Welcome to E-XFL.COM

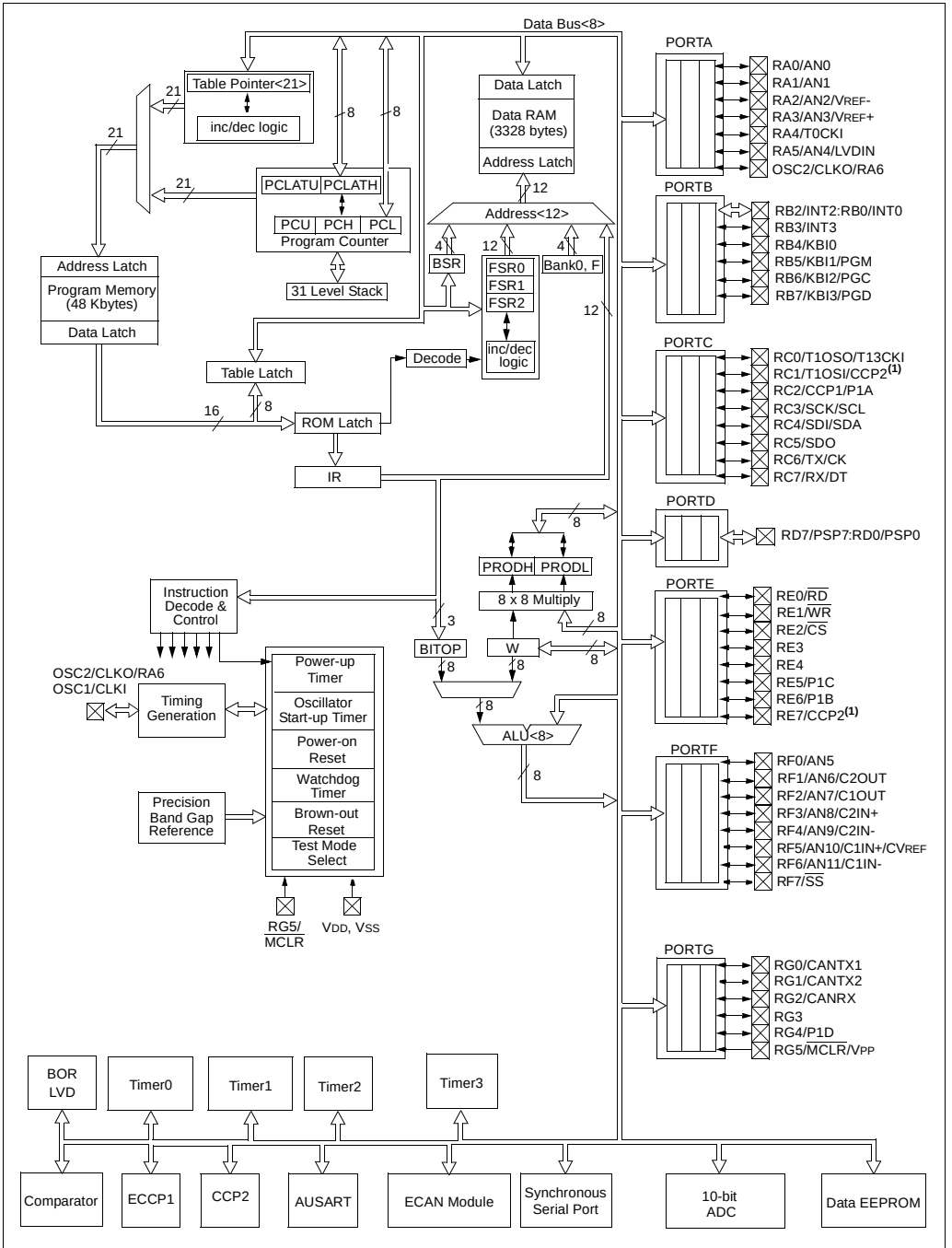
What is "Embedded - Microcontrollers"?

"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)	4.25V ~ 5.5V
Data Converters	A/D 12x10b
Oscillator Type	External
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	68-LCC (J-Lead)
Supplier Device Package	68-PLCC (24.23x24.23)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic18f6585t-i-l

FIGURE 1-1: PIC18F6X8X BLOCK DIAGRAM



Note 1: The CCP2 pin placement depends on the CCP2MX and Processor mode settings.

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 4-2: SPECIAL FUNCTION REGISTER MAP (CONTINUED)

Address	Name	Address	Name	Address	Name	Address	Name
EFFh	—(1)	EDFh	—(1)	EBFh	—(1)	E9Fh	—(1)
EFEh	—(1)	EDEh	—(1)	EBEh	—(1)	E9Eh	—(1)
EFDh	—(1)	EDDh	—(1)	EBDh	—(1)	E9Dh	—(1)
EFCh	—(1)	EDCh	—(1)	EBCh	—(1)	E9Ch	—(1)
EFBh	—(1)	EDBh	—(1)	EBBh	—(1)	E9Bh	—(1)
EFAh	—(1)	EDAh	—(1)	EBAh	—(1)	E9Ah	—(1)
EF9h	—(1)	ED9h	—(1)	EB9h	—(1)	E99h	—(1)
EF8h	—(1)	ED8h	—(1)	EB8h	—(1)	E98h	—(1)
EF7h	—(1)	ED7h	—(1)	EB7h	—(1)	E97h	—(1)
EF6h	—(1)	ED6h	—(1)	EB6h	—(1)	E96h	—(1)
EF5h	—(1)	ED5h	—(1)	EB5h	—(1)	E95h	—(1)
EF4h	—(1)	ED4h	—(1)	EB4h	—(1)	E94h	—(1)
EF3h	—(1)	ED3h	—(1)	EB3h	—(1)	E93h	—(1)
EF2h	—(1)	ED2h	—(1)	EB2h	—(1)	E92h	—(1)
EF1h	—(1)	ED1h	—(1)	EB1h	—(1)	E91h	—(1)
EF0h	—(1)	ED0h	—(1)	EB0h	—(1)	E90h	—(1)
EEFh	—(1)	ECFh	—(1)	EAFh	—(1)	E8Fh	—(1)
EEEh	—(1)	ECEh	—(1)	EAEh	—(1)	E8Eh	—(1)
EEDh	—(1)	ECDh	—(1)	EADh	—(1)	E8Dh	—(1)
EECh	—(1)	ECCh	—(1)	EACh	—(1)	E8Ch	—(1)
EEBh	—(1)	ECBh	—(1)	EABh	—(1)	E8Bh	—(1)
EEAh	—(1)	ECAh	—(1)	EAAh	—(1)	E8Ah	—(1)
EE9h	—(1)	EC9h	—(1)	EA9h	—(1)	E89h	—(1)
EE8h	—(1)	EC8h	—(1)	EA8h	—(1)	E88h	—(1)
EE7h	—(1)	EC7h	—(1)	EA7h	—(1)	E87h	—(1)
EE6h	—(1)	EC6h	—(1)	EA6h	—(1)	E86h	—(1)
EE5h	—(1)	EC5h	—(1)	EA5h	—(1)	E85h	—(1)
EE4h	—(1)	EC4h	—(1)	EA4h	—(1)	E84h	—(1)
EE3h	—(1)	EC3h	—(1)	EA3h	—(1)	E83h	—(1)
EE2h	—(1)	EC2h	—(1)	EA2h	—(1)	E82h	—(1)
EE1h	—(1)	EC1h	—(1)	EA1h	—(1)	E81h	—(1)
EE0h	—(1)	EC0h	—(1)	EA0h	—(1)	E80h	—(1)

Note 1: Unimplemented registers are read as '0'.

