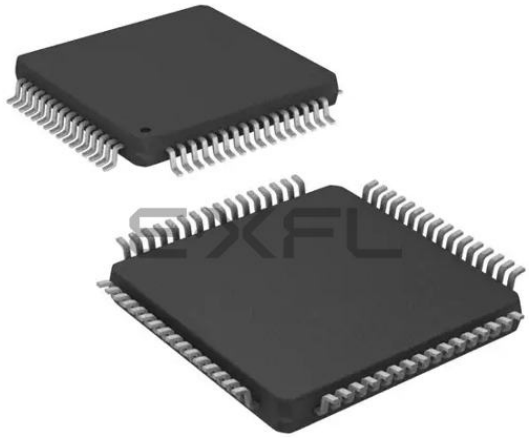


Welcome to E-XFL.COM

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	40MHz
Connectivity	CANbus, I²C, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, LVD, POR, PWM, WDT
Number of I/O	52
Program Memory Size	48KB (24K x 16)
Program Memory Type	FLASH
EEPROM Size	1K x 8
RAM Size	3.25K x 8
Voltage - Supply (Vcc/Vdd)	4.2V ~ 5.5V
Data Converters	A/D 12x10b
Oscillator Type	External
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	64-TQFP
Supplier Device Package	64-TQFP (10x10)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic18f6585-i-pt

2.6.1 SYSTEM CLOCK SWITCH BIT

The system clock source switching is performed under software control. The System Clock Switch bits, SCS1:SCS0 (OSCCON<1:0>), control the clock switching. When the SCS0 bit is '0', the system clock source comes from the main oscillator that is selected by the FOSC configuration bits in configuration register, CONFIG1H. When the SCS0 bit is set, the system clock source will come from the Timer1 oscillator. The SCS0 bit is cleared on all forms of Reset.

When FOSC bits are programmed for software PLL mode, the SCS1 bit can be used to select between primary oscillator/clock and PLL output. The SCS1 bit will only have an effect on the system clock if the PLL is enabled (PLLEN = 1) and locked (LOCK = 1), else it will be forced clear. When programmed with Configuration Controlled PLL mode, the SCS1 bit will be forced clear.

Note:	The Timer1 oscillator must be enabled and operating to switch the system clock source. The Timer1 oscillator is enabled by setting the T1OSCEN bit in the Timer1 Control register (T1CON). If the Timer1 oscillator is not enabled, then any write to the SCS0 bit will be ignored (SCS0 bit forced cleared) and the main oscillator will continue to be the system clock source.

REGISTER 2-1: OSCCON REGISTER

U-0	U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	—	LOCK	PLLEN	SCS1	SCS0
bit 7							bit 0

bit 7-4 **Unimplemented:** Read as '0'

bit 3 **LOCK:** Phase Lock Loop Lock Status bit
1 = Phase Lock Loop output is stable as system clock
0 = Phase Lock Loop output is not stable and output cannot be used as system clock

bit 2 **PLLEN(1):** Phase Lock Loop Enable bit
1 = Enable Phase Lock Loop output as system clock
0 = Disable Phase Lock Loop

bit 1 **SCS1:** System Clock Switch bit 1
When PLLEN and LOCK bits are set:
1 = Use PLL output
0 = Use primary oscillator/clock input pin
When PLLEN or LOCK bit is cleared:
Bit is forced clear.

bit 0 **SCS0(2):** System Clock Switch bit 0
When $\overline{\text{OSCSEN}}$ configuration bit = 0 and T1OSCEN bit = 1:
1 = Switch to Timer1 oscillator/clock pin
0 = Use primary oscillator/clock input pin
When $\overline{\text{OSCSEN}}$ and T1OSCEN are in other states:
Bit is forced clear.

Note 1: PLLEN bit is ignored when configured for ECIO+PLL and HS+PLL. This bit is used in ECIO+SPLL and HS+SPLL modes only.

2: The setting of SCS0 = 1 supersedes SCS1 = 1.

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

TABLE 3-3: INITIALIZATION CONDITIONS FOR ALL REGISTERS (CONTINUED)

Register	Applicable Devices		Power-on Reset, Brown-out Reset	$\overline{\text{MCLR}}$ Resets WDT Reset RESET Instruction Stack Resets	Wake-up via WDT or Interrupt
PORTJ	PIC18F6X8X	PIC18F8X8X	xxxx xxxx	uuuu uuuu	uuuu uuuu
PORTH	PIC18F6X8X	PIC18F8X8X	0000 xxxx	0000 uuuu	uuuu uuuu
PORTG	PIC18F6X8X	PIC18F8X8X	--xx xxxx	--uu uuuu	--uu uuuu
PORTF	PIC18F6X8X	PIC18F8X8X	x000 0000	u000 0000	u000 0000
PORTE	PIC18F6X8X	PIC18F8X8X	---- -000	---- -000	---- -uuu
PORTD	PIC18F6X8X	PIC18F8X8X	xxxx xxxx	uuuu uuuu	uuuu uuuu
PORTC	PIC18F6X8X	PIC18F8X8X	xxxx xxxx	uuuu uuuu	uuuu uuuu
PORTB	PIC18F6X8X	PIC18F8X8X	xxxx xxxx	uuuu uuuu	uuuu uuuu
PORTA(5,6)	PIC18F6X8X	PIC18F8X8X	-x0x 0000(5)	-u0u 0000(5)	-uuu uuuu(5)
SPBRGH	PIC18F6X8X	PIC18F8X8X	0000 0000	0000 0000	uuuu uuuu
BAUDCON	PIC18F6X8X	PIC18F8X8X	-1-0 0-00	-1-0 0-00	-u-u u-uu
ECCP1DEL	PIC18F6X8X	PIC18F8X8X	0000 0000	0000 0000	uuuu uuuu
ECANCON	PIC18F6X8X	PIC18F8X8X	0001 0000	0001 0000	uuuu uuuu
TXERRCNT	PIC18F6X8X	PIC18F8X8X	0000 0000	0000 0000	uuuu uuuu
RXERRCNT	PIC18F6X8X	PIC18F8X8X	0000 0000	0000 0000	uuuu uuuu
COMSTAT	PIC18F6X8X	PIC18F8X8X	0000 0000	0000 0000	uuuu uuuu
CIOCON	PIC18F6X8X	PIC18F8X8X	0000 ----	0000 ----	uuuu ----
BRGCON3	PIC18F6X8X	PIC18F8X8X	00-- -000	00-- -000	uu-- -uuu
BRGCON2	PIC18F6X8X	PIC18F8X8X	0000 0000	0000 0000	uuuu uuuu
BRGCON1	PIC18F6X8X	PIC18F8X8X	0000 0000	0000 0000	uuuu uuuu
CANCON	PIC18F6X8X	PIC18F8X8X	1000 000-	1000 000-	uuuu uuu-
CANSTAT	PIC18F6X8X	PIC18F8X8X	100- 000-	100- 000-	uuu- uuu-
RXB0D7	PIC18F6X8X	PIC18F8X8X	xxxx xxxx	uuuu uuuu	uuuu uuuu
RXB0D6	PIC18F6X8X	PIC18F8X8X	xxxx xxxx	uuuu uuuu	uuuu uuuu
RXB0D5	PIC18F6X8X	PIC18F8X8X	xxxx xxxx	uuuu uuuu	uuuu uuuu
RXB0D4	PIC18F6X8X	PIC18F8X8X	xxxx xxxx	uuuu uuuu	uuuu uuuu
RXB0D3	PIC18F6X8X	PIC18F8X8X	xxxx xxxx	uuuu uuuu	uuuu uuuu
RXB0D2	PIC18F6X8X	PIC18F8X8X	xxxx xxxx	uuuu uuuu	uuuu uuuu
RXB0D1	PIC18F6X8X	PIC18F8X8X	xxxx xxxx	uuuu uuuu	uuuu uuuu
RXB0D0	PIC18F6X8X	PIC18F8X8X	xxxx xxxx	uuuu uuuu	uuuu uuuu
RXB0DLC	PIC18F6X8X	PIC18F8X8X	-xxx xxxx	-uuu uuuu	-uuu uuuu

