



dsPIC30F Data Sheet

General Purpose and Sensor Families

High-Performance
Digital Signal Controllers

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contained in this publication regarding device applications and the like is intended through suggestion only and may be superseded by updates. It is your responsibility to ensure that your application meets with your specifications. No representation or warranty is given and no liability is assumed by Microchip Technology Incorporated with respect to the accuracy or use of such information, or infringement of patents or other intellectual property rights arising from such use or otherwise. Use of Microchip's products as critical components in life support systems is not authorized except with express written approval by Microchip. No licenses are conveyed, implicitly or otherwise, under any intellectual property rights.

Trademarks

The Microchip name and logo, the Microchip logo, Accuron, dsPIC, KEELOQ, microID, MPLAB, PIC, PICmicro, PICSTART, PRO MATE, PowerSmart, rPIC, and SmartShunt are registered trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.


AmpLab, FilterLab, MXDEV, MXLAB, PICMASTER, 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, dsPICDEM, dsPICDEM.net, dsPICworks, ECAN, ECONOMONITOR, FanSense, FlexROM, fuzzyLAB, In-Circuit Serial Programming, ICSP, ICEPIC, Migratable Memory, MPASM, MPLIB, MPLINK, MPSIM, PICKit, PICDEM, PICDEM.net, PICLAB, PICTail, PowerCal, PowerInfo, PowerMate, PowerTool, rLAB, rPICDEM, Select Mode, Smart Serial, SmartTel and Total Endurance 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.

© 2004, 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 quality system certification for its worldwide headquarters, design and wafer fabrication facilities in Chandler and Tempe, Arizona and Mountain View, California in October 2003. The Company's quality system processes and procedures are for its PICmicro® 8-bit MCUs, 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.

dsPIC30F Enhanced Flash 16-bit Digital Signal Controllers General Purpose and Sensor Families

Note: This data sheet summarizes features of this group of dsPIC30F devices and is not intended to be a complete reference source. For more information on the CPU, peripherals, register descriptions and general device functionality, refer to the *dsPIC30F Family Reference Manual* (DS70046). For more information on the device instruction set and programming, refer to the *dsPIC30F Programmer's Reference Manual* (DS70030).

High Performance Modified RISC CPU:

- Modified Harvard architecture
- C compiler optimized instruction set architecture
- 84 base instructions
- 24-bit wide instructions, 16-bit wide data path
- Linear program memory addressing up to 4M instruction words
- Linear data memory addressing up to 64 Kbytes
- Up to 144 Kbytes on-chip Flash program space
- Up to 48K instruction words
- Up to 8 Kbytes of on-chip data RAM
- Up to 4 Kbytes of non-volatile data EEPROM
- 16 x 16-bit working register array
- Three Address Generation Units that enable:
 - Dual data fetch
 - Accumulator write back for DSP operations
- Flexible Addressing modes supporting:
 - Indirect, Modulo and Bit-Reversed modes
- Two 40-bit wide accumulators with optional saturation logic
- 17-bit x 17-bit single cycle hardware fractional/integer multiplier
- Single cycle Multiply-Accumulate (MAC) operation
- 40-stage Barrel Shifter
- Up to 30 MIPs operation:
 - DC to 40 MHz external clock input
 - 4 MHz-10 MHz oscillator input with PLL active (4x, 8x, 16x)
- Up to 41 interrupt sources:
 - 8 user selectable priority levels
- Vector table with up to 62 vectors:
 - 54 interrupt vectors
 - 8 processor exceptions and software traps

Peripheral Features:

- High current sink/source I/O pins: 25 mA/25 mA
- Up to 5 external interrupt sources
- Timer module with programmable prescaler:
 - Up to five 16-bit timers/counters; optionally pair up 16-bit timers into 32-bit timer modules
- 16-bit Capture input functions
- 16-bit Compare/PWM output functions:
 - Dual Compare mode available
- Data Converter Interface (DCI) supports common audio Codec protocols, including I²S and AC'97
- 3-wire SPI™ modules (supports 4 Frame modes)
- I²C™ module supports Multi-Master/Slave mode and 7-bit/10-bit addressing
- Addressable UART modules supporting:
 - Interrupt on address bit
 - Wake-up on Start bit
 - 4 characters deep TX and RX FIFO buffers
- CAN bus modules

Analog Features:

- 12-bit Analog-to-Digital Converter (A/D) with:
 - 100 Ksps conversion rate
 - Up to 16 input channels
 - Conversion available during Sleep and Idle
- Programmable Low Voltage Detection (PLVD)
- Programmable Brown-out Detection and Reset generation

dsPIC30F

Special Microcontroller Features:

- Enhanced Flash program memory:
 - 10,000 erase/write cycle (min.) for industrial temperature range, 100K (typical)
- Data EEPROM memory:
 - 100,000 erase/write cycle (min.) for industrial temperature range, 1M (typical)
- Self-reprogrammable under software control
- Power-on Reset (POR), Power-up Timer (PWRT) and Oscillator Start-up Timer (OST)
- Flexible Watchdog Timer (WDT) with on-chip low power RC oscillator for reliable operation
- Fail-Safe Clock Monitor operation:
 - Detects clock failure and switches to on-chip low power RC oscillator
- Programmable code protection
- In-Circuit Serial Programming™ (ICSP™) via 3 pins and power/ground
- Selectable Power Management modes:
 - Sleep, Idle and Alternate Clock modes

CMOS Technology:

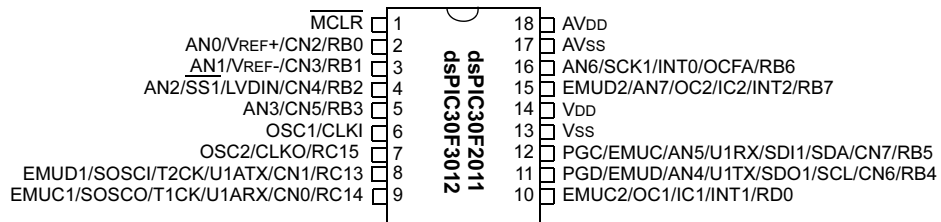
- Low power, high speed Flash technology
- Wide operating voltage range (2.5V to 5.5V)
- Industrial and Extended temperature ranges
- Low power consumption

dsPIC30F Sensor Processor Family

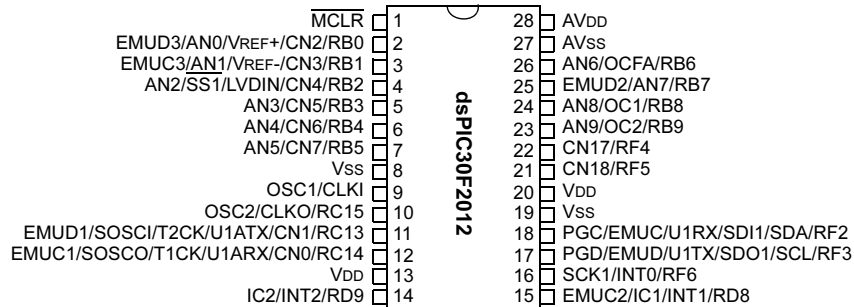
| Device | Pins | Program Memory | | SRAM Bytes | EEPROM Bytes | Timer 16-bit | Input Cap | Output Comp/ Std PWM | A/D 12-bit 100 Ksps | UART | SPI™ | I²C™ |
|--------------|------|----------------|--------------|------------|--------------|--------------|-----------|----------------------|---------------------|------|------|------|
| | | Bytes | Instructions | | | | | | | | | |
| dsPIC30F2011 | 18 | 12K | 4K | 1024 | 0 | 3 | 2 | 2 | 8 ch | 1 | 1 | 1 |
| dsPIC30F3012 | 18 | 24K | 8K | 2048 | 1024 | 3 | 2 | 2 | 8 ch | 1 | 1 | 1 |
| dsPIC30F2012 | 28 | 12K | 4K | 1024 | 0 | 3 | 2 | 2 | 10 ch | 1 | 1 | 1 |
| dsPIC30F3013 | 28 | 24K | 8K | 2048 | 1024 | 3 | 2 | 2 | 10 ch | 2 | 1 | 1 |

Pin Diagrams

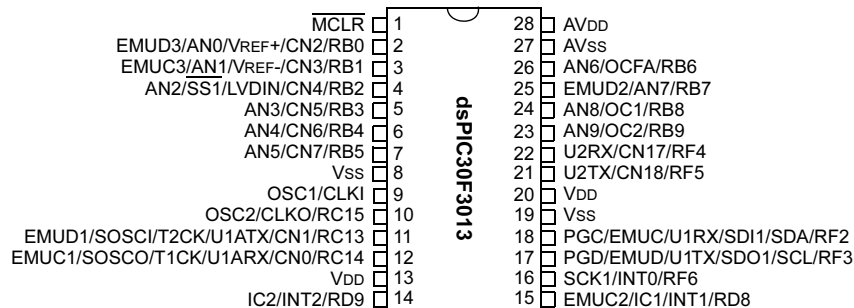
18-Pin SOIC and PDIP



28-Pin PDIP and SOIC



28-Pin PDIP and SOIC

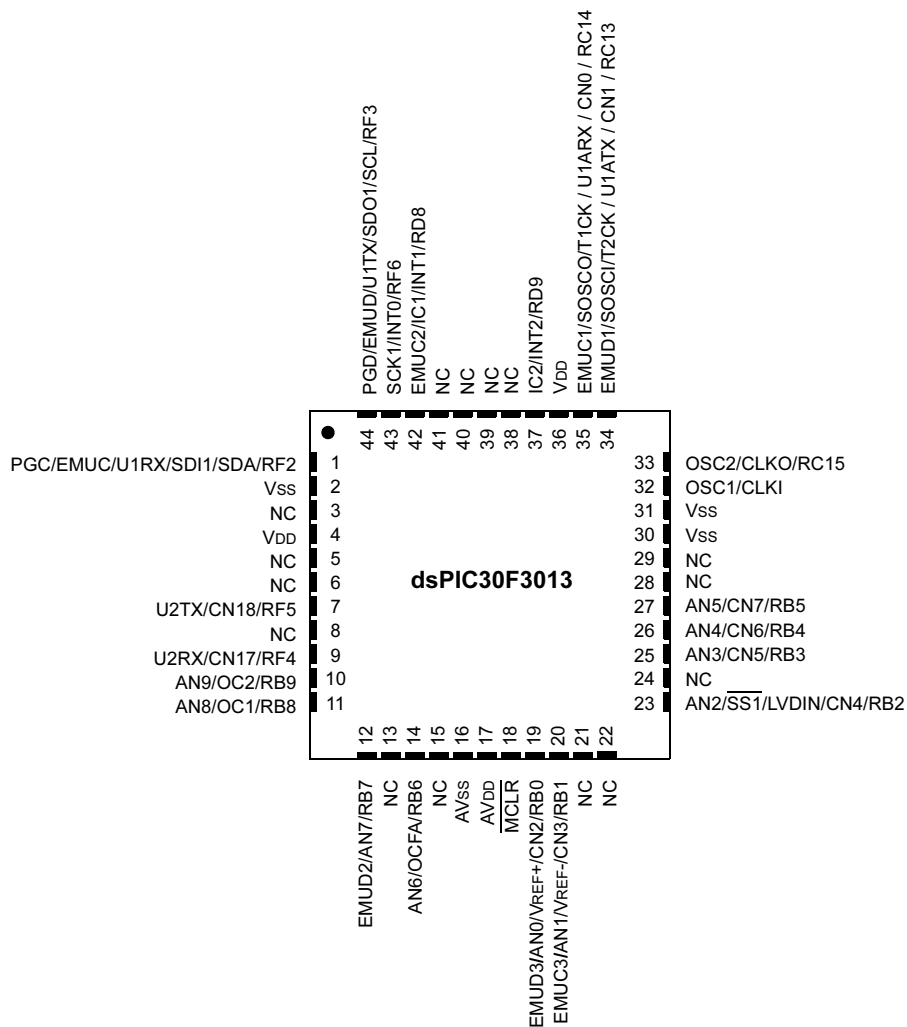


Note: For descriptions of individual pins, see Section 1.0.

dsPIC30F

Pin Diagrams (Continued)

44-Pin QFN

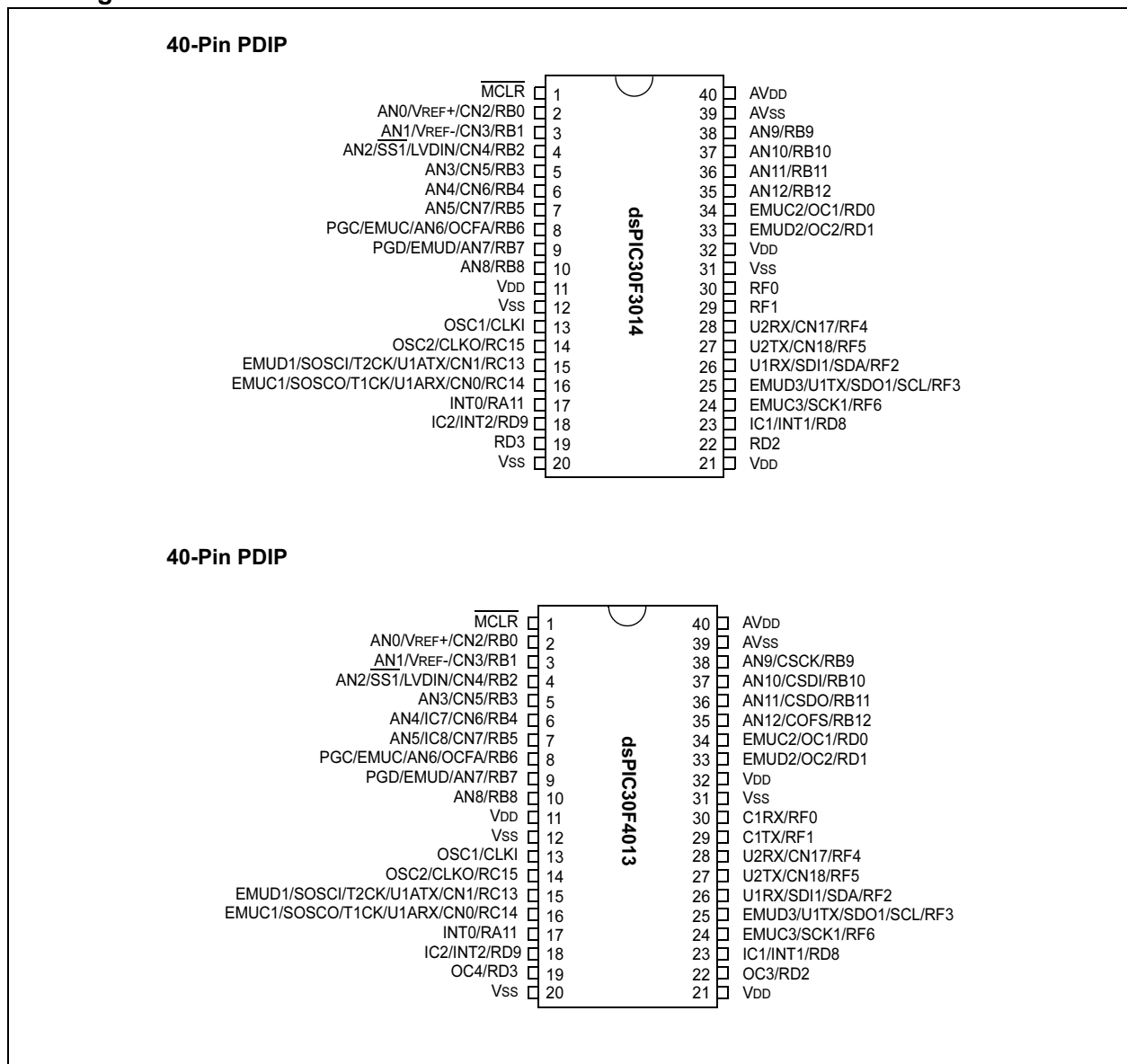


Note: For descriptions of individual pins, see Section 1.0.

dsPIC30F General Purpose Controller Family

| Device | Pins | Program Memory | | SRAM Bytes | EEPROM Bytes | Timer 16-bit | Input Cap | Output Comp/Std PWM | Codec Interface | A/D 12-bit 100 Ksps | UART | SPI™ | I ² C™ | CAN |
|--------------|-------|----------------|--------------|------------|--------------|--------------|-----------|---------------------|-------------------------|---------------------|------|------|-------------------|-----|
| | | Bytes | Instructions | | | | | | | | | | | |
| dsPIC30F3014 | 40/44 | 24K | 8K | 2048 | 1024 | 3 | 2 | 2 | — | 13 ch | 2 | 1 | 1 | - |
| dsPIC30F4013 | 40/44 | 48K | 16K | 2048 | 1024 | 5 | 4 | 4 | AC'97, I ² S | 13 ch | 2 | 1 | 1 | 1 |
| dsPIC30F5011 | 64 | 66K | 22K | 4096 | 1024 | 5 | 8 | 8 | AC'97, I ² S | 16 ch | 2 | 2 | 1 | 2 |
| dsPIC30F6011 | 64 | 132K | 44K | 6144 | 2048 | 5 | 8 | 8 | — | 16 ch | 2 | 2 | 1 | 2 |
| dsPIC30F6012 | 64 | 144K | 48K | 8192 | 4096 | 5 | 8 | 8 | AC'97, I ² S | 16 ch | 2 | 2 | 1 | 2 |
| dsPIC30F5013 | 80 | 66K | 22K | 4096 | 1024 | 5 | 8 | 8 | AC'97, I ² S | 16 ch | 2 | 2 | 1 | 2 |
| dsPIC30F6013 | 80 | 132K | 44K | 6144 | 2048 | 5 | 8 | 8 | — | 16 ch | 2 | 2 | 1 | 2 |
| dsPIC30F6014 | 80 | 144K | 48K | 8192 | 4096 | 5 | 8 | 8 | AC'97, I ² S | 16 ch | 2 | 2 | 1 | 2 |

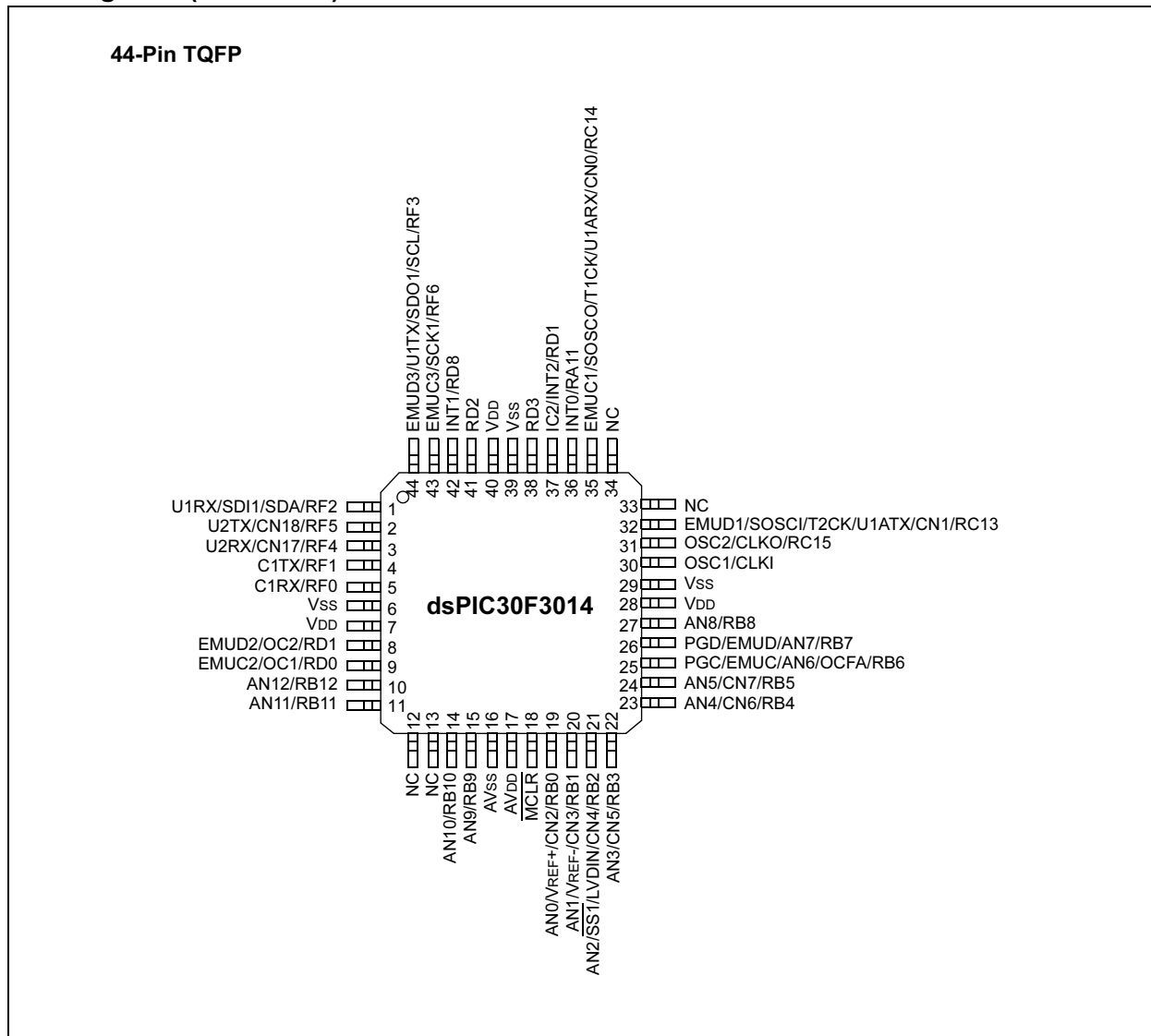
Pin Diagrams



Note: For descriptions of individual pins, see Section 1.0.

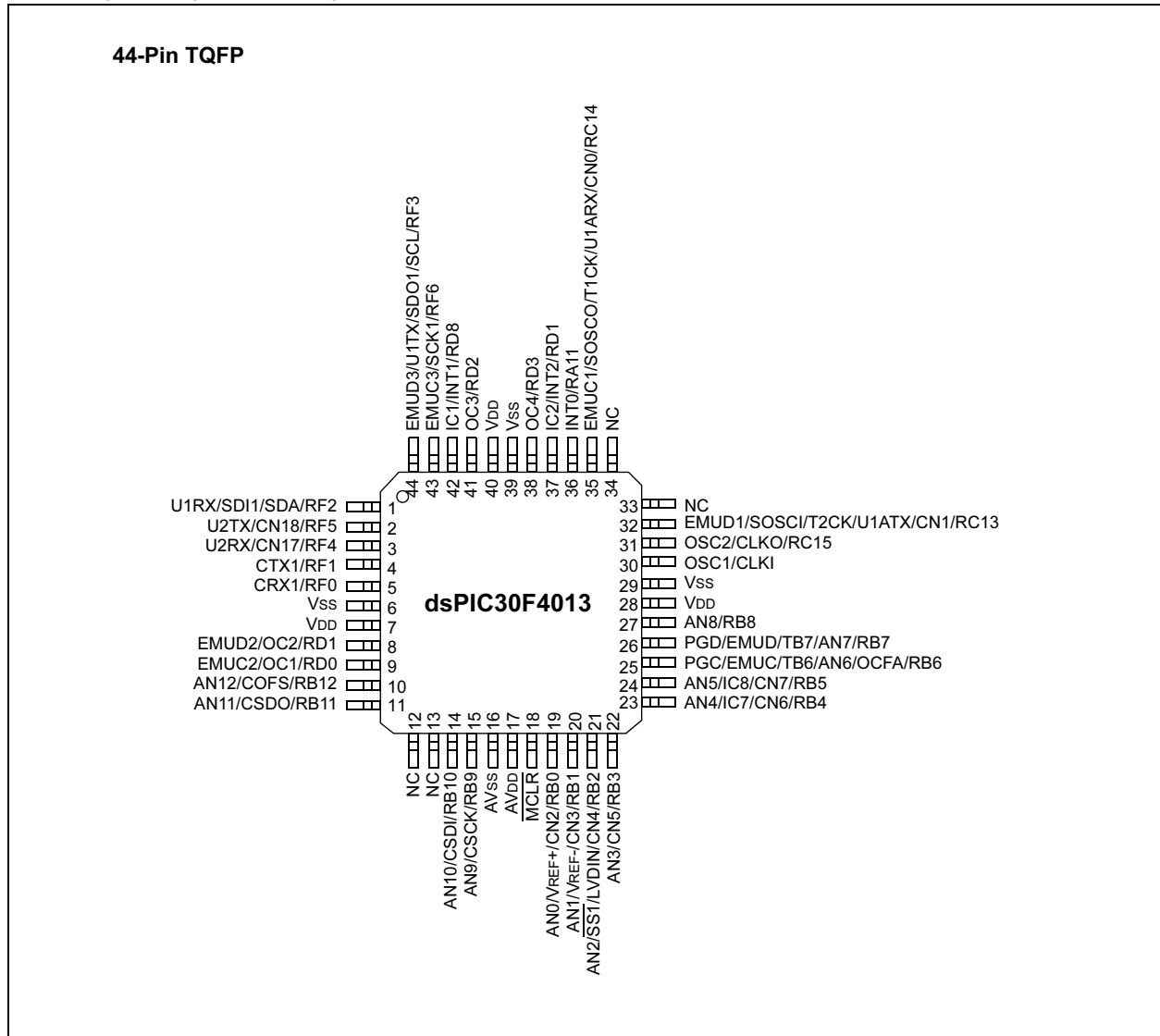
dsPIC30F

Pin Diagrams (Continued)



Note: For descriptions of individual pins, see Section 1.0.

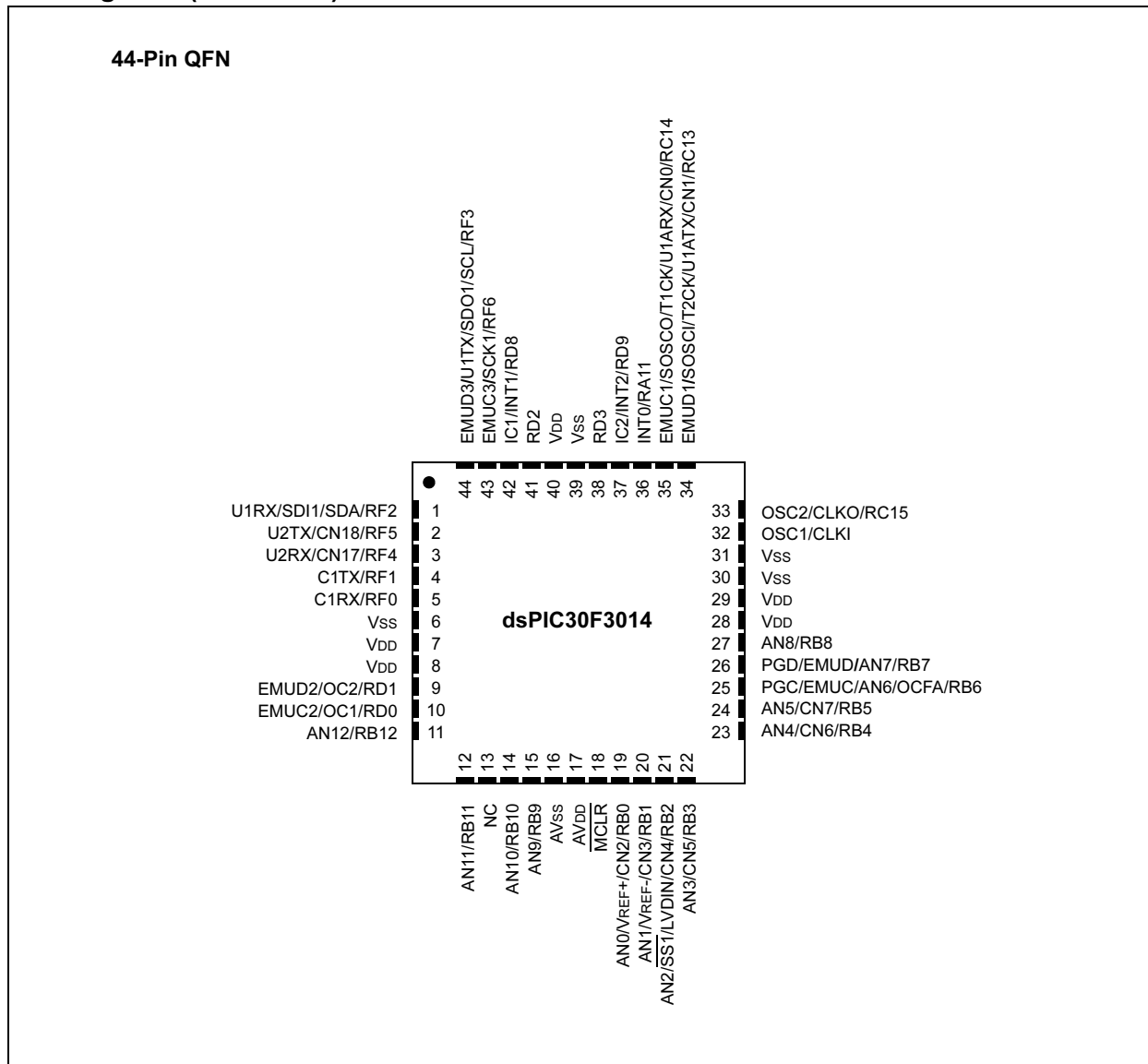
Pin Diagrams (Continued)



Note: For descriptions of individual pins, see Section 1.0.

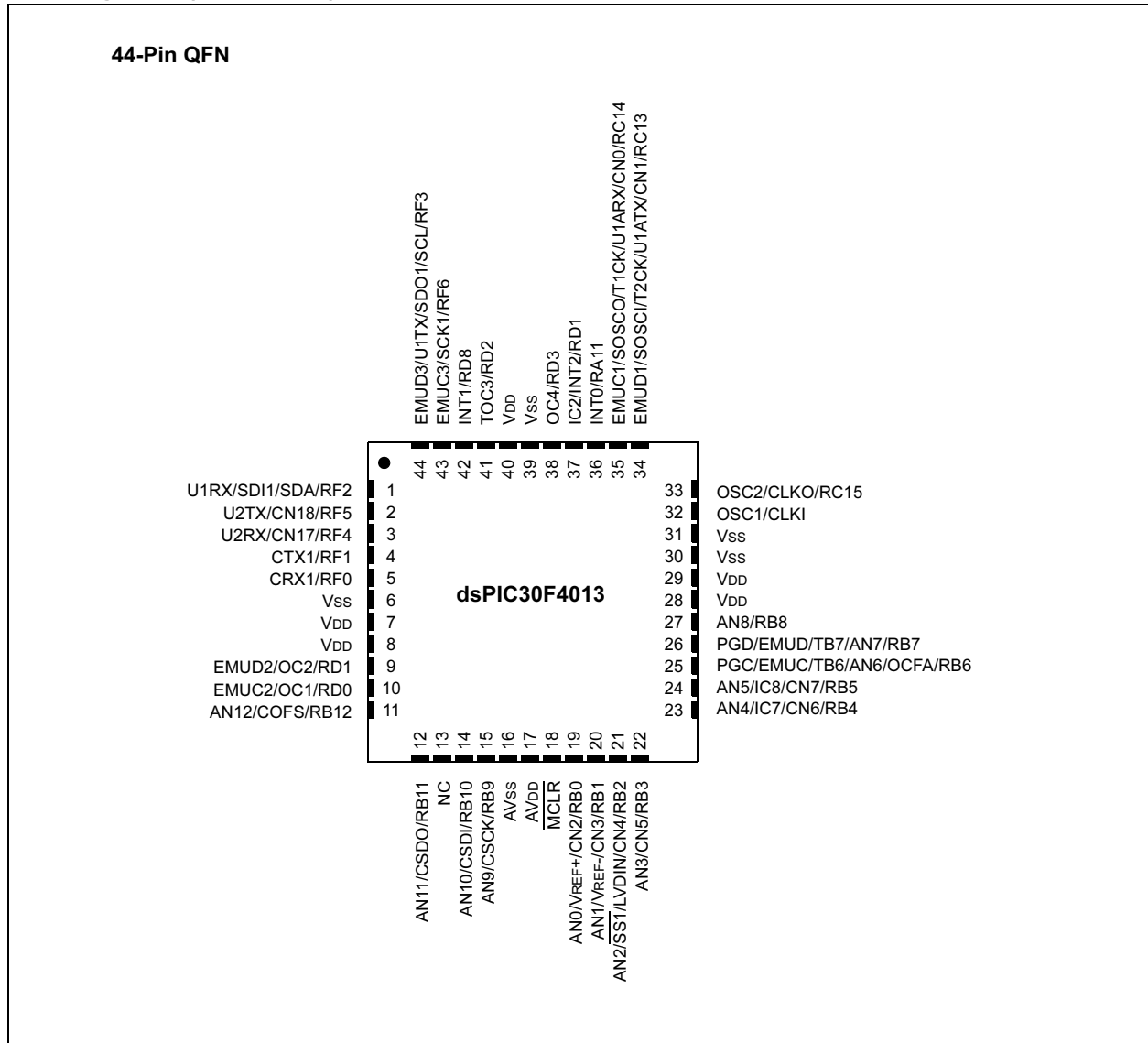
dsPIC30F

Pin Diagrams (Continued)



Note: For descriptions of individual pins, see Section 1.0.

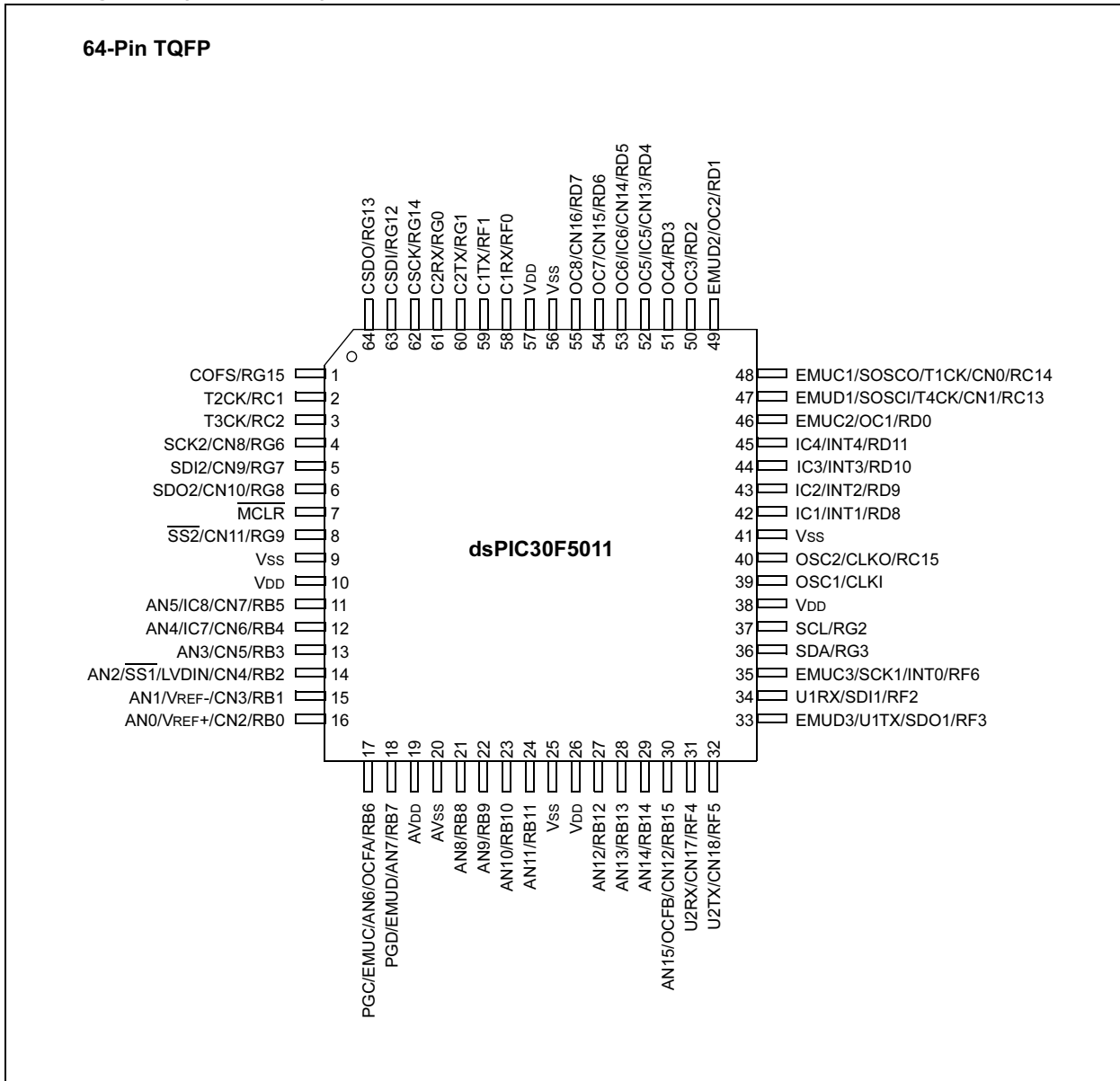
Pin Diagrams (Continued)



Note: For descriptions of individual pins, see Section 1.0.

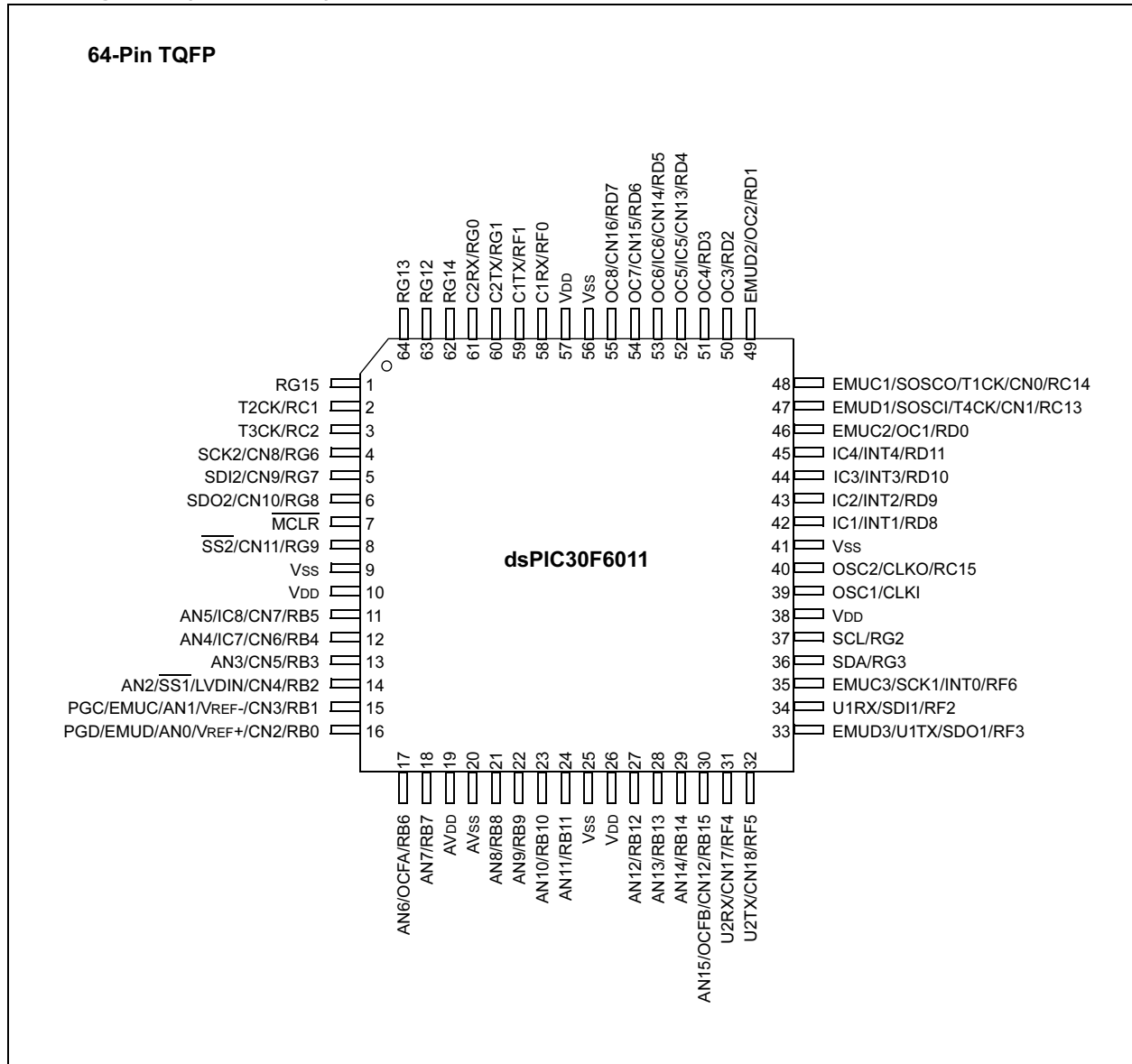
dsPIC30F

Pin Diagrams (Continued)



Note: For descriptions of individual pins, see Section 1.0.

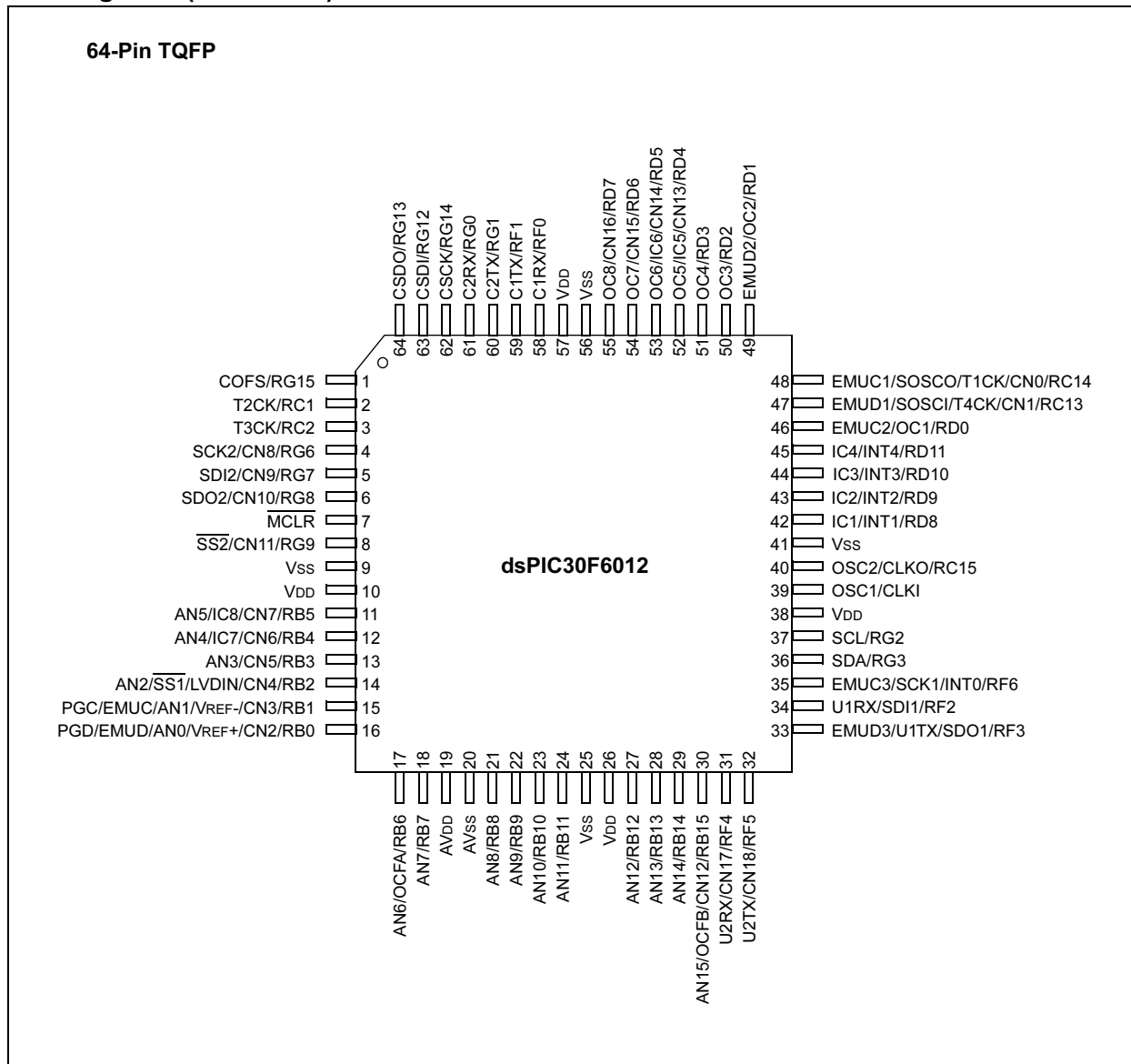
Pin Diagrams (Continued)



Note: For descriptions of individual pins, see Section 1.0.

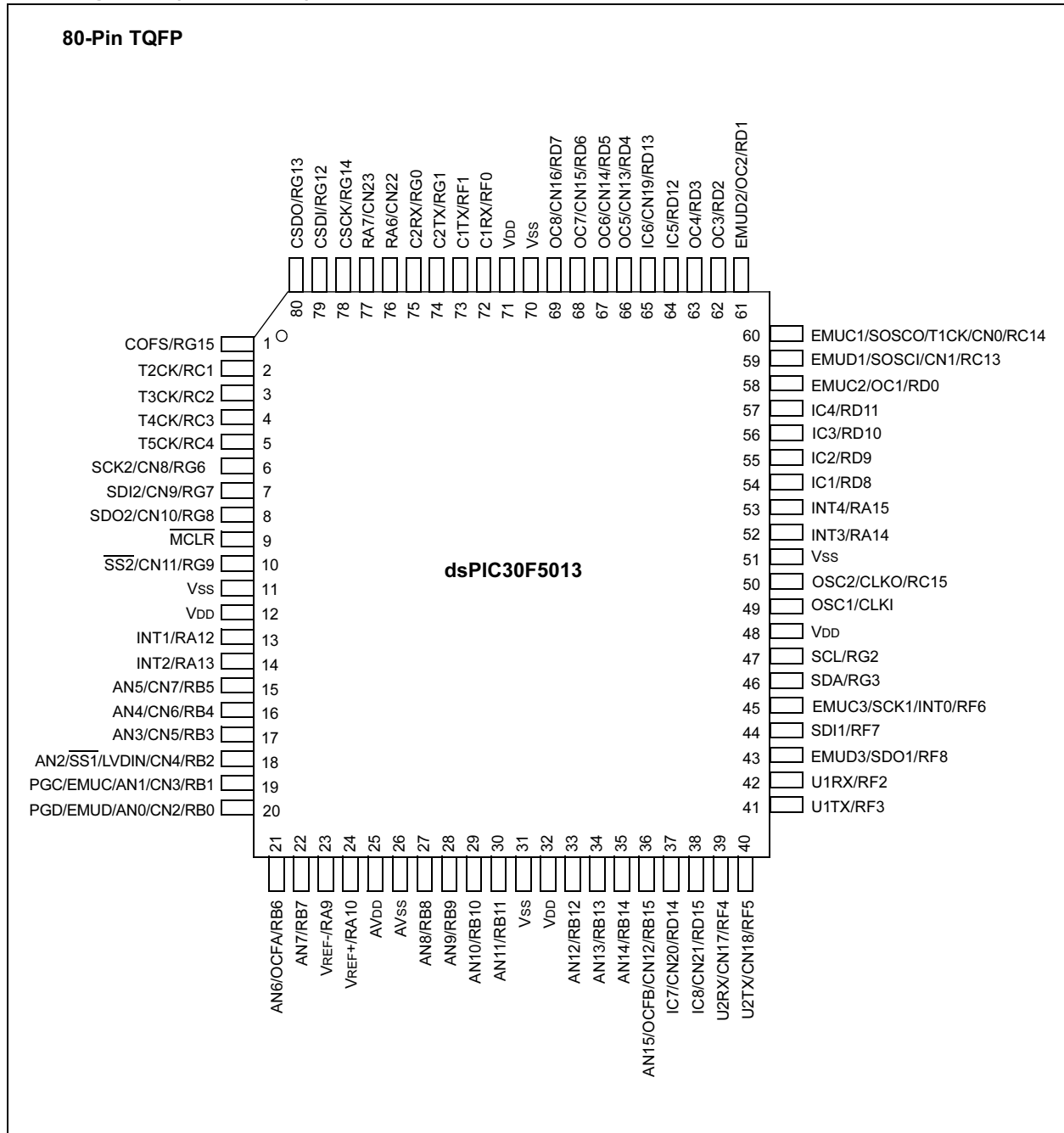
dsPIC30F

Pin Diagrams (Continued)



Note: For descriptions of individual pins, see Section 1.0.

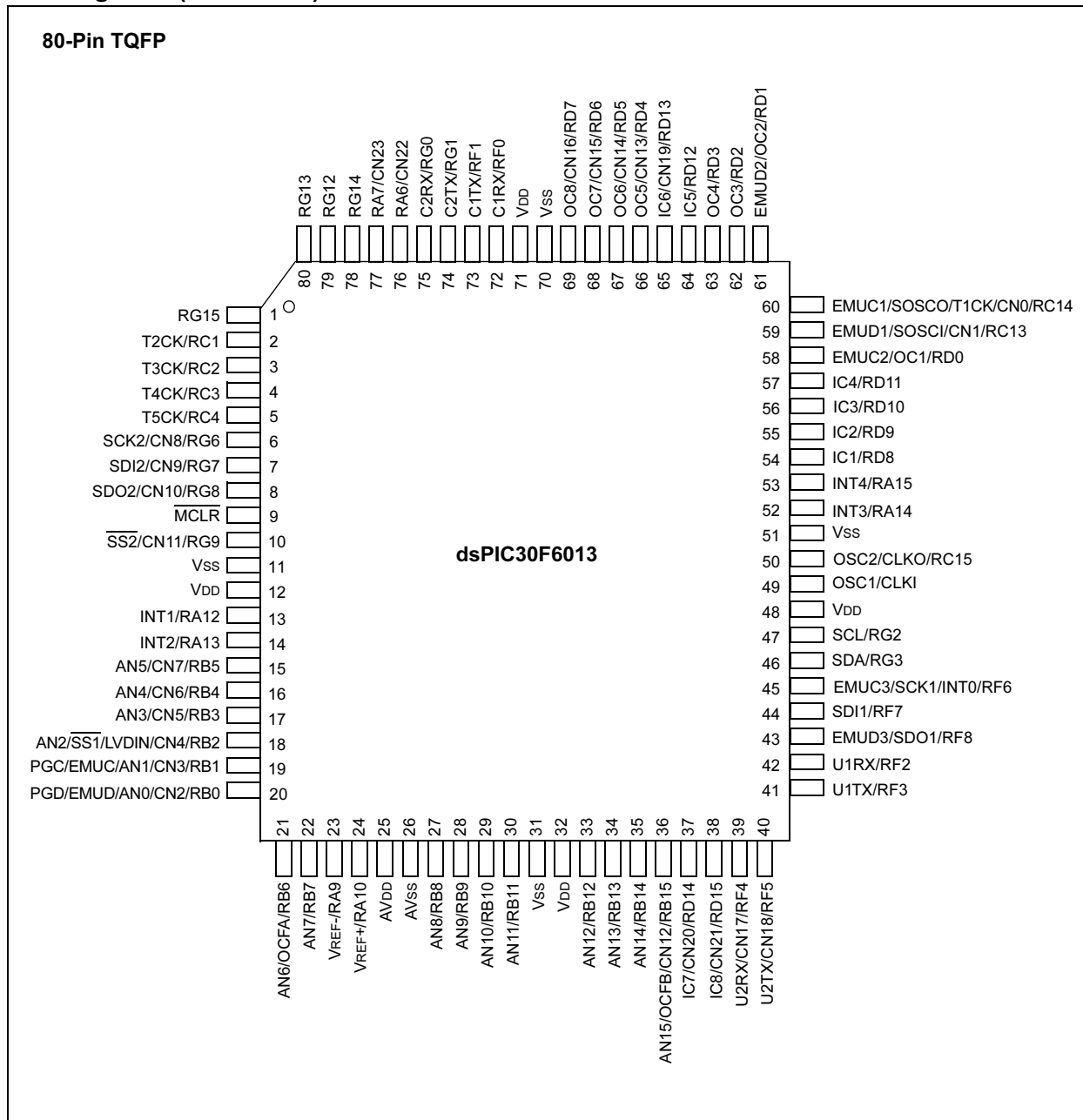
Pin Diagrams (Continued)



Note: For descriptions of individual pins, see Section 1.0.

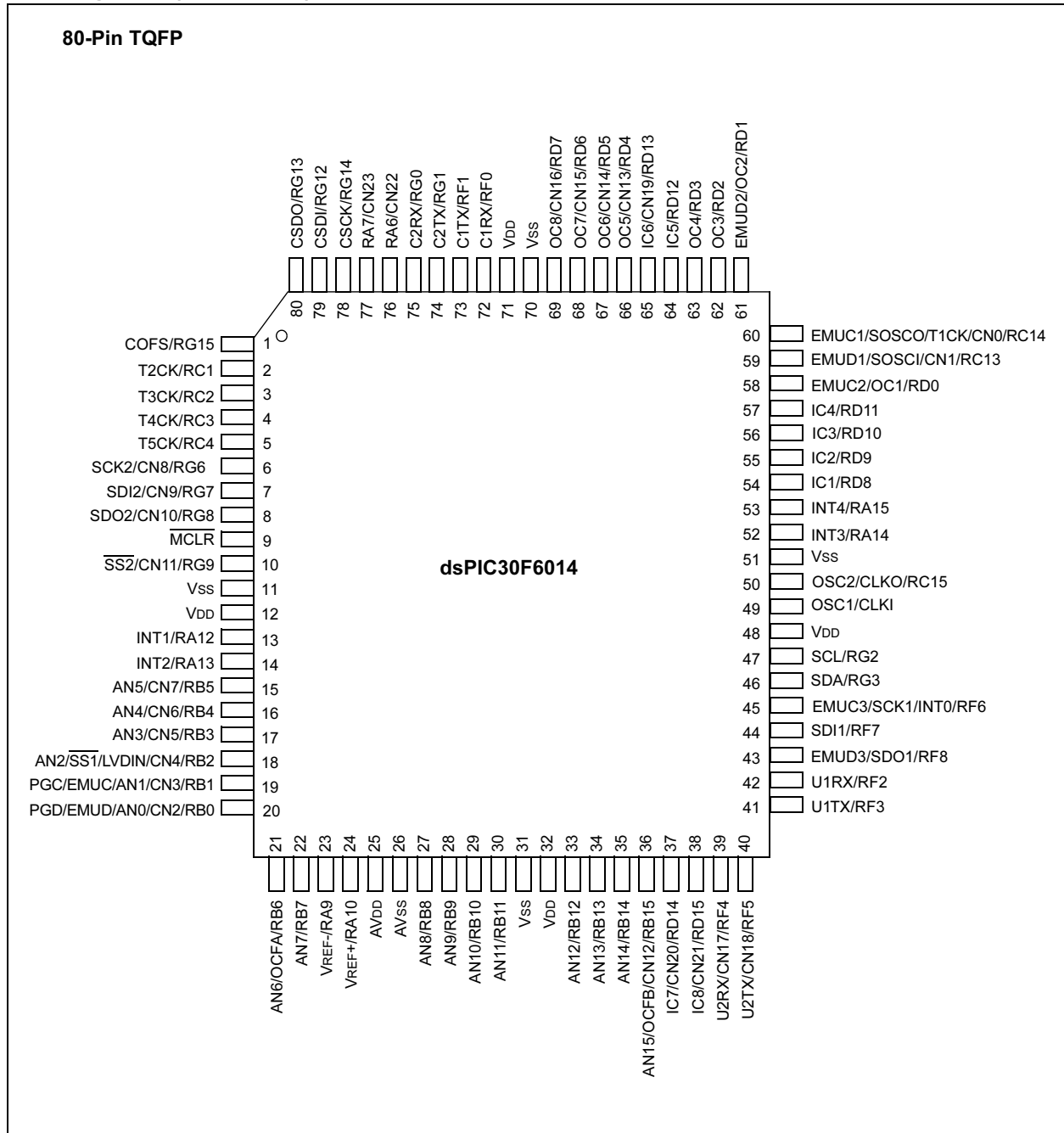
dsPIC30F

Pin Diagrams (Continued)



Note: For descriptions of individual pins, see Section 1.0.

Pin Diagrams (Continued)



Note: For descriptions of individual pins, see Section 1.0.

dsPIC30F

Table of Contents

| | | |
|------|---|-----|
| 1.0 | Device Overview | 17 |
| 2.0 | CPU Architecture Overview..... | 21 |
| 3.0 | Memory Organization..... | 35 |
| 4.0 | Address Generator Units..... | 47 |
| 5.0 | Interrupts | 55 |
| 6.0 | Flash Program Memory..... | 63 |
| 7.0 | Data EEPROM Memory | 69 |
| 8.0 | I/O Ports | 75 |
| 9.0 | Timer1 Module | 81 |
| 10.0 | Timer2/3 Module | 85 |
| 11.0 | Timer4/5 Module | 91 |
| 12.0 | Input Capture Module..... | 95 |
| 13.0 | Output Compare Module..... | 99 |
| 14.0 | SPI Module..... | 103 |
| 15.0 | I2C Module..... | 107 |
| 16.0 | Universal Asynchronous Receiver Transmitter (UART) Module | 115 |
| 17.0 | CAN Module..... | 123 |
| 18.0 | Data Converter Interface (DCI) Module..... | 135 |
| 19.0 | 12-bit Analog-to-Digital Converter (A/D) Module..... | 145 |
| 20.0 | System Integration | 153 |
| 21.0 | Instruction Set Summary | 169 |
| 22.0 | Development Support..... | 177 |
| 23.0 | Electrical Characteristics | 183 |
| 24.0 | Packaging Information..... | 223 |
| | Index | 237 |
| | On-Line Support..... | 243 |
| | Systems Information and Upgrade Hot Line | 243 |
| | Reader Response | 244 |
| | Product Identification System..... | 245 |

TO OUR VALUED CUSTOMERS

It is our intention to provide our valued customers with the best documentation possible to ensure successful use of your Microchip products. To this end, we will continue to improve our publications to better suit your needs. Our publications will be refined and enhanced as new volumes and updates are introduced.

If you have any questions or comments regarding this publication, please contact the Marketing Communications Department via E-mail at docerrors@mail.microchip.com or fax the **Reader Response Form** in the back of this data sheet to (480) 792-4150. We welcome your feedback.

Most Current Data Sheet

To obtain the most up-to-date version of this data sheet, please register at our Worldwide Web site at:

<http://www.microchip.com>

You can determine the version of a data sheet by examining its literature number found on the bottom outside corner of any page. The last character of the literature number is the version number, (e.g., DS30000A is version A of document DS30000).

Errata

An errata sheet, describing minor operational differences from the data sheet and recommended workarounds, may exist for current devices. As device/documentation issues become known to us, we will publish an errata sheet. The errata will specify the revision of silicon and revision of document to which it applies.

To determine if an errata sheet exists for a particular device, please check with one of the following:

- Microchip's Worldwide Web site; <http://www.microchip.com>
- Your local Microchip sales office (see last page)
- The Microchip Corporate Literature Center; U.S. FAX: (480) 792-7277

When contacting a sales office or the literature center, please specify which device, revision of silicon and data sheet (include literature number) you are using.

Customer Notification System

Register on our web site at www.microchip.com/cn to receive the most current information on all of our products.

1.0 DEVICE OVERVIEW

Note: This data sheet summarizes features of this group of dsPIC30F devices and is not intended to be a complete reference source. For more information on the CPU, peripherals, register descriptions and general device functionality, refer to the *dsPIC30F Family Reference Manual* (DS70046). For more information on the device instruction set and programming, refer to the *dsPIC30F Programmer's Reference Manual* (DS70030).

This document contains device family specific information for the dsPIC30F family of Digital Signal Controller (DSC) devices. The dsPIC30F devices contain extensive Digital Signal Processor (DSP) functionality within a high performance 16-bit microcontroller (MCU) architecture.

Figure 1-1 shows a sample device block diagram.

Note: The device(s) depicted in this block diagram are representative of the corresponding device family. Other devices of the same family may vary in terms of number of pins and multiplexing of pin functions. Typically, smaller devices in the family contain a subset of the peripherals present in the device(s) shown in this diagram.

dsPIC30F

FIGURE 1-1: dsPIC30F5013/6013/6014 BLOCK DIAGRAM

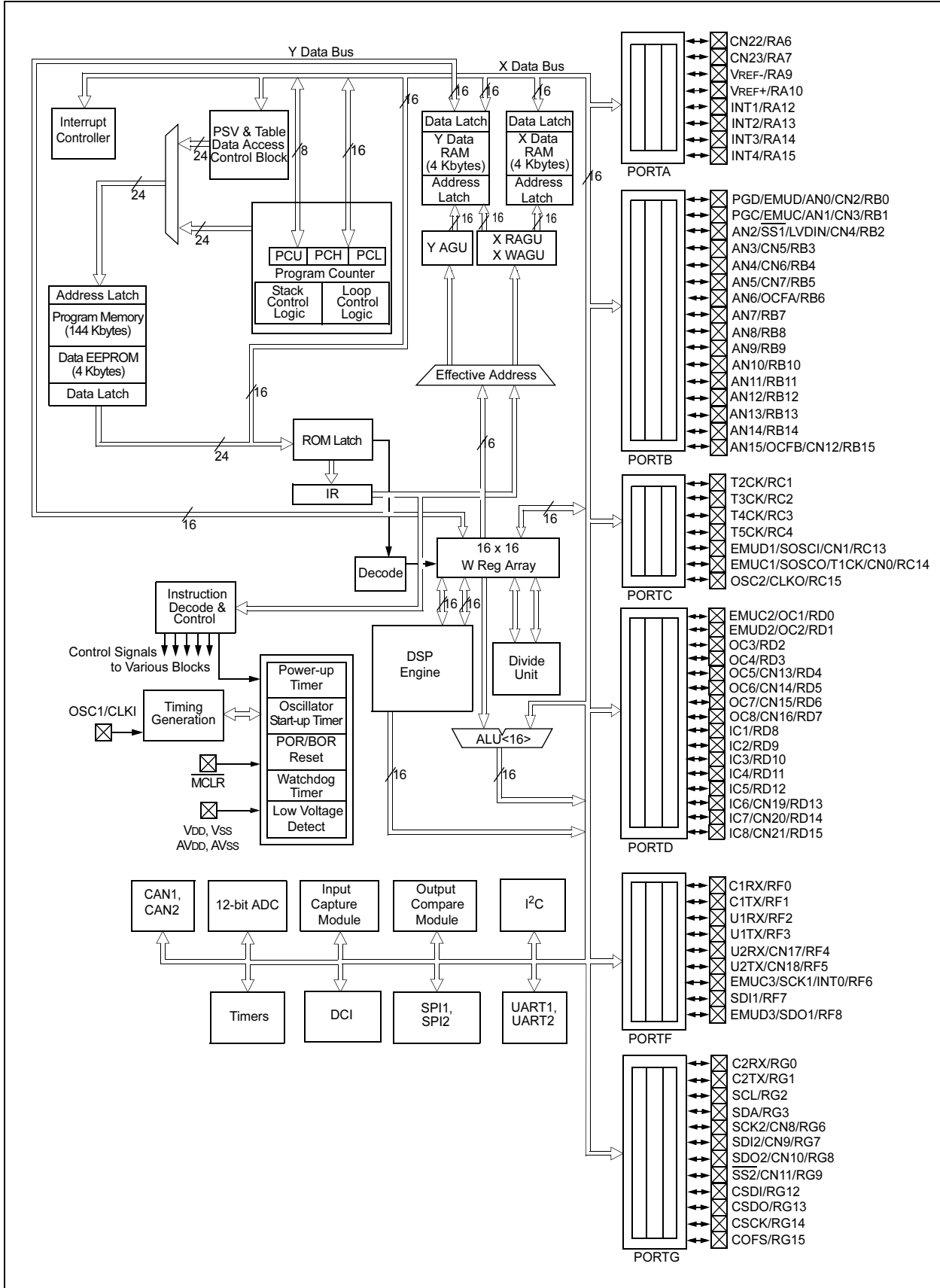


Table 1-1 provides a brief description of device I/O pinouts and the functions that may be multiplexed to a port pin. Multiple functions may exist on one port pin. When multiplexing occurs, the peripheral module's functional requirements may force an override of the data direction of the port pin.

TABLE 1-1: PINOUT I/O DESCRIPTIONS

| Pin Name | Pin Type | Buffer Type | Description |
|----------|----------|-------------|--|
| AN0-AN15 | I | Analog | Analog input channels. AN0 and AN1 are also used for device programming data and clock inputs, respectively. |
| AVDD | P | P | Positive supply for analog module. |
| AVSS | P | P | Ground reference for analog module. |
| CLKI | I | ST/CMOS | External clock source input. Always associated with OSC1 pin function. |
| CLKO | O | — | Oscillator crystal output. Connects to crystal or resonator in Crystal Oscillator mode. Optionally functions as CLKO in RC and EC modes. Always associated with OSC2 pin function. |
| CN0-CN23 | I | ST | Input change notification inputs. Can be software programmed for internal weak pull-ups on all inputs. |
| COFS | I/O | ST | Data Converter Interface frame synchronization pin. |
| CSCK | I/O | ST | Data Converter Interface serial clock input/output pin. |
| CSDI | I | ST | Data Converter Interface serial data input pin. |
| CSDO | O | — | Data Converter Interface serial data output pin. |
| C1RX | I | ST | CAN1 bus receive pin. |
| C1TX | O | — | CAN1 bus transmit pin. |
| C2RX | I | ST | CAN2 bus receive pin. |
| C2TX | O | — | CAN2 bus transmit pin. |
| EMUD | I/O | ST | ICD Primary Communication Channel data input/output pin. |
| EMUC | I/O | ST | ICD Primary Communication Channel clock input/output pin. |
| EMUD1 | I/O | ST | ICD Secondary Communication Channel data input/output pin. |
| EMUC1 | I/O | ST | ICD Secondary Communication Channel clock input/output pin. |
| EMUD2 | I/O | ST | ICD Tertiary Communication Channel data input/output pin. |
| EMUC2 | I/O | ST | ICD Tertiary Communication Channel clock input/output pin. |
| EMUD3 | I/O | ST | ICD Quaternary Communication Channel data input/output pin. |
| EMUC3 | I/O | ST | ICD Quaternary Communication Channel clock input/output pin. |
| IC1-IC8 | I | ST | Capture inputs 1 through 8. |
| INT0 | I | ST | External interrupt 0. |
| INT1 | I | ST | External interrupt 1. |
| INT2 | I | ST | External interrupt 2. |
| INT3 | I | ST | External interrupt 3. |
| INT4 | I | ST | External interrupt 4. |
| LVDIN | I | Analog | Low Voltage Detect Reference Voltage input pin. |
| MCLR | I/P | ST | Master Clear (Reset) input or programming voltage input. This pin is an active low Reset to the device. |
| OCFA | I | ST | Compare Fault A input (for Compare channels 1, 2, 3 and 4). |
| OCFB | I | ST | Compare Fault B input (for Compare channels 5, 6, 7 and 8). |
| OC1-OC8 | O | — | Compare outputs 1 through 8. |

Legend: CMOS = CMOS compatible input or output Analog = Analog input
 ST = Schmitt Trigger input with CMOS levels O = Output
 I = Input P = Power

dsPIC30F

TABLE 1-1: PINOUT I/O DESCRIPTIONS (CONTINUED)

| Pin Name | Pin Type | Buffer Type | Description |
|-----------|----------|-------------|--|
| OSC1 | I | ST/CMOS | Oscillator crystal input. ST buffer when configured in RC mode; CMOS otherwise. |
| OSC2 | I/O | — | Oscillator crystal output. Connects to crystal or resonator in Crystal Oscillator mode. Optionally functions as CLKO in RC and EC modes. |
| PGD | I/O | ST | In-Circuit Serial Programming data input/output pin. |
| PGC | I | ST | In-Circuit Serial Programming clock input pin. |
| RA6-RA7 | I/O | ST | PORTA is a bidirectional I/O port. |
| RA9-RA10 | I/O | ST | |
| RA12-RA15 | I/O | ST | |
| RB0-RB15 | I/O | ST | PORTB is a bidirectional I/O port. |
| RC1-RC4 | I/O | ST | PORTC is a bidirectional I/O port. |
| RC13-RC15 | I/O | ST | |
| RD0-RD15 | I/O | ST | PORTD is a bidirectional I/O port. |
| RF0-RF8 | I/O | ST | PORTF is a bidirectional I/O port. |
| RG0-RG3 | I/O | ST | PORTG is a bidirectional I/O port. |
| RG6-RG9 | I/O | ST | |
| RG12-RG15 | I/O | ST | |
| SCK1 | I/O | ST | Synchronous serial clock input/output for SPI1. |
| SDI1 | I | ST | SPI1 Data In. |
| SDO1 | O | — | SPI1 Data Out. |
| SS1 | I | ST | SPI1 Slave Synchronization. |
| SCK2 | I/O | ST | Synchronous serial clock input/output for SPI2. |
| SDI2 | I | ST | SPI2 Data In. |
| SDO2 | O | — | SPI2 Data Out. |
| SS2 | I | ST | SPI2 Slave Synchronization. |
| SCL | I/O | ST | Synchronous serial clock input/output for I ² C. |
| SDA | I/O | ST | Synchronous serial data input/output for I ² C. |
| SOSCO | O | — | 32 kHz low power oscillator crystal output. |
| SOSCI | I | ST/CMOS | 32 kHz low power oscillator crystal input. ST buffer when configured in RC mode; CMOS otherwise. |
| T1CK | I | ST | Timer1 external clock input. |
| T2CK | I | ST | Timer2 external clock input. |
| T3CK | I | ST | Timer3 external clock input. |
| T4CK | I | ST | Timer4 external clock input. |
| T5CK | I | ST | Timer5 external clock input. |
| U1RX | I | ST | UART1 Receive. |
| U1TX | O | — | UART1 Transmit. |
| U1ARX | I | ST | UART1 Alternate Receive. |
| U1ATX | O | — | UART1 Alternate Transmit. |
| U2RX | I | ST | UART2 Receive. |
| U2TX | O | — | UART2 Transmit. |
| VDD | P | — | Positive supply for logic and I/O pins. |
| VSS | P | — | Ground reference for logic and I/O pins. |
| VREF+ | I | Analog | Analog Voltage Reference (High) input. |
| VREF- | I | Analog | Analog Voltage Reference (Low) input. |

Legend: CMOS = CMOS compatible input or output Analog = Analog input
 ST = Schmitt Trigger input with CMOS levels O = Output
 I = Input P = Power

2.0 CPU ARCHITECTURE OVERVIEW

Note: This data sheet summarizes features of this group of dsPIC30F devices and is not intended to be a complete reference source. For more information on the CPU, peripherals, register descriptions and general device functionality, refer to the *dsPIC30F Family Reference Manual* (DS70046). For more information on the device instruction set and programming, refer to the *dsPIC30F Programmer's Reference Manual* (DS70030).

2.1 Core Overview

The core has a 24-bit instruction word. The Program Counter (PC) is 23-bits wide with the Least Significant (LS) bit always clear (refer to Section 3.1), and the Most Significant (MS) bit is ignored during normal program execution, except for certain specialized instructions. Thus, the PC can address up to 4M instruction words of user program space. An instruction pre-fetch mechanism is used to help maintain throughput. Program loop constructs, free from loop count management overhead, are supported using the `DO` and `REPEAT` instructions, both of which are interruptible at any point.

The working register array consists of 16 x 16-bit registers, each of which can act as data, address or offset registers. One working register (W15) operates as a software stack pointer for interrupts and calls.

The data space is 64 Kbytes (32K words) and is split into two blocks, referred to as X and Y data memory. Each block has its own independent Address Generation Unit (AGU). Most instructions operate solely through the X memory, AGU, which provides the appearance of a single unified data space. The Multiply-Accumulate (MAC) class of dual source DSP instructions operate through both the X and Y AGUs, splitting the data address space into two parts (see Section 3.2). The X and Y data space boundary is device specific and cannot be altered by the user. Each data word consists of 2 bytes, and most instructions can address data either as words or bytes.

There are two methods of accessing data stored in program memory:

- The upper 32 Kbytes of data space memory can be mapped into the lower half (user space) of program space at any 16K program word boundary, defined by the 8-bit Program Space Visibility Page (PSVPAG) register. This lets any instruction access program space as if it were data space, with a limitation that the access requires an additional cycle. Moreover, only the lower 16 bits of each instruction word can be accessed using this method.
- Linear indirect access of 32K word pages within program space is also possible using any working register, via table read and write instructions. Table read and write instructions can be used to access all 24 bits of an instruction word.

Overhead-free circular buffers (modulo addressing) are supported in both X and Y address spaces. This is primarily intended to remove the loop overhead for DSP algorithms.

The X AGU also supports bit-reversed addressing on destination effective addresses to greatly simplify input or output data reordering for radix-2 FFT algorithms. Refer to Section 4.0 for details on modulo and bit-reversed addressing.

The core supports Inherent (no operand), Relative, Literal, Memory Direct, Register Direct, Register Indirect, Register Offset and Literal Offset Addressing modes. Instructions are associated with predefined Addressing modes, depending upon their functional requirements.

For most instructions, the core is capable of executing a data (or program data) memory read, a working register (data) read, a data memory write and a program (instruction) memory read per instruction cycle. As a result, 3-operand instructions are supported, allowing $C = A+B$ operations to be executed in a single cycle.

A DSP engine has been included to significantly enhance the core arithmetic capability and throughput. It 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. Data in the accumulator or any working register can be shifted up to 15 bits right, or 16 bits 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 class of instructions can concurrently fetch two data operands from memory while multiplying two W registers. To enable this concurrent fetching of data operands, the data space has been split for these instructions and linear for all others. This has been achieved in a transparent and flexible manner, by dedicating certain working registers to each address space for the MAC class of instructions.

The core does not support a multi-stage instruction pipeline. However, a single stage instruction pre-fetch mechanism is used, which accesses and partially decodes instructions a cycle ahead of execution, in order to maximize available execution time. Most instructions execute in a single cycle with certain exceptions, as outlined in Section 2.3.

The core features a vectored exception processing structure for traps and interrupts, with 62 independent vectors. The exceptions consist of up to 8 traps (of which 4 are reserved) and 54 interrupts. Each interrupt is prioritized based on a user assigned priority between 1 and 7 (1 being the lowest priority and 7 being the highest), in conjunction with a predetermined 'natural order'. Traps have fixed priorities ranging from 8 to 15.

2.2 Programmer's Model

The programmer's model is shown in Figure 2-1 and consists of 16 x 16-bit working registers (W0 through W15), 2 x 40-bit accumulators (AccA and AccB), STATUS register (SR), Data Table Page register (TBLPAG), Program Space Visibility Page register (PSVPAG), DO and REPEAT registers (DOSTART, DOEND, DCOUNT and RCOUNT) and Program Counter (PC). The working registers can act as data, address or offset registers. All registers are memory mapped. W0 acts as the W register for file register addressing.

Some of these registers have a shadow register associated with each of them, as shown in Figure 2-1. The shadow register is used as a temporary holding register and can transfer its contents to or from its host register upon the occurrence of an event. None of the shadow registers are accessible directly. The following rules apply for transfer of registers into and out of shadows.

- PUSH.S and POP.S
W0, W1, W2, W3, SR (DC, N, OV, Z and C bits only) are transferred.
- DO instruction
DOSTART, DOEND, DCOUNT shadows are pushed on loop start, and popped on loop end.

When a byte operation is performed on a working register, only the Least Significant Byte of the target register is affected. However, a benefit of memory mapped working registers is that both the Least and Most Significant Bytes can be manipulated through byte wide data memory space accesses.

2.2.1 SOFTWARE STACK POINTER/ FRAME POINTER

The dsPIC[®] devices contain a software stack. W15 is the dedicated software Stack Pointer (SP), and will be automatically modified by exception processing and subroutine calls and returns. However, W15 can be referenced by any instruction in the same manner as all other W registers. This simplifies the reading, writing and manipulation of the stack pointer (e.g., creating stack frames).

Note: In order to protect against misaligned stack accesses, W15<0> is always clear.

W15 is initialized to 0x0800 during a Reset. The user may reprogram the SP during initialization to any location within data space.

W14 has been dedicated as a stack frame pointer as defined by the LNK and ULNK instructions. However, W14 can be referenced by any instruction in the same manner as all other W registers.

2.2.2 STATUS REGISTER

The dsPIC core has a 16-bit STATUS register (SR), the LS Byte of which is referred to as the SR Low byte (SRL) and the MS Byte as the SR High byte (SRH). See Figure 2-1 for SR layout.

SRL contains all the MCU ALU operation status flags (including the Z bit), as well as the CPU Interrupt Priority Level status bits, IPL<2:0> and the Repeat Active status bit, RA. During exception processing, SRL is concatenated with the MS Byte of the PC to form a complete word value which is then stacked.

The upper byte of the STATUS register contains the DSP Adder/Subtractor status bits, the DO Loop Active bit (DA) and the Digit Carry (DC) status bit.

Most SR bits are read/write. Exceptions are:

1. The DA bit: DA is read and clear only because accidentally setting it could cause erroneous operation.
2. The RA bit: RA is a read only bit because accidentally setting it could cause erroneous operation. RA is only set on entry into a REPEAT loop, and cannot be directly cleared by software.
3. The OV, OA, OB and OAB bits: These bits are read only and can only be set by the DSP engine overflow logic.
4. The SA, SB and SAB bits: These are read and clear only and can only be set by the DSP engine saturation logic. Once set, these flags remain set until cleared by the user, irrespective of the results from any subsequent DSP operations.

Note 1: Clearing the SAB bit will also clear both the SA and SB bits.

2: When the memory mapped STATUS register (SR) is the destination address for an operation which affects any of the SR bits, data writes are disabled to all bits.

2.2.2.1 Z Status Bit

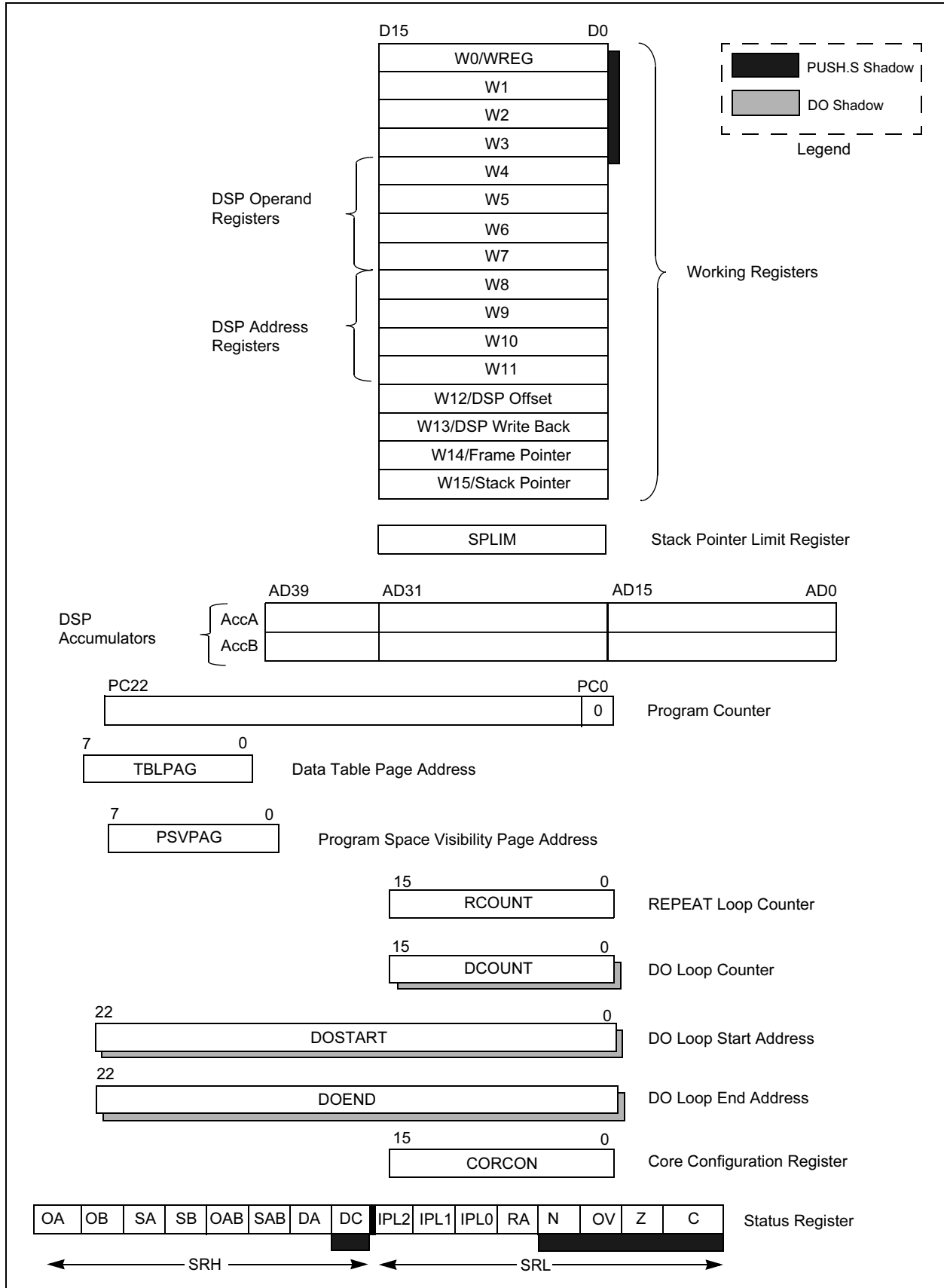
Instructions that use a carry/borrow input (ADDC, CPB, SUBB and SUBBR) will only be able to clear Z (for a non-zero result) and can never set it. A multi-precision sequence of instructions, starting with an instruction with no carry/borrow input, will thus automatically logically AND the successive results of the zero test. All results must be zero for the Z flag to remain set by the end of the sequence.

All other instructions can set as well as clear the Z bit.

2.2.3 PROGRAM COUNTER

The program counter is 23-bits wide; bit 0 is always clear. Therefore, the PC can address up to 4M instruction words.

FIGURE 2-1: PROGRAMMER'S MODEL



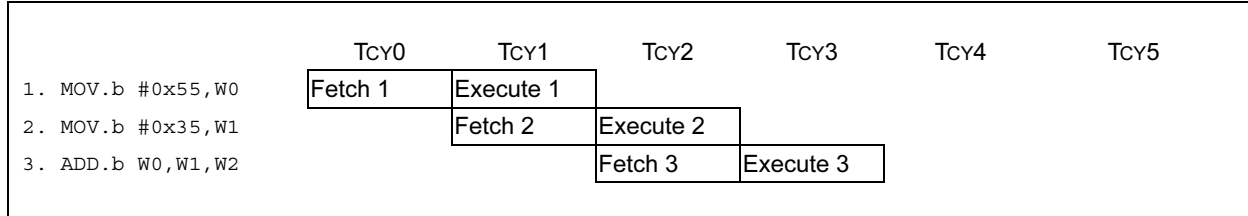
dsPIC30F

2.3 Instruction Flow

There are 8 types of instruction flows:

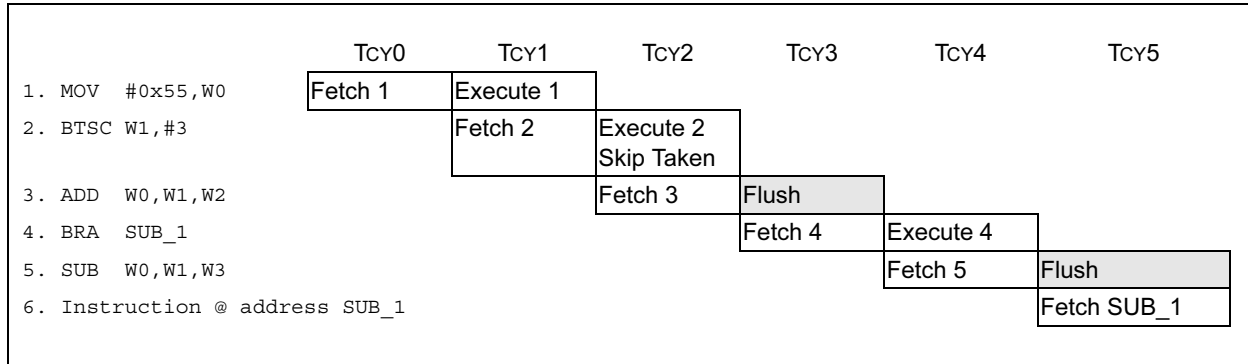
1. Normal one-word, one-cycle instructions: these instructions take one effective cycle to execute as shown in Figure 2-2.

FIGURE 2-2: INSTRUCTION PIPELINE FLOW: 1-WORD, 1-CYCLE



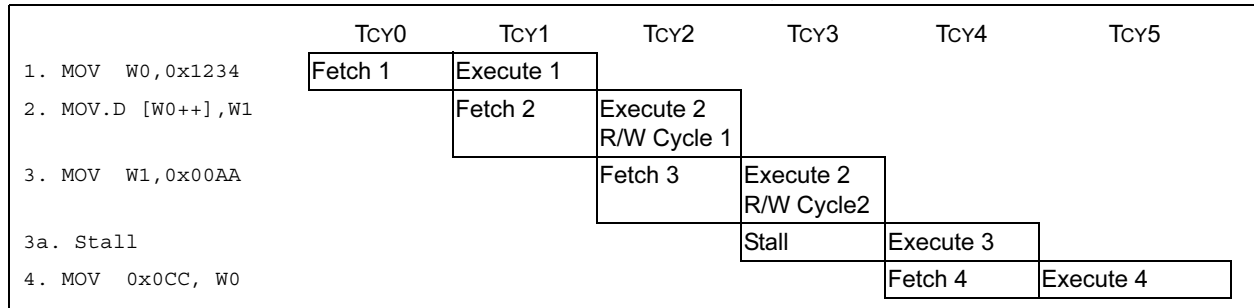
2. One-word, two-cycle (or three-cycle) instructions that are flow control instructions: these instructions include the relative branches, relative call, skips and returns. When an instruction changes the PC (other than to increment it), the pipelined fetch is discarded. This causes the instruction to take two effective cycles to execute as shown in Figure 2-3. Some instructions that change program flow require 3 cycles, such as the RETURN, RETFIE and RETLW instructions, and instructions that skip over 2-word instructions.

FIGURE 2-3: INSTRUCTION PIPELINE FLOW: 1-WORD, 2-CYCLE



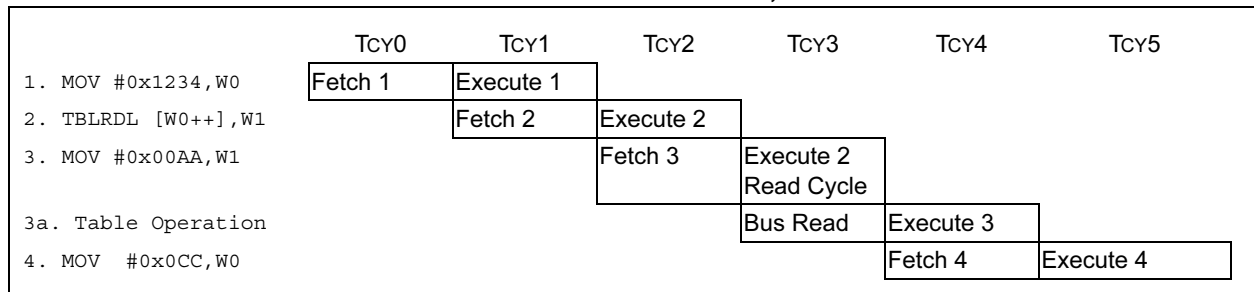
- One-word, two-cycle instructions that are not flow control instructions: the only instructions of this type are the `MOV.D` (load and store double-word) instructions, as shown in Figure 2-4.

FIGURE 2-4: INSTRUCTION PIPELINE FLOW: 1-WORD, 2-CYCLE `MOV.D` OPERATIONS



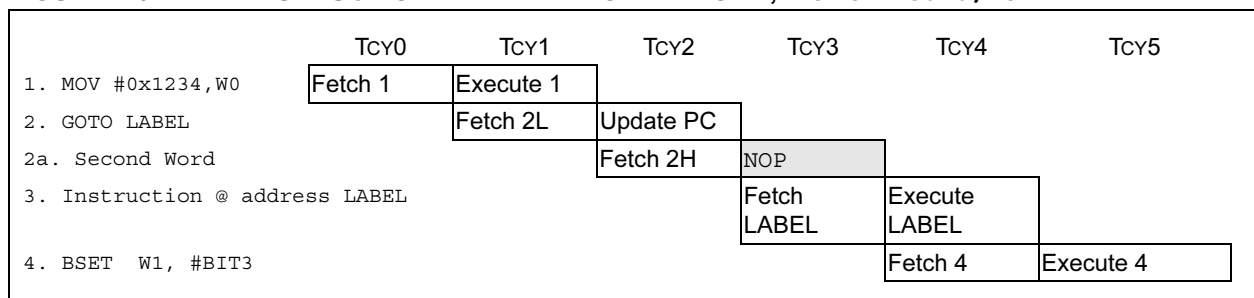
- Table read/write instructions: these instructions will suspend the fetching to insert a read or write cycle to the program memory. The instruction fetched while executing the table operation is saved for 1 cycle and executed in the cycle immediately after the table operation as shown in Figure 2-5.

FIGURE 2-5: INSTRUCTION PIPELINE FLOW: 1-WORD, 2-CYCLE TABLE OPERATIONS



- Two-word instructions for `CALL` and `GOTO`: in these instructions, the fetch after the instruction provides the remainder of the jump or call destination address. These instructions require 2 cycles to execute, 1 cycle to fetch the 2 instruction words (enabled by a high speed path on the second fetch), and 1 cycle to flush the pipeline as shown in Figure 2-6.

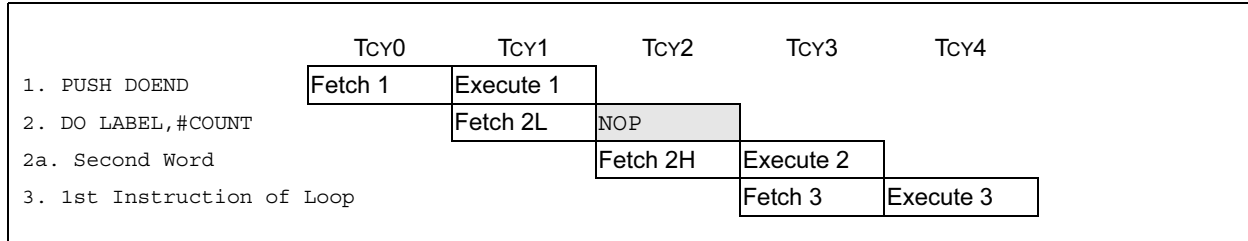
FIGURE 2-6: INSTRUCTION PIPELINE FLOW: 2-WORD, 2-CYCLE `GOTO`, `CALL`



dsPIC30F

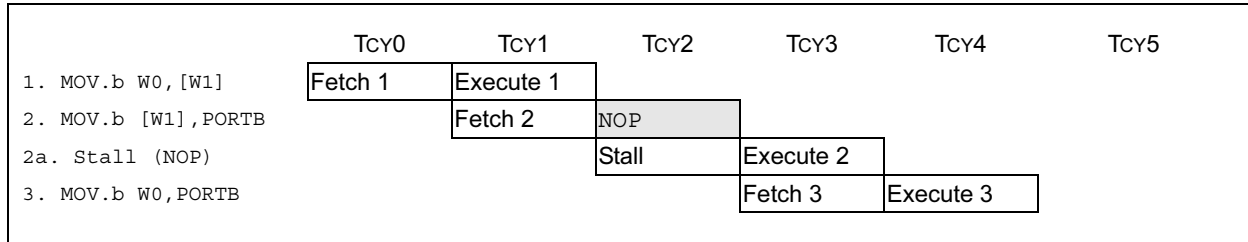
6. Two-word instructions for DO: in these instructions, the fetch after the instruction contains an address offset. This address offset is added to the first instruction address to generate the last loop instruction address. Therefore, these instructions require 2 cycles as shown in Figure 2-7.

