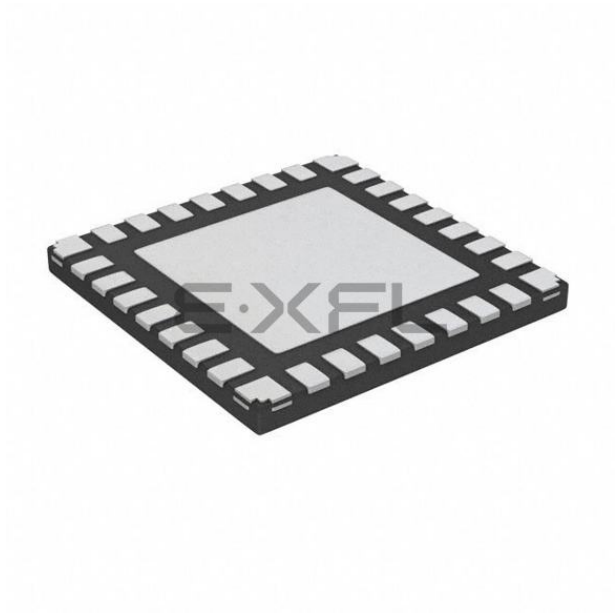


Welcome to E-XFL.COM

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, DMA, HLVD, POR, PWM, WDT
Number of I/O	25
Program Memory Size	64KB (64K x 8)
Program Memory Type	FLASH
EEPROM Size	1K x 8
RAM Size	4K x 8
Voltage - Supply (Vcc/Vdd)	2.3V ~ 5.5V
Data Converters	A/D 24x12b; D/A 1x5b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 125°C (TA)
Mounting Type	Surface Mount
Package / Case	28-UQFN Exposed Pad
Supplier Device Package	28-UQFN (6x6)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic18f26k42-e-mx

EXAMPLE 13-4: WRITING TO PROGRAM FLASH MEMORY

```
                MOVLW     D'64'                 ; number of bytes in erase block
                MOVWF     COUNTER
                MOVLW     BUFFER_ADDR_HIGH      ; point to buffer
                MOVWF     FSR0H
                MOVLW     BUFFER_ADDR_LOW
                MOVWF     FSR0L
                MOVLW     CODE_ADDR_UPPER       ; Load TBLPTR with the base
                MOVWF     TBLPTRU               ; address of the memory block
                MOVLW     CODE_ADDR_HIGH
                MOVWF     TBLPTRH
                MOVLW     CODE_ADDR_LOW
                MOVWF     TBLPTRL
READ_BLOCK
                TBLRD*+                         ; read into TABLAT, and inc
                MOVF      TABLAT, W             ; get data
                MOVWF     POSTINC0              ; store data
                DECFSZ    COUNTER               ; done?
                BRA       READ_BLOCK            ; repeat
MODIFY_WORD
                MOVLW     BUFFER_ADDR_HIGH      ; point to buffer
                MOVWF     FSR0H
                MOVLW     BUFFER_ADDR_LOW
                MOVWF     FSR0L
                MOVLW     NEW_DATA_LOW          ; update buffer word
                MOVWF     POSTINC0
                MOVLW     NEW_DATA_HIGH
                MOVWF     INDF0
ERASE_BLOCK
                MOVLW     CODE_ADDR_UPPER       ; load TBLPTR with the base
                MOVWF     TBLPTRU               ; address of the memory block
                MOVLW     CODE_ADDR_HIGH
                MOVWF     TBLPTRH
                MOVLW     CODE_ADDR_LOW
                MOVWF     TBLPTRL
                BCF       NVMCON1, REG0         ; point to Program Flash Memory
                BSF       NVMCON1, REG1         ; point to Program Flash Memory
                BSF       NVMCON1, WREN         ; enable write to memory
                BSF       NVMCON1, FREE         ; enable Erase operation
                BCF       INTCON0, GIE          ; disable interrupts
                MOVLW     55h
Required        MOVWF     NVMCON2               ; write 55h
Sequence        MOVLW     AAh
                MOVWF     NVMCON2               ; write 0AAh
                BSF       NVMCON1, WR           ; start erase (CPU stall)
                BSF       INTCON0, GIE          ; re-enable interrupts
                TBLRD*-                         ; dummy read decrement
                MOVLW     BUFFER_ADDR_HIGH      ; point to buffer
                MOVWF     FSR0H
                MOVLW     BUFFER_ADDR_LOW
                MOVWF     FSR0L
WRITE_BUFFER_BACK
                MOVLW     BlockSize             ; number of bytes in holding register
                MOVWF     COUNTER
                MOVLW     D'64'/BlockSize       ; number of write blocks in 64 bytes
                MOVWF     COUNTER2
```

REGISTER 16-9: RxyI2C: I²C PAD Rxy CONTROL REGISTER

U-0	R/W-0/0	R/W-0/0	R/W-0/0	U-0	U-0	R/W-0/0	R/W-0/0
—	SLEW	PU<1:0>		—	—	TH<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	HS = Hardware set

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

bit 6 **SLEW**: I²C specific slew rate limiting is enabled
- 1 = I²C specific slew rate limiting is enabled. Standard pad slew limiting is disabled. The SLRxy bit is ignored.
- 0 = Standard GPIO Slew Rate; enabled/disabled via SLRxy bit.

bit 5-4 **PU<1:0>**: I²C Pull-up Selection bits
- 11 = Reserved
- 10 = 10x current of standard weak pull-up
- 01 = 2x current of standard weak pull-up
- 00 = Standard GPIO weak pull-up, enabled via WPUxy bit

bit 3-2 **Unimplemented**: Read as '0'

bit 1-0 **TH<1:0>**: I²C Input Threshold Selection bits
- 11 = SMBus 3.0 (1.35 V) input threshold
- 10 = SMBus 2.0 (2.1 V) input threshold
- 01 = I²C specific input thresholds
- 00 = Standard GPIO Input pull-up, enabled via INLVLxy registers

TABLE 16-10: I2C PAD CONTROL REGISTERS

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
RB1I2C	—	SLEW	PU<1:0>		—	—	TH<1:0>	
RB2I2C	—	SLEW	PU<1:0>		—	—	TH<1:0>	
RC3I2C	—	SLEW	PU<1:0>		—	—	TH<1:0>	
RC4I2C	—	SLEW	PU<1:0>		—	—	TH<1:0>	
RD0I2C(1)	—	SLEW	PU<1:0>		—	—	TH<1:0>	
RD1I2C(1)	—	SLEW	PU<1:0>		—	—	TH<1:0>	

Note 1: Unimplemented in PIC18(L)F26/27K42.

REGISTER 21-2: TxGCON: TIMERx GATE CONTROL REGISTER

R/W-0/u	R/W-0/u	R/W-0/u	R/W-0/u	R/W-0/u	R-x	U-0	U-0
GE	GPOL	GTM	GSPM	GGO/$\overline{\text{DONE}}$	GVAL	—	—
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 **GE:** Timerx Gate Enable bit

If TMRxON = 1:

1 = Timerx counting is controlled by the Timerx gate function

0 = Timerx is always counting

If TMRxON = 0:

This bit is ignored

bit 6 **GPOL:** Timerx Gate Polarity bit

1 = Timerx gate is active-high (Timerx counts when gate is high)

0 = Timerx gate is active-low (Timerx counts when gate is low)

bit 5 **GTM:** Timerx Gate Toggle Mode bit

1 = Timerx Gate Toggle mode is enabled

0 = Timerx Gate Toggle mode is disabled and Toggle flip-flop is cleared

Timerx Gate Flip Flop Toggles on every rising edge

bit 4 **GSPM:** Timerx Gate Single Pulse Mode bit

1 = Timerx Gate Single Pulse mode is enabled and is controlling Timerx gate)

0 = Timerx Gate Single Pulse mode is disabled

bit 3 **GGO/$\overline{\text{DONE}}$:** Timerx Gate Single Pulse Acquisition Status bit

1 = Timerx Gate Single Pulse Acquisition is ready, waiting for an edge

0 = Timerx Gate Single Pulse Acquisition has completed or has not been started.

