



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 "<u>Embedded -</u> <u>Microcontrollers</u>"

#### Details

Product Status	Active
Core Processor	PIC
Core Size	8-Bit
Speed	32MHz
Connectivity	I <sup>2</sup> C, LINbus, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, POR, PSMC, PWM, WDT
Number of I/O	24
Program Memory Size	3.5KB (2K x 14)
Program Memory Type	FLASH
EEPROM Size	256 x 8
RAM Size	256 x 8
Voltage - Supply (Vcc/Vdd)	1.8V ~ 3.6V
Data Converters	A/D 11x12b; D/A 1x8b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 125°C (TA)
Mounting Type	Surface Mount
Package / Case	28-VQFN Exposed Pad
Supplier Device Package	28-QFN (6x6)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic16lf1782-e-ml

Email: info@E-XFL.COM

Address: Room A, 16/F, Full Win Commercial Centre, 573 Nathan Road, Mongkok, Hong Kong

# 3.3.1 SPECIAL FUNCTION REGISTER

The Special Function Registers are registers used by the application to control the desired operation of peripheral functions in the device. The Special Function Registers occupy the 20 bytes after the core registers of every data memory bank (addresses x0Ch/x8Ch through x1Fh/x9Fh). The registers associated with the operation of the peripherals are described in the appropriate peripheral chapter of this data sheet.

#### 3.3.2 GENERAL PURPOSE RAM

There are up to 80 bytes of GPR in each data memory bank. The Special Function Registers occupy the 20 bytes after the core registers of every data memory bank (addresses x0Ch/x8Ch through x1Fh/x9Fh).

#### 3.3.2.1 Linear Access to GPR

The general purpose RAM can be accessed in a non-banked method via the FSRs. This can simplify access to large memory structures. See **Section 3.6.2** "Linear Data Memory" for more information.

#### 3.3.3 COMMON RAM

There are 16 bytes of common RAM accessible from all banks.

#### FIGURE 3-3: BANKED MEMORY PARTITIONING

7-bit Bank Offset	Memory Region	
00h 0Bh	Core Registers (12 bytes)	
0Ch	Special Function Registers (20 bytes maximum)	
1Fh 20h		
ŝEb	General Purpose RAM (80 bytes maximum)	
70h 7Fh	Common RAM (16 bytes)	

# PIC16(L)F1782/3

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
Banl	Bank 0										
00Ch	PORTA	PORTA Data L	PORTA Data Latch when written: PORTA pins when read							XXXX XXXX	uuuu uuuu
00Dh	PORTB	PORTB Data L	PORTB Data Latch when written: PORTB pins when read								uuuu uuuu
00Eh	PORTC	PORTC Data I	_atch when w	ritten: PORTC	oins when read					XXXX XXXX	uuuu uuuu
00Fh	—	Unimplemente	d							_	_
010h	PORTE	—	_	—	—	RE3	—	—	_	x	u
011h	PIR1	TMR1GIF	ADIF	RCIF	TXIF	SSP1IF	CCP1IF	TMR2IF	TMR1IF	0000 0000	0000 0000
012h	PIR2	OSFIF	C2IF	C1IF	EEIF	BCL1IF	—	C3IF	CCP2IF	0000 0-00	0000 0-00
013h	—	Unimplemente	d	•				•	•	_	_
014h	PIR4	_	_	PSMC2TIF	PSMC1TIF	—	_	PSMC2SIF	PSMC1SIF	0000	0000
015h	TMR0	Timer0 Module	Register							XXXX XXXX	uuuu uuuu
016h	TMR1L	Holding Regist	er for the Lea	st Significant B	yte of the 16-bit	TMR1 Regist	er			XXXX XXXX	uuuu uuuu
017h	TMR1H	Holding Regist	er for the Mo	st Significant B	te of the 16-bit	TMR1 Registe	er			XXXX XXXX	uuuu uuuu
018h	T1CON	TMR1CS1	TMR1CS0	T1CKPS1	T1CKPS0	T10SCEN	T1SYNC	—	TMR10N	0000 00-0	uuuu uu-u
019h	T1GCON	TMR1GE	T1GPOL	T1GTM	T1GSPM	T <u>1GGO</u> / DONE	T1GVAL	T1GSS	S<1:0>	0000 0x00	uuuu uxuu
016h	TMR2	Holding Regist	er for the Lea	st Significant B	yte of the 16-bit	TMR2 Regist	er			XXXX XXXX	uuuu uuuu
017h	PR2	Holding Regist	er for the Mo	st Significant By	/te of the 16-bit	TMR2 Registe	er			XXXX XXXX	uuuu uuuu
018h	T2CON	_		T2OUT	PS<3:0>		TMR2ON	T2CKP	S<1:0>	-000 0000	-000 0000
01Dh to 01Fh	_	Unimplemented								_	_
Ban	k 1										
08Ch	TRISA	PORTA Data Direction Register								1111 1111	1111 1111
08Dh	TRISB	PORTB Data Direction Register								1111 1111	1111 1111
08Eh	TRISC	PORTC Data I	Direction Regi	ister						1111 1111	1111 1111
08Fh	—	Unimplemente	d							—	
090h	TRISE	—	—	_		(2)		—	—	1	1
091h	PIE1	TMR1GIE	ADIE	RCIE	TXIE	SSP1IE	CCP1IE	TMR2IE	TMR1IE	0000 0000	0000 0000
092h	PIE2	OSEIE	C2IE	C1IE	EEIE	BCL1IE	_	C3IE	CCP2IE	0000 0-00	0000 0-00
093h	—	Unimplemente	d							—	-
094h	PIE4	_	_	PSMC2TIE	PSMC1TIE	_	_	PSMC2SIE	PSMC1SIE	0000	0000
095h	OPTION_REG	WPUEN	INTEDG	TMR0CS	TMR0SE	PSA		PS<2:0>		1111 1111	1111 1111
096h	PCON	STKOVF	STKUNF	—	RWDT	RMCLR	RI	POR	BOR	00-1 11qq	qq-q qquu
097h	WDTCON	—	_		١	VDTPS<4:0>			SWDTEN	01 0110	01 0110
098h	OSCTUNE	_	_			TUN<	5:0>			00 0000	00 0000
099h	OSCCON	SPLLEN		IRCF	<3:0>		—	SCS	<1:0>	0011 1-00	0011 1-00
09Ah	OSCSTAT	T10SCR	PLLR	OSTS	HFIOFR	HFIOFL	MFIOFR	LFIOFR	HFIOFS	00q000	dddd0d
09Bh	ADRESL	A/D Result Re	gister Low		-					XXXX XXXX	uuuu uuuu
09Ch	ADRESH	A/D Result Re	gister High							XXXX XXXX	uuuu uuuu
09Dh	ADCON0	ADRMD			CHS<4:0>			GO/DONE	ADON	0000 0000	0000 0000
09Eh	ADCON1	ADFM		ADCS<2:0>		—	ADNREF	ADPRE	F<1:0>	0000 -000	0000 -000
09Fh	ADCON2	TRIGSEL<3:0> CHSN<3:0>							000000	000000	

