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	Obsolete
Core Processor	PIC
Core Size	8-Bit
Speed	40MHz
Connectivity	CANbus, EBI/EMI, I²C, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, LVD, POR, PWM, WDT
Number of I/O	68
Program Memory Size	64KB (32K x 16)
Program Memory Type	FLASH
EEPROM Size	1K x 8
RAM Size	3.25K x 8
Voltage - Supply (Vcc/Vdd)	2V ~ 5.5V
Data Converters	A/D 16x10b
Oscillator Type	External
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	80-TQFP
Supplier Device Package	80-TQFP (12x12)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic18lf8680t-i-pt

TABLE 1-2: PIC18F6585/8585/6680/8680 PINOUT I/O DESCRIPTIONS (CONTINUED)

Pin Name	Pin Number			Pin Type	Buffer Type	Description
	PIC18F6X8X		PIC18F8X8X			
	TQFP	PLCC	TQFP			
						PORTD is a bidirectional I/O port. These pins have TTL input buffers when external memory is enabled.
RD0/PSP0/AD0	58	3	72			
RD0				I/O	ST	Digital I/O.
PSP0(6)				I/O	TTL	Parallel Slave Port data.
AD0(3)				I/O	TTL	External memory address/data 0.
RD1/PSP1/AD1	55	67	69			
RD1				I/O	ST	Digital I/O.
PSP1(6)				I/O	TTL	Parallel Slave Port data.
AD1(3)				I/O	TTL	External memory address/data 1.
RD2/PSP2/AD2	54	66	68			
RD2				I/O	ST	Digital I/O.
PSP2(6)				I/O	TTL	Parallel Slave Port data.
AD2(3)				I/O	TTL	External memory address/data 2.
RD3/PSP3/AD3	53	65	67			
RD3				I/O	ST	Digital I/O.
PSP3(6)				I/O	TTL	Parallel Slave Port data.
AD3(3)				I/O	TTL	External memory address/data 3.
RD4/PSP4/AD4	52	64	66			
RD4				I/O	ST	Digital I/O.
PSP4(6)				I/O	TTL	Parallel Slave Port data.
AD4(3)				I/O	TTL	External memory address/data 4.
RD5/PSP5/AD5	51	63	65			
RD5				I/O	ST	Digital I/O.
PSP5(6)				I/O	TTL	Parallel Slave Port data.
AD5(3)				I/O	TTL	External memory address/data 5.
RD6/PSP6/AD6	50	62	64			
RD6				I/O	ST	Digital I/O.
PSP6(6)				I/O	TTL	Parallel Slave Port data.
AD6(3)				I/O	TTL	External memory address/data 6.
RD7/PSP7/AD7	49	61	63			
RD7				I/O	ST	Digital I/O.
PSP7(6)				I/O	TTL	Parallel Slave Port data.
AD7(3)				I/O	TTL	External memory address/data 7.

Legend: TTL = TTL compatible input; CMOS = CMOS compatible input or output; ST = Schmitt Trigger input with CMOS levels; Analog = Analog input; I = Input; O = Output; P = Power; OD = Open-Drain (no P diode to V_{DD})

Note 1: Alternate assignment for CCP2 in all operating modes except Microcontroller – applies to PIC18F8X8X only.

2: Default assignment when CCP2MX is set.

3: External memory interface functions are only available on PIC18F8X8X devices.

4: CCP2 is multiplexed with this pin by default when configured in Microcontroller mode; otherwise, it is multiplexed with either RB3 or RC1.

5: PORTH and PORTJ are only available on PIC18F8X8X (80-pin) devices.

6: PSP is available in Microcontroller mode only.

7: On PIC18F8X8X devices, these pins can be multiplexed with RH7/RH6 by changing the ECCPMX configuration bit.

NOTES:

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

Register	Applicable Devices		Power-on Reset, Brown-out Reset	MCLR Resets WDT Reset RESET Instruction Stack Resets	Wake-up via WDT or Interrupt
IPR3	PIC18F6X8X	PIC18F8X8X	1111 1111	1111 1111	uuuu uuuu
PIR3	PIC18F6X8X	PIC18F8X8X	0000 0000	0000 0000	uuuu uuuu
PIE3	PIC18F6X8X	PIC18F8X8X	0000 0000	0000 0000	uuuu uuuu
IPR2	PIC18F6X8X	PIC18F8X8X	-1-1 1111	-1-1 1111	-u-u uuuu
PIR2	PIC18F6X8X	PIC18F8X8X	-0-0 0000	-0-0 0000	-u-u uuuu(1)
PIE2	PIC18F6X8X	PIC18F8X8X	-0-0 0000	-0-0 0000	-u-u uuuu
IPR1	PIC18F6X8X	PIC18F8X8X	1111 1111	1111 1111	uuuu uuuu
PIR1	PIC18F6X8X	PIC18F8X8X	0000 0000	0000 0000	uuuu uuuu(1)
PIE1	PIC18F6X8X	PIC18F8X8X	0000 0000	0000 0000	uuuu uuuu
MEMCON	PIC18F6X8X	PIC18F8X8X	0-00 --00	0-00 --00	u-uu --uu
TRISJ	PIC18F6X8X	PIC18F8X8X	1111 1111	1111 1111	uuuu uuuu
TRISH	PIC18F6X8X	PIC18F8X8X	1111 1111	1111 1111	uuuu uuuu
TRISG	PIC18F6X8X	PIC18F8X8X	---1 1111	---1 1111	---u uuuu
TRISF	PIC18F6X8X	PIC18F8X8X	1111 1111	1111 1111	uuuu uuuu
TRISE	PIC18F6X8X	PIC18F8X8X	0000 -111	0000 -111	uuuu -uuu
TRISD	PIC18F6X8X	PIC18F8X8X	1111 1111	1111 1111	uuuu uuuu
TRISC	PIC18F6X8X	PIC18F8X8X	1111 1111	1111 1111	uuuu uuuu
TRISB	PIC18F6X8X	PIC18F8X8X	1111 1111	1111 1111	uuuu uuuu
TRISA(5,6)	PIC18F6X8X	PIC18F8X8X	-111 1111(5)	-111 1111(5)	-uuu uuuu(5)
LATJ	PIC18F6X8X	PIC18F8X8X	xxxx xxxx	uuuu uuuu	uuuu uuuu
LATH	PIC18F6X8X	PIC18F8X8X	xxxx xxxx	uuuu uuuu	uuuu uuuu
LATG	PIC18F6X8X	PIC18F8X8X	---x xxxx	---u uuuu	---u uuuu
LATF	PIC18F6X8X	PIC18F8X8X	xxxx xxxx	uuuu uuuu	uuuu uuuu
LATE	PIC18F6X8X	PIC18F8X8X	xxxx xxxx	uuuu uuuu	uuuu uuuu
LATD	PIC18F6X8X	PIC18F8X8X	xxxx xxxx	uuuu uuuu	uuuu uuuu
LATC	PIC18F6X8X	PIC18F8X8X	xxxx xxxx	uuuu uuuu	uuuu uuuu
LATB	PIC18F6X8X	PIC18F8X8X	xxxx xxxx	uuuu uuuu	uuuu uuuu
LATA(5,6)	PIC18F6X8X	PIC18F8X8X	-xxx xxxx(5)	-uuu uuuu(5)	-uuu uuuu(5)