FIGURE 2-7: INSTRUCTION PIPELINE FLOW: 2-WORD, 2-CYCLE DO, DOW



7. Instructions that are subjected to a stall due to a data dependency between the X RAGU and X WAGU: an additional cycle is inserted to resolve the resource conflict as shown in Figure 2-7. Instruction stalls caused by data dependencies are further discussed in Section 4.0.

FIGURE 2-8: INSTRUCTION PIPELINE FLOW: 1-WORD, 2-CYCLE WITH INSTRUCTION STALL



8. Interrupt recognition execution: refer to Section 6.0 for details on interrupts.

2.4 Divide Support

The dsPIC devices feature a 16/16-bit signed fractional divide operation, as well as 32/16-bit and 16/16-bit signed and unsigned integer divide operations, in the form of single instruction iterative divides. The following instructions and data sizes are supported:

1. `DIVF` - 16/16 signed fractional divide
2. `DIV.sd` - 32/16 signed divide
3. `DIV.ud` - 32/16 unsigned divide
4. `DIV.sw` - 16/16 signed divide
5. `DIV.uw` - 16/16 unsigned divide

The 16/16 divides are similar to the 32/16 (same number of iterations), but the dividend is either zero-extended or sign-extended during the first iteration.

The quotient for all divide instructions is stored in `W0`, and the remainder in `W1`. `DIV` and `DIVF` can specify any `W` register for both the 16-bit dividend and divisor. All other divides can specify any `W` register for the 16-bit divisor, but the 32-bit dividend must be in an aligned `W` register pair, such as `W1:W0`, `W3:W2`, etc.

The non-restoring divide algorithm requires one cycle for an initial dividend shift (for integer divides only), one cycle per divisor bit, and a remainder/quotient correction cycle. The correction cycle is the last cycle of the iteration loop but must be performed (even if the remainder is not required) because it may also adjust the quotient. A consequence of this is that `DIVF` will also produce a valid remainder (though it is of little use in fractional arithmetic).

The divide instructions must be executed within a `REPEAT` loop. Any other form of execution (e.g., a series of discrete divide instructions) will not function correctly because the instruction flow depends on `RCOUNT`. The divide instruction does not automatically set up the `RCOUNT` value and it must, therefore, be explicitly and correctly specified in the `REPEAT` instruction as shown in Table 2-1 (`REPEAT` will execute the target instruction {operand value+1} times). The `REPEAT` loop count must be setup for 18 iterations of the `DIV/DIVF` instruction. Thus, a complete divide operation requires 19 cycles.

Note: The divide flow is interruptible. However, the user needs to save the context as appropriate.

TABLE 2-1: DIVIDE INSTRUCTIONS

| Instruction | Function |
|--|--|
| <code>DIVF</code> | Signed fractional divide: $W_m/W_n \rightarrow W0$; Rem $\rightarrow W1$ |
| <code>DIV.sd</code> | Signed divide: $(W_{m+1}:W_m)/W_n \rightarrow W0$; Rem $\rightarrow W1$ |
| <code>DIV.sw</code> or <code>DIV.s</code> | Signed divide: $W_m/W_n \rightarrow W0$; Rem $\rightarrow W1$ |
| <code>DIV.ud</code> | Unsigned divide: $(W_{m+1}:W_m)/W_n \rightarrow W0$; Rem $\rightarrow W1$ |
| <code>DIV.uw</code> or <code>DIV.u</code> | Unsigned divide: $W_m/W_n \rightarrow W0$; Rem $\rightarrow W1$ |

2.5 DSP Engine

Concurrent operation of the DSP engine with MCU instruction flow is not possible, though both the MCU ALU and DSP engine resources may be used concurrently by the same instruction (e.g., ED and EDAC instructions).

The DSP engine consists of a high speed 17-bit x 17-bit multiplier, a barrel shifter and a 40-bit adder/subtractor (with two target accumulators, round and saturation logic).

Data input to the DSP engine is derived from one of the following:

1. Directly from the W array (registers W4, W5, W6 or W7) via the X and Y data buses for the MAC class of instructions (MAC, MSC, MPY, MPY.N, ED, EDAC, CLR and MOVSAAC).
2. From the X bus for all other DSP instructions.
3. From the X bus for all MCU instructions which use the barrel shifter.

Data output from the DSP engine is written to one of the following:

1. The target accumulator, as defined by the DSP instruction being executed.
2. The X bus for MAC, MSC, CLR and MOVSAAC accumulator writes, where the EA is derived from W13 only. (MPY, MPY.N, ED and EDAC do not offer an accumulator write option.)
3. The X bus for all MCU instructions which use the barrel shifter.

The DSP engine also has the capability to perform inherent accumulator-to-accumulator operations, which require no additional data. These instructions are ADD, SUB and NEG.

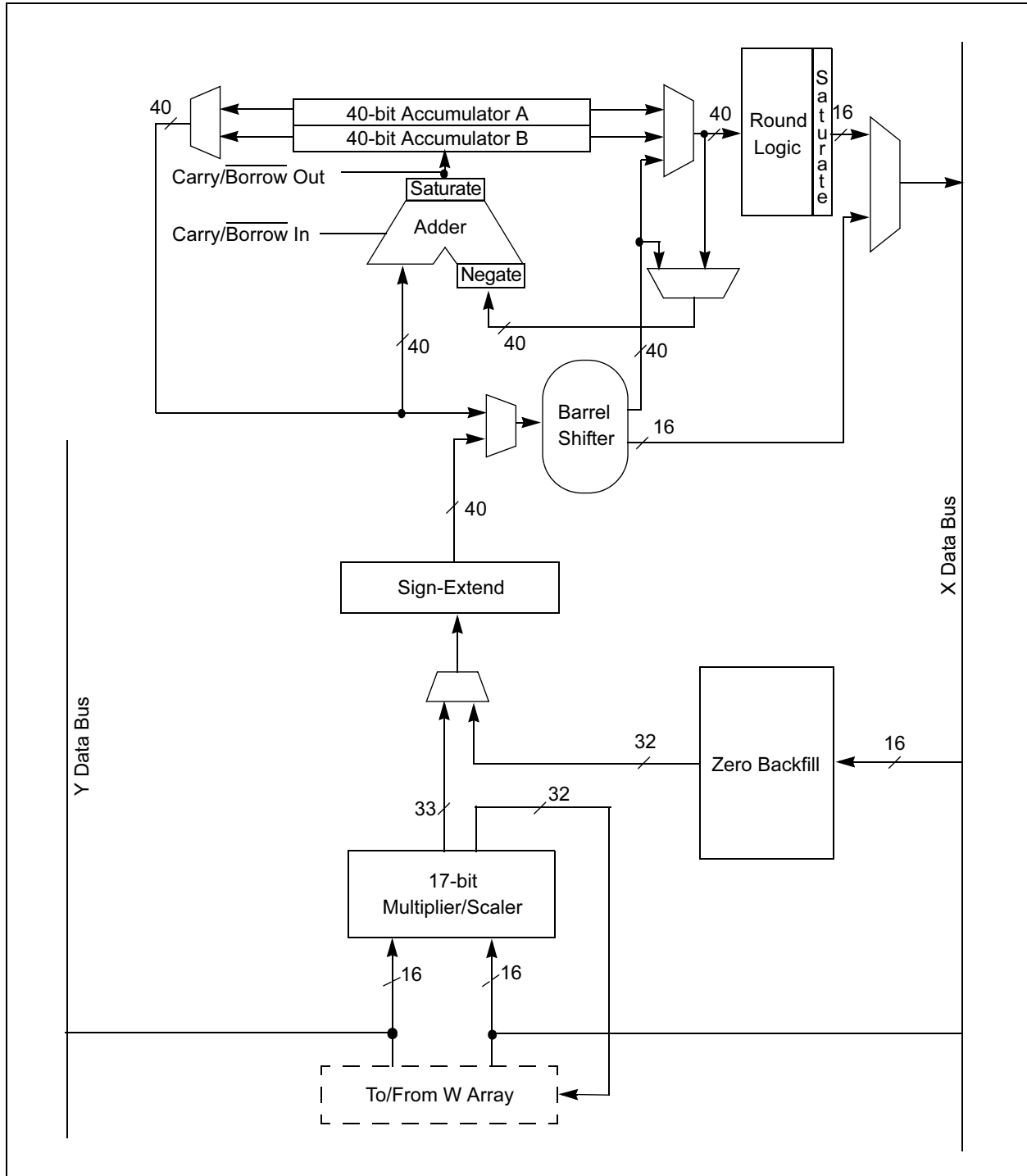
The DSP engine has various options selected through various bits in the CPU Core Configuration register (CORCON), as listed below:

1. Fractional or integer DSP multiply (IF).
2. Signed or unsigned DSP multiply (US).
3. Conventional or convergent rounding (RND).
4. Automatic saturation on/off for AccA (SATA).
5. Automatic saturation on/off for AccB (SATB).
6. Automatic saturation on/off for writes to data memory (SATDW).
7. Accumulator Saturation mode selection (ACCSAT).

Note: For CORCON layout, see Table 4-3.

A block diagram of the DSP engine is shown in Figure 2-9.

FIGURE 2-9: DSP ENGINE BLOCK DIAGRAM



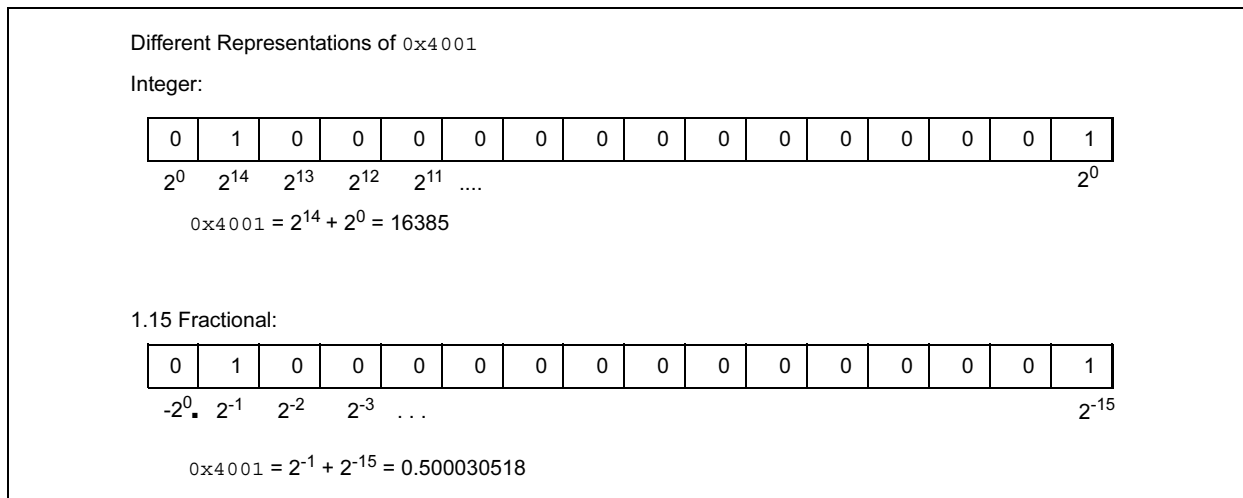
dsPIC30F

2.5.1 MULTIPLIER

The 17 x 17-bit multiplier is capable of signed or unsigned operation and can multiplex its output using a scaler to support either 1.31 fractional (Q31) or 32-bit integer results. The respective number representation formats are shown in Figure 2-10. Unsigned operands are zero-extended into the 17th bit of the multiplier input value. Signed operands are sign-extended into the 17th bit of the multiplier input value. The output of the 17 x 17-bit multiplier/scaler is a 33-bit value which is sign-extended to 40 bits. Integer data is inherently represented as a signed two's complement value, where the MSB is defined as a sign bit. Generally speaking, the range of an N-bit two's complement integer is -2^{N-1} to $2^{N-1} - 1$. For a 16-bit integer, the data range is -32768 (0x8000) to 32767 (0x7FFF) including '0' (see Figure 2-10). For a 32-bit integer, the data range is -2,147,483,648 (0x8000 0000) to 2,147,483,645 (0x7FFF FFFF).

When the multiplier is configured for fractional multiplication, the data is represented as a two's complement fraction, where the MSB is defined as a sign bit and the radix point is implied to lie just after the sign bit (QX format). The range of an N-bit two's complement fraction with this implied radix point is -1.0 to $(1 - 2^{1-N})$. For a 16-bit fraction, the Q15 data range is -1.0 (0x8000) to 0.999969482 (0x7FFF) including '0' and has a precision of 3.01518×10^{-5} . In Fractional mode, the 16x16 multiply operation generates a 1.31 product which has a precision of 4.65661×10^{-10} .

FIGURE 2-10: 16-BIT INTEGER AND FRACTIONAL MODES



Certain multiply operations always operate on signed data. These include the MAC/MSC, MPY [.N] and ED [AC] instructions. The 40-bit adder/subtractor may also optionally negate one of its operand inputs to change the result sign (without changing the operands). This is used to create a multiply and subtract (MSC), or multiply and negate (MPY .N) operation.

In the special case when both input operands are 1.15 fractions and equal to 0x8000 (-1_{10}), the result of the multiplication is corrected to 0x7FFFFFFF (as the closest approximation to +1) by hardware before it is used.

It should be noted that with the exception of DSP multiplies, the dsPIC30F ALU operates identically on integer and fractional data. Namely, an addition of two integers will yield the same result (binary number) as the addition of two fractional numbers. The only difference is how the result is interpreted by the user. However, multiplies performed by DSP operations are

different. In these instructions, data format selection is made with the IF bit (CORCON<0>) and US bits (CORCON<12>), and it must be set accordingly ('0' for Fractional mode, '1' for Integer mode in the case of the IF bit, and '0' for Signed mode, '1' for Unsigned mode in the case of the US bit). This is required because of the implied radix point used by dsPIC30F fractions. In Integer mode, multiplying two 16-bit integers produces a 32-bit integer result. However, multiplying two 1.15 values generates a 2.30 result. Since the dsPIC30F uses 1.31 format for the accumulators, a DSP multiply in Fractional mode also includes a left shift by one bit to keep the radix point properly aligned. This feature reduces the resolution of the DSP multiplier to 2^{-30} , but has no other effect on the computation.

The same multiplier is used to support the MCU multiply instructions which include integer 16-bit signed, unsigned and mixed sign multiplies. Additional data paths are provided to allow these instructions to write the result back into the W array and X data bus (via the W array). These paths are placed prior to the data scaler. The IF bit in the CORCON register, therefore, only affects the result of the MAC class of DSP instructions. All other multiply operations are assumed to be integer operations. If the user executes a MAC instruction on fractional data without clearing the IF bit, the result must be explicitly shifted left by the user program after multiplication in order to obtain the correct result.

The MUL instruction may be directed to use byte or word sized operands. Byte operands will direct a 16-bit result, and word operands will direct a 32-bit result to the specified register(s) in the W array.

2.5.2 DATA ACCUMULATORS AND ADDER/SUBTRACTER

The data accumulator consists of a 40-bit adder/subtractor with automatic sign extension logic. It can select one of two accumulators (A or B) as its pre-accumulation source and post-accumulation destination. For the ADD and LAC instructions, the data to be accumulated or loaded can be optionally scaled via the barrel shifter, prior to accumulation.

2.5.2.1 Adder/Subtractor, Overflow and Saturation

The adder/subtractor is a 40-bit adder with an optional zero input into one side and either true, or complement data into the other input. In the case of addition, the carry/borrow input is active high and the other input is true data (not complemented), whereas in the case of subtraction, the carry/borrow input is active low and the other input is complemented. The adder/subtractor generates overflow status bits SA/SB and OA/OB, which are latched and reflected in the STATUS register:

- Overflow from bit 39: this is a catastrophic overflow in which the sign of the accumulator is destroyed.
- Overflow into guard bits 32 through 39: this is a recoverable overflow. This bit is set whenever all the guard bits bits are not identical to each other.

The adder has an additional saturation block which controls accumulator data saturation, if selected. It uses the result of the adder, the overflow status bits described above, and the SATA/B (CORCON<7:6>) and ACCSAT (CORCON<4>) mode control bits to determine when and to what value to saturate.

Six STATUS register bits have been provided to support saturation and overflow; they are:

1. OA:
AccA overflowed into guard bits
2. OB:
AccB overflowed into guard bits
3. SA:
AccA saturated (bit 31 overflow and saturation)
or
AccA overflowed into guard bits and saturated (bit 39 overflow and saturation)
4. SB:
AccB saturated (bit 31 overflow and saturation)
or
AccB overflowed into guard bits and saturated (bit 39 overflow and saturation)
5. OAB:
Logical OR of OA and OB
6. SAB:
Logical OR of SA and SB

The OA and OB bits are modified each time data passes through the adder/subtractor. When set, they indicate that the most recent operation has overflowed into the accumulator guard bits (bits 32 through 39). The OA and OB bits can also optionally generate an arithmetic warning trap when set and the corresponding overflow trap flag enable bit (OVATEN, OVBTEN) in the INTCON1 register (refer to Section 5.0) is set. This allows the user to take immediate action, for example, to correct system gain.

The SA and SB bits are modified each time data passes through the adder/subtractor but can only be cleared by the user. When set, they indicate that the accumulator has overflowed its maximum range (bit 31 for 32-bit saturation, or bit 39 for 40-bit saturation) and will be saturated (if saturation is enabled). When saturation is not enabled, SA and SB default to bit 39 overflow and thus indicate that a catastrophic overflow has occurred. If the COVTE bit in the INTCON1 register is set, SA and SB bits will generate an arithmetic warning trap when saturation is disabled.

The overflow and saturation status bits can optionally be viewed in the STATUS register (SR) as the logical OR of OA and OB (in bit OAB) and the logical OR of SA and SB (in bit SAB). This allows programmers to check one bit in the STATUS register to determine if either accumulator has overflowed, or one bit to determine if either accumulator has saturated. This would be useful for complex number arithmetic which typically uses both the accumulators.

The device supports three saturation and overflow modes:

1. **Bit 39 Overflow and Saturation:**
When bit 39 overflow and saturation occurs, the saturation logic loads the maximally positive 9.31 ($0x7FFFFFFF$), or maximally negative 9.31 value ($0x80000000$) into the target accumulator. The SA or SB bit is set and remains set until cleared by the user. This is referred to as 'super saturation' and provides protection against erroneous data, or unexpected algorithm problems (e.g., gain calculations).
2. **Bit 31 Overflow and Saturation:**
When bit 31 overflow and saturation occurs, the saturation logic then loads the maximally positive 1.31 value ($0x007FFFFFFF$), or maximally negative 1.31 value ($0x00800000$) into the target accumulator. The SA or SB bit is set and remains set until cleared by the user. When this Saturation mode is in effect, the guard bits are not used (so the OA, OB or OAB bits are never set).
3. **Bit 39 Catastrophic Overflow:**
The bit 39 overflow status bit from the adder is used to set the SA or SB bit which remain set until cleared by the user. No saturation operation is performed and the accumulator is allowed to overflow (destroying its sign). If the COVTE bit in the INTCON1 register is set, a catastrophic overflow can initiate a trap exception.

2.5.2.2 Accumulator 'Write Back'

The MAC class of instructions (with the exception of MPY, MPY.N, ED and EDAC) can optionally write a rounded version of the high word (bits 31 through 16) of the accumulator that is not targeted by the instruction into data space memory. The write is performed across the X bus into combined X and Y address space. The following Addressing modes are supported:

1. **W13, Register Direct:**
The rounded contents of the non-target accumulator are written into W13 as a 1.15 fraction.
2. **[W13] += 2, Register Indirect with Post-Increment:**
The rounded contents of the non-target accumulator are written into the address pointed to by W13 as a 1.15 fraction. W13 is then incremented by 2 (for a word write).

2.5.2.3 Round Logic

The round logic is a combinational block which performs a conventional (biased) or convergent (unbiased) round function during an accumulator write (store). The Round mode is determined by the state of the RND bit in the CORCON register. It generates a 16-bit, 1.15 data value which is passed to the data space write saturation logic. If rounding is not indicated by the instruction, a truncated 1.15 data value is stored and the LS Word is simply discarded.

The two Rounding modes are shown in Figure 2-10. Conventional rounding takes bit 15 of the accumulator, zero-extends it and adds it to the ACCxH word (bits 16 through 31 of the accumulator). If the ACCxL word (bits 0 through 15 of the accumulator) is between $0x8000$ and $0xFFFF$ ($0x8000$ included), ACCxH is incremented. If ACCxL is between $0x0000$ and $0x7FFF$, ACCxH is left unchanged. A consequence of this algorithm is that over a succession of random rounding operations, the value will tend to be biased slightly positive.

Convergent (or unbiased) rounding operates in the same manner as conventional rounding, except when ACCxL equals $0x8000$. If this is the case, the LS bit (bit 16 of the accumulator) of ACCxH is examined. If it is '1', ACCxH is incremented. If it is '0', ACCxH is not modified. Assuming that bit 16 is effectively random in nature, this scheme will remove any rounding bias that may accumulate.

The SAC and SAC.R instructions store either a truncated (SAC) or rounded (SAC.R) version of the contents of the target accumulator to data memory via the X bus (subject to data saturation, see Section 2.5.2.4). Note that for the MAC class of instructions, the accumulator write back operation will function in the same manner, addressing combined MCU (X and Y) data space though the X bus. For this class of instructions, the data is always subject to rounding.

2.5.2.4 Data Space Write Saturation

In addition to adder/subtractor saturation, writes to data space may also be saturated but without affecting the contents of the source accumulator. The data space write saturation logic block accepts a 16-bit, 1.15 fractional value from the round logic block as its input, together with overflow status from the original source (accumulator) and the 16-bit round adder. These are combined and used to select the appropriate 1.15 fractional value as output to write to data space memory.

If the SATDW bit in the CORCON register is set, data (after rounding or truncation) is tested for overflow and adjusted accordingly. For input data greater than $0x007FFF$, data written to memory is forced to the maximum positive 1.15 value, $0x7FFF$. For input data less than $0xFF8000$, data written to memory is forced to the maximum negative 1.15 value, $0x8000$. The MS bit of the source (bit 39) is used to determine the sign of the operand being tested.

If the SATDW bit in the CORCON register is not set, the input data is always passed through unmodified under all conditions.

2.5.3 BARREL SHIFTER

The barrel shifter is capable of performing up to 15-bit arithmetic or logic right shifts, or up to 16-bit left shifts in a single cycle. The source can be either of the two DSP accumulators, or the X bus (to support multi-bit shifts of register or memory data).

The shifter requires a signed binary value to determine both the magnitude (number of bits) and direction of the shift operation. A positive value will shift the operand right. A negative value will shift the operand left. A value of '0' will not modify the operand.

The barrel shifter is 40-bits wide, thereby obtaining a 40-bit result for DSP shift operations and a 16-bit result for MCU shift operations. Data from the X bus is presented to the barrel shifter between bit positions 16 to 31 for right shifts, and bit positions 0 to 15 for left shifts.

dsPIC30F

NOTES:

3.0 MEMORY ORGANIZATION

Note: This data sheet summarizes features of this group of dsPIC30F devices and is not intended to be a complete reference source. For more information on the CPU, peripherals, register descriptions and general device functionality, refer to the *dsPIC30F Family Reference Manual* (DS70046). For more information on the device instruction set and programming, refer to the *dsPIC30F Programmer's Reference Manual* (DS70030).

User program space access is restricted to the lower 4M instruction word address range (0x000000 to 0x7FFFFE) for all accesses other than TBLRD/TBLWT, which use TBLPAG<7> to determine user or configuration space access. In Table 3-1, Program Space Address Construction, bit 23 allows access to the Device ID, the User ID and the configuration bits. Otherwise, bit 23 is always clear.

3.1 Program Address Space

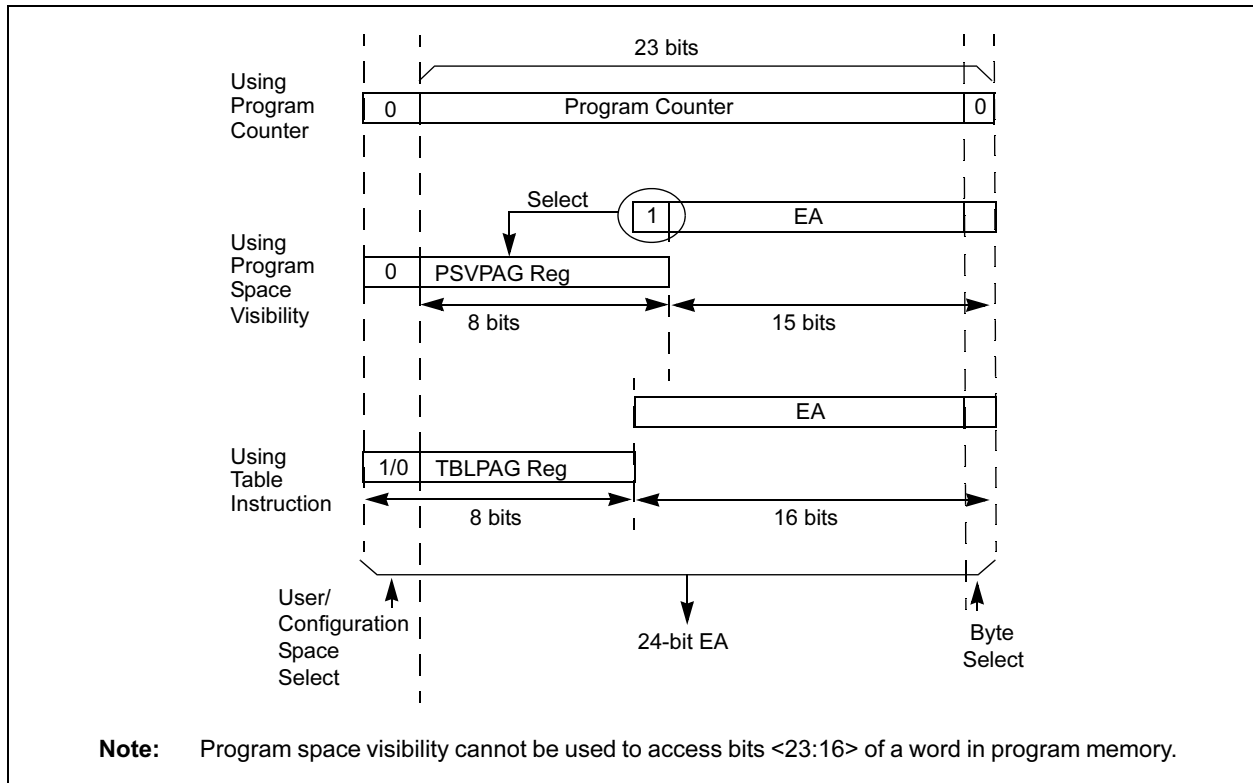
The program address space is 4M instruction words. It is addressable by a 24-bit value from either the 23-bit PC, table instruction EA, or data space EA, when program space is mapped into data space as defined by Table 3-1. Note that the program space address is incremented by two between successive program words in order to provide compatibility with data space addressing.

Note: The address map shown in Figure 3-5 is conceptual, and the actual memory configuration may vary across individual devices depending on available memory.

TABLE 3-1: PROGRAM SPACE ADDRESS CONSTRUCTION

| Access Type | Access Space | Program Space Address | | | | |
|--------------------------|----------------------------------|-----------------------|-------------|------|---------------|-----|
| | | <23> | <22:16> | <15> | <14:1> | <0> |
| Instruction Access | User | 0 | PC<22:1> | | | 0 |
| TBLRD/TBLWT | User (TBLPAG<7> = 0) | TBLPAG<7:0> | | | Data EA<15:0> | |
| TBLRD/TBLWT | Configuration (TBLPAG<7> = 1) | TBLPAG<7:0> | | | Data EA<15:0> | |
| Program Space Visibility | User | 0 | PSVPAG<7:0> | | Data EA<14:0> | |

FIGURE 3-1: DATA ACCESS FROM PROGRAM SPACE ADDRESS GENERATION



dsPIC30F

3.1.1 PROGRAM SPACE ALIGNMENT AND DATA ACCESS USING TABLE INSTRUCTIONS

This architecture fetches 24-bit wide program memory. Consequently, instructions are always aligned. However, as the architecture is modified Harvard, data can also be present in program space.

There are two methods by which program space can be accessed: via special table instructions, or through the remapping of a 16K word program space page into the upper half of data space (see Section 3.1.2). The TBLRDH and TBLWTH instructions offer a direct method of reading or writing the LS Word of any address within program space, without going through data space. The TBLRDH and TBLWTH instructions are the only method whereby the upper 8 bits of a program space word can be accessed as data.

The PC is incremented by two for each successive 24-bit program word. This allows program memory addresses to directly map to data space addresses. Program memory can thus be regarded as two 16-bit word wide address spaces, residing side by side, each with the same address range. TBLRDH and TBLWTH access the space which contains the LS Data Word, and TBLRDH and TBLWTH access the space which contains the MS Data Byte.

Figure 3-1 shows how the EA is created for table operations and data space accesses (PSV = 1). Here, P<23:0> refers to a program space word, whereas D<15:0> refers to a data space word.

A set of table instructions are provided to move byte or word sized data to and from program space.

1. TBLRDH: Table Read High
Word: Read the MS Word of the program address; P<23:16> maps to D<7:0>; D<15:8> will always be = 0.
Byte: Read one of the MS Bytes of the program address; P<23:16> maps to the destination byte when byte select = 0; The destination byte will always be = 0 when byte select = 1.
2. TBLWTH: Table Write High (refer to Section 6.0 for details on Flash Programming)
3. TBLRDH: Table Read High
Word: Read the MS Word of the program address; P<23:16> maps to D<7:0>; D<15:8> will always be = 0.
Byte: Read one of the MS Bytes of the program address; P<23:16> maps to the destination byte when byte select = 0; The destination byte will always be = 0 when byte select = 1.
4. TBLWTH: Table Write High (refer to Section 6.0 for details on Flash Programming)

FIGURE 3-2: PROGRAM DATA TABLE ACCESS (LS WORD)

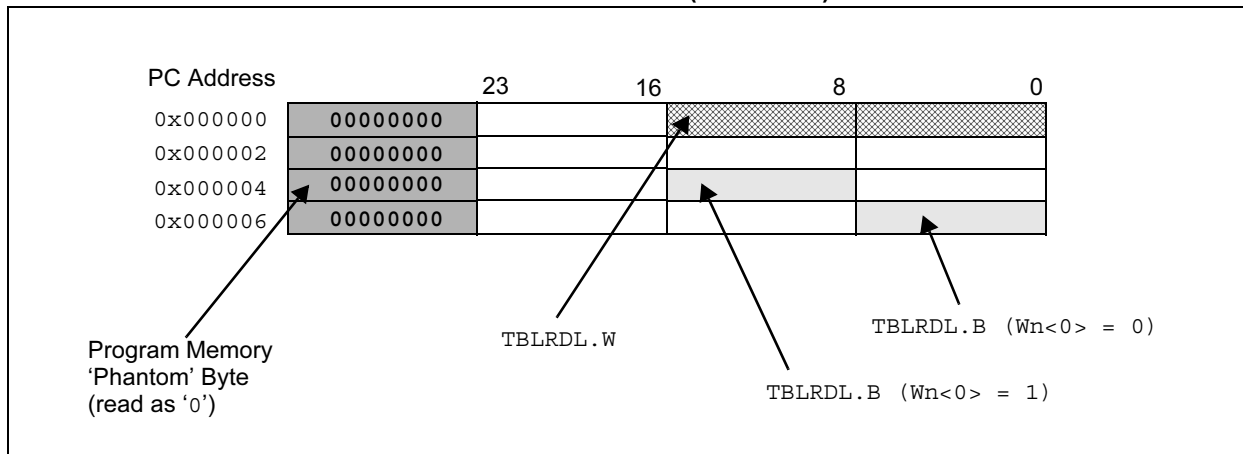
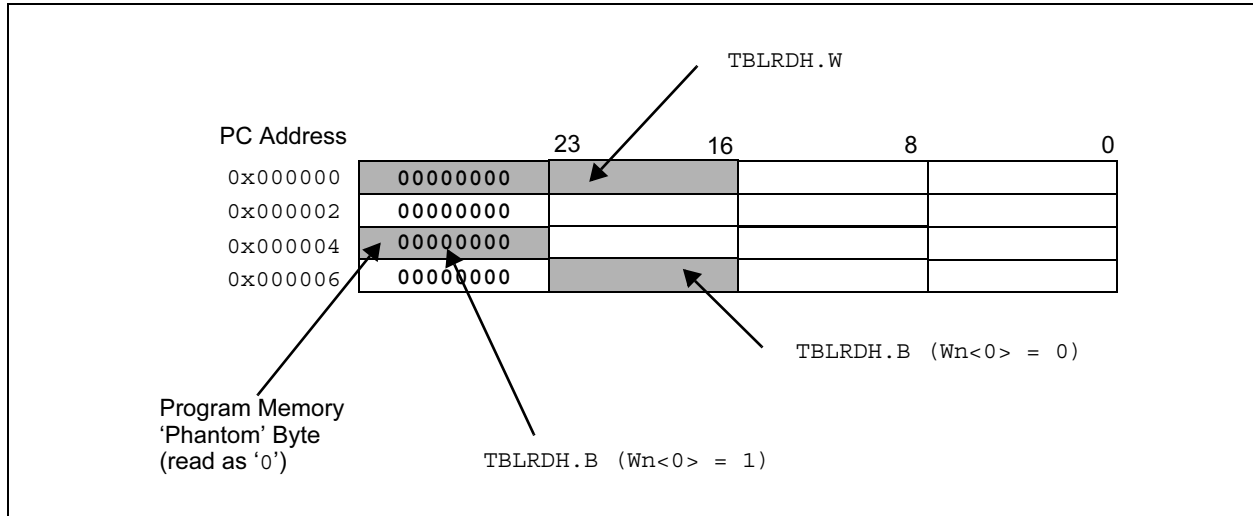


FIGURE 3-3: PROGRAM DATA TABLE ACCESS (MS BYTE)



3.1.2 PROGRAM SPACE VISIBILITY FROM DATA SPACE

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

Program space access through the data space occurs if the MS bit of the data space EA is set and program space visibility is enabled by setting the PSV bit in the Core Control register (CORCON). The functions of CORCON are discussed in Section 2.5, DSP Engine.

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

Note that the upper half of addressable data space is always part of the X data space. Therefore, when a DSP operation uses program space mapping to access this memory region, Y data space should typically contain state (variable) data for DSP operations, whereas X data space should typically contain coefficient (constant) data.

Although each data space address, 0x8000 and higher, maps directly into a corresponding program memory address (see Figure 3-4), only the lower 16 bits of the 24-bit program word are used to contain the data. The upper 8 bits should be programmed to force an illegal instruction to maintain machine robustness. Refer to the Programmer's Reference Manual (DS70030) for details on instruction encoding.

Note that by incrementing the PC by 2 for each program memory word, the LS 15 bits of data space addresses directly map to the LS 15 bits in the corresponding program space addresses. The remaining bits are provided by the Program Space Visibility Page register, PSVPAG<7:0>, as shown in Figure 3-4.

Note: PSV access is temporarily disabled during table reads/writes.

For instructions that use PSV which are executed outside a REPEAT loop:

- The following instructions will require one instruction cycle in addition to the specified execution time:
 - MAC class of instructions with data operand pre-fetch
 - MOV instructions
 - MOV.D instructions
- All other instructions will require two instruction cycles in addition to the specified execution time of the instruction.

For instructions that use PSV which are executed inside a REPEAT loop:

- The following instances will require two instruction cycles in addition to the specified execution time of the instruction:
 - Execution in the first iteration
 - Execution in the last iteration
 - Execution prior to exiting the loop due to an interrupt
 - Execution upon re-entering the loop after an interrupt is serviced
- Any other iteration of the REPEAT loop will allow the instruction accessing data, using PSV, to execute in a single cycle.

FIGURE 3-4: DATA SPACE WINDOW INTO PROGRAM SPACE OPERATION

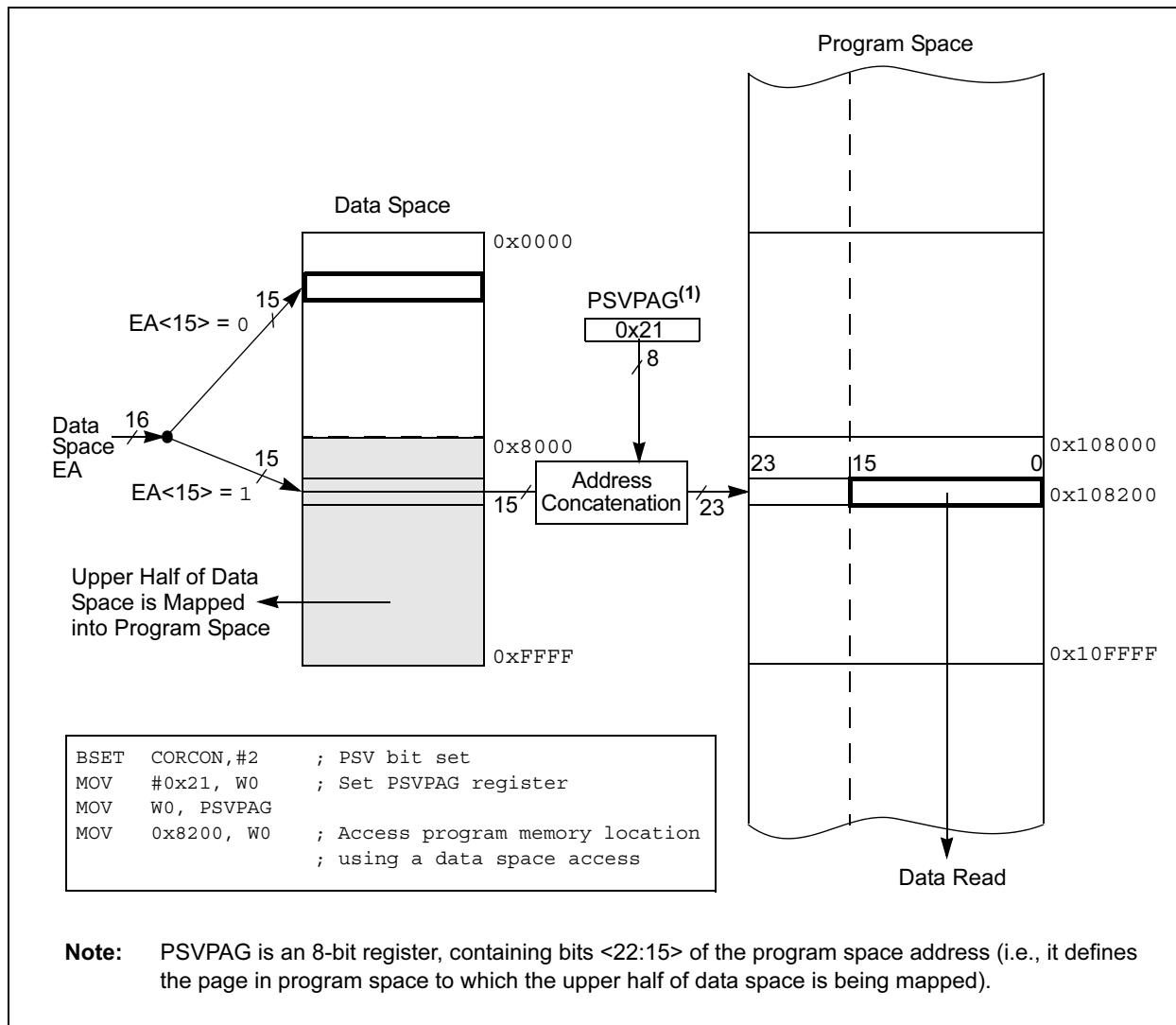
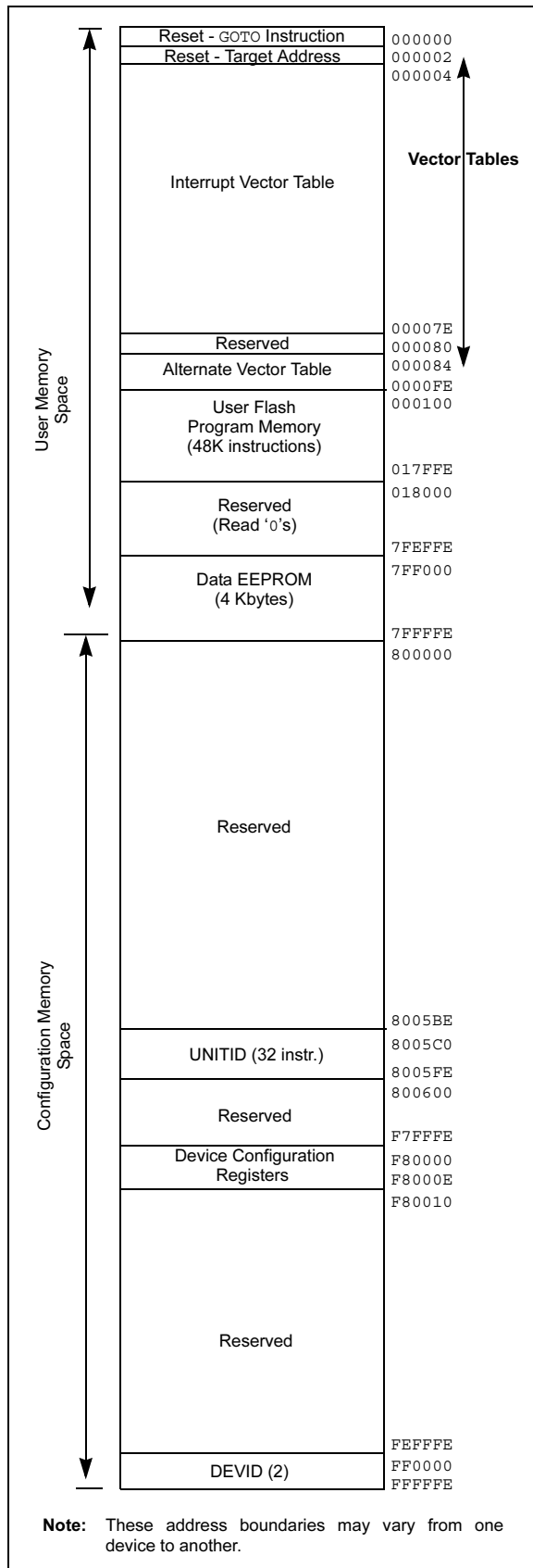


FIGURE 3-5: SAMPLE PROGRAM SPACE MEMORY MAP



3.2 Data Address Space

The core has two data spaces. The data spaces can be considered either separate (for some DSP instructions), or as one unified linear address range (for MCU instructions). The data spaces are accessed using two Address Generation Units (AGUs) and separate data paths.

3.2.1 DATA SPACES

The X data space is used by all instructions and supports all Addressing modes. There are separate read and write data buses. The X read data bus is the return data path for all instructions that view data space as combined X and Y address space. It is also the X address space data path for the dual operand read instructions (MAC class). The X write data bus is the only write path to data space for all instructions.

The X data space also supports modulo addressing for all instructions, subject to Addressing mode restrictions. Bit-reversed addressing is only supported for writes to X data space.

The Y data space is used in concert with the X data space by the MAC class of instructions (CLR, ED, EDAC, MAC, MOV SAC, MPY, MPY.N and MSC) to provide two concurrent data read paths. No writes occur across the Y bus. This class of instructions dedicates two W register pointers, W10 and W11, to always address Y data space, independent of X data space, whereas W8 and W9 always address X data space. Note that during accumulator write back, the data address space is considered a combination of X and Y data spaces, so the write occurs across the X bus. Consequently, the write can be to any address in the entire data space.

The Y data space can only be used for the data pre-fetch operation associated with the MAC class of instructions. It also supports modulo addressing for automated circular buffers. Of course, all other instructions can access the Y data address space through the X data path as part of the composite linear space.

The boundary between the X and Y data spaces is defined as shown in Figure 3-8 and is not user programmable. Should an EA point to data outside its own assigned address space, or to a location outside physical memory, an all zero word/byte will be returned. For example, although Y address space is visible by all non-MAC instructions using any Addressing mode, an attempt by a MAC instruction to fetch data from that space using W8 or W9 (X space pointers) will return 0x0000.

TABLE 3-2: EFFECT OF INVALID MEMORY ACCESSES

| Attempted Operation | Data Returned |
|---|---------------|
| EA = an unimplemented address | 0x0000 |
| W8 or W9 used to access Y data space in a MAC instruction | 0x0000 |
| W10 or W11 used to access X data space in a MAC instruction | 0x0000 |

All effective addresses are 16 bits wide and point to bytes within the data space. Therefore, the data space address range is 64 Kbytes or 32K words.

3.2.2 DATA SPACE WIDTH

The core data width is 16 bits. All internal registers are organized as 16-bit wide words. Data space memory is organized in byte addressable, 16-bit wide blocks.

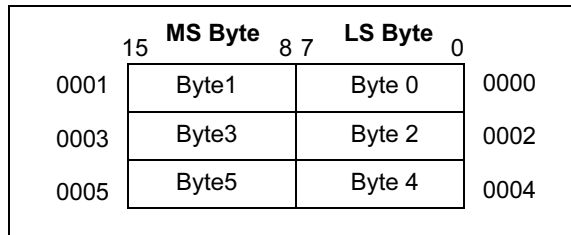
3.2.3 DATA ALIGNMENT

To help maintain backward compatibility with PICmicro® devices and improve data space memory usage efficiency, the dsPIC30F instruction set supports both word and byte operations. Data is aligned in data memory and registers as words, but all data space EAs resolve to bytes. Data byte reads will read the complete word which contains the byte, using the LS bit of any EA to determine which byte to select. The selected byte is placed onto the LS Byte of the X data path (no byte accesses are possible from the Y data path as the MAC class of instruction can only fetch words). 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 which matches the byte address.

As a consequence of this byte accessibility, all effective address calculations (including those generated by the DSP operations which are restricted to word sized data) are internally scaled to step through word aligned memory. For example, the core would recognize that Post-Modified Register Indirect Addressing mode [Ws++] will result in a value of Ws+1 for byte operations and Ws+2 for word operations.

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. Should a misaligned read or write be attempted, an address error trap will be generated. If the error occurred on a read, the instruction underway is completed, whereas if it occurred on a write, the instruction will be executed but the write will not occur. In either case, a trap will then be executed, allowing the system and/or user to examine the machine state prior to execution of the address fault.

FIGURE 3-6: DATA ALIGNMENT



All byte loads into any W register are loaded into the LS Byte. The MSB is not modified.

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

Although most instructions are capable of operating on word or byte data sizes, it should be noted that some instructions, including the DSP instructions, operate only on words.

3.2.4 DATA SPACE MEMORY MAP

The data space memory is split into two blocks, X and Y data space. A key element of this architecture is that Y space is a subset of X space, and is fully contained within X space. In order to provide an apparent linear addressing space, X and Y spaces have contiguous addresses.

When executing any instruction other than one of the MAC class of instructions, the X block consists of the 64-Kbyte data address space (including all Y addresses). When executing one of the MAC class of instructions, the X block consists of the 64-Kbyte data address space excluding the Y address block (for data reads only). In other words, all other instructions regard the entire data memory as one composite address space. The MAC class instructions extract the Y address space from data space and address it using EAs sourced from W10 and W11. The remaining X data space is addressed using W8 and W9. Both address spaces are concurrently accessed only with the MAC class instructions.

An example data space memory map is shown in Figure 3-8.

3.2.5 NEAR DATA SPACE

An 8-Kbyte 'near' data space is reserved in X address memory space between $0x0000$ and $0x1FFF$, which is directly addressable via a 13-bit absolute address field within all memory direct instructions. The remaining X address space and all of the Y address space is addressable indirectly. Additionally, the whole of X data space is addressable using MOV instructions, which support memory direct addressing with a 16-bit address field.

The stack pointer always points to the first available free word and grows from lower addresses towards higher addresses. It pre-decrements for stack pops and post-increments for stack pushes as shown in Figure 3-7. Note that 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 will concatenate the SRL register to the MSB of the PC prior to the push.

3.2.6 SOFTWARE STACK

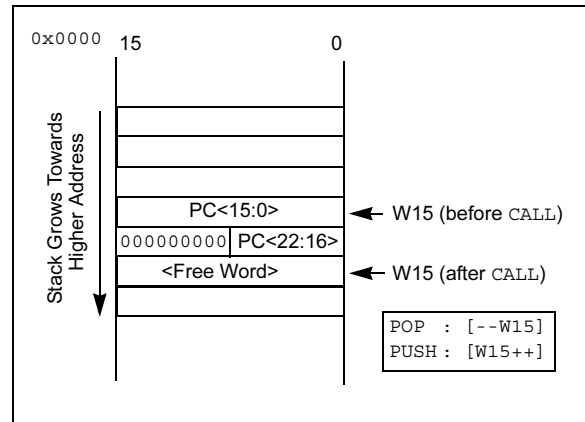
The dsPIC devices contain a software stack. W15 is used as the stack pointer.

There is a Stack Pointer Limit register (SPLIM) associated with the stack pointer. 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 effective address (EA) is generated using W15 as a source or destination pointer, the address thus generated 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 $0x0800$, thus preventing 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 3-7: CALL STACK FRAME



dsPIC30F

FIGURE 3-8: SAMPLE DATA SPACE MEMORY MAP

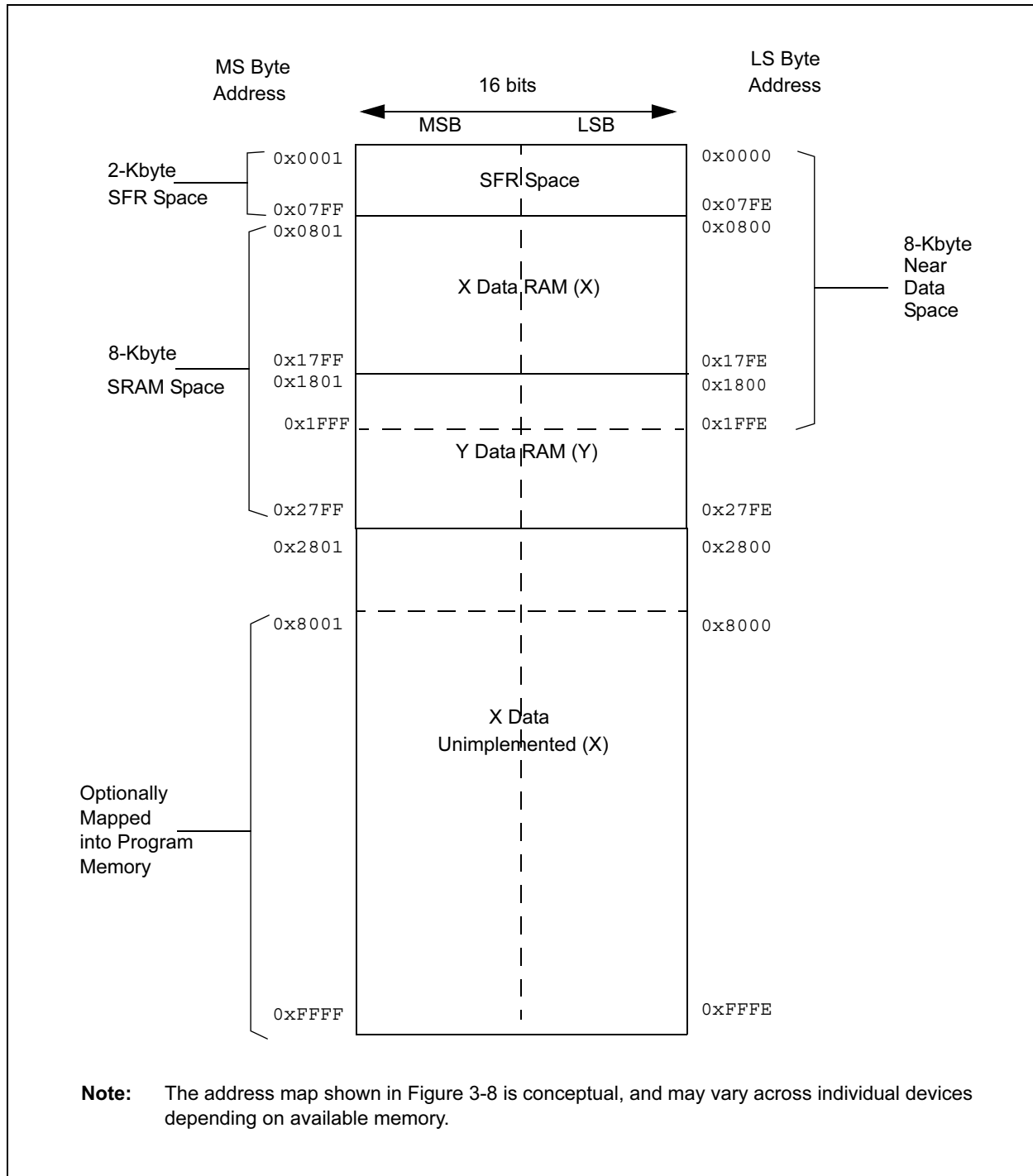


FIGURE 3-9: DATA SPACE FOR MCU AND DSP (MAC CLASS) INSTRUCTIONS EXAMPLE

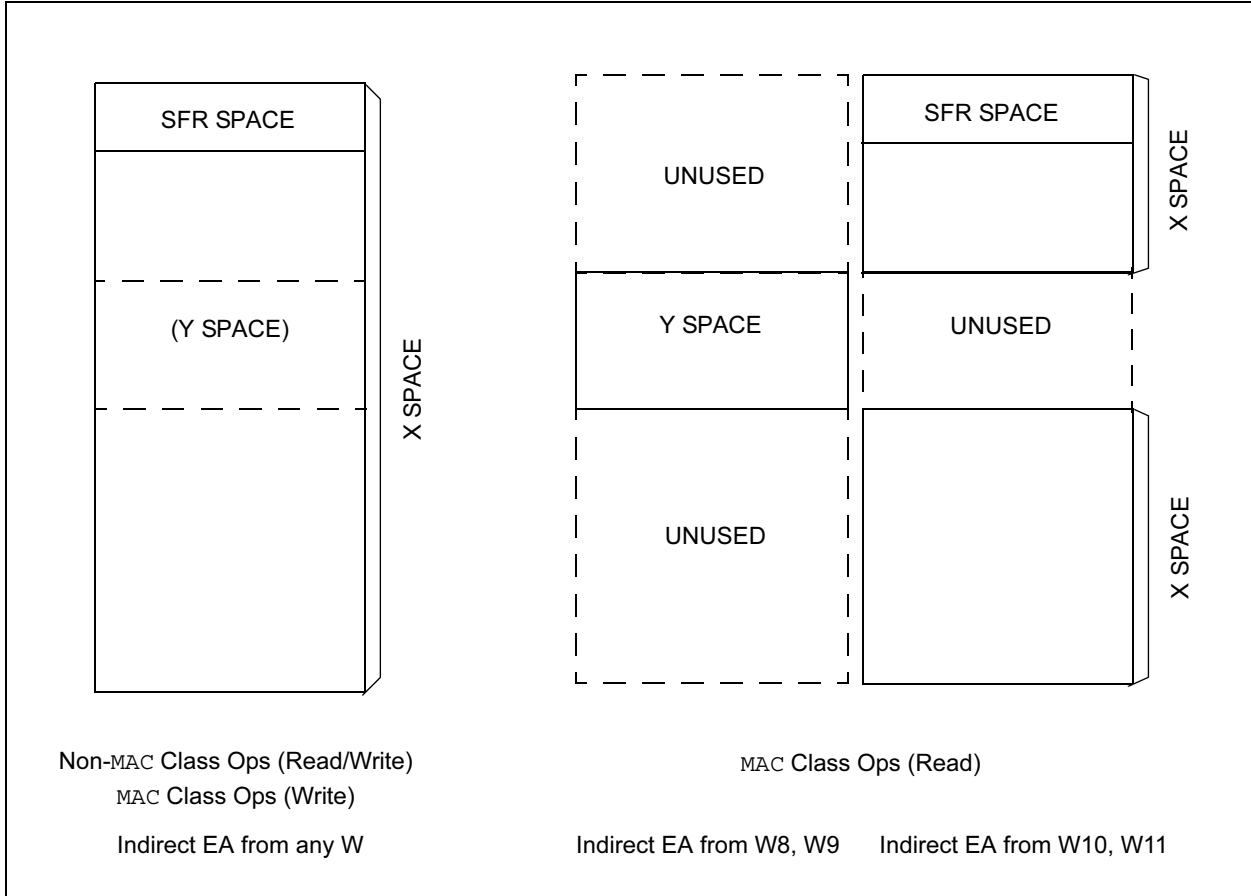


TABLE 3-3: CORE REGISTER MAP

| SFR Name | Address (Home) | 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 | Reset State |
|----------|----------------|---------------------------|--------|--------|--------|--------|--------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|---------------------|
| W0 | 0000 | W0 / WREG | | | | | | | | | | | | | | | | 0000 0000 0000 0000 |
| W1 | 0002 | W1 | | | | | | | | | | | | | | | | 0000 0000 0000 0000 |
| W2 | 0004 | W2 | | | | | | | | | | | | | | | | 0000 0000 0000 0000 |
| W3 | 0006 | W3 | | | | | | | | | | | | | | | | 0000 0000 0000 0000 |
| W4 | 0008 | W4 | | | | | | | | | | | | | | | | 0000 0000 0000 0000 |
| W5 | 000A | W5 | | | | | | | | | | | | | | | | 0000 0000 0000 0000 |
| W6 | 000C | W6 | | | | | | | | | | | | | | | | 0000 0000 0000 0000 |
| W7 | 000E | W7 | | | | | | | | | | | | | | | | 0000 0000 0000 0000 |
| W8 | 0010 | W8 | | | | | | | | | | | | | | | | 0000 0000 0000 0000 |
| W9 | 0012 | W9 | | | | | | | | | | | | | | | | 0000 0000 0000 0000 |
| W10 | 0014 | W10 | | | | | | | | | | | | | | | | 0000 0000 0000 0000 |
| W11 | 0016 | W11 | | | | | | | | | | | | | | | | 0000 0000 0000 0000 |
| W12 | 0018 | W12 | | | | | | | | | | | | | | | | 0000 0000 0000 0000 |
| W13 | 001A | W13 | | | | | | | | | | | | | | | | 0000 0000 0000 0000 |
| W14 | 001C | W14 | | | | | | | | | | | | | | | | 0000 0000 0000 0000 |
| W15 | 001E | W15 | | | | | | | | | | | | | | | | 0000 1000 0000 0000 |
| SPLIM | 0020 | SPLIM | | | | | | | | | | | | | | | | 0000 0000 0000 0000 |
| ACCAL | 0022 | ACCAL | | | | | | | | | | | | | | | | 0000 0000 0000 0000 |
| ACCAH | 0024 | ACCAH | | | | | | | | | | | | | | | | 0000 0000 0000 0000 |
| ACCAU | 0026 | Sign-Extension (ACCA<39>) | | | | | | | | | | | | | | | | 0000 0000 0000 0000 |
| ACCBH | 0028 | ACCBH | | | | | | | | | | | | | | | | 0000 0000 0000 0000 |
| ACCBH | 002A | ACCBH | | | | | | | | | | | | | | | | 0000 0000 0000 0000 |
| ACCBH | 002C | Sign-Extension (ACCB<39>) | | | | | | | | | | | | | | | | 0000 0000 0000 0000 |
| ACCBH | 002E | PCL | | | | | | | | | | | | | | | | 0000 0000 0000 0000 |
| PCH | 0030 | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | 0000 0000 0000 0000 |
| TBLPAG | 0032 | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | 0000 0000 0000 0000 |
| PSVPAG | 0034 | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | 0000 0000 0000 0000 |
| RCOUNT | 0036 | RCOUNT | | | | | | | | | | | | | | | | uuuu uuuu uuuu uuuu |
| DCOUNT | 0038 | DCOUNT | | | | | | | | | | | | | | | | uuuu uuuu uuuu uuuu |
| DOSTARTL | 003A | DOSTARTL | | | | | | | | | | | | | | | | uuuu uuuu uuuu uuuu |
| DOSTARTH | 003C | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | 0000 0000 0000 0000 |
| DOENDL | 003E | DOENDL | | | | | | | | | | | | | | | | uuuu uuuu uuuu uuuu |
| DOENDH | 0040 | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | 0000 0000 0000 0000 |
| SR | 0042 | OA | OB | SA | SB | OAB | SAB | DA | DC | IPL2 | IPL1 | IPL0 | RA | N | OV | Z | C | 0000 0000 0000 0000 |

Legend: u = uninitialized bit

TABLE 3-3: CORE REGISTER MAP (CONTINUED)

| SFR Name | Address (Home) | 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 | Reset State |
|----------|----------------|--------|--------|--------|--------|--------|----------|----------|-------|---------------|----------|-------|--------|-------|----------|-------|-------|---------------------|
| CORCON | 0044 | — | — | — | US | EDT | DL2 | DL1 | DL0 | SATA | SATB | SATDW | ACCSAT | IPL3 | PSV | RND | IF | 0000 0000 0010 0000 |
| MODCON | 0046 | XMODEN | YMODEN | — | — | — | BWM<3:0> | — | — | — | YWM<3:0> | — | — | — | XWM<3:0> | — | — | 0000 0000 0000 0000 |
| XMODSRT | 0048 | — | — | — | — | — | — | XS<15:1> | — | — | — | — | — | — | — | — | 0 | uuuu uuuu uuuu uuu0 |
| XMODEND | 004A | — | — | — | — | — | — | XE<15:1> | — | — | — | — | — | — | — | — | 1 | uuuu uuuu uuuu uuu1 |
| YMODSRT | 004C | — | — | — | — | — | — | YS<15:1> | — | — | — | — | — | — | — | — | 0 | uuuu uuuu uuuu uuu0 |
| YMODEND | 004E | — | — | — | — | — | — | YE<15:1> | — | — | — | — | — | — | — | — | 1 | uuuu uuuu uuuu uuu1 |
| XBREV | 0050 | BREN | — | — | — | — | — | — | — | XB<14:0> | — | — | — | — | — | — | — | uuuu uuuu uuuu uuuu |
| DISICNT | 0052 | — | — | — | — | — | — | — | — | DISICNT<13:0> | — | — | — | — | — | — | — | 0000 0000 0000 0000 |

Legend: u = uninitialized bit

Note: Refer to *dsPIC30F Family Reference Manual (DS70046)* for descriptions of register bit fields.

dsPIC30F

NOTES:

4.0 ADDRESS GENERATOR UNITS

Note: This data sheet summarizes features of this group of dsPIC30F devices and is not intended to be a complete reference source. For more information on the CPU, peripherals, register descriptions and general device functionality, refer to the *dsPIC30F Family Reference Manual* (DS70046). For more information on the device instruction set and programming, refer to the *dsPIC30F Programmer's Reference Manual* (DS70030).

The dsPIC core contains two independent address generator units: the X AGU and Y AGU. Further, the X AGU has two parts: X RAGU (Read AGU) and X WAGU (Write AGU). The X RAGU and X WAGU support byte and word sized data space reads and writes for both MCU and DSP instructions. The Y AGU supports word sized data reads for the DSP MAC class of instructions only. They are each capable of supporting two types of data addressing:

- Linear Addressing
- Modulo (Circular) Addressing

In addition, the X WAGU can support:

- Bit-Reversed Addressing

Linear and Modulo Data Addressing modes can be applied to data space or program space. Bit-reversed addressing is only applicable to data space addresses.

4.1 Data Space Organization

Although the data space memory is organized as 16-bit words, all effective addresses (EAs) are byte addresses. Instructions can thus access individual bytes as well as properly aligned words. Word addresses must be aligned at even boundaries. Misaligned word accesses are not supported, and if attempted, will initiate an address error trap.

When executing instructions which require just one source operand to be fetched from data space, the X RAGU and X WAGU are used to calculate the effective address. The X RAGU and X WAGU can generate any address in the 64-Kbyte data space. They support all MCU Addressing modes and modulo addressing for low overhead circular buffers. The X WAGU also supports bit-reversed addressing to facilitate FFT data reorganization.

When executing instructions which require two source operands to be concurrently fetched (i.e., the MAC class of DSP instructions), both the X RAGU and Y AGU are used simultaneously and the data space is split into two independent address spaces, X and Y. The Y AGU supports register indirect post-modified and modulo addressing only.

Note: The data write phase of the MAC class of instructions does not split X and Y address space. The write EA is calculated using the X WAGU and the data space is configured for full 64-Kbyte access.

In the Split Data Space mode, some W register address pointers are dedicated to X RAGU, and others to Y AGU. The EAs of each operand must, therefore, be restricted within different address spaces. If they are not, one of the EAs will be outside the address space of the corresponding data space (and will fetch the bus default value, 0x0000).

4.2 Instruction Addressing Modes

The Addressing modes in Table 4-1 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.

Some Addressing mode combinations may lead to a one-cycle stall during instruction execution, or are not allowed, as discussed in Section 4.3.

TABLE 4-1: 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 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 | The sum of Wn and Wb forms the EA. |
| Register Indirect with Literal Offset | The sum of Wn and a literal forms the EA. |

4.2.1 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. These memory locations are known as File registers. 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 can use a 16-bit address field.

4.2.2 MCU INSTRUCTIONS

The three-operand MCU instructions are of the 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 the W register fetched from data memory or 5-bit literal. In two-operand instructions, the result location is the same as that of one of the operands. Certain MCU instructions are one-operand operations. 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

Note: Not all instructions support all the Addressing modes given above. Individual instructions may support different subsets of these Addressing modes.

4.2.3 MOVE AND ACCUMULATOR INSTRUCTIONS

Move instructions and the DSP accumulator class of instructions provide a greater degree of addressing flexibility than 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 between 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 given above. Individual instructions may support different subsets of these Addressing modes.

4.2.4 MAC INSTRUCTIONS

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

The 2 source operand pre-fetch registers must be a member of the set {W8, W9, W10, W11}. For data reads, W8 and W9 will always be directed to the X RAGU and W10 and W11 will always be 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 is only available 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)

4.2.5 OTHER INSTRUCTIONS

Besides the various Addressing modes outlined above, 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 `NOB`, do not have any operands.

4.3 Instruction Stalls

4.3.1 INTRODUCTION

In order to maximize data space, EA calculation and operand fetch time, the X data space read and write accesses are partially pipelined. The latter half of the read phase overlaps the first half of the write phase of an instruction, as shown in Section 2.0.

Address register data dependencies, also known as 'Read After Write' (RAW) dependencies may, therefore, arise between successive read and write operations using common registers. They occur across instruction boundaries and are detected by the hardware.

An example of a RAW dependency is a write operation (in the current instruction) that modifies W5, followed by a read operation (in the next instruction) that uses W5 as a source address pointer. W5 will not be valid for the read operation until the earlier write completes. This problem is resolved by stalling the instruction execution for one instruction cycle, thereby allowing the write to complete before the next read is started.

4.3.2 RAW DEPENDENCY DETECTION

During the instruction pre-decode, the core determines if any address register dependency is imminent across an instruction boundary. The stall detection logic compares the W register (if any) used for the destination EA of the instruction currently being executed, with the W register to be used by the source EA (if any) of the pre-fetched instruction. As the W registers are also memory mapped, the stall detection logic also derives an SFR address from the W register being used by the destination EA, and determines whether this address is being issued during the write phase of the instruction currently being executed.

When it observes a match between the destination and source registers, a set of rules is applied to decide whether or not to stall the instruction by one cycle. Table 4-2 lists out the various RAW conditions which cause an instruction execution stall.

TABLE 4-2: RAW DEPENDENCY RULES (DETECTION BY HARDWARE)

| Destination Addressing Mode Using Wn | Source Addressing Mode Using Wn | Status | Examples (Wn = W2) |
|---|---|----------|---|
| Direct | Direct | No Stall | ADD.w W0, W1, W2 MOV.w W2, W3 |
| Direct | Indirect | Stall | ADD.w W0, W1, W2 MOV.w [W2], W3 |
| Direct | Indirect with Pre- or Post-Modification | Stall | ADD.w W0, W1, W2 MOV.w [W2++], W3 |
| Indirect | Direct | No Stall | ADD.w W0, W1, [W2] MOV.w W2, W3 |
| Indirect | Indirect | No Stall | ADD.w W0, W1, [W2] MOV.w [W2], W3 |
| Indirect | Indirect | Stall | ADD.w W0, W1, [W2] ; W2=0x0004 (mapped W2) MOV.w [W2], W3 ; (i.e. if W2 = addr. of W2) |
| Indirect | Indirect with Pre- or Post-Modification | No Stall | ADD.w W0, W1, [W2] MOV.w [W2++], W3 |
| Indirect | Indirect with Pre- or Post-Modification | Stall | ADD.w W0, W1, [W2] ; W2=0x0004 (mapped W2) MOV.w [W2++], W3 ; (i.e. if W2 = addr. of W2) |
| Indirect with Pre- or Post-Modification | Direct | No Stall | ADD.w W0, W1, [W2++] MOV.w W2, W3 |
| Indirect with Pre- or Post-Modification | Indirect | Stall | ADD.w W0, W1, [W2++] MOV.w [W2], W3 |
| Indirect with Pre- or Post-Modification | Indirect with Pre- or Post-Modification | Stall | ADD.w W0, W1, [W2++] MOV.w [W2++], W3 |

4.4 Modulo Addressing

Modulo addressing is a method of providing an automated means to support circular data buffers using hardware. The objective is to remove the need for software to perform data address boundary checks when executing tightly looped code, as is typical in many DSP algorithms.

Modulo addressing can operate in either data or program space (since the data pointer mechanism is essentially the same for both). One circular buffer can be supported in each of the X (which also provides the pointers into program space) and Y data spaces. Modulo addressing can operate on any W register pointer. However, it is not advisable to use W14 or W15 for modulo addressing since these two registers are used as the stack frame pointer and stack pointer, respectively.

In general, any particular circular buffer can only be configured to operate in one direction, as there are certain restrictions on the buffer start address (for incrementing buffers), or end address (for decrementing buffers) based upon the direction of the buffer.

The only exception to the usage restrictions is for buffers which have a power-of-2 length. As these buffers satisfy the start and end address criteria, they may operate in a Bidirectional mode (i.e., address boundary checks will be performed on both the lower and upper address boundaries).

4.4.1 START AND END ADDRESS

The modulo addressing scheme requires that a starting and an ending address be specified and loaded into the 16-bit Modulo Buffer Address registers: XMODSRT, XMODEND, YMODSRT, YMODEND (see Table 3-3).

Note: The start and end addresses are the first and last byte addresses of the buffer (irrespective of whether it is a word or byte buffer, or an increasing or decreasing buffer). Moreover, the start address must be even and the end address must be odd (for both word and byte buffers).

If the length of an incrementing buffer is greater than $M = 2^{N-1}$, but not greater than $M = 2^N$ bytes, then the last 'N' bits of the data buffer start address must be zeros. There are no such restrictions on the end address of an incrementing buffer. For example, if the buffer size (modulus value) is chosen to be 100 bytes ($0x64$), then the buffer start address for an incrementing buffer must contain 7 Least Significant zeros. Valid start addresses may, therefore, be $0xXX00$ and $0xXX80$, where 'X' is any hexadecimal value. Adding the buffer length to this value and subtracting '1' will give the end address to be written into X/YMODEND.

For example, if the start address was chosen to be $0x2000$, then the X/YMODEND would be set to $(0x2000 + 0x0064 - 1) = 0x2063$.

Note: 'Start address' refers to the smallest address boundary of the circular buffer. The first access of the buffer may be at any address within the modulus range (see Section 4.4.4).

In the case of a decrementing buffer, the last 'N' bits of the data buffer end address must be ones. There are no such restrictions on the start address of a decrementing buffer. For example, if the buffer size (modulus value) is chosen to be 100 bytes ($0x64$), then the buffer end address for a decrementing buffer must contain 7 Least Significant ones. Valid end addresses may, therefore, be $0xFFFF$ and $0xFF7F$, where 'x' is any hexadecimal value. Subtracting the buffer length from this value and adding 1 will give the start address to be written into X/YMODSRT. For example, if the end address was chosen to be $0x207F$, then the start address would be $(0x207F - 0x0064 + 1) = 0x201C$, which is the first physical address of the buffer.

Note: Y space modulo addressing EA calculations assume word sized data (LS bit of every EA is always clear).

The length of a circular buffer is not directly specified. It is determined by the difference between the corresponding start and end addresses. The maximum possible length of the circular buffer is 32K words (64 Kbytes).

A write operation to the MODCON register should not be immediately followed by an indirect read operation using any W register.

Note 1: Using a POP instruction to pop the contents of the top-of-stack (TOS) location into MODCON also constitutes a write to MODCON. Therefore, the instruction immediately following such a POP cannot be any instruction performing an indirect read operation.

2: It should be noted that some instructions perform an indirect read operation implicitly. These are: POP, RETURN, RETFIE, RETLW and ULNK.

4.4.2 W ADDRESS REGISTER SELECTION

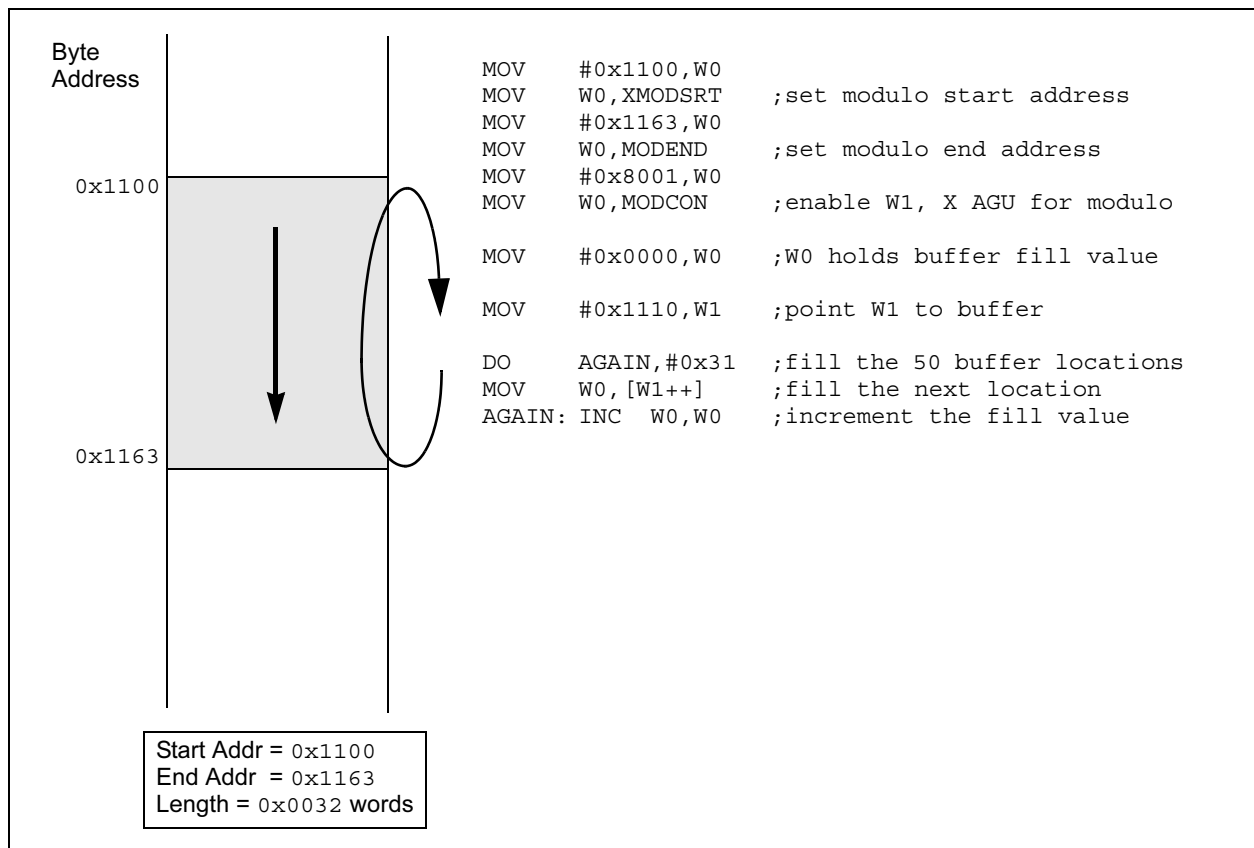
The Modulo and Bit-Reversed Addressing Control register MODCON<15:0> contains enable flags as well as a W register field to specify the W address registers. The XWM and YWM fields select which registers will operate with modulo addressing. If XWM = 15, X RAGU and X WAGU modulo addressing is disabled. Similarly, if YWM = 15, Y AGU modulo addressing is disabled.

Note: The XMODSRT and XMODEND registers and the XWM register selection are shared between X RAGU and X WAGU.

The X Address Space Pointer W register (XWM), to which modulo addressing is to be applied, is stored in MODCON<3:0> (see Table 3-3). Modulo addressing is enabled for X data space when XWM is set to any value other than '15' and the XMODEN bit is set at MODCON<15>.

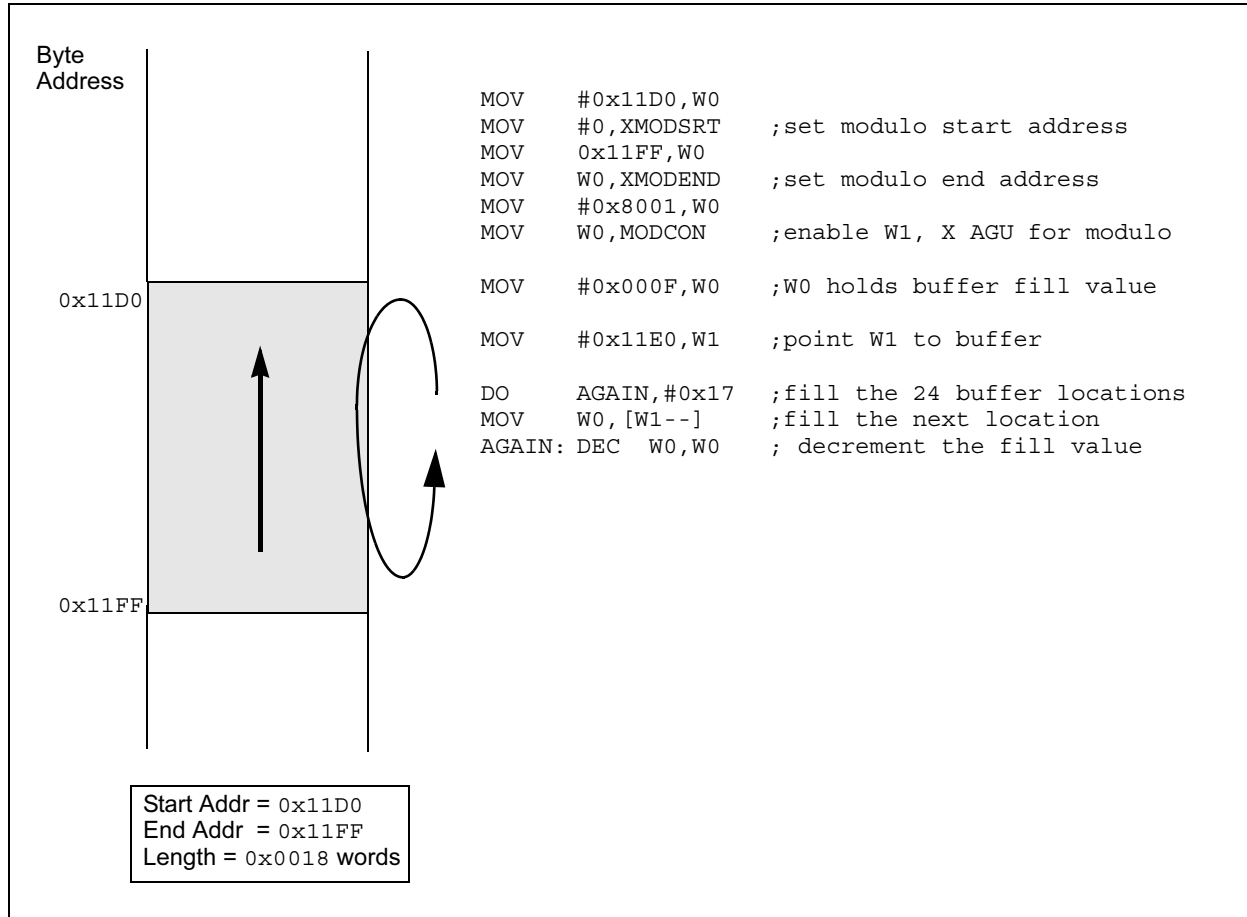
The Y Address Space Pointer W register (YWM), to which modulo addressing is to be applied, is stored in MODCON<7:4>. Modulo addressing is enabled for Y data space when YWM is set to any value other than '15' and the YMODEN bit is set at MODCON<14>.

FIGURE 4-1: INCREMENTING BUFFER MODULO ADDRESSING OPERATION EXAMPLE



dsPIC30F

FIGURE 4-2: DECREMENTING BUFFER MODULO ADDRESSING OPERATION EXAMPLE



4.4.3 MODULO ADDRESSING APPLICABILITY

Modulo addressing can be applied to the effective address (EA) calculation associated with any *W* register. It is important to realize that the address boundaries check for addresses less than, or greater than the upper (for incrementing buffers), and lower (for decrementing buffers) boundary addresses (not just equal to). Address changes may, therefore, jump over boundaries and still be adjusted correctly (see Section 4.4.4 for restrictions).

Note: The modulo corrected effective address is written back to the register only when Pre-Modify or Post-Modify Addressing mode is used to compute the effective address. When an address offset (e.g., $[W7+W2]$) is used, modulo address correction is performed but the contents of the register remain unchanged.

4.4.4 MODULO ADDRESSING RESTRICTIONS

For an incrementing buffer, the circular buffer start address (lower boundary) is arbitrary but must be at a 'zero' power-of-two boundary (see Section 4.4.1). For a decrementing buffer, the circular buffer end address is arbitrary but must be at a 'ones' boundary.

There are no restrictions regarding how much an EA calculation can exceed the address boundary being checked and still be successfully corrected.

Once configured, the direction of successive addresses into a buffer should not be changed. Although all EAs will continue to be generated correctly, irrespective of offset sign, only one address boundary is checked for each type of buffer. Thus, if a buffer is set up to be an incrementing buffer by choosing an appropriate starting address, then correction of the effective address will be performed by the AGU at the upper address boundary, but no address correction will occur if the EA crosses the lower address boundary. Similarly, for a decrementing boundary, address correction will be performed by the AGU at the lower address boundary, but no address correction will take place if the EA crosses the upper address boundary. The circular buffer pointer may be freely modified in both directions without a possibility of out-of-range address access only when the start address satisfies the condition for an incrementing buffer (last 'N' bits are zeroes) and the end address satisfies the condition for a decrementing buffer (last 'N' bits are ones). Thus, the modulo addressing capability is truly bidirectional only for modulo-2 length buffers.

4.5 Bit-Reversed Addressing

Bit-reversed addressing is intended to simplify data re-ordering for radix-2 FFT algorithms. It is supported by the X WAGU only (i.e., for data writes only).

The modifier, which may be a constant value or register contents, is regarded as having its bit order reversed. The address source and destination are kept in normal order. Thus, the only operand requiring reversal is the modifier.

4.5.1 BIT-REVERSED ADDRESSING IMPLEMENTATION

Bit-reversed addressing is enabled when:

1. BWM (*W* register selection) in the MODCON register is any value other than '15' (the stack cannot be accessed using bit-reversed addressing) **and**
2. the BREN bit is set in the XBREV register **and**
3. the Addressing mode used is Register Indirect with Pre-Increment or Post-Increment.

If the length of a bit-reversed buffer is $M = 2^N$ bytes, then the last 'N' bits of the data buffer start address must be zeros.

$XB<14:0>$ is the bit-reversed address modifier or 'pivot point' which is typically a constant. In the case of an FFT computation, its value is equal to half of the FFT data buffer size.

Note: All bit-reversed EA calculations assume word sized data (LS bit of every EA is always clear). The XB value is scaled accordingly to generate compatible (byte) addresses.

When enabled, bit-reversed addressing will only be executed for register indirect with pre-increment or post-increment addressing and word sized data writes. It will not function for any other Addressing mode or for byte sized data, and normal addresses will be generated instead. When bit-reversed addressing is active, the *W* address pointer will always be added to the address modifier (XB) and the offset associated with the Register Indirect Addressing mode will be ignored. In addition, as word sized data is a requirement, the LS bit of the EA is ignored (and always clear).

Note: Modulo addressing and bit-reversed addressing should not be enabled together. In the event that the user attempts to do this, bit-reversed addressing will assume priority when active for the X WAGU, and X WAGU modulo addressing will be disabled. However, modulo addressing will continue to function in the X RAGU.

If bit-reversed addressing has already been enabled by setting the BREN (XBREV<15>) bit, then a write to the XBREV register should not be immediately followed by an indirect read operation using the *W* register that has been designated as the bit-reversed pointer.

dsPIC30F

FIGURE 4-3: BIT-REVERSED ADDRESS EXAMPLE

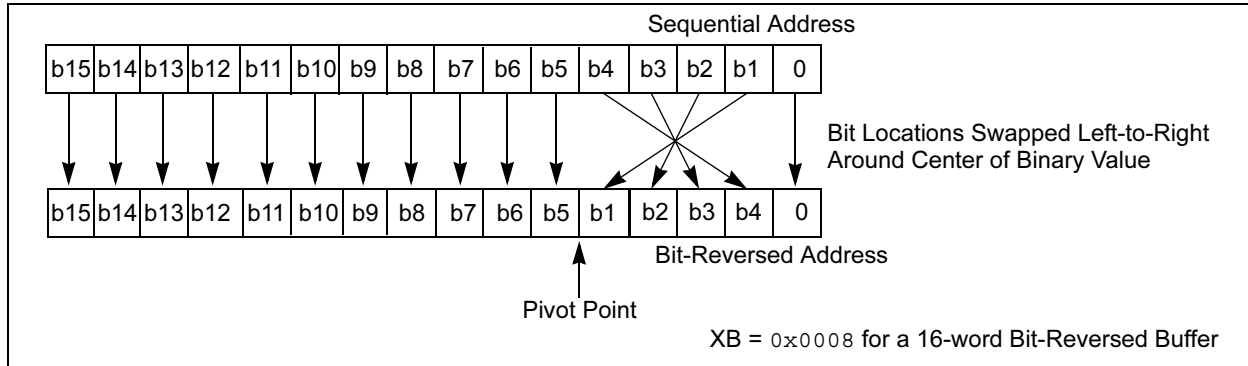


TABLE 4-3: BIT-REVERSED ADDRESS SEQUENCE (16-ENTRY)

| Normal Address | | | | | Bit-Reversed Address | | | | |
|----------------|----|----|----|---------|----------------------|----|----|----|---------|
| A3 | A2 | A1 | A0 | Decimal | A3 | A2 | A1 | A0 | Decimal |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 8 |
| 0 | 0 | 1 | 0 | 2 | 0 | 1 | 0 | 0 | 4 |
| 0 | 0 | 1 | 1 | 3 | 1 | 1 | 0 | 0 | 12 |
| 0 | 1 | 0 | 0 | 4 | 0 | 0 | 1 | 0 | 2 |
| 0 | 1 | 0 | 1 | 5 | 1 | 0 | 1 | 0 | 10 |
| 0 | 1 | 1 | 0 | 6 | 0 | 1 | 1 | 0 | 6 |
| 0 | 1 | 1 | 1 | 7 | 1 | 1 | 1 | 0 | 14 |
| 1 | 0 | 0 | 0 | 8 | 0 | 0 | 0 | 1 | 1 |
| 1 | 0 | 0 | 1 | 9 | 1 | 0 | 0 | 1 | 9 |
| 1 | 0 | 1 | 0 | 10 | 0 | 1 | 0 | 1 | 5 |
| 1 | 0 | 1 | 1 | 11 | 1 | 1 | 0 | 1 | 13 |
| 1 | 1 | 0 | 0 | 12 | 0 | 0 | 1 | 1 | 3 |
| 1 | 1 | 0 | 1 | 13 | 1 | 0 | 1 | 1 | 11 |
| 1 | 1 | 1 | 0 | 14 | 0 | 1 | 1 | 1 | 7 |
| 1 | 1 | 1 | 1 | 15 | 1 | 1 | 1 | 1 | 15 |

TABLE 4-4: BIT-REVERSED ADDRESS MODIFIER VALUES

| Buffer Size (Words) | XB<14:0> Bit-Reversed Address Modifier Value |
|---------------------|--|
| 32768 | 0x4000 |
| 16384 | 0x2000 |
| 8192 | 0x1000 |
| 4096 | 0x0800 |
| 2048 | 0x0400 |
| 1024 | 0x0200 |
| 512 | 0x0100 |
| 256 | 0x0080 |
| 128 | 0x0040 |
| 64 | 0x0020 |
| 32 | 0x0010 |
| 16 | 0x0008 |
| 8 | 0x0004 |
| 4 | 0x0002 |
| 2 | 0x0001 |

5.0 INTERRUPTS

Note: This data sheet summarizes features of this group of dsPIC30F devices and is not intended to be a complete reference source. For more information on the CPU, peripherals, register descriptions and general device functionality, refer to the *dsPIC30F Family Reference Manual* (DS70046). For more information on the device instruction set and programming, refer to the *dsPIC30F Programmer's Reference Manual* (DS70030).

The dsPIC30F Sensor and General Purpose Family has up to 41 interrupt sources and 4 processor exceptions (traps) which must be arbitrated based on a priority scheme.

The CPU is responsible for reading the Interrupt Vector Table (IVT) and transferring the address contained in the interrupt vector to the program counter. The interrupt vector is transferred from the program data bus into the program counter via a 24-bit wide multiplexer on the input of the program counter.

The Interrupt Vector Table (IVT) and Alternate Interrupt Vector Table (AIVT) are placed near the beginning of program memory (0x000004). The IVT and AIVT are shown in Table 5-2.

The interrupt controller is responsible for pre-processing the interrupts and processor exceptions prior to them being presented to the processor core. The peripheral interrupts and traps are enabled, prioritized and controlled using centralized Special Function Registers:

- IFS0<15:0>, IFS1<15:0>, IFS2<15:0>
All interrupt request flags are maintained in these three registers. The flags are set by their respective peripherals or external signals, and they are cleared via software.
- IEC0<15:0>, IEC1<15:0>, IEC2<15:0>
All interrupt enable control bits are maintained in these three registers. These control bits are used to individually enable interrupts from the peripherals or external signals.
- IPC0<15:0>... IPC10<7:0>
The user assignable priority level associated with each of these 41 interrupts is held centrally in these twelve registers.
- IPL<3:0>
The current CPU priority level is explicitly stored in the IPL bits. IPL<3> is present in the CORCON register, whereas IPL<2:0> are present in the STATUS register (SR) in the processor core.

- INTCON1<15:0>, INTCON2<15:0>
Global interrupt control functions are derived from these two registers. INTCON1 contains the control and status flags for the processor exceptions. The INTCON2 register controls the external interrupt request signal behavior and the use of the alternate vector table.

Note: Interrupt flag bits get set when an interrupt condition occurs, regardless of the state of its corresponding enable bit. User software should ensure the appropriate interrupt flag bits are clear prior to enabling an interrupt.

All interrupt sources can be user assigned to one of 7 priority levels, 1 through 7, via the IPCx registers. Each interrupt source is associated with an interrupt vector, as shown in Table 5-2. Levels 7 and 1 represent the highest and lowest maskable priorities, respectively.

Note: Assigning a priority level of '0' to an interrupt source is equivalent to disabling that interrupt.

If the NSTDIS bit (INTCON1<15>) is set, nesting of interrupts is prevented. Thus, if an interrupt is currently being serviced, processing of a new interrupt is prevented even if the new interrupt is of higher priority than the one currently being serviced.

Note: The IPL bits become read only whenever the NSTDIS bit has been set to '1'.

Certain interrupts have specialized control bits for features like edge or level triggered interrupts, interrupt-on-change, etc. Control of these features remains within the peripheral module which generates the interrupt.

The DISI instruction can be used to disable the processing of interrupts of priorities 6 and lower for a certain number of instructions, during which the DISI bit (INTCON2<14>) remains set.

