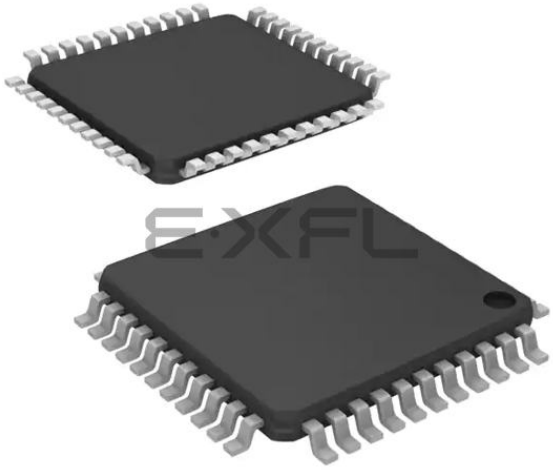
# Microchip Technology - DSPIC33EP128GM604-H/PT Datasheet



**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 | - |
| Connectivity | CANbus, I²C, IrDA, LINbus, QEI, SPI, UART/USART |
| Peripherals | Brown-out Detect/Reset, DMA, I²S, Motor Control PWM, POR, PWM, WDT |
| Number of I/O | 35 |
| Program Memory Size | 128KB (43K x 24) |
| Program Memory Type | FLASH |
| EEPROM Size | - |
| RAM Size | 16K x 8 |
| Voltage - Supply (Vcc/Vdd) | - |
| Data Converters | A/D 18x10b/12b |
| Oscillator Type | Internal |
| Operating Temperature | -40°C ~ 150°C (TA) |
| Mounting Type | Surface Mount |
| Package / Case | 44-TQFP |
| Supplier Device Package | 44-TQFP (10x10) |
| Purchase URL | https://www.e-xfl.com/product-detail/microchip-technology/dspic33ep128gm604-h-pt |

**NOTES:**

**FIGURE 2-5: SINGLE-PHASE SYNCHRONOUS BUCK CONVERTER**

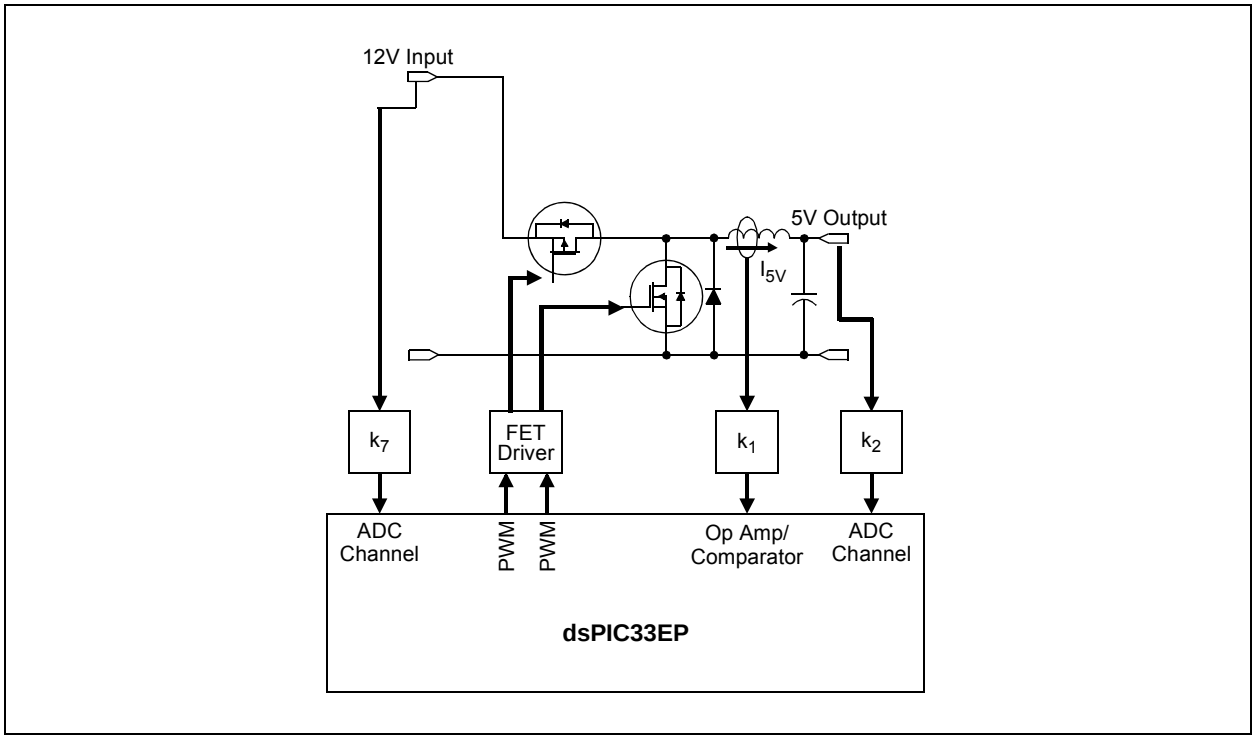
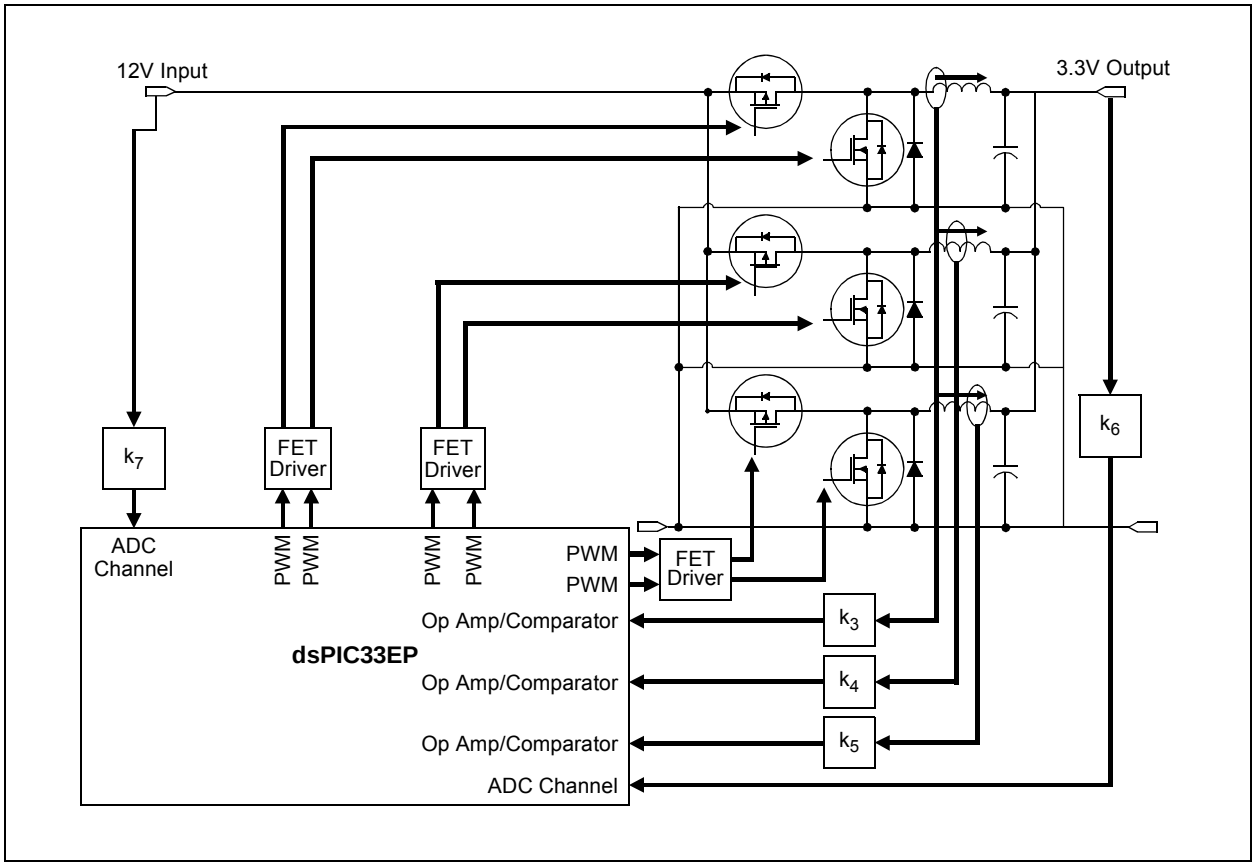