#### SPECIAL FUNCTION REGISTER SUMMARY **TABLE 3-8:**

 ${\rm x}$  = unknown,  ${\rm u}$  = unchanged,  ${\rm q}$  = value depends on condition, - = unimplemented, read as '0',  ${\rm r}$  = reserved. Shaded locations are unimplemented, read as '0'. These registers can be addressed from any bank. Unimplemented, read as '1'. PIC16F1782/3 only. Legend:

Note

1: 2: 3:

# PIC16(L)F1782/3

TABLE 3-8:	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	Value on all other
										PUR, BUR	Resets
Ban	Jank 31										
F8Ch to FE3h	_	Unimplemente	d							_	_
FE4h	STATUS_ SHAD	—	—	—	—	—	Z	DC	С	xxx	uuu
FE5h	WREG_SHAD	Working Regis	ter Shadow							XXXX XXXX	uuuu uuuu
FE6h	BSR_SHAD	—	-	-	Bank Select R	egister Shadov	v			x xxxx	u uuuu
FE7h	PCLATH_ SHAD	—	Program Counter Latch High Register Shadow						-xxx xxxx	սսսս սսսս	
FE8h	FSR0L_SHAD	Indirect Data N	ndirect Data Memory Address 0 Low Pointer Shadow							XXXX XXXX	uuuu uuuu
FE9h	FSR0H_ SHAD	Indirect Data Memory Address 0 High Pointer Shadow					****	սսսս սսսս			
FEAh	FSR1L_SHAD	Indirect Data N	lemory Addre	ess 1 Low Point	er Shadow					XXXX XXXX	uuuu uuuu
FEBh	FSR1H_ SHAD	Indirect Data Memory Address 1 High Pointer Shadow					****	uuuu uuuu			
FECh	_	Unimplemented					—	—			
FEDh	STKPTR	—	—	—	Current Stack	Pointer				1 1111	1 1111
FEEh	TOSL	Top of Stack L	ow byte							XXXX XXXX	uuuu uuuu
FEFh	TOSH	_	Top of Stack	High byte						-xxx xxxx	-uuu uuuu

x = unknown, u = unchanged, q = value depends on condition, - = unimplemented, read as '0', r = reserved. Shaded locations are unimplemented, read as '0'. These registers can be addressed from any bank. Unimplemented, read as '1'. PIC16F1782/3 only. Legend:

Note

1:

2: 3:

#### 3.6.1 TRADITIONAL DATA MEMORY

The traditional data memory is a region from FSR address 0x000 to FSR address 0xFFF. The addresses correspond to the absolute addresses of all SFR, GPR and common registers.