Legend: u = unchanged, x = unknown, - = unimplemented bit, read as '0', q = value depends on condition. Shaded cells indicate conditions do not apply for the designated device.

Note 1: One or more bits in the INTCONx or PIRx registers will be affected (to cause wake-up).

2: When the wake-up is due to an interrupt and the GIEL or GIEH bit is set, the PC is loaded with the interrupt vector (0008h or 0018h).

3: When the wake-up is due to an interrupt and the GIEL or GIEH bit is set, the TOSU, TOSH and TOSL are updated with the current value of the PC. The STKPTR is modified to point to the next location in the hardware stack.

4: See Table 3-2 for Reset value for specific condition.

5: Bit 6 of PORTA, LATA, and TRISA are enabled in ECIO and RCIO Oscillator modes only. In all other oscillator modes, they are disabled and read '0'.

6: Bit 6 of PORTA, LATA and TRISA are not available on all devices. When unimplemented, they read '0'.

7: This register reads all '0's until ECAN is set up in Mode 1 or Mode 2.

6.2 16-bit Mode

The external memory interface implemented in PIC18F8X8X devices operates only in 16-bit mode. The mode selection is not software configurable but is programmed via the configuration bits.

The WM<1:0> bits in the MEMCON register determine three types of connections in 16-bit mode. They are referred to as:

- 16-bit Byte Write
- 16-bit Word Write
- 16-bit Byte Select

These three different configurations allow the designer maximum flexibility in using 8-bit and 16-bit memory devices.

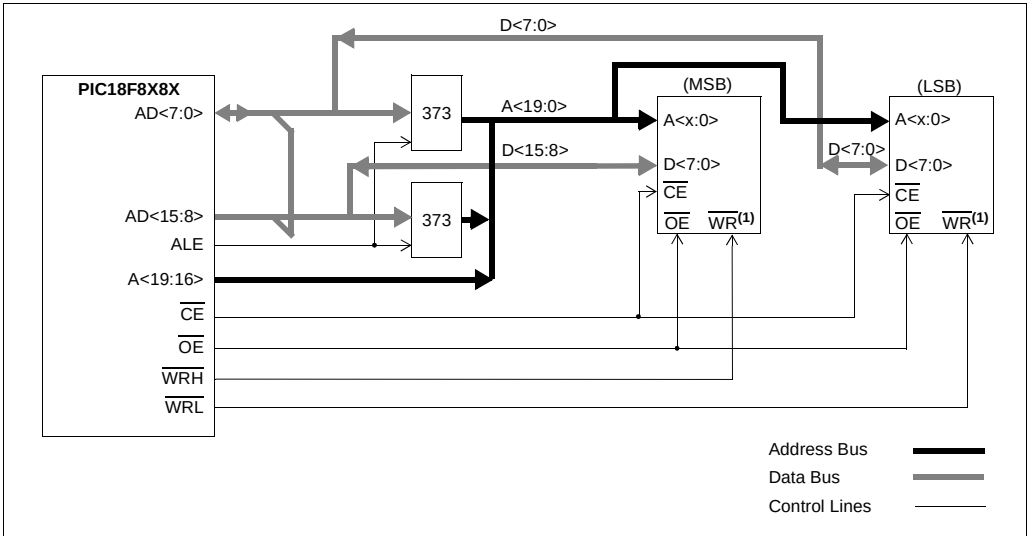
For all 16-bit modes, the Address Latch Enable (ALE) pin indicates that the Address bits (A<15:0>) are available on the external memory interface bus. Following the address latch, the Output Enable signal ($\overline{OE}$) will enable both bytes of program memory at once to form a 16-bit instruction word.

In Byte Select mode, JEDEC standard Flash memories will require BA0 for the byte address line, and one I/O line to select between Byte and Word mode. The other 16-bit modes do not need BA0. JEDEC standard static RAM memories will use the $\overline{UB}$ or $\overline{LB}$ signals for byte selection.

6.2.1 16-BIT BYTE WRITE MODE

Figure 6-1 shows an example of 16-bit Byte Write mode for PIC18F8X8X devices.

FIGURE 6-1: 16-BIT BYTE WRITE MODE EXAMPLE



Note 1: This signal only applies to table writes. See **Section 5.1 "Table Reads and Table Writes"**.

7.0 DATA EEPROM MEMORY

The data EEPROM is readable and writable during normal operation over the entire VDD range. The data memory is not directly mapped in the register file space. Instead, it is indirectly addressed through the Special Function Registers (SFR).

There are five SFRs used to read and write the program and data EEPROM memory. These registers are:

- EECON1
- EECON2
- EEDATA
- EEADR
- EEADRH

The EEPROM data memory allows byte read and write. When interfacing to the data memory block, EEDATA holds the 8-bit data for read/write and EEADR holds the address of the EEPROM location being accessed. These devices have 1024 bytes of data EEPROM with an address range from 0h to 3FFh.

The EEPROM data memory is rated for high erase/write cycles. A byte write automatically erases the location and writes the new data (erase-before-write). The write time is controlled by an on-chip timer. The write time will vary with voltage and temperature as well as from chip to chip. Please refer to parameter D122 (Electrical Characteristics, **Section 27.0 "Electrical Characteristics"**) for exact limits.

7.1 EEADRH:EEADR

The address register pair, EEADRH:EEADR, can address up to a maximum of 1024 bytes of data EEPROM.

