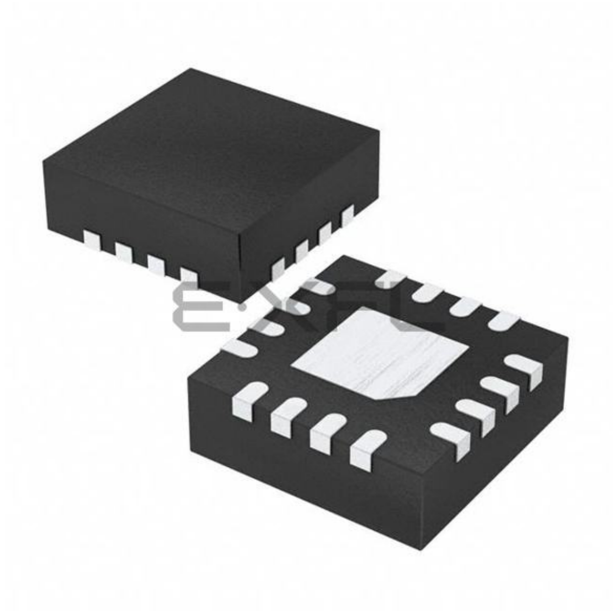
Microchip Technology - PIC16F1764T-I/ML Datasheet



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	32MHz
Connectivity	I²C, LINbus, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, POR, PWM, WDT
Number of I/O	12
Program Memory Size	7KB (4K x 14)
Program Memory Type	FLASH
EEPROM Size	128 x 8
RAM Size	512 x 8
Voltage - Supply (Vcc/Vdd)	2.3V ~ 5.5V
Data Converters	A/D 8x10b; D/A 1x5b, 1x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	16-VQFN Exposed Pad
Supplier Device Package	16-QFN (4x4)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic16f1764t-i-ml

1.1 Register and Bit Naming Conventions

1.1.1 REGISTER NAMES

When there are multiple instances of the same peripheral in a device, the peripheral control registers will be depicted as the concatenation of a peripheral identifier, peripheral instance and control identifier. The control registers section will show just one instance of all the register names with an 'x' in the place of the peripheral instance number. This naming convention may also be applied to peripherals when there is only one instance of that peripheral in the device to maintain compatibility with other devices in the family that contain more than one.

1.1.2 BIT NAMES

There are two variants for bit names:

- Short name: Bit function abbreviation
- Long name: Peripheral abbreviation + short name

1.1.2.1 Short Bit Names

Short bit names are an abbreviation for the bit function. For example, some peripherals are enabled with the EN bit. The bit names shown in the registers are the short name variant.

Short bit names are useful when accessing bits in C programs. The general format for accessing bits by the short name is *RegisterName*bits.*ShortName*. For example, the enable bit, EN, in the COG1CON0 register can be set in C programs with the instruction `COG1CON0bits.EN = 1`.

Short names are generally not useful in assembly programs because the same name may be used by different peripherals in different bit positions. When this occurs, during the include file generation, all instances of that short bit name are appended with an underscore plus the name of the register in which the bit resides to avoid naming contentions.

1.1.2.2 Long Bit Names

Long bit names are constructed by adding a peripheral abbreviation prefix to the short name. The prefix is unique to the peripheral thereby making every long bit name unique. The long bit name for the COG1 enable bit is the COG1 prefix, G1, appended with the enable bit short name, EN, resulting in the unique bit name G1EN.

Long bit names are useful in both C and assembly programs. For example, in C the COG1CON0 enable bit can be set with the `G1EN = 1` instruction. In assembly, this bit can be set with the `BSF COG1CON0,G1EN` instruction.

1.1.2.3 Bit Fields

Bit fields are two or more adjacent bits in the same register. Bit fields adhere only to the short bit naming convention. For example, the three Least Significant bits of the COG1CON0 register contain the mode control bits. The short name for this field is MD. There is no long bit name variant. Bit field access is only possible in C programs. The following example demonstrates a C program instruction for setting the COG1 to the Push-Pull mode:

```
COG1CON0bits.MD = 0x5;
```

Individual bits in a bit field can also be accessed with long and short bit names. Each bit is the field name appended with the number of the bit position within the field. For example, the Most Significant mode bit has the short bit name MD2, and the long bit name is G1MD2. The following two examples demonstrate assembly program sequences for setting the COG1 to Push-Pull mode:

Example 1:

```
MOVLW  ~(1<<G1MD1)
ANDWF  COG1CON0,F
MOVLW  1<<G1MD2 | 1<<G1MD0
IORWF  COG1CON0,F
```

Example 2:

```
BSF    COG1CON0,G1MD2
BCF    COG1CON0,G1MD1
BSF    COG1CON0,G1MD0
```

1.1.3 REGISTER AND BIT NAMING EXCEPTIONS

1.1.3.1 Status, Interrupt and Mirror Bits

Status, interrupt enables, interrupt flags and mirror bits are contained in registers that span more than one peripheral. In these cases, the bit name shown is unique, so there is no prefix or short name variant.

1.1.3.2 Legacy Peripherals

There are some peripherals that do not strictly adhere to these naming conventions. Peripherals that have existed for many years and are present in almost every device are the exceptions. These exceptions were necessary to limit the adverse impact of the new conventions on legacy code. Peripherals that do adhere to the new convention will include a table in the registers section indicating the long name prefix for each peripheral instance. Peripherals that fall into the exception category will not have this table. These peripherals include, but are not limited to, the following:

- EUSART
- MSSP

TABLE 3-16: SPECIAL FUNCTION REGISTER 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											
00Ch	PORTA	—	—	RA<5:0>						`--xx xxxx`	`--uu uuuu`
00Dh	PORTB(2)	RB<7:4>				—	—	—	—	`xxxx ----`	`uuuu ----`
00Eh	PORTC	RC<7:6>(2)		RC<5:0>						`xxxx xxxx`	`uuuu uuuu`
00Fh	—	Unimplemented								—	—
010h	—	Unimplemented								—	—
011h	PIR1	TMR1GIF	ADIF	RCIF	TXIF	SSP1IF	CCP1IF	TMR2IF	TMR1IF	`0000 0000`	`0000 0000`
012h	PIR2	OSFIF	C2IF	C1IF	—	BCL1IF	C4IF(2)	C3IF(2)	CCP2IF(2)	`000- 0000`	`000- 0000`
013h	PIR3	PWM6IF(2)	PWM5IF	COG1IF	ZCDIF	COG2IF(2)	CLC3IF	CLC2IF	CLC1IF	`0000 0000`	`0000 0000`
014h	PIR4	—	—	TMR5GIF	TMR5IF	TMR3GIF	TMR3IF	TMR6IF	TRM4IF	`--00 0000`	`--00 0000`
015h	TMR0	Timer0 Module Register								`0000 0000`	`0000 0000`
016h	TMR1L	Holding Register for the Least Significant Byte of the 16-Bit TMR1 Register								`xxxx xxxx`	`uuuu uuuu`
017h	TMR1H	Holding Register for the Most Significant Byte of the 16-Bit TMR1 Register								`xxxx xxxx`	`uuuu uuuu`
018h	T1CON	CS<1:0>		CKPS<1:0>		OSCEN	$\overline{\text{SYNC}}$	—	ON	`0000 00-0`	`uuuu uu-u`
019h	T1GCON	GE	GPOL	GTM	GSPM	GGO/$\overline{\text{DONE}}$	GVAL	GSS<1:0>		`0000 0x00`	`uuuu uxuu`
01Ah	T2TMR	Holding Register for the 8-Bit TMR2 Register								`0000 0000`	`0000 0000`
01Bh	T2PR	TMR2 Period Register								`1111 1111`	`1111 1111`
01Ch	T2CON	ON	CKPS<2:0>			OUTPS<3:0>				`0000 0000`	`0000 0000`
01Dh	T2HLT	PSYNC	CKPOL	CKSYNC	MODE<4:0>					`0000 0000`	`0000 0000`
01Eh	T2CLKCON	—	—	—	—	CS<3:0>				`---- 0000`	`---- 0000`
01Fh	T2RST	—	—	—	—	RSEL<3:0>				`---- 0000`	`---- 0000`
Bank 1											
08Ch	TRISA	—	—	TRISA<5:4>		—(1)	TRISA<2:0>			`--11 1111`	`--11 1111`
08Dh	TRISB(2)	TRISB<7:4>				—	—	—	—	`1111 ----`	`1111 ----`
08Eh	TRISC	TRISC<7:6>(2)		TRISC<5:0>						`1111 1111`	`1111 1111`
08Fh	—	Unimplemented								—	—
090h	—	Unimplemented								—	—
091h	PIE1	TMR1GIE	ADIE	RCIE	TXIE	SSP1IE	CCP1IE	TMR2IE	TMR1IE	`0000 0000`	`0000 0000`
092h	PIE2	OSFIE	C2IE	C1IE	—	BCL1IE	C4IE(2)	C3IE(2)	CCP2IE(2)	`000- 0000`	`000- 0000`
093h	PIE3	PWM6IE(2)	PWM5IE	COG1IE	ZCDIE	COG2IE(2)	CLC3IE	CLC2IE	CLC1IE	`0000 0000`	`0000 0000`
094h	PIE4	—	—	TMR5GIE	TMR5IE	TMR3GIE	TMR3IE	TMR6IE	TRM4IE	`--00 0000`	`--00 0000`
095h	OPTION_REG	$\overline{\text{WPUEN}}$	INTEDG	TMR0CS	TMR0SE	PSA	PS<2:0>			`1111 1111`	`1111 1111`
096h	PCON	STKOVF	STKUNF	—	$\overline{\text{RWDT}}$	$\overline{\text{RMCLR}}$	$\overline{\text{RI}}$	$\overline{\text{POR}}$	$\overline{\text{BOR}}$	`00-1 11qq`	`qq-q qquu`
097h	WDTCON	—	—	WDTPS<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	SOSCR	PLLR	OSTS	HFIOFR	HFIOFL	MFIOFR	LFIOFR	HFIOFS	`00q0 0q0q`	`qqqq qq0q`
09Bh	ADRESL	ADC Result Register Low								`xxxx xxxx`	`uuuu uuuu`
09Ch	ADRESH	ADC Result Register High								`xxxx xxxx`	`uuuu uuuu`
09Dh	ADCON0	—	CHS<4:0>					GO/$\overline{\text{DONE}}$	ADON	`-000 0000`	`-000 0000`
09Eh	ADCON1	ADFM	ADCS<2:0>			—	ADNREF	ADPREF<1:0>		`0000 -000`	`0000 -000`
09Fh	ADCON2	TRIGSEL<4:0>					—	—	—	`0000 0---`	`0000 0---`

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: Unimplemented, read as '`1`'.
2: PIC16(L)F1768/9 only.
3: PIC16(L)F1764/5 only.
4: Unimplemented on PIC16LF1764/5/8/9.

TABLE 3-16: 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 14											
70Ch	—	Unimplemented								—	—
70Dh	COG2PHR(2)	—	—	COG Rising Edge Phase Delay Count Register						--00 0000	--00 0000
70Eh	COG2PHF(2)	—	—	COG Falling Edge Phase Delay Count Register						--00 0000	--00 0000
70Fh	COG2BLKR(2)	—	—	COG Rising Edge Blanking Count Register						--00 0000	--00 0000
710h	COG2BLKF(2)	—	—	COG Falling Edge Blanking Count Register						--00 0000	--00 0000
711h	COG2DBR(2)	—	—	COG Rising Edge Dead-band Count Register						--00 0000	--00 0000
712h	COG2DBF(2)	—	—	COG Falling Edge Dead-band Count Register						--00 0000	--00 0000
713h	COG2CON0(2)	EN	LD	—	CS<1:0>		MD<2:0>			00-0 0000	00-0 0000
714h	COG2CON1(2)	RDBS	FDBS	—	—	POLD	POLC	POLB	POLA	00-- 0000	00-- 0000
715h	COG2RIS0(2)	RIS<7:0>								0000 0000	0000 0000
716h	COG2RIS1(2)	RIS<15:8>								0000 0000	0000 0000
717h	COG2RSIM0(2)	RSIM<7:0>								0000 0000	0000 0000
718h	COG2RSIM1(2)	RSIM<15:8>								0000 0000	0000 0000
719h	COG2FIS0(2)	FIS<7:0>								0000 0000	0000 0000
71Ah	COG2FIS1(2)	FIS<15:8>								0000 0000	0000 0000
71Bh	COG2FSIM0(2)	FSIM<7:0>								0000 0000	0000 0000
71Ch	COG2FSIM1(2)	FSIM<15:8>								0000 0000	0000 0000
71Dh	COG2ASD0(2)	ASE	ARSEN	ASDBD<1:0>		ASDAC<1:0>		—	—	0001 01--	0001 01--
71Eh	COG2ASD1(2)	AS7E	AS6E	AS5E	AS4E	AS3E	AS2E	AS1E	AS0E	0000 0000	0000 0000
71Fh	COG2STR(2)	SDATD	SDATC	SDATB	SDATA	STRD	STRC	STRB	STRA	0000 0000	0000 0000
Bank 15											
78Ch — 793h	—	Unimplemented								—	—
794h	PRG1RTSS	—	—	—	—	RTSS<3:0>				---- 0000	---- 0000
795h	PRG1FTSS	—	—	—	—	FTSS<3:0>				---- 0000	---- 0000
796h	PRG1INS	—	—	—	—	INS<3:0>				---- 0000	---- 0000
797h	PRG1CON0	EN	—	FEDG	REDG	MODE<1:0>		OS	GO	0-00 0000	0-00 0000
798h	PRG1CON1	—	—	—	—	—	RDY	FPOL	RPOL	---- -000	---- -000
799h	PRG1CON2	—	—	—	ISET<4:0>					---0 0000	---0 0000
79Ah	PRG2RTSS(2)	—	—	—	—	RTSS<3:0>				---- 0000	---- 0000
79Bh	PRG2FTSS(2)	—	—	—	—	FTSS<3:0>				---- 0000	---- 0000
79Ch	PRG2INS(2)	—	—	—	—	INS<3:0>				---- 0000	---- 0000
79Dh	PRG2CON0(2)	EN	—	FEDG	REDG	MODE<1:0>		OS	GO	0-00 0000	0-00 0000
79Eh	PRG2CON1(2)	—	—	—	—	—	RDY	FPOL	RPOL	---- -000	---- -000
79Fh	PRG2CON2(2)	—	—	—	ISET<4:0>					---0 0000	---0 0000
Bank 16-26											
x0Ch/ x8Ch — x1Fh/ x9Fh	—	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: Unimplemented, read as ‘1’.
2: PIC16(L)F1768/9 only.
3: PIC16(L)F1764/5 only.
4: Unimplemented on PIC16LF1764/5/8/9.