2: This register is not available on PIC18F6X8X devices.

3: This is not a physical register.

15.4 PWM Mode

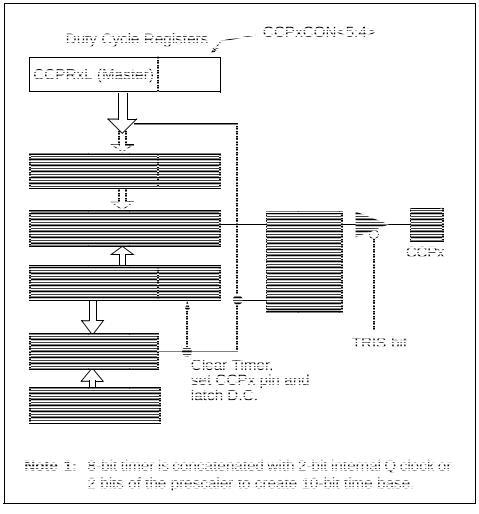
In Pulse Width Modulation (PWM) mode, the CCPx pin produces up to a 10-bit resolution PWM output. For PWM mode to function properly, the TRIS bit for the CCPx pin must be cleared to make it an output.

Note:	Clearing the CCPxCON register will force the CCPx PWM output latch to the default low level. This is not the port data latch.

Figure 15-3 shows a simplified block diagram of the CCP module in PWM mode.

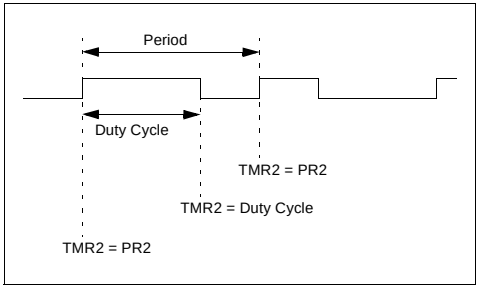
For a step-by-step procedure on how to set up the CCP module for PWM operation, see **Section 15.4.3 "Setup for PWM Operation"**.

FIGURE 15-3: SIMPLIFIED PWM BLOCK DIAGRAM



A PWM output (Figure 15-4) has a time base (period) and a time that the output stays high (duty cycle). The frequency of the PWM is the inverse of the period (1/period).

FIGURE 15-4: PWM OUTPUT



15.4.1 PWM PERIOD

The PWM period is specified by writing to the PR2 register. The PWM period can be calculated using the following formula.

EQUATION 15-1:

$$\text{PWM Period} = [(PR2) + 1] \cdot 4 \cdot T_{OSC} \cdot (\text{TMR2 Prescale Value})$$

PWM frequency is defined as 1/[PWM period].

When TMR2 is equal to PR2, the following three events occur on the next increment cycle:

- TMR2 is cleared
- The CCP1 pin is set (exception: if PWM duty cycle = 0%, the CCP1 pin will not be set)
- The PWM duty cycle is latched from CCPR1L into CCPR1H

Note:	The Timer2 postscaler (see **Section 13.0 "Timer2 Module"**) is not used in the determination of the PWM frequency. The postscaler could be used to have a servo update rate at a different frequency than the PWM output.

15.4.2 PWM DUTY CYCLE

The PWM duty cycle is specified by writing to the CCPRxL register and to the CCPxCON<5:4> bits. Up to 10-bit resolution is available. The CCPRxL contains the eight MSbs and the CCPxCON<5:4> contain the two LSbs. This 10-bit value is represented by CCPRxL:CCPxCON<5:4>. The following equation is used to calculate the PWM duty cycle in time.

EQUATION 15-2:

$$\text{PWM Duty Cycle} = (\text{CCPRxL:CCPxCON<5:4>}) \cdot T_{OSC} \cdot (\text{TMR2 Prescale Value})$$

CCPRxL and CCPxCON<5:4> can be written to at any time but the duty cycle value is not latched into CCPRxH until after a match between PR2 and TMR2 occurs (i.e., the period is complete). In PWM mode, CCPRxH is a read-only register.

The CCPRxH register and a 2-bit internal latch are used to double-buffer the PWM duty cycle. This double-buffering is essential for glitchless PWM operation.

When the CCPRxH and 2-bit latch match TMR2, concatenated with an internal 2-bit Q clock or 2 bits of the TMR2 prescaler, the CCPx pin is cleared.

The maximum PWM resolution (bits) for a given PWM frequency is given by the following equation.

EQUATION 15-3:

$$\text{PWM Resolution (max)} = \frac{\log\left(\frac{FOSC}{FPWM}\right)}{\log(2)}\text{bits}$$

Note:	If the PWM duty cycle value is longer than the PWM period, the CCP1 pin will not be cleared.

15.4.3 SETUP FOR PWM OPERATION

The following steps should be taken when configuring the CCP module for PWM operation:

1. Set the PWM period by writing to the PR2 register.
2. Set the PWM duty cycle by writing to the CCPRxL register and CCPxCON<5:4> bits.
3. Make the CCPx pin an output by clearing corresponding TRIS bit.
4. Set the TMR2 prescale value and enable Timer2 by writing to T2CON.
5. Configure the CCPx module for PWM operation.