# 6.0 OSCILLATOR MODULE (WITH FAIL-SAFE CLOCK MONITOR)

# 6.1 Overview

The oscillator module has a wide variety of clock sources and selection features that allow it to be used in a wide range of applications while maximizing performance and minimizing power consumption. Figure 6-1 illustrates a block diagram of the oscillator module.

Clock sources can be supplied from external oscillators, quartz crystal resonators, ceramic resonators and Resistor-Capacitor (RC) circuits. In addition, the system clock source can be supplied from one of two internal oscillators and PLL circuits, with a choice of speeds selectable via software. Additional clock features include:

- Selectable system clock source between external or internal sources via software.
- Two-Speed Start-up mode, which minimizes latency between external oscillator start-up and code execution.
- Fail-Safe Clock Monitor (FSCM) designed to detect a failure of the external clock source (LP, XT, HS, EC or RC modes) and switch automatically to the internal oscillator.
- Oscillator Start-up Timer (OST) ensures stability of crystal oscillator sources

The oscillator module can be configured in one of eight clock modes.

- 1. ECL External Clock Low-Power mode (0 MHz to 0.5 MHz)
- 2. ECM External Clock Medium-Power mode (0.5 MHz to 4 MHz)
- 3. ECH External Clock High-Power mode (4 MHz to 32 MHz)
- 4. LP 32 kHz Low-Power Crystal mode.
- 5. XT Medium Gain Crystal or Ceramic Resonator Oscillator mode (up to 4 MHz)
- 6. HS High Gain Crystal or Ceramic Resonator mode (4 MHz to 20 MHz)
- 7. RC External Resistor-Capacitor (RC).
- 8. INTOSC Internal oscillator (31 kHz to 32 MHz).

Clock Source modes are selected by the FOSC<2:0> bits in the Configuration Words. The FOSC bits determine the type of oscillator that will be used when the device is first powered.

The EC clock mode relies on an external logic level signal as the device clock source. The LP, XT, and HS clock modes require an external crystal or resonator to be connected to the device. Each mode is optimized for a different frequency range. The RC clock mode requires an external resistor and capacitor to set the oscillator frequency.

The INTOSC internal oscillator block produces low, medium, and high-frequency clock sources, designated LFINTOSC, MFINTOSC and HFINTOSC. (see Internal Oscillator Block, Figure 6-1). A wide selection of device clock frequencies may be derived from these three clock sources.

### 6.2.2.1 HFINTOSC

The High-Frequency Internal Oscillator (HFINTOSC) is a factory calibrated 16 MHz internal clock source. The frequency of the HFINTOSC can be altered via software using the OSCTUNE register (Register 6-3).

The output of the HFINTOSC connects to a postscaler and multiplexer (see Figure 6-1). One of multiple frequencies derived from the HFINTOSC can be selected via software using the IRCF<3:0> bits of the OSCCON register. See Section 6.2.2.7 "Internal Oscillator Clock Switch Timing" for more information.

The HFINTOSC is enabled by:

- Configure the IRCF<3:0> bits of the OSCCON register for the desired HF frequency, and
- FOSC<2:0> = 100, or
- Set the System Clock Source (SCS) bits of the OSCCON register to '1x'.

A fast startup oscillator allows internal circuits to power up and stabilize before switching to HFINTOSC.

The High Frequency Internal Oscillator Ready bit (HFIOFR) of the OSCSTAT register indicates when the HFINTOSC is running.

The High Frequency Internal Oscillator Status Locked bit (HFIOFL) of the OSCSTAT register indicates when the HFINTOSC is running within 2% of its final value.

The High Frequency Internal Oscillator Stable bit (HFIOFS) of the OSCSTAT register indicates when the HFINTOSC is running within 0.5% of its final value.

#### 6.2.2.2 MFINTOSC

The Medium-Frequency Internal Oscillator (MFINTOSC) is a factory calibrated 500 kHz internal clock source. The frequency of the MFINTOSC can be altered via software using the OSCTUNE register (Register 6-3).

The output of the MFINTOSC connects to a postscaler and multiplexer (see Figure 6-1). One of nine frequencies derived from the MFINTOSC can be selected via software using the IRCF<3:0> bits of the OSCCON register. See Section 6.2.2.7 "Internal Oscillator Clock Switch Timing" for more information.

The MFINTOSC is enabled by:

- Configure the IRCF<3:0> bits of the OSCCON register for the desired HF frequency, and
- FOSC<2:0> = 100, or
- Set the System Clock Source (SCS) bits of the OSCCON register to '1x'

The Medium Frequency Internal Oscillator Ready bit (MFIOFR) of the OSCSTAT register indicates when the MFINTOSC is running.