REGISTER 11-3: LATA: PORTA DATA LATCH REGISTER

U-0	U-0	R/W-x/u	R/W-x/u	U-0	R/W-x/u	R/W-x/u	R/W-x/u
—	—	LATA<5:4>(1)		—	LATA<2:0>(1)		
bit 7							bit 0

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

bit 7-6 **Unimplemented:** Read as ‘0’

bit 5-4 **LATA<5:4>**: RA<5:4> Output Latch Value bits(1)

bit 3 **Unimplemented:** Read as ‘0’

bit 2-0 **LATA<2:0>**: RA<2:0> Output Latch Value bits(1)

Note 1: Writes to PORTA are actually written to the corresponding LATA register. Reads from PORTA are the return of actual I/O pin values.

REGISTER 11-4: ANSELA: PORTA ANALOG SELECT REGISTER

U-0	U-0	U-0	R/W-1/1	U-0	R/W-1/1	R/W-1/1	R/W-1/1
—	—	—	ANSA4	—	ANSA<2:0>		
bit 7							bit 0

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

bit 7-5 **Unimplemented:** Read as ‘0’

bit 4 **ANSA4**: Analog Select Between Analog or Digital Function on RA4 Pin bit
1 = Analog input; pin is assigned as an analog input, digital input buffer is disabled(1)
0 = Digital I/O; pin is assigned to port or digital special function

bit 3 **Unimplemented:** Read as ‘0’

bit 2-0 **ANSA<2:0>**: Analog Select Between Analog or Digital Function on RA<2:0> Pins bits
1 = Analog input; pin is assigned as an analog input, digital input buffer is disabled(1)
0 = Digital I/O; pin is assigned to port or digital special function

Note 1: When setting a pin to an analog input, the corresponding TRISx bit must be set to Input mode in order to allow external control of the voltage on the pin.

REGISTER 11-15: SLRCONB: PORTB SLEW RATE CONTROL REGISTER

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

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

bit 7-4 **SLRB<7:4>:** PORTB Slew Rate Enable bits
For RB<7:4> Pins:
1 = Port pin slew rate is limited
0 = Port pin slews at maximum rate

bit 3-0 **Unimplemented:** Read as ‘0’

REGISTER 11-16: INLVLB: PORTB INPUT LEVEL CONTROL REGISTER

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	U-0	U-0	U-0	U-0
INLVLB<7:4>				—	—	—	—
bit 7							bit 0

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

bit 7-4 **INLVLB<7:4>:** PORTB Input Level Select bits
For RB<7:4> Pins:
1 = Port pin digital input operates with ST thresholds
0 = Port pin digital input operates with TTL thresholds

bit 3-0 **Unimplemented:** Read as ‘0’

TABLE 11-4: SUMMARY OF REGISTERS ASSOCIATED WITH PORTB

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Register on Page
ANSELB	ANSB<7:4>				—	—	—	—	143
INLVLB	INLVLB<7:4>				—	—	—	—	145
LATB	LATB<7:4>				—	—	—	—	143
ODCONB	ODB<7:4>				—	—	—	—	144
PORTB	RB<7:4>				—	—	—	—	142
SLRCONB	SLRB<7:4>				—	—	—	—	145
TRISB	TRISB<7:4>				—	—	—	—	145
WPUB	WPUB<7:4>				—	—	—	—	144

Legend: — = unimplemented location, read as ‘0’. Shaded cells are not used by PORTB.

14.4 Register Definitions: FVR Control

REGISTER 14-1: FVRCON: FIXED VOLTAGE REFERENCE CONTROL REGISTER

R/W-0/0	R-q/q	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
FVREN	FVRRDY[1]	TSEN[3]	TSRNG[3]	CDAFVR<1:0>		ADFVR<1:0>	
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	q = Value depends on condition

bit 7 **FVREN:** Fixed Voltage Reference Enable bit

1 = Fixed Voltage Reference is enabled
0 = Fixed Voltage Reference is disabled

bit 6 **FVRRDY:** Fixed Voltage Reference Ready Flag bit[1]

1 = Fixed Voltage Reference output is ready for use
0 = Fixed Voltage Reference output is not ready or not enabled

bit 5 **TSEN:** Temperature Indicator Enable bit[3]

1 = Temperature indicator is enabled
0 = Temperature indicator is disabled

bit 4 **TSRNG:** Temperature Indicator Range Selection bit[3]

1 = $V_{OUT} = V_{DD} - 4\ V_T$ (High Range)
0 = $V_{OUT} = V_{DD} - 2\ V_T$ (Low Range)

bit 3-2 **CDAFVR<1:0>:** Comparator/DAC FVR Buffer Gain Selection bits

11 = Comparator/DAC FVR buffer gain is 4x with output, V_{CDAFVR} = 4x V_{FVR}[2]
10 = Comparator/DAC FVR buffer gain is 2x with output, V_{CDAFVR} = 2x V_{FVR}[2]
01 = Comparator/DAC FVR buffer gain is 1x with output, V_{CDAFVR} = 1x V_{FVR}
00 = Comparator/DAC FVR buffer is off

bit 1-0 **ADFVR<1:0>:** ADC FVR Buffer Gain Selection bits

11 = ADC FVR buffer gain is 4x with output, V_{ADFVR} = 4x V_{FVR}[2]
10 = ADC FVR buffer gain is 2x with output, V_{ADFVR} = 2x V_{FVR}[2]
01 = ADC FVR buffer gain is 1x with output, V_{ADFVR} = 1x V_{FVR}
00 = ADC FVR buffer is off

Note 1: FVRRDY is always ‘1’ on PIC16F1764/5/8/9 only.
2: Fixed Voltage Reference output cannot exceed V_{DD}.
3: See **Section 15.0 “Temperature Indicator Module”** for additional information.

TABLE 14-2: SUMMARY OF REGISTERS ASSOCIATED WITH FIXED VOLTAGE REFERENCE

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Register on Page
FVRCON	FVREN	FVRRDY	TSEN	TSRNG	CDAFVR<1:0>		ADFVR<1:0>		169

