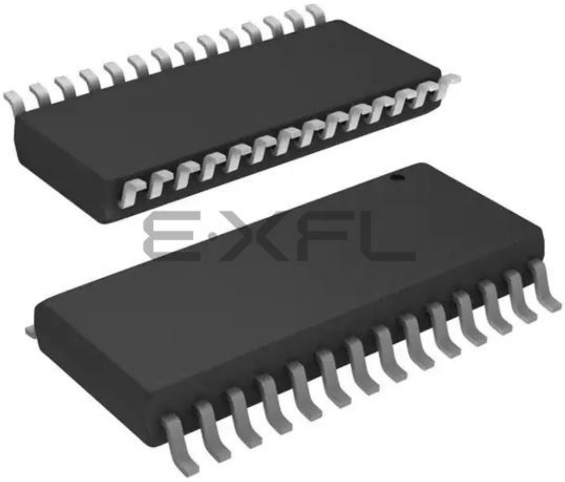


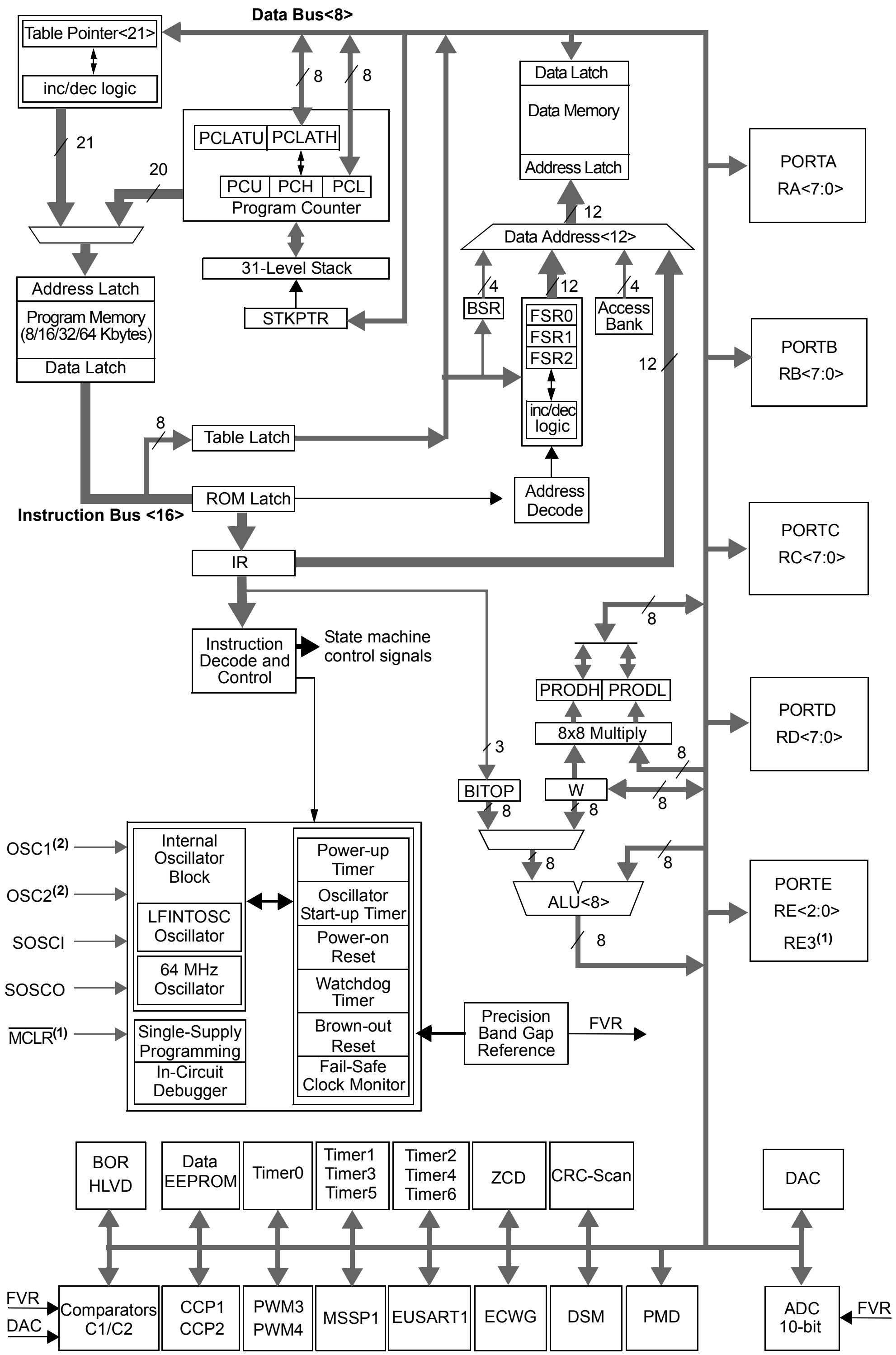
What is "Embedded - Microcontrollers"?

"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	64MHz
Connectivity	I²C, LINbus, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, LVD, POR, PWM, WDT
Number of I/O	25
Program Memory Size	32KB (16K x 16)
Program Memory Type	FLASH
EEPROM Size	256 x 8
RAM Size	2K x 8
Voltage - Supply (Vcc/Vdd)	2.3V ~ 5.5V
Data Converters	A/D 35x10b; D/A 1x5b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 125°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/pic18f25k40-e-so

FIGURE 1-1: PIC18(L)F24/25K40 FAMILY BLOCK DIAGRAM



Note 1: RE3 is only available when $\overline{\text{MCLR}}$ functionality is disabled.

2: OSC1/CLKIN and OSC2/CLKOUT are only available in select oscillator modes and when these pins are not being used as digital I/O. Refer to **Section 4.0 “Oscillator Module (with Fail-Safe Clock Monitor)”** for additional information.

TABLE 10-5: REGISTER FILE SUMMARY FOR PIC18(L)F24/25K40 DEVICES (CONTINUED)