**FIGURE 2-6: MULTIPHASE SYNCHRONOUS BUCK CONVERTER**



## 3.5 Programmer's Model

The programmer's model for the dsPIC33EPXXXGM3XX/6XX/7XX devices is shown in Figure 3-2. All registers in the programmer's model are memory-mapped and can be manipulated directly by instructions. Table 3-1 lists a description of each register.

In addition to the registers contained in the programmer's model, the dsPIC33EPXXXGM3XX/6XX/7XX devices contain control registers for Modulo Addressing and Bit-Reversed Addressing, and interrupts. These registers are described in subsequent sections of this document.

All registers associated with the programmer's model are memory-mapped, as shown in Table 4-1.

**TABLE 3-1: PROGRAMMER'S MODEL REGISTER DESCRIPTIONS**

| Register(s) Name | Description |
|---|---|
| W0 through W15 | Working Register Array |
| ACCA, ACCB | 40-Bit DSP Accumulators |
| PC | 23-Bit Program Counter |
| SR | ALU and DSP Engine Status register |
| SPLIM | Stack Pointer Limit Value register |
| TBLPAG | Table Memory Page Address register |
| DSRPAG | Extended Data Space (EDS) Read Page register |
| DSWPAG | Extended Data Space (EDS) Write Page register |
| RCOUNT | REPEAT Loop Count register |
| DCOUNT | DO Loop Count register |
| DOSTARTH(1), DOSTARTL(1) | DO Loop Start Address register (High and Low) |
| DOENDH, DOENDL | DO Loop End Address register (High and Low) |
| CORCON | Contains DSP Engine, DO Loop Control and Trap Status bits |

**Note 1:** The DOSTARTH and DOSTARTL registers are read-only.

## 4.2 Data Address Space

The dsPIC33EPXXXGM3XX/6XX/7XX CPU has a separate 16-bit wide data memory space. The Data Space is accessed using separate Address Generation Units (AGUs) for read and write operations. The data memory maps, which are presented by device family and memory size, are shown in Figure 4-5 through Figure 4-7.

All Effective Addresses (EAs) in the data memory space are 16 bits wide and point to bytes within the Data Space. This arrangement gives a Base Data Space address range of 64 Kbytes or 32K words.

The Base Data Space address is used in conjunction with a Data Space Read or Write Page register (DSRPAG or DSWPAG) to form an Extended Data Space, which has a total address range of 16 Mbytes.

dsPIC33EPXXXGM3XX/6XX/7XX devices implement up to 52 Kbytes of data memory (4 Kbytes of data memory for Special Function Registers and up to 48 Kbytes of data memory for RAM). If an EA points to a location outside of this area, an all zero word or byte is returned.

### 4.2.1 DATA SPACE WIDTH

The data memory space is organized in byte-addressable, 16-bit wide blocks. Data is aligned in data memory and registers as 16-bit words, but all Data Space EAs resolve to bytes. The Least Significant Bytes (LSBs) of each word have even addresses, while the Most Significant Bytes (MSBs) have odd addresses.

### 4.2.2 DATA MEMORY ORGANIZATION AND ALIGNMENT

To maintain backward compatibility with PIC® MCU devices and improve Data Space memory usage efficiency, the dsPIC33EPXXXGM3XX/6XX/7XX instruction set supports both word and byte operations. As a consequence of byte accessibility, all Effective Address calculations are internally scaled to step through word-aligned memory. For example, the core recognizes that Post-Modified Register Indirect Addressing mode [Ws++] results in a value of Ws + 1 for byte operations and Ws + 2 for word operations.

A data byte read, reads the complete word that contains the byte, using the LSb of any EA to determine which byte to select. The selected byte is placed onto the LSB of the data path. That is, data memory and registers are organized as two parallel, byte-wide entities with shared (word) address decode but separate write lines. Data byte writes only write to the corresponding side of the array or register that matches the byte address.

All word accesses must be aligned to an even address. Misaligned word data fetches are not supported, so care must be taken when mixing byte and word operations, or translating from 8-bit MCU code. If a misaligned read or write is attempted, an address error trap is generated. If the error occurred on a read, the instruction underway is completed. If the error occurred on a write, the instruction is executed but the write does not occur. In either case, a trap is then executed, allowing the system and/or user application to examine the machine state prior to execution of the address Fault.

All byte loads into any W register are loaded into the LSB; the MSB is not modified.

A Sign-Extend (SE) instruction is provided to allow user applications to translate 8-bit signed data to 16-bit signed values. Alternatively, for 16-bit unsigned data, user applications can clear the MSB of any W register by executing a Zero-Extend (ZE) instruction on the appropriate address.

### 4.2.3 SFR SPACE

The first 4 Kbytes of the Near Data Space, from 0x0000 to 0x0FFF, is primarily occupied by Special Function Registers (SFRs). These are used by the dsPIC33EPXXXGM3XX/6XX/7XX core and peripheral modules for controlling the operation of the device.

SFRs are distributed among the modules that they control and are generally grouped together by module. Much of the SFR space contains unused addresses; these are read as '0'.

| | |
|---|---|
| **Note:** | The actual set of peripheral features and interrupts varies by the device. Refer to the corresponding device tables and pinout diagrams for device-specific information. |

### 4.2.4 NEAR DATA SPACE

The 8-Kbyte area, between 0x0000 and 0x1FFF, is referred to as the Near Data Space. Locations in this space are directly addressable through a 13-bit absolute address field within all memory direct instructions. Additionally, the whole Data Space is addressable using MOV instructions, which support Memory Direct Addressing mode with a 16-bit address field, or by using Indirect Addressing mode using a Working register as an Address Pointer.

