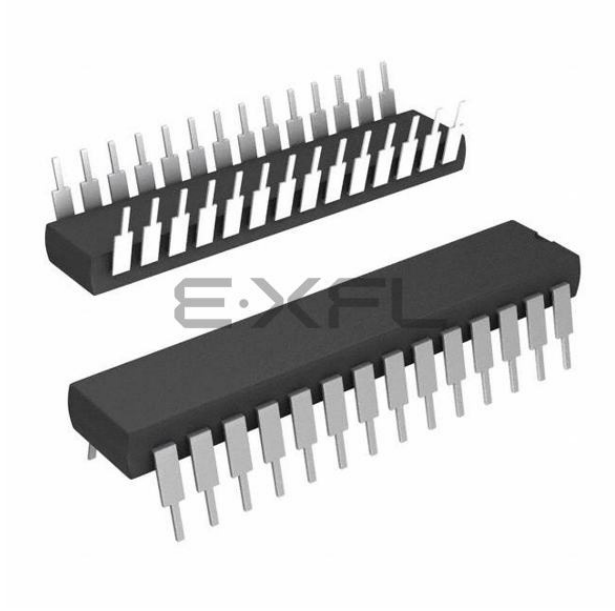


Welcome to E-XFL.COM

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	CANbus, I²C, LINbus, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, DMA, POR, PWM, WDT
Number of I/O	25
Program Memory Size	32KB (16K x 16)
Program Memory Type	FLASH
EEPROM Size	1K x 8
RAM Size	2K 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	Through Hole
Package / Case	28-DIP (0.300", 7.62mm)
Supplier Device Package	28-SPDIP
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic18f25k83-e-sp

TO OUR VALUED CUSTOMERS

It is our intention to provide our valued customers with the best documentation possible to ensure successful use of your Microchip products. To this end, we will continue to improve our publications to better suit your needs. Our publications will be refined and enhanced as new volumes and updates are introduced.

If you have any questions or comments regarding this publication, please contact the Marketing Communications Department via E-mail at **docerrors@microchip.com**. We welcome your feedback.

Most Current Data Sheet

To obtain the most up-to-date version of this data sheet, please register at our Worldwide Website at:

http://www.microchip.com

You can determine the version of a data sheet by examining its literature number found on the bottom outside corner of any page. The last character of the literature number is the version number, (e.g., DS30000000A is version A of document DS30000000).

Errata

An errata sheet, describing minor operational differences from the data sheet and recommended workarounds, may exist for current devices. As device/documentation issues become known to us, we will publish an errata sheet. The errata will specify the revision of silicon and revision of document to which it applies.

To determine if an errata sheet exists for a particular device, please check with one of the following:

- Microchip's Worldwide Website; **http://www.microchip.com**
- Your local Microchip sales office (see last page)

When contacting a sales office, please specify which device, revision of silicon and data sheet (include literature number) you are using.

Customer Notification System

Register on our website at **www.microchip.com** to receive the most current information on all of our products.

4.3.2 LOOK-UP TABLES IN PROGRAM MEMORY

There may be programming situations that require the creation of data structures, or look-up tables, in program memory. For PIC18 devices, look-up tables can be implemented in two ways:

- Computed GOTO
- Table Reads

4.3.2.1 Computed GOTO

A computed GOTO is accomplished by adding an offset to the Program Counter. An example is shown in Example 4-2.

A look-up table can be formed with an ADDWF PCL instruction and a group of RETLW nn instructions. The W register is loaded with an offset into the table before executing a call to that table. The first instruction of the called routine is the ADDWF PCL instruction. The next instruction executed will be one of the RETLW nn instructions that returns the value 'nn' to the calling function.

The offset value (in WREG) specifies the number of bytes that the Program Counter should advance and should be multiples of two (LSb = 0).

In this method, only one data byte may be stored in each instruction location and room on the return address stack is required.

EXAMPLE 4-2: COMPUTED GOTO USING AN OFFSET VALUE

```
        MOVF    OFFSET, W
        CALL    TABLE
ORG     nn00h
TABLE   ADDWF   PCL
        RETLW   nnh
        RETLW   nnh
        RETLW   nnh
        .
        .
        .
```

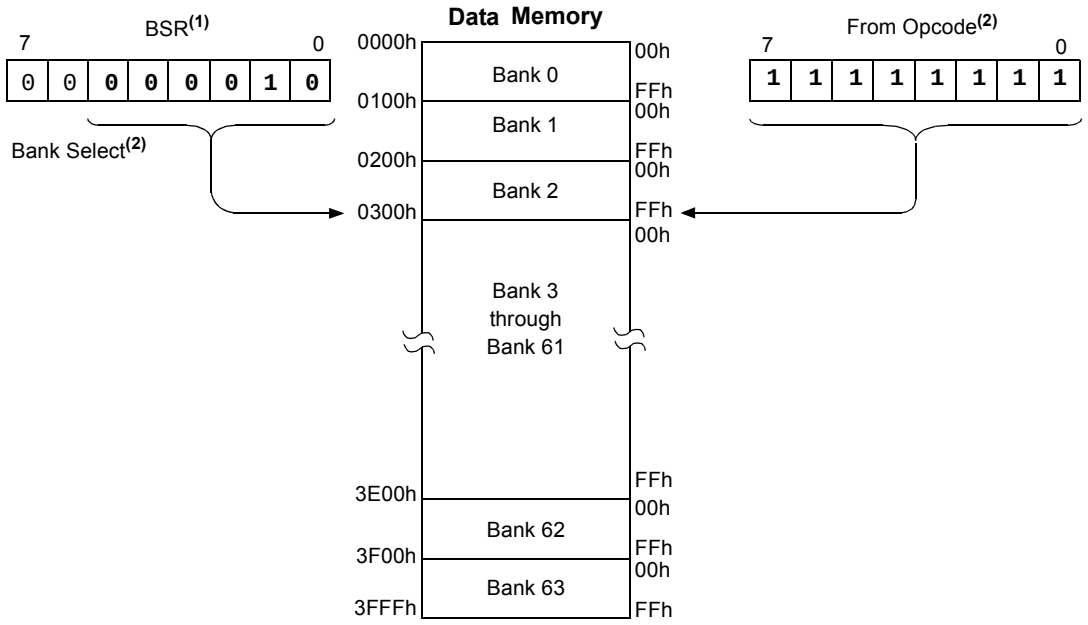
4.3.2.2 Table Reads and Table Writes

A better method of storing data in program memory allows two bytes of data to be stored in each instruction location.

Look-up table data may be stored two bytes per program word by using table reads and writes. The Table Pointer (TBLPTR) register specifies the byte address and the Table Latch (TABLAT) register contains the data that is read from or written to program memory.