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.

TABLE 4-2: SPECIAL FUNCTION REGISTER MAP

Address	Name	Address	Name	Address	Name	Address	Name
FFFh	TOSU	FDFh	INDF2(3)	FBFh	CCPR1H	F9Fh	IPR1
FFEh	TOSH	FDEh	POSTINC2(3)	FBEh	CCPR1L	F9Eh	PIR1
FFDh	TOSL	FDDh	POSTDEC2(3)	FBDh	CCP1CON	F9Dh	PIE1
FFCh	STKPTR	FDCh	PREINC2(3)	FBCh	CCPR2H	F9Ch	MEMCON(2)
FFBh	PCLATU	FDBh	PLUSW2(3)	FBBh	CCPR2L	F9Bh	—(1)
FFAh	PCLATH	FDAh	FSR2H	FBAh	CCP2CON	F9Ah	TRISJ(2)
FF9h	PCL	FD9h	FSR2L	FB9h	—(1)	F99h	TRISH(2)
FF8h	TBLPTRU	FD8h	STATUS	FB8h	—(1)	F98h	TRISG
FF7h	TBLPTRH	FD7h	TMR0H	FB7h	—(1)	F97h	TRISF
FF6h	TBLPTRL	FD6h	TMR0L	FB6h	ECCP1AS	F96h	TRISE
FF5h	TABLAT	FD5h	T0CON	FB5h	CVRCON	F95h	TRISD
FF4h	PRODH	FD4h	—(1)	FB4h	CMCON	F94h	TRISC
FF3h	PRODL	FD3h	OSCCON	FB3h	TMR3H	F93h	TRISB
FF2h	INTCON	FD2h	LVDCON	FB2h	TMR3L	F92h	TRISA
FF1h	INTCON2	FD1h	WDTCON	FB1h	T3CON	F91h	LATJ(2)
FF0h	INTCON3	FD0h	RCON	FB0h	PSPCON	F90h	LATH(2)
FEFh	INDF0(3)	FCFh	TMR1H	FAFh	SPBRG	F8Fh	LATG
FEEh	POSTINC0(3)	FCEh	TMR1L	FAEh	RCREG	F8Eh	LATF
FEDh	POSTDEC0(3)	FCDh	T1CON	FADh	TXREG	F8Dh	LATE
FECh	PREINC0(3)	FCCh	TMR2	FACh	TXSTA	F8Ch	LATD
FEBh	PLUSW0(3)	FCBh	PR2	FABh	RCSTA	F8Bh	LATC
FEAh	FSR0H	FCAh	T2CON	FAAh	EEADRH	F8Ah	LATB
FE9h	FSR0L	FC9h	SSPBUF	FA9h	EEADR	F89h	LATA
FE8h	WREG	FC8h	SSPADD	FA8h	EEDATA	F88h	PORTJ(2)
FE7h	INDF1(3)	FC7h	SSPSTAT	FA7h	EECON2	F87h	PORTH(2)
FE6h	POSTINC1(3)	FC6h	SSPCON1	FA6h	EECON1	F86h	PORTG
FE5h	POSTDEC1(3)	FC5h	SSPCON2	FA5h	IPR3	F85h	PORTF
FE4h	PREINC1(3)	FC4h	ADRESH	FA4h	PIR3	F84h	PORTE
FE3h	PLUSW1(3)	FC3h	ADRESL	FA3h	PIE3	F83h	PORTD
FE2h	FSR1H	FC2h	ADCON0	FA2h	IPR2	F82h	PORTC
FE1h	FSR1L	FC1h	ADCON1	FA1h	PIR2	F81h	PORTB
FE0h	BSR	FC0h	ADCON2	FA0h	PIE2	F80h	PORTA

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

2: This register is not available on PIC18F6X8X devices.

3: This is not a physical register.

