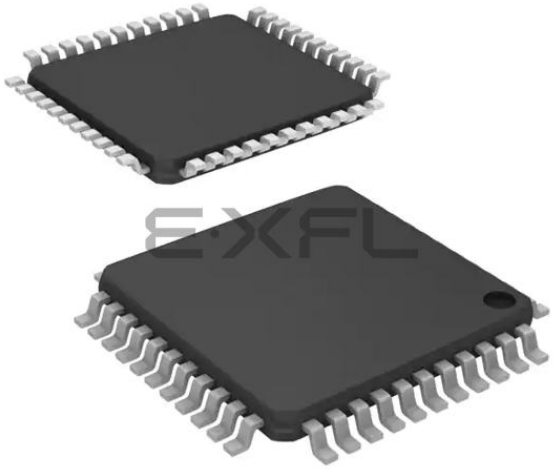


Welcome to E-XFL.COM

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	64MHz
Connectivity	I²C, LINbus, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, LVD, POR, PWM, WDT
Number of I/O	36
Program Memory Size	64KB (32K x 16)
Program Memory Type	FLASH
EEPROM Size	1K x 8
RAM Size	3.6K 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 ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	44-TQFP
Supplier Device Package	44-TQFP (10x10)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic18f46k40-i-pt

Pin Allocation Tables

TABLE 1: 28-PIN ALLOCATION TABLE (PIC18(L)F26K40)

I/O(2)	28-Pin SPDIP, SOIC, SSOP	28-Pin (U)QFN	A/D	Reference	Comparator	Timers	CCP	CWG	ZCD	Interrupt	EUSART	DSM	MSSP	Pull-up	Basic
RA0	2	27	ANA0	—	C1IN0- C2IN0-	—	—	—	—	IOCA0	—	—	—	Y	—
RA1	3	28	ANA1	—	C1IN1- C2IN1-	—	—	—	—	IOCA1	—	—	—	Y	—
RA2	4	1	ANA2	DAC1OUT1 VREF- (DAC) VREF- (ADC)	C1IN0+ C2IN0+	—	—	—	—	IOCA2	—	—	—	Y	—
RA3	5	2	ANA3	VREF+ (DAC) VREF+ (ADC)	C1IN1+	—	—	—	—	IOCA3	—	MDCIN1(1)	—	Y	—
RA4	6	3	ANA4	—	—	T0CKI(1)	—	—	—	IOCA4	—	MDCIN2(1)	—	Y	—
RA5	7	4	ANA5	—	—	—	—	—	—	IOCA5	—	MDMIN(1)	$\overline{SS1}$(1)	Y	—
RA6	10	7	ANA6	—	—	—	—	—	—	IOCA6	—	—	—	Y	CLKOUT OSC2
RA7	9	6	ANA7	—	—	—	—	—	—	IOCA7	—	—	—	Y	OSC1 CLKIN
RB0	21	18	ANB0	—	C2IN1+	—	—	CWG1(1)	ZCDIN	IOCB0 INT0(1)	—	—	$\overline{SS2}$(1)	Y	—
RB1	22	19	ANB1	—	C1IN3- C2IN3-	—	—	—	—	IOCB1 INT1(1)	—	—	SCK2(1) SCL2(3,4)	Y	—
RB2	23	20	ANB2	—	—	—	—	—	—	IOCB2 INT2(1)	—	—	SDI2(1) SDA2(3,4)	Y	—
RB3	24	21	ANB3	—	C1IN2- C2IN2-	—	—	—	—	IOCB3	—	—	—	Y	—
RB4	25	22	ANB4	—	—	T5G(1)	—	—	—	IOCB4	—	—	—	Y	—
RB5	26	23	ANB5	—	—	T1G(1)	—	—	—	IOCB5	—	—	—	Y	—
RB6	27	24	ANB6	—	—	—	—	—	—	IOCB6	CK2(1)	—	—	Y	ICSPCLK
RB7	28	25	ANB7	DAC1OUT2	—	T6AIN(1)	—	—	—	IOCB7	RX2/DT2(1)	—	—	Y	ICSPDAT

Note 1: Default peripheral input. Input can be moved to any other pin with the PPS input selection registers (Register 17-1).

2: All pin outputs default to PORT latch data. Any pin can be selected as a peripheral digital output with the PPS output selection registers.

3: These peripheral functions are bidirectional. The output pin selections must be the same as the input pin selections.

4: These pins are configured for I^2C logic levels; The SCLx/SDAx signals may be assigned to any of these pins. PPS assignments to the other pins (e.g., RB1) will operate, but input logic levels will be standard TTL/ST as selected by the INLVL register, instead of the I^2C specific or SMBus input buffer thresholds.

Table of Contents

REGISTER 7-6: PMD5: PMD CONTROL REGISTER 5

U-0	U-0	U-0	U-0	U-0	U-0	U-0	R/W-0/0
—	—	—	—	—	—	—	DSMMD
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 7-1 **Unimplemented:** Read as '0'

bit 0 **DSMMD:** Disable Data Signal Modulator bit
1 = DSM module disabled
0 = DSM module enabled

TABLE 7-1: SUMMARY OF REGISTERS ASSOCIATED WITH PMD

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Register on page
PMD0	SYSCMD	FVRMD	HLVDMD	CRCMD	SCANMD	NVMMD	CLKRMD	IOCMD	68
PMD1	—	TMR6MD	TMR5MD	TMR4MD	TMR3MD	TMR2MD	TMR1MD	TMR0MD	69
PMD2	—	DACMD	ADCMD	—	—	CMP2MD	CMP1MD	ZCDMD	70
PMD3	—	—	—	—	PWM4MD	PWM3MD	CCP2MD	CCP1MD	71
PMD4	UART2MD	UART1MD	MSSP2MD	MSSP1MD	—	—	—	CWG1MD	72
PMD5	—	—	—	—	—	—	—	DSMMD	73

Legend: — = unimplemented, read as '0'. Shaded cells are not used by the PMD.

10.2 Register Definitions: Stack Pointer

REGISTER 10-1: STKPTR: STACK POINTER REGISTER

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

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented	C = Clearable only bit
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7-5 **Unimplemented:** Read as '0'

bit 4-0 **STKPTR<4:0>:** Stack Pointer Location bits

10.2.1 STACK OVERFLOW AND UNDERFLOW RESETS

Device Resets on Stack Overflow and Stack Underflow conditions are enabled by setting the STVREN bit in Configuration Register 4L. When STVREN is set, a Full or Underflow condition will set the appropriate STKOVF or STKUNF bit and then cause a device Reset. When STVREN is cleared, a Full or Underflow condition will set the appropriate STKOVF or STKUNF bit but not cause a device Reset. The STKOVF or STKUNF bits are cleared by the user software or a Power-on Reset.