#### 6.2.2.3 Internal Oscillator Frequency Adjustment

The 500 kHz internal oscillator is factory calibrated. This internal oscillator can be adjusted in software by writing to the OSCTUNE register (Register 6-3). Since the HFINTOSC and MFINTOSC clock sources are derived from the 500 kHz internal oscillator a change in the OSCTUNE register value will apply to both.

The default value of the OSCTUNE register is '0'. The value is a 6-bit two's complement number. A value of 1Fh will provide an adjustment to the maximum frequency. A value of 20h will provide an adjustment to the minimum frequency.

When the OSCTUNE register is modified, the oscillator frequency will begin shifting to the new frequency. Code execution continues during this shift. There is no indication that the shift has occurred.

OSCTUNE does not affect the LFINTOSC frequency. Operation of features that depend on the LFINTOSC clock source frequency, such as the Power-up Timer (PWRT), Watchdog Timer (WDT), Fail-Safe Clock Monitor (FSCM) and peripherals, are *not* affected by the change in frequency.

### 6.2.2.4 LFINTOSC

The Low-Frequency Internal Oscillator (LFINTOSC) is an uncalibrated 31 kHz internal clock source.

The output of the LFINTOSC connects to a multiplexer (see Figure 6-1). Select 31 kHz, via software, using the IRCF<3:0> bits of the OSCCON register. See Section 6.2.2.7 "Internal Oscillator Clock Switch Timing" for more information. The LFINTOSC is also the frequency for the Power-up Timer (PWRT), Watchdog Timer (WDT) and Fail-Safe Clock Monitor (FSCM).

The LFINTOSC is enabled by selecting 31 kHz (IRCF<3:0> bits of the OSCCON register = 000) as the system clock source (SCS bits of the OSCCON register = 1x), or when any of the following are enabled:

- Configure the IRCF<3:0> bits of the OSCCON register for the desired LF frequency, and
- FOSC<2:0> = 100, or
- Set the System Clock Source (SCS) bits of the OSCCON register to '1x'

Peripherals that use the LFINTOSC are:

- Power-up Timer (PWRT)
- Watchdog Timer (WDT)
- Fail-Safe Clock Monitor (FSCM)

The Low-Frequency Internal Oscillator Ready bit (LFIOFR) of the OSCSTAT register indicates when the LFINTOSC is running.

### 9.1.1 WAKE-UP USING INTERRUPTS

When global interrupts are disabled (GIE cleared) and any interrupt source has both its interrupt enable bit and interrupt flag bit set, one of the following will occur:

- If the interrupt occurs **before** the execution of a SLEEP instruction
  - SLEEP instruction will execute as a NOP.
  - WDT and WDT prescaler will not be cleared
  - TO bit of the STATUS register will not be set
  - PD bit of the STATUS register will not be cleared.

- If the interrupt occurs **during or after** the execution of a **SLEEP** instruction
  - SLEEP instruction will be completely executed
  - Device will immediately wake-up from Sleep
  - WDT and WDT prescaler will be cleared
  - TO bit of the STATUS register will be set
  - PD bit of the STATUS register will be cleared.

Even if the flag bits were checked before executing a SLEEP instruction, it may be possible for flag bits to become set before the SLEEP instruction completes. To determine whether a SLEEP instruction executed, test the PD bit. If the PD bit is set, the SLEEP instruction was executed as a NOP.

CLKIN <sup>(1)</sup> CLKOUT <sup>(2)</sup>	Q1 Q2 Q3 Q4 /~_/~_/~_/ //	Q1 Q2 Q3  Q4 /~_/~_/	Q1	T1osc <sup>(3</sup>	Q1 Q2 Q3 Q4 /~_/~_/~_/ )/	Q1 Q2 Q3 Q4 /~_/~_/~_/ //	Q1  Q2  Q3  Q4 /~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~	Q1 Q2 Q3 Q4 
Interrupt flag	ı i 	י י י	·/		Interrupt Laten	су <sup>(4)</sup>	· - -	
GIE bit (INTCON reg.)	' '	<u>.</u> 	Processor in	· · ·		<u>.</u>	<u>.                                    </u>	
Instruction Flow			; ; {/			; ; {/		
PC,		X PC+1	<u>X PC</u>	+2	<u>χ PC+2</u>	<u>X PC+2</u>	<u>X 0004n</u>	<u>x 0005n</u>
Instruction { Fetched	Inst(PC) = Sleep	Inst(PC + 1)	1 1		Inst(PC + 2)	1 1	Inst(0004h)	Inst(0005h)
Instruction { Executed {	Inst(PC - 1)	Sleep	1 1 1		Inst(PC + 1)	Forced NOP	Forced NOP	Inst(0004h)
Note 1: E 2: ( 3: 1 4: (	External clock. Hig CLKOUT is shown T1osc; See Sectio GIE = 1 assumed.	h, Medium, Low n here for timing re on 30.0 "Electrica In this case after v	node assumed ference. I <b>I Specificatio</b> wake-up, the	d. <b>ons</b> ". processo	r calls the ISR at 0	0004h. If GIE = 0,	execution will cont	tinue in-line.

