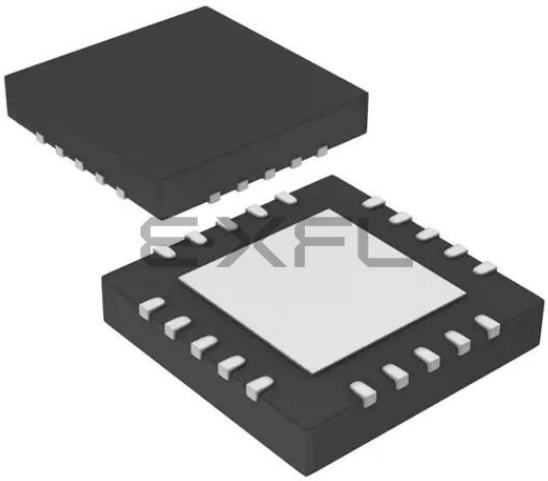


Welcome to E-XFL.COM

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	16-Bit
Speed	32MHz
Connectivity	I²C, IrDA, LINbus, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, HLVD, POR, PWM, WDT
Number of I/O	17
Program Memory Size	8KB (2.75K x 24)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	512 x 8
Voltage - Supply (Vcc/Vdd)	1.8V ~ 3.6V
Data Converters	A/D 12x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	20-VQFN Exposed Pad
Supplier Device Package	20-VQFN (5x5)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic24f08kl201-i-mq

TABLE 1-4: PIC24F16KL40X/30X FAMILY PINOUT DESCRIPTIONS (CONTINUED)

Function	Pin Number				I/O	Buffer	Description
	20-Pin PDIP/ SSOP/ SOIC	20-Pin QFN	28-Pin SPDIP/ SSOP/ SOIC	28-Pin QFN			
SOSCI	9	6	11	8	I	ANA	Secondary Oscillator Input
SOSCO	10	7	12	9	O	ANA	Secondary Oscillator Output
$\overline{SS1}$	12	9	26	23	O	—	SPI1 Slave Select
$\overline{SS2}$	15	12	23	20	O	—	SPI2 Slave Select
T1CK	13	10	18	15	I	ST	Timer1 Clock
T3CK	18	15	26	23	I	ST	Timer3 Clock
T3G	6	3	6	3	I	ST	Timer3 External Gate Input
U1CTS	12	9	17	14	I	ST	UART1 Clear-to-Send Input
U1RTS	13	10	18	15	O	—	UART1 Request-to-Send Output
U1RX	6	3	6	3	I	ST	UART1 Receive
U1TX	11	8	16	13	O	—	UART1 Transmit
U2CTS	10	7	12	9	I	ST	UART2 Clear-to-Send Input
U2RTS	9	6	11	8	O	—	UART2 Request-to-Send Output
U2RX	5	2	5	2	I	ST	UART2 Receive
U2TX	4	1	4	1	O	—	UART2 Transmit
ULPWU	4	1	4	1	I	ANA	Ultra Low-Power Wake-up Input
V_{DD}	20	17	13, 28	10, 25	P	—	Positive Supply for Peripheral Digital Logic and I/O Pins
V_{REF+}	2	19	2	27	I	ANA	A/D Reference Voltage Input (+)
V_{REF-}	3	20	3	28	I	ANA	A/D Reference Voltage Input (-)
V_{SS}	19	16	8, 27	5, 24	P	—	Ground Reference for Logic and I/O Pins

Legend: TTL = TTL input buffer
ANA = Analog level input/output
ST = Schmitt Trigger input buffer
I^2C = I^2C™/SMBus input buffer

TABLE 4-6: TIMER REGISTER MAP

File Name	Addr	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	All Resets
TMR1	0100	Timer1 Register																0000
PR1	0102	Timer1 Period Register																FFFF
T1CON	0104	TON	—	TSIDL	—	—	—	T1ECS1	T1ECS0	—	TGATE	TCKPS1	TCKPS0	—	TSYNC	TCS	—	0000
TMR2	0106	—	—	—	—	—	—	—	—	Timer2 Register								0000
PR2	0108	—	—	—	—	—	—	—	—	Timer2 Period Register								00FF
T2CON	010A	—	—	—	—	—	—	—	—	—	T2OUTPS3	T2OUTPS2	T2OUTPS1	T2OUTPS0	TMR2ON	T2CKPS1	T2CKPS0	0000
TMR3	010C	Timer3 Register																0000
T3GCON	010E	—	—	—	—	—	—	—	—	TMR3GE	T3GPOL	T3GTM	T3GSPM	T3GGO/$\overline{\text{T3DONE}}$	T3GVAL	T3GSS1	T3GSS0	0000
T3CON	0110	—	—	—	—	—	—	—	—	TMR3CS1	TMR3CS0	T3CKPS1	T3CKPS0	T3OSCEN	$\overline{\text{T3SYNC}}$	—	TMR3ON	0000
TMR4(1)	0112	—	—	—	—	—	—	—	—	Timer4 Register								0000
PR4(1)	0114	—	—	—	—	—	—	—	—	Timer4 Period Register								00FF
T4CON(1)	0116	—	—	—	—	—	—	—	—	—	T4OUTPS3	T4OUTPS2	T4OUTPS1	T4OUTPS0	TMR4ON	T4CKPS1	T4CKPS0	0000
CCPTMRS0(1)	013C	—	—	—	—	—	—	—	—	—	C3TSEL0(1)	—	—	C2TSEL0	—	—	C1TSEL0	0000

Legend: — = unimplemented, read as '0'. Reset values are shown in hexadecimal.

Note 1: These bits and/or registers are unimplemented on PIC24FXXKL10X and PIC24FXXKL20X family devices; read as '0'.