When an interrupt is serviced, the PC is loaded with the address stored in the vector location in program memory that corresponds to the interrupt. There are 63 different vectors within the IVT (refer to Table 5-2). These vectors are contained in locations 0x000004 through 0x0000FE of program memory (refer to Table 5-2). These locations contain 24-bit addresses and in order to preserve robustness, an address error trap will take place should the PC attempt to fetch any of these words during normal execution. This prevents execution of random data as a result of accidentally decrementing a PC into vector space, accidentally mapping a data space address into vector space, or the PC rolling over to 0x000000 after reaching the end of implemented program memory space. Execution of a GOTO instruction to this vector space will also generate an address error trap.

dsPIC30F

5.1 Interrupt Priority

The user assignable interrupt priority (IP<2:0>) bits for each individual interrupt source are located in the LS 3 bits of each nibble within the IPCx register(s). Bit 3 of each nibble is not used and is read as a '0'. These bits define the priority level assigned to a particular interrupt by the user.

Note: The user selectable priority levels start at 0 as the lowest priority and level 7 as the highest priority.

Since more than one interrupt request source may be assigned to a specific user specified priority level, a means is provided to assign priority within a given level. This method is called "Natural Order Priority".

Table 5-1 lists the interrupt numbers and interrupt sources for the dsPIC device and their associated vector numbers.

Note 1: The natural order priority scheme has 0 as the highest priority and 53 as the lowest priority.
2: The natural order priority number is the same as the INT number.

The ability for the user to assign every interrupt to one of seven priority levels implies that the user can assign a very high overall priority level to an interrupt with a low natural order priority. For example, the PLVD (Low Voltage Detect) can be given a priority of 7. The INT0 (External Interrupt 0) may be assigned to priority level 1, thus giving it a very low effective priority.

TABLE 5-1: NATURAL ORDER PRIORITY

| INT Number | Vector Number | Interrupt Source |
|--------------------------------|---------------|--|
| Highest Natural Order Priority | | |
| 0 | 8 | INT0 - External Interrupt 0 |
| 1 | 9 | IC1 - Input Capture 1 |
| 2 | 10 | OC1 - Output Compare 1 |
| 3 | 11 | T1 - Timer 1 |
| 4 | 12 | IC2 - Input Capture 2 |
| 5 | 13 | OC2 - Output Compare 2 |
| 6 | 14 | T2 - Timer 2 |
| 7 | 15 | T3 - Timer 3 |
| 8 | 16 | SPI1 |
| 9 | 17 | U1RX - UART1 Receiver |
| 10 | 18 | U1TX - UART1 Transmitter |
| 11 | 19 | ADC - ADC Convert Done |
| 12 | 20 | NVM - NVM Write Complete |
| 13 | 21 | SI2C - I ² C Slave Interrupt |
| 14 | 22 | MI2C - I ² C Master Interrupt |
| 15 | 23 | Input Change Interrupt |
| 16 | 24 | INT1 - External Interrupt 1 |
| 17 | 25 | IC7 - Input Capture 7 |
| 18 | 26 | IC8 - Input Capture 8 |
| 19 | 27 | OC3 - Output Compare 3 |
| 20 | 28 | OC4 - Output Compare 4 |
| 21 | 29 | T4 - Timer 4 |
| 22 | 30 | T5 - Timer 5 |
| 23 | 31 | INT2 - External Interrupt 2 |
| 24 | 32 | U2RX - UART2 Receiver |
| 25 | 33 | U2TX - UART2 Transmitter |
| 26 | 34 | SPI2 |
| 27 | 35 | C1 - Combined IRQ for CAN1 |
| 28 | 36 | IC3 - Input Capture 3 |
| 29 | 37 | IC4 - Input Capture 4 |
| 30 | 38 | IC5 - Input Capture 5 |
| 31 | 39 | IC6 - Input Capture 6 |
| 32 | 40 | OC5 - Output Compare 5 |
| 33 | 41 | OC6 - Output Compare 6 |
| 34 | 42 | OC7 - Output Compare 7 |
| 35 | 43 | OC8 - Output Compare 8 |
| 36 | 44 | INT3 - External Interrupt 3 |
| 37 | 45 | INT4 - External Interrupt 4 |
| 38 | 46 | C2 - Combined IRQ for CAN2 |
| 39-40 | 47-48 | Reserved |
| 41 | 49 | DCI - Codec Transfer Done |
| 42 | 50 | LVD - Low Voltage Detect |
| 43-53 | 51-61 | Reserved |
| Lowest Natural Order Priority | | |

5.2 Reset Sequence

A Reset is not a true exception, because the interrupt controller is not involved in the Reset process. The processor initializes its registers in response to a Reset which forces the PC to zero. The processor then begins program execution at location `0x000000`. A `GOTO` instruction is stored in the first program memory location immediately followed by the address target for the `GOTO` instruction. The processor executes the `GOTO` to the specified address and then begins operation at the specified target (start) address.

5.2.1 RESET SOURCES

In addition to external Reset and Power-on Reset (POR), there are 6 sources of error conditions which 'trap' to the Reset vector.

- **Watchdog Time-out:**
The watchdog has timed out, indicating that the processor is no longer executing the correct flow of code.
- **Uninitialized W Register Trap:**
An attempt to use an uninitialized W register as an address pointer will cause a Reset.
- **Illegal Instruction Trap:**
Attempted execution of any unused opcodes will result in an illegal instruction trap. Note that a fetch of an illegal instruction does not result in an illegal instruction trap if that instruction is flushed prior to execution due to a flow change.
- **Brown-out Reset (BOR):**
A momentary dip in the power supply to the device has been detected which may result in malfunction.
- **Trap Lockout:**
Occurrence of multiple trap conditions simultaneously will cause a Reset.

5.3 Traps

Traps can be considered as non-maskable, non-stable interrupts, which adhere to a predefined priority, as shown in Table 5-2. They are intended to provide the user a means to correct erroneous operation during debug and when operating within the application.

Note: If the user does not intend to take corrective action in the event of a trap error condition, these vectors must be loaded with the address of a default handler that simply contains the `RESET` instruction. If, on the other hand, one of the vectors containing an invalid address is called, an address error trap is generated.

Note that many of these trap conditions can only be detected when they occur. Consequently, the questionable instruction is allowed to complete prior to trap exception processing. If the user chooses to recover from the error, the result of the erroneous action that caused the trap may have to be corrected.

There are 8 fixed priority levels for traps: level 8 through level 15, which implies that the IPL3 is always set during processing of a trap.

If the user is not currently executing a trap, and he sets the IPL<3:0> bits to a value of '0111' (level 7), then all interrupts are disabled but traps can still be processed.

5.3.1 TRAP SOURCES

The following traps are provided with increasing priority. However, since all traps can be nested, priority has little effect.

- **Math Error Trap:**
The math error trap executes under the following four circumstances:
 1. Should an attempt be made to divide by zero, the divide operation will be aborted on a cycle boundary and the trap taken.
 2. If enabled, a math error trap will be taken when an arithmetic operation on either accumulator A or B causes an overflow from bit 31 and the accumulator guard bits are not utilized.
 3. If enabled, a math error trap will be taken when an arithmetic operation on either accumulator A or B causes a catastrophic overflow from bit 39 and all saturation is disabled.
 4. If the shift amount specified in a shift instruction is greater than the maximum allowed shift amount, a trap will occur.

- Address Error Trap:

This trap is initiated when any of the following circumstances occurs:

1. A misaligned data word access is attempted.
2. A data fetch from an unimplemented data memory location is attempted.
3. A data fetch from an unimplemented program memory location is attempted.
4. An instruction fetch from vector space is attempted.

Note: In the MAC class of instructions, wherein the data space is split into X and Y data space, unimplemented X space includes all of Y space, and unimplemented Y space includes all of X space.

5. Execution of a "BRA #literal" instruction or a "GOTO #literal" instruction, where *literal* is an unimplemented program memory address.
6. Executing instructions after modifying the PC to point to unimplemented program memory addresses. The PC may be modified by loading a value into the stack and executing a RETURN instruction.

- Stack Error Trap:

This trap is initiated under the following conditions:

1. The stack pointer is loaded with a value which is greater than the (user programmable) limit value written into the SPLIM register (stack overflow).
2. The stack pointer is loaded with a value which is less than 0x0800 (simple stack underflow).

- Oscillator Fail Trap:

This trap is initiated if the external oscillator fails and operation becomes reliant on an internal RC backup.

5.3.2 HARD AND SOFT TRAPS

It is possible that multiple traps can become active within the same cycle (e.g., a misaligned word stack write to an overflowed address). In such a case, the fixed priority shown in Figure 5-2 is implemented, which may require the user to check if other traps are pending in order to completely correct the fault.

'Soft' traps include exceptions of priority level 8 through level 11, inclusive. The arithmetic error trap (level 11) falls into this category of traps. Soft traps can be treated like non-maskable sources of interrupt that adhere to the priority assigned by their position in the IVT. Soft traps are processed like interrupts and require 2 cycles to be sampled and Acknowledged prior to exception processing. Therefore, additional instructions may be executed before a soft trap is Acknowledged.

'Hard' traps include exceptions of priority level 12 through level 15, inclusive. The address error (level 12), stack error (level 13) and oscillator error (level 14) traps fall into this category.

Like soft traps, hard traps can also be viewed as non-maskable sources of interrupt. The difference between hard traps and soft traps is that hard traps force the CPU to stop code execution after the instruction causing the trap has completed. Normal program execution flow will not resume until after the trap has been Acknowledged and processed.

If a higher priority trap occurs while any lower priority trap is in progress, processing of the lower priority trap will be suspended and the higher priority trap will be Acknowledged and processed. The lower priority trap will remain pending until processing of the higher priority trap completes.

Each hard trap that occurs must be Acknowledged before code execution of any type may continue. If a lower priority hard trap occurs while a higher priority trap is pending, Acknowledged, or is being processed, a hard trap conflict will occur. The conflict occurs because the lower priority trap cannot be Acknowledged until processing for the higher priority trap completes.

The device is automatically reset in a hard trap conflict condition. The TRAPR status bit (RCON<15>) is set when the Reset occurs so that the condition may be detected in software.

In the case of a math error trap or oscillator failure trap, the condition that causes the trap to occur must be removed before the respective trap flag bit in the INTCON1 register may be cleared.

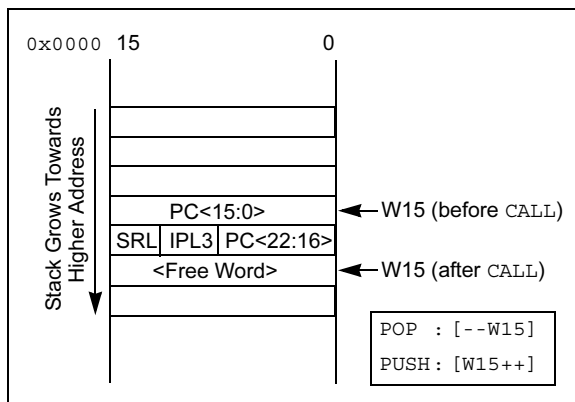
5.4 Interrupt Sequence

All interrupt event flags are sampled in the beginning of each instruction cycle by the IFSx registers. A pending interrupt request (IRQ) is indicated by the flag bit being equal to a '1' in an IFSx register. The IRQ will cause an interrupt to occur if the corresponding bit in the Interrupt Enable (IECx) register is set. For the remainder of the instruction cycle, the priorities of all pending interrupt requests are evaluated.

If there is a pending IRQ with a priority level greater than the current processor priority level in the IPL bits, the processor will be interrupted.

The processor then stacks the current program counter and the low byte of the processor STATUS register (SRL), as shown in Figure 5-1. The low byte of the STATUS register contains the processor priority level at the time prior to the beginning of the interrupt cycle. The processor then loads the priority level for this interrupt into the STATUS register. This action will disable all lower priority interrupts until the completion of the Interrupt Service Routine.

FIGURE 5-1: INTERRUPT STACK FRAME

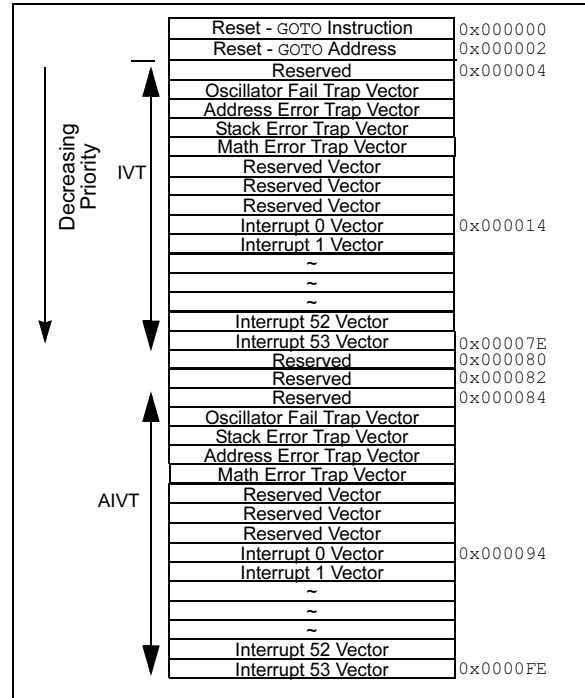


Note 1: The user can always lower the priority level by writing a new value into SR. The Interrupt Service Routine must clear the interrupt flag bits in the IFSx register before lowering the processor interrupt priority, in order to avoid recursive interrupts.

2: The IPL3 bit (CORCON<3>) is always clear when interrupts are being processed. It is set only during execution of traps.

The RETFIE (return from interrupt) instruction will unstack the program counter and STATUS registers to return the processor to its state prior to the interrupt sequence.

FIGURE 5-2: EXCEPTION VECTORS



5.5 Alternate Vector Table

In program memory, the Interrupt Vector Table (IVT) is followed by the Alternate Interrupt Vector Table (AIVT), as shown in Table 5-2. Access to the alternate vector table is provided by the ALTIVT bit in the INTCON2 register. If the ALTIVT bit is set, all interrupt and exception processes will use the alternate vectors instead of the default vectors. The alternate vectors are organized in the same manner as the default vectors. The AIVT supports emulation and debugging efforts by providing a means to switch between an application and a support environment without requiring the interrupt vectors to be reprogrammed. This feature also enables switching between applications for evaluation of different software algorithms at run time.

If the AIVT is not required, the program memory allocated to the AIVT may be used for other purposes. AIVT is not a protected section and may be freely programmed by the user.

5.6 Fast Context Saving

A context saving option is available using shadow registers. Shadow registers are provided for the DC, N, OV, Z and C bits in SR, and the registers W0 through W3. The shadows are only one level deep. The shadow registers are accessible using the `PUSH.S` and `POP.S` instructions only.

When the processor vectors to an interrupt, the `PUSH.S` instruction can be used to store the current value of the aforementioned registers into their respective shadow registers.

If an ISR of a certain priority uses the `PUSH.S` and `POP.S` instructions for fast context saving, then a higher priority ISR should not include the same instructions. Users must save the key registers in software during a lower priority interrupt if the higher priority ISR uses fast context saving.

5.7 External Interrupt Requests

The interrupt controller supports up to five external interrupt request signals, INT0-INT4. These inputs are edge sensitive; they require a low-to-high or a high-to-low transition to generate an interrupt request. The INTCON2 register has five bits, INT0EP-INT4EP, that select the polarity of the edge detection circuitry.

5.8 Wake-up from Sleep and Idle

The interrupt controller may be used to wake-up the processor from either Sleep or Idle modes, if Sleep or Idle mode is active when the interrupt is generated.

If an enabled interrupt request of sufficient priority is received by the interrupt controller, then the standard interrupt request is presented to the processor. At the same time, the processor will wake-up from Sleep or Idle and begin execution of the Interrupt Service Routine (ISR) needed to process the interrupt request.

TABLE 5-2: INTERRUPT CONTROLLER REGISTER MAP

| SFR Name | ADR | 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 | Reset State |
|----------|------|--------|-------------|--------|--------|--------|-------------|--------|--------|--------|-------|-------------|---------|---------|--------|-------------|--------|---------------------|
| INTCON1 | 0080 | NSTDIS | — | — | — | OVATE | — | OVBTE | COVTE | — | — | — | MATHERR | ADDRERR | STKERR | OSCFAIL | — | 0000 0000 0000 0000 |
| INTCON2 | 0082 | ALTIVT | — | — | — | — | — | — | — | — | — | — | INT4EP | INT3EP | INT2EP | INT1EP | INT0EP | 0000 0000 0000 0000 |
| IFS0 | 0084 | CNIF | M12CIF | S12CIF | IC4IF | ADIF | U1TXIF | U1RXIF | SP11IF | T3IF | T2IF | OC2IF | IC2IF | T1IF | OC1IF | IC1IF | INT0IF | 0000 0000 0000 0000 |
| IFS1 | 0086 | IC6IF | IC5IF | IC4IF | IC3IF | C1IF | SP12IF | U2TXIF | U2RXIF | INT2IF | T5IF | T4IF | OC4IF | OC3IF | IC8IF | IC7IF | INT1IF | 0000 0000 0000 0000 |
| IFS2 | 0088 | — | — | — | — | — | LVDIF | DCIF | — | — | C2IF | INT4IF | INT3IF | OC8IF | OC7IF | OC6IF | OC5IF | 0000 0000 0000 0000 |
| IEC0 | 008C | CNIE | M12CIE | S12CIE | NVMIE | ADIE | U1TXIE | U1RXIE | SP11IE | T3IE | T2IE | OC2IE | IC2IE | T1IE | OC1IE | IC1IE | INT0IE | 0000 0000 0000 0000 |
| IEC1 | 008E | IC6IE | IC5IE | IC4IE | IC3IE | C1IE | SP12IE | U2TXIE | U2RXIE | INT2IE | T5IE | T4IE | OC4IE | OC3IE | IC8IE | IC7IE | INT1IE | 0000 0000 0000 0000 |
| IEC2 | 0090 | — | — | — | — | — | LVDIE | DCIE | — | — | C2IE | INT4IE | INT3IE | OC8IE | OC7IE | OC6IE | OC5IE | 0000 0000 0000 0000 |
| IPC0 | 0094 | — | T1IP<2:0> | — | — | — | OC1IP<2:0> | — | — | — | — | IC1IP<2:0> | — | — | — | INT0IP<2:0> | — | 0100 0100 0100 0100 |
| IPC1 | 0096 | — | T31P<2:0> | — | — | — | T2IP<2:0> | — | — | — | — | OC2IP<2:0> | — | — | — | IC2IP<2:0> | — | 0100 0100 0100 0100 |
| IPC2 | 0098 | — | ADIP<2:0> | — | — | — | U1TXIP<2:0> | — | — | — | — | U1RXIP<2:0> | — | — | — | SP11IP<2:0> | — | 0100 0100 0100 0100 |
| IPC3 | 009A | — | CNIP<2:0> | — | — | — | M12CIP<2:0> | — | — | — | — | S12CIP<2:0> | — | — | — | NVMIP<2:0> | — | 0100 0100 0100 0100 |
| IPC4 | 009C | — | OC3IP<2:0> | — | — | — | IC8IP<2:0> | — | — | — | — | IC7IP<2:0> | — | — | — | INT1IP<2:0> | — | 0100 0100 0100 0100 |
| IPC5 | 009E | — | INT2IP<2:0> | — | — | — | T5IP<2:0> | — | — | — | — | T4IP<2:0> | — | — | — | OC4IP<2:0> | — | 0100 0100 0100 0100 |
| IPC6 | 00A0 | — | C1IP<2:0> | — | — | — | SP12IP<2:0> | — | — | — | — | U2TXIP<2:0> | — | — | — | U2RXIP<2:0> | — | 0100 0100 0100 0100 |
| IPC7 | 00A2 | — | IC6IP<2:0> | — | — | — | IC5IP<2:0> | — | — | — | — | IC4IP<2:0> | — | — | — | IC3IP<2:0> | — | 0100 0100 0100 0100 |
| IPC8 | 00A4 | — | OC8IP<2:0> | — | — | — | OC7IP<2:0> | — | — | — | — | OC6IP<2:0> | — | — | — | OC5IP<2:0> | — | 0100 0100 0100 0100 |
| IPC9 | 00A6 | — | — | — | — | — | C2IP<2:0> | — | — | — | — | INT4IP<2:0> | — | — | — | INT3IP<2:0> | — | 0000 0100 0100 0100 |
| IPC10 | 00A8 | — | — | — | — | — | LVDIP<2:0> | — | — | — | — | DCIIP<2:0> | — | — | — | — | — | 0000 0100 0100 0000 |

Legend: u = uninitialized bit

Note: Refer to *dsPIC30F Family Reference Manual (DS70046)* for descriptions of register bit fields.

dsPIC30F

NOTES:

6.0 FLASH PROGRAM MEMORY

Note: This data sheet summarizes features of this group of dsPIC30F devices and is not intended to be a complete reference source. For more information on the CPU, peripherals, register descriptions and general device functionality, refer to the *dsPIC30F Family Reference Manual* (DS70046). For more information on the device instruction set and programming, refer to the *dsPIC30F Programmer's Reference Manual* (DS70030).

The dsPIC30F family of devices contains internal program Flash memory for executing user code. There are two methods by which the user can program this memory:

1. Run-Time Self-Programming (RTSP)
2. In-Circuit Serial Programming™ (ICSP™)

6.1 In-Circuit Serial Programming (ICSP)

dsPIC30F devices can be serially programmed while in the end application circuit. This is simply done with two lines for Programming Clock and Programming Data (which are named PGC and PGD respectively), and three other lines for Power (VDD), Ground (VSS) and Master Clear (MCLR). This allows customers to manufacture boards with unprogrammed devices, and then program the microcontroller just before shipping the product. This also allows the most recent firmware or a custom firmware to be programmed.

6.2 Run-Time Self-Programming (RTSP)

RTSP is accomplished using TBLRD (table read) and TBLWT (table write) instructions. With RTSP, the user may erase and program 32 instructions (96 bytes) at a time.

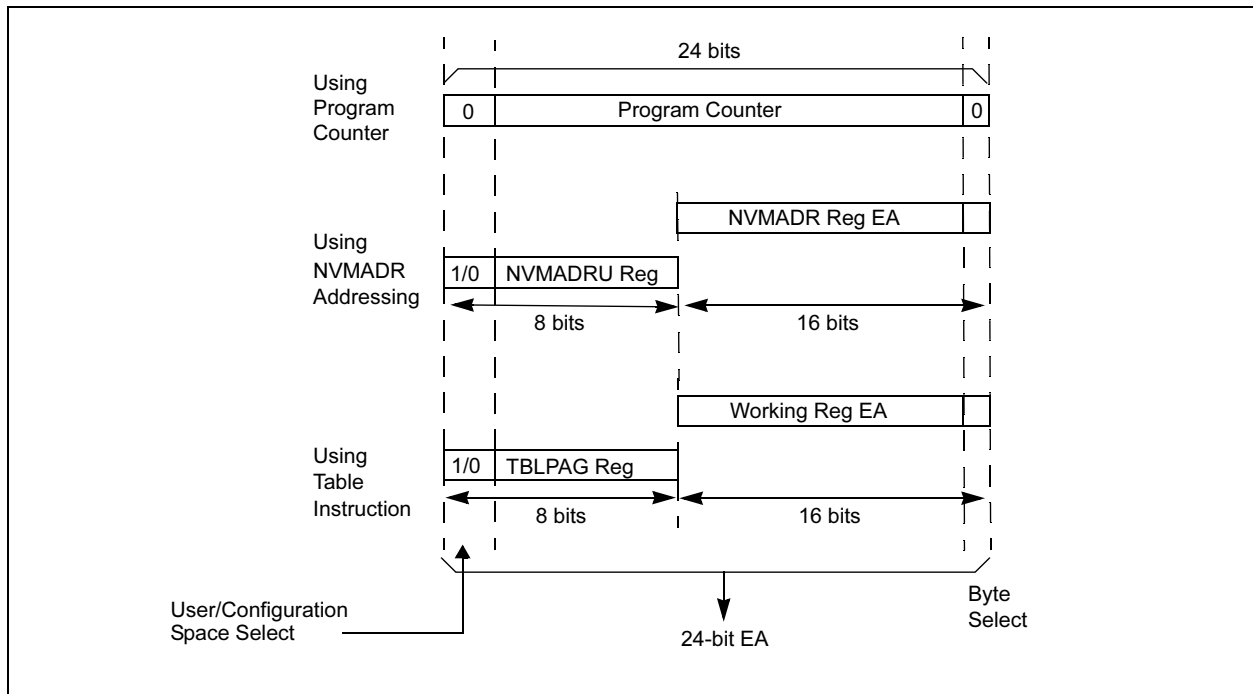
6.3 Table Instruction Operation Summary

The TBLRDL and the TBLWTL instructions are used to read or write to bits<15:0> of program memory. TBLRDL and TBLWTL can access program memory in Word or Byte mode.

The TBLRDH and TBLWTH instructions are used to read or write to bits<23:16> of program memory. TBLRDH and TBLWTH can access program memory in Word or Byte mode.

A 24-bit program memory address is formed using bits<7:0> of the TBLPAG register and the effective address (EA) from a W register specified in the table instruction, as shown in Figure 6-1.

FIGURE 6-1: ADDRESSING FOR TABLE AND NVM REGISTERS



6.4 RTSP Operation

The dsPIC30F Flash program memory is organized into rows and panels. Each row consists of 32 instructions or 96 bytes. Each panel consists of 128 rows or 4K x 24 instructions. RTSP allows the user to erase and program one row (32 instructions) at a time. RTSP may be used to program multiple program memory panels, but the table pointer must be changed at each panel boundary.

Each panel of program memory contains write latches that hold 32 instructions of programming data. Prior to the actual programming operation, the write data must be loaded into the panel write latches. The data to be programmed into the panel is loaded in sequential order into the write latches: instruction 0, instruction 1, etc. The instruction words loaded must always be from a group of 32 boundary.

The basic sequence for RTSP programming is to set up a table pointer, then do a series of `TBLWT` instructions to load the write latches. Programming is performed by setting the special bits in the `NVMCON` register. Four `TBLWTL` and four `TBLWTH` instructions are required to load the four instructions. To fully program a row of program memory, eight cycles of four `TBLWTL` and four `TBLWTH` are required. If multiple panel programming is required, the table pointer needs to be changed and the next set of multiple write latches written.

All of the table write operations are single word writes (2 instruction cycles) because only the table latches are written. A total of 32 programming passes, each writing 4 instruction words, are required per row.

The Flash program memory is readable, writable, and erasable during normal operation over the entire `VDD` range.

6.5 Control Registers

The four SFRs used to read and write the program Flash memory are:

- `NVMCON`
- `NVMADR`
- `NVMADRU`
- `NVMKEY`

6.5.1 NVMCON REGISTER

The `NVMCON` register controls which blocks are to be erased, which memory type is to be programmed and start of the programming cycle.

6.5.2 NVMADR REGISTER

The `NVMADR` register is used to hold the lower two bytes of the effective address. The `NVMADR` register captures the `EA<15:0>` of the last table instruction that has been executed and selects the row to write.

6.5.3 NVMADRU REGISTER

The `NVMADRU` register is used to hold the upper byte of the effective address. The `NVMADRU` register captures the `EA<23:16>` of the last table instruction that has been executed.

6.5.4 NVMKEY REGISTER

`NVMKEY` is a write only register that is used for write protection. To start a programming or an erase sequence, the user must consecutively write `0x55` and `0xAA` to the `NVMKEY` register. Refer to Section 6.6 for further details.

6.6 Programming Operations

A complete programming sequence is necessary for programming or erasing the internal Flash in RTSP mode. A programming operation is nominally 2 msec in duration and the processor stalls (waits) until the operation is finished. Setting the WR bit (NVMCON<15>) starts the operation, and the WR bit is automatically cleared when the operation is finished.

6.6.1 PROGRAMMING ALGORITHM FOR PROGRAM FLASH

The user can erase one row of program Flash memory at a time. The user can program one block (4 instruction words) of Flash memory at a time. The general process is:

1. Read one row of program Flash (32 instruction words) and store into data RAM as a data "image".
2. Update the data image with the desired new data.
3. Erase program Flash row.
 - a) Setup NVMCON register for multi-word, program Flash, erase, and set WREN bit.
 - b) Write address of row to be erased into NVMADRU/NVMADR.
 - c) Write '55' to NVMKEY.
 - d) Write 'AA' to NVMKEY.
 - e) Set the WR bit. This will begin erase cycle.
 - f) CPU will stall for the duration of the erase cycle.
 - g) The WR bit is cleared when erase cycle ends.

4. Write 32 instruction words of data from data RAM into the program Flash write latches.
5. Program 32 instruction words into program Flash.
 - a) Setup NVMCON register for multi-word, program Flash, program, and set WREN bit.
 - b) Write '55' to NVMKEY.
 - c) Write 'AA' to NVMKEY.
 - d) Set the WR bit. This will begin program cycle.
 - e) CPU will stall for duration of the program cycle.
 - f) The WR bit is cleared by the hardware when program cycle ends.
6. Repeat steps 1 through 5 as needed to program desired amount of program Flash memory.

6.6.2 ERASING A ROW OF PROGRAM MEMORY

Example 6-1 shows a code sequence that can be used to erase a row (32 instructions) of program memory.

EXAMPLE 6-1: ERASING A ROW OF PROGRAM MEMORY

```

; Setup NVMCON for erase operation, multi word write
; program memory selected, and writes enabled
    MOV    #0x4041,W0                ;
    MOV    W0,NVMCON                 ; Init NVMCON SFR
; Init pointer to row to be ERASED
    MOV    #tblpage(PROG_ADDR),W0    ;
    MOV    W0,NVMADRU                ; Initialize PM Page Boundary SFR
    MOV    #tbloffset(PROG_ADDR),W0 ; Intialize in-page EA[15:0] pointer
    MOV    W0, NVMADR                ; Initialize NVMADR SFR
    DISI   #5                        ; Block all interrupts with priority <7 for
                                        ; next 5 instructions

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

```

dsPIC30F

6.6.3 LOADING WRITE LATCHES

Example 6-2 shows a sequence of instructions that can be used to load the 96 bytes of write latches. Four TBLWTL and four TBLWTH instructions are needed to load the write latches selected by the table pointer.

EXAMPLE 6-2: LOADING WRITE LATCHES

```
; 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
.
.
; 31st_program_word
MOV    #LOW_WORD_3,W2           ;
MOV    #HIGH_BYTE_3,W3         ;
TBLWTL W2,[W0]                 ; Write PM low word into program latch
TBLWTH W3,[W0++]              ; Write PM high byte into program latch
```

Note: In Example 6-2, the contents of the upper byte of W3 has no effect.

6.6.4 INITIATING THE PROGRAMMING SEQUENCE

For protection, 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.

EXAMPLE 6-3: INITIATING A PROGRAMMING SEQUENCE

```
DISI   #5                        ; Block all interrupts with priority <7 for
                                        ; next 5 instructions
MOV    #0x55,W0                  ;
MOV    W0,NVMKEY                 ; Write the 0x55 key
MOV    #0xAA,W1                  ;
MOV    W1,NVMKEY                 ; Write the 0xAA key
BSET   NVMCON,#WR                ; Start the erase sequence
NOP                                         ; Insert two NOPs after the erase
NOP                                         ; command is asserted
```

TABLE 6-1: NVM REGISTER MAP

| File Name | Addr. | Bit 15 | Bit 14 | Bit 13 | Bit 12 | Bit 11 | Bit 10 | Bit 9 | Bit 8 | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | All RESETS |
|-----------|-------|--------------|--------|--------|--------|--------|--------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|---------------------|
| NVMCON | 0760 | WR | WREN | WRERR | — | — | — | — | TWRI | — | — | — | — | — | — | — | — | 0000 0000 0000 0000 |
| NVMADR | 0762 | NVMADR<15:0> | | | | | | | | | | | | | | | | |
| NVMADRU | 0764 | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | uuuu uuuu uuuu uuuu |
| NVMKEY | 0766 | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | 0000 0000 uuuu uuuu |
| | | | | | | | | | | | | | | | | | | 0000 0000 0000 0000 |

Legend: u = uninitialized bit

Note: Refer to *dsPIC30F Family Reference Manual (DS70046)* for descriptions of register bit fields.

dsPIC30F

NOTES:

7.0 DATA EEPROM MEMORY

Note: This data sheet summarizes features of this group of dsPIC30F devices and is not intended to be a complete reference source. For more information on the CPU, peripherals, register descriptions and general device functionality, refer to the *dsPIC30F Family Reference Manual* (DS70046). For more information on the device instruction set and programming, refer to the *dsPIC30F Programmer's Reference Manual* (DS70030).

The Data EEPROM Memory is readable and writable during normal operation over the entire VDD range. The data EEPROM memory is directly mapped in the program memory address space.

The four SFRs used to read and write the program Flash memory are used to access data EEPROM memory, as well. As described in Section 6.5, these registers are:

- NVMCON
- NVMADR
- NVMADRU
- NVMKEY

The EEPROM data memory allows read and write of single words and 16-word blocks. When interfacing to data memory, NVMADR in conjunction with the NVMADRU register are used to address the EEPROM location being accessed. TBLRD and TBLWT instructions are used to read and write data EEPROM. The dsPIC30F devices have up to 8 Kbytes (4K words) of data EEPROM with an address range from 0x7FF000 to 0x7FFFFE.

A word write operation should be preceded by an erase of the corresponding memory location(s). The write typically requires 2 ms to complete but the write time will vary with voltage and temperature.

A program or erase operation on the data EEPROM does not stop the instruction flow. The user is responsible for waiting for the appropriate duration of time before initiating another data EEPROM write/erase operation. Attempting to read the data EEPROM while a programming or erase operation is in progress results in unspecified data.

Control bit WR initiates write operations similar to program Flash writes. This bit cannot be cleared, only set, in software. They are cleared in hardware at the completion of the write operation. The inability to clear the WR bit in software prevents the accidental or premature termination of a write operation.

The WREN bit, when set, will allow a write operation. On power-up, the WREN bit is clear. The WRERR bit is set when a write operation is interrupted by a MCLR Reset or a WDT Time-out Reset during normal operation. In these situations, following Reset, the user can check the WRERR bit and rewrite the location. The address register NVMADR remains unchanged.

Note: Interrupt flag bit NVMIF in the IFS0 register is set when write is complete. It must be cleared in software.

7.1 Reading the Data EEPROM

A TBLRD instruction reads a word at the current program word address. This example uses W0 as a pointer to data EEPROM. The result is placed in register W4 as shown in Example 7-1.

EXAMPLE 7-1: DATA EEPROM READ

```
MOV    #LOW_ADDR_WORD,W0 ; Init Pointer
MOV    #HIGH_ADDR_WORD,W1
MOV    W1,TBLPAG
TBLRD  [W0],W4           ; read data EEPROM
```

dsPIC30F

7.2 Erasing Data EEPROM

7.2.1 ERASING A BLOCK OF DATA EEPROM

In order to erase a block of data EEPROM, the NVMADRU and NVMADR registers must initially point to the block of memory to be erased. Configure NVMCON for erasing a block of data EEPROM, and set the ERASE and WREN bits in the NVMCON register. Setting the WR bit initiates the erase as shown in Example 7-2.

EXAMPLE 7-2: DATA EEPROM BLOCK ERASE

```
; Select data EEPROM block, ERASE, WREN bits
MOV    #4045,W0
MOV    W0,NVMCON                ; Initialize NVMCON SFR

; Start erase cycle by setting WR after writing key sequence
DISI   #5                        ; Block all interrupts with priority <7 for
                                ; next 5 instructions
MOV    #0x55,W0                  ;
MOV    W0,NVMKEY                 ; Write the 0x55 key
MOV    #0xAA,W1                  ;
MOV    W1,NVMKEY                 ; Write the 0xAA key
BSET   NVMCON,#WR               ; Initiate erase sequence
NOP
NOP

; Erase cycle will complete in 2mS. CPU is not stalled for the Data Erase Cycle
; User can poll WR bit, use NVMIF or Timer IRQ to determine erasure complete
```

7.2.2 ERASING A WORD OF DATA EEPROM

The TBLPAG and NVMADR registers must point to the block. Select erase a block of data Flash, and set the ERASE and WREN bits in the NVMCON register. Setting the WR bit initiates the erase as shown in Example 7-3.

EXAMPLE 7-3: DATA EEPROM WORD ERASE

```
; Select data EEPROM word, ERASE, WREN bits
MOV    #4044,W0
MOV    W0,NVMCON

; Start erase cycle by setting WR after writing key sequence
DISI   #5                        ; Block all interrupts with priority <7 for
                                ; next 5 instructions
MOV    #0x55,W0                  ;
MOV    W0,NVMKEY                 ; Write the 0x55 key
MOV    #0xAA,W1                  ;
MOV    W1,NVMKEY                 ; Write the 0xAA key
BSET   NVMCON,#WR               ; Initiate erase sequence
NOP
NOP

; Erase cycle will complete in 2mS. CPU is not stalled for the Data Erase Cycle
; User can poll WR bit, use NVMIF or Timer IRQ to determine erasure complete
```


7.3 Writing to the Data EEPROM

To write an EEPROM data location, the following sequence must be followed:

1. Erase data EEPROM word.
 - a) Select word, data EEPROM erase, and set WREN bit in NVMCON register.
 - b) Write address of word to be erased into NVMADR.
 - c) Enable NVM interrupt (optional).
 - d) Write '55' to NVMKEY.
 - e) Write 'AA' to NVMKEY.
 - f) Set the WR bit. This will begin erase cycle.
 - g) Either poll NVMIF bit or wait for NVMIF interrupt.
 - h) The WR bit is cleared when the erase cycle ends.
2. Write data word into data EEPROM write latches.
3. Program 1 data word into data EEPROM.
 - a) Select word, data EEPROM program, and set WREN bit in NVMCON register.
 - b) Enable NVM write done interrupt (optional).
 - c) Write '55' to NVMKEY.
 - d) Write 'AA' to NVMKEY.
 - e) Set the WR bit. This will begin program cycle.
 - f) Either poll NVMIF bit or wait for NVM interrupt.
 - g) The WR bit is cleared when the write cycle ends.

The write will not initiate if the above sequence is not exactly followed (write 0x55 to NVMKEY, write 0xAA to NVMCON, then set WR bit) for each word. It is strongly recommended that interrupts be disabled during this code segment.

Additionally, the WREN bit in NVMCON must be set to enable writes. This mechanism prevents accidental writes to data EEPROM due to unexpected code execution. The WREN bit should be kept clear at all times except when updating the EEPROM. The WREN bit is not cleared by hardware.

After a write sequence has been initiated, clearing the WREN bit will not affect the current write cycle. The WR bit will be inhibited from being set unless the WREN bit is set. The WREN bit must be set on a previous instruction. Both WR and WREN cannot be set with the same instruction.

At the completion of the write cycle, the WR bit is cleared in hardware and the Non-Volatile Memory Write Complete Interrupt Flag bit (NVMIF) is set. The user may either enable this interrupt or poll this bit. NVMIF must be cleared by software.

7.3.1 WRITING A WORD OF DATA EEPROM

Once the user has erased the word to be programmed, then a table write instruction is used to write one write latch, as shown in Example 7-4.

EXAMPLE 7-4: DATA EEPROM WORD WRITE

```

; Point to data memory
MOV      #LOW_ADDR_WORD,W0          ; Init pointer
MOV      #HIGH_ADDR_WORD,W1
MOV      W1,TBLPAG
MOV      #LOW(WORD),W2             ; Get data
TBLWTL   W2,[ W0]                 ; Write data
; The NVMADR captures last table access address
; Select data EEPROM for 1 word op
MOV      #0x4004,W0
MOV      W0,NVMCON

; Operate key to allow write operation
DISI     #5                        ; Block all interrupts with priority <7 for
                                   ; next 5 instructions

MOV      #0x55,W0
MOV      W0,NVMKEY                 ; Write the 0x55 key
MOV      #0xAA,W1
MOV      W1,NVMKEY                 ; Write the 0xAA key
BSET     NVMCON,#WR                ; Initiate program sequence
NOP
NOP

; Write cycle will complete in 2mS. CPU is not stalled for the Data Write Cycle
; User can poll WR bit, use NVMIF or Timer IRQ to determine write complete

```

dsPIC30F

7.3.2 WRITING A BLOCK OF DATA EEPROM

To write a block of data EEPROM, write to all sixteen latches first, then set the NVMCON register and program the block.

EXAMPLE 7-5: DATA EEPROM BLOCK WRITE

```
MOV      #LOW_ADDR_WORD,W0 ; Init pointer
MOV      #HIGH_ADDR_WORD,W1
MOV      W1,TBLPAG
MOV      #data1,W2          ; Get 1st data
TBLWTL  W2,[W0]++          ; write data
MOV      #data2,W2          ; Get 2nd data
TBLWTL  W2,[W0]++          ; write data
MOV      #data3,W2          ; Get 3rd data
TBLWTL  W2,[W0]++          ; write data
MOV      #data4,W2          ; Get 4th data
TBLWTL  W2,[W0]++          ; write data
MOV      #data5,W2          ; Get 5th data
TBLWTL  W2,[W0]++          ; write data
MOV      #data6,W2          ; Get 6th data
TBLWTL  W2,[W0]++          ; write data
MOV      #data7,W2          ; Get 7th data
TBLWTL  W2,[W0]++          ; write data
MOV      #data8,W2          ; Get 8th data
TBLWTL  W2,[W0]++          ; write data
MOV      #data9,W2          ; Get 9th data
TBLWTL  W2,[W0]++          ; write data
MOV      #data10,W2         ; Get 10th data
TBLWTL  W2,[W0]++          ; write data
MOV      #data11,W2         ; Get 11th data
TBLWTL  W2,[W0]++          ; write data
MOV      #data12,W2         ; Get 12th data
TBLWTL  W2,[W0]++          ; write data
MOV      #data13,W2         ; Get 13th data
TBLWTL  W2,[W0]++          ; write data
MOV      #data14,W2         ; Get 14th data
TBLWTL  W2,[W0]++          ; write data
MOV      #data15,W2         ; Get 15th data
TBLWTL  W2,[W0]++          ; write data
MOV      #data16,W2         ; Get 16th data
TBLWTL  W2,[W0]++          ; write data. The NVMADR captures last table access address.
MOV      #0x400A,W0         ; Select data EEPROM for multi word op
MOV      W0,NVMCON          ; Operate Key to allow program operation
DISI    #5                  ; Block all interrupts with priority <7 for
                                ; next 5 instructions

MOV      #0x55,W0
MOV      W0,NVMKEY          ; Write the 0x55 key
MOV      #0xAA,W1
MOV      W1,NVMKEY          ; Write the 0xAA key
BSET    NVMCON,#WR         ; Start write cycle
NOP
NOP
```

7.4 Write Verify

Depending on the application, good programming practice may dictate that the value written to the memory should be verified against the original value. This should be used in applications where excessive writes can stress bits near the specification limit.

7.5 Protection Against Spurious Write

There are conditions when the device may not want to write to the data EEPROM memory. To protect against spurious EEPROM writes, various mechanisms have been built-in. On power-up, the WREN bit is cleared; also, the Power-up Timer prevents EEPROM write.

The write initiate sequence and the WREN bit together help prevent an accidental write during brown-out, power glitch, or software malfunction.

dsPIC30F

NOTES:

8.0 I/O PORTS

Note: This data sheet summarizes features of this group of dsPIC30F devices and is not intended to be a complete reference source. For more information on the CPU, peripherals, register descriptions and general device functionality, refer to the *dsPIC30F Family Reference Manual* (DS70046).

All of the device pins (except VDD, VSS, $\overline{\text{MCLR}}$, and OSC1/CLKI) are shared between the peripherals and the parallel I/O ports.

All I/O input ports feature Schmitt Trigger inputs for improved noise immunity.

8.1 Parallel I/O (PIO) Ports

When a peripheral is enabled and the peripheral is actively driving an associated pin, the use of the pin as a general purpose output pin is disabled. The I/O pin may be read but the output driver for the parallel port bit will be disabled. If a peripheral is enabled but the peripheral is not actively driving a pin, that pin may be driven by a port.

All port pins have three registers directly associated with the operation of the port pin. The Data Direction register (TRISx) determines whether the pin is an input or an output. If the data direction bit is a '1', then the pin is an input. All port pins are defined as inputs after a Reset. Reads from the latch (LATx), read the latch. Writes to the latch, write the latch (LATx). Reads from the port (PORTx), read the port pins and writes to the port pins, write the latch (LATx).

Any bit and its associated data and control registers that are not valid for a particular device will be disabled. That means the corresponding LATx and TRISx registers and the port pin will read as zeros.

When a pin is shared with another peripheral or function that is defined as an input only, it is nevertheless regarded as a dedicated port because there is no other competing source of outputs. An example is the INT4 pin.

The format of the registers for PORTA are shown in Table 8-1.

The TRISA (Data Direction Control) register controls the direction of the RA<7:0> pins, as well as the INTx pins and the VREF pins. The LATA register supplies data to the outputs and is readable/writable. Reading the PORTA register yields the state of the input pins, while writing the PORTA register modifies the contents of the LATA register.

A parallel I/O (PIO) port that shares a pin with a peripheral is, in general, subservient to the peripheral. The peripheral's output buffer data and control signals are provided to a pair of multiplexers. The multiplexers select whether the peripheral or the associated port has ownership of the output data and control signals of the I/O pad cell. Figure 8-2 shows how ports are shared with other peripherals and the associated I/O cell (pad) to which they are connected. Table 8-2 through Table 8-6 show the formats of the registers for the shared ports, PORTB through PORTG.

Note: The actual bits in use vary between devices.

FIGURE 8-1: BLOCK DIAGRAM OF A DEDICATED PORT STRUCTURE

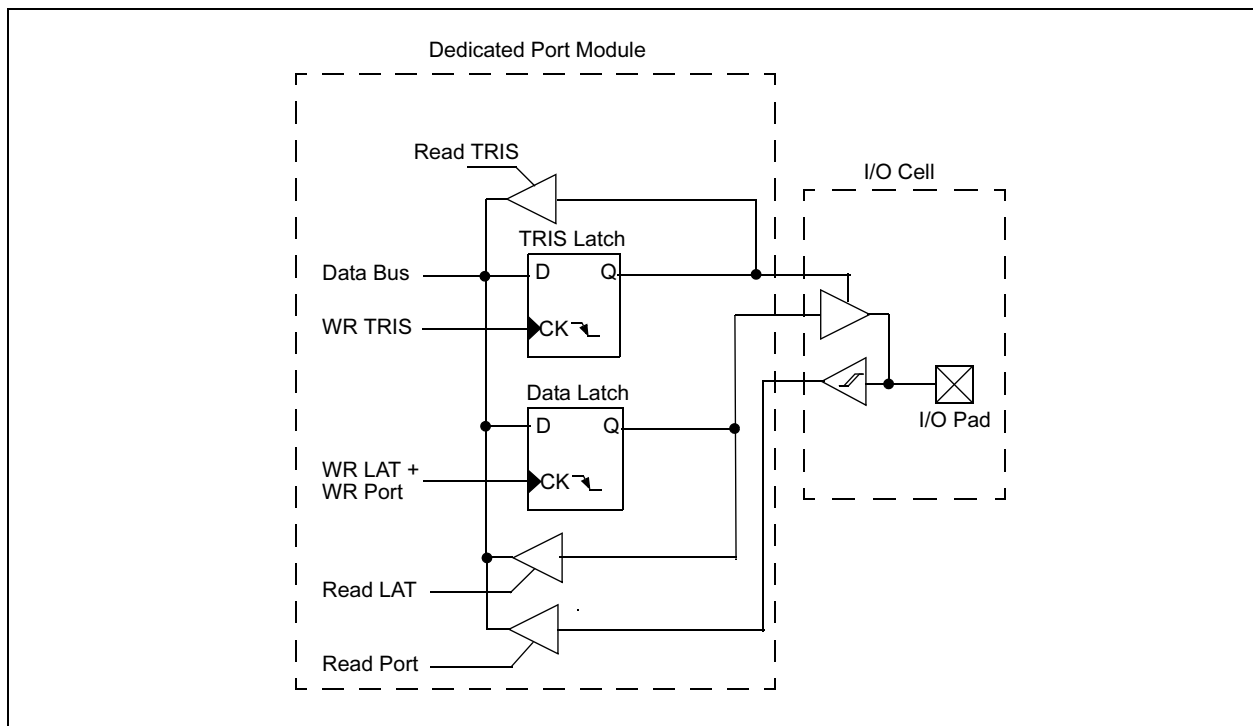
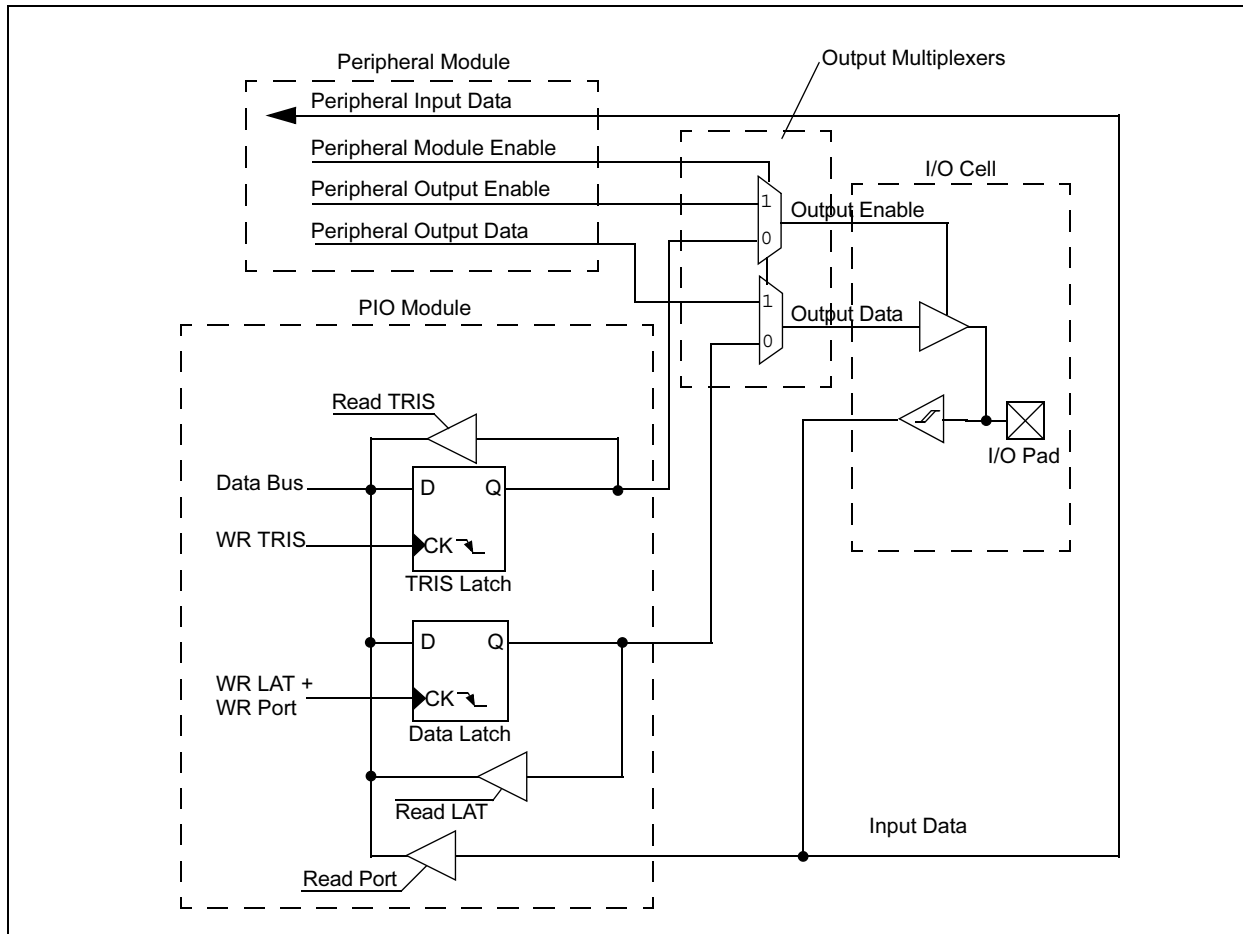


FIGURE 8-2: BLOCK DIAGRAM OF A SHARED PORT STRUCTURE



8.2 Configuring Analog Port Pins

The use of the ADPCFG and TRIS registers control the operation of the A/D port pins. The port pins that are desired as analog inputs must have their corresponding TRIS bit set (input). If the TRIS bit is cleared (output), the digital output level (VOH or VOL) will be converted.

When reading the Port register, all pins configured as analog input channels will read as cleared (a low level).

Pins configured as digital inputs will not convert an analog input. Analog levels on any pin that is defined as a digital input (including the ANx pins) may cause the input buffer to consume current that exceeds the device specifications.

8.2.1 I/O PORT WRITE/READ TIMING

One instruction cycle is required between a port direction change or port write operation and a read operation of the same port. Typically this instruction would be a NOP.

EXAMPLE 8-1: PORT WRITE/READ EXAMPLE

```

MOV    0xFF00, W0 ; Configure PORTB<15:8>
                ; as inputs
MOV    W0, TRISB ; and PORTB<7:0> as outputs
NOP                    ; additional instruction
                ; cycle
btss   PORTB, #13 ; bit test RB13 and skip if
                ; set
    
```

TABLE 8-1: PORTA 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 | Reset State |
|----------|-------|---------|---------|---------|---------|--------|---------|--------|-------|--------|--------|-------|-------|-------|-------|-------|-------|---------------------|
| TRISA | 02C0 | TRISA15 | TRISA14 | TRISA13 | TRISA12 | — | TRISA10 | TRISA9 | — | TRISA7 | TRISA6 | — | — | — | — | — | — | 1111 0110 1100 0000 |
| PORTA | 02C2 | RA15 | RA14 | RA13 | RA12 | — | RA10 | RA9 | — | RA7 | RA6 | — | — | — | — | — | — | 0000 0000 0000 0000 |
| LATA | 02C4 | LATA15 | LATA14 | LATA13 | LATA12 | — | LATA10 | LATA9 | — | LATA7 | LATA6 | — | — | — | — | — | — | 0000 0000 0000 0000 |

Legend: u = uninitialized bit

TABLE 8-2: PORTB 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 | Reset State |
|----------|-------|---------|---------|---------|---------|---------|---------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|---------------------|
| TRISB | 02C6 | TRISB15 | TRISB14 | TRISB13 | TRISB12 | TRISB11 | TRISB10 | TRISB9 | TRISB8 | TRISB7 | TRISB6 | TRISB5 | TRISB4 | TRISB3 | TRISB2 | TRISB1 | TRISB0 | 1111 1111 1111 1111 |
| PORTB | 02C8 | RB15 | RB14 | RB13 | RB12 | RB11 | RB10 | RB9 | RB8 | RB7 | RB6 | RB5 | RB4 | RB3 | RB2 | RB1 | RB0 | 0000 0000 0000 0000 |
| LATB | 02CB | LATB15 | LATB14 | LATB13 | LATB12 | LATB11 | LATB10 | LATB9 | LATB8 | LATB7 | LATB6 | LATB5 | LATB4 | LATB3 | LATB2 | LATB1 | LATB0 | 0000 0000 0000 0000 |

Legend: u = uninitialized bit

TABLE 8-3: PORTC 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 | Reset State |
|----------|-------|---------|---------|---------|--------|--------|--------|-------|-------|-------|-------|--------|--------|--------|--------|-------|-------|---------------------|
| TRISC | 02CC | TRISC15 | TRISC14 | TRISC13 | — | — | — | — | — | — | — | TRISC4 | TRISC3 | TRISC2 | TRISC1 | — | — | 1110 0000 0001 1110 |
| PORTC | 02CE | RC15 | RC14 | RC13 | — | — | — | — | — | — | — | RC4 | RC3 | RC2 | RC1 | — | — | 0000 0000 0000 0000 |
| LATC | 02D0 | LATC15 | LATC14 | LATC13 | — | — | — | — | — | — | — | LATC4 | LATC3 | LATC2 | LATC1 | — | — | 0000 0000 0000 0000 |

Legend: u = uninitialized bit

TABLE 8-4: PORTD 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 | Reset State |
|----------|-------|---------|---------|---------|---------|---------|---------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|---------------------|
| TRISD | 02D2 | TRISD15 | TRISD14 | TRISD13 | TRISD12 | TRISD11 | TRISD10 | TRISD9 | TRISD8 | TRISD7 | TRISD6 | TRISD5 | TRISD4 | TRISD3 | TRISD2 | TRISD1 | TRISD0 | 1111 1111 1111 1111 |
| PORTD | 02D4 | RD15 | RD14 | RD13 | RD12 | RD11 | RD10 | RD9 | RD8 | RD7 | RD6 | RD5 | RD4 | RD3 | RD2 | RD1 | RD0 | 0000 0000 0000 0000 |
| LATD | 02D6 | LATD15 | LATD14 | LATD13 | LATD12 | LATD11 | LATD10 | LATD9 | LATD8 | LATD7 | LATD6 | LATD5 | LATD4 | LATD3 | LATD2 | LATD1 | LATD0 | 0000 0000 0000 0000 |

Legend: u = uninitialized bit

TABLE 8-5: PORTF 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 | Reset State |
|----------|-------|--------|--------|--------|--------|--------|--------|-------|--------|--------|--------|--------|--------|--------|--------|--------|--------|---------------------|
| TRISF | 02DE | — | — | — | — | — | — | — | TRISF8 | TRISF7 | TRISF6 | TRISF5 | TRISF4 | TRISF3 | TRISF2 | TRISF1 | TRISF0 | 0000 0001 1111 1111 |
| PORTF | 02E0 | — | — | — | — | — | — | — | RF8 | RF7 | RF6 | RF5 | RF4 | RF3 | RF2 | RF1 | RF0 | 0000 0000 0000 0000 |
| LATF | 02E2 | — | — | — | — | — | — | — | LATF8 | LATF7 | LATF6 | LATF5 | LATF4 | LATF3 | LATF2 | LATF1 | LATF0 | 0000 0000 0000 0000 |

Legend: u = uninitialized bit

TABLE 8-6: PORTG 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 | Reset State |
|----------|-------|---------|---------|---------|---------|--------|--------|--------|--------|--------|--------|-------|-------|--------|--------|--------|--------|---------------------|
| TRISG | 02E4 | TRISG15 | TRISG14 | TRISG13 | TRISG12 | — | — | TRISG9 | TRISG8 | TRISG7 | TRISG6 | — | — | TRISG3 | TRISG2 | TRISG1 | TRISG0 | 1111 0011 1100 1111 |
| PORTG | 02E6 | RG15 | RG14 | RG13 | RG12 | — | — | RG9 | RG8 | RG7 | RG6 | — | — | RG3 | RG2 | RG1 | RG0 | 0000 0000 0000 0000 |
| LATG | 02E8 | LATG15 | LATG14 | LATG13 | LATG12 | — | — | LATG9 | LATG8 | LATG7 | LATG6 | — | — | LATG3 | LATG2 | LATG1 | LATG0 | 0000 0000 0000 0000 |

Legend: u = uninitialized bit

Note: Refer to *dsPIC30F Family Reference Manual (DS70046)* for descriptions of register bit fields.

8.3 Input Change Notification Module

The input change notification module provides the dsPIC30F devices the ability to generate interrupt requests to the processor, in response to a change of state on selected input pins. This module is capable of detecting input change of states even in Sleep mode, when the clocks are disabled. There are up to 24 external signals (CN0 through CN23) that may be selected (enabled) for generating an interrupt request on a change of state.

TABLE 8-7: INPUT CHANGE NOTIFICATION REGISTER MAP (BITS 15-8)

| SFR Name | Addr. | Bit 15 | Bit 14 | Bit 13 | Bit 12 | Bit 11 | Bit 10 | Bit 9 | Bit 8 | Reset State |
|----------|-------|---------|---------|---------|---------|---------|---------|--------|--------|---------------------|
| CNEN1 | 00C0 | CN15IE | CN14IE | CN13IE | CN12IE | CN11IE | CN10IE | CN9IE | CN8IE | 0000 0000 0000 0000 |
| CNEN2 | 00C2 | — | — | — | — | — | — | — | — | 0000 0000 0000 0000 |
| CNPU1 | 00C4 | CN15PUE | CN14PUE | CN13PUE | CN12PUE | CN11PUE | CN10PUE | CN9PUE | CN8PUE | 0000 0000 0000 0000 |
| CNPU2 | 00C6 | — | — | — | — | — | — | — | — | 0000 0000 0000 0000 |

Legend: u = uninitialized bit

TABLE 8-8: INPUT CHANGE NOTIFICATION REGISTER MAP (BITS 7-0)

| SFR Name | Addr. | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Reset State |
|----------|-------|---------|---------|---------|---------|---------|---------|---------|---------|---------------------|
| CNEN1 | 00C0 | CN7IE | CN6IE | CN5IE | CN4IE | CN3IE | CN2IE | CN1IE | CN0IE | 0000 0000 0000 0000 |
| CNEN2 | 00C2 | CN23IE | CN22IE | CN21IE | CN20IE | CN19IE | CN18IE | CN17IE | CN16IE | 0000 0000 0000 0000 |
| CNPU1 | 00C4 | CN7PUE | CN6PUE | CN5PUE | CN4PUE | CN3PUE | CN2PUE | CN1PUE | CN0PUE | 0000 0000 0000 0000 |
| CNPU2 | 00C6 | CN23PUE | CN22PUE | CN21PUE | CN20PUE | CN19PUE | CN18PUE | CN17PUE | CN16PUE | 0000 0000 0000 0000 |

Legend: u = uninitialized bit

Note: Refer to *dsPIC30F Family Reference Manual* (DS70046) for descriptions of register bit fields.

dsPIC30F

NOTES:

9.0 TIMER1 MODULE

Note: This data sheet summarizes features of this group of dsPIC30F devices and is not intended to be a complete reference source. For more information on the CPU, peripherals, register descriptions and general device functionality, refer to the *dsPIC30F Family Reference Manual* (DS70046).

This section describes the 16-bit General Purpose (GP) Timer1 module and associated Operational modes. Figure 9-1 depicts the simplified block diagram of the 16-bit Timer1 module.

The following sections provide a detailed description including setup and control registers, along with associated block diagrams for the Operational modes of the timers.

The Timer1 module is a 16-bit timer which can serve as the time counter for the real-time clock, or operate as a free-running interval timer/counter. The 16-bit timer has the following modes:

- 16-bit Timer
- 16-bit Synchronous Counter
- 16-bit Asynchronous Counter

Further, the following operational characteristics are supported:

- Timer gate operation
- Selectable prescaler settings
- Timer operation during CPU Idle and Sleep modes
- Interrupt on 16-bit Period register match or falling edge of external gate signal

These Operating modes are determined by setting the appropriate bit(s) in the 16-bit SFR, T1CON. Figure 9-1 presents a block diagram of the 16-bit timer module.

16-bit Timer Mode: In the 16-bit Timer mode, the timer increments on every instruction cycle up to a match value preloaded into the Period register PR1, then resets to '0' and continues to count.

When the CPU goes into the Idle mode, the timer will stop incrementing unless the TSIDL (T1CON<13>) bit = 0. If TSIDL = 1, the timer module logic will resume the incrementing sequence upon termination of the CPU Idle mode.

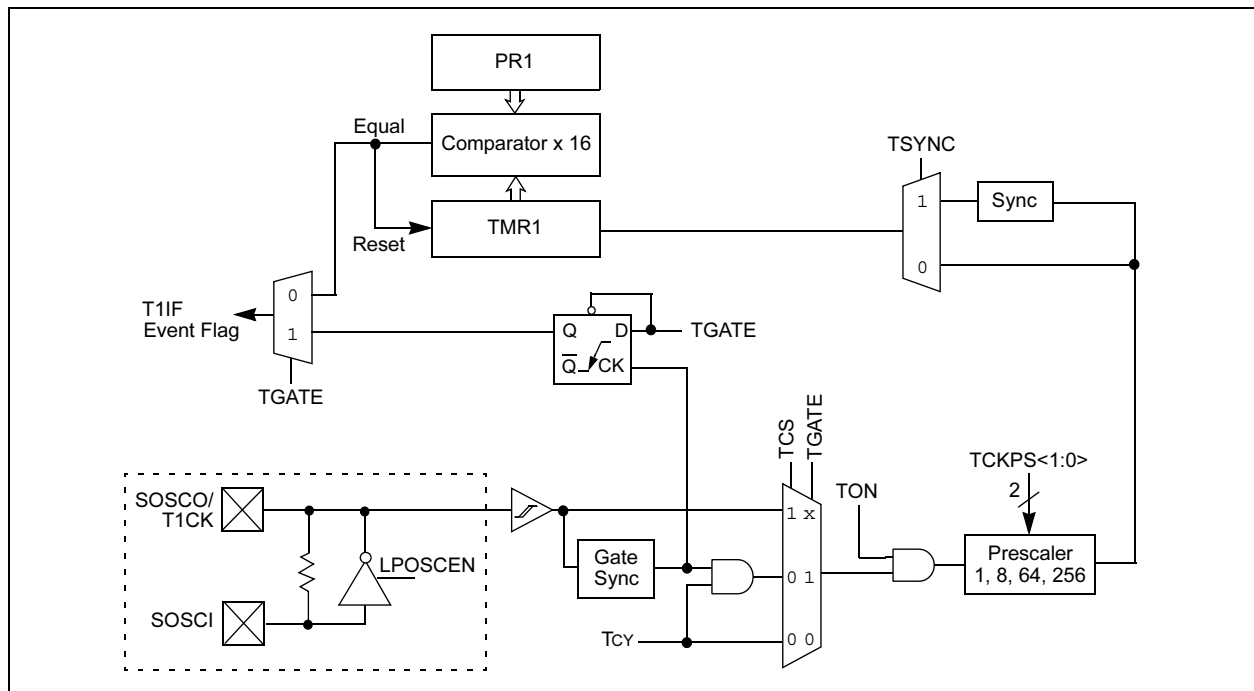
16-bit Synchronous Counter Mode: In the 16-bit Synchronous Counter mode, the timer increments on the rising edge of the applied external clock signal which is synchronized with the internal phase clocks. The timer counts up to a match value preloaded in PR1, then resets to '0' and continues.

When the CPU goes into the Idle mode, the timer will stop incrementing unless the respective TSIDL bit = 0. If TSIDL = 1, the timer module logic will resume the incrementing sequence upon termination of the CPU Idle mode.

16-bit Asynchronous Counter Mode: In the 16-bit Asynchronous Counter mode, the timer increments on every rising edge of the applied external clock signal. The timer counts up to a match value preloaded in PR1, then resets to '0' and continues.

When the timer is configured for the Asynchronous mode of operation and the CPU goes into the Idle mode, the timer will stop incrementing if TSIDL = 1.

FIGURE 9-1: 16-BIT TIMER1 MODULE BLOCK DIAGRAM



9.1 Timer Gate Operation

The 16-bit timer can be placed in the Gated Time Accumulation mode. This mode allows the internal TCY to increment the respective timer when the gate input signal (T1CK pin) is asserted high. Control bit TGATE (T1CON<6>) must be set to enable this mode. The timer must be enabled (TON = 1) and the timer clock source set to internal (TCS = 0).

When the CPU goes into the Idle mode, the timer will stop incrementing unless TSIDL = 0. If TSIDL = 1, the timer will resume the incrementing sequence upon termination of the CPU Idle mode.

9.2 Timer Prescaler

The input clock (FOSC/4 or external clock) to the 16-bit Timer has a prescale option of 1:1, 1:8, 1:64 and 1:256, selected by control bits TCKPS<1:0> (T1CON<5:4>). The prescaler counter is cleared when any of the following occurs:

- a write to the TMR1 register
- clearing of the TON bit (T1CON<15>)
- device Reset, such as POR and BOR

However, if the timer is disabled (TON = 0), then the timer prescaler cannot be reset since the prescaler clock is halted.

TMR1 is not cleared when T1CON is written. It is cleared by writing to the TMR1 register.

9.3 Timer Operation During Sleep Mode

During CPU Sleep mode, the timer will operate if:

- The timer module is enabled (TON = 1) and
- The timer clock source is selected as external (TCS = 1) and
- The TSYNC bit (T1CON<2>) is asserted to a logic '0' which defines the external clock source as asynchronous.

When all three conditions are true, the timer will continue to count up to the Period register and be reset to 0x0000.

When a match between the timer and the Period register occurs, an interrupt can be generated if the respective timer interrupt enable bit is asserted.

9.4 Timer Interrupt

The 16-bit timer has the ability to generate an interrupt on period match. When the timer count matches the Period register, the T1IF bit is asserted and an interrupt will be generated if enabled. The T1IF bit must be cleared in software. The timer interrupt flag, T1IF, is located in the IFS0 Control register in the interrupt controller.

When the Gated Time Accumulation mode is enabled, an interrupt will also be generated on the falling edge of the gate signal (at the end of the accumulation cycle).

Enabling an interrupt is accomplished via the respective timer interrupt enable bit, T1IE. The timer interrupt enable bit is located in the IEC0 Control register in the interrupt controller.

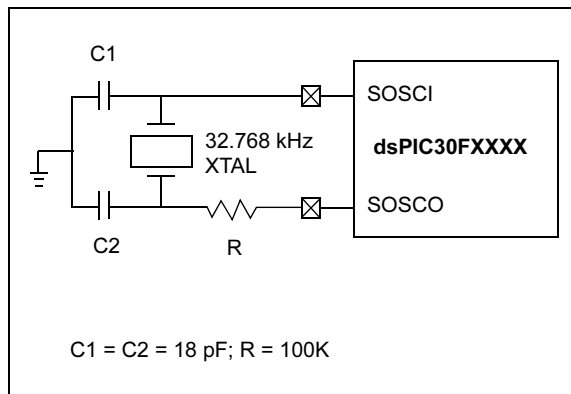
9.5 Real-Time Clock

Timer1, when operating in Real-Time Clock (RTC) mode, provides time of day and event time-stamping capabilities. Key operational features of the RTC are:

- Operation from 32 kHz LP oscillator
- 8-bit prescaler
- Low power
- Real-Time Clock interrupts

These Operating modes are determined by setting the appropriate bit(s) in the T1CON Control register.

FIGURE 9-2: RECOMMENDED COMPONENTS FOR TIMER1 LP OSCILLATOR RTC



9.5.1 RTC OSCILLATOR OPERATION

When the $TON = 1$, $TCS = 1$ and $TGATE = 0$, the timer increments on the rising edge of the 32 kHz LP oscillator output signal, up to the value specified in the Period register and is then reset to '0'.

The $TSYNC$ bit must be asserted to a logic '0' (Asynchronous mode) for correct operation.

Enabling $LPOSCEN$ ($OSCCON<1>$) will disable the normal Timer and Counter modes and enable a timer carry-out wake-up event.

When the CPU enters Sleep mode, the RTC will continue to operate provided the 32 kHz external crystal oscillator is active and the control bits have not been changed. The $TSIDL$ bit should be cleared to '0' in order for RTC to continue operation in Idle mode.

9.5.2 RTC INTERRUPTS

When an interrupt event occurs, the respective interrupt flag, $T1IF$, is asserted and an interrupt will be generated if enabled. The $T1IF$ bit must be cleared in software. The respective Timer interrupt flag, $T1IF$, is located in the $IFS0$ Status register in the interrupt controller.

Enabling an interrupt is accomplished via the respective timer interrupt enable bit, $T1IE$. The timer interrupt enable bit is located in the $IEC0$ Control register in the interrupt controller.

TABLE 9-1: TIMER1 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 | Reset State |
|----------|-------|-------------------|--------|--------|--------|--------|--------|-------|-------|-------|-------|--------|--------|-------|-------|-------|-------|---------------------|
| TMR1 | 0100 | Timer1 Register | | | | | | | | | | | | | | | | |
| PR1 | 0102 | Period Register 1 | | | | | | | | | | | | | | | | |
| T1CON | 0104 | TON | — | TSIDL | — | — | — | — | — | — | TGATE | TCKPS1 | TCKPS0 | — | TSYNC | TCS | — | 0000 0000 0000 0000 |

Legend: u = uninitialized bit

Note: Refer to *dsPIC30F Family Reference Manual (DS70046)* for descriptions of register bit fields.

10.0 TIMER2/3 MODULE

Note: This data sheet summarizes features of this group of dsPIC30F devices and is not intended to be a complete reference source. For more information on the CPU, peripherals, register descriptions and general device functionality, refer to the *dsPIC30F Family Reference Manual* (DS70046).

This section describes the 32-bit General Purpose (GP) Timer module (Timer2/3) and associated Operational modes. Figure 10-1 depicts the simplified block diagram of the 32-bit Timer2/3 module. Figure 10-2 and Figure 10-3 show Timer2/3 configured as two independent 16-bit timers, Timer2 and Timer3, respectively.

The Timer2/3 module is a 32-bit timer (which can be configured as two 16-bit timers) with selectable Operating modes. These timers are utilized by other peripheral modules, such as:

- Input Capture
- Output Compare/Simple PWM

The following sections provide a detailed description, including setup and control registers, along with associated block diagrams for the Operational modes of the timers.

The 32-bit timer has the following modes:

- Two independent 16-bit timers (Timer2 and Timer3) with all 16-bit Operating modes (except Asynchronous Counter mode)
- Single 32-bit timer operation
- Single 32-bit synchronous counter

Further, the following operational characteristics are supported:

- ADC event trigger
- Timer gate operation
- Selectable prescaler settings
- Timer operation during Idle and Sleep modes
- Interrupt on a 32-bit period register match

These Operating modes are determined by setting the appropriate bit(s) in the 16-bit T2CON and T3CON SFRs.

For 32-bit timer/counter operation, Timer2 is the LS Word and Timer3 is the MS Word of the 32-bit timer.

Note: For 32-bit timer operation, T3CON control bits are ignored. Only T2CON control bits are used for setup and control. Timer2 clock and gate inputs are utilized for the 32-bit timer module but an interrupt is generated with the Timer3 interrupt flag (T3IF) and the interrupt is enabled with the Timer3 interrupt enable bit (T3IE).

16-bit Timer Mode: In the 16-bit mode, Timer2 and Timer3 can be configured as two independent 16-bit timers. Each timer can be set up in either 16-bit Timer mode or 16-bit Synchronous Counter mode. See Section 9.0, Timer1 Module for details on these two Operating modes.

The only functional difference between Timer2 and Timer3 is that Timer2 provides synchronization of the clock prescaler output. This is useful for high frequency external clock inputs.

32-bit Timer Mode: In the 32-bit Timer mode, the timer increments on every instruction cycle, up to a match value preloaded into the combined 32-bit Period register PR3/PR2, then resets to '0' and continues to count.

For synchronous 32-bit reads of the Timer2/Timer3 pair, reading the LS Word (TMR2 register) will cause the MS word to be read and latched into a 16-bit holding register, termed TMR3HLD.

For synchronous 32-bit writes, the holding register (TMR3HLD) must first be written to. When followed by a write to the TMR2 register, the contents of TMR3HLD will be transferred and latched into the MSB of the 32-bit timer (TMR3).

32-bit Synchronous Counter Mode: In the 32-bit Synchronous Counter mode, the timer increments on the rising edge of the applied external clock signal which is synchronized with the internal phase clocks. The timer counts up to a match value preloaded in the combined 32-bit period register PR3/PR2, then resets to '0' and continues.

When the timer is configured for the Synchronous Counter mode of operation and the CPU goes into the Idle mode, the timer will stop incrementing unless the TSIDL (T2CON<13>) bit = 0. If TSIDL = 1, the timer module logic will resume the incrementing sequence upon termination of the CPU Idle mode.

Note: In some devices, one or more of the TxCK pins may be absent. Therefore, for such timers, the following modes should not be used:

1. TCS = 1 (16-bit counter)
2. TCS = 0, TGATE = 1 (gated time accumulation).

dsPIC30F

FIGURE 10-1: 32-BIT TIMER2/3 BLOCK DIAGRAM

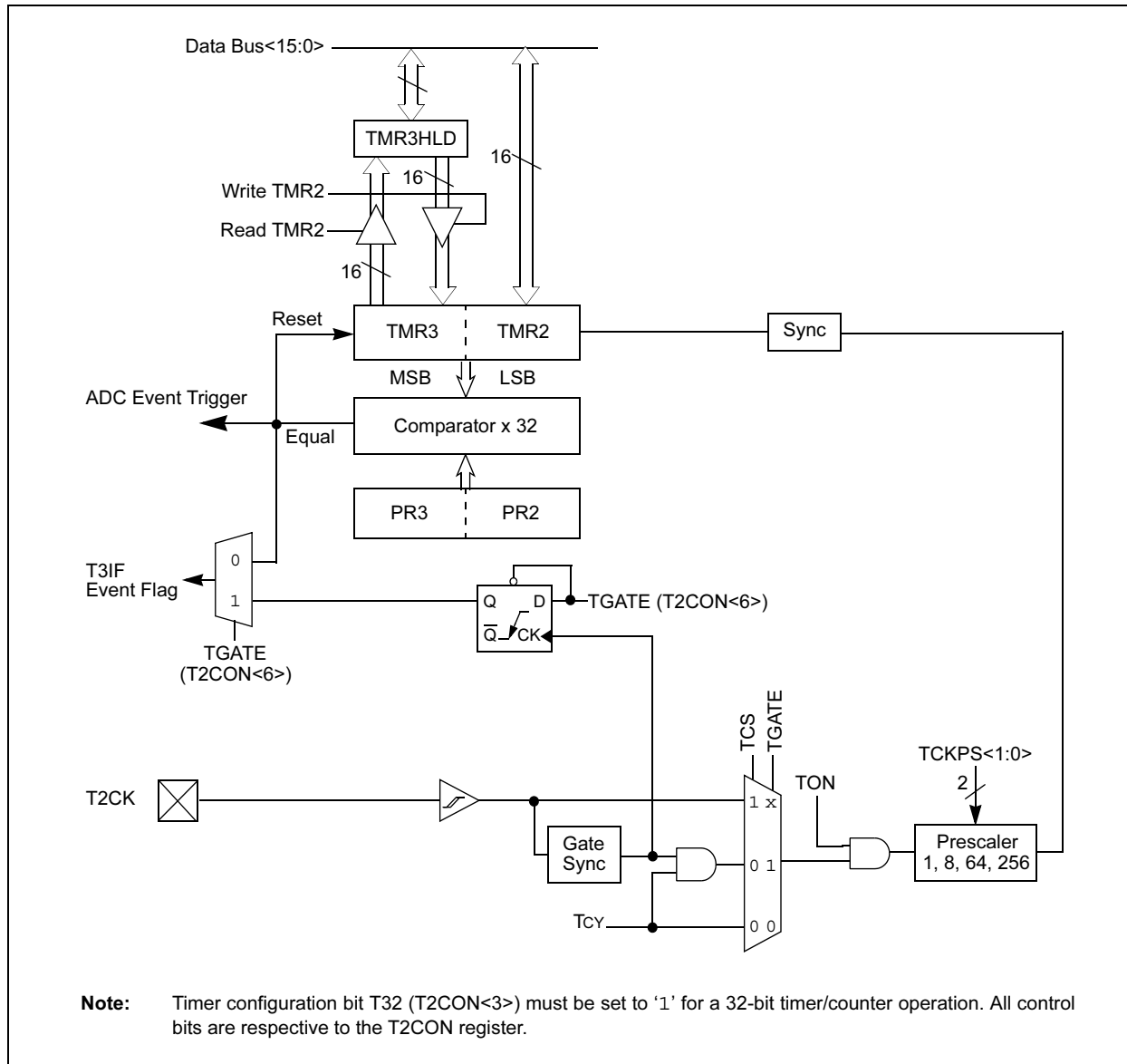


FIGURE 10-2: 16-BIT TIMER2 BLOCK DIAGRAM

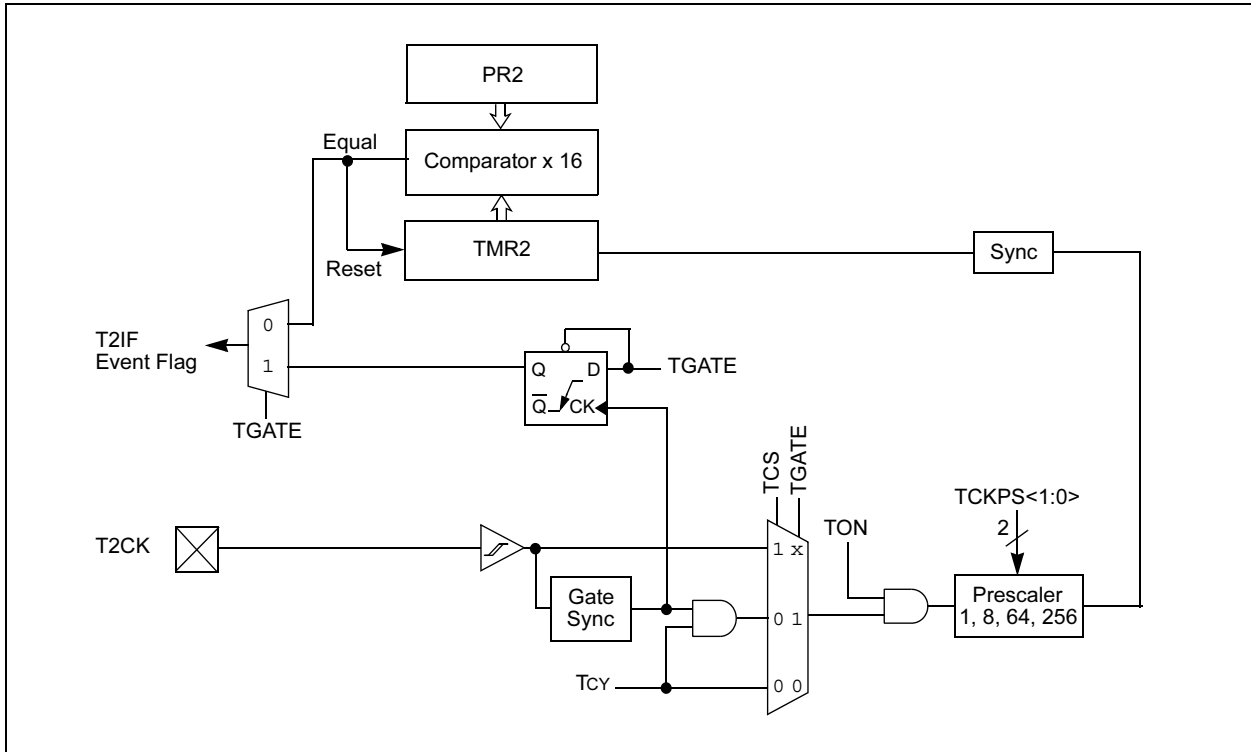
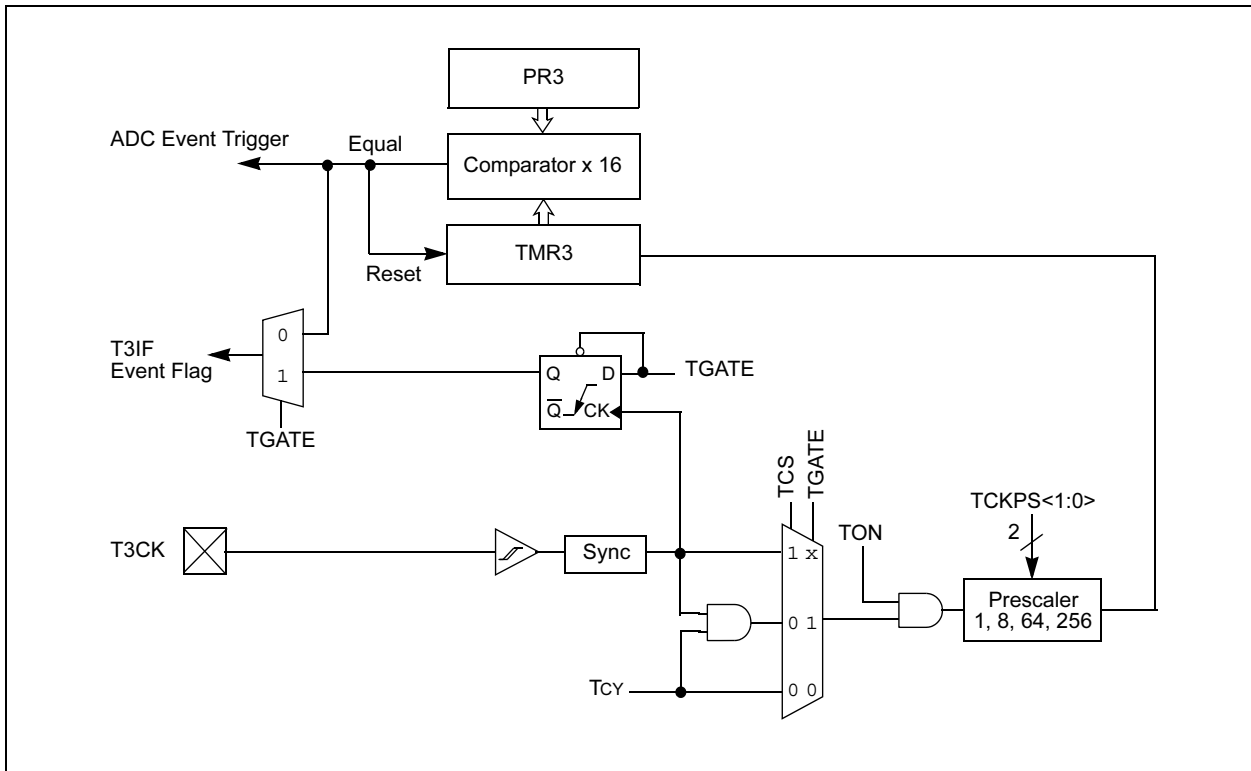


FIGURE 10-3: 16-BIT TIMER3 BLOCK DIAGRAM



10.1 Timer Gate Operation

The 32-bit timer can be placed in the Gated Time Accumulation mode. This mode allows the internal TCY to increment the respective timer when the gate input signal (T2CK pin) is asserted high. Control bit TGATE (T2CON<6>) must be set to enable this mode. When in this mode, Timer2 is the originating clock source. The TGATE setting is ignored for Timer3. The timer must be enabled (TON = 1) and the timer clock source set to internal (TCS = 0).

The falling edge of the external signal terminates the count operation but does not reset the timer. The user must reset the timer in order to start counting from zero.

10.2 ADC Event Trigger

When a match occurs between the 32-bit timer (TMR3/TMR2) and the 32-bit combined period register (PR3/PR2), or between the 16-bit timer (TMR3) and the 16-bit period register (PR3), a special ADC trigger event signal is generated by Timer3.

10.3 Timer Prescaler

The input clock (FOSC/4 or external clock) to the timer has a prescale option of 1:1, 1:8, 1:64, and 1:256, selected by control bits TCKPS<1:0> (T2CON<5:4> and T3CON<5:4>). For the 32-bit timer operation, the originating clock source is Timer2. The prescaler operation for Timer3 is not applicable in this mode. The prescaler counter is cleared when any of the following occurs:

- a write to the TMR2/TMR3 register
- clearing either of the TON (T2CON<15> or T3CON<15>) bits to '0'
- device Reset, such as POR and BOR

However, if the timer is disabled (TON = 0), then the Timer 2 prescaler cannot be reset since the prescaler clock is halted.

TMR2/TMR3 is not cleared when T2CON/T3CON is written.

10.4 Timer Operation During Sleep Mode

During CPU Sleep mode, the timer will not operate because the internal clocks are disabled.

10.5 Timer Interrupt

The 32-bit timer module can generate an interrupt on period match or on the falling edge of the external gate signal. When the 32-bit timer count matches the respective 32-bit period register, or the falling edge of the external "gate" signal is detected, the T3IF bit (IFS0<7>) is asserted and an interrupt will be generated if enabled. In this mode, the T3IF interrupt flag is used as the source of the interrupt. The T3IF bit must be cleared in software.

Enabling an interrupt is accomplished via the respective timer interrupt enable bit, T3IE (IEC0<7>).

TABLE 10-1: TIMER2/3 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 | Reset State | |
|----------|-------|--|--------|--------|--------|--------|--------|-------|-------|-------|-------|--------|--------|-------|-------|-------|-------|---------------------|---------------------|
| TMR2 | 0106 | Timer2 Register | | | | | | | | | | | | | | | | | uuuu uuuu uuuu uuuu |
| TMR3HLD | 0108 | Timer3 Holding Register (for 32-bit timer operations only) | | | | | | | | | | | | | | | | | uuuu uuuu uuuu uuuu |
| TMR3 | 010A | Timer3 Register | | | | | | | | | | | | | | | | | uuuu uuuu uuuu uuuu |
| PR2 | 010C | Period Register 2 | | | | | | | | | | | | | | | | | 1111 1111 1111 1111 |
| PR3 | 010E | Period Register 3 | | | | | | | | | | | | | | | | | 1111 1111 1111 1111 |
| T2CON | 0110 | TON | — | TSIDL | — | — | — | — | — | — | TGATE | TCKPS1 | TCKPS0 | T32 | — | TCS | — | 0000 0000 0000 0000 | |
| T3CON | 0112 | TON | — | TSIDL | — | — | — | — | — | — | TGATE | TCKPS1 | TCKPS0 | — | — | TCS | — | 0000 0000 0000 0000 | |

Legend: u = uninitialized bit

Note: Refer to *dsPIC30F Family Reference Manual (DS70046)* for descriptions of register bit fields.

dsPIC30F

NOTES:

11.0 TIMER4/5 MODULE

Note: This data sheet summarizes features of this group of dsPIC30F devices and is not intended to be a complete reference source. For more information on the CPU, peripherals, register descriptions and general device functionality, refer to the *dsPIC30F Family Reference Manual* (DS70046).

This section describes the second 32-bit General Purpose (GP) Timer module (Timer4/5) and associated Operational modes. Figure 11-1 depicts the simplified block diagram of the 32-bit Timer4/5 module. Figure 11-2 and Figure 11-3 show Timer4/5 configured as two independent 16-bit timers, Timer4 and Timer5, respectively.

The Timer4/5 module is similar in operation to the Timer2/3 module. However, there are some differences which are listed below:

- The Timer4/5 module does not support the ADC event trigger feature
- Timer4/5 can not be utilized by other peripheral modules, such as input capture and output compare

The Operating modes of the Timer4/5 module are determined by setting the appropriate bit(s) in the 16-bit T4CON and T5CON SFRs.

For 32-bit timer/counter operation, Timer4 is the LS Word and Timer5 is the MS Word of the 32-bit timer.

Note: For 32-bit timer operation, T5CON control bits are ignored. Only T4CON control bits are used for setup and control. Timer4 clock and gate inputs are utilized for the 32-bit timer module but an interrupt is generated with the Timer5 interrupt flag (T5IF) and the interrupt is enabled with the Timer5 interrupt enable bit (T5IE).