**TABLE 4-3: INTERRUPT CONTROLLER REGISTER MAP FOR dsPIC33EPXXXGM3XX DEVICES (CONTINUED)**

| 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 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| IPC11 | 0856 | — | T6IP2 | T6IP1 | T6IP0 | — | — | — | — | — | PMPIP2(1) | PMPIP1(1) | PMPIP0(1) | — | OC8IP2 | OC8IP1 | OC8IP0 | 4444 |
| IPC12 | 0858 | — | T8IP2 | T8IP1 | T8IP0 | — | MI2C2IP2 | MI2C2IP1 | MI2C2IP0 | — | SI2C2IP2 | SI2C2IP1 | SI2C2IP0 | — | T7IP2 | T7IP1 | T7IP0 | 4444 |
| IPC13 | 085A | — | — | — | — | — | INT4IP2 | INT4IP1 | INT4IP0 | — | INT3IP2 | INT3IP1 | INT3IP0 | — | T9IP2 | T9IP1 | T9IP0 | 4444 |
| IPC14 | 085C | — | DCIEIP2 | DCIEIP1 | DCIEIP0 | — | QEI1IP2 | QEI1IP2 | QEI1IP0 | — | PCEPIP2 | PCEPIP1 | PCEPIP0 | — | — | — | — | 4444 |
| IPC15 | 085E | — | FLT1IP2 | FLT1IP1 | FLT1IP0 | — | RTCCIP2(2) | RTCCIP1(2) | RTCCIP0(2) | — | — | — | — | — | DCIIP2 | DCIIP1 | DCIIP0 | 0404 |
| IPC16 | 0860 | — | CRCIP2 | CRCIP1 | CRCIP0 | — | U2EIP2 | U2EIP1 | U2EIP0 | — | U1EIP2 | U1EIP1 | U1EIP0 | — | FLT2IP2 | FLT2IP1 | FLT2IP0 | 4440 |
| IPC18 | 0864 | — | C2TXIP2 | C2TXIP1 | C2TXIP0 | — | FLT3IP2 | FLT3IP1 | FLT3IP0 | — | PCESIP2 | PCESIP1 | PCESIP0 | — | — | — | — | 4040 |
| IPC19 | 0866 | — | — | — | — | — | — | — | — | — | CTMUIP2 | CTMUIP1 | CTMUIP0 | — | FLT4IP2 | FLT4IP1 | FLT4IP0 | 0004 |
| IPC20 | 0868 | — | U3TXIP2 | U3TXIP1 | U3TXIP0 | — | U3RXIP2 | U3RXIP1 | U3RXIP0 | — | U3EIP2 | U3EIP1 | U3EIP0 | — | — | — | — | 0000 |
| IPC21 | 086A | — | U4EIP2 | U4EIP1 | U4EIP0 | — | — | — | — | — | — | — | — | — | — | — | — | 0000 |
| IPC22 | 086C | — | SPI3IP2 | SPI3IP1 | SPI3IP0 | — | SPI3EIP2 | SPI3EIP1 | SPI3EIP0 | — | U4TXIP2 | U4TXIP1 | U4TXIP0 | — | U4RXIP2 | U4RXIP1 | U4RXIP0 | 0000 |
| IPC23 | 086E | — | PGC2IP2 | PGC2IP1 | PGC2IP0 | — | PWM1IP2 | PWM1IP1 | PWM1IP0 | — | — | — | — | — | — | — | — | 4400 |
| IPC24 | 0870 | — | PWM6IP2 | PWM6IP1 | PWM6IP0 | — | PWM5IP2 | PWM5IP1 | PWM5IP0 | — | PWM4IP2 | PWM4IP1 | PWM4IP0 | — | PWM3IP2 | PWM3IP1 | PWM3IP0 | 4444 |
| IPC35 | 0886 | — | JTAGIP2 | JTAGIP1 | JTAGIP0 | — | ICDIP2 | ICDIP1 | ICDIP0 | — | — | — | — | — | — | — | — | 4400 |
| IPC36 | 0888 | — | PTG0IP2 | PTG0IP1 | PTG0IP0 | — | PTGWDTIP2 | PTGWDTIP1 | PTGWDTIP0 | — | PTGSTEPIP2 | PTGSTEPIP1 | PTGSTEPIP0 | — | — | — | — | 4440 |
| IPC37 | 088A | — | — | — | — | — | PTG3IP2 | PTG3IP1 | PTG3IP0 | — | PTG2IP2 | PTG2IP1 | PTG2IP0 | — | PTG1IP2 | PTG1IP1 | PTG1IP0 | 0444 |
| INTTREG | 08C8 | — | — | — | — | ILR3 | ILR2 | ILR1 | ILR0 | VECNUM7 | VECNUM6 | VECNUM5 | VECNUM4 | VECNUM3 | VECNUM2 | VECNUM1 | VECNUM0 | 0000 |

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

**Note 1:** The PMPIF/PMPIE/PMPIPx flags are not available on 44-pin devices.

**2:** The RTCCIF/RTCCIE/RTCCIPx flags are not available on 44-pin devices.

**TABLE 4-19: UART3 AND UART4 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 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| U3MODE | 0250 | UARTEN | — | USIDL | IREN | RTSMD | — | UEN1 | UEN0 | WAKE | LPBACK | ABAUD | URXINV | BRGH | PDSEL1 | PDSEL0 | STSEL | 0000 |
| U3STA | 0252 | UTXISEL1 | UTXINV | UTXISEL0 | — | UTXBRK | UTXEN | UTXBF | TRMT | URXISEL1 | URXISEL0 | ADDEN | RIDLE | PERR | FERR | OERR | URXDA | 0110 |
| U3TXREG | 0254 | — | — | — | — | — | — | — | UART3 Transmit Register | | | | | | | | | xxxx |
| U3RXREG | 0256 | — | — | — | — | — | — | — | UART3 Receive Register | | | | | | | | | 0000 |
| U3BRG | 0258 | Baud Rate Generator Prescaler | | | | | | | | | | | | | | | | 0000 |
| U4MODE | 02B0 | UARTEN | — | USIDL | IREN | RTSMD | — | UEN1 | UEN0 | WAKE | LPBACK | ABAUD | URXINV | BRGH | PDSEL1 | PDSEL0 | STSEL | 0000 |
| U4STA | 02B2 | UTXISEL1 | UTXINV | UTXISEL0 | — | UTXBRK | UTXEN | UTXBF | TRMT | URXISEL1 | URXISEL0 | ADDEN | RIDLE | PERR | FERR | OERR | URXDA | 0110 |
| U4TXREG | 02B4 | — | — | — | — | — | — | — | UART4 Transmit Register | | | | | | | | | xxxx |
| U4RXREG | 02B6 | — | — | — | — | — | — | — | UART4 Receive Register | | | | | | | | | 0000 |
| U4BRG | 02B8 | Baud Rate Generator Prescaler | | | | | | | | | | | | | | | | 0000 |