This bit is automatically cleared when TxGSPM is cleared.

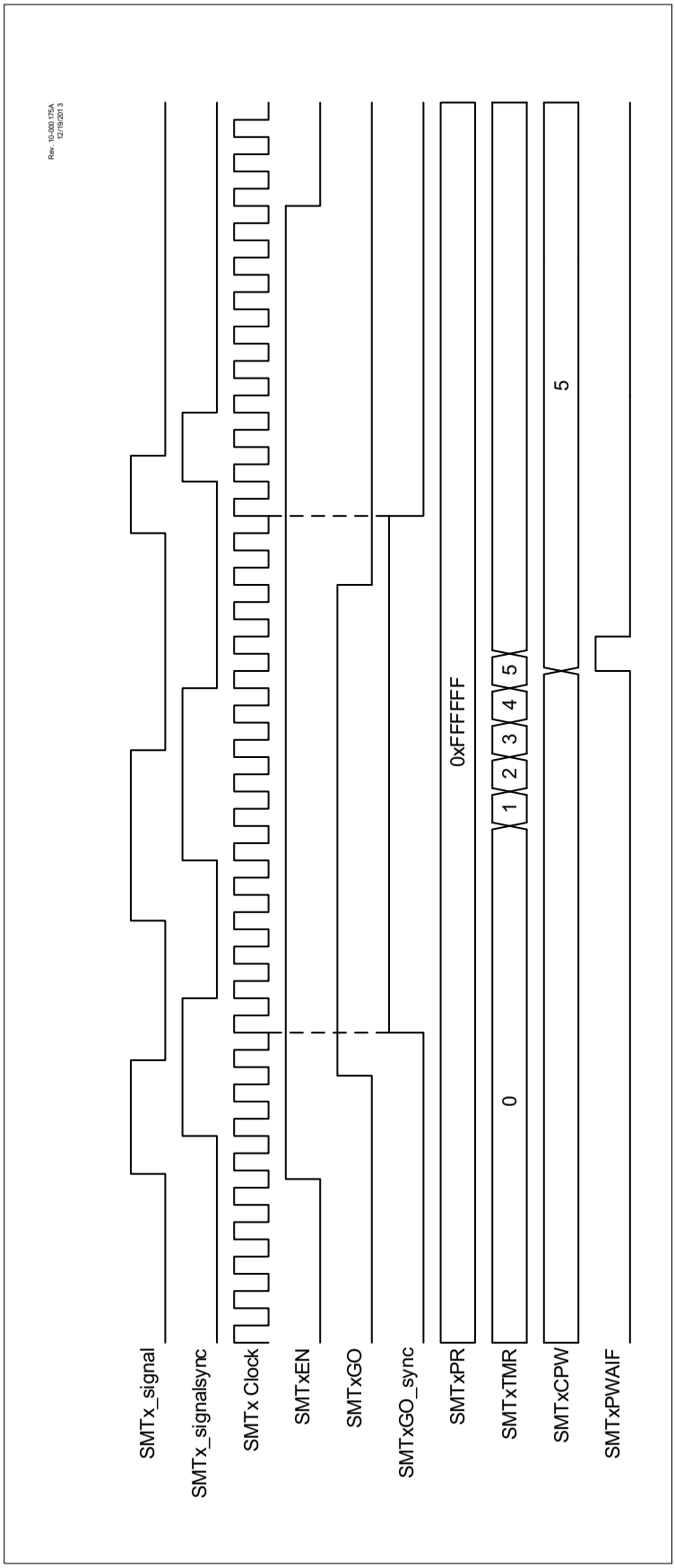
bit 2 **GVAL:** Timerx Gate Current State bit

Indicates the current state of the Timerx gate that could be provided to TMRxH:TMRxL

Unaffected by Timerx Gate Enable (TMRxGE)

bit 1-0 **Unimplemented:** Read as '0'

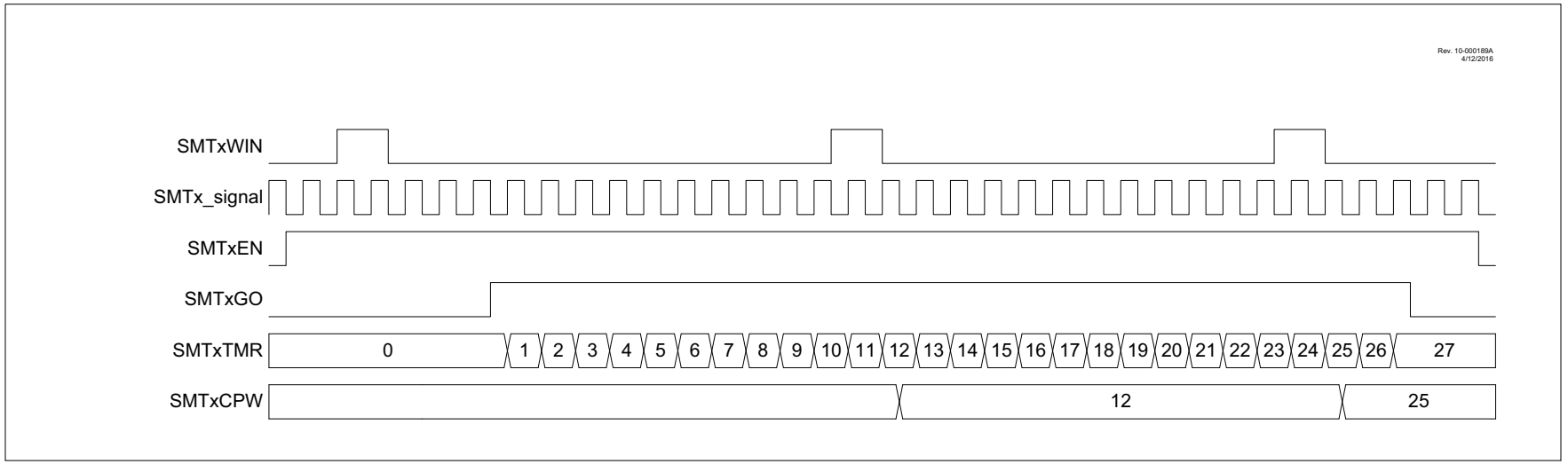
FIGURE 25-5: GATED TIMER MODE SINGLE ACQUISITION TIMING DIAGRAM



25.6.9 COUNTER MODE

This mode increments the timer on each pulse of the SMT1_signal input. This mode is asynchronous to the SMT clock and uses the SMT1_signal as a time source. The SMT1CPW register will be updated with the current SMT1TMR value on the rising edge of the SMT1WIN input. See Figure 25-18.

FIGURE 25-18: COUNTER MODE TIMING DIAGRAM



25.6.10 GATED COUNTER MODE

This mode counts pulses on the SMT1_signal input, gated by the SMT1WIN input. It begins incrementing the timer upon seeing a rising edge of the SMT1WIN input and updates the SMT1CPW register upon a falling edge on the SMT1WIN input. See Figure 25-19 and Figure 25-20.

