



Welcome to [E-XFL.COM](https://www.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	dsPIC
Core Size	16-Bit
Speed	40 MIPS
Connectivity	CANbus, I <sup>2</sup> C, IrDA, LINbus, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, DMA, Motor Control PWM, POR, PWM, QEI, WDT
Number of I/O	85
Program Memory Size	256KB (256K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	30K x 8
Voltage - Supply (Vcc/Vdd)	3V ~ 3.6V
Data Converters	A/D 24x10/12b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	100-TQFP
Supplier Device Package	100-TQFP (12x12)
Purchase URL	<a href="https://www.e-xfl.com/product-detail/microchip-technology/dspic33fj256mc710t-i-pt">https://www.e-xfl.com/product-detail/microchip-technology/dspic33fj256mc710t-i-pt</a>

---

**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.

---

Information contained in this publication regarding device applications and the like is provided only for your convenience and may be superseded by updates. It is your responsibility to ensure that your application meets with your specifications. MICROCHIP MAKES NO REPRESENTATIONS OR WARRANTIES OF ANY KIND WHETHER EXPRESS OR IMPLIED, WRITTEN OR ORAL, STATUTORY OR OTHERWISE, RELATED TO THE INFORMATION, INCLUDING BUT NOT LIMITED TO ITS CONDITION, QUALITY, PERFORMANCE, MERCHANTABILITY OR FITNESS FOR PURPOSE. Microchip disclaims all liability arising from this information and its use. Use of Microchip devices in life support and/or safety applications is entirely at the buyer's risk, and the buyer agrees to defend, indemnify and hold harmless Microchip from any and all damages, claims, suits, or expenses resulting from such use. No licenses are conveyed, implicitly or otherwise, under any Microchip intellectual property rights.

#### **Trademarks**

The Microchip name and logo, the Microchip logo, Accuron, dsPIC, KEELOQ, KEELOQ logo, MPLAB, PIC, PICmicro, PICSTART, rfPIC, SmartShunt and UNI/O are registered trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.


FilterLab, Linear Active Thermistor, MXDEV, MXLAB, SEEVAL, SmartSensor and The Embedded Control Solutions Company are registered trademarks of Microchip Technology Incorporated in the U.S.A.

Analog-for-the-Digital Age, Application Maestro, CodeGuard, dsPICDEM, dsPICDEM.net, dsPICworks, dsSPEAK, ECAN, ECONOMONITOR, FanSense, In-Circuit Serial Programming, ICSP, ICEPIC, Mindi, MiWi, MPASM, MPLAB Certified logo, MPLIB, MPLINK, mTouch, nanoWatt XLP, PICKit, PICDEM, PICDEM.net, PICTail, PIC<sup>32</sup> logo, PowerCal, PowerInfo, PowerMate, PowerTool, REAL ICE, rfLAB, Select Mode, Total Endurance, TSHARC, WiperLock and ZENA are trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

SQTP is a service mark of Microchip Technology Incorporated in the U.S.A.

All other trademarks mentioned herein are property of their respective companies.

© 2009, Microchip Technology Incorporated, Printed in the U.S.A., All Rights Reserved.

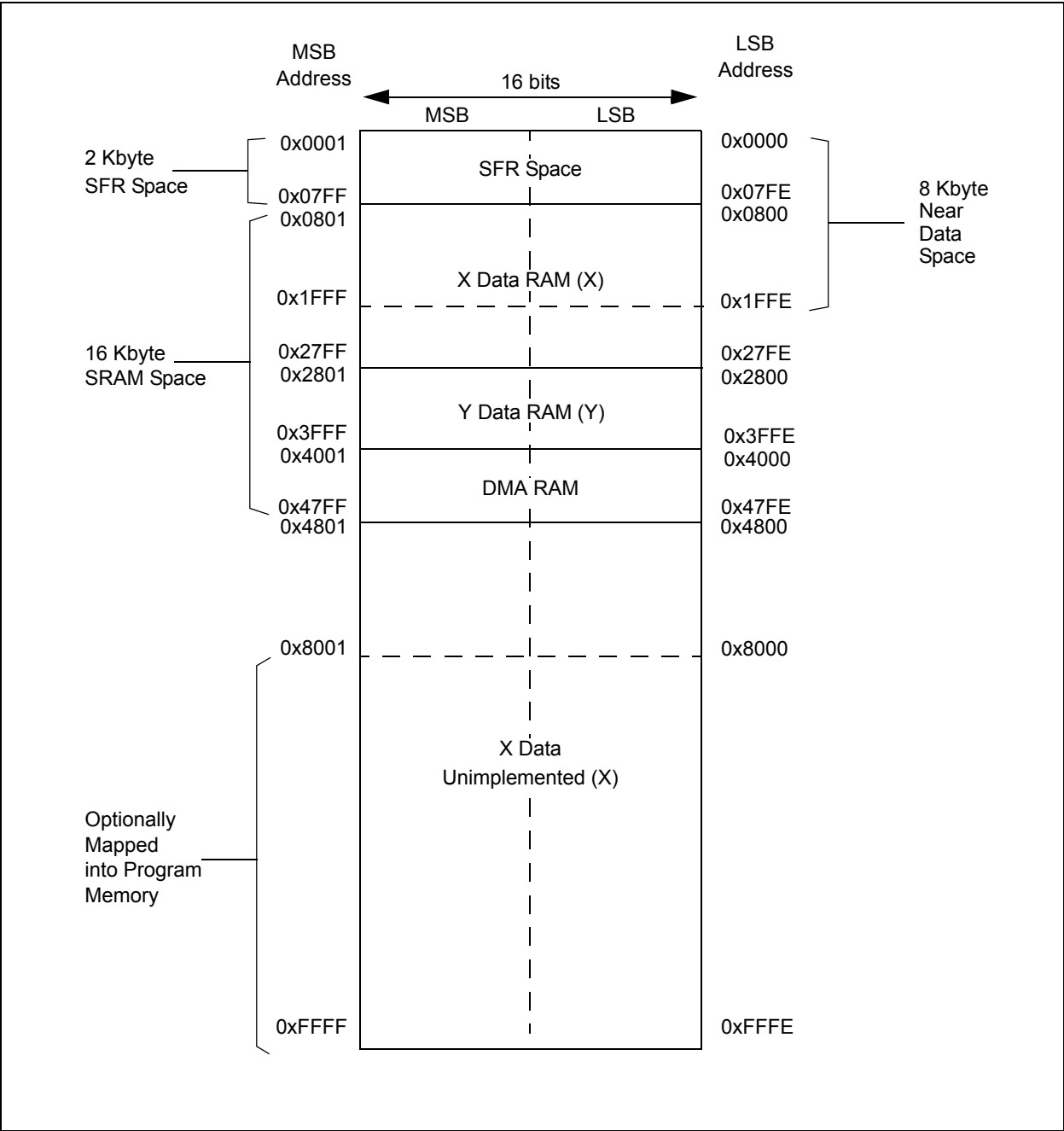
 Printed on recycled paper.