TABLE 4-7: CCP/ECCP REGISTER MAP

File Name	Addr	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	All Resets
CCP1CON	0190	—	—	—	—	—	—	—	—	PM1(1)	PM0(1)	DC1B1	DC1B0	CCP1M3	CCP1M2	CCP1M1	CCP1M0	0000
CCPR1L	0192	—	—	—	—	—	—	—	—	Capture/Compare/PWM1 Register Low Byte								0000
CCPR1H	0194	—	—	—	—	—	—	—	—	Capture/Compare/PWM1 Register High Byte								0000
ECCP1DEL(1)	0196	—	—	—	—	—	—	—	—	PRSEN	PDC6	PDC5	PDC4	PDC3	PDC2	PDC1	PDC0	0000
ECCP1AS(1)	0198	—	—	—	—	—	—	—	—	ECCPASE	ECCPAS2	ECCPAS1	ECCPAS0	PSSAC1	PSSAC0	PSSBD1	PSSBD0	0000
PSTR1CON(1)	019A	—	—	—	—	—	—	—	—	CMPL1	CMPL0	—	STRSYNC	STRD	STRC	STRB	STRA	0001
CCP2CON	019C	—	—	—	—	—	—	—	—	—	—	DC2B1	DC2B0	CCP2M3	CCP2M2	CCP2M1	CCP2M0	0000
CCPR2L	019E	—	—	—	—	—	—	—	—	Capture/Compare/PWM2 Register Low Byte								0000
CCPR2H	01A0	—	—	—	—	—	—	—	—	Capture/Compare/PWM2 Register High Byte								0000
CCP3CON(1)	01A8	—	—	—	—	—	—	—	—	—	—	DC3B1	DC3B0	CCP3M3	CCP3M2	CCP3M1	CCP3M0	0000
CCPR3L(1)	01AA	—	—	—	—	—	—	—	—	Capture/Compare/PWM3 Register Low Byte								0000
CCPR3H(1)	01AC	—	—	—	—	—	—	—	—	Capture/Compare/PWM3 Register High Byte								0000

Legend: — = unimplemented, read as '0'. Reset values are shown in hexadecimal.

Note 1: These bits and/or registers are unimplemented on PIC24FXXKL10X and PIC24FXXKL20X family devices; read as '0'.

TABLE 4-8: MSSP REGISTER MAP

File Name	Addr	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	All Resets
SSP1BUF	0200	—	—	—	—	—	—	—	—	MSSP1 Receive Buffer/Transmit Register								00xx
SSP1CON1	0202	—	—	—	—	—	—	—	—	WCOL	SSPOV	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0	0000
SSP1CON2	0204	—	—	—	—	—	—	—	—	GCEN	ACKSTAT	ACKDT	ACKEN	RCEN	PEN	RSEN	SEN	0000
SSP1CON3	0206	—	—	—	—	—	—	—	—	ACKTIM	PCIE	SCIE	BOEN	SDAHT	SBCDE	AHEN	DHEN	0000
SSP1STAT	0208	—	—	—	—	—	—	—	—	SMP	CKE	D/$\overline{A}$	P	S	R/$\overline{W}$	UA	BF	0000
SSP1ADD	020A	—	—	—	—	—	—	—	—	MSSP1 Address Register (I^2C™ Slave Mode) MSSP1 Baud Rate Reload Register (I^2C Master Mode)								0000
SSP1MSK	020C	—	—	—	—	—	—	—	—	MSSP1 Address Mask Register (I^2C Slave Mode)								00FF
SSP2BUF(1)	0210	—	—	—	—	—	—	—	—	MSSP2 Receive Buffer/Transmit Register								00xx
SSP2CON1(1)	0212	—	—	—	—	—	—	—	—	WCOL	SSPOV	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0	0000
SSP2CON2(1)	0214	—	—	—	—	—	—	—	—	GCEN	ACKSTAT	ACKDT	ACKEN	RCEN	PEN	RSEN	SEN	0000
SSP2CON3(1)	0216	—	—	—	—	—	—	—	—	ACKTIM	PCIE	SCIE	BOEN	SDAHT	SBCDE	AHEN	DHEN	0000
SSP2STAT(1)	0218	—	—	—	—	—	—	—	—	SMP	CKE	D/$\overline{A}$	P	S	R/$\overline{W}$	UA	BF	0000
SSP2ADD(1)	021A	—	—	—	—	—	—	—	—	MSSP2 Address Register (I^2C Slave Mode) MSSP2 Baud Rate Reload Register (I^2C Master Mode)								0000
SSP2MSK(1)	021C	—	—	—	—	—	—	—	—	MSSP2 Address Mask Register (I^2C Slave Mode)								00FF

Legend: — = unimplemented, read as '0'. Reset values are shown in hexadecimal.

Note 1: These bits and/or registers are unimplemented on PIC24FXXKL10X and PIC24FXXKL20X family devices; read as '0'.

TABLE 4-9: UART REGISTER MAP

File Name	Addr	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	All Resets
U1MODE	0220	UARTEN	—	USIDL	IREN	RTSMD	—	UEN1	UEN0	WAKE	LPBACK	ABAUD	RXINV	BRGH	PDSEL1	PDSEL0	STSEL	0000
U1STA	0222	UTXISEL1	UTXINV	UTXISEL0	—	UTXBRK	UTXEN	UTXBF	TRMT	URXISEL1	URXISEL0	ADDEN	RIDLE	PERR	FERR	OERR	URXDA	0110
U1TXREG	0224	—	—	—	—	—	—	—	UART1 Transmit Register									xxxx
U1RXREG	0226	—	—	—	—	—	—	—	UART1 Receive Register									0000
U1BRG	0228	Baud Rate Generator Prescaler Register																0000
U2MODE	0230	UARTEN	—	USIDL	IREN	RTSMD	—	UEN1	UEN0	WAKE	LPBACK	ABAUD	RXINV	BRGH	PDSEL1	PDSEL0	STSEL	0000
U2STA	0232	UTXISEL1	UTXINV	UTXISEL0	—	UTXBRK	UTXEN	UTXBF	TRMT	URXISEL1	URXISEL0	ADDEN	RIDLE	PERR	FERR	OERR	URXDA	0110
U2TXREG	0234	—	—	—	—	—	—	—	UART2 Transmit Register									xxxx
U2RXREG	0236	—	—	—	—	—	—	—	UART2 Receive Register									0000
U2BRG	0238	Baud Rate Generator Prescaler Register																0000

Legend: — = unimplemented, read as '0'. Reset values are shown in hexadecimal.

