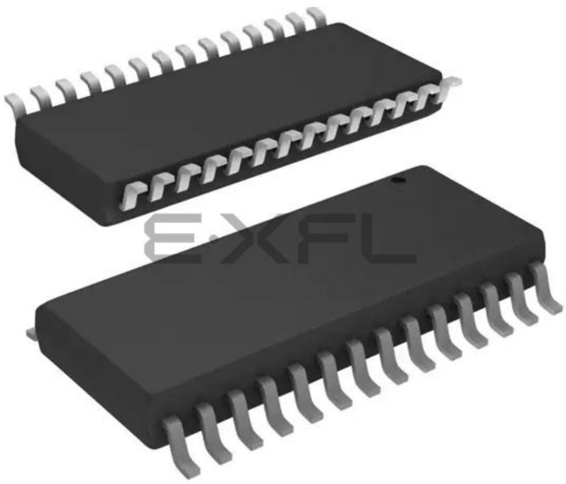


Welcome to E-XFL.COM

What is "Embedded - Microcontrollers"?

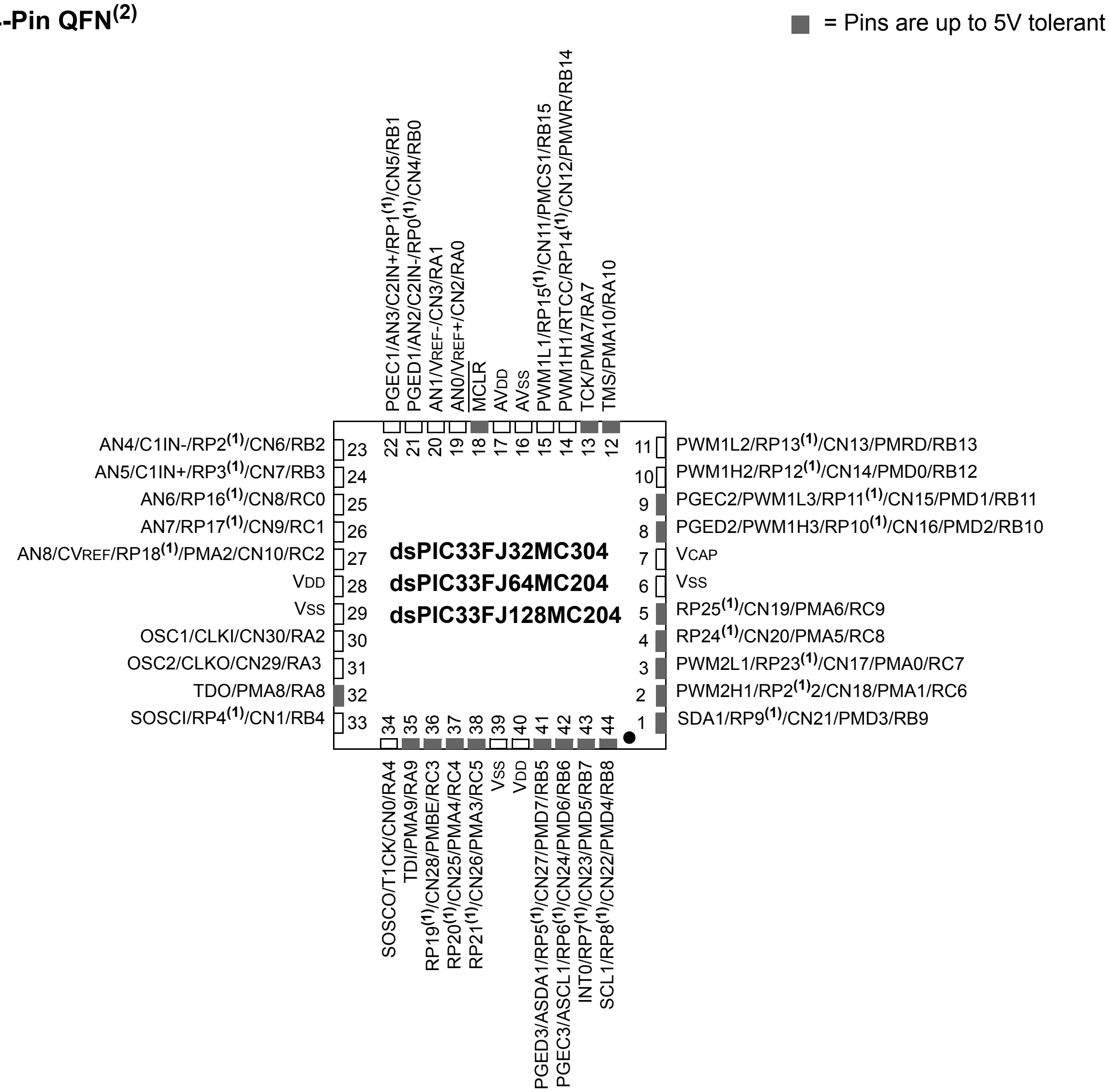
"Embedded - Microcontrollers" refer to small, integrated circuits designed to perform specific tasks within larger systems. These microcontrollers are essentially compact computers on a single chip, containing a processor core, memory, and programmable input/output peripherals. They are called "embedded" 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	dsPIC
Core Size	16-Bit
Speed	40 MIPs
Connectivity	I²C, IrDA, LINbus, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, DMA, Motor Control PWM, POR, PWM, QEI, WDT
Number of I/O	21
Program Memory Size	32KB (32K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	4K x 8
Voltage - Supply (Vcc/Vdd)	3V ~ 3.6V
Data Converters	A/D 6x10b/12b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	28-SOIC (0.295", 7.50mm Width)
Supplier Device Package	28-SOIC
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/dspic33fj32mc302t-i-so

Pin Diagrams (Continued)

44-Pin QFN(2)



Note 1: The RPx pins can be used by any remappable peripheral. See Table 1 in this section for the list of available peripherals.

2: The metal plane at the bottom of the device is not connected to any pins and is recommended to be connected to VSS externally.