Table read and table write operations are discussed further in **Section 13.1.1 "Table Reads and Table Writes"**.

FIGURE 4-5: USE OF THE BANK SELECT REGISTER (DIRECT ADDRESSING)



Note 1: The Access RAM bit of the instruction can be used to force an override of the selected bank (BSR<5:0>) to the registers of the Access Bank.

© 2017 Microchip Technology Inc.

6.4 Low-Power Brown-out Reset (LPBOR)

The Low-Power Brown-out Reset (LPBOR) provides an additional BOR circuit for low power operation. Refer to Figure 6-2 to see how the BOR interacts with other modules.

The LPBOR is used to monitor the external VDD pin. When too low of a voltage is detected, the device is held in Reset.

6.4.1 ENABLING LPBOR

The LPBOR is controlled by the LPBOREN bit of Configuration Word 2L. When the device is erased, the LPBOR module defaults to disabled.

6.4.1.1 LPBOR Module Output

The output of the LPBOR module is a signal indicating whether or not a Reset is to be asserted. This signal is OR'd together with the Reset signal of the BOR module to provide the generic BOR signal, which goes to the PCON0 register and to the power control block.

6.5 MCLR

The MCLR is an optional external input that can reset the device. The MCLR function is controlled by the MCLRE bit of Configuration Words and the LVP bit of Configuration Words (Table 6-2). The RMCLR bit in the PCON0 register will be set to '0' if a MCLR Reset has occurred.

TABLE 6-2: MCLR CONFIGURATION

MCLRE	LVP	MCLR
x	1	Enabled
1	0	Enabled
0	0	Disabled

6.5.1 MCLR ENABLED

When MCLR is enabled and the pin is held low, the device is held in Reset. The MCLR pin is connected to VDD through an internal weak pull-up.

The device has a noise filter in the MCLR Reset path. The filter will detect and ignore small pulses.

Note:	An internal Reset event (RESET instruction, BOR, WWDT, POR stack), does not drive the MCLR pin low.

6.5.2 MCLR DISABLED

When MCLR is disabled, the MCLR pin becomes input-only and pin functions such as internal weak pull-ups are under software control. See **Section 16.2 "I/O Priorities"** for more information.

6.6 Windowed Watchdog Timer (WWDT) Reset

The Windowed Watchdog Timer generates a Reset if the firmware does not issue a CLRWDT instruction within the time-out period or window set. The TO and PD bits in the STATUS register and the RWDT bit in the PCON0 register are changed to indicate a WWDT Reset. The WDTWV bit in the PCON0 register indicates if the WDT Reset has occurred due to a time out or a window violation. See **Section 11.0 "Windowed Watchdog Timer (WWDT)"** for more information.

6.7 RESET Instruction

A RESET instruction will cause a device Reset. The RI bit in the PCON0 register will be set to '0'. See Table 6-3 for default conditions after a RESET instruction has occurred.

6.8 Stack Overflow/Underflow Reset

The device can reset when the Stack Overflows or Underflows. The STKOVF or STKUNF bits of the PCON0 register indicate the Reset condition. These Resets are enabled by setting the STVREN bit in Configuration Words. See **Section 4.2.5 "Return Address Stack"** for more information.

6.9 Programming Mode Exit

Upon exit of Programming mode, the device will behave as if a POR has just occurred.

6.10 Power-up Timer (PWRT)

The Power-up Timer provides a selected time-out duration on POR or Brown-out Reset.

The device is held in Reset as long as PWRT is active. The PWRT delay allows additional time for the VDD to rise to an acceptable level. The Power-up Timer is selected by setting the PWRTS<1:0> Configuration bits, appropriately.

The Power-up Timer starts after the release of the POR and BOR/LPBOR if enabled, as shown in Figure 6-1.

REGISTER 7-2: OSCCON2: OSCILLATOR CONTROL REGISTER 2

U-0	R-f/f	R-f/f	R-f/f	R-f/f	R-f/f	R-f/f	R-f/f
—	COSC<2:0>			CDIV<3:0>			
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 **Unimplemented:** Read as ‘0’

bit 6-4 **COSC<2:0>:** Current Oscillator Source Select bits (read-only)(1)
Indicates the current source oscillator and PLL combination per Table 7-1.

bit 3-0 **CDIV<3:0>:** Current Divider Select bits (read-only)(1)
Indicates the current postscaler division ratio per Table 7-1.

Note 1: The POR value is the value present when user code execution begins.

REGISTER 7-3: OSCCON3: OSCILLATOR CONTROL REGISTER 3

R/W/HC-0/0	R/W-0/0	U-0	R-0/0	R-0/0	U-0	U-0	U-0
CSWHOLD	SOSCPWR	—	ORDY	NOSCR	—	—	—
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	HC = Bit is cleared by hardware

bit 7 **CSWHOLD:** Clock Switch Hold bit
1 = Clock switch will hold (with interrupt) when the oscillator selected by NOSC is ready
0 = Clock switch may proceed when the oscillator selected by NOSC is ready; NOSCR becomes ‘1’, the switch will occur

bit 6 **SOSCPWR:** Secondary Oscillator Power Mode Select bit
1 = Secondary oscillator operating in High-Power mode
0 = Secondary oscillator operating in Low-Power mode

bit 5 **Unimplemented:** Read as ‘0’

bit 4 **ORDY:** Oscillator Ready bit (read-only)
1 = OSCCON1 = OSCCON2; the current system clock is the clock specified by NOSC
0 = A clock switch is in progress

bit 3 **NOSCR:** New Oscillator is Ready bit (read-only)(1)
1 = A clock switch is in progress and the oscillator selected by NOSC indicates a “ready” condition
0 = A clock switch is not in progress, or the NOSC-selected oscillator is not yet ready

bit 2-0 **Unimplemented:** Read as ‘0’