26.7 Rising Edge and Reverse Dead Band

In Half-Bridge mode, the rising edge dead band delays the turn-on of the CWGxA output after the rising edge of the CWG data input. In Full-Bridge mode, the reverse dead-band delay is only inserted when changing directions from Forward mode to Reverse mode, and only the modulated output CWGxB is affected.

The CWGxDBR register determines the duration of the dead-band interval on the rising edge of the input source signal. This duration is from 0 to 64 periods of the CWG clock.

Dead band is always initiated on the edge of the input source signal. A count of zero indicates that no dead band is present.

If the input source signal reverses polarity before the dead-band count is completed, then no signal will be seen on the respective output.

The CWGxDBR register value is double-buffered. When EN = 0 (Register 26-1), the buffer is loaded when CWGxDBR is written. If EN = 1, then the buffer will be loaded at the rising edge following the first falling edge of the data input, after the LD bit (Register 26-1) is set. Refer to Figure 26-12 for an example.

26.8 Falling Edge and Forward Dead Band

In Half-Bridge mode, the falling edge dead band delays the turn-on of the CWGxB output at the falling edge of the CWG data input. In Full-Bridge mode, the forward dead-band delay is only inserted when changing directions from Reverse mode to Forward mode, and only the modulated output CWGxD is affected.

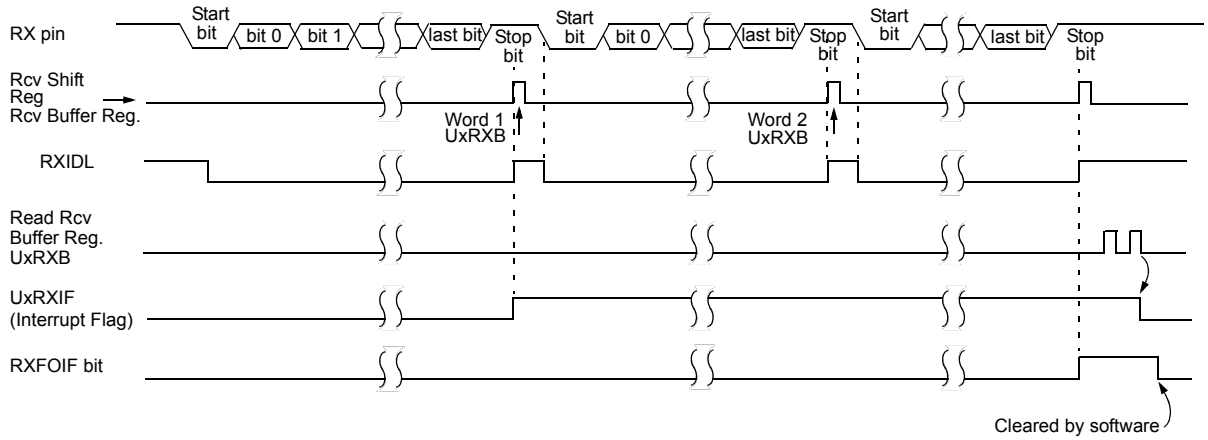
The CWGxDBF register determines the duration of the dead-band interval on the falling edge of the input source signal. This duration is from zero to 64 periods of CWG clock.

Dead-band delay is always initiated on the edge of the input source signal. A count of zero indicates that no dead band is present.

If the input source signal reverses polarity before the dead-band count is completed, then no signal will be seen on the respective output.

The CWGxDBF register value is double-buffered. When EN = 0 (Register 26-1), the buffer is loaded when CWGxDBF is written. If EN = 1, then the buffer will be loaded at the rising edge following the first falling edge of the data input after the LD (Register 26-1) is set. Refer to Figure 26-13 for an example.

FIGURE 31-5: ASYNCHRONOUS RECEPTION



Note: This timing diagram shows three words appearing on the RX input. The UxRXB (receive buffer) is not read before the third word is received, causing the RXFOIF (FIFO overrun) bit to be set. STPMD = 0, STP<1:0> = 00.

31.3 Asynchronous Address Mode

A special Address Detection mode is available for use when multiple receivers share the same transmission line, such as in RS-485 systems.

When Asynchronous Address mode is enabled, all data is transmitted and received as 9-bit characters. The 9th bit determines whether the character is an address or data. When the 9th bit is set, the eight Least Significant bits are the address. When the 9th bit is clear, the Least Significant bits are data. In either case, the 9th bit is stored in PERIF when the byte is written to the receive FIFO. When PERIE is also set, the RXIF will be suppressed, thereby suspending DMA transfers allowing software to process the received address.

An address character will enable all receivers that match the address and disable all other receivers. Once a receiver is enabled, all non-address characters will be received until an address character is received that does not match.

31.3.1 ADDRESS MODE TRANSMIT

The UART transmitter is enabled for asynchronous address operation by configuring the following control bits:

- TXEN = 1
- MODE<3:0> = 0100
- UxBRGH:L = desired baud rate
- RxyPPS = code for desired output pin
- ON = 1

Addresses are sent by writing to the UxP1L register. This transmits the written byte with the 9th bit set, which indicates that the byte is an address.

Data is sent by writing to the UxTXB register. This transmits the written byte with the 9th bit cleared, which indicates that the byte is data.

To send data to a particular device on the transmission bus, first transmit the address of the intended device. All subsequent data will be accepted only by that device until an address of another device is transmitted.

Writes to UxP1L take precedence over writes to UxTXB. When both the UxP1L and UxTXB registers are written while the TSR is busy, the next byte to be transmitted will be from UxP1L.

To ensure that all data intended for one device is sent before the address is changed, wait until the TXMTIF bit is high before writing UxP1L with the new address.

31.3.2 ADDRESS MODE RECEIVE

The UART receiver is enabled for asynchronous address operation by configuring the following control bits:

- RXEN = 1
- MODE<3:0> = 0100
- UxBRGH:L = desired baud rate
- RXPPS = code for desired input pin
- Input pin ANSEL bit = 0
- UxP2L = receiver address
- UxP3L = address mask
- ON = 1

In Address mode, no data will be transferred to the input FIFO until a valid address is received. This is the default state. Any of the following conditions will cause the UART to revert to the default state:

- ON = 0
- RXEN = 0
- Received address does not match

When a character with the 9th bit set is received, the Least Significant eight bits of that character will be qualified by the values in the UxP2L and UxP3L registers.

The byte is XOR'd with UxP2L then AND'd with UxP3L. A match occurs when the result is 0h, in which case, the unaltered received character is stored in the receive FIFO, thereby setting the UxRXIF interrupt bit. The 9th bit is stored in the corresponding PERIF bit, identifying this byte as an address.

An address match also enables the receiver for all data such that all subsequent characters without the 9th bit set will be stored in the receive FIFO.