TABLE 4-3: REGISTER FILE SUMMARY

File Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR, BOR	Details on page:
TOSU	—	—	—	Top-of-Stack Upper Byte (TOS<20:16>)					---0 0000	36, 54
TOSH	Top-of-Stack High Byte (TOS<15:8>)								0000 0000	36, 54
TOSL	Top-of-Stack Low Byte (TOS<7:0>)								0000 0000	36, 54
STKPTR	STKFUL	STKUNF	—	Return Stack Pointer					00-0 0000	36, 55
PCLATU	—	—	bit 21	Holding Register for PC<20:16>					--00 0000	36, 56
PCLATH	Holding Register for PC<15:8>								0000 0000	36, 56
PCL	PC Low Byte (PC<7:0>)								0000 0000	36, 56
TBLPTRU	—	—	bit 21(2)	Program Memory Table Pointer Upper Byte (TBLPTR<20:16>)					--00 0000	36, 86
TBLPTRH	Program Memory Table Pointer High Byte (TBLPTR<15:8>)								0000 0000	36, 86
TBLPTRL	Program Memory Table Pointer Low Byte (TBLPTR<7:0>)								0000 0000	36, 86
TABLAT	Program Memory Table Latch								0000 0000	36, 86
PRODH	Product Register High Byte								xxxx xxxx	36, 107
PRODL	Product Register Low Byte								xxxx xxxx	36, 107
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	0000 000x	36, 111
INTCON2	RBPU	INTEDG0	INTEDG1	INTEDG2	INTEDG3	TMR0IP	INT3IP	RBIP	1111 1111	36, 112
INTCON3	INT2IP	INT1IP	INT3IE	INT2IE	INT1IE	INT3IF	INT2IF	INT1IF	1100 0000	36, 113
INDF0	Uses contents of FSR0 to address data memory – value of FSR0 not changed (not a physical register)								n/a	79
POSTINC0	Uses contents of FSR0 to address data memory – value of FSR0 post-incremented (not a physical register)								n/a	79
POSTDEC0	Uses contents of FSR0 to address data memory – value of FSR0 post-decremented (not a physical register)								n/a	79
PREINC0	Uses contents of FSR0 to address data memory – value of FSR0 pre-incremented (not a physical register)								n/a	79
PLUSW0	Uses contents of FSR0 to address data memory – value of FSR0 pre-incremented (not a physical register) – value of FSR0 offset by value in WREG								n/a	79
FSR0H	—	—	—	—	Indirect Data Memory Address Pointer 0 High Byte				---- 0000	36, 79
FSR0L	Indirect Data Memory Address Pointer 0 Low Byte								xxxx xxxx	36, 79
WREG	Working Register								xxxx xxxx	36
INDF1	Uses contents of FSR1 to address data memory – value of FSR1 not changed (not a physical register)								n/a	79
POSTINC1	Uses contents of FSR1 to address data memory – value of FSR1 post-incremented (not a physical register)								n/a	79
POSTDEC1	Uses contents of FSR1 to address data memory – value of FSR1 post-decremented (not a physical register)								n/a	79
PREINC1	Uses contents of FSR1 to address data memory – value of FSR1 pre-incremented (not a physical register)								n/a	79
PLUSW1	Uses contents of FSR1 to address data memory – value of FSR1 pre-incremented (not a physical register) – value of FSR1 offset by value in WREG								n/a	79
FSR1H	—	—	—	—	Indirect Data Memory Address Pointer 1 High Byte				---- 0000	37, 79
FSR1L	Indirect Data Memory Address Pointer 1 Low Byte								xxxx xxxx	37, 79
BSR	—	—	—	—	Bank Select Register				---- 0000	37, 78
INDF2	Uses contents of FSR2 to address data memory – value of FSR2 not changed (not a physical register)								n/a	79
POSTINC2	Uses contents of FSR2 to address data memory – value of FSR2 post-incremented (not a physical register)								n/a	79
POSTDEC2	Uses contents of FSR2 to address data memory – value of FSR2 post-decremented (not a physical register)								n/a	79
PREINC2	Uses contents of FSR2 to address data memory – value of FSR2 pre-incremented (not a physical register)								n/a	79
PLUSW2	Uses contents of FSR2 to address data memory – value of FSR2 pre-incremented (not a physical register) – value of FSR2 offset by value in WREG								n/a	79
FSR2H	—	—	—	—	Indirect Data Memory Address Pointer 2 High Byte				---- 0000	37, 79
FSR2L	Indirect Data Memory Address Pointer 2 Low Byte								xxxx xxxx	37, 79

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

Note 1: RA6 and associated bits are configured as port pins in RCIO and ECIO Oscillator mode only and read '0' in all other oscillator modes.

2: Bit 21 of the TBLPTRU allows access to the device configuration bits.

3: These registers are unused on PIC18F6X80 devices; always maintain these clear.

4: These bits have multiple functions depending on the CAN module mode selection.

5: Meaning of this register depends on whether this buffer is configured as transmit or receive.

6: RG5 is available as an input when $\overline{\text{MCLR}}$ is disabled.

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

12.2 Timer1 Oscillator

A crystal oscillator circuit is built-in between pins T1OSI (input) and T1OSO (amplifier output). It is enabled by setting control bit, T1OSCEN (T1CON<3>). The oscillator is a low-power oscillator rated up to 200 kHz. It will continue to run during Sleep. It is primarily intended for a 32 kHz crystal. Table 12-1 shows the capacitor selection for the Timer1 oscillator.

The user must provide a software time delay to ensure proper start-up of the Timer1 oscillator.