7.2 EECON1 and EECON2 Registers

EECON1 is the control register for EEPROM memory accesses.

EECON2 is not a physical register. Reading EECON2 will read all '0's. The EECON2 register is used exclusively in the EEPROM write sequence.

Control bits RD and WR initiate read and write operations, respectively. These bits cannot be cleared, only set in software. They are cleared in hardware at the completion of the read or write operation. The inability to clear the WR bit in software prevents the accidental or premature termination of a write operation.

The WREN bit, when set, will allow a write operation. On power-up, the WREN bit is clear. The WRERR bit is set when a write operation is interrupted by a $\overline{\text{MCLR}}$ Reset or a WDT Time-out Reset during normal operation. In these situations, the user can check the WRERR bit and rewrite the location. It is necessary to reload the data and address registers (EEDATA and EEADR) due to the Reset condition forcing the contents of the registers to zero.

Note:	Interrupt flag bit, EEIF in the PIR2 register, is set when write is complete. It must be cleared in software.

7.5 Write Verify

Depending on the application, good programming practice may dictate that the value written to the memory should be verified against the original value. This should be used in applications where excessive writes can stress bits near the specification limit.

7.6 Protection Against Spurious Write

There are conditions when the device may not want to write to the data EEPROM memory. To protect against spurious EEPROM writes, various mechanisms have been built-in. On power-up, the WREN bit is cleared. Also, the Power-up Timer (72 ms duration) prevents EEPROM write.

The write initiate sequence and the WREN bit together help prevent an accidental write during brown-out, power glitch, or software malfunction.

7.7 Operation During Code-Protect

Data EEPROM memory has its own code-protect mechanism. External read and write operations are disabled if either of these mechanisms are enabled.

The microcontroller itself can both read and write to the internal data EEPROM regardless of the state of the code-protect configuration bit. Refer to **Section 24.0 "Special Features of the CPU"** for additional information.

7.8 Using the Data EEPROM

The data EEPROM is a high endurance, byte addressable array that has been optimized for the storage of frequently changing information (e.g., program variables or other data that are updated often). Frequently changing values will typically be updated more often than specification D124. If this is not the case, an array refresh must be performed. For this reason, variables that change infrequently (such as constants, IDs, calibration, etc.) should be stored in Flash program memory.

A simple data EEPROM refresh routine is shown in Example 7-3.

Note:	If data EEPROM is only used to store constants and/or data that changes rarely, an array refresh is likely not required. See specification D124.

EXAMPLE 7-3: DATA EEPROM REFRESH ROUTINE

```
        CLRF    EEADRH              ;
        CLRF    EEADR               ; Start at address 0
        BCF     EECON1, CFGS        ; Set for memory
        BCF     EECON1, EEPGD       ; Set for Data EEPROM
        BCF     INTCON, GIE         ; Disable interrupts
        BSF     EECON1, WREN        ; Enable writes
Loop                                ; Loop to refresh array
        BSF     EECON1, RD          ; Read current address
        MOVLW   55h                 ;
        MOVWF   EECON2              ; Write 55h
        MOVLW   0AAh                ;
        MOVWF   EECON2              ; Write 0AAh
        BSF     EECON1, WR          ; Set WR bit to begin write
        BTFSC   EECON1, WR          ; Wait for write to complete
        BRA     $-2
        INCFSZ  EEADR, F            ; Increment address
        BRA     Loop                ; Not zero, do it again

        INCFS2  EEADRH, F           ;
        BRA     Loop                ;
        BCF     EECON1, WREN        ; Disable writes
        BSF     INTCON, GIE         ; Enable interrupts
```

9.4 IPR Registers

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

REGISTER 9-10: IPR1: PERIPHERAL INTERRUPT PRIORITY REGISTER 1

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
PSPIP(1)	ADIP	RCIP	TXIP	SSPIP	CCP1IP	TMR2IP	TMR1IP
bit 7							bit 0

bit 7 **PSPIP:** Parallel Slave Port Read/Write Interrupt Priority bit(1)
1 = High priority
0 = Low priority

Note 1: Available in Microcontroller mode only.

bit 6 **ADIP**: A/D Converter Interrupt Priority bit
1 = High priority
0 = Low priority

bit 5 **RCIP**: USART Receive Interrupt Priority bit
1 = High priority
0 = Low priority

bit 4 **TXIP**: USART Transmit Interrupt Priority bit
1 = High priority
0 = Low priority

bit 3 **SSPIP**: Master Synchronous Serial Port Interrupt Priority bit
1 = High priority
0 = Low priority

bit 2 **CCP1IP**: CCP1 Interrupt Priority bit
1 = High priority
0 = Low priority

bit 1 **TMR2IP**: TMR2 to PR2 Match Interrupt Priority bit
1 = High priority
0 = Low priority

bit 0 **TMR1IP**: TMR1 Overflow Interrupt Priority bit
1 = High priority
0 = Low priority

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

9.5 RCON Register

The RCON register contains the IPEN bit which is used to enable prioritized interrupts. The functions of the other bits in this register are discussed in more detail in **Section 4.14 “RCON Register”**.

REGISTER 9-13: RCON REGISTER

R/W-0	U-0	U-0	R/W-1	R-1	R-1	R/W-0	R/W-0
IPEN	—	—	$\overline{RI}$	$\overline{TO}$	$\overline{PD}$	$\overline{POR}$	$\overline{BOR}$
bit 7							bit 0

bit 7 **IPEN:** Interrupt Priority Enable bit
1 = Enable priority levels on interrupts
0 = Disable priority levels on interrupts (PIC16 Compatibility mode)

bit 6-5 **Unimplemented:** Read as ‘0’

bit 4 **$\overline{RI}$:** RESET Instruction Flag bit
For details of bit operation, see Register 4-4.

bit 3 **$\overline{TO}$:** Watchdog Time-out Flag bit
For details of bit operation, see Register 4-4.

bit 2 **$\overline{PD}$:** Power-down Detection Flag bit
For details of bit operation, see Register 4-4.

bit 1 **$\overline{POR}$:** Power-on Reset Status bit
For details of bit operation, see Register 4-4.

bit 0 **$\overline{BOR}$:** Brown-out Reset Status bit
For details of bit operation, see Register 4-4.

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

11.0 TIMER0 MODULE

The Timer0 module has the following features:

- Software selectable as an 8-bit or 16-bit timer/counter
- Readable and writable
- Dedicated 8-bit software programmable prescaler
- Clock source selectable to be external or internal
- Interrupt-on-overflow from 0FFh to 00h in 8-bit mode and 0FFFFh to 0000h in 16-bit mode
- Edge select for external clock

Figure 11-1 shows a simplified block diagram of the Timer0 module in 8-bit mode and Figure 11-2 shows a simplified block diagram of the Timer0 module in 16-bit mode.