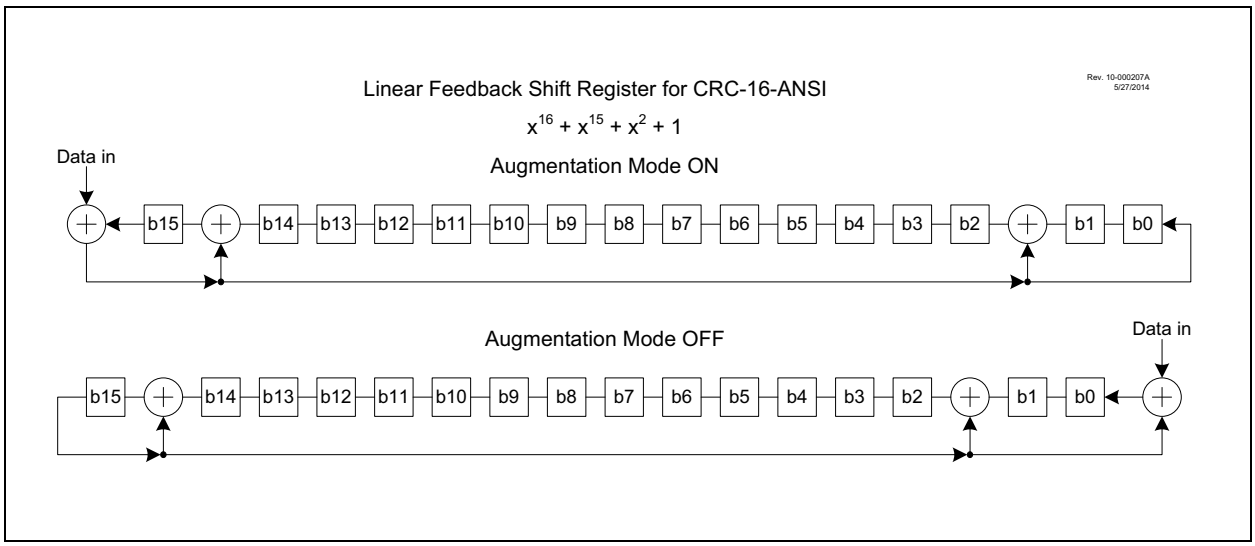
Note 1: If CSWHOLD = 0, the user may not see this bit set because, when the oscillator becomes ready there may be a delay of one instruction clock before this bit is set. The clock switch occurs in the next instruction cycle and this bit is cleared.

14.3 CRC Polynomial Implementation

Any polynomial can be used. The polynomial and accumulator sizes are determined by the PLEN<3:0> bits. For an n-bit accumulator, PLEN = n-1 and the corresponding polynomial is n+1 bits. Therefore the accumulator can be any size up to 16 bits with a corresponding polynomial up to 17 bits. The MSb and LSb of the polynomial are always '1' which is forced by hardware. All polynomial bits between the MSb and LSb are specified by the CRCXOR registers. For example, when using CRC-16-ANSI, the polynomial is defined as $X^{16}+X^{15}+X^2+1$.

The X^{16} and X^0 = 1 terms are the MSb and LSb controlled by hardware. The X^{15} and X^2 terms are specified by setting the corresponding CRCXOR<15:0> bits with the value of '0x8004'. The actual value is '0x8005' because the hardware sets the LSb to 1. However, the LSb of the CRCXORL register is unimplemented and always reads as '0'. Refer to Example 14-1.

EXAMPLE 14-2: CRC LFSR EXAMPLE



14.4 CRC Data Sources

Data can be input to the CRC module in two ways:

- User data using the CRCDAT registers (CRCDATH and CRCDATL)
- Program memory using the Program Memory Scanner

To set the number of bits of data, up to 16 bits, the DLEN bits of CRCCON1 must be set accordingly. Only data bits in CRCDAT registers up to DLEN will be used, other data bits in CRCDAT registers will be ignored.

Data is moved into the CRCSHIFT as an intermediate to calculate the check value located in the CRCACC registers.

The SHIFTM bit is used to determine the bit order of the data being shifted into the accumulator. If SHIFTM is not set, the data will be shifted in MSb first (Big Endian). The value of DLEN will determine the MSb. If SHIFTM bit is set, the data will be shifted into the accumulator in reversed order, LSb first (Little Endian).

The CRC module can be seeded with an initial value by setting the CRCACC<15:0> registers to the appropriate value before beginning the CRC.

14.4.1 CRC FROM USER DATA

To use the CRC module on data input from the user, the user must write the data to the CRCDAT registers. The data from the CRCDAT registers will be latched into the shift registers on any write to the CRCDATL register.

14.4.2 CRC FROM FLASH

To use the CRC module on data located in Program memory, the user can initialize the Program Memory Scanner as defined in **Section 14.8, Scanner Module Overview**.

21.3 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.

21.4 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 21.4.1 "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.

21.4.1 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.

21.5 Timer1/3/5 16-Bit Read/Write Mode

Timer1/3/5 can be configured to read and write all 16 bits of data, to and from, the 8-bit TMRxL and TMRxH registers, simultaneously. The 16-bit read and write operations are enabled by setting the RD16 bit of the TxCON register.

To accomplish this function, the TMRxH register value is mapped to a buffer register called the TMRxH buffer register. While in 16-Bit mode, the TMRxH register is not directly readable or writable and all read and write operations take place through the use of this TMRxH buffer register.

When a read from the TMRxL register is requested, the value of the TMRxH register is simultaneously loaded into the TMRxH buffer register. When a read from the TMRxH register is requested, the value is provided from the TMRxH buffer register instead. This provides the user with the ability to accurately read all 16 bits of the Timer1/3/5 value from a single instance in time. Reference the block diagram in Figure 21-2 for more details.

In contrast, when not in 16-Bit mode, the user must read each register separately and determine if the values have become invalid due to a rollover that may have occurred between the read operations.

When a write request of the TMRxL register is requested, the TMRxH buffer register is simultaneously updated with the contents of the TMRxH register. The value of TMRxH must be preloaded into the TMRxH buffer register prior to the write request for the TMRxL register. This provides the user with the ability to write all 16 bits to the TMRxL:TMRxH register pair at the same time.

Any requests to write to the TMRxH directly does not clear the Timer1/3/5 prescaler value. The prescaler value is only cleared through write requests to the TMRxL register.