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR, BOR
FE2h	FSR1H	—	—	—	—	Indirect Data Memory Address Pointer 1 High				----xxxx
FE1h	FSR1L	Indirect Data Memory Address Pointer 1 Low								xxxxxxxx
FE0h	BSR	—	—	—	—	Bank Select Register				----0000
FDFh	INDF2	Uses contents of FSR0 to address data memory – value of FSR2 not changed (not a physical register)								--------
FDEh	POSTINC2	Uses contents of FSR0 to address data memory – value of FSR2 post-incremented (not a physical register)								--------
FDDh	POSTDEC2	Uses contents of FSR0 to address data memory – value of FSR2 post-decremented (not a physical register)								--------
FDCh	PREINC2	Uses contents of FSR0 to address data memory – value of FSR2 pre-incremented (not a physical register)								--------
FDBh	PLUSW2	Uses contents of FSR0 to address data memory – value of FSR2 pre-incremented (not a physical register) – value of FSR0 offset by W								--------
FDAh	FSR2H	—	—	—	—	Indirect Data Memory Address Pointer 2 High				----xxxx
FD9h	FSR2L	Indirect Data Memory Address Pointer 2 Low								xxxxxxxx
FD8h	STATUS	—	$\overline{\text{TO}}$	$\overline{\text{PD}}$	N	OV	Z	DC	C	-1100000
FD7h	PCON0	STKOVF	STKUNF	$\overline{\text{WDTWV}}$	$\overline{\text{RWDT}}$	$\overline{\text{RMCLR}}$	$\overline{\text{RI}}$	$\overline{\text{POR}}$	$\overline{\text{BOR}}$	0011110q
FD6h	T0CON1	T0CS<2:0>			T0ASYNC	T0CKPS<3:0>				00000000
FD5h	T0CON0	T0EN	—	T0OUT	T016BIT	T0OUTPS<3:0>				0-000000
FD4h	TMR0H	Holding Register for the Most Significant Byte of the 16-bit TMR0 Register								11111111
FD3h	TMR0L	Holding Register for the Least Significant Byte of the 16-bit TMR0 Register								00000000
FD2h	T1CLK	—	—	—	—	CS<3:0>				----0000
FD1h	T1GATE	—	—	—	—	GSS<3:0>				----0000
FD0h	T1GCON	GE	GPOL	GTM	GSPM	GO/$\overline{\text{DONE}}$	GVAL	—	—	00000x--
FCFh	T1CON	—	—	CKPS<1:0>		—	$\overline{\text{SYNC}}$	RD16	ON	--00-000
FCEh	TMR1H	Holding Register for the Most Significant Byte of the 16-bit TMR1 Register								00000000
FCDh	TMR1L	Holding Register for the Least Significant Byte of the 16-bit TMR1 Register								00000000
FCCh	T3CLK	—	—	—	—	CS<3:0>				----0000
FCBh	T3GATE	—	—	—	—	GSS<3:0>				----0000
FCAh	T3GCON	GE	GPOL	GTM	GSPM	GO/$\overline{\text{DONE}}$	GVAL	—	—	00000x--
FC9h	T3CON	—	—	CKPS<1:0>		—	$\overline{\text{SYNC}}$	RD16	ON	--00-000
FC8h	TMR3H	Holding Register for the Most Significant Byte of the 16-bit TMR3 Register								00000000
FC7h	TMR3L	Holding Register for the Least Significant Byte of the 16-bit TMR3 Register								00000000
FC6h	TMR5CLK	—	—	—	—	CS<3:0>				----0000
FC5h	T5GATE	—	—	—	—	GSS<3:0>				----0000
FC4h	T5GCON	GE	GPOL	GTM	GSPM	GO/$\overline{\text{DONE}}$	GVAL	—	—	00000x--
FC3h	T5CON	—	—	CKPS<1:0>		—	$\overline{\text{SYNC}}$	RD16	ON	--00-000
FC2h	TMR5H	Holding Register for the Most Significant Byte of the 16-bit TMR5 Register								00000000

Legend: x = unknown, u = unchanged, — = unimplemented, q = value depends on condition
Note 1: Not available on LF devices.

TABLE 13-5: SUMMARY OF REGISTERS ASSOCIATED WITH CRC

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Register on Page
CRCACCH	ACC<15:8>								148
CRCACCL	ACC<7:0>								149
CRCCON0	EN	GO	BUSY	ACCM	—	—	SHIFTM	FULL	147
CRCCON1	DLEN<3:0>				PLEN<3:0>				147
CRCDATH	DATA<15:8>								148
CRCDATL	DATA<7:0>								148
CRCSHIFTH	SHIFT<15:8>								149
CRCSHIFTL	SHIFT<7:0>								149
CRCXORH	X<15:8>								150
CRCXORL	X<7:1>							—	150
PMD0	SYSCMD	FVRMD	HLVDMD	CRCMD	SCANMD	NVMMD	CLKRMD	IOCMD	64
SCANCON0	SCANEN	SCANGO	BUSY	INVALID	INTM	—	MODE<1:0>		151
SCANHADRU	—	—	HADR<21:16>						153
SCANHADRH	HADR<15:8>								154
SCANHADRL	HADR<7:0>								154
SCANLADRU	—	—	LADR<21:16>						152
SCANLADRH	LADR<15:8>								152
SCANLADRL	LADR<7:0>								153
SCANTRIG	—	—	—	—	TSEL<3:0>				155
INTCON	GIE/GIEH	PEIE/GIEL	IPEN	—	—	INT2EDG	INT1EDG	INT0EDG	166
PIR7	SCANIF	CRCIF	NVMIF	—	—	—	—	CWG1IF	174
PIE7	SCANIE	CRCIE	NVMIE	—	—	—	—	CWG1IE	182
IPR7	SCANIP	CRCIP	NVMIP	—	—	—	—	CWG1IP	190

Legend: — = unimplemented location, read as ‘0’. Shaded cells are not used for the CRC module.

REGISTER 17-3: PPSLOCK: PPS LOCK REGISTER

U-0	U-0	U-0	U-0	U-0	U-0	U-0	R/W-0/0
—	—	—	—	—	—	—	PPSLOCKED
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 7-1 **Unimplemented:** Read as ‘0’

bit 0 **PPSLOCKED:** PPS Locked bit
1 = PPS is locked. PPS selections can not be changed.
0 = PPS is not locked. PPS selections can be changed.

TABLE 17-2: SUMMARY OF REGISTERS ASSOCIATED WITH THE PPS MODULE

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Register on page
PPSLOCK	—	—	—	—	—	—	—	PPSLOCKED	214
INT0PPS	—	—	—	INT0PPS<4:0>					211
INT1PPS	—	—	—	INT1PPS<4:0>					211
INT2PPS	—	—	—	INT2PPS<4:0>					211
T0CKIPPS	—	—	—	T0CKIPPS<4:0>					211
T1CKIPPS	—	—	—	T1CKIPPS<4:0>					211
T1GPPS	—	—	—	T1GPPS<4:0>					211
T3CKIPPS	—	—	—	T3CKIPPS<4:0>					211
T3GPPS	—	—	—	T3GPPS<4:0>					211
T5CKIPPS	—	—	—	T5CKIPPS<4:0>					211
T5GPPS	—	—	—	T5GPPS<4:0>					211
T2INPPS	—	—	—	T2INPPS<4:0>					211
T4INPPS	—	—	—	T4INPPS<4:0>					211
T6INPPS	—	—	—	T6INPPS<4:0>					211
CCP1PPS	—	—	—	CCP1PPS<4:0>					211
CCP2PPS	—	—	—	CCP2PPS<4:0>					211
CWG1PPS	—	—	—	CWG1PPS<4:0>					211
MDCARLPPS	—	—	—	MDCARLPPS<4:0>					211
MDCARHPPS	—	—	—	MDCARHPPS<4:0>					211
MDSRCPPS	—	—	—	MDSRCPPS<4:0>					211
ADACTPPS	—	—	—	ADACTPPS<4:0>					211
SSP1CLKPPS	—	—	—	SSP1CLKPPS<4:0>					211
SSP1DATPPS	—	—	—	SSP1DATPPS<4:0>					211
SSP1SSPPS	—	—	—	SSP1SSPPS<4:0>					211
RX1PPS	—	—	—	RX1PPS<4:0>					213
TX1PPS	—	—	—	TX1PPS<4:0>					211
RxyPPS	—	—	—	RxyPPS<4:0>					213

Legend: — = unimplemented, read as ‘0’. Shaded cells are unused by the PPS module.

19.4 Timer1/3/5 Prescaler

Timer1/3/5 has four prescaler options allowing 1, 2, 4 or 8 divisions of the clock input. The CKPS bits of the TxCON register control the prescale counter. The prescale counter is not directly readable or writable; however, the prescaler counter is cleared upon a write to TMRxH or TMRxL.

19.5 Secondary Oscillator

A secondary low-power 32.768 kHz oscillator circuit is built-in between pins SOSCI (input) and SOSCO (amplifier output). This internal circuit is to be used in conjunction with an external 32.768 kHz crystal. The secondary oscillator is not dedicated only to Timer1/3/5; it can also be used by other modules.

The oscillator circuit is enabled by setting the SOSCEN bit of the OSCEN register (Register 4-7). This can be used as the clock source to the Timer using the TMRxCLK bits.The oscillator will continue to run during Sleep.

> **Note:** The oscillator requires a start-up and stabilization time before use. Thus, the SOSCEN bit of the OSCEN register should be set and a suitable delay observed prior to enabling Timer1/3/5. A software check can be performed to confirm if the secondary oscillator is enabled and ready to use. This is done by polling the SOR bit of the OSCSTAT (Register 4-4).