3.3 DSP Engine Overview

The DSP engine features a high-speed 17-bit by 17-bit multiplier, a 40-bit ALU, two 40-bit saturating accumulators and a 40-bit bidirectional barrel shifter. The barrel shifter is capable of shifting a 40-bit value up to 16 bits right or left, in a single cycle. The DSP instructions operate seamlessly with all other instructions and have been designed for optimal real-time performance. The `MAC` instruction and other associated instructions can concurrently fetch two data operands from memory while multiplying two W registers and accumulating and optionally saturating the result in the same cycle. This instruction functionality requires that the RAM data space be split for these instructions and linear for all others. Data space partitioning is achieved in a transparent and flexible manner by dedicating certain working registers to each address space.

3.4 Special MCU Features

The dsPIC33FJ32MC302/304, dsPIC33FJ64MCX02/X04 and dsPIC33FJ128MCX02/X04 features a 17-bit by 17-bit single-cycle multiplier that is shared by both the MCU ALU and DSP engine. The multiplier can perform signed, unsigned and mixed-sign multiplication. Using a 17-bit by 17-bit multiplier for 16-bit by 16-bit multiplication not only allows you to perform mixed-sign multiplication, it also achieves accurate results for special operations, such as (-1.0) x (-1.0).

The dsPIC33FJ32MC302/304, dsPIC33FJ64MCX02/X04 and dsPIC33FJ128MCX02/X04 supports 16/16 and 32/16 divide operations, both fractional and integer. All divide instructions are iterative operations. They must be executed within a `REPEAT` loop, resulting in a total execution time of 19 instruction cycles. The divide operation can be interrupted during any of those 19 cycles without loss of data.

A 40-bit barrel shifter is used to perform up to a 16-bit left or right shift in a single cycle. The barrel shifter can be used by both MCU and DSP instructions.

FIGURE 3-2: dsPIC33FJ32MC302/304, dsPIC33FJ64MCX02/X04 AND dsPIC33FJ128MCX02/X04 PROGRAMMER'S MODEL

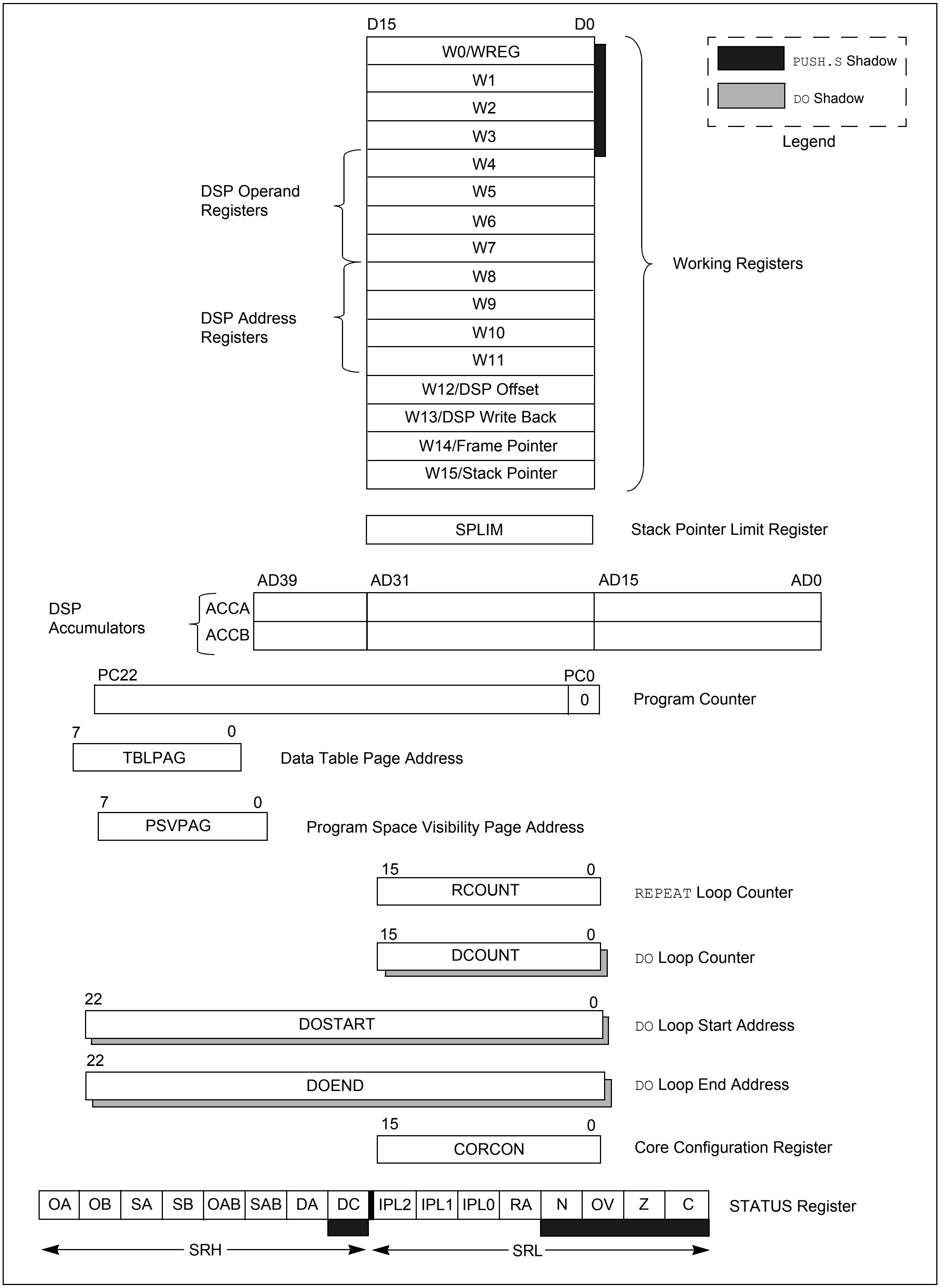


TABLE 4-12: I2C1 REGISTER MAP

SFR 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
I2C1RCV	0200	—	—	—	—	—	—	—	—	Receive Register								0000
I2C1TRN	0202	—	—	—	—	—	—	—	—	Transmit Register								00FF
I2C1BRG	0204	—	—	—	—	—	—	—	Baud Rate Generator Register									0000
I2C1CON	0206	I2CEN	—	I2CSIDL	SCLREL	IPMIEN	A10M	DISSLW	SMEN	GCEN	STREN	ACKDT	ACKEN	RCEN	PEN	RSEN	SEN	1000
I2C1STAT	0208	ACKSTAT	TRSTAT	—	—	—	BCL	GCSTAT	ADD10	IWCOL	I2COV	D_A	P	S	R_W	RBF	TBF	0000
I2C1ADD	020A	—	—	—	—	—	—	Address Register										0000
I2C1MSK	020C	—	—	—	—	—	—	Address Mask Register										0000