The T0CON register (Register 11-1) is a readable and writable register that controls all the aspects of Timer0, including the prescale selection.

Note:	Timer0 is enabled on POR.

REGISTER 11-1: T0CON: TIMER0 CONTROL REGISTER

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
TMR0ON	T08BIT	T0CS	T0SE	PSA	T0PS2	T0PS1	T0PS0
bit 7							bit 0

bit 7 **TMR0ON:** Timer0 On/Off Control bit
1 = Enables Timer0
0 = Stops Timer0

bit 6 **T08BIT**: Timer0 8-bit/16-bit Control bit
1 = Timer0 is configured as an 8-bit timer/counter
0 = Timer0 is configured as a 16-bit timer/counter

bit 5 **T0CS**: Timer0 Clock Source Select bit
1 = Transition on T0CKI pin
0 = Internal instruction cycle clock (CLKO)

bit 4 **T0SE**: Timer0 Source Edge Select bit
1 = Increment on high-to-low transition on T0CKI pin
0 = Increment on low-to-high transition on T0CKI pin

bit 3 **PSA**: Timer0 Prescaler Assignment bit
1 = TImer0 prescaler is not assigned. Timer0 clock input bypasses prescaler.
0 = Timer0 prescaler is assigned. Timer0 clock input comes from prescaler output.

bit 2-0 **T0PS2:T0PS0**: Timer0 Prescaler Select bits
111 = 1:256 prescale value
110 = 1:128 prescale value
101 = 1:64 prescale value
100 = 1:32 prescale value
011 = 1:16 prescale value
010 = 1:8 prescale value
001 = 1:4 prescale value
000 = 1:2 prescale value

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

FIGURE 17-5: SPI MODE WAVEFORM (SLAVE MODE WITH CKE = 0)

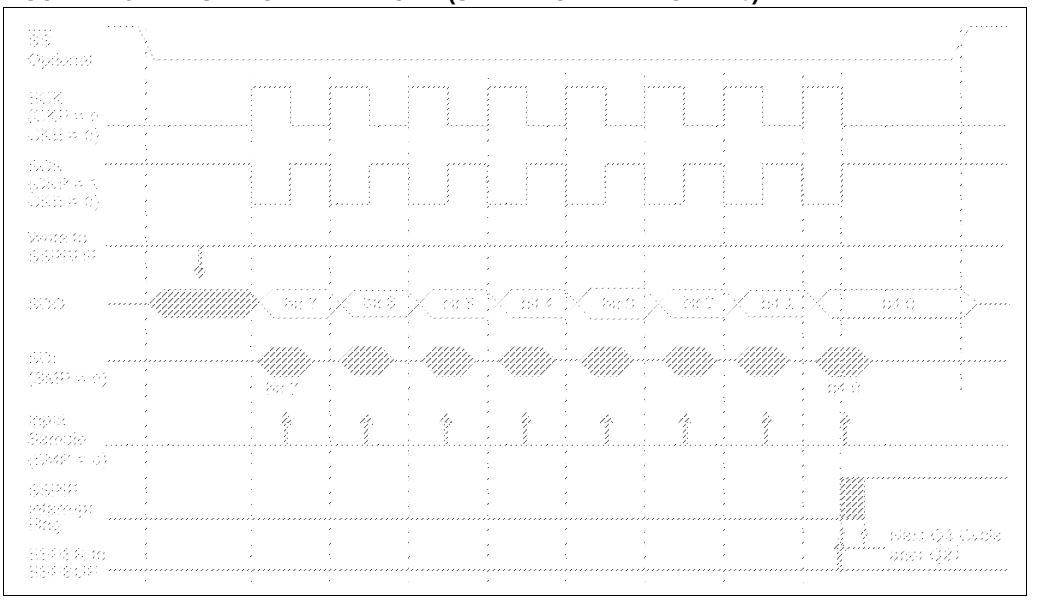
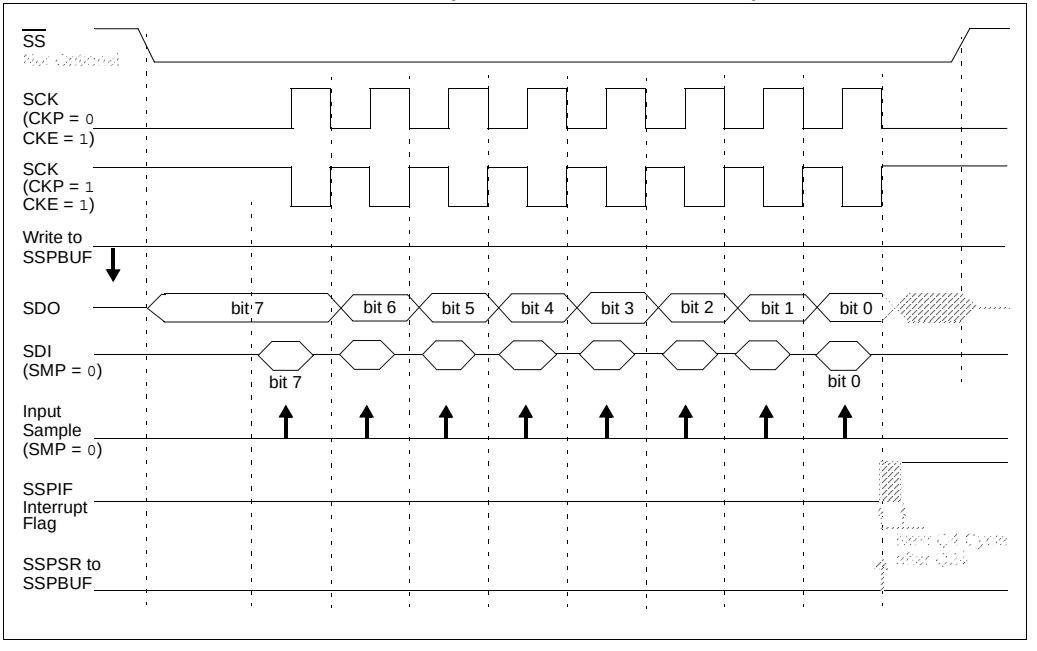


FIGURE 17-6: SPI MODE WAVEFORM (SLAVE MODE WITH CKE = 1)



17.4.7.1 Clock Arbitration

Clock arbitration occurs when the master, during any receive, transmit or Repeated Start/Stop condition, deasserts the SCL pin (SCL allowed to float high). When the SCL pin is allowed to float high, the Baud Rate Generator (BRG) is suspended from counting until the SCL pin is actually sampled high. When the SCL pin is sampled high, the Baud Rate Generator is reloaded with the contents of SSPADD<6:0> and begins counting. This ensures that the SCL high time will always be at least one BRG rollover count in the event that the clock is held low by an external device (Figure 17-18).