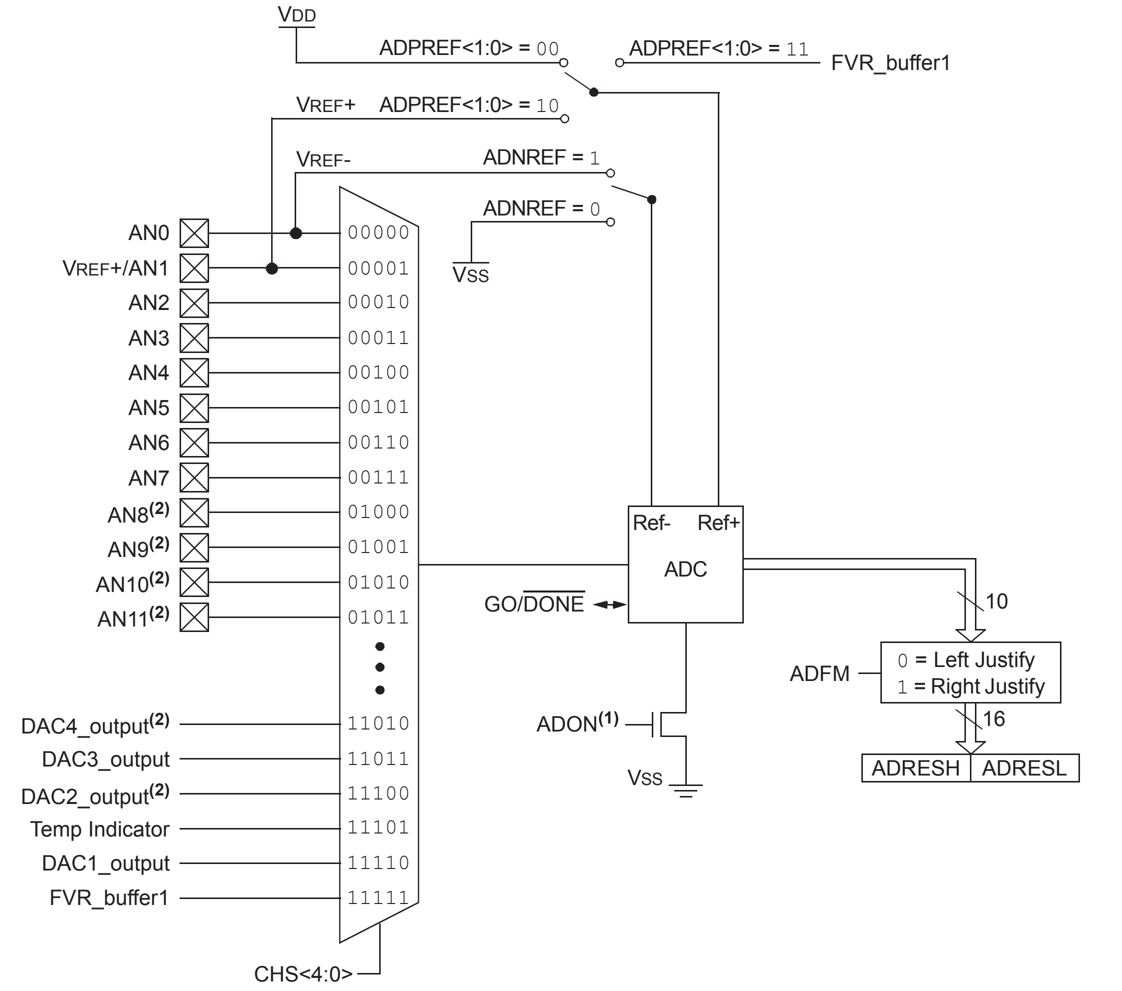
16.0 ANALOG-TO-DIGITAL CONVERTER (ADC) MODULE

The Analog-to-Digital Converter (ADC) allows conversion of an analog input signal to a 10-bit binary representation of that signal. This device uses analog inputs, which are multiplexed into a single Sample-and-Hold (S&H) circuit. The output of the Sample-and-Hold is connected to the input of the converter. The converter generates a 10-bit binary result via successive approximation and stores the conversion result into the ADC Result registers (ADRESH:ADRESL register pair). Figure 16-1 shows the block diagram of the ADC.

The ADC voltage reference is software-selectable to be either internally generated or externally supplied.

The ADC can generate an interrupt upon completion of a conversion. This interrupt can be used to wake-up the device from Sleep.

FIGURE 16-1: ADC BLOCK DIAGRAM



Note 1: When ADON = 0, all multiplexer inputs are disconnected.
2: PIC16(L)F1768/9 only

TABLE 16-1: ADC CLOCK PERIOD (TAD) VS. DEVICE OPERATING FREQUENCIES

ADC Clock Period (TAD)		Device Frequency (FOSC)					
ADC Clock Source	**ADCS<2:0>**	**32 MHz**	**20 MHz**	**16 MHz**	**8 MHz**	**4 MHz**	**1 MHz**
FOSC/2	000	62.5ns(2)	100 ns(2)	125 ns(2)	250 ns(2)	500 ns(2)	2.0 µs
FOSC/4	100	125 ns(2)	200 ns(2)	250 ns(2)	500 ns(2)	1.0 µs	4.0 µs
FOSC/8	001	0.5 µs(2)	400 ns(2)	0.5 µs(2)	1.0 µs	2.0 µs	8.0 µs(3)
FOSC/16	101	800 ns	800 ns	1.0 µs	2.0 µs	4.0 µs	16.0 µs(2)
FOSC/32	010	1.0 µs	1.6 µs	2.0 µs	4.0 µs	8.0 µs(3)	32.0 µs(2)
FOSC/64	110	2.0 µs	3.2 µs	4.0 µs	8.0 µs(3)	16.0 µs(2)	64.0 µs(2)
FRC	x11	1.0-6.0 µs(1,4)	1.0-6.0 µs(1,4)	1.0-6.0 µs(1,4)	1.0-6.0 µs(1,4)	1.0-6.0 µs(1,4)	1.0-6.0 µs(1,4)

Legend: Shaded cells are outside of the recommended range.

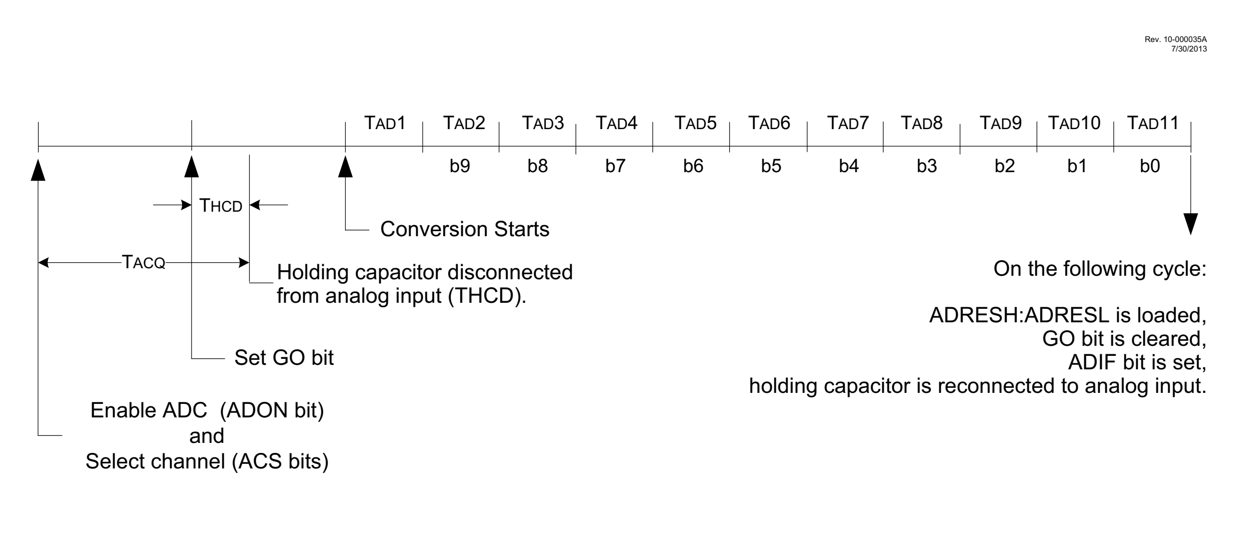
Note 1: See the TAD parameter for FRC source typical TAD value.

2: These values violate the required TAD time.

3: Outside the recommended TAD time.

4: The ADC clock period (TAD) and total ADC conversion time can be minimized when the ADC clock is derived from the system clock, FOSC. However, the FRC oscillator source must be used when conversions are to be performed with the device in Sleep mode.