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

**TABLE 4-20: SPI1, SPI2 AND SPI3 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 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SPI1STAT | 0240 | SPIEN | — | SPISIDL | — | — | SPIBEC2 | SPIBEC1 | SPIBEC0 | SRMPT | SPIROV | SRXMPT | SISEL2 | SISEL1 | SISEL0 | SPITBF | SPIRBF | 0000 |
| SPI1CON1 | 0242 | — | — | — | DISSCK | DISSDO | MODE16 | SMP | CKE | SSEN | CKP | MSTEN | SPRE2 | SPRE1 | SPRE0 | PPRE1 | PPRE0 | 0000 |
| SPI1CON2 | 0244 | FRMEN | SPIFSD | FRMPOL | — | — | — | — | — | — | — | — | — | — | — | FRMDLY | SPIBEN | 0000 |
| SPI1BUF | 0248 | SPI1 Transmit and Receive Buffer Register | | | | | | | | | | | | | | | | 0000 |
| SPI2STAT | 0260 | SPIEN | — | SPISIDL | — | — | SPIBEC2 | SPIBEC1 | SPIBEC0 | SRMPT | SPIROV | SRXMPT | SISEL2 | SISEL1 | SISEL0 | SPITBF | SPIRBF | 0000 |
| SPI2CON1 | 0262 | — | — | — | DISSCK | DISSDO | MODE16 | SMP | CKE | SSEN | CKP | MSTEN | SPRE2 | SPRE1 | SPRE0 | PPRE1 | PPRE0 | 0000 |
| SPI2CON2 | 0264 | FRMEN | SPIFSD | FRMPOL | — | — | — | — | — | — | — | — | — | — | — | FRMDLY | SPIBEN | 0000 |
| SPI2BUF | 0268 | SPI2 Transmit and Receive Buffer Register | | | | | | | | | | | | | | | | 0000 |
| SPI3STAT | 02A0 | SPIEN | — | SPISIDL | — | — | SPIBEC2 | SPIBEC1 | SPIBEC0 | SRMPT | SPIROV | SRXMPT | SISEL2 | SISEL1 | SISEL0 | SPITBF | SPIRBF | 0000 |
| SPI3CON1 | 02A2 | — | — | — | DISSCK | DISSDO | MODE16 | SMP | CKE | SSEN | CKP | MSTEN | SPRE2 | SPRE1 | SPRE0 | PPRE1 | PPRE0 | 0000 |
| SPI3CON2 | 02A4 | FRMEN | SPIFSD | FRMPOL | — | — | — | — | — | — | — | — | — | — | — | FRMDLY | SPIBEN | 0000 |
| SPI3BUF | 02A8 | SPI3 Transmit and Receive Buffer Register | | | | | | | | | | | | | | | | 0000 |

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

**REGISTER 7-4: INTCON2: INTERRUPT CONTROL REGISTER 2**

| R/W-1 | R/W-0 | R/W-0 | U-0 | U-0 | U-0 | U-0 | U-0 |
|---|---|---|---|---|---|---|---|
| GIE | DISI | SWTRAP | — | — | — | — | — |
| bit 15 | | | | | | | bit 8 |

| U-0 | U-0 | U-0 | U-0 | U-0 | R/W-0 | R/W-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| — | — | — | — | — | INT2EP | INT1EP | INT0EP |
| 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 **GIE:** Global Interrupt Enable bit
1 = Interrupts and associated IECx bits are enabled
0 = Interrupts are disabled, but traps are still enabled

bit 14 **DISI:** DISI Instruction Status bit
1 = DISI instruction is active
0 = DISI instruction is not active

bit 13 **SWTRAP:** Software Trap Status bit
1 = Software trap is enabled
0 = Software trap is disabled

bit 12-3 **Unimplemented:** Read as '0'

bit 2 **INT2EP:** External Interrupt 2 Edge Detect Polarity Select bit
1 = Interrupt on negative edge
0 = Interrupt on positive edge

bit 1 **INT1EP:** External Interrupt 1 Edge Detect Polarity Select bit
1 = Interrupt on negative edge
0 = Interrupt on positive edge

bit 0 **INT0EP:** External Interrupt 0 Edge Detect Polarity Select bit
1 = Interrupt on negative edge
0 = Interrupt on positive edge

**REGISTER 9-1: OSCCON: OSCILLATOR CONTROL REGISTER(1,3)**

| U-0 | R-0 | R-0 | R-0 | U-0 | R/W-y | R/W-y | R/W-y |
|---|---|---|---|---|---|---|---|
| — | COSC2 | COSC1 | COSC0 | — | NOSC2(2) | NOSC1(2) | NOSC0(2) |
| bit 15 | | | | | | | bit 8 |

| R/W-0 | R/W-0 | R-0 | U-0 | R/W-0 | U-0 | R/W-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| CLKLOCK | IOLOCK | LOCK | — | CF(5) | — | LPOSCEN | OSWEN |
| bit 7 | | | | | | | bit 0 |

| **Legend:** | y = Value set from Configuration bits on POR | | |
|---|---|---|---|
| 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 **COSC<2:0>:** Current Oscillator Selection bits (read-only)
111 = Fast RC Oscillator (FRC) with Divide-by-N
110 = Fast RC Oscillator (FRC) with Divide-by-16
101 = Low-Power RC Oscillator (LPRC)
100 = Secondary Oscillator (SOSC)(4)
011 = Primary Oscillator (MS, HS, EC) with PLL
010 = Primary Oscillator (MS, HS, EC)
001 = Fast RC Oscillator (FRC) Divided by N and PLL
000 = Fast RC Oscillator (FRC)

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

bit 10-8 **NOSC<2:0>:** New Oscillator Selection bits(2)
111 = Fast RC Oscillator (FRC) with Divide-by-N
110 = Fast RC Oscillator (FRC) with Divide-by-16
101 = Low-Power RC Oscillator (LPRC)
100 = Secondary Oscillator (SOSC)(4)
011 = Primary Oscillator (MS, HS, EC) with PLL
010 = Primary Oscillator (MS, HS, EC)
001 = Fast RC Oscillator (FRC) Divided by N and PLL
000 = Fast RC Oscillator (FRC)

bit 7 **CLKLOCK:** Clock Lock Enable bit
1 = If FCKSM0 = 1, then clock and PLL configurations are locked; if FCKSM0 = 0, then clock and PLL configurations may be modified
0 = Clock and PLL selections are not locked, configurations may be modified

bit 6 **IOLOCK:** I/O Lock Enable bit
1 = I/O lock is active
0 = I/O lock is not active

**Note 1:** Writes to this register require an unlock sequence. Refer to the *"dsPIC33/PIC24 Family Reference Manual"*, **"Oscillator"** (DS70580), available from the Microchip web site for details.

**2:** Direct clock switches between any primary oscillator mode with PLL and FRCPLL mode are not permitted. This applies to clock switches in either direction. In these instances, the application must switch to FRC mode as a transitional clock source between the two PLL modes.

**3:** This register resets only on a Power-on Reset (POR).

**4:** Secondary Oscillator (SOSC) selection is valid on 64-pin and 100-pin devices, and defaults to FRC/N on 44-pin devices.

**5:** Only '0' should be written to the CF bit in order to clear it. If a '1' is written to CF, it will have the same effect as a detected clock failure, including an oscillator fail trap.