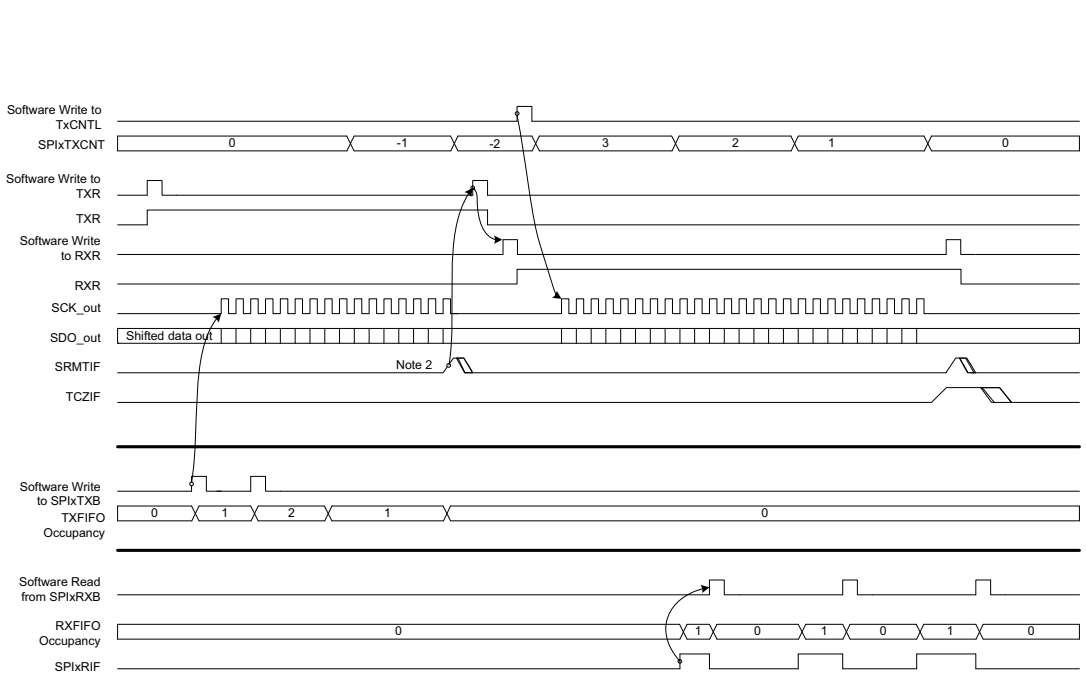
When the 9th bit is set and a match does not occur, the character is not stored in the receive FIFO and all subsequent data is ignored.

The UxP3L register mask allows a range of addresses to be accepted. Software can then determine the sub-address of the range by processing the received address character.

32.5.3 RECEIVE ONLY MODE

When RXR is set and TXR is clear, the SPI master is in Receive Only mode. In this mode, data transfers when the RXFIFO is not full and the Transfer Counter is non-zero. In this mode, writing a value to SPIxTCNTL will start the clocks for transfer. The clocks will suspend while the RXFIFO is full and cease when the SPIxTCNT reaches zero (see **Section 32.4 "Transfer Counter"**). If there is any data in the TXFIFO, the first data written to the TXFIFO will be transmitted on each data exchange, although the TXFIFO occupancy will not change, meaning that the same message will be sent on each transmission. If there is no data in the TXFIFO, the most recently received data will instead be transmitted. Figure 32-5 shows an example of sending a command using **Section 32.5.2 "Transmit Only Mode"** and then receiving a byte of data using this mode.

FIGURE 32-5: SPI MASTER OPERATION, COMMAND+READ DATA, TXR/RXR=0/1



Note: 1. SS(out) is not shown
2. Software must wait for shift-register empty (SRMTIF) before changing TXR, RXR, BYTES and BITS controls. This is not considered an imposition in this case, because the slave probably needs time to load output data (see also Figure 4-14).

32.5.4 TRANSFER OFF MODE

When both TXR and RXR are cleared, the SPI master is in Transfer Off mode. In this mode, SCK will not toggle and no data is exchanged. However, writes to SPIxTXB will be transferred to the TXFIFO which will be transmitted if the TXR bit is set.

33.3.2 BYTE FORMAT

All communication in I^2C 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 by the receiver. After the 8th falling edge of the SCL line, the device transmitting data on the SDA line releases control of that pin to an input, and reads in an acknowledge value on the next clock pulse. The clock signal is provided by the master. Data is valid to change while the SCL line is low, and sampled on the rising edge of the clock. Changes on the SDA line while the SCL line is high define Start and Stop conditions on the bus which are explained further in the chapter.

33.3.3 SDA AND SCL PINS

The user must configure these pins as open-drain inputs. This is done by clearing the appropriate TRIS bits and setting the appropriate and ODCON bits. The user may also select the input threshold, slew-rate and internal pull-up settings using the RxyI2C control registers (Register 16-9).

33.3.4 SDA HOLD TIME

The hold time of the SDA pin is selected by the SDAHT<1:0> bits of the I2CxCON2 register. Hold time is the time SDA is held valid after the falling edge of SCL. A longer hold time setting may help on buses with large capacitance.

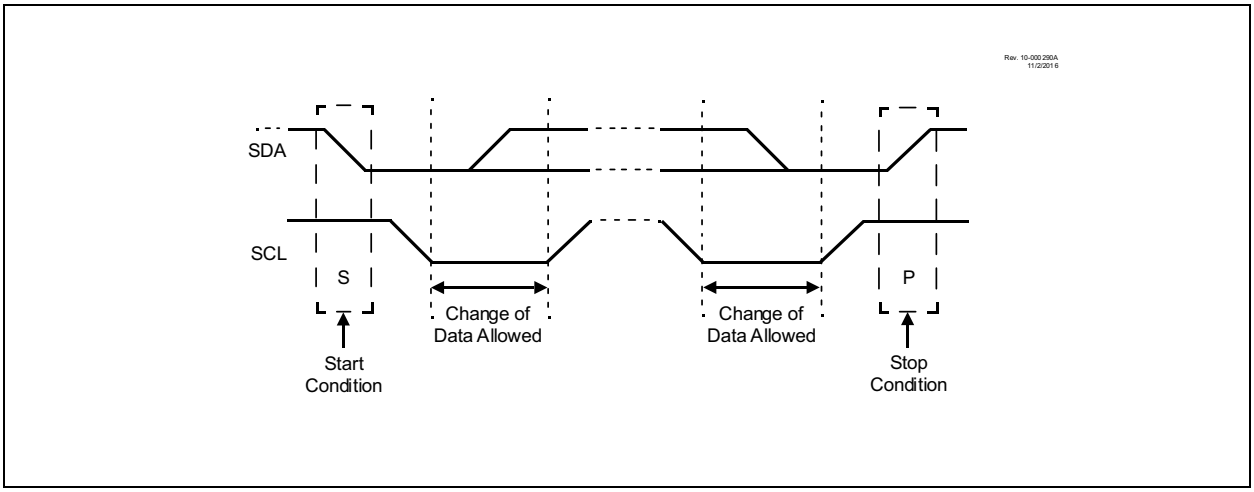
33.3.5 START CONDITION

The I^2C specification defines a Start condition as a transition of SDA line from a high to a low state while SCL line is high. A Start condition is always generated by the master and signifies the transition of the bus from an Idle to an Active state. Figure 33-3 shows waveforms for Start conditions. Master hardware waits for the BFRE bit of I2CxSTAT0 to be set, before asserting a Start condition on the SCL and SDA lines. If two masters assert a start at the same time, a collision will occur during the addressing phase.