#### FIGURE 9-1: WAKE-UP FROM SLEEP THROUGH INTERRUPT

# 13.7 PORTC Registers

#### 13.7.1 DATA REGISTER

PORTC is an 8-bit wide bidirectional port. The corresponding data direction register is TRISC (Register 13-19). Setting a TRISC bit (= 1) will make the corresponding PORTC pin an input (i.e., put the corresponding output driver in a High-Impedance mode). Clearing a TRISC bit (= 0) will make the corresponding PORTC pin an output (i.e., enable the output driver and put the contents of the output latch on the selected pin). Example 13-1 shows how to initialize an I/O port.

Reading the PORTC register (Register 13-18) reads the status of the pins, whereas writing to it will write to the PORT latch. All write operations are read-modify-write operations. Therefore, a write to a port implies that the port pins are read, this value is modified and then written to the PORT data latch (LATC).

### 13.7.2 DIRECTION CONTROL

The TRISC register (Register 13-19) controls the PORTC pin output drivers, even when they are being used as analog inputs. The user should ensure the bits in the TRISC register are maintained set when using them as analog inputs. I/O pins configured as analog inputs always read '0'.

### 13.7.3 OPEN DRAIN CONTROL

The ODCONC register (Register 13-22) controls the open-drain feature of the port. Open drain operation is independently selected for each pin. When an ODCONC bit is set, the corresponding port output becomes an open drain driver capable of sinking current only. When an ODCONC bit is cleared, the corresponding port output pin is the standard push-pull drive capable of sourcing and sinking current.

### 13.7.4 SLEW RATE CONTROL

The SLRCONC register (Register 13-23) controls the slew rate option for each port pin. Slew rate control is independently selectable for each port pin. When an SLRCONC bit is set, the corresponding port pin drive is slew rate limited. When an SLRCONC bit is cleared, The corresponding port pin drive slews at the maximum rate possible.

### 13.7.5 INPUT THRESHOLD CONTROL

The INLVLC register (Register 13-24) controls the input voltage threshold for each of the available PORTC input pins. A selection between the Schmitt Trigger CMOS or the TTL Compatible thresholds is available. The input threshold is important in determining the value of a read of the PORTC register and also the level at which an interrupt-on-change occurs, if that feature is enabled. See Section TABLE 30-1: "Supply

Voltage" for more information on threshold levels.

Note: Changing the input threshold selection should be performed while all peripheral modules are disabled. Changing the threshold level during the time a module is active may inadvertently generate a transition associated with an input pin, regardless of the actual voltage level on that pin.

### 13.7.6 PORTC FUNCTIONS AND OUTPUT PRIORITIES

Each PORTC pin is multiplexed with other functions. The pins, their combined functions and their output priorities are shown in Table 13-7.

When multiple outputs are enabled, the actual pin control goes to the peripheral with the highest priority. Analog input and some digital input functions are not included in the list below. These input functions can remain active when the pin is configured as an output. Certain digital input functions override other port functions and are included in the priority list.

Pin Name	Function Priority <sup>(1)</sup>
RC0	T1OSO PSMC1A RC0
RC1	PSMC1B CCP2 RC1
RC2	PSMC1C CCP1 RC2
RC3	PSMC1D SCL SCK RC3
RC4	PSMC1E SDA RC4
RC5	PSMC1F SDO RC5
RC6	PSMC2A TX/CK RC6
RC7	PSMC2B DT RC7

TABLE 13-7: PORTC OUTPUT PRIORITY

Note 1: Priority listed from highest to lowest.

# 17.2 ADC Operation

### 17.2.1 STARTING A CONVERSION

To enable the ADC module, the ADON bit of the ADCON0 register must be set to a '1'. Setting the GO/DONE bit of the ADCON0 register to a '1' will clear the ADRESH and ADRESL registers and start the Analog-to-Digital conversion.

Note:	The GO/DONE bit should not be set in the
	same instruction that turns on the ADC.
	Refer to Section 17.2.6 "A/D Conversion
	Procedure".

# 17.2.2 COMPLETION OF A CONVERSION

When the conversion is complete, the ADC module will:

- Clear the GO/DONE bit
- Set the ADIF Interrupt Flag bit

#### 17.2.3 TERMINATING A CONVERSION

When a conversion is terminated before completion by clearing the GO/DONE bit then the partial results are discarded and the results in the ADRESH and ADRESL registers from the previous conversion remain unchanged.