**REGISTER 11-27: RPINR39: PERIPHERAL PIN SELECT INPUT REGISTER 39**

| U-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| — | DTCMP3R<6:0> | | | | | | |
| bit 15 | | | | | | | bit 8 |

| U-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| — | DTCMP2R<6:0> | | | | | | |
| 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 14-8 **DTCMP3R<6:0>:** Assign PWM Dead-Time Compensation Input 3 to the Corresponding RPn Pin bits (see Table 11-2 for input pin selection numbers)

1111100 = Input tied to RPI124
•
•
•
0000001 = Input tied to CMP1
0000000 = Input tied to Vss

bit 7 **Unimplemented:** Read as ‘0’

bit 6-0 **DTCMP2R<6:0>:** Assign PWM Dead-Time Compensation Input 2 to the Corresponding RPn Pin bits (see Table 11-2 for input pin selection numbers)

1111100 = Input tied to RPI124
•
•
•
0000001 = Input tied to CMP1
0000000 = Input tied to Vss

## 23.3 ADCx Control Registers

**REGISTER 23-1: ADxCON1: ADCx CONTROL REGISTER 1**

| R/W-0 | U-0 | R/W-0 | R/W-0 | U-0 | R/W-0 | R/W-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| ADON | — | ADSIDL | ADDMABM | — | AD12B | FORM1 | FORM0 |
| 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, HC, HS | R/C-0, HC, HS |
|---|---|---|---|---|---|---|---|
| SSRC2 | SSRC1 | SSRC0 | SSRCG | SIMSAM | ASAM | SAMP | DONE(2) |
| bit 7 | | | | | | | bit 0 |

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

bit 15 **ADON:** ADCx Operating Mode bit
1 = ADCx module is operating
0 = ADCx is off

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

bit 13 **ADSIDL:** ADCx Stop in Idle Mode bit
1 = Discontinues module operation when device enters Idle mode
0 = Continues module operation in Idle mode

bit 12 **ADDMABM:** ADCx DMA Buffer Build Mode bit
1 = DMA buffers are written in the order of conversion; the module provides an address to the DMA channel that is the same as the address used for the non-DMA stand-alone buffer
0 = DMA buffers are written in Scatter/Gather mode; the module provides a Scatter/Gather address to the DMA channel based on the index of the analog input and the size of the DMA buffer

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

bit 10 **AD12B:** 10-Bit or 12-Bit ADCx Operation Mode bit
1 = 12-bit, 1-channel ADCx operation
0 = 10-bit, 4-channel ADCx operation

bit 9-8 **FORM<1:0>:** Data Output Format bits
For 10-Bit Operation:
11 = Signed fractional (DOUT = sddd dddd dd00 0000, where s = .NOT.d<9>)
10 = Fractional (DOUT = dddd dddd dd00 0000)
01 = Signed integer (DOUT = ssss sssd dddd dddd, where s = .NOT.d<9>)
00 = Integer (DOUT = 0000 00dd dddd dddd)
For 12-Bit Operation:
11 = Signed fractional (DOUT = sddd dddd dddd 0000, where s = .NOT.d<11>)
10 = Fractional (DOUT = dddd dddd dddd 0000)
01 = Signed integer (DOUT = ssss sddd dddd dddd, where s = .NOT.d<11>)
00 = Integer (DOUT = 0000 dddd dddd dddd)

**Note 1:** See **Section 25.0 "Peripheral Trigger Generator (PTG) Module"** for information on this selection.
**2:** Do not clear the DONE bit in software if ADCx Sample Auto-Start bit is enabled (ASAM = 1).

**REGISTER 27-3: ALCFGRPT: ALARM CONFIGURATION 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 |
|---|---|---|---|---|---|---|---|
| ALRMEN | CHIME | AMASK3 | AMASK2 | AMASK1 | AMASK0 | ALRMPTR1 | ALRMPTR0 |
| 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 |
|---|---|---|---|---|---|---|---|
| ARPT7 | ARPT6 | ARPT5 | ARPT4 | ARPT3 | ARPT2 | ARPT1 | ARPT0 |
| 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 **ALRMEN:** Alarm Enable bit

1 = Alarm is enabled (cleared automatically after an alarm event whenever ARPT<7:0> = 0x00 and CHIME = 0)
0 = Alarm is disabled

bit 14 **CHIME:** Chime Enable bit

1 = Chime is enabled; ARPT<7:0> bits are allowed to roll over from 0x00 to 0xFF
0 = Chime is disabled; ARPT<7:0> bits stop once they reach 0x00

bit 13-10 **AMASK<3:0>:** Alarm Mask Configuration bits

0000 = Every half second
0001 = Every second
0010 = Every 10 seconds
0011 = Every minute
0100 = Every 10 minutes
0101 = Every hour
0110 = Once a day
0111 = Once a week
1000 = Once a month
1001 = Once a year (except when configured for February 29th, once every 4 years)
101x = Reserved – do not use
11xx = Reserved – do not use

bit 9-8 **ALRMPTR<1:0>:** Alarm Value Register Window Pointer bits

Points to the corresponding Alarm Value registers when reading the ALRMVAL register. The ALRMPTR<1:0> value decrements on every read or write of ALRMVAL until it reaches '00'.

bit 7-0 **ARPT<7:0>:** Alarm Repeat Counter Value bits

11111111 = Alarm will repeat 255 more times
•
•
•
00000000 = Alarm will not repeat

The counter decrements on any alarm event. The counter is prevented from rolling over from 0x00 to 0xFF unless CHIME = 1.

## 29.0 PROGRAMMABLE CYCLIC REDUNDANCY CHECK (CRC) GENERATOR

**Note 1:** This data sheet summarizes the features of the dsPIC33EPXXXGM3XX/6XX/7XX family of devices. It is not intended to be a comprehensive reference source. To complement the information in this data sheet, refer to the "*dsPIC33/PIC24 Family Reference Manual*", **"32-Bit Programmable Cyclic Redundancy Check (CRC)"** (DS70346), 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 programmable CRC generator offers the following features:

- User-Programmable (up to 32nd order) polynomial CRC equation
- Interrupt Output
- Data FIFO

The programmable CRC generator provides a hardware-implemented method of quickly generating checksums for various networking and security applications. It offers the following features:

- User-programmable CRC polynomial equation, up to 32 bits
- Programmable shift direction (little or big-endian)
- Independent data and polynomial lengths
- Configurable interrupt output
- Data FIFO

A simplified block diagram of the CRC generator is shown in Figure 29-1. A simple version of the CRC shift engine is shown in Figure 29-2.