TABLE 15-4: EXAMPLE PWM FREQUENCIES AND RESOLUTIONS AT 40 MHz

PWM Frequency	2.44 kHz	9.76 kHz	39.06 kHz	156.3 kHz	312.5 kHz	416.6 kHz
Timer Prescaler (1, 4, 16)	16	4	1	1	1	1
PR2 Value	0FFh	0FFh	0FFh	3Fh	1Fh	17h
Maximum Resolution (bits)	10	10	10	8	7	5.5

TABLE 15-5: REGISTERS ASSOCIATED WITH PWM AND TIMER2

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR, BOR	Value on all other Resets
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	0000 000x	0000 000u
PIR1	PSPIF	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	0000 0000	0000 0000
PIE1	PSPIE	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	0000 0000	0000 0000
IPR1	PSPIP	ADIP	RCIP	TXIP	SSPIP	CCP1IP	TMR2IP	TMR1IP	1111 1111	1111 1111
TRISC	PORTC Data Direction Register								1111 1111	1111 1111
TMR2	Timer2 Module Register								0000 0000	0000 0000
PR2	Timer2 Module Period Register								1111 1111	1111 1111
T2CON	—	TOUTPS3	TOUTPS2	TOUTPS1	TOUTPS0	TMR2ON	T2CKPS1	T2CKPS0	-000 0000	-000 0000
CCPR1L	Capture/Compare/PWM Register 1 (LSB)								xxxx xxxx	uuuu uuuu
CCPR1H	Capture/Compare/PWM Register 1 (MSB)								xxxx xxxx	uuuu uuuu
CCP1CON	P1M1	P1M0	DC1B1	DC1B0	CCP1M3	CCP1M2	CCP1M1	CCP1M0	0000 0000	0000 0000
CCPR2L	Capture/Compare/PWM Register 2 (LSB)								xxxx xxxx	uuuu uuuu
CCPR2H	Capture/Compare/PWM Register 2 (MSB)								xxxx xxxx	uuuu uuuu
CCP2CON	—	—	DC2B1	DC2B0	CCP2M3	CCP2M2	CCP2M1	CCP2M0	--00 0000	--00 0000

Legend: x = unknown, u = unchanged, – = unimplemented, read as '0'. Shaded cells are not used by PWM and Timer2.

17.4.4 CLOCK STRETCHING

Both 7- and 10-bit Slave modes implement automatic clock stretching during a transmit sequence.

The SEN bit (SSPCON2<0>) allows clock stretching to be enabled during receives. Setting SEN will cause the SCL pin to be held low at the end of each data receive sequence.

17.4.4.1 Clock Stretching for 7-bit Slave Receive Mode (SEN = 1)

In 7-bit Slave Receive mode, on the falling edge of the ninth clock at the end of the ACK sequence if the BF bit is set, the CKP bit in the SSPCON1 register is automatically cleared, forcing the SCL output to be held low. The CKP being cleared to '0' will assert the SCL line low. The CKP bit must be set in the user's ISR before reception is allowed to continue. By holding the SCL line low, the user has time to service the ISR and read the contents of the SSPBUF before the master device can initiate another receive sequence. This will prevent buffer overruns from occurring (see Figure 17-13).

Note 1:	If the user reads the contents of the SSPBUF before the falling edge of the ninth clock, thus clearing the BF bit, the CKP bit will not be cleared and clock stretching will not occur.
2:	The CKP bit can be set in software regardless of the state of the BF bit. The user should be careful to clear the BF bit in the ISR before the next receive sequence in order to prevent an overflow condition.

17.4.4.2 Clock Stretching for 10-bit Slave Receive Mode (SEN = 1)

In 10-bit Slave Receive mode, during the address sequence, clock stretching automatically takes place but CKP is not cleared. During this time, if the UA bit is set after the ninth clock, clock stretching is initiated. The UA bit is set after receiving the upper byte of the 10-bit address and following the receive of the second byte of the 10-bit address with the R/$\overline{W}$ bit cleared to '0'. The release of the clock line occurs upon updating SSPADD. Clock stretching will occur on each data receive sequence as described in 7-bit mode.

Note:	If the user polls the UA bit and clears it by updating the SSPADD register before the falling edge of the ninth clock occurs and if the user hasn't cleared the BF bit by reading the SSPBUF register before that time, then the CKP bit will still NOT be asserted low. Clock stretching on the basis of the state of the BF bit only occurs during a data sequence, not an address sequence.

17.4.4.3 Clock Stretching for 7-bit Slave Transmit Mode

7-bit Slave Transmit mode implements clock stretching by clearing the CKP bit after the falling edge of the ninth clock, if the BF bit is clear. This occurs regardless of the state of the SEN bit.

The user's ISR must set the CKP bit before transmission is allowed to continue. By holding the SCL line low, the user has time to service the ISR and load the contents of the SSPBUF before the master device can initiate another transmit sequence (see Figure 17-9).

Note 1:	If the user loads the contents of SSPBUF, setting the BF bit before the falling edge of the ninth clock, the CKP bit will not be cleared and clock stretching will not occur.
2:	The CKP bit can be set in software regardless of the state of the BF bit.

17.4.4.4 Clock Stretching for 10-bit Slave Transmit Mode

In 10-bit Slave Transmit mode, clock stretching is controlled during the first two address sequences by the state of the UA bit, just as it is in 10-bit Slave Receive mode. The first two addresses are followed by a third address sequence which contains the high order bits of the 10-bit address and the R/$\overline{W}$ bit set to '1'. After the third address sequence is performed, the UA bit is not set, the module is now configured in Transmit mode, and clock stretching is controlled by the BF flag as in 7-bit Slave Transmit mode (see Figure 17-11).

17.4.6.1 I²C Master Mode Operation

The master device generates all of the serial clock pulses and the Start and Stop conditions. A transfer is ended with a Stop condition or with a Repeated Start condition. Since the Repeated Start condition is also the beginning of the next serial transfer, the I²C bus will not be released.

In Master Transmitter mode, serial data is output through SDA while SCL outputs the serial clock. The first byte transmitted contains the slave address of the receiving device (7 bits) and the Read/Write (R/$\overline{W}$) bit. In this case, the R/$\overline{W}$ bit will be logic '0'. Serial data is transmitted 8 bits at a time. After each byte is transmitted, an Acknowledge bit is received. Start and Stop conditions are output to indicate the beginning and the end of a serial transfer.

In Master Receive mode, the first byte transmitted contains the slave address of the transmitting device (7 bits) and the R/$\overline{W}$ bit. In this case, the R/$\overline{W}$ bit will be logic '1'. Thus, the first byte transmitted is a 7-bit slave address followed by a '1' to indicate a receive bit. Serial data is received via SDA while SCL outputs the serial clock. Serial data is received 8 bits at a time. After each byte is received, an Acknowledge bit is transmitted. Start and Stop conditions indicate the beginning and end of transmission.

The Baud Rate Generator used for the SPI mode operation is used to set the SCL clock frequency for either 100 kHz, 400 kHz or 1 MHz I²C operation. See **Section 17.4.7 "Baud Rate Generator"** for more detail.

A typical transmit sequence would go as follows:

1. The user generates a Start condition by setting the Start enable bit, SEN (SSPCON2<0>).
2. SSPIF is set. The MSSP module will wait the required start time before any other operation takes place.
3. The user loads the SSPBUF with the slave address to transmit.
4. Address is shifted out the SDA pin until all 8 bits are transmitted.
5. The MSSP module shifts in the ACK bit from the slave device and writes its value into the SSPCON2 register (SSPCON2<6>).
6. The MSSP module generates an interrupt at the end of the ninth clock cycle by setting the SSPIF bit.
7. The user loads the SSPBUF with eight bits of data.
8. Data is shifted out the SDA pin until all 8 bits are transmitted.
9. The MSSP module shifts in the ACK bit from the slave device and writes its value into the SSPCON2 register (SSPCON2<6>).
10. The MSSP module generates an interrupt at the end of the ninth clock cycle by setting the SSPIF bit.
11. The user generates a Stop condition by setting the Stop enable bit PEN (SSPCON2<2>).
12. Interrupt is generated once the Stop condition is complete.