19.6 Timer1/3/5 Operation in Asynchronous Counter Mode

If control bit $\overline{\text{SYNC}}$ of the TxCON register is set, the external clock input is not synchronized. The timer increments asynchronously to the internal phase clocks. If external clock source is selected then the timer will continue to run during Sleep and can generate an interrupt on overflow, which will wake-up the processor. However, special precautions in software are needed to read/write the timer (see **Section 19.6.1 "Reading and Writing Timer1/3/5 in Asynchronous Counter Mode"**).

> **Note:** When switching from synchronous to asynchronous operation, it is possible to skip an increment. When switching from asynchronous to synchronous operation, it is possible to produce an additional increment.

19.6.1 READING AND WRITING TIMER1/3/5 IN ASYNCHRONOUS COUNTER MODE

Reading TMRxH or TMRxL while the timer is running from an external asynchronous clock will ensure a valid read (taken care of in hardware). However, the user should keep in mind that reading the 16-bit timer in two 8-bit values itself, poses certain problems, since the timer may overflow between the reads. For writes, it is recommended that the user simply stop the timer and write the desired values. A write contention may occur by writing to the timer registers, while the register is incrementing. This may produce an unpredictable value in the TMRxH:TMRxL register pair.

25.2 DSM Operation

The DSM module can be enabled by setting the MDEN bit in the MDCON0 register. Clearing the MDEN bit in the MDCON0 register, disables the DSM module output and switches the carrier high and carrier low signals to the default option of MDCARHPPS and MDCARLPPS, respectively. The modulator signal source is also switched to the MDBIT in the MDCON0 register. This not only assures that the DSM module is inactive, but that it is also consuming the least amount of current.

The values used to select the carrier high, carrier low, and modulator sources held by the Modulation Source, Modulation High Carrier, and Modulation Low Carrier control registers are not affected when the MDEN bit is cleared and the DSM module is disabled. The values inside these registers remain unchanged while the DSM is inactive. The sources for the carrier high, carrier low and modulator signals will once again be selected when the MDEN bit is set and the DSM module is again enabled and active.

The modulated output signal can be disabled without shutting down the DSM module. The DSM module will remain active and continue to mix signals, but the output value will not be sent to the DSM pin. During the time that the output is disabled, the DSM pin will remain low. The modulated output can be disabled by clearing the MDEN bit in the MDCON register.

25.3 Modulator Signal Sources

The modulator signal can be supplied from the following sources:

- External signal on pin selected by MDSRCPPS
- MDBIT bit in the MDCON0 register
- CCP1/2 Output
- PWM3/4 Output
- Comparator C1/C2 Output
- EUSART RX Signal
- EUSART TX Signal
- MSSP SDO Signal (SPI Mode Only)

The modulator signal is selected by configuring the MDSRCS<3:0> bits in the MDSRC register.

25.4 Carrier Signal Sources

The carrier high signal and carrier low signal can be supplied from the following sources:

- External signal on pin selected by MDCARHPPS/ MDCARLPPS
- FOSC (system clock)
- HFINTOSC
- Reference Clock Module Signal
- CCP1/2 Output Signal
- PWM3/4 Output

The carrier high signal is selected by configuring the MDCHS<2:0> bits in the MDCARH register. The carrier low signal is selected by configuring the MDCLS<2:0> bits in the MDCARL register.

25.5 Carrier Synchronization

During the time when the DSM switches between carrier high and carrier low signal sources, the carrier data in the modulated output signal can become truncated. To prevent this, the carrier signal can be synchronized to the modulator signal. When synchronization is enabled, the carrier pulse that is being mixed at the time of the transition is allowed to transition low before the DSM switches over to the next carrier source.

Synchronization is enabled separately for the carrier high and carrier low signal sources. Synchronization for the carrier high signal is enabled by setting the MDCHSYNC bit in the MDCON1 register. Synchronization for the carrier low signal is enabled by setting the MDCLSYNC bit in the MDCON1 register.

Figure 25-2 through Figure 25-6 show timing diagrams of using various synchronization methods.

REGISTER 26-8: SSPxCON2: MSSPx CONTROL REGISTER 2 (I^{2}C MASTER MODE)

R/W-0	R/W/HC-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
GCEN	ACKSTAT	ACKDT(1)	ACKEN(2)	RCEN(2)	PEN(2)	RSEN(2)	SEN(2)
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	HC = Bit is cleared by hardware	
-n = Value at POR	‘1’ = Bit is set	‘0’ = Bit is cleared	x = Bit is unknown

bit 7 **GCEN:** General Call Enable bit
Unused in Master mode.

bit 6 **ACKSTAT:** Acknowledge Status bit (Master Transmit mode only)
1 = Acknowledge was not received from slave
0 = Acknowledge was received from slave

bit 5 **ACKDT:** Acknowledge Data bit (Master Receive mode only)(1)
1 = Not Acknowledge
0 = Acknowledge

bit 4 **ACKEN:** Acknowledge Sequence Enable bit(2)
1 = Initiates Acknowledge sequence on SDAx and SCLx pins and transmits ACKDT data bit; automatically cleared by hardware
0 = Acknowledge sequence is Idle

bit 3 **RCEN:** Receive Enable bit (Master Receive mode only)(2)
1 = Enables Receive mode for I^{2}C
0 = Receive is Idle

bit 2 **PEN:** Stop Condition Enable bit(2)
1 = Initiates Stop condition on SDAx and SCLx pins; automatically cleared by hardware
0 = Stop condition is Idle

bit 1 **RSEN:** Repeated Start Condition Enable bit(2)
1 = Initiates Repeated Start condition on SDAx and SCLx pins; automatically cleared by hardware
0 = Repeated Start condition is Idle

bit 0 **SEN:** Start Condition Enable bit(2)
1 = Initiates Start condition on SDAx and SCLx pins; automatically cleared by hardware
0 = Start condition is Idle

Note 1: The value that will be transmitted when the user initiates an Acknowledge sequence at the end of a receive.

2: If the I^{2}C module is active, these bits may not be set (no spooling) and the SSPxBUF may not be written (or writes to the SSPxBUF are disabled).

26.8 I^{2}C Mode Operation

All MSSP I^{2}C communication is byte oriented and shifted out MSb first. Six SFR registers and two interrupt flags interface the module with the PIC® microcontroller and user software. Two pins, SDA and SCL, are exercised by the module to communicate with other external I^{2}C devices.

26.8.1 BYTE FORMAT

All communication in I^{2}C is done in 9-bit segments. A byte is sent from a master to a slave or vice-versa, followed by an Acknowledge bit sent back. After the eighth falling edge of the SCL line, the device outputting data on the SDA changes that pin to an input and reads in an acknowledge value on the next clock pulse.

The clock signal, SCL, is provided by the master. Data is valid to change while the SCL signal is low, and sampled on the rising edge of the clock. Changes on the SDA line while the SCL line is high define special conditions on the bus, explained below.

26.8.2 DEFINITION OF I^{2}C TERMINOLOGY

There is language and terminology in the description of I^{2}C communication that have definitions specific to I^{2}C. That word usage is defined below and may be used in the rest of this document without explanation. This table was adapted from the Philips I^{2}C specification.

26.8.3 SDA AND SCL PINS

Selection of any I^{2}C mode with the SSPEN bit set, forces the SCL and SDA pins to be open-drain. These pins should be set by the user to inputs by setting the appropriate TRIS bits.

Note 1: Data is tied to output zero when an I^{2}C mode is enabled.

2: Any device pin can be selected for SDA and SCL functions with the PPS peripheral. These functions are bidirectional. The SDA input is selected with the SSPxDATPPS registers. The SCL input is selected with the SSPxCLKPPS registers. Outputs are selected with the RxyPPS registers. It is the user's responsibility to make the selections so that both the input and the output for each function is on the same pin.