FIGURE 16-2: ANALOG-TO-DIGITAL CONVERSION TAD CYCLES



16.2.6 ADC CONVERSION PROCEDURE

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

1. Configure the PORT:
   - Disable the pin output driver (refer to the TRISx register)
   - Configure pin as an analog (refer to the ANSELx register)
   - Disable weak pull-ups either globally (Refer to the OPTION_REG register) or individually (Refer to the appropriate WPUx 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 16.4 "ADC Acquisition Requirements"**.

EXAMPLE 16-1: ADC CONVERSION

```
;This code block configures the ADC
;for polling, Vdd and Vss references, FRC
;oscillator and AN0 input.
;
;Conversion start & polling for completion
; are included.
;
BANKSEL   ADCON1       ;
MOVLW     B'11110000'  ;Right justify, FRC
                       ;oscillator
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   WPUA
BCF       WPUA,0       ;Disable weak
                       ;pull-up on RA0
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
```

REGISTER 16-3: ADCON2: ADC CONTROL REGISTER 2

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	U-0	U-0	U-0
TRIGSEL<4:0>(1)					—	—	—
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 7-3 **TRIGSEL<4:0>:** Auto-Conversion Trigger Selection bits(1)

`11111` = Reserved
•
•
•
`11011` = Reserved
`11010` = PWM6 – OF6_match(2)
`11001` = PWM6 – PH6_match(2)
`11000` = PWM6 – PR6_match(2)
`10111` = PWM6 – DC6_match(2)
`10110` = PWM5 – OF5_match
`10101` = PWM5 – PH5_match
`10100` = PWM5 – PR5_match
`10011` = PWM5 – DC5_match
`10010` = PWM4 – PWM4OUT(2)
`10001` = PWM3 – PWM3OUT
`10000` = CCP2 – CCP2_trigger(2)
`01111` = CCP1 – CCP1_trigger
`01110` = CLC3 – LC3_out
`01101` = CLC2 – LC2_out
`01100` = CLC1 – LC1_out
`01011` = Comparator C4 – sync_C4OUT(2)
`01010` = Comparator C3 – sync_C3OUT(2)
`01001` = Comparator C2 – sync_C2OUT
`01000` = Comparator C1 – sync_C1OUT
`00111` = Timer6 – T6_postscaled
`00110` = Timer5 – T5_overflow
`00101` = Timer4 – T4_postscaled
`00100` = Timer3 – T3_overflow
`00011` = Timer2 – T2_postscaled
`00010` = Timer1 – T1_overflow
`00001` = Timer0 – T0_overflow
`00000` = No auto-conversion trigger selected

bit 3-0 **Unimplemented:** Read as ‘`0`’

Note 1: This is a rising edge-sensitive input for all sources.
2: PIC16(L)F1768/9 only; reserved otherwise.

TABLE 19-4: SUMMARY OF REGISTERS ASSOCIATED WITH COMPARATOR MODULE

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Register on Page
ANSELA	—	—	—	ANSA4	—	ANSA2	ANSA<1:0>		137
ANSELB(1)	ANSB<7:6>		ANSB<5:4>		—	—	—	—	143
ANSELC	ANSC7(1)	ANSC6(1)	—	—	ANSC<3:0>				148
CM1CON0	ON	OUT	—	POL	ZLF	Reserved	HYS	SYNC	202
CM2CON0	ON	OUT	—	POL	ZLF	Reserved	HYS	SYNC	202
CM3CON0(1)	ON	OUT	—	POL	ZLF	Reserved	HYS	SYNC	202
CM4CON0(1)	ON	OUT	—	POL	ZLF	Reserved	HYS	SYNC	202
CM1CON1	—	—	—	—	—	—	INTP	INTN	203
CM2CON1	—	—	—	—	—	—	INTP	INTN	203
CM3CON1(1)	—	—	—	—	—	—	INTP	INTN	203
CM4CON1(1)	—	—	—	—	—	—	INTP	INTN	203
CM1NSEL	—	—	—	—	—	NCH<2:0>			203
CM2NSEL	—	—	—	—	—	NCH<2:0>			203
CM3NSEL(1)	—	—	—	—	—	NCH<2:0>			203
CM4NSEL(1)	—	—	—	—	—	NCH<2:0>			203
CM1PSEL	—	—	—	—	PCH<3:0>				204
CM2PSEL	—	—	—	—	PCH<3:0>				204
CM3PSEL(1)	—	—	—	—	PCH<3:0>				204
CM4PSEL(1)	—	—	—	—	PCH<3:0>				204
CMOUT	—	—	—	—	MC4OUT(1)	MC3OUT(1)	MC2OUT	MC1OUT	204
FVRCON	FVREN	FVRRDY	TSEN	TSRNG	CDAFVR<1:0>		ADFVR<1:0>		169
DAC1CON0	EN	FM	OE1	—	PSS<1:0>		—	NSS	193
DAC2CON0(1)	EN	FM	OE1	—	PSS<1:0>		—	NSS	193
DAC3CON0	EN	—	OE1	—	PSS<1:0>		—	NSS	188
DAC4CON0(1)	EN	—	OE1	—	PSS<1:0>		—	NSS	188
DAC3REF	---	---	---	REF<4:0>					189
DAC4REF(1)	---	---	---	REF<4:0>					189
DAC1REFH	REF<9:x> (x Depends on FM bit)								194
DAC2REFH(1)	REF<9:x> (x Depends on FM bit)								194
DAC1REFL	REF<x-1:0> (x Depends on FM bit)								194
DAC2REFL(1)	REF<x-1:0> (x Depends on FM bit)								194
INTCON	GIE	PEIE	TMR0IE	INTE	IOCIE	TMR0IF	INTF	IOCIF	101
PIE2	OSFIE	C2IE	C1IE	—	BCL1IE	C4IE(1)	C3IE(1)	CCP2IE(1)	103
PIR2	OSFIF	C2IF	C1IF	—	BCL1IF	C4IF(1)	C3IF(1)	CCP2IF(1)	106
TRISA	—	—	TRISA<5:4>		—(2)	TRISA2	TRISA<1:0>		136
TRISB(1)	TRISB<7:6>		TRISB<5:4>		—	—	—	—	142
TRISC	TRISC<7:6>(1)		TRISC<5:4>		TRISC<3:0>				147

Legend: — = unimplemented location, read as '0'. Shaded cells are unused by the comparator module.

Note 1: PIC16(L)F1768/9 only.

2: Unimplemented, read as '1'.

23.6 Operation Examples

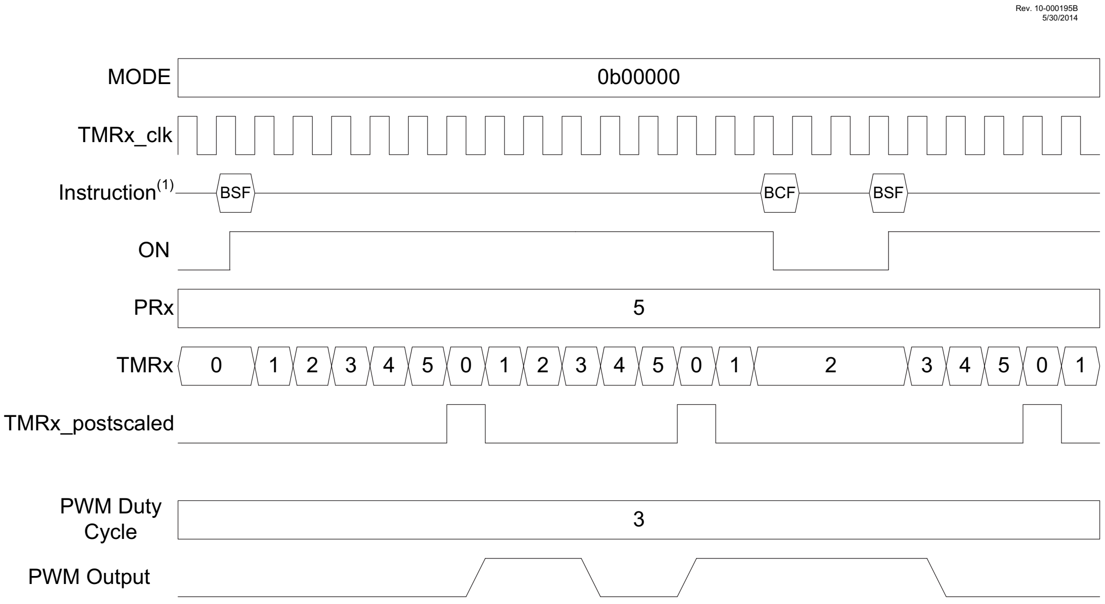
Unless otherwise specified, the following notes apply to the following timing diagrams:

- Both the prescaler and postscaler are set to 1:1 (both the CKPSx and OUTPSx bits in the TxCON register are cleared).
- The diagrams illustrate any clock except FOSC/4 and show clock-sync delays of at least two full cycles for both ON and Timer2_ers. When using FOSC/4, the clock-sync delay is at least one instruction period for Timer2_ers; ON applies in the next instruction period.
- The PWM duty cycle and PWM output are illustrated assuming that the timer is used for the PWM function of the CCP module, as described in **Section 24.6 "CCP/PWM Clock Selection"**. The signals are not a part of the Timer2 module.

23.6.1 SOFTWARE GATE MODE

This mode corresponds to legacy Timer2 operation. The timer increments with each clock input when ON = 1 and does not increment when ON = 0. When the TMRx count equals the PRx period count, the timer resets on the next clock and continues counting from 0. Operation with the ON bit software controlled is illustrated in Figure 23-4. With PRx = 5, the counter advances until TMRx = 5, and goes to zero with the next clock.

FIGURE 23-4: SOFTWARE GATE MODE TIMING DIAGRAM (MODE<4:0> = 00000)



Note 1: BSF and BCF represent Bit-Set File and Bit-Clear File instructions executed by the CPU to set or clear the ON bit of TxCON. CPU execution is asynchronous to the timer clock input.

REGISTER 26-15: PWMxTMRH: PWMx TIMER HIGH 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/W-0/0
TMR<15:8>							
bit 7							bit 0

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

bit 7-0	**TMR<15:8>**: PWMx Timer High bits Upper eight bits of PWMx timer counter.

REGISTER 26-16: PWMxTMRL: PWMx TIMER LOW 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/W-0/0
TMR<7:0>							
bit 7							bit 0

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

bit 7-0	**TMR<7:0>**: PWMx Timer Low bits Lower eight bits of PWMx timer counter.

REGISTER 27-7: COGxFIS0: COGx FALLING EVENT INPUT SELECTION REGISTER 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/W-0/0	R/W-0/0
FIS7	FIS6	FIS5	FIS4	FIS3	FIS2	FIS1	FIS0
bit 7							bit 0

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

bit 7 **FIS7:** COGx Falling Event Input Source 7 Enable bit
1 = PWM3 output is enabled as a falling event input
0 = PWM3 output has no effect on the falling event

bit 6 **FIS6:** COGx Falling Event Input Source 6 Enable bit
1 = CCP2 output is enabled as a falling event input
0 = CCP2 output has no effect on the falling event

bit 5 **FIS5:** COGx Falling Event Input Source 5 Enable bit
1 = CCP1 output is enabled as a falling event input
0 = CCP1 output has no effect on the falling event

bit 4 **FIS4:** COGx Falling Event Input Source 4 Enable bit
1 = Comparator 4 output is enabled as a falling event input
0 = Comparator 4 output has no effect on the falling event

bit 3 **FIS3:** COGx Falling Event Input Source 3 Enable bit
1 = Comparator 3 output is enabled as a falling event input
0 = Comparator 3 output has no effect on the falling event

bit 2 **FIS2:** COGx Falling Event Input Source 2 Enable bit
1 = Comparator 2 output is enabled as a falling event input
0 = Comparator 2 output has no effect on the falling event

bit 1 **FIS1:** COGx Falling Event Input Source 1 Enable bit
1 = Comparator 1 output is enabled as a falling event input
0 = Comparator 1 output has no effect on the falling event

bit 0 **FIS0:** COGx Falling Event Input Source 0 Enable bit
1 = Pin selected with COGxINPPS register is enabled as falling event input
0 = Pin selected with COGxINPPS register has no effect on the falling event

REGISTER 27-9: COGxFSIM0: COGx FALLING EVENT SOURCE INPUT MODE REGISTER 0 (CONTINUED)

bit 1 **FSIM1:** COGx Falling Event Input Source 1 Mode bit

FIS1 = 1:
1 = Comparator 1 high-to-low transition will cause a falling event after falling event phase delay
0 = Comparator 1 low level will cause an immediate falling event

FIS1 = 0:
Comparator 1 has no effect on falling event.

bit 0 **FSIM0:** COGx Falling Event Input Source 0 Mode bit

FIS0 = 1:
1 = Pin selected with COGxINPPS control high-to-low transition will cause a falling event after falling event phase delay
0 = Pin selected with COGxINPPS control low level will cause an immediate falling event

FIS0 = 0:
Pin selected with COGxINPPS control has no effect on falling event.

32.5.4 SLAVE MODE 10-BIT ADDRESS RECEPTION

This section describes a standard sequence of events for the MSSP module configured as an I^{2}C slave in 10-Bit Addressing mode.

Figure 32-20 is used as a visual reference for this description.

This is a step by step process of what must be done by slave software to accomplish I^{2}C communication.

1. Bus starts Idle.
2. Master sends Start condition. S bit of SSPxSTAT is set; SSPxIF is set if interrupt on Start detect is enabled.
3. Master sends matching high address with R/$\overline{\text{W}}$ bit clear; UA bit of the SSPxSTAT register is set.
4. Slave sends $\overline{\text{ACK}}$ and SSPxIF is set.
5. Software clears the SSPxIF bit.
6. Software reads received address from SSPxBUF, clearing the BF flag.
7. Slave loads low address into SSPxADD, releasing SCL.
8. Master sends matching low address byte to the slave; UA bit is set.

> **Note:** Updates to the SSPxADD register are not allowed until after the $\overline{\text{ACK}}$ sequence.

9. Slave sends $\overline{\text{ACK}}$ and SSPxIF is set.

> **Note:** If the low address does not match, SSPxIF and UA are still set so that the slave software can set SSPxADD back to the high address. BF is not set because there is no match. CKP is unaffected.

10. Slave clears SSPxIF.
11. Slave reads the received matching address from SSPxBUF, clearing BF.
12. Slave loads high address into SSPxADD.
13. Master clocks a data byte to the slave and clocks out the slave's $\overline{\text{ACK}}$ on the 9th SCL pulse; SSPxIF is set.
14. If SEN bit of SSPxCON2 is set, CKP is cleared by hardware and the clock is stretched.
15. Slave clears SSPxIF.
16. Slave reads the received byte from SSPxBUF, clearing BF.
17. If SEN is set, the slave sets CKP to release the SCL.
18. Steps 13-17 repeat for each received byte.
19. Master sends Stop to end the transmission.