17.4.17.1 Bus Collision During a Start Condition

During a Start condition, a bus collision occurs if:

a) SDA or SCL are sampled low at the beginning of the Start condition (Figure 17-26).

b) SCL is sampled low before SDA is asserted low (Figure 17-27).

During a Start condition, both the SDA and the SCL pins are monitored.

If the SDA pin is already low or the SCL pin is already low, then all of the following occur:

- the Start condition is aborted,
- the BCLIF flag is set, and
- the MSSP module is reset to its Idle state (Figure 17-26).

The Start condition begins with the SDA and SCL pins deasserted. When the SDA pin is sampled high, the Baud Rate Generator is loaded from SSPADD<6:0> and counts down to '0'. If the SCL pin is sampled low while SDA is high, a bus collision occurs because it is assumed that another master is attempting to drive a data '1' during the Start condition.

If the SDA pin is sampled low during this count, the BRG is reset and the SDA line is asserted early (Figure 17-28). If, however, a '1' is sampled on the SDA pin, the SDA pin is asserted low at the end of the BRG count. The Baud Rate Generator is then reloaded and counts down to '0' and during this time, if the SCL pins are sampled as '0', a bus collision does not occur. At the end of the BRG count, the SCL pin is asserted low.

Note:	The reason that bus collision is not a factor during a Start condition is that no two bus masters can assert a Start condition at the exact same time. Therefore, one master will always assert SDA before the other. This condition does not cause a bus collision because the two masters must be allowed to arbitrate the first address following the Start condition. If the address is the same, arbitration must be allowed to continue into the data portion, Repeated Start or Stop conditions.

FIGURE 17-26: BUS COLLISION DURING START CONDITION (SDA ONLY)