4.3.3 READING DATA FROM PROGRAM MEMORY USING PROGRAM SPACE VISIBILITY

The upper 32 Kbytes of data space may optionally be mapped into a 16K word page of the program space. This provides transparent access of stored constant data from the data space without the need to use special instructions (i.e., TBLRDL/H).

Program space access through the data space occurs if the MSb of the data space EA is '1' and PSV is enabled by setting the PSV bit in the CPU Control (CORCON<2>) register. The location of the program memory space to be mapped into the data space is determined by the Program Space Visibility Page Address (PSVPAG) register. This 8-bit register defines any one of 256 possible pages of 16K words in program space. In effect, PSVPAG functions as the upper 8 bits of the program memory address, with 15 bits of the EA functioning as the lower bits.

By incrementing the PC by 2 for each program memory word, the lower 15 bits of data space addresses directly map to the lower 15 bits in the corresponding program space addresses.

Data reads from this area add an additional cycle to the instruction being executed, since two program memory fetches are required.

Although each data space address, 8000h and higher, maps directly into a corresponding program memory address (see Figure 4-7), only the lower 16 bits of the 24-bit program word are used to contain the data. The upper 8 bits of any program space location, used as data, should be programmed with '1111 1111' or '0000 0000' to force a NOP. This prevents possible issues should the area of code ever be accidentally executed.

Note:	PSV access is temporarily disabled during Table Reads/Writes.

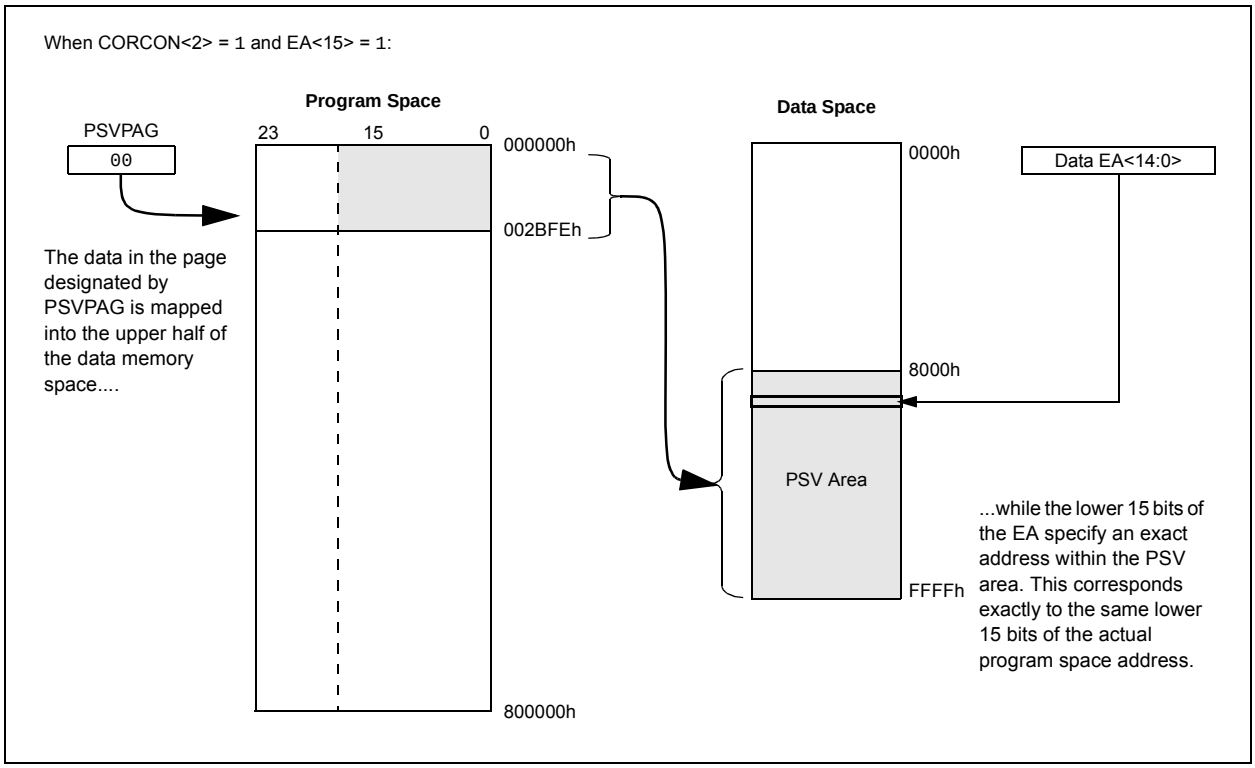
For operations that use PSV and are executed outside of a REPEAT loop, the MOV and MOV.D instructions will require one instruction cycle, in addition to the specified execution time. All other instructions will require two instruction cycles in addition to the specified execution time.

For operations that use PSV, which are executed inside a REPEAT loop, there will be some instances that require two instruction cycles, in addition to the specified execution time of the instruction:

- Execution in the first iteration
- Execution in the last iteration
- Execution prior to exiting the loop due to an interrupt
- Execution upon re-entering the loop after an interrupt is serviced

Any other iteration of the REPEAT loop will allow the instruction accessing data, using PSV, to execute in a single cycle.

FIGURE 4-7: PROGRAM SPACE VISIBILITY OPERATION



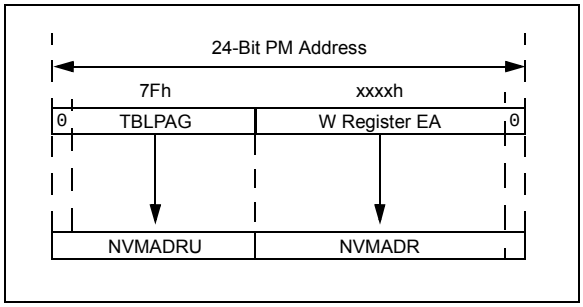
6.3 NVM Address Register

As with Flash program memory, the NVM Address Registers, NVMADRU and NVMADR, form the 24-bit Effective Address (EA) of the selected row or word for data EEPROM operations. The NVMADRU register is used to hold the upper 8 bits of the EA, while the NVMADR register is used to hold the lower 16 bits of the EA. These registers are not mapped into the Special Function Register (SFR) space; instead, they directly capture the EA<23:0> of the last Table Write instruction that has been executed and selects the data EEPROM row to erase. Figure 6-1 depicts the program memory EA that is formed for programming and erase operations.