**FIGURE 29-1: CRC BLOCK DIAGRAM**

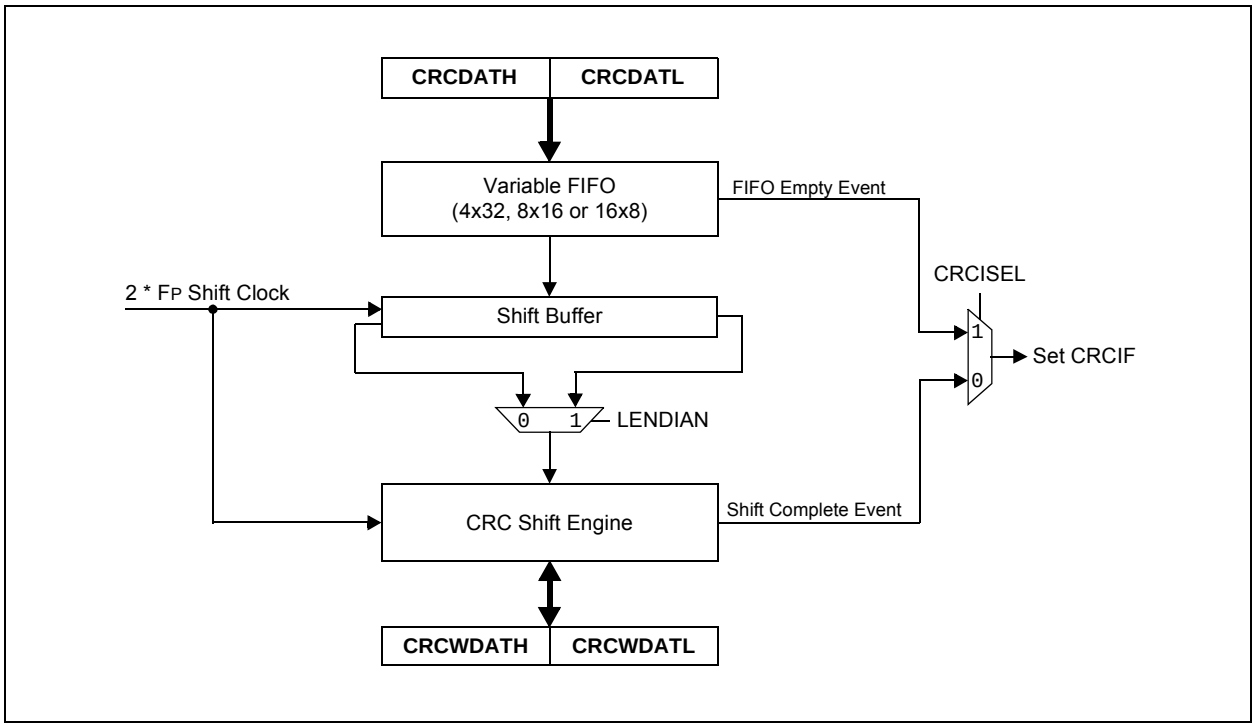


**TABLE 31-2: INSTRUCTION SET OVERVIEW (CONTINUED)**

| Base Instr # | Assembly Mnemonic | Assembly Syntax | Description | # of Words | # of Cycles | Status Flags Affected |
|---|---|---|---|---|---|---|
| 8 | BSW | BSW.C Ws,Wb | Write C bit to Ws<Wb> | 1 | 1 | None |
| | | BSW.Z Ws,Wb | Write Z bit to Ws<Wb> | 1 | 1 | None |
| 9 | BTG | BTG f,#bit4 | Bit Toggle f | 1 | 1 | None |
| | | BTG Ws,#bit4 | Bit Toggle Ws | 1 | 1 | None |
| 10 | BTSC | BTSC f,#bit4 | Bit Test f, Skip if Clear | 1 | 1 (2 or 3) | None |
| | | BTSC Ws,#bit4 | Bit Test Ws, Skip if Clear | 1 | 1 (2 or 3) | None |
| 11 | BTSS | BTSS f,#bit4 | Bit Test f, Skip if Set | 1 | 1 (2 or 3) | None |
| | | BTSS Ws,#bit4 | Bit Test Ws, Skip if Set | 1 | 1 (2 or 3) | None |
| 12 | BTST | BTST f,#bit4 | Bit Test f | 1 | 1 | Z |
| | | BTST.C Ws,#bit4 | Bit Test Ws to C | 1 | 1 | C |
| | | BTST.Z Ws,#bit4 | Bit Test Ws to Z | 1 | 1 | Z |
| | | BTST.C Ws,Wb | Bit Test Ws<Wb> to C | 1 | 1 | C |
| | | BTST.Z Ws,Wb | Bit Test Ws<Wb> to Z | 1 | 1 | Z |
| 13 | BTSTS | BTSTS f,#bit4 | Bit Test then Set f | 1 | 1 | Z |
| | | BTSTS.C Ws,#bit4 | Bit Test Ws to C, then Set | 1 | 1 | C |
| | | BTSTS.Z Ws,#bit4 | Bit Test Ws to Z, then Set | 1 | 1 | Z |
| 14 | CALL | CALL lit23 | Call subroutine | 2 | 4 | SFA |
| | | CALL Wn | Call indirect subroutine | 1 | 4 | SFA |
| | | CALL.L Wn | Call indirect subroutine (long address) | 1 | 4 | SFA |
| 15 | CLR | CLR f | f = 0x0000 | 1 | 1 | None |
| | | CLR WREG | WREG = 0x0000 | 1 | 1 | None |
| | | CLR Ws | Ws = 0x0000 | 1 | 1 | None |
| | | CLR Acc,Wx,Wxd,Wy,Wyd,AWB | Clear Accumulator | 1 | 1 | OA,OB,SA, SB |
| 16 | CLRWDT | CLRWDT | Clear Watchdog Timer | 1 | 1 | WDTO,Sleep |
| 17 | COM | COM f | $f = \overline{f}$ | 1 | 1 | N,Z |
| | | COM f,WREG | $WREG = \overline{f}$ | 1 | 1 | N,Z |
| | | COM Ws,Wd | $Wd = \overline{Ws}$ | 1 | 1 | N,Z |
| 18 | CP | CP f | Compare f with WREG | 1 | 1 | C,DC,N,OV,Z |
| | | CP Wb,#lit8 | Compare Wb with lit8 | 1 | 1 | C,DC,N,OV,Z |
| | | CP Wb,Ws | Compare Wb with Ws (Wb – Ws) | 1 | 1 | C,DC,N,OV,Z |
| 19 | CP0 | CP0 f | Compare f with 0x0000 | 1 | 1 | C,DC,N,OV,Z |
| | | CP0 Ws | Compare Ws with 0x0000 | 1 | 1 | C,DC,N,OV,Z |
| 20 | CPB | CPB f | Compare f with WREG, with Borrow | 1 | 1 | C,DC,N,OV,Z |
| | | CPB Wb,#lit8 | Compare Wb with lit8, with Borrow | 1 | 1 | C,DC,N,OV,Z |
| | | CPB Wb,Ws | Compare Wb with Ws, with Borrow ($Wb – Ws – \overline{C}$) | 1 | 1 | C,DC,N,OV,Z |
| 21 | CPSEQ | CPSEQ Wb,Wn | Compare Wb with Wn, skip if = | 1 | 1 (2 or 3) | None |
| | CPBEQ | CPBEQ Wb,Wn,Expr | Compare Wb with Wn, branch if = | 1 | 1 (5) | None |
| 22 | CPSGT | CPSGT Wb,Wn | Compare Wb with Wn, skip if > | 1 | 1 (2 or 3) | None |
| | CPBGT | CPBGT Wb,Wn,Expr | Compare Wb with Wn, branch if > | 1 | 1 (5) | None |
| 23 | CPSLT | CPSLT Wb,Wn | Compare Wb with Wn, skip if < | 1 | 1 (2 or 3) | None |
| | CPBLT | CPBLT Wb,Wn,Expr | Compare Wb with Wn, branch if < | 1 | 1 (5) | None |
| 24 | CPSNE | CPSNE Wb,Wn | Compare Wb with Wn, skip if ≠ | 1 | 1 (2 or 3) | None |
| | CPBNE | CPBNE Wb,Wn,Expr | Compare Wb with Wn, branch if ≠ | 1 | 1 (5) | None |

**Note:** Read and Read-Modify-Write (e.g., bit operations and logical operations) on non-CPU SFRs incur an additional instruction cycle.