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

TABLE 4-13: UART1 REGISTER MAP

SFR 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	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	—	—	—	—	—	—	—	UTX8	UART Transmit Register								xxxx
U1RXREG	0226	—	—	—	—	—	—	—	URX8	UART Received 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	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	—	—	—	—	—	—	—	UTX8	UART Transmit Register								xxxx
U2RXREG	0236	—	—	—	—	—	—	—	URX8	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 4-40: FUNDAMENTAL ADDRESSING MODES SUPPORTED

Addressing Mode	Description
File Register Direct	The address of the file register is specified explicitly.
Register Direct	The contents of a register are accessed directly.
Register Indirect	The contents of Wn forms the Effective Address (EA).
Register Indirect Post-Modified	The contents of Wn forms the EA. Wn is post-modified (incremented or decremented) by a constant value.
Register Indirect Pre-Modified	Wn is pre-modified (incremented or decremented) by a signed constant value to form the EA.
Register Indirect with Register Offset (Register Indexed)	The sum of Wn and Wb forms the EA.
Register Indirect with Literal Offset	The sum of Wn and a literal forms the EA.

4.5.3 MOVE AND ACCUMULATOR INSTRUCTIONS

Move instructions and the DSP accumulator class of instructions provide a greater degree of addressing flexibility than any other instructions. In addition to the addressing modes supported by most MCU instructions, move and accumulator instructions also support Register Indirect with Register Offset Addressing mode, also referred to as Register Indexed mode.

> **Note:** For the MOV instructions, the addressing mode specified in the instruction can differ for the source and destination EA. However, the 4-bit Wb (Register Offset) field is shared by both source and destination (but typically only used by one).

In summary, the following addressing modes are supported by move and accumulator instructions:

- Register Direct
- Register Indirect
- Register Indirect Post-Modified
- Register Indirect Pre-Modified
- Register Indirect with Register Offset (Indexed)
- Register Indirect with Literal Offset
- 8-bit Literal
- 16-bit Literal

> **Note:** Not all instructions support all the addressing modes listed above. Individual instructions may support different subsets of these addressing modes.

4.5.4 MAC INSTRUCTIONS

The dual source operand DSP instructions (CLR, ED, EDAC, MAC, MPY, MPY.N, MOVSAC and MSC), also referred to as MAC instructions, use a simplified set of addressing modes to allow the user application to effectively manipulate the data pointers through register indirect tables.

The two-source operand prefetch registers must be members of the set {W8, W9, W10, W11}. For data reads, W8 and W9 are always directed to the X RAGU, and W10 and W11 are always directed to the Y AGU. The effective addresses generated (before and after modification) must, therefore, be valid addresses within X data space for W8 and W9 and Y data space for W10 and W11.

> **Note:** Register Indirect with Register Offset Addressing mode is available only for W9 (in X space) and W11 (in Y space).

In summary, the following addressing modes are supported by the MAC class of instructions:

- Register Indirect
- Register Indirect Post-Modified by 2
- Register Indirect Post-Modified by 4
- Register Indirect Post-Modified by 6
- Register Indirect with Register Offset (Indexed)

4.5.5 OTHER INSTRUCTIONS

Apart from the addressing modes outlined previously, some instructions use literal constants of various sizes. For example, BRA (branch) instructions use 16-bit signed literals to specify the branch destination directly, whereas, the DISI instruction uses a 14-bit unsigned literal field. In some instructions, such as ADD Acc, the source of an operand or result is implied by the opcode itself. Certain operations, such as NOP, do not have any operands.

EXAMPLE 5-2: LOADING THE WRITE BUFFERS

```
; Set up NVMCON for row programming operations
       MOV     #0x4001, W0                ;
       MOV     W0, NVMCON                 ; Initialize NVMCON
; Set up a pointer to the first program memory location to be written
; program memory selected, and writes enabled
       MOV     #0x0000, W0                ;
       MOV     W0, TBLPAG                 ; Initialize PM Page Boundary SFR
       MOV     #0x6000, W0                ; An example program memory address
; Perform the TBLWT instructions to write the latches
; 0th_program_word
       MOV     #LOW_WORD_0, W2            ;
       MOV     #HIGH_BYTE_0, W3           ;
       TBLWTL  W2, [W0]                   ; Write PM low word into program latch
       TBLWTH  W3, [W0++]                 ; Write PM high byte into program latch
; 1st_program_word
       MOV     #LOW_WORD_1, W2            ;
       MOV     #HIGH_BYTE_1, W3           ;
       TBLWTL  W2, [W0]                   ; Write PM low word into program latch
       TBLWTH  W3, [W0++]                 ; Write PM high byte into program latch
;  2nd_program_word
       MOV     #LOW_WORD_2, W2            ;
       MOV     #HIGH_BYTE_2, W3           ;
       TBLWTL  W2, [W0]                   ; Write PM low word into program latch
       TBLWTH  W3, [W0++]                 ; Write PM high byte into program latch
       •
       •
       •
; 63rd_program_word
       MOV     #LOW_WORD_31, W2           ;
       MOV     #HIGH_BYTE_31, W3          ;
       TBLWTL  W2, [W0]                   ; Write PM low word into program latch
       TBLWTH  W3, [W0++]                 ; Write PM high byte into program latch
```

EXAMPLE 5-3: INITIATING A PROGRAMMING SEQUENCE

```
       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
       NOP                                ; erase command is asserted
```