**Note:** A device Reset forces all registers to their Reset state. Thus, the ADC module is turned off and any pending conversion is terminated.

### 17.2.4 ADC OPERATION DURING SLEEP

The ADC module can operate during Sleep. This requires the ADC clock source to be set to the FRC option. When the FRC oscillator source is selected, the ADC waits one additional instruction before starting the conversion. This allows the SLEEP instruction to be executed, which can reduce system noise during the conversion. If the ADC interrupt is enabled, the device will wake-up from Sleep when the conversion completes. If the ADC interrupt is disabled, the ADC module is turned off after the conversion completes, although the ADON bit remains set.

When the ADC clock source is something other than FRC, a SLEEP instruction causes the present conversion to be aborted and the ADC module is turned off, although the ADON bit remains set.

### 17.2.5 AUTO-CONVERSION TRIGGER

The Auto-conversion Trigger allows periodic ADC measurements without software intervention. When a rising edge of the selected source occurs, the GO/DONE bit is set by hardware.

The Auto-conversion Trigger source is selected with the TRIGSEL<3:0> bits of the ADCON2 register.

Using the Auto-conversion Trigger does not assure proper ADC timing. It is the user's responsibility to ensure that the ADC timing requirements are met.

Auto-conversion sources are:

- CCP1
- CCP2
- PSMC1<sup>(1)</sup>
- PSMC2<sup>(1)</sup>

**Note:** The PSMC clock frequency, after the prescaler, must be less than Fosc/4 to ensure that the ADC detects the auto-conversion trigger. This limitation can be overcome by synchronizing a slave PSMC, running at the required slower clock frequency, to the first PSMC and triggering the conversion from the slave PSMC.

# 23.0 TIMER2 MODULE

The Timer2 module incorporates the following features:

- 8-bit Timer and Period registers (TMR2 and PR2, respectively)
- Readable and writable (both registers)
- Software programmable prescaler (1:1, 1:4, 1:16, and 1:64)
- Software programmable postscaler (1:1 to 1:16)
- Interrupt on TMR2 match with PR2
- Optional use as the shift clock for the MSSP module

See Figure 23-1 for a block diagram of Timer2.



### FIGURE 23-1: TIMER2 BLOCK DIAGRAM

# 25.0 CAPTURE/COMPARE/PWM MODULES

The Capture/Compare/PWM module is a peripheral that allows the user to time and control different events, and to generate Pulse-Width Modulation (PWM) signals. In Capture mode, the peripheral allows the timing of the duration of an event. The Compare mode allows the user to trigger an external event when a predetermined amount of time has expired. The PWM mode can generate Pulse-Width Modulated signals of varying frequency and duty cycle.

This family of devices contains two standard Capture/Compare/PWM modules (CCP1 and CCP2).

The Capture and Compare functions are identical for all CCP modules.

- Note 1: In devices with more than one CCP module, it is very important to pay close attention to the register names used. A number placed after the module acronym is used to distinguish between separate modules. For example, the CCP1CON and CCP2CON control the same operational aspects of two completely different CCP modules.
  - 2: Throughout this section, generic references to a CCP module in any of its operating modes may be interpreted as being equally applicable to CCPx module. Register names, module signals, I/O pins, and bit names may use the generic designator 'x' to indicate the use of a numeral to distinguish a particular module, when required.

When one device is transmitting a logical one, or letting the line float, and a second device is transmitting a logical zero, or holding the line low, the first device can detect that the line is not a logical one. This detection, when used on the SCL line, is called clock stretching. Clock stretching gives slave devices a mechanism to control the flow of data. When this detection is used on the SDA line, it is called arbitration. Arbitration ensures that there is only one master device communicating at any single time.

# 26.3.1 CLOCK STRETCHING

When a slave device has not completed processing data, it can delay the transfer of more data through the process of clock stretching. An addressed slave device may hold the SCL clock line low after receiving or sending a bit, indicating that it is not yet ready to continue. The master that is communicating with the slave will attempt to raise the SCL line in order to transfer the next bit, but will detect that the clock line has not yet been released. Because the SCL connection is open-drain, the slave has the ability to hold that line low until it is ready to continue communicating.

Clock stretching allows receivers that cannot keep up with a transmitter to control the flow of incoming data.

# 26.3.2 ARBITRATION

Each master device must monitor the bus for Start and Stop bits. If the device detects that the bus is busy, it cannot begin a new message until the bus returns to an Idle state.

However, two master devices may try to initiate a transmission on or about the same time. When this occurs, the process of arbitration begins. Each transmitter checks the level of the SDA data line and compares it to the level that it expects to find. The first transmitter to observe that the two levels do not match, loses arbitration, and must stop transmitting on the SDA line.

For example, if one transmitter holds the SDA line to a logical one (lets it float) and a second transmitter holds it to a logical zero (pulls it low), the result is that the SDA line will be low. The first transmitter then observes that the level of the line is different than expected and concludes that another transmitter is communicating.