10.2.2 FAST REGISTER STACK

A fast register stack is provided for the Status, WREG and BSR registers, to provide a "fast return" option for interrupts. The stack for each register is only one level deep and is neither readable nor writable. It is loaded with the current value of the corresponding register when the processor vectors for an interrupt. All interrupt sources will push values into the stack registers. The values in the registers are then loaded back into their associated registers if the `RETFIE, FAST` instruction is used to return from the interrupt.

If both low and high priority interrupts are enabled, the stack registers cannot be used reliably to return from low priority interrupts. If a high priority interrupt occurs while servicing a low priority interrupt, the stack register values stored by the low priority interrupt will be overwritten. In these cases, users must save the key registers by software during a low priority interrupt.

If interrupt priority is not used, all interrupts may use the fast register stack for returns from interrupt. If no interrupts are used, the fast register stack can be used to restore the STATUS, WREG and BSR registers at the end of a subroutine call. To use the fast register stack for a subroutine call, a `CALL label, FAST` instruction must be executed to save the `STATUS`, WREG and BSR registers to the fast register stack. A `RETURN, FAST` instruction is then executed to restore these registers from the fast register stack.

Example 10-1 shows a source code example that uses the fast register stack during a subroutine call and return.

EXAMPLE 10-1: FAST REGISTER STACK CODE EXAMPLE

```
CALL SUB1, FAST    ;STATUS, WREG, BSR
                   ;SAVED IN FAST REGISTER
                   ;STACK
       •
       •
       •
SUB1   •
       •
       •
   RETURN, FAST    ;RESTORE VALUES SAVED
                   ;IN FAST REGISTER STACK
```

14.4 INTCON Registers

The INTCON registers are readable and writable registers, which contain various enable and priority bits.

14.5 PIR Registers

The PIR registers contain the individual flag bits for the peripheral interrupts. Due to the number of peripheral interrupt sources, there are eight Peripheral Interrupt Request Flag registers (PIR0, PIR1, PIR2, PIR3, PIR4, PIR5, PIR6 and PIR7).

14.6 PIE Registers

The PIE registers contain the individual enable bits for the peripheral interrupts. Due to the number of peripheral interrupt sources, there are eight Peripheral Interrupt Enable registers (PIE0, PIE1, PIE2, PIE3, PIE4, PIE5, PIE6 and PIE7). When IPEN = 0, the PEIE/GIEL bit must be set to enable any of these peripheral interrupts.

14.7 IPR Registers

The IPR registers contain the individual priority bits for the peripheral interrupts. Due to the number of peripheral interrupt sources, there are eight Peripheral Interrupt Priority registers (IPR0, IPR1, IPR2, IPR3, IPR4 and IPR5, IPR6 and IPR7). Using the priority bits requires that the Interrupt Priority Enable (IPEN) bit be set.

REGISTER 14-18: IPR0: PERIPHERAL INTERRUPT PRIORITY REGISTER 0

U-0	U-0	R/W-1/1	R/W-1/1	U-0	R/W-1/1	R/W-1/1	R/W-1/1
—	—	TMR0IP	IOCIP	—	INT2IP	INT1IP	INT0IP
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

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

bit 5 **TMR0IP:** Timer0 Interrupt Priority bit
1 = High priority
0 = Low priority

bit 4 **IOCIP:** Interrupt-on-Change Priority bit
1 = High priority
0 = Low priority

bit 3 **Unimplemented:** Read as '0'

bit 2 **INT2IP:** External Interrupt 2 Priority bit
1 = High priority
0 = Low priority

bit 1 **INT1IP:** External Interrupt 1 Priority bit
1 = High priority
0 = Low priority

bit 0 **INT0IP:** External Interrupt 0 Priority bit
1 = High priority
0 = Low priority

FIGURE 16-1: INTERRUPT-ON-CHANGE BLOCK DIAGRAM (PORTA EXAMPLE)

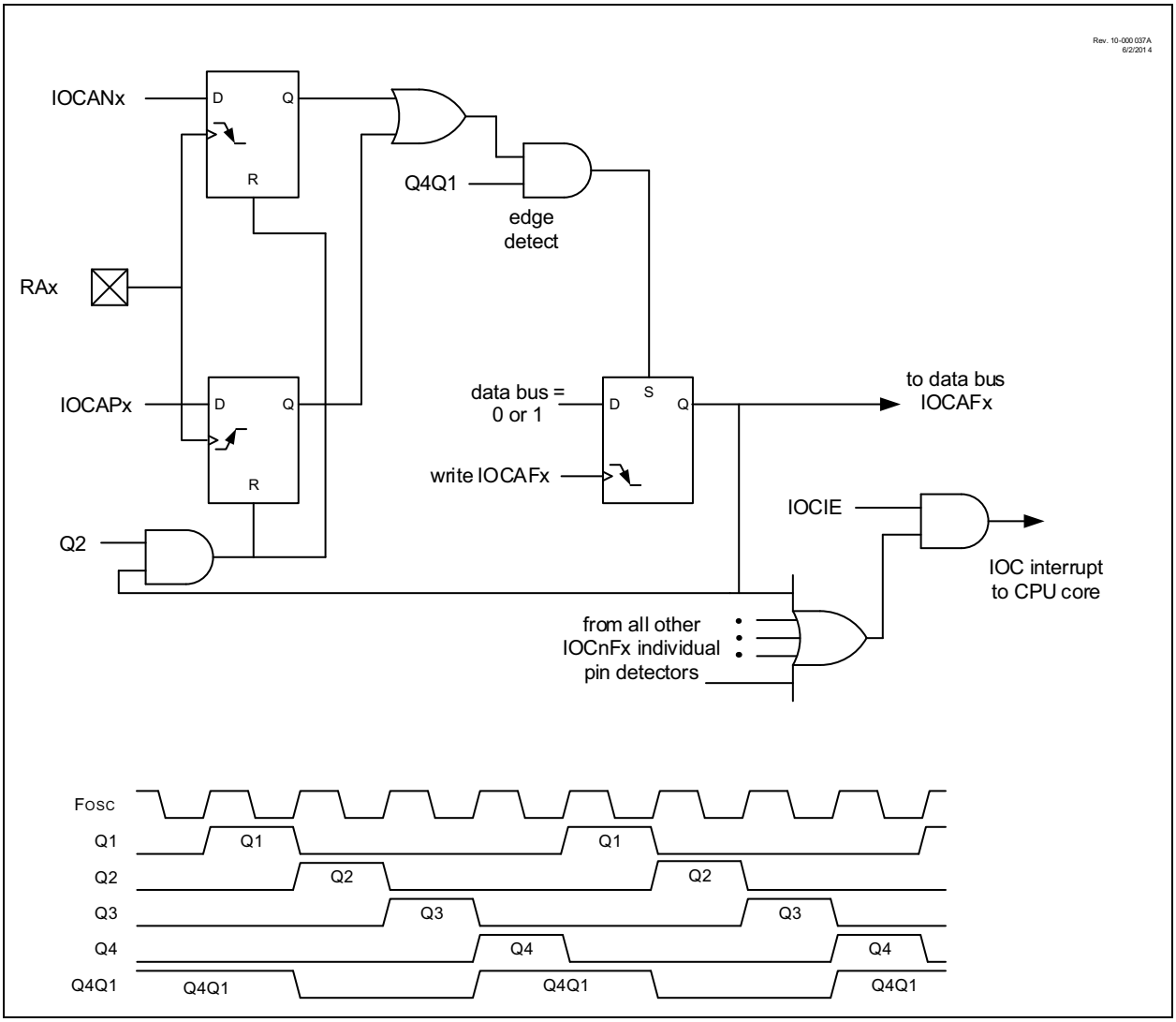
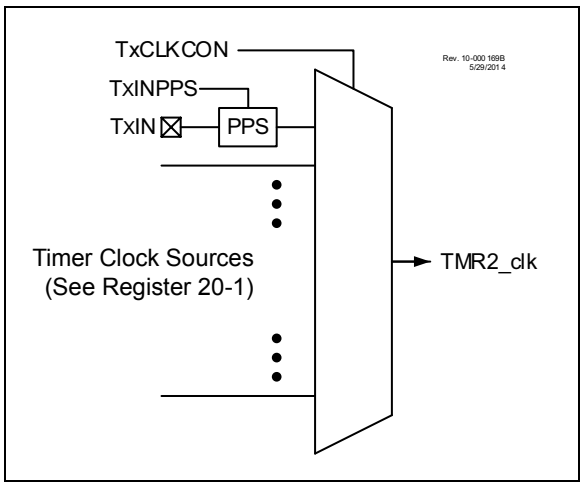