Note: In some devices, one or more of the TxCK pins may be absent. Therefore, for such timers, the following modes should not be used:

1. TCS = 1 (16-bit counter)
2. TCS = 0, TGATE = 1 (gated time accumulation)

FIGURE 11-1: 32-BIT TIMER4/5 BLOCK DIAGRAM

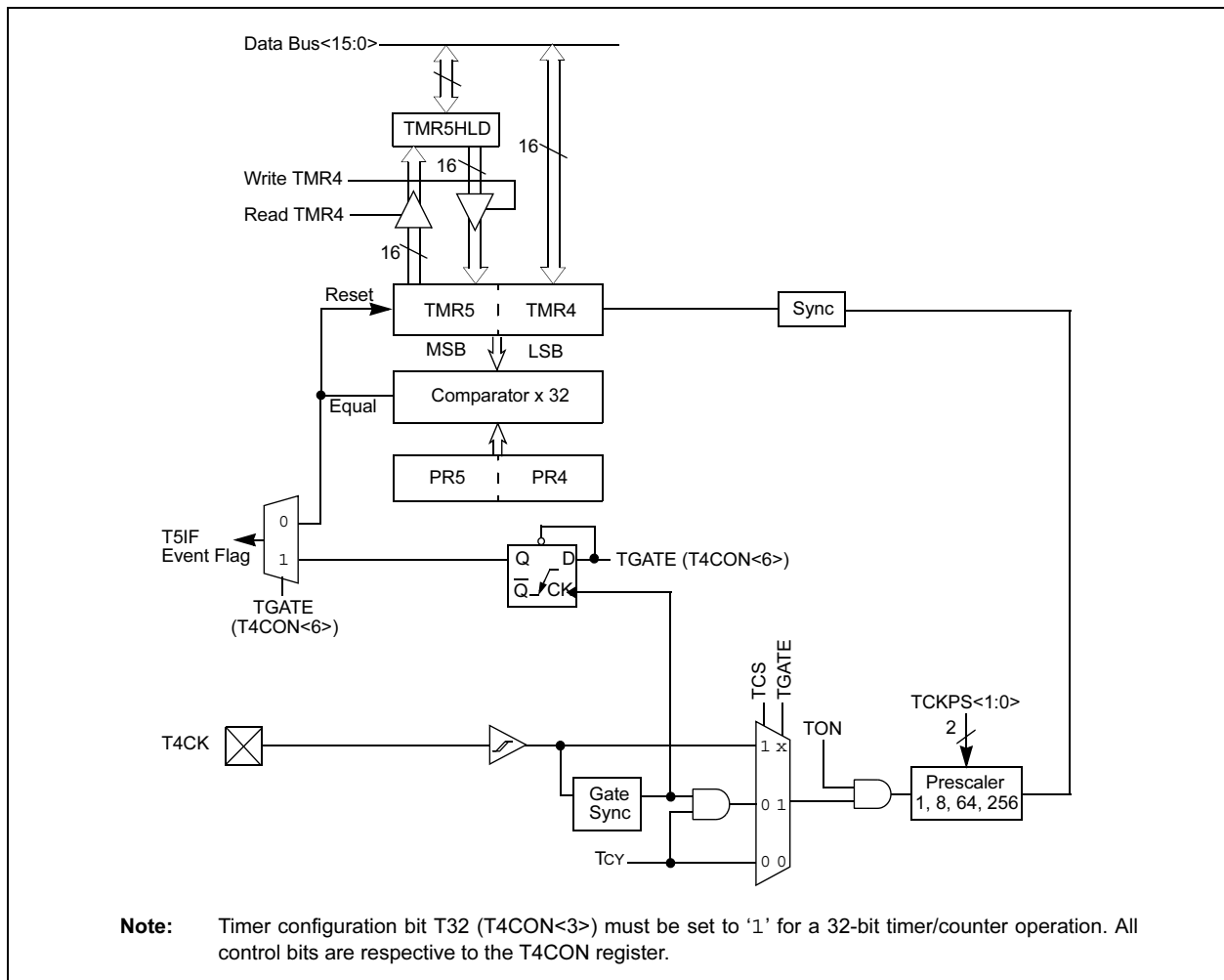


FIGURE 11-2: 16-BIT TIMER4 BLOCK DIAGRAM

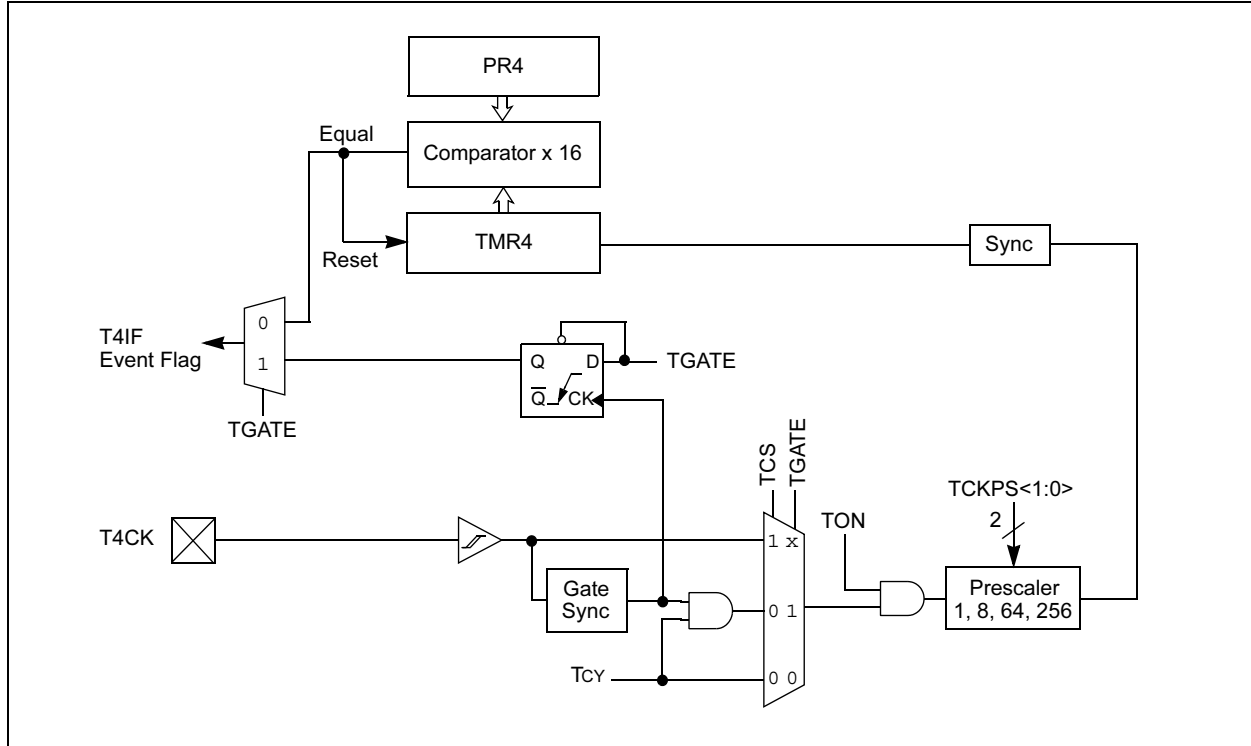


FIGURE 11-3: 16-BIT TIMER5 BLOCK DIAGRAM

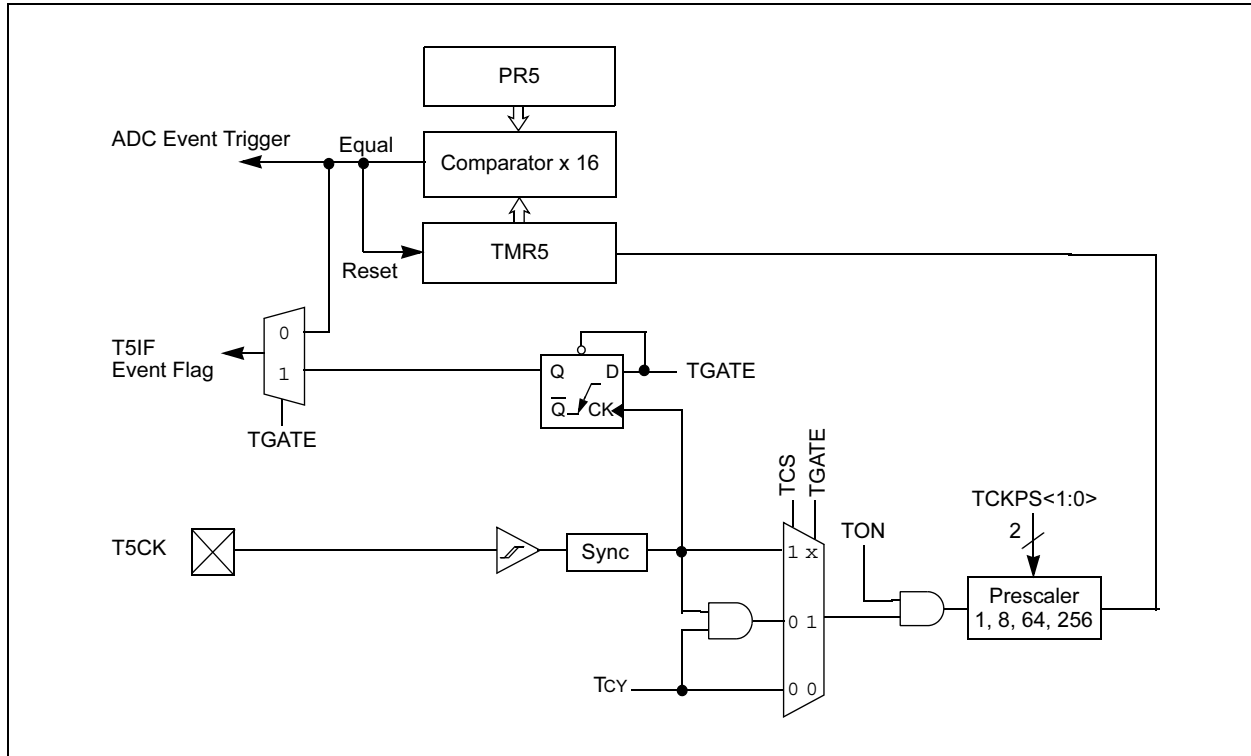


TABLE 11-1: TIMER4/5 REGISTER MAP

| 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 | Reset State | |
|----------|-------|---|--------|--------|--------|--------|--------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------------|---------------------|
| TMR4 | 0114 | Timer 4 Register | | | | | | | | | | | | | | | | | uuuu uuuu uuuu uuuu |
| TMR5HLD | 0116 | Timer 5 Holding Register (for 32-bit operations only) | | | | | | | | | | | | | | | | | uuuu uuuu uuuu uuuu |
| TMR5 | 0118 | Timer 5 Register | | | | | | | | | | | | | | | | | uuuu uuuu uuuu uuuu |
| PR4 | 011A | Period Register 4 | | | | | | | | | | | | | | | | | 1111 1111 1111 1111 |
| PR5 | 011C | Period Register 5 | | | | | | | | | | | | | | | | | 1111 1111 1111 1111 |
| T4CON | 011E | TON | — | TSIDL | — | — | — | — | — | — | — | — | — | — | — | — | — | — | 0000 0000 0000 0000 |
| T5CON | 0120 | TON | — | TSIDL | — | — | — | — | — | — | — | — | — | — | — | — | — | — | 0000 0000 0000 0000 |

Legend: u = uninitialized

Note: Refer to *dsPIC30F Family Reference Manual (DS70046)* for descriptions of register bit fields.

dsPIC30F

NOTES:

12.0 INPUT CAPTURE MODULE

Note: This data sheet summarizes features of this group of dsPIC30F devices and is not intended to be a complete reference source. For more information on the CPU, peripherals, register descriptions and general device functionality, refer to the *dsPIC30F Family Reference Manual* (DS70046).

This section describes the input capture module and associated Operational modes. The features provided by this module are useful in applications requiring frequency (period) and pulse measurement. Figure 12-1 depicts a block diagram of the input capture module. Input capture is useful for such modes as:

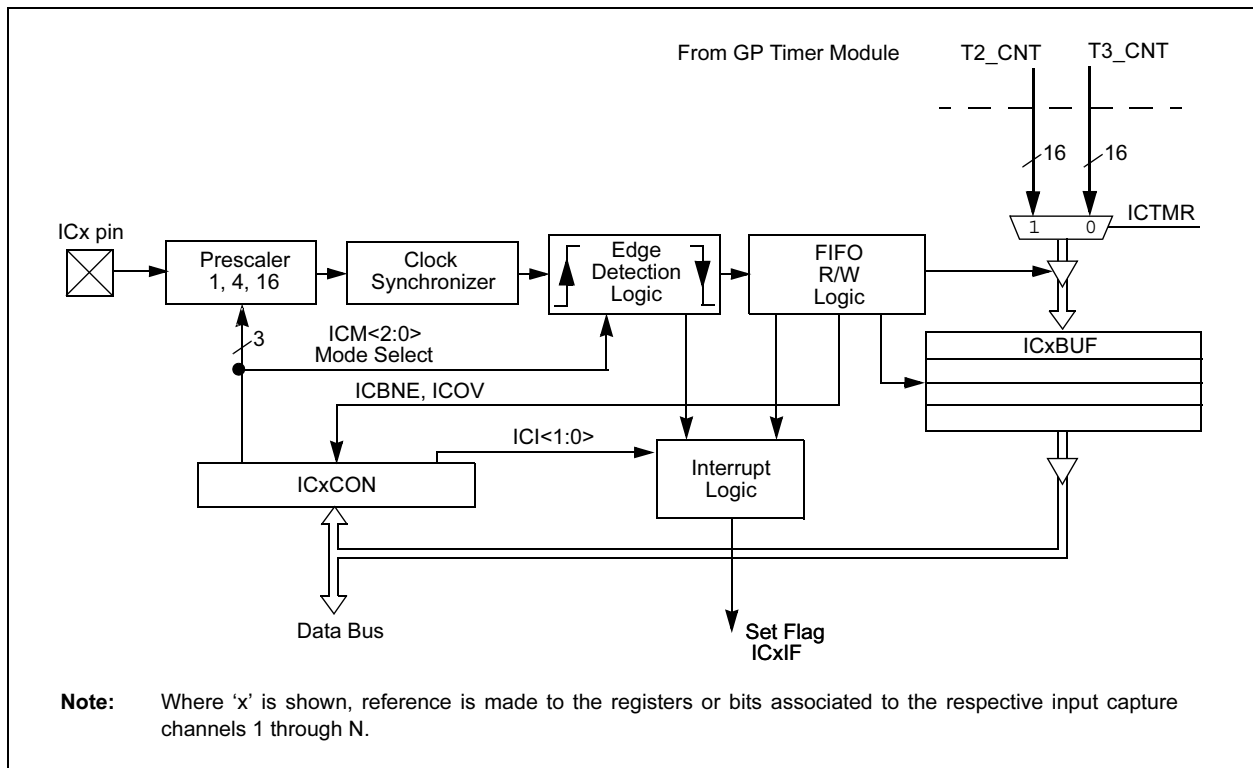
- Frequency/Period/Pulse Measurements
- Additional Sources of External Interrupts

The key operational features of the input capture module are:

- Simple Capture Event mode
- Timer2 and Timer3 mode selection
- Interrupt on input capture event

These Operating modes are determined by setting the appropriate bits in the ICxCON register (where x = 1,2,...,N). The dsPIC devices contain up to 8 capture channels (i.e., the maximum value of N is 8).

FIGURE 12-1: INPUT CAPTURE MODE BLOCK DIAGRAM



12.1 Simple Capture Event Mode

The simple capture events in the dsPIC30F product family are:

- Capture every falling edge
- Capture every rising edge
- Capture every 4th rising edge
- Capture every 16th rising edge
- Capture every rising and falling edge

These simple Input Capture modes are configured by setting the appropriate bits ICM<2:0> (ICxCON<2:0>).

12.1.1 CAPTURE PRESCALER

There are four input capture prescaler settings specified by bits ICM<2:0> (ICxCON<2:0>). Whenever the capture channel is turned off, the prescaler counter will be cleared. In addition, any Reset will clear the prescaler counter.

12.1.2 CAPTURE BUFFER OPERATION

Each capture channel has an associated FIFO buffer which is four 16-bit words deep. There are two status flags which provide status on the FIFO buffer:

- ICBFNE - Input Capture Buffer Not Empty
- ICOV - Input Capture Overflow

The ICBFNE will be set on the first input capture event and remain set until all capture events have been read from the FIFO. As each word is read from the FIFO, the remaining words are advanced by one position within the buffer.

In the event that the FIFO is full with four capture events and a fifth capture event occurs prior to a read of the FIFO, an overflow condition will occur and the ICOV bit will be set to a logic '1'. The fifth capture event is lost and is not stored in the FIFO. No additional events will be captured until all four events have been read from the buffer.

If a FIFO read is performed after the last read and no new capture event has been received, the read will yield indeterminate results.

12.1.3 TIMER2 AND TIMER3 SELECTION MODE

The input capture module consists of up to 8 input capture channels. Each channel can select between one of two timers for the time base, Timer2 or Timer3.

Selection of the timer resource is accomplished through SFR bit, ICTMR (ICxCON<7>). Timer3 is the default timer resource available for the input capture module.

12.1.4 HALL SENSOR MODE

When the input capture module is set for capture on every edge, rising and falling, ICM<2:0> = 001, the following operations are performed by the input capture logic:

- The input capture interrupt flag is set on every edge, rising and falling.
- The interrupt on Capture mode setting bits, ICI<1:0>, is ignored since every capture generates an interrupt.
- A capture overflow condition is not generated in this mode.

12.2 Input Capture Operation During Sleep and Idle Modes

An input capture event will generate a device wake-up or interrupt, if enabled, if the device is in CPU Idle or Sleep mode.

Independent of the timer being enabled, the input capture module will wake-up from the CPU Sleep or Idle mode when a capture event occurs if ICM<2:0> = 111 and the interrupt enable bit is asserted. The same wake-up can generate an interrupt if the conditions for processing the interrupt have been satisfied. The wake-up feature is useful as a method of adding extra external pin interrupts.

12.2.1 INPUT CAPTURE IN CPU SLEEP MODE

CPU Sleep mode allows input capture module operation with reduced functionality. In the CPU Sleep mode, the ICI<1:0> bits are not applicable and the input capture module can only function as an external interrupt source.

The capture module must be configured for interrupt only on rising edge (ICM<2:0> = 111) in order for the input capture module to be used while the device is in Sleep mode. The prescale settings of 4:1 or 16:1 are not applicable in this mode.

12.2.2 INPUT CAPTURE IN CPU IDLE MODE

CPU Idle mode allows input capture module operation with full functionality. In the CPU Idle mode, the Interrupt mode selected by the ICI<1:0> bits is applicable, as well as the 4:1 and 16:1 capture prescale settings which are defined by control bits ICM<2:0>. This mode requires the selected timer to be enabled. Moreover, the ICSIDL bit must be asserted to a logic '0'.

If the input capture module is defined as ICM<2:0> = 111 in CPU Idle mode, the input capture pin will serve only as an external interrupt pin.

12.3 Input Capture Interrupts

The input capture channels have the ability to generate an interrupt based upon the selected number of capture events. The selection number is set by control bits ICI<1:0> (ICxCON<6:5>).

Each channel provides an interrupt flag (ICxIF) bit. The respective capture channel interrupt flag is located in the corresponding IFSx Status register.

Enabling an interrupt is accomplished via the respective capture channel interrupt enable (ICxIE) bit. The capture interrupt enable bit is located in the corresponding IEC Control register.

TABLE 12-1: INPUT CAPTURE 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 | Reset State |
|----------|-------|--------|--------|--------|--------|--------|--------|--------------------------|-------|-------|----------|-------|-------|-------|----------|-------|-------|---------------------|
| IC1BUF | 0140 | | | | | | | Input 1 Capture Register | | | | | | | | | | uuuu uuuu uuuu uuuu |
| IC1CON | 0142 | — | — | ICSIDL | — | — | — | — | — | ICTMR | ICI<1:0> | ICOV | ICBNE | ICBNE | ICM<2:0> | | | 0000 0000 0000 0000 |
| IC2BUF | 0144 | | | | | | | Input 2 Capture Register | | | | | | | | | | uuuu uuuu uuuu uuuu |
| IC2CON | 0146 | — | — | ICSIDL | — | — | — | — | — | ICTMR | ICI<1:0> | ICOV | ICBNE | ICBNE | ICM<2:0> | | | 0000 0000 0000 0000 |
| IC3BUF | 0148 | | | | | | | Input 3 Capture Register | | | | | | | | | | uuuu uuuu uuuu uuuu |
| IC3CON | 014A | — | — | ICSIDL | — | — | — | — | — | ICTMR | ICI<1:0> | ICOV | ICBNE | ICBNE | ICM<2:0> | | | 0000 0000 0000 0000 |
| IC4BUF | 014C | | | | | | | Input 4 Capture Register | | | | | | | | | | uuuu uuuu uuuu uuuu |
| IC4CON | 014E | — | — | ICSIDL | — | — | — | — | — | ICTMR | ICI<1:0> | ICOV | ICBNE | ICBNE | ICM<2:0> | | | 0000 0000 0000 0000 |
| IC5BUF | 0150 | | | | | | | Input 5 Capture Register | | | | | | | | | | uuuu uuuu uuuu uuuu |
| IC5CON | 0152 | — | — | ICSIDL | — | — | — | — | — | ICTMR | ICI<1:0> | ICOV | ICBNE | ICBNE | ICM<2:0> | | | 0000 0000 0000 0000 |
| IC6BUF | 0154 | | | | | | | Input 6 Capture Register | | | | | | | | | | uuuu uuuu uuuu uuuu |
| IC6CON | 0156 | — | — | ICSIDL | — | — | — | — | — | ICTMR | ICI<1:0> | ICOV | ICBNE | ICBNE | ICM<2:0> | | | 0000 0000 0000 0000 |
| IC7BUF | 0158 | | | | | | | Input 7 Capture Register | | | | | | | | | | uuuu uuuu uuuu uuuu |
| IC7CON | 015A | — | — | ICSIDL | — | — | — | — | — | ICTMR | ICI<1:0> | ICOV | ICBNE | ICBNE | ICM<2:0> | | | 0000 0000 0000 0000 |
| IC8BUF | 015C | | | | | | | Input 8 Capture Register | | | | | | | | | | uuuu uuuu uuuu uuuu |
| IC8CON | 015E | — | — | ICSIDL | — | — | — | — | — | ICTMR | ICI<1:0> | ICOV | ICBNE | ICBNE | ICM<2:0> | | | 0000 0000 0000 0000 |

Legend: u = uninitialized bit

Note: Refer to *dsPIC30F Family Reference Manual (DS70046)* for descriptions of register bit fields.

13.0 OUTPUT COMPARE MODULE

Note: This data sheet summarizes features of this group of dsPIC30F devices and is not intended to be a complete reference source. For more information on the CPU, peripherals, register descriptions and general device functionality, refer to the *dsPIC30F Family Reference Manual* (DS70046).

This section describes the output compare module and associated Operational modes. The features provided by this module are useful in applications requiring Operational modes, such as:

- Generation of Variable Width Output Pulses
- Power Factor Correction

Figure 13-1 depicts a block diagram of the output compare module.

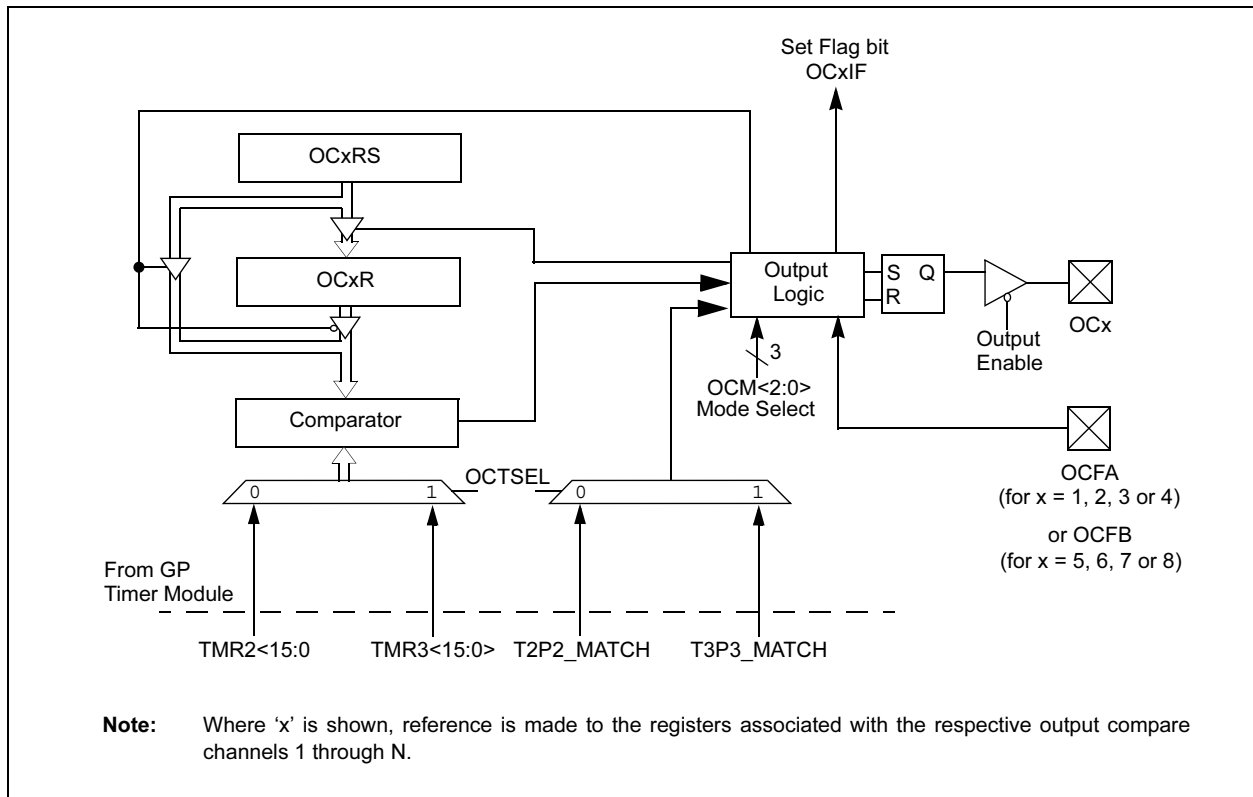
The key operational features of the output compare module include:

- Timer2 and Timer3 Selection mode
- Simple Output Compare Match mode
- Dual Output Compare Match mode
- Simple PWM mode
- Output Compare During Sleep and Idle modes
- Interrupt on Output Compare/PWM Event

These Operating modes are determined by setting the appropriate bits in the 16-bit OCxCON SFR (where $x = 1, 2, 3, \dots, N$). The dsPIC devices contain up to 8 compare channels (i.e., the maximum value of N is 8).

OCxRS and OCxR in Figure 13-1 represent the Dual Compare registers. In the Dual Compare mode, the OCxR register is used for the first compare and OCxRS is used for the second compare.

FIGURE 13-1: OUTPUT COMPARE MODE BLOCK DIAGRAM



13.1 Timer2 and Timer3 Selection Mode

Each output compare channel can select between one of two 16-bit timers, Timer2 or Timer3.

The selection of the timers is controlled by the OCTSEL bit (OCxCON<3>). Timer2 is the default timer resource for the output compare module.

13.2 Simple Output Compare Match Mode

When control bits OCM<2:0> (OCxCON<2:0>) = 001, 010 or 011, the selected output compare channel is configured for one of three simple Output Compare Match modes:

- Compare forces I/O pin low
- Compare forces I/O pin high
- Compare toggles I/O pin

The OCxR register is used in these modes. The OCxR register is loaded with a value and is compared to the selected incrementing timer count. When a compare occurs, one of these Compare Match modes occurs. If the counter resets to zero before reaching the value in OCxR, the state of the OCx pin remains unchanged.

13.3 Dual Output Compare Match Mode

When control bits OCM<2:0> (OCxCON<2:0>) = 100 or 101, the selected output compare channel is configured for one of two Dual Output Compare modes, which are:

- Single Output Pulse mode
- Continuous Output Pulse mode

13.3.1 SINGLE PULSE MODE

For the user to configure the module for the generation of a single output pulse, the following steps are required (assuming timer is off):

- Determine instruction cycle time T_{CY} .
- Calculate desired pulse width value based on T_{CY} .
- Calculate time to start pulse from timer start value of $0x0000$.
- Write pulse width start and stop times into OCxR and OCxRS Compare registers (x denotes channel 1, 2, ...,N).
- Set Timer Period register to value equal to, or greater than value in OCxRS Compare register.
- Set OCM<2:0> = 100.
- Enable timer, TON (TxCON<15>) = 1.

To initiate another single pulse, issue another write to set OCM<2:0> = 100.

13.3.2 CONTINUOUS PULSE MODE

For the user to configure the module for the generation of a continuous stream of output pulses, the following steps are required:

- Determine instruction cycle time T_{CY} .
- Calculate desired pulse value based on T_{CY} .
- Calculate timer to start pulse width from timer start value of $0x0000$.
- Write pulse width start and stop times into OCxR and OCxRS (x denotes channel 1, 2, ...,N) Compare registers, respectively.
- Set Timer Period register to value equal to, or greater than value in OCxRS Compare register.
- Set OCM<2:0> = 101.
- Enable timer, TON (TxCON<15>) = 1.

13.4 Simple PWM Mode

When control bits OCM<2:0> (OCxCON<2:0>) = 110 or 111, the selected output compare channel is configured for the PWM mode of operation. When configured for the PWM mode of operation, OCxR is the main latch (read only) and OCxRS is the secondary latch. This enables glitchless PWM transitions.

The user must perform the following steps in order to configure the output compare module for PWM operation:

1. Set the PWM period by writing to the appropriate period register.
2. Set the PWM duty cycle by writing to the OCxRS register.
3. Configure the output compare module for PWM operation.
4. Set the TMRx prescale value and enable the Timer, TON (TxCON<15>) = 1.

13.4.1 INPUT PIN FAULT PROTECTION FOR PWM

When control bits OCM<2:0> (OCxCON<2:0>) = 111, the selected output compare channel is again configured for the PWM mode of operation with the additional feature of input FAULT protection. While in this mode, if a logic '0' is detected on the OCFA/B pin, the respective PWM output pin is placed in the high impedance input state. The OCFLT bit (OCxCON<4>) indicates whether a FAULT condition has occurred. This state will be maintained until both of the following events have occurred:

- The external FAULT condition has been removed.
- The PWM mode has been re-enabled by writing to the appropriate control bits.

13.4.2 PWM PERIOD

The PWM period is specified by writing to the PRx register. The PWM period can be calculated using Equation 13-1.

EQUATION 13-1:

$$\text{PWM period} = [(PRx) + 1] \cdot 4 \cdot \text{TOSC} \cdot (\text{TMRx prescale value})$$

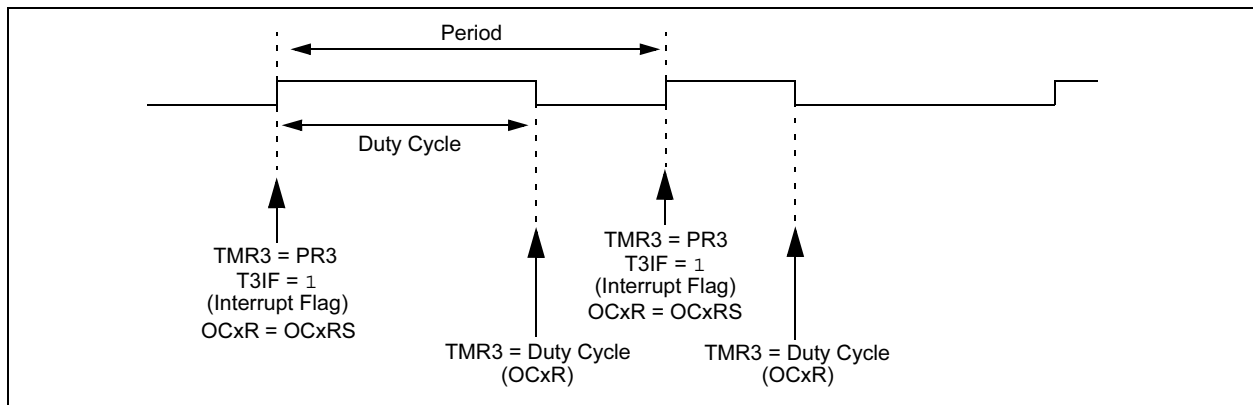
PWM frequency is defined as $1 / [\text{PWM period}]$.

When the selected TMRx is equal to its respective period register, PRx, the following four events occur on the next increment cycle:

- TMRx is cleared.
- The OCx pin is set.
 - Exception 1: If PWM duty cycle is 0x0000, the OCx pin will remain low.
 - Exception 2: If duty cycle is greater than PRx, the pin will remain high.
- The PWM duty cycle is latched from OCxRS into OCxR.
- The corresponding timer interrupt flag is set.

See Figure 13-2 for key PWM period comparisons. Timer3 is referred to in Figure 13-2 for clarity.

FIGURE 13-2: PWM OUTPUT TIMING



13.5 Output Compare Operation During CPU Sleep Mode

When the CPU enters Sleep mode, all internal clocks are stopped. Therefore, when the CPU enters the Sleep state, the output compare channel will drive the pin to the active state that was observed prior to entering the CPU Sleep state.

For example, if the pin was high when the CPU entered the Sleep state, the pin will remain high. Likewise, if the pin was low when the CPU entered the Sleep state, the pin will remain low. In either case, the output compare module will resume operation when the device wakes up.

13.6 Output Compare Operation During CPU Idle Mode

When the CPU enters the Idle mode, the output compare module can operate with full functionality.

The output compare channel will operate during the CPU Idle mode if the OCSIDL bit (OCxCON<13>) is at logic '0' and the selected time base (Timer2 or Timer3) is enabled and the TSIDL bit of the selected timer is set to logic '0'.

13.7 Output Compare Interrupts

The output compare channels have the ability to generate an interrupt on a compare match, for whichever Match mode has been selected.

For all modes except the PWM mode, when a compare event occurs, the respective interrupt flag (OCxIF) is asserted and an interrupt will be generated if enabled. The OCxIF bit is located in the corresponding IFS Status register and must be cleared in software. The interrupt is enabled via the respective compare interrupt enable (OCxIE) bit located in the corresponding IEC Control register.

For the PWM mode, when an event occurs, the respective timer interrupt flag (T2IF or T3IF) is asserted and an interrupt will be generated if enabled. The IF bit is located in the IFS0 Status register and must be cleared in software. The interrupt is enabled via the respective timer interrupt enable bit (T2IE or T3IE) located in the IEC0 Control register. The output compare interrupt flag is never set during the PWM mode of operation.

TABLE 13-1: OUTPUT COMPARE 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 | Reset State | |
|----------|-------|-------------------------------------|--------|--------|--------|--------|--------|-------|-------|-------|-------|-------|-------|--------|----------|----------|----------|---------------------|---------------------|
| OC1RS | 0180 | Output Compare 1 Secondary Register | | | | | | | | | | | | | | | | | 0000 0000 0000 0000 |
| OC1R | 0182 | Output Compare 1 Main Register | | | | | | | | | | | | | | | | | 0000 0000 0000 0000 |
| OC1CON | 0184 | — | — | OCSIDL | — | — | — | — | — | — | — | — | OCFLT | OCTSEL | OCM<2:0> | OCM<2:0> | OCM<2:0> | 0000 0000 0000 0000 | |
| OC2RS | 0186 | Output Compare 2 Secondary Register | | | | | | | | | | | | | | | | | 0000 0000 0000 0000 |
| OC2R | 0188 | Output Compare 2 Main Register | | | | | | | | | | | | | | | | | 0000 0000 0000 0000 |
| OC2CON | 018A | — | — | OCSIDL | — | — | — | — | — | — | — | — | OCFLT | OCTSE | OCM<2:0> | OCM<2:0> | OCM<2:0> | 0000 0000 0000 0000 | |
| OC3RS | 018C | Output Compare 3 Secondary Register | | | | | | | | | | | | | | | | | 0000 0000 0000 0000 |
| OC3R | 018E | Output Compare 3 Main Register | | | | | | | | | | | | | | | | | 0000 0000 0000 0000 |
| OC3CON | 0190 | — | — | OCSIDL | — | — | — | — | — | — | — | — | OCFLT | OCTSEL | OCM<2:0> | OCM<2:0> | OCM<2:0> | 0000 0000 0000 0000 | |
| OC4RS | 0192 | Output Compare 4 Secondary Register | | | | | | | | | | | | | | | | | 0000 0000 0000 0000 |
| OC4R | 0194 | Output Compare 4 Main Register | | | | | | | | | | | | | | | | | 0000 0000 0000 0000 |
| OC4CON | 0196 | — | — | OCSIDL | — | — | — | — | — | — | — | — | OCFLT | OCTSEL | OCM<2:0> | OCM<2:0> | OCM<2:0> | 0000 0000 0000 0000 | |
| OC5RS | 0198 | Output Compare 5 Secondary Register | | | | | | | | | | | | | | | | | 0000 0000 0000 0000 |
| OC5R | 019A | Output Compare 5 Main Register | | | | | | | | | | | | | | | | | 0000 0000 0000 0000 |
| OC5CON | 019C | — | — | OCSIDL | — | — | — | — | — | — | — | — | OCFLT | OCTSEL | OCM<2:0> | OCM<2:0> | OCM<2:0> | 0000 0000 0000 0000 | |
| OC6RS | 019E | Output Compare 6 Secondary Register | | | | | | | | | | | | | | | | | 0000 0000 0000 0000 |
| OC6R | 01A0 | Output Compare 6 Main Register | | | | | | | | | | | | | | | | | 0000 0000 0000 0000 |
| OC6CON | 01A2 | — | — | OCSIDL | — | — | — | — | — | — | — | — | OCFLT | OCTSEL | OCM<2:0> | OCM<2:0> | OCM<2:0> | 0000 0000 0000 0000 | |
| OC7RS | 01A4 | Output Compare 7 Secondary Register | | | | | | | | | | | | | | | | | 0000 0000 0000 0000 |
| OC7R | 01A6 | Output Compare 7 Main Register | | | | | | | | | | | | | | | | | 0000 0000 0000 0000 |
| OC7CON | 01A8 | — | — | OCSIDL | — | — | — | — | — | — | — | — | OCFLT | OCTSEL | OCM<2:0> | OCM<2:0> | OCM<2:0> | 0000 0000 0000 0000 | |
| OC8RS | 01AA | Output Compare 8 Secondary Register | | | | | | | | | | | | | | | | | 0000 0000 0000 0000 |
| OC8R | 01AC | Output Compare 8 Main Register | | | | | | | | | | | | | | | | | 0000 0000 0000 0000 |
| OC8CON | 01AE | — | — | OCSIDL | — | — | — | — | — | — | — | — | OCFLT | OCTSEL | OCM<2:0> | OCM<2:0> | OCM<2:0> | 0000 0000 0000 0000 | |

Legend: u = uninitialized bit

Note: Refer to *dsPIC30F Family Reference Manual (DS70046)* for descriptions of register bit fields.

14.0 SPI MODULE

Note: This data sheet summarizes features of this group of dsPIC30F devices and is not intended to be a complete reference source. For more information on the CPU, peripherals, register descriptions and general device functionality, refer to the *dsPIC30F Family Reference Manual* (DS70046).

The Serial Peripheral Interface (SPI) module is a synchronous serial interface. It is useful for communicating with other peripheral devices, such as EEPROMs, shift registers, display drivers and A/D converters, or other microcontrollers. It is compatible with Motorola's SPI™ and SIOP interfaces.

14.1 Operating Function Description

Each SPI module consists of a 16-bit shift register, SPIxSR (where x = 1 or 2), used for shifting data in and out, and a buffer register, SPIxBUF. A control register, SPIxCON, configures the module. Additionally, a status register, SPIxSTAT, indicates various status conditions.

The serial interface consists of 4 pins: SDIx (serial data input), SDOx (serial data output), SCKx (shift clock input or output), and SSx (active low slave select).

In Master mode operation, SCK is a clock output but in Slave mode, it is a clock input.

A series of eight (8) or sixteen (16) clock pulses shift out bits from the SPIxSR to SDOx pin and simultaneously shift in data from SDIx pin. An interrupt is generated when the transfer is complete and the corresponding interrupt flag bit (SPI1IF or SPI2IF) is set. This interrupt can be disabled through an interrupt enable bit (SPI1IE or SPI2IE).

The receive operation is double-buffered. When a complete byte is received, it is transferred from SPIxSR to SPIxBUF.

If the receive buffer is full when new data is being transferred from SPIxSR to SPIxBUF, the module will set the SPIROV bit indicating an overflow condition. The transfer of the data from SPIxSR to SPIxBUF will not be completed and the new data will be lost. The module will not respond to SCL transitions while SPIROV is '1', effectively disabling the module until SPIxBUF is read by user software.

Transmit writes are also double-buffered. The user writes to SPIxBUF. When the master or slave transfer is completed, the contents of the shift register (SPIxSR) are moved to the receive buffer. If any transmit data has been written to the buffer register, the contents of the transmit buffer are moved to SPIxSR. The received data is thus placed in SPIxBUF and the transmit data in SPIxSR is ready for the next transfer.

Note: Both the transmit buffer (SPIxTXB) and the receive buffer (SPIxRXB) are mapped to the same register address, SPIxBUF.

In Master mode, the clock is generated by prescaling the system clock. Data is transmitted as soon as a value is written to SPIxBUF. The interrupt is generated at the middle of the transfer of the last bit.

In Slave mode, data is transmitted and received as external clock pulses appear on SCK. Again, the interrupt is generated when the last bit is latched. If SSx control is enabled, then transmission and reception are enabled only when SSx = low. The SDOx output will be disabled in SSx mode with SSx high.

The clock provided to the module is (Fosc/4). This clock is then prescaled by the primary (PPRE<1:0>) and the secondary (SPRE<2:0>) prescale factors. The CKE bit determines whether transmit occurs on transition from active clock state to Idle clock state, or vice versa. The CKP bit selects the Idle state (high or low) for the clock.

14.1.1 WORD AND BYTE COMMUNICATION

A control bit, MODE16 (SPIxCON<10>), allows the module to communicate in either 16-bit or 8-bit mode. 16-bit operation is identical to 8-bit operation except that the number of bits transmitted is 16 instead of 8.

The user software must disable the module prior to changing the MODE16 bit. The SPI module is reset when the MODE16 bit is changed by the user.

A basic difference between 8-bit and 16-bit operation is that the data is transmitted out of bit 7 of the SPIxSR for 8-bit operation, and data is transmitted out of bit 15 of the SPIxSR for 16-bit operation. In both modes, data is shifted into bit 0 of the SPIxSR.

14.1.2 SDOx DISABLE

A control bit, DISSDO, is provided to the SPIxCON register to allow the SDOx output to be disabled. This will allow the SPI module to be connected in an input only configuration. SDO can also be used for general purpose I/O.

14.2 Framed SPI Support

The module supports a basic framed SPI protocol in Master or Slave mode. The control bit FRMEN enables framed SPI support and causes the SSx pin to perform the frame synchronization pulse (FSYNC) function. The control bit SPIFSD determines whether the SSx pin is an input or an output (i.e., whether the module receives or generates the frame synchronization pulse). The frame pulse is an active high pulse for a single SPI clock cycle. When frame synchronization is enabled, the data transmission starts only on the subsequent transmit edge of the SPI clock.

dsPIC30F

FIGURE 14-1: SPI BLOCK DIAGRAM

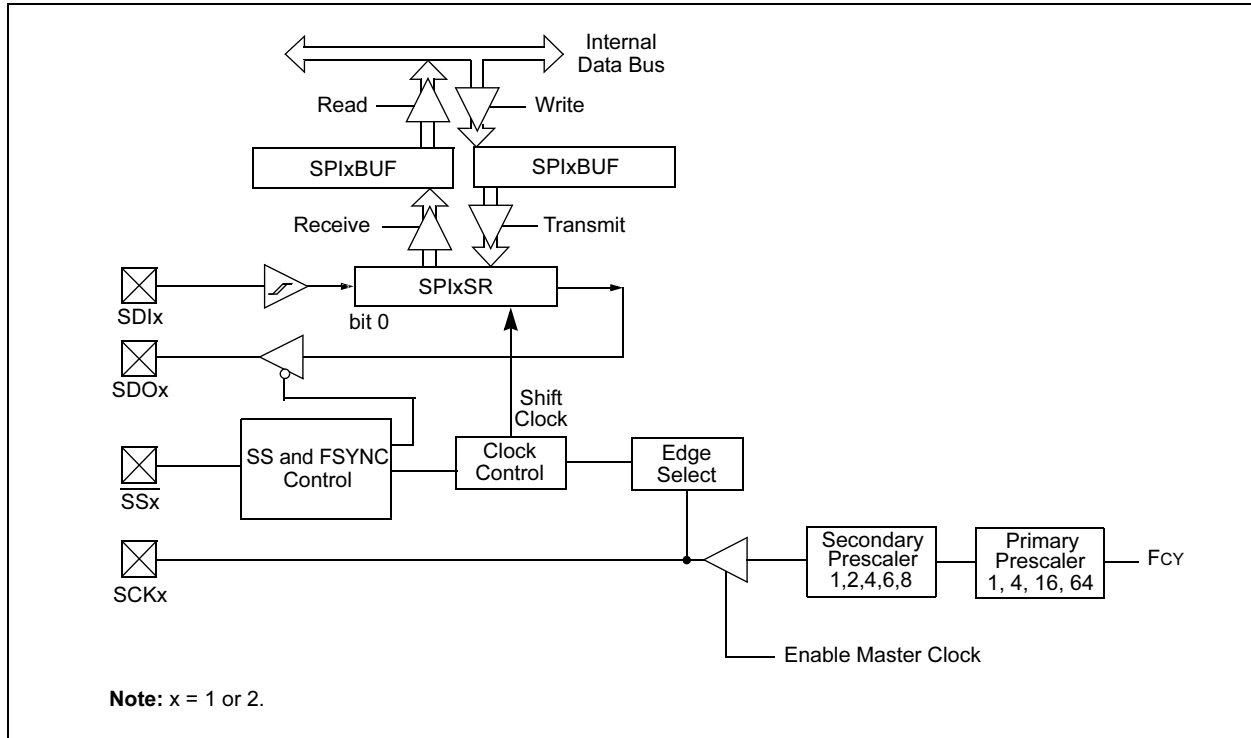
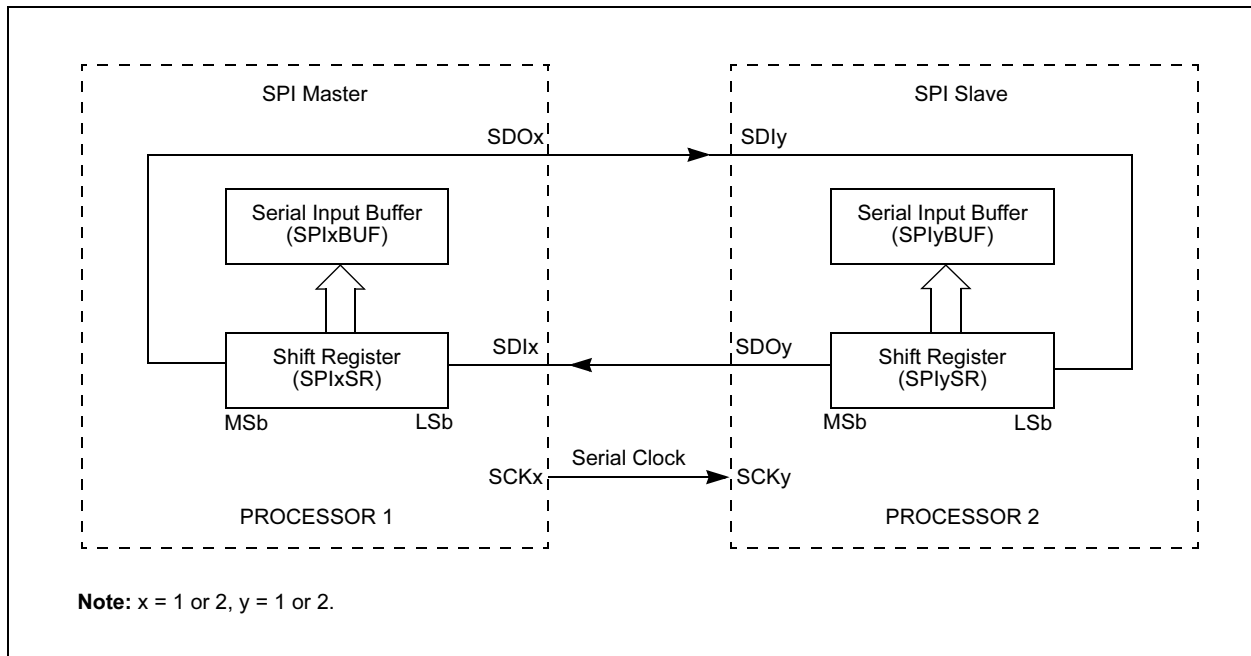


FIGURE 14-2: SPI MASTER/SLAVE CONNECTION



14.3 Slave Select Synchronization

The \overline{SSx} pin allows a Synchronous Slave mode. The SPI must be configured in SPI Slave mode with \overline{SSx} pin control enabled ($SSEN = 1$). When the \overline{SSx} pin is low, transmission and reception are enabled and the SDOx pin is driven. When \overline{SSx} pin goes high, the SDOx pin is no longer driven. Also, the SPI module is re-synchronized, and all counters/control circuitry are reset. Therefore, when the \overline{SSx} pin is asserted low again, transmission/reception will begin at the MS bit even if \overline{SSx} had been de-asserted in the middle of a transmit/receive.

14.4 SPI Operation During CPU Sleep Mode

During Sleep mode, the SPI module is shutdown. If the CPU enters Sleep mode while an SPI transaction is in progress, then the transmission and reception is aborted.

The transmitter and receiver will stop in Sleep mode. However, register contents are not affected by entering or exiting Sleep mode.

14.5 SPI Operation During CPU Idle Mode

When the device enters Idle mode, all clock sources remain functional. The SPISIDL bit ($SPIxSTAT<13>$) selects if the SPI module will stop or continue on Idle. If $SPISIDL = 0$, the module will continue to operate when the CPU enters Idle mode. If $SPISIDL = 1$, the module will stop when the CPU enters Idle mode.

TABLE 14-1: SPI1 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 | Reset State |
|----------|-------|-----------------------------|--------|--------|--------|--------|--------|-------|-------|-------|--------|-------|-------|-------|-------|---------|---------|---------------------|
| SPI1STAT | 0220 | SPIEN | — | SPIIDL | — | — | — | — | — | — | SPIROV | MSTEN | SPRE2 | SPRE1 | SPRE0 | SPI1TBF | SPI0RBF | 0000 0000 0000 0000 |
| SPI1CON | 0222 | — | FRMEN | SPIFSD | — | DISSDO | MODE16 | SMP | CKE | SSEN | CKP | — | — | — | — | PPRE1 | PPRE0 | 0000 0000 0000 0000 |
| SPI1BUF | 0224 | Transmit and Receive Buffer | | | | | | | | | | | | | | | | 0000 0000 0000 0000 |

Legend: u = uninitialized bit

TABLE 14-2: SPI2 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 | Reset State |
|----------|-------|-----------------------------|--------|--------|--------|--------|--------|-------|-------|-------|--------|-------|-------|-------|-------|---------|---------|---------------------|
| SPI2STAT | 0226 | SPIEN | — | SPIIDL | — | — | — | — | — | — | SPIROV | — | — | — | — | SPI1TBF | SPI0RBF | 0000 0000 0000 0000 |
| SPI2CON | 0228 | — | FRMEN | SPIFSD | — | DISSDO | MODE16 | SMP | CKE | SSEN | CKP | MSTEN | SPRE2 | SPRE1 | SPRE0 | PPRE1 | PPRE0 | 0000 0000 0000 0000 |
| SPI2BUF | 022A | Transmit and Receive Buffer | | | | | | | | | | | | | | | | 0000 0000 0000 0000 |

Legend: u = uninitialized bit

Note: Refer to *dsPIC30F Family Reference Manual (DS70046)* for descriptions of register bit fields.

15.0 I²C MODULE

Note: This data sheet summarizes features of this group of dsPIC30F devices and is not intended to be a complete reference source. For more information on the CPU, peripherals, register descriptions and general device functionality, refer to the *dsPIC30F Family Reference Manual* (DS70046).

The Inter-Integrated Circuit (I²C™) module provides complete hardware support for both Slave and Multi-Master modes of the I²C serial communication standard, with a 16-bit interface.

This module offers the following key features:

- I²C interface supporting both master and slave operation.
- I²C Slave mode supports 7 and 10-bit address.
- I²C Master mode supports 7 and 10-bit address.
- I²C port allows bidirectional transfers between master and slaves.

- Serial clock synchronization for I²C port can be used as a handshake mechanism to suspend and resume serial transfer (SCLREL control).
- I²C supports multi-master operation; detects bus collision and will arbitrate accordingly.

15.1 Operating Function Description

The hardware fully implements all the master and slave functions of the I²C Standard and Fast mode specifications, as well as 7 and 10-bit addressing.

Thus, the I²C module can operate either as a slave or a master on an I²C bus.

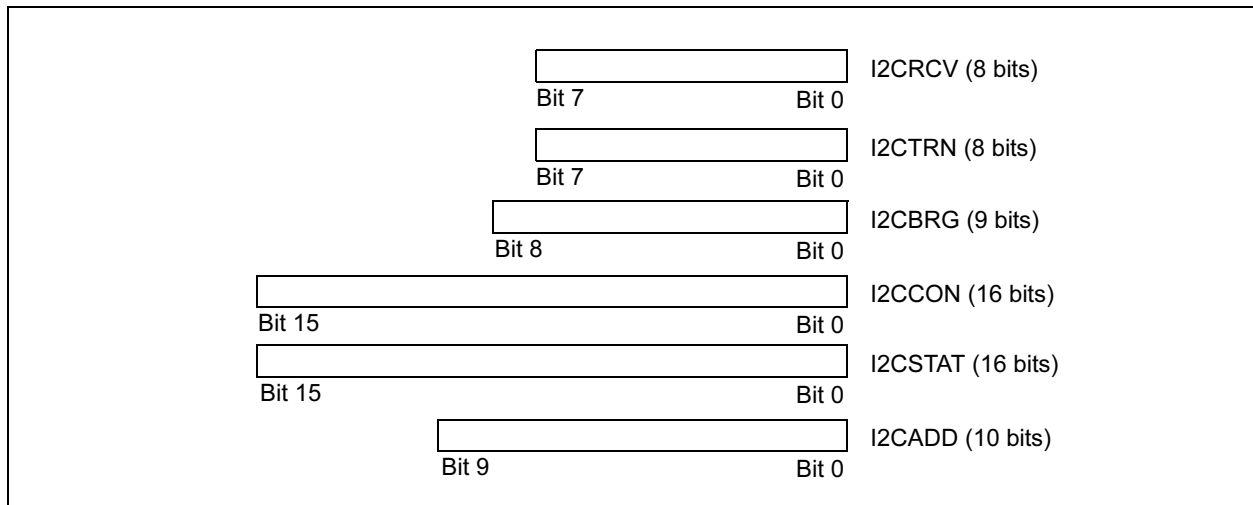
15.1.1 VARIOUS I²C MODES

The following types of I²C operation are supported:

- I²C slave operation with 7-bit address
- I²C slave operation with 10-bit address
- I²C master operation with 7 or 10-bit address

See the I²C programmer's model in Figure 15-1.

FIGURE 15-1: PROGRAMMER'S MODEL



15.1.2 PIN CONFIGURATION IN I²C MODE

I²C has a 2-pin interface: the SCL pin is clock and the SDA pin is data.

15.1.3 I²C REGISTERS

I2CCON and I2CSTAT are control and status registers, respectively. The I2CCON register is readable and writable. The lower 6 bits of I2CSTAT are read only. The remaining bits of the I2CSTAT are read/write.

I2CRSR is the shift register used for shifting data, whereas I2CRCV is the buffer register to which data bytes are written, or from which data bytes are read. I2CRCV is the receive buffer as shown in Figure 15-1. I2CTRN is the transmit register to which bytes are written during a transmit operation, as shown in Figure 15-2.

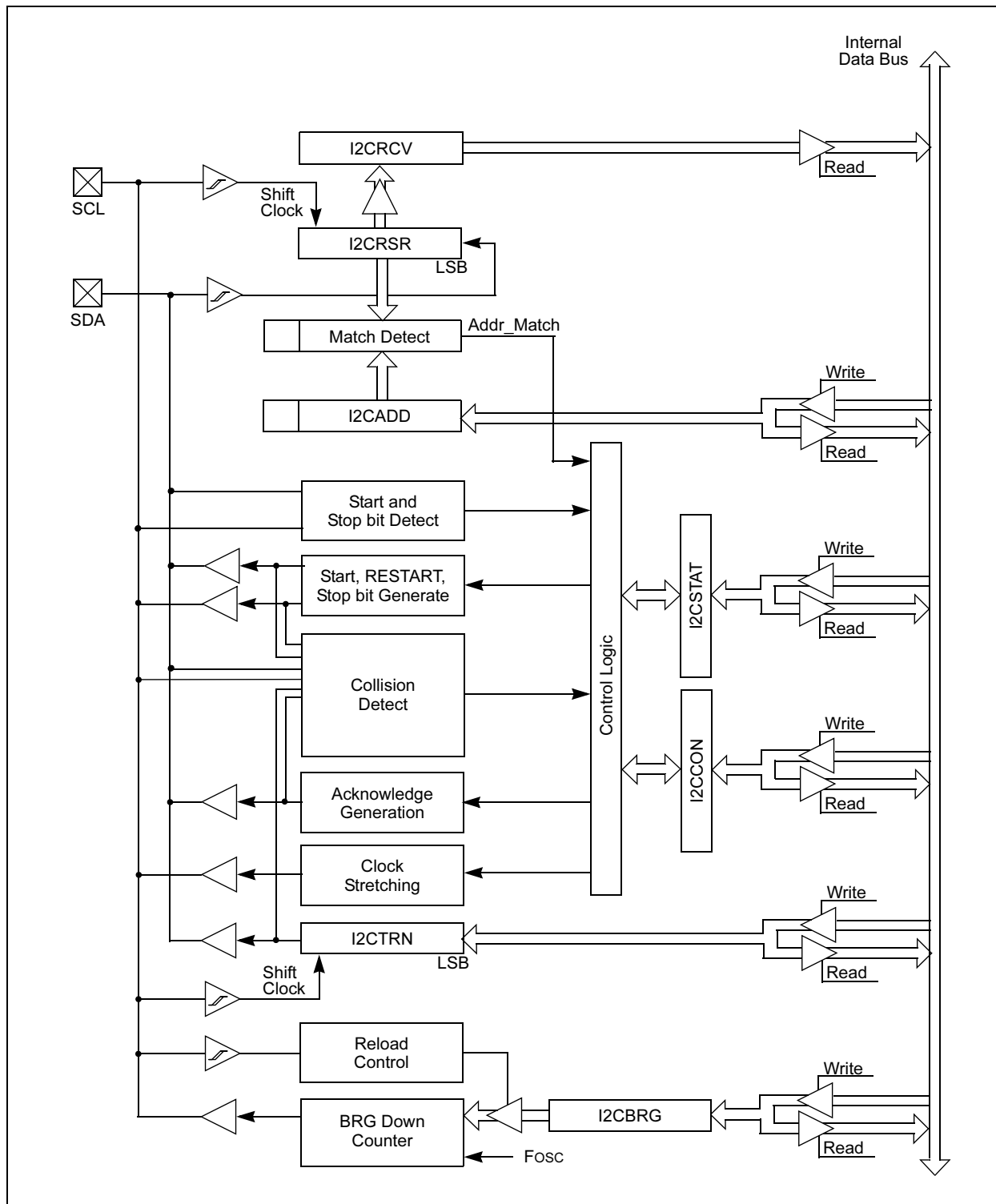
The I2CADD register holds the slave address. A status bit, ADD10, indicates 10-bit Address mode. The I2CBRG acts as the baud rate generator reload value.

In receive operations, I2CRSR and I2CRCV together form a double-buffered receiver. When I2CRSR receives a complete byte, it is transferred to I2CRCV and an interrupt pulse is generated. During transmission, the I2CTRN is not double-buffered.

Note: Following a RESTART condition in 10-bit mode, the user only needs to match the first 7-bit address.

dsPIC30F

FIGURE 15-2: I²C BLOCK DIAGRAM



15.2 I²C Module Addresses

The I2CADD register contains the Slave mode addresses. The register is a 10-bit register.

If the A10M bit (I2CCON<10>) is '0', the address is interpreted by the module as a 7-bit address. When an address is received, it is compared to the 7 LS bits of the I2CADD register.

If the A10M bit is '1', the address is assumed to be a 10-bit address. When an address is received, it will be compared with the binary value '11110 A9 A8' (where A9 and A8 are two Most Significant bits of I2CADD). If that value matches, the next address will be compared with the Least Significant 8 bits of I2CADD, as specified in the 10-bit addressing protocol.

7-bit I²C Slave Addresses supported by dsPIC30F:

| | |
|-----------|---------------------------------------|
| 0x00 | General call address or start byte |
| 0x01-0x03 | Reserved |
| 0x04-0x77 | Valid 7-bit addresses |
| 0x78-0x7b | Valid 10-bit addresses (lower 7 bits) |
| 0x7c-0x7f | Reserved |

15.3 I²C 7-bit Slave Mode Operation

Once enabled (I2CEN = 1), the slave module will wait for a Start bit to occur (i.e., the I²C module is 'Idle'). Following the detection of a Start bit, 8 bits are shifted into I2CRSR and the address is compared against I2CADD. In 7-bit mode (A10M = 0), bits I2CADD<6:0> are compared against I2CRSR<7:1> and I2CRSR<0> is the R_W bit. All incoming bits are sampled on the rising edge of SCL.

If an address match occurs, an Acknowledgement will be sent, and the slave event interrupt flag (SI2CIF) is set on the falling edge of the ninth (ACK) bit. The address match does not affect the contents of the I2CRCV buffer or the RBF bit.

15.3.1 SLAVE TRANSMISSION

If the R_W bit received is a '1', then the serial port will go into Transmit mode. It will send $\overline{\text{ACK}}$ on the ninth bit and then hold SCL to '0' until the CPU responds by writing to I2CTRN. SCL is released by setting the SCLREL bit, and 8 bits of data are shifted out. Data bits are shifted out on the falling edge of SCL, such that SDA is valid during SCL high. The interrupt pulse is sent on the falling edge of the ninth clock pulse, regardless of the status of the ACK received from the master.

15.3.2 SLAVE RECEPTION

If the R_W bit received is a '0' during an address match, then Receive mode is initiated. Incoming bits are sampled on the rising edge of SCL. After 8 bits are

received, if I2CRCV is not full or I2COV is not set, I2CRSR is transferred to I2CRCV. $\overline{\text{ACK}}$ is sent on the ninth clock.

If the RBF flag is set, indicating that I2CRCV is still holding data from a previous operation (RBF = 1), then $\overline{\text{ACK}}$ is not sent; however, the interrupt pulse is generated. In the case of an overflow, the contents of the I2CRSR are not loaded into the I2CRCV.

Note: The I2CRCV will be loaded if the I2COV bit = 1 and the RBF flag = 0. In this case, a read of the I2CRCV was performed but the user did not clear the state of the I2COV bit before the next receive occurred. The Acknowledgement is not sent ($\overline{\text{ACK}}$ = 1) and the I2CRCV is updated.

15.4 I²C 10-bit Slave Mode Operation

In 10-bit mode, the basic receive and transmit operations are the same as in the 7-bit mode. However, the criteria for address match is more complex.

The I²C specification dictates that a slave must be addressed for a write operation with two address bytes following a Start bit.

The A10M bit is a control bit that signifies that the address in I2CADD is a 10-bit address rather than a 7-bit address. The address detection protocol for the first byte of a message address is identical for 7-bit and 10-bit messages, but the bits being compared are different.

I2CADD holds the entire 10-bit address. Upon receiving an address following a Start bit, I2CRSR <7:3> is compared against a literal '11110' (the default 10-bit address) and I2CRSR<2:1> are compared against I2CADD<9:8>. If a match occurs and if R_W = 0, the interrupt pulse is sent. The ADD10 bit will be cleared to indicate a partial address match. If a match fails or R_W = 1, the ADD10 bit is cleared and the module returns to the Idle state.

The low byte of the address is then received and compared with I2CADD<7:0>. If an address match occurs, the interrupt pulse is generated and the ADD10 bit is set, indicating a complete 10-bit address match. If an address match did not occur, the ADD10 bit is cleared and the module returns to the Idle state.

15.4.1 10-BIT MODE SLAVE TRANSMISSION

Once a slave is addressed in this fashion with the full 10-bit address (we will refer to this state as "PRIOR_ADDR_MATCH"), the master can begin sending data bytes for a slave reception operation.

15.4.2 10-BIT MODE SLAVE RECEPTION

Once addressed, the master can generate a Repeated Start, reset the high byte of the address and set the R_W bit without generating a Stop bit, thus initiating a slave transmit operation.

15.5 Automatic Clock Stretch

In the Slave modes, the module can synchronize buffer reads and write to the master device by clock stretching.

15.5.1 TRANSMIT CLOCK STRETCHING

Both 10-bit and 7-bit Transmit modes implement clock stretching by asserting the SCLREL bit after the falling edge of the ninth clock, if the TBF bit is cleared, indicating the buffer is empty.

In Slave Transmit modes, clock stretching is always performed irrespective of the STREN bit.

Clock synchronization takes place following the ninth clock of the transmit sequence. If the device samples an $\overline{\text{ACK}}$ on the falling edge of the ninth clock and if the TBF bit is still clear, then the SCLREL bit is automatically cleared. The SCLREL being cleared to '0' will assert the SCL line low. The user's ISR must set the SCLREL bit before transmission is allowed to continue. By holding the SCL line low, the user has time to service the ISR and load the contents of the I2CTRN before the master device can initiate another transmit sequence.

Note 1: If the user loads the contents of I2CTRN, setting the TBF bit before the falling edge of the ninth clock, the SCLREL bit will not be cleared and clock stretching will not occur.

2: The SCLREL bit can be set in software, regardless of the state of the TBF bit.

15.5.2 RECEIVE CLOCK STRETCHING

The STREN bit in the I2CCON register can be used to enable clock stretching in Slave Receive mode. When the STREN bit is set, the SCL pin will be held low at the end of each data receive sequence.

15.5.3 CLOCK STRETCHING DURING 7-BIT ADDRESSING (STREN = 1)

When the STREN bit is set in Slave Receive mode, the SCL line is held low when the buffer register is full. The method for stretching the SCL output is the same for both 7 and 10-bit Addressing modes.

Clock stretching takes place following the ninth clock of the receive sequence. On the falling edge of the ninth clock at the end of the ACK sequence, if the RBF bit is set, the SCLREL bit is automatically cleared, forcing the SCL output to be held low. The user's ISR must set the SCLREL bit before reception is allowed to continue. By holding the SCL line low, the user has time to ser-

vice the ISR and read the contents of the I2CRCV before the master device can initiate another receive sequence. This will prevent buffer overruns from occurring.

Note 1: If the user reads the contents of the I2CRCV, clearing the RBF bit before the falling edge of the ninth clock, the SCLREL bit will not be cleared and clock stretching will not occur.

2: The SCLREL bit can be set in software regardless of the state of the RBF bit. The user should be careful to clear the RBF bit in the ISR before the next receive sequence in order to prevent an overflow condition.

15.5.4 CLOCK STRETCHING DURING 10-BIT ADDRESSING (STREN = 1)

Clock stretching takes place automatically during the addressing sequence. Because this module has a register for the entire address, it is not necessary for the protocol to wait for the address to be updated.

After the address phase is complete, clock stretching will occur on each data receive or transmit sequence as was described earlier.

15.6 Software Controlled Clock Stretching (STREN = 1)

When the STREN bit is '1', the SCLREL bit may be cleared by software to allow software to control the clock stretching. The logic will synchronize writes to the SCLREL bit with the SCL clock. Clearing the SCLREL bit will not assert the SCL output until the module detects a falling edge on the SCL output and SCL is sampled low. If the SCLREL bit is cleared by the user while the SCL line has been sampled low, the SCL output will be asserted (held low). The SCL output will remain low until the SCLREL bit is set, and all other devices on the I²C bus have de-asserted SCL. This ensures that a write to the SCLREL bit will not violate the minimum high time requirement for SCL.

If the STREN bit is '0', a software write to the SCLREL bit will be disregarded and have no effect on the SCLREL bit.

15.7 Interrupts

The I²C module generates two interrupt flags, MI2CIF (I²C Master Interrupt Flag) and SI2CIF (I²C Slave Interrupt Flag). The MI2CIF interrupt flag is activated on completion of a master message event. The SI2CIF interrupt flag is activated on detection of a message directed to the slave.

15.8 Slope Control

The I²C standard requires slope control on the SDA and SCL signals for Fast mode (400 kHz). The control bit, DISSLW, enables the user to disable slew rate control if desired. It is necessary to disable the slew rate control for 1 MHz mode.

15.9 IPMI Support

The control bit, IPMIEN, enables the module to support Intelligent Peripheral Management Interface (IPMI). When this bit is set, the module accepts and acts upon all addresses.

15.10 General Call Address Support

The general call address can address all devices. When this address is used, all devices should, in theory, respond with an Acknowledgement.

The general call address is one of eight addresses reserved for specific purposes by the I²C protocol. It consists of all '0's with R_W = 0.

The general call address is recognized when the General Call Enable (GCEN) bit is set (I2CCON<15> = 1). Following a Start bit detection, 8 bits are shifted into I2CRSR and the address is compared with I2CADD, and is also compared with the general call address which is fixed in hardware.

If a general call address match occurs, the I2CRSR is transferred to the I2CRCV after the eighth clock, the RBF flag is set and on the falling edge of the ninth bit (ACK bit), the master event interrupt flag (MI2CIF) is set.

When the interrupt is serviced, the source for the interrupt can be checked by reading the contents of the I2CRCV to determine if the address was device specific or a general call address.

15.11 I²C Master Support

As a master device, six operations are supported:

- Assert a Start condition on SDA and SCL.
- Assert a RESTART condition on SDA and SCL.
- Write to the I2CTRN register initiating transmission of data/address.
- Generate a Stop condition on SDA and SCL.
- Configure the I²C port to receive data.
- Generate an ACK condition at the end of a received byte of data.

15.12 I²C Master Operation

The master device generates all of the serial clock pulses and the Start and Stop conditions. A transfer is ended with a Stop condition or with a Repeated Start condition. Since the Repeated Start condition is also the beginning of the next serial transfer, the I²C bus will not be released.

In Master Transmitter mode, serial data is output through SDA, while SCL outputs the serial clock. The first byte transmitted contains the slave address of the receiving device (7 bits) and the data direction bit. In this case, the data direction bit (R_W) is logic '0'. Serial data is transmitted 8 bits at a time. After each byte is transmitted, an ACK bit is received. Start and Stop conditions are output to indicate the beginning and the end of a serial transfer.

In Master Receive mode, the first byte transmitted contains the slave address of the transmitting device (7 bits) and the data direction bit. In this case, the data direction bit (R_W) is logic '1'. Thus, the first byte transmitted is a 7-bit slave address, followed by a '1' to indicate receive bit. Serial data is received via SDA while SCL outputs the serial clock. Serial data is received 8 bits at a time. After each byte is received, an ACK bit is transmitted. Start and Stop conditions indicate the beginning and end of transmission.

15.12.1 I²C MASTER TRANSMISSION

Transmission of a data byte, a 7-bit address, or the second half of a 10-bit address is accomplished by simply writing a value to I2CTRN register. The user should only write to I2CTRN when the module is in a WAIT state. This action will set the Buffer Full Flag (TBF) and allow the baud rate generator to begin counting and start the next transmission. Each bit of address/data will be shifted out onto the SDA pin after the falling edge of SCL is asserted. The Transmit Status Flag, TRSTAT (I2CSTAT<14>), indicates that a master transmit is in progress.

15.12.2 I²C MASTER RECEPTION

Master mode reception is enabled by programming the Receive Enable bit, RCEN (I2CCON<11>). The I²C module must be Idle before the RCEN bit is set, otherwise the RCEN bit will be disregarded. The baud rate generator begins counting and on each rollover, the state of the SCL pin ACK and data are shifted into the I2CRSR on the rising edge of each clock.

15.12.3 BAUD RATE GENERATOR

In I²C Master mode, the reload value for the BRG is located in the I2CBRG register. When the BRG is loaded with this value, the BRG counts down to '0' and stops until another reload has taken place. If clock arbitration is taking place, for instance, the BRG is reloaded when the SCL pin is sampled high.

As per the I²C standard, F_{SCL} may be 100 kHz or 400 kHz. However, the user can specify any baud rate up to 1 MHz. I2CBRG values of '0' or '1' are illegal.

EQUATION 15-1: SERIAL CLOCK RATE

$$I2CBRG = \left(\frac{FCY}{F_{SCL}} - \frac{FCY}{1,111,111} \right) - 1$$

15.12.4 CLOCK ARBITRATION

Clock arbitration occurs when the master de-asserts the SCL pin (SCL allowed to float high) during any receive, transmit, or RESTART/Stop condition. When the SCL pin is allowed to float high, the baud rate generator (BRG) is suspended from counting until the SCL pin is actually sampled high. When the SCL pin is sampled high, the baud rate generator is reloaded with the contents of I2CBRG and begins counting. This ensures that the SCL high time will always be at least one BRG rollover count in the event that the clock is held low by an external device.

15.12.5 MULTI-MASTER COMMUNICATION, BUS COLLISION, AND BUS ARBITRATION

Multi-master operation support is achieved by bus arbitration. When the master outputs address/data bits onto the SDA pin, arbitration takes place when the master outputs a '1' on SDA by letting SDA float high while another master asserts a '0'. When the SCL pin floats high, data should be stable. If the expected data on SDA is a '1' and the data sampled on the SDA pin = 0, then a bus collision has taken place. The master will set the MI2CIF pulse and reset the master portion of the I²C port to its Idle state.

If a transmit was in progress when the bus collision occurred, the transmission is halted, the TBF flag is cleared, the SDA and SCL lines are de-asserted and a value can now be written to I2CTRN. When the user services the I²C master event Interrupt Service Routine, if the I²C bus is free (i.e., the P bit is set), the user can resume communication by asserting a Start condition.

If a Start, RESTART, Stop or Acknowledge condition was in progress when the bus collision occurred, the condition is aborted, the SDA and SCL lines are de-asserted, and the respective control bits in the I2CCON register are cleared to '0'. When the user services the bus collision Interrupt Service Routine, and if the I²C bus is free, the user can resume communication by asserting a Start condition.

The master will continue to monitor the SDA and SCL pins, and if a Stop condition occurs, the MI2CIF bit will be set.

A write to the I2CTRN will start the transmission of data at the first data bit regardless of where the transmitter left off when bus collision occurred.

In a multi-master environment, the interrupt generation on the detection of Start and Stop conditions allows the determination of when the bus is free. Control of the I²C bus can be taken when the P bit is set in the I2CSTAT register, or the bus is Idle and the S and P bits are cleared.

15.13 I²C Module Operation During CPU Sleep and Idle Modes

15.13.1 I²C OPERATION DURING CPU SLEEP MODE

When the device enters Sleep mode, all clock sources to the module are shutdown and stay at logic '0'. If Sleep occurs in the middle of a transmission and the state machine is partially into a transmission as the clocks stop, then the transmission is aborted. Similarly, if Sleep occurs in the middle of a reception, then the reception is aborted.

15.13.2 I²C OPERATION DURING CPU IDLE MODE

For the I²C, the I2CSIDL bit selects if the module will stop on Idle or continue on Idle. If I2CSIDL = 0, the module will continue operation on assertion of the Idle mode. If I2CSIDL = 1, the module will stop on Idle.

TABLE 15-1: I²C 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 | Reset State |
|----------|-------|---------|--------|---------|--------|--------|--------|--------|-------|-------|-------|---------------------|-------------------|-------|-------|-------|---------------------|---------------------|
| I2CRCV | 0200 | — | — | — | — | — | — | — | — | — | — | — | Receive Register | | | | | 0000 0000 0000 0000 |
| I2CTRN | 0202 | — | — | — | — | — | — | — | — | — | — | — | Transmit Register | | | | | 0000 0000 1111 1111 |
| I2CBRG | 0204 | — | — | — | — | — | — | — | — | — | — | Baud Rate Generator | | | | | 0000 0000 0000 0000 | |
| I2CCON | 0206 | I2CEN | — | I2CSIDL | SCLREL | IPMIEN | A10M | DISSLW | SMEN | GCEN | STREN | ACKDT | ACKEN | RCEN | PEN | RSEN | SEN | 0001 0000 0000 0000 |
| I2CSTAT | 0208 | ACKSTAT | TRSTAT | — | — | — | BCL | GCSTAT | ADD10 | IWCOL | I2COV | D_A | P | S | R_W | RBF | TBF | 0000 0000 0000 0000 |
| I2CADD | 020A | — | — | — | — | — | — | — | — | — | — | Address Register | | | | | 0000 0000 0000 0000 | |

Legend: u = uninitialized bit

Note: Refer to *dsPIC30F Family Reference Manual (DS70046)* for descriptions of register bit fields.

dsPIC30F

NOTES:

16.0 UNIVERSAL ASYNCHRONOUS RECEIVER TRANSMITTER (UART) MODULE

Note: This data sheet summarizes features of this group of dsPIC30F devices and is not intended to be a complete reference source. For more information on the CPU, peripherals, register descriptions and general device functionality, refer to the *dsPIC30F Family Reference Manual (DS70046)*.

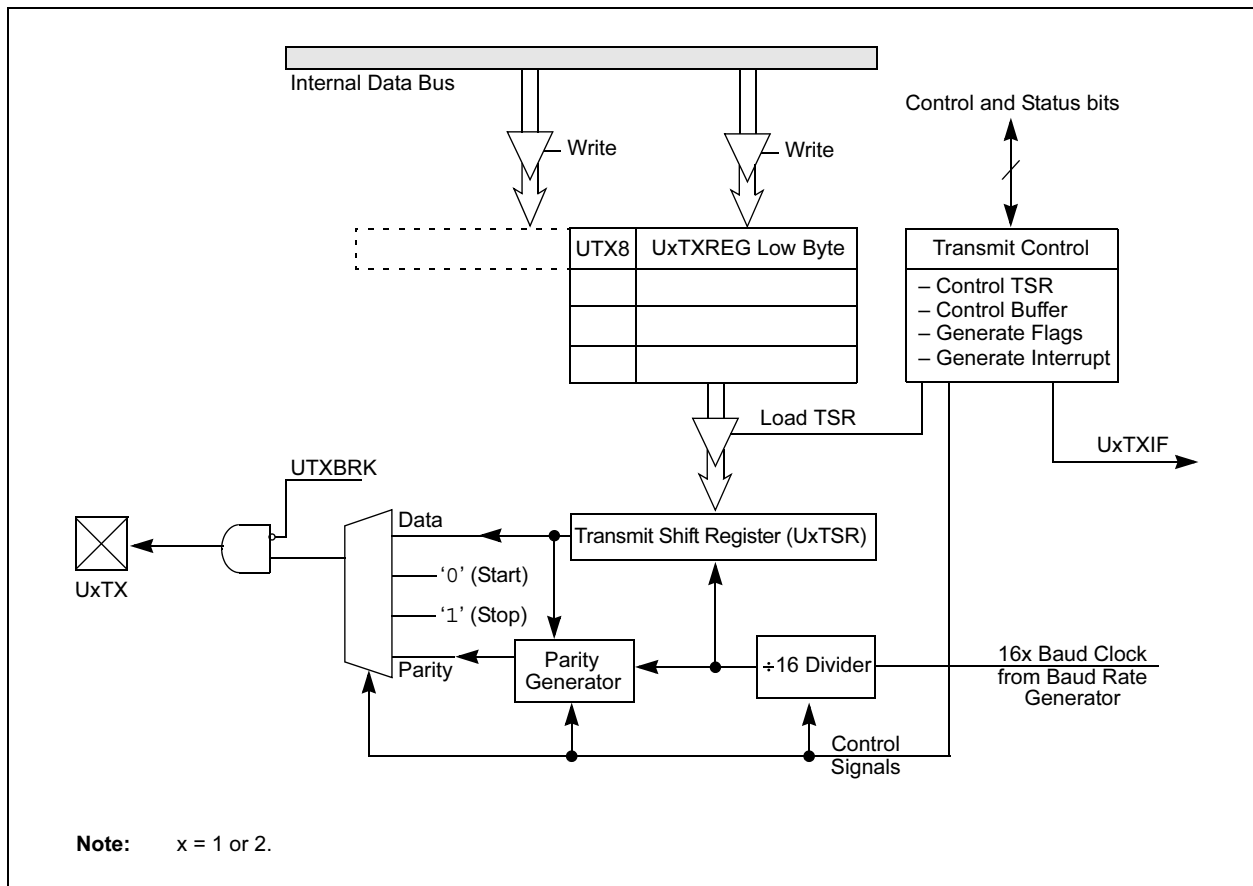
This section describes the Universal Asynchronous Receiver/Transmitter Communications module.

16.1 UART Module Overview

The key features of the UART module are:

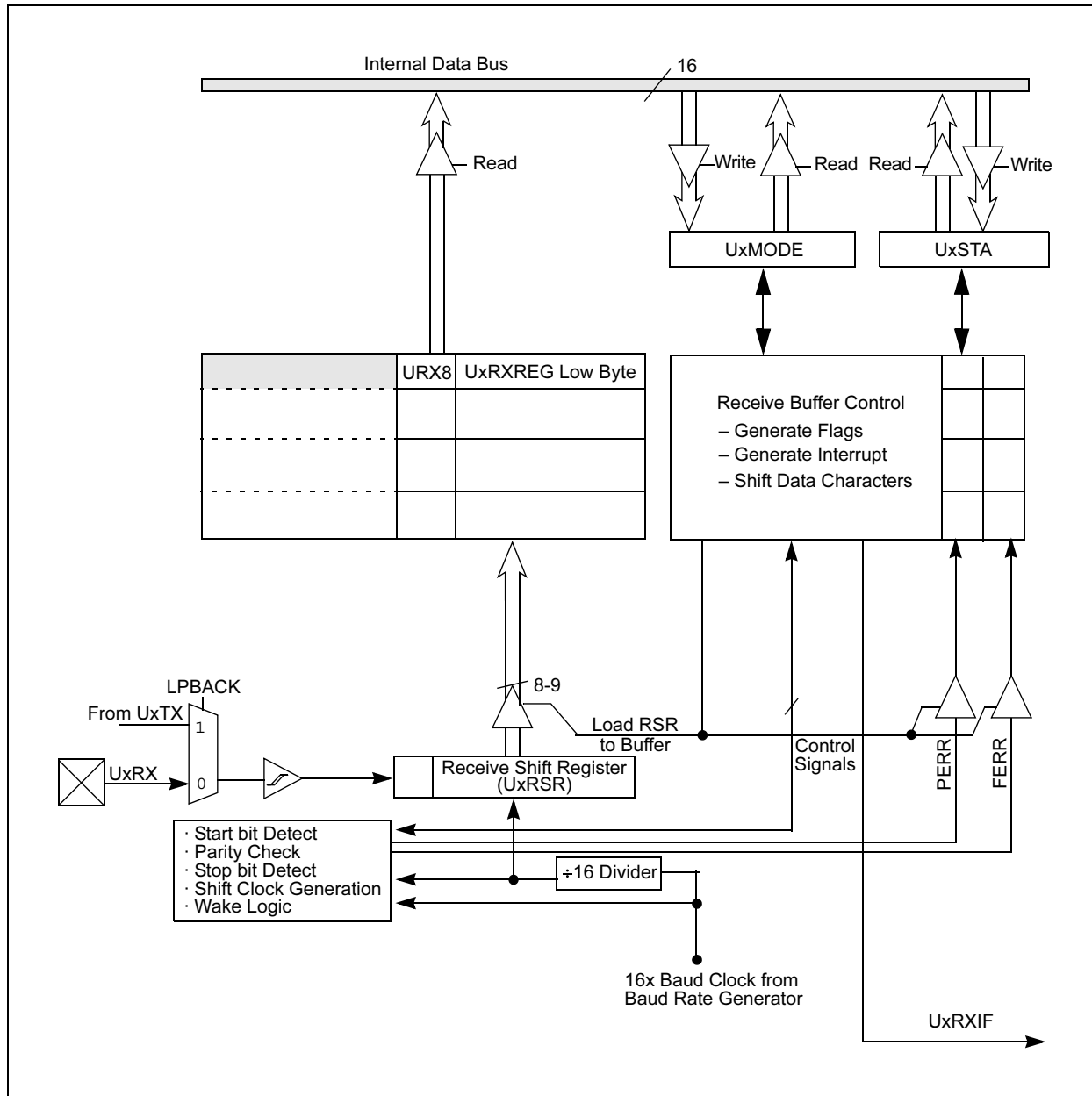
- Full-duplex, 8 or 9-bit data communication
- Even, odd or no parity options (for 8-bit data)
- One or two Stop bits
- Fully integrated baud rate generator with 16-bit prescaler
- Baud rates range from 38 bps to 1.875 Mbps at a 30 MHz instruction rate
- 4-word deep transmit data buffer
- 4-word deep receive data buffer
- Parity, framing and buffer overrun error detection
- Support for interrupt only on address detect (9th bit = 1)
- Separate transmit and receive interrupts
- Loopback mode for diagnostic support

FIGURE 16-1: UART TRANSMITTER BLOCK DIAGRAM



dsPIC30F

FIGURE 16-2: UART RECEIVER BLOCK DIAGRAM



16.2 Enabling and Setting Up UART

16.2.1 ENABLING THE UART

The UART module is enabled by setting the UARTEN bit in the UxMODE register (where x = 1 or 2). Once enabled, the UxTX and UxRX pins are configured as an output and an input respectively, overriding the TRIS and LATCH register bit settings for the corresponding I/O port pins. The UxTX pin is at logic '1' when no transmission is taking place.

16.2.2 DISABLING THE UART

The UART module is disabled by clearing the UARTEN bit in the UxMODE register. This is the default state after any Reset. If the UART is disabled, all I/O pins operate as port pins under the control of the latch and TRIS bits of the corresponding port pins.

Disabling the UART module resets the buffers to empty states. Any data characters in the buffers are lost and the baud rate counter is reset.

All error and status flags associated with the UART module are reset when the module is disabled. The URXDA, OERR, FERR, PERR, UTXEN, UTXBRK and UTXBF bits are cleared, whereas RIDLE and TRMT are set. Other control bits, including ADDEN, URXISEL<1:0>, UTXISEL, as well as the UxMODE and UxBRG registers, are not affected.

Clearing the UARTEN bit while the UART is active will abort all pending transmissions and receptions and reset the module as defined above. Re-enabling the UART will restart the UART in the same configuration.

16.2.3 ALTERNATE I/O

The alternate I/O function is enabled by setting the ALTIO bit (UxMODE<10>). If ALTIO = 1, the UxATX and UxARX pins (alternate transmit and alternate receive pins, respectively) are used by the UART module instead of the UxTX and UxRX pins. If ALTIO = 0, the UxTX and UxRX pins are used by the UART module.

16.2.4 SETTING UP DATA, PARITY AND STOP BIT SELECTIONS

Control bits PDSEL<1:0> in the UxMODE register are used to select the data length and parity used in the transmission. The data length may either be 8 bits with even, odd or no parity, or 9 bits with no parity.

The STSEL bit determines whether one or two Stop bits will be used during data transmission.

The default (power-on) setting of the UART is 8 bits, no parity and 1 Stop bit (typically represented as 8, N, 1).

16.3 Transmitting Data

16.3.1 TRANSMITTING IN 8-BIT DATA MODE

The following steps must be performed in order to transmit 8-bit data:

1. Set up the UART:
First, the data length, parity and number of Stop bits must be selected. Then, the transmit and receive interrupt enable and priority bits are setup in the UxMODE and UxSTA registers. Also, the appropriate baud rate value must be written to the UxBRG register.
2. Enable the UART by setting the UARTEN bit (UxMODE<15>).
3. Set the UTXEN bit (UxSTA<10>), thereby enabling a transmission.
4. Write the byte to be transmitted to the lower byte of UxTXREG. The value will be transferred to the Transmit Shift register (UxTSR) immediately and the serial bit stream will start shifting out during the next rising edge of the baud clock. Alternatively, the data byte may be written while UTXEN = 0, following which, the user may set UTXEN. This will cause the serial bit stream to begin immediately because the baud clock will start from a cleared state.
5. A transmit interrupt will be generated, depending on the value of the interrupt control bit UTXISEL (UxSTA<15>).

16.3.2 TRANSMITTING IN 9-BIT DATA MODE

The sequence of steps involved in the transmission of 9-bit data is similar to 8-bit transmission, except that a 16-bit data word (of which the upper 7 bits are always clear) must be written to the UxTXREG register.

16.3.3 TRANSMIT BUFFER (UxTXB)

The transmit buffer is 9 bits wide and 4 characters deep. Including the Transmit Shift register (UxTSR), the user effectively has a 5-deep FIFO (First-In, First-Out) buffer. The UTXBF status bit (UxSTA<9>) indicates whether the transmit buffer is full.

If a user attempts to write to a full buffer, the new data will not be accepted into the FIFO, and no data shift will occur within the buffer. This enables recovery from a buffer overrun condition.

The FIFO is reset during any device Reset but is not affected when the device enters or wakes up from a Power Saving mode.

16.3.4 TRANSMIT INTERRUPT

The transmit interrupt flag (U1TXIF or U2TXIF) is located in the corresponding interrupt flag register.

The transmitter generates an edge to set the UxTXIF bit. The condition for generating the interrupt depends on the UTXISEL control bit:

- a) If UTXISEL = 0, an interrupt is generated when a word is transferred from the transmit buffer to the Transmit Shift register (UxTSR). This implies that the transmit buffer has at least one empty word.
- b) If UTXISEL = 1, an interrupt is generated when a word is transferred from the transmit buffer to the Transmit Shift register (UxTSR) and the transmit buffer is empty.

Switching between the two Interrupt modes during operation is possible and sometimes offers more flexibility.

16.3.5 TRANSMIT BREAK

Setting the UTXBRK bit (UxSTA<11>) will cause the UxTX line to be driven to logic '0'. The UTXBRK bit overrides all transmission activity. Therefore, the user should generally wait for the transmitter to be Idle before setting UTXBRK.

To send a break character, the UTXBRK bit must be set by software and must remain set for a minimum of 13 baud clock cycles. The UTXBRK bit is then cleared by software to generate Stop bits. The user must wait for a duration of at least one or two baud clock cycles in order to ensure a valid Stop bit(s) before reloading the UxTXB, or starting other transmitter activity. Transmission of a break character does not generate a transmit interrupt.

16.4 Receiving Data

16.4.1 RECEIVING IN 8-BIT OR 9-BIT DATA MODE

The following steps must be performed while receiving 8-bit or 9-bit data:

1. Set up the UART (see Section 16.3.1).
2. Enable the UART (see Section 16.3.1).
3. A receive interrupt will be generated when one or more data words have been received, depending on the receive interrupt settings specified by the URXISEL bits (UxSTA<7:6>).
4. Read the OERR bit to determine if an overrun error has occurred. The OERR bit must be reset in software.
5. Read the received data from UxRXREG. The act of reading UxRXREG will move the next word to the top of the receive FIFO, and the PERR and FERR values will be updated.

16.4.2 RECEIVE BUFFER (UxRXB)

The receive buffer is 4 words deep. Including the Receive Shift register (UxRSR), the user effectively has a 5-word deep FIFO buffer.

URXDA (UxSTA<0>) = 1 indicates that the receive buffer has data available. URXDA = 0 implies that the buffer is empty. If a user attempts to read an empty buffer, the old values in the buffer will be read and no data shift will occur within the FIFO.

The FIFO is reset during any device Reset. It is not affected when the device enters or wakes up from a Power Saving mode.

16.4.3 RECEIVE INTERRUPT

The receive interrupt flag (U1RXIF or U2RXIF) can be read from the corresponding interrupt flag register. The interrupt flag is set by an edge generated by the receiver. The condition for setting the receive interrupt flag depends on the settings specified by the URXISEL<1:0> (UxSTA<7:6>) control bits.

- a) If URXISEL<1:0> = 00 or 01, an interrupt is generated every time a data word is transferred from the Receive Shift register (UxRSR) to the receive buffer. There may be one or more characters in the receive buffer.
- b) If URXISEL<1:0> = 10, an interrupt is generated when a word is transferred from the Receive Shift register (UxRSR) to the receive buffer, which as a result of the transfer, contains 3 characters.
- c) If URXISEL<1:0> = 11, an interrupt is set when a word is transferred from the Receive Shift register (UxRSR) to the receive buffer, which as a result of the transfer, contains 4 characters (i.e., becomes full).

Switching between the Interrupt modes during operation is possible, though generally not advisable during normal operation.

16.5 Reception Error Handling

16.5.1 RECEIVE BUFFER OVERRUN ERROR (OERR BIT)

The OERR bit (UxSTA<1>) is set if all of the following conditions occur:

- a) The receive buffer is full.
- b) The Receive Shift register is full, but unable to transfer the character to the receive buffer.
- c) The Stop bit of the character in the UxRSR is detected, indicating that the UxRSR needs to transfer the character to the buffer.

Once OERR is set, no further data is shifted in UxRSR (until the OERR bit is cleared in software or a Reset occurs). The data held in UxRSR and UxRXREG remains valid.