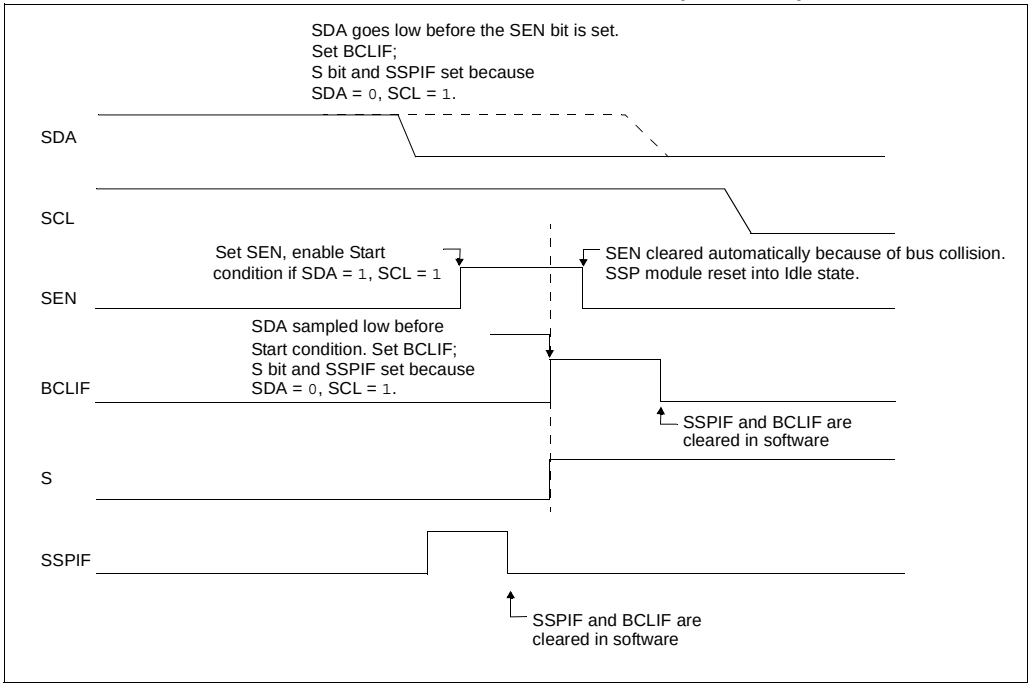


FIGURE 18-2: USART TRANSMIT BLOCK DIAGRAM

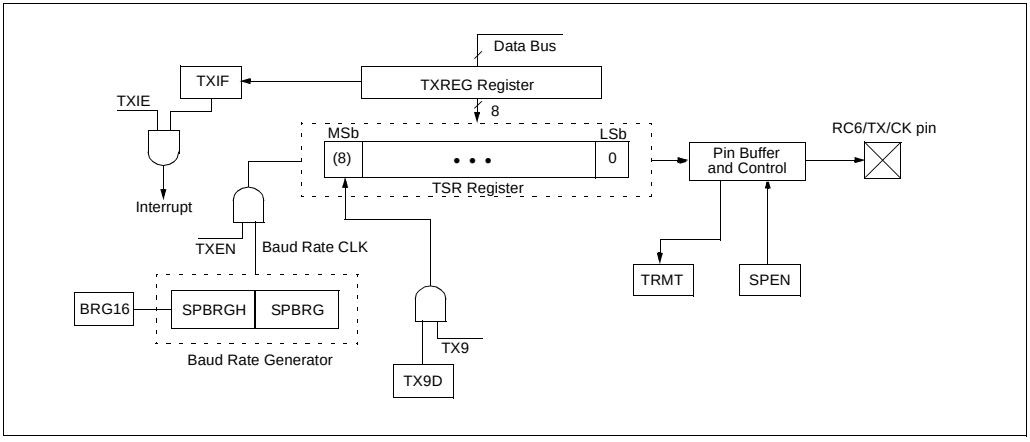


FIGURE 18-3: ASYNCHRONOUS TRANSMISSION

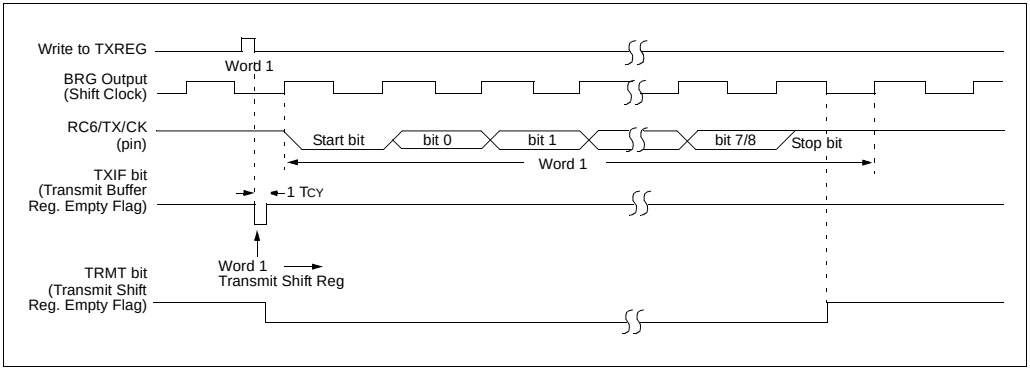
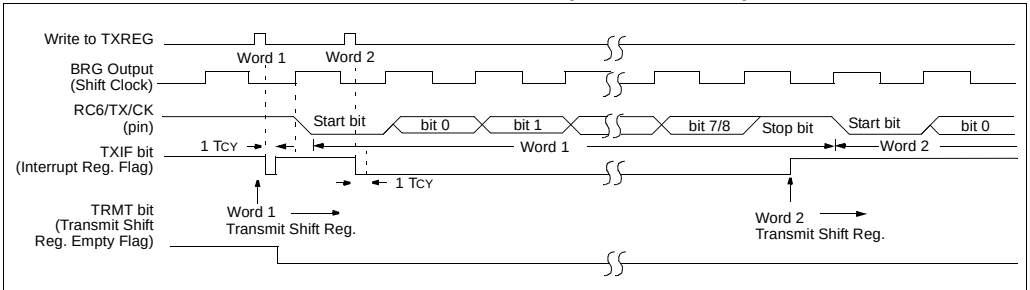


FIGURE 18-4: ASYNCHRONOUS TRANSMISSION (BACK TO BACK)



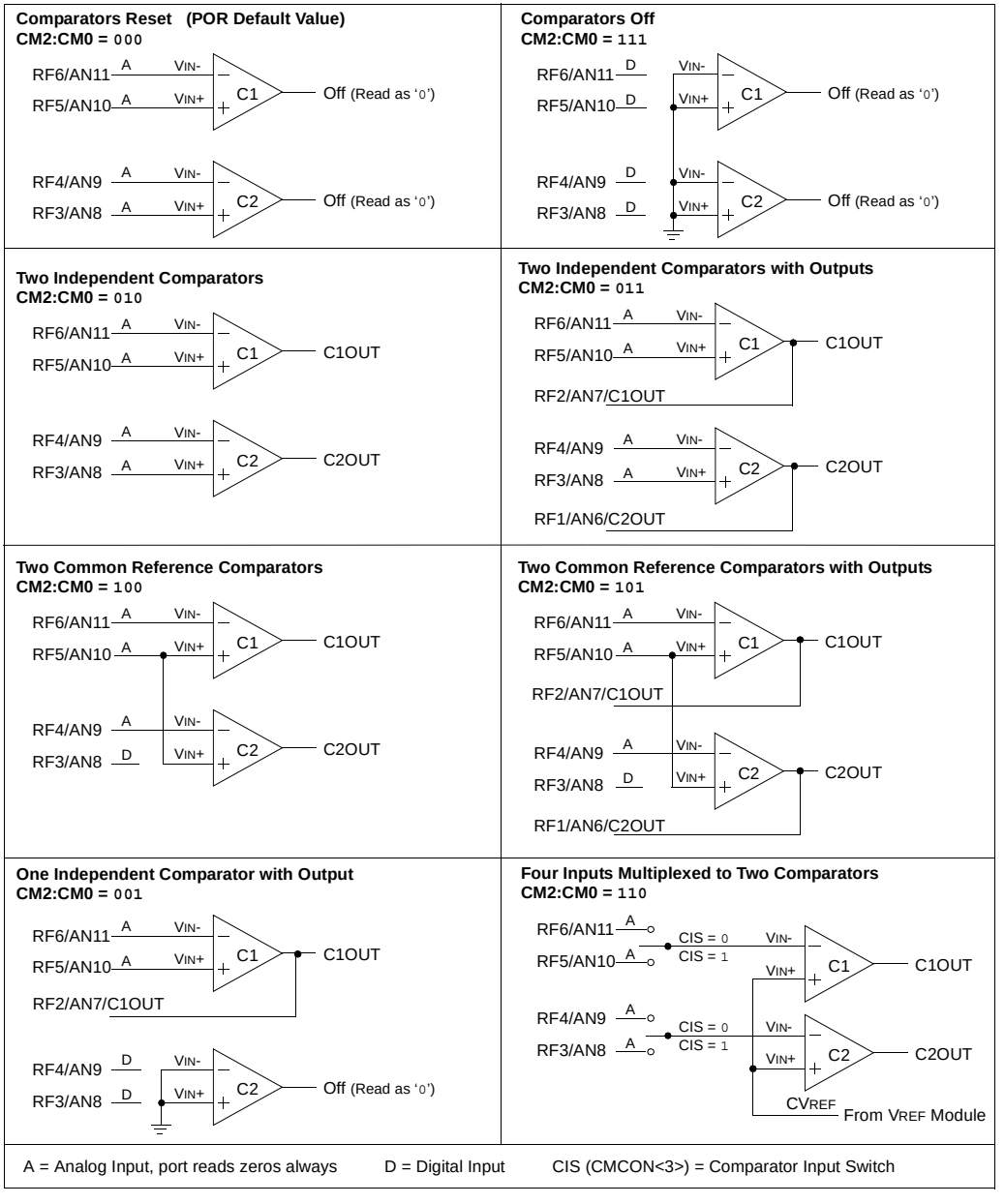
Note: This timing diagram shows two consecutive transmissions.

20.1 Comparator Configuration

There are eight modes of operation for the comparators. The CMCON register is used to select these modes. Figure 20-1 shows the eight possible modes. The TRISF register controls the data direction of the comparator pins for each mode. If the Comparator mode is changed, the comparator output level may not be valid for the specified mode change delay shown in **Section 27.0 "Electrical Characteristics"**.

Note:	Comparator interrupts should be disabled during a Comparator mode change. Otherwise, a false interrupt may occur.

FIGURE 20-1: COMPARATOR I/O OPERATING MODES



21.0 COMPARATOR VOLTAGE REFERENCE MODULE

The comparator voltage reference is a 16-tap resistor ladder network that provides a selectable voltage reference. The resistor ladder is segmented to provide two ranges of CVREF values and has a power-down function to conserve power when the reference is not being used. The CVRCON register controls the operation of the reference as shown in Register 21-1. The block diagram is given in Figure 21-1.

The comparator reference supply voltage can come from either VDD or VSS, or the external VREF+ and VREF- that are multiplexed with RA3 and RA2. The comparator reference supply voltage is controlled by the CVRSS bit.

21.1 Configuring the Comparator Voltage Reference

The comparator voltage reference can output 16 distinct voltage levels for each range. The equations used to calculate the output of the comparator voltage reference are as follows:

If CVRR = 1:
CVREF = (CVR<3:0>/24) x CVRSRC

If CVRR = 0:
CVREF = (CVDD x 1/4) + (CVR<3:0>/32) x CVRSRC

The settling time of the comparator voltage reference must be considered when changing the CVREF output (**Section 27.0 "Electrical Characteristics"**).