TABLE 17-2: SUMMARY OF REGISTERS ASSOCIATED WITH THE PPS MODULE (CONTINUED)

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Register on page
SSP2SSPPS	—	—	—	SSP2SSPPS<4:0>					216
RX2PPS	—	—	—	RX2PPS<4:0>					218
TX2PPS	—	—	—	TX2PPS<4:0>					216
RxyPPS	—	—	—	RxyPPS<4:0>					218

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

FIGURE 20-2: TIMER2 CLOCK SOURCE BLOCK DIAGRAM



20.1 Timer2 Operation

Timer2 operates in three major modes:

- Free Running Period
- One-shot
- Monostable

Within each mode there are several options for starting, stopping, and reset. Table 20-1 lists the options.

In all modes, the TMR2 count register is incremented on the rising edge of the clock signal from the programmable prescaler. When TMR2 equals T2PR, a high level is output to the postscaler counter. TMR2 is cleared on the next clock input.

An external signal from hardware can also be configured to gate the timer operation or force a TMR2 count Reset. In Gate modes the counter stops when the gate is disabled and resumes when the gate is enabled. In Reset modes the TMR2 count is reset on either the level or edge from the external source.

The TMR2 and T2PR registers are both directly readable and writable. The TMR2 register is cleared and the T2PR register initializes to FFh on any device Reset. Both the prescaler and postscaler counters are cleared on the following events:

- a write to the TMR2 register
- a write to the T2CON register
- any device Reset
- External Reset Source event that resets the timer.

Note: TMR2 is not cleared when T2CON is written.

20.1.1 FREE RUNNING PERIOD MODE

The value of TMR2 is compared to that of the Period register, T2PR, on each clock cycle. When the two values match, the comparator resets the value of TMR2 to 00h on the next cycle and increments the output postscaler counter. When the postscaler count equals the value in the OUTPS<4:0> bits of the TMRxCON1 register then a one clock period wide pulse occurs on the TMR2_postscaled output, and the postscaler count is cleared.

20.1.2 ONE-SHOT MODE

The One-Shot mode is identical to the Free Running Period mode except that the ON bit is cleared and the timer is stopped when TMR2 matches T2PR and will not restart until the T2ON bit is cycled off and on. Postscaler OUTPS<4:0> values other than 0 are meaningless in this mode because the timer is stopped at the first period event and the postscaler is reset when the timer is restarted.

20.1.3 MONOSTABLE MODE

Monostable modes are similar to One-Shot modes except that the ON bit is not cleared and the timer can be restarted by an external Reset event.

20.2 Timer2 Output

The Timer2 module's primary output is TMR2_postscaled, which pulses for a single TMR2_clk period when the postscaler counter matches the value in the OUTPS bits of the TMR2CON register. The T2PR postscaler is incremented each time the TMR2 value matches the T2PR value. This signal can be selected as an input to several other input modules:

- The ADC module, as an Auto-conversion Trigger
- COG, as an auto-shutdown source

In addition, the Timer2 is also used by the CCP module for pulse generation in PWM mode. Both the actual TMR2 value as well as other internal signals are sent to the CCP module to properly clock both the period and pulse width of the PWM signal. See **Section 21.0 "Capture/Compare/PWM Module"** for more details on setting up Timer2 for use with the CCP, as well as the timing diagrams in **Section 20.5 "Operation Examples"** for examples of how the varying Timer2 modes affect CCP PWM output.

20.3 External Reset Sources

In addition to the clock source, the Timer2 also takes in an external Reset source. This external Reset source is selected for Timer2, Timer4 and Timer6 with the T2RST, T4RST and T6RST registers, respectively. This source can control starting and stopping of the timer, as well as resetting the timer, depending on which mode the timer is in. The mode of the timer is controlled by the MODE<4:0> bits of the TMRxHLT register. Edge-Triggered modes require six Timer clock periods between external triggers. Level-Triggered modes require the triggering level to be at least three Timer clock periods long. External triggers are ignored while in Debug Freeze mode.