Like program memory operations, the Least Significant bit (LSb) of NVMADR is restricted to even addresses. This is because any given address in the data EEPROM space consists of only the lower word of the program memory width; the upper word, including the uppermost "phantom byte", is unavailable. This means that the LSb of a data EEPROM address will always be '0'.

Similarly, the Most Significant bit (MSb) of NVMADRU is always '0', since all addresses lie in the user program space.

FIGURE 6-1: DATA EEPROM ADDRESSING WITH TBLPAG AND NVM ADDRESS REGISTERS



6.4 Data EEPROM Operations

The EEPROM block is accessed using Table Read and Table Write operations, similar to those used for program memory. The TBLWTH and TBLRDH instructions are not required for data EEPROM operations since the memory is only 16 bits wide (data on the lower address is valid only). The following programming operations can be performed on the data EEPROM:

- Erase one, four or eight words
- Bulk erase the entire data EEPROM
- Write one word
- Read one word

Note: Unexpected results will be obtained if the user attempts to read the EEPROM while a programming or erase operation is underway.

The C30 C compiler includes library procedures to automatically perform the Table Read and Table Write operations, manage the Table Pointer and write buffers, and unlock and initiate memory write sequences. This eliminates the need to create assembler macros or time critical routines in C for each application.

The library procedures are used in the code examples detailed in the following sections. General descriptions of each process are provided for users who are not using the C30 compiler libraries.

REGISTER 8-24: IPC7: INTERRUPT PRIORITY CONTROL REGISTER 7

U-0	R/W-1	R/W-0	R/W-0	U-0	R/W-1	R/W-0	R/W-0
—	U2TXIP2(1)	U2TXIP1(1)	U2TXIP0(1)	—	U2RXIP2(1)	U2RXIP1(1)	U2RXIP0(1)
bit 15							bit 8

U-0	R/W-1	R/W-0	R/W-0	U-0	U-0	U-0	U-0
—	INT2IP2	INT2IP1	INT2IP0	—	—	—	—
bit 7							bit 0

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

bit 15 **Unimplemented:** Read as '0'

bit 14-12 **U2TXIP<2:0>:** UART2 Transmitter Interrupt Priority bits(1)
111 = Interrupt is Priority 7 (highest priority interrupt)
•
•
•
001 = Interrupt is Priority 1
000 = Interrupt source is disabled

bit 11 **Unimplemented:** Read as '0'

bit 10-8 **U2RXIP<2:0>:** UART2 Receiver Interrupt Priority bits(1)
111 = Interrupt is Priority 7 (highest priority interrupt)
•
•
•
001 = Interrupt is Priority 1
000 = Interrupt source is disabled

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

bit 6-4 **INT2IP<2:0>:** External Interrupt 2 Priority bits
111 = Interrupt is Priority 7 (highest priority interrupt)
•
•
•
001 = Interrupt is Priority 1
000 = Interrupt source is disabled

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

Note 1: These bits are unimplemented on PIC24FXXKL10X and PIC24FXXKL20X devices.

REGISTER 8-25: IPC9: INTERRUPT PRIORITY CONTROL REGISTER 9

U-0	U-0	U-0	U-0	U-0	U-0	U-0	U-0
—	—	—	—	—	—	—	—
bit 15							bit 8

U-0	R/W-1	R/W-0	R/W-0	U-0	U-0	U-0	U-0
—	T3GIP2	T3GIP1	T3GIP0	—	—	—	—
bit 7							bit 0

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

bit 15-7 **Unimplemented:** Read as ‘0’

bit 6-4 **T3GIP<2:0>:** Timer3 External Gate Interrupt Priority bits
111 = Interrupt is Priority 7 (highest priority interrupt)
•
•
•
001 = Interrupt is Priority 1
000 = Interrupt source is disabled

bit 3-0 **Unimplemented:** Read as ‘0’

10.0 POWER-SAVING FEATURES

Note: This data sheet summarizes the features of this group of PIC24F devices. It is not intended to be a comprehensive reference source. For more information on Power-Saving Features, refer to the *"dsPIC33/PIC24 Family Reference Manual"*, **"Power-Saving Features with Deep Sleep"** (DS39727).

The PIC24F16KL402 family of devices provides the ability to manage power consumption by selectively managing clocking to the CPU and the peripherals. In general, a lower clock frequency and a reduction in the number of circuits being clocked constitutes lower consumed power. All PIC24F devices manage power consumption using several strategies:

- Clock frequency
- Instruction-based Idle and Sleep modes
- Hardware-based periodic wake-up from Sleep
- Software Controlled Doze mode
- Selective peripheral control in software

Combinations of these methods can be used to selectively tailor an application's power consumption, while still maintaining critical application features, such as timing-sensitive communications.

10.1 Clock Frequency and Clock Switching

PIC24F devices allow for a wide range of clock frequencies to be selected under application control. If the system clock configuration is not locked, users can choose low-power or high-precision oscillators by simply changing the NOSCx bits. The process of changing a system clock during operation, as well as limitations to the process, are discussed in more detail in **Section 9.0 "Oscillator Configuration"**.

10.2 Instruction-Based Power-Saving Modes

PIC24F devices have two special power-saving modes that are entered through the execution of a special PWRSAV instruction. Sleep mode stops clock operation and halts all code execution; Idle mode halts the CPU and code execution, but allows peripheral modules to continue operation.

The assembly syntax of the PWRSAV instruction is shown in Example 10-1.

Note: SLEEP_MODE and IDLE_MODE are constants defined in the assembler include file for the selected device.