TABLE 12-1: CAPACITOR SELECTION FOR THE ALTERNATE OSCILLATOR

Osc Type	Freq	C1	C2
LP	32 kHz	TBD(1)	TBD(1)
Crystal to be Tested:			
32.768 kHz	Epson C-001R32.768K-A	± 20 PPM	

Note 1: Microchip suggests 33 pF as a starting point in validating the oscillator circuit.

2: Higher capacitance increases the stability of the oscillator but also increases the start-up time.

3: Since each resonator/crystal has its own characteristics, the user should consult the resonator/crystal manufacturer for appropriate values of external components.

4: Capacitor values are for design guidance only.

12.3 Timer1 Interrupt

The TMR1 register pair (TMR1H:TMR1L) increments from 0000h to 0FFFFh and rolls over to 0000h. The TMR1 interrupt, if enabled, is generated on overflow which is latched in interrupt flag bit, TMR1IF (PIR1<0>). This interrupt can be enabled/disabled by setting/clearing TMR1 interrupt enable bit, TMR1IE (PIE1<0>).

12.4 Resetting Timer1 Using a CCP Trigger Output

If the CCP module is configured in Compare mode to generate a "special event trigger" (CCP1M3:CCP1M0 = 1011), this signal will reset Timer1 and start an A/D conversion (if the A/D module is enabled).

Note: The special event triggers from the CCP1 module will not set interrupt flag bit TMR1IF (PIR1<0>).

Timer1 must be configured for either Timer or Synchronized Counter mode to take advantage of this feature. If Timer1 is running in Asynchronous Counter mode, this Reset operation may not work.

In the event that a write to Timer1 coincides with a special event trigger from CCP1, the write will take precedence.

In this mode of operation, the CCPR1H:CCPR1L register pair effectively becomes the period register for Timer1.

12.5 Timer1 16-Bit Read/Write Mode

Timer1 can be configured for 16-bit reads and writes (see Figure 12-2). When the RD16 control bit (T1CON<7>) is set, the address for TMR1H is mapped to a buffer register for the high byte of Timer1. A read from TMR1L will load the contents of the high byte of Timer1 into the Timer1 high byte buffer. This provides the user with the ability to accurately read all 16 bits of Timer1 without having to determine whether a read of the high byte, followed by a read of the low byte, is valid due to a rollover between reads.

A write to the high byte of Timer1 must also take place through the TMR1H Buffer register. Timer1 high byte is updated with the contents of TMR1H when a write occurs to TMR1L. This allows a user to write all 16 bits to both the high and low bytes of Timer1 at once.

The high byte of Timer1 is not directly readable or writable in this mode. All reads and writes must take place through the Timer1 High Byte Buffer register. Writes to TMR1H do not clear the Timer1 prescaler. The prescaler is only cleared on writes to TMR1L.

TABLE 12-2: REGISTERS ASSOCIATED WITH TIMER1 AS A TIMER/COUNTER

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 0000	0000 0000
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	0111 1111	0111 1111
TMR1L	Holding Register for the Least Significant Byte of the 16-bit TMR1 Register								xxxx xxxx	uuuu uuuu
TMR1H	Holding Register for the Most Significant Byte of the 16-bit TMR1 Register								xxxx xxxx	uuuu uuuu
T1CON	RD16	—	T1CKPS1	T1CKPS0	T1OSCEN	T1SYNC	TMR1CS	TMR1ON	0-00 0000	u-uu uuuu

Legend: x = unknown, u = unchanged, – = unimplemented, read as '0'. Shaded cells are not used by the Timer1 module.

17.3.2 OPERATION

When initializing the SPI, several options need to be specified. This is done by programming the appropriate control bits (SSPCON1<5:0> and SSPSTAT<7:6>). These control bits allow the following to be specified:

- Master mode (SCK is the clock output)
- Slave mode (SCK is the clock input)
- Clock Polarity (Idle state of SCK)
- Data Input Sample Phase (middle or end of data output time)
- Clock Edge (output data on rising/falling edge of SCK)
- Clock Rate (Master mode only)
- Slave Select mode (Slave mode only)

The MSSP consists of a Transmit/Receive Shift register (SSPSR) and a Buffer register (SSPBUF). The SSPSR shifts the data in and out of the device, MSb first. The SSPBUF holds the data that was written to the SSPSR, until the received data is ready. Once the 8 bits of data have been received, that byte is moved to the SSPBUF register. Then the Buffer Full detect bit, BF (SSPSTAT<0>) and the interrupt flag bit, SSPIF, are set. This double-buffering of the received data (SSPBUF) allows the next byte to start reception before reading the data that was just received. Any write to the SSPBUF register during transmission/reception of data will be ignored and the Write Collision detect bit, WCOL (SSPCON1<7>), will be set. User software must clear the WCOL bit so that it can be determined if the following write(s) to the SSPBUF register completed successfully.

When the application software is expecting to receive valid data, the SSPBUF should be read before the next byte of data to transfer is written to the SSPBUF. Buffer Full bit, BF (SSPSTAT<0>), indicates when SSPBUF has been loaded with the received data (transmission is complete). When the SSPBUF is read, the BF bit is cleared. This data may be irrelevant if the SPI is only a transmitter. Generally, the MSSP interrupt is used to determine when the transmission/reception has completed. The SSPBUF must be read and/or written. If the interrupt method is not going to be used, then software polling can be done to ensure that a write collision does not occur. Example 17-1 shows the loading of the SSPBUF (SSPSR) for data transmission.

The SSPSR is not directly readable or writable and can only be accessed by addressing the SSPBUF register. Additionally, the MSSP Status register (SSPSTAT) indicates the various status conditions.