TABLE 20-3: SUMMARY OF REGISTERS ASSOCIATED WITH TIMER2

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Register on Page
INTCON	GIE/GIEH	PEIE/GIEL	IPEN	—	—	INT2EDG	INT1EDG	INT0EDG	170
PIE4	—	—	TMR6IE	TMR5IE	TMR4IE	TMR3IE	TMR2IE	TMR1IE	183
PIR4	—	—	TMR6IF	TMR5IF	TMR4IF	TMR3IF	TMR2IF	TMR1IF	175
IPR4	—	—	TMR6IP	TMR5IP	TMR4IP	TMR3IP	TMR2IP	TMR1IP	191
PMD1	—	TMR6MD	TMR5MD	TMR4MD	TMR3MD	TMR2MD	TMR1MD	TMR0MD	69
PR2	Timer2 Module Period Register								244*
TMR2	Holding Register for the 8-bit TMR2 Register								244*
T2CON	ON	CKPS<2:0>			OUTPS<3:0>				262
T2CLKCON	—	—	—	—	—	CS<2:0>			264
T2RST	—	—	—	—	RSEL<3:0>				265
T2HLT	PSYNC	CPOL	CSYNC	MODE<4:0>					263
PR4	Timer4 Module Period Register								244*
TMR4	Holding Register for the 8-bit TMR4 Register								244*
T4CON	ON	CKPS<2:0>			OUTPS<3:0>				262
T4CLKCON	—	—	—	—	—	CS<2:0>			264
T4RST	—	—	—	—	RSEL<3:0>				265
T4HLT	PSYNC	CPOL	CSYNC	MODE<4:0>					263
PR6	Timer6 Module Period Register								244*
TMR6	Holding Register for the 8-bit TMR6 Register								244*
T6CON	ON	CKPS<2:0>			OUTPS<3:0>				262
T6CLKCON	—	—	—	—	—	CS<2:0>			264
T6RST	—	—	—	—	RSEL<3:0>				265
T6HLT	PSYNC	CPOL	CSYNC	MODE<4:0>					263
T2INPPS	—	—	—	T2INPPS<4:0>					216
T4INPPS	—	—	—	T4INPPS<4:0>					216
T6INPPS	—	—	—	T6INPPS<4:0>					216

Legend: — = unimplemented location, read as '0'. Shaded cells are not used for Timer2 module.

* Page provides register information.

REGISTER 24-5: CWG1STR(1): CWG STEERING CONTROL REGISTER

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
OVRD	OVRC	OVRB	OVRA	STRD(2)	STRC(2)	STRB(2)	STRA(2)
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 **OVRD:** Steering Data D bit

bit 6 **OVRC:** Steering Data C bit

bit 5 **OVRB:** Steering Data B bit

bit 4 **OVRA:** Steering Data A bit

bit 3 **STRD:** Steering Enable bit D(2)
1 = CWG1D output has the CWG data input waveform with polarity control from POLD bit
0 = CWG1D output is assigned to value of OVRD bit

bit 2 **STRC:** Steering Enable bit C(2)
1 = CWG1C output has the CWG data input waveform with polarity control from POLC bit
0 = CWG1C output is assigned to value of OVRC bit

bit 1 **STRB:** Steering Enable bit B(2)
1 = CWG1B output has the CWG data input waveform with polarity control from POLB bit
0 = CWG1B output is assigned to value of OVRB bit

bit 0 **STRA:** Steering Enable bit A(2)
1 = CWG1A output has the CWG data input waveform with polarity control from POLA bit
0 = CWG1A output is assigned to value of OVRA bit

Note 1: The bits in this register apply only when MODE<2:0> = 00x (Register 24-1, Steering modes).
2: This bit is double-buffered when MODE<2:0> = 001.

26.5.4 SLAVE SELECT SYNCHRONIZATION

The Slave Select can also be used to synchronize communication. The Slave Select line is held high until the master device is ready to communicate. When the Slave Select line is pulled low, the slave knows that a new transmission is starting.

If the slave fails to receive the communication properly, it will be reset at the end of the transmission, when the Slave Select line returns to a high state. The slave is then ready to receive a new transmission when the Slave Select line is pulled low again. If the Slave Select line is not used, there is a risk that the slave will eventually become out of sync with the master. If the slave misses a bit, it will always be one bit off in future transmissions. Use of the Slave Select line allows the slave and master to align themselves at the beginning of each transmission.

The $\overline{SS}$ pin allows a Synchronous Slave mode. The SPI must be in Slave mode with $\overline{SS}$ pin control enabled (SSPxCON1<3:0> = 0100).

When the $\overline{SS}$ pin is low, transmission and reception are enabled and the SDO pin is driven.

When the $\overline{SS}$ pin goes high, the SDO pin is no longer driven, even if in the middle of a transmitted byte and becomes a floating output. External pull-up/pull-down resistors may be desirable depending on the application.

Note 1:	When the SPI is in Slave mode with $\overline{SS}$ pin control enabled (SSPxCON1<3:0> = 0100), the SPI module will reset if the $\overline{SS}$ pin is set to VDD.
2:	When the SPI is used in Slave mode with CKE set; the user must enable $\overline{SS}$ pin control.
3:	While operated in SPI Slave mode the SMP bit of the SSPxSTAT register must remain clear.

When the SPI module resets, the bit counter is forced to '0'. This can be done by either forcing the $\overline{SS}$ pin to a high level or clearing the SSPEN bit.