16.5.2 FRAMING ERROR (FERR)

The FERR bit (UxSTA<2>) is set if a '0' is detected instead of a Stop bit. If two Stop bits are selected, both Stop bits must be '1', otherwise FERR will be set. The read only FERR bit is buffered along with the received data. It is cleared on any Reset.

16.5.3 PARITY ERROR (PERR)

The PERR bit (UxSTA<3>) is set if the parity of the received word is incorrect. This error bit is applicable only if a Parity mode (odd or even) is selected. The read only PERR bit is buffered along with the received data bytes. It is cleared on any Reset.

16.5.4 IDLE STATUS

When the receiver is active (i.e., between the initial detection of the Start bit and the completion of the Stop bit), the RIDLE bit (UxSTA<4>) is '0'. Between the completion of the Stop bit and detection of the next Start bit, the RIDLE bit is '1', indicating that the UART is Idle.

16.5.5 RECEIVE BREAK

The receiver will count and expect a certain number of bit times based on the values programmed in the PDSEL (UxMODE<2:1>) and STSEL (UxMODE<0>) bits.

If the break is longer than 13 bit times, the reception is considered complete after the number of bit times specified by PDSEL and STSEL. The URXDA bit is set, FERR is set, zeros are loaded into the receive FIFO, interrupts are generated if appropriate and the RIDLE bit is set.

When the module receives a long break signal and the receiver has detected the Start bit, the data bits and the invalid Stop bit (which sets the FERR), the receiver must wait for a valid Stop bit before looking for the next Start bit. It cannot assume that the break condition on the line is the next Start bit.

Break is regarded as a character containing all '0's with the FERR bit set. The break character is loaded into the buffer. No further reception can occur until a Stop bit is received. Note that RIDLE goes high when the Stop bit has not yet been received.

16.6 Address Detect Mode

Setting the ADDEN bit (UxSTA<5>) enables this special mode in which a 9th bit (URX8) value of '1' identifies the received word as an address, rather than data. This mode is only applicable for 9-bit data communication. The URXISEL control bit does not have any impact on interrupt generation in this mode since an interrupt (if enabled) will be generated every time the received word has the 9th bit set.

16.7 Loopback Mode

Setting the LPBACK bit enables this special mode in which the UxTX pin is internally connected to the UxRX pin. When configured for the Loopback mode, the UxRX pin is disconnected from the internal UART receive logic. However, the UxTX pin still functions as in a normal operation.

To select this mode:

- Configure UART for desired mode of operation.
- Set LPBACK = 1 to enable Loopback mode.
- Enable transmission as defined in Section 16.3.

16.8 Baud Rate Generator

The UART has a 16-bit baud rate generator to allow maximum flexibility in baud rate generation. The baud rate generator register (UxBRG) is readable and writable. The baud rate is computed as follows:

$$\text{BRG} = 16\text{-bit value held in UxBRG register (0 through 65535)}$$

$$\text{FCY} = \text{Instruction Clock Rate (1/TcY)}$$

The Baud Rate is given by Equation 16-1.

EQUATION 16-1: BAUD RATE

$$\text{Baud Rate} = \text{FCY} / (16 * (\text{BRG} + 1))$$

Therefore, the maximum baud rate possible is

$$\text{FCY} / 16 \quad (\text{if BRG} = 0),$$

and the minimum baud rate possible is

$$\text{FCY} / (16 * 65536).$$

With a full 16-bit baud rate generator at 30 MIPs operation, the minimum baud rate achievable is 28.5 bps.

16.9 Auto Baud Support

To allow the system to determine baud rates of received characters, the input can be optionally linked to a capture input (IC1 for UART1, IC2 for UART2). To enable this mode, the user must program the input capture module to detect the falling and rising edges of the Start bit.

16.10 UART Operation During CPU Sleep and Idle Modes

16.10.1 UART OPERATION DURING CPU SLEEP MODE

When the device enters Sleep mode, all clock sources to the module are shutdown and stay at logic '0'. If entry into Sleep mode occurs while a transmission is in progress, then the transmission is aborted. The UxTX pin is driven to logic '1'. Similarly, if entry into Sleep mode occurs while a reception is in progress, then the reception is aborted. The UxSTA, UxMODE, transmit and receive registers and buffers, and the UxBRG register are not affected by Sleep mode.

If the WAKE bit (UxSTA<7>) is set before the device enters Sleep mode, then a falling edge on the UxRX pin will generate a receive interrupt. The Receive Interrupt Select mode bit (URXISEL) has no effect for this function. If the receive interrupt is enabled, then this will wake-up the device from Sleep. The URTEN bit must be set in order to generate a wake-up interrupt.

16.10.2 UART OPERATION DURING CPU IDLE MODE

For the UART, the USIDL bit selects if the module will stop operation when the device enters Idle mode or whether the module will continue on Idle. If USIDL = 0, the module will continue operation during Idle mode. If USIDL = 1, the module will stop on Idle.

TABLE 16-1: 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 | Reset State |
|----------|-------|-------------------------------|--------|--------|--------|--------|--------|-------|-------|-------------------|----------|-------|-------|-------|--------|--------|-------|---------------------|
| U1MODE | 020C | UARTEN | — | USIDL | — | — | ALTIO | — | — | WAKE | LPBACK | ABAUD | — | — | PDSEL1 | PDSEL0 | STSEL | 0000 0000 0000 0000 |
| U1STA | 020E | UTXISEL | — | — | — | UTXBRK | UTXEN | UTXBF | TRMT | URXISEL1 | URXISEL0 | ADDEN | RIDLE | PERR | FERR | OERR | URXDA | 0000 0001 0001 0000 |
| U1TXREG | 0210 | — | — | — | — | — | — | — | UTX8 | Transmit Register | | | | | | | | 0000 000u uuuu uuuu |
| U1RXREG | 0212 | — | — | — | — | — | — | — | URX8 | Receive Register | | | | | | | | 0000 0000 0000 0000 |
| U1BRG | 0214 | Baud Rate Generator Prescaler | | | | | | | | | | | | | | | | 0000 0000 0000 0000 |

Legend: u = uninitialized bit

TABLE 16-2: 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 | Reset State |
|----------|-------|-------------------------------|--------|--------|--------|--------|--------|-------|-------|-------------------|----------|-------|-------|-------|--------|--------|-------|---------------------|
| U2MODE | 0216 | UARTEN | — | USIDL | — | — | — | — | — | WAKE | LPBACK | ABAUD | — | — | PDSEL1 | PDSEL0 | STSEL | 0000 0000 0000 0000 |
| U2STA | 0218 | UTXISEL | — | — | — | UTXBRK | UTXEN | UTXBF | TRMT | URXISEL1 | URXISEL0 | ADDEN | RIDLE | PERR | FERR | OERR | URXDA | 0000 0001 0001 0000 |
| U2TXREG | 021A | — | — | — | — | — | — | — | UTX8 | Transmit Register | | | | | | | | 0000 000u uuuu uuuu |
| U2RXREG | 021C | — | — | — | — | — | — | — | URX8 | Receive Register | | | | | | | | 0000 0000 0000 0000 |
| U2BRG | 021E | Baud Rate Generator Prescaler | | | | | | | | | | | | | | | | 0000 0000 0000 0000 |

Legend: u = uninitialized bit

Note: Refer to *dsPIC30F Family Reference Manual (DS70046)* for descriptions of register bit fields.

dsPIC30F

NOTES:

17.0 CAN MODULE

Note: This data sheet summarizes features of this group of dsPIC30F devices and is not intended to be a complete reference source. For more information on the CPU, peripherals, register descriptions and general device functionality, refer to the *dsPIC30F Family Reference Manual* (DS70046).

17.1 Overview

The Controller Area Network (CAN) module is a serial interface, useful for communicating with other CAN modules or microcontroller devices. This interface/protocol was designed to allow communications within noisy environments.

The CAN module is a communication controller implementing the CAN 2.0 A/B protocol, as defined in the BOSCH specification. The module will support CAN 1.2, CAN 2.0A, CAN 2.0B Passive, and CAN 2.0B Active versions of the protocol. The module implementation is a full CAN system. The CAN specification is not covered within this data sheet. The reader may refer to the BOSCH CAN specification for further details.

The module features are as follows:

- Implementation of the CAN protocol CAN 1.2, CAN 2.0A and CAN 2.0B
- Standard and extended data frames
- 0-8 bytes data length
- Programmable bit rate up to 1 Mbit/sec
- Support for remote frames
- Double-buffered receiver with two prioritized received message storage buffers (each buffer may contain up to 8 bytes of data)
- 6 full (standard/extended identifier) acceptance filters, 2 associated with the high priority receive buffer and 4 associated with the low priority receive buffer
- 2 full acceptance filter masks, one each associated with the high and low priority receive buffers
- Three transmit buffers with application specified prioritization and abort capability (each buffer may contain up to 8 bytes of data)
- Programmable wake-up functionality with integrated low-pass filter
- Programmable Loopback mode supports self-test operation
- Signaling via interrupt capabilities for all CAN receiver and transmitter error states
- Programmable clock source
- Programmable link to Input Capture module (IC2, for both CAN1 and CAN2) for time-stamping and network synchronization
- Low power Sleep and Idle mode

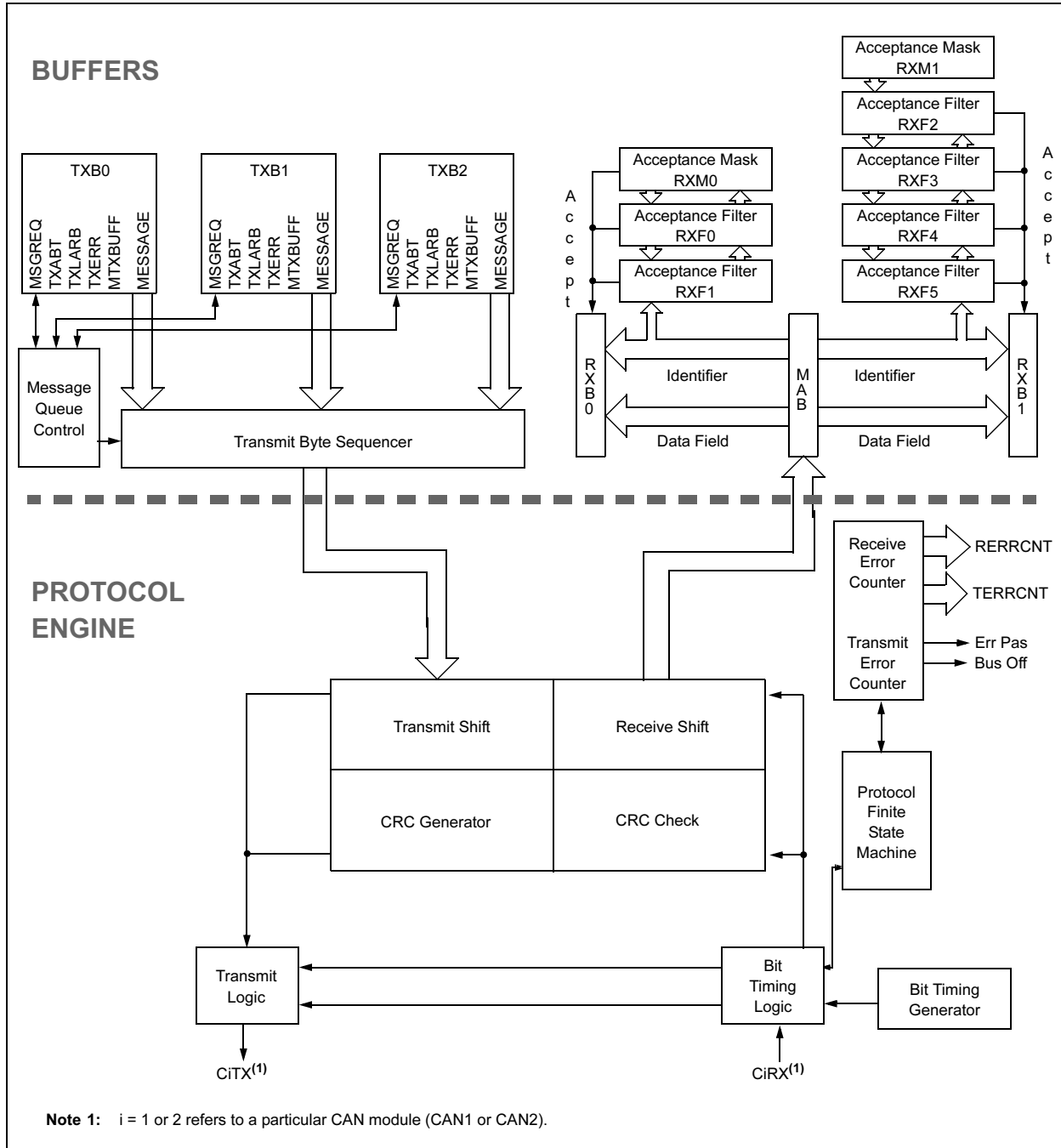
The CAN bus module consists of a protocol engine and message buffering/control. The CAN protocol engine handles all functions for receiving and transmitting messages on the CAN bus. Messages are transmitted by first loading the appropriate data registers. Status and errors can be checked by reading the appropriate registers. Any message detected on the CAN bus is checked for errors and then matched against filters to see if it should be received and stored in one of the receive registers.

17.2 Frame Types

The CAN module transmits various types of frames which include data messages or remote transmission requests initiated by the user, as other frames that are automatically generated for control purposes. The following frame types are supported:

- **Standard Data Frame:**
A standard data frame is generated by a node when the node wishes to transmit data. It includes an 11-bit standard identifier (SID) but not an 18-bit extended identifier (EID).
- **Extended Data Frame:**
An extended data frame is similar to a standard data frame but includes an extended identifier as well.
- **Remote Frame:**
It is possible for a destination node to request the data from the source. For this purpose, the destination node sends a remote frame with an identifier that matches the identifier of the required data frame. The appropriate data source node will then send a data frame as a response to this remote request.
- **Error Frame:**
An error frame is generated by any node that detects a bus error. An error frame consists of 2 fields: an error flag field and an error delimiter field.
- **Overload Frame:**
An overload frame can be generated by a node as a result of 2 conditions. First, the node detects a dominant bit during interframe space which is an illegal condition. Second, due to internal conditions, the node is not yet able to start reception of the next message. A node may generate a maximum of 2 sequential overload frames to delay the start of the next message.
- **Interframe Space:**
Interframe space separates a preceding frame (of whatever type) from a following data or remote frame.

FIGURE 17-1: CAN BUFFERS AND PROTOCOL ENGINE BLOCK DIAGRAM



17.3 Modes of Operation

The CAN module can operate in one of several Operation modes selected by the user. These modes include:

- Initialization Mode
- Disable Mode
- Normal Operation Mode
- Listen Only Mode
- Loopback Mode
- Error Recognition Mode

Modes are requested by setting the REQOP<2:0> bits (CiCTRL<10:8>), except the Error Recognition mode which is requested through the RXM<1:0> bits (CiRXnCON<6:5>, where n = 0 or 1 represents a particular receive buffer). Entry into a mode is Acknowledged by monitoring the OPMODE<2:0> bits (CiCTRL<7:5>). The module will not change the mode and the OPMODE bits until a change in mode is acceptable, generally during bus Idle time which is defined as at least 11 consecutive recessive bits.

17.3.1 INITIALIZATION MODE

In the Initialization mode, the module will not transmit or receive. The error counters are cleared and the interrupt flags remain unchanged. The programmer will have access to configuration registers that are access restricted in other modes. The module will protect the user from accidentally violating the CAN protocol through programming errors. All registers which control the configuration of the module can not be modified while the module is on-line. The CAN module will not be allowed to enter the Configuration mode while a transmission is taking place. The Configuration mode serves as a lock to protect the following registers.

- All Module Control Registers
- Baud Rate and Interrupt Configuration Registers
- Bus Timing Registers
- Identifier Acceptance Filter Registers
- Identifier Acceptance Mask Registers

17.3.2 DISABLE MODE

In Disable mode, the module will not transmit or receive. The module has the ability to set the WAKIF bit due to bus activity, however, any pending interrupts will remain and the error counters will retain their value.

If the REQOP<2:0> bits (CiCTRL<10:8>) = 001, the module will enter the Module Disable mode. If the module is active, the module will wait for 11 recessive bits on the CAN bus, detect that condition as an Idle bus, then accept the module disable command. When the OPMODE<2:0> bits (CiCTRL<7:5>) = 001, that indicates whether the module successfully went into Module Disable mode. The I/O pins will revert to normal I/O function when the module is in the Module Disable mode.

The module can be programmed to apply a low-pass filter function to the CiRX input line while the module or the CPU is in Sleep mode. The WAKFIL bit (CiCFG2<14>) enables or disables the filter.

Note: Typically, if the CAN module is allowed to transmit in a particular mode of operation and a transmission is requested immediately after the CAN module has been placed in that mode of operation, the module waits for 11 consecutive recessive bits on the bus before starting transmission. If the user switches to Disable Mode within this 11-bit period, then this transmission is aborted and the corresponding TXABT bit is set and TXREQ bit is cleared.

17.3.3 NORMAL OPERATION MODE

Normal Operating mode is selected when REQOP<2:0> = 000. In this mode, the module is activated and the I/O pins will assume the CAN bus functions. The module will transmit and receive CAN bus messages via the CxTX and CxRX pins.

17.3.4 LISTEN ONLY MODE

If the Listen Only mode is activated, the module on the CAN bus is passive. The transmitter buffers revert to the port I/O function. The receive pins remain inputs. For the receiver, no error flags or Acknowledge signals are sent. The error counters are deactivated in this state. The Listen Only mode can be used for detecting the baud rate on the CAN bus. To use this, it is necessary that there are at least two further nodes that communicate with each other.

17.3.5 LISTEN ALL MESSAGES MODE

The module can be set to ignore all errors and receive any message. The Error Recognition mode is activated by setting REQOP<2:0> = 111. In this mode, the data which is in the message assembly buffer until the time an error occurred, is copied in the receive buffer and can be read via the CPU interface.

17.3.6 LOOPBACK MODE

If the Loopback mode is activated, the module will connect the internal transmit signal to the internal receive signal at the module boundary. The transmit and receive pins revert to their port I/O function.

17.4 Message Reception

17.4.1 RECEIVE BUFFERS

The CAN bus module has 3 receive buffers. However, one of the receive buffers is always committed to monitoring the bus for incoming messages. This buffer is called the Message Assembly Buffer (MAB). So there are 2 receive buffers visible, RXB0 and RXB1, that can essentially instantaneously receive a complete message from the protocol engine.

All messages are assembled by the MAB and are transferred to the RXBn buffers only if the acceptance filter criterion are met. When a message is received, the RXnIF flag (CiINTF<0> or CiINRF<1>) will be set. This bit can only be set by the module when a message is received. The bit is cleared by the CPU when it has completed processing the message in the buffer. If the RXnIE bit (CiINTE<0> or CiINTE<1>) is set, an interrupt will be generated when a message is received.

RXF0 and RXF1 filters with RXM0 mask are associated with RXB0. The filters RXF2, RXF3, RXF4, and RXF5 and the mask RXM1 are associated with RXB1.

17.4.2 MESSAGE ACCEPTANCE FILTERS

The message acceptance filters and masks are used to determine if a message in the message assembly buffer should be loaded into either of the receive buffers. Once a valid message has been received into the Message Assembly Buffer (MAB), the identifier fields of the message are compared to the filter values. If there is a match, that message will be loaded into the appropriate receive buffer.

The acceptance filter looks at incoming messages for the RXIDE bit (CiRXnSID<0>) to determine how to compare the identifiers. If the RXIDE bit is clear, the message is a standard frame and only filters with the EXIDE bit (CiRXFnSID<0>) clear are compared. If the RXIDE bit is set, the message is an extended frame, and only filters with the EXIDE bit set are compared. Configuring the RXM<1:0> bits to '01' or '10' can override the EXIDE bit.

17.4.3 MESSAGE ACCEPTANCE FILTER MASKS

The mask bits essentially determine which bits to apply the filter to. If any mask bit is set to a zero, then that bit will automatically be accepted regardless of the filter bit. There are 2 programmable acceptance filter masks associated with the receive buffers, one for each buffer.

17.4.4 RECEIVE OVERRUN

An overrun condition occurs when the Message Assembly Buffer (MAB) has assembled a valid received message, the message is accepted through the acceptance filters, and when the receive buffer associated with the filter has not been designated as clear of the previous message.

The overrun error flag, RXnOVR (CiINTF<15> or CiINTF<14>), and the ERRIF bit (CiINTF<5>) will be set and the message in the MAB will be discarded.

If the DBEN bit is clear, RXB1 and RXB0 operate independently. When this is the case, a message intended for RXB0 will not be diverted into RXB1 if RXB0 contains an unread message and the RXOVR bit will be set.

If the DBEN bit is set, the overrun for RXB0 is handled differently. If a valid message is received for RXB0 and RXFUL = 1 indicates that RXB0 is full and RXFUL = 0 indicates that RXB1 is empty, the message for RXB0 will be loaded into RXB1. An overrun error will not be generated for RXB0. If a valid message is received for RXB0 and RXFUL = 1, indicating that both RXB0 and RXB1 are full, the message will be lost and an overrun will be indicated for RXB1.

17.4.5 RECEIVE ERRORS

The CAN module will detect the following receive errors:

- Cyclic Redundancy Check (CRC) Error
- Bit Stuffing Error
- Invalid Message Receive Error

These receive errors do not generate an interrupt. However, the receive error counter is incremented by one in case one of these errors occur. The RXWAR bit (CiINTF<9>) indicates that the receive error counter has reached the CPU warning limit of 96 and an interrupt is generated.

17.4.6 RECEIVE INTERRUPTS

Receive interrupts can be divided into 3 major groups, each including various conditions that generate interrupts:

- Receive Interrupt:
A message has been successfully received and loaded into one of the receive buffers. This interrupt is activated immediately after receiving the End of Frame (EOF) field. Reading the RXnIF flag will indicate which receive buffer caused the interrupt.
- Wake-up Interrupt:
The CAN module has woken up from Disable mode or the device has woken up from Sleep mode.

- Receive Error Interrupts:

A receive error interrupt will be indicated by the ERRIF bit. This bit shows that an error condition occurred. The source of the error can be determined by checking the bits in the CAN Interrupt Status register, CiINTF.

- Invalid Message Received:

If any type of error occurred during reception of the last message, an error will be indicated by the IVRIF bit.

- Receiver Overrun:

The RXnOVR bit indicates that an overrun condition occurred.

- Receiver Warning:

The RXWAR bit indicates that the receive error counter (RERRCNT<7:0>) has reached the warning limit of 96.

- Receiver Error Passive:

The RXEP bit indicates that the receive error counter has exceeded the error passive limit of 127 and the module has gone into error passive state.

17.5 Message Transmission

17.5.1 TRANSMIT BUFFERS

The CAN module has three transmit buffers. Each of the three buffers occupies 14 bytes of data. Eight of the bytes are the maximum 8 bytes of the transmitted message. Five bytes hold the standard and extended identifiers and other message arbitration information.

17.5.2 TRANSMIT MESSAGE PRIORITY

Transmit priority is a prioritization within each node of the pending transmittable messages. There are 4 levels of transmit priority. If TXPRI<1:0> (CiTXnCON<1:0>, where n = 0, 1 or 2 represents a particular transmit buffer) for a particular message buffer is set to '11', that buffer has the highest priority. If TXPRI<1:0> for a particular message buffer is set to '10' or '01', that buffer has an intermediate priority. If TXPRI<1:0> for a particular message buffer is '00', that buffer has the lowest priority.

17.5.3 TRANSMISSION SEQUENCE

To initiate transmission of the message, the TXREQ bit (CiTXnCON<3>) must be set. The CAN bus module resolves any timing conflicts between setting of the TXREQ bit and the Start of Frame (SOF), ensuring that if the priority was changed, it is resolved correctly before the SOF occurs. When TXREQ is set, the TXABT (CiTXnCON<6>), TXLAR (CiTXnCON<5>) and TXERR (CiTXnCON<4>) flag bits are automatically cleared.

Setting TXREQ bit simply flags a message buffer as enqueued for transmission. When the module detects an available bus, it begins transmitting the message which has been determined to have the highest priority.

If the transmission completes successfully on the first attempt, the TXREQ bit is cleared automatically, and an interrupt is generated if TXIE was set.

If the message transmission fails, one of the error condition flags will be set, and the TXREQ bit will remain set indicating that the message is still pending for transmission. If the message encountered an error condition during the transmission attempt, the TXERR bit will be set, and the error condition may cause an interrupt. If the message loses arbitration during the transmission attempt, the TXLAR bit is set. No interrupt is generated to signal the loss of arbitration.

17.5.4 ABORTING MESSAGE TRANSMISSION

The system can also abort a message by clearing the TXREQ bit associated with each message buffer. Setting the ABAT bit (CiCTRL<12>) will request an abort of all pending messages. If the message has not yet started transmission, or if the message started but is interrupted by loss of arbitration or an error, the abort will be processed. The abort is indicated when the module sets the TXABT bit and the TXnIF flag is not automatically set.

17.5.5 TRANSMISSION ERRORS

The CAN module will detect the following transmission errors:

- Acknowledge Error
- Form Error
- Bit Error

These transmission errors will not necessarily generate an interrupt but are indicated by the transmission error counter. However, each of these errors will cause the transmission error counter to be incremented by one. Once the value of the error counter exceeds the value of 96, the ERRIF (CiINTF<5>) and the TXWAR bit (CiINTF<10>) are set. Once the value of the error counter exceeds the value of 96, an interrupt is generated and the TXWAR bit in the Error Flag register is set.

17.5.6 TRANSMIT INTERRUPTS

Transmit interrupts can be divided into 2 major groups, each including various conditions that generate interrupts:

- **Transmit Interrupt:**
At least one of the three transmit buffers is empty (not scheduled) and can be loaded to schedule a message for transmission. Reading the TXnIF flags will indicate which transmit buffer is available and caused the interrupt.
- **Transmit Error Interrupts:**
A transmission error interrupt will be indicated by the ERRIF flag. This flag shows that an error condition occurred. The source of the error can be determined by checking the error flags in the CAN Interrupt Status register, CiINTF. The flags in this register are related to receive and transmit errors.
 - **Transmitter Warning Interrupt:**
The TXWAR bit indicates that the transmit error counter has reached the CPU warning limit of 96.
 - **Transmitter Error Passive:**
The TXEP bit (CiINTF<12>) indicates that the transmit error counter has exceeded the error passive limit of 127 and the module has gone to error passive state.
 - **Bus Off:**
The TXBO bit (CiINTF<13>) indicates that the transmit error counter has exceeded 255 and the module has gone to the bus off state.

17.6 Baud Rate Setting

All nodes on any particular CAN bus must have the same nominal bit rate. In order to set the baud rate, the following parameters have to be initialized:

- Synchronization Jump Width
- Baud Rate Prescaler
- Phase Segments
- Length determination of Phase Segment 2
- Sample Point
- Propagation Segment bits

17.6.1 BIT TIMING

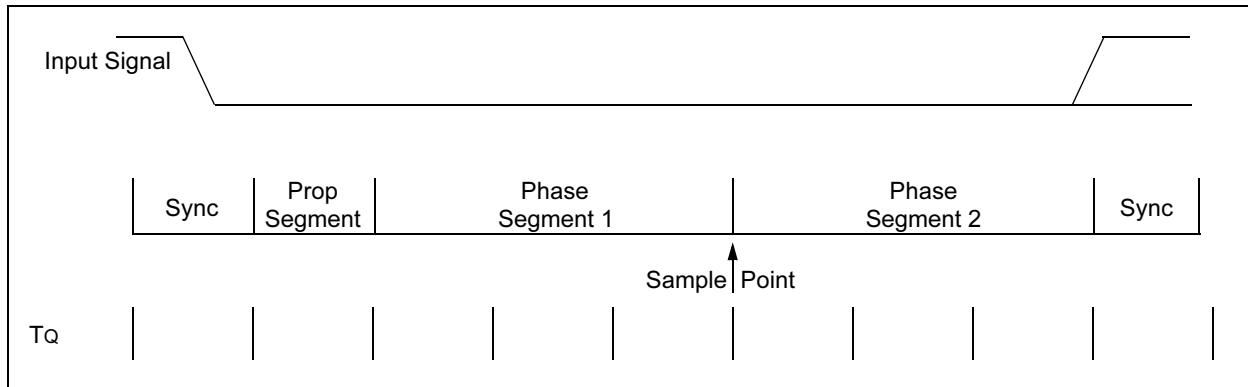
All controllers on the CAN bus must have the same baud rate and bit length. However, different controllers are not required to have the same master oscillator clock. At different clock frequencies of the individual controllers, the baud rate has to be adjusted by adjusting the number of time quanta in each segment.

The nominal bit time can be thought of as being divided into separate non-overlapping time segments. These segments are shown in Figure 17-2.

- Synchronization Segment (Sync Seg)
- Propagation Time Segment (Prop Seg)
- Phase Segment 1 (Phase1 Seg)
- Phase Segment 2 (Phase2 Seg)

The time segments and also the nominal bit time are made up of integer units of time called time quanta or T_Q. By definition, the nominal bit time has a minimum of 8 T_Q and a maximum of 25 T_Q. Also, by definition, the minimum nominal bit time is 1 μsec corresponding to a maximum bit rate of 1 MHz.

FIGURE 17-2: CAN BIT TIMING



17.6.2 PRESCALER SETTING

There is a programmable prescaler with integral values ranging from 1 to 64, in addition to a fixed divide-by-2 for clock generation. The time quantum (T_Q) is a fixed unit of time derived from the oscillator period, and is given by Equation 17-1, where F_{CAN} is F_{CY} (if the CANCKS bit is set) or 4F_{CY} (if CANCKS is clear).

Note: F_{CAN} must not exceed 30 MHz. If CANCKS = 0, then F_{CY} must not exceed 7.5 MHz.

EQUATION 17-1: TIME QUANTUM FOR CLOCK GENERATION

$$TQ = 2 (BRP<5:0> + 1) / F_{CAN}$$

17.6.3 PROPAGATION SEGMENT

This part of the bit time is used to compensate physical delay times within the network. These delay times consist of the signal propagation time on the bus line and the internal delay time of the nodes. The Prop Seg can be programmed from 1 T_Q to 8 T_Q by setting the PRSEG<2:0> bits (CiCFG2<2:0>).

17.6.4 PHASE SEGMENTS

The phase segments are used to optimally locate the sampling of the received bit within the transmitted bit time. The sampling point is between Phase1 Seg and Phase2 Seg. These segments are lengthened or shortened by resynchronization. The end of the Phase1 Seg determines the sampling point within a bit period. The segment is programmable from 1 T_Q to 8 T_Q. Phase2 Seg provides delay to the next transmitted data transition. The segment is programmable from 1 T_Q to 8 T_Q, or it may be defined to be equal to the greater of Phase1 Seg or the information processing time (2 T_Q). The Phase1 Seg is initialized by setting bits SEG1PH<2:0> (CiCFG2<5:3>), and Phase2 Seg is initialized by setting SEG2PH<2:0> (CiCFG2<10:8>).

The following requirement must be fulfilled while setting the lengths of the phase segments:

$$\text{Prop Seg} + \text{Phase1 Seg} > = \text{Phase2 Seg}$$

17.6.5 SAMPLE POINT

The sample point is the point of time at which the bus level is read and interpreted as the value of that respective bit. The location is at the end of Phase1 Seg. If the bit timing is slow and contains many T_Q, it is possible to specify multiple sampling of the bus line at the sample point. The level determined by the CAN bus then corresponds to the result from the majority decision of three values. The majority samples are taken at the sample point and twice before with a distance of T_Q/2. The CAN module allows the user to choose between sampling three times at the same point or once at the same point, by setting or clearing the SAM bit (CiCFG2<6>).

Typically, the sampling of the bit should take place at about 60 - 70% through the bit time, depending on the system parameters.

17.6.6 SYNCHRONIZATION

To compensate for phase shifts between the oscillator frequencies of the different bus stations, each CAN controller must be able to synchronize to the relevant signal edge of the incoming signal. When an edge in the transmitted data is detected, the logic will compare the location of the edge to the expected time (Synchronous Segment). The circuit will then adjust the values of Phase1 Seg and Phase2 Seg. There are 2 mechanisms used to synchronize.

17.6.6.1 Hard Synchronization

Hard synchronization is only done whenever there is a 'recessive' to 'dominant' edge during bus Idle indicating the start of a message. After hard synchronization, the bit time counters are restarted with the Sync Seg. Hard synchronization forces the edge which has caused the hard synchronization to lie within the synchronization segment of the restarted bit time. If a hard synchronization is done, there will not be a resynchronization within that bit time.

17.6.6.2 Resynchronization

As a result of resynchronization, Phase1 Seg may be lengthened or Phase2 Seg may be shortened. The amount of lengthening or shortening of the phase buffer segment has an upper bound known as the synchronization jump width, and is specified by the SJW<1:0> bits (CiCFG1<7:6>). The value of the synchronization jump width will be added to Phase1 Seg or subtracted from Phase2 Seg. The resynchronization jump width is programmable between 1 T_Q and 4 T_Q.

The following requirement must be fulfilled while setting the SJW<1:0> bits:

$$\text{Phase2 Seg} > \text{Synchronization Jump Width}$$

TABLE 17-1: CAN1 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 | Reset State |
|------------|-------|---|--------|--------|--------|--------|--------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|---------------------|
| C1RXFOSID | 0300 | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | 000u uuuu uuuu uu0u |
| C1RXFOEIDH | 0302 | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | 0000 uuuu uuuu uuuu |
| C1RXFOEIDL | 0304 | Receive Acceptance Filter 0 Extended Identifier <5:0> | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | uuuu uu00 0000 0000 |
| C1RXF1SID | 0308 | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | 000u uuuu uuuu uu0u |
| C1RXF1EIDH | 030A | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | 0000 uuuu uuuu uuuu |
| C1RXF1EIDL | 030C | Receive Acceptance Filter 1 Extended Identifier <5:0> | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | uuuu uu00 0000 0000 |
| C1RXF2SID | 0310 | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | 000u uuuu uuuu uu0u |
| C1RXF2EIDH | 0312 | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | 0000 uuuu uuuu uuuu |
| C1RXF2EIDL | 0314 | Receive Acceptance Filter 2 Extended Identifier <5:0> | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | uuuu uu00 0000 0000 |
| C1RXF3SID | 0318 | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | 000u uuuu uuuu uu0u |
| C1RXF3EIDH | 031A | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | 0000 uuuu uuuu uuuu |
| C1RXF3EIDL | 031C | Receive Acceptance Filter 3 Extended Identifier <5:0> | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | uuuu uu00 0000 0000 |
| C1RXF4SID | 0320 | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | 000u uuuu uuuu uu0u |
| C1RXF4EIDH | 0322 | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | 0000 uuuu uuuu uuuu |
| C1RXF4EIDL | 0324 | Receive Acceptance Filter 4 Extended Identifier <5:0> | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | uuuu uu00 0000 0000 |
| C1RXF5SID | 0328 | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | 000u uuuu uuuu uu0u |
| C1RXF5EIDH | 032A | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | 0000 uuuu uuuu uuuu |
| C1RXF5EIDL | 032C | Receive Acceptance Filter 5 Extended Identifier <5:0> | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | uuuu uu00 0000 0000 |
| C1RXM0SID | 0330 | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | 000u uuuu uuuu uu0u |
| C1RXM0EIDH | 0332 | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | 0000 uuuu uuuu uuuu |
| C1RXM0EIDL | 0334 | Receive Acceptance Mask 0 Extended Identifier <5:0> | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | uuuu uu00 0000 0000 |
| C1RXM1SID | 0338 | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | 000u uuuu uuuu uu0u |
| C1RXM1EIDH | 033A | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | 0000 uuuu uuuu uuuu |
| C1RXM1EIDL | 033C | Receive Acceptance Mask 1 Extended Identifier <5:0> | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | uuuu uu00 0000 0000 |
| C1TX2SID | 0340 | Transmit Buffer 2 Standard Identifier <10:6> | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | uuuu uu00 0000 0000 |
| C1TX2EID | 0342 | Transmit Buffer 2 Extended Identifier <17:14> | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | uuuu 0000 uuuu uuuu |
| C1TX2DLC | 0344 | Transmit Buffer 2 Extended Identifier <5:0> | — | — | — | — | — | — | — | TXRB0 | TXRB1 | TXRTR | TXRB1 | TXRB0 | TXRB1 | TXRTR | TXRB1 | uuuu uuuu uuuu u000 |
| C1TX2B1 | 0346 | Transmit Buffer 2 Byte 1 | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | uuuu uuuu uuuu uuuu |
| C1TX2B2 | 0348 | Transmit Buffer 2 Byte 3 | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | uuuu uuuu uuuu uuuu |
| C1TX2B3 | 034A | Transmit Buffer 2 Byte 5 | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | uuuu uuuu uuuu uuuu |
| C1TX2B4 | 034C | Transmit Buffer 2 Byte 7 | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | uuuu uuuu uuuu uuuu |
| C1TX2CON | 034E | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | uuuu uuuu uuuu uuuu |
| C1TX1SID | 0350 | Transmit Buffer 1 Standard Identifier <10:6> | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | 0000 0000 0000 0000 |
| C1TX1EID | 0352 | Transmit Buffer 1 Extended Identifier <17:14> | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | uuuu uu00 uuuu uuuu |
| C1TX1DLC | 0354 | Transmit Buffer 1 Extended Identifier <5:0> | — | — | — | — | — | — | — | TXRB0 | TXRB1 | TXRTR | TXRB1 | TXRB0 | TXRB1 | TXRTR | TXRB1 | uuuu uuuu uuuu u000 |

Legend: u = uninitialized bit

TABLE 17-1: CAN1 REGISTER MAP (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 | Reset State | |
|----------|-------|---|--------|--------|---|------------------------------|-------------|-------|-------|-------|--|----------|-------|---------|--------------------------|------------|---------|---------------------|---------------------|
| C1TX1B1 | 0356 | | | | Transmit Buffer 1 Byte 1 | | | | | | | | | | Transmit Buffer 1 Byte 0 | | | | uuuu uuuu uuuu uuuu |
| C1TX1B2 | 0358 | | | | Transmit Buffer 1 Byte 3 | | | | | | | | | | Transmit Buffer 1 Byte 2 | | | | uuuu uuuu uuuu uuuu |
| C1TX1B3 | 035A | | | | Transmit Buffer 1 Byte 5 | | | | | | | | | | Transmit Buffer 1 Byte 4 | | | | uuuu uuuu uuuu uuuu |
| C1TX1B4 | 035C | | | | Transmit Buffer 1 Byte 7 | | | | | | | | | | Transmit Buffer 1 Byte 6 | | | | uuuu uuuu uuuu uuuu |
| C1TX1CON | 035E | — | — | — | — | — | — | — | — | — | TXABT | TXLARB | TXERR | TXREQ | — | TXPRI<1:0> | | 0000 0000 0000 0000 | |
| C1TX0SID | 0360 | Transmit Buffer 0 Standard Identifier <10:6> | | | | — | — | — | — | — | Transmit Buffer 0 Standard Identifier <5:0> | | | | SRR | TXIDE | | uuuu u000 uuuu uuuu | |
| C1TX0EID | 0362 | Transmit Buffer 0 Extended Identifier <17:14> | | | | — | — | — | — | — | Transmit Buffer 0 Extended Identifier <13:6> | | | | | | | uuuu 0000 uuuu uuuu | |
| C1TX0DLC | 0364 | Transmit Buffer 0 Extended Identifier <5:0> | | | | TXRTR | TXRB1 | | | TXRB0 | | DLC<3:0> | | | | | | uuuu uuuu uuuu u000 | |
| C1TX0B1 | 0366 | Transmit Buffer 0 Byte 1 | | | | | | | | | | | | | | | | uuuu uuuu uuuu uuuu | |
| C1TX0B2 | 0368 | Transmit Buffer 0 Byte 3 | | | | | | | | | | | | | | | | uuuu uuuu uuuu uuuu | |
| C1TX0B3 | 036A | Transmit Buffer 0 Byte 5 | | | | | | | | | | | | | | | | uuuu uuuu uuuu uuuu | |
| C1TX0B4 | 036C | Transmit Buffer 0 Byte 7 | | | | | | | | | | | | | | | | uuuu uuuu uuuu uuuu | |
| C1TX0CON | 036E | — | — | — | — | — | — | — | — | — | TXABT | TXLARB | TXERR | TXREQ | — | TXPRI<1:0> | | 0000 0000 0000 0000 | |
| C1RX1SID | 0370 | — | — | — | Receive Buffer 1 Standard Identifier <10:0> | | | | | | | | | | | SRR | RXIDE | 000u uuuu uuuu uuuu | |
| C1RX1EID | 0372 | — | — | — | Receive Buffer 1 Extended Identifier <17:6> | | | | | | | | | | | | | 0000 uuuu uuuu uuuu | |
| C1RX1DLC | 0374 | Receive Buffer 1 Extended Identifier <5:0> | | | | RXRTR | RXRB1 | | | | | | | | | DLC<3:0> | | uuuu uuuu 000u uuuu | |
| C1RX1B1 | 0376 | Receive Buffer 1 Byte 1 | | | | | | | | | | | | | | | | uuuu uuuu uuuu uuuu | |
| C1RX1B2 | 0378 | Receive Buffer 1 Byte 3 | | | | | | | | | | | | | | | | uuuu uuuu uuuu uuuu | |
| C1RX1B3 | 037A | Receive Buffer 1 Byte 5 | | | | | | | | | | | | | | | | uuuu uuuu uuuu uuuu | |
| C1RX1B4 | 037C | Receive Buffer 1 Byte 7 | | | | | | | | | | | | | | | | uuuu uuuu uuuu uuuu | |
| C1RX1CON | 037E | — | — | — | — | — | — | — | — | — | RXFUL | — | — | RXRTRRO | FILHIT<2:0> | | | 0000 0000 0000 0000 | |
| C1RX0SID | 0380 | — | — | — | Receive Buffer 0 Standard Identifier <10:0> | | | | | | | | | | | SRR | RXIDE | 000u uuuu uuuu uuuu | |
| C1RX0EID | 0382 | — | — | — | Receive Buffer 0 Extended Identifier <17:6> | | | | | | | | | | | | | 0000 uuuu uuuu uuuu | |
| C1RX0DLC | 0384 | Receive Buffer 0 Extended Identifier <5:0> | | | | RXRTR | RXRB1 | | | | | | | | | DLC<3:0> | | uuuu uuuu 000u uuuu | |
| C1RX0B1 | 0386 | Receive Buffer 0 Byte 1 | | | | | | | | | | | | | | | | uuuu uuuu uuuu uuuu | |
| C1RX0B2 | 0388 | Receive Buffer 0 Byte 3 | | | | | | | | | | | | | | | | uuuu uuuu uuuu uuuu | |
| C1RX0B3 | 038A | Receive Buffer 0 Byte 5 | | | | | | | | | | | | | | | | uuuu uuuu uuuu uuuu | |
| C1RX0B4 | 038C | Receive Buffer 0 Byte 7 | | | | | | | | | | | | | | | | uuuu uuuu uuuu uuuu | |
| C1RX0CON | 038E | — | — | — | — | — | — | — | — | — | RXFUL | — | — | RXRTRRO | DBEN | JTOFF | FILHITO | 0000 0000 0000 0000 | |
| C1CTRL | 0390 | CANCAP | — | CSIDLE | ABAT | CANCKS | REQOP<2:0> | — | — | — | OPMODE<2:0> | — | — | — | — | — | — | 0000 0100 1000 0000 | |
| C1CFG1 | 0392 | — | — | — | — | — | — | — | — | — | SJW<1:0> | — | — | — | BRP<5:0> | — | — | 0000 0000 0000 0000 | |
| C1CFG2 | 0394 | — | WAKFIL | — | — | — | SEG2PH<2:0> | — | — | — | SEG2PHTS | SAM | — | — | SEG1PH<2:0> | — | — | 0u00 0uuu uuuu uuuu | |
| C1INTF | 0396 | RX0OVR | RX1OVR | TXBO | TXEP | RXEP | TXWAR | RXWAR | EWARN | IVRIF | WAKIF | ERRIF | TX2IF | TX1IF | TX0IF | RX1IF | RX0IF | 0000 0000 0000 0000 | |
| C1INTE | 0398 | — | — | — | — | — | — | — | — | — | IVRIE | WAKIE | TX2IE | TX1IE | TX0IE | RX1IE | RX0IE | 0000 0000 0000 0000 | |
| C1EC | 039A | Transmit Error Count Register | | | | Receive Error Count Register | | | | | | | | | | | | 0000 0000 0000 0000 | |

Legend: u = uninitialized bit

TABLE 17-2: CAN2 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 | Reset State | |
|------------|-------|---|--------|--------|--------|--------|--------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------------|---------------------|
| C2RXF0SID | 03C0 | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | EXIDE | 000u uuuu uuuu uu0u |
| C2RXF0EIDL | 03C2 | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | 0000 uuuu uuuu uuuu |
| C2RXF0EIDL | 03C4 | Receive Acceptance Filter 0 Extended Identifier <5:0> | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | uuuu uu00 0000 0000 |
| C2RXF1SID | 03C8 | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | 000u uuuu uuuu uu0u |
| C2RXF1EIDL | 03CA | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | 0000 uuuu uuuu uuuu |
| C2RXF1EIDL | 03CC | Receive Acceptance Filter 1 Extended Identifier <5:0> | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | uuuu uu00 0000 0000 |
| C2RXF2SID | 03D0 | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | 000u uuuu uuuu uu0u |
| C2RXF2EIDL | 03D2 | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | 0000 uuuu uuuu uuuu |
| C2RXF2EIDL | 03D4 | Receive Acceptance Filter 2 Extended Identifier <5:0> | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | uuuu uu00 0000 0000 |
| C2RXF3SID | 03D8 | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | 000u uuuu uuuu uu0u |
| C2RXF3EIDL | 03DA | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | 0000 uuuu uuuu uuuu |
| C2RXF3EIDL | 03DC | Receive Acceptance Filter 3 Extended Identifier <5:0> | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | uuuu uu00 0000 0000 |
| C2RXF4SID | 03E0 | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | 000u uuuu uuuu uu0u |
| C2RXF4EIDL | 03E2 | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | 0000 uuuu uuuu uuuu |
| C2RXF4EIDL | 03E4 | Receive Acceptance Filter 4 Extended Identifier <5:0> | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | uuuu uu00 0000 0000 |
| C2RXF5SID | 03E8 | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | 000u uuuu uuuu uu0u |
| C2RXF5EIDL | 03EA | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | 0000 uuuu uuuu uuuu |
| C2RXF5EIDL | 03EC | Receive Acceptance Filter 5 Extended Identifier <5:0> | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | uuuu uu00 0000 0000 |
| C2RXM0SID | 03F0 | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | 000u uuuu uuuu uu0u |
| C2RXM0EIDL | 03F2 | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | 0000 uuuu uuuu uuuu |
| C2RXM0EIDL | 03F4 | Receive Acceptance Mask 0 Extended Identifier <17:6> | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | uuuu uu00 0000 0000 |
| C2RXM1SID | 03F8 | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | 000u uuuu uuuu uu0u |
| C2RXM1EIDL | 03FA | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | 0000 uuuu uuuu uuuu |
| C2RXM1EIDL | 03FC | Receive Acceptance Mask 1 Extended Identifier <5:0> | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | uuuu uu00 0000 0000 |
| C2TX2SID | 0400 | Transmit Buffer 2 Standard Identifier <10:6> | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | uuuu uu00 0000 0000 |
| C2TX2EID | 0402 | Transmit Buffer 2 Extended Identifier <17:14> | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | uuuu uu00 0000 0000 |
| C2TX2DLC | 0404 | Transmit Buffer 2 Extended Identifier <5:0> | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | uuuu uu00 0000 0000 |
| C2TX2B1 | 0406 | Transmit Buffer 2 Byte 1 | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | uuuu uu00 0000 0000 |
| C2TX2B2 | 0408 | Transmit Buffer 2 Byte 3 | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | uuuu uu00 0000 0000 |
| C2TX2B3 | 040A | Transmit Buffer 2 Byte 5 | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | uuuu uu00 0000 0000 |
| C2TX2B4 | 040C | Transmit Buffer 2 Byte 7 | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | uuuu uu00 0000 0000 |
| C2TX2CON | 040E | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | 0000 0000 0000 0000 |
| C2TX1SID | 0410 | Transmit Buffer 1 Standard Identifier <10:6> | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | uuuu uu00 0000 0000 |
| C2TX1EID | 0412 | Transmit Buffer 1 Extended Identifier <17:14> | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | uuuu 0000 uuuu uuuu |
| C2TX1DLC | 0414 | Transmit Buffer 1 Extended Identifier <5:0> | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | uuuu uu00 0000 0000 |

Legend: u = uninitialized bit

TABLE 17-2: CAN2 REGISTER MAP (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 | Reset State |
|----------|-------|--------|--------|--------|--------|--------|--------|-------------------------------|-------|----------|-------------|------------------------------|-------|--|------------|---------------------|---------|---------------------|
| C2TX1B1 | 0416 | | | | | | | | | | | | | Transmit Buffer 1 Byte 0 | | | | uuuu uuuu uuuu uuuu |
| C2TX1B2 | 0418 | | | | | | | | | | | | | Transmit Buffer 1 Byte 2 | | | | uuuu uuuu uuuu uuuu |
| C2TX1B3 | 041A | | | | | | | | | | | | | Transmit Buffer 1 Byte 4 | | | | uuuu uuuu uuuu uuuu |
| C2TX1B4 | 041C | | | | | | | | | | | | | Transmit Buffer 1 Byte 6 | | | | uuuu uuuu uuuu uuuu |
| C2TX1CON | 041E | | | | | | | | | | TXABT | TXLARB | TXERR | TXREQ | | TXPRI<1:0> | | 0000 0000 0000 0000 |
| C2TX0SID | 0420 | | | | | | | | | | | | | Transmit Buffer 0 Standard Identifier <5:0> | | | | uuuu u000 uuuu uuuu |
| C2TX0EID | 0422 | | | | | | | | | | | | | Transmit Buffer 0 Extended Identifier <13:6> | | | | uuuu 0000 uuuu uuuu |
| C2TX0DLC | 0424 | | | | | | | TXRTR | TXRB1 | TXRB0 | | | | DLC<3:0> | | | | uuuu uuuu uuuu u000 |
| C2TX0B1 | 0426 | | | | | | | | | | | | | Transmit Buffer 0 Byte 0 | | | | uuuu uuuu uuuu uuuu |
| C2TX0B2 | 0428 | | | | | | | | | | | | | Transmit Buffer 0 Byte 2 | | | | uuuu uuuu uuuu uuuu |
| C2TX0B3 | 042A | | | | | | | | | | | | | Transmit Buffer 0 Byte 4 | | | | uuuu uuuu uuuu uuuu |
| C2TX0B4 | 042C | | | | | | | | | | | | | Transmit Buffer 0 Byte 6 | | | | uuuu uuuu uuuu uuuu |
| C2TX0CON | 042E | | | | | | | | | | TXABT | TXLARB | TXERR | TXREQ | | TXPRI<1:0> | | 0000 0000 0000 0000 |
| C2RX1SID | 0430 | | | | | | | | | | | | | Receive Buffer 1 Standard Identifier <10:0> | | | | 000u uuuu uuuu uuuu |
| C2RX1EID | 0432 | | | | | | | | | | | | | Receive Buffer 1 Extended Identifier <17:6> | | | | 0000 uuuu uuuu uuuu |
| C2RX1DLC | 0434 | | | | | | | RXRTR | RXRB1 | RXRB0 | | | | DLC<3:0> | | | | uuuu uuuu uuuu uuuu |
| C2RX1B1 | 0436 | | | | | | | | | | | | | Receive Buffer 1 Byte 0 | | | | uuuu uuuu uuuu uuuu |
| C2RX1B2 | 0438 | | | | | | | | | | | | | Receive Buffer 1 Byte 2 | | | | uuuu uuuu uuuu uuuu |
| C2RX1B3 | 043A | | | | | | | | | | | | | Receive Buffer 1 Byte 4 | | | | uuuu uuuu uuuu uuuu |
| C2RX1B4 | 043C | | | | | | | | | | | | | Receive Buffer 1 Byte 6 | | | | uuuu uuuu uuuu uuuu |
| C2RX1CON | 043E | | | | | | | | | RXFUL | | | | RXRTRRO | | FILHIT<2:0> | | 0000 0000 0000 0000 |
| C2RX0SID | 0440 | | | | | | | | | | | | | Receive Buffer 0 Standard Identifier <10:0> | | | | 000u uuuu uuuu uuuu |
| C2RX0EID | 0442 | | | | | | | | | | | | | Receive Buffer 0 Extended Identifier <17:6> | | | | 0000 uuuu uuuu uuuu |
| C2RX0DLC | 0444 | | | | | | | RXRTR | RXRB1 | RXRB0 | | | | DLC<3:0> | | | | uuuu uuuu uuuu uuuu |
| C2RX0B1 | 0446 | | | | | | | | | | | | | Receive Buffer 0 Byte 0 | | | | uuuu uuuu uuuu uuuu |
| C2RX0B2 | 0448 | | | | | | | | | | | | | Receive Buffer 0 Byte 2 | | | | uuuu uuuu uuuu uuuu |
| C2RX0B3 | 044A | | | | | | | | | | | | | Receive Buffer 0 Byte 4 | | | | uuuu uuuu uuuu uuuu |
| C2RX0B4 | 044C | | | | | | | | | | | | | Receive Buffer 0 Byte 6 | | | | uuuu uuuu uuuu uuuu |
| C2RX0CON | 044E | | | | | | | | | RXFUL | | | | RXRTRRO | DBEN | JTOFF | FILHITO | 0000 0000 0000 0000 |
| C2CTRL | 0450 | | | | | | | REQOP<2:0> | | | OPMODE<2:0> | | | | ICODE<2:0> | | | 0000 0100 1000 0000 |
| C2CFG1 | 0452 | | | | | | | | | | SJW<1:0> | | | BRP<5:0> | | | | 0000 0000 0000 0000 |
| C2CFG2 | 0454 | | | | | | | SEG2PH<2:0> | | SEG2PHTS | SAM | | | SEG1PH<2:0> | | | | 0u00 0uuu uuuu uuuu |
| C2INTF | 0456 | | | | | | | TXWAR | RXWAR | EWARN | IVRIF | WAKIF | ERRIF | TX2IF | TX0IF | RX1IF | RX0IF | 0000 0000 0000 0000 |
| C2INTE | 0458 | | | | | | | | | | IVRIE | WAKIE | ERRIE | TX1IE | TX0IE | RX1IE | RX0IE | 0000 0000 0000 0000 |
| C2EC | 045A | | | | | | | Transmit Error Count Register | | | | Receive Error Count Register | | | | 0000 0000 0000 0000 | | |

Legend: u = uninitialized bit

Note: Refer to *dsPIC30F Family Reference Manual (DS70046)* for descriptions of register bit fields.

dsPIC30F

NOTES:

18.0 DATA CONVERTER INTERFACE (DCI) MODULE

Note: This data sheet summarizes features of this group of dsPIC30F devices and is not intended to be a complete reference source. For more information on the CPU, peripherals, register descriptions and general device functionality, refer to the *dsPIC30F Family Reference Manual* (DS70046).

18.1 Module Introduction

The dsPIC30F Data Converter Interface (DCI) module allows simple interfacing of devices, such as audio coder/decoders (Codecs), A/D converters and D/A converters. The following interfaces are supported:

- Framed Synchronous Serial Transfer (Single or Multi-Channel)
- Inter-IC Sound (I²S) Interface
- AC-Link Compliant mode

The DCI module provides the following general features:

- Programmable word size up to 16 bits
- Support for up to 16 time slots, for a maximum frame size of 256 bits
- Data buffering for up to 4 samples without CPU overhead

18.2 Module I/O Pins

There are four I/O pins associated with the module. When enabled, the module controls the data direction of each of the four pins.

18.2.1 CCLK PIN

The CCLK pin provides the serial clock for the DCI module. The CCLK pin may be configured as an input or output using the CCLKD control bit in the DCICON2 SFR. When configured as an output, the serial clock is provided by the dsPIC30F. When configured as an input, the serial clock must be provided by an external device.

18.2.2 CSDO PIN

The serial data output (CSDO) pin is configured as an output only pin when the module is enabled. The CSDO pin drives the serial bus whenever data is to be transmitted. The CSDO pin is tri-stated or driven to '0' during CCLK periods when data is not transmitted, depending on the state of the CSDOM control bit. This allows other devices to place data on the serial bus during transmission periods not used by the DCI module.

18.2.3 CSDI PIN

The serial data input (CSDI) pin is configured as an input only pin when the module is enabled.

18.2.3.1 COFS PIN

The Codec frame synchronization (COFS) pin is used to synchronize data transfers that occur on the CSDO and CSDI pins. The COFS pin may be configured as an input or an output. The data direction for the COFS pin is determined by the COFSD control bit in the DCICON1 register.

The DCI module accesses the shadow registers while the CPU is in the process of accessing the memory mapped buffer registers.

18.2.4 BUFFER DATA ALIGNMENT

Data values are always stored left justified in the buffers since most Codec data is represented as a signed 2's complement fractional number. If the received word length is less than 16 bits, the unused LS bits in the receive buffer registers are set to '0' by the module. If the transmitted word length is less than 16 bits, the unused LS bits in the transmit buffer register are ignored by the module. The word length setup is described in subsequent sections of this document.

18.2.5 TRANSMIT/RECEIVE SHIFT REGISTER

The DCI module has a 16-bit shift register for shifting serial data in and out of the module. Data is shifted in/out of the shift register MS bit first, since audio PCM data is transmitted in signed 2's complement format.

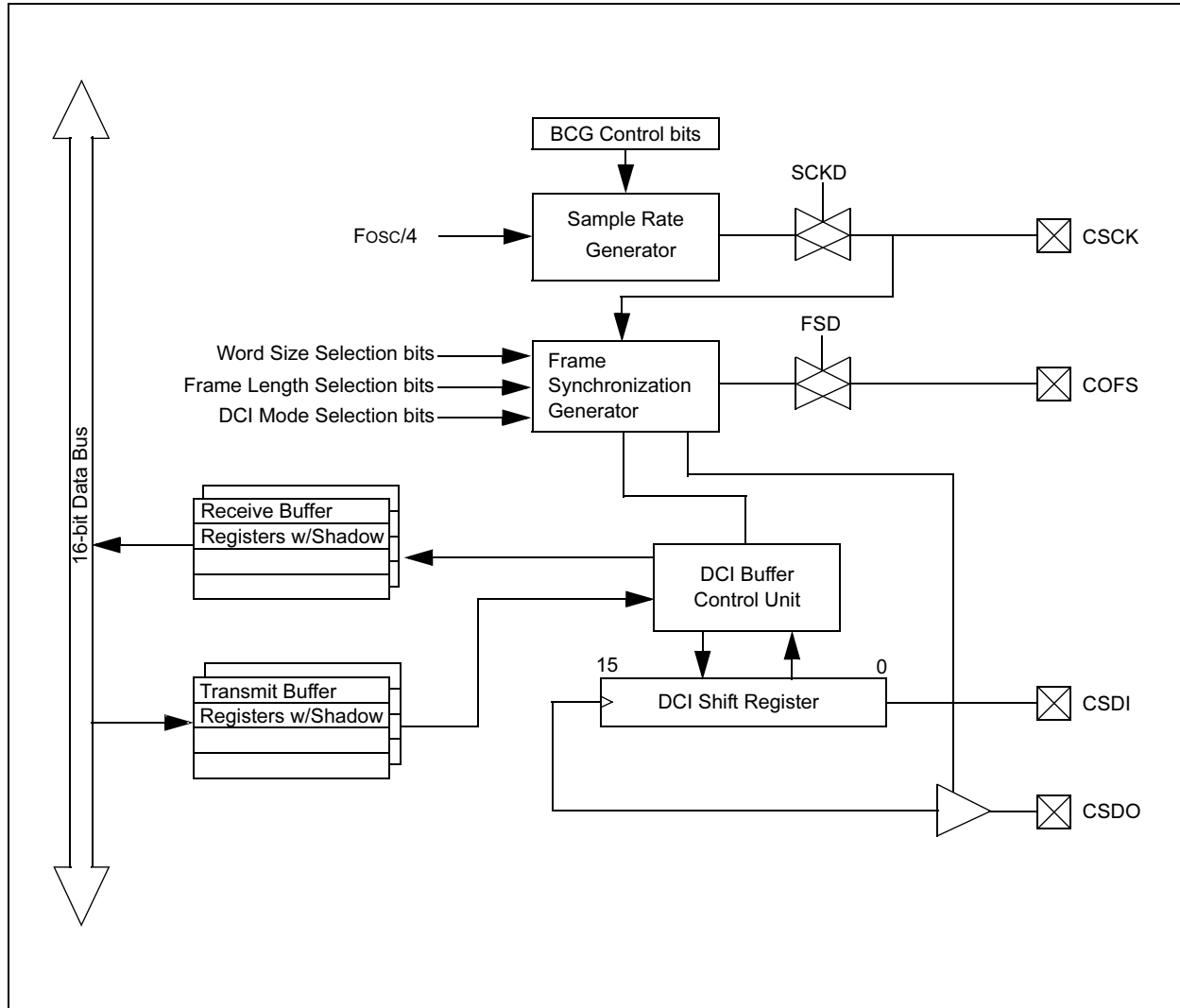
18.2.6 DCI BUFFER CONTROL

The DCI module contains a buffer control unit for transferring data between the shadow buffer memory and the serial shift register. The buffer control unit is a simple 2-bit address counter that points to word locations in the shadow buffer memory. For the receive memory space (high address portion of DCI buffer memory), the address counter is concatenated with a '0' in the MSb location to form a 3-bit address. For the transmit memory space (high portion of DCI buffer memory), the address counter is concatenated with a '1' in the MSb location.

Note: The DCI buffer control unit always accesses the same relative location in the transmit and receive buffers, so only one address counter is provided.

dsPIC30F

FIGURE 18-1: DCI MODULE BLOCK DIAGRAM



18.3 DCI Module Operation

18.3.1 MODULE ENABLE

The DCI module is enabled or disabled by setting/clearing the DCIEN control bit in the DCICON1 SFR. Clearing the DCIEN control bit has the effect of resetting the module. In particular, all counters associated with CSCK generation, frame sync, and the DCI buffer control unit are reset.

The DCI clocks are shutdown when the DCIEN bit is cleared.

When enabled, the DCI controls the data direction for the four I/O pins associated with the module. The Port, LAT and TRIS register values for these I/O pins are overridden by the DCI module when the DCIEN bit is set.

It is also possible to override the CSCK pin separately when the bit clock generator is enabled. This permits the bit clock generator to operate without enabling the rest of the DCI module.

18.3.2 WORD SIZE SELECTION BITS

The WS<3:0> word size selection bits in the DCICON2 SFR determine the number of bits in each DCI data word. Essentially, the WS<3:0> bits determine the counting period for a 4-bit counter clocked from the CSCK signal.

Any data length, up to 16-bits, may be selected. The value loaded into the WS<3:0> bits is one less the desired word length. For example, a 16-bit data word size is selected when WS<3:0> = 1111.

Note: These WS<3:0> control bits are used only in the Multi-Channel and I²S modes. These bits have no effect in AC-Link mode since the data slot sizes are fixed by the protocol.

18.3.3 FRAME SYNC GENERATOR

The frame sync generator (COFSG) is a 4-bit counter that sets the frame length in data words. The frame sync generator is incremented each time the word size counter is reset (refer to Section 18.3.2). The period for the frame synchronization generator is set by writing the COFSG<3:0> control bits in the DCICON2 SFR. The COFSG period in clock cycles is determined by the following formula:

EQUATION 18-1: COFSG PERIOD

$$\text{Frame Length} = \text{Word Length} \cdot (\text{FSG Value} + 1)$$

Frame lengths, up to 16 data words, may be selected. The frame length in CSCK periods can vary up to a maximum of 256 depending on the word size that is selected.

Note: The COFSG control bits will have no effect in AC-Link mode since the frame length is set to 256 CSCK periods by the protocol.

18.3.4 FRAME SYNC MODE CONTROL BITS

The type of frame sync signal is selected using the Frame Synchronization mode control bits (COFSM<1:0>) in the DCICON1 SFR. The following operating modes can be selected:

- Multi-Channel mode
- I²S mode
- AC-Link mode (16-bit)
- AC-Link mode (20-bit)

The operation of the COFSM control bits depends on whether the DCI module generates the frame sync signal as a master device, or receives the frame sync signal as a slave device.

The master device in a DSP/Codec pair is the device that generates the frame sync signal. The frame sync signal initiates data transfers on the CSDI and CSDO pins and usually has the same frequency as the data sample rate (COFS).

The DCI module is a frame sync master if the COFSD control bit is cleared and is a frame sync slave if the COFSD control bit is set.

18.3.5 MASTER FRAME SYNC OPERATION

When the DCI module is operating as a frame sync master device (COFSD = 0), the COFSM mode bits determine the type of frame sync pulse that is generated by the frame sync generator logic.

A new COFS signal is generated when the frame sync generator resets to '0'.

In the Multi-Channel mode, the frame sync pulse is driven high for the CSCK period to initiate a data transfer. The number of CSCK cycles between successive frame sync pulses will depend on the word size and frame sync generator control bits. A timing diagram for the frame sync signal in Multi-Channel mode is shown in Figure 18-2.

In the AC-Link mode of operation, the frame sync signal has a fixed period and duty cycle. The AC-Link frame sync signal is high for 16 CSCK cycles and is low for 240 CSCK cycles. A timing diagram with the timing details at the start of an AC-Link frame is shown in Figure 18-3.

In the I²S mode, a frame sync signal having a 50% duty cycle is generated. The period of the I²S frame sync signal in CSCK cycles is determined by the word size and frame sync generator control bits. A new I²S data transfer boundary is marked by a high-to-low or a low-to-high transition edge on the COFS pin.

dsPIC30F

18.3.6 SLAVE FRAME SYNC OPERATION

When the DCI module is operating as a frame sync slave (COFSD = 1), data transfers are controlled by the Codec device attached to the DCI module. The COFSM control bits control how the DCI module responds to incoming COFS signals.

In the Multi-Channel mode, a new data frame transfer will begin one CSMK cycle after the COFS pin is sampled high (see Figure 18-2). The pulse on the COFS pin resets the frame sync generator logic.

In the I²S mode, a new data word will be transferred one CSMK cycle after a low-to-high or a high-to-low transition is sampled on the COFS pin. A rising or falling edge on the COFS pin resets the frame sync generator logic.

In the AC-Link mode, the tag slot and subsequent data slots for the next frame will be transferred one CSMK cycle after the COFS pin is sampled high.

The COFSG and WS bits must be configured to provide the proper frame length when the module is operating in the Slave mode. Once a valid frame sync pulse has been sampled by the module on the COFS pin, an entire data frame transfer will take place. The module will not respond to further frame sync pulses until the data frame transfer has completed.

FIGURE 18-2: FRAME SYNC TIMING, MULTI-CHANNEL MODE

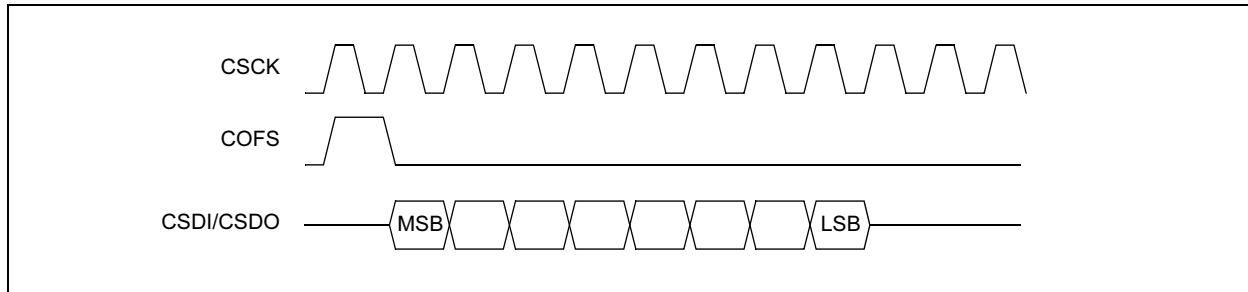


FIGURE 18-3: FRAME SYNC TIMING, AC-LINK START OF FRAME

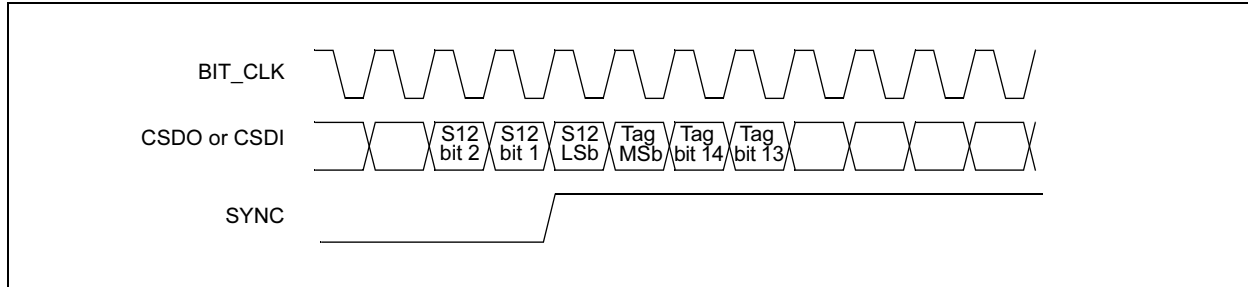
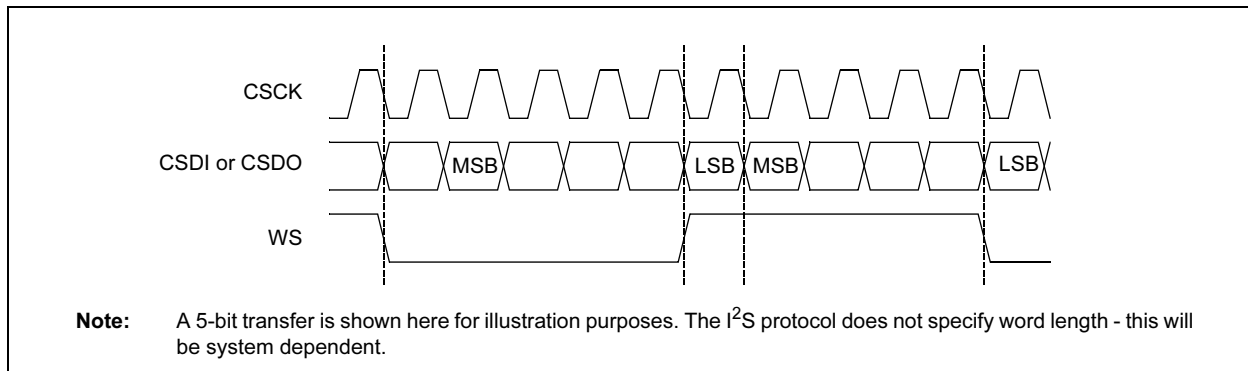


FIGURE 18-4: I²S INTERFACE FRAME SYNC TIMING



18.3.7 BIT CLOCK GENERATOR

The DCI module has a dedicated 12-bit time base that produces the bit clock. The bit clock rate (period) is set by writing a non-zero 12-bit value to the BCG<11:0> control bits in the DCICON1 SFR.

When the BCG<11:0> bits are set to zero, the bit clock will be disabled. If the BCG<11:0> bits are set to a non-zero value, the bit clock generator is enabled. These bits should be set to '0' and the CSCKD bit set to '1' if the serial clock for the DCI is received from an external device.

The formula for the bit clock frequency is given in Equation 18-2.

EQUATION 18-2: BIT CLOCK FREQUENCY

$$F_{BCK} = \frac{F_{CY}}{2 \cdot (BCG + 1)}$$

The required bit clock frequency will be determined by the system sampling rate and frame size. Typical bit clock frequencies range from 16x to 512x the converter sample rate depending on the data converter and the communication protocol that is used.

To achieve bit clock frequencies associated with common audio sampling rates, the user will need to select a crystal frequency that has an 'even' binary value. Examples of such crystal frequencies are listed in Table 18-1.

TABLE 18-1: DEVICE FREQUENCIES FOR COMMON CODEC CSCK FREQUENCIES

| Fosc | PLL | Fcyc |
|-----------|-----|-------------|
| 2.048 MHz | 16x | 32.768 MIPs |
| 4.096 MHz | 8x | 32.768 MIPs |
| 4.800 MHz | 8x | 38.4 MIPs |
| 9.600 MHz | 4x | 38.4 MIPs |

Note 1: When the CSCK signal is applied externally (CSCKD = 1), the BCG<11:0> bits have no effect on the operation of the DCI module.

2: When the CSCK signal is applied externally (CSCKD = 1), the external clock high and low times must meet the device timing requirements.

18.3.8 SAMPLE CLOCK EDGE CONTROL BIT

The sample clock edge (CSCKE) control bit determines the sampling edge for the CSCK signal. If the CSCK bit is cleared (default), data will be sampled on the falling edge of the CSCK signal. The AC-Link protocols and most Multi-Channel formats require that data be sampled on the falling edge of the CSCK signal. If the CSCK bit is set, data will be sampled on the rising edge of CSCK. The I²S protocol requires that data be sampled on the rising edge of the CSCK signal.

18.3.9 DATA JUSTIFICATION CONTROL BIT

In most applications, the data transfer begins one CSCK cycle after the COFS signal is sampled active. This is the default configuration of the DCI module. An alternate data alignment can be selected by setting the DJST control bit in the DCICON2 SFR. When DJST = 1, data transfers will begin during the same CSCK cycle when the COFS signal is sampled active.

18.3.10 TRANSMIT SLOT ENABLE BITS

The TSCON SFR has control bits that are used to enable up to 16 time slots for transmission. These control bits are the TSE<15:0> bits. The size of each time slot is determined by the WS<3:0> word size selection bits and can vary up to 16 bits.

If a transmit time slot is enabled via one of the TSE bits (TSE_x = 1), the contents of the current transmit shadow buffer location will be loaded into the CSDO Shift register and the DCI buffer control unit is incremented to point to the next location.

During an unused transmit time slot, the CSDO pin will drive '0's or will be tri-stated during all disabled time slots depending on the state of the CSDOM bit in the DCICON1 SFR.

The data frame size in bits is determined by the chosen data word size and the number of data word elements in the frame. If the chosen frame size has less than 16 elements, the additional slot enable bits will have no effect.

Each transmit data word is written to the 16-bit transmit buffer as left justified data. If the selected word size is less than 16 bits, then the LS bits of the transmit buffer memory will have no effect on the transmitted data. The user should write '0's to the unused LS bits of each transmit buffer location.

18.3.11 RECEIVE SLOT ENABLE BITS

The RSCON SFR contains control bits that are used to enable up to 16 time slots for reception. These control bits are the RSE<15:0> bits. The size of each receive time slot is determined by the WS<3:0> word size selection bits and can vary from 1 to 16 bits.

If a receive time slot is enabled via one of the RSE bits (RSE_x = 1), the shift register contents will be written to the current DCI receive shadow buffer location and the buffer control unit will be incremented to point to the next buffer location.

Data is not packed in the receive memory buffer locations if the selected word size is less than 16 bits. Each received slot data word is stored in a separate 16-bit buffer location. Data is always stored in a left justified format in the receive memory buffer.

18.3.12 SLOT ENABLE BITS OPERATION WITH FRAME SYNC

The TSE and RSE control bits operate in concert with the DCI frame sync generator. In the Master mode, a COFS signal is generated whenever the frame sync generator is reset. In the Slave mode, the frame sync generator is reset whenever a COFS pulse is received.

The TSE and RSE control bits allow up to 16 consecutive time slots to be enabled for transmit or receive. After the last enabled time slot has been transmitted/received, the DCI will stop buffering data until the next occurring COFS pulse.

18.3.13 SYNCHRONOUS DATA TRANSFERS

The DCI buffer control unit will be incremented by one word location whenever a given time slot has been enabled for transmission or reception. In most cases, data input and output transfers will be synchronized, which means that a data sample is received for a given channel at the same time a data sample is transmitted. Therefore, the transmit and receive buffers will be filled with equal amounts of data when a DCI interrupt is generated.

In some cases, the amount of data transmitted and received during a data frame may not be equal. As an example, assume a two-word data frame is used. Furthermore, assume that data is only received during slot #0 but is transmitted during slot #0 and slot #1. In this case, the buffer control unit counter would be incremented twice during a data frame but only one receive register location would be filled with data.

18.3.14 BUFFER LENGTH CONTROL

The amount of data that is buffered between interrupts is determined by the buffer length (BLEN<1:0>) control bits in the DCISTAT SFR. The size of the transmit and receive buffers may be varied from 1 to 4 data words using the BLEN control bits. The BLEN control bits are compared to the current value of the DCI buffer control unit address counter. When the 2 LS bits of the DCI address counter match the BLEN<1:0> value, the buffer control unit will be reset to '0'. In addition, the contents of the receive shadow registers are transferred to the receive buffer registers and the contents of the transmit buffer registers are transferred to the transmit shadow registers.

18.3.15 BUFFER ALIGNMENT WITH DATA FRAMES

There is no direct coupling between the position of the AGU address pointer and the data frame boundaries. This means that there will be an implied assignment of each transmit and receive buffer that is a function of the BLEN control bits and the number of enabled data slots via the TSE and RSE control bits.

As an example, assume that a 4-word data frame is chosen and that we want to transmit on all four time slots in the frame. This configuration would be established by setting the TSE0, TSE1, TSE2, and TSE3 control bits in the TSCON SFR. With this module setup, the TXBUF0 register would be naturally assigned to slot #0, the TXBUF1 register would be naturally assigned to slot #1, and so on.

Note: When more than four time slots are active within a data frame, the user code must keep track of which time slots are to be read/written at each interrupt. In some cases, the alignment between transmit/receive buffers and their respective slot assignments could be lost. Examples of such cases include an emulation breakpoint or a hardware trap. In these situations, the user should poll the SLOT status bits to determine what data should be loaded into the buffer registers to resynchronize the software with the DCI module.

18.3.16 TRANSMIT STATUS BITS

There are two transmit status bits in the DCISTAT SFR.

The TMPTY bit is set when the contents of the transmit buffer registers are transferred to the transmit shadow registers. The TMPTY bit may be polled in software to determine when the transmit buffer registers may be written. The TMPTY bit is cleared automatically by the hardware when a write to one of the four transmit buffers occurs.

The TUNF bit is read only and indicates that a transmit underflow has occurred for at least one of the transmit buffer registers that is in use. The TUNF bit is set at the time the transmit buffer registers are transferred to the transmit shadow registers. The TUNF status bit is cleared automatically when the buffer register that underflowed is written by the CPU.

Note: The transmit status bits only indicate status for buffer locations that are used by the module. If the buffer length is set to less than four words, for example, the unused buffer locations will not affect the transmit status bits.

18.3.17 RECEIVE STATUS BITS

There are two receive status bits in the DCISTAT SFR.

The RFUL status bit is read only and indicates that new data is available in the receive buffers. The RFUL bit is cleared automatically when all receive buffers in use have been read by the CPU.

The ROV status bit is read only and indicates that a receive overflow has occurred for at least one of the receive buffer locations. A receive overflow occurs when the buffer location is not read by the CPU before new data is transferred from the shadow registers. The ROV status bit is cleared automatically when the buffer register that caused the overflow is read by the CPU.

When a receive overflow occurs for a specific buffer location, the old contents of the buffer are overwritten.

Note: The receive status bits only indicate status for buffer locations that are used by the module. If the buffer length is set to less than four words, for example, the unused buffer locations will not affect the transmit status bits.

18.3.18 SLOT STATUS BITS

The SLOT<3:0> status bits in the DCISTAT SFR indicate the current active time slot. These bits will correspond to the value of the frame sync generator counter. The user may poll these status bits in software when a DCI interrupt occurs to determine what time slot data was last received and which time slot data should be loaded into the TXBUF registers.

18.3.19 CSDO MODE BIT

The CSDOM control bit controls the behavior of the CSDO pin during unused transmit slots. A given transmit time slot is unused if its corresponding TSEx bit in the TSCON SFR is cleared.

If the CSDOM bit is cleared (default), the CSDO pin will be low during unused time slot periods. This mode will be used when there are only two devices attached to the serial bus.

If the CSDOM bit is set, the CSDO pin will be tri-stated during unused time slot periods. This mode allows multiple devices to share the same CSDO line in a multi-channel application. Each device on the CSDO line is configured so that it will only transmit data during specific time slots. No two devices will transmit data during the same time slot.

18.3.20 DIGITAL LOOPBACK MODE

Digital Loopback mode is enabled by setting the DLOOP control bit in the DCISTAT SFR. When the DLOOP bit is set, the module internally connects the CSDO signal to CSDI. The actual data input on the CSDI I/O pin will be ignored in Digital Loopback mode.

18.3.21 UNDERFLOW MODE CONTROL BIT

When an underflow occurs, one of two actions may occur depending on the state of the Underflow mode (UNFM) control bit in the DCICON2 SFR. If the UNFM bit is cleared (default), the module will transmit '0's on the CSDO pin during the active time slot for the buffer location. In this Operating mode, the Codec device attached to the DCI module will simply be fed digital 'silence'. If the UNFM control bit is set, the module will transmit the last data written to the buffer location. This Operating mode permits the user to send continuous data to the Codec device without consuming CPU overhead.

18.4 DCI Module Interrupts

The frequency of DCI module interrupts is dependent on the BLEN<1:0> control bits in the DCICON2 SFR. An interrupt to the CPU is generated each time the set buffer length has been reached and a shadow register transfer takes place. A shadow register transfer is defined as the time when the previously written TXBUF values are transferred to the transmit shadow registers and new received values in the receive shadow registers are transferred into the RXBUF registers.

18.5 DCI Module Operation During CPU Sleep and Idle Modes

18.5.1 DCI MODULE OPERATION DURING CPU SLEEP MODE

The DCI module has the ability to operate while in Sleep mode and wake the CPU when the CSCK signal is supplied by an external device (CSCKD = 1). The DCI module will generate an asynchronous interrupt when a DCI buffer transfer has completed and the CPU is in Sleep mode.

18.5.2 DCI MODULE OPERATION DURING CPU IDLE MODE

If the DCISIDL control bit is cleared (default), the module will continue to operate normally even in Idle mode. If the DCISIDL bit is set, the module will halt when Idle mode is asserted.

18.6 AC-Link Mode Operation

The AC-Link protocol is a 256-bit frame with one 16-bit data slot, followed by twelve 20-bit data slots. The DCI module has two Operating modes for the AC-Link protocol. These Operating modes are selected by the COFSM<1:0> control bits in the DCICON1 SFR. The first AC-Link mode is called '16-bit AC-Link mode' and is selected by setting COFSM<1:0> = 10. The second AC-Link mode is called '20-bit AC-Link mode' and is selected by setting COFSM<1:0> = 11.

18.6.1 16-BIT AC-LINK MODE

In the 16-bit AC-Link mode, data word lengths are restricted to 16 bits. Note that this restriction only affects the 20-bit data time slots of the AC-Link protocol. For received time slots, the incoming data is simply truncated to 16 bits. For outgoing time slots, the 4 LS bits of the data word are set to '0' by the module. This truncation of the time slots limits the A/D and DAC data to 16 bits but permits proper data alignment in the TXBUF and RXBUF registers. Each RXBUF and TXBUF register will contain one data time slot value.

18.6.2 20-BIT AC-LINK MODE

The 20-bit AC-Link mode allows all bits in the data time slots to be transmitted and received but does not maintain data alignment in the TXBUF and RXBUF registers.

The 20-bit AC-Link mode functions similar to the Multi-Channel mode of the DCI module, except for the duty cycle of the frame synchronization signal. The AC-Link frame synchronization signal should remain high for 16 CSCK cycles and should be low for the following 240 cycles.

The 20-bit mode treats each 256-bit AC-Link frame as sixteen, 16-bit time slots. In the 20-bit AC-Link mode, the module operates as if COFSG<3:0> = 1111 and WS<3:0> = 1111. The data alignment for 20-bit data slots is ignored. For example, an entire AC-Link data frame can be transmitted and received in a packed fashion by setting all bits in the TSCON and RSCON SFRs. Since the total available buffer length is 64 bits, it would take 4 consecutive interrupts to transfer the AC-Link frame. The application software must keep track of the current AC-Link frame segment.

18.7 I²S Mode Operation

The DCI module is configured for I²S mode by writing a value of '01' to the COFSM<1:0> control bits in the DCICON1 SFR. When operating in the I²S mode, the DCI module will generate frame synchronization signals with a 50% duty cycle. Each edge of the frame synchronization signal marks the boundary of a new data word transfer.

The user must also select the frame length and data word size using the COFSG and WS control bits in the DCICON2 SFR.

18.7.1 I²S FRAME AND DATA WORD LENGTH SELECTION

The WS and COFSG control bits are set to produce the period for one half of an I²S data frame. That is, the frame length is the total number of CSCK cycles required for a left or a right data word transfer.

The BLEN bits must be set for the desired buffer length. Setting BLEN<1:0> = 01 will produce a CPU interrupt, once per I²S frame.

18.7.2 I²S DATA JUSTIFICATION

As per the I²S specification, a data word transfer will, by default, begin one CSCK cycle after a transition of the WS signal. A 'MS bit left justified' option can be selected using the DJST control bit in the DCICON2 SFR.

If DJST = 1, the I²S data transfers will be MS bit left justified. The MS bit of the data word will be presented on the CSDO pin during the same CSCK cycle as the rising or falling edge of the COFS signal. The CSDO pin is tri-stated after the data word has been sent.

TABLE 18-2: DCI 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 | Reset State |
|----------|-------|----------------------------------|--------|---------|--------|--------|--------|-------|-------|------------|-----------|-------|-------|-------|---------|--------|--------|---------------------|
| DCICON1 | 0240 | DCIEN | — | DCISIDL | — | DLOOP | CSCKD | CSCKE | COFSD | UNFM | CSDOM | DJUST | — | — | — | COFSM1 | COFSM0 | 0000 0000 0000 0000 |
| DCICON2 | 0242 | — | — | — | — | BLEN1 | BLENO | — | — | COFSG<3:0> | BCG<11:0> | — | — | — | WS<3:0> | — | — | 0000 0000 0000 0000 |
| DCICON3 | 0244 | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | 0000 0000 0000 0000 |
| DCISTAT | 0246 | — | — | — | — | SLOT3 | SLOT2 | SLOT1 | SLOT0 | — | — | — | — | ROV | RFUL | TUNF | TMPTY | 0000 0000 0000 0000 |
| TSCON | 0248 | TSE15 | TSE14 | TSE13 | TSE12 | TSE11 | TSE10 | TSE9 | TSE8 | TSE7 | TSE6 | TSE5 | TSE4 | TSE3 | TSE2 | TSE1 | TSE0 | 0000 0000 0000 0000 |
| RSCON | 024C | RSE15 | RSE14 | RSE13 | RSE12 | RSE11 | RSE10 | RSE9 | RSE8 | RSE7 | RSE6 | RSE5 | RSE4 | RSE3 | RSE2 | RSE1 | RSE0 | 0000 0000 0000 0000 |
| RXBUF0 | 0250 | Receive Buffer #0 Data Register | | | | | | | | | | | | | | | | |
| RXBUF1 | 0252 | Receive Buffer #1 Data Register | | | | | | | | | | | | | | | | |
| RXBUF2 | 0254 | Receive Buffer #2 Data Register | | | | | | | | | | | | | | | | |
| RXBUF3 | 0256 | Receive Buffer #3 Data Register | | | | | | | | | | | | | | | | |
| TXBUF0 | 0258 | Transmit Buffer #0 Data Register | | | | | | | | | | | | | | | | |
| TXBUF1 | 025A | Transmit Buffer #1 Data Register | | | | | | | | | | | | | | | | |
| TXBUF2 | 025C | Transmit Buffer #2 Data Register | | | | | | | | | | | | | | | | |
| TXBUF3 | 025E | Transmit Buffer #3 Data Register | | | | | | | | | | | | | | | | |

Legend: u = uninitialized bit

Note: Refer to *dsPIC30F Family Reference Manual (DS70046)* for descriptions of register bit fields.

dsPIC30F

NOTES:

19.0 12-BIT ANALOG-TO-DIGITAL CONVERTER (A/D) MODULE

Note: This data sheet summarizes features of this group of dsPIC30F devices and is not intended to be a complete reference source. For more information on the CPU, peripherals, register descriptions and general device functionality, refer to the *dsPIC30F Family Reference Manual* (DS70046).

The 12-bit Analog-to-Digital converter (A/D) allows conversion of an analog input signal to a 12-bit digital number. This module is based on a Successive Approximation Register (SAR) architecture and provides a maximum sampling rate of 100 ksp/s. The A/D module has up to 16 analog inputs which are multiplexed into a sample and hold amplifier. The output of the sample and hold is the input into the converter which generates the result. The analog reference voltage is software selectable to either the device supply voltage (AVDD/AVSS) or the voltage level on the (VREF+/VREF-) pin. The A/D converter has a unique feature of being able to operate while the device is in Sleep mode with RC oscillator selection.

The A/D module has six 16-bit registers:

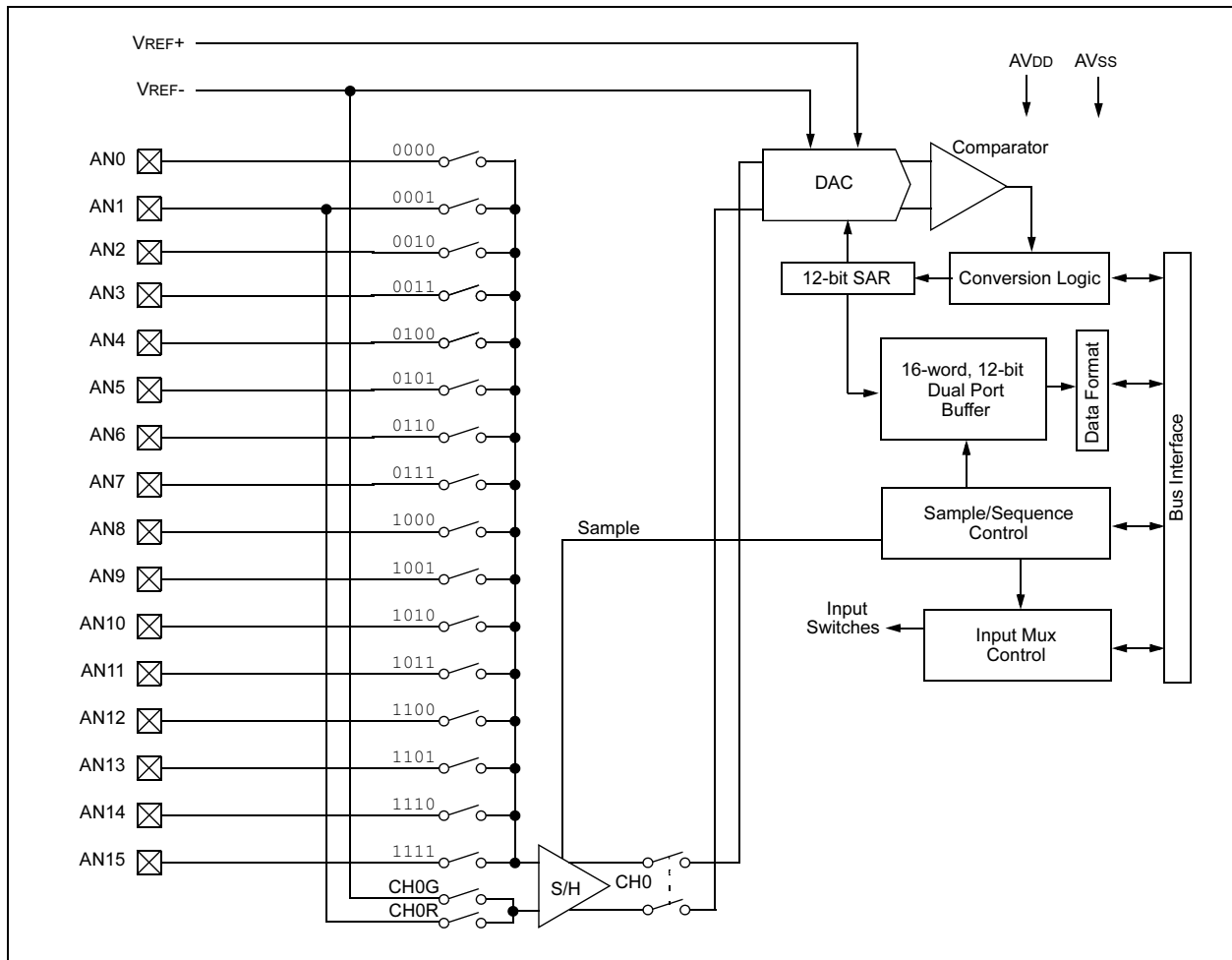
- A/D Control Register 1 (ADCON1)
- A/D Control Register 2 (ADCON2)
- A/D Control Register 3 (ADCON3)
- A/D Input Select Register (ADCHS)
- A/D Port Configuration Register (ADPCFG)
- A/D Input Scan Selection Register (ADCSSL)

The ADCON1, ADCON2 and ADCON3 registers control the operation of the A/D module. The ADCHS register selects the input channels to be converted. The ADPCFG register configures the port pins as analog inputs or as digital I/O. The ADCSSL register selects inputs for scanning.