33.3.6 STOP CONDITION

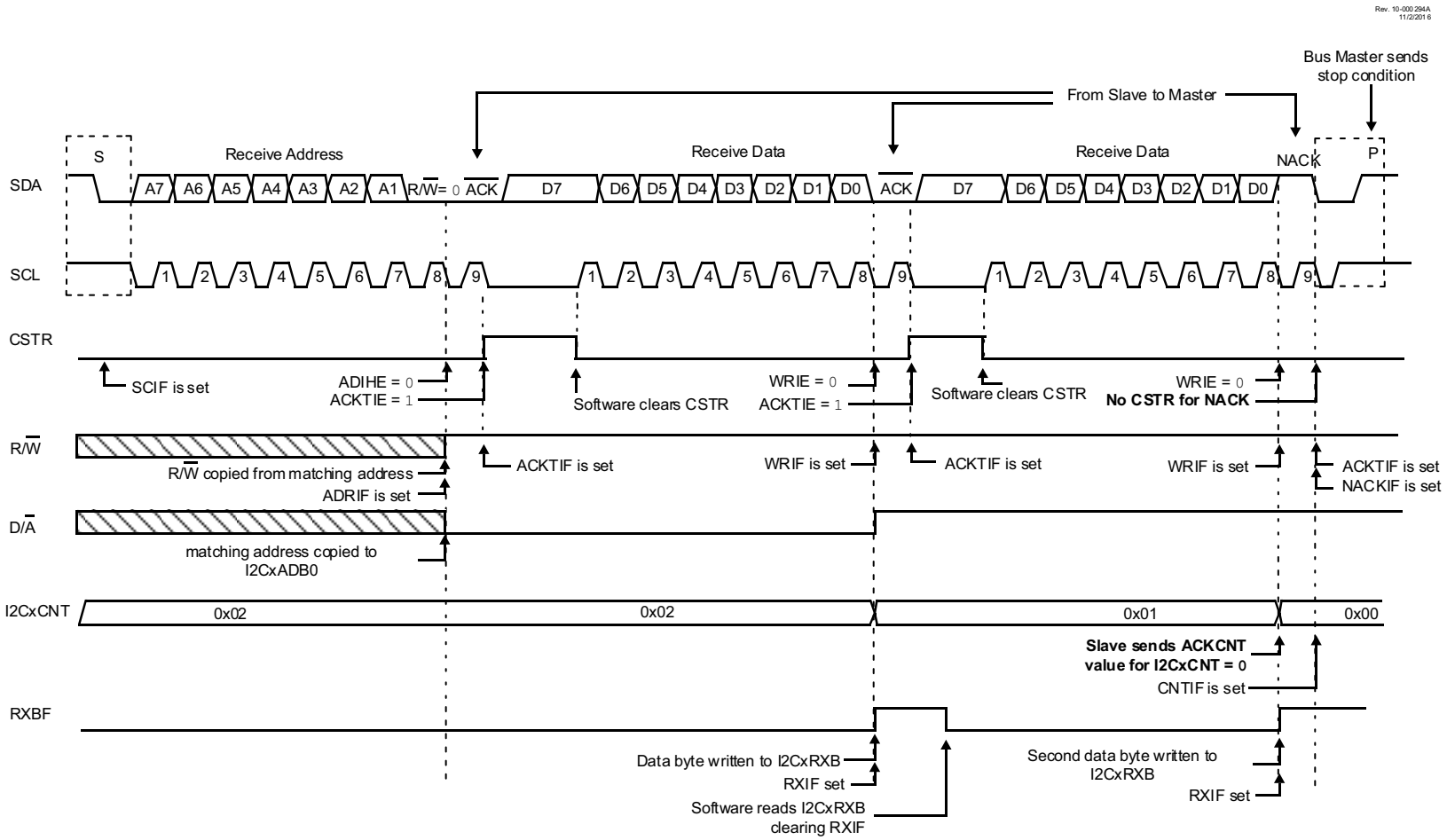
A Stop condition is a transition of the SDA line from low to high while the SCL line is high. Figure 33-3 shows waveforms for Stop conditions.

FIGURE 33-3: START AND STOP CONDITIONS



Note:	At least one SCL low time must appear before a Stop is valid. Therefore if the SDA line goes low then high again while the SCL line is high, only the Start condition is detected.

FIGURE 33-7: I²C SLAVE, 7-BIT ADDRESS, RECEPTION WITH I2CxCNT (ACKTIE = 1, ADRIE = 0, WRIE = 0)



36.1.5 INTERRUPTS

The ADC module allows for the ability to generate an interrupt upon completion of an Analog-to-Digital conversion. The ADC Interrupt Flag is the ADIF bit in the PIRx register. The ADC Interrupt Enable is the ADIE bit in the PIEx register. The ADIF bit must be cleared in software.

Note 1:	The ADIF bit is set at the completion of every conversion, regardless of whether or not the ADC interrupt is enabled.
2:	The ADC operates during Sleep only when the FRC oscillator is selected.

This interrupt can be generated while the device is operating or while in Sleep. If the device is in Sleep, the interrupt will wake up the device. Upon waking from Sleep, the next instruction following the SLEEP instruction is always executed. If the user is attempting to wake up from Sleep and resume in-line code execution, the ADIE bit of the PIEx register and the GIE bits of the INTCON0 register must both be set. If all these bits are set, the execution will switch to the Interrupt Service Routine.

36.1.6 RESULT FORMATTING

The 12-bit ADC conversion result can be supplied in two formats, left justified or right justified. The FM bits of the ADCON0 register controls the output format.

Figure 36-3 shows the two output formats.

Writes to the ADRES register pair are always right justified regardless of the selected format mode. Therefore, data read after writing to ADRES when ADFRM0 = 0 will be shifted left four places.

FIGURE 36-3: 12-BIT ADC CONVERSION RESULT FORMAT

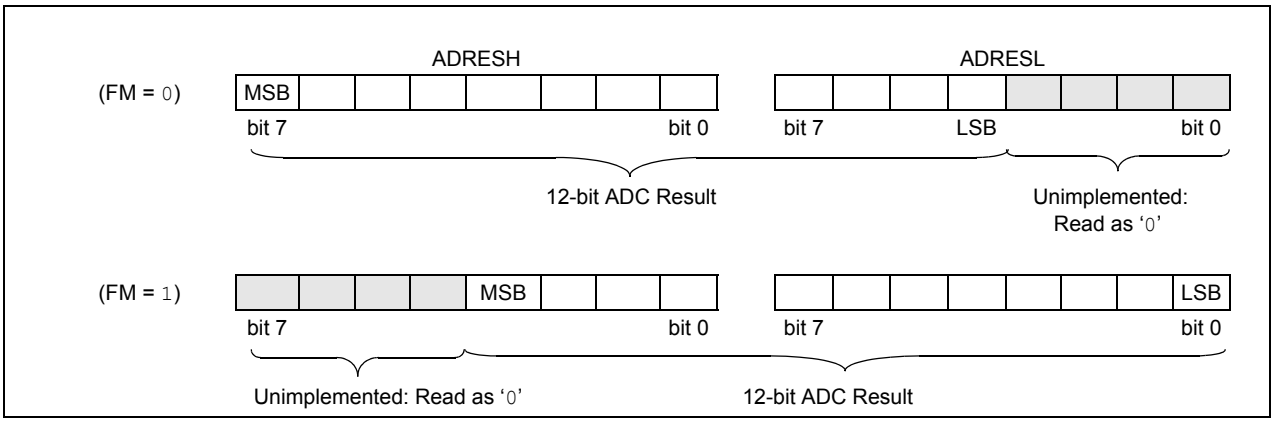
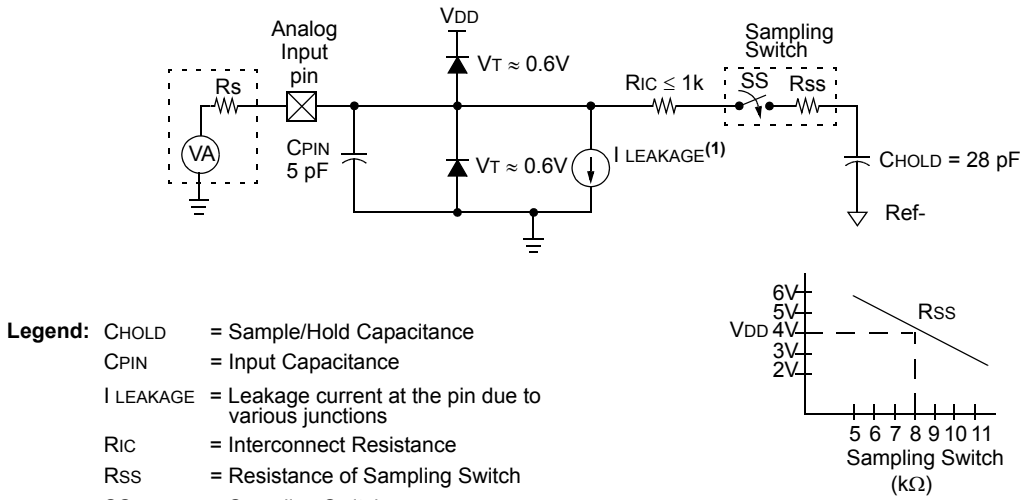