Sleep and Idle modes can be exited as a result of an enabled interrupt, WDT time-out or a device Reset. When the device exits these modes, it is said to "wake-up".

EXAMPLE 10-1: PWRSAV INSTRUCTION SYNTAX

```
PWRSAV     #SLEEP_MODE    ; Put the device into SLEEP mode
PWRSAV     #IDLE_MODE     ; Put the device into IDLE mode
```

10.4 Doze Mode

Generally, changing clock speed and invoking one of the power-saving modes are the preferred strategies for reducing power consumption. There may be circumstances, however, where this is not practical. For example, it may be necessary for an application to maintain uninterrupted, synchronous communication, even while it is doing nothing else. Reducing system clock speed may introduce communication errors, while using a power-saving mode may stop communications completely.

Doze mode is a simple and effective alternative method to reduce power consumption while the device is still executing code. In this mode, the system clock continues to operate from the same source and at the same speed. Peripheral modules continue to be clocked at the same speed, while the CPU clock speed is reduced. Synchronization between the two clock domains is maintained, allowing the peripherals to access the SFRs while the CPU executes code at a slower rate.

Doze mode is enabled by setting the DOZEN bit (CLKDIV<11>). The ratio between peripheral and core clock speed is determined by the DOZE<2:0> bits (CLKDIV<14:12>). There are eight possible configurations, from 1:1 to 1:128, with 1:1 being the default.

It is also possible to use Doze mode to selectively reduce power consumption in event driven applications. This allows clock-sensitive functions, such as synchronous communications, to continue without interruption. Meanwhile, the CPU Idles, waiting for something to invoke an interrupt routine. Enabling the automatic return to full-speed CPU operation on interrupts is enabled by setting the ROI bit (CLKDIV<15>). By default, interrupt events have no effect on Doze mode operation.

10.5 Selective Peripheral Module Control

Idle and Doze modes allow users to substantially reduce power consumption by slowing or stopping the CPU clock. Even so, peripheral modules still remain clocked and thus, consume power. There may be cases where the application needs what these modes do not provide: the allocation of power resources to CPU processing, with minimal power consumption from the peripherals.

PIC24F devices address this requirement by allowing peripheral modules to be selectively disabled, reducing or eliminating their power consumption. This can be done with two control bits:

- The Peripheral Enable bit, generically named, “XXXEN”, located in the module’s main control SFR.
- The Peripheral Module Disable (PMD) bit, generically named, “XXXMD”, located in one of the PMD Control registers.

Both bits have similar functions in enabling or disabling its associated module. Setting the PMD bit for a module disables all clock sources to that module, reducing its power consumption to an absolute minimum. In this state, the control and status registers associated with the peripheral will also be disabled, so writes to those registers will have no effect, and read values will be invalid. Many peripheral modules have a corresponding PMD bit.

In contrast, disabling a module by clearing its XXXEN bit, disables its functionality, but leaves its registers available to be read and written to. Power consumption is reduced, but not by as much as when the PMD bits are used.

To achieve more selective power savings, peripheral modules can also be selectively disabled when the device enters Idle mode. This is done through the control bit of the generic name format, “XXXIDL”. By default, all modules that can operate during Idle mode will do so. Using the disable on Idle feature disables the module while in Idle mode, allowing further reduction of power consumption during Idle mode. This enhances power savings for extremely critical power applications.

REGISTER 14-1: T3CON: TIMER3 CONTROL REGISTER

U-0	U-0	U-0	U-0	U-0	U-0	U-0	U-0
—	—	—	—	—	—	—	—
bit 15							bit 8

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	U-0	R/W-0
TMR3CS1	TMR3CS0	T3CKPS1	T3CKPS0	T3OSCEN	$\overline{\text{T3SYNC}}$	—	TMR3ON
bit 7							bit 0

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

bit 15-8 **Unimplemented:** Read as '0'

bit 7-6 **TMR3CS<1:0>:** Timer3 Clock Source Select bits
11 = Low-Power RC Oscillator (LPRC)
10 = External clock source (selected by T3CON<3>)
01 = Instruction clock (FOSC/2)
00 = System clock (FOSC)**(1)**

bit 5-4 **T3CKPS<1:0>**: Timer3 Input Clock Prescale Select bits
11 = 1:8 Prescale value
10 = 1:4 Prescale value
01 = 1:2 Prescale value
00 = 1:1 Prescale value

bit 3 **T3OSCEN**: Timer3 Oscillator Enable bit
1 = SOSC (Secondary Oscillator) is used as a clock source
0 = T3CK digital input pin is used as a clock source

bit 2 **$\overline{\text{T3SYNC}}$:** Timer3 External Clock Input Synchronization Control bit
When TMR3CS<1:0> = 1x:
1 = Does not synchronize the external clock input
0 = Synchronizes the external clock input**(2)**
When TMR3CS<1:0> = 0x:
This bit is ignored; Timer3 uses the internal clock.

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

bit 0 **TMR3ON:** Timer3 On bit
1 = Enables Timer3
0 = Stops Timer3

Note 1: The FOSC clock source should not be selected if the timer will be used with the ECCP capture or compare features.
2: This option must be selected when the timer will be used with ECCP/CCP.

REGISTER 16-4: ECCP1DEL: ECCP1 ENHANCED PWM CONTROL REGISTER(1)

U-0	U-0	U-0	U-0	U-0	U-0	U-0	U-0
—	—	—	—	—	—	—	—
bit 15							bit 8

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
PRSEN	PDC6	PDC5	PDC4	PDC3	PDC2	PDC1	PDC0
bit 7							bit 0

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

bit 15-8 **Unimplemented:** Read as '0'

bit 7 **PRSEN:** PWM Restart Enable bit

1 = Upon auto-shutdown, the ECCPASE bit clears automatically once the shutdown event goes away; the PWM restarts automatically

0 = Upon auto-shutdown, ECCPASE must be cleared by software to restart the PWM

bit 6-0 **PDC<6:0>:** PWM Delay Count bits

PDCn = Number of FCY (FOSC/2) cycles between the scheduled time when a PWM signal **should** transition active and the **actual** time it transitions active.