26.8.4 SDA HOLD TIME

The hold time of the SDA pin is selected by the SDAHT bit of the SSPxCON3 register. Hold time is the time SDA is held valid after the falling edge of SCL. Setting the SDAHT bit selects a longer 300 ns minimum hold time and may help on buses with large capacitance.

TABLE 26-2: I^{2}C BUS TERMS

TERM	Description
Transmitter	The device which shifts data out onto the bus.
Receiver	The device which shifts data in from the bus.
Master	The device that initiates a transfer, generates clock signals and terminates a transfer.
Slave	The device addressed by the master.
Multi-master	A bus with more than one device that can initiate data transfers.
Arbitration	Procedure to ensure that only one master at a time controls the bus. Winning arbitration ensures that the message is not corrupted.
Synchronization	Procedure to synchronize the clocks of two or more devices on the bus.
Idle	No master is controlling the bus, and both SDA and SCL lines are high.
Active	Any time one or more master devices are controlling the bus.
Addressed Slave	Slave device that has received a matching address and is actively being clocked by a master.
Matching Address	Address byte that is clocked into a slave that matches the value stored in SSPxADD.
Write Request	Slave receives a matching address with R/$\overline{W}$ bit clear, and is ready to clock in data.
Read Request	Master sends an address byte with the R/$\overline{W}$ bit set, indicating that it wishes to clock data out of the Slave. This data is the next and all following bytes until a Restart or Stop.
Clock Stretching	When a device on the bus hold SCL low to stall communication.
Bus Collision	Any time the SDA line is sampled low by the module while it is outputting and expected high state.

26.9.3 SLAVE TRANSMISSION

When the R/$\overline{\text{W}}$ bit of the incoming address byte is set and an address match occurs, the R/$\overline{\text{W}}$ bit of the SSPxSTAT register is set. The received address is loaded into the SSPxBUF register, and an $\overline{\text{ACK}}$ pulse is sent by the slave on the ninth bit.

Following the $\overline{\text{ACK}}$, slave hardware clears the CKP bit and the SCL pin is held low (see **Section 26.9.6 "Clock Stretching"** for more detail). By stretching the clock, the master will be unable to assert another clock pulse until the slave is done preparing the transmit data.

The transmit data must be loaded into the SSPxBUF register which also loads the SSPSR register. Then the SCL pin should be released by setting the CKP bit of the SSPxCON1 register. The eight data bits are shifted out on the falling edge of the SCL input. This ensures that the SDA signal is valid during the SCL high time.

The $\overline{\text{ACK}}$ pulse from the master-receiver is latched on the rising edge of the ninth SCL input pulse. This $\overline{\text{ACK}}$ value is copied to the ACKSTAT bit of the SSPxCON2 register. If ACKSTAT is set (not $\overline{\text{ACK}}$), then the data transfer is complete. In this case, when the not $\overline{\text{ACK}}$ is latched by the slave, the slave goes idle and waits for another occurrence of the Start bit. If the SDA line was low ($\overline{\text{ACK}}$), the next transmit data must be loaded into the SSPxBUF register. Again, the SCL pin must be released by setting bit CKP.

An MSSP interrupt is generated for each data transfer byte. The SSPxIF bit must be cleared by software and the SSPxSTAT register is used to determine the status of the byte. The SSPxIF bit is set on the falling edge of the ninth clock pulse.

26.9.3.1 Slave Mode Bus Collision

A slave receives a Read request and begins shifting data out on the SDA line. If a bus collision is detected and the SBCDE bit of the SSPxCON3 register is set, the BCLxIF bit of the PIR register is set. Once a bus collision is detected, the slave goes idle and waits to be addressed again. User software can use the BCLxIF bit to handle a slave bus collision.

26.9.3.2 7-bit Transmission

A master device can transmit a read request to a slave, and then clock data out of the slave. The list below outlines what software for a slave will need to do to accomplish a standard transmission. Figure 26-18 can be used as a reference to this list.

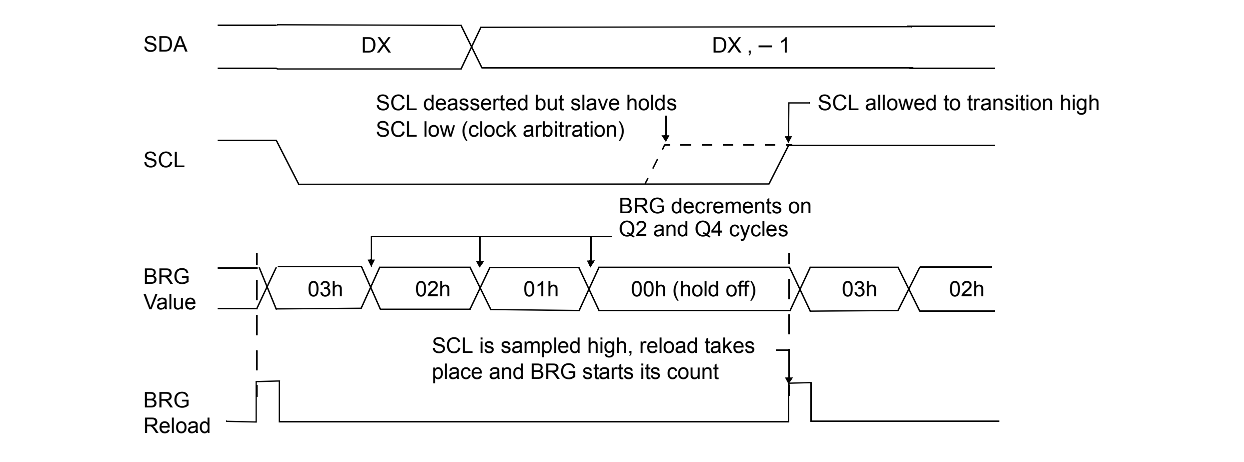
1. Master sends a Start condition on SDA and SCL.
2. S bit of SSPxSTAT is set; SSPxIF is set if interrupt on Start detect is enabled.
3. Matching address with R/$\overline{\text{W}}$ bit set is received by the Slave setting SSPxIF bit.
4. Slave hardware generates an $\overline{\text{ACK}}$ and sets SSPxIF.
5. SSPxIF bit is cleared by user.
6. Software reads the received address from SSPxBUF, clearing BF.
7. R/$\overline{\text{W}}$ is set so CKP was automatically cleared after the $\overline{\text{ACK}}$.
8. The slave software loads the transmit data into SSPxBUF.
9. CKP bit is set releasing SCL, allowing the master to clock the data out of the slave.
10. SSPxIF is set after the $\overline{\text{ACK}}$ response from the master is loaded into the ACKSTAT register.
11. SSPxIF bit is cleared.
12. The slave software checks the ACKSTAT bit to see if the master wants to clock out more data.

Note 1: If the master $\overline{\text{ACK}}$s the clock will be stretched.

2: ACKSTAT is the only bit updated on the rising edge of SCL (9th) rather than the falling.

13. Steps 9-13 are repeated for each transmitted byte.
14. If the master sends a not $\overline{\text{ACK}}$; the clock is not held, but SSPxIF is still set.
15. The master sends a Restart condition or a Stop.
16. The slave is no longer addressed.

FIGURE 26-25: BAUD RATE GENERATOR TIMING WITH CLOCK ARBITRATION


SDA
DX
DX , – 1
SCL deasserted but slave holds
SCL low (clock arbitration)
SCL allowed to transition high
SCL
BRG decrements on
Q2 and Q4 cycles
BRG
Value
03h
02h
01h
00h (hold off)
03h
02h
SCL is sampled high, reload takes
place and BRG starts its count
BRG
Reload

26.10.3 WCOL STATUS FLAG

If the user writes the SSPxBUF when a Start, Restart, Stop, Receive or Transmit sequence is in progress, the WCOL bit is set and the contents of the buffer are unchanged (the write does not occur). Any time the WCOL bit is set it indicates that an action on SSPxBUF was attempted while the module was not idle.