EXAMPLE 17-1: LOADING THE SSPBUF (SSPSR) REGISTER

```
LOOP    BTFSS   SSPSTAT, BF     ;Has data been received(transmit complete)?
        BRA     LOOP            ;No
        MOVF    SSPBUF, W       ;WREG reg = contents of SSPBUF
        MOVWF   RXDATA          ;Save in user RAM, if data is meaningful
        MOVF    TXDATA, W       ;W reg = contents of TXDATA
        MOVWF   SSPBUF          ;New data to xmit
```

TABLE 18-5: REGISTERS ASSOCIATED WITH ASYNCHRONOUS TRANSMISSION

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
RCSTA	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	0000 000x	0000 000x
TXREG	USART Transmit Register								0000 0000	0000 0000
TXSTA	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	0000 0010	0000 0010
BAUDCON	—	RCIDL	—	SCKP	BRG16	—	WUE	ABDEN	-1-1 0-00	-1-1 0-00
SPBRGH	Baud Rate Generator Register, High Byte								0000 0000	0000 0000
SPBRG	Baud Rate Generator Register, Low Byte								0000 0000	0000 0000

Legend: x = unknown, – = unimplemented locations read as '0'. Shaded cells are not used for asynchronous transmission.

18.2.5 BREAK CHARACTER SEQUENCE

The enhanced USART module has the capability of sending the special break character sequences that are required by the LIN bus standard. The break character transmit consists of a Start bit, followed by twelve '0' bits and a Stop bit. The frame break character is sent whenever the SENDB and TXEN bits (TXSTA<3> and TXSTA<5>) are set while the Transmit Shift register is loaded with data. Note that the value of data written to TXREG will be ignored and all '0's will be transmitted.

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

Note that the data value written to the TXREG for the break character is ignored. The write simply serves the purpose of initiating the proper sequence.

The TRMT bit indicates when the transmit operation is active or Idle, just as it does during normal transmission. See Figure 18-9 for the timing of the break character sequence.

18.2.5.1 Break and Sync Transmit Sequence

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

1. Configure the USART for the desired mode.
2. Set the TXEN and SENDB bits to set up the break character.
3. Load the TXREG with a dummy character to initiate transmission (the value is ignored).
4. Write '55h' to TXREG to load the sync character into the transmit FIFO buffer.
5. After the break has been sent, the SENDB bit is reset by hardware. The sync character now transmits in the preconfigured mode.

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

18.2.6 RECEIVING A BREAK CHARACTER

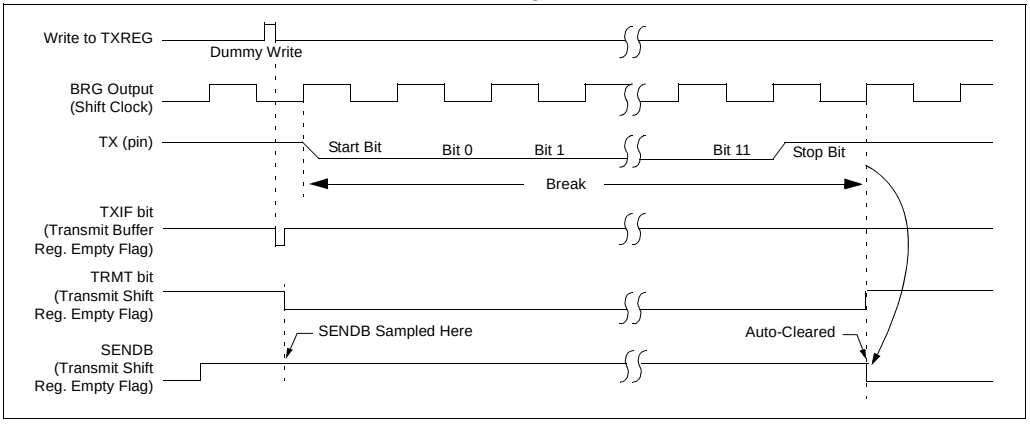
The enhanced USART module can receive a break character in two ways.

The first method forces the configuration of the baud rate at a frequency of 9/13 the typical speed. This allows for the Stop bit transition to be at the correct sampling location (13 bits for break versus Start bit and 8 data bits for typical data).

The second method uses the auto-wake-up feature described in **Section 18.2.4 "Auto-Wake-up on Sync Break Character"**. By enabling this feature, the USART will sample the next two transitions on RX/DT, cause an RCIF interrupt, and receive the next data byte followed by another interrupt.

Note that following a break character, the user will typically want to enable the auto-baud rate detect feature. For both methods, the user can set the ABD bit once the TXIF interrupt is observed.

FIGURE 18-9: SEND BREAK CHARACTER SEQUENCE



23.2.3.1 Programmable TX/RX and Auto RTR Buffers

The ECAN module contains 6 message buffers that can be programmed as transmit or receive buffers. Any of these buffers can also be programmed to automatically handle RTR messages.

Note:	These registers are not used in Mode 0.