FIGURE 17-18: BAUD RATE GENERATOR TIMING WITH CLOCK ARBITRATION

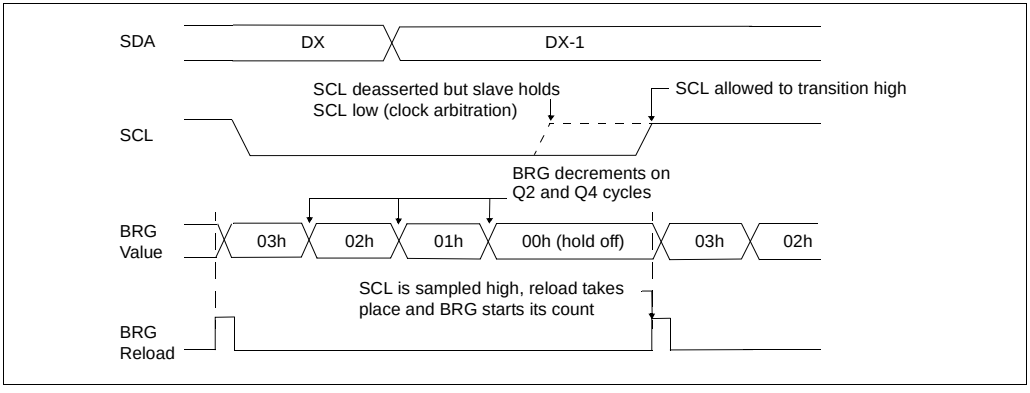


FIGURE 17-27: BUS COLLISION DURING START CONDITION (SCL = 0)

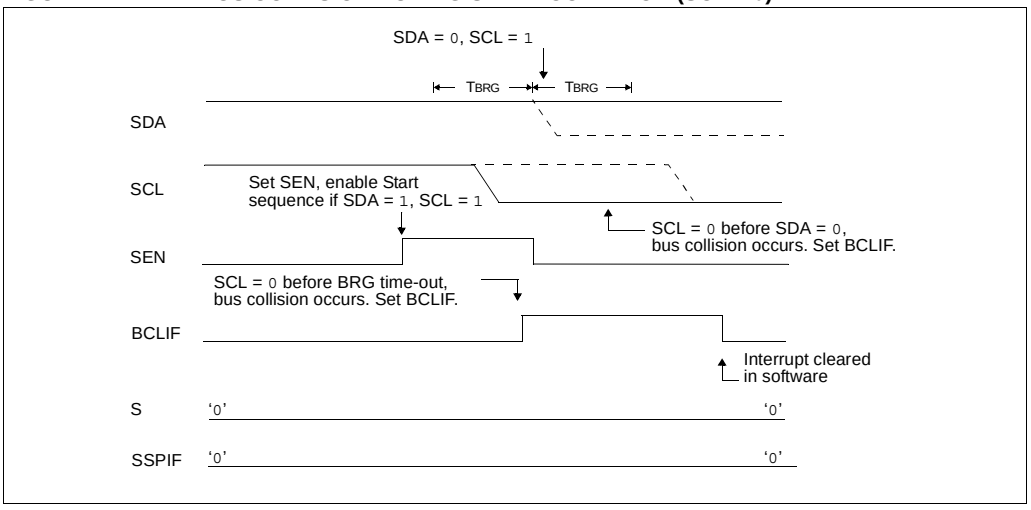
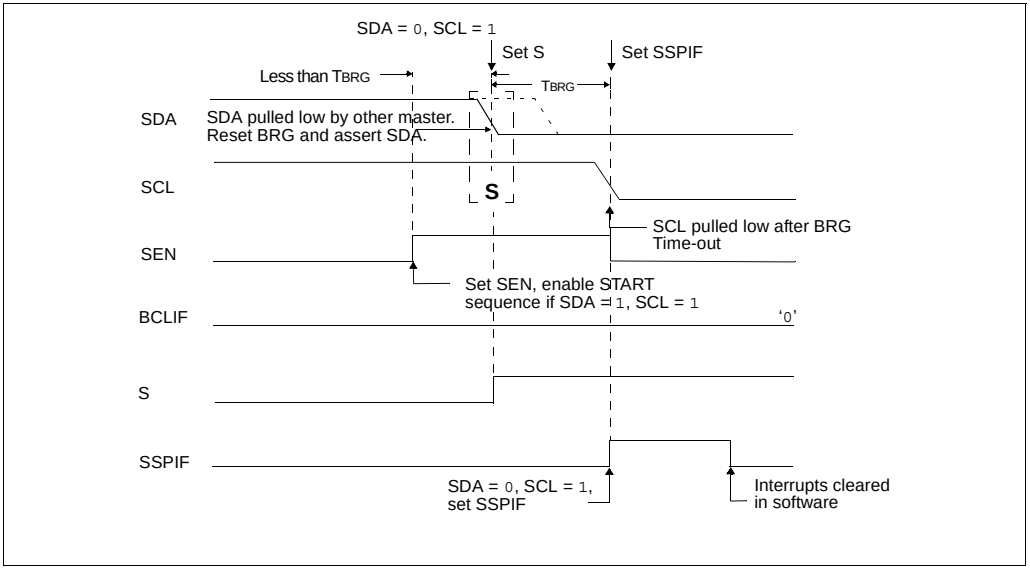


FIGURE 17-28: BRG RESET DUE TO SDA ARBITRATION DURING START CONDITION



17.4.17.2 Bus Collision During a Repeated Start Condition

During a Repeated Start condition, a bus collision occurs if:

a) A low level is sampled on SDA when SCL goes from low level to high level.

b) SCL goes low before SDA is asserted low, indicating that another master is attempting to transmit a data '1'.

When the user deasserts SDA and the pin is allowed to float high, the BRG is loaded with SSPADD<6:0> and counts down to '0'. The SCL pin is then deasserted and when sampled high, the SDA pin is sampled.

If SDA is low, a bus collision has occurred (i.e., another master is attempting to transmit a data '0', see Figure 17-29). If SDA is sampled high, the BRG is reloaded and begins counting. If SDA goes from high to low before the BRG times out, no bus collision occurs because no two masters can assert SDA at exactly the same time.

If SCL goes from high to low before the BRG times out and SDA has not already been asserted, a bus collision occurs. In this case, another master is attempting to transmit a data '1' during the Repeated Start condition (see Figure 17-30).

If, at the end of the BRG time-out, both SCL and SDA are still high, the SDA pin is driven low and the BRG is reloaded and begins counting. At the end of the count, regardless of the status of the SCL pin, the SCL pin is driven low and the Repeated Start condition is complete.