FIGURE 31-7: MANCHESTER TIMING

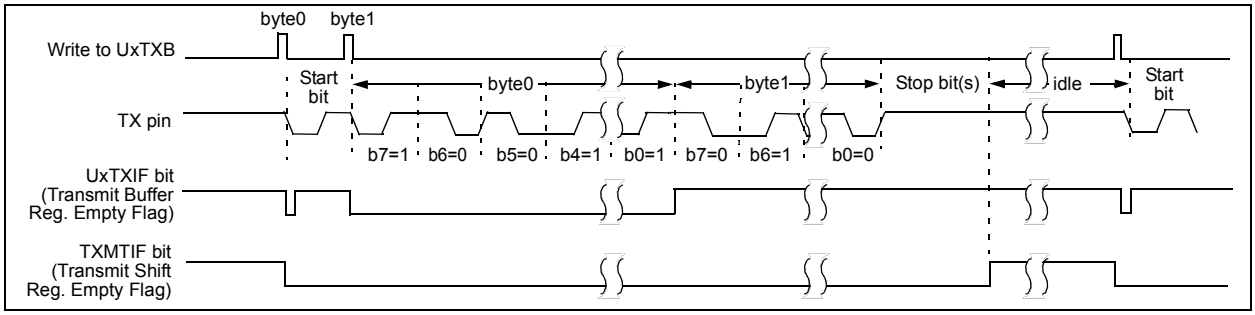


FIGURE 31-8: DALI FRAME TIMING

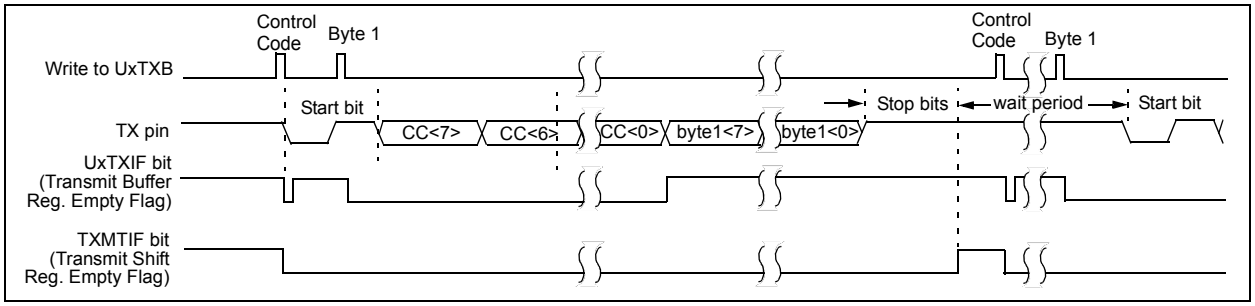


FIGURE 31-9: DALI FORWARD/BACK FRAME TIMING

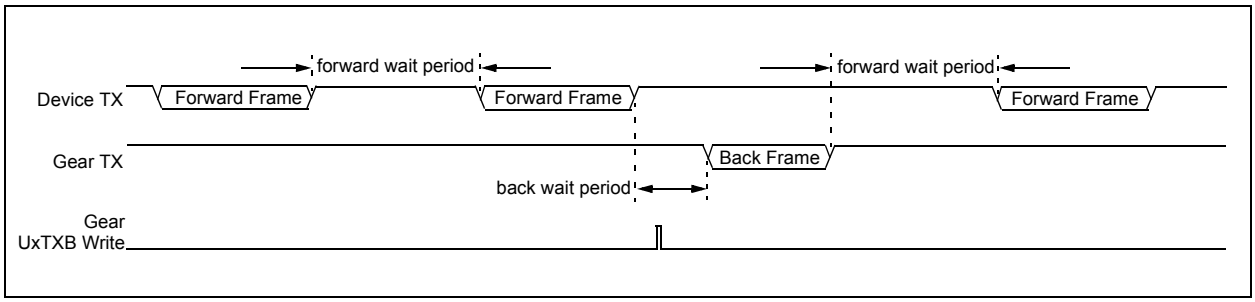


FIGURE 32-12: TRADITIONAL SPI DAISY – CHAIN CONNECTION

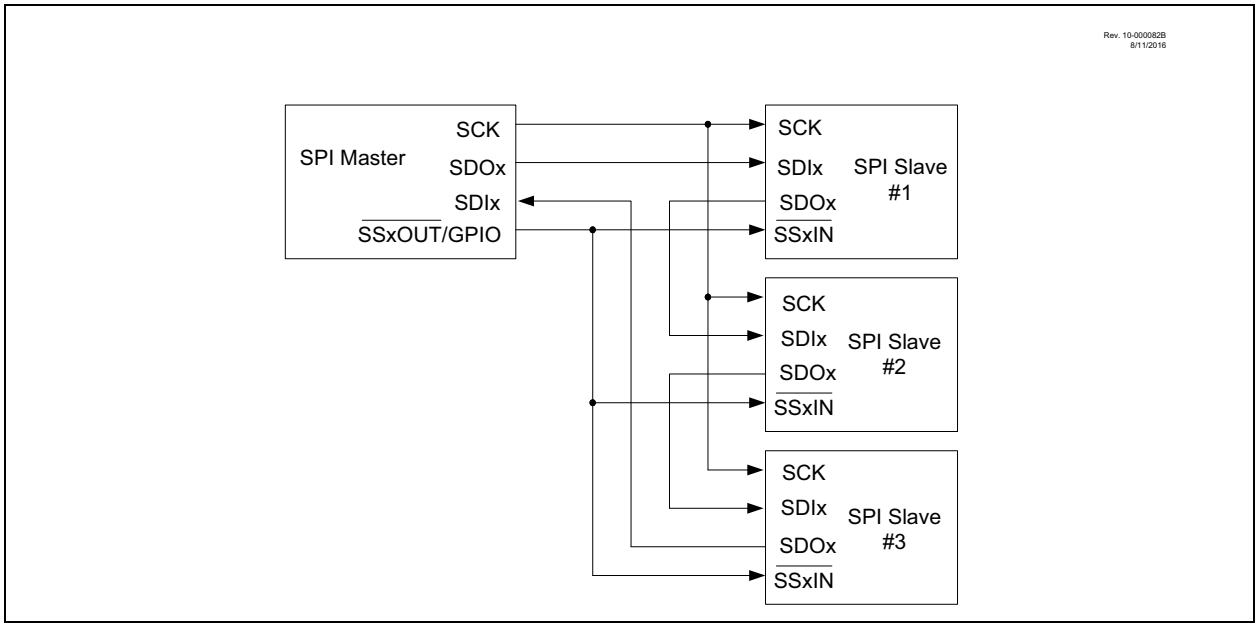
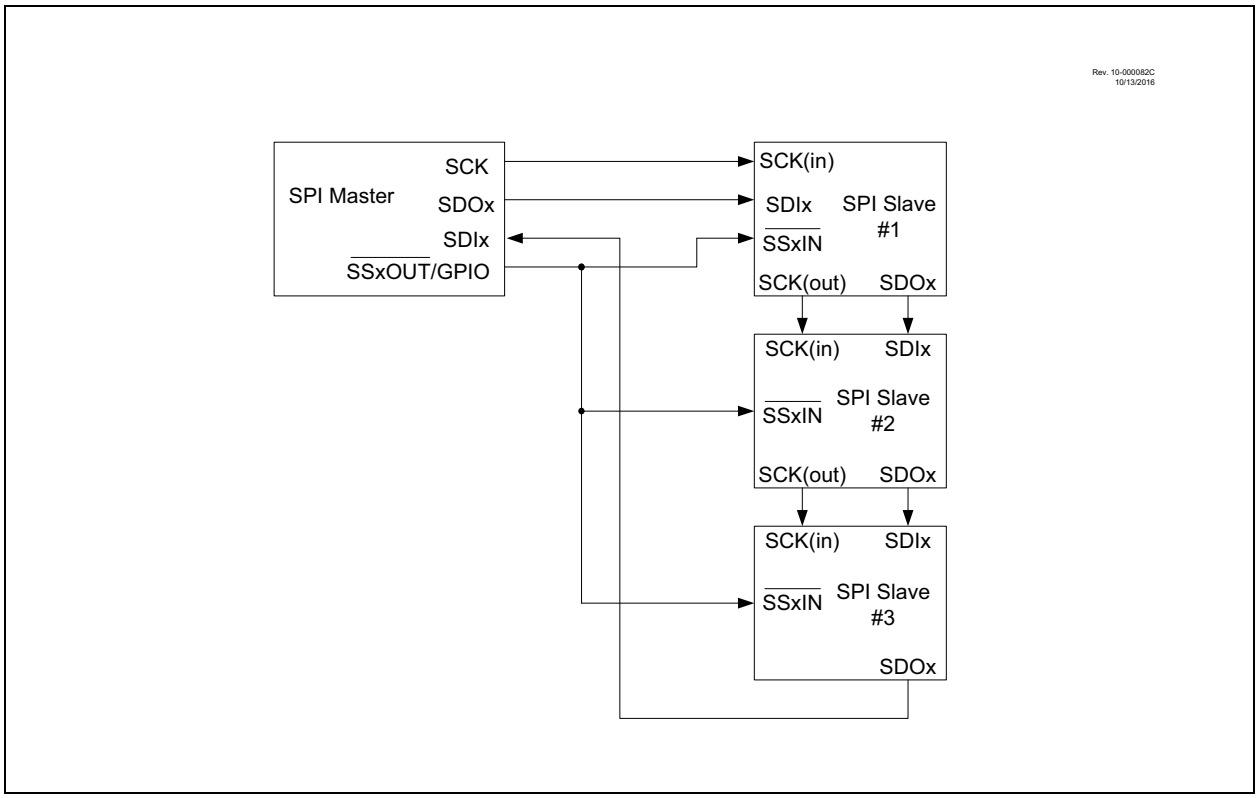