The first transmitter to notice this difference is the one that loses arbitration and must stop driving the SDA line. If this transmitter is also a master device, it also must stop driving the SCL line. It then can monitor the lines for a Stop condition before trying to reissue its transmission. In the meantime, the other device that has not noticed any difference between the expected and actual levels on the SDA line continues with its original transmission. It can do so without any complications, because so far, the transmission appears exactly as expected with no other transmitter disturbing the message.

Slave Transmit mode can also be arbitrated, when a master addresses multiple slaves, but this is less common.

If two master devices are sending a message to two different slave devices at the address stage, the master sending the lower slave address always wins arbitration. When two master devices send messages to the same slave address, and addresses can sometimes refer to multiple slaves, the arbitration process must continue into the data stage.

Arbitration usually occurs very rarely, but it is a necessary process for proper multi-master support.

# PIC16(L)F1782/3



#### 26.6.6 I<sup>2</sup>C MASTER MODE TRANSMISSION

Transmission of a data byte, a 7-bit address or the other half of a 10-bit address is accomplished by simply writing a value to the SSPBUF register. This action will set the Buffer Full flag bit, BF and allow the Baud Rate Generator to begin counting and start the next transmission. Each bit of address/data will be shifted out onto the SDA pin after the falling edge of SCL is asserted. SCL is held low for one Baud Rate Generator rollover count (TBRG). Data should be valid before SCL is released high. When the SCL pin is released high, it is held that way for TBRG. The data on the SDA pin must remain stable for that duration and some hold time after the next falling edge of SCL. After the eighth bit is shifted out (the falling edge of the eighth clock), the BF flag is cleared and the master releases SDA. This allows the slave device being addressed to respond with an ACK bit during the ninth bit time if an address match occurred, or if data was received properly. The status of ACK is written into the ACKSTAT bit on the rising edge of the ninth clock. If the master receives an Acknowledge, the Acknowledge Status bit, ACKSTAT, is cleared. If not, the bit is set. After the ninth clock, the SSP1IF bit is set and the master clock (Baud Rate Generator) is suspended until the next data byte is loaded into the SSPBUF, leaving SCL low and SDA unchanged (Figure 26-27).

After the write to the SSPBUF, each bit of the address will be shifted out on the falling edge of SCL until all seven address bits and the R/W bit are completed. On the falling edge of the eighth clock, the master will release the SDA pin, allowing the slave to respond with an Acknowledge. On the falling edge of the ninth clock, the master will sample the SDA pin to see if the address was recognized by a slave. The status of the ACK bit is loaded into the ACKSTAT Status bit of the SSPCON2 register. Following the falling edge of the ninth clock transmission of the address, the SSP1IF is set, the BF flag is cleared and the Baud Rate Generator is turned off until another write to the SSPBUF takes place, holding SCL low and allowing SDA to float.

#### 26.6.6.1 BF Status Flag

In Transmit mode, the BF bit of the SSPSTAT register is set when the CPU writes to SSPBUF and is cleared when all 8 bits are shifted out.

#### 26.6.6.2 WCOL Status Flag

If the user writes the SSPBUF when a transmit is already in progress (i.e., SSPSR is still shifting out a data byte), the WCOL is set and the contents of the buffer are unchanged (the write does not occur).

WCOL must be cleared by software before the next transmission.

#### 26.6.6.3 ACKSTAT Status Flag

In Transmit mode, the ACKSTAT bit of the SSPCON2 register is cleared when the slave has sent an Acknowledge (ACK = 0) and is set when the slave does not Acknowledge (ACK = 1). A slave sends an Acknowledge when it has recognized its address (including a general call), or when the slave has properly received its data.

26.6.6.4 Typical transmit sequence:

- 1. The user generates a Start condition by setting the SEN bit of the SSPCON2 register.
- 2. SSP1IF is set by hardware on completion of the Start.
- 3. SSP1IF is cleared by software.
- 4. The MSSP module will wait the required start time before any other operation takes place.
- 5. The user loads the SSPBUF with the slave address to transmit.
- 6. Address is shifted out the SDA pin until all 8 bits are transmitted. Transmission begins as soon as SSPBUF is written to.
- The MSSP module shifts in the ACK bit from the slave device and writes its value into the ACKSTAT bit of the SSPCON2 register.
- 8. The MSSP module generates an interrupt at the end of the ninth clock cycle by setting the SSP1IF bit.
- 9. The user loads the SSPBUF with eight bits of data.
- 10. Data is shifted out the SDA pin until all 8 bits are transmitted.
- 11. The MSSP module shifts in the ACK bit from the slave device and writes its value into the ACKSTAT bit of the SSPCON2 register.
- 12. Steps 8-11 are repeated for all transmitted data bytes.
- The user generates a Stop or Restart condition by setting the PEN or RSEN bits of the SSP-CON2 register. Interrupt is generated once the Stop/Restart condition is complete.