**QUALITY MANAGEMENT SYSTEM**  
**CERTIFIED BY DNV**  
**== ISO/TS 16949:2002 ==**

*Microchip received ISO/TS-16949:2002 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.*

# dsPIC33FJXXMCX06/X08/X10

**FIGURE 4-4: DATA MEMORY MAP FOR dsPIC33FJXXMCX06/X08/X10 DEVICES WITH 16 KB RAM**



**TABLE 4-13: UART1 REGISTER MAP**

SFR Name	SFR 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	RTSMO	—	UEN1	UEN0	WAKE	LPBACK	ABAUD	URXINV	BRGH	PDSEL<1:0>		STSEL	0000
U1STA	0222	UTXISEL1	UTXINV	UTXISEL0	—	UTXBRK	UTXEN	UTXBF	TRMT	URXISEL<1:0>		ADDEN	RIDLE	PERR	FERR	OERR	URXDA	0110
U1TXREG	0224	—	—	—	—	—	—	—	UART Transmit Register									xxxxx
U1RXREG	0226	—	—	—	—	—	—	—	UART Receive Register									0000
U1BRG	0228	Baud Rate Generator Prescaler																0000

**Legend:** x = unknown value on Reset, — = unimplemented, read as '0'. Reset values are shown in hexadecimal.

**TABLE 4-14: UART2 REGISTER MAP**

SFR Name	SFR 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
U2MODE	0230	UARTEN	—	USIDL	IREN	RTSMD	—	UEN1	UEN0	WAKE	LPBACK	ABAUD	URXINV	BRGH	PDSEL<1:0>		STSEL	0000
U2STA	0232	UTXISEL1	UTXINV	UTXISEL0	—	UTXBRK	UTXEN	UTXBF	TRMT	URXISEL<1:0>		ADDEN	RIDLE	PERR	FERR	OERR	URXDA	0110
U2TXREG	0234	—	—	—	—	—	—	—	UART Transmit Register									xxxx
U2RXREG	0236	—	—	—	—	—	—	—	UART Receive Register									0000
U2BRG	0238	Baud Rate Generator Prescaler																0000

**Legend:** x = unknown value on Reset, — = unimplemented, read as '0'. Reset values are shown in hexadecimal.

**TABLE 4-15: SPI1 REGISTER MAP**

SFR Name	SFR 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
SPI1STAT	0240	SPIEN	—	SPISIDL	—	—	—	—	—	—	SPROV	—	—	—	—	SPITBF	SPIRBF	0000
SPI1CON1	0242	—	—	—	DISSCK	DISSDO	MODE16	SMP	CKE	SSEN	CKP	MSTEN	SPRE<2:0>		PPRE<1:0>			0000
SPI1CON2	0244	FRMEN	SPIFSD	FRMPOL	—	—	—	—	—	—	—	—	—	—	—	FRMDLY	—	0000
SPI1BUF	0248	SPI1 Transmit and Receive Buffer Register																0000

**Legend:** x = unknown value on Reset, — = unimplemented, read as '0'. Reset values are shown in hexadecimal.

**TABLE 4-16: SPI2 REGISTER MAP**

SFR Name	SFR 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
SPI2STAT	0260	SPIEN	—	SPISIDL	—	—	—	—	—	—	SPROV	—	—	—	—	SPITBF	SPIRBF	0000
SPI2CON1	0262	—	—	—	DISSCK	DISSDO	MODE16	SMP	CKE	SSEN	CKP	MSTEN	SPRE<2:0>		PPRE<1:0>			0000
SPI2CON2	0264	FRMEN	SPIFSD	FRMPOL	—	—	—	—	—	—	—	—	—	—	—	FRMDLY	—	0000
SPI2BUF	0268	SPI2 Transmit and Receive Buffer Register																0000

**Legend:** x = unknown value on Reset, — = unimplemented, read as '0'. Reset values are shown in hexadecimal.

# dsPIC33FJXXMCX06/X08/X10

#### 4.2.7 SOFTWARE STACK

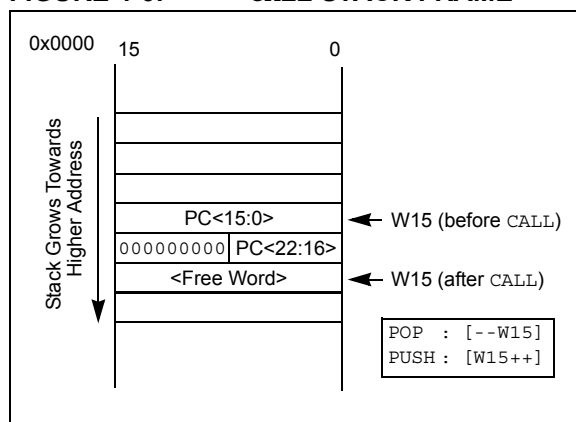
In addition to its use as a working register, the W15 register in the dsPIC33FJXXMCX06/X08/X10 devices is also used as a software Stack Pointer. The Stack Pointer always points to the first available free word and grows from lower to higher addresses. It pre-decrements for stack pops and post-increments for stack pushes, as shown in Figure 4-6. For a PC push during any CALL instruction, the MSb of the PC is zero-extended before the push, ensuring that the MSb is always clear.

**Note:** A PC push during exception processing concatenates the SRL register to the MSb of the PC prior to the push.

The Stack Pointer Limit register (SPLIM) associated with the Stack Pointer sets an upper address boundary for the stack. SPLIM is uninitialized at Reset. As is the case for the Stack Pointer, SPLIM<0> is forced to '0' because all stack operations must be word-aligned. Whenever an EA is generated using W15 as a source or destination pointer, the resulting address is compared with the value in SPLIM. If the contents of the Stack Pointer (W15) and the SPLIM register are equal and a push operation is performed, a stack error trap will not occur. The stack error trap will occur on a subsequent push operation. Thus, for example, if it is desirable to cause a stack error trap when the stack grows beyond address 0x2000 in RAM, initialize the SPLIM with the value 0x1FFE.