Note 1: This register is implemented only on PIC24FXXKL40X/30X devices.

18.2 Transmitting in 8-Bit Data Mode

1. Set up the UART:
   a) Write appropriate values for data, parity and Stop bits.
   b) Write appropriate baud rate value to the UxBRG register.
   c) Set up transmit and receive interrupt enable and priority bits.
2. Enable the UART.
3. Set the UTXEN bit (causes a transmit interrupt, two cycles after being set).
4. Write data byte to lower byte of UxTXREG word. The value will be immediately transferred to the Transmit Shift Register (TSR) and the serial bit stream will start shifting out with the next rising edge of the baud clock.
5. Alternately, the data byte may be transferred while UTXEN = 0 and then, the user may set UTXEN. This will cause the serial bit stream to begin immediately, because the baud clock will start from a cleared state.
6. A transmit interrupt will be generated as per interrupt control bit, UTXISELx.

18.3 Transmitting in 9-Bit Data Mode

1. Set up the UART (as described in **Section 18.2 "Transmitting in 8-Bit Data Mode"**).
2. Enable the UART.
3. Set the UTXEN bit (causes a transmit interrupt, two cycles after being set).
4. Write UxTXREG as a 16-bit value only.
5. A word write to UxTXREG triggers the transfer of the 9-bit data to the TSR. The serial bit stream will start shifting out with the first rising edge of the baud clock.
6. A transmit interrupt will be generated as per the setting of control bit, UTXISELx.

18.4 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.

1. Configure the UART for the desired mode.
2. Set UTXEN and UTXBRK – sets up the Break character.
3. Load the UxTXREG with a dummy character to initiate transmission (value is ignored).
4. Write '55h' to UxTXREG – loads the Sync character into the transmit FIFO.
5. After the Break has been sent, the UTXBRK bit is reset by hardware. The Sync character now transmits.

18.5 Receiving in 8-Bit or 9-Bit Data Mode

1. Set up the UART (as described in **Section 18.2 "Transmitting in 8-Bit Data Mode"**).
2. Enable the UART.
3. A receive interrupt will be generated when one or more data characters have been received as per interrupt control bit, URXISELx.
4. Read the OERR bit to determine if an overrun error has occurred. The OERR bit must be reset in software.
5. Read UxRXREG.

The act of reading the UxRXREG character will move the next character to the top of the receive FIFO, including a new set of PERR and FERR values.

18.6 Operation of $\overline{\text{UxCTS}}$ and $\overline{\text{UxRTS}}$ Control Pins

UARTx Clear-to-Send ($\overline{\text{UxCTS}}$) and Request-to-Send ($\overline{\text{UxRTS}}$) are the two hardware-controlled pins that are associated with the UART module. These two pins allow the UART to operate in Simplex and Flow Control modes. They are implemented to control the transmission and reception between the Data Terminal Equipment (DTE). The UEN<1:0> bits in the UxMODE register configure these pins.

18.7 Infrared Support

The UART module provides two types of infrared UART support: one is the IrDA clock output to support an external IrDA encoder and decoder device (legacy module support), and the other is the full implementation of the IrDA encoder and decoder.

As the IrDA modes require a 16x baud clock, they will only work when the BRGH bit (UxMODE<3>) is '0'.

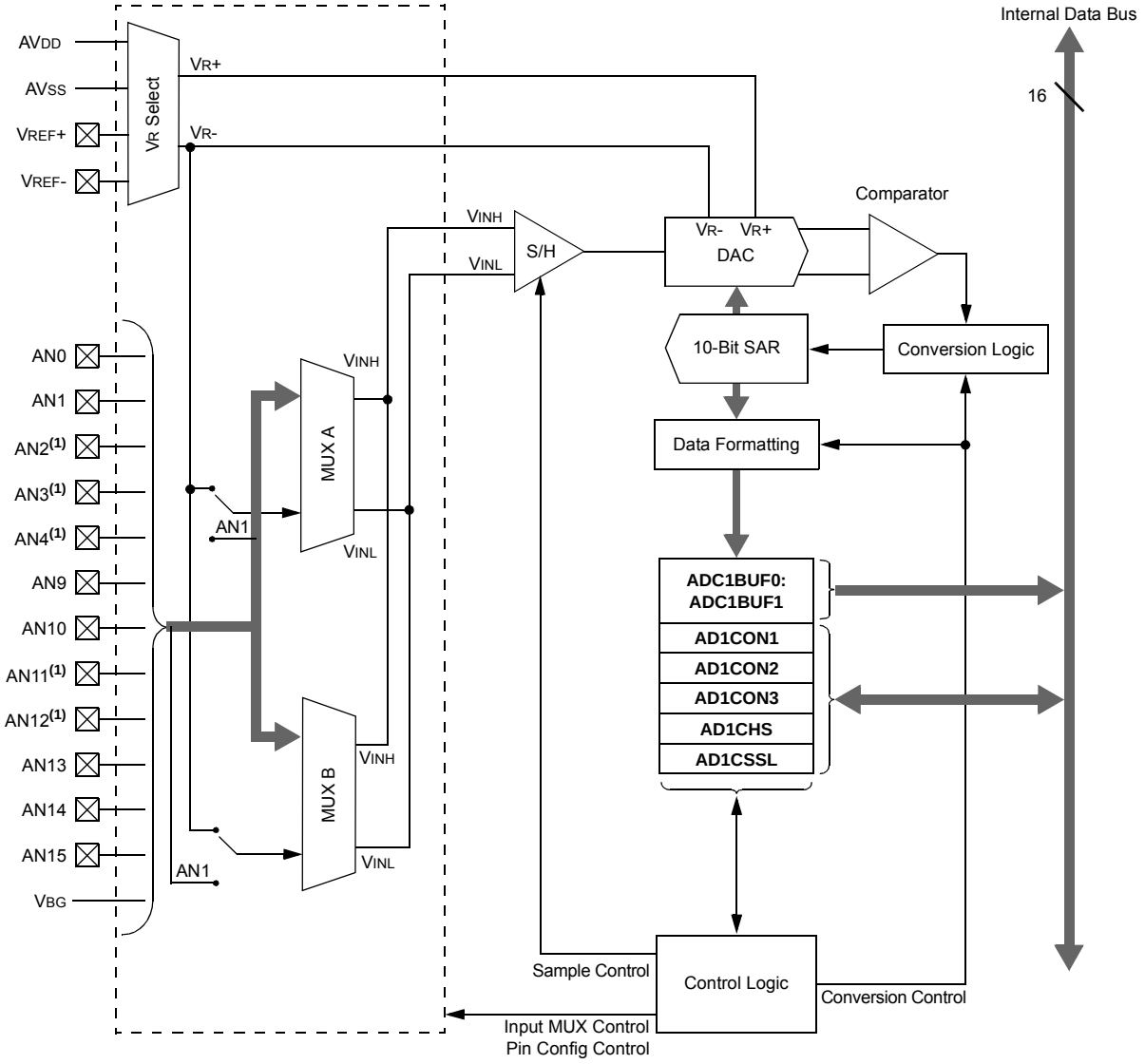
18.7.1 EXTERNAL IrDA SUPPORT – IrDA CLOCK OUTPUT