#### REGISTER 26-1: SSPSTAT: SSP STATUS REGISTER (CONTINUED)

bit 0

BF: Buffer Full Status bit

- Receive (SPI and I<sup>2</sup>C modes):
- 1 = Receive complete, SSPBUF is full
  0 = Receive not complete, SSPBUF is empty
- <u>Transmit (I<sup>2</sup>C mode only):</u>
- 1 = Data transmit in progress (does not include the ACK and Stop bits), SSPBUF is full
- 0 = Data transmit complete (does not include the ACK and Stop bits), SSPBUF is empty

# 28.0 IN-CIRCUIT SERIAL PROGRAMMING™ (ICSP™)

ICSP<sup>™</sup> programming allows customers to manufacture circuit boards with unprogrammed devices. Programming can be done after the assembly process, allowing the device to be programmed with the most recent firmware or a custom firmware. Five pins are needed for ICSP<sup>™</sup> programming:

- ICSPCLK
- ICSPDAT
- MCLR/VPP
- VDD
- Vss

In Program/Verify mode the program memory, user IDs and the Configuration Words are programmed through serial communications. The ICSPDAT pin is a bidirectional I/O used for transferring the serial data and the ICSPCLK pin is the clock input. For more information on ICSP™ refer to the "*PIC16(L)F178X Memory Programming Specification*" (DS41457).

# 28.1 High-Voltage Programming Entry Mode

The device is placed into High-Voltage Programming Entry mode by holding the ICSPCLK and ICSPDAT pins low then raising the voltage on MCLR/VPP to VIHH.

# 28.2 Low-Voltage Programming Entry Mode

The Low-Voltage Programming Entry mode allows the PIC<sup>®</sup> Flash MCUs to be programmed using VDD only, without high voltage. When the LVP bit of Configuration Words is set to '1', the low-voltage ICSP programming entry is enabled. To disable the Low-Voltage ICSP mode, the LVP bit must be programmed to '0'.

Entry into the Low-Voltage Programming Entry mode requires the following steps:

- 1. MCLR is brought to VIL.
- 2. A 32-bit key sequence is presented on ICSPDAT, while clocking ICSPCLK.

Once the key sequence is complete,  $\overline{\text{MCLR}}$  must be held at VIL for as long as Program/Verify mode is to be maintained.

If low-voltage programming is enabled (LVP = 1), the  $\overline{\text{MCLR}}$  Reset function is automatically enabled and cannot be disabled. See **Section 5.5 "MCLR"** for more information.

The LVP bit can only be reprogrammed to '0' by using the High-Voltage Programming mode.

# 28.3 Common Programming Interfaces

Connection to a target device is typically done through an ICSP<sup>™</sup> header. A commonly found connector on development tools is the RJ-11 in the 6P6C (6-pin, 6 connector) configuration. See Figure 28-1.





Another connector often found in use with the PICkit<sup>™</sup> programmers is a standard 6-pin header with 0.1 inch spacing. Refer to Figure 28-2.

For additional interface recommendations, refer to your specific device programmer manual prior to PCB design.

It is recommended that isolation devices be used to separate the programming pins from other circuitry. The type of isolation is highly dependent on the specific application and may include devices such as resistors, diodes, or even jumpers. See Figure 28-3 for more information.









Note: Unless otherwise noted, VIN = 5V, Fosc = 300 kHz, CIN = 0.1  $\mu$ F, TA = 25°C.



FIGURE 31-55: VOH vs. IOH Over Temperature, VDD = 1.8V, PIC16LF1782/3 Only.



Temperature, VDD = 1.8V, PIC16LF1782/3 Only.



FIGURE 31-57: LFINTOSC Frequency, PIC16LF1782/3 Only.



FIGURE 31-58: LFINTOSC Frequency, PIC16F1782/3 Only.



PIC16F1782/3 Only.



FIGURE 31-60: WDT Time-Out Period PIC16LF1782/3 Only.

# 32.11 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<sup>™</sup> and dsPICDEM<sup>™</sup> demonstration/development board series of circuits, Microchip has a line of evaluation kits and demonstration software for analog filter design, KEELOQ<sup>®</sup> security ICs, CAN, IrDA<sup>®</sup>, PowerSmart battery management, SEEVAL<sup>®</sup> 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.

# 32.12 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<sup>®</sup> and Olimex
- Embedded Ethernet Solutions from companies, such as EZ Web Lynx, WIZnet and IPLogika<sup>®</sup>

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



	MILLIMETERS				
Dimension	MIN	NOM	MAX		
Contact Pitch	E	0.65 BSC			
Contact Pad Spacing	С		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