REGISTER 21-1: CVRCON REGISTER

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
CVREN	CVROE	CVRR	CVRSS	CVR3	CVR2	CVR1	CVR0
bit 7							bit 0

bit 7 **CVREN**: Comparator Voltage Reference Enable bit
1 = CVREF circuit powered on
0 = CVREF circuit powered down

bit 6 **CVROE**: Comparator VREF Output Enable bit(1)
1 = CVREF voltage level is also output on the RF5/AN10/C1IN+/CVREF pin
0 = CVREF voltage is disconnected from the RF5/AN10/C1IN+/CVREF pin

bit 5 **CVRR**: Comparator VREF Range Selection bit
1 = 0.00 CVRSRC to 0.625 CVRSRC with CVRSRC/24 step size
0 = 0.25 CVRSRC to 0.71875 CVRSRC with CVRSRC/32 step size

bit 4 **CVRSS**: Comparator VREF Source Selection bit
1 = Comparator reference source, CVRSRC = VREF+ – VREF-
0 = Comparator reference source, CVRSRC = VDD – VSS

Note: To select (VREF+ – VREF-) as the comparator voltage reference source, the voltage reference configuration bits in the ADCON1 register (ADCON1<5:4>) must also be set to '11'.

bit 3-0 **CVR3:CVR0:** Comparator VREF Value Selection bits (0 ≤ VR3:VR0 ≤ 15)
When CVRR = 1:
CVREF = (CVR<3:0>/24) • (CVRSRC)
When CVRR = 0:
CVREF = 1/4 • (CVRSRC) + (CVR3:CVR0/32) • (CVRSRC)

Note 1: If enabled for output, RF5 must also be configured as an input by setting TRISF<5> to '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

REGISTER 23-2: CANSTAT: CAN STATUS REGISTER (CONTINUED)

bit 4-0 Mode 1,2:

EICODE4:EICODE0: Interrupt Code bits in Mode 1 and Mode 2
When an interrupt occurs, a prioritized coded interrupt value will be present in these bits. This code indicates the source of the interrupt. Unlike ICODE bits in Mode 0, these bits may not be copied directly to EWIN bits to map interrupted buffer to Access Bank area. If required, user software may maintain a table in program memory to map EICODE bits to EWIN bits and access interrupt buffer in Access Bank area.

	EICODE4:EICODE0 Value
No interrupt	00000
Error interrupt	00010
TXB2 interrupt	00100
TXB1 interrupt	00110
TXB0 interrupt	01000
RXB1 interrupt	10001/10000(2)
RXB0 interrupt	10000
Wake-up interrupt	01110
RX/TX B0 interrupt	10010(2)
RX/TX B1 interrupt	10011(2)
RX/TX B2 interrupt	10100(2)
RX/TX B3 interrupt	10101(2)
RX/TX B4 interrupt	10110(2)
RX/TX B4 interrupt	10111(2)

Note 1: To achieve maximum power saving and/or able to wake-up on CAN bus activity, switch CAN module to Disable mode before putting the device to Sleep.

2: In Mode 2, if the buffer is configured as a receiver, EICODE bits will always contain '10000' upon interrupt.

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

REGISTER 23-6: TXBnSIDH: TRANSMIT BUFFER n STANDARD IDENTIFIER REGISTERS, HIGH BYTE [0 ≤ n ≤ 2]

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
SID10	SID9	SID8	SID7	SID6	SID5	SID4	SID3
bit 7							bit 0

bit 7-0 **SID10:SID3:** Standard Identifier bits, if EXIDE (TXBnSIDL<3>) = 0;
Extended Identifier bits EID28:EID21, if EXIDE = 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

REGISTER 23-7: TXBnSIDL: TRANSMIT BUFFER n STANDARD IDENTIFIER REGISTERS, LOW BYTE [0 ≤ n ≤ 2]

R/W-x	R/W-x	R/W-x	U-0	R/W-x	U-0	R/W-x	R/W-x
SID2	SID1	SID0	—	EXIDE	—	EID17	EID16
bit 7							bit 0

bit 7-5 **SID2:SID0:** Standard Identifier bits, if EXIDE (TXBnSIDL<3>) = 0;
Extended Identifier bits EID20:EID18, if EXIDE = 1.

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

bit 3 **EXIDE:** Extended Identifier Enable bit
1 = Message will transmit extended ID, SID10:SID0 becomes EID28:EID18
0 = Message will transmit standard ID, EID17:EID0 are ignored

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

bit 1-0 **EID17:EID16:** Extended Identifier bits

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

REGISTER 23-8: TXBnEIDH: TRANSMIT BUFFER n EXTENDED IDENTIFIER REGISTERS, HIGH BYTE [0 ≤ n ≤ 2]

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
EID15	EID14	EID13	EID12	EID11	EID10	EID9	EID8
bit 7							bit 0

bit 7-0 **EID15:EID8:** Extended Identifier bits (not used when transmitting standard identifier message)

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

REGISTER 23-24: BnSIDH: TX/RX BUFFER n STANDARD IDENTIFIER REGISTERS, HIGH BYTE IN RECEIVE MODE [0 ≤ n ≤ 5, TXnEN (BSEL0<n>) = 0](1)

R-x	R-x	R-x	R-x	R-x	R-x	R-x	R-x
SID10	SID9	SID8	SID7	SID6	SID5	SID4	SID3
bit 7							bit 0

bit 7-0 **SID10:SID3:** Standard Identifier bits, if EXIDE (BnSIDL<3>) = 0;
Extended Identifier bits EID28:EID21, if EXIDE = 1.

Note 1: These registers are available in Mode 1 and 2 only.

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

REGISTER 23-25: BnSIDH: TX/RX BUFFER n STANDARD IDENTIFIER REGISTERS, HIGH BYTE IN TRANSMIT MODE [0 ≤ n ≤ 5, TXnEN (BSEL0<n>) = 1](1)

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
SID10	SID9	SID8	SID7	SID6	SID5	SID4	SID3
bit 7							bit 0

bit 7-0 **SID10:SID3:** Standard Identifier bits, if EXIDE (BnSIDL<3>) = 0;
Extended Identifier bits EID28:EID21, if EXIDE = 1.

Note 1: These registers are available in Mode 1 and 2 only.

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

REGISTER 23-30: BnEIDL: TX/RX BUFFER n EXTENDED IDENTIFIER REGISTERS, LOW BYTE IN RECEIVE MODE [0 ≤ n ≤ 5, TXnEN (BSEL<n>) = 0](1)

R-x	R-x	R-x	R-x	R-x	R-x	R-x	R-x
EID7	EID6	EID5	EID4	EID3	EID2	EID1	EID0
bit 7							bit 0

bit 7-0 **EID7:EID0:** Extended Identifier bits

Note 1: These registers are available in Mode 1 and 2 only.