FIGURE 17-29: BUS COLLISION DURING A REPEATED START CONDITION (CASE 1)

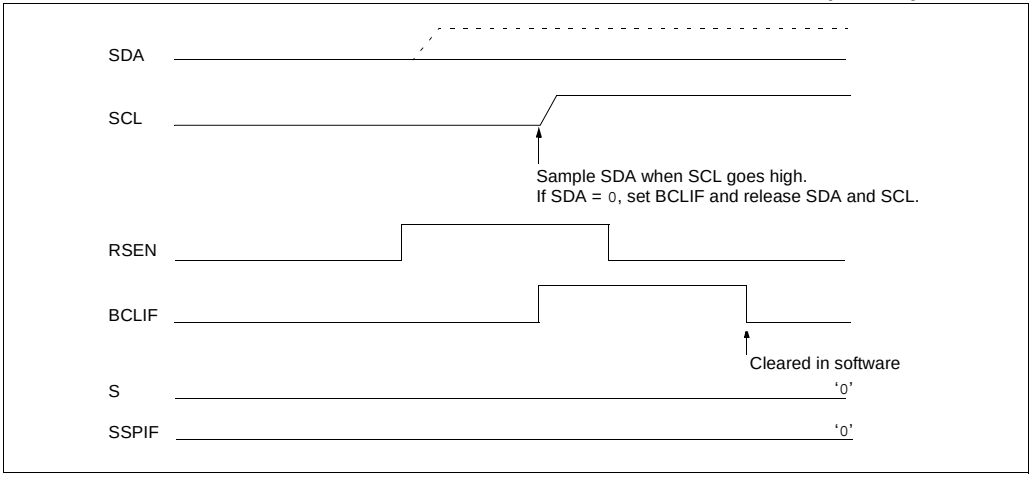
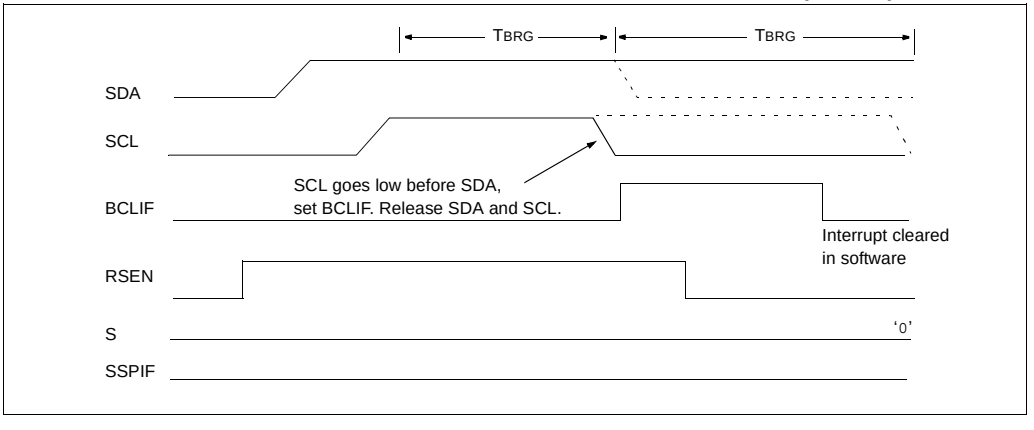


FIGURE 17-30: BUS COLLISION DURING REPEATED START CONDITION (CASE 2)



REGISTER 19-2: ADCON1 REGISTER

U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	VCFG1	VCFG0	PCFG3	PCFG2	PCFG1	PCFG0
bit 7							bit 0

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

bit 5-4 **VCFG1:VCFG0:** Voltage Reference Configuration bits

	A/D VREF+	**A/D VREF-**
00	AVDD	AVSS
01	External VREF+	AVSS
10	AVDD	External VREF-
11	External VREF+	External VREF-

bit 3-0 **PCFG3:PCFG0:** A/D Port Configuration Control bits

	AN15	**AN14**	**AN13**	**AN12**	**AN11**	**AN10**	**AN9**	**AN8**	**AN7**	**AN6**	**AN5**	**AN4**	**AN3**	**AN2**	**AN1**	**AN0**
0000	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A
0001	D	D	A	A	A	A	A	A	A	A	A	A	A	A	A	A
0010	D	D	D	A	A	A	A	A	A	A	A	A	A	A	A	A
0011	D	D	D	D	A	A	A	A	A	A	A	A	A	A	A	A
0100	D	D	D	D	D	A	A	A	A	A	A	A	A	A	A	A
0101	D	D	D	D	D	D	A	A	A	A	A	A	A	A	A	A
0110	D	D	D	D	D	D	D	A	A	A	A	A	A	A	A	A
0111	D	D	D	D	D	D	D	D	A	A	A	A	A	A	A	A
1000	D	D	D	D	D	D	D	D	D	A	A	A	A	A	A	A
1001	D	D	D	D	D	D	D	D	D	D	A	A	A	A	A	A
1010	D	D	D	D	D	D	D	D	D	D	D	A	A	A	A	A
1011	D	D	D	D	D	D	D	D	D	D	D	D	A	A	A	A
1100	D	D	D	D	D	D	D	D	D	D	D	D	D	A	A	A
1101	D	D	D	D	D	D	D	D	D	D	D	D	D	D	A	A
1110	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	A
1111	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D

A = Analog input D = Digital I/O

Shaded cells = Additional channels available on the PIC18F8X8X devices

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

Note: Channels AN15 through AN12 are not available on the 68-pin devices.

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.

23.7.3 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 8 buffers deep FIFO (First In First Out) 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 8 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 transmit buffer, the actual FIFO will consist of RXB0, RXB1, B0, B1 and B2, a total of 5 buffers. If B0 is configured as a transmit buffer, the FIFO length will be 2. If none of the programmable buffers are configured as a transmit buffer, the FIFO will be 8 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 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 FP<3:0> bits to access 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.

23.7.4 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 CCP1, which in turn captures the value of either Timer1 or Timer3. This value can be used as the message time-stamp.

To use the time-stamp capability, the CANCAP bit (CIOCAN<4>) must be set. This replaces the capture input for CCP1 with the signal generated from the CAN module. In addition, CCP1CON<3:0> must be set to '0011' to enable the CCP special event trigger for CAN events.

23.8 Message Acceptance Filters and Masks

The message acceptance filters and masks are used to determine if a message in the message assembly buffer should be loaded into any of the receive buffers. Once a valid message has been received into the MAB, the identifier fields of the message are compared to the filter values. If there is a match, that message will be loaded into the appropriate receive buffer. The filter masks are used to determine which bits in the identifier are examined with the filters. A truth table is shown below in Table 23-2 that indicates how each bit in the identifier is compared to the masks and filters to determine if a message should be loaded into a receive buffer. The mask essentially determines which bits to apply the acceptance filters to. If any mask bit is set to a zero, then that bit will automatically be accepted regardless of the filter bit.