REGISTER 23-22: BnCON: TX/RX BUFFER n CONTROL REGISTERS IN RECEIVE MODE [0 ≤ n ≤ 5, TXnEN (BSEL0<n>) = 0](1)

R/C-0	R/W-0	R-0	R-0	R-0	R-0	R-0	R-0
RXFUL	RXM1	RTRRO	FILHIT4	FILHIT3	FILHIT2	FILHIT1	FILHIT0
bit 7							bit 0

bit 7 **RXFUL:** Receive Full Status bit(1)
1 = Receive buffer contains a received message
0 = Receive buffer is open to receive a new message

Note: This bit is set by the CAN module upon receiving a message and must be cleared by software after the buffer is read. As long as RXFUL is set, no new message will be loaded and buffer will be considered full.

bit 6 **RXM1:** Receive Buffer Mode bit
1 = Receive all messages including partial and invalid (acceptance filters are ignored)
0 = Receive all valid messages as per acceptance filters

bit 5 **RTRRO:** Read-Only Remote Transmission Request bit for Received Message
1 = Received message is a remote transmission request
0 = Received message is not a remote transmission request

bit 4-0 **FILHIT4:FILHIT0:** Filter Hit bits
These bits indicate which acceptance filter enabled the last message reception into this buffer.
01111 = Acceptance Filter 15 (RXF15)
01110 = Acceptance Filter 14 (RXF14)
...
00001 = Acceptance Filter 1 (RXF1)
00000 = Acceptance Filter 0 (RXF0)

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

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 23-23: BnCON: TX/RX BUFFER n CONTROL REGISTERS IN TRANSMIT MODE [0 ≤ n ≤ 5, TXnEN (BSEL0<n>) = 1](1)

R/W-0	R-0	R-0	R-0	R/W-0	R/W-0	R/W-0	R/W-0
TXBIF	TXABT	TXLARB	TXERR	TXREQ	RTREN	TXPRI1	TXPRI0
bit 7							bit 0

bit 7 **TXBIF:** Transmit Buffer Interrupt Flag bit(1)
1 = A message is successfully transmitted
0 = No message was transmitted

bit 6 **TXABT:** Transmission Aborted Status bit(1)
1 = Message was aborted
0 = Message was not aborted

bit 5 **TXLARB:** Transmission Lost Arbitration Status bit(2)
1 = Message lost arbitration while being sent
0 = Message did not lose arbitration while being sent

bit 4 **TXERR:** Transmission Error Detected Status bit(2)
1 = A bus error occurred while the message was being sent
0 = A bus error did not occur while the message was being sent

bit 3 **TXREQ:** Transmit Request Status bit(3)
1 = Requests sending a message; clears the TXABT, TXLARB, and TXERR bits
0 = Automatically cleared when the message is successfully sent

Note: Clearing this bit in software while the bit is set will request a message abort.

bit 2 **RTREN:** Automatic Remote Transmission Request Enable bit
1 = When a remote transmission request is received, TXREQ will be automatically set
0 = When a remote transmission request is received, TXREQ will be unaffected

bit 1-0 **TXPRI1:TXPRI0:** Transmit Priority bits(4)
11 = Priority Level 3 (highest priority)
10 = Priority Level 2
01 = Priority Level 1
00 = Priority Level 0 (lowest priority)

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

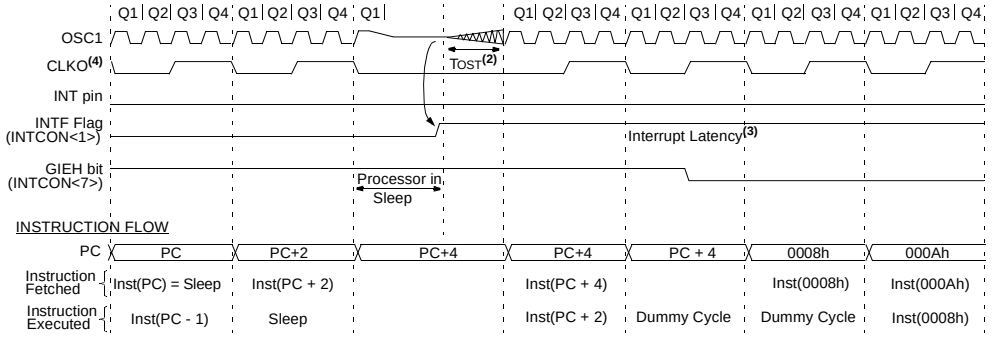
2: This bit is automatically cleared when TXREQ is set.

3: While TXREQ is set or transmission is in progress, transmit buffer registers remain read-only.

4: These bits set the order in which the transmit buffer will be transferred. They do not alter the CAN message identifier.

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 24-2: WAKE-UP FROM SLEEP THROUGH INTERRUPT(1,2)



Note 1: XT, HS or LP Oscillator mode assumed.
2: GIE = 1 assumed. In this case after wake-up, the processor jumps to the interrupt routine. If GIE = 0, execution will continue in-line.
3: TOST = 1024 TOSC (drawing not to scale). This delay will not occur for RC and EC Oscillator modes.
4: CLKO is not available in these oscillator modes but shown here for timing reference.

BNC **Branch if Not Carry**

Syntax: [*label*] BNC n

Operands: -128 ≤ n ≤ 127

Operation: if carry bit is '0'
(PC) + 2 + 2n → PC

Status Affected: None

Encoding:

1110	0011	nnnn	nnnn

Description: If the Carry bit is '0', then the program will branch.
The 2's complement number '2n' is added to the PC. Since the PC will have incremented to fetch the next instruction, the new address will be PC+2+2n. This instruction is then a two-cycle instruction.

Words: 1

Cycles: 1(2)

Q Cycle Activity:
If Jump:

Q1	Q2	Q3	Q4
Decode	Read literal 'n'	Process Data	Write to PC
No operation	No operation	No operation	No operation

If No Jump:

Q1	Q2	Q3	Q4
Decode	Read literal 'n'	Process Data	No operation

Example: `HERE BNC Jump`

Before Instruction
PC = address `(HERE)`
After Instruction
If Carry = 0;
PC = address `(Jump)`
If Carry = 1;
PC = address `(HERE+2)`

BNN **Branch if Not Negative**

Syntax: [*label*] BNN n

Operands: -128 ≤ n ≤ 127

Operation: if negative bit is '0'
(PC) + 2 + 2n → PC

Status Affected: None

Encoding:

1110	0111	nnnn	nnnn

Description: If the Negative bit is '0', then the program will branch.
The 2's complement number '2n' is added to the PC. Since the PC will have incremented to fetch the next instruction, the new address will be PC+2+2n. This instruction is then a two-cycle instruction.