Similarly, a Stack Pointer underflow (stack error) trap is generated when the Stack Pointer address is found to be less than 0x0800. This prevents the stack from interfering with the Special Function Register (SFR) space.

A write to the SPLIM register should not be immediately followed by an indirect read operation using W15.

**FIGURE 4-6: CALL STACK FRAME**

#### 4.2.8 DATA RAM PROTECTION FEATURE

The dsPIC33FJXXXMCX06/X08/X10 devices supports Data RAM protection features which enable segments of RAM to be protected when used in conjunction with Boot and Secure Code Segment Security. BSRAM (Secure RAM segment for BS) is accessible only from the Boot Segment Flash code when enabled. SSRAM (Secure RAM segment for RAM) is accessible only from the Secure Segment Flash code when enabled. See Table 4-1 for an overview of the BSRAM and SSRAM SFRs.

### 4.3 Instruction Addressing Modes

The addressing modes in Table 4-36 form the basis of the addressing modes optimized to support the specific features of individual instructions. The addressing modes provided in the **MAC** class of instructions are somewhat different from those in the other instruction types.

### 4.3.1 FILE REGISTER INSTRUCTIONS

Most file register instructions use a 13-bit address field (f) to directly address data present in the first 8192 bytes of data memory (Near Data Space). Most file register instructions employ a working register, W0, which is denoted as WREG in these instructions. The destination is typically either the same file register or WREG (with the exception of the `MUL` instruction), which writes the result to a register or register pair. The `MOV` instruction allows additional flexibility and can access the entire data space.

### 4.3.2 MCU INSTRUCTIONS

The 3-operand MCU instructions are of the following form:

Operand 3 = Operand 1 <function> Operand 2

where Operand 1 is always a working register (i.e., the addressing mode can only be register direct) which is referred to as Wb. Operand 2 can be a W register fetched from data memory or a 5-bit literal. The result location can be either a W register or a data memory location. The following addressing modes are supported by MCU instructions:

- Register Direct
- Register Indirect
- Register Indirect Post-Modified
- Register Indirect Pre-Modified
- 5-bit or 10-bit Literal

<b>Note:</b>	Not all instructions support all the addressing modes given above. Individual instructions may support different subsets of these addressing modes.
--------------	---

# dsPIC33FJXXXMCX06/X08/X10

## 5.4.1 PROGRAMMING ALGORITHM FOR FLASH PROGRAM MEMORY

The user can program one row of program Flash memory at a time. To do this, it is necessary to erase the 8-row erase page that contains the desired row. The general process is as follows:

1. Read eight rows of program memory (512 instructions) and store it in data RAM.
2. Update the program data in RAM with the desired new data.
3. Erase the block (see Example 5-1):
  - a) Set the NVMOP bits (NVMCON<3:0>) to '0010' to configure for block erase. Set the ERASE (NVMCON<6>) and WREN (NVMCON<14>) bits.
  - b) Write the starting address of the page to be erased into the TBLPAG and W registers.
  - c) Write 0x55 to NVMKEY.
  - d) Write 0xAA to NVMKEY.
  - e) Set the WR bit (NVMCON<15>). The erase cycle begins and the CPU stalls for the duration of the erase cycle. When the erase is done, the WR bit is cleared automatically.
4. Write the first 64 instructions from data RAM into the program memory buffers (see Example 5-2).
5. Write the program block to Flash memory:
  - a) Set the NVMOP bits to '0001' to configure for row programming. Clear the ERASE bit and set the WREN bit.
  - b) Write 0x55 to NVMKEY.
  - c) Write 0xAA to NVMKEY.
  - d) Set the WR bit. The programming cycle begins and the CPU stalls for the duration of the write cycle. When the write to Flash memory is done, the WR bit is cleared automatically.
6. Repeat steps 4 and 5 using the next available 64 instructions from the block in data RAM by incrementing the value in TBLPAG until all 512 instructions are written back to Flash memory.

For protection against accidental operations, the write initiate sequence for NVMKEY must be used to allow any erase or program operation to proceed. After the programming command has been executed, the user must wait for the programming time until programming is complete. The two instructions following the start of the programming sequence should be NOPs, as shown in Example 5-3.

### EXAMPLE 5-1: ERASING A PROGRAM MEMORY PAGE

```
; Set up NVMCON for block erase operation
MOV    #0x4042, W0          ;
MOV     W0, NVMCON          ; Initialize NVMCON
; Init pointer to row to be ERASED
MOV     #tblpage(PROG_ADDR), W0 ;
MOV     W0, TBLPAG          ; Initialize PM Page Boundary SFR
MOV     #tbloffset(PROG_ADDR), W0 ; Initialize in-page EA[15:0] pointer
TBLWTL  W0, [W0]            ; Set base address of erase block
DISI    #5                  ; Block all interrupts with priority <7
                                ; for next 5 instructions

MOV     #0x55, W0
MOV     W0, NVMKEY          ; Write the 55 key
MOV     #0xAA, W1
MOV     W1, NVMKEY          ; Write the AA key
BSET    NVMCON, #WR         ; Start the erase sequence
NOP                                           ; Insert two NOPs after the erase
NOP                                           ; command is asserted
```

# dsPIC33FJXXXMCX06/X08/X10

---

## REGISTER 7-3: INTCON1: INTERRUPT CONTROL REGISTER 1 (CONTINUED)

bit 3	<b>ADDRERR:</b> Address Error Trap Status bit 1 = Address error trap has occurred 0 = Address error trap has not occurred
bit 2	<b>STKERR:</b> Stack Error Trap Status bit 1 = Stack error trap has occurred 0 = Stack error trap has not occurred
bit 1	<b>OSCFAIL:</b> Oscillator Failure Trap Status bit 1 = Oscillator failure trap has occurred 0 = Oscillator failure trap has not occurred
bit 0	<b>Unimplemented:</b> Read as '0'