REGISTER 7-7: IFS2: INTERRUPT FLAG STATUS REGISTER 2

U-0	R/W-0	R/W-0	U-0	U-0	U-0	U-0	U-0
—	DMA4IF	PMPIF	—	—	—	—	—
bit 15							bit 8

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	DMA3IF	C1IF(1)	C1RXIF(1)	SPI2IF	SPI2EIF
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 **DMA4IF:** DMA Channel 4 Data Transfer Complete Interrupt Flag Status bit
1 = Interrupt request has occurred
0 = Interrupt request has not occurred

bit 13 **PMPIF:** Parallel Master Port Interrupt Flag Status bit
1 = Interrupt request has occurred
0 = Interrupt request has not occurred

bit 12-5 **Unimplemented:** Read as ‘0’

bit 4 **DMA3IF:** DMA Channel 3 Data Transfer Complete Interrupt Flag Status bit
1 = Interrupt request has occurred
0 = Interrupt request has not occurred

bit 3 **C1IF:** ECAN1 Event Interrupt Flag Status bit(1)
1 = Interrupt request has occurred
0 = Interrupt request has not occurred

bit 2 **C1RXIF:** ECAN1 Receive Data Ready Interrupt Flag Status bit(1)
1 = Interrupt request has occurred
0 = Interrupt request has not occurred

bit 1 **SPI2IF:** SPI2 Event Interrupt Flag Status bit
1 = Interrupt request has occurred
0 = Interrupt request has not occurred

bit 0 **SPI2EIF:** SPI2 Error Interrupt Flag Status bit
1 = Interrupt request has occurred
0 = Interrupt request has not occurred

Note 1: Interrupts are disabled on devices without an ECAN™ module.

REGISTER 7-22: 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
—	U2TXIP<2:0>			—	U2RXIP<2:0>		
bit 15							bit 8

U-0	R/W-1	R/W-0	R/W-0	U-0	R/W-1	R/W-0	R/W-0
—	INT2IP<2:0>			—	T5IP<2:0>		
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 **Unimplemented:** Read as ‘0’

bit 14-12 **U2TXIP<2:0>:** UART2 Transmitter Interrupt Priority bits
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
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 **Unimplemented:** Read as ‘0’

bit 2-0 **T5IP<2:0>:** Timer5 Interrupt Priority bits
111 = Interrupt is priority 7 (highest priority interrupt)
•
•
•
001 = Interrupt is priority 1
000 = Interrupt source is disabled

7.6 Interrupt Setup Procedures

7.6.1 INITIALIZATION

To configure an interrupt source at initialization:

1. Set the NSTDIS bit (INTCON1<15>) if nested interrupts are not desired.
2. Select the user-assigned priority level for the interrupt source by writing the control bits in the appropriate IPCx register. The priority level depends on the specific application and type of interrupt source. If multiple priority levels are not desired, the IPCx register control bits for all enabled interrupt sources can be programmed to the same non-zero value.

> **Note:** At a device Reset, the IPCx registers are initialized such that all user interrupt sources are assigned to priority level 4.

3. Clear the interrupt flag status bit associated with the peripheral in the associated IFSx register.
4. Enable the interrupt source by setting the interrupt enable control bit associated with the source in the appropriate IECx register.

7.6.2 INTERRUPT SERVICE ROUTINE

The method used to declare an ISR and initialize the IVT with the correct vector address depends on the programming language (C or assembler) and the language development tool suite used to develop the application.

In general, the user application must clear the interrupt flag in the appropriate IFSx register for the source of interrupt that the ISR handles. Otherwise, the program re-enters the ISR immediately after exiting the routine. If the ISR is coded in assembly language, it must be terminated using a `RETFIE` instruction to unstack the saved PC value, SRL value and old CPU priority level.

7.6.3 TRAP SERVICE ROUTINE

A Trap Service Routine (TSR) is coded like an ISR, except that the appropriate trap status flag in the INTCON1 register must be cleared to avoid re-entry into the TSR.

7.6.4 INTERRUPT DISABLE

All user interrupts can be disabled using this procedure:

1. Push the current SR value onto the software stack using the `PUSH` instruction.
2. Force the CPU to priority level 7 by inclusive ORing the value OEh with SRL.

To enable user interrupts, the `POP` instruction can be used to restore the previous SR value.

> **Note:** Only user interrupts with a priority level of 7 or lower can be disabled. Trap sources (level 8-level 15) cannot be disabled.

The `DISI` instruction provides a convenient way to disable interrupts of priority levels 1-6 for a fixed period of time. Level 7 interrupt sources are not disabled by the `DISI` instruction.

9.0 OSCILLATOR CONFIGURATION

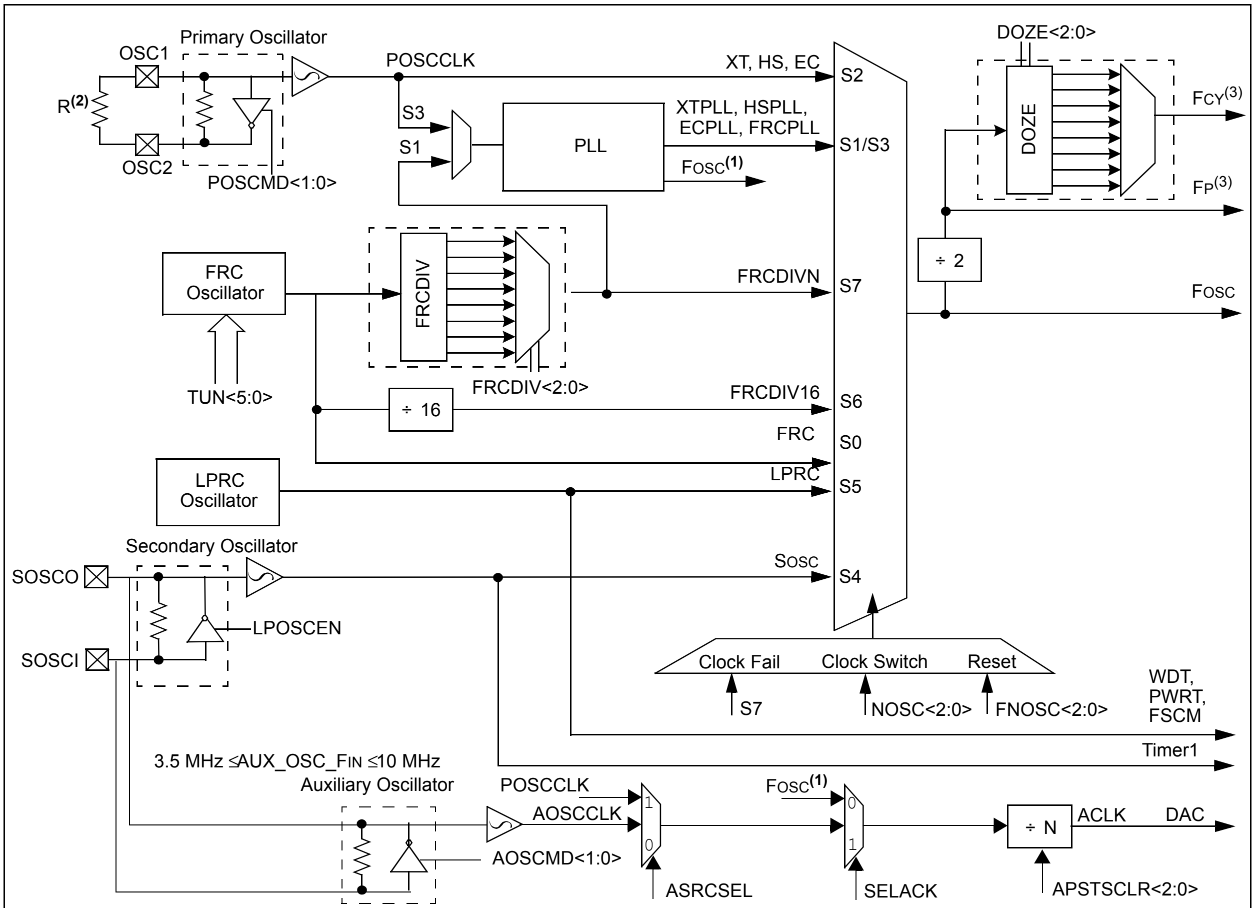
> **Note 1:** This data sheet summarizes the features of the dsPIC33FJ32MC302/304, dsPIC33FJ64MCX02/X04 and dsPIC33FJ128MCX02/X04 family of devices. It is not intended to be a comprehensive reference source. To complement the information in this data sheet, refer to **Section 39. "Oscillator (Part III)"** (DS70216) of the "*dsPIC33F/PIC24H Family Reference Manual*", which is available from the Microchip web site (www.microchip.com).
>
> **2:** Some registers and associated bits described in this section may not be available on all devices. Refer to **Section 4.0 "Memory Organization"** in this data sheet for device-specific register and bit information.