Note: The SSRC<2:0>, ASAM, SMPI<3:0>, BUFM and ALTS bits, as well as the ADCON3 and ADCSSL registers, must not be written to while ADON = 1. This would lead to indeterminate results.

The block diagram of the 12-bit A/D module is shown in Figure 19-1.

FIGURE 19-1: 12-BIT A/D FUNCTIONAL BLOCK DIAGRAM



19.1 A/D Result Buffer

The module contains a 16-word dual port read only buffer, called ADCBUF0...ADCBUFF, to buffer the A/D results. The RAM is 12 bits wide but the data obtained is represented in one of four different 16-bit data formats. The contents of the sixteen A/D Conversion Result Buffer registers, ADCBUF0 through ADCBUFF, cannot be written by user software.

19.2 Conversion Operation

After the A/D module has been configured, the sample acquisition is started by setting the SAMP bit. Various sources, such as a programmable bit, timer time-outs and external events, will terminate acquisition and start a conversion. When the A/D conversion is complete, the result is loaded into ADCBUF0...ADCBUFF, and the DONE bit and the A/D interrupt flag ADIF are set after the number of samples specified by the SMPI bit. The ADC module can be configured for different interrupt rates as described in Section 19.3.

The following steps should be followed for doing an A/D conversion:

1. Configure the A/D module:
 - Configure analog pins, voltage reference and digital I/O
 - Select A/D input channels
 - Select A/D conversion clock
 - Select A/D conversion trigger
 - Turn on A/D module
2. Configure A/D interrupt (if required):
 - Clear ADIF bit
 - Select A/D interrupt priority
3. Start sampling.
4. Wait the required acquisition time.
5. Trigger acquisition end, start conversion:
6. Wait for A/D conversion to complete, by either:
 - Waiting for the A/D interrupt, or
 - Waiting for the DONE bit to get set.
7. Read A/D result buffer, clear ADIF if required.

19.3 Selecting the Conversion Sequence

Several groups of control bits select the sequence in which the A/D connects inputs to the sample/hold channel, converts a channel, writes the buffer memory and generates interrupts.

The sequence is controlled by the sampling clocks.

The SMPI bits select the number of acquisition/conversion sequences that would be performed before an interrupt occurs. This can vary from 1 sample per interrupt to 16 samples per interrupt.

The BUFM bit will split the 16-word results buffer into two 8-word groups. Writing to the 8-word buffers will be alternated on each interrupt event.

Use of the BUFM bit will depend on how much time is available for the moving of the buffers after the interrupt.

If the processor can quickly unload a full buffer within the time it takes to acquire and convert one channel, the BUFM bit can be '0' and up to 16 conversions (corresponding to the 16 input channels) may be done per interrupt. The processor will have one acquisition and conversion time to move the sixteen conversions.

If the processor cannot unload the buffer within the acquisition and conversion time, the BUFM bit should be '1'. For example, if $SMPI<3:0> (ADCON2<5:2>) = 0111$, then eight conversions will be loaded into 1/2 of the buffer, following which an interrupt occurs. The next eight conversions will be loaded into the other 1/2 of the buffer. The processor will have the entire time between interrupts to move the eight conversions.

The ALTS bit can be used to alternate the inputs selected during the sampling sequence. The input multiplexer has two sets of sample inputs: MUX A and MUX B. If the ALTS bit is '0', only the MUX A inputs are selected for sampling. If the ALTS bit is '1' and $SMPI<3:0> = 0000$ on the first sample/convert sequence, the MUX A inputs are selected and on the next acquire/convert sequence, the MUX B inputs are selected.

The CSCNA bit ($ADCON2<10>$) will allow the multiplexer input to be alternately scanned across a selected number of analog inputs for the MUX A group. The inputs are selected by the ADCSSL register. If a particular bit in the ADCSSL register is '1', the corresponding input is selected. The inputs are always scanned from lower to higher numbered inputs, starting after each interrupt. If the number of inputs selected is greater than the number of samples taken per interrupt, the higher numbered inputs are unused.

19.4 Programming the Start of Conversion Trigger

The conversion trigger will terminate acquisition and start the requested conversions.

The SSRC<2:0> bits select the source of the conversion trigger. The SSRC bits provide for up to 4 alternate sources of conversion trigger.

When SSRC<2:0> = 000, the conversion trigger is under software control. Clearing the SAMP bit will cause the conversion trigger.

When SSRC<2:0> = 111 (Auto-Start mode), the conversion trigger is under A/D clock control. The SAMC bits select the number of A/D clocks between the start of acquisition and the start of conversion. This provides the fastest conversion rates on multiple channels. SAMC must always be at least 1 clock cycle.

Other trigger sources can come from timer modules or external interrupts.

19.5 Aborting a Conversion

Clearing the ADON bit during a conversion will abort the current conversion and stop the sampling sequencing until the next sampling trigger. The ADCBUF will not be updated with the partially completed A/D conversion sample. That is, the ADCBUF will continue to contain the value of the last completed conversion (or the last value written to the ADCBUF register).

If the clearing of the ADON bit coincides with an auto-start, the clearing has a higher priority and a new conversion will not start.

After the A/D conversion is aborted, a 2 TAD wait is required before the next sampling may be started by setting the SAMP bit.

19.6 Selecting the A/D Conversion Clock

The A/D conversion requires 15 TAD. The source of the A/D conversion clock is software selected, using a six-bit counter. There are 64 possible options for TAD.

EQUATION 19-1: A/D CONVERSION CLOCK

$$TAD = T_{CY} * (0.5 * (ADCS<5:0> + 1))$$

The internal RC oscillator is selected by setting the ADRC bit.

For correct A/D conversions, the A/D conversion clock (TAD) must be selected to ensure a minimum TAD time of 667 nsec (for VDD = 5V). Refer to the Electrical Specifications section for minimum TAD under other operating conditions.

Example 19-1 shows a sample calculation for the ADCS<5:0> bits, assuming a device operating speed of 30 MIPS.

EXAMPLE 19-1: A/D CONVERSION CLOCK CALCULATION

$$\begin{aligned} \text{Minimum TAD} &= 667 \text{ nsec} \\ T_{CY} &= 33 \text{ nsec (30 MIPS)} \end{aligned}$$

$$\begin{aligned} ADCS<5:0> &= 2 \frac{TAD}{T_{CY}} - 1 \\ &= 2 \cdot \frac{667 \text{ nsec}}{33 \text{ nsec}} - 1 \\ &= 39.4 \end{aligned}$$

Therefore,
Set ADCS<5:0> = 40

$$\begin{aligned} \text{Actual TAD} &= \frac{T_{CY}}{2} (ADCS<5:0> + 1) \\ &= \frac{33 \text{ nsec}}{2} (40 + 1) \\ &= 677 \text{ nsec} \end{aligned}$$

dsPIC30F

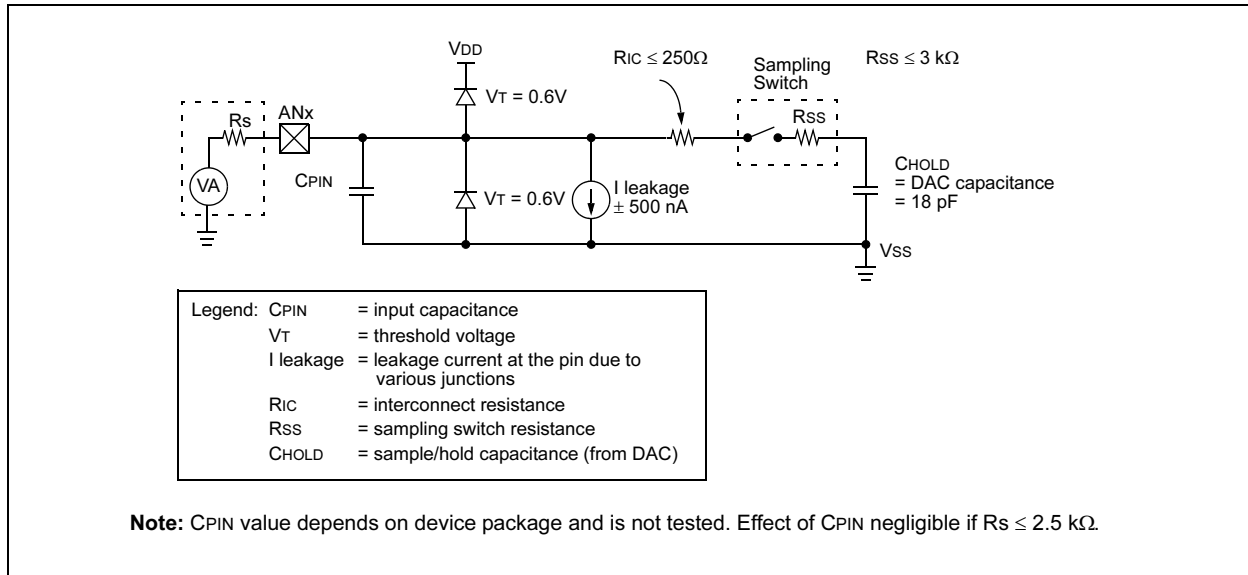
19.7 A/D Acquisition Requirements

The analog input model of the 12-bit A/D converter is shown in Figure 19-2. The total sampling time for the A/D is a function of the internal amplifier settling time and the holding capacitor charge time.

For the A/D converter to meet its specified accuracy, the charge holding capacitor (CHOLD) must be allowed to fully charge to the voltage level on the analog input pin. The source impedance (R_s), the interconnect impedance (R_{IC}), and the internal sampling switch (R_{SS}) impedance combine to directly affect the time

required to charge the capacitor CHOLD. The combined impedance of the analog sources must therefore be small enough to fully charge the holding capacitor within the chosen sample time. To minimize the effects of pin leakage currents on the accuracy of the A/D converter, the maximum recommended source impedance, R_s , is $2.5\text{ k}\Omega$. After the analog input channel is selected (changed), this sampling function must be completed prior to starting the conversion. The internal holding capacitor will be in a discharged state prior to each sample operation.

FIGURE 19-2: 12-BIT A/D CONVERTER ANALOG INPUT MODEL



19.8 Module Power-down Modes

The module has 2 internal Power modes.

When the ADON bit is '1', the module is in Active mode; it is fully powered and functional.

When ADON is '0', the module is in Off mode. The digital and analog portions of the circuit are disabled for maximum current savings.

In order to return to the Active mode from Off mode, the user must wait for the ADC circuitry to stabilize.

19.9 A/D Operation During CPU Sleep and Idle Modes

19.9.1 A/D OPERATION DURING CPU SLEEP MODE

When the device enters Sleep mode, all clock sources to the module are shutdown and stay at logic '0'.

If Sleep occurs in the middle of a conversion, the conversion is aborted. The converter will not continue with a partially completed conversion on exit from Sleep mode.

Register contents are not affected by the device entering or leaving Sleep mode.

The A/D module can operate during Sleep mode if the A/D clock source is set to RC (ADRC = 1). When the RC clock source is selected, the A/D module waits one instruction cycle before starting the conversion. This allows the SLEEP instruction to be executed which eliminates all digital switching noise from the conversion. When the conversion is complete, the CONV bit will be cleared and the result loaded into the ADCBUF register.

If the A/D interrupt is enabled, the device will wake-up from Sleep. If the A/D interrupt is not enabled, the A/D module will then be turned off, although the ADON bit will remain set.

19.9.2 A/D OPERATION DURING CPU IDLE MODE

The ADSIDL bit selects if the module will stop on Idle or continue on Idle. If ADSIDL = 0, the module will continue operation on assertion of Idle mode. If ADSIDL = 1, the module will stop on Idle.

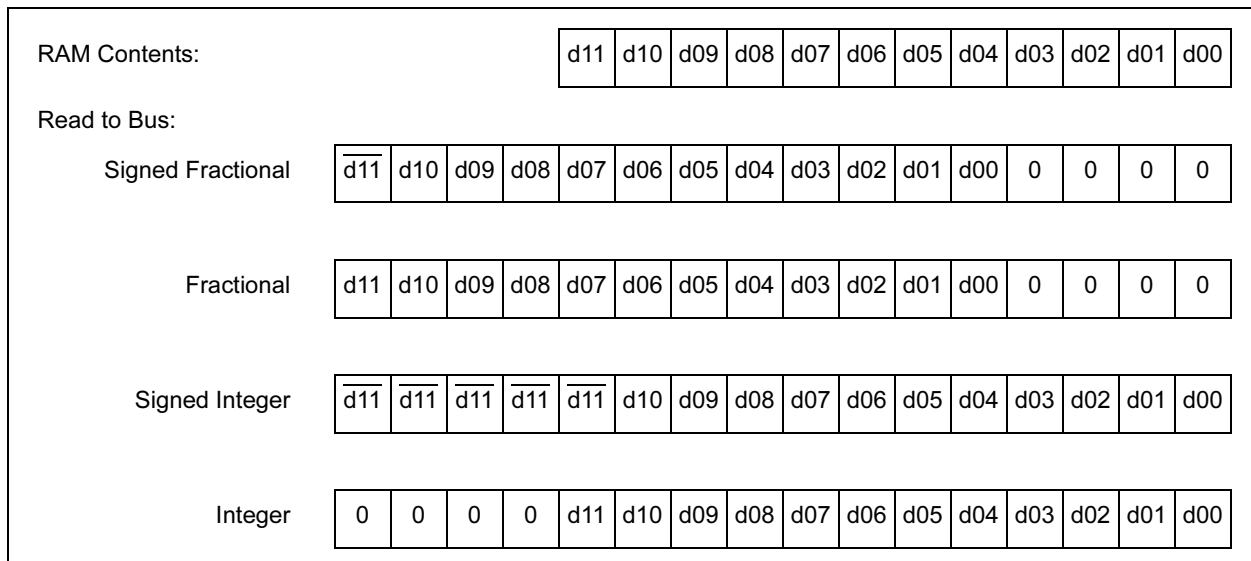
19.10 Effects of a Reset

A device Reset forces all registers to their Reset state. This forces the A/D module to be turned off, and any conversion and sampling sequence is aborted. The values that are in the ADCBUF registers are not modified. The A/D Result register will contain unknown data after a Power-on Reset.

19.11 Output Formats

The A/D result is 12 bits wide. The data buffer RAM is also 12 bits wide. The 12-bit data can be read in one of four different formats. The FORM<1:0> bits select the format. Each of the output formats translates to a 16-bit result on the data bus. Write data will always be in right justified (integer) format.

FIGURE 19-3: A/D OUTPUT DATA FORMATS



19.12 Configuring Analog Port Pins

The use of the ADPCFG and TRIS registers control the operation of the A/D port pins. The port pins that are desired as analog inputs must have their corresponding TRIS bit set (input). If the TRIS bit is cleared (output), the digital output level (VOH or VOL) will be converted.

The A/D operation is independent of the state of the CH0SA<3:0>/CH0SB<3:0> bits and the TRIS bits.

When reading the Port register, all pins configured as analog input channels will read as cleared.

Pins configured as digital inputs will not convert an analog input. Analog levels on any pin that is defined as a digital input (including the ANx pins) may cause the input buffer to consume current that exceeds the device specifications.

19.13 Connection Considerations

The analog inputs have diodes to VDD and VSS as ESD protection. This requires that the analog input be between VDD and VSS. If the input voltage exceeds this range by greater than 0.3V (either direction), one of the diodes becomes forward biased and it may damage the device if the input current specification is exceeded.

An external RC filter is sometimes added for anti-aliasing of the input signal. The R component should be selected to ensure that the sampling time requirements are satisfied. Any external components connected (via high impedance) to an analog input pin (capacitor, zener diode, etc.) should have very little leakage current at the pin.

TABLE 19-1: A/D CONVERTER 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 | Reset State |
|----------|-------|--------|--------|--------|--------|--------|--------|-----------|-------|-------|-----------|-------|-------|-------|-------|-------|-------|---------------------|
| ADCBUF0 | 0280 | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | 0000 0000 0000 0000 |
| ADCBUF1 | 0282 | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | 0000 0000 0000 0000 |
| ADCBUF2 | 0284 | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | 0000 0000 0000 0000 |
| ADCBUF3 | 0286 | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | 0000 0000 0000 0000 |
| ADCBUF4 | 0288 | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | 0000 0000 0000 0000 |
| ADCBUF5 | 028A | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | 0000 0000 0000 0000 |
| ADCBUF6 | 028C | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | 0000 0000 0000 0000 |
| ADCBUF7 | 028E | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | 0000 0000 0000 0000 |
| ADCBUF8 | 0290 | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | 0000 0000 0000 0000 |
| ADCBUF9 | 0292 | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | 0000 0000 0000 0000 |
| ADCBUFA | 0294 | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | 0000 0000 0000 0000 |
| ADCBUFB | 0296 | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | 0000 0000 0000 0000 |
| ADCBUFC | 0298 | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | 0000 0000 0000 0000 |
| ADCBUFD | 029A | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | 0000 0000 0000 0000 |
| ADCBUFE | 029C | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | 0000 0000 0000 0000 |
| ADCBUFF | 029E | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | 0000 0000 0000 0000 |
| ADCON1 | 02A0 | ADON | — | ADSIDL | — | — | — | FORM<1:0> | — | — | SSRC<2:0> | — | — | — | ASAM | SAMP | DONE | 0000 0000 0000 0000 |
| ADCON2 | 02A2 | — | — | — | — | — | CSCNA | — | — | — | — | — | — | — | — | — | — | 0000 0000 0000 0000 |
| ADCON3 | 02A4 | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | 0000 0000 0000 0000 |
| ADCHS | 02A6 | — | — | — | CH0NB | — | — | — | — | — | — | — | — | — | — | — | — | 0000 0000 0000 0000 |
| ADPCFG | 02A8 | PCFG15 | PCFG14 | PCFG13 | PCFG12 | PCFG11 | PCFG10 | PCFG9 | PCFG8 | PCFG7 | PCFG6 | PCFG5 | PCFG4 | PCFG3 | PCFG2 | PCFG1 | PCFG0 | 0000 0000 0000 0000 |
| ADCSSL | 02AA | CSSL15 | CSSL14 | CSSL13 | CSSL12 | CSSL11 | CSSL10 | CSSL9 | CSSL8 | CSSL7 | CSSL6 | CSSL5 | CSSL4 | CSSL3 | CSSL2 | CSSL1 | CSSL0 | 0000 0000 0000 0000 |

Legend: u = uninitialized bit

Note: Refer to dsPIC30F Family Reference Manual (DS70046) for descriptions of register bit fields.

dsPIC30F

NOTES:

20.0 SYSTEM INTEGRATION

Note: This data sheet summarizes features of this group of dsPIC30F devices and is not intended to be a complete reference source. For more information on the CPU, peripherals, register descriptions and general device functionality, refer to the *dsPIC30F Family Reference Manual* (DS70046). For more information on the device instruction set and programming, refer to the *dsPIC30F Programmer's Reference Manual* (DS70030).

There are several features intended to maximize system reliability, minimize cost through elimination of external components, provide Power Saving Operating modes and offer code protection:

- Oscillator Selection
- Reset
 - Power-on Reset (POR)
 - Power-up Timer (PWRT)
 - Oscillator Start-up Timer (OST)
 - Programmable Brown-out Reset (BOR)
- Watchdog Timer (WDT)
- Power Saving Modes (Sleep and Idle)
- Code Protection
- Unit ID Locations
- In-Circuit Serial Programming (ICSP)

dsPIC30F devices have a Watchdog Timer which is permanently enabled via the configuration bits or can be software controlled. It runs off its own RC oscillator for added reliability. There are two timers that offer necessary delays on power-up. One is the Oscillator Start-up Timer (OST), intended to keep the chip in Reset until the crystal oscillator is stable. The other is the Power-up Timer (PWRT) which provides a delay on power-up only, designed to keep the part in Reset while the power supply stabilizes. With these two timers on-chip, most applications need no external Reset circuitry.

Sleep mode is designed to offer a very low current Power-down mode. The user can wake-up from Sleep through external Reset, Watchdog Timer Wake-up, or through an interrupt. Several oscillator options are also made available to allow the part to fit a wide variety of applications. In the Idle mode, the clock sources are still active but the CPU is shut-off. The RC oscillator option saves system cost while the LP crystal option saves power.

20.1 Oscillator System Overview

The dsPIC30F oscillator system has the following modules and features:

- Various external and internal oscillator options as clock sources
- An on-chip PLL to boost internal operating frequency
- A clock switching mechanism 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
- Clock Control register (OSCCON)
- Configuration bits for main oscillator selection

Configuration bits determine the clock source upon Power-on Reset (POR) and Brown-out Reset (BOR). Thereafter, the clock source can be changed between permissible clock sources. The OSCCON register controls the clock switching and reflects system clock related status bits.

Table 20-1 provides a summary of the dsPIC30F Oscillator Operating modes. A simplified diagram of the oscillator system is shown in Figure 20-1.

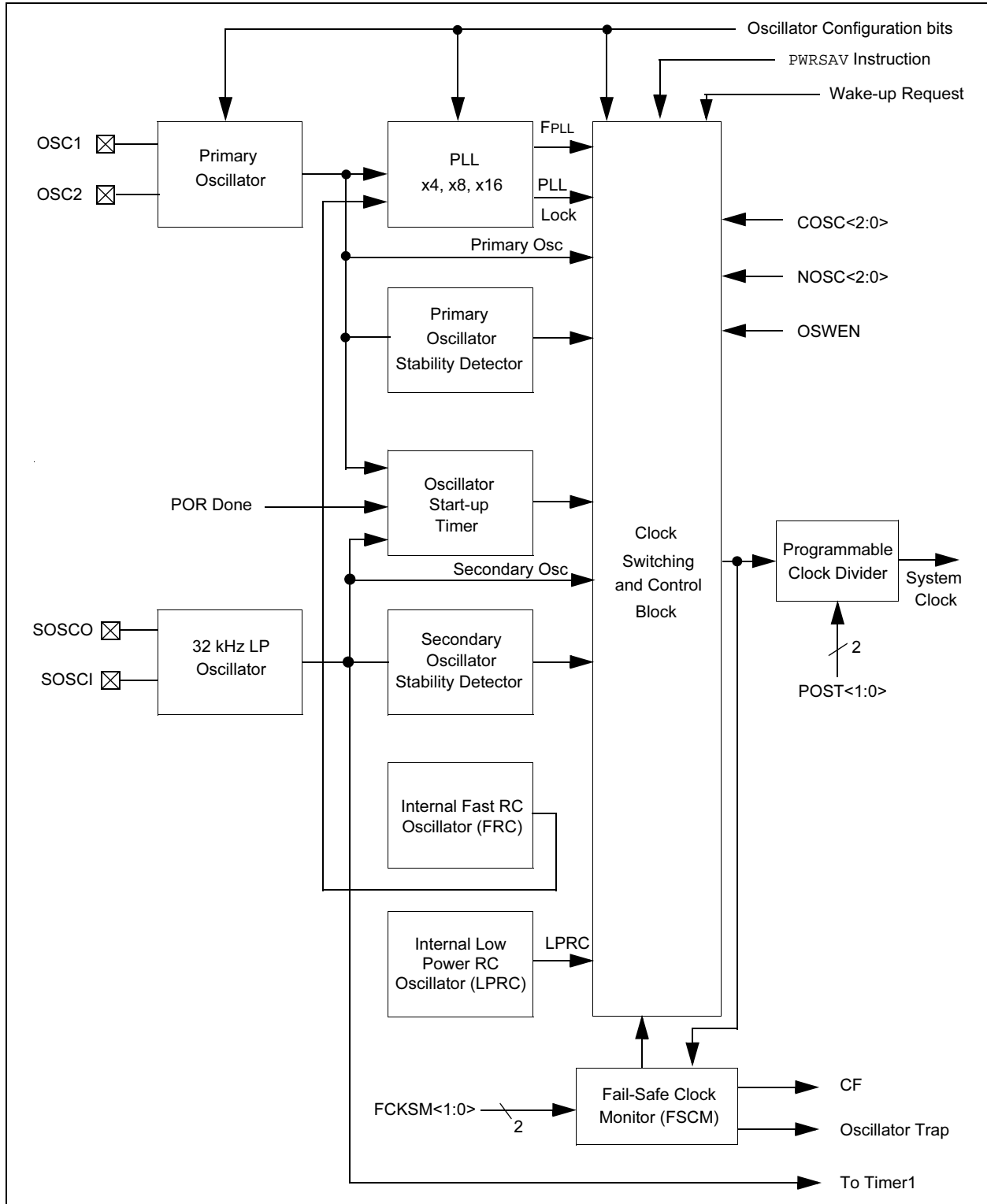
dsPIC30F

TABLE 20-1: OSCILLATOR OPERATING MODES

| Oscillator Mode | Description |
|-----------------|--|
| XTL | 200 kHz-4 MHz crystal on OSC1:OSC2. |
| XT | 4 MHz-10 MHz crystal on OSC1:OSC2. |
| XT w/ PLL 4x | 4 MHz-10 MHz crystal on OSC1:OSC2, 4x PLL enabled. |
| XT w/ PLL 8x | 4 MHz-10 MHz crystal on OSC1:OSC2, 8x PLL enabled. |
| XT w/ PLL 16x | 4 MHz-10 MHz crystal on OSC1:OSC2, 16x PLL enabled ⁽¹⁾ . |
| LP | 32 kHz crystal on SOSCO:SOSCI ⁽²⁾ . |
| HS | 10 MHz-25 MHz crystal. |
| HS/2 w/ PLL 4x | 10 MHz-25 MHz crystal, divide by 2, 4x PLL enabled. |
| HS/2 w/ PLL 8x | 10 MHz-25 MHz crystal, divide by 2, 8x PLL enabled. |
| HS/2 w/ PLL 16x | 10 MHz-25 MHz crystal, divide by 2, 16x PLL enabled. |
| HS/3 w/ PLL 4x | 10 MHz-25 MHz crystal, divide by 3, 4x PLL enabled. |
| HS/3 w/ PLL 8x | 10 MHz-25 MHz crystal, divide by 3, 8x PLL enabled. |
| HS/3 w/ PLL 16x | 10 MHz-25 MHz crystal, divide by 3, 16x PLL enabled. |
| EC | External clock input (0-40 MHz). |
| ECIO | External clock input (0-40 MHz), OSC2 pin is I/O. |
| EC w/ PLL 4x | External clock input (0-40 MHz), OSC2 pin is I/O, 4x PLL enabled ⁽¹⁾ . |
| EC w/ PLL 8x | External clock input (0-40 MHz), OSC2 pin is I/O, 8x PLL enabled ⁽¹⁾ . |
| EC w/ PLL 16x | External clock input (0-40 MHz), OSC2 pin is I/O, 16x PLL enabled ⁽¹⁾ . |
| ERC | External RC oscillator, OSC2 pin is Fosc/4 output ⁽³⁾ . |
| ERCIO | External RC oscillator, OSC2 pin is I/O ⁽³⁾ . |
| FRC | 8 MHz internal RC oscillator. |
| FRC w/ PLL 4x | 8 MHz Internal RC oscillator, 4x PLL enabled. |
| FRC w/ PLL 8x | 8 MHz Internal RC oscillator, 8x PLL enabled. |
| FRC w/ PLL 16x | 7.5 MHz Internal RC oscillator, 16x PLL enabled. |
| LPRC | 512 kHz internal RC oscillator. |

- Note 1:** dsPIC30F maximum operating frequency of 120 MHz must be met.
- 2:** LP oscillator can be conveniently shared as system clock, as well as real-time clock for Timer1.
- 3:** Requires external R and C. Frequency operation up to 4 MHz.
- 4:** Some devices support a subset of the above modes. Refer to individual device data sheets for details.

FIGURE 20-1: OSCILLATOR SYSTEM BLOCK DIAGRAM



dsPIC30F

20.2 Oscillator Configurations

The selection is as shown in Table 20-2.

20.2.1 INITIAL CLOCK SOURCE SELECTION

While coming out of Power-on Reset or Brown-out Reset, the device selects its clock source based on:

- FOS<2:0> configuration bits that select one of four oscillator groups,
- and FPR<4:0> configuration bits that select one of 13 oscillator choices within the primary group.

Note: Some devices may have different FOS and FPR values from Table 20-2, depending on the oscillator modes supported. Refer to individual device data sheets for details.

TABLE 20-2: CONFIGURATION BIT VALUES FOR CLOCK SELECTION

| Oscillator Mode | Oscillator Source | FOS<2:0> | | | FPR<4:0> | | | | | OSC2 Function |
|-----------------|-------------------|----------|---|---|----------|---|---|---|---|---------------|
| | | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 1 | |
| ECIO w/ PLL 4x | PLL | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 1 | I/O |
| ECIO w/ PLL 8x | PLL | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 0 | I/O |
| ECIO w/ PLL 16x | PLL | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | I/O |
| FRC w/ PLL 4x | PLL | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | I/O |
| FRC w/ PLL 8x | PLL | 1 | 1 | 1 | 0 | 1 | 0 | 1 | 0 | I/O |
| FRC w/ PLL 16x | PLL | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 1 | I/O |
| XT w/ PLL 4x | PLL | 1 | 1 | 1 | 0 | 0 | 1 | 0 | 1 | OSC2 |
| XT w/ PLL 8x | PLL | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | OSC2 |
| XT w/ PLL 16x | PLL | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | OSC2 |
| HS2 w/ PLL 4x | PLL | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 1 | OSC2 |
| HS2 w/ PLL 8x | PLL | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 0 | OSC2 |
| HS2 w/ PLL 16x | PLL | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | OSC2 |
| HS3 w/ PLL 4x | PLL | 1 | 1 | 1 | 1 | 0 | 1 | 0 | 1 | OSC2 |
| HS3 w/ PLL 8x | PLL | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 0 | OSC2 |
| HS3 w/ PLL 16x | PLL | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | OSC2 |
| ECIO | External | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | I/O |
| XT | External | 0 | 1 | 1 | 0 | 0 | 1 | 0 | 0 | OSC2 |
| HS | External | 0 | 1 | 1 | 0 | 0 | 0 | 1 | 0 | OSC2 |
| EC | External | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | CLKOUT |
| ERC | External | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 1 | CLKOUT |
| ERCIO | External | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | I/O |
| XTL | External | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | OSC2 |
| LP | Secondary | 0 | 0 | 0 | X | X | X | X | X | (Notes 1, 2) |
| FRC | Internal FRC | 0 | 0 | 1 | X | X | X | X | X | (Notes 1, 2) |
| LPRC | Internal LPRC | 0 | 1 | 0 | X | X | X | X | X | (Notes 1, 2) |

Note 1: OSC2 pin function is determined by (FPR<4:0>).

Note 2: Note that OSC1 pin cannot be used as an I/O pin even if the secondary oscillator or an internal clock source is selected at all times.

20.2.2 OSCILLATOR START-UP TIMER (OST)

In order to ensure that a crystal oscillator (or ceramic resonator) has started and stabilized, an Oscillator Start-up Timer is included. It is a simple 10-bit counter that counts 1024 TOSC cycles before releasing the oscillator clock to the rest of the system. The time-out period is designated as TOST. The TOST time is involved every time the oscillator has to restart (i.e., on POR, BOR and wake-up from Sleep). The Oscillator Start-up Timer is applied to the LP oscillator, XT, XTL, and HS modes (upon wake-up from Sleep, POR and BOR) for the primary oscillator.

20.2.3 LP OSCILLATOR CONTROL

Enabling the LP oscillator is controlled with two elements:

1. The current oscillator group bits COSC<2:0>.
2. The LPOSCEN bit (OSCON register).

The LP oscillator is on (even during Sleep mode) if LPOSCEN = 1. The LP oscillator is the device clock if:

- COSC<2:0> = 00 (LP selected as main oscillator) and
- LPOSCEN = 1

Keeping the LP oscillator on at all times allows for a fast switch to the 32 kHz system clock for lower power operation. Returning to the faster main oscillator will still require a start-up time

20.2.4 PHASE LOCKED LOOP (PLL)

The PLL multiplies the clock which is generated by the primary oscillator. The PLL is selectable to have either gains of x4, x8, and x16. Input and output frequency ranges are summarized in Table 20-3.

TABLE 20-3: PLL FREQUENCY RANGE

| F _{IN} | PLL Multiplier | F _{OUT} |
|-----------------|----------------|------------------|
| 4 MHz-10 MHz | x4 | 16 MHz-40 MHz |
| 4 MHz-10 MHz | x8 | 32 MHz-80 MHz |
| 4 MHz-7.5 MHz | x16 | 64 MHz-160 MHz |

The PLL features a lock output which is asserted when the PLL enters a phase locked state. Should the loop fall out of lock (e.g., due to noise), the lock signal will be rescinded. The state of this signal is reflected in the read only LOCK bit in the OSCCON register.

20.2.5 FAST RC OSCILLATOR (FRC)

The FRC oscillator is a fast (8 MHz nominal) internal RC oscillator. This oscillator is intended to provide reasonable device operating speeds without the use of an external crystal, ceramic resonator, or RC network. The FRC oscillator can be used with the PLL to obtain higher clock frequencies.

The dsPIC30F operates from the FRC oscillator whenever the current oscillator selection control bits in the OSCCON register (OSCCON<14:12>) are set to '001'.

The four bit field specified by TUN<3:0> (OSCON<15:14> and OSCON<11:10>) allows the user to tune the internal fast RC oscillator (nominal 8.0 MHz). The user can tune the FRC oscillator within a range of +10.5% (840 kHz) and -12% (960 kHz) in steps of 1.50% around the factory-calibrated setting, see Table 20-4.

If OSCCON<14:12> are set to '111' and FPR<4:0> are set to '00101', '00110' or '00111', then a PLL multiplier of 4, 8 or 16 (respectively) is applied.

Note: When a 16x PLL is used, the FRC frequency must not be tuned to a frequency greater than 7.5 MHz.

TABLE 20-4: FRC TUNING

| TUN<3:0> Bits | FRC Frequency |
|---------------|--|
| 0111 | + 10.5% |
| 0110 | + 9.0% |
| 0101 | + 7.5% |
| 0100 | + 6.0% |
| 0011 | + 4.5% |
| 0010 | + 3.0% |
| 0001 | + 1.5% |
| 0000 | Center Frequency (oscillator is running at calibrated frequency) |
| 1111 | - 1.5% |
| 1110 | - 3.0% |
| 1101 | - 4.5% |
| 1100 | - 6.0% |
| 1011 | - 7.5% |
| 1010 | - 9.0% |
| 1001 | - 10.5% |
| 1000 | - 12.0% |

20.2.6 LOW POWER RC OSCILLATOR (LPRC)

The LPRC oscillator is a component of the Watchdog Timer (WDT) and oscillates at a nominal frequency of 512 kHz. The LPRC oscillator is the clock source for the Power-up Timer (PWRT) circuit, WDT, and clock monitor circuits. It may also be used to provide a low frequency clock source option for applications where power consumption is critical and timing accuracy is not required.

The LPRC oscillator is always enabled at a Power-on Reset because it is the clock source for the PWRT. After the PWRT expires, the LPRC oscillator will remain on if one of the following is TRUE:

- The Fail-Safe Clock Monitor is enabled
- The WDT is enabled
- The LPRC oscillator is selected as the system clock via the COSC<2:0> control bits in the OSCCON register

If one of the above conditions is not true, the LPRC will shut-off after the PWRT expires.

Note 1: OSC2 pin function is determined by the Primary Oscillator mode selection (FPR<4:0>).

2: OSC1 pin cannot be used as an I/O pin even if the secondary oscillator or an internal clock source is selected at all times.

20.2.7 FAIL-SAFE CLOCK MONITOR

The Fail-Safe Clock Monitor (FSCM) allows the device to continue to operate even in the event of an oscillator failure. The FSCM function is enabled by appropriately programming the FCKSM configuration bits (clock switch and monitor selection bits) in the FOSC Device Configuration register. If the FSCM function is enabled, the LPRC internal oscillator will run at all times (except during Sleep mode) and will not be subject to control by the SWDTEN bit.

In the event of an oscillator failure, the FSCM will generate a clock failure trap event and will switch the system clock over to the FRC oscillator. The user will then have the option to either attempt to restart the oscillator or execute a controlled shutdown. The user may decide to treat the trap as a warm Reset by simply loading the Reset address into the oscillator fail trap vector. In this event, the CF (Clock Fail) status bit (OSCCON<3>) is also set whenever a clock failure is recognized.

In the event of a clock failure, the WDT is unaffected and continues to run on the LPRC clock.

If the oscillator has a very slow start-up time coming out of POR, BOR or Sleep, it is possible that the PWRT timer will expire before the oscillator has started. In such cases, the FSCM will be activated and the FSCM will initiate a clock failure trap, and the COSC<2:0> bits are loaded with FRC oscillator selection. This will effectively shut-off the original oscillator that was trying to start.

The user may detect this situation and restart the oscillator in the clock fail trap ISR.

Upon a clock failure detection, the FSCM module will initiate a clock switch to the FRC oscillator as follows:

1. The COSC bits (OSCCON<14:12>) are loaded with the FRC oscillator selection value.
2. CF bit is set (OSCCON<3>).
3. OSWEN control bit (OSCCON<0>) is cleared.

For the purpose of clock switching, the clock sources are sectioned into four groups:

1. Primary
2. Secondary
3. Internal FRC
4. Internal LPRC

The user can switch between these functional groups but cannot switch between options within a group. If the primary group is selected, then the choice within the group is always determined by the FPR<4:0> configuration bits.

The OSCCON register holds the control and status bits related to clock switching.

- COSC<1:0>: Read only status bits always reflect the current oscillator group in effect.
- NOSC<2:0>: Control bits which are written to indicate the new oscillator group of choice.
 - On POR and BOR, COSC<2:0> and NOSC<1:0> are both loaded with the configuration bit values FOS<2:0>.
- LOCK: The LOCK status bit indicates a PLL lock.
- CF: Read only status bit indicating if a clock fail detect has occurred.
- OSWEN: Control bit changes from a '0' to a '1' when a clock transition sequence is initiated. Clearing the OSWEN control bit will abort a clock transition in progress (used for hang-up situations).

If configuration bits FCKSM<2:0> = 1x, then the clock switching and Fail-Safe Clock monitoring functions are disabled. This is the default configuration bit setting.

If clock switching is disabled, then the FOS<2:0> and FPR<4:0> bits directly control the oscillator selection and the COSC<2:0> bits do not control the clock selection. However, these bits will reflect the clock source selection.

Note: The application should not attempt to switch to a clock of frequency lower than 100 KHz when the fail-safe clock monitor is enabled. If such clock switching is performed, the device may generate an oscillator fail trap and switch to the Fast RC oscillator.

20.2.8 PROTECTION AGAINST ACCIDENTAL WRITES TO OSCCON

A write to the OSCCON register is intentionally made difficult because it controls clock switching and clock scaling.

To write to the OSCCON low byte, the following code sequence must be executed without any other instructions in between:

```
Byte Write "0x46" to OSCCON low
Byte Write "0x57" to OSCCON low
```

Byte write is allowed for one instruction cycle. Write the desired value or use bit manipulation instruction.

To write to the OSCCON high byte, the following instructions must be executed without any other instructions in between:

```
Byte Write "0x78" to OSCCON high
Byte Write "0x9A" to OSCCON high
```

Byte write is allowed for one instruction cycle. Write the desired value or use bit manipulation instruction.

20.3 Reset

The dsPIC30F differentiates between various kinds of Reset:

- a) Power-on Reset (POR)
- b) $\overline{\text{MCLR}}$ Reset during normal operation
- c) $\overline{\text{MCLR}}$ Reset during Sleep
- d) Watchdog Timer (WDT) Reset (during normal operation)
- e) Programmable Brown-out Reset (BOR)
- f) RESET Instruction
- g) Reset caused by trap lockup (TRAPR)
- h) Reset caused by illegal opcode or by using an uninitialized W register as an address pointer (IOPUWR)

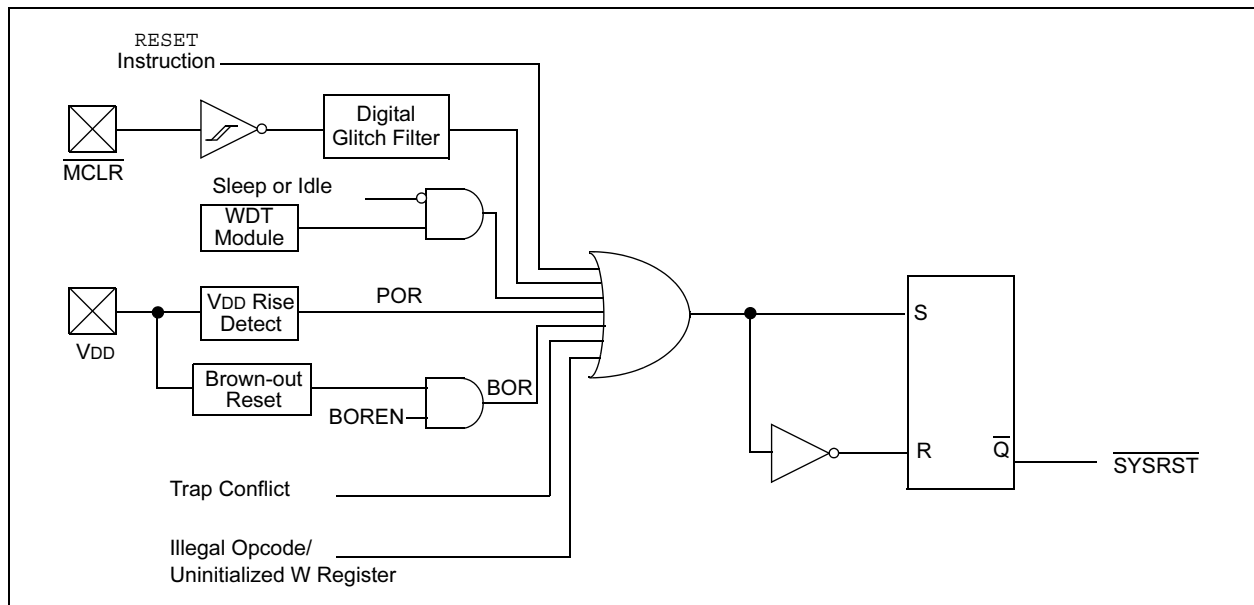
Different registers are affected in different ways by various Reset conditions. Most registers are not affected by a WDT wake-up since this is viewed as the resumption of normal operation. Status bits from the RCON register are set or cleared differently in different Reset situations, as indicated in Table 20-5. These bits are used in software to determine the nature of the Reset.

A block diagram of the On-Chip Reset Circuit is shown in Figure 20-2.

A $\overline{\text{MCLR}}$ noise filter is provided in the $\overline{\text{MCLR}}$ Reset path. The filter detects and ignores small pulses.

Internally generated Resets do not drive $\overline{\text{MCLR}}$ pin low.

FIGURE 20-2: RESET SYSTEM BLOCK DIAGRAM



dsPIC30F

20.3.1 POR: POWER-ON RESET

A power-on event will generate an internal POR pulse when a VDD rise is detected. The Reset pulse will occur at the POR circuit threshold voltage (VPOR) which is nominally 1.85V. The device supply voltage characteristics must meet specified starting voltage and rise rate requirements. The POR pulse will reset a POR timer and place the device in the Reset state. The POR also selects the device clock source identified by the oscillator configuration fuses.

The POR circuit inserts a small delay, TPOR, which is nominally 10 μs and ensures that the device bias circuits are stable. Furthermore, a user selected power-up time-out (TPWRT) is applied. The TPWRT parameter is based on device configuration bits and can be 0 ms (no delay), 4 ms, 16 ms, or 64 ms. The total delay is at device power-up, TPOR + TPWRT. When these delays have expired, $\overline{\text{SYSRST}}$ will be negated on the next leading edge of the Q1 clock and the PC will jump to the Reset vector.

The timing for the $\overline{\text{SYSRST}}$ signal is shown in Figure 20-3 through Figure 20-5.

FIGURE 20-3: TIME-OUT SEQUENCE ON POWER-UP (MCLR TIED TO VDD)

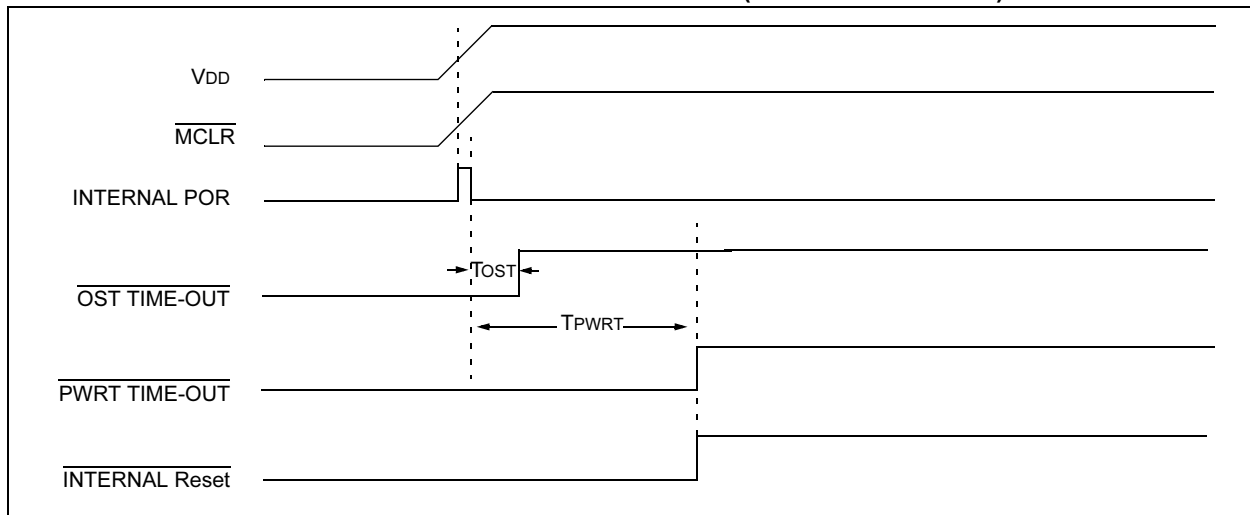


FIGURE 20-4: TIME-OUT SEQUENCE ON POWER-UP (MCLR NOT TIED TO VDD): CASE 1

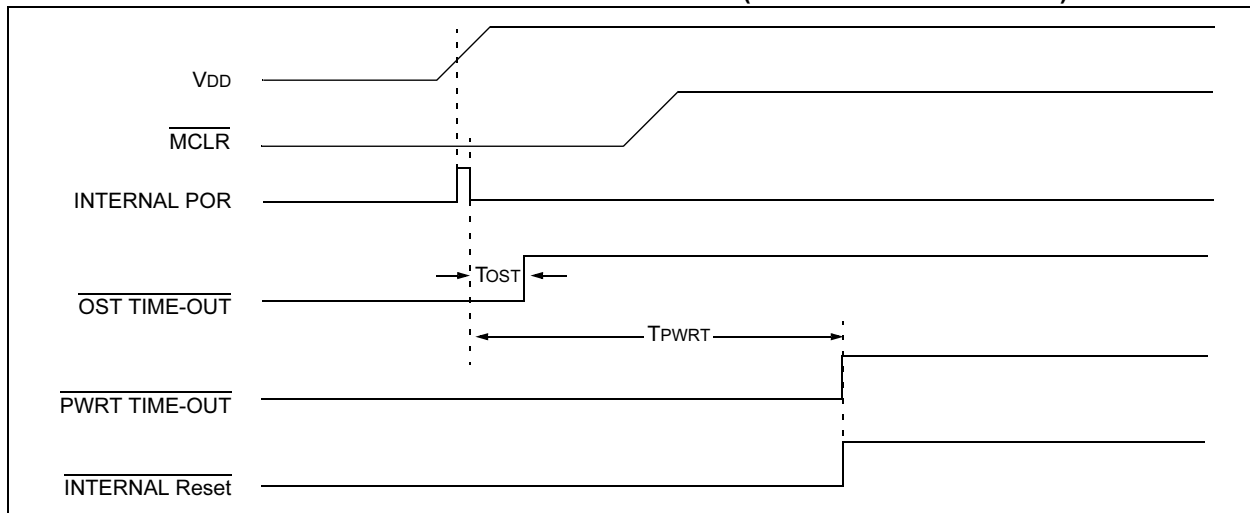
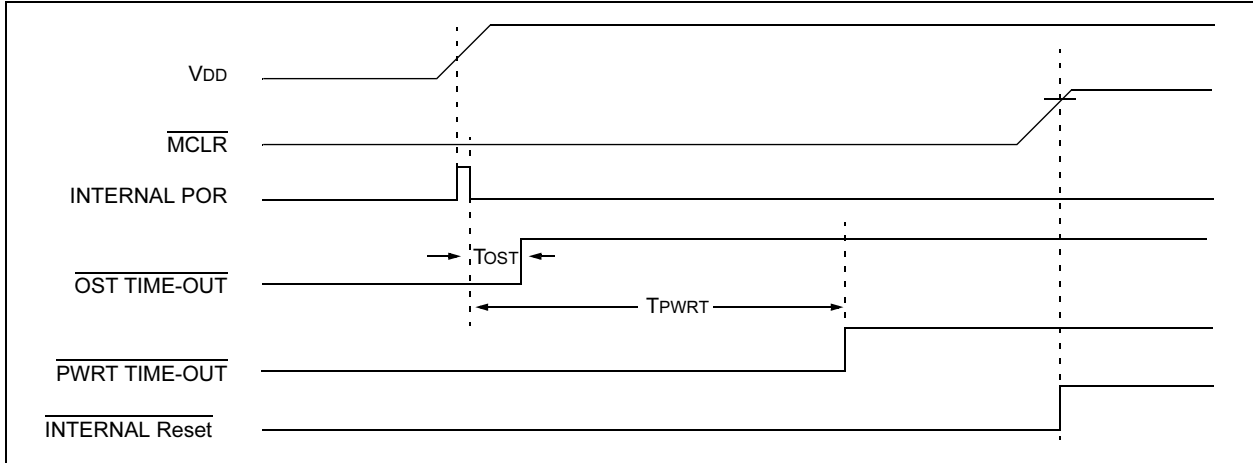


FIGURE 20-5: TIME-OUT SEQUENCE ON POWER-UP ($\overline{\text{MCLR}}$ NOT TIED TO V_{DD}): CASE 2



20.3.1.1 POR with Long Crystal Start-up Time (with FSCM Enabled)

The oscillator start-up circuitry is not linked to the POR circuitry. Some crystal circuits (especially low frequency crystals) will have a relatively long start-up time. Therefore, one or more of the following conditions is possible after the POR timer and the PWRT have expired:

- The oscillator circuit has not begun to oscillate.
- The Oscillator Start-up Timer has not expired (if a crystal oscillator is used).
- The PLL has not achieved a LOCK (if PLL is used).

If the FSCM is enabled and one of the above conditions is true, then a clock failure trap will occur. The device will automatically switch to the FRC oscillator and the user can switch to the desired crystal oscillator in the trap ISR.

20.3.1.2 Operating without FSCM and PWRT

If the FSCM is disabled and the Power-up Timer (PWRT) is also disabled, then the device will exit rapidly from Reset on power-up. If the clock source is FRC, LPRC, EXTRC or EC, it will be active immediately.

If the FSCM is disabled and the system clock has not started, the device will be in a frozen state at the Reset vector until the system clock starts. From the user's perspective, the device will appear to be in Reset until a system clock is available.

20.3.2 BOR: PROGRAMMABLE BROWN-OUT RESET

The BOR (Brown-out Reset) module is based on an internal voltage reference circuit. The main purpose of the BOR module is to generate a device Reset when a brown-out condition occurs. Brown-out conditions are generally caused by glitches on the AC mains (i.e., missing portions of the AC cycle waveform due to bad power transmission lines, or voltage sags due to excessive current draw when a large inductive load is turned on).

The BOR module allows selection of one of the following voltage trip points:

- 2.0V
- 2.7V
- 4.2V
- 4.5V

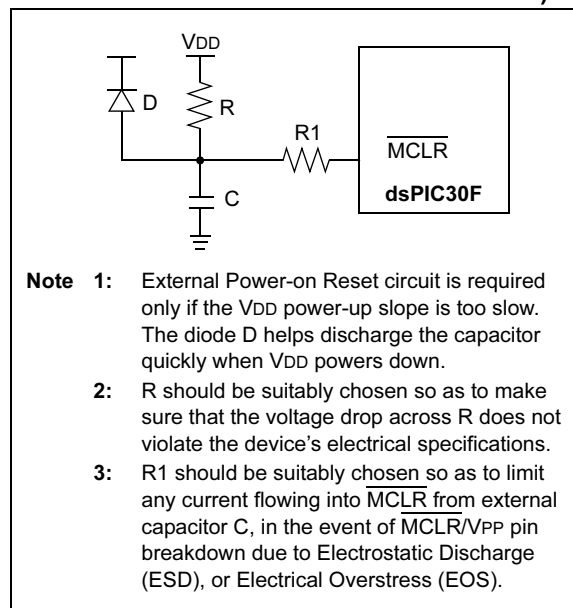
Note: The BOR voltage trip points indicated here are nominal values provided for design guidance only. Refer to the Electrical Specifications in the specific device data sheet for BOR voltage limit specifications.

A BOR will generate a Reset pulse which will reset the device. The BOR will select the clock source based on the device configuration bit values (FOS<1:0> and FPR<3:0>). Furthermore, if an Oscillator mode is selected, the BOR will activate the Oscillator Start-up Timer (OST). The system clock is held until OST expires. If the PLL is used, then the clock will be held until the LOCK bit (OSCCON<5>) is '1'.

Concurrently, the POR time-out (TPOR) and the PWRT time-out (TPWRT) will be applied before the internal Reset is released. If TPWRT = 0 and a crystal oscillator is being used, then a nominal delay of TFSCM = 100 μ s is applied. The total delay in this case is (TPOR + TFSCM).

The BOR status bit (RCON<1>) will be set to indicate that a BOR has occurred. The BOR circuit, if enabled, will continue to operate while in Sleep or Idle modes and will reset the device should VDD fall below the BOR threshold voltage.

FIGURE 20-6: EXTERNAL POWER-ON RESET CIRCUIT (FOR SLOW VDD POWER-UP)



Note: Dedicated supervisory devices, such as the MCP1XX and MCP8XX, may also be used as an external Power-on Reset circuit.

Table 20-5 shows the Reset conditions for the RCON register. Since the control bits within the RCON register are R/W, the information in the table implies that all the bits are negated prior to the action specified in the condition column.

TABLE 20-5: INITIALIZATION CONDITION FOR RCON REGISTER: CASE 1

| Condition | Program Counter | TRAPR | IOPUWR | EXTR | SWR | WDTO | Idle | Sleep | POR | BOR |
|--|-----------------------|-------|--------|------|-----|------|------|-------|-----|-----|
| Power-on Reset | 0x000000 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 |
| Brown-out Reset | 0x000000 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| MCLR Reset during normal operation | 0x000000 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| Software Reset during normal operation | 0x000000 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| MCLR Reset during Sleep | 0x000000 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 |
| MCLR Reset during Idle | 0x000000 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 |
| WDT Time-out Reset | 0x000000 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| WDT Wake-up | PC + 2 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 |
| Interrupt Wake-up from Sleep | PC + 2 ⁽¹⁾ | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| Clock Failure Trap | 0x000004 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Trap Reset | 0x000000 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Illegal Operation Trap | 0x000000 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Legend: u = unchanged, x = unknown, - = unimplemented bit, read as '0'

Note 1: When the wake-up is due to an enabled interrupt, the PC is loaded with the corresponding interrupt vector.

dsPIC30F

Table 20-6 shows a second example of the bit conditions for the RCON register. In this case, it is not assumed the user has set/cleared specific bits prior to action specified in the condition column.

TABLE 20-6: INITIALIZATION CONDITION FOR RCON REGISTER: CASE 2

| Condition | Program Counter | TRAPR | IOPUWR | EXTR | SWR | WDTO | Idle | Sleep | POR | BOR |
|--|-----------------------|-------|--------|------|-----|------|------|-------|-----|-----|
| Power-on Reset | 0x000000 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 |
| Brown-out Reset | 0x000000 | u | u | u | u | u | u | u | 0 | 1 |
| MCLR Reset during normal operation | 0x000000 | u | u | 1 | 0 | 0 | 0 | 0 | u | u |
| Software Reset during normal operation | 0x000000 | u | u | 0 | 1 | 0 | 0 | 0 | u | u |
| MCLR Reset during Sleep | 0x000000 | u | u | 1 | u | 0 | 0 | 1 | u | u |
| MCLR Reset during Idle | 0x000000 | u | u | 1 | u | 0 | 1 | 0 | u | u |
| WDT Time-out Reset | 0x000000 | u | u | 0 | 0 | 1 | 0 | 0 | u | u |
| WDT Wake-up | PC + 2 | u | u | u | u | 1 | u | 1 | u | u |
| Interrupt Wake-up from Sleep | PC + 2 ⁽¹⁾ | u | u | u | u | u | u | 1 | u | u |
| Clock Failure Trap | 0x000004 | u | u | u | u | u | u | u | u | u |
| Trap Reset | 0x000000 | 1 | u | u | u | u | u | u | u | u |
| Illegal Operation Reset | 0x000000 | u | 1 | u | u | u | u | u | u | u |

Legend: u = unchanged, x = unknown, - = unimplemented bit, read as '0'

Note 1: When the wake-up is due to an enabled interrupt, the PC is loaded with the corresponding interrupt vector.

20.4 Watchdog Timer (WDT)

20.4.1 WATCHDOG TIMER OPERATION

The primary function of the Watchdog Timer (WDT) is to reset the processor in the event of a software malfunction. The WDT is a free-running timer which runs off an on-chip RC oscillator, requiring no external component. Therefore, the WDT timer will continue to operate even if the main processor clock (e.g., the crystal oscillator) fails.

20.4.2 ENABLING AND DISABLING THE WDT

The Watchdog Timer can be “Enabled” or “Disabled” only through a configuration bit (FWDTEN) in the Configuration register, FWDT.

Setting FWDTEN = 1 enables the Watchdog Timer. The enabling is done when programming the device. By default, after chip erase, FWDTEN bit = 1. Any device programmer capable of programming dsPIC30F devices allows programming of this and other configuration bits.

If enabled, the WDT will increment until it overflows or “times out”. A WDT time-out will force a device Reset (except during Sleep). To prevent a WDT time-out, the user must clear the Watchdog Timer using a CLRWDT instruction.

If a WDT times out during Sleep, the device will wake-up. The WDTO bit in the RCON register will be cleared to indicate a wake-up resulting from a WDT time-out.

Setting FWDTEN = 0 allows user software to enable/disable the Watchdog Timer via the SWDTEN (RCON<5>) control bit.

20.5 Low Voltage Detect

The Low Voltage Detect (LVD) module is used to detect when the VDD of the device drops below a threshold value, VLVD, which is determined by the LVDL<3:0> bits (RCON<11:8>) and is thus user programmable. The internal voltage reference circuitry requires a nominal amount of time to stabilize, and the BGST bit (RCON<13>) indicates when the voltage reference has stabilized.

In some devices, the LVD threshold voltage may be applied externally on the LVDIN pin.

The LVD module is enabled by setting the LVDEN bit (RCON<12>).

20.6 Power Saving Modes

There are two power saving states that can be entered through the execution of a special instruction, PWRSAV; these are Sleep and Idle.

The format of the PWRSAV instruction is as follows:

PWRSAV <parameter>, where ‘parameter’ defines Idle or Sleep mode.

20.6.1 SLEEP MODE

In Sleep mode, the clock to the CPU and peripherals is shutdown. If an on-chip oscillator is being used, it is shutdown.

The Fail-Safe Clock Monitor is not functional during Sleep since there is no clock to monitor. However, LPRC clock remains active if WDT is operational during Sleep.

The brown-out protection circuit and the Low Voltage Detect circuit, if enabled, will remain functional during Sleep.

The processor wakes up from Sleep if at least one of the following conditions has occurred:

- any interrupt that is individually enabled and meets the required priority level
- any Reset (POR, BOR and MCLR)
- WDT time-out

On waking up from Sleep mode, the processor will restart the same clock that was active prior to entry into Sleep mode. When clock switching is enabled, bits COSC<1:0> will determine the oscillator source that will be used on wake-up. If clock switch is disabled, then there is only one system clock.

Note: If a POR or BOR occurred, the selection of the oscillator is based on the FOS<1:0> and FPR<3:0> configuration bits.

If the clock source is an oscillator, the clock to the device will be held off until OST times out (indicating a stable oscillator). If PLL is used, the system clock is held off until LOCK = 1 (indicating that the PLL is stable). In either case, TPOR, TLOCK and TPWRT delays are applied.

If EC, FRC, LPRC or EXTRC oscillators are used, then a delay of TPOR (~ 10 μs) is applied. This is the smallest delay possible on wake-up from Sleep.

Moreover, if LP oscillator was active during Sleep and LP is the oscillator used on wake-up, then the start-up delay will be equal to TPOR. PWRT delay and OST timer delay are not applied. In order to have the smallest possible start-up delay when waking up from Sleep, one of these faster wake-up options should be selected before entering Sleep.

Any interrupt that is individually enabled (using the corresponding IE bit) and meets the prevailing priority level will be able to wake-up the processor. The processor will process the interrupt and branch to the ISR. The Sleep status bit in the RCON register is set upon wake-up.

Note: In spite of various delays applied (TPOR, TLOCK and TPWRT), the crystal oscillator (and PLL) may not be active at the end of the time-out (e.g., for low frequency crystals). In such cases, if FSCM is enabled, then the device will detect this as a clock failure and process the clock failure trap, the FRC oscillator will be enabled and the user will have to re-enable the crystal oscillator. If FSCM is not enabled, then the device will simply suspend execution of code until the clock is stable and will remain in Sleep until the oscillator clock has started.

All Resets will wake-up the processor from Sleep mode. Any Reset, other than POR, will set the Sleep status bit. In a POR, the Sleep bit is cleared.

If the Watchdog Timer is enabled, then the processor will wake-up from Sleep mode upon WDT time-out. The Sleep and WDTO status bits are both set.

20.6.2 IDLE MODE

In Idle mode, the clock to the CPU is shutdown while peripherals keep running. Unlike Sleep mode, the clock source remains active.

Several peripherals have a control bit in each module that allows them to operate during Idle.

LPRC Fail-Safe Clock remains active if clock failure detect is enabled.

The processor wakes up from Idle if at least one of the following conditions has occurred:

- any interrupt that is individually enabled (IE bit is '1') and meets the required priority level
- any Reset (POR, BOR, MCLR)
- WDT time-out

Upon wake-up from Idle mode, the clock is re-applied to the CPU and instruction execution begins immediately, starting with the instruction following the PWRSAV instruction.

Any interrupt that is individually enabled (using IE bit) and meets the prevailing priority level will be able to wake-up the processor. The processor will process the interrupt and branch to the ISR. The Idle status bit in the RCON register is set upon wake-up.

Any Reset other than POR will set the Idle status bit. On a POR, the Idle bit is cleared.

If Watchdog Timer is enabled, then the processor will wake-up from Idle mode upon WDT time-out. The Idle and WDTO status bits are both set.

Unlike wake-up from Sleep, there are no time delays involved in wake-up from Idle.

20.7 Device Configuration Registers

The configuration bits in each device configuration register specify some of the Device modes and are programmed by a device programmer, or by using the In-Circuit Serial Programming™ (ICSP™) feature of the device. Each device configuration register is a 24-bit register, but only the lower 16 bits of each register are used to hold configuration data. There are four device configuration registers available to the user:

1. FOSC (0xF80000): Oscillator Configuration Register
2. FWDT (0xF80002): Watchdog Timer Configuration Register
3. FBORPOR (0xF80004): BOR and POR Configuration Register
4. FGS (0xF8000A): General Code Segment Configuration Register

The placement of the configuration bits is automatically handled when you select the device in your device programmer. The desired state of the configuration bits may be specified in the source code (dependent on the language tool used), or through the programming interface. After the device has been programmed, the application software may read the configuration bit values through the table read instructions. For additional information, please refer to the Programming Specifications of the device.

Note: If the code protection configuration fuse bits (FGS<GCP> and FGS<GWRP>) have been programmed, an erase of the entire code-protected device is only possible at voltages $V_{DD} \geq 4.5V$.

20.8 Peripheral Module Disable (PMD) Registers

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 will also be disabled so writes to those registers will have no effect and read values will be invalid.

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

Note: 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).

Note 1: In the dsPIC30F6011 and dsPIC30F6013 devices, the DCIMD bit is readable and writeable, and will be read as “1” when set.

2: In the dsPIC30F3014 device, the T4MD, T5MD, IC7MD, IC8MD, OC3MD, OC4MD DCIMD and C1MD bits are readable and writeable, and will be read as “1” when set.

3: In the dsPIC30F2011, dsPIC30F3012 and dsPIC30F2012 devices, the U2MD bit is readable and writeable, and will be read as “1” when set.

20.9 In-Circuit Debugger

When MPLAB ICD2 is selected as a Debugger, the In-Circuit Debugging functionality is enabled. This function allows simple debugging functions when used with MPLAB IDE. When the device has this feature enabled, some of the resources are not available for general use. These resources include the first 80 bytes of Data RAM and two I/O pins.

One of four pairs of Debug I/O pins may be selected by the user using configuration options in MPLAB IDE. These pin pairs are named EMUD/EMUC, EMUD1/EMUC1, EMUD2/EMUC2 and MUD3/EMUC3.

In each case, the selected EMUD pin is the Emulation/Debug Data line, and the EMUC pin is the Emulation/Debug Clock line. These pins will interface to the MPLAB ICD 2 module available from Microchip. The selected pair of Debug I/O pins is used by MPLAB ICD 2 to send commands and receive responses, as well as to send and receive data. To use the In-Circuit Debugger function of the device, the design must implement ICSP connections to MCLR, VDD, VSS, PGC, PGD, and the selected EMUDx/EMUCx pin pair.

This gives rise to two possibilities:

1. If EMUD/EMUC is selected as the Debug I/O pin pair, then only a 5-pin interface is required, as the EMUD and EMUC pin functions are multiplexed with the PGD and PGC pin functions in all dsPIC30F devices.
2. If EMUD1/EMUC1, EMUD2/EMUC2 or EMUD3/EMUC3 is selected as the Debug I/O pin pair, then a 7-pin interface is required, as the EMUDx/EMUCx pin functions (x = 1, 2 or 3) are not multiplexed with the PGD and PGC pin functions.

TABLE 20-7: SYSTEM INTEGRATION 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 | Reset State |
|----------|-------|--------|--------|--------|--------|--------|-----------|-------|-------|-----------|-------|--------|--------|--------|-------|---------|-------|---------------------|
| RCON | 0740 | TRAPR | IOPUWR | BGST | LVDEN | — | LVDL<3:0> | | — | EXTR | SWR | SWDTEN | WDTO | Sleep | Idle | BOR | POR | (Note 1) |
| OSCCON | 0742 | — | — | — | — | — | NOSC<2:0> | | — | POST<1:0> | — | LOCK | — | CF | — | LPOSCEN | OSWEN | (Note 2) |
| OSCTUN | 0774 | — | — | — | — | — | — | — | — | — | — | — | — | TUN3 | TUN2 | TUN1 | TUN0 | 0000 0000 0000 0000 |
| PMD1 | 0770 | T5MD | T4MD | T3MD | T2MD | T1MD | — | — | DC1MD | I2CMD | U2MD | U1MD | SPI2MD | SPI1MD | C2MD | C1MD | ADCMD | 0000 0000 0000 0000 |
| PMD2 | 0772 | IC8MD | IC7MD | IC6MD | IC5MD | IC4MD | IC3MD | IC2MD | IC1MD | OC8MD | OC7MD | OC6MD | OC5MD | OC4MD | OC3MD | OC2MD | OC1MD | 0000 0000 0000 0000 |

Note 1: Reset state depends on type of Reset.

Note 2: Reset state depends on configuration bits.

TABLE 20-8: DEVICE CONFIGURATION REGISTER MAP

| File Name | Addr. | Bits 23-16 | 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 |
|-----------|--------|------------|------------|--------|--------|--------|--------|----------|-------|-------|-------|-------|------------|----------|------------|-------|-------|-------|
| FOSC | F80000 | — | FCKSM<1:0> | | — | — | — | FOS<2:0> | | — | — | — | — | FPR<4:0> | | | | |
| FWDT | F80002 | — | — | — | — | — | — | — | — | — | — | — | FWPSA<1:0> | | FWPSB<3:0> | | | |
| FBORPOR | F80004 | — | — | — | — | — | — | — | — | — | BOREN | — | BORV<1:0> | | FPWRT<1:0> | | | |
| FGS | F8000A | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | GCP | GWRP |

Note: Refer to dsPIC30F Family Reference Manual (DS70046) for descriptions of register bit fields.

21.0 INSTRUCTION SET SUMMARY

Note: This data sheet summarizes features of this group of dsPIC30F devices and is not intended to be a complete reference source. For more information on the CPU, peripherals, register descriptions and general device functionality, refer to the *dsPIC30F Family Reference Manual* (DS70046). For more information on the device instruction set and programming, refer to the *dsPIC30F Programmer's Reference Manual* (DS70030).

The dsPIC30F instruction set adds many enhancements to the previous PICmicro® instruction sets, while maintaining an easy migration from PICmicro instruction sets.

Most instructions are a single program memory word (24 bits). Only three instructions require two program memory locations.

Each single word instruction is a 24-bit word divided into an 8-bit opcode which specifies the instruction type, and one or more operands which further specify the operation of the instruction.

The instruction set is highly orthogonal and is grouped into five basic categories:

- Word or byte-oriented operations
- Bit-oriented operations
- Literal operations
- DSP operations
- Control operations

Table 21-1 shows the general symbols used in describing the instructions.

The dsPIC30F instruction set summary in Table 21-2 lists all the instructions, along with the status flags affected by each instruction.

Most word or byte-oriented W register instructions (including barrel shift instructions) have three operands:

- The first source operand which is typically a register 'Wb' without any address modifier
- The second source operand which is typically a register 'Ws' with or without an address modifier
- The destination of the result which is typically a register 'Wd' with or without an address modifier

However, word or byte-oriented file register instructions have two operands:

- The file register specified by the value 'f'
- The destination, which could either be the file register 'f' or the W0 register, which is denoted as 'WREG'

Most bit-oriented instructions (including simple rotate/shift instructions) have two operands:

- The W register (with or without an address modifier) or file register (specified by the value of 'Ws' or 'f')
- The bit in the W register or file register (specified by a literal value or indirectly by the contents of register 'Wb')

The literal instructions that involve data movement may use some of the following operands:

- A literal value to be loaded into a W register or file register (specified by the value of 'k')
- The W register or file register where the literal value is to be loaded (specified by 'Wb' or 'f')

However, literal instructions that involve arithmetic or logical operations use some of the following operands:

- The first source operand which is a register 'Wb' without any address modifier
- The second source operand which is a literal value
- The destination of the result (only if not the same as the first source operand) which is typically a register 'Wd' with or without an address modifier

The MAC class of DSP instructions may use some of the following operands:

- The accumulator (A or B) to be used (required operand)
- The W registers to be used as the two operands
- The X and Y address space pre-fetch operations
- The X and Y address space pre-fetch destinations
- The accumulator write back destination

The other DSP instructions do not involve any multiplication, and may include:

- The accumulator to be used (required)
- The source or destination operand (designated as Wso or Wdo, respectively) with or without an address modifier
- The amount of shift specified by a W register 'Wn' or a literal value

The control instructions may use some of the following operands:

- A program memory address
- The mode of the table read and table write instructions

dsPIC30F

All instructions are a single word, except for certain double-word instructions, which were made double-word instructions so that all the required information is available in these 48 bits. In the second word, the 8 MSBs are '0's. If this second word is executed as an instruction (by itself), it will execute as a NOP.

Most single word instructions are executed in a single instruction cycle, unless a conditional test is true or the program counter is changed as a result of the instruction. In these cases, the execution takes two instruction cycles with the additional instruction cycle(s) executed as a NOP. Notable exceptions are the BRA (unconditional/computed branch), indirect CALL/GOTO, all table reads and writes, and RETURN/RETFIE instructions,

which are single word instructions but take two or three cycles. Certain instructions that involve skipping over the subsequent instruction require either two or three cycles if the skip is performed, depending on whether the instruction being skipped is a single word or two-word instruction. Moreover, double-word moves require two cycles. The double-word instructions execute in two instruction cycles.

Note: For more details on the instruction set, refer to the Programmer's Reference Manual.

TABLE 21-1: SYMBOLS USED IN OPCODE DESCRIPTIONS

| Field | Description |
|-----------------|---|
| #text | Means literal defined by "text" |
| (text) | Means "content of text" |
| [text] | Means "the location addressed by text" |
| { } | Optional field or operation |
| <n:m> | Register bit field |
| .b | Byte mode selection |
| .d | Double-Word mode selection |
| .S | Shadow register select |
| .w | Word mode selection (default) |
| Acc | One of two accumulators {A, B} |
| AWB | Accumulator write back destination address register $\in \{W13, [W13]+2\}$ |
| bit4 | 4-bit bit selection field (used in word addressed instructions) $\in \{0...15\}$ |
| C, DC, N, OV, Z | MCU status bits: Carry, Digit Carry, Negative, Overflow, Sticky Zero |
| Expr | Absolute address, label or expression (resolved by the linker) |
| f | File register address $\in \{0x0000...0x1FFF\}$ |
| lit1 | 1-bit unsigned literal $\in \{0,1\}$ |
| lit4 | 4-bit unsigned literal $\in \{0...15\}$ |
| lit5 | 5-bit unsigned literal $\in \{0...31\}$ |
| lit8 | 8-bit unsigned literal $\in \{0...255\}$ |
| lit10 | 10-bit unsigned literal $\in \{0...255\}$ for Byte mode, $\{0:1023\}$ for Word mode |
| lit14 | 14-bit unsigned literal $\in \{0...16384\}$ |
| lit16 | 16-bit unsigned literal $\in \{0...65535\}$ |
| lit23 | 23-bit unsigned literal $\in \{0...8388608\}$; LSB must be 0 |
| None | Field does not require an entry, may be blank |
| OA, OB, SA, SB | DSP status bits: AccA Overflow, AccB Overflow, AccA Saturate, AccB Saturate |
| PC | Program Counter |
| Slit10 | 10-bit signed literal $\in \{-512...511\}$ |
| Slit16 | 16-bit signed literal $\in \{-32768...32767\}$ |
| Slit6 | 6-bit signed literal $\in \{-16...16\}$ |

TABLE 21-1: SYMBOLS USED IN OPCODE DESCRIPTIONS (CONTINUED)

| Field | Description |
|-------|--|
| Wb | Base W register $\in \{W0..W15\}$ |
| Wd | Destination W register $\in \{Wd, [Wd], [Wd++] , [Wd--], [++Wd], [--Wd] \}$ |
| Wdo | Destination W register $\in \{Wnd, [Wnd], [Wnd++] , [Wnd--], [++Wnd], [--Wnd], [Wnd+Wb] \}$ |
| Wm,Wn | Dividend, Divisor working register pair (direct addressing) |
| Wm*Wm | Multiplicand and Multiplier working register pair for Square instructions $\in \{W4*W4, W5*W5, W6*W6, W7*W7\}$ |
| Wm*Wn | Multiplicand and Multiplier working register pair for DSP instructions $\in \{W4*W5, W4*W6, W4*W7, W5*W6, W5*W7, W6*W7\}$ |
| Wn | One of 16 working registers $\in \{W0..W15\}$ |
| Wnd | One of 16 destination working registers $\in \{W0..W15\}$ |
| Wns | One of 16 source working registers $\in \{W0..W15\}$ |
| WREG | W0 (working register used in file register instructions) |
| Ws | Source W register $\in \{Ws, [Ws], [Ws++] , [Ws--], [++Ws], [--Ws] \}$ |
| Wso | Source W register $\in \{Wns, [Wns], [Wns++] , [Wns--], [++Wns], [--Wns], [Wns+Wb] \}$ |
| Wx | X data space pre-fetch address register for DSP instructions $\in \{[W8] += 6, [W8] += 4, [W8] += 2, [W8], [W8] -= 6, [W8] -= 4, [W8] -= 2, [W9] += 6, [W9] += 4, [W9] += 2, [W9], [W9] -= 6, [W9] -= 4, [W9] -= 2, [W9+W12], \text{none}\}$ |
| Wxd | X data space pre-fetch destination register for DSP instructions $\in \{W4..W7\}$ |
| Wy | Y data space pre-fetch address register for DSP instructions $\in \{[W10] += 6, [W10] += 4, [W10] += 2, [W10], [W10] -= 6, [W10] -= 4, [W10] -= 2, [W11] += 6, [W11] += 4, [W11] += 2, [W11], [W11] -= 6, [W11] -= 4, [W11] -= 2, [W11+W12], \text{none}\}$ |
| Wyd | Y data space pre-fetch destination register for DSP instructions $\in \{W4..W7\}$ |

dsPIC30F

TABLE 21-2: INSTRUCTION SET OVERVIEW

| Base Instr # | Assembly Mnemonic | Assembly Syntax | Description | # of Words | # of Cycles | Status Flags Affected |
|--------------|-------------------|--------------------|--|------------|-------------|-----------------------|
| 1 | ADD | ADD Acc | Add Accumulators | 1 | 1 | OA,OB,SA,SB |
| | | ADD f | $f = f + WREG$ | 1 | 1 | C,DC,N,OV,Z |
| | | ADD f,WREG | $WREG = f + WREG$ | 1 | 1 | C,DC,N,OV,Z |
| | | ADD #lit10,Wn | $Wd = lit10 + Wd$ | 1 | 1 | C,DC,N,OV,Z |
| | | ADD Wb,Ws,Wd | $Wd = Wb + Ws$ | 1 | 1 | C,DC,N,OV,Z |
| | | ADD Wb,#lit5,Wd | $Wd = Wb + lit5$ | 1 | 1 | C,DC,N,OV,Z |
| | | ADD Wso,#Slit4,Acc | 16-bit Signed Add to Accumulator | 1 | 1 | OA,OB,SA,SB |
| 2 | ADDC | ADDC f | $f = f + WREG + (C)$ | 1 | 1 | C,DC,N,OV,Z |
| | | ADDC f,WREG | $WREG = f + WREG + (C)$ | 1 | 1 | C,DC,N,OV,Z |
| | | ADDC #lit10,Wn | $Wd = lit10 + Wd + (C)$ | 1 | 1 | C,DC,N,OV,Z |
| | | ADDC Wb,Ws,Wd | $Wd = Wb + Ws + (C)$ | 1 | 1 | C,DC,N,OV,Z |
| | | ADDC Wb,#lit5,Wd | $Wd = Wb + lit5 + (C)$ | 1 | 1 | C,DC,N,OV,Z |
| 3 | AND | AND f | $f = f .AND. WREG$ | 1 | 1 | N,Z |
| | | AND f,WREG | $WREG = f .AND. WREG$ | 1 | 1 | N,Z |
| | | AND #lit10,Wn | $Wd = lit10 .AND. Wd$ | 1 | 1 | N,Z |
| | | AND Wb,Ws,Wd | $Wd = Wb .AND. Ws$ | 1 | 1 | N,Z |
| | | AND Wb,#lit5,Wd | $Wd = Wb .AND. lit5$ | 1 | 1 | N,Z |
| 4 | ASR | ASR f | $f = \text{Arithmetic Right Shift } f$ | 1 | 1 | C,N,OV,Z |
| | | ASR f,WREG | $WREG = \text{Arithmetic Right Shift } f$ | 1 | 1 | C,N,OV,Z |
| | | ASR Ws,Wd | $Wd = \text{Arithmetic Right Shift } Ws$ | 1 | 1 | C,N,OV,Z |
| | | ASR Wb,Wns,Wnd | $Wnd = \text{Arithmetic Right Shift } Wb \text{ by } Wns$ | 1 | 1 | N,Z |
| | | ASR Wb,#lit5,Wnd | $Wnd = \text{Arithmetic Right Shift } Wb \text{ by } lit5$ | 1 | 1 | N,Z |
| 5 | BCLR | BCLR f,#bit4 | Bit Clear f | 1 | 1 | None |
| | | BCLR Ws,#bit4 | Bit Clear Ws | 1 | 1 | None |
| 6 | BRA | BRA C,Expr | Branch if Carry | 1 | 1 (2) | None |
| | | BRA GE,Expr | Branch if greater than or equal | 1 | 1 (2) | None |
| | | BRA GEU,Expr | Branch if unsigned greater than or equal | 1 | 1 (2) | None |
| | | BRA GT,Expr | Branch if greater than | 1 | 1 (2) | None |
| | | BRA GTU,Expr | Branch if unsigned greater than | 1 | 1 (2) | None |
| | | BRA LE,Expr | Branch if less than or equal | 1 | 1 (2) | None |
| | | BRA LEU,Expr | Branch if unsigned less than or equal | 1 | 1 (2) | None |
| | | BRA LT,Expr | Branch if less than | 1 | 1 (2) | None |
| | | BRA LTU,Expr | Branch if unsigned less than | 1 | 1 (2) | None |
| | | BRA N,Expr | Branch if Negative | 1 | 1 (2) | None |
| | | BRA NC,Expr | Branch if Not Carry | 1 | 1 (2) | None |
| | | BRA NN,Expr | Branch if Not Negative | 1 | 1 (2) | None |
| | | BRA NOV,Expr | Branch if Not Overflow | 1 | 1 (2) | None |
| | | BRA NZ,Expr | Branch if Not Zero | 1 | 1 (2) | None |
| | | BRA OA,Expr | Branch if Accumulator A overflow | 1 | 1 (2) | None |
| | | BRA OB,Expr | Branch if Accumulator B overflow | 1 | 1 (2) | None |
| | | BRA OV,Expr | Branch if Overflow | 1 | 1 (2) | None |
| | | BRA SA,Expr | Branch if Accumulator A saturated | 1 | 1 (2) | None |
| | | BRA SB,Expr | Branch if Accumulator B saturated | 1 | 1 (2) | None |
| | | BRA Expr | Branch Unconditionally | 1 | 2 | None |
| BRA Z,Expr | Branch if Zero | 1 | 1 (2) | None | | |
| BRA Wn | Computed Branch | 1 | 2 | None | | |
| 7 | BSET | BSET f,#bit4 | Bit Set f | 1 | 1 | None |
| | | BSET Ws,#bit4 | Bit Set Ws | 1 | 1 | None |
| 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 |

TABLE 21-2: INSTRUCTION SET OVERVIEW (CONTINUED)

| Base Instr # | Assembly Mnemonic | Assembly Syntax | Description | # of Words | # of Cycles | Status Flags Affected |
|--------------|-------------------|---------------------------|---|------------|---------------|-----------------------|
| 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 | 2 | None |
| | | CALL Wn | Call indirect subroutine | 1 | 2 | None |
| 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 = \bar{f} | 1 | 1 | N,Z |
| | | COM f,WREG | WREG = \bar{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,#lit5 | Compare Wb with lit5 | 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 | CP1 | CP1 f | Compare f with 0xFFFF | 1 | 1 | C,DC,N,OV,Z |
| | | CP1 Ws | Compare Ws with 0xFFFF | 1 | 1 | C,DC,N,OV,Z |
| 21 | CPB | CPB f | Compare f with WREG, with Borrow | 1 | 1 | C,DC,N,OV,Z |
| | | CPB Wb,#lit5 | Compare Wb with lit5, with Borrow | 1 | 1 | C,DC,N,OV,Z |
| | | CPB Wb,Ws | Compare Wb with Ws, with Borrow (Wb - Ws - C) | 1 | 1 | C,DC,N,OV,Z |
| 22 | CPSEQ | CPSEQ Wb, Wn | Compare Wb with Wn, skip if = | 1 | 1 (2 or 3) | None |
| 23 | CPSGT | CPSGT Wb, Wn | Compare Wb with Wn, skip if > | 1 | 1 (2 or 3) | None |
| 24 | CPSLT | CPSLT Wb, Wn | Compare Wb with Wn, skip if < | 1 | 1 (2 or 3) | None |
| 25 | CPSNE | CPSNE Wb, Wn | Compare Wb with Wn, skip if ≠ | 1 | 1 (2 or 3) | None |
| 26 | DAW | DAW Wn | Wn = decimal adjust Wn | 1 | 1 | C |
| 27 | DEC | DEC f | f = f - 1 | 1 | 1 | C,DC,N,OV,Z |
| | | DEC f,WREG | WREG = f - 1 | 1 | 1 | C,DC,N,OV,Z |
| | | DEC Ws,Wd | Wd = Ws - 1 | 1 | 1 | C,DC,N,OV,Z |
| 28 | DEC2 | DEC2 f | f = f - 2 | 1 | 1 | C,DC,N,OV,Z |
| | | DEC2 f,WREG | WREG = f - 2 | 1 | 1 | C,DC,N,OV,Z |
| | | DEC2 Ws,Wd | Wd = Ws - 2 | 1 | 1 | C,DC,N,OV,Z |

dsPIC30F

TABLE 21-2: INSTRUCTION SET OVERVIEW (CONTINUED)

| Base Instr # | Assembly Mnemonic | Assembly Syntax | Description | # of Words | # of Cycles | Status Flags Affected |
|--------------|-------------------|---------------------------------|---|------------|-------------|-----------------------|
| 29 | DISI | DISI #lit14 | Disable Interrupts for k instruction cycles | 1 | 1 | None |
| 30 | DIV | DIV.S Wm,Wn | Signed 16/16-bit Integer Divide | 1 | 18 | N,Z,C,OV |
| | | DIV.SD Wm,Wn | Signed 32/16-bit Integer Divide | 1 | 18 | N,Z,C,OV |
| | | DIV.U Wm,Wn | Unsigned 16/16-bit Integer Divide | 1 | 18 | N,Z,C,OV |
| | | DIV.UD Wm,Wn | Unsigned 32/16-bit Integer Divide | 1 | 18 | N,Z,C,OV |
| 31 | DIVF | DIVF Wm,Wn | Signed 16/16-bit Fractional Divide | 1 | 18 | N,Z,C,OV |
| 32 | DO | DO #lit14,Expr | Do code to PC+Expr, lit14+1 times | 2 | 2 | None |
| | | DO Wn,Expr | Do code to PC+Expr, (Wn)+1 times | 2 | 2 | None |
| 33 | ED | ED Wm*Wm,Acc,Wx,Wy,Wxd | Euclidean Distance (no accumulate) | 1 | 1 | OA,OB,OAB,SA,SB,SAB |
| 34 | EDAC | EDAC Wm*Wm,Acc,Wx,Wy,Wxd | Euclidean Distance | 1 | 1 | OA,OB,OAB,SA,SB,SAB |
| 35 | EXCH | EXCH Wns,Wnd | Swap Wns with Wnd | 1 | 1 | None |
| 36 | FBCL | FBCL Ws,Wnd | Find Bit Change from Left (MSb) Side | 1 | 1 | C |
| 37 | FF1L | FF1L Ws,Wnd | Find First One from Left (MSb) Side | 1 | 1 | C |
| 38 | FF1R | FF1R Ws,Wnd | Find First One from Right (LSb) Side | 1 | 1 | C |
| 39 | GOTO | GOTO Expr | Go to address | 2 | 2 | None |
| | | GOTO Wn | Go to indirect | 1 | 2 | None |
| 40 | INC | INC f | $f = f + 1$ | 1 | 1 | C,DC,N,OV,Z |
| | | INC f,WREG | $WREG = f + 1$ | 1 | 1 | C,DC,N,OV,Z |
| | | INC Ws,Wd | $Wd = Ws + 1$ | 1 | 1 | C,DC,N,OV,Z |
| 41 | INC2 | INC2 f | $f = f + 2$ | 1 | 1 | C,DC,N,OV,Z |
| | | INC2 f,WREG | $WREG = f + 2$ | 1 | 1 | C,DC,N,OV,Z |
| | | INC2 Ws,Wd | $Wd = Ws + 2$ | 1 | 1 | C,DC,N,OV,Z |
| 42 | IOR | IOR f | $f = f .IOR. WREG$ | 1 | 1 | N,Z |
| | | IOR f,WREG | $WREG = f .IOR. WREG$ | 1 | 1 | N,Z |
| | | IOR #lit10,Wn | $Wd = lit10 .IOR. Wd$ | 1 | 1 | N,Z |
| | | IOR Wb,Ws,Wd | $Wd = Wb .IOR. Ws$ | 1 | 1 | N,Z |
| | | IOR Wb,#lit5,Wd | $Wd = Wb .IOR. lit5$ | 1 | 1 | N,Z |
| 43 | LAC | LAC Wso,#Slit4,Acc | Load Accumulator | 1 | 1 | OA,OB,OAB,SA,SB,SAB |
| 44 | LNK | LNK #lit14 | Link frame pointer | 1 | 1 | None |
| 45 | LSR | LSR f | $f = \text{Logical Right Shift } f$ | 1 | 1 | C,N,OV,Z |
| | | LSR f,WREG | $WREG = \text{Logical Right Shift } f$ | 1 | 1 | C,N,OV,Z |
| | | LSR Ws,Wd | $Wd = \text{Logical Right Shift } Ws$ | 1 | 1 | C,N,OV,Z |
| | | LSR Wb,Wns,Wnd | $Wnd = \text{Logical Right Shift } Wb \text{ by } Wns$ | 1 | 1 | N,Z |
| | | LSR Wb,#lit5,Wnd | $Wnd = \text{Logical Right Shift } Wb \text{ by } lit5$ | 1 | 1 | N,Z |
| 46 | MAC | MAC Wm*Wn,Acc,Wx,Wxd,Wy,Wyd,AWB | Multiply and Accumulate | 1 | 1 | OA,OB,OAB,SA,SB,SAB |
| | | MAC Wm*Wm,Acc,Wx,Wxd,Wy,Wyd | Square and Accumulate | 1 | 1 | OA,OB,OAB,SA,SB,SAB |
| 47 | MOV | MOV f,Wn | Move f to Wn | 1 | 1 | None |
| | | MOV f | Move f to f | 1 | 1 | N,Z |
| | | MOV f,WREG | Move f to WREG | 1 | 1 | N,Z |
| | | MOV #lit16,Wn | Move 16-bit literal to Wn | 1 | 1 | None |
| | | MOV.b #lit8,Wn | Move 8-bit literal to Wn | 1 | 1 | None |
| | | MOV Wn,f | Move Wn to f | 1 | 1 | None |
| | | MOV Wso,Wdo | Move Ws to Wd | 1 | 1 | None |
| | | MOV WREG,f | Move WREG to f | 1 | 1 | N,Z |
| | | MOV.D Wns,Wd | Move Double from W(ns):W(ns+1) to Wd | 1 | 2 | None |
| | | MOV.D Ws,Wnd | Move Double from Ws to W(nd+1):W(nd) | 1 | 2 | None |
| 48 | MOVSAC | MOVSAC Acc,Wx,Wxd,Wy,Wyd,AWB | Pre-fetch and store accumulator | 1 | 1 | None |

TABLE 21-2: INSTRUCTION SET OVERVIEW (CONTINUED)

| Base Instr # | Assembly Mnemonic | Assembly Syntax | Description | # of Words | # of Cycles | Status Flags Affected |
|--------------|-------------------|---------------------------------|--|------------|-------------|-----------------------|
| 49 | MPY | MPY Wm*Wn,Acc,Wx,Wxd,Wy,Wyd | Multiply Wm by Wn to Accumulator | 1 | 1 | OA,OB,OAB,SA,SB,SAB |
| | | MPY Wm*Wm,Acc,Wx,Wxd,Wy,Wyd | Square Wm to Accumulator | 1 | 1 | OA,OB,OAB,SA,SB,SAB |
| 50 | MPY.N | MPY.N Wm*Wn,Acc,Wx,Wxd,Wy,Wyd | -(Multiply Wm by Wn) to Accumulator | 1 | 1 | None |
| 51 | MSC | MSC Wm*Wm,Acc,Wx,Wxd,Wy,Wyd,AWB | Multiply and Subtract from Accumulator | 1 | 1 | OA,OB,OAB,SA,SB,SAB |
| 52 | MUL | MUL.SS Wb,Ws,Wnd | {Wnd+1, Wnd} = signed(Wb) * signed(Ws) | 1 | 1 | None |
| | | MUL.SU Wb,Ws,Wnd | {Wnd+1, Wnd} = signed(Wb) * unsigned(Ws) | 1 | 1 | None |
| | | MUL.US Wb,Ws,Wnd | {Wnd+1, Wnd} = unsigned(Wb) * signed(Ws) | 1 | 1 | None |
| | | MUL.UU Wb,Ws,Wnd | {Wnd+1, Wnd} = unsigned(Wb) * unsigned(Ws) | 1 | 1 | None |
| | | MUL.SU Wb,#lit5,Wnd | {Wnd+1, Wnd} = signed(Wb) * unsigned(lit5) | 1 | 1 | None |
| | | MUL.UU Wb,#lit5,Wnd | {Wnd+1, Wnd} = unsigned(Wb) * unsigned(lit5) | 1 | 1 | None |
| | | MUL f | W3:W2 = f * WREG | 1 | 1 | None |
| 53 | NEG | NEG Acc | Negate Accumulator | 1 | 1 | OA,OB,OAB,SA,SB,SAB |
| | | NEG f | $f = \bar{f} + 1$ | 1 | 1 | C,DC,N,OV,Z |
| | | NEG f,WREG | WREG = $\bar{f} + 1$ | 1 | 1 | C,DC,N,OV,Z |
| | | NEG Ws,Wd | $Wd = \bar{Ws} + 1$ | 1 | 1 | C,DC,N,OV,Z |
| 54 | NOP | NOP | No Operation | 1 | 1 | None |
| | | NOPR | No Operation | 1 | 1 | None |
| 55 | POP | POP f | Pop f from top-of-stack (TOS) | 1 | 1 | None |
| | | POP Wdo | Pop from top-of-stack (TOS) to Wdo | 1 | 1 | None |
| | | POP.D Wnd | Pop from top-of-stack (TOS) to W(nd):W(nd+1) | 1 | 2 | None |
| | | POP.S | Pop Shadow Registers | 1 | 1 | All |
| 56 | PUSH | PUSH f | Push f to top-of-stack (TOS) | 1 | 1 | None |
| | | PUSH Wso | Push Wso to top-of-stack (TOS) | 1 | 1 | None |
| | | PUSH.D Wns | Push W(ns):W(ns+1) to top-of-stack (TOS) | 1 | 2 | None |
| | | PUSH.S | Push Shadow Registers | 1 | 1 | None |
| 57 | PWRSVAV | PWRSVAV #lit1 | Go into Sleep or Idle mode | 1 | 1 | WDTO,Sleep |
| 58 | RCALL | RCALL Expr | Relative Call | 1 | 2 | None |
| | | RCALL Wn | Computed Call | 1 | 2 | None |
| 59 | REPEAT | REPEAT #lit14 | Repeat Next Instruction lit14+1 times | 1 | 1 | None |
| | | REPEAT Wn | Repeat Next Instruction (Wn)+1 times | 1 | 1 | None |
| 60 | RESET | RESET | Software device Reset | 1 | 1 | None |
| 61 | RETFIE | RETFIE | Return from interrupt | 1 | 3 (2) | None |
| 62 | RETLW | RETLW #lit10,Wn | Return with literal in Wn | 1 | 3 (2) | None |
| 63 | RETURN | RETURN | Return from Subroutine | 1 | 3 (2) | None |
| 64 | RLC | RLC f | f = Rotate Left through Carry f | 1 | 1 | C,N,Z |
| | | RLC f,WREG | WREG = Rotate Left through Carry f | 1 | 1 | C,N,Z |
| | | RLC Ws,Wd | Wd = Rotate Left through Carry Ws | 1 | 1 | C,N,Z |
| 65 | RLNC | RLNC f | f = Rotate Left (No Carry) f | 1 | 1 | N,Z |
| | | RLNC f,WREG | WREG = Rotate Left (No Carry) f | 1 | 1 | N,Z |
| | | RLNC Ws,Wd | Wd = Rotate Left (No Carry) Ws | 1 | 1 | N,Z |
| 66 | RRC | RRC f | f = Rotate Right through Carry f | 1 | 1 | C,N,Z |
| | | RRC f,WREG | WREG = Rotate Right through Carry f | 1 | 1 | C,N,Z |
| | | RRC Ws,Wd | Wd = Rotate Right through Carry Ws | 1 | 1 | C,N,Z |
| 67 | RRNC | RRNC f | f = Rotate Right (No Carry) f | 1 | 1 | N,Z |
| | | RRNC f,WREG | WREG = Rotate Right (No Carry) f | 1 | 1 | N,Z |
| | | RRNC Ws,Wd | Wd = Rotate Right (No Carry) Ws | 1 | 1 | N,Z |

dsPIC30F

TABLE 21-2: INSTRUCTION SET OVERVIEW (CONTINUED)

| Base Instr # | Assembly Mnemonic | Assembly Syntax | Description | # of Words | # of Cycles | Status Flags Affected |
|--------------|-------------------|----------------------|---------------------------------------|------------|-------------|-----------------------|
| 68 | SAC | SAC Acc,#Slit4,Wdo | Store Accumulator | 1 | 1 | None |
| | | SAC.R Acc,#Slit4,Wdo | Store Rounded Accumulator | 1 | 1 | None |
| 69 | SE | SE Ws,Wnd | Wnd = sign-extended Ws | 1 | 1 | C,N,Z |
| 70 | SETM | SETM f | f = 0xFFFF | 1 | 1 | None |
| | | SETM WREG | WREG = 0xFFFF | 1 | 1 | None |
| | | SETM Ws | Ws = 0xFFFF | 1 | 1 | None |
| 71 | SFTAC | SFTAC Acc,Wn | Arithmetic Shift Accumulator by (Wn) | 1 | 1 | OA,OB,OAB,SA,SB,SAB |
| | | SFTAC Acc,#Slit6 | Arithmetic Shift Accumulator by Slit6 | 1 | 1 | OA,OB,OAB,SA,SB,SAB |
| 72 | SL | SL f | f = Left Shift f | 1 | 1 | C,N,OV,Z |
| | | SL f,WREG | WREG = Left Shift f | 1 | 1 | C,N,OV,Z |
| | | SL Ws,Wd | Wd = Left Shift Ws | 1 | 1 | C,N,OV,Z |
| | | SL Wb,Wns,Wnd | Wnd = Left Shift Wb by Wns | 1 | 1 | N,Z |
| | | SL Wb,#lit5,Wnd | Wnd = Left Shift Wb by lit5 | 1 | 1 | N,Z |
| 73 | SUB | SUB Acc | Subtract Accumulators | 1 | 1 | OA,OB,OAB,SA,SB,SAB |
| | | SUB f | f = f - WREG | 1 | 1 | C,DC,N,OV,Z |
| | | SUB f,WREG | WREG = f - WREG | 1 | 1 | C,DC,N,OV,Z |
| | | SUB #lit10,Wn | Wn = Wn - lit10 | 1 | 1 | C,DC,N,OV,Z |
| | | SUB Wb,Ws,Wd | Wd = Wb - Ws | 1 | 1 | C,DC,N,OV,Z |
| | | SUB Wb,#lit5,Wd | Wd = Wb - lit5 | 1 | 1 | C,DC,N,OV,Z |
| 74 | SUBB | SUBB f | f = f - WREG - (\overline{C}) | 1 | 1 | C,DC,N,OV,Z |
| | | SUBB f,WREG | WREG = f - WREG - (\overline{C}) | 1 | 1 | C,DC,N,OV,Z |
| | | SUBB #lit10,Wn | Wn = Wn - lit10 - (\overline{C}) | 1 | 1 | C,DC,N,OV,Z |
| | | SUBB Wb,Ws,Wd | Wd = Wb - Ws - (\overline{C}) | 1 | 1 | C,DC,N,OV,Z |
| | | SUBB Wb,#lit5,Wd | Wd = Wb - lit5 - (\overline{C}) | 1 | 1 | C,DC,N,OV,Z |
| 75 | SUBR | SUBR f | f = WREG - f | 1 | 1 | C,DC,N,OV,Z |
| | | SUBR f,WREG | WREG = WREG - f | 1 | 1 | C,DC,N,OV,Z |
| | | SUBR Wb,Ws,Wd | Wd = Ws - Wb | 1 | 1 | C,DC,N,OV,Z |
| | | SUBR Wb,#lit5,Wd | Wd = lit5 - Wb | 1 | 1 | C,DC,N,OV,Z |
| 76 | SUBBR | SUBBR f | f = WREG - f - (\overline{C}) | 1 | 1 | C,DC,N,OV,Z |
| | | SUBBR f,WREG | WREG = WREG - f - (\overline{C}) | 1 | 1 | C,DC,N,OV,Z |
| | | SUBBR Wb,Ws,Wd | Wd = Ws - Wb - (\overline{C}) | 1 | 1 | C,DC,N,OV,Z |
| | | SUBBR Wb,#lit5,Wd | Wd = lit5 - Wb - (\overline{C}) | 1 | 1 | C,DC,N,OV,Z |
| 77 | SWAP | SWAP.b Wn | Wn = nibble swap Wn | 1 | 1 | None |
| | | SWAP Wn | Wn = byte swap Wn | 1 | 1 | None |
| 78 | TBLRDH | TBLRDH Ws,Wd | Read Prog<23:16> to Wd<7:0> | 1 | 2 | None |
| 79 | TBLRDL | TBLRDL Ws,Wd | Read Prog<15:0> to Wd | 1 | 2 | None |
| 80 | TBLWTH | TBLWTH Ws,Wd | Write Ws<7:0> to Prog<23:16> | 1 | 2 | None |
| 81 | TBLWTL | TBLWTL Ws,Wd | Write Ws to Prog<15:0> | 1 | 2 | None |
| 82 | ULNK | ULNK | Unlink frame pointer | 1 | 1 | None |
| 83 | XOR | XOR f | f = f .XOR. WREG | 1 | 1 | N,Z |
| | | XOR f,WREG | WREG = f .XOR. WREG | 1 | 1 | N,Z |
| | | XOR #lit10,Wn | Wd = lit10 .XOR. Wd | 1 | 1 | N,Z |
| | | XOR Wb,Ws,Wd | Wd = Wb .XOR. Ws | 1 | 1 | N,Z |
| | | XOR Wb,#lit5,Wd | Wd = Wb .XOR. lit5 | 1 | 1 | N,Z |
| 84 | ZE | ZE Ws,Wnd | Wnd = Zero-extend Ws | 1 | 1 | C,Z,N |

22.0 DEVELOPMENT SUPPORT

The PICmicro® microcontrollers are supported with a full range of hardware and software development tools:

- Integrated Development Environment
 - MPLAB® IDE Software
- Assemblers/Compilers/Linkers
 - MPASM™ Assembler
 - MPLAB C17 and MPLAB C18 C Compilers
 - MPLINK™ Object Linker/
MPLIB™ Object Librarian
 - MPLAB C30 C Compiler
 - MPLAB ASM30 Assembler/Linker/Library
- Simulators
 - MPLAB SIM Software Simulator
 - MPLAB dsPIC30 Software Simulator
- Emulators
 - MPLAB ICE 2000 In-Circuit Emulator
 - MPLAB ICE 4000 In-Circuit Emulator
- In-Circuit Debugger
 - MPLAB ICD 2
- Device Programmers
 - PRO MATE® II Universal Device Programmer
 - PICSTART® Plus Development Programmer
- Low Cost Demonstration Boards
 - PICDEM™ 1 Demonstration Board
 - PICDEM.net™ Demonstration Board
 - PICDEM 2 Plus Demonstration Board
 - PICDEM 3 Demonstration Board
 - PICDEM 4 Demonstration Board
 - PICDEM 17 Demonstration Board
 - PICDEM 18R Demonstration Board
 - PICDEM LIN Demonstration Board
 - PICDEM USB Demonstration Board
- Evaluation Kits
 - KEELOQ®
 - PICDEM MSC
 - microID®
 - CAN
 - PowerSmart®
 - Analog

22.1 MPLAB Integrated Development Environment Software

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

- An interface to debugging tools
 - simulator
 - programmer (sold separately)
 - 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
- Extensive on-line help

The MPLAB IDE allows you to:

- Edit your source files (either assembly or C)
- One touch assemble (or compile) and download to PICmicro emulator and simulator tools (automatically updates all project information)
- Debug using:
 - source files (assembly or C)
 - absolute listing file (mixed assembly and C)
 - 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 increasing flexibility and power.