FIGURE 26-5: SPI DAISY-CHAIN CONNECTION

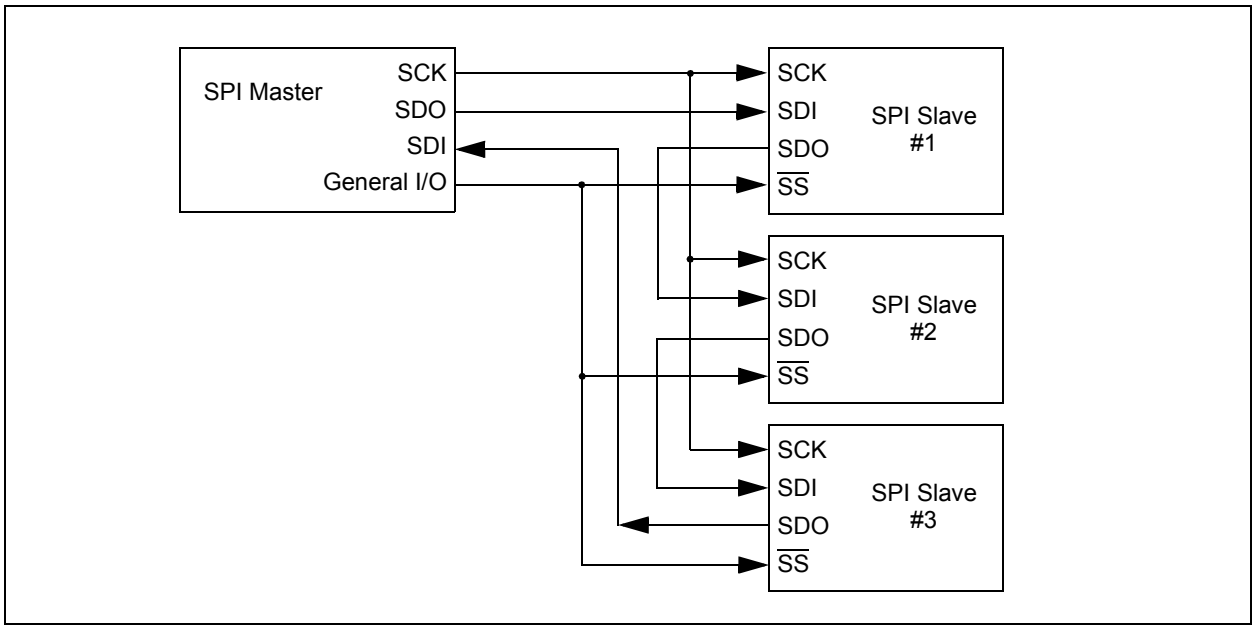


FIGURE 26-7: SPI MODE WAVEFORM (SLAVE MODE WITH CKE = 0)

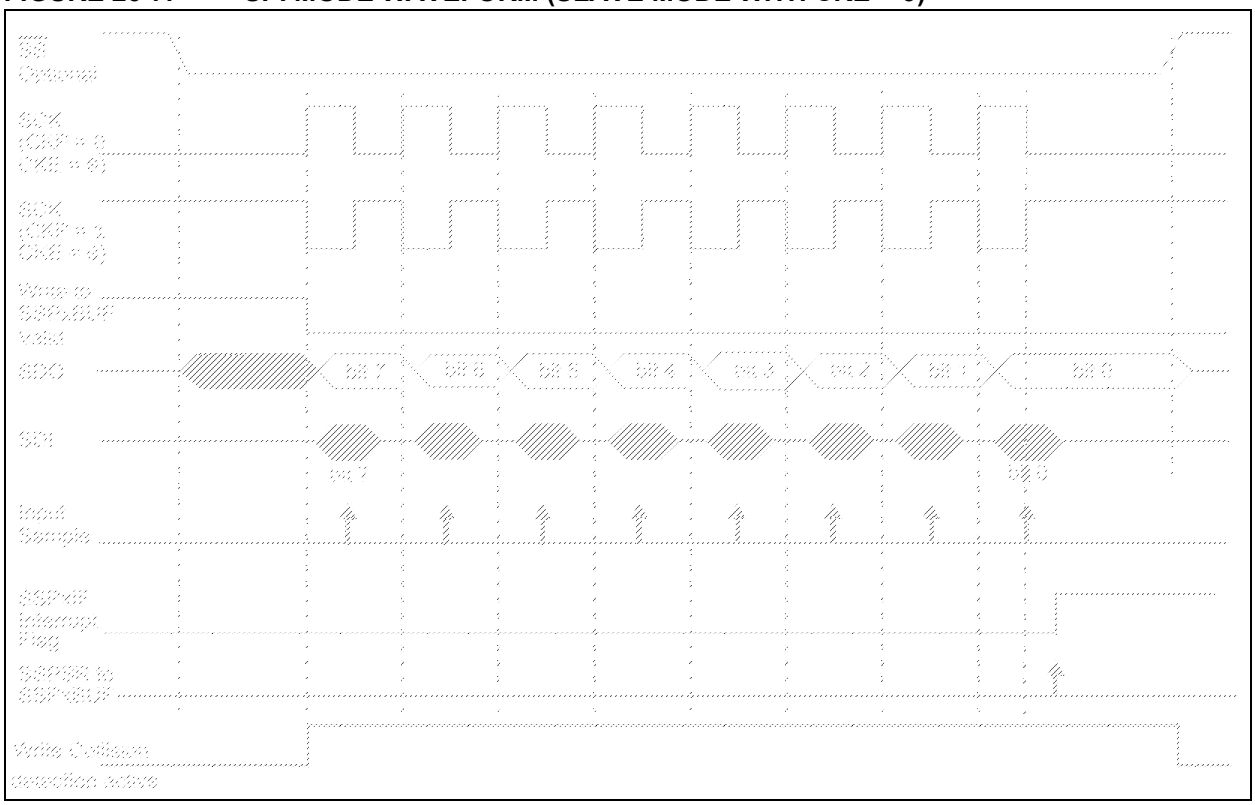


FIGURE 26-8: SPI MODE WAVEFORM (SLAVE MODE WITH CKE = 1)