The dsPIC33FJ32MC302/304, dsPIC33FJ64MCX02/X04 and dsPIC33FJ128MCX02/X04 oscillator system provides:

- External and internal oscillator options as clock sources
- An on-chip Phase-Locked Loop (PLL) to scale the internal operating frequency to the required system clock frequency
- An internal FRC oscillator that can also be used with the PLL, thereby allowing full-speed operation without any external clock generation hardware
- Clock switching between various clock sources
- Programmable clock postscaler for system power savings
- A Fail-Safe Clock Monitor (FSCM) that detects clock failure and takes fail-safe measures
- An Oscillator Control register (OSCCON)
- Non-volatile Configuration bits for main oscillator selection
- An auxiliary crystal oscillator for audio DAC

A simplified diagram of the oscillator system is shown in Figure 9-1.

FIGURE 9-1: OSCILLATOR SYSTEM DIAGRAM



Note 1: See Figure 9-2 for PLL details.

2: If the Oscillator is used with XT or HS modes, an external parallel resistor with the value of 1 MΩ must be connected.

3: The term FP refers to the clock source for all the peripherals, while FCY refers to the clock source for the CPU. Throughout this document FCY and FP are used interchangeably, except in the case of Doze mode. FP and FCY will be different when Doze mode is used in any ratio other than 1:1, which is the default.

REGISTER 10-1: PMD1: PERIPHERAL MODULE DISABLE CONTROL REGISTER 1 (CONTINUED)

bit 2	**Unimplemented:** Read as '0'
bit 1	**C1MD:** ECAN1 Module Disable bit 1 = ECAN1 module is disabled 0 = ECAN1 module is enabled
bit 0	**AD1MD:** ADC1 Module Disable bit 1 = ADC1 module is disabled 0 = ADC1 module is enabled

REGISTER 11-23: RPOR2: PERIPHERAL PIN SELECT OUTPUT REGISTER 2

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	RP5R<4:0>				
bit 15							bit 8

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	RP4R<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 15-13 **Unimplemented:** Read as ‘0’

bit 12-8 **RP5R<4:0>:** Peripheral Output Function is Assigned to RP5 Output Pin bits (see Table 11-2 for peripheral function numbers)

bit 7-5 **Unimplemented:** Read as ‘0’

bit 4-0 **RP4R<4:0>:** Peripheral Output Function is Assigned to RP4 Output Pin bits (see Table 11-2 for peripheral function numbers)

REGISTER 11-24: RPOR3: PERIPHERAL PIN SELECT OUTPUT REGISTER 3

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	RP7R<4:0>				
bit 15							bit 8

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	RP6R<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 15-13 **Unimplemented:** Read as ‘0’

bit 12-8 **RP7R<4:0>:** Peripheral Output Function is Assigned to RP7 Output Pin bits (see Table 11-2 for peripheral function numbers)

bit 7-5 **Unimplemented:** Read as ‘0’

bit 4-0 **RP6R<4:0>:** Peripheral Output Function is Assigned to RP6 Output Pin bits (see Table 11-2 for peripheral function numbers)

REGISTER 19-2: I2CxSTAT: I2Cx STATUS REGISTER

R-0, HSC	R-0, HSC	U-0	U-0	U-0	R/C-0, HS	R-0, HSC	R-0, HSC
ACKSTAT	TRSTAT	—	—	—	BCL	GCSTAT	ADD10
bit 15							bit 8

R/C-0, HS	R/C-0, HS	R-0, HSC	R/C-0, HSC	R/C-0, HSC	R-0, HSC	R-0, HSC	R-0, HSC
IWCOL	I2COV	D_A	P	S	R_W	RBF	TBF
bit 7							bit 0

Legend:	C = Clear only bit	U = Unimplemented bit, read as ‘0’	
R = Readable bit	W = Writable bit	HS = Set in hardware	HSC = Hardware set/cleared
-n = Value at POR	‘1’ = Bit is set	‘0’ = Bit is cleared	x = Bit is unknown

bit 15 **ACKSTAT:** Acknowledge Status bit
(when operating as I^{2}C™ master, applicable to master transmit operation)
1 = NACK received from slave
0 = ACK received from slave
Hardware set or clear at end of slave Acknowledge.

bit 14 **TRSTAT:** Transmit Status bit (when operating as I^{2}C master, applicable to master transmit operation)
1 = Master transmit is in progress (8 bits + ACK)
0 = Master transmit is not in progress
Hardware set at beginning of master transmission. Hardware clear at end of slave Acknowledge.

bit 13-11 **Unimplemented:** Read as ‘0’

bit 10 **BCL:** Master Bus Collision Detect bit
1 = A bus collision has been detected during a master operation
0 = No collision
Hardware set at detection of bus collision.

bit 9 **GCSTAT:** General Call Status bit
1 = General call address was received
0 = General call address was not received
Hardware set when address matches general call address. Hardware clear at Stop detection.

bit 8 **ADD10:** 10-bit Address Status bit
1 = 10-bit address was matched
0 = 10-bit address was not matched
Hardware set at match of 2nd byte of matched 10-bit address. Hardware clear at Stop detection.