TABLE 23-2: FILTER/MASK TRUTH TABLE

Mask bit n	Filter bit n	Message Identifier bit n001	Accept or Reject bit n
0	x	x	Accept
1	0	0	Accept
1	0	1	Reject
1	1	0	Reject
1	1	1	Accept

Legend: x = don't care

In Mode 0, acceptance filters RXF0 and RXF1 and filter mask RXM0 are associated with RXB0. Filters RXF2, RXF3, RXF4 and RXF5 and mask RXM1 are associated with RXB1.

24.4.2 DATA EEPROM CODE PROTECTION

The entire data EEPROM is protected from external reads and writes by two bits: CPD and WRTD. CPD inhibits external reads and writes of data EEPROM. WRTD inhibits external writes to data EEPROM. The CPU can continue to read and write data EEPROM regardless of the protection bit settings.

24.4.3 CONFIGURATION REGISTER PROTECTION

The Configuration registers can be write-protected. The WRTC bit controls protection of the Configuration registers. In User mode, the WRTC bit is readable only. WRTC can only be written via ICSP or an external programmer.

24.5 ID Locations

Eight memory locations (200000h-200007h) are designated as ID locations where the user can store checksum or other code identification numbers. These locations are accessible during normal execution through the TBLRD and TBLWT instructions or during program/verify. The ID locations can be read when the device is code-protected.

24.6 In-Circuit Serial Programming

PIC18FXX80/XX85 microcontrollers can be serially programmed while in the end application circuit. This is simply done with two lines for clock and data, and three other lines for power, ground and the programming voltage. This allows customers to manufacture boards with unprogrammed devices and then program the microcontroller just before shipping the product. This also allows the most recent firmware or a custom firmware to be programmed.

24.7 In-Circuit Debugger

When the DEBUG bit in Configuration register, CONFIG4L, is programmed to a '0', the in-circuit debugger functionality is enabled. This function allows simple debugging functions when used with MPLAB® IDE. When the microcontroller has this feature enabled, some of the resources are not available for general use. Table 24-4 shows which features are consumed by the background debugger.

TABLE 24-4: DEBUGGER RESOURCES

I/O pins	RB6, RB7
Stack	2 levels
Program Memory	512 bytes
Data Memory	10 bytes

To use the in-circuit debugger function of the microcontroller, the design must implement In-Circuit Serial Programming connections to $\overline{MCLR}$/VPP, VDD, GND, RB7 and RB6. This will interface to the in-circuit debugger module available from Microchip or one of the third party development tool companies.

MULLW **Multiply Literal with W**

Syntax: [*label*] MULLW k

Operands: $0 \le k \le 255$

Operation: (W) x k → PRODH:PRODL

Status Affected: None

Encoding:

0000	1101	kkkk	kkkk

Description: An unsigned multiplication is carried out between the contents of W and the 8-bit literal 'k'. The 16-bit result is placed in PRODH:PRODL register pair. PRODH contains the high byte. W is unchanged.
None of the status flags are affected.
Note that neither overflow nor carry is possible in this operation. A zero result is possible but not detected.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'k'	Process Data	Write registers PRODH: PRODL

Example: `MULLW 0xC4`

Before Instruction
W = 0xE2
PRODH = ?
PRODL = ?

After Instruction
W = 0xE2
PRODH = 0xAD
PRODL = 0x08

MULWF **Multiply W with f**

Syntax: [*label*] MULWF f [,a]

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

Operation: (W) x (f) → PRODH:PRODL

Status Affected: None

Encoding:

0000	001a	ffff	ffff

Description:' An unsigned multiplication is carried out between the contents of W and the register file location 'f'. The 16-bit result is stored in the PRODH:PRODL register pair. PRODH contains the high byte.
Both W and 'f' are unchanged.
None of the status flags are affected.
Note that neither overflow nor carry is possible in this operation. A zero result is possible but not detected. If 'a' is '0', the Access Bank will be selected, overriding the BSR value. If 'a' = 1, then the bank will be selected as per the BSR value (default).

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write registers PRODH: PRODL

Example: `MULWF REG, 1`

Before Instruction
W = 0xC4
REG = 0xB5
PRODH = ?
PRODL = ?

After Instruction
W = 0xC4
REG = 0xB5
PRODH = 0x8A
PRODL = 0x94

26.9 MPLAB ICE 2000 High-Performance Universal In-Circuit Emulator

The MPLAB ICE 2000 universal in-circuit emulator is intended to provide the product development engineer with a complete microcontroller design tool set for PIC microcontrollers. Software control of the MPLAB ICE 2000 in-circuit emulator is advanced by the MPLAB Integrated Development Environment, which allows editing, building, downloading and source debugging from a single environment.

The MPLAB ICE 2000 is a full-featured emulator system with enhanced trace, trigger and data monitoring features. Interchangeable processor modules allow the system to be easily reconfigured for emulation of different processors. The universal architecture of the MPLAB ICE in-circuit emulator allows expansion to support new PIC microcontrollers.

The MPLAB ICE 2000 in-circuit emulator system has been designed as a real-time emulation system with advanced features that are typically found on more expensive development tools. The PC platform and Microsoft® Windows 32-bit operating system were chosen to best make these features available in a simple, unified application.

26.10 MPLAB ICE 4000 High-Performance Universal In-Circuit Emulator

The MPLAB ICE 4000 universal in-circuit emulator is intended to provide the product development engineer with a complete microcontroller design tool set for high-end PIC microcontrollers. Software control of the MPLAB ICE in-circuit emulator is provided by the MPLAB Integrated Development Environment, which allows editing, building, downloading and source debugging from a single environment.

The MPLAB ICD 4000 is a premium emulator system, providing the features of MPLAB ICE 2000, but with increased emulation memory and high-speed performance for dsPIC30F and PIC18XXXX devices. Its advanced emulator features include complex triggering and timing, up to 2 Mb of emulation memory and the ability to view variables in real-time.

The MPLAB ICE 4000 in-circuit emulator system has been designed as a real-time emulation system with advanced features that are typically found on more expensive development tools. The PC platform and Microsoft Windows 32-bit operating system were chosen to best make these features available in a simple, unified application.

26.11 MPLAB ICD 2 In-Circuit Debugger

Microchip's In-Circuit Debugger, MPLAB ICD 2, is a powerful, low-cost, run-time development tool, connecting to the host PC via an RS-232 or high-speed USB interface. This tool is based on the Flash PIC MCUs and can be used to develop for these and other PIC microcontrollers. The MPLAB ICD 2 utilizes the in-circuit debugging capability built into the Flash devices. This feature, along with Microchip's In-Circuit Serial Programming™ (ICSP™) protocol, offers cost effective in-circuit Flash debugging from the graphical user interface of the MPLAB Integrated Development Environment. This enables a designer to develop and debug source code by setting breakpoints, single-stepping and watching variables, CPU status and peripheral registers. Running at full speed enables testing hardware and applications in real-time. MPLAB ICD 2 also serves as a development programmer for selected PIC devices.