FIGURE 32-13: SPI DAISY-CHAIN CONNECTION WITH CHAINED SCK



REGISTER 33-9: I2CxCNT: I²C BYTE COUNT REGISTER

R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u
CNT<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	HS = Hardware set HC = Hardware clear

bit 7-0 **CNT<7:0>:** I²C Byte Count Register bits

If receiving data,

decremented 8th SCL edge, when a new data byte is loaded into I2CxRXB

If transmitting data,

decremented 9th SCL edge, when a new data byte is moved from I2CxTXB

CNTIF flag is set on 9th falling SCL edge, when I2CxCNT = 0. (Byte count cannot decrement past '0')

Note 1: It is recommended to write this register only when the module is IDLE (MMA = 0, SMA = 0) or when clock stretching (CSTR = 1 || MDR = 1).

messages regardless of the values of the acceptance filters. Also, if a message has an error before the end of frame, that portion of the message assembled in the MAB before the error frame will be loaded into the buffer. This mode may serve as a valuable debugging tool for a given CAN network. It should not be used in an actual system environment as the actual system will always have some bus errors and all nodes on the bus are expected to ignore them.

In Mode 1 and 2, when a programmable buffer is configured as a transmit buffer and one or more acceptance filters are associated with it, all incoming messages matching this acceptance filter criteria will be discarded. To avoid this scenario, user firmware must make sure that there are no acceptance filters associated with a buffer configured as a transmit buffer.

34.6.2 RECEIVE PRIORITY

When in Mode 0, RXB0 is the higher priority buffer and has two message acceptance filters associated with it. RXB1 is the lower priority buffer and has four acceptance filters associated with it. The lower number of acceptance filters makes the match on RXB0 more restrictive and implies a higher priority for that buffer. Additionally, the RXB0CON register can be configured such that if RXB0 contains a valid message and another valid message is received, an overflow error will not occur and the new message will be moved into RXB1 regardless of the acceptance criteria of RXB1. There are also two programmable acceptance filter masks available, one for each receive buffer (see **Section 34.4 "CAN Message Buffers"**).

In Mode 1 and 2, there are a total of 16 acceptance filters available and each can be dynamically assigned to any of the receive buffers. A buffer with a lower number has higher priority. Given this, if an incoming message matches with two or more receive buffer acceptance criteria, the buffer with the lower number will be loaded with that message.

34.6.3 ENHANCED FIFO MODE

When configured for Mode 2, two of the dedicated receive buffers in combination with one or more programmable transmit/receive buffers, are used to create a maximum of an eight buffers deep FIFO buffer. In this mode, there is no direct correlation between filters and receive buffer registers. Any filter that has been enabled can generate an acceptance. When a message has been accepted, it is stored in the next available receive buffer register and an internal Write Pointer is incremented. The FIFO can be a maximum of eight buffers deep. The entire FIFO must consist of contiguous receive buffers. The FIFO head begins at RXB0 buffer and its tail spans toward B5. The maximum length of the FIFO is limited by the presence or absence of the first transmit buffer starting from B0. If a buffer is configured as a transmit buffer, the FIFO length is reduced accordingly. For instance, if B3 is configured as a transmit buffer, the actual FIFO will consist of RXB0, RXB1, B0, B1 and B2, a total of five buffers. If B0 is configured as a transmit buffer, the FIFO length will be two. If none of the programmable buffers are configured as a transmit buffer, the FIFO will be eight buffers deep. A system that requires more transmit buffers should try to locate transmit buffers at the very end of B0-B5 buffers to maximize available FIFO length.

When a message is received in FIFO mode, the Interrupt Flag Code bits (EICODE<4:0>) in the CANSTAT register will have a value of '10000', indicating the FIFO has received a message. FIFO Pointer bits, FP<3:0> in the CANCON register, point to the buffer that contains data not yet read. The FIFO Pointer bits, in this sense, serve as the FIFO Read Pointer. The user should use the FP bits and read corresponding buffer data. When receive data is no longer needed, the RXFUL bit in the current buffer must be cleared, causing FP<3:0> to be updated by the module.

To determine whether FIFO is empty or not, the user may use the FP<3:0> bits to access the RXFUL bit in the current buffer. If RXFUL is cleared, the FIFO is considered to be empty. If it is set, the FIFO may contain one or more messages. In Mode 2, the module also provides a bit called FIFO High Water Mark (FIFOWM) in the ECANCON register. This bit can be used to cause an interrupt whenever the FIFO contains only one or four empty buffers. The FIFO high water mark interrupt can serve as an early warning to a full FIFO condition.

34.6.4 TIME-STAMPING

The CAN module can be programmed to generate a time-stamp for every message that is received. When enabled, the module generates a capture signal for the CCP modules, which in turn captures the value of Timer1, Timer3 or Timer5. This value can be used as the message time-stamp.