Words: 1

Cycles: 1(2)

Q Cycle Activity:
If Jump:

Q1	Q2	Q3	Q4
Decode	Read literal 'n'	Process Data	Write to PC
No operation	No operation	No operation	No operation

If No Jump:

Q1	Q2	Q3	Q4
Decode	Read literal 'n'	Process Data	No operation

Example: `HERE BNN Jump`

Before Instruction
PC = address `(HERE)`
After Instruction
If Negative = 0;
PC = address `(Jump)`
If Negative = 1;
PC = address `(HERE+2)`

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 27-15: EXAMPLE SPI MASTER MODE TIMING (CKE = 0)

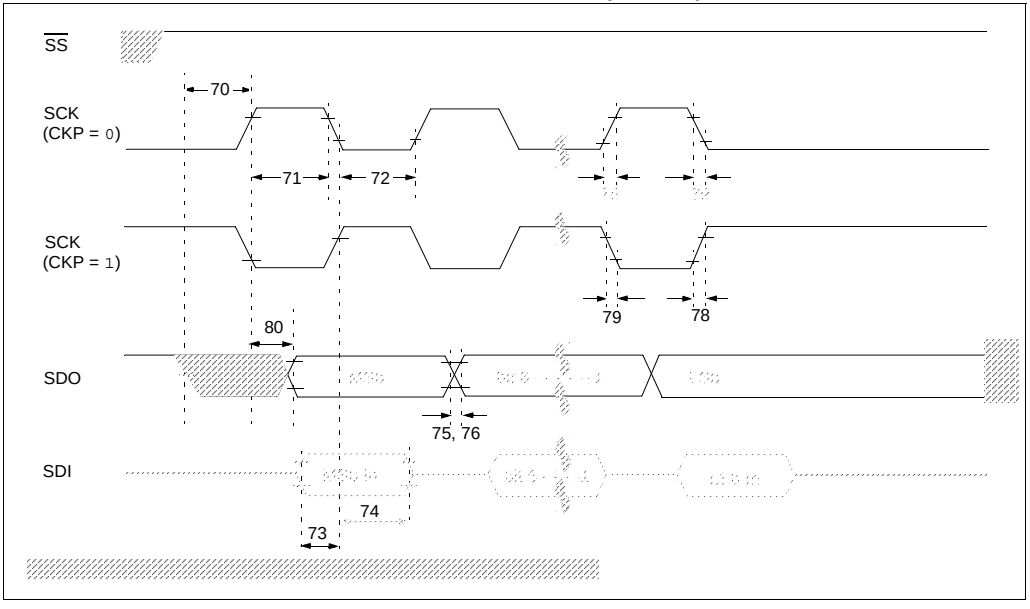


TABLE 27-15: EXAMPLE SPI MODE REQUIREMENTS (MASTER MODE, CKE = 0)

Param. No.	Symbol	Characteristic		Min	Max	Units	Conditions
70	TssL2scH, TssL2scL	$\overline{SS}$ ↓ to SCK ↓ or SCK ↑ Input		Tcy	—	ns	
71	TscH	SCK Input High Time (Slave mode)	Continuous	1.25 Tcy + 30	—	ns	
71A			Single Byte	40	—	ns	**(Note 1)**
72	TscL	SCK Input Low Time (Slave mode)	Continuous	1.25 Tcy + 30	—	ns	
72A			Single Byte	40	—	ns	**(Note 1)**
73	TdiV2scH, TdiV2scL	Setup Time of SDI Data Input to SCK Edge		100	—	ns	
73A	Tb2b	Last Clock Edge of Byte 1 to the 1st Clock Edge of Byte 2		1.5 Tcy + 40	—	ns	**(Note 2)**
74	TscH2diL, TscL2diL	Hold Time of SDI Data Input to SCK Edge		100	—	ns	
75	TdoR	SDO Data Output Rise Time	PIC18FXX8X	—	25	ns	
			PIC18LFXX8X	—	45	ns	
76	TdoF	SDO Data Output Fall Time		—	25	ns	
78	TscR	SCK Output Rise Time (Master mode)	PIC18FXX8X	—	25	ns	
			PIC18LFXX8X	—	45	ns	
79	TscF	SCK Output Fall Time (Master mode)		—	25	ns	
80	TscH2doV, TscL2doV	SDO Data Output Valid after SCK Edge	PIC18FXX8X	—	50	ns	
			PIC18LFXX8X	—	100	ns	

Note 1: Requires the use of Parameter #73A.

2: Only if Parameter #71A and #72A are used.