## 32.2 MPLAB XC Compilers

The MPLAB XC Compilers are complete ANSI C compilers for all of Microchip's 8, 16 and 32-bit MCU and DSC devices. These compilers provide powerful integration capabilities, superior code optimization and ease of use. MPLAB XC Compilers run on Windows, Linux or MAC OS X.

For easy source level debugging, the compilers provide debug information that is optimized to the MPLAB X IDE.

The free MPLAB XC Compiler editions support all devices and commands, with no time or memory restrictions, and offer sufficient code optimization for most applications.

MPLAB XC Compilers include an assembler, linker and utilities. The assembler generates relocatable object files that can then be archived or linked with other relocatable object files and archives to create an executable file. MPLAB XC Compiler uses the assembler to produce its object file. Notable features of the assembler include:

- Support for the entire device instruction set
- Support for fixed-point and floating-point data
- Command-line interface
- Rich directive set
- Flexible macro language
- MPLAB X IDE compatibility

## 32.3 MPASM Assembler

The MPASM Assembler is a full-featured, universal macro assembler for PIC10/12/16/18 MCUs.

The MPASM Assembler generates relocatable object files for the MPLINK Object Linker, Intel® standard HEX files, MAP files to detail memory usage and symbol reference, absolute LST files that contain source lines and generated machine code, and COFF files for debugging.

The MPASM Assembler features include:

- Integration into MPLAB X IDE projects
- User-defined macros to streamline assembly code
- Conditional assembly for multipurpose source files
- Directives that allow complete control over the assembly process

## 32.4 MPLINK Object Linker/ MPLIB Object Librarian

The MPLINK Object Linker combines relocatable objects created by the MPASM Assembler. It can link relocatable objects from precompiled libraries, using directives from a linker script.

The MPLIB Object Librarian manages the creation and modification of library files of precompiled code. When a routine from a library is called from a source file, only the modules that contain that routine will be linked in with the application. This allows large libraries to be used efficiently in many different applications.

The object linker/library features include:

- Efficient linking of single libraries instead of many smaller files
- Enhanced code maintainability by grouping related modules together
- Flexible creation of libraries with easy module listing, replacement, deletion and extraction

## 32.5 MPLAB Assembler, Linker and Librarian for Various Device Families

MPLAB Assembler produces relocatable machine code from symbolic assembly language for PIC24, PIC32 and dsPIC DSC devices. MPLAB XC Compiler uses the assembler to produce its object file. The assembler generates relocatable object files that can then be archived or linked with other relocatable object files and archives to create an executable file. Notable features of the assembler include:

- Support for the entire device instruction set
- Support for fixed-point and floating-point data
- Command-line interface
- Rich directive set
- Flexible macro language
- MPLAB X IDE compatibility

**TABLE 33-6: DC CHARACTERISTICS: OPERATING CURRENT (IDD)**

| DC CHARACTERISTICS | | | Standard Operating Conditions: 3.0V to 3.6V (unless otherwise stated)<br>Operating temperature -40°C ≤ TA ≤ +85°C for Industrial<br>-40°C ≤ TA ≤ +125°C for Extended | | | |
|---|---|---|---|---|---|---|
| **Param.** | **Typ.(2)** | **Max.** | **Units** | **Conditions** | | |
| **Operating Current (IDD)(1)** | | | | | | |
| DC20d | 6.0 | 18.0 | mA | -40°C | 3.3V | 10 MIPS |
| DC20a | 6.0 | 18.0 | mA | +25°C | | |
| DC20b | 6.0 | 18.0 | mA | +85°C | | |
| DC20c | 6.0 | 18.0 | mA | +125°C | | |
| DC21d | 11.0 | 20.0 | mA | -40°C | 3.3V | 20 MIPS |
| DC21a | 11.0 | 20.0 | mA | +25°C | | |
| DC21b | 11.0 | 20.0 | mA | +85°C | | |
| DC21c | 11.0 | 20.0 | mA | +125°C | | |
| DC22d | 17.0 | 30.0 | mA | -40°C | 3.3V | 40 MIPS |
| DC22a | 17.0 | 30.0 | mA | +25°C | | |
| DC22b | 17.0 | 30.0 | mA | +85°C | | |
| DC22c | 17.0 | 30.0 | mA | +125°C | | |
| DC23d | 25.0 | 50.0 | mA | -40°C | 3.3V | 60 MIPS |
| DC23a | 25.0 | 50.0 | mA | +25°C | | |
| DC23b | 25.0 | 50.0 | mA | +85°C | | |
| DC23c | 25.0 | 50.0 | mA | +125°C | | |
| DC24d | 30.0 | 60.0 | mA | -40°C | 3.3V | 70 MIPS |
| DC24a | 30.0 | 60.0 | mA | +25°C | | |
| DC24b | 30.0 | 60.0 | mA | +85°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 and external clock is active, 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 outputs and driving low
- $\overline{\text{MCLR}}$ = VDD, WDT and FSCM are disabled
- CPU, SRAM, program memory and data memory are operational
- No peripheral modules are operating or being clocked (defined PMDx bits are all ones)
- CPU executing

```
while(1)
{
NOP();
}
```