Note: Because queuing of events is not allowed, writing to the lower five bits of SSPxCON2 is disabled until the Start condition is complete.

26.10.4 I^{2}C MASTER MODE START CONDITION TIMING

To initiate a Start condition (Figure 26-26), the user sets the Start Enable bit, SEN bit of the SSPxCON2 register. If the SDA and SCL pins are sampled high, the Baud Rate Generator is reloaded with the contents of SSPxADD<7:0> and starts its count. If SCL and SDA are both sampled high when the Baud Rate Generator times out (TBRG), the SDA pin is driven low. The action of the SDA being driven low while SCL is high is the Start condition and causes the S bit of the SSPxSTAT1 register to be set. Following this, the Baud Rate Generator is reloaded with the contents of SSPxADD<7:0> and resumes its count. When the Baud Rate Generator times out (TBRG), the SEN bit of the SSPxCON2 register will be automatically cleared by hardware; the Baud Rate Generator is suspended, leaving the SDA line held low and the Start condition is complete.

Note 1: If at the beginning of the Start condition, the SDA and SCL pins are already sampled low, or if during the Start condition, the SCL line is sampled low before the SDA line is driven low, a bus collision occurs, the Bus Collision Interrupt Flag, BCLxIF, is set, the Start condition is aborted and the I^{2}C module is reset into its Idle state.

2: The Philips I^{2}C specification states that a bus collision cannot occur on a Start.

FIGURE 26-26: FIRST START BIT TIMING

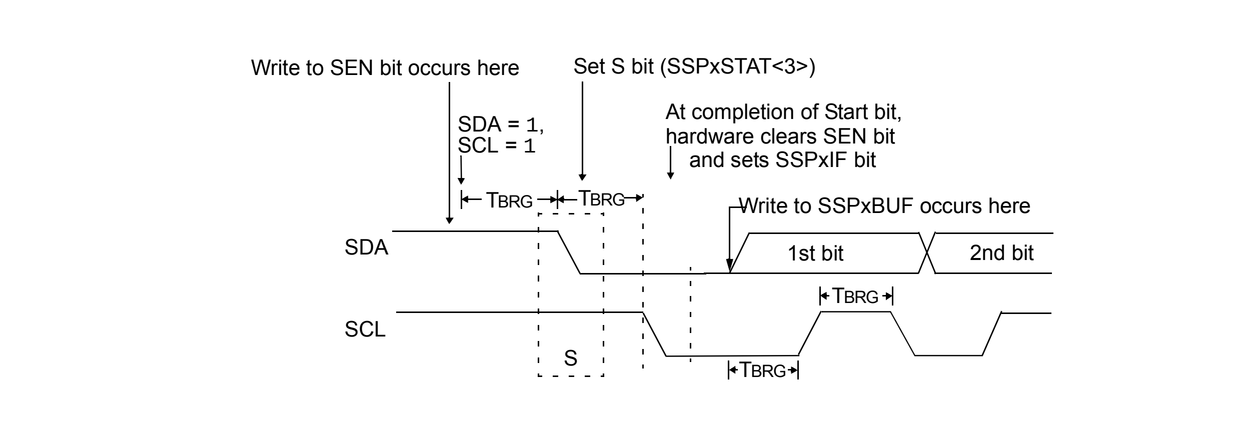


FIGURE 26-34: BUS COLLISION DURING START CONDITION (SCL = 0)

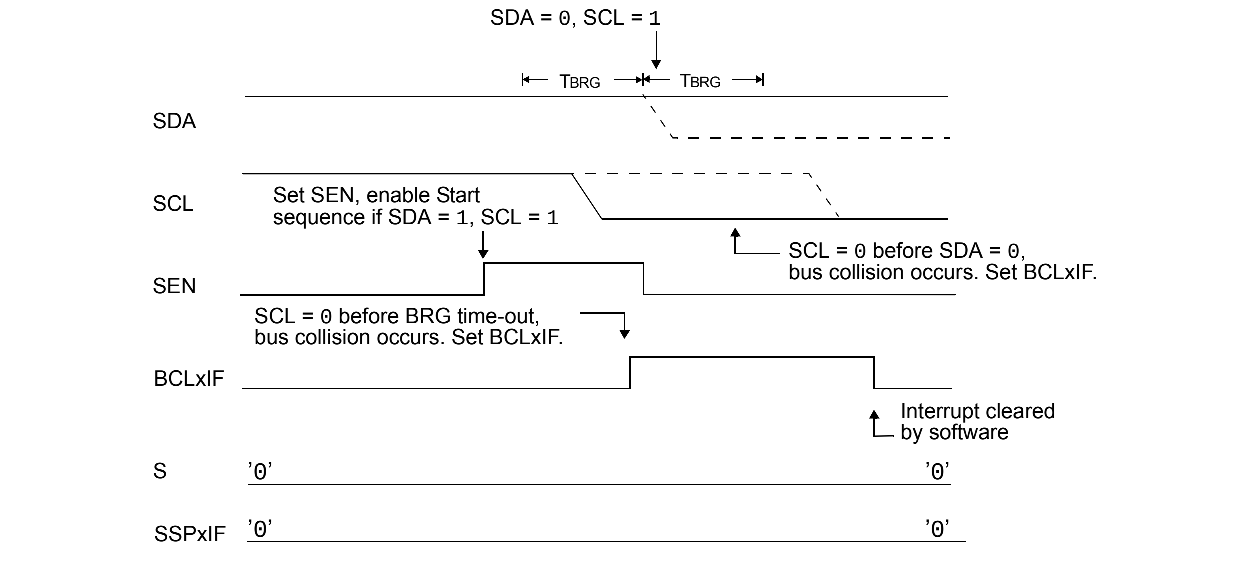

SDA = 0, SCL = 1
TBRG
TBRG
SDA
SCL
Set SEN, enable Start sequence if SDA = 1, SCL = 1
SCL = 0 before SDA = 0, bus collision occurs. Set BCLxIF.
SEN
SCL = 0 before BRG time-out, bus collision occurs. Set BCLxIF.
BCLxIF
Interrupt cleared by software
S
'0'
'0'
SSPxIF
'0'
'0'