32.5.5 10-BIT ADDRESSING WITH ADDRESS OR DATA HOLD

Reception using 10-Bit Addressing with AHEN or DHEN set is the same as with 7-bit modes. The only difference is the need to update the SSPxADD register using the UA bit. All functionality, specifically when the CKP bit is cleared and the SCL line is held low, are the same. Figure 32-21 can be used as a reference of a slave in 10-Bit Addressing with AHEN set.

Figure 32-22 shows a standard waveform for a slave transmitter in 10-Bit Addressing mode.

33.1.2 EUSART ASYNCHRONOUS RECEIVER

The Asynchronous mode is typically used in RS-232 systems. The receiver block diagram is shown in Figure 33-2. The data is received on the RX/DT pin and drives the data recovery block. The data recovery block is actually a high-speed shifter operating at 16 times the baud rate, whereas the serial Receive Shift Register (RSR) operates at the bit rate. When all eight or nine bits of the character have been shifted in, they are immediately transferred to a two-character First-In-First-Out (FIFO) memory. The FIFO buffering allows reception of two complete characters and the start of a third character before software must start servicing the EUSART receiver. The FIFO and RSR registers are not directly accessible by software. Access to the received data is via the RCxREG register.

33.1.2.1 Enabling the Receiver

The EUSART receiver is enabled for asynchronous operation by configuring the following three control bits:

- CREN = 1
- SYNC = 0
- SPEN = 1

All other EUSART control bits are assumed to be in their default state.

Setting the CREN bit of the RCxSTA register enables the receiver circuitry of the EUSART. Clearing the SYNC bit of the TXxSTA register configures the EUSART for asynchronous operation. Setting the SPEN bit of the RCxSTA register enables the EUSART. The programmer must set the corresponding TRIS bit to configure the RX/DT I/O pin as an input.

Note:	If the RX/DT function is on an analog pin, the corresponding ANSELx bit must be cleared for the receiver to function.

33.1.2.2 Receiving Data

The receiver data recovery circuit initiates character reception on the falling edge of the first bit. The first bit, also known as the Start bit, is always a zero. The data recovery circuit counts one-half bit time to the center of the Start bit and verifies that the bit is still a zero. If it is not a zero then the data recovery circuit aborts character reception, without generating an error, and resumes looking for the falling edge of the Start bit. If the Start bit zero verification succeeds, then the data recovery circuit counts a full bit time to the center of the next bit. The bit is then sampled by a majority detect circuit and the resulting '0' or '1' is shifted into the RSR. This repeats until all data bits have been sampled and shifted into the RSR. One final bit time is measured and the level sampled. This is the Stop bit, which is always a '1'. If the data recovery circuit samples a '0' in the Stop bit position then a framing error is set for this character, otherwise the framing error is cleared for this character. See **Section 33.1.2.4 "Receive Framing Error"** for more information on framing errors.

Immediately after all data bits and the Stop bit have been received, the character in the RSR is transferred to the EUSART receive FIFO and the RCIF interrupt flag bit of the PIR1 register is set. The top character in the FIFO is transferred out of the FIFO by reading the RCxREG register.

Note:	If the receive FIFO is overrun, no additional characters will be received until the overrun condition is cleared. See **Section 33.1.2.5 "Receive Overrun Error"** for more information on overrun errors.

33.1.2.3 Receive Interrupts

The RCIF interrupt flag bit of the PIR1 register is set whenever the EUSART receiver is enabled and there is an unread character in the receive FIFO. The RCIF interrupt flag bit is read-only; it cannot be set or cleared by software.

RCIF interrupts are enabled by setting all of the following bits:

- RCIE, Interrupt Enable bit of the PIE1 register
- PEIE, Peripheral Interrupt Enable bit of the INTCON register
- GIE, Global Interrupt Enable bit of the INTCON register

The RCIF interrupt flag bit will be set when there is an unread character in the FIFO, regardless of the state of interrupt enable bits.

TABLE 33-5: BAUD RATES FOR ASYNCHRONOUS MODES (CONTINUED)

BAUD RATE	SYNC = 0, BRGH = 1, BRG16 = 1 or SYNC = 1, BRG16 = 1											
	FOSC = 32.000 MHz			FOSC = 20.000 MHz			FOSC = 18.432 MHz			FOSC = 11.0592 MHz		
	Actual Rate	% Error	SPxBRG value (decimal)	Actual Rate	% Error	SPxBRG value (decimal)	Actual Rate	% Error	SPxBRG value (decimal)	Actual Rate	% Error	SPxBRG value (decimal)
300	300.0	0.00	26666	300.0	0.00	16665	300.0	0.00	15359	300.0	0.00	9215
1200	1200	0.00	6666	1200	-0.01	4166	1200	0.00	3839	1200	0.00	2303
2400	2400	0.01	3332	2400	0.02	2082	2400	0.00	1919	2400	0.00	1151
9600	9604	0.04	832	9597	-0.03	520	9600	0.00	479	9600	0.00	287
10417	10417	0.00	767	10417	0.00	479	10425	0.08	441	10433	0.16	264
19.2k	19.18k	-0.08	416	19.23k	0.16	259	19.20k	0.00	239	19.20k	0.00	143
57.6k	57.55k	-0.08	138	57.47k	-0.22	86	57.60k	0.00	79	57.60k	0.00	47
115.2k	115.9k	0.64	68	116.3k	0.94	42	115.2k	0.00	39	115.2k	0.00	23

BAUD RATE	SYNC = 0, BRGH = 1, BRG16 = 1 or SYNC = 1, BRG16 = 1											
	FOSC = 8.000 MHz			FOSC = 4.000 MHz			FOSC = 3.6864 MHz			FOSC = 1.000 MHz		
	Actual Rate	% Error	SPxBRG value (decimal)	Actual Rate	% Error	SPxBRG value (decimal)	Actual Rate	% Error	SPxBRG value (decimal)	Actual Rate	% Error	SPxBRG value (decimal)
300	300.0	0.00	6666	300.0	0.01	3332	300.0	0.00	3071	300.1	0.04	832
1200	1200	-0.02	1666	1200	0.04	832	1200	0.00	767	1202	0.16	207
2400	2401	0.04	832	2398	0.08	416	2400	0.00	383	2404	0.16	103
9600	9615	0.16	207	9615	0.16	103	9600	0.00	95	9615	0.16	25
10417	10417	0	191	10417	0.00	95	10473	0.53	87	10417	0.00	23
19.2k	19.23k	0.16	103	19.23k	0.16	51	19.20k	0.00	47	19.23k	0.16	12
57.6k	57.14k	-0.79	34	58.82k	2.12	16	57.60k	0.00	15	—	—	—
115.2k	117.6k	2.12	16	111.1k	-3.55	8	115.2k	0.00	7	—	—	—

Note: Unless otherwise noted, V_{IN} = 5V, F_{OSC} = 300 kHz, C_{IN} = 0.1 µF, T_A = 25°C.