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

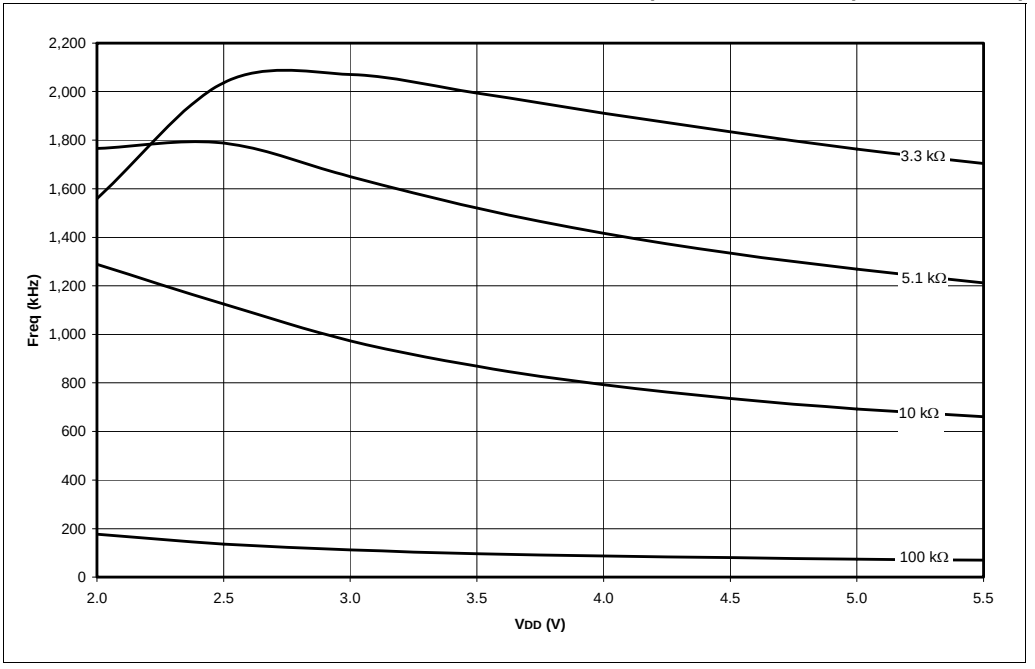
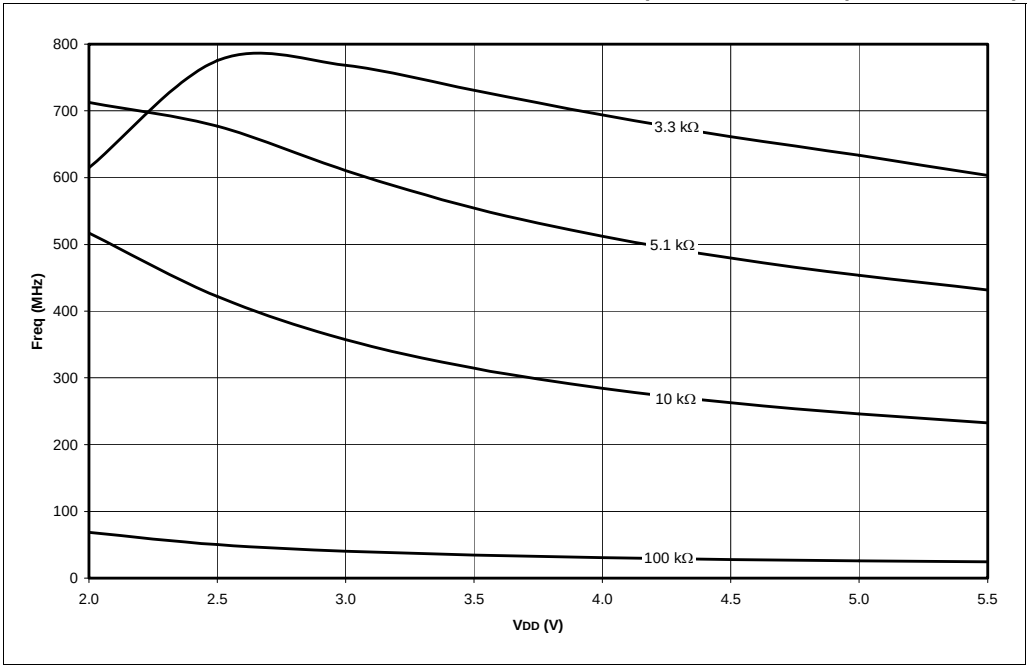


FIGURE 28-14: AVERAGE FOSC vs. VDD FOR VARIOUS R's (RC MODE, C = 300 pF, TEMP = 25°C)



C

NOTES:

Note the following details of the code protection feature on Microchip devices:

- Microchip products meet the specification contained in their particular Microchip Data Sheet.
- Microchip believes that its family of products is one of the most secure families of its kind on the market today, when used in the intended manner and under normal conditions.
- There are dishonest and possibly illegal methods used to breach the code protection feature. All of these methods, to our knowledge, require using the Microchip products in a manner outside the operating specifications contained in Microchip's Data Sheets. Most likely, the person doing so is engaged in theft of intellectual property.
- Microchip is willing to work with the customer who is concerned about the integrity of their code.
- Neither Microchip nor any other semiconductor manufacturer can guarantee the security of their code. Code protection does not mean that we are guaranteeing the product as "unbreakable."

Code protection is constantly evolving. We at Microchip are committed to continuously improving the code protection features of our products. Attempts to break Microchip's code protection feature may be a violation of the Digital Millennium Copyright Act. If such acts allow unauthorized access to your software or other copyrighted work, you may have a right to sue for relief under that Act.

Trademarks

Printed on recycled paper.

ISBN: 9781620769638

QUALITY MANAGEMENT SYSTEM
CERTIFIED BY DNV
═ ISO/TS 16949 ═

Microchip received ISO/TS-16949:2009 certification for its worldwide headquarters, design and wafer fabrication facilities in Chandler and Tempe, Arizona; Gresham, Oregon and design centers in California and India. The Company's quality system processes and procedures are for its PIC® MCUs and dsPIC® DSCs, KEELOQ® code hopping devices, Serial EEPROMs, microperipherals, nonvolatile memory and analog products. In addition, Microchip's quality system for the design and manufacture of development systems is ISO 9001:2000 certified.