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

REGISTER 23-31: BnEIDL: TX/RX BUFFER n EXTENDED IDENTIFIER REGISTERS, LOW BYTE IN TRANSMIT MODE [0 ≤ n ≤ 5, TXnEN (BSEL<n>) = 1](1)

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
EID7	EID6	EID5	EID4	EID3	EID2	EID1	EID0
bit 7							bit 0

bit 7-0 **EID7:EID0:** Extended Identifier bits

Note 1: These registers are available in Mode 1 and 2 only.

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

NOTES:

REGISTER 24-1: CONFIG1H: CONFIGURATION REGISTER 1 HIGH (BYTE ADDRESS 300001h)

U-0	U-0	R/P-1	U-0	R/P-1	R/P-1	R/P-1	R/P-1
—	—	$\overline{\text{OSCSEN}}$	—	FOSC3	FOSC2	FOSC1	FOSC0
bit 7							bit 0

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

bit 5 **$\overline{\text{OSCSEN}}$**: Oscillator System Clock Switch Enable bit

1 = Oscillator system clock switch option is disabled (main oscillator is source)
0 = Timer1 oscillator system clock switch option is enabled (oscillator switching is enabled)

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

bit 3-0 **FOSC3:FOSC0**: Oscillator Selection bits

1111 = RC oscillator with OSC2 configured as RA6
1110 = HS oscillator with SW enabled 4x PLL
1101 = EC oscillator with OSC2 configured as RA6 and SW enabled 4x PLL
1100 = EC oscillator with OSC2 configured as RA6 and HW enabled 4x PLL
1011 = Reserved; do not use
1010 = Reserved; do not use
1001 = Reserved; do not use
1000 = Reserved; do not use
0111 = RC oscillator with OSC2 configured as RA6
0110 = HS oscillator with HW enabled 4x PLL
0101 = EC oscillator with OSC2 configured as RA6
0100 = EC oscillator with OSC2 configured as divide by 4 clock output
0011 = RC oscillator with OSC2 configured as divide by 4 clock output
0010 = HS oscillator
0001 = XT oscillator
0000 = LP oscillator

Legend:		
R = Readable bit	P = Programmable bit	U = Unimplemented bit, read as '0'
- n = Value when device is unprogrammed		u = Unchanged from programmed state

26.14 PICSTART Plus Development Programmer

The PICSTART Plus development programmer is an easy-to-use, low-cost, prototype programmer. It connects to the PC via a COM (RS-232) port. MPLAB Integrated Development Environment software makes using the programmer simple and efficient. The PICSTART Plus development programmer supports most PIC devices up to 40 pins. Larger pin count devices, such as the PIC16C92X and PIC17C76X, may be supported with an adapter socket. The PICSTART Plus development programmer is CE compliant.

26.15 PICDEM 1 PIC MCU Demonstration Board

The PICDEM 1 demonstration board demonstrates the capabilities of the PIC16C5X (PIC16C54 to PIC16C58A), PIC16C61, PIC16C62X, PIC16C71, PIC16C8X, PIC17C42, PIC17C43 and PIC17C44. All necessary hardware and software is included to run basic demo programs. The sample microcontrollers provided with the PICDEM 1 demonstration board can be programmed with a PRO MATE II device programmer or a PICSTART Plus development programmer. The PICDEM 1 demonstration board can be connected to the MPLAB ICE in-circuit emulator for testing. A prototype area extends the circuitry for additional application components. Features include an RS-232 interface, a potentiometer for simulated analog input, push button switches and eight LEDs.

26.16 PICDEM.net Internet/Ethernet Demonstration Board

The PICDEM.net demonstration board is an Internet/Ethernet demonstration board using the PIC18F452 microcontroller and TCP/IP firmware. The board supports any 40-pin DIP device that conforms to the standard pinout used by the PIC16F877 or PIC18C452. This kit features a user friendly TCP/IP stack, web server with HTML, a 24L256 Serial EEPROM for Xmodem download to web pages into Serial EEPROM, ICSP/MPLAB ICD 2 interface connector, an Ethernet interface, RS-232 interface and a 16 x 2 LCD display. Also included is the book and CD-ROM *"TCP/IP Lean, Web Servers for Embedded Systems,"* by Jeremy Bentham

26.17 PICDEM 2 Plus Demonstration Board

The PICDEM 2 Plus demonstration board supports many 18, 28 and 40-pin microcontrollers, including PIC16F87X and PIC18FXX2 devices. All the necessary hardware and software is included to run the demonstration programs. The sample microcontrollers provided with the PICDEM 2 demonstration board can be programmed with a PRO MATE II device programmer, PICSTART Plus development programmer, or MPLAB ICD 2 with a Universal Programmer Adapter. The MPLAB ICD 2 and MPLAB ICE in-circuit emulators may also be used with the PICDEM 2 demonstration board to test firmware. A prototype area extends the circuitry for additional application components. Some of the features include an RS-232 interface, a 2 x 16 LCD display, a piezo speaker, an on-board temperature sensor, four LEDs and sample PIC18F452 and PIC16F877 Flash microcontrollers.

26.18 PICDEM 3 PIC16C92X Demonstration Board

The PICDEM 3 demonstration board supports the PIC16C923 and PIC16C924 in the PLCC package. All the necessary hardware and software is included to run the demonstration programs.

26.19 PICDEM 4 8/14/18-Pin Demonstration Board

The PICDEM 4 can be used to demonstrate the capabilities of the 8, 14 and 18-pin PIC16XXXX and PIC18XXXX MCUs, including the PIC16F818/819, PIC16F87/88, PIC16F62XA and the PIC18F1320 family of microcontrollers. PICDEM 4 is intended to showcase the many features of these low pin count parts, including LIN and Motor Control using ECCP. Special provisions are made for low-power operation with the supercapacitor circuit and jumpers allow on-board hardware to be disabled to eliminate current draw in this mode. Included on the demo board are provisions for Crystal, RC or Canned Oscillator modes, a five volt regulator for use with a nine volt wall adapter or battery, DB-9 RS-232 interface, ICD connector for programming via ICSP and development with MPLAB ICD 2, 2 x 16 liquid crystal display, PCB footprints for H-Bridge motor driver, LIN transceiver and EEPROM. Also included are: header for expansion, eight LEDs, four potentiometers, three push buttons and a prototyping area. Included with the kit is a PIC16F627A and a PIC18F1320. Tutorial firmware is included along with the User's Guide.