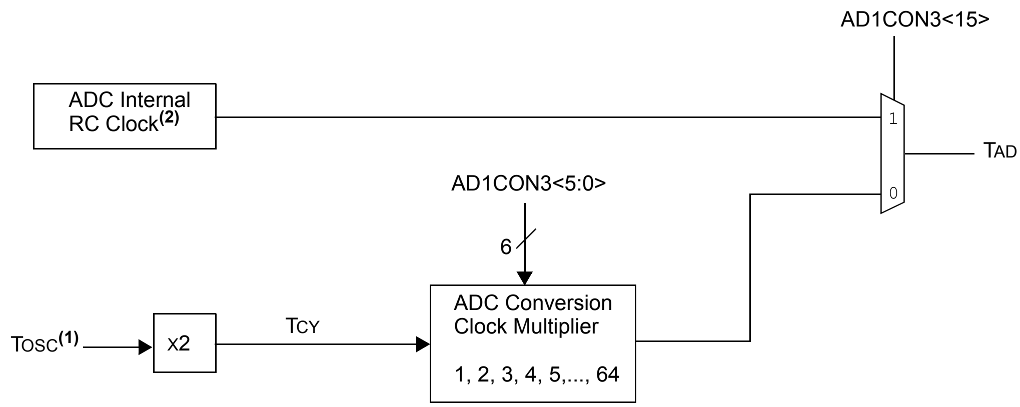
bit 7 **IWCOL:** Write Collision Detect bit
1 = An attempt to write the I2CxTRN register failed because the I^{2}C module is busy
0 = No collision
Hardware set at occurrence of write to I2CxTRN while busy (cleared by software).

bit 6 **I2COV:** Receive Overflow Flag bit
1 = A byte was received while the I2CxRCV register is still holding the previous byte
0 = No overflow
Hardware set at attempt to transfer I2CxRSR to I2CxRCV (cleared by software).

bit 5 **D_A:** Data/Address bit (when operating as I^{2}C slave)
1 = Indicates that the last byte received was data
0 = Indicates that the last byte received was device address
Hardware clear at device address match. Hardware set by reception of slave byte.

bit 4 **P:** Stop bit
1 = Indicates that a Stop bit has been detected last
0 = Stop bit was not detected last
Hardware set or clear when Start, Repeated Start or Stop detected.

FIGURE 22-3: ADC CONVERSION CLOCK PERIOD BLOCK DIAGRAM



Note 1: Refer to Figure 9-2 for the derivation of Fosc when the PLL is enabled. If the PLL is not used, Fosc is equal to the clock source frequency. Tosc = 1/Fosc.

2: See the ADC electrical characteristics for the exact RC clock value.

30.0 DEVELOPMENT SUPPORT

The PIC® microcontrollers and dsPIC® digital signal controllers are supported with a full range of software and hardware development tools:

- Integrated Development Environment
  - MPLAB® IDE Software
- Compilers/Assemblers/Linkers
  - MPLAB C Compiler for Various Device Families
  - HI-TECH C for Various Device Families
  - MPASM™ Assembler
  - MPLINK™ Object Linker/ MPLIB™ Object Librarian
  - MPLAB Assembler/Linker/Librarian for Various Device Families
- Simulators
  - MPLAB SIM Software Simulator
- Emulators
  - MPLAB REAL ICE™ In-Circuit Emulator
- In-Circuit Debuggers
  - MPLAB ICD 3
  - PICkit™ 3 Debug Express
- Device Programmers
  - PICkit™ 2 Programmer
  - MPLAB PM3 Device Programmer
- Low-Cost Demonstration/Development Boards, Evaluation Kits, and Starter Kits

30.1 MPLAB Integrated Development Environment Software

The MPLAB IDE software brings an ease of software development previously unseen in the 8/16/32-bit microcontroller market. The MPLAB IDE is a Windows® operating system-based application that contains:

- A single graphical interface to all debugging tools
  - Simulator
  - Programmer (sold separately)
  - In-Circuit Emulator (sold separately)
  - In-Circuit Debugger (sold separately)
- A full-featured editor with color-coded context
- A multiple project manager
- Customizable data windows with direct edit of contents
- High-level source code debugging
- Mouse over variable inspection
- Drag and drop variables from source to watch windows
- Extensive on-line help
- Integration of select third party tools, such as IAR C Compilers

The MPLAB IDE allows you to:

- Edit your source files (either C or assembly)
- One-touch compile or assemble, and download to emulator and simulator tools (automatically updates all project information)
- Debug using:
  - Source files (C or assembly)
  - Mixed C and assembly
  - Machine code

MPLAB IDE supports multiple debugging tools in a single development paradigm, from the cost-effective simulators, through low-cost in-circuit debuggers, to full-featured emulators. This eliminates the learning curve when upgrading to tools with increased flexibility and power.