To support external IrDA encoder and decoder devices, the UxBCLK pin (same as the $\overline{\text{UxRTS}}$ pin) can be configured to generate the 16x baud clock. When UEN<1:0> = 11, the UxBCLK pin will output the 16x baud clock if the UART module is enabled; it can be used to support the IrDA codec chip.

18.7.2 BUILT-IN IrDA ENCODER AND DECODER

The UART has full implementation of the IrDA encoder and decoder as part of the UART module. The built-in IrDA encoder and decoder functionality is enabled using the IREN bit (UxMODE<12>). When enabled (IREN = 1), the receive pin (UxRX) acts as the input from the infrared receiver. The transmit pin (UxTX) acts as the output to the infrared transmitter.

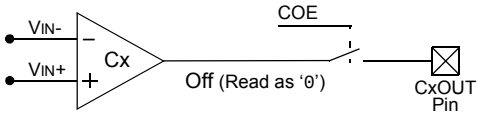
FIGURE 19-1: 10-BIT HIGH-SPEED A/D CONVERTER BLOCK DIAGRAM



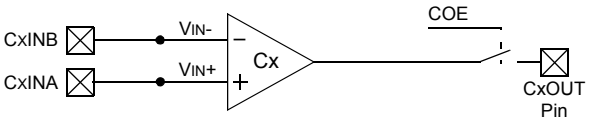
Note 1: Unimplemented in 14-pin devices.

FIGURE 20-2: INDIVIDUAL COMPARATOR CONFIGURATIONS

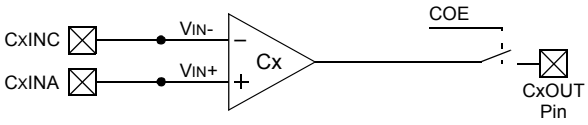
Comparator Off
CON = 0, CREF = x, CCH<1:0> = xx



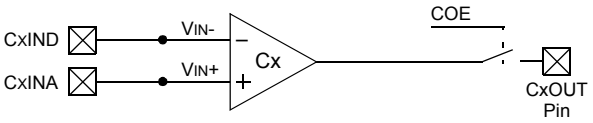
Comparator CxINB > CxINA Compare
CON = 1, CREF = 0, CCH<1:0> = 00



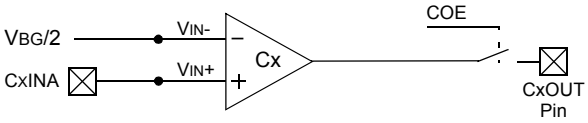
Comparator CxINC > CxINA Compare(1)
CON = 1, CREF = 0, CCH<1:0> = 01



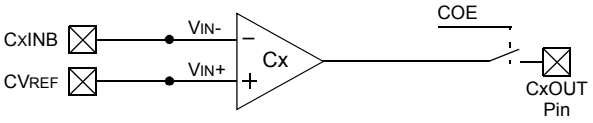
Comparator CxIND > CxINA Compare(1)
CON = 1, CREF = 0, CCH<1:0> = 10



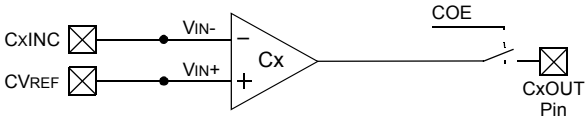
Comparator VBG > CxINA Compare
CON = 1, CREF = 0, CCH<1:0> = 11



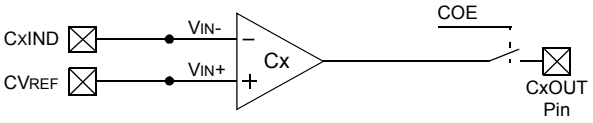
Comparator CxINB > CVREF Compare
CON = 1, CREF = 1, CCH<1:0> = 00



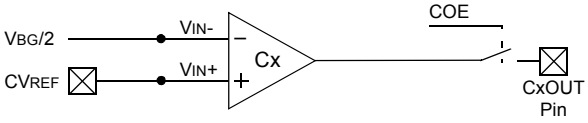
Comparator CxINC > CVREF Compare(1)
CON = 1, CREF = 1, CCH<1:0> = 01



Comparator CxIND > CVREF Compare(1)
CON = 1, CREF = 1, CCH<1:0> = 10



Comparator VBG > CVREF Compare
CON = 1, CREF = 1, CCH<1:0> = 11



Note 1: This configuration is unavailable on 14-pin (PIC24FXXKL100/200) devices.