FIGURE 36-4: ANALOG INPUT MODEL

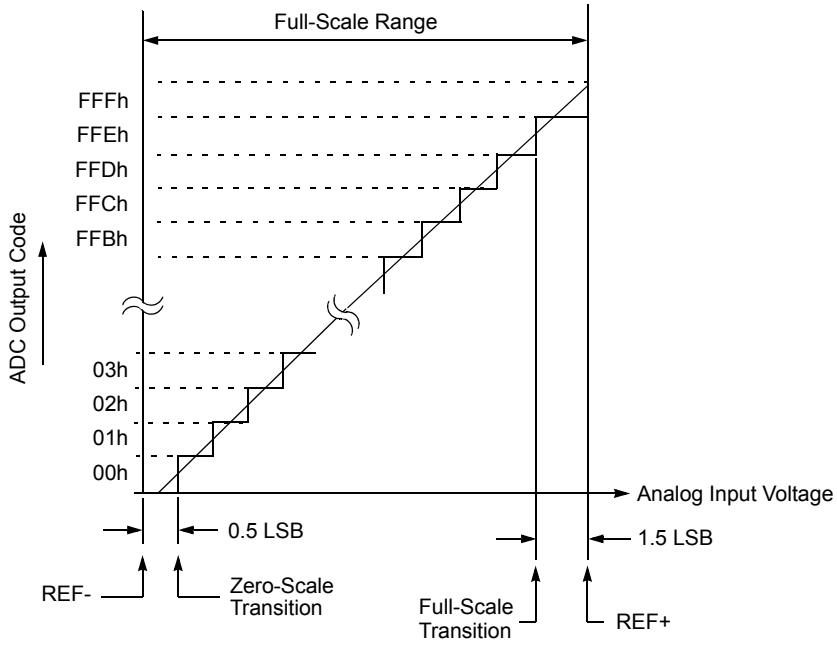


Legend:

CHOLD	= Sample/Hold Capacitance
CPIN	= Input Capacitance
I LEAKAGE	= Leakage current at the pin due to various junctions
RIC	= Interconnect Resistance
RSS	= Resistance of Sampling Switch
SS	= Sampling Switch
VT	= Threshold Voltage

Note 1: Refer to Table 44-6 (parameter D340 and D341).

FIGURE 36-5: ADC TRANSFER FUNCTION



41.2.2 EXTENDED INSTRUCTION SET

ADDULNK — Add Literal to FSR2 and Return

Syntax:	ADDULNK k
Operands:	$0 \le k \le 63$
Operation:	FSR2 + k → FSR2, (TOS) → PC
Status Affected:	None
Encoding:	1110 | 1000 | 11kk | kkkk
Description:	The 6-bit literal 'k' is added to the contents of FSR2. A RETURN is then executed by loading the PC with the TOS. The instruction takes two cycles to execute; a NOP is performed during the second cycle. This may be thought of as a special case of the ADDFSR instruction, where f = 3 (binary '11'); it operates only on FSR2.
Words:	1
Cycles:	2

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'k'	Process Data	Write to FSR
No Operation	No Operation	No Operation	No Operation

Example: ADDULNK 23h

Before Instruction
FSR2 = 03FFh
PC = 0100h

After Instruction
FSR2 = 0422h
PC = (TOS)

Note: All PIC18 instructions may take an optional label argument preceding the instruction mnemonic for use in symbolic addressing. If a label is used, the instruction syntax then becomes: {label} instruction argument(s).

TABLE 42-1: REGISTER FILE SUMMARY FOR PIC18(L)F26/27/45/46/47/55/56/57K42 DEVICES

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Register on page
3FA1h	T4HLT	PSYNC	CKPOL	CKSYNC	MODE					339
3FA0h	T4CON	ON	CKPS			OUTPS				338
3F9Fh	T4PR	PR4								337
3F9Eh	T4TMR	TMR4								337
3F9Dh	T5CLK	CS								335
3F9Ch	T5GATE	GSS								316
3F9Bh	T5GCON	GE	GPOL	GTM	GSPM	GGO	GVAL	—	—	314
3F9Ah	T5CON	—	—	CKPS		—	NOT_SYNC	RD16	ON	338
3F99h	TMR5H	TMR5H								317
3F98h	TMR5L	TMR5L								317
3F97h	T6RST	—	—	—	RSEL					336
3F96h	T6CLK	—	—	—	—	CS				315
3F95h	T6HLT	PSYNC	CKPOL	CKSYNC	MODE					339
3F94h	T6CON	ON	CKPS			OUTPS				338
3F93h	T6PR	PR6								337
3F92h	T6TMR	TMR6								337
3F91h - 3F80h	—	Unimplemented								
3F7Fh	CCP1CAP	CTS								352
3F7Eh	CCP1CON	EN	—	OUT	FMT	MODE				350
3F7Dh	CCPR1H	RH								353
3F7Ch	CCPR1L	RL								352
3F7Bh	CCP2CAP	CTS								352
3F7Ah	CCP2CON	EN	—	OUT	FMT	MODE				350
3F79h	CCPR2H	RH								353
3F78h	CCPR2L	RL								352
3F77h	CCP3CAP	CTS								352
3F76h	CCP3CON	EN	—	OUT	FMT	MODE				350
3F75h	CCPR3H	RH								353
3F74h	CCPR3L	RL								352
3F73h	CCP4CAP	CTS								352
3F72h	CCP4CON	EN	—	OUT	FMT	MODE				350
3F71h	CCPR4H	RH								353
3F70h	CCPR4L	RL								352
3F6Fh	—	Unimplemented								
3F6Eh	PWM5CON	EN	—	OUT	POL	—	—	—	—	358
3F6Dh	PWM5DCH	DC9	DC8	DC7	DC6	DC5	DC4	DC3	DC2	360
3F6Dh	PWM5DCH	DC8								360
3F6Ch	PWM5DCL	DC1	DC0	—	—	—	—	—	—	360
3F6Ch	PWM5DCL	DC		—	—	—	—	—	—	360
3F6Bh	—	Unimplemented								
3F6Ah	PWM6CON	EN	—	OUT	POL	—	—	—	—	358
3F69h	PWM6DCH	DC9		DC7	DC6	DC5	DC4	DC3	DC2	360
3F69h	PWM6DCH	DC								360
3F68h	PWM6DCL	DC1	DC0	—	—	—	—	—	—	360
3F68h	PWM6DCL	DC		—	—	—	—	—	—	360
3F67h	—	Unimplemented								

Legend: x = unknown, u = unchanged, — = unimplemented, q = value depends on condition

Note 1: Unimplemented in LF devices.
2: Unimplemented in PIC18(L)F26/27K42.
3: Unimplemented on PIC18(L)F26/27/45/46/47K42 devices.
4: Unimplemented in PIC18(L)F45/55K42.