FIGURE 26-35: BRG RESET DUE TO SDA ARBITRATION DURING START CONDITION

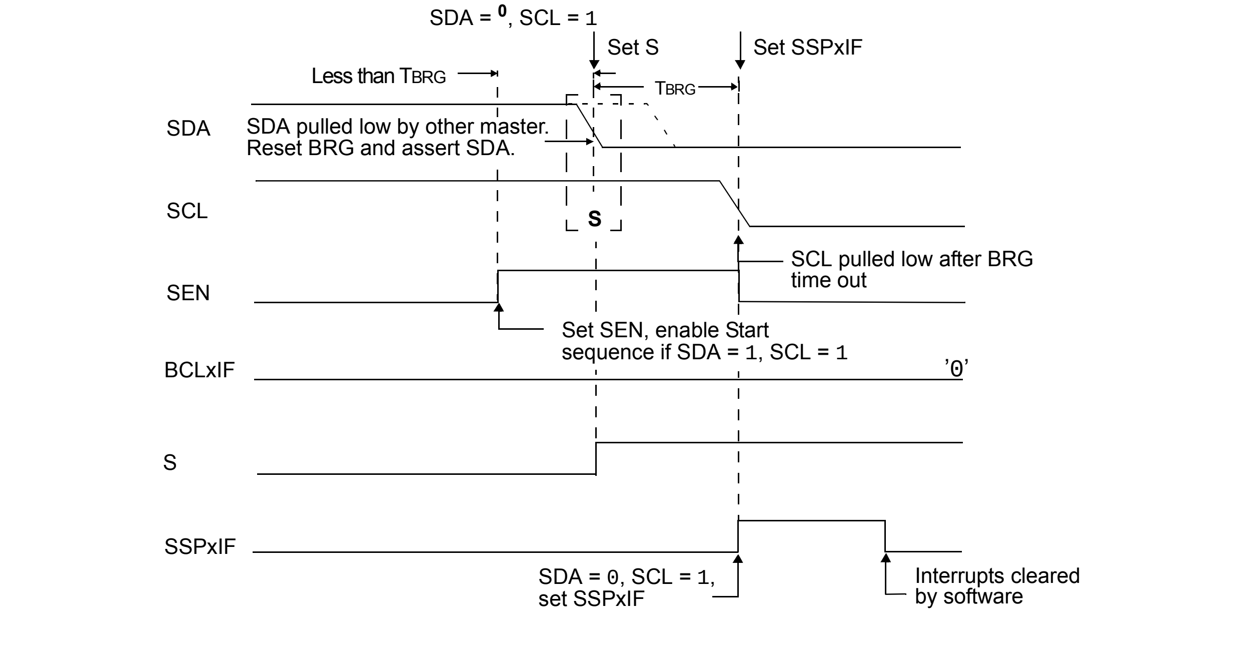
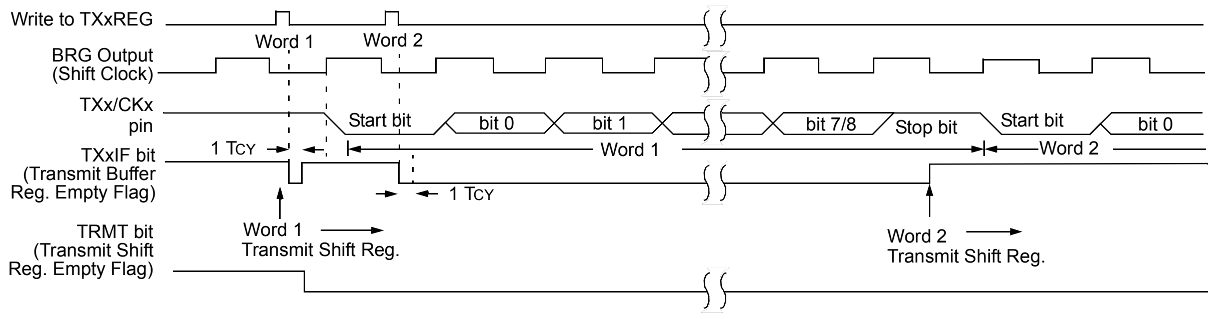

SDA = 0, SCL = 1
Set S
Set SSPxIF
Less than TBRG
TBRG
SDA
SDA pulled low by other master. Reset BRG and assert SDA.
SCL
S
SCL pulled low after BRG time out
SEN
Set SEN, enable Start sequence if SDA = 1, SCL = 1
BCLxIF
'0'
S
SSPxIF
SDA = 0, SCL = 1, set SSPxIF
Interrupts cleared by software

FIGURE 27-4: ASYNCHRONOUS TRANSMISSION (BACK-TO-BACK)



Note: This timing diagram shows two consecutive transmissions.

TABLE 27-1: SUMMARY OF REGISTERS ASSOCIATED WITH ASYNCHRONOUS TRANSMISSION

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Register on Page
BAUDxCON	ABDOVF	RCIDL	—	SCKP	BRG16	—	WUE	ABDEN	389
INTCON	GIE/GIEH	PEIE/GIEL	IPEN	—	—	INT2EDG	INT1EDG	INT0EDG	166
PIE3	—	—	RC1IE	TX1IE	—	—	BCL1IE	SSP1IE	178
PIR3	—	—	RC1IF	TX1IF	—	—	BCL1IF	SSP1IF	170
IPR3	—	—	RC1IP	TX1IP	—	—	BCL1IP	SSP1IP	186
RCxSTA	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	388
RxyPPS	—	—	—	RxyPPS<4:0>					213
TXxPPS	—	—	—	TXPPS<4:0>					211
SPxBRGH	EUSARTx Baud Rate Generator, High Byte								398*
SPxBRGL	EUSARTx Baud Rate Generator, Low Byte								398*
TXxREG	EUSARTx Transmit Register								390*
TXxSTA	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	387

Legend: — = unimplemented location, read as ‘0’. Shaded cells are not used for asynchronous transmission.
* Page provides register information.

31.2 ADC Operation

31.2.1 STARTING A CONVERSION

To enable the ADC module, the ADON bit of the ADCON0 register must be set to a '1'. A conversion may be started by any of the following:

- Software setting the ADGO bit of ADCON0 to '1'
- An external trigger (selected by Register 31-3)
- A continuous-mode retrigger (see section **Section 31.5.8 "Continuous Sampling mode"**)

.

> **Note:** The ADGO bit should not be set in the same instruction that turns on the ADC. Refer to **Section 31.2.6 "ADC Conversion Procedure (Basic Mode)"**.

31.2.2 COMPLETION OF A CONVERSION

When any individual conversion is complete, the value already in ADRES is written into ADPREV (if ADPSIS = 1) and the new conversion results appear in ADRES. When the conversion completes, the ADC module will:

- Clear the ADGO bit (unless the ADCONT bit of ADCON0 is set)
- Set the ADIF Interrupt Flag bit
- Set the ADMATH bit
- Update ADACC

When ADDSEN = 0 then after every conversion, or when ADDSEN = 1 then after every other conversion, the following events occur:

- ADERR is calculated
- ADTIF is set if ADERR calculation meets threshold comparison

Importantly, filter and threshold computations occur after the conversion itself is complete. As such, interrupt handlers responding to ADIF should check ADTIF before reading filter and threshold results.

31.2.3 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.

31.2.4 EXTERNAL TRIGGER DURING SLEEP

If the external trigger is received during sleep while ADC clock source is set to the FRC, ADC module will perform the conversion and set the ADIF bit upon completion.

If an external trigger is received when the ADC clock source is something other than FRC, the trigger will be recorded, but the conversion will not begin until the device exits Sleep.

31.2.5 AUTO-CONVERSION TRIGGER

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

The Auto-conversion Trigger source is selected with the ADACT<4:0> bits of the ADACT 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. See Table 31-2 for auto-conversion sources.

TABLE 31-2: ADC AUTO-CONVERSION TABLE

Source Peripheral	Description
ADCACTPPS	Pin selected by ADCACTPPS
TMR0	Timer0 overflow condition
TMR1	Timer1 overflow condition
TMR2	Match between Timer2 postscaled value and PR2
CCP1	CCP1 output
PWM3/4	PWM3/4 output
C1/2	Comparator C1/2 output
IOC	Interrupt-on-change interrupt trigger
ADERR	Read of ADERRH register
ADRESH	Read of ADRESH register
ADPCH	Write of ADPCH register

31.2.6 ADC CONVERSION PROCEDURE (BASIC MODE)

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

1. Configure Port:
   - Disable pin output driver (Refer to the TRISx register)
   - Configure pin as analog (Refer to the ANSELx register)
2. Configure the ADC module:
   - Select ADC conversion clock
   - Configure voltage reference
   - Select ADC input channel (precharge+acquisition)
   - Turn on ADC module
3. Configure ADC interrupt (optional):
   - Clear ADC interrupt flag
   - Enable ADC interrupt
   - Enable peripheral interrupt (PEIE bit)
   - Enable global interrupt (GIE bit)[(1)]
4. If ADACQ = `0`, software must wait the required acquisition time[(2)].
5. Start conversion by setting the ADGO bit.
6. Wait for ADC conversion to complete by one of the following:
   - Polling the ADGO 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 31.3 "ADC Acquisition Requirements"**.