REGISTER 23-3: FOSCSEL: OSCILLATOR SELECTION CONFIGURATION REGISTER

R/P-1	R/P-1	R/P-1	U-0	U-0	R/P-0	R/P-0	R/P-1
IESO	LPRCSEL	SOSCSRC	—	—	FNOSC2	FNOSC1	FNOSC0
bit 7							bit 0

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

bit 7 **IESO:** Internal External Switchover bit
1 = Internal External Switchover mode is enabled (Two-Speed Start-up is enabled)
0 = Internal External Switchover mode is disabled (Two-Speed Start-up is disabled)

bit 6 **LPRCSEL:** Internal LPRC Oscillator Power Select bit
1 = High-Power/High-Accuracy mode
0 = Low-Power/Low-Accuracy mode

bit 5 **SOSCSRC:** Secondary Oscillator Clock Source Configuration bit
1 = SOSC analog crystal function is available on the SOSCI/SOSCO pins
0 = SOSC crystal is disabled; digital SCLKI function is selected on the SOSCO pin

bit 4-3 **Unimplemented:** Read as '0'

bit 2-0 **FNOSC<2:0>:** Oscillator Selection bits
111 = 8 MHz FRC Oscillator with Divide-by-N (FRCDIV)
110 = 500 kHz Low-Power FRC Oscillator with Divide-by-N (LPFRCDIV)
101 = Low-Power RC Oscillator (LPRC)
100 = Secondary Oscillator (SOSC)
011 = Primary Oscillator with PLL module (HS+PLL, EC+PLL)
010 = Primary Oscillator (XT, HS, EC)
001 = 8 MHz FRC Oscillator with Divide-by-N with PLL module (FRCDIV+PLL)
000 = 8 MHz FRC Oscillator (FRC)

24.11 Demonstration/Development Boards, Evaluation Kits and Starter Kits

A wide variety of demonstration, development and evaluation boards for various PIC MCUs and dsPIC DSCs allows quick application development on fully functional systems. Most boards include prototyping areas for adding custom circuitry and provide application firmware and source code for examination and modification.

The boards support a variety of features, including LEDs, temperature sensors, switches, speakers, RS-232 interfaces, LCD displays, potentiometers and additional EEPROM memory.

The demonstration and development boards can be used in teaching environments, for prototyping custom circuits and for learning about various microcontroller applications.

In addition to the PICDEM™ and dsPICDEM™ demonstration/development board series of circuits, Microchip has a line of evaluation kits and demonstration software for analog filter design, KEELOQ® security ICs, CAN, IrDA®, PowerSmart battery management, SEEVAL® evaluation system, Sigma-Delta ADC, flow rate sensing, plus many more.

Also available are starter kits that contain everything needed to experience the specified device. This usually includes a single application and debug capability, all on one board.

Check the Microchip web page (www.microchip.com) for the complete list of demonstration, development and evaluation kits.

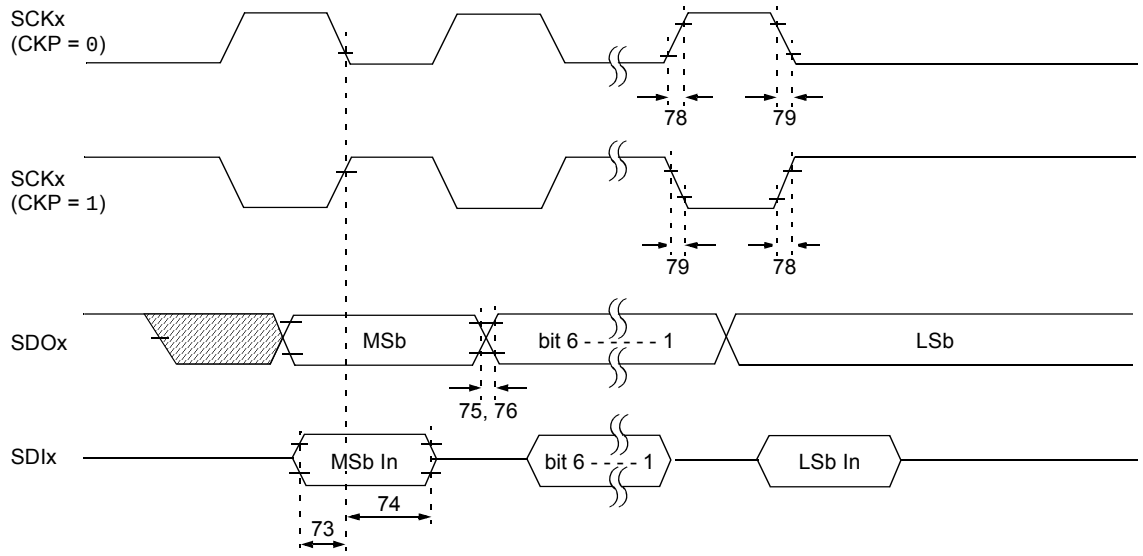
24.12 Third-Party Development Tools

Microchip also offers a great collection of tools from third-party vendors. These tools are carefully selected to offer good value and unique functionality.

- Device Programmers and Gang Programmers from companies, such as SoftLog and CCS
- Software Tools from companies, such as Gimpel and Trace Systems
- Protocol Analyzers from companies, such as Saleae and Total Phase
- Demonstration Boards from companies, such as MikroElektronika, Digilent® and Olimex
- Embedded Ethernet Solutions from companies, such as EZ Web Lynx, WIZnet and IPLogika®

NOTES:

FIGURE 26-7: EXAMPLE SPI MASTER MODE TIMING (CKE = 0)



Note: Refer to Figure 26-3 for load conditions.

TABLE 26-27: EXAMPLE SPI MODE REQUIREMENTS (MASTER MODE, CKE = 0)

Param No.	Symbol	Characteristic	Min	Max	Units	Conditions
73	TDIV2SCH, TDIV2SCL	Setup Time of SDIx Data Input to SCKx Edge	20	—	ns	
74	TSCH2DIL, TSCL2DIL	Hold Time of SDIx Data Input to SCKx Edge	40	—	ns	
75	TDOR	SDOx Data Output Rise Time	—	25	ns	
76	TDOF	SDOx Data Output Fall Time	—	25	ns	
78	TSCR	SCKx Output Rise Time (Master mode)	—	25	ns	
79	TSCF	SCKx Output Fall Time (Master mode)	—	25	ns	
	FSCK	SCKx Frequency	—	10	MHz	

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: 978-1-62077-620-9

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.