FIGURE 28-11: TYPICAL AND MAXIMUM I_{T1OSC} vs. V_{DD} (TIMER1 AS SYSTEM CLOCK)

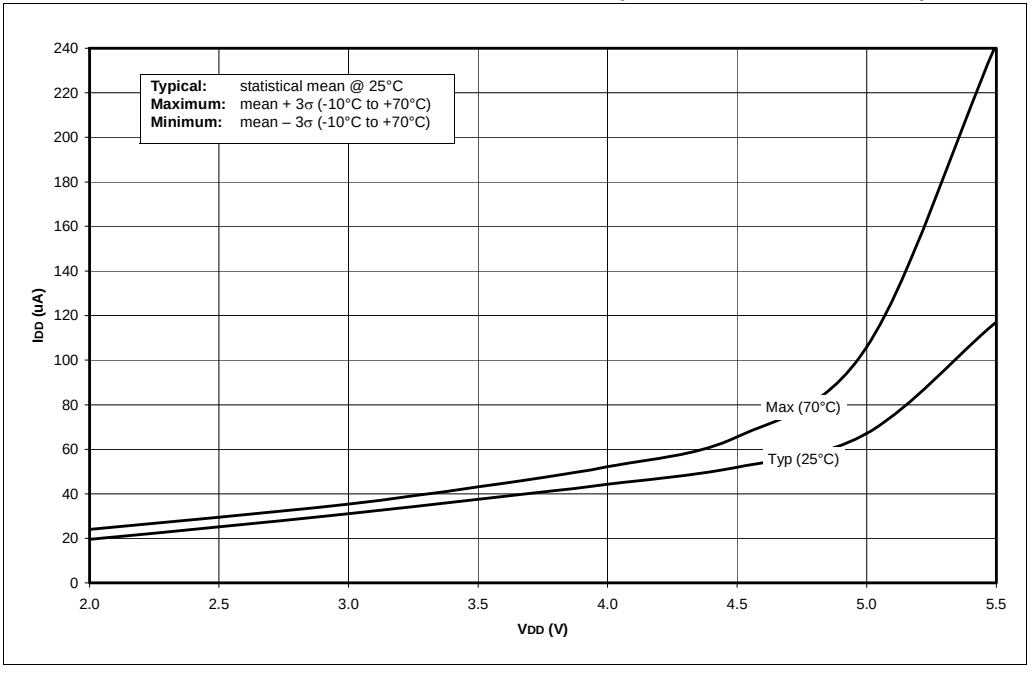


FIGURE 28-12: AVERAGE F_{OSC} vs. V_{DD} FOR VARIOUS R's (RC MODE, C = 20 pF, TEMP = 25°C)

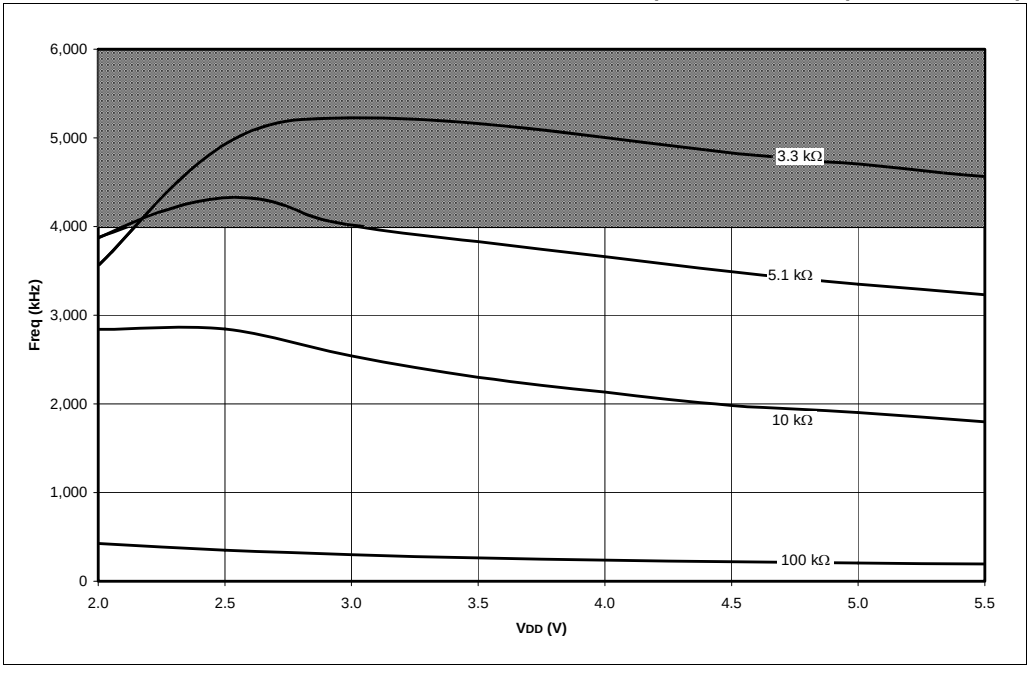


FIGURE 28-13: AVERAGE FOSC vs. VDD FOR VARIOUS R's (RC MODE, C = 100 pF, TEMP = 25°C)

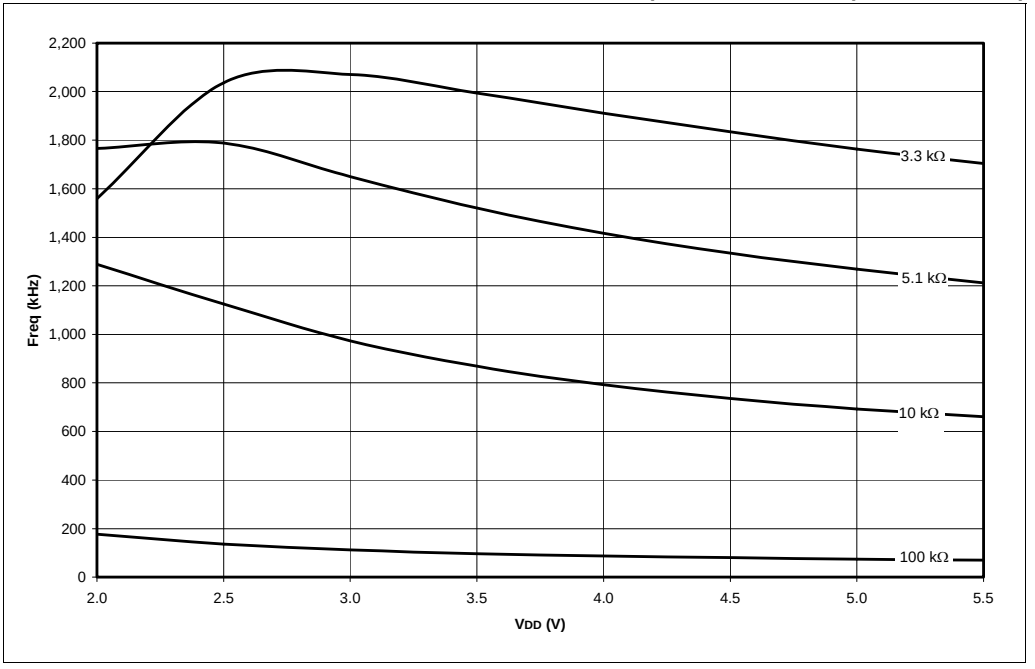


FIGURE 28-14: AVERAGE FOSC vs. VDD FOR VARIOUS R's (RC MODE, C = 300 pF, TEMP = 25°C)