To use the time-stamp capability, set the CTS<3:0> bits of the appropriate CCPxCAP register to '1000' to configure the CCP module capture input to the CAN_rx_timestamp signal.

In addition, the CAN_rx_timestamp can be chosen as a signal input for the Signal Measurement Timer, which can be used for a variety of other timing applications.

EXAMPLE 34-4: TRANSMITTING A CAN MESSAGE USING WIN BITS

```
; Need to transmit Standard Identifier message 123h using TXB0 buffer.
; To successfully transmit, CAN module must be either in Normal or Loopback mode.
; TXB0 buffer is not in access bank. Use WIN bits to map it to RXB0 area.
MOVF   CANCON, W                    ; WIN bits are in lower 4 bits only. Read CANCON
                                    ; register to preserve all other bits. If operation
                                    ; mode is already known, there is no need to preserve
                                    ; other bits.
ANDLW  B'11110000'                  ; Clear WIN bits.
IORLW  B'00001000'                  ; Select Transmit Buffer 0
MOVWF  CANCON                       ; Apply the changes.
; Now TXB0 is mapped in place of RXB0. All future access to RXB0 registers will actually
; yield TXB0 register values.

; Load transmit data into TXB0 buffer.
MOVLW  MY_DATA_BYTE1                ; Load first data byte into buffer
MOVWF  RXB0D0                       ; Access TXB0D0 via RXB0D0 address.
; Load rest of the data bytes - up to 8 bytes into "TXB0" buffer using RXB0 registers.
...
; Load message identifier
MOVLW  60H                          ; Load SID2:SID0, EXIDE = 0
MOVWF  RXB0SIDL
MOVLW  24H                          ; Load SID10:SID3
MOVWF  RXB0SIDH
; No need to load RXB0EIDL:RXB0EIDH, as we are transmitting Standard Identifier Message only.

; Now that all data bytes are loaded, mark it for transmission.
MOVLW  B'00001000'                  ; Normal priority; Request transmission
MOVWF  RXB0CON

; If required, wait for message to get transmitted
BTFSC  RXB0CON, TXREQ               ; Is it transmitted?
BRA    $-2                          ; No. Continue to wait...

; Message is transmitted.
; If required, reset the WIN bits to default state.
```

REGISTER 37-3: ADCON2: ADC CONTROL REGISTER 2

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W/HC-0	R/W-0/0	R/W-0/0	R/W-0/0
PSIS	CRS<2:0>			ACLR	MD<2:0>		
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	HC = Bit is cleared by hardware

bit 7 **PSIS:** ADC Previous Sample Input Select bits
1 = PREV is the FLTR value at start-of-conversion
0 = PREV is the RES value at start-of-conversion

bit 6-4 **CRS<2:0>:** ADC Accumulated Calculation Right Shift Select bits
If ADMD = 100:
Low-pass filter time constant is 2^{ADCRS}, filter gain is 1:1
If ADMD = 001, 010 or 011:
The accumulated value is right-shifted by CRS (divided by 2^{ADCRS})**(1,2)**
Otherwise:
Bits are ignored

bit 3 **ACLR:** A/D Accumulator Clear Command bit**(3)**
1 = ACC, AOV and CNT registers are cleared
0 = Clearing action is complete (or not started)

bit 2-0 **MD<2:0>:** ADC Operating Mode Selection bits**(4)**
111-101 = Reserved
100 = Low-pass Filter mode
011 = Burst Average mode
010 = Average mode
001 = Accumulate mode
000 = Basic mode

Note 1: To correctly calculate an average, the number of samples (set in RPT) must be 2^{ADCRS}.
2: ADCRS = 3'b111 is a reserved option.
3: This bit is cleared by hardware when the accumulator operation is complete; depending on oscillator selections, the delay may be many instructions.
4: See Table 37-2 for Full mode descriptions.

REGISTER 37-6: ADCLK: ADC CLOCK SELECTION REGISTER

U-0	U-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
—	—	CS<5:0>					
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 7-6 **Unimplemented**: Read as '0'

bit 5-0 **CS<5:0>**: ADC Conversion Clock Select bits
111111 = FOSC/128
111110 = FOSC/126
111101 = FOSC/124
•
•
•
000000 = FOSC/2

REGISTER 37-7: ADREF: ADC REFERENCE SELECTION REGISTER

U-0	U-0	U-0	R/W-0/0	U-0	U-0	R/W-0/0	R/W-0/0
—	—	—	NREF	—	—	PREF<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	

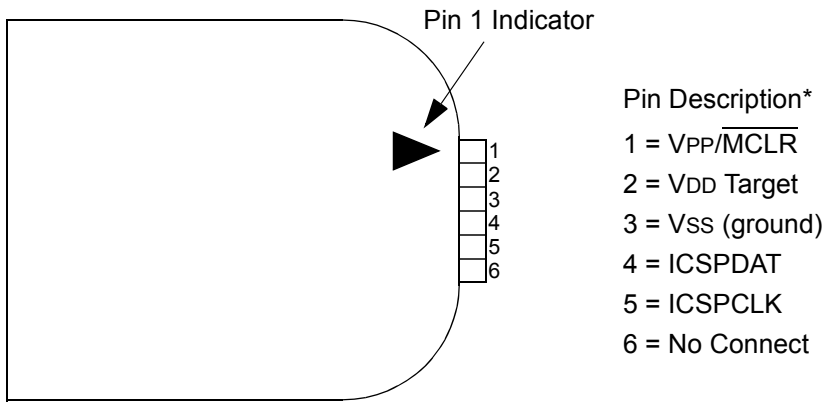
bit 7-5 **Unimplemented**: Read as '0'

bit 4 **NREF**: ADC Negative Voltage Reference Selection bit
1 = VREF- is connected to external VREF-
0 = VREF- is connected to VSS