22.2 MPASM Assembler

The MPASM assembler is a full-featured, universal macro assembler for all PICmicro 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 IDE projects
- User defined macros to streamline assembly code
- Conditional assembly for multi-purpose source files
- Directives that allow complete control over the assembly process

22.3 MPLAB C17 and MPLAB C18 C Compilers

The MPLAB C17 and MPLAB C18 Code Development Systems are complete ANSI C compilers for Microchip's PIC17CXXX and PIC18CXXX family of microcontrollers. These compilers provide powerful integration capabilities, superior code optimization and ease of use not found with other compilers.

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

22.4 MPLINK Object Linker/ MPLIB Object Librarian

The MPLINK object linker combines relocatable objects created by the MPASM assembler and the MPLAB C17 and MPLAB C18 C compilers. It can link relocatable objects from pre-compiled libraries, using directives from a linker script.

The MPLIB object librarian manages the creation and modification of library files of pre-compiled 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

22.5 MPLAB C30 C Compiler

The MPLAB C30 C compiler is a full-featured, ANSI compliant, optimizing compiler that translates standard ANSI C programs into dsPIC30F assembly language source. The compiler also supports many command-line options and language extensions to take full advantage of the dsPIC30F device hardware capabilities, and afford fine control of the compiler code generator.

MPLAB C30 is distributed with a complete ANSI C standard library. All library functions have been validated and conform to the ANSI C library standard. The library includes functions for string manipulation, dynamic memory allocation, data conversion, time-keeping, and math functions (trigonometric, exponential and hyperbolic). The compiler provides symbolic information for high level source debugging with the MPLAB IDE.

22.6 MPLAB ASM30 Assembler, Linker, and Librarian

MPLAB ASM30 assembler produces relocatable machine code from symbolic assembly language for dsPIC30F devices. MPLAB C30 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 dsPIC30F instruction set
- Support for fixed-point and floating-point data
- Command line interface
- Rich directive set
- Flexible macro language
- MPLAB IDE compatibility

22.7 MPLAB SIM Software Simulator

The MPLAB SIM software simulator allows code development in a PC hosted environment by simulating the PICmicro series microcontrollers on an instruction level. On any given instruction, the data areas can be examined or modified and stimuli can be applied from a file, or user defined key press, to any pin. The execution can be performed in Single-Step, Execute Until Break, or Trace mode.

The MPLAB SIM simulator fully supports symbolic debugging using the MPLAB C17 and MPLAB C18 C Compilers, as well as the MPASM assembler. The software simulator offers the flexibility to develop and debug code outside of the laboratory environment, making it an excellent, economical software development tool.

22.8 MPLAB SIM30 Software Simulator

The MPLAB SIM30 software simulator allows code development in a PC hosted environment by simulating the dsPIC30F series microcontrollers on an instruction level. On any given instruction, the data areas can be examined or modified and stimuli can be applied from a file, or user defined key press, to any of the pins.

The MPLAB SIM30 simulator fully supports symbolic debugging using the MPLAB C30 C Compiler and MPLAB ASM30 assembler. The simulator runs in either a Command Line mode for automated tasks, or from MPLAB IDE. This high speed simulator is designed to debug, analyze and optimize time intensive DSP routines.

22.9 MPLAB ICE 2000 High Performance Universal In-Circuit Emulator

The MPLAB ICE 2000 universal in-circuit emulator is intended to provide the product development engineer with a complete microcontroller design tool set for PICmicro microcontrollers. Software control of the MPLAB ICE 2000 in-circuit emulator is advanced by the MPLAB Integrated Development Environment, which allows editing, building, downloading and source debugging from a single environment.

The MPLAB ICE 2000 is a full-featured emulator system with enhanced trace, trigger and data monitoring features. Interchangeable processor modules allow the system to be easily reconfigured for emulation of different processors. The universal architecture of the MPLAB ICE in-circuit emulator allows expansion to support new PICmicro microcontrollers.

The MPLAB ICE 2000 in-circuit emulator system has been designed as a real-time emulation system with advanced features that are typically found on more expensive development tools. The PC platform and Microsoft® Windows 32-bit operating system were chosen to best make these features available in a simple, unified application.

22.10 MPLAB ICE 4000 High Performance Universal In-Circuit Emulator

The MPLAB ICE 4000 universal in-circuit emulator is intended to provide the product development engineer with a complete microcontroller design tool set for high-end PICmicro microcontrollers. Software control of the MPLAB ICE in-circuit emulator is provided by the MPLAB Integrated Development Environment, which allows editing, building, downloading and source debugging from a single environment.

The MPLAB ICE 4000 is a premium emulator system, providing the features of MPLAB ICE 2000, but with increased emulation memory and high speed performance for dsPIC30F and PIC18XXXX devices. Its advanced emulator features include complex triggering and timing, up to 2 Mb of emulation memory, and the ability to view variables in real-time.

The MPLAB ICE 4000 in-circuit emulator system has been designed as a real-time emulation system with advanced features that are typically found on more expensive development tools. The PC platform and Microsoft Windows 32-bit operating system were chosen to best make these features available in a simple, unified application.

22.11 MPLAB ICD 2 In-Circuit Debugger

Microchip's In-Circuit Debugger, MPLAB ICD 2, is a powerful, low cost, run-time development tool, connecting to the host PC via an RS-232 or high speed USB interface. This tool is based on the Flash PICmicro MCUs and can be used to develop for these and other PICmicro microcontrollers. The MPLAB ICD 2 utilizes the in-circuit debugging capability built into the Flash devices. This feature, along with Microchip's In-Circuit Serial Programming™ (ICSP™) protocol, offers cost effective in-circuit Flash debugging from the graphical user interface of the MPLAB Integrated Development Environment. This enables a designer to develop and debug source code by setting breakpoints, single-stepping and watching variables, CPU status and peripheral registers. Running at full speed enables testing hardware and applications in real-time. MPLAB ICD 2 also serves as a development programmer for selected PICmicro devices.

22.12 PRO MATE II Universal Device Programmer

The PRO MATE II is a universal, CE compliant device programmer with programmable voltage verification at VDDMIN and VDDMAX for maximum reliability. It features an LCD display for instructions and error messages and a modular detachable socket assembly to support various package types. In Stand-Alone mode, the PRO MATE II device programmer can read, verify, and program PICmicro devices without a PC connection. It can also set code protection in this mode.

22.13 PICSTART Plus Development Programmer

The PICSTART Plus development programmer is an easy-to-use, low cost, prototype programmer. It connects to the PC via a COM (RS-232) port. MPLAB Integrated Development Environment software makes using the programmer simple and efficient. The PICSTART Plus development programmer supports most PICmicro devices up to 40 pins. Larger pin count devices, such as the PIC16C92X and PIC17C76X, may be supported with an adapter socket. The PICSTART Plus development programmer is CE compliant.

22.14 PICDEM 1 PICmicro Demonstration Board

The PICDEM 1 demonstration board demonstrates the capabilities of the PIC16C5X (PIC16C54 to PIC16C58A), PIC16C61, PIC16C62X, PIC16C71, PIC16C8X, PIC17C42, PIC17C43 and PIC17C44. All necessary hardware and software is included to run basic demo programs. The sample microcontrollers provided with the PICDEM 1 demonstration board can be programmed with a PRO MATE II device programmer, or a PICSTART Plus development programmer. The PICDEM 1 demonstration board can be connected to the MPLAB ICE in-circuit emulator for testing. A prototype area extends the circuitry for additional application components. Features include an RS-232 interface, a potentiometer for simulated analog input, push button switches and eight LEDs.

22.15 PICDEM.net Internet/Ethernet Demonstration Board

The PICDEM.net demonstration board is an Internet/Ethernet demonstration board using the PIC18F452 microcontroller and TCP/IP firmware. The board supports any 40-pin DIP device that conforms to the standard pinout used by the PIC16F877 or PIC18C452. This kit features a user friendly TCP/IP stack, web server with HTML, a 24L256 Serial EEPROM for Xmodem download to web pages into Serial EEPROM, ICSP/MPLAB ICD 2 interface connector, an Ethernet interface, RS-232 interface, and a 16 x 2 LCD display. Also included is the book and CD-ROM *"TCP/IP Lean, Web Servers for Embedded Systems,"* by Jeremy Bentham

22.16 PICDEM 2 Plus Demonstration Board

The PICDEM 2 Plus demonstration board supports many 18-, 28-, and 40-pin microcontrollers, including PIC16F87X and PIC18FXX2 devices. All the necessary hardware and software is included to run the demonstration programs. The sample microcontrollers provided with the PICDEM 2 demonstration board can be programmed with a PRO MATE II device programmer, PICSTART Plus development programmer, or MPLAB ICD 2 with a Universal Programmer Adapter. The MPLAB ICD 2 and MPLAB ICE in-circuit emulators may also be used with the PICDEM 2 demonstration board to test firmware. A prototype area extends the circuitry for additional application components. Some of the features include an RS-232 interface, a 2 x 16 LCD display, a piezo speaker, an on-board temperature sensor, four LEDs, and sample PIC18F452 and PIC16F877 Flash microcontrollers.

22.17 PICDEM 3 PIC16C92X Demonstration Board

The PICDEM 3 demonstration board supports the PIC16C923 and PIC16C924 in the PLCC package. All the necessary hardware and software is included to run the demonstration programs.

22.18 PICDEM 4 8/14/18-Pin Demonstration Board

The PICDEM 4 can be used to demonstrate the capabilities of the 8-, 14-, and 18-pin PIC16XXXX and PIC18XXXX MCUs, including the PIC16F818/819, PIC16F87/88, PIC16F62XA and the PIC18F1320 family of microcontrollers. PICDEM 4 is intended to showcase the many features of these low pin count parts, including LIN and Motor Control using ECCP. Special provisions are made for low power operation with the super capacitor circuit, and jumpers allow on-board hardware to be disabled to eliminate current draw in this mode. Included on the demo board are provisions for Crystal, RC or Canned Oscillator modes, a five volt regulator for use with a nine volt wall adapter or battery, DB-9 RS-232 interface, ICD connector for programming via ICSP and development with MPLAB ICD 2, 2x16 liquid crystal display, PCB footprints for H-Bridge motor driver, LIN transceiver and EEPROM. Also included are: header for expansion, eight LEDs, four potentiometers, three push buttons and a prototyping area. Included with the kit is a PIC16F627A and a PIC18F1320. Tutorial firmware is included along with the User's Guide.

22.19 PICDEM 17 Demonstration Board

The PICDEM 17 demonstration board is an evaluation board that demonstrates the capabilities of several Microchip microcontrollers, including PIC17C752, PIC17C756A, PIC17C762 and PIC17C766. A programmed sample is included. The PRO MATE II device programmer, or the PICSTART Plus development programmer, can be used to reprogram the device for user tailored application development. The PICDEM 17 demonstration board supports program download and execution from external on-board Flash memory. A generous prototype area is available for user hardware expansion.

22.20 PICDEM 18R PIC18C601/801 Demonstration Board

The PICDEM 18R demonstration board serves to assist development of the PIC18C601/801 family of Microchip microcontrollers. It provides hardware implementation of both 8-bit Multiplexed/De-multiplexed and 16-bit Memory modes. The board includes 2 Mb external Flash memory and 128 Kb SRAM memory, as well as serial EEPROM, allowing access to the wide range of memory types supported by the PIC18C601/801.

22.21 PICDEM LIN PIC16C43X Demonstration Board

The powerful LIN hardware and software kit includes a series of boards and three PICmicro microcontrollers. The small footprint PIC16C432 and PIC16C433 are used as slaves in the LIN communication and feature on-board LIN transceivers. A PIC16F874 Flash microcontroller serves as the master. All three microcontrollers are programmed with firmware to provide LIN bus communication.

22.22 PICKit™ 1 Flash Starter Kit

A complete "development system in a box", the PICKit Flash Starter Kit includes a convenient multi-section board for programming, evaluation, and development of 8/14-pin Flash PIC® microcontrollers. Powered via USB, the board operates under a simple Windows GUI. The PICKit 1 Starter Kit includes the user's guide (on CD ROM), PICKit 1 tutorial software and code for various applications. Also included are MPLAB® IDE (Integrated Development Environment) software, software and hardware "Tips 'n Tricks for 8-pin Flash PIC® Microcontrollers" Handbook and a USB Interface Cable. Supports all current 8/14-pin Flash PIC microcontrollers, as well as many future planned devices.

22.23 PICDEM USB PIC16C7X5 Demonstration Board

The PICDEM USB Demonstration Board shows off the capabilities of the PIC16C745 and PIC16C765 USB microcontrollers. This board provides the basis for future USB products.

22.24 Evaluation and Programming Tools

In addition to the PICDEM series of circuits, Microchip has a line of evaluation kits and demonstration software for these products.

- KEELOQ evaluation and programming tools for Microchip's HCS Secure Data Products
- CAN developers kit for automotive network applications
- Analog design boards and filter design software
- PowerSmart battery charging evaluation/calibration kits
- IrDA® development kit
- microID development and rflab™ development software
- SEEVAL® designer kit for memory evaluation and endurance calculations
- PICDEM MSC demo boards for Switching mode power supply, high power IR driver, delta sigma ADC, and flow rate sensor

Check the Microchip web page and the latest Product Line Card for the complete list of demonstration and evaluation kits.

dsPIC30F

NOTES:

23.0 ELECTRICAL CHARACTERISTICS

This section provides an overview of dsPIC30F electrical characteristics. Additional information will be provided in future revisions of this document as it becomes available.

For detailed information about the dsPIC30F architecture and core, refer to *dsPIC30F Family Reference Manual* (DS70046).

Absolute maximum ratings for the dsPIC30F family are listed below. Exposure to these maximum rating conditions for extended periods may affect device reliability. Functional operation of the device at these or any other conditions above the parameters indicated in the operation listings of this specification is not implied.

Note: The specifications listed in this section are provided for design guidance only. Please refer to the individual device data sheets for electrical characteristics specific to each device.

Absolute Maximum Ratings^(†)

| | |
|--|-----------------------|
| Ambient temperature under bias..... | -40°C to +125°C |
| Storage temperature | -65°C to +150°C |
| Voltage on any pin with respect to VSS (except VDD and MCLR) | -0.3V to (VDD + 0.3V) |
| Voltage on VDD with respect to VSS | -0.3V to +5.5V |
| Voltage on MCLR with respect to VSS (Note 1) | 0V to +13.25V |
| Total power dissipation (Note 2) | 1.0W |
| Maximum current out of VSS pin | 300 mA |
| Maximum current into VDD pin | 250 mA |
| Input clamp current, I _{IK} (V _I < 0 or V _I > VDD)..... | ±20 mA |
| Output clamp current, I _{OK} (V _O < 0 or V _O > VDD) | ±20 mA |
| Maximum output current sunk by any I/O pin..... | 25 mA |
| Maximum output current sourced by any I/O pin | 25 mA |
| Maximum current sunk by all ports | 200 mA |
| Maximum current sourced by all ports | 200 mA |

Note 1: Power dissipation is calculated as follows:

$$P_{dis} = V_{DD} \times \{I_{DD} - \sum I_{OH}\} + \sum \{(V_{DD} - V_{OH}) \times I_{OH}\} + \sum (V_{OL} \times I_{OL})$$

- 2:** Voltage spikes below VSS at the MCLR/VPP pin, inducing currents greater than 80 mA, may cause latchup. Thus, a series resistor of 50-100Ω should be used when applying a “low” level to the MCLR/VPP pin, rather than pulling this pin directly to VSS.

[†]NOTICE: Stresses above those listed under “Absolute Maximum Ratings” may cause permanent damage to the device. This is a stress rating only and functional operation of the device at those or any other conditions above those indicated in the operation listings of this specification is not implied. Exposure to maximum rating conditions for extended periods may affect device reliability.

Note: All peripheral electrical characteristics are specified. For exact peripherals available on specific devices, please refer the the Family Cross Reference Table.

dsPIC30F

23.1 DC Characteristics

TABLE 23-1: OPERATING MIPS VS. VOLTAGE

| VDD Range | Temp Range | Max MIPS | | |
|-----------|----------------|-----------------|-----------------|-----------------|
| | | dsPIC30FXXX-30I | dsPIC30FXXX-20I | dsPIC30FXXX-20E |
| 4.5-5.5V | -40°C to 85°C | 30 | 20 | — |
| 4.5-5.5V | -40°C to 125°C | — | — | 20 |
| 3.0-3.6V | -40°C to 85°C | 15 | 10 | — |
| 3.0-3.6V | -40°C to 125°C | — | — | 10 |
| 2.5-3.0V | -40°C to 85°C | 10 | 7.5 | — |

TABLE 23-2: DC TEMPERATURE AND VOLTAGE SPECIFICATIONS

| DC CHARACTERISTICS | | | Standard Operating Conditions: 2.5V to 5.5V (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 ⁽¹⁾ | Max | Units | Conditions |
| Operating Voltage⁽²⁾ | | | | | | | |
| DC10 | VDD | Supply Voltage | 2.5 | — | 5.5 | V | Industrial temperature |
| DC11 | VDD | Supply Voltage | 3.0 | — | 5.5 | V | Extended temperature |
| DC12 | VDR | RAM Data Retention Voltage⁽³⁾ | — | 1.5 | — | V | |
| DC16 | VPOR | VDD Start Voltage to ensure internal Power-on Reset signal | — | VSS | — | V | |
| DC17 | SVDD | VDD Rise Rate to ensure internal Power-on Reset signal | 0.05 | | | V/ms | 0-5V in 0.1 sec 0-3V in 60 ms |

Note 1: Data in "Typ" column is at 5V, 25°C unless otherwise stated. Parameters are for design guidance only and are not tested.

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

3: This is the limit to which VDD can be lowered without losing RAM data.

TABLE 23-3: DC CHARACTERISTICS: OPERATING CURRENT (IDD)

| DC CHARACTERISTICS | | | Standard Operating Conditions: 2.5V to 5.5V (unless otherwise stated) Operating temperature $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$ for Industrial $-40^{\circ}\text{C} \leq T_A \leq +125^{\circ}\text{C}$ for Extended | | |
|--|------------------------|-----|---|------------|---|
| Parameter No. | Typical ⁽¹⁾ | Max | Units | Conditions | |
| Operating Current (IDD)⁽²⁾ | | | | | |
| DC20 | — | — | mA | -40°C | 3.3V 5V 1 MIPS EC mode |
| DC20a | 4 | — | mA | 25°C | |
| DC20b | — | — | mA | 85°C | |
| DC20c | — | — | mA | 125°C | |
| DC20d | — | — | mA | -40°C | |
| DC20e | 7 | — | mA | 25°C | |
| DC20f | — | — | mA | 85°C | |
| DC20g | — | — | mA | 125°C | |
| DC23 | — | — | mA | -40°C | 3.3V 5V 4 MIPS EC mode, 4X PLL |
| DC23a | 13 | — | mA | 25°C | |
| DC23b | — | — | mA | 85°C | |
| DC23c | — | — | mA | 125°C | |
| DC23d | — | — | mA | -40°C | |
| DC23e | 22 | — | mA | 25°C | |
| DC23f | — | — | mA | 85°C | |
| DC23g | — | — | mA | 125°C | |
| DC24 | — | — | mA | -40°C | 3.3V 5V 10 MIPS EC mode, 4X PLL |
| DC24a | 29 | — | mA | 25°C | |
| DC24b | — | — | mA | 85°C | |
| DC24c | — | — | mA | 125°C | |
| DC24d | — | — | mA | -40°C | |
| DC24e | 50 | — | mA | 25°C | |
| DC24f | — | — | mA | 85°C | |
| DC24g | — | — | mA | 125°C | |
| DC25 | — | — | mA | -40°C | 3.3V 5V 8 MIPS EC mode, 8X PLL |
| DC25a | 23 | — | mA | 25°C | |
| DC25b | — | — | mA | 85°C | |
| DC25c | — | — | mA | 125°C | |
| DC25d | — | — | mA | -40°C | |
| DC25e | 41 | — | mA | 25°C | |
| DC25f | — | — | mA | 85°C | |
| DC25g | — | — | mA | 125°C | |

Note 1: Data in “Typical” column is at 5V, 25°C unless otherwise stated. Parameters are for design guidance only and are not tested.

- 2:** The supply current is mainly 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: OSC1 driven with external square wave from rail to rail. All I/O pins are configured as Inputs and pulled to VDD. MCLR = VDD, WDT, FSCM, LVD and BOR are disabled. CPU, SRAM, Program Memory and Data Memory are operational. No peripheral modules are operating.

dsPIC30F

TABLE 23-3: DC CHARACTERISTICS: OPERATING CURRENT (IDD) (CONTINUED)

| DC CHARACTERISTICS | | | Standard Operating Conditions: 2.5V to 5.5V (unless otherwise stated) Operating temperature $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$ for Industrial $-40^{\circ}\text{C} \leq T_A \leq +125^{\circ}\text{C}$ for Extended | | |
|--|------------------------|-----|---|------------|--|
| Parameter No. | Typical ⁽¹⁾ | Max | Units | Conditions | |
| Operating Current (IDD)⁽²⁾ | | | | | |
| DC27 | — | — | mA | -40°C | 3.3V 5V 20 MIPS EC mode, 8X PLL |
| DC27a | 50 | — | mA | 25°C | |
| DC27b | — | — | mA | 85°C | |
| DC27c | — | — | mA | -40°C | |
| DC27d | 90 | — | mA | 25°C | |
| DC27e | — | — | mA | 85°C | |
| DC27f | — | — | mA | 125°C | |
| DC28 | — | — | mA | -40°C | 3.3V 5V 16 MIPS EC mode, 16X PLL |
| DC28a | 42 | — | mA | 25°C | |
| DC28b | — | — | mA | 85°C | |
| DC28c | — | — | mA | -40°C | |
| DC28d | 76 | — | mA | 25°C | |
| DC28e | — | — | mA | 85°C | |
| DC28f | — | — | mA | 125°C | |
| DC29 | — | — | mA | -40°C | 5V 30 MIPS EC mode, 16X PLL |
| DC29a | 146 | — | mA | 25°C | |
| DC29b | — | — | mA | 85°C | |
| DC29c | — | — | mA | 125°C | |
| DC30 | — | — | mA | -40°C | 3.3V 5V FRC (~ 2 MIPS) |
| DC30a | 7.0 | — | mA | 25°C | |
| DC30b | — | — | mA | 85°C | |
| DC30c | — | — | mA | 125°C | |
| DC30d | — | — | mA | -40°C | |
| DC30e | 12 | — | mA | 25°C | |
| DC30f | — | — | mA | 85°C | |
| DC30g | — | — | mA | 125°C | |
| DC31 | — | — | mA | -40°C | 3.3V 5V LPRC (~ 512 kHz) |
| DC31a | 1.5 | — | mA | 25°C | |
| DC31b | — | — | mA | 85°C | |
| DC31c | — | — | mA | 125°C | |
| DC31d | — | — | mA | -40°C | |
| DC31e | 2.5 | — | mA | 25°C | |
| DC31f | — | — | mA | 85°C | |
| DC31g | — | — | mA | 125°C | |

Note 1: Data in "Typical" column is at 5V, 25°C unless otherwise stated. Parameters are for design guidance only and are not tested.

- 2:** The supply current is mainly 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: OSC1 driven with external square wave from rail to rail. All I/O pins are configured as Inputs and pulled to VDD. MCLR = VDD, WDT, FSCM, LVD and BOR are disabled. CPU, SRAM, Program Memory and Data Memory are operational. No peripheral modules are operating.

TABLE 23-4: DC CHARACTERISTICS: IDLE CURRENT (IDLE)

| DC CHARACTERISTICS | | | Standard Operating Conditions: 2.5V to 5.5V (unless otherwise stated) Operating temperature $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$ for Industrial $-40^{\circ}\text{C} \leq T_A \leq +125^{\circ}\text{C}$ for Extended | |
|--|------------------------|-----|---|------------|
| Parameter No. | Typical ⁽¹⁾ | Max | Units | Conditions |
| Idle Current (IDLE): Core OFF Clock ON Base Current⁽²⁾ | | | | |
| DC40 | — | — | mA | -40°C |
| DC40a | 3 | — | mA | 25°C |
| DC40b | — | — | mA | 85°C |
| DC40c | — | — | mA | 125°C |
| DC40d | — | — | mA | -40°C |
| DC40e | 5 | — | mA | 25°C |
| DC40f | — | — | mA | 85°C |
| DC40g | — | — | mA | 125°C |
| DC43 | — | — | mA | -40°C |
| DC43a | 7.7 | — | mA | 25°C |
| DC43b | — | — | mA | 85°C |
| DC43c | — | — | mA | 125°C |
| DC43d | — | — | mA | -40°C |
| DC43e | 13 | — | mA | 25°C |
| DC43f | — | — | mA | 85°C |
| DC43g | — | — | mA | 125°C |
| DC44 | — | — | mA | -40°C |
| DC44a | 15 | — | mA | 25°C |
| DC44b | — | — | mA | 85°C |
| DC44c | — | — | mA | 125°C |
| DC44d | — | — | mA | -40°C |
| DC44e | 29 | — | mA | 25°C |
| DC44f | — | — | mA | 85°C |
| DC44g | — | — | mA | 125°C |
| DC45 | — | — | mA | -40°C |
| DC45a | 13 | — | mA | 25°C |
| DC45b | — | — | mA | 85°C |
| DC45c | — | — | mA | 125°C |
| DC45d | — | — | mA | -40°C |
| DC45e | 24 | — | mA | 25°C |
| DC45f | — | — | mA | 85°C |
| DC45g | — | — | mA | 125°C |

Note 1: Data in “Typical” column is at 5V, 25°C unless otherwise stated. Parameters are for design guidance only and are not tested.

2: Base IDLE current is measured with Core off, Clock on and all modules turned off.

dsPIC30F

TABLE 23-4: DC CHARACTERISTICS: IDLE CURRENT (I_{IDLE}) (CONTINUED)

| DC CHARACTERISTICS | | | Standard Operating Conditions: 2.5V to 5.5V (unless otherwise stated) Operating temperature -40°C ≤ T _A ≤ +85°C for Industrial -40°C ≤ T _A ≤ +125°C for Extended | |
|--|------------------------|-----|---|------------|
| Parameter No. | Typical ⁽¹⁾ | Max | Units | Conditions |
| Idle Current (I_{IDLE}): Core OFF Clock ON Base Current⁽²⁾ | | | | |
| DC47 | — | — | mA | -40°C |
| DC47a | 29 | — | mA | 25°C |
| DC47b | — | — | mA | 85°C |
| DC47c | — | — | mA | -40°C |
| DC47d | 52 | — | mA | 25°C |
| DC47e | — | — | mA | 85°C |
| DC47f | — | — | mA | 125°C |
| DC48 | — | — | mA | -40°C |
| DC48a | 24 | — | mA | 25°C |
| DC48b | — | — | mA | 85°C |
| DC48c | — | — | mA | -40°C |
| DC48d | 43 | — | mA | 25°C |
| DC48e | — | — | mA | 85°C |
| DC48f | — | — | mA | 125°C |
| DC49 | — | — | mA | -40°C |
| DC49a | 73 | — | mA | 25°C |
| DC49b | — | — | mA | 85°C |
| DC49c | — | — | mA | 125°C |
| DC50 | — | — | mA | -40°C |
| DC50a | 4.0 | — | mA | 25°C |
| DC50b | — | — | mA | 85°C |
| DC50c | — | — | mA | 125°C |
| DC50d | — | — | mA | -40°C |
| DC50e | 7.0 | — | mA | 25°C |
| DC50f | — | — | mA | 85°C |
| DC50g | — | — | mA | 125°C |
| DC51 | — | — | mA | -40°C |
| DC51a | 1.0 | — | mA | 25°C |
| DC51b | — | — | mA | 85°C |
| DC51c | — | — | mA | 125°C |
| DC51d | — | — | mA | -40°C |
| DC51e | 1.5 | — | mA | 25°C |
| DC51f | — | — | mA | 85°C |
| DC51g | — | — | mA | 125°C |

Note 1: Data in "Typical" column is at 5V, 25°C unless otherwise stated. Parameters are for design guidance only and are not tested.

2: Base I_{IDLE} current is measured with Core off, Clock on and all modules turned off.

TABLE 23-5: DC CHARACTERISTICS: POWER-DOWN CURRENT (IPD)

| DC CHARACTERISTICS | | | Standard Operating Conditions: 2.5V to 5.5V (unless otherwise stated) Operating temperature $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$ for Industrial $-40^{\circ}\text{C} \leq T_A \leq +125^{\circ}\text{C}$ for Extended | | |
|---|------------------------|-----|---|-----------------------|---|
| Parameter No. | Typical ⁽¹⁾ | Max | Units | Conditions | |
| Power Down Current (IPD)⁽²⁾ | | | | | |
| DC60 | — | — | μA | -40°C | 3.3V Base Power Down Current ⁽³⁾ |
| DC60a | 0.1 | — | μA | 25°C | |
| DC60b | — | — | μA | 85°C | |
| DC60c | — | — | μA | 125°C | |
| DC60d | — | — | μA | -40°C | |
| DC60e | 0.2 | — | μA | 25°C | |
| DC60f | — | — | μA | 85°C | |
| DC60g | — | — | μA | 125°C | 5V |
| DC61 | — | — | μA | -40°C | 3.3V Watchdog Timer Current: ΔI_{WDT} ⁽³⁾ |
| DC61a | 6.8 | — | μA | 25°C | |
| DC61b | — | — | μA | 85°C | |
| DC61c | — | — | μA | 125°C | |
| DC61d | — | — | μA | -40°C | |
| DC61e | 16 | — | μA | 25°C | |
| DC61f | — | — | μA | 85°C | |
| DC61g | — | — | μA | 125°C | 5V |
| DC62 | — | — | μA | -40°C | 3.3V Timer 1 w/32 kHz Crystal: $\Delta I_{\text{T}132}$ ⁽³⁾ |
| DC62a | 5.5 | — | μA | 25°C | |
| DC62b | — | — | μA | 85°C | |
| DC62c | — | — | μA | 125°C | |
| DC62d | — | — | μA | -40°C | |
| DC62e | 7.5 | — | μA | 25°C | |
| DC62f | — | — | μA | 85°C | |
| DC62g | — | — | μA | 125°C | 5V |
| DC63 | — | — | μA | -40°C | 3.3V BOR On: ΔI_{BOR} ⁽³⁾ |
| DC63a | 32 | — | μA | 25°C | |
| DC63b | — | — | μA | 85°C | |
| DC63c | — | — | μA | 125°C | |
| DC63d | — | — | μA | -40°C | |
| DC63e | 38 | — | μA | 25°C | |
| DC63f | — | — | μA | 85°C | |
| DC63g | — | — | μA | 125°C | 5V |

- Note 1:** Data in the Typical column is at 5V, 25°C unless otherwise stated. Parameters are for design guidance only and are not tested.
- 2:** Base IPD is measured with all peripherals and clocks shut down. All I/Os are configured as inputs and pulled high. LVD, BOR, WDT, etc. are all switched off.
- 3:** The Δ current is the additional current consumed when the module is enabled. This current should be added to the base IPD current.

dsPIC30F

TABLE 23-5: DC CHARACTERISTICS: POWER-DOWN CURRENT (IPD) (CONTINUED)

| DC CHARACTERISTICS | | | Standard Operating Conditions: 2.5V to 5.5V (unless otherwise stated) Operating temperature -40°C ≤ Ta ≤ +85°C for Industrial -40°C ≤ Ta ≤ +125°C for Extended | | | |
|---|------------------------|-----|---|------------|------|--|
| Parameter No. | Typical ⁽¹⁾ | Max | Units | Conditions | | |
| Power Down Current (IPD)⁽²⁾ | | | | | | |
| DC66 | — | — | μA | -40°C | 3.3V | Low Voltage Detect: ΔILVD ⁽³⁾ |
| DC66a | 25 | — | μA | 25°C | | |
| DC66b | — | — | μA | 85°C | | |
| DC66c | — | — | μA | 125°C | | |
| DC66d | — | — | μA | -40°C | 5V | |
| DC66e | 30 | — | μA | 25°C | | |
| DC66f | — | — | μA | 85°C | | |
| DC66g | — | — | μA | 125°C | | |

- Note 1:** Data in the Typical column is at 5V, 25°C unless otherwise stated. Parameters are for design guidance only and are not tested.
- 2:** Base IPD is measured with all peripherals and clocks shut down. All I/Os are configured as inputs and pulled high. LVD, BOR, WDT, etc. are all switched off.
- 3:** The Δ current is the additional current consumed when the module is enabled. This current should be added to the base IPD current.

TABLE 23-6: DC CHARACTERISTICS: I/O PIN INPUT SPECIFICATIONS

| DC CHARACTERISTICS | | | Standard Operating Conditions: 2.5V to 5.5V (unless otherwise stated) Operating temperature $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$ for Industrial $-40^{\circ}\text{C} \leq T_A \leq +125^{\circ}\text{C}$ for Extended | | | | |
|--------------------|----------|---|---|--------------------|--------------|---------------|---|
| Param No. | Symbol | Characteristic | Min | Typ ⁽¹⁾ | Max | Units | Conditions |
| DI10 | V_{IL} | Input Low Voltage⁽²⁾ I/O pins: with Schmitt Trigger buffer | V_{SS} | — | $0.2 V_{DD}$ | V | |
| DI15 | | $\overline{\text{MCLR}}$ | V_{SS} | — | $0.2 V_{DD}$ | V | |
| DI16 | | OSC1 (in XT, HS and LP modes) | V_{SS} | — | $0.2 V_{DD}$ | V | |
| DI17 | | OSC1 (in RC mode) ⁽³⁾ | V_{SS} | — | $0.3 V_{DD}$ | V | |
| DI18 | | SDA, SCL | TBD | — | TBD | V | SM bus disabled |
| DI19 | | SDA, SCL | TBD | — | TBD | V | SM bus enabled |
| DI20 | V_{IH} | Input High Voltage⁽²⁾ I/O pins: with Schmitt Trigger buffer | $0.8 V_{DD}$ | — | V_{DD} | V | |
| DI25 | | $\overline{\text{MCLR}}$ | $0.8 V_{DD}$ | — | V_{DD} | V | |
| DI26 | | OSC1 (in XT, HS and LP modes) | $0.7 V_{DD}$ | — | V_{DD} | V | |
| DI27 | | OSC1 (in RC mode) ⁽³⁾ | $0.9 V_{DD}$ | — | V_{DD} | V | |
| DI28 | | SDA, SCL | TBD | — | TBD | V | SM bus disabled |
| DI29 | | SDA, SCL | TBD | — | TBD | V | SM bus enabled |
| DI30 | ICNPU | CNxx Pull-up Current⁽²⁾ | 50 | 250 | 400 | μA | $V_{DD} = 5\text{V}, V_{PIN} = V_{SS}$ |
| DI31 | | | TBD | TBD | TBD | μA | $V_{DD} = 3\text{V}, V_{PIN} = V_{SS}$ |
| DI50 | I_{IL} | Input Leakage Current⁽²⁾⁽⁴⁾⁽⁵⁾ I/O ports | — | 0.01 | ± 1 | μA | $V_{SS} \leq V_{PIN} \leq V_{DD}$, Pin at hi-impedance |
| DI51 | | Analog input pins | — | 0.50 | — | μA | $V_{SS} \leq V_{PIN} \leq V_{DD}$, Pin at hi-impedance |
| DI55 | | $\overline{\text{MCLR}}$ | — | 0.05 | ± 5 | μA | $V_{SS} \leq V_{PIN} \leq V_{DD}$ |
| DI56 | | OSC1 | — | 0.05 | ± 5 | μA | $V_{SS} \leq V_{PIN} \leq V_{DD}$, XT, HS and LP Osc mode |

Note 1: Data in "Typ" column is at 5V, 25°C unless otherwise stated. Parameters are for design guidance only and are not tested.

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

3: In RC oscillator configuration, the OSC1/CLKI pin is a Schmitt Trigger input. It is not recommended that the dsPIC30F device be driven with an external clock while in RC mode.

4: The leakage current on the $\overline{\text{MCLR}}$ pin is strongly dependent on the applied voltage level. The specified levels represent normal operating conditions. Higher leakage current may be measured at different input voltages.

5: Negative current is defined as current sourced by the pin.

dsPIC30F

TABLE 23-7: DC CHARACTERISTICS: I/O PIN OUTPUT SPECIFICATIONS

| DC CHARACTERISTICS | | | Standard Operating Conditions: 2.5V to 5.5V (unless otherwise stated) Operating temperature $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$ for Industrial $-40^{\circ}\text{C} \leq T_A \leq +125^{\circ}\text{C}$ for Extended | | | | |
|--------------------|--------|--|---|--------------------|-----|-------|--|
| Param No. | Symbol | Characteristic | Min | Typ ⁽¹⁾ | Max | Units | Conditions |
| DO10 | VOL | Output Low Voltage⁽²⁾ I/O ports | — | — | 0.6 | V | $I_{OL} = 8.5 \text{ mA}$, $V_{DD} = 5\text{V}$ |
| DO16 | | OSC2/CLKOUT (RC or EC Osc mode) | — | — | TBD | V | $I_{OL} = 2.0 \text{ mA}$, $V_{DD} = 3\text{V}$ |
| DO20 | VOH | Output High Voltage⁽²⁾ I/O ports | $V_{DD} - 0.7$ TBD | — | — | V | $I_{OH} = -3.0 \text{ mA}$, $V_{DD} = 5\text{V}$ $I_{OH} = -2.0 \text{ mA}$, $V_{DD} = 3\text{V}$ |
| DO26 | | OSC2/CLKOUT (RC or EC Osc mode) | $V_{DD} - 0.7$ TBD | — | — | V | $I_{OH} = -1.3 \text{ mA}$, $V_{DD} = 5\text{V}$ $I_{OH} = -2.0 \text{ mA}$, $V_{DD} = 3\text{V}$ |
| DO50 | Cosco | Capacitive Loading Specs on Output Pins⁽²⁾ OSC2/SOSCO pin | — | — | 15 | pF | In XTL, XT, HS and LP modes when external clock is used to drive OSC1. |
| DO56 | Cio | All I/O pins and OSC2 | — | — | 50 | pF | RC or EC Osc mode |
| DO58 | CB | SCL, SDA | — | — | 400 | pF | In I ² C mode |

Note 1: Data in "Typ" column is at 5V, 25°C unless otherwise stated. Parameters are for design guidance only and are not tested.

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

FIGURE 23-1: LOW-VOLTAGE DETECT CHARACTERISTICS

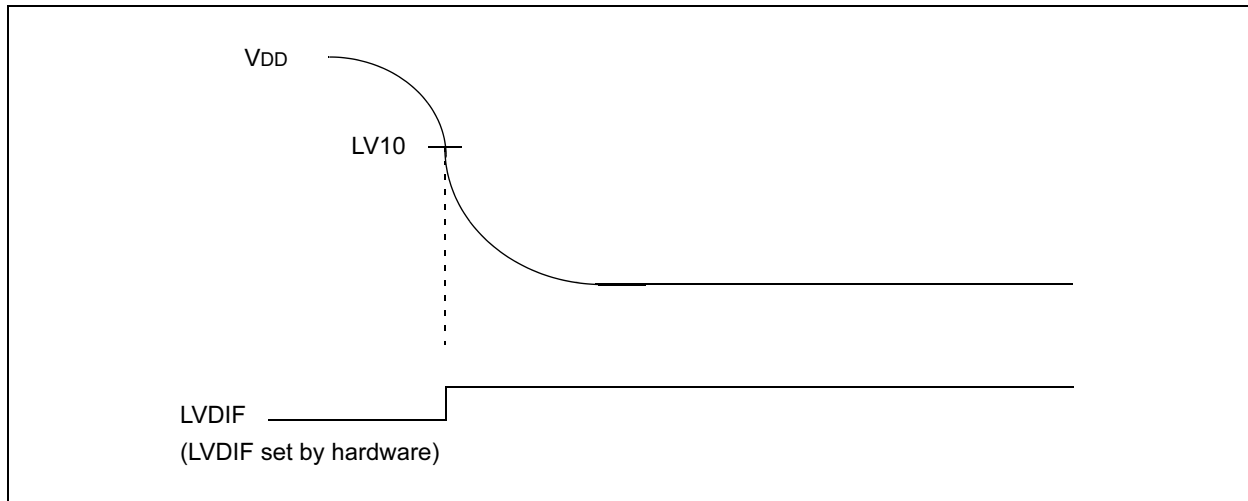


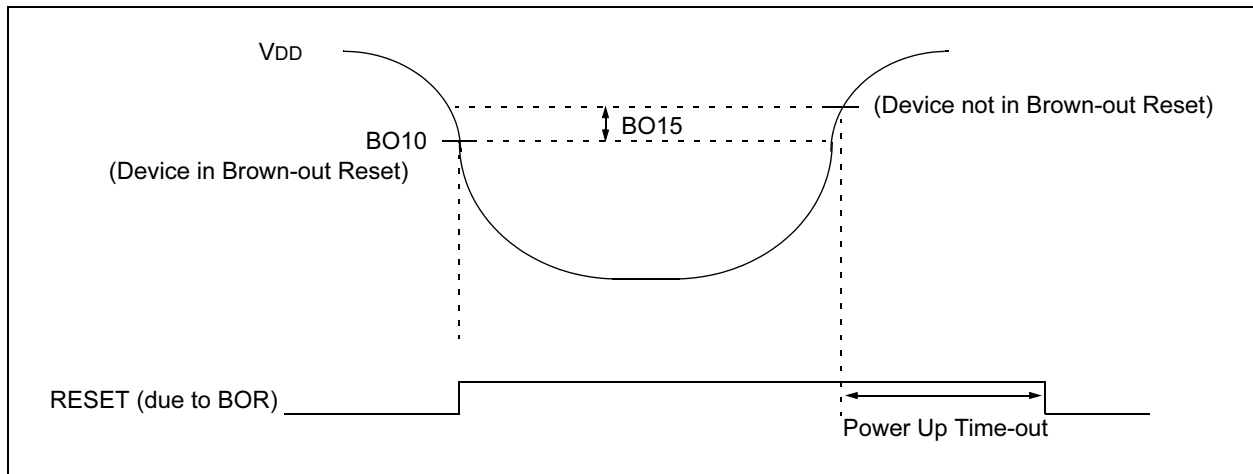
TABLE 23-8: ELECTRICAL CHARACTERISTICS: LVDL

| DC CHARACTERISTICS | | Standard Operating Conditions: 2.5V to 5.5V (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 | |
| LV10 | VPLVD | LVDL Voltage on VDD transition high to low | LVDL = 0000 ⁽²⁾ | — | — | — | V | |
| | | | LVDL = 0001 ⁽²⁾ | — | — | — | V | |
| | | | LVDL = 0010 ⁽²⁾ | — | — | — | V | |
| | | | LVDL = 0011 ⁽²⁾ | — | — | — | V | |
| | | | LVDL = 0100 | 2.50 | — | 2.65 | V | |
| | | | LVDL = 0101 | 2.70 | — | 2.86 | V | |
| | | | LVDL = 0110 | 2.80 | — | 2.97 | V | |
| | | | LVDL = 0111 | 3.00 | — | 3.18 | V | |
| | | | LVDL = 1000 | 3.30 | — | 3.50 | V | |
| | | | LVDL = 1001 | 3.50 | — | 3.71 | V | |
| | | | LVDL = 1010 | 3.60 | — | 3.82 | V | |
| | | | LVDL = 1011 | 3.80 | — | 4.03 | V | |
| | | | LVDL = 1100 | 4.00 | — | 4.24 | V | |
| | | | LVDL = 1101 | 4.20 | — | 4.45 | V | |
| LV15 | VLVDIN | External LVD input pin threshold voltage | LVDL = 1111 | — | — | — | V | |

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

Note 2: These values not in usable operating range.

FIGURE 23-2: BROWN-OUT RESET CHARACTERISTICS



dsPIC30F

TABLE 23-9: ELECTRICAL CHARACTERISTICS: BOR

| DC CHARACTERISTICS | | | Standard Operating Conditions: 2.5V to 5.5V (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 ⁽¹⁾ | Max | Units | Conditions | |
| BO10 | VBOR | BOR Voltage ⁽²⁾ on VDD transition high to low | BORV = 00 ⁽³⁾ | — | — | — | V | Not in operating range |
| | | | BORV = 01 | 2.7 | — | 2.86 | V | |
| | | | BORV = 10 | 4.2 | — | 4.46 | V | |
| | | | BORV = 11 | 4.5 | — | 4.78 | V | |
| BO15 | VBHYS | | — | 5 | — | mV | | |

- Note 1:** Data in “Typ” column is at 5V, 25°C unless otherwise stated. Parameters are for design guidance only and are not tested.
2: These parameters are characterized but not tested in manufacturing.
3: 00 values not in usable operating range.

TABLE 23-10: DC CHARACTERISTICS: PROGRAM AND EEPROM

| DC CHARACTERISTICS | | | Standard Operating Conditions: 2.5V to 5.5V (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 ⁽¹⁾ | Max | Units | Conditions | |
| Data EEPROM Memory⁽²⁾ | | | | | | | | |
| D120 | ED | Byte Endurance | 100K | 1M | — | E/W | -40°C ≤ TA ≤ +85°C Using NVMCON to read/write VMIN = Minimum operating voltage | |
| D121 | VDRW | VDD for Read/Write | VMIN | — | 5.5 | V | | |
| D122 | TDEW | Erase/Write Cycle Time | — | 2 | — | ms | | |
| D123 | TRETD | Characteristic Retention | 40 | 100 | — | Year | | |
| D124 | IDEW | IDD during Programming | — | 10 | 30 | mA | Row erase | |
| Program FLASH Memory⁽²⁾ | | | | | | | | |
| D130 | EP | Cell Endurance | 10K | 100K | — | E/W | -40°C ≤ TA ≤ +85°C VMIN = Minimum operating voltage | |
| D131 | VPR | VDD for Read | VMIN | — | 5.5 | V | | |
| D132 | VEB | VDD for Bulk Erase | 3.0 | — | 5.5 | V | | |
| D133 | VPEW | VDD for Erase/Write | 3.0 | — | 5.5 | V | | |
| D134 | TPEW | Erase/Write Cycle Time | — | 2 | — | ms | Provided no other specifications are violated | |
| D135 | TRETD | Characteristic Retention | 40 | 100 | — | Year | | |
| D136 | TEB | ICSP Block Erase Time | — | 4 | — | ms | | |
| D137 | IPEW | IDD during Programming | — | 10 | 30 | mA | Row erase | |
| D138 | IEB | IDD during Programming | — | 10 | 30 | mA | Bulk erase | |

- Note 1:** Data in “Typ” column is at 5V, 25°C unless otherwise stated.
2: These parameters are characterized but not tested in manufacturing.

23.2 AC Characteristics and Timing Parameters

The information contained in this section defines dsPIC30F AC characteristics and timing parameters.

TABLE 23-11: TEMPERATURE AND VOLTAGE SPECIFICATIONS – AC

| | |
|---------------------------|---|
| AC CHARACTERISTICS | Standard Operating Conditions: 2.5V to 5.5V (unless otherwise stated) |
| | Operating temperature $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$ for Industrial $-40^{\circ}\text{C} \leq T_A \leq +125^{\circ}\text{C}$ for Extended Operating voltage V_{DD} range as described in DC Spec Section 23.0. |

FIGURE 23-3: LOAD CONDITIONS FOR DEVICE TIMING SPECIFICATIONS

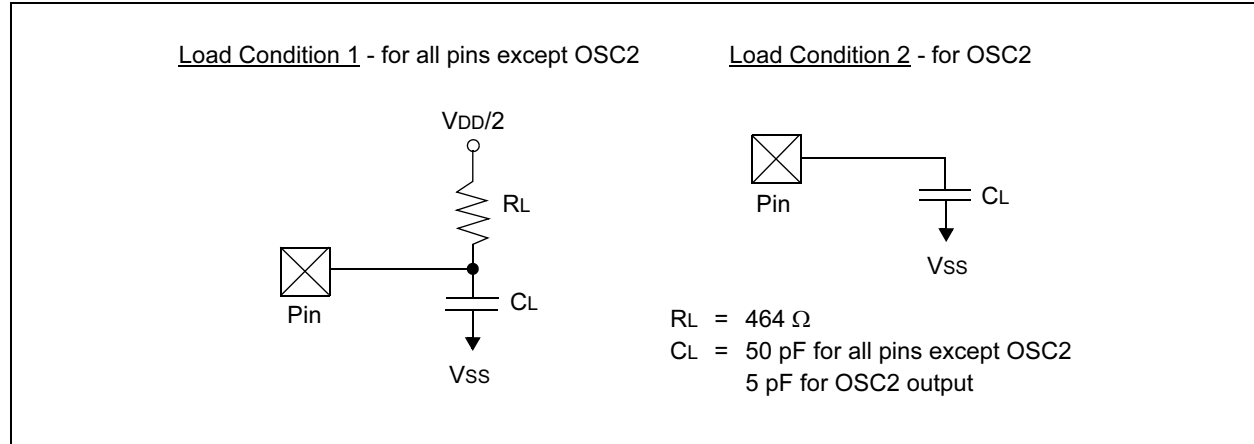
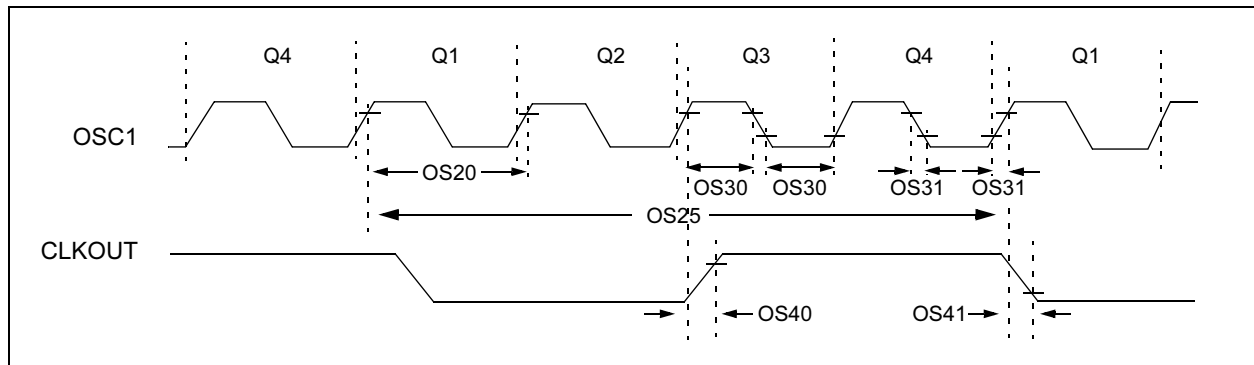


FIGURE 23-4: EXTERNAL CLOCK TIMING



dsPIC30F

TABLE 23-12: EXTERNAL CLOCK TIMING REQUIREMENTS

| AC CHARACTERISTICS | | | Standard Operating Conditions: 2.5V to 5.5V (unless otherwise stated) Operating temperature $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$ for Industrial $-40^{\circ}\text{C} \leq T_A \leq +125^{\circ}\text{C}$ for Extended | | | | |
|--------------------|---------------|--|---|--------------------|-----------------|-------|-----------------------------------|
| Param No. | Symbol | Characteristic | Min | Typ ⁽¹⁾ | Max | Units | Conditions |
| OS10 | Fosc | External CLKIN Frequency ⁽²⁾ (External clocks allowed only in EC mode) | DC | — | 40 | MHz | EC |
| | | | 4 | — | 10 | MHz | EC with 4x PLL |
| 4 | — | | 10 | MHz | EC with 8x PLL | | |
| 4 | — | | 7.5 | MHz | EC with 16x PLL | | |
| OS10 | Fosc | Oscillator Frequency ⁽²⁾ | DC | — | 4 | MHz | RC |
| | | | 0.4 | — | 4 | MHz | XTL |
| | | | 4 | — | 10 | MHz | XT |
| | | | 4 | — | 10 | MHz | XT with 4x PLL |
| | | | 4 | — | 10 | MHz | XT with 8x PLL |
| | | | 4 | — | 7.5 | MHz | XT with 16x PLL |
| | | | 10 | — | 25 | MHz | HS |
| | | | 31 | — | 33 | kHz | LP |
| | | | — | 8 | — | MHz | FRC internal |
| | | | — | 512 | — | kHz | LPRC internal |
| OS20 | Tosc | Tosc = 1/Fosc | — | — | — | — | See parameter OS10 for Fosc value |
| OS25 | Tcy | Instruction Cycle Time ⁽²⁾⁽³⁾ | 33 | — | DC | ns | See Table 23-14 |
| OS30 | TosL, TosH | External Clock ⁽²⁾ in (OSC1) High or Low Time | .45 x Tosc | — | — | ns | EC |
| OS31 | TosR, TosF | External Clock ⁽²⁾ in (OSC1) Rise or Fall Time | — | — | 20 | ns | EC |
| OS40 | TckR | CLKOUT Rise Time ⁽²⁾⁽⁴⁾ | — | 6 | 10 | ns | |
| OS41 | TckF | CLKOUT Fall Time ⁽²⁾⁽⁴⁾ | — | 6 | 10 | ns | |

- Note 1:** Data in “Typ” column is at 5V, 25°C unless otherwise stated. Parameters are for design guidance only and are not tested.
- 2:** These parameters are characterized but not tested in manufacturing.
- 3:** Instruction cycle period (Tcy) equals four times the input oscillator time-base period. All specified values are based on characterization data for that particular oscillator type under standard operating conditions with the device executing code. Exceeding these specified limits may result in an unstable oscillator operation and/or higher than expected current consumption. All devices are tested to operate at “min.” values with an external clock applied to the OSC1/CLKI pin. When an external clock input is used, the “Max.” cycle time limit is “DC” (no clock) for all devices.
- 4:** Measurements are taken in EC or ERC modes. The CLKOUT signal is measured on the OSC2 pin. CLKOUT is low for the Q1-Q2 period (1/2 Tcy) and high for the Q3-Q4 period (1/2 Tcy).

TABLE 23-13: PLL CLOCK TIMING SPECIFICATIONS (V_{DD} = 2.5 TO 5.5 V)

| AC CHARACTERISTICS | | Standard Operating Conditions: 2.5V to 5.5V (unless otherwise stated) Operating temperature -40°C ≤ T _A ≤ +85°C for Industrial -40°C ≤ T _A ≤ +125°C for Extended | | | | | |
|--------------------|--------|---|-----|--------------------|-----|-------|-----------------------------|
| Param No. | Symbol | Characteristic ⁽¹⁾ | Min | Typ ⁽²⁾ | Max | Units | Conditions |
| OS50 | FPLLI | PLL Input Frequency Range ⁽²⁾ | 4 | — | 10 | MHz | EC, XT modes with PLL |
| OS51 | FSYS | On-chip PLL Output ⁽²⁾ | 16 | — | 120 | MHz | EC, XT modes with PLL |
| OS52 | TLOC | PLL Start-up Time (Lock Time) | — | 20 | 50 | μs | |
| OS53 | DCLK | CLKOUT Stability (Jitter) | TBD | 1 | TBD | % | Measured over 100 ms period |

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

Note 2: Data in “Typ” column is at 5V, 25°C unless otherwise stated. Parameters are for design guidance only and are not tested.

TABLE 23-14: INTERNAL CLOCK TIMING EXAMPLES

| Clock Oscillator Mode | Fosc (MHz) ⁽¹⁾ | T _{CY} (μsec) ⁽²⁾ | MIPS ⁽³⁾ w/o PLL | MIPS ⁽³⁾ w PLL x4 | MIPS ⁽³⁾ w PLL x8 | MIPS ⁽³⁾ w PLL x16 |
|-----------------------|---------------------------|---------------------------------------|-----------------------------|------------------------------|------------------------------|-------------------------------|
| EC | 0.200 | 20.0 | 0.05 | — | — | — |
| | 4 | 1.0 | 1.0 | 4.0 | 8.0 | 16.0 |
| | 10 | 0.4 | 2.5 | 10.0 | 20.0 | — |
| | 25 | 0.16 | 25.0 | — | — | — |
| XT | 4 | 1.0 | 1.0 | 4.0 | 8.0 | 16.0 |
| | 10 | 0.4 | 2.5 | 10.0 | 20.0 | — |

Note 1: Assumption: Oscillator Postscaler is divide by 1.

Note 2: Instruction Execution Cycle Time: T_{CY} = 1 / MIPS.

Note 3: Instruction Execution Frequency: MIPS = (Fosc * PLLx) / 4 [since there are 4 Q clocks per instruction cycle].

TABLE 23-15: INTERNAL RC ACCURACY

| AC CHARACTERISTICS | | Standard Operating Conditions: 2.5V to 5.5V (unless otherwise stated) Operating temperature -40°C ≤ T _A ≤ +85°C for Industrial -40°C ≤ T _A ≤ +125°C for Extended | | | | | |
|--------------------|--------------------------------------|---|-----|-----|-------|----------------|------------------------|
| Param No. | Characteristic | Min | Typ | Max | Units | Conditions | |
| | FRC @ Freq = 8 MHz ⁽¹⁾ | | | | | | |
| F16 | | TBD | — | TBD | % | -40°C to +85°C | V _{DD} = 3.3V |
| F19 | | TBD | — | TBD | % | -40°C to +85°C | V _{DD} = 5V |
| | LPRC @ Freq = 512 kHz ⁽²⁾ | | | | | | |
| F20 | | TBD | — | TBD | % | -40°C to +85°C | V _{DD} = 3V |
| F21 | | TBD | — | TBD | % | -40°C to +85°C | V _{DD} = 5V |

Note 1: Frequency calibrated at 25°C and 5V. TUN bits can be used to compensate for temperature drift.

Note 2: LPRC frequency after calibration.

Note 3: Change of LPRC frequency as V_{DD} changes.

dsPIC30F

FIGURE 23-5: CLKOUT AND I/O TIMING CHARACTERISTICS

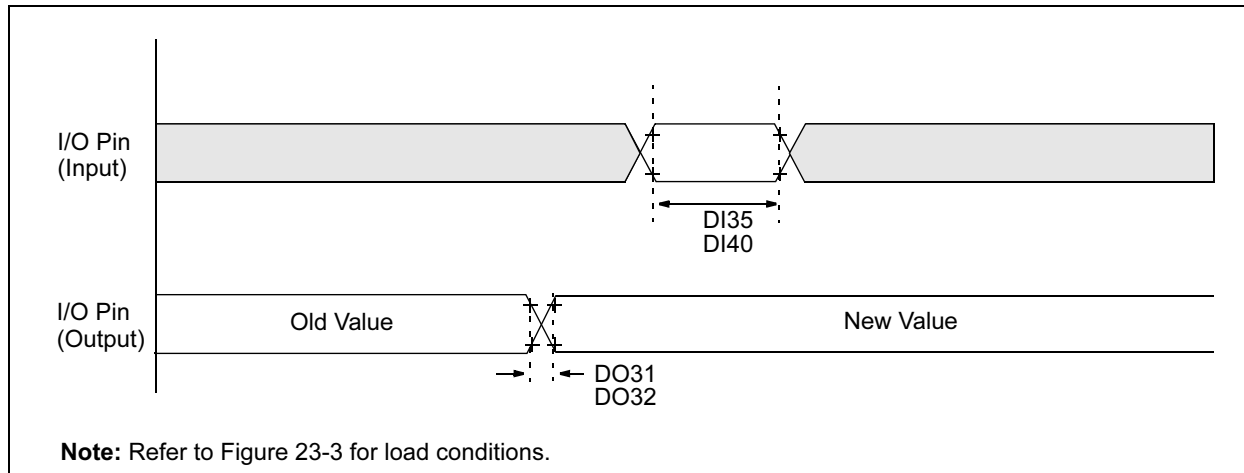
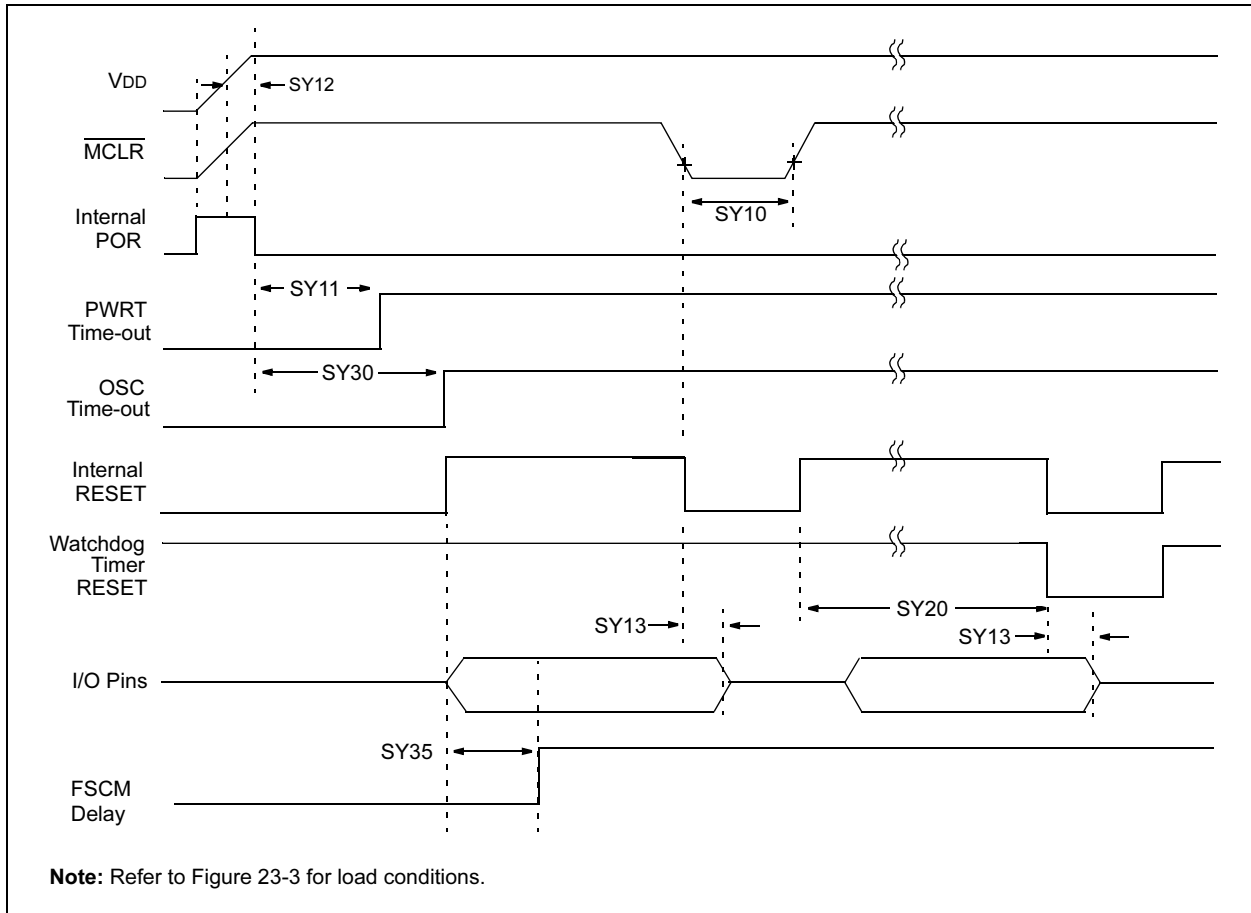


TABLE 23-16: CLKOUT AND I/O TIMING REQUIREMENTS

| AC CHARACTERISTICS | | Standard Operating Conditions: 2.5V to 5.5V (unless otherwise stated) Operating temperature -40°C ≤ TA ≤ +85°C for Industrial -40°C ≤ TA ≤ +125°C for Extended | | | | | |
|--------------------|--------|---|-------|--------------------|-----|-------|------------|
| Param No. | Symbol | Characteristic ⁽¹⁾⁽²⁾⁽³⁾ | Min | Typ ⁽⁴⁾ | 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 | — | — | ns | — |

- Note 1:** These parameters are asynchronous events not related to any internal clock edges
Note 2: Measurements are taken in RC mode and EC mode where CLKOUT output is 4 x TOSC.
Note 3: These parameters are characterized but not tested in manufacturing.
Note 4: Data in “Typ” column is at 5V, 25°C unless otherwise stated.

FIGURE 23-6: RESET, WATCHDOG TIMER, OSCILLATOR START-UP TIMER AND POWER-UP TIMER TIMING CHARACTERISTICS



dsPIC30F

TABLE 23-17: RESET, WATCHDOG TIMER, OSCILLATOR START-UP TIMER, POWER-UP TIMER AND BROWN-OUT RESET TIMING REQUIREMENTS

| AC CHARACTERISTICS | | | Standard Operating Conditions: 2.5V to 5.5V (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 | MCLR Pulse Width (low) | 2 | — | — | μs | -40°C to +85°C |
| SY11 | TPWRT | Power-up Timer Period | TBD TBD TBD TBD | 0 4 16 64 | TBD TBD TBD TBD | 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 Hi-impedance from MCLR Low or Watchdog Timer Reset | — | — | 100 | ns | |
| SY20 | TWDT1 | Watchdog Timer Time-out Period (No Prescaler) | 1.8 | 2.0 | 2.2 | ms | VDD = 5V, -40°C to +85°C |
| | TWDT2 | | 1.9 | 2.1 | 2.3 | ms | VDD = 3V, -40°C to +85°C |
| SY25 | TBOR | Brown-out Reset Pulse Width ⁽³⁾ | 100 | — | — | μs | VDD ≤ VBOR (D034) |
| SY30 | TOST | Oscillation Start-up Timer Period | — | 1024 TOSC | — | — | TOSC = OSC1 period |
| SY35 | TFSCM | Fail-Safe Clock Monitor Delay | — | 100 | — | μs | -40°C to +85°C |

- Note 1:** These parameters are characterized but not tested in manufacturing.
Note 2: Data in "Typ" column is at 5V, 25°C unless otherwise stated.
Note 3: Refer to Figure 23-2 and Table 23-9 for BOR.

FIGURE 23-7: BAND GAP START-UP TIME CHARACTERISTICS

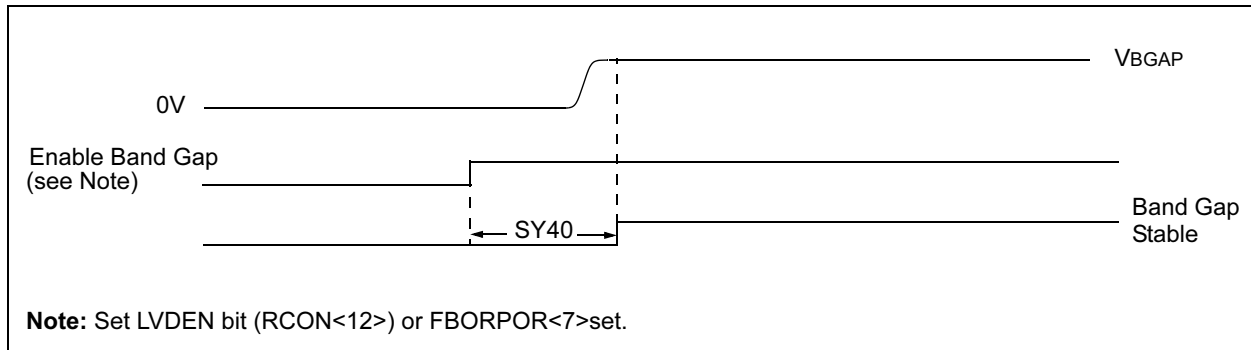


TABLE 23-18: BAND GAP START-UP TIME REQUIREMENTS

| AC CHARACTERISTICS | | | Standard Operating Conditions: 2.5V to 5.5V (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 |
| SY40 | TBGAP | Band Gap Start-up Time | — | 20 | 50 | μs | Defined as the time between the instant that the band gap is enabled and the moment that the band gap reference voltage is stable. RCON<13>Status bit |

- Note 1:** These parameters are characterized but not tested in manufacturing.
Note 2: Data in "Typ" column is at 5V, 25°C unless otherwise stated.

FIGURE 23-8: TYPE A, B AND C TIMER EXTERNAL CLOCK TIMING CHARACTERISTICS

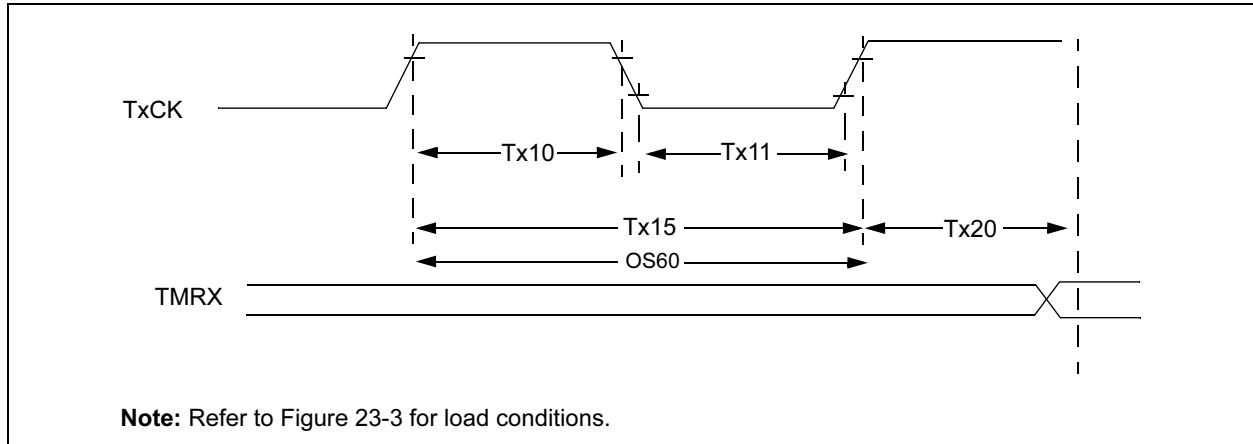


TABLE 23-19: TYPE A TIMER (TIMER1) EXTERNAL CLOCK TIMING REQUIREMENTS

| AC CHARACTERISTICS | | Standard Operating Conditions: 2.5V to 5.5V (unless otherwise stated) Operating temperature $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$ for Industrial $-40^{\circ}\text{C} \leq T_A \leq +125^{\circ}\text{C}$ for Extended | | | | | | |
|--------------------|-----------------------|---|-----------------------------|--|-----|-------------|-------|------------------------------------|
| Param No. | Symbol | Characteristic | | Min | Typ | Max | Units | Conditions |
| TA10 | T _{TXH} | TxCK High Time | Synchronous, no prescaler | $0.5 T_{CY} + 20$ | — | — | ns | Must also meet parameter TA15 |
| | | | Synchronous, with prescaler | 10 | — | — | ns | |
| | | | Asynchronous | 10 | — | — | ns | |
| TA11 | T _{TXL} | TxCK Low Time | Synchronous, no prescaler | $0.5 T_{CY} + 20$ | — | — | ns | Must also meet parameter TA15 |
| | | | Synchronous, with prescaler | 10 | — | — | ns | |
| | | | Asynchronous | 10 | — | — | ns | |
| TA15 | T _{TXP} | TxCK Input Period | Synchronous, no prescaler | $T_{CY} + 10$ | — | — | ns | N = prescale value (1, 8, 64, 256) |
| | | | Synchronous, with prescaler | Greater of: 20 ns or $(T_{CY} + 40)/N$ | — | — | — | |
| | | | Asynchronous | 20 | — | — | ns | |
| OS60 | F _{t1} | SOSCI/T1CK oscillator input frequency range (oscillator enabled by setting bit TCS (T1CON, bit 1)) | | DC | — | 50 | kHz | |
| TA20 | T _{CKEXTMRL} | Delay from External TQCK Clock Edge to Timer Increment | | $2 T_{OSC}$ | | $6 T_{OSC}$ | — | |

Note: Timer1 is a Type A.

dsPIC30F

TABLE 23-20: TYPE B TIMER (TIMER2 AND TIMER4) EXTERNAL CLOCK TIMING REQUIREMENTS

| AC CHARACTERISTICS | | | Standard Operating Conditions: 2.5V to 5.5V (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 |
| 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 + 10 | — | — | ns | N = prescale value (1, 8, 64, 256) |
| | | | Synchronous, with prescaler | Greater of: 20 ns or (Tcy + 40)/N | | | | |
| TB20 | TCKEXTMRL | Delay from External TQCK Clock Edge to Timer Increment | | 2 Tosc | — | 6 Tosc | — | |

Note: Timer2 and Timer4 are Type B.

TABLE 23-21: TYPE C TIMER (TIMER3 AND TIMER5) EXTERNAL CLOCK TIMING REQUIREMENTS

| AC CHARACTERISTICS | | | Standard Operating Conditions: 2.5V to 5.5V (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 |
| TC10 | TtxH | TxCK High Time | Synchronous | 0.5 Tcy + 20 | — | — | ns | Must also meet parameter TC15 |
| TC11 | TtxL | TxCK Low Time | Synchronous | 0.5 Tcy + 20 | — | — | ns | Must also meet parameter TC15 |
| TC15 | TtxP | TxCK Input Period | Synchronous, no prescaler | Tcy + 10 | — | — | ns | N = prescale value (1, 8, 64, 256) |
| | | | Synchronous, with prescaler | Greater of: 20 ns or (Tcy + 40)/N | | | | |
| TC20 | TCKEXTMRL | Delay from External TQCK Clock Edge to Timer Increment | | 2 Tosc | — | 6 Tosc | — | |

Note: Timer3 and Timer5 are Type C.

FIGURE 23-9: INPUT CAPTURE (CAPx) TIMING CHARACTERISTICS

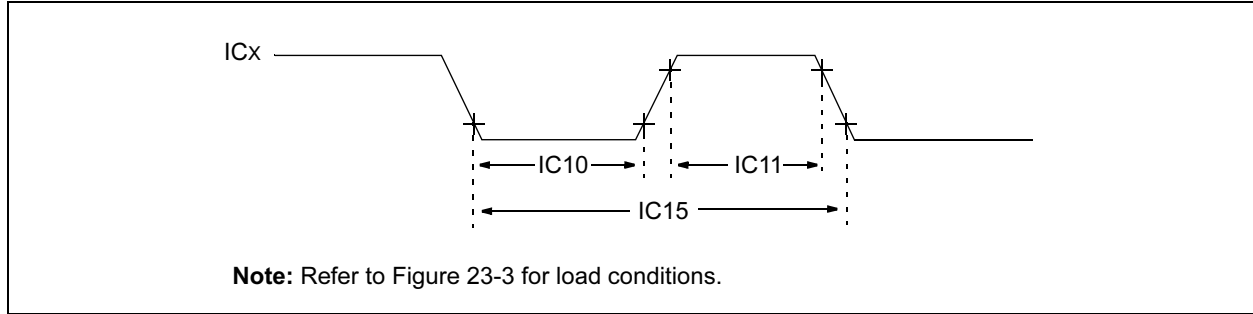


TABLE 23-22: INPUT CAPTURE TIMING REQUIREMENTS

| AC CHARACTERISTICS | | Standard Operating Conditions: 2.5V to 5.5V (unless otherwise stated) Operating temperature $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$ for Industrial $-40^{\circ}\text{C} \leq T_A \leq +125^{\circ}\text{C}$ for Extended | | | | | |
|--------------------|--------|---|----------------|---------------------|-----|-------|-------------------------------|
| Param No. | Symbol | Characteristic ⁽¹⁾ | | Min | Max | Units | Conditions |
| IC10 | TccL | ICx Input Low Time | No Prescaler | $0.5 T_{CY} + 20$ | — | ns | |
| | | | With Prescaler | 10 | — | ns | |
| IC11 | TccH | ICx Input High Time | No Prescaler | $0.5 T_{CY} + 20$ | — | ns | |
| | | | With Prescaler | 10 | — | ns | |
| IC15 | TccP | ICx Input Period | | $(2 T_{CY} + 40)/N$ | — | ns | N = prescale value (1, 4, 16) |

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

FIGURE 23-10: OUTPUT COMPARE MODULE (OCx) TIMING CHARACTERISTICS

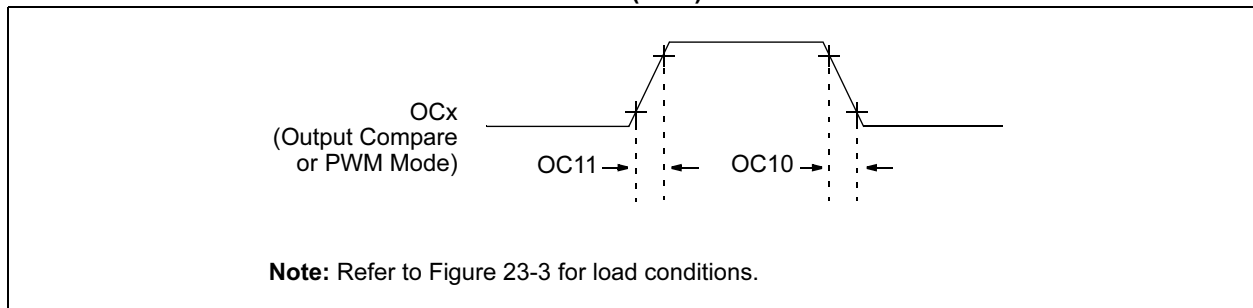


TABLE 23-23: OUTPUT COMPARE MODULE TIMING REQUIREMENTS

| AC CHARACTERISTICS | | Standard Operating Conditions: 2.5V to 5.5V (unless otherwise stated) Operating temperature $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$ for Industrial $-40^{\circ}\text{C} \leq T_A \leq +125^{\circ}\text{C}$ for Extended | | | | | |
|--------------------|--------|---|-----|--------------------|-----|-------|------------|
| Param No. | Symbol | Characteristic ⁽¹⁾ | Min | Typ ⁽²⁾ | Max | Units | Conditions |
| OC10 | TccF | OCx Output Fall Time | — | 10 | 25 | ns | — |
| OC11 | TccR | OCx Output Rise Time | — | 10 | 25 | ns | — |

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

Note 2: Data in "Typ" column is at 5V, 25°C unless otherwise stated. Parameters are for design guidance only and are not tested.

dsPIC30F

FIGURE 23-11: OC/PWM MODULE TIMING CHARACTERISTICS

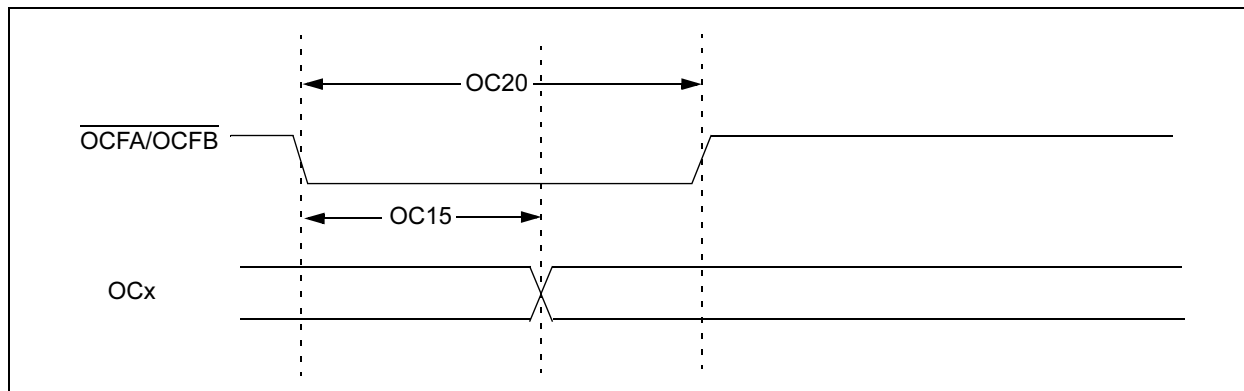


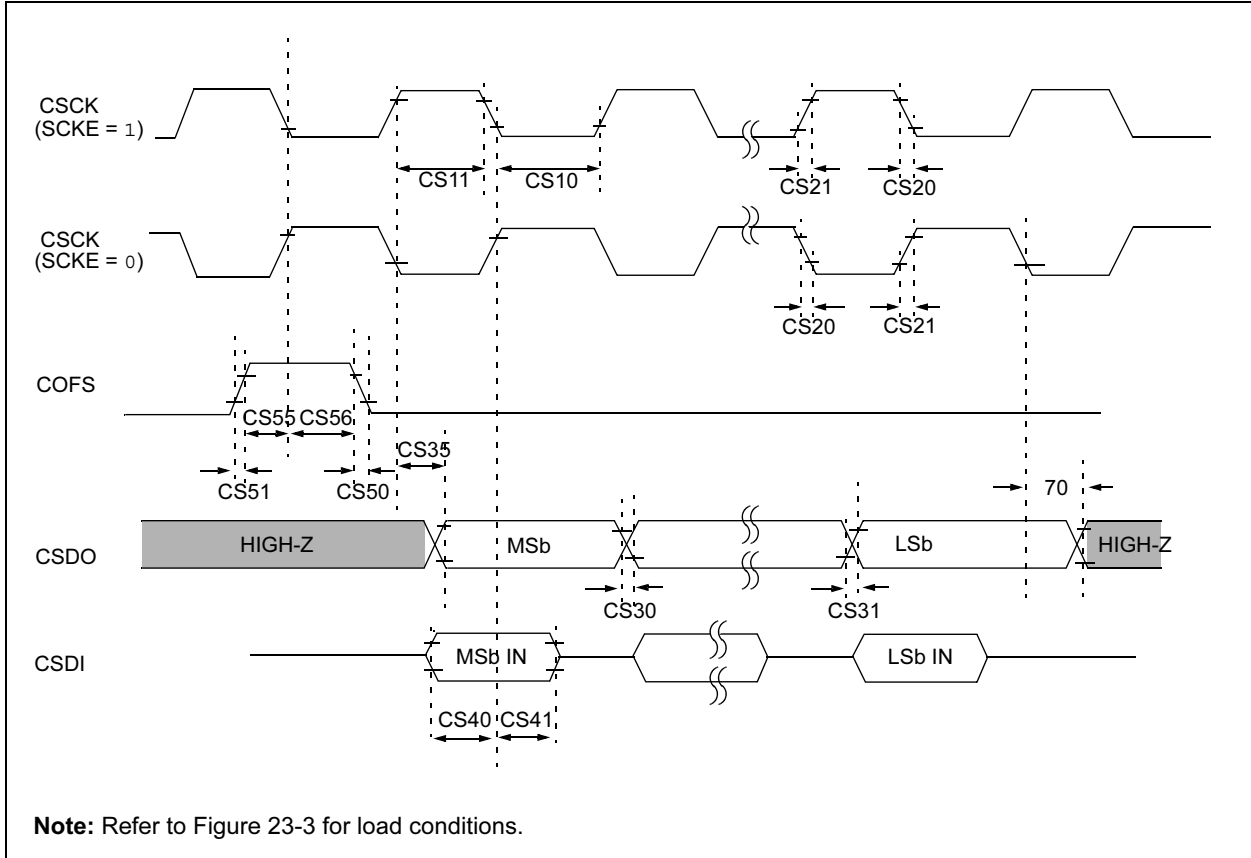
TABLE 23-24: SIMPLE OC/PWM MODE TIMING REQUIREMENTS

| AC CHARACTERISTICS | | | | Standard Operating Conditions: 2.5V to 5.5V (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 | |
| OC15 | TFD | Fault Input to PWM I/O Change | — | — | 25 | ns | VDD = 3V | -40°C to +85°C |
| | | | | | TBD | ns | VDD = 5V | |
| OC20 | TFLT | Fault Input Pulse Width | — | — | 50 | ns | VDD = 3V | -40°C to +85°C |
| | | | | | TBD | ns | VDD = 5V | |

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

Note 2: Data in “Typ” column is at 5V, 25°C unless otherwise stated. Parameters are for design guidance only and are not tested.