EXAMPLE 31-1: 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   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
```

31.5.5 BURST AVERAGE MODE

The Burst Average mode (ADMD = `011`) acts the same as the Average mode in most respects. The one way it differs is that it continuously retriggers ADC sampling until the ADCNT value is greater than or equal to ADRPT, even if Continuous Sampling mode (see **Section 31.5.8 "Continuous Sampling mode"**) is not enabled. This allows for a threshold comparison on the average of a short burst of ADC samples.

31.5.6 LOW-PASS FILTER MODE

The Low-pass Filter mode (ADMD = `100`) acts similarly to the Average mode in how it handles samples (accumulates samples until ADCNT value greater than or equal to ADRPT, then triggers threshold comparison), but instead of a simple average, it performs a low-pass filter operation on all of the samples, reducing the effect of high-frequency noise on the average, then performs a threshold comparison on the results. (see Table 31-3 for a more detailed description of the mathematical operation). In this mode, the ADCRS bits determine the cut-off frequency of the low-pass filter (as demonstrated by Table 31-4).

31.5.7 THRESHOLD COMPARISON

At the end of each computation:

- The conversion results are latched and held stable at the end-of-conversion.
- The error is calculated based on a difference calculation which is selected by the ADCALC<2:0> bits in the ADCON3 register. The value can be one of the following calculations (see Register 31-4 for more details):
  - The first derivative of single measurements
  - The CVD result in CVD mode
  - The current result vs. a setpoint
  - The current result vs. the filtered/average result
  - The first derivative of the filtered/average value
  - Filtered/average value vs. a setpoint
- The result of the calculation (ADERR) is compared to the upper and lower thresholds, ADUTH<ADUTHH:ADUTHL> and ADLTH<ADLTHH:ADLTHL> registers, to set the ADUTHR and ADLTHR flag bits. The threshold logic is selected by ADTMD<2:0> bits in the ADCON3 register. The threshold trigger option can be one of the following:
  - Never interrupt
  - Error is less than lower threshold
  - Error is greater than or equal to lower threshold
  - Error is between thresholds (inclusive)
  - Error is outside of thresholds
  - Error is less than or equal to upper threshold
  - Error is greater than upper threshold
  - Always interrupt regardless of threshold test results
  - If the threshold condition is met, the threshold interrupt flag ADTIF is set.

Note 1: The threshold tests are signed operations.

2: If ADAOV is set, a threshold interrupt is signaled.

REGISTER 31-11: ADCAP: ADC ADDITIONAL SAMPLE CAPACITOR SELECTION REGISTER

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

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

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

bit 4-0 **ADCAP<4:0>**: ADC Additional Sample Capacitor Selection bits
11111 = 31 pF
11110 = 30 pF
11101 = 29 pF
•
•
•
00011 = 3 pF
00010 = 2 pF
00001 = 1 pF
00000 = No additional capacitance

REGISTER 31-12: ADRPT: ADC REPEAT SETTING 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
ADRPT<7:0>							
bit 7							bit 0

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

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

bit 7-0 **ADRPT<7:0>**: ADC Repeat Threshold bits
Counts the number of times that the ADC has been triggered and is used along with ADCNT to determine when the error threshold is checked when the computation is Low-pass Filter, Burst Average, or Average modes. See Table 31-3 for more details.

32.4 Comparator Hysteresis

A selectable amount of separation voltage can be added to the input pins of each comparator to provide a hysteresis function to the overall operation. Hysteresis is enabled by setting the CxHYS bit of the CMxCON0 register.

See Comparator Specifications in Table 37-15 for more information.

32.5 Timer1/3/5 Gate Operation

The output resulting from a comparator operation can be used as a source for gate control of Timer1/3/5. See **Section 19.8 "Timer1/3/5 Gate"** for more information. This feature is useful for timing the duration or interval of an analog event.

It is recommended that the comparator output be synchronized to Timer1. This ensures that Timer1 does not increment while a change in the comparator is occurring.

32.5.1 COMPARATOR OUTPUT SYNCHRONIZATION

The output from a comparator can be synchronized with Timer1 by setting the SYNC bit of the CMxCON0 register.

Once enabled, the comparator output is latched on the falling edge of the Timer1 source clock. If a prescaler is used with Timer1, the comparator output is latched after the prescaling function. To prevent a race condition, the comparator output is latched on the falling edge of the Timer1 clock source and Timer1 increments on the rising edge of its clock source. See the Comparator Block Diagram (Figure 32-2) and the Timer1 Block Diagram (Figure 19-1) for more information.

32.6 Comparator Interrupt

An interrupt can be generated upon a change in the output value of the comparator for each comparator, a rising edge detector and a falling edge detector are present.

When either edge detector is triggered and its associated enable bit is set (CxINTP and/or CxINTN bits of the CMxCON1 register), the Corresponding Interrupt Flag bit (CxIF bit of the PIR2 register) will be set.

To enable the interrupt, you must set the following bits:

- EN and POL bits of the CMxCON0 register
- CxIE bit of the PIE2 register
- INTP bit of the CMxCON1 register (for a rising edge detection)
- INTN bit of the CMxCON1 register (for a falling edge detection)
- PEIE and GIE bits of the INTCON register

The associated interrupt flag bit, CxIF bit of the PIR2 register, must be cleared in software. If another edge is detected while this flag is being cleared, the flag will still be set at the end of the sequence.

Note:	Although a comparator is disabled, an interrupt can be generated by changing the output polarity with the CxPOL bit of the CMxCON0 register, or by switching the comparator on or off with the CxEN bit of the CMxCON0 register.

32.7 Comparator Positive Input Selection

Configuring the PCH<2:0> bits of the CMxPCH register directs an internal voltage reference or an analog pin to the non-inverting input of the comparator:

- CxIN0+, CxIN1+ analog pin
- DAC output
- FVR (Fixed Voltage Reference)
- AVSS (Ground)

See **Section 28.0 "Fixed Voltage Reference (FVR)"** for more information on the Fixed Voltage Reference module.

See **Section 30.0 "5-Bit Digital-to-Analog Converter (DAC) Module"** for more information on the DAC input signal.

Any time the comparator is disabled (CxEN = `0`), all comparator inputs are disabled.

32.8 Comparator Negative Input Selection

The NCH<2:0> bits of the CMxNCH register direct an analog input pin and internal reference voltage or analog ground to the inverting input of the comparator:

- CxIN0-, CxIN1-, CxIN2-, CxIN3- analog pin
- FVR (Fixed Voltage Reference)
- Analog Ground

Note:	To use CxINy+ and CxINy- pins as analog input, the appropriate bits must be set in the ANSEL register and the corresponding TRIS bits must also be set to disable the output drivers.

BZ Branch if Zero

Syntax: BZ n

Operands: -128 ≤ n ≤ 127

Operation: if ZERO bit is ‘1’
(PC) + 2 + 2n → PC

Status Affected: None

Encoding:

1110	0000	nnnn	nnnn

Description: If the ZERO bit is ‘1’, then the program will branch.
The 2’s complement number ‘2n’ is added to the PC. Since the PC will have incremented to fetch the next instruction, the new address will be PC + 2 + 2n. This instruction is then a 2-cycle instruction.

Words: 1

Cycles: 1(2)

Q Cycle Activity:
If Jump:

Q1	Q2	Q3	Q4
Decode	Read literal ‘n’	Process Data	Write to PC
No operation	No operation	No operation	No operation

If No Jump:

Q1	Q2	Q3	Q4
Decode	Read literal ‘n’	Process Data	No operation