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

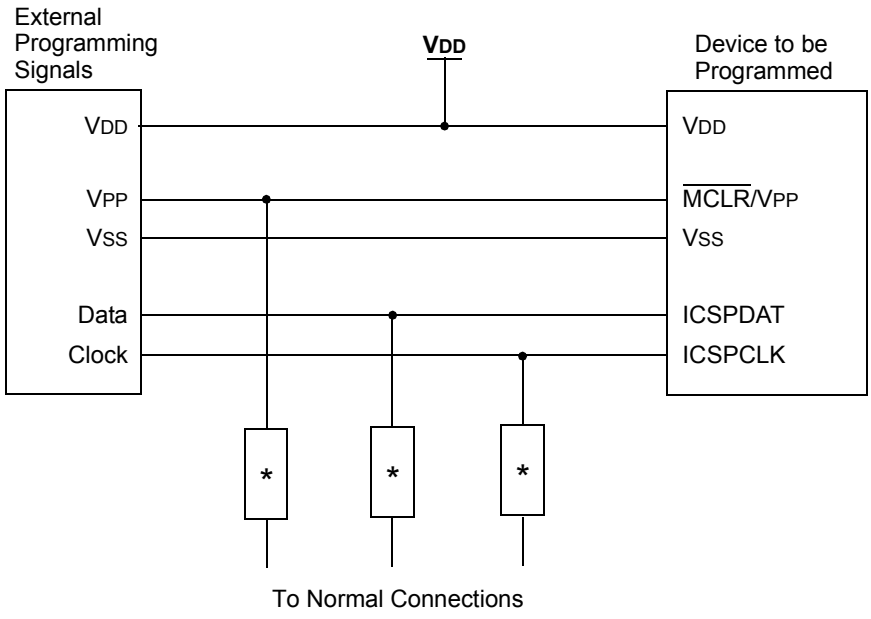
bit 1-0 **PREF**: ADC Positive Voltage Reference Selection bits
11 = VREF+ is connected to internal Fixed Voltage Reference (FVR) module
10 = VREF+ is connected to external VREF+
01 = Reserved
00 = VREF+ is connected to VDD

FIGURE 41-2: PICkit™ PROGRAMMER STYLE CONNECTOR INTERFACE



* The 6-pin header (0.100" spacing) accepts 0.025" square pins.

FIGURE 41-3: TYPICAL CONNECTION FOR ICSP™ PROGRAMMING



* Isolation devices (as required).

TABLE 42-2: INSTRUCTION SET (CONTINUED)

Mnemonic, Operands		Description	Cycles	16-Bit Instruction Word				Status Affected	Notes
				MSb			LSb		
CONTROL INSTRUCTIONS									
BC	n	Branch if Carry	1 (2)	1110	0010	nnnn	nnnn	None	1
BN	n	Branch if Negative	1 (2)	1110	0110	nnnn	nnnn	None	1
BNC	n	Branch if Not Carry	1 (2)	1110	0011	nnnn	nnnn	None	1
BNN	n	Branch if Not Negative	1 (2)	1110	0111	nnnn	nnnn	None	1
BNOV	n	Branch if Not Overflow	1 (2)	1110	0101	nnnn	nnnn	None	1
BNZ	n	Branch if Not Zero	2	1110	0001	nnnn	nnnn	None	1
BOV	n	Branch if Overflow	1 (2)	1110	0100	nnnn	nnnn	None	1
BRA	n	Branch Unconditionally	1 (2)	1101	0nnn	nnnn	nnnn	None	
BZ	n	Branch if Zero	1 (2)	1110	0000	nnnn	nnnn	None	1
CALL	n, s	Call subroutine 1st word	2	1110	110s	nnnn	nnnn	None	2
		2nd word		1111	nnnn	nnnn	nnnn		
GOTO	n	Go to address 1st word	2	1110	1111	nnnn	nnnn	None	2
	—	2nd word		1111	nnnn	nnnn	nnnn		
CALLW	—	W -> PCL and Call subroutine	2	0000	0000	0001	0100	None	1
RCALL	n	Relative Call	2	1101	1nnn	nnnn	nnnn	None	1
RETFIE	s	Return from interrupt enable	2	0000	0000	0001	000s	None	1
RETLW	k	Return with literal in WREG	2	0000	1100	kkkk	kkkk	None	1
RETURN	s	Return from Subroutine	2	0000	0000	0001	001s	None	1
INHERENT INSTRUCTIONS									
CLRWDT	—	Clear Watchdog Timer	1	0000	0000	0000	0100	None	
DAW	—	Decimal Adjust WREG	1	0000	0000	0000	0111	C	
NOP	—	No Operation	1	0000	0000	0000	0000	None	
NOP	—	No Operation	1	1111	xxxx	xxxx	xxxx	None	2
POP	—	Pop top of return stack (TOS)	1	0000	0000	0000	0110	None	
PUSH	—	Push top of return stack (TOS)	1	0000	0000	0000	0101	None	
RESET		Software device Reset	1	0000	0000	1111	1111	All	
SLEEP	—	Go into Standby mode	1	0000	0000	0000	0011	None	

Note 1: If Program Counter (PC) is modified or a conditional test is true, the instruction requires an additional cycle. The extra cycle is executed as a NOP.

2: Some instructions are multi word instructions. The second/third words of these instructions will be decoded as a NOP, unless the first word of the instruction retrieves the information embedded in these 16-bits. This ensures that all program memory locations have a valid instruction.

3: f_s and f_d do not cover the full memory range. 2 MSBs of bank selection are forced to ‘b00 to limit the range of these instructions to lower 4k addressing space.

SWAPF **Swap f**

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

Operands: $0 \le f \le 255$
$d \in [0,1]$
$a \in [0,1]$

Operation: (f<3:0>) → dest<7:4>,
(f<7:4>) → dest<3:0>

Status Affected: None

Encoding:

0011	10da	ffff	ffff

Description: The upper and lower nibbles of register 'f' are exchanged. If 'd' is '0', the result is placed in W. If 'd' is '1', the result is placed 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 42.2.3 "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: SWAPF REG, 1, 0

Before Instruction
REG = 53h
After Instruction
REG = 35h

TBLWT **Table Write**

Syntax: TBLWT (*; *+; *-; +*)

Operands: None

Operation:
if TBLWT*,
(TABLAT) → Holding Register;
TBLPTR – No Change;
if TBLWT*+,
(TABLAT) → Holding Register;
(TBLPTR) + 1 → TBLPTR;
if TBLWT*-,
(TABLAT) → Holding Register;
(TBLPTR) – 1 → TBLPTR;
if TBLWT+*,
(TBLPTR) + 1 → TBLPTR;
(TABLAT) → Holding Register;

Status Affected: None

Encoding:

0000	0000	0000	11nn nn=0 * =1 *+ =2 *- =3 +*

Description: This instruction uses the three LSBs of TBLPTR to determine which of the eight holding registers the TABLAT is written to. The holding registers are used to program the contents of Program Memory (P.M.). (Refer to **Section 13.1 "Program Flash Memory"** for additional details on programming Flash memory.)
The TBLPTR (a 21-bit pointer) points to each byte in the program memory. TBLPTR has a 2-MByte address range. The LSb of the TBLPTR selects which byte of the program memory location to access.