TABLE 31-5: DC CHARACTERISTICS: OPERATING CURRENT (IDD)

DC CHARACTERISTICS			Standard Operating Conditions: 3.0V to 3.6V (unless otherwise stated) Operating temperature -40°C ≤ TA ≤ + 85°C for Industrial -40°C ≤ TA ≤ +125°C for Extended			
Parameter No.[3]	Typical[2]	Max	Units	Conditions		
Operating Current (IDD)[1]						
DC20d	18	21	mA	-40°C	3.3V	10 MIPS
DC20a	18	22	mA	+25°C		
DC20b	18	22	mA	+85°C		
DC20c	18	25	mA	+125°C		
DC21d	30	35	mA	-40°C	3.3V	16 MIPS
DC21a	30	34	mA	+25°C		
DC21b	30	34	mA	+85°C		
DC21c	30	36	mA	+125°C		
DC22d	34	42	mA	-40°C	3.3V	20 MIPS
DC22a	34	41	mA	+25°C		
DC22b	34	42	mA	+85°C		
DC22c	35	44	mA	+125°C		
DC23d	49	58	mA	-40°C	3.3V	30 MIPS
DC23a	49	57	mA	+25°C		
DC23b	49	57	mA	+85°C		
DC23c	49	60	mA	+125°C		
DC24d	63	75	mA	-40°C	3.3V	40 MIPS
DC24a	63	74	mA	+25°C		
DC24b	63	74	mA	+85°C		
DC24c	63	76	mA	+125°C		

Note 1: IDD is primarily a function of the operating voltage and frequency. Other factors, such as I/O pin loading and switching rate, oscillator type, internal code execution pattern and temperature, also have an impact on the current consumption. The test conditions for all IDD measurements are as follows:

- Oscillator is configured in EC mode, no PLL until 10 MIPS, OSC1 is driven with external square wave from rail-to-rail (EC clock overshoot/undershoot < 250 mV required)
- CLKO is configured as an I/O input pin in the Configuration word
- All I/O pins are configured as inputs and pulled to VSS
- $\overline{\text{MCLR}}$ = VDD, WDT and FSCM are disabled
- CPU, SRAM, program memory and data memory are operational
- No peripheral modules are operating; however, every peripheral is being clocked (defined PMDx bits are set to zero)
- CPU executing `while(1)` statement
- JTAG is disabled

2: Data in “Typ” column is at 3.3V, +25ºC unless otherwise stated.

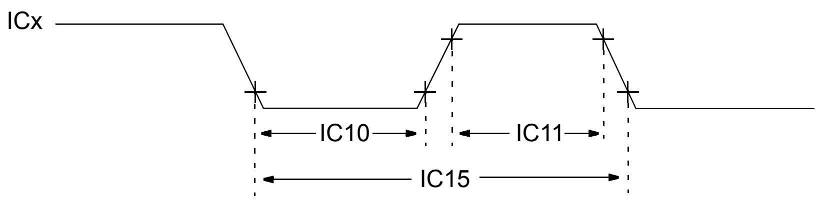
3: These parameters are characterized but not tested in manufacturing.

TABLE 31-21: RESET, WATCHDOG TIMER, OSCILLATOR START-UP TIMER, POWER-UP TIMER TIMING REQUIREMENTS

AC CHARACTERISTICS			Standard Operating Conditions: 3.0V to 3.6V (unless otherwise stated) Operating temperature -40°C ≤ TA ≤ +85°C for Industrial -40°C ≤ TA ≤ +125°C for Extended				
Param No.	**Symbol**	**Characteristic[1]**	**Min**	**Typ[2]**	**Max**	**Units**	**Conditions**
SY10	TMCL	$\overline{\text{MCLR}}$ Pulse Width (low)	2	—	—	μs	-40°C to +85°C
SY11	TPWRT	Power-up Timer Period	—	2 4 8 16 32 64 128	—	ms	-40°C to +85°C User programmable
SY12	TPOR	Power-on Reset Delay	3	10	30	μs	-40°C to +85°C
SY13	TIOZ	I/O High-Impedance from $\overline{\text{MCLR}}$ Low or Watchdog Timer Reset	0.68	0.72	1.2	μs	—
SY20	TWDT1	Watchdog Timer Time-out Period	—	—	—	—	See **Section 28.4 "Watchdog Timer (WDT)"** and LPRC specification F21 (Table 31-19)
SY30	TOST	Oscillator Start-up Time	—	1024 TOSC	—	—	TOSC = OSC1 period
SY35	TFSCM	Fail-Safe Clock Monitor Delay	—	500	900	μs	-40°C to +85°C

Note 1: These parameters are characterized but not tested in manufacturing.
2: Data in "Typ" column is at 3.3V, 25°C unless otherwise stated.

FIGURE 31-7: INPUT CAPTURE (CAPx) TIMING CHARACTERISTICS



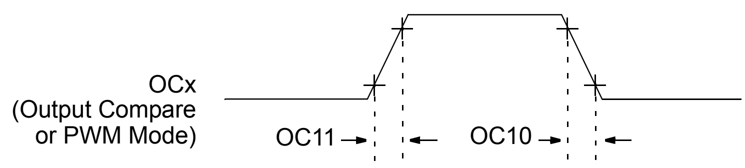
Note: Refer to Figure 31-1 for load conditions.

TABLE 31-26: INPUT CAPTURE TIMING REQUIREMENTS

AC CHARACTERISTICS			Standard Operating Conditions: 3.0V to 3.6V (unless otherwise stated) Operating temperature -40°C ≤ TA ≤ +85°C for Industrial -40°C ≤ TA ≤ +125°C for Extended				
Param No.	**Symbol**	**Characteristic(1)**		**Min**	**Max**	**Units**	**Conditions**
IC10	TccL	ICx Input Low Time	No Prescaler	0.5 TCY + 20	—	ns	
			With Prescaler	10	—	ns	
IC11	TccH	ICx Input High Time	No Prescaler	0.5 TCY + 20	—	ns	
			With Prescaler	10	—	ns	
IC15	TccP	ICx Input Period		(TCY + 40)/N	—	ns	N = prescale value (1, 4, 16)

Note 1: These parameters are characterized but not tested in manufacturing.

FIGURE 31-8: OUTPUT COMPARE MODULE (OCx) TIMING CHARACTERISTICS



Note: Refer to Figure 31-1 for load conditions.

TABLE 31-27: OUTPUT COMPARE MODULE TIMING REQUIREMENTS

AC CHARACTERISTICS			Standard Operating Conditions: 3.0V to 3.6V (unless otherwise stated) Operating temperature -40°C ≤ TA ≤ +85°C for Industrial -40°C ≤ TA ≤ +125°C for Extended				
Param No.	**Symbol**	**Characteristic(1)**	**Min**	**Typ**	**Max**	**Units**	**Conditions**
OC10	TccF	OCx Output Fall Time	—	—	—	ns	See parameter D032
OC11	TccR	OCx Output Rise Time	—	—	—	ns	See parameter D031

Note 1: These parameters are characterized but not tested in manufacturing.