```
Example:        HERE        BZ   Jump

     Before Instruction
          PC         =     address (HERE)
     After Instruction
          If ZERO    =     1;
               PC    =     address (Jump)
          If ZERO    =     0;
               PC    =     address (HERE + 2)
```

CALL Subroutine Call

Syntax: CALL k {,s}

Operands: 0 ≤ k ≤ 1048575
s ∈ [0,1]

Operation: (PC) + 4 → TOS,
k → PC<20:1>,
if s = 1
(W) → WS,
(Status) → STATUSS,
(BSR) → BSRS

Status Affected: None

Encoding:

1st word (k<7:0>)	1110	110s	$k_7$kkk	kkkk$_0$
2nd word(k<19:8>)	1111	k_{19}kkk	kkkk	kkkk$_8$

Description: Subroutine call of entire 2-Mbyte memory range. First, return address (PC + 4) is pushed onto the return stack. If ‘s’ = 1, the W, Status and BSR registers are also pushed into their respective shadow registers, WS, STATUSS and BSRS. If ‘s’ = 0, no update occurs (default). Then, the 20-bit value ‘k’ is loaded into PC<20:1>. CALL is a 2-cycle instruction.

Words: 2

Cycles: 2

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal ‘k’<7:0>,	PUSH PC to stack	Read literal ‘k’<19:8>, Write to PC
No operation	No operation	No operation	No operation

```
Example:        HERE       CALL   THERE, 1

     Before Instruction
          PC        =     address (HERE)
     After Instruction
          PC        =     address (THERE)
          TOS       =     address (HERE + 4)
          WS        =     W
          BSRS      =     BSR
          STATUSS =       Status
```

SUBLW Subtract W from literal

Syntax: SUBLW k

Operands: 0 ≤ k ≤ 255

Operation: k – (W) → W

Status Affected: N, OV, C, DC, Z

Encoding:

`0000`	`1000`	`kkkk`	`kkkk`

Description: W is subtracted from the 8-bit literal ‘k’. The result is placed in W.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal ‘k’	Process Data	Write to W

Example 1: `SUBLW 02h`

```
Before Instruction
    W    =  01h
    C    =  ?
After Instruction
    W    =  01h
    C    =  1     ; result is positive
    Z    =  0
    N    =  0
```

Example 2: `SUBLW 02h`

```
Before Instruction
    W    =  02h
    C    =  ?
After Instruction
    W    =  00h
    C    =  1     ; result is zero
    Z    =  1
    N    =  0
```

Example 3: `SUBLW 02h`

```
Before Instruction
    W    =  03h
    C    =  ?
After Instruction
    W    =  FFh  ; (2’s complement)
    C    =  0    ; result is negative
    Z    =  0
    N    =  1
```

SUBWF Subtract W from f

Syntax: SUBWF f {,d {,a}}

Operands: 0 ≤ f ≤ 255
d ∈ [0,1]
a ∈ [0,1]

Operation: (f) – (W) → dest

Status Affected: N, OV, C, DC, Z

Encoding:

`0101`	`11da`	`ffff`	`ffff`

Description: Subtract W from register ‘f’ (2’s complement method). If ‘d’ is ‘`0`’, the result is stored in W. If ‘d’ is ‘`1`’, the result is stored back in register ‘f’ (default).
If ‘a’ is ‘`0`’, the Access Bank is selected. If ‘a’ is ‘`1`’, the BSR is used to select the GPR bank.
If ‘a’ is ‘`0`’ and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever f ≤ 95 (5Fh). See **Section 35.2.3 “Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode”** for details.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register ‘f’	Process Data	Write to destination

Example 1: `SUBWF REG, 1, 0`

```
Before Instruction
    REG  =  3
    W    =  2
    C    =  ?
After Instruction
    REG  =  1
    W    =  2
    C    =  1     ; result is positive
    Z    =  0
    N    =  0
```

Example 2: `SUBWF REG, 0, 0`

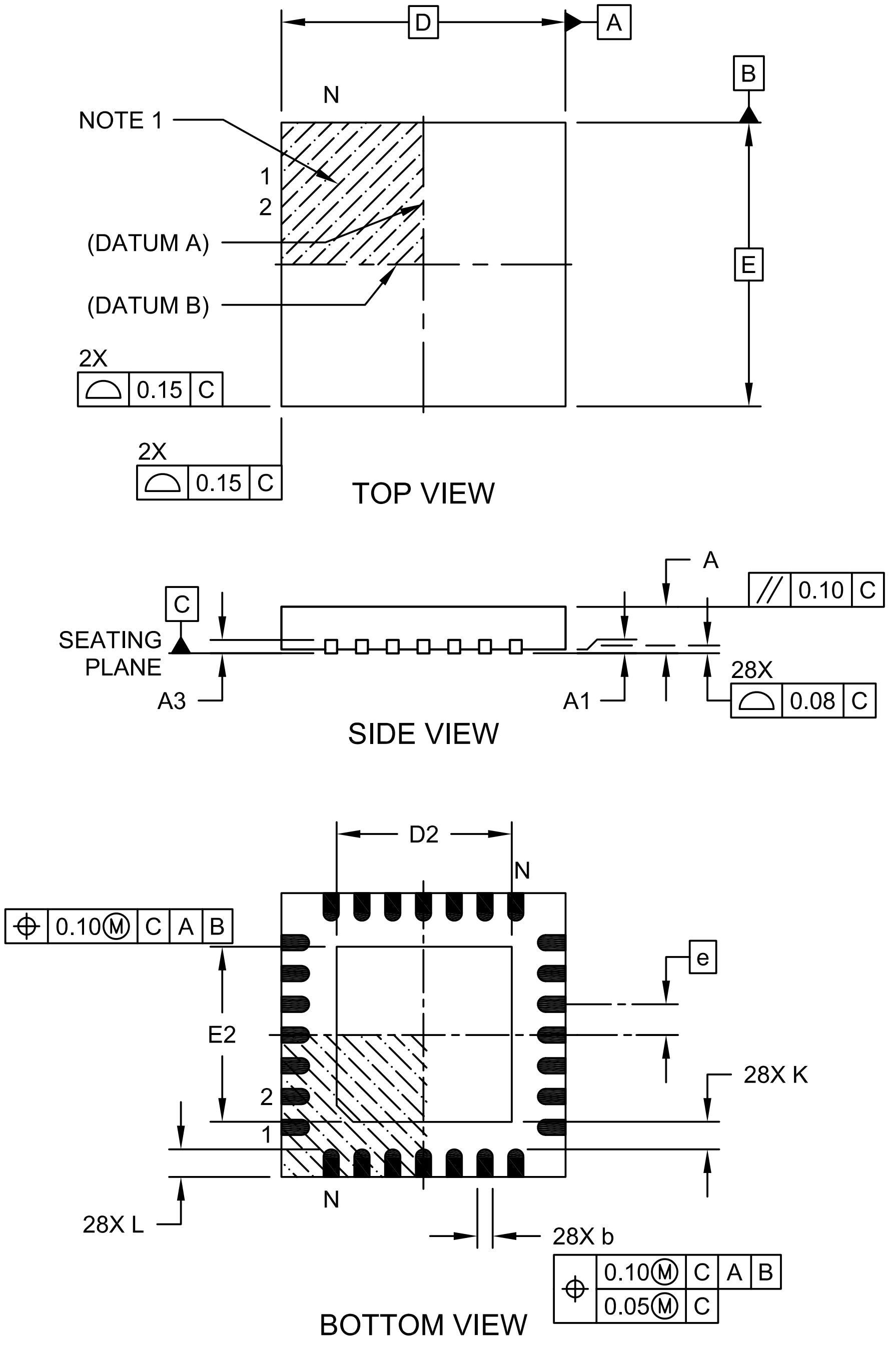
```
Before Instruction
    REG  =  2
    W    =  2
    C    =  ?
After Instruction
    REG  =  2
    W    =  0
    C    =  1     ; result is zero
    Z    =  1
    N    =  0
```

Example 3: `SUBWF REG, 1, 0`

```
Before Instruction
    REG  =  1
    W    =  2
    C    =  ?
After Instruction
    REG  =  FFh  ;(2’s complement)
    W    =  2
    C    =  0     ; result is negative
    Z    =  0
    N    =  1
```

28-Lead Plastic Quad Flat, No Lead Package (ML) - 6x6 mm Body [QFN] With 0.55 mm Terminal Length

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



Microchip Technology Drawing C04-105C Sheet 1 of 2