TABLE 42-1: REGISTER FILE SUMMARY FOR PIC18(L)F26/27/45/46/47/55/56/57K42 DEVICES

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Register on page
3BFEh	DMA1AIRQ	—	AIRQ							256
3BFDh	DMA1CON1	EN	SIRQEN	DGO	—	—	AIRQEN	—	XIP	249
3BFCh	DMA1CON0	DMODE		DSTP	SMR		SMODE		SSTP	248
3BFBh	DMA1SSAU	—	—	SSA						251
3BFAh	DMA1SSAH	SSA								250
3BF9h	DMA1SSAL	SSA								250
3BF8h	DMA1SSZH	—	—	—	—	SSZ				252
3BF7h	DMA1SSZL	SSZ								252
3BF6h	DMA1SPTRU	—	—	SPTR						252
3BF5h	DMA1SPTRH	SPTR								251
3BF4h	DMA1SPTRL	SPTR								251
3BF3h	DMA1SCNTH	—	—	—	—	SCNT				253
3BF2h	DMA1SCNTL	SCNT								253
3BF1h	DMA1DSAH	DSA								254
3BF0h	DMA1DSAL	SSA								253
3BEFh	DMA1DSZH	—	—	—	—	DSZ				255
3BEEh	DMA1DSZL	DSZ								255
3BEDh	DMA1DPTRH	DPTR								254
3BECh	DMA1DPTRL	DPTR								254
3BEBh	DMA1DCNTH	—	—	—	—	DCNT				256
3BEAh	DMA1DCNTL	DCNT								255
3BE9h	DMA1BUF	BUF								250
3BE8h - 3BE0h	—	Unimplemented								
3BDFh	DMA2SIRQ	—	SIRQ							256
3BDEh	DMA2AIRQ	—	AIRQ							256
3BDDh	DMA2CON1	EN	SIRQEN	DGO	—	—	AIRQEN	—	XIP	249
3BDCh	DMA2CON0	DMODE		DSTP	SMR		SMODE		SSTP	248
3BDBh	DMA2SSAU	—	—	SSA						251
3BDAh	DMA2SSAH	SSA								250
3BD9h	DMA2SSAL	SSA								250
3BD8h	DMA2SSZH	—	—	—	—	SSZ				252
3BD7h	DMA2SSZL	SSZ								252
3BD6h	DMA2SPTRU	—	—	SPTR						252
3BD5h	DMA2SPTRH	SPTR								251
3BD4h	DMA2SPTRL	SPTR								251
3BD3h	DMA2SCNTH	—	—	—	—	SCNT				253
3BD2h	DMA2SCNTL	SCNT								253
3BD1h	DMA2DSAH	DSA								254
3BD0h	DMA2DSAL	SSA								253
3BCFh	DMA2DSZH	—	—	—	—	DSZ				255
3BCEh	DMA2DSZL	DSZ								255
3BCDh	DMA2DPTRH	DPTR								254
3BCCh	DMA2DPTRL	DPTR								254
3BCBh	DMA2DCNTH	—	—	—	—	DCNT				256
3BCAh	DMA2DCNTL	DCNT								255
3BC9h	DMA2BUF	BUF								250

Legend: x = unknown, u = unchanged, — = unimplemented, q = value depends on condition

Note 1: Unimplemented in LF devices.
2: Unimplemented in PIC18(L)F26/27K42.
3: Unimplemented on PIC18(L)F26/27/45/46/47K42 devices.
4: Unimplemented in PIC18(L)F45/55K42.

TABLE 44-15: ANALOG-TO-DIGITAL CONVERTER (ADC) ACCURACY SPECIFICATIONS(1,2)

Operating Conditions (unless otherwise stated) VDD = 3.0V, TA = 25°C, TAD = 1µs							
Param No.	**Sym.**	**Characteristic**	**Min.**	**Typ†**	**Max.**	**Units**	**Conditions**
AD01	NR	Resolution	—	—	12	bit	
AD02	EIL	Integral Error	—	±0.1	±2.0	LSb	ADCREF+ = 3.0V, ADCREF-= 0V
AD03	EDL	Differential Error	—	±0.1	±1.0	LSb	ADCREF+ = 3.0V, ADCREF-= 0V
AD04	EOFF	Offset Error	—	0.5	6.0	LSb	ADCREF+ = 3.0V, ADCREF-= 0V
AD05	EGN	Gain Error	—	±0.2	±6.0	LSb	ADCREF+ = 3.0V, ADCREF-= 0V
AD06	VADREF	ADC Reference Voltage (ADREF+ - ADREF-)	1.8	—	VDD	V	
AD07	VAIN	Full-Scale Range	ADREF-	—	ADREF+	V	
AD08	ZAIN	Recommended Impedance of Analog Voltage Source	—	1	—	kΩ	
AD09	RVREF	ADC Voltage Reference Ladder Impedance	—	50	—	kΩ	**Note 3**

* These parameters are characterized but not tested.

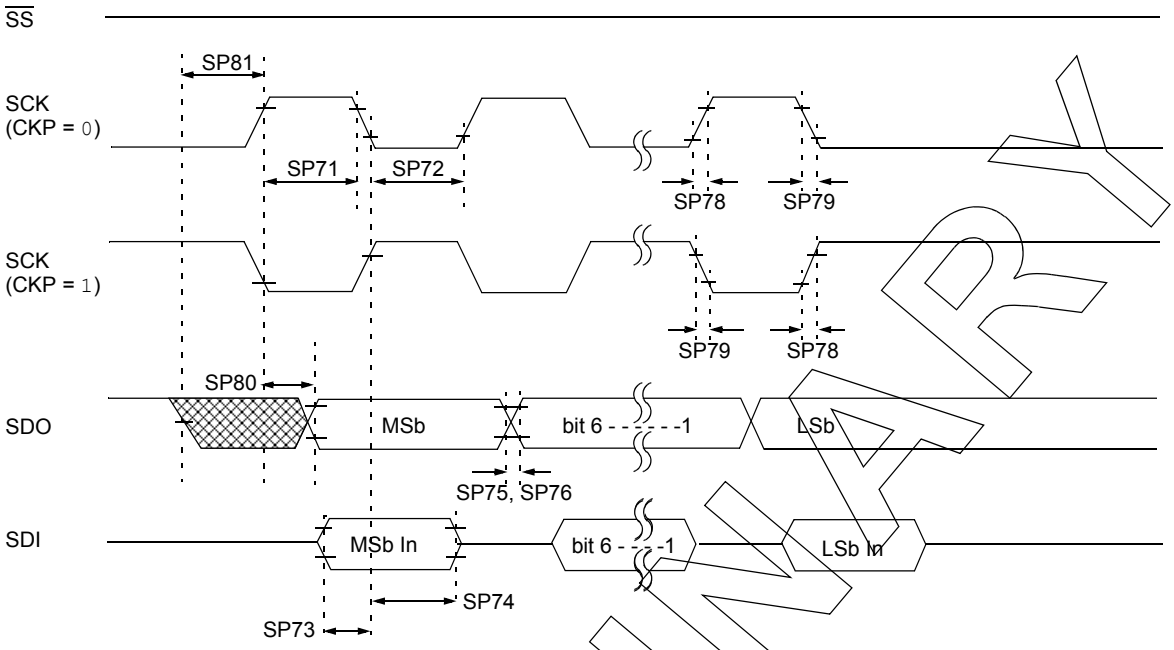
† Data in "Typ" column is at 3.0V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

Note 1: Total Absolute Error is the sum of the offset, gain and integral non-linearity (INL) errors.