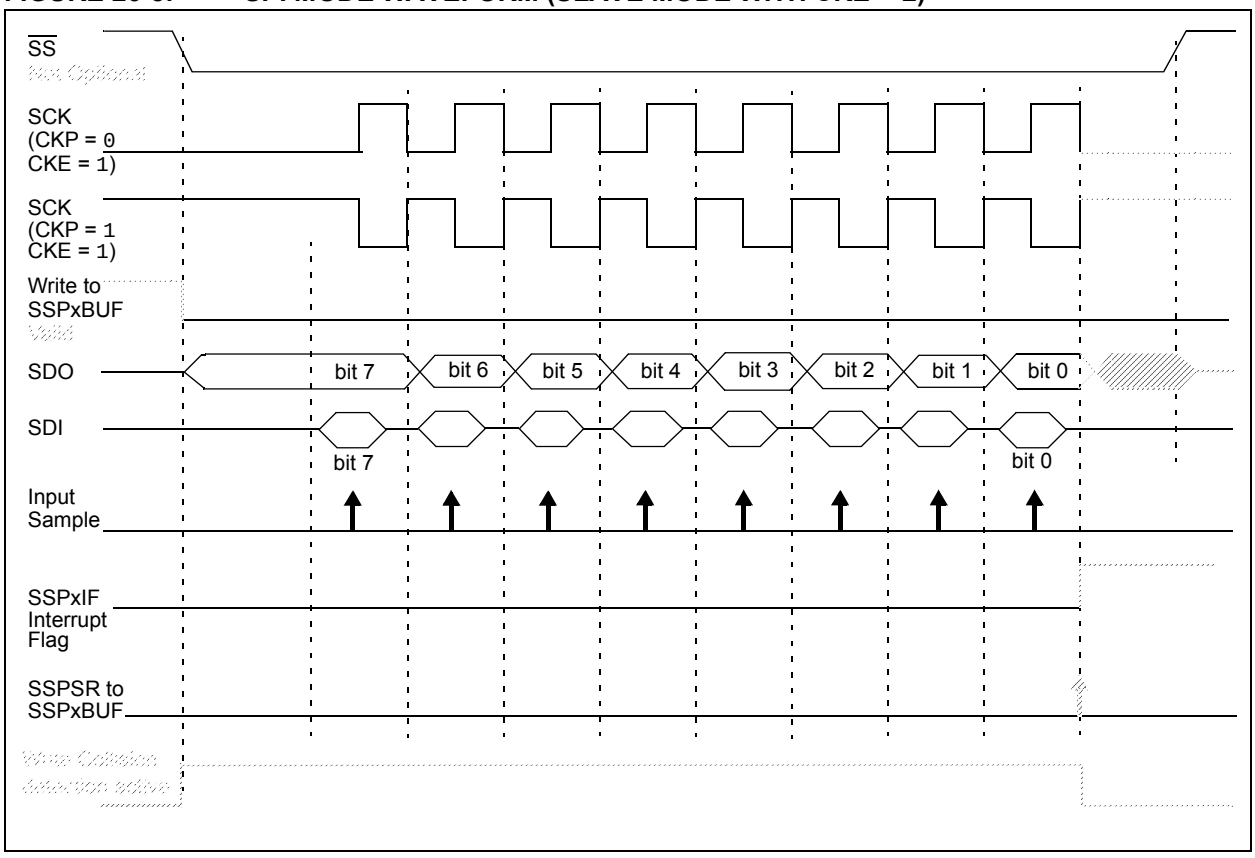
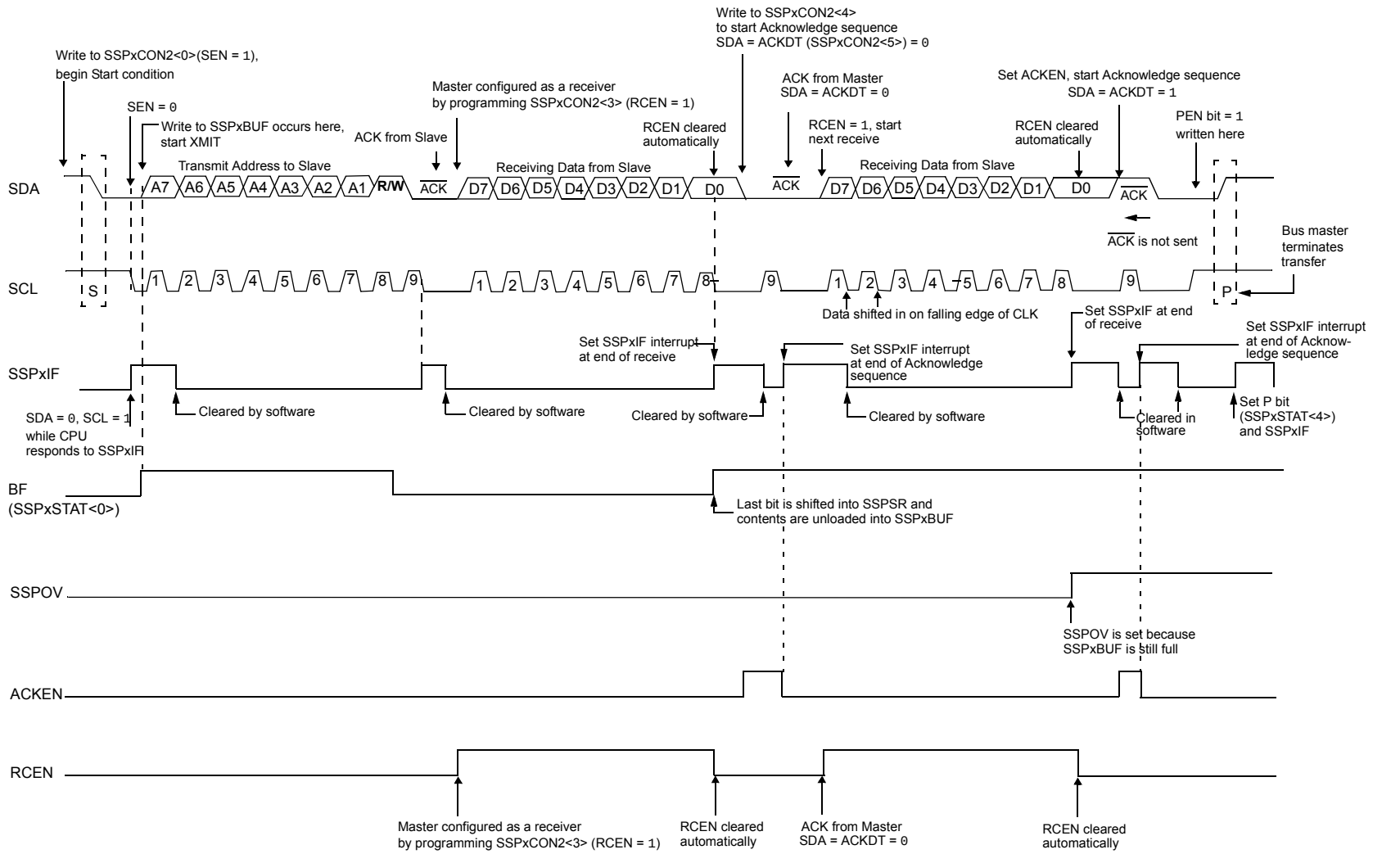


FIGURE 26-29: I^2C MASTER MODE WAVEFORM (RECEPTION, 7-BIT ADDRESS)



27.4.4 BREAK CHARACTER SEQUENCE

The EUSART module has the capability of sending the special Break character sequences that are required by the LIN bus standard. A Break character consists of a Start bit, followed by 12 '0' bits and a Stop bit.

To send a Break character, set the SENDB and TXEN bits of the TXxSTA register. The Break character transmission is then initiated by a write to the TXxREG. The value of data written to TXxREG will be ignored and all '0's will be transmitted.

The SENDB bit is automatically reset by hardware after the corresponding Stop bit is sent. This allows the user to preload the transmit FIFO with the next transmit byte following the Break character (typically, the Sync character in the LIN specification).

The TRMT bit of the TXxSTA register indicates when the transmit operation is active or idle, just as it does during normal transmission. See Figure 27-9 for the timing of the Break character sequence.