FIGURE 31-24: I2Cx BUS START/STOP BITS TIMING CHARACTERISTICS (SLAVE MODE)

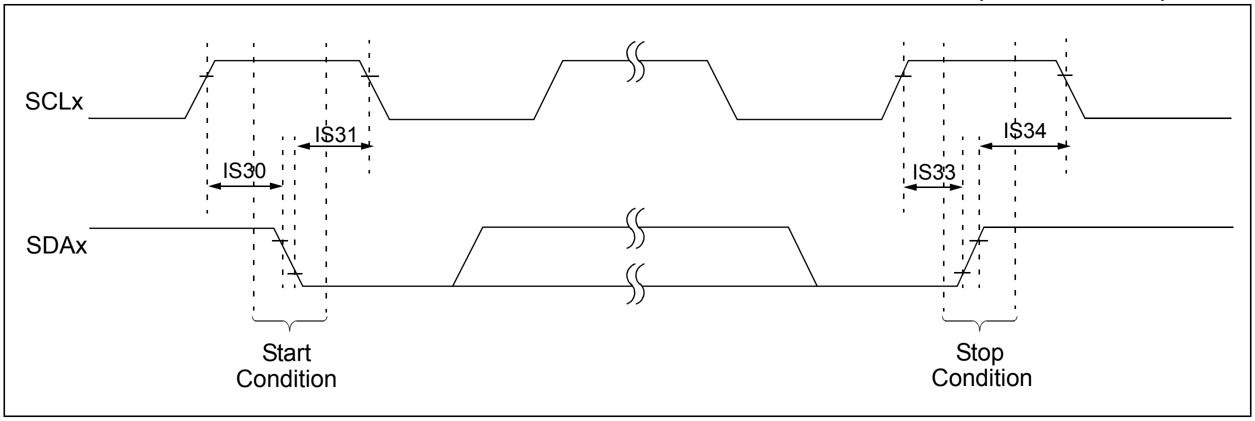


FIGURE 31-25: I2Cx BUS DATA TIMING CHARACTERISTICS (SLAVE MODE)

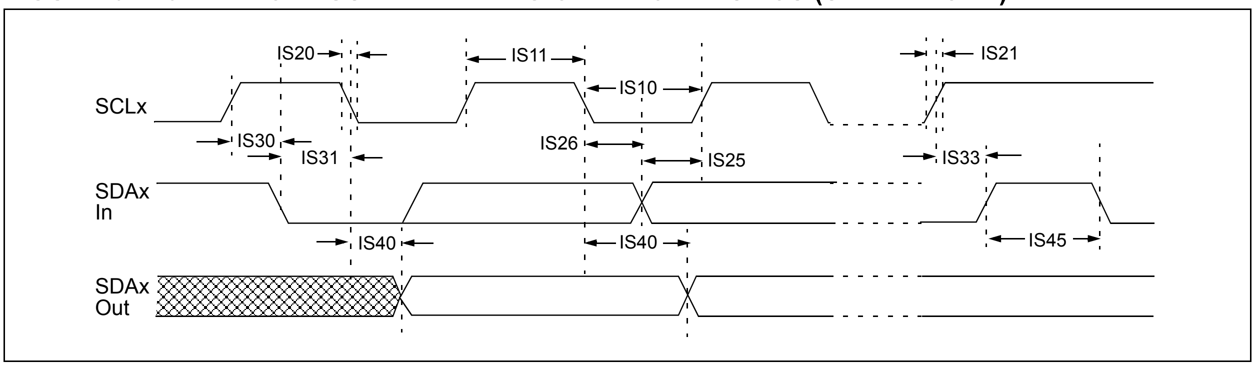


TABLE 31-45: ADC MODULE SPECIFICATIONS (10-BIT MODE)

AC CHARACTERISTICS			Standard Operating Conditions: 3.0V to 3.6V (unless otherwise stated) Operating temperature -40°C ≤ TA ≤ +85°C for Industrial -40°C ≤ TA ≤ +125°C for Extended				
Param No.	**Symbol**	**Characteristic**	**Min**	**Typ**	**Max**	**Units**	**Conditions**
ADC Accuracy (10-bit Mode) – Measurements with external VREF+/VREF-							
AD20b	Nr	Resolution[1]	10 data bits			bits	
AD21b	INL	Integral Nonlinearity	-1.5	—	+1.5	LSb	VINL = AVSS = VREFL = 0V, AVDD = VREFH = 3.6V
AD22b	DNL	Differential Nonlinearity	>-1	—	<1	LSb	VINL = AVSS = VREFL = 0V, AVDD = VREFH = 3.6V
AD23b	GERR	Gain Error	—	3	6	LSb	VINL = AVSS = VREFL = 0V, AVDD = VREFH = 3.6V
AD24b	EOFF	Offset Error	—	2	5	LSb	VINL = AVSS = VREFL = 0V, AVDD = VREFH = 3.6V
AD25b	—	Monotonicity	—	—	—	—	Guaranteed
ADC Accuracy (10-bit Mode) – Measurements with internal VREF+/VREF-							
AD20b	Nr	Resolution[1]	10 data bits			bits	
AD21b	INL	Integral Nonlinearity	-1	—	+1	LSb	VINL = AVSS = 0V, AVDD = 3.6V
AD22b	DNL	Differential Nonlinearity	>-1	—	<1	LSb	VINL = AVSS = 0V, AVDD = 3.6V
AD23b	GERR	Gain Error	3	7	15	LSb	VINL = AVSS = 0V, AVDD = 3.6V
AD24b	EOFF	Offset Error	1.5	3	7	LSb	VINL = AVSS = 0V, AVDD = 3.6V
AD25b	—	Monotonicity	—	—	—	—	Guaranteed
Dynamic Performance (10-bit Mode)							
AD30b	THD	Total Harmonic Distortion	—	—	-64	dB	—
AD31b	SINAD	Signal to Noise and Distortion	57	58.5	—	dB	—
AD32b	SFDR	Spurious Free Dynamic Range	72	—	—	dB	—
AD33b	FNYQ	Input Signal Bandwidth	—	—	550	kHz	—
AD34b	ENOB	Effective Number of Bits	9.16	9.4	—	bits	—

Note 1: Injection currents > | 0 | can affect the ADC results by approximately 4-6 counts.