- JTAG is disabled

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

**FIGURE 33-21: SPI2 AND SPI3 SLAVE MODE (FULL-DUPLEX, CKE = 0, CKP = 1, SMP = 0) TIMING CHARACTERISTICS**

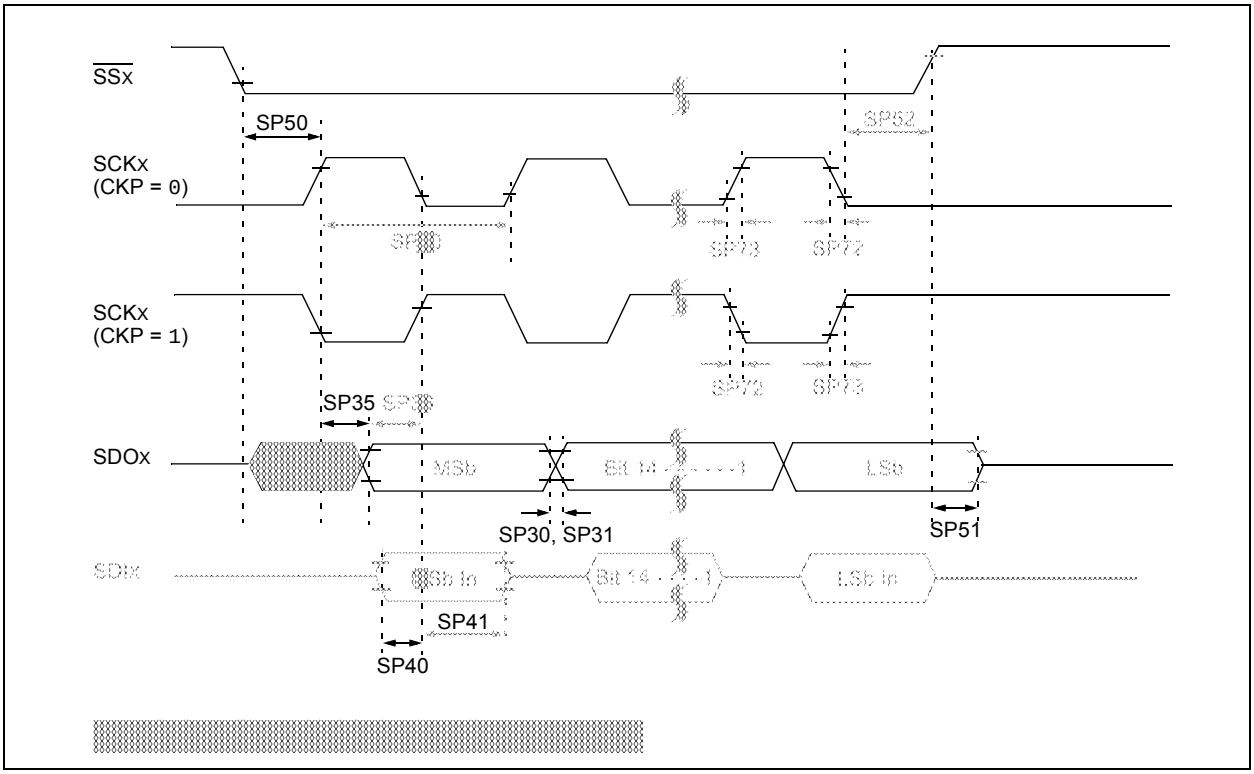
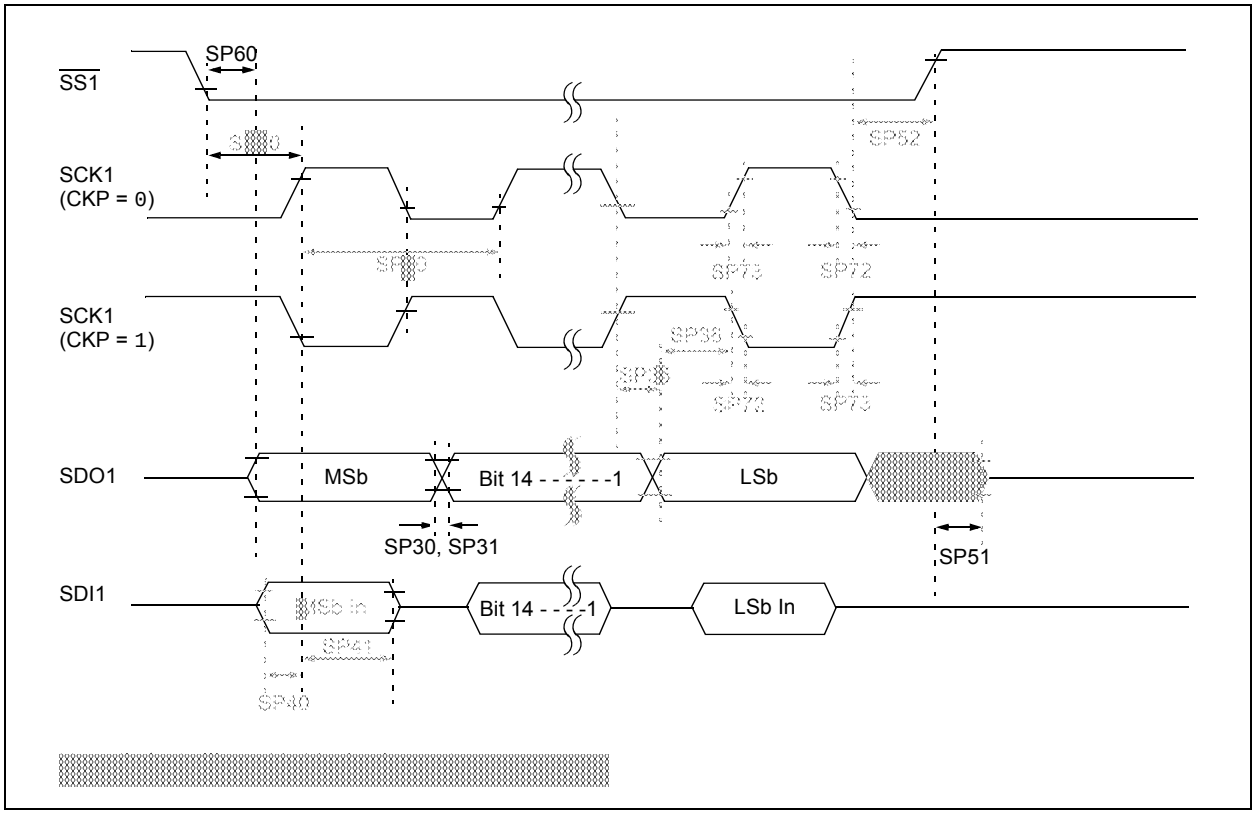


**FIGURE 33-27: SPI1 SLAVE MODE (FULL-DUPLEX, CKE = 1, CKP = 0, SMP = 0) TIMING CHARACTERISTICS**



**NOTES:**

## THE MICROCHIP WEB SITE

Microchip provides online support via our WWW site at www.microchip.com. This web site is used as a means to make files and information easily available to customers. Accessible by using your favorite Internet browser, the web site contains the following information:

- **Product Support** – Data sheets and errata, application notes and sample programs, design resources, user's guides and hardware support documents, latest software releases and archived software
- **General Technical Support** – Frequently Asked Questions (FAQ), technical support requests, online discussion groups, Microchip consultant program member listing
- **Business of Microchip** – Product selector and ordering guides, latest Microchip press releases, listing of seminars and events, listings of Microchip sales offices, distributors and factory representatives

## CUSTOMER CHANGE NOTIFICATION SERVICE

Microchip's customer notification service helps keep customers current on Microchip products. Subscribers will receive e-mail notification whenever there are changes, updates, revisions or errata related to a specified product family or development tool of interest.

To register, access the Microchip web site at www.microchip.com. Under "Support", click on "Customer Change Notification" and follow the registration instructions.

## CUSTOMER SUPPORT

Users of Microchip products can receive assistance through several channels:

- Distributor or Representative
- Local Sales Office
- Field Application Engineer (FAE)
- Technical Support

Customers should contact their distributor, representative or Field Application Engineer (FAE) for support. Local sales offices are also available to help customers. A listing of sales offices and locations is included in the back of this document.

**Technical support is available through the web site at: http://microchip.com/support**