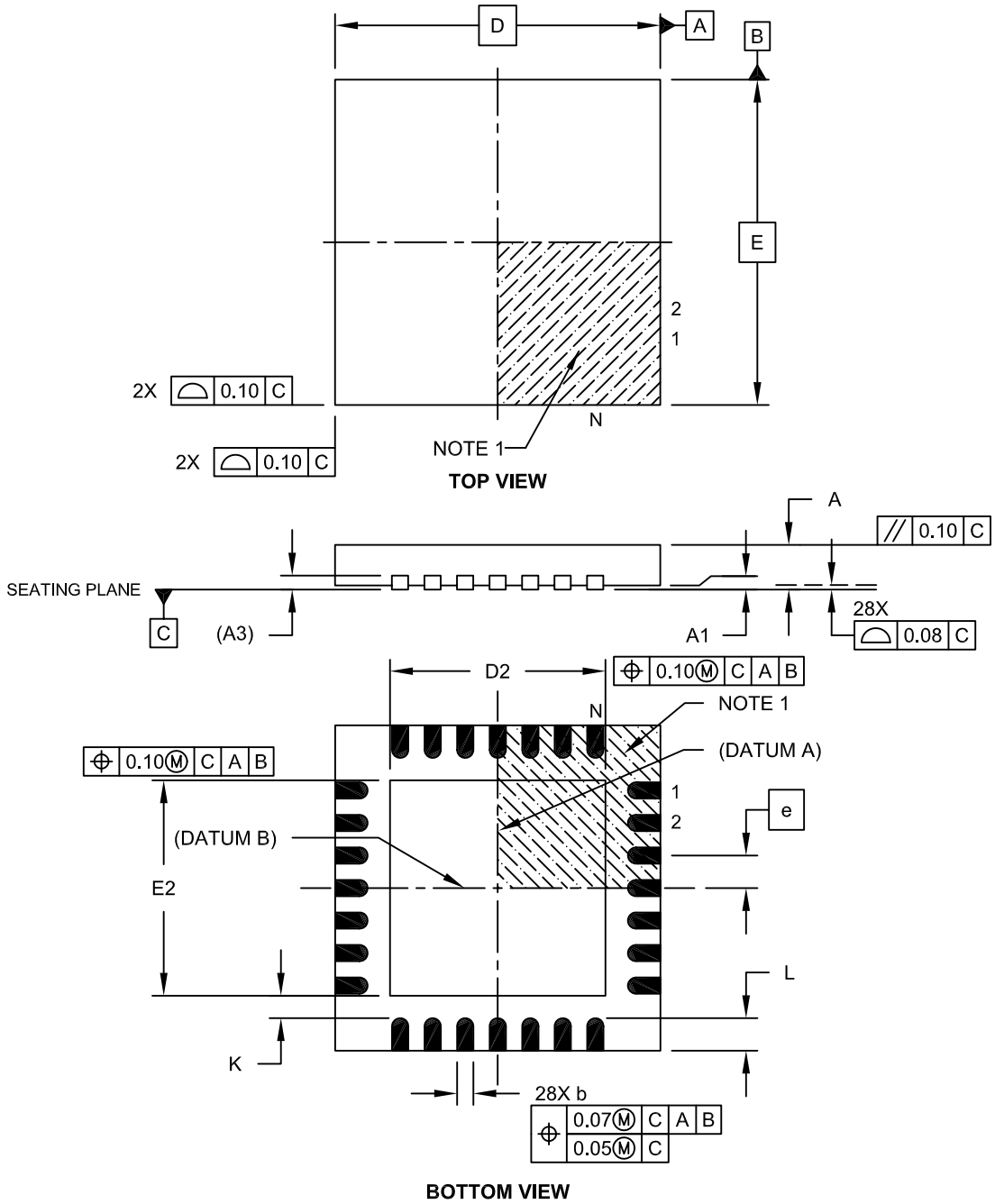
23.5 MPLAB Assembler, Linker and Librarian for Various Device Families

MPLAB Assembler produces relocatable machine code from symbolic assembly language for PIC24, PIC32 and dsPIC DSC devices. MPLAB XC Compiler uses the assembler to produce its object file. The assembler generates relocatable object files that can then be archived or linked with other relocatable object files and archives to create an executable file. Notable features of the assembler include:

- Support for the entire device instruction set
- Support for fixed-point and floating-point data
- Command-line interface
- Rich directive set
- Flexible macro language
- MPLAB X IDE compatibility

28-Lead Plastic Ultra Thin Quad Flat, No Lead Package (MV) – 4x4x0.5 mm Body [UQFN]

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



Microchip Technology Drawing C04-152A Sheet 1 of 2

Note the following details of the code protection feature on Microchip devices:

- Microchip products meet the specification contained in their particular Microchip Data Sheet.
- Microchip believes that its family of products is one of the most secure families of its kind on the market today, when used in the intended manner and under normal conditions.
- There are dishonest and possibly illegal methods used to breach the code protection feature. All of these methods, to our knowledge, require using the Microchip products in a manner outside the operating specifications contained in Microchip's Data Sheets. Most likely, the person doing so is engaged in theft of intellectual property.
- Microchip is willing to work with the customer who is concerned about the integrity of their code.
- Neither Microchip nor any other semiconductor manufacturer can guarantee the security of their code. Code protection does not mean that we are guaranteeing the product as "unbreakable."

Code protection is constantly evolving. We at Microchip are committed to continuously improving the code protection features of our products. Attempts to break Microchip's code protection feature may be a violation of the Digital Millennium Copyright Act. If such acts allow unauthorized access to your software or other copyrighted work, you may have a right to sue for relief under that Act.

Microchip received ISO/TS-16949:2009 certification for its worldwide headquarters, design and wafer fabrication facilities in Chandler and Tempe, Arizona; Gresham, Oregon and design centers in California and India. The Company's quality system processes and procedures are for its PIC® MCUs and dsPIC® DSCs, KEELOQ® code hopping devices, Serial EEPROMs, microperipherals, nonvolatile memory and analog products. In addition, Microchip's quality system for the design and manufacture of development systems is ISO 9001:2000 certified.

QUALITY MANAGEMENT SYSTEM
CERTIFIED BY DNV
═ ISO/TS 16949 ═

Trademarks

ISBN: 978-1-5224-0546-7