TBLPTR[0] = 0: Least Significant Byte of Program Memory Word
TBLPTR[0] = 1: Most Significant Byte of Program Memory Word

The TBLWT instruction can modify the value of TBLPTR as follows:

- no change
- post-increment
- post-decrement
- pre-increment

Words: 1

Cycles: 2

Q Cycle Activity:

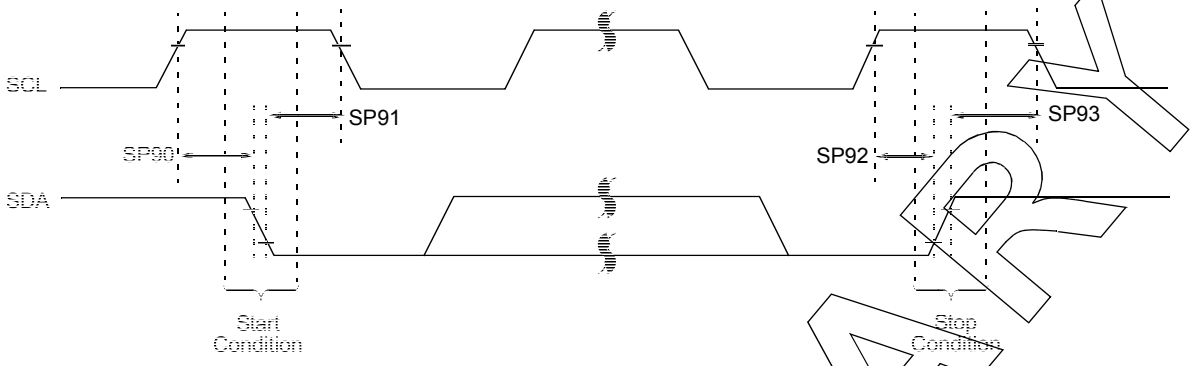
Q1	Q2	Q3	Q4
Decode	No operation	No operation	No operation
No operation	No operation (Read TABLAT)	No operation	No operation (Write to Holding Register)

TBLWT **Table Write (Continued)**

Example1:
```
TBLWT *+;
```
Before Instruction
TABLAT = 55h
TBLPTR = 00A356h
HOLDING REGISTER (00A356h) = FFh
After Instructions (table write completion)
TABLAT = 55h
TBLPTR = 00A357h
HOLDING REGISTER (00A356h) = 55h

Example 2:
```
TBLWT +*;
```
Before Instruction
TABLAT = 34h
TBLPTR = 01389Ah
HOLDING REGISTER (01389Ah) = FFh
HOLDING REGISTER (01389Bh) = FFh
After Instruction (table write completion)
TABLAT = 34h
TBLPTR = 01389Bh
HOLDING REGISTER (01389Ah) = FFh
HOLDING REGISTER (01389Bh) = 34h

FIGURE 45-18: I²C BUS START/STOP BITS TIMING



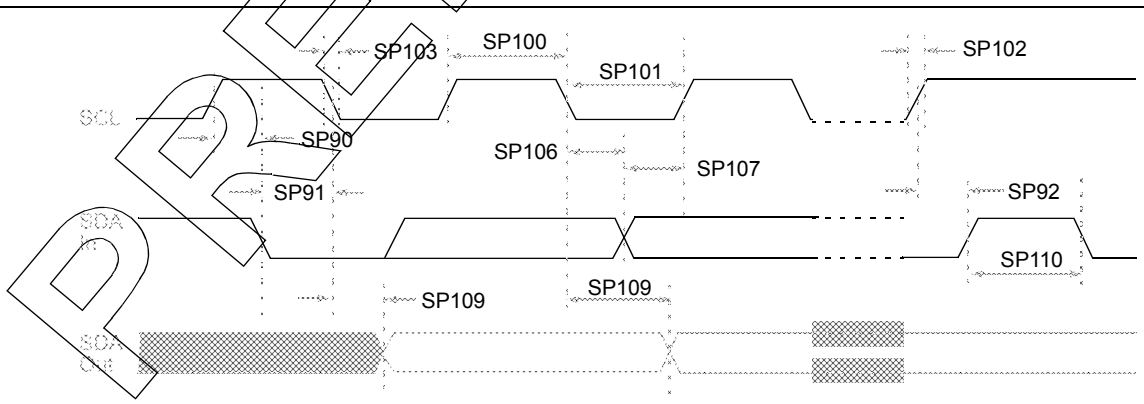
Note: Refer to Figure 45-4 for load conditions.

TABLE 45-22: I²C BUS START/STOP BITS REQUIREMENTS

Standard Operating Conditions (unless otherwise stated)								
Param No.	**Symbol**	**Characteristic**		**Min.**	**Typ**	**Max.**	**Units**	**Conditions**
SP90*	TSU:STA	Start condition Setup time	100 kHz mode	4700	—	—	ns	Only relevant for Repeated Start condition
			400 kHz mode	600	—	—		
SP91*	THD:STA	Start condition Hold time	100 kHz mode	4000	—	—	ns	After this period, the first clock pulse is generated
			400 kHz mode	600	—	—		
SP92*	TSU:STO	Stop condition Setup time	100 kHz mode	4700	—	—	ns	
			400 kHz mode	600	—	—		
SP93	THD:STO	Stop condition Hold time	100 kHz mode	4000	—	—	ns	
			400 kHz mode	600	—	—		

* These parameters are characterized but not tested.

FIGURE 45-19: I²C BUS DATA TIMING

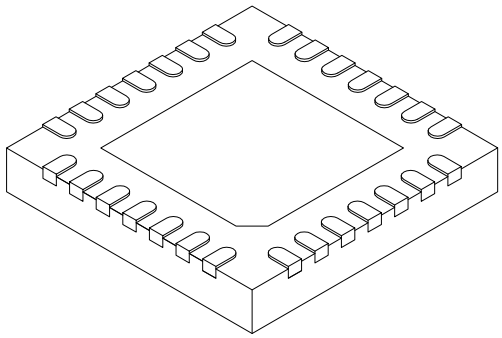


Note: Refer to Figure 45-4 for load conditions.

PRELIMINARY

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



	Units	MILLIMETERS		
Dimension	Limits	MIN	NOM	MAX
Number of Pins	N		28	
Pitch	e		0.65 BSC	
Overall Height	A	0.80	0.90	1.00
Standoff	A1	0.00	0.02	0.05
Terminal Thickness	A3		0.20 REF	
Overall Width	E		6.00 BSC	
Exposed Pad Width	E2	3.65	3.70	4.20
Overall Length	D		6.00 BSC	
Exposed Pad Length	D2	3.65	3.70	4.20
Terminal Width	b	0.23	0.30	0.35
Terminal Length	L	0.50	0.55	0.70
Terminal-to-Exposed Pad	K	0.20	-	-

Notes:

1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. Package is saw singulated
3. Dimensioning and tolerancing per ASME Y14.5M.
   BSC: Basic Dimension. Theoretically exact value shown without tolerances.
   REF: Reference Dimension, usually without tolerance, for information purposes only.

Microchip Technology Drawing C04-105C Sheet 2 of 2