26.12 PRO MATE II Universal Device Programmer

The PRO MATE II is a universal, CE compliant device programmer with programmable voltage verification at V_{DDMIN} and V_{DDMAX} for maximum reliability. It features an LCD display for instructions and error messages and a modular detachable socket assembly to support various package types. In Stand-Alone mode, the PRO MATE II device programmer can read, verify and program PIC devices without a PC connection. It can also set code protection in this mode.

26.13 MPLAB PM3 Device Programmer

The MPLAB PM3 is a universal, CE compliant device programmer with programmable voltage verification at V_{DDMIN} and V_{DDMAX} for maximum reliability. It features a large LCD display (128 x 64) for menus and error messages and a modular detachable socket assembly to support various package types. The ICSP™ cable assembly is included as a standard item. In Stand-Alone mode, the MPLAB PM3 device programmer can read, verify and program PIC devices without a PC connection. It can also set code protection in this mode. MPLAB PM3 connects to the host PC via an RS-232 or USB cable. MPLAB PM3 has high-speed communications and optimized algorithms for quick programming of large memory devices and incorporates an SD/MMC card for file storage and secure data applications.

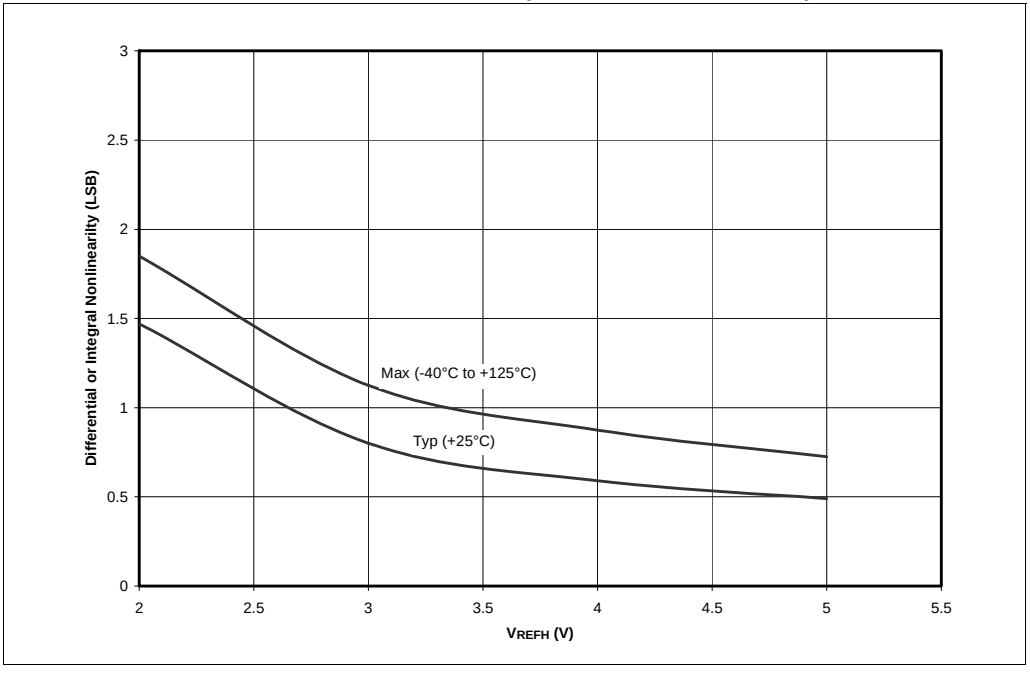
TABLE 27-20: I^2C BUS DATA REQUIREMENTS (SLAVE MODE)

Param. No.	Symbol	Characteristic		Min	Max	Units	Conditions
100	THIGH	Clock High Time	100 kHz mode	4.0	—	µs	PIC18FXX8X must operate at a minimum of 1.5 MHz
			400 kHz mode	0.6	—	µs	PIC18FXX8X must operate at a minimum of 10 MHz
			SSP Module	1.5 TCY	—		
101	TLOW	Clock Low Time	100 kHz mode	4.7	—	µs	PIC18FXX8X must operate at a minimum of 1.5 MHz
			400 kHz mode	1.3	—	µs	PIC18FXX8X must operate at a minimum of 10 MHz
			SSP Module	1.5 TCY	—		
102	TR	SDA and SCL Rise Time	100 kHz mode	—	1000	ns	
			400 kHz mode	20 + 0.1 CB	300	ns	CB is specified to be from 10 to 400 pF
103	TF	SDA and SCL Fall Time	100 kHz mode	—	300	ns	
			400 kHz mode	20 + 0.1 CB	300	ns	CB is specified to be from 10 to 400 pF
90	TSU:STA	Start Condition Setup Time	100 kHz mode	4.7	—	µs	Only relevant for Repeated Start condition
			400 kHz mode	0.6	—	µs	
91	THD:STA	Start Condition Hold Time	100 kHz mode	4.0	—	µs	After this period, the first clock pulse is generated
			400 kHz mode	0.6	—	µs	
106	THD:DAT	Data Input Hold Time	100 kHz mode	0	—	ns	
			400 kHz mode	0	0.9	µs	
107	TSU:DAT	Data Input Setup Time	100 kHz mode	250	—	ns	**(Note 2)**
			400 kHz mode	100	—	ns	
92	TSU:STO	Stop Condition Setup Time	100 kHz mode	4.7	—	µs	
			400 kHz mode	0.6	—	µs	
109	TAA	Output Valid from Clock	100 kHz mode	—	3500	ns	**(Note 1)**
			400 kHz mode	—	—	ns	
110	TBUF	Bus Free Time	100 kHz mode	4.7	—	µs	Time the bus must be free before a new transmission can Start
			400 kHz mode	1.3	—	µs	
D102	CB	Bus Capacitive Loading		—	400	pF	

Note 1: As a transmitter, the device must provide this internal minimum delay time to bridge the undefined region (min. 300 ns) of the falling edge of SCL to avoid unintended generation of Start or Stop conditions.

2: A Fast mode I^2C bus device can be used in a Standard mode I^2C bus system but the requirement, TSU:DAT ≥ 250 ns, must then be met. This will automatically be the case if the device does not stretch the low period of the SCL signal. If such a device does stretch the low period of the SCL signal, it must output the next data bit to the SDA line.
TR max. + TSU:DAT = 1000 + 250 = 1250 ns (according to the Standard mode I^2C bus specification) before the SCL line is released.

FIGURE 28-29: A/D NON-LINEARITY vs. VREFH (VDD = 5V, -40°C TO +125°C)



S

T