FIGURE 23-12: DCI MODULE (MULTICHANNEL, I²S MODES) TIMING CHARACTERISTICS



dsPIC30F

TABLE 23-25: DCI MODULE (MULTICHANNEL, I²S MODES) TIMING REQUIREMENTS

| AC CHARACTERISTICS | | | Standard Operating Conditions: 2.5V to 5.5V (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 |
| CS10 | TcSCKL | CSCK Input Low Time (CSCK pin is an input) | T _{CY} / 2 + 20 | — | — | ns | — |
| | | CSCK Output Low Time ⁽³⁾ (CSCK pin is an output) | 30 | — | — | ns | — |
| CS11 | TcSCKH | CSCK Input High Time (CSCK pin is an input) | T _{CY} / 2 + 20 | — | — | ns | — |
| | | CSCK Output High Time ⁽³⁾ (CSCK pin is an output) | 30 | — | — | ns | — |
| CS20 | TcSCKF | CSCK Output Fall Time ⁽⁴⁾ (CSCK pin is an output) | — | 10 | 25 | ns | — |
| CS21 | TcSCKR | CSCK Output Rise Time ⁽⁴⁾ (CSCK pin is an output) | — | 10 | 25 | ns | — |
| CS30 | TcSDOF | CSDO Data Output Fall Time ⁽⁴⁾ | — | 10 | 25 | ns | — |
| CS31 | TcSDOR | CSDO Data Output Rise Time ⁽⁴⁾ | — | 10 | 25 | ns | — |
| CS35 | TdV | Clock edge to CSDO data valid | — | — | 10 | ns | — |
| CS36 | TdIV | Clock edge to CSDO tri-stated | 10 | — | 20 | ns | — |
| CS40 | TcSDI | Setup time of CSDI data input to CSCK edge (CSCK pin is input or output) | 20 | — | — | ns | — |
| CS41 | THCSDI | Hold time of CSDI data input to CSCK edge (CSCK pin is input or output) | 20 | — | — | ns | — |
| CS50 | TcoFSF | COFS Fall Time (COFS pin is output) | — | 10 | 25 | ns | Note 1 |
| CS51 | TcoFSR | COFS Rise Time (COFS pin is output) | — | 10 | 25 | ns | Note 1 |
| CS55 | TscoFS | Setup time of COFS data input to CSCK edge (COFS pin is input) | 20 | — | — | ns | — |
| CS56 | THCOFS | Hold time of COFS data input to CSCK edge (COFS pin is input) | 20 | — | — | ns | — |

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

- 2: Data in "Typ" column is at 5V, 25°C unless otherwise stated. Parameters are for design guidance only and are not tested.
- 3: The minimum clock period for CSCK is 100 ns. Therefore, the clock generated in Master mode must not violate this specification.
- 4: Assumes 50 pF load on all DCI pins.

FIGURE 23-13: DCI MODULE (AC-LINK MODE) TIMING CHARACTERISTICS

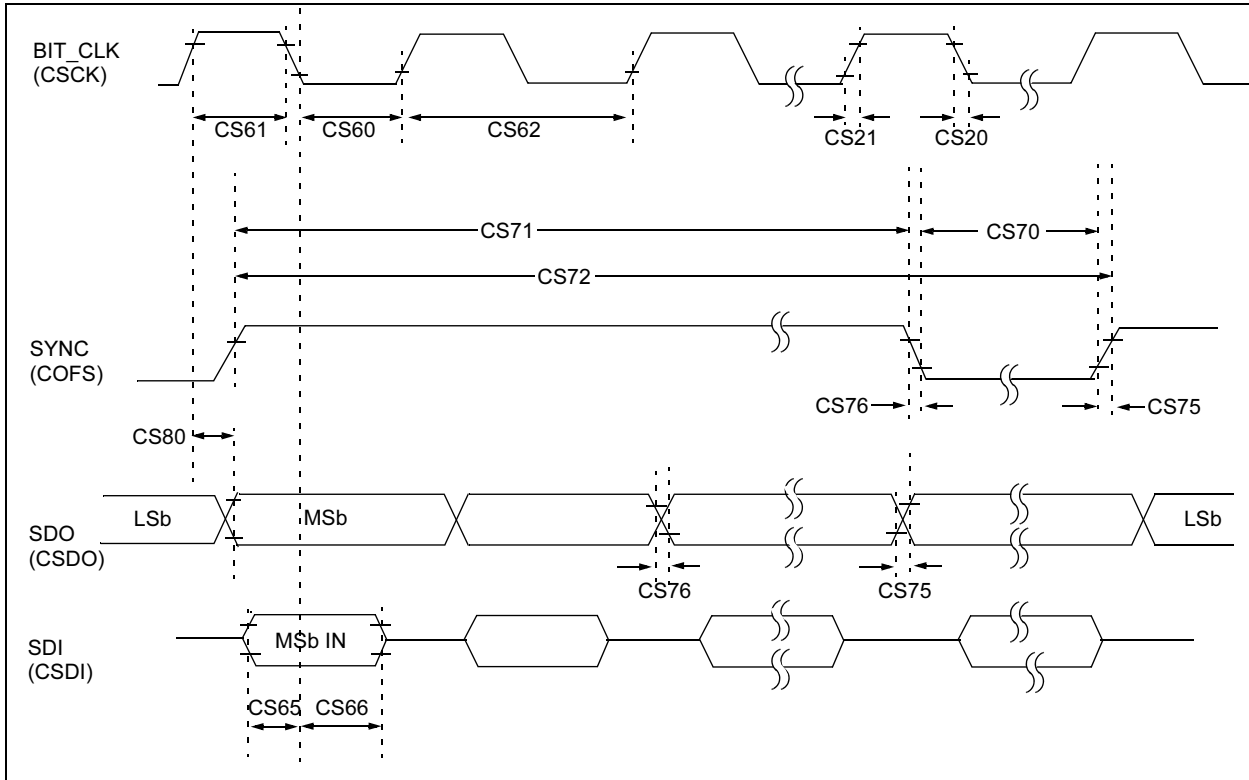


TABLE 23-26: DCI MODULE (AC-LINK MODE) TIMING REQUIREMENTS

| AC CHARACTERISTICS | | | Standard Operating Conditions: 2.5V to 5.5V (unless otherwise stated) Operating temperature -40°C ≤ TA ≤ +85°C for Industrial -40°C ≤ TA ≤ +125°C for Extended | | | | |
|--------------------|--------|--|---|--------------------|-----|-------|-------------------------|
| Param No. | Symbol | Characteristic ⁽¹⁾⁽²⁾ | Min | Typ ⁽³⁾ | Max | Units | Conditions |
| CS60 | TBCLKL | BIT_CLK Low Time | 36 | 40.7 | 45 | ns | — |
| CS61 | TBCLKH | BIT_CLK High Time | 36 | 40.7 | 45 | ns | — |
| CS62 | TBCLK | BIT_CLK Period | — | 81.4 | — | ns | Bit clock is input |
| CS65 | TSACL | Input Setup Time to Falling Edge of BIT_CLK | — | — | 10 | ns | — |
| CS66 | THACL | Input Hold Time from Falling Edge of BIT_CLK | — | — | 10 | ns | — |
| CS70 | TSYNCL | SYNC Data Output Low Time | — | 19.5 | — | μs | Note 1 |
| CS71 | TSYNCH | SYNC Data Output High Time | — | 1.3 | — | μs | Note 1 |
| CS72 | TSYNC | SYNC Data Output Period | — | 20.8 | — | μs | Note 1 |
| CS75 | TRACL | Rise Time, SYNC, SDATA_OUT | — | 10 | 25 | ns | CLOAD = 50 pF, VDD = 5V |
| CS76 | TFACL | Fall Time, SYNC, SDATA_OUT | — | 10 | 25 | ns | CLOAD = 50 pF, VDD = 5V |
| CS77 | TRACL | Rise Time, SYNC, SDATA_OUT | — | TBD | TBD | ns | CLOAD = 50 pF, VDD = 3V |
| CS78 | TFACL | Fall Time, SYNC, SDATA_OUT | — | TBD | TBD | ns | CLOAD = 50 pF, VDD = 3V |
| CS80 | TOVDA | Output valid delay from rising edge of BIT_CLK | — | — | 15 | ns | — |

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

2: These values assume BIT_CLK frequency is 12.288 MHz.

3: Data in "Typ" column is at 5V, 25°C unless otherwise stated. Parameters are for design guidance only and are not tested.

dsPIC30F

FIGURE 23-14: SPI MODULE MASTER MODE (CKE = 0) TIMING CHARACTERISTICS

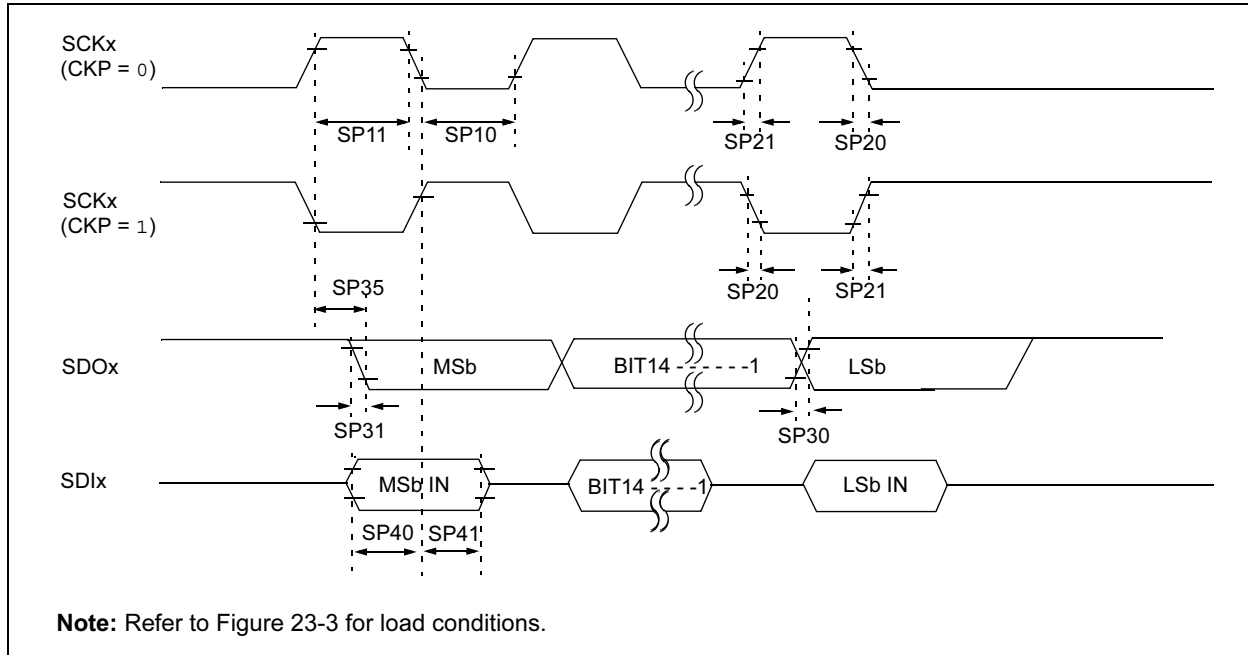


TABLE 23-27: SPI MASTER MODE (CKE = 0) TIMING REQUIREMENTS

| AC CHARACTERISTICS | | | Standard Operating Conditions: 2.5V to 5.5V (unless otherwise stated) Operating temperature $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$ for Industrial $-40^{\circ}\text{C} \leq T_A \leq +125^{\circ}\text{C}$ for Extended | | | | |
|--------------------|-----------------------|--|---|--------------------|-----|-------|------------|
| Param No. | Symbol | Characteristic ⁽¹⁾ | Min | Typ ⁽²⁾ | Max | Units | Conditions |
| SP10 | TscL | SCKx Output Low Time ⁽³⁾ | $T_{CY} / 2$ | — | — | ns | — |
| SP11 | TscH | SCKx Output High Time ⁽³⁾ | $T_{CY} / 2$ | — | — | ns | — |
| SP20 | TscF | SCKx Output Fall Time ⁽⁴⁾ | — | 10 | 25 | ns | — |
| SP21 | TscR | SCKx Output Rise Time ⁽⁴⁾ | — | 10 | 25 | ns | — |
| SP30 | TdoF | SDOx Data Output Fall Time ⁽⁴⁾ | — | 10 | 25 | ns | — |
| SP31 | TdoR | SDOx Data Output Rise Time ⁽⁴⁾ | — | 10 | 25 | ns | — |
| SP35 | TscH2doV, TscL2doV | SDOx Data Output Valid after SCKx Edge | — | — | 30 | ns | — |
| SP40 | TdiV2scH, TdiV2scL | Setup Time of SDIx Data Input to SCKx Edge | 20 | — | — | ns | — |
| SP41 | TscH2diL, TscL2diL | Hold Time of SDIx Data Input to SCKx Edge | 20 | — | — | ns | — |

- Note 1:** These parameters are characterized but not tested in manufacturing.
Note 2: Data in “Typ” column is at 5V, 25°C unless otherwise stated. Parameters are for design guidance only and are not tested.
Note 3: The minimum clock period for SCK is 100 ns. Therefore, the clock generated in Master mode must not violate this specification.
Note 4: Assumes 50 pF load on all SPI pins.

FIGURE 23-15: SPI MODULE MASTER MODE (CKE = 1) TIMING CHARACTERISTICS

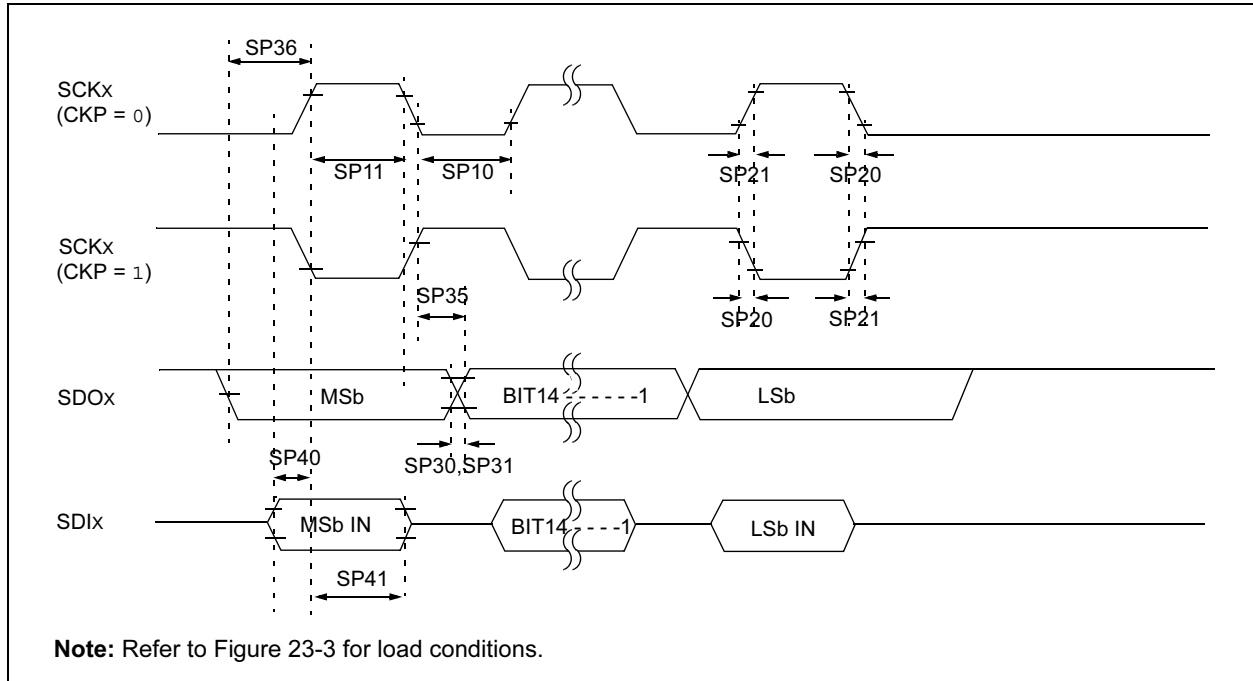


TABLE 23-28: SPI MODULE MASTER MODE (CKE = 1) TIMING REQUIREMENTS

| AC CHARACTERISTICS | | | Standard Operating Conditions: 2.5V to 5.5V (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 |
| SP10 | TscL | SCKx output low time ⁽³⁾ | Tcy / 2 | — | — | ns | — |
| SP11 | TscH | SCKx output high time ⁽³⁾ | Tcy / 2 | — | — | ns | — |
| SP20 | TscF | SCKx output fall time ⁽⁴⁾ | — | 10 | 25 | ns | — |
| SP21 | TscR | SCKx output rise time ⁽⁴⁾ | — | 10 | 25 | ns | — |
| SP30 | TdoF | SDOx data output fall time ⁽⁴⁾ | — | 10 | 25 | ns | — |
| SP31 | TdoR | SDOx data output rise time ⁽⁴⁾ | — | 10 | 25 | ns | — |
| SP35 | Tsch2doV, TscL2doV | SDOx data output valid after SCKx edge | — | — | 30 | ns | — |
| SP36 | TdoV2sc, TdoV2scL | SDOx data output setup to first SCKx edge | 30 | — | — | ns | — |
| SP40 | TdiV2scH, TdiV2scL | Setup time of SDIx data input to SCKx edge | 20 | — | — | ns | — |
| SP41 | Tsch2diL, TscL2diL | Hold time of SDIx data input to SCKx edge | 20 | — | — | ns | — |

- Note 1:** These parameters are characterized but not tested in manufacturing.
Note 2: Data in "Typ" column is at 5V, 25°C unless otherwise stated. Parameters are for design guidance only and are not tested.
Note 3: The minimum clock period for SCK is 100 ns. Therefore, the clock generated in master mode must not violate this specification.
Note 4: Assumes 50 pF load on all SPI pins.

dsPIC30F

FIGURE 23-16: SPI MODULE SLAVE MODE (CKE = 0) TIMING CHARACTERISTICS

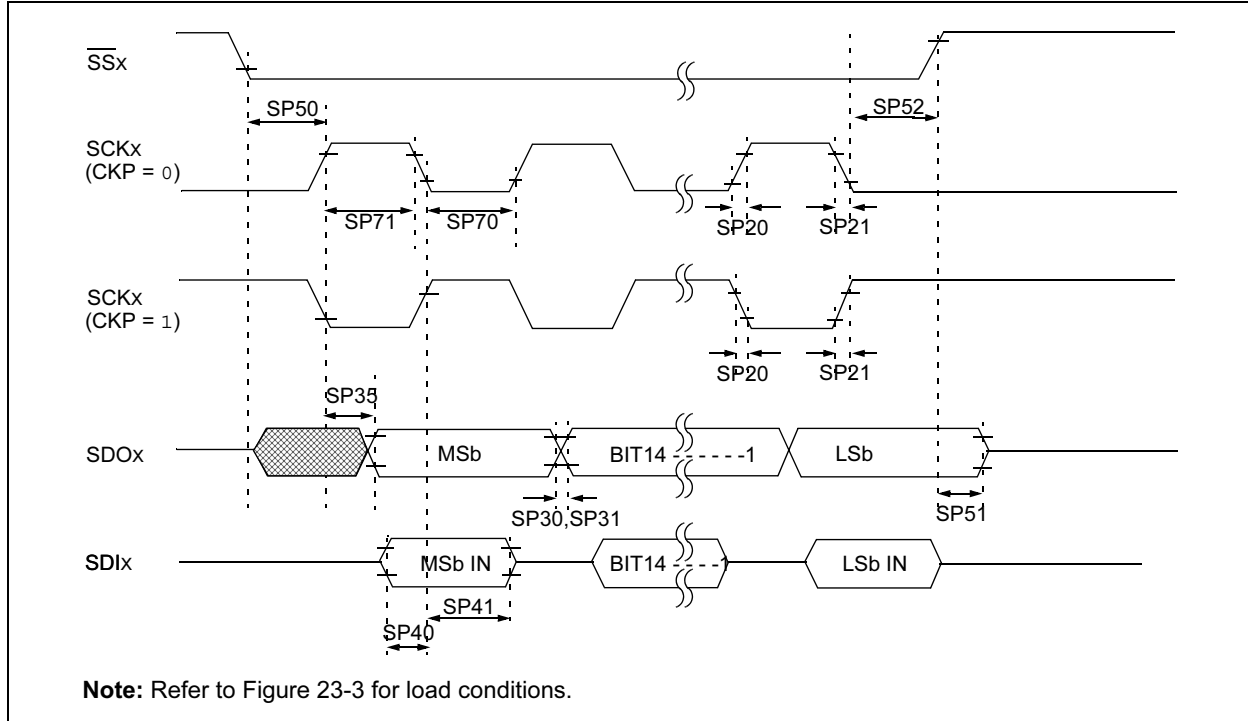


TABLE 23-29: SPI MODULE SLAVE MODE (CKE = 0) TIMING REQUIREMENTS

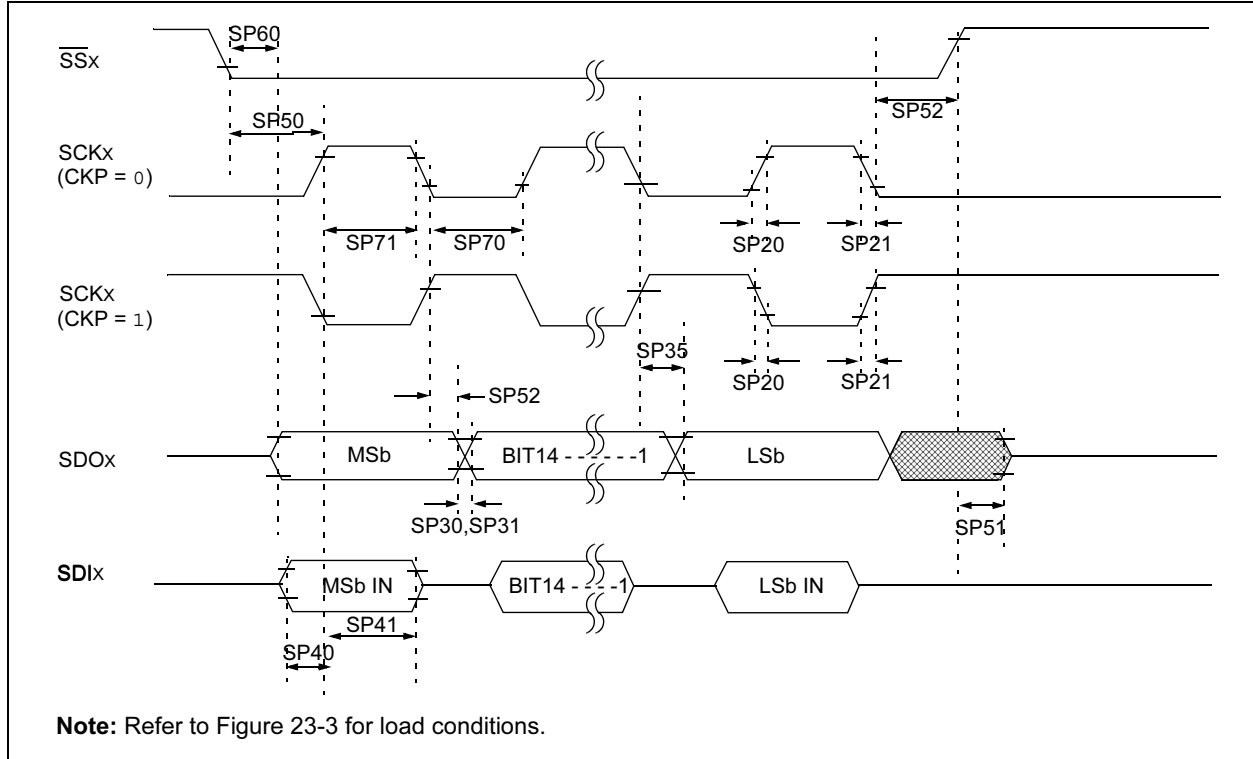
| AC CHARACTERISTICS | | | Standard Operating Conditions: 2.5V to 5.5V (unless otherwise stated) Operating temperature $-40^{\circ}\text{C} \leq T_a \leq +85^{\circ}\text{C}$ for Industrial $-40^{\circ}\text{C} \leq T_a \leq +125^{\circ}\text{C}$ for Extended | | | | |
|--------------------|-----------------------|---|---|--------------------|-----|-------|------------|
| Param No. | Symbol | Characteristic ⁽¹⁾ | Min | Typ ⁽²⁾ | Max | Units | Conditions |
| SP70 | TscL | SCKx Input Low Time | 30 | — | — | ns | — |
| SP71 | TscH | SCKx Input High Time | 30 | — | — | ns | — |
| SP20 | TscF | SCKx Output Fall Time ⁽³⁾ | — | 10 | 25 | ns | — |
| SP21 | TscR | SCKx Output Rise Time ⁽³⁾ | — | 10 | 25 | ns | — |
| SP30 | TdoF | SDOx Data Output Fall Time ⁽³⁾ | — | 10 | 25 | ns | — |
| SP31 | TdoR | SDOx Data Output Rise Time ⁽³⁾ | — | 10 | 25 | ns | — |
| SP35 | Tsch2doV, TscL2doV | SDOx Data Output Valid after SCKx Edge | — | — | 30 | ns | — |
| SP40 | TdiV2scH, TdiV2scL | Setup Time of SDIx Data Input to SCKx Edge | 20 | — | — | ns | — |
| SP41 | Tsch2diL, TscL2diL | Hold Time of SDIx Data Input to SCKx Edge | 20 | — | — | ns | — |
| SP50 | TssL2scH, TssL2scL | SSx↓ to SCKx↑ or SCKx↓ Input | 120 | — | — | ns | — |
| SP51 | TssH2doZ | SSx↑ to SDOx Output Hi-Impedance ⁽³⁾ | 10 | — | 50 | ns | — |
| SP52 | Tsch2ssH, TscL2ssH | SSx after SCK Edge | 1.5 TcY + 40 | — | — | ns | — |

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

Note 2: Data in “Typ” column is at 5V, 25°C unless otherwise stated. Parameters are for design guidance only and are not tested.

Note 3: Assumes 50 pF load on all SPI pins.

FIGURE 23-17: SPI MODULE SLAVE MODE (CKE = 1) TIMING CHARACTERISTICS



dsPIC30F

TABLE 23-30: SPI MODULE SLAVE MODE (CKE = 1) TIMING REQUIREMENTS

| AC CHARACTERISTICS | | | Standard Operating Conditions: 2.5V to 5.5V (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 |
| SP70 | TscL | SCKx Input Low Time | 30 | — | — | ns | — |
| SP71 | TscH | SCKx Input High Time | 30 | — | — | ns | — |
| SP20 | TscF | SCKx Output Fall Time ⁽³⁾ | — | 10 | 25 | ns | — |
| SP21 | TscR | SCKx Output Rise Time ⁽³⁾ | — | 10 | 25 | ns | — |
| SP30 | TdoF | SDOx Data Output Fall Time ⁽³⁾ | — | 10 | 25 | ns | — |
| SP31 | TdoR | SDOx Data Output Rise Time ⁽³⁾ | — | 10 | 25 | ns | — |
| SP35 | Tsch2doV, TscL2doV | SDOx Data Output Valid after SCKx Edge | — | — | 30 | ns | — |
| SP40 | TdiV2scH, TdiV2scL | Setup Time of SDIx Data Input to SCKx Edge | 20 | — | — | ns | — |
| SP41 | Tsch2diL, TscL2diL | Hold Time of SDIx Data Input to SCKx Edge | 20 | — | — | ns | — |
| SP50 | TssL2scH, TssL2scL | \overline{SSx} ↓ to SCKx↓ or SCKx↑ input | 120 | — | — | ns | — |
| SP51 | TssH2doZ | \overline{SS} ↑ to SDOx Output Hi-Impedance ⁽⁴⁾ | 10 | — | 50 | ns | — |
| SP52 | Tsch2ssH TscL2ssH | \overline{SSx} ↑ after SCKx Edge | 1.5 TcY + 40 | — | — | ns | — |
| SP60 | TssL2doV | SDOx Data Output Valid after \overline{SSx} Edge | — | — | 50 | ns | — |

- Note 1:** These parameters are characterized but not tested in manufacturing.
Note 2: Data in “Typ” column is at 5V, 25°C unless otherwise stated. Parameters are for design guidance only and are not tested.
Note 3: The minimum clock period for SCK is 100 ns. Therefore, the clock generated in master mode must not violate this specification.
Note 4: Assumes 50 pF load on all SPI pins.

FIGURE 23-18: I²C BUS START/STOP BITS TIMING CHARACTERISTICS (MASTER MODE)

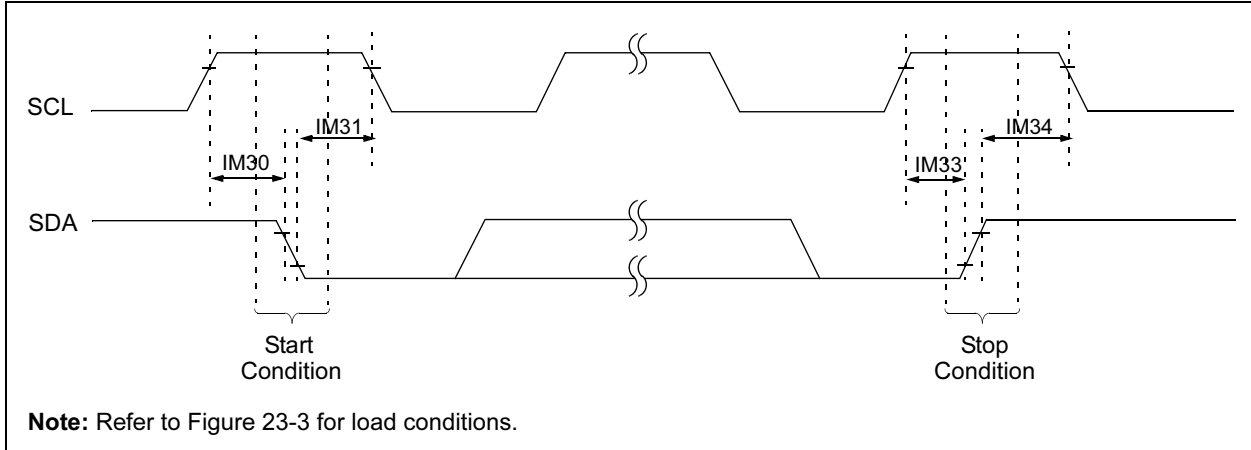
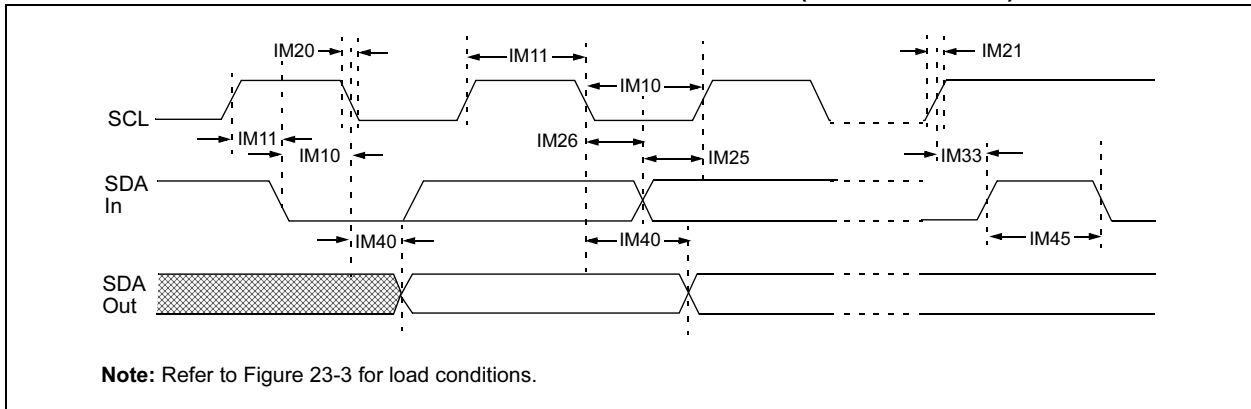


FIGURE 23-19: I²C BUS DATA TIMING CHARACTERISTICS (MASTER MODE)



dsPIC30F

TABLE 23-31: I²C BUS DATA TIMING REQUIREMENTS (MASTER MODE)

| AC CHARACTERISTICS | | | Standard Operating Conditions: 2.5V to 5.5V (unless otherwise stated) Operating temperature -40°C ≤ T _A ≤ +85°C for Industrial -40°C ≤ T _A ≤ +125°C for Extended | | | | |
|--------------------|---------|----------------------------|---|-------------------------------|-------|------------|---|
| Param No. | Symbol | Characteristic | Min ⁽¹⁾ | Max | Units | Conditions | |
| IM10 | TLO:SCL | Clock Low Time | 100 kHz mode | T _{CY} / 2 (BRG + 1) | — | ms | — |
| | | | 400 kHz mode | T _{CY} / 2 (BRG + 1) | — | ms | — |
| | | | 1 MHz mode ⁽²⁾ | T _{CY} / 2 (BRG + 1) | — | ms | — |
| IM11 | THI:SCL | Clock High Time | 100 kHz mode | T _{CY} / 2 (BRG + 1) | — | ms | — |
| | | | 400 kHz mode | T _{CY} / 2 (BRG + 1) | — | ms | — |
| | | | 1 MHz mode ⁽²⁾ | T _{CY} / 2 (BRG + 1) | — | ms | — |
| IM20 | TF:SCL | SDA and SCL Fall Time | 100 kHz mode | — | 300 | ns | CB is specified to be from 10 to 400 pF |
| | | | 400 kHz mode | 20 + 0.1 C _B | 300 | ns | |
| | | | 1 MHz mode ⁽²⁾ | — | 100 | ns | |
| IM21 | TR:SCL | SDA and SCL Rise Time | 100 kHz mode | — | 1000 | ns | CB is specified to be from 10 to 400 pF |
| | | | 400 kHz mode | 20 + 0.1 C _B | 300 | ns | |
| | | | 1 MHz mode ⁽²⁾ | — | 300 | ns | |
| IM25 | TSU:DAT | Data Input Setup Time | 100 kHz mode | 250 | — | ns | — |
| | | | 400 kHz mode | 100 | — | ns | |
| | | | 1 MHz mode ⁽²⁾ | TBD | — | ns | |
| IM26 | THD:DAT | Data Input Hold Time | 100 kHz mode | 0 | — | ns | — |
| | | | 400 kHz mode | 0 | 0.9 | ms | |
| | | | 1 MHz mode ⁽²⁾ | TBD | — | ns | |
| IM30 | TSU:STA | Start Condition Setup Time | 100 kHz mode | T _{CY} / 2 (BRG + 1) | — | ms | Only relevant for repeated Start condition |
| | | | 400 kHz mode | T _{CY} / 2 (BRG + 1) | — | ms | |
| | | | 1 MHz mode ⁽²⁾ | T _{CY} / 2 (BRG + 1) | — | ms | |
| IM31 | THD:STA | Start Condition Hold Time | 100 kHz mode | T _{CY} / 2 (BRG + 1) | — | ms | After this period the first clock pulse is generated |
| | | | 400 kHz mode | T _{CY} / 2 (BRG + 1) | — | ms | |
| | | | 1 MHz mode ⁽²⁾ | T _{CY} / 2 (BRG + 1) | — | ms | |
| IM33 | TSU:STO | Stop Condition Setup Time | 100 kHz mode | T _{CY} / 2 (BRG + 1) | — | ms | — |
| | | | 400 kHz mode | T _{CY} / 2 (BRG + 1) | — | ms | |
| | | | 1 MHz mode ⁽²⁾ | T _{CY} / 2 (BRG + 1) | — | ms | |
| IM34 | THD:STO | Stop Condition Hold Time | 100 kHz mode | T _{CY} / 2 (BRG + 1) | — | ns | — |
| | | | 400 kHz mode | T _{CY} / 2 (BRG + 1) | — | ns | |
| | | | 1 MHz mode ⁽²⁾ | T _{CY} / 2 (BRG + 1) | — | ns | |
| IM40 | TAA:SCL | Output Valid From Clock | 100 kHz mode | — | 3500 | ns | — |
| | | | 400 kHz mode | — | 1000 | ns | |
| | | | 1 MHz mode ⁽²⁾ | — | — | ns | |
| IM45 | TBF:SDA | Bus Free Time | 100 kHz mode | 4.7 | — | ms | Time the bus must be free before a new transmission can start |
| | | | 400 kHz mode | 1.3 | — | ms | |
| | | | 1 MHz mode ⁽²⁾ | TBD | — | ms | |
| IM50 | CB | Bus Capacitive Loading | — | 400 | pF | — | |

Note 1: BRG is the value of the I²C Baud Rate Generator. Refer to **Section 21 “Inter-Integrated Circuit™ (I²C)”** in the *dsPIC30F Family Reference Manual*.

2: Maximum pin capacitance = 10 pF for all I²C pins (for 1 MHz mode only).

FIGURE 23-20: I²C BUS START/STOP BITS TIMING CHARACTERISTICS (SLAVE MODE)

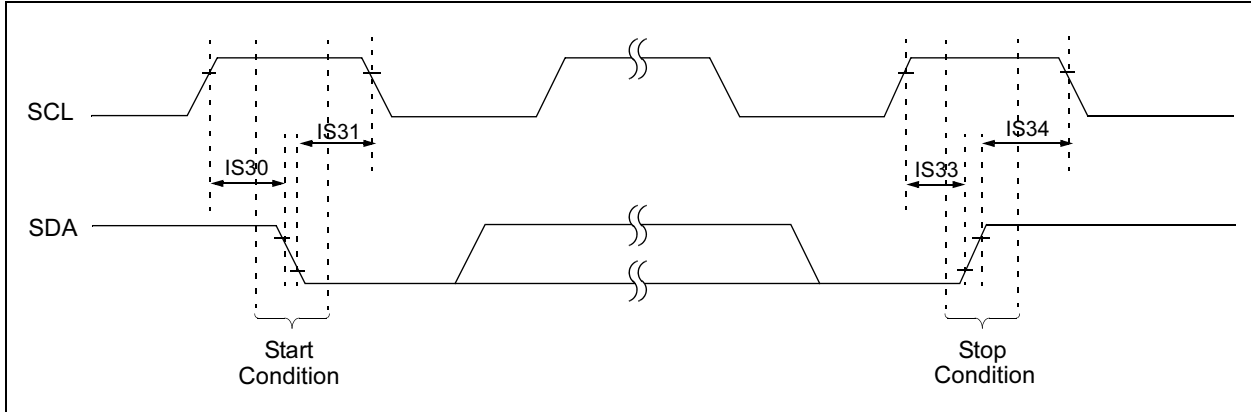
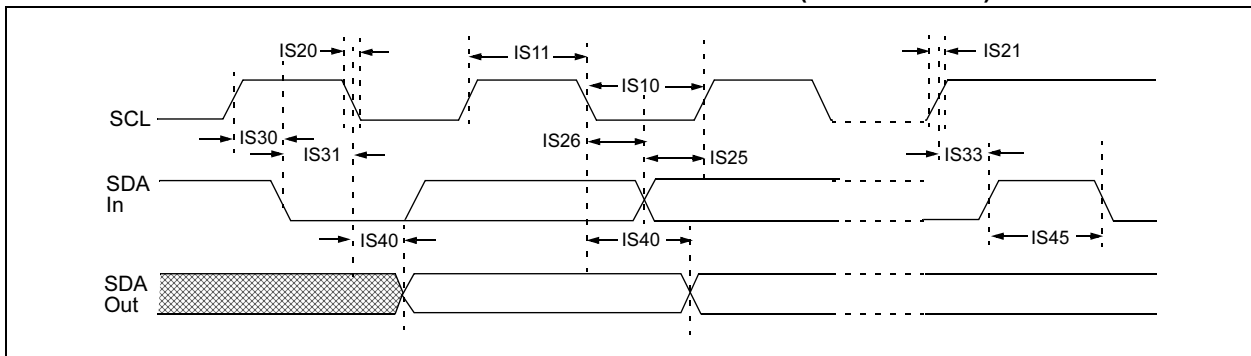


FIGURE 23-21: I²C BUS DATA TIMING CHARACTERISTICS (SLAVE MODE)



dsPIC30F

TABLE 23-32: I²C BUS DATA TIMING REQUIREMENTS (SLAVE MODE)

| AC CHARACTERISTICS | | Standard Operating Conditions: 2.5V to 5.5V (unless otherwise stated) Operating temperature -40°C ≤ TA ≤ +85°C for Industrial -40°C ≤ TA ≤ +125°C for Extended | | | | | |
|--------------------|---------|---|---------------------------|-------------|-------|------------|---|
| Param No. | Symbol | Characteristic | Min | Max | Units | Conditions | |
| IS10 | TLO:SCL | Clock Low Time | 100 kHz mode | 4.7 | — | μs | Device must operate at a minimum of 1.5 MHz |
| | | | 400 kHz mode | 1.3 | — | μs | Device must operate at a minimum of 10 MHz. |
| | | | 1 MHz mode ⁽¹⁾ | 0.5 | — | μs | — |
| IS11 | THI:SCL | Clock High Time | 100 kHz mode | 4.0 | — | μs | Device must operate at a minimum of 1.5 MHz |
| | | | 400 kHz mode | 0.6 | — | μs | Device must operate at a minimum of 10 MHz |
| | | | 1 MHz mode ⁽¹⁾ | 0.5 | — | μs | — |
| IS20 | TF:SCL | SDA and SCL Fall Time | 100 kHz mode | — | 300 | ns | CB is specified to be from 10 to 400 pF |
| | | | 400 kHz mode | 20 + 0.1 CB | 300 | ns | |
| | | | 1 MHz mode ⁽¹⁾ | — | 100 | ns | |
| IS21 | TR:SCL | SDA and SCL Rise Time | 100 kHz mode | — | 1000 | ns | CB is specified to be from 10 to 400 pF |
| | | | 400 kHz mode | 20 + 0.1 CB | 300 | ns | |
| | | | 1 MHz mode ⁽¹⁾ | — | 300 | ns | |
| IS25 | TSU:DAT | Data Input Setup Time | 100 kHz mode | 250 | — | ns | — |
| | | | 400 kHz mode | 100 | — | ns | |
| | | | 1 MHz mode ⁽¹⁾ | 100 | — | ns | |
| IS26 | THD:DAT | Data Input Hold Time | 100 kHz mode | 0 | — | ns | — |
| | | | 400 kHz mode | 0 | 0.9 | μs | |
| | | | 1 MHz mode ⁽¹⁾ | 0 | 0.3 | μs | |
| IS30 | TSU:STA | Start Condition Setup Time | 100 kHz mode | 4.7 | — | μs | Only relevant for repeated Start condition |
| | | | 400 kHz mode | 0.6 | — | μs | |
| | | | 1 MHz mode ⁽¹⁾ | 0.25 | — | μs | |
| IS31 | THD:STA | Start Condition Hold Time | 100 kHz mode | 4.0 | — | μs | After this period the first clock pulse is generated |
| | | | 400 kHz mode | 0.6 | — | μs | |
| | | | 1 MHz mode ⁽¹⁾ | 0.25 | — | μs | |
| IS33 | TSU:STO | Stop Condition Setup Time | 100 kHz mode | 4.7 | — | μs | — |
| | | | 400 kHz mode | 0.6 | — | μs | |
| | | | 1 MHz mode ⁽¹⁾ | 0.6 | — | μs | |
| IS34 | THD:STO | Stop Condition Hold Time | 100 kHz mode | 4000 | — | ns | — |
| | | | 400 kHz mode | 600 | — | ns | |
| | | | 1 MHz mode ⁽¹⁾ | 250 | — | ns | |
| IS40 | TAA:SCL | Output Valid From Clock | 100 kHz mode | 0 | 3500 | ns | — |
| | | | 400 kHz mode | 0 | 1000 | ns | |
| | | | 1 MHz mode ⁽¹⁾ | 0 | 350 | ns | |
| IS45 | TBF:SDA | Bus Free Time | 100 kHz mode | 4.7 | — | μs | Time the bus must be free before a new transmission can start |
| | | | 400 kHz mode | 1.3 | — | μs | |
| | | | 1 MHz mode ⁽¹⁾ | 0.5 | — | μs | |
| IS50 | CB | Bus Capacitive Loading | — | 400 | pF | — | |

Note 1: Maximum pin capacitance = 10 pF for all I²C pins (for 1 MHz mode only).

FIGURE 23-22: CAN MODULE I/O TIMING CHARACTERISTICS

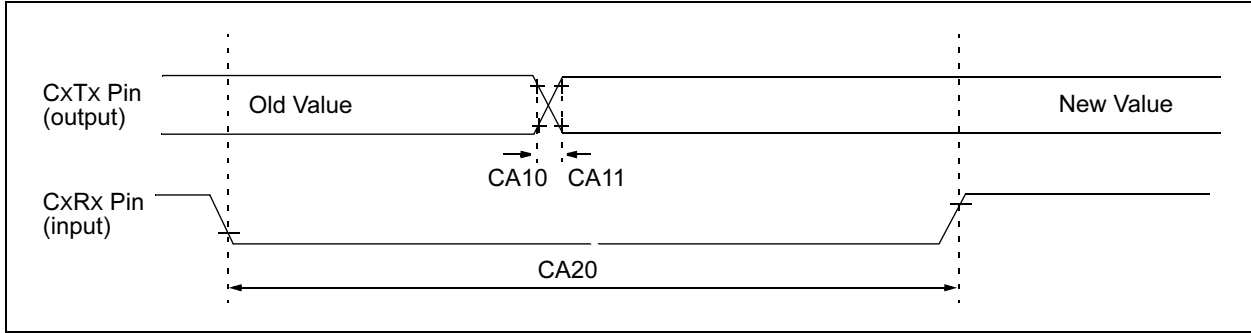


TABLE 23-33: CAN MODULE I/O TIMING REQUIREMENTS

| AC CHARACTERISTICS | | Standard Operating Conditions: 2.5V to 5.5V (unless otherwise stated) Operating temperature $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$ for Industrial $-40^{\circ}\text{C} \leq T_A \leq +125^{\circ}\text{C}$ for Extended | | | | | |
|--------------------|--------|---|-----|--------------------|-----|-------|------------|
| Param No. | Symbol | Characteristic ⁽¹⁾ | Min | Typ ⁽²⁾ | Max | Units | Conditions |
| CA10 | TioF | Port Output Fall Time | — | 10 | 25 | ns | — |
| CA11 | TioR | Port Output Rise Time | — | 10 | 25 | ns | — |
| CA20 | Tcwf | Pulse Width to Trigger CAN Wakeup Filter | 500 | | | ns | — |

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

Note 2: Data in "Typ" column is at 5V, 25°C unless otherwise stated. Parameters are for design guidance only and are not tested.

dsPIC30F

TABLE 23-34: 12-BIT A/D MODULE SPECIFICATIONS

| AC CHARACTERISTICS | | | Standard Operating Conditions: 2.7V to 5.5V (unless otherwise stated) Operating temperature $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$ for Industrial $-40^{\circ}\text{C} \leq T_A \leq +125^{\circ}\text{C}$ for Extended | | | | |
|-------------------------|-----------|--|---|-------------|----------------------------|--------------------------------|--|
| Param No. | Symbol | Characteristic | Min. | Typ | Max. | Units | Conditions |
| Device Supply | | | | | | | |
| AD01 | AVDD | Module VDD Supply | Greater of VDD - 0.3 or 2.7 | — | Lesser of VDD + 0.3 or 5.5 | V | — |
| AD02 | AVSS | Module VSS Supply | VSS - 0.3 | — | VSS + 0.3 | V | — |
| Reference Inputs | | | | | | | |
| AD05 | VREFH | Reference Voltage High | AVSS + 2.7 | — | AVDD | V | — |
| AD06 | VREFL | Reference Voltage Low | AVSS | — | AVDD - 2.7 | V | — |
| AD07 | VREF | Absolute Reference Voltage | AVSS - 0.3 | — | AVDD + 0.3 | V | — |
| AD08 | IREF | Current Drain | — | 200 .001 | 300 3 | μA μA | A/D operating A/D off |
| Analog Input | | | | | | | |
| AD10 | VINH-VINL | Full-Scale Input Span | VREFL | | VREFH | V | See Note |
| AD11 | VIN | Absolute Input Voltage | AVSS - 0.3 | | AVDD + 0.3 | V | — |
| AD12 | — | Leakage Current | — | ± 0.001 | ± 0.610 | μA | VINL = AVSS = VREFL = 0V, AVDD = VREFH = 5V Source Impedance = 2.5 k Ω |
| AD13 | — | Leakage Current | — | ± 0.001 | ± 0.610 | μA | VINL = AVSS = VREFL = 0V, AVDD = VREFH = 3V Source Impedance = 2.5 k Ω |
| AD15 | RSS | Switch Resistance | — | 3.2K | — | Ω | — |
| AD16 | CSAMPLE | Sample Capacitor | — | 18 | | pF | — |
| AD17 | RIN | Recommended Impedance of Analog Voltage Source | — | — | 2.5K | Ω | — |
| DC Accuracy | | | | | | | |
| AD20 | Nr | Resolution | 12 data bits | | | bits | |
| AD21 | INL | Integral Nonlinearity | — | ± 0.75 | TBD | LSb | VINL = AVSS = VREFL = 0V, AVDD = VREFH = 5V |
| AD21A | INL | Integral Nonlinearity | — | ± 0.75 | TBD | LSb | VINL = AVSS = VREFL = 0V, AVDD = VREFH = 3V |
| AD22 | DNL | Differential Nonlinearity | — | ± 0.5 | $< \pm 1$ | LSb | VINL = AVSS = VREFL = 0V, AVDD = VREFH = 5V |
| AD22A | DNL | Differential Nonlinearity | — | ± 0.5 | $< \pm 1$ | LSb | VINL = AVSS = VREFL = 0V, AVDD = VREFH = 3V |
| AD23 | GERR | Gain Error | — | ± 1.25 | TBD | LSb | VINL = AVSS = VREFL = 0V, AVDD = VREFH = 5V |
| AD23A | GERR | Gain Error | — | ± 1.25 | TBD | LSb | VINL = AVSS = VREFL = 0V, AVDD = VREFH = 3V |

Note 1: The A/D conversion result never decreases with an increase in the input voltage, and has no missing codes.

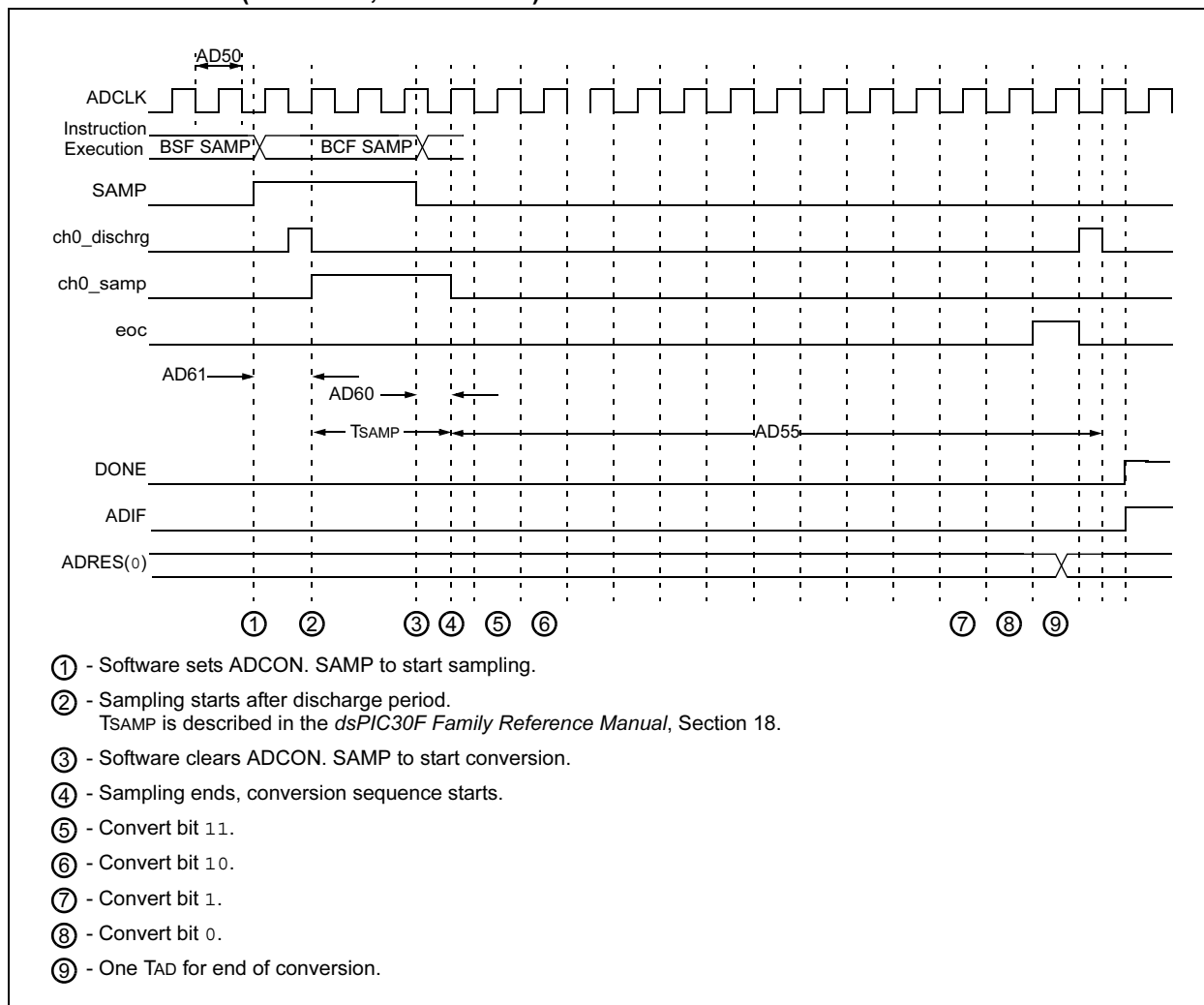
TABLE 23-34: 12-BIT A/D MODULE SPECIFICATIONS (CONTINUED)

| AC CHARACTERISTICS | | | Standard Operating Conditions: 2.7V to 5.5V (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 |
| AD24 | E _{OFF} | Offset Error | — | ±1.25 | TBD | LSb | V _{INL} = AV _{SS} = V _{REFL} = 0V, AV _{DD} = V _{REFH} = 5V |
| AD24A | E _{OFF} | Offset Error | — | ±1.25 | TBD | LSb | V _{INL} = AV _{SS} = V _{REFL} = 0V, AV _{DD} = V _{REFH} = 3V |
| AD25 | — | Monotonicity ⁽¹⁾ | — | — | — | — | Guaranteed |
| AD26 | CMRR | Common-Mode Rejection | — | TBD | — | dB | — |
| AD27 | PSRR | Power Supply Rejection Ratio | — | TBD | — | dB | — |
| AD28 | CTLK | Channel to Channel Crosstalk | — | TBD | — | dB | — |
| Dynamic Performance | | | | | | | |
| AD30 | THD | Total Harmonic Distortion | — | — | — | dB | — |
| AD31 | SINAD | Signal to Noise and Distortion | — | TBD | — | dB | — |
| AD32 | SFDR | Spurious Free Dynamic Range | — | TBD | — | dB | — |
| AD33 | F _{NYQ} | Input Signal Bandwidth | — | — | 50 | kHz | — |
| AD34 | ENOB | Effective Number of Bits | — | TBD | TBD | bits | — |

Note 1: The A/D conversion result never decreases with an increase in the input voltage, and has no missing codes.

dsPIC30F

FIGURE 23-23: 12-BIT A/D CONVERSION TIMING CHARACTERISTICS (ASAM = 0, SSRC = 000)



- ① - Software sets ADCON. SAMP to start sampling.
- ② - Sampling starts after discharge period.
TsAMP is described in the *dsPIC30F Family Reference Manual*, Section 18.
- ③ - Software clears ADCON. SAMP to start conversion.
- ④ - Sampling ends, conversion sequence starts.
- ⑤ - Convert bit 11.
- ⑥ - Convert bit 10.
- ⑦ - Convert bit 1.
- ⑧ - Convert bit 0.
- ⑨ - One TAD for end of conversion.

TABLE 23-35: 12-BIT A/D CONVERSION TIMING REQUIREMENTS

| AC CHARACTERISTICS | | | Standard Operating Conditions: 2.7V to 5.5V (unless otherwise stated) Operating temperature $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$ for Industrial $-40^{\circ}\text{C} \leq T_A \leq +125^{\circ}\text{C}$ for Extended | | | | |
|--------------------------|--------|---|---|--------|---------|---------------|---|
| Param No. | Symbol | Characteristic | Min. | Typ | Max. | Units | Conditions |
| Clock Parameters | | | | | | | |
| AD50 | TAD | A/D Clock Period | — | 667 | — | ns | VDD = 3-5.5V (Note 1) |
| AD51 | tRC | A/D Internal RC Oscillator Period | 1.2 | 1.5 | 1.8 | μs | — |
| Conversion Rate | | | | | | | |
| AD55 | tCONV | Conversion Time | — | 14 TAD | — | ns | — |
| AD56 | FCNV | Throughput Rate | — | — | 100 | ksps | VDD = VREF = 5V |
| AD57 | TSAMP | Sample Time | — | 1 TAD | — | ns | VDD = 3-5.5V source resistance Rs = 0-2.5 k Ω |
| Timing Parameters | | | | | | | |
| AD60 | tPCS | Conversion Start from Sample Trigger | — | — | TAD | ns | — |
| AD61 | tPSS | Sample Start from Setting Sample (SAMP) Bit | 0.5 TAD | — | 1.5 TAD | ns | — |
| AD62 | tCSS | Conversion Completion to Sample Start (ASAM = 1) | — | — | TBD | ns | — |
| AD63 | tDPU | Time to Stabilize Analog Stage from A/D Off to A/D On | — | — | TBD | μs | — |

Note 1: Because the sample caps will eventually lose charge, clock rates below 10 kHz can affect linearity performance, especially at elevated temperatures.

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

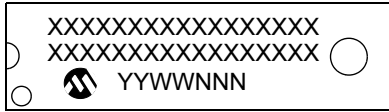
dsPIC30F

NOTES:

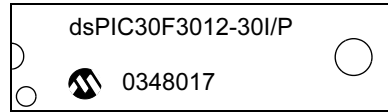
24.0 PACKAGING INFORMATION

24.1 Package Marking Information

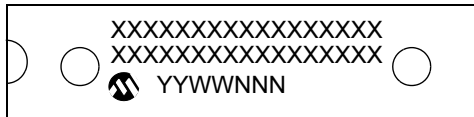
18-Lead PDIP



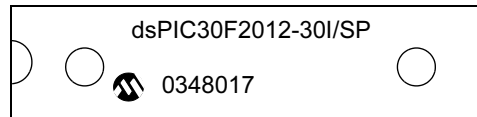
Example



28-Lead PDIP



Example



18-Lead SOIC



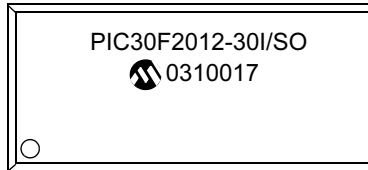
Example



28-Lead SOIC (.300")



Example



Legend: XX...X Customer specific information*
 Y Year code (last digit of calendar year)
 YY Year code (last 2 digits of calendar year)
 WW Week code (week of January 1 is week '01')
 NNN Alphanumeric traceability code

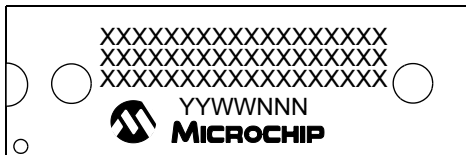
Note: In the event the full Microchip part number cannot be marked on one line, it will be carried over to the next line thus limiting the number of available characters for customer specific information.

* Standard device marking consists of Microchip part number, year code, week code, and traceability code. For device marking beyond this, certain price adders apply. Please check with your Microchip Sales Office. For QTP devices, any special marking adders are included in QTP price.

dsPIC30F

24.1 Package Marking Information (Continued)

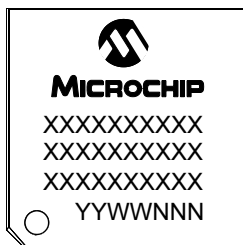
40-Lead PDIP



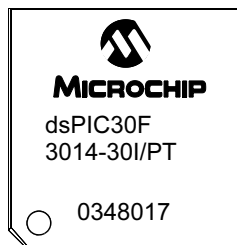
Example



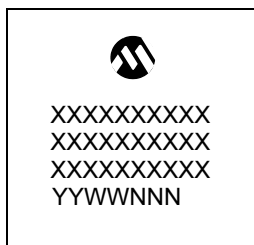
44-Lead TQFP



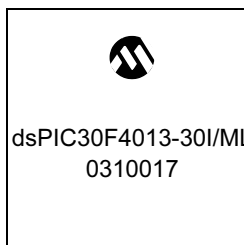
Example



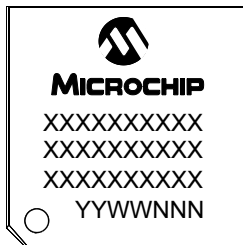
44-Lead QFN



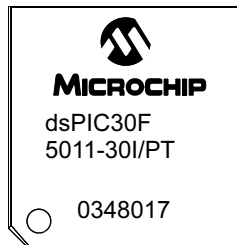
Example



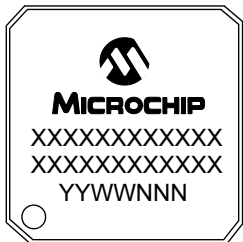
64-Lead TQFP (10x10x1mm)



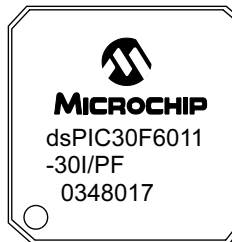
Example



64-Lead TQFP (14x14x1mm)

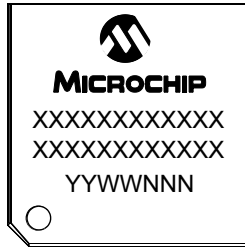


Example

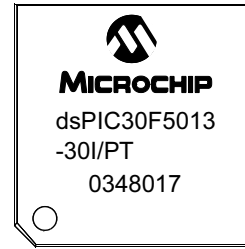


24.1 Package Marking Information (Continued)

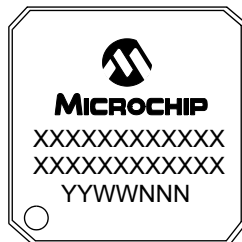
80-Lead TQFP (12x12x1mm)



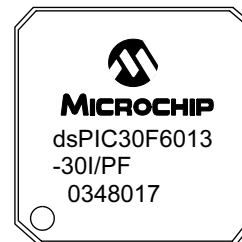
Example



80-Lead TQFP (14x14x1mm)

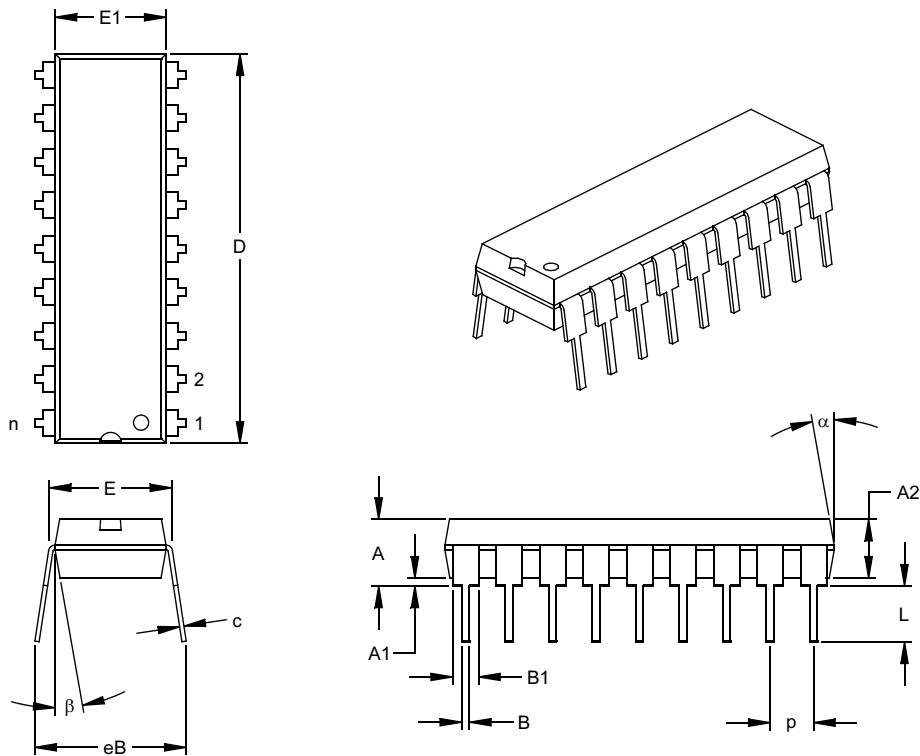


Example



dsPIC30F

18-Lead Plastic Dual In-line (P) – 300 mil (PDIP)



| Units | | INCHES* | | | MILLIMETERS | | |
|----------------------------|------|---------|------|------|-------------|-------|-------|
| Dimension Limits | | MIN | NOM | MAX | MIN | NOM | MAX |
| Number of Pins | n | | 18 | | | 18 | |
| Pitch | p | | .100 | | | 2.54 | |
| Top to Seating Plane | A | .140 | .155 | .170 | 3.56 | 3.94 | 4.32 |
| Molded Package Thickness | A2 | .115 | .130 | .145 | 2.92 | 3.30 | 3.68 |
| Base to Seating Plane | A1 | .015 | | | 0.38 | | |
| Shoulder to Shoulder Width | E | .300 | .313 | .325 | 7.62 | 7.94 | 8.26 |
| Molded Package Width | E1 | .240 | .250 | .260 | 6.10 | 6.35 | 6.60 |
| Overall Length | D | .890 | .898 | .905 | 22.61 | 22.80 | 22.99 |
| Tip to Seating Plane | L | .125 | .130 | .135 | 3.18 | 3.30 | 3.43 |
| Lead Thickness | c | .008 | .012 | .015 | 0.20 | 0.29 | 0.38 |
| Upper Lead Width | B1 | .045 | .058 | .070 | 1.14 | 1.46 | 1.78 |
| Lower Lead Width | B | .014 | .018 | .022 | 0.36 | 0.46 | 0.56 |
| Overall Row Spacing | § eB | .310 | .370 | .430 | 7.87 | 9.40 | 10.92 |
| Mold Draft Angle Top | α | 5 | 10 | 15 | 5 | 10 | 15 |
| Mold Draft Angle Bottom | β | 5 | 10 | 15 | 5 | 10 | 15 |

* Controlling Parameter

§ Significant Characteristic

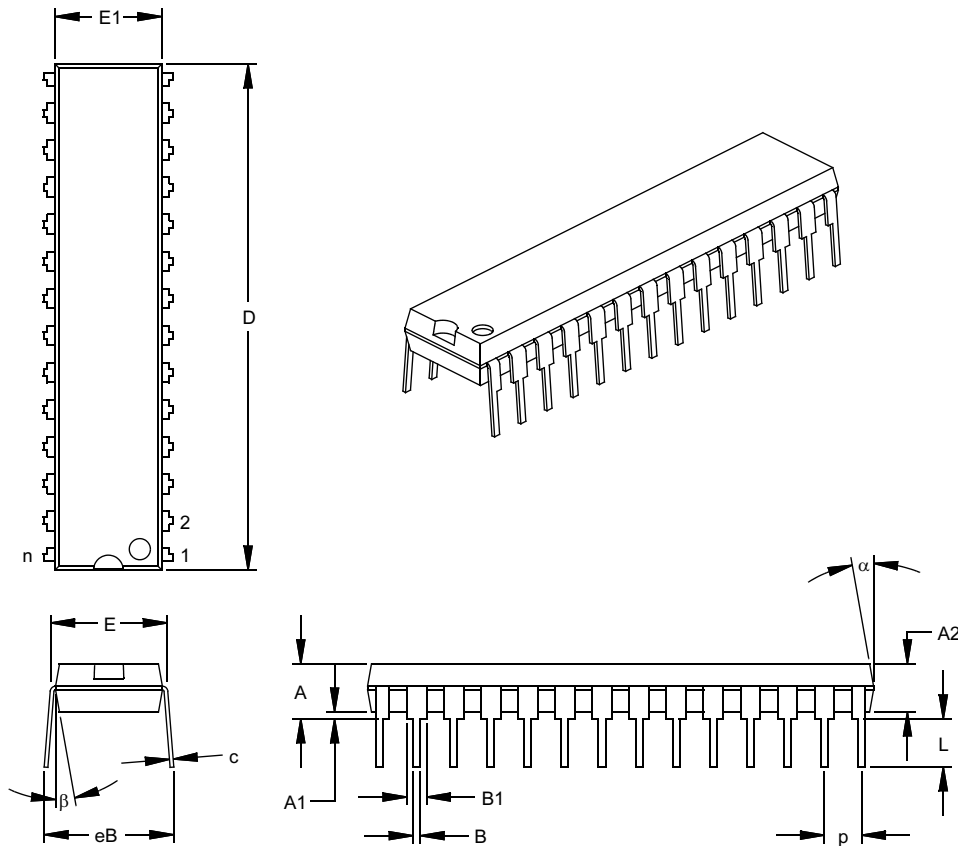
Notes:

Dimensions D and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed .010" (0.254mm) per side.

JEDEC Equivalent: MS-001

Drawing No. C04-007

28-Lead Plastic Dual In-line (SP) – 300 mil (PDIP)



| Units | | INCHES* | | | MILLIMETERS | | |
|----------------------------|------|---------|-------|-------|-------------|-------|-------|
| Dimension Limits | | MIN | NOM | MAX | MIN | NOM | MAX |
| Number of Pins | n | | 28 | | | 28 | |
| Pitch | p | | .100 | | | 2.54 | |
| Top to Seating Plane | A | .140 | .150 | .160 | 3.56 | 3.81 | 4.06 |
| Molded Package Thickness | A2 | .125 | .130 | .135 | 3.18 | 3.30 | 3.43 |
| Base to Seating Plane | A1 | .015 | | | 0.38 | | |
| Shoulder to Shoulder Width | E | .300 | .310 | .325 | 7.62 | 7.87 | 8.26 |
| Molded Package Width | E1 | .275 | .285 | .295 | 6.99 | 7.24 | 7.49 |
| Overall Length | D | 1.345 | 1.365 | 1.385 | 34.16 | 34.67 | 35.18 |
| Tip to Seating Plane | L | .125 | .130 | .135 | 3.18 | 3.30 | 3.43 |
| Lead Thickness | c | .008 | .012 | .015 | 0.20 | 0.29 | 0.38 |
| Upper Lead Width | B1 | .040 | .053 | .065 | 1.02 | 1.33 | 1.65 |
| Lower Lead Width | B | .016 | .019 | .022 | 0.41 | 0.48 | 0.56 |
| Overall Row Spacing | § eB | .320 | .350 | .430 | 8.13 | 8.89 | 10.92 |
| Mold Draft Angle Top | α | 5 | 10 | 15 | 5 | 10 | 15 |
| Mold Draft Angle Bottom | β | 5 | 10 | 15 | 5 | 10 | 15 |

* Controlling Parameter

§ Significant Characteristic

Notes:

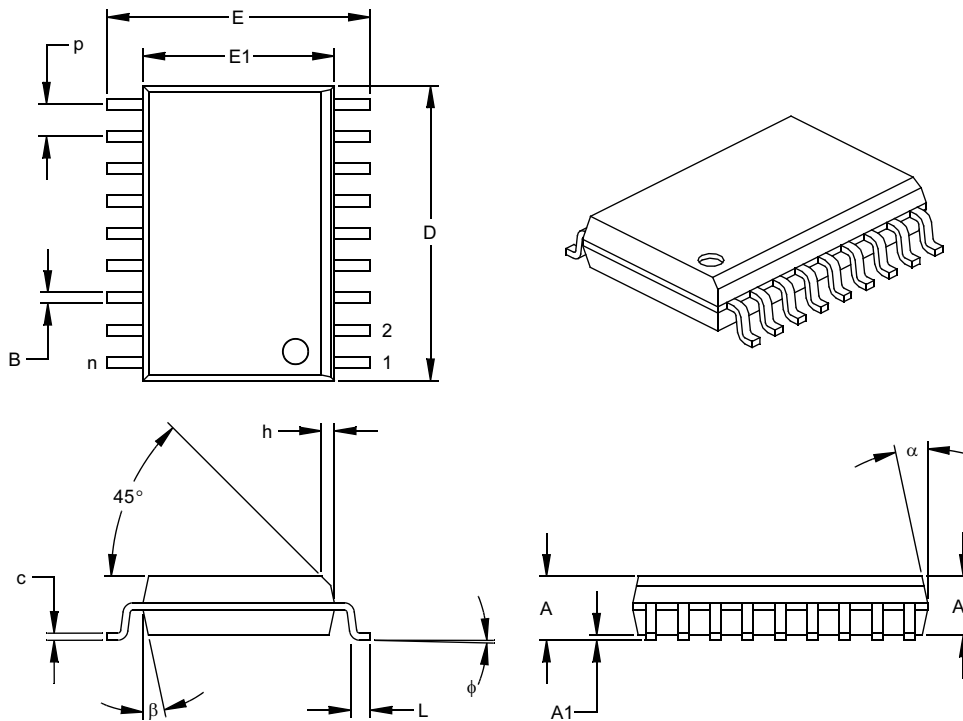
Dimension D and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed .010" (0.254mm) per side.

JEDEC Equivalent: MO-095

Drawing No. C04-070

dsPIC30F

18-Lead Plastic Small Outline (SO) – Wide, 300 mil (SOIC)



| Units | | INCHES* | | | MILLIMETERS | | |
|--------------------------|----|---------|------|------|-------------|-------|-------|
| Dimension Limits | | MIN | NOM | MAX | MIN | NOM | MAX |
| Number of Pins | n | | 18 | | | 18 | |
| Pitch | p | | .050 | | | 1.27 | |
| Overall Height | A | .093 | .099 | .104 | 2.36 | 2.50 | 2.64 |
| Molded Package Thickness | A2 | .088 | .091 | .094 | 2.24 | 2.31 | 2.39 |
| Standoff § | A1 | .004 | .008 | .012 | 0.10 | 0.20 | 0.30 |
| Overall Width | E | .394 | .407 | .420 | 10.01 | 10.34 | 10.67 |
| Molded Package Width | E1 | .291 | .295 | .299 | 7.39 | 7.49 | 7.59 |
| Overall Length | D | .446 | .454 | .462 | 11.33 | 11.53 | 11.73 |
| Chamfer Distance | h | .010 | .020 | .029 | 0.25 | 0.50 | 0.74 |
| Foot Length | L | .016 | .033 | .050 | 0.41 | 0.84 | 1.27 |
| Foot Angle | φ | 0 | 4 | 8 | 0 | 4 | 8 |
| Lead Thickness | c | .009 | .011 | .012 | 0.23 | 0.27 | 0.30 |
| Lead Width | B | .014 | .017 | .020 | 0.36 | 0.42 | 0.51 |
| Mold Draft Angle Top | α | 0 | 12 | 15 | 0 | 12 | 15 |
| Mold Draft Angle Bottom | β | 0 | 12 | 15 | 0 | 12 | 15 |

* Controlling Parameter
 § Significant Characteristic

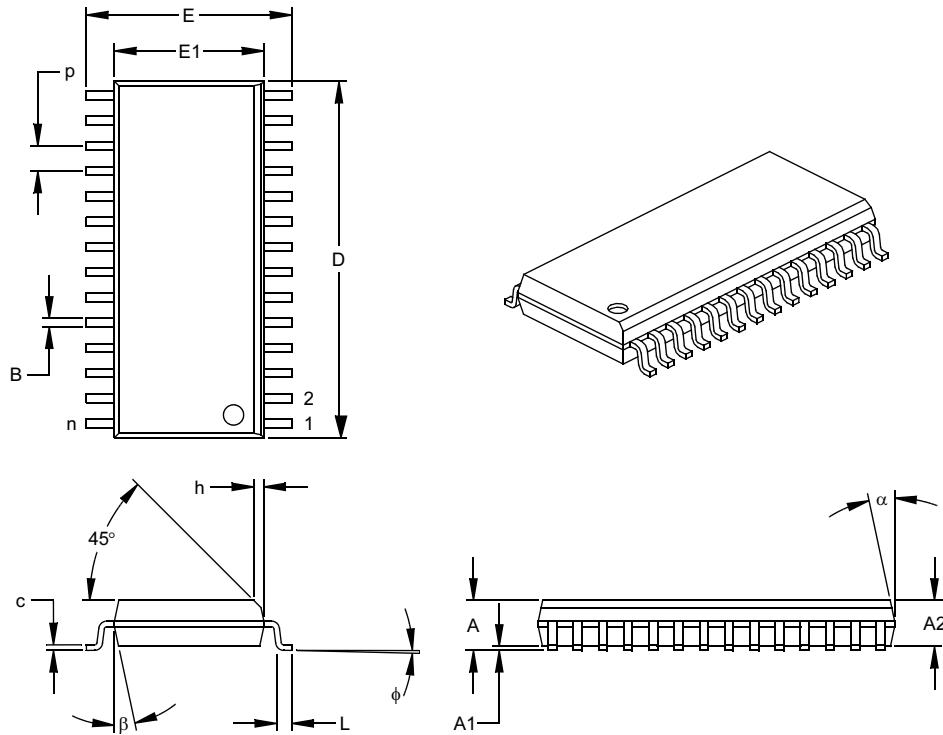
Notes:

Dimensions D and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed .010" (0.254mm) per side.

JEDEC Equivalent: MS-013

Drawing No. C04-051

28-Lead Plastic Small Outline (SO) – Wide, 300 mil (SOIC)



| Dimension Limits | Units | INCHES* | | | MILLIMETERS | | |
|--------------------------|-------|---------|------|------|-------------|-------|-------|
| | | MIN | NOM | MAX | MIN | NOM | MAX |
| Number of Pins | n | | 28 | | | 28 | |
| Pitch | p | | .050 | | | 1.27 | |
| Overall Height | A | .093 | .099 | .104 | 2.36 | 2.50 | 2.64 |
| Molded Package Thickness | A2 | .088 | .091 | .094 | 2.24 | 2.31 | 2.39 |
| Standoff § | A1 | .004 | .008 | .012 | 0.10 | 0.20 | 0.30 |
| Overall Width | E | .394 | .407 | .420 | 10.01 | 10.34 | 10.67 |
| Molded Package Width | E1 | .288 | .295 | .299 | 7.32 | 7.49 | 7.59 |
| Overall Length | D | .695 | .704 | .712 | 17.65 | 17.87 | 18.08 |
| Chamfer Distance | h | .010 | .020 | .029 | 0.25 | 0.50 | 0.74 |
| Foot Length | L | .016 | .033 | .050 | 0.41 | 0.84 | 1.27 |
| Foot Angle Top | φ | 0 | 4 | 8 | 0 | 4 | 8 |
| Lead Thickness | c | .009 | .011 | .013 | 0.23 | 0.28 | 0.33 |
| Lead Width | B | .014 | .017 | .020 | 0.36 | 0.42 | 0.51 |
| Mold Draft Angle Top | α | 0 | 12 | 15 | 0 | 12 | 15 |
| Mold Draft Angle Bottom | β | 0 | 12 | 15 | 0 | 12 | 15 |

* Controlling Parameter
 § Significant Characteristic

Notes:

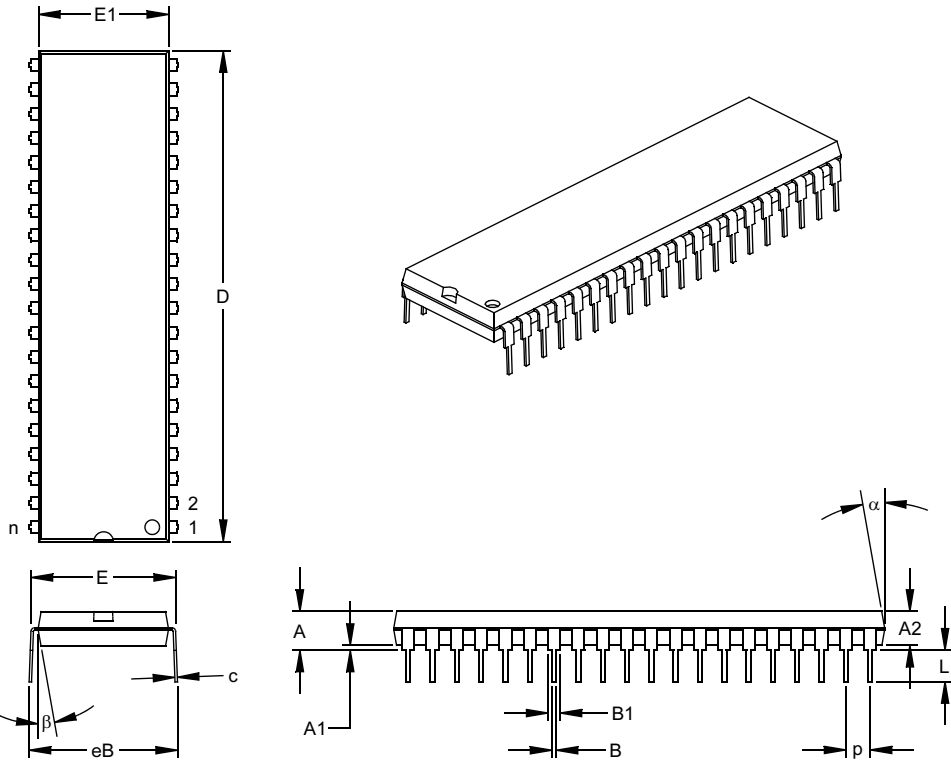
Dimensions D and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed .010" (0.254mm) per side.

JEDEC Equivalent: MS-013

Drawing No. C04-052

dsPIC30F

40-Lead Plastic Dual In-line (P) – 600 mil (PDIP)



| Units | | INCHES* | | | MILLIMETERS | | |
|----------------------------|------|---------|-------|-------|-------------|-------|-------|
| Dimension Limits | | MIN | NOM | MAX | MIN | NOM | MAX |
| Number of Pins | n | | 40 | | | 40 | |
| Pitch | p | | .100 | | | 2.54 | |
| Top to Seating Plane | A | .160 | .175 | .190 | 4.06 | 4.45 | 4.83 |
| Molded Package Thickness | A2 | .140 | .150 | .160 | 3.56 | 3.81 | 4.06 |
| Base to Seating Plane | A1 | .015 | | | 0.38 | | |
| Shoulder to Shoulder Width | E | .595 | .600 | .625 | 15.11 | 15.24 | 15.88 |
| Molded Package Width | E1 | .530 | .545 | .560 | 13.46 | 13.84 | 14.22 |
| Overall Length | D | 2.045 | 2.058 | 2.065 | 51.94 | 52.26 | 52.45 |
| Tip to Seating Plane | L | .120 | .130 | .135 | 3.05 | 3.30 | 3.43 |
| Lead Thickness | c | .008 | .012 | .015 | 0.20 | 0.29 | 0.38 |
| Upper Lead Width | B1 | .030 | .050 | .070 | 0.76 | 1.27 | 1.78 |
| Lower Lead Width | B | .014 | .018 | .022 | 0.36 | 0.46 | 0.56 |
| Overall Row Spacing | § eB | .620 | .650 | .680 | 15.75 | 16.51 | 17.27 |
| Mold Draft Angle Top | α | 5 | 10 | 15 | 5 | 10 | 15 |
| Mold Draft Angle Bottom | β | 5 | 10 | 15 | 5 | 10 | 15 |

* Controlling Parameter

§ Significant Characteristic

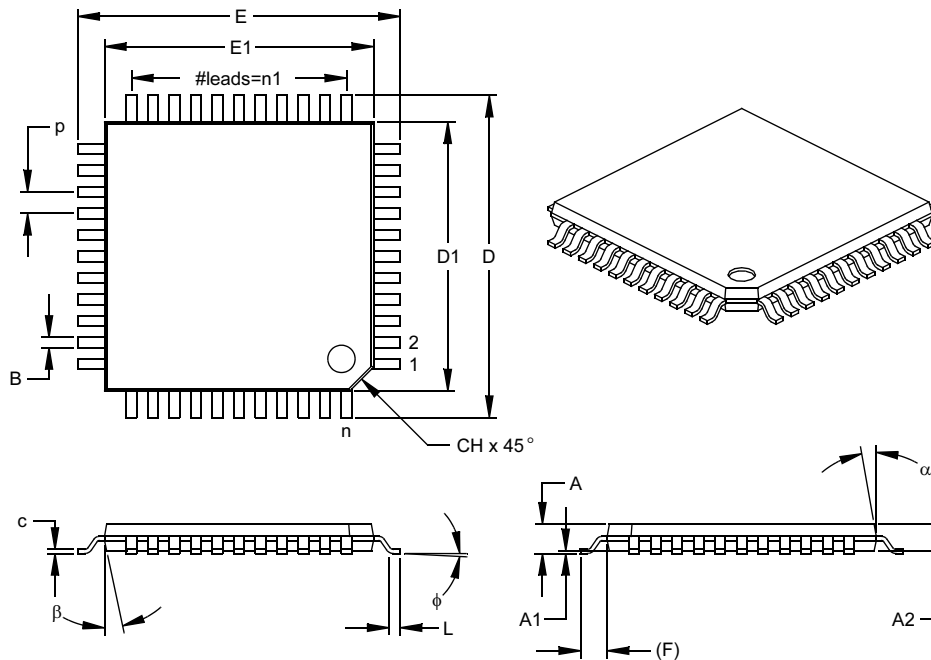
Notes:

Dimensions D and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed .010" (0.254mm) per side.

JEDEC Equivalent: MO-011

Drawing No. C04-016

44-Lead Plastic Thin Quad Flatpack (PT) 10x10x1 mm Body, 1.0/0.10 mm Lead Form (TQFP)



| Dimension Limits | Units | INCHES | | | MILLIMETERS* | | |
|--------------------------|-------|--------|------|------|--------------|-------|-------|
| | | MIN | NOM | MAX | MIN | NOM | MAX |
| Number of Pins | n | | 44 | | | 44 | |
| Pitch | p | | .031 | | | 0.80 | |
| Pins per Side | n1 | | 11 | | | 11 | |
| Overall Height | A | .039 | .043 | .047 | 1.00 | 1.10 | 1.20 |
| Molded Package Thickness | A2 | .037 | .039 | .041 | 0.95 | 1.00 | 1.05 |
| Standoff § | A1 | .002 | .004 | .006 | 0.05 | 0.10 | 0.15 |
| Foot Length | L | .018 | .024 | .030 | 0.45 | 0.60 | 0.75 |
| Footprint (Reference) | (F) | | .039 | | 1.00 | | |
| Foot Angle | phi | 0 | 3.5 | 7 | 0 | 3.5 | 7 |
| Overall Width | E | .463 | .472 | .482 | 11.75 | 12.00 | 12.25 |
| Overall Length | D | .463 | .472 | .482 | 11.75 | 12.00 | 12.25 |
| Molded Package Width | E1 | .390 | .394 | .398 | 9.90 | 10.00 | 10.10 |
| Molded Package Length | D1 | .390 | .394 | .398 | 9.90 | 10.00 | 10.10 |
| Lead Thickness | c | .004 | .006 | .008 | 0.09 | 0.15 | 0.20 |
| Lead Width | B | .012 | .015 | .017 | 0.30 | 0.38 | 0.44 |
| Pin 1 Corner Chamfer | CH | .025