27.4.4.1 Break and Sync Transmit Sequence

The following sequence will start a message frame header made up of a Break, followed by an auto-baud Sync byte. This sequence is typical of a LIN bus master.

1. Configure the EUSART for the desired mode.
2. Set the TXEN and SENDB bits to enable the Break sequence.
3. Load the TXxREG with a dummy character to initiate transmission (the value is ignored).
4. Write '55h' to TXxREG to load the Sync character into the transmit FIFO buffer.
5. After the Break has been sent, the SENDB bit is reset by hardware and the Sync character is then transmitted.

When the TXxREG becomes empty, as indicated by the TXxIF, the next data byte can be written to TXxREG.

27.4.5 RECEIVING A BREAK CHARACTER

The Enhanced EUSART module can receive a Break character in two ways.

The first method to detect a Break character uses the FERR bit of the RCxSTA register and the received data as indicated by RCxREG. The Baud Rate Generator is assumed to have been initialized to the expected baud rate.

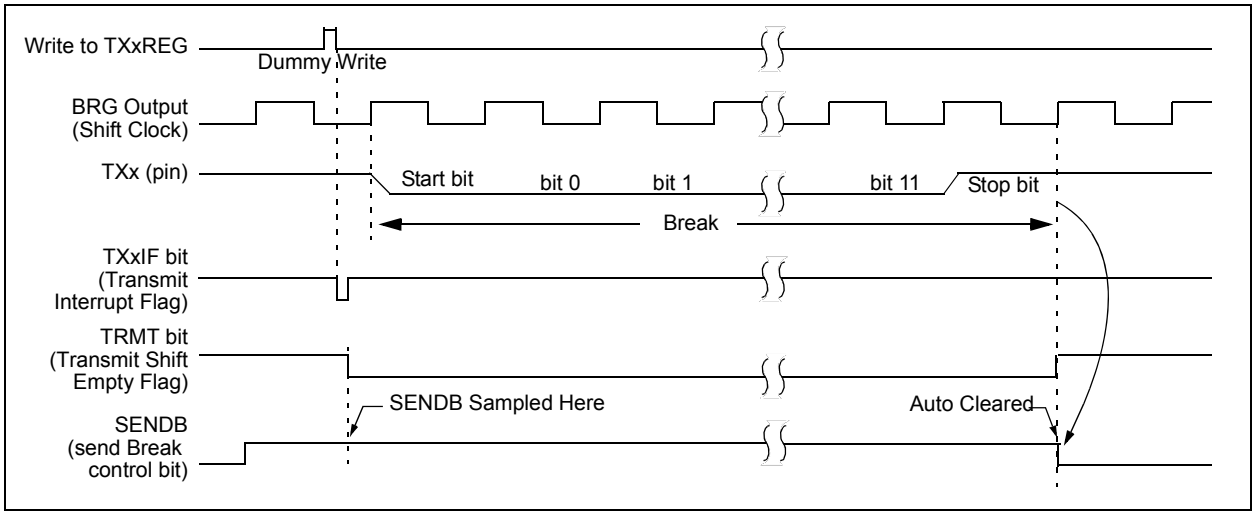
A Break character has been received when;

- RCxIF bit is set
- FERR bit is set
- RCxREG = 00h

The second method uses the Auto-Wake-up feature described in **Section 27.4.3 "Auto-Wake-up on Break"**. By enabling this feature, the EUSART will sample the next two transitions on RX/DT, cause an RCxIF interrupt, and receive the next data byte followed by another interrupt.

Note that following a Break character, the user will typically want to enable the Auto-Baud Detect feature. For both methods, the user can set the ABDEN bit of the BAUDxCON register before placing the EUSART in Sleep mode.

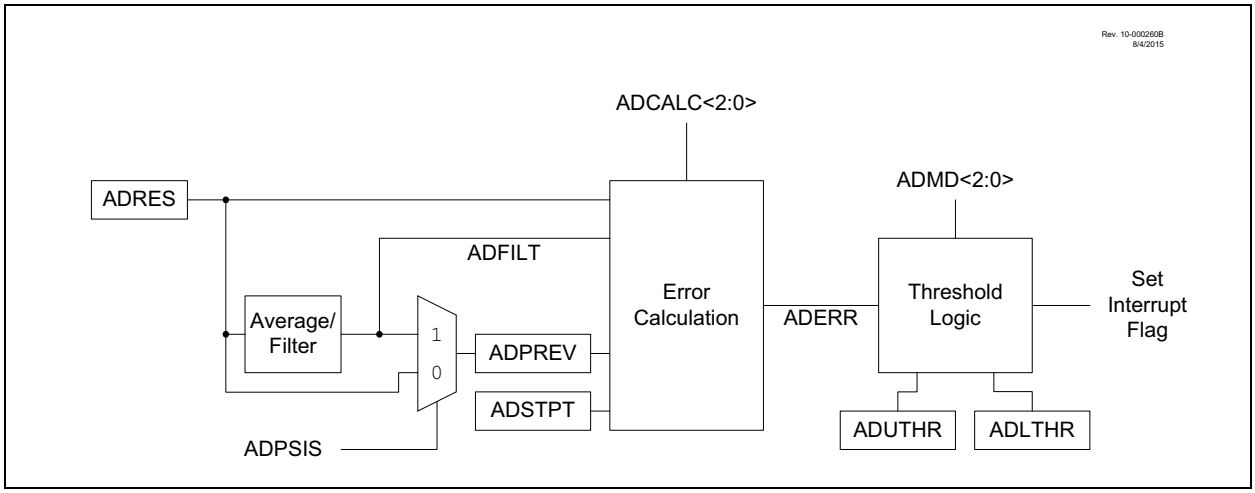
FIGURE 27-9: SEND BREAK CHARACTER SEQUENCE



31.5 Computation Operation

The ADC module hardware is equipped with post conversion computation features. These features provide data post-processing functions that can be operated on the ADC conversion result, including digital filtering/averaging and threshold comparison functions.

FIGURE 31-11: COMPUTATIONAL FEATURES SIMPLIFIED BLOCK DIAGRAM



The operation of the ADC computational features is controlled by ADMD <2:0> bits in the ADCON2 register.

The module can be operated in one of five modes:

• **Basic**: This is a legacy mode. In this mode, ADC conversion occurs on single (ADDSEN = 0) or double (ADDSEN = 1) samples. ADIF is set after all the conversion are complete.

• **Accumulate**: With each trigger, the ADC conversion result is added to accumulator and ADCNT increments. ADIF is set after each conversion. ADTIF is set according to the calculation mode.