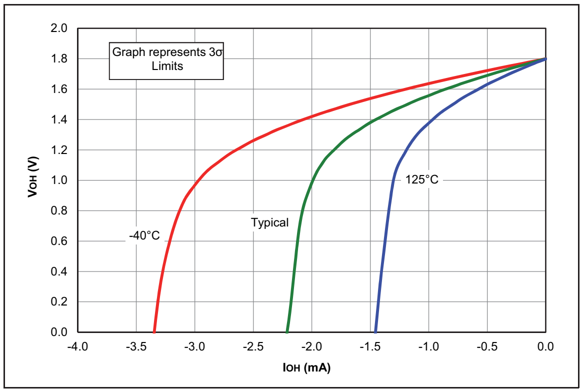


FIGURE 37-49: *Standard IO V_{OH} vs. I_{OH} Over Temperature, V_{DD} = 1.8V, PIC16LF1764/5/8/9 Only.*

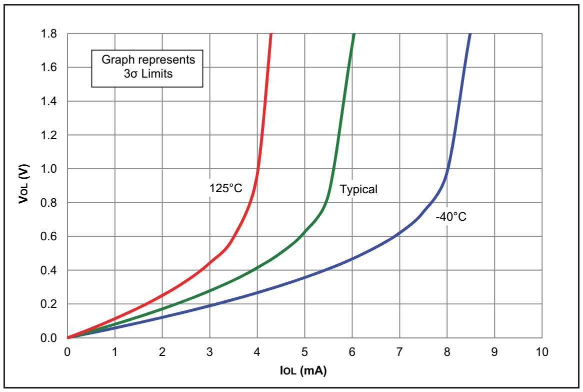


FIGURE 37-50: *Standard IO V_{OL} vs. I_{OL} Over Temperature, V_{DD} = 1.8V, PIC16LF1764/5/8/9 Only.*

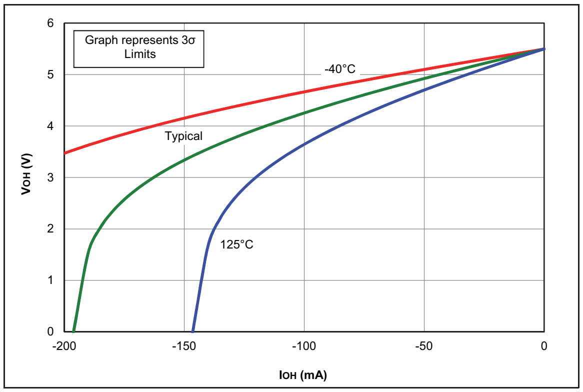


FIGURE 37-51: *100mA IO V_{OH} vs. I_{OH} Over Temperature, V_{DD} = 5.5V, PIC16F1764/5/8/9 Only.*

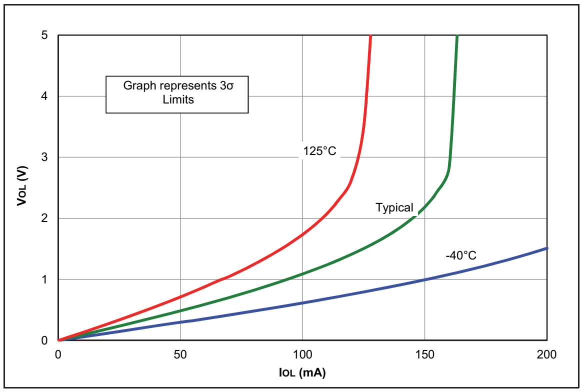


FIGURE 37-52: *100mA IO V_{OL} vs. I_{OL} Over Temperature, V_{DD} = 5.5V, PIC16F1764/5/8/9 Only.*

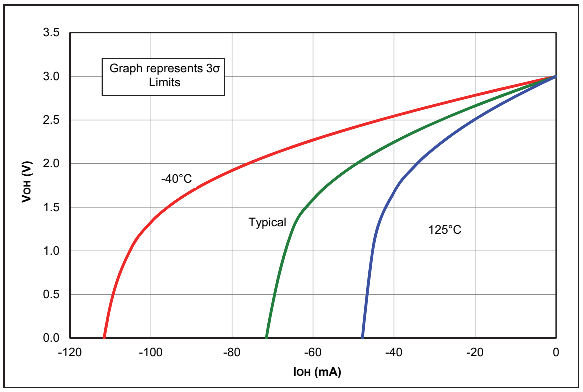


FIGURE 37-53: *100mA IO V_{OH} vs. I_{OH} Over Temperature, V_{DD} = 3.0V.*

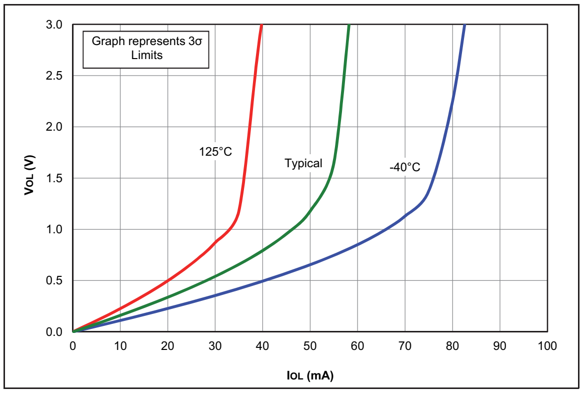
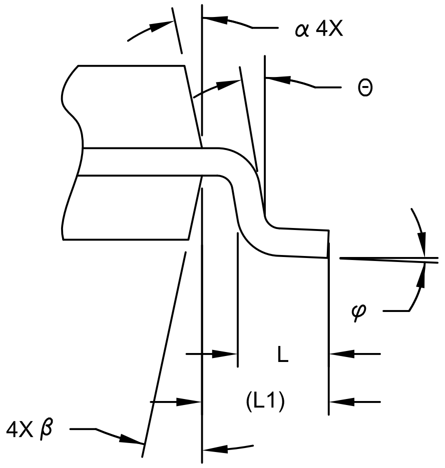
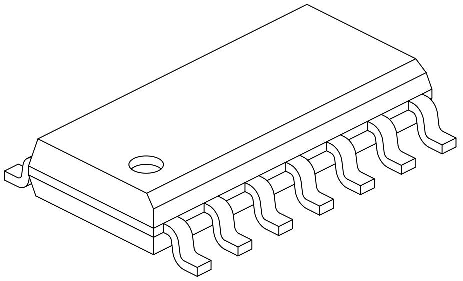


FIGURE 37-54: *V_{OL} vs. I_{OL} Over Temperature, V_{DD} = 3.0V.*

14-Lead Plastic Small Outline (SL) - Narrow, 3.90 mm Body [SOIC]

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





VIEW C

Units		MILLIMETERS		
Dimension Limits		MIN	NOM	MAX
Number of Pins	N	14		
Pitch	e	1.27 BSC		
Overall Height	A	-	-	1.75
Molded Package Thickness	A2	1.25	-	-
Standoff §	A1	0.10	-	0.25
Overall Width	E	6.00 BSC		
Molded Package Width	E1	3.90 BSC		
Overall Length	D	8.65 BSC		
Chamfer (Optional)	h	0.25	-	0.50
Foot Length	L	0.40	-	1.27
Footprint	L1	1.04 REF		
Lead Angle	Θ	0°	-	-
Foot Angle	φ	0°	-	8°
Lead Thickness	c	0.10	-	0.25
Lead Width	b	0.31	-	0.51
Mold Draft Angle Top	α	5°	-	15°
Mold Draft Angle Bottom	β	5°	-	15°

Notes:

1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. § Significant Characteristic
3. Dimension D does not include mold flash, protrusions or gate burrs, which shall not exceed 0.15 mm per end. Dimension E1 does not include interlead flash or protrusion, which shall not exceed 0.25 mm per side.
4. Dimensioning and tolerancing per ASME Y14.5M

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

   REF: Reference Dimension, usually without tolerance, for information purposes only.
5. Datums A & B to be determined at Datum H.

Microchip Technology Drawing No. C04-065C Sheet 2 of 2