2: The ADC conversion result never decreases with an increase in the input and has no missing codes.

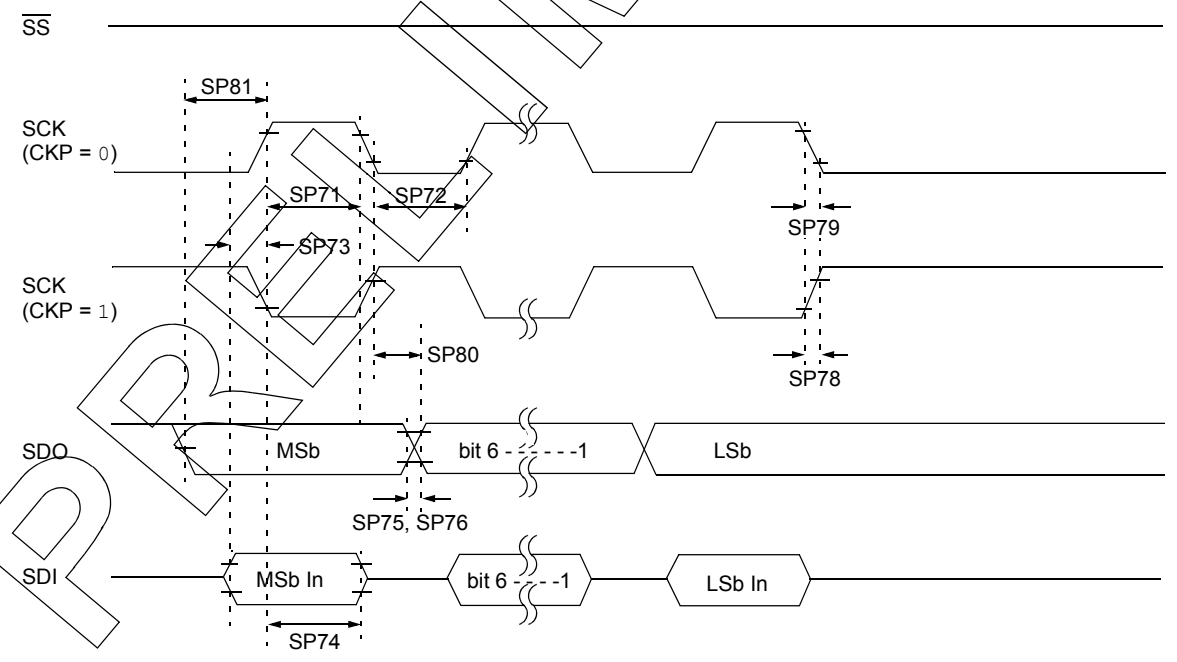
3: This is the impedance seen by the VREF pads when the external reference pads are selected.

FIGURE 44-14: SPI MASTER MODE TIMING (CKE = 0, SMP = 0)



Note: Refer to Figure 44-4 for load conditions.

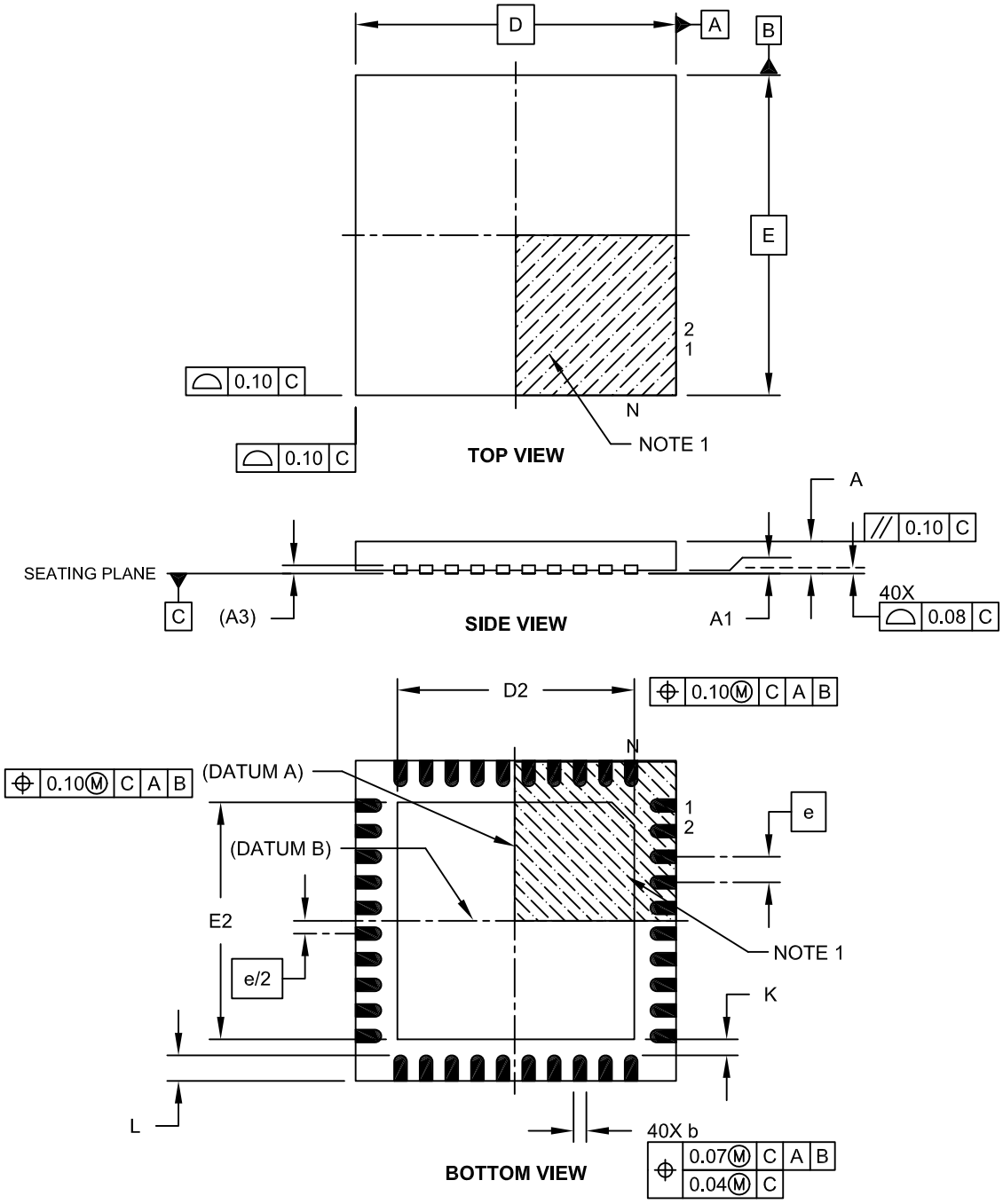
FIGURE 44-15: SPI MASTER MODE TIMING (CKE = 1, SMP = 1)



Note: Refer to Figure 44-4 for load conditions.

40-Lead Ultra Thin Plastic Quad Flat, No Lead Package (MV) – 5x5x0.5 mm Body [UQFN]

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



Microchip Technology Drawing C04-156A Sheet 1 of 2