• **Average**: With each trigger, the ADC conversion result is added to the accumulator. When the ADRPT number of samples have been accumulated, a threshold test is performed. Upon the next trigger, the accumulator is cleared. For the subsequent tests, additional ADRPT samples are required to be accumulated.

• **Burst Average**: At the trigger, the accumulator is cleared. The ADC conversion results are then collected repetitively until ADRPT samples are accumulated and finally the threshold is tested.

• **Low-Pass Filter (LPF)**: With each trigger, the ADC conversion result is sent through a filter. When ADRPT samples have occurred, a threshold test is performed. Every trigger after that the ADC conversion result is sent through the filter and another threshold test is performed.

The five modes are summarized in Table 31-3 below.

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.

TABLE 37-7: EXTERNAL CLOCK/OSCILLATOR TIMING REQUIREMENTS (CONTINUED)

Standard Operating Conditions (unless otherwise stated)							
Param No.	**Sym.**	**Characteristic**	**Min.**	**Typ†**	**Max.**	**Units**	**Conditions**
OS21	F_{CY}	Instruction Frequency	—	FOSC/4	—	MHz	
OS22	T_{CY}	Instruction Period	62.5	$1/F_{CY}$	—	ns	

* These parameters are characterized but not tested.

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

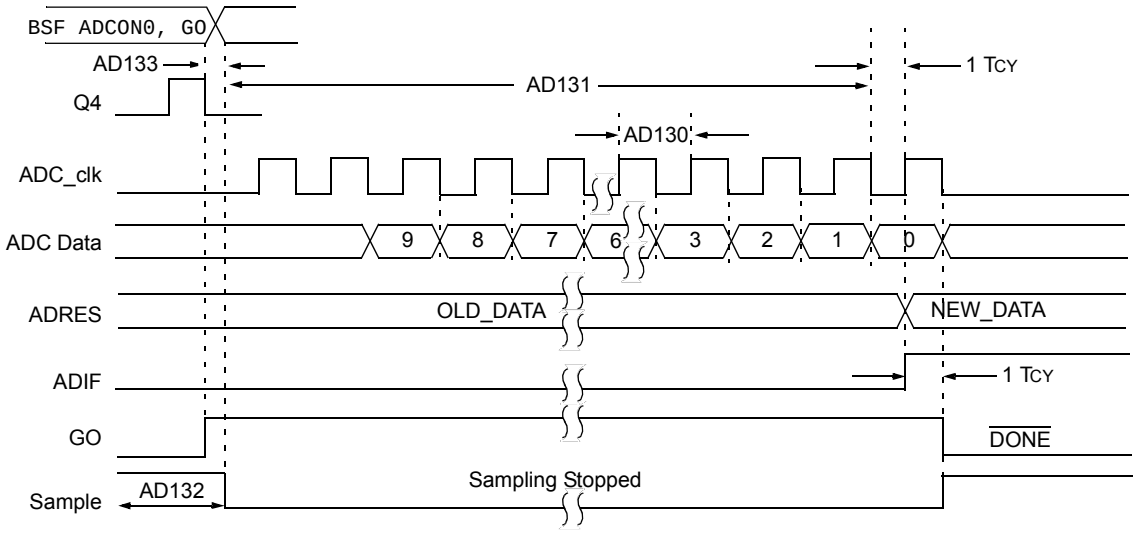
Note 1: Instruction cycle period (TCY) equals four times the input oscillator time base period. All specified values are based on characterization data for that particular oscillator type under standard operating conditions with the device executing code. Exceeding these specified limits may result in an unstable oscillator operation and/or higher than expected current consumption. All devices are tested to operate at "min" values with an external clock applied to OSC1 pin. When an external clock input is used, the "max" cycle time limit is "DC" (no clock) for all devices.

2: The system clock frequency (FOSC) is selected by the "main clock switch controls" as described in **Section 6.0 "Power-Saving Operation Modes"**.

3: The system clock frequency (FOSC) must meet the voltage requirements defined in the **Section 37.2 "Standard Operating Conditions"**.

4: LP, XT and HS oscillator modes require an appropriate crystal or resonator to be connected to the device. For clocking the device with the external square wave, one of the EC mode selections must be used.

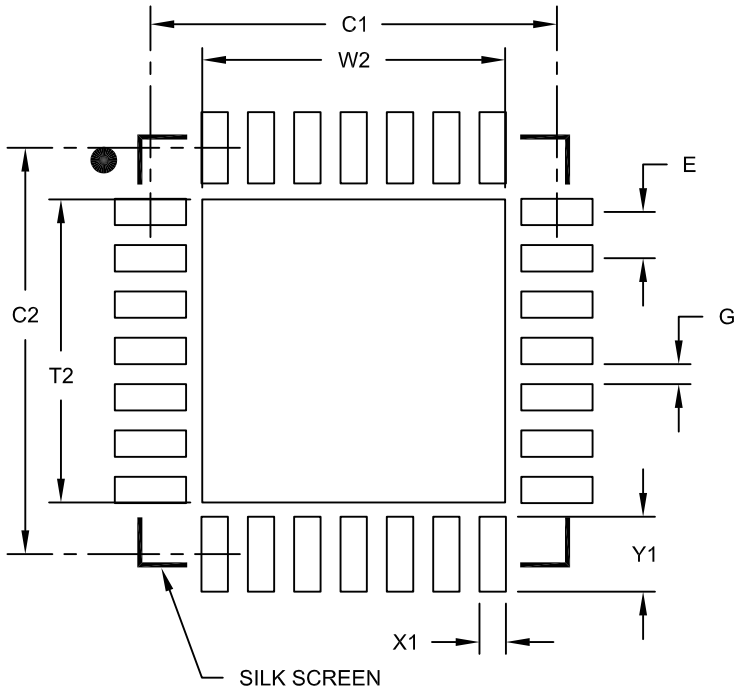
FIGURE 37-11: ADC CONVERSION TIMING (ADC CLOCK FROM ADCRC)



Note 1: If the ADC clock source is selected as ADCRC, a time of TCY is added before the ADC clock starts. This allows the SLEEP instruction to be executed.

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

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



RECOMMENDED LAND PATTERN

Units		MILLIMETERS		
Dimension Limits		MIN	NOM	MAX
Contact Pitch	E		0.65 BSC	
Optional Center Pad Width	W2			4.25
Optional Center Pad Length	T2			4.25
Contact Pad Spacing	C1		5.70	
Contact Pad Spacing	C2		5.70	
Contact Pad Width (X28)	X1			0.37
Contact Pad Length (X28)	Y1			1.00
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. C04-2105A