# dsPIC33FJXXXMCX06/X08/X10

---

## REGISTER 7-11: IEC1: INTERRUPT ENABLE CONTROL REGISTER 1 (CONTINUED)

- bit 3      **CNIE:** Input Change Notification Interrupt Enable bit  
            1 = Interrupt request enabled  
            0 = Interrupt request not enabled
- bit 2      **Unimplemented:** Read as '0'
- bit 1      **MI2C1IE:** I2C1 Master Events Interrupt Enable bit  
            1 = Interrupt request enabled  
            0 = Interrupt request not enabled
- bit 0      **SI2C1IE:** I2C1 Slave Events Interrupt Enable bit  
            1 = Interrupt request enabled  
            0 = Interrupt request not enabled



## 10.2.2 IDLE MODE

Idle mode has the following features:

- The CPU stops executing instructions.
- The WDT is automatically cleared.
- The system clock source remains active. By default, all peripheral modules continue to operate normally from the system clock source, but can also be selectively disabled (see **Section 10.4 “Peripheral Module Disable”**).
- If the WDT or FSCM is enabled, the LPRC also remains active.

The device will wake from Idle mode on any of the following events:

- Any interrupt that is individually enabled
- Any device Reset
- A WDT time-out

On wake-up from Idle, the clock is reapplied to the CPU and instruction execution begins immediately, starting with the instruction following the `PWRSV` instruction or the first instruction in the ISR.

## 10.2.3 INTERRUPTS COINCIDENT WITH POWER SAVE INSTRUCTIONS

Any interrupt that coincides with the execution of a `PWRSV` instruction is held off until entry into Sleep or Idle mode has completed. The device then wakes up from Sleep or Idle mode.

## 10.3 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 setting.

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 while 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.

For example, suppose the device is operating at 20 MIPS and the CAN module has been configured for 500 kbps based on this device operating speed. If the device is now placed in Doze mode with a clock frequency ratio of 1:4, the CAN module continues to communicate at the required bit rate of 500 kbps, but the CPU now starts executing instructions at a frequency of 5 MIPS.

## 10.4 Peripheral Module Disable

The Peripheral Module Disable (PMD) registers provide a method to disable a peripheral module by stopping all clock sources supplied to that module. When a peripheral is disabled via the appropriate PMD control bit, the peripheral is in a minimum power consumption state. The control and status registers associated with the peripheral are also disabled, so writes to those registers will have no effect and read values will be invalid.

A peripheral module is only enabled if both the associated bit in the PMD register is cleared and the peripheral is supported by the specific dsPIC® DSC variant. If the peripheral is present in the device, it is enabled in the PMD register by default.

<b>Note:</b>	If a PMD bit is set, the corresponding module is disabled after a delay of 1 instruction cycle. Similarly, if a PMD bit is cleared, the corresponding module is enabled after a delay of 1 instruction cycle (assuming the module control registers are already configured to enable module operation).
--------------	---

# dsPIC33FJXXXMCX06/X08/X10

## REGISTER 12-1: T1CON: TIMER1 CONTROL REGISTER

R/W-0	U-0	R/W-0	U-0	U-0	U-0	U-0	U-0
TON	—	TSIDL	—	—	—	—	—
bit 15							bit 8

U-0	R/W-0	R/W-0	R/W-0	U-0	R/W-0	R/W-0	U-0
—	TGATE	TCKPS<1:0>		—	TSYNC	TCS	—
bit 7							bit 0

### Legend:

R = Readable bit

W = Writable bit

U = Unimplemented bit, read as '0'

-n = Value at POR

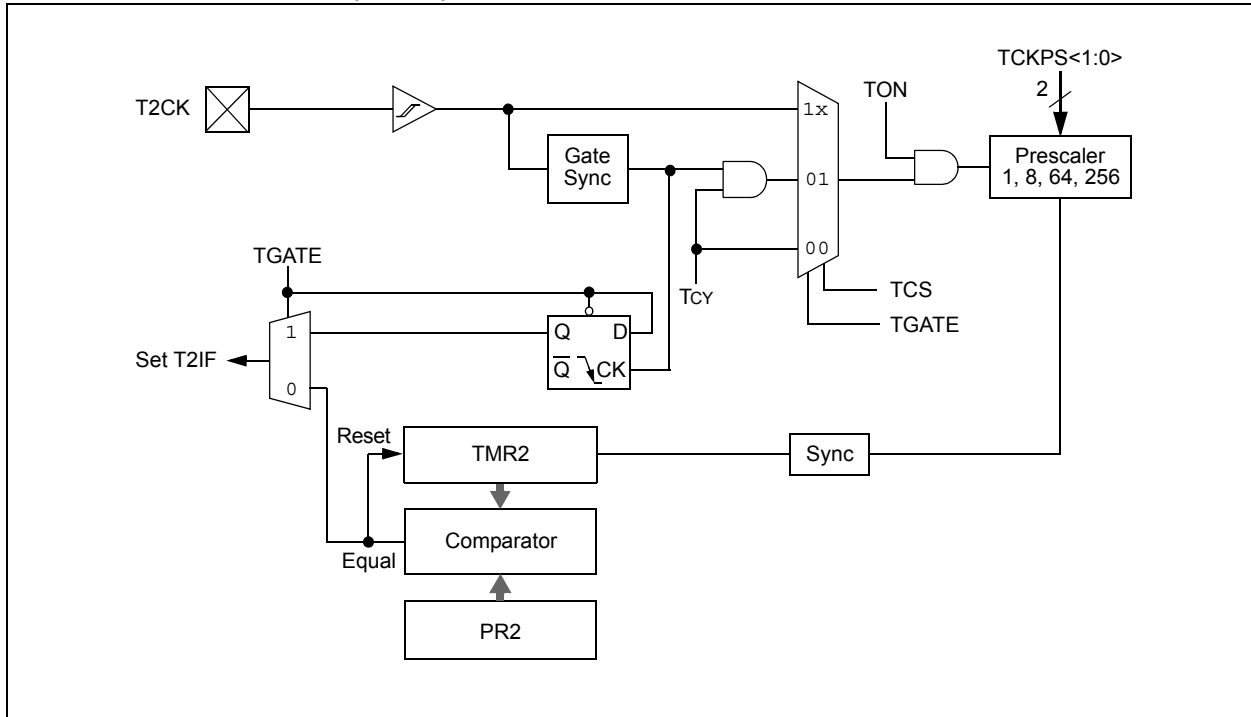
'1' = Bit is set

'0' = Bit is cleared

x = Bit is unknown

- bit 15      **TON:** Timer1 On bit  
               1 = Starts 16-bit Timer1  
               0 = Stops 16-bit Timer1
- bit 14      **Unimplemented:** Read as '0'
- bit 13      **TSIDL:** Stop in Idle Mode bit  
               1 = Discontinue module operation when device enters Idle mode  
               0 = Continue module operation in Idle mode
- bit 12-7    **Unimplemented:** Read as '0'
- bit 6        **TGATE:** Timer1 Gated Time Accumulation Enable bit  
               When T1CS = 1:  
               This bit is ignored.  
               When T1CS = 0:  
               1 = Gated time accumulation enabled  
               0 = Gated time accumulation disabled
- bit 5-4     **TCKPS<1:0>:** Timer1 Input Clock Prescale Select bits  
               11 = 1:256  
               10 = 1:64  
               01 = 1:8  
               00 = 1:1
- bit 3        **Unimplemented:** Read as '0'
- bit 2        **TSYNC:** Timer1 External Clock Input Synchronization Select bit  
               When TCS = 1:  
               1 = Synchronize external clock input  
               0 = Do not synchronize external clock input  
               When TCS = 0:  
               This bit is ignored.
- bit 1        **TCS:** Timer1 Clock Source Select bit  
               1 = External clock from pin T1CK (on the rising edge)  
               0 = Internal clock (Fcy)
- bit 0        **Unimplemented:** Read as '0'

**FIGURE 13-2: TIMER2 (16-BIT) BLOCK DIAGRAM**



# dsPIC33FJXXXMCX06/X08/X10

**REGISTER 13-1: TxCON (T2CON, T4CON, T6CON OR T8CON) CONTROL REGISTER**

R/W-0	U-0	R/W-0	U-0	U-0	U-0	U-0	U-0
TON	—	TSIDL	—	—	—	—	—
bit 15							bit 8

U-0	R/W-0	R/W-0	R/W-0	R/W-0	U-0	R/W-0	U-0
—	TGATE	TCKPS<1:0>		T32	—	TCS <sup>(1)</sup>	—
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 **TON:** Timerx On bit

When T32 = 1:

1 = Starts 32-bit Timerx/y

0 = Stops 32-bit Timerx/y

When T32 = 0:

1 = Starts 16-bit Timerx

0 = Stops 16-bit Timerx

bit 14 **Unimplemented:** Read as '0'

bit 13 **TSIDL:** Stop in Idle Mode bit

1 = Discontinue module operation when device enters Idle mode

0 = Continue module operation in Idle mode

bit 12-7 **Unimplemented:** Read as '0'

bit 6 **TGATE:** Timerx Gated Time Accumulation Enable bit

When TCS = 1:

This bit is ignored

When TCS = 0:

1 = Gated time accumulation enabled

0 = Gated time accumulation disabled

bit 5-4 **TCKPS<1:0>:** Timerx Input Clock Prescale Select bits

11 = 1:256

10 = 1:64

01 = 1:8

00 = 1:1

bit 3 **T32:** 32-bit Timer Mode Select bit

1 = Timerx and Timery form a single 32-bit timer

0 = Timerx and Timery act as two 16-bit timers

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

bit 1 **TCS:** Timerx Clock Source Select bit<sup>(1)</sup>

1 = External clock from pin TxCK (on the rising edge)

0 = Internal clock (Fcy)

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

**Note 1:** The TxCK pin is not available on all timers. Refer to the “Pin Diagrams” section for the available pins.

# dsPIC33FJXXMCMX06/X08/X10

## REGISTER 16-1: P<sub>x</sub>TCON: PWM TIME BASE CONTROL REGISTER

R/W-0	U-0	R/W-0	U-0	U-0	U-0	U-0	U-0
PTEN	—	PTSIDL	—	—	—	—	—
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
PTOPS<3:0>				PTCKPS<1:0>		PTMOD<1:0>	
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 **PTEN:** PWM Time Base Timer Enable bit

1 = PWM time base is on

0 = PWM time base is off

bit 14 **Unimplemented:** Read as '0'

bit 13 **PTSIDL:** PWM Time Base Stop in Idle Mode bit

1 = PWM time base halts in CPU Idle mode

0 = PWM time base runs in CPU Idle mode

bit 12-8 **Unimplemented:** Read as '0'

bit 7-4 **PTOPS<3:0>:** PWM Time Base Output Postscale Select bits

1111 = 1:16 postscale

•  
•  
•

0001 = 1:2 postscale

0000 = 1:1 postscale

bit 3-2 **PTCKPS<1:0>:** PWM Time Base Input Clock Prescale Select bits

11 = PWM time base input clock period is 64 T<sub>CY</sub> (1:64 prescale)

10 = PWM time base input clock period is 16 T<sub>CY</sub> (1:16 prescale)

01 = PWM time base input clock period is 4 T<sub>CY</sub> (1:4 prescale)

00 = PWM time base input clock period is T<sub>CY</sub> (1:1 prescale)

bit 1-0 **PTMOD<1:0>:** PWM Time Base Mode Select bits

11 = PWM time base operates in a Continuous Up/Down Count mode with interrupts for double PWM updates

10 = PWM time base operates in a Continuous Up/Down Count mode

01 = PWM time base operates in Single Pulse mode

00 = PWM time base operates in a Free-Running mode

# dsPIC33FJXXMCX06/X08/X10

## REGISTER 16-4: PxSECMP: SPECIAL EVENT COMPARE 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
SEVTDIR <sup>(1)</sup>	SEVTCMP<14:8> <sup>(2)</sup>						
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
SEVTCMP<7:0> <sup>(2)</sup>							
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      **SEVTDIR:** Special Event Trigger Time Base Direction bit<sup>(1)</sup>

1 = A Special Event Trigger will occur when the PWM time base is counting downwards

0 = A Special Event Trigger will occur when the PWM time base is counting upwards

bit 14-0      **SEVTCMP<14:0>:** Special Event Compare Value bits<sup>(2)</sup>

**Note 1:** SEVTDIR is compared with PTDIR (PTMR<15>) to generate the Special Event Trigger.

**2:** SEVTCMP<14:0> is compared with PTMR<14:0> to generate the Special Event Trigger.

# dsPIC33FJXXMCMX06/X08/X10

**REGISTER 21-26: CiTRmnCON: ECAN™ TX/RX BUFFER m CONTROL REGISTER (m = 0,2,4,6; n = 1,3,5,7)**

R/W-0	R-0	R-0	R-0	R/W-0	R/W-0	R/W-0	R/W-0
TXENn	TXABTn	TXLARBn	TXERRn	TXREQn	RTRENn	TXnPRI<1:0>	
bit 15							bit 8

R/W-0	R-0	R-0	R-0	R/W-0	R/W-0	R/W-0	R/W-0
TXENm	TXABTm <sup>(1)</sup>	TXLARBm <sup>(1)</sup>	TXERRm <sup>(1)</sup>	TXREQm	RTRENm	TXmPRI<1:0>	
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 **See Definition for Bits 7-0, Controls Buffer n**

bit 7 **TXENm:** TX/RX Buffer Selection bit

1 = Buffer TRBn is a transmit buffer

0 = Buffer TRBn is a receive buffer

bit 6 **TXABTm:** Message Aborted bit<sup>(1)</sup>

1 = Message was aborted

0 = Message completed transmission successfully

bit 5 **TXLARBm:** Message Lost Arbitration bit<sup>(1)</sup>

1 = Message lost arbitration while being sent

0 = Message did not lose arbitration while being sent

bit 4 **TXERRm:** Error Detected During Transmission bit<sup>(1)</sup>

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 **TXREQm:** Message Send Request bit

Setting this bit to '1' requests sending a message. The bit will automatically clear when the message is successfully sent. Clearing the bit to '0' while set will request a message abort.

bit 2 **RTRENm:** Auto-Remote Transmit Enable bit

1 = When a remote transmit is received, TXREQ will be set

0 = When a remote transmit is received, TXREQ will be unaffected

bit 1-0 **TXmPRI<1:0>:** Message Transmission Priority bits

11 = Highest message priority

10 = High intermediate message priority

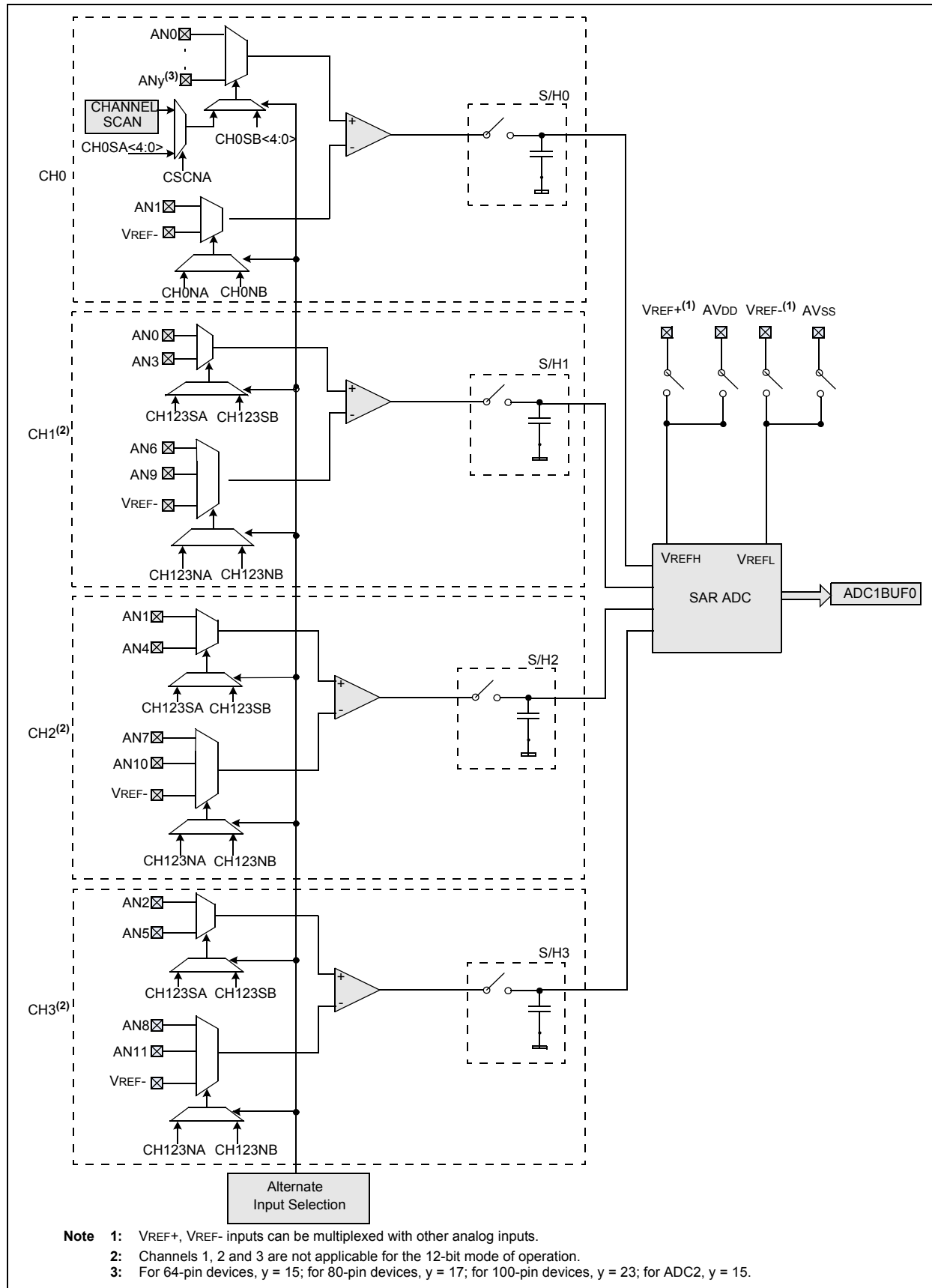
01 = Low intermediate message priority

00 = Lowest message priority

**Note 1:** This bit is cleared when TXREQ is set.

# dsPIC33FJXXXMCX06/X08/X10

**FIGURE 22-1: ADC1 MODULE BLOCK DIAGRAM**





# dsPIC33FJXXXMCX06/X08/X10

## REGISTER 22-6: ADxCHS0: ADCx INPUT CHANNEL 0 SELECT REGISTER

R/W-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
CH0NB	—	—	CH0SB<4:0>				
bit 15							bit 8

R/W-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
CH0NA	—	—	CH0SA<4:0>				
bit 7							bit 0

### Legend:

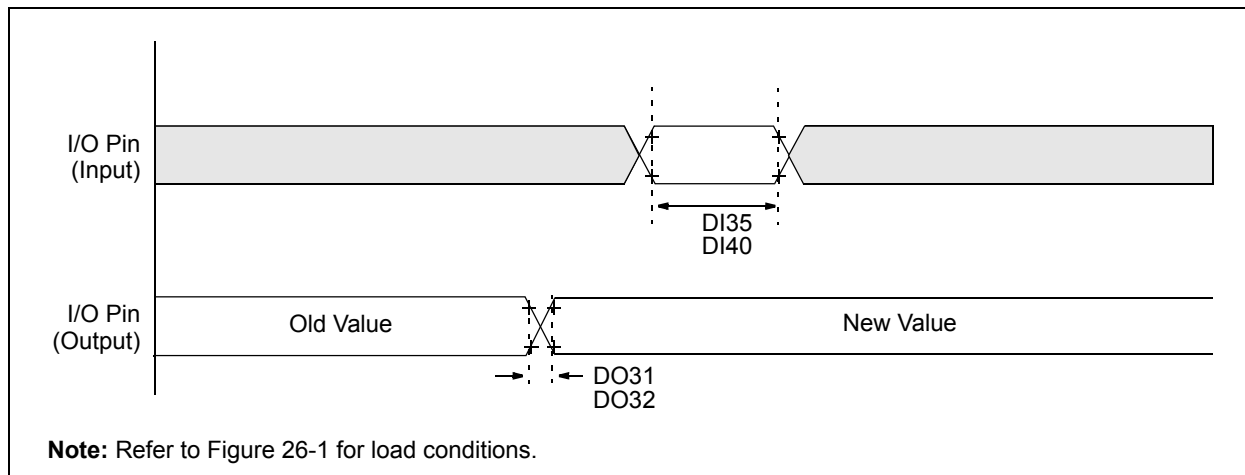
R = Readable bit      W = Writable bit      U = Unimplemented bit, read as '0'  
 -n = Value at POR      '1' = Bit is set      '0' = Bit is cleared      x = Bit is unknown

- bit 15      **CH0NB:** Channel 0 Negative Input Select for Sample B bit  
 Same definition as bit 7.
- bit 14-13      **Unimplemented:** Read as '0'
- bit 12-8      **CH0SB<4:0>:** Channel 0 Positive Input Select for Sample B bits  
 Same definition as bit<4:0>.
- bit 7      **CH0NA:** Channel 0 Negative Input Select for Sample A bit  
 1 = Channel 0 negative input is AN1  
 0 = Channel 0 negative input is VREF-
- bit 6-5      **Unimplemented:** Read as '0'
- bit 4-0      **CH0SA<4:0>:** Channel 0 Positive Input Select for Sample A bits  
 11111 = Channel 0 positive input is AN31  
 11110 = Channel 0 positive input is AN30  
 .  
 .  
 .  
 00010 = Channel 0 positive input is AN2  
 00001 = Channel 0 positive input is AN1  
 00000 = Channel 0 positive input is AN0

**Note:** ADC2 can only select AN0-AN15 as positive inputs.

# dsPIC33FJXXMCX06/X08/X10

**FIGURE 26-3: CLKO AND I/O TIMING CHARACTERISTICS**



**TABLE 26-20: I/O TIMING REQUIREMENTS**

AC CHARACTERISTICS			Standard Operating Conditions: 3.0V to 3.6V (unless otherwise stated) Operating temperature $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$ for Industrial				
Param No.	Symbol	Characteristic	Min	Typ <sup>(1)</sup>	Max	Units	Conditions
DO31	TioR	Port Output Rise Time	—	10	25	ns	—
DO32	TioF	Port Output Fall Time	—	10	25	ns	—
DI35	TINP	INTx Pin High or Low Time (output)	20	—	—	ns	—
DI40	TRBP	CNx High or Low Time (input)	2	—	—	TcY	—

**Note 1:** Data in “Typ” column is at 3.3V, 25°C unless otherwise stated.

# dsPIC33FJXXMCX06/X08/X10

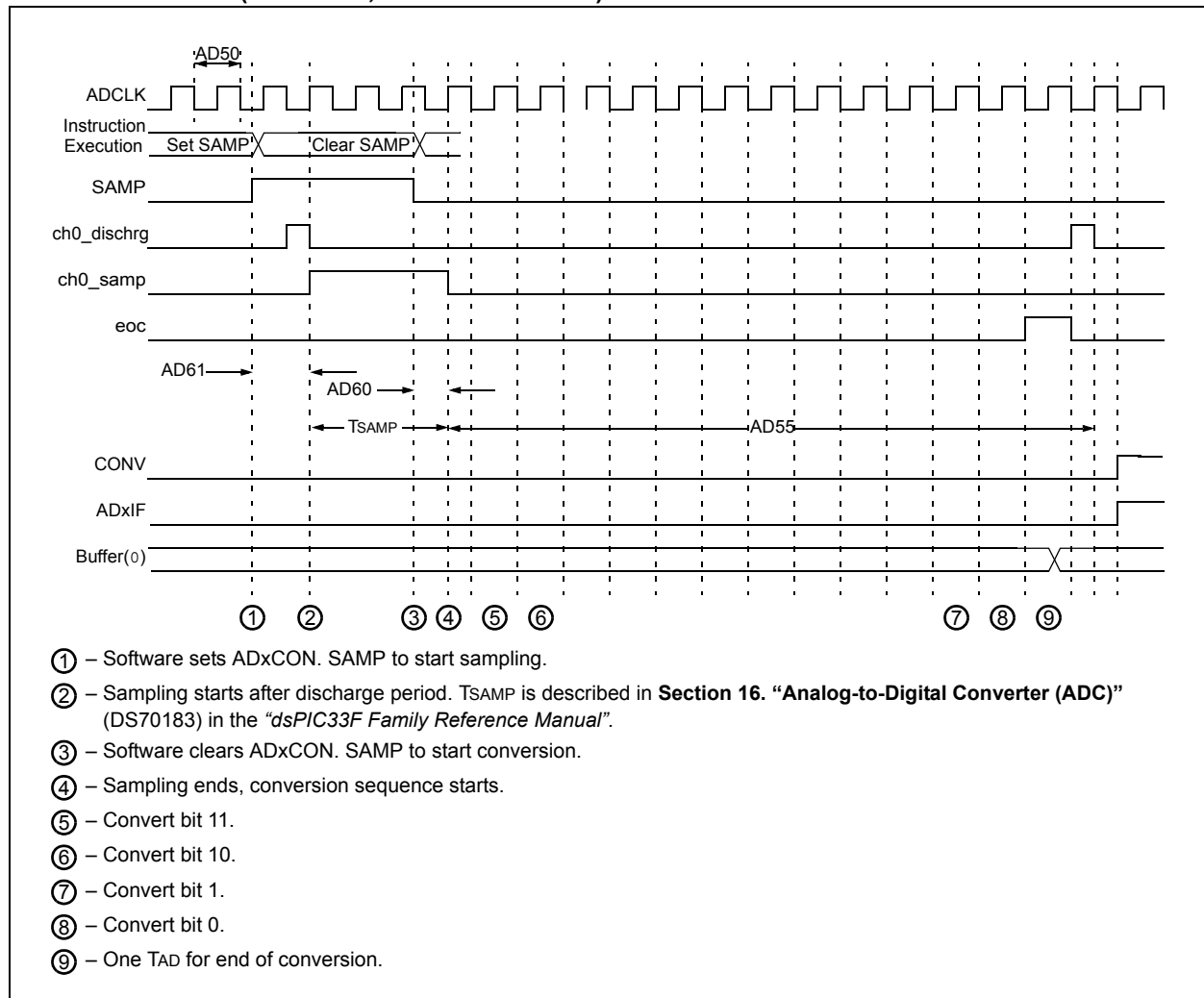
**TABLE 26-23: TIMER2, TIMER4, TIMER6 AND TIMER8 EXTERNAL CLOCK TIMING REQUIREMENTS**

AC CHARACTERISTICS				Standard Operating Conditions: 3.0V to 3.6V (unless otherwise stated) Operating temperature    -40°C ≤ TA ≤ +85°C for Industrial				
Param No.	Symbol	Characteristic		Min	Typ	Max	Units	Conditions
TB10	TtxH	TxCK High Time	Synchronous, no prescaler	0.5 Tcy + 20	—	—	ns	Must also meet parameter TB15
			Synchronous, with prescaler	10	—	—	ns	
TB11	TtxL	TxCK Low Time	Synchronous, no prescaler	0.5 Tcy + 20	—	—	ns	Must also meet parameter TB15
			Synchronous, with prescaler	10	—	—	ns	
TB15	TtxP	TxCK Input Period	Synchronous, no prescaler	Tcy + 40	—	—	ns	N = prescale value (1, 8, 64, 256)
			Synchronous, with prescaler	Greater of: 20 ns or (Tcy + 40)/N				
TB20	TCKEXT-MRL	Delay from External TxCK Clock Edge to Timer Increment		0.5 Tcy	—	1.5 Tcy	—	—

**TABLE 26-24: TIMER3, TIMER5, TIMER7 AND TIMER9 EXTERNAL CLOCK TIMING REQUIREMENTS**

AC CHARACTERISTICS				Standard Operating Conditions: 3.0V to 3.6V (unless otherwise stated) Operating temperature $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$ for Industrial				
Param No.	Symbol	Characteristic		Min	Typ	Max	Units	Conditions
TC10	TtxH	TxCK High Time	Synchronous	$0.5 T_{CY} + 20$	—	—	ns	Must also meet parameter TC15
TC11	TtxL	TxCK Low Time	Synchronous	$0.5 T_{CY} + 20$	—	—	ns	Must also meet parameter TC15
TC15	TtxP	TxCK Input Period	Synchronous, no prescaler	$T_{CY} + 40$	—	—	ns	N = prescale value (1, 8, 64, 256)
			Synchronous, with prescaler	Greater of: 20 ns or $(T_{CY} + 40)/N$				
TC20	TCKEXTMRL	Delay from External TxCK Clock Edge to Timer Increment		$0.5 T_{CY}$	—	$1.5 T_{CY}$	—	—

**FIGURE 26-23: ADC CONVERSION (12-BIT MODE) TIMING CHARACTERISTICS**  
(ASAM = 0, SSRC<2:0> = 000)



# dsPIC33FJXXMCX06/X08/X10

## PRODUCT IDENTIFICATION SYSTEM

To order or obtain information, e.g., on pricing or delivery, refer to the factory or the listed sales office.

	dsPIC	33	FJ	256	MC7	10	T	I	/ PT	- XXX
Microchip Trademark	_____	_____	_____	_____	_____	_____	_____	_____	_____	_____
Architecture	_____	_____	_____	_____	_____	_____	_____	_____	_____	_____
Flash Memory Family	_____	_____	_____	_____	_____	_____	_____	_____	_____	_____
Program Memory Size (KB)	_____	_____	_____	_____	_____	_____	_____	_____	_____	_____
Product Group	_____	_____	_____	_____	_____	_____	_____	_____	_____	_____
Pin Count	_____	_____	_____	_____	_____	_____	_____	_____	_____	_____
Tape and Reel Flag (if applicable)	_____	_____	_____	_____	_____	_____	_____	_____	_____	_____
Temperature Range	_____	_____	_____	_____	_____	_____	_____	_____	_____	_____
Package	_____	_____	_____	_____	_____	_____	_____	_____	_____	_____
Pattern	_____	_____	_____	_____	_____	_____	_____	_____	_____	_____

Architecture:            33        =    16-bit Digital Signal Controller

Flash Memory Family:   FJ        =    Flash program memory, 3.3V

Product Group:           MC5    =    Motor Control family  
                             MC7    =    Motor Control family

Pin Count:                06        =    64-pin  
                              08        =    80-pin  
                              10        =    100-pin

Temperature Range:    I         =    -40°C to   +85°C    (Industrial)

Package:                 PT        =    10x10 or 12x12 mm TQFP (Thin Quad Flat-pack)  
                              PF        =    14x14 mm TQFP (Thin Quad Flatpack)

Pattern                    Three-digit QTP, SQTP, Code or Special Requirements  
                              (blank otherwise)

**Examples:**

a)    dsPIC33FJ64MC706I/PT:  
      Motor Control dsPIC33, 64 KB program  
      memory, 64-pin, Industrial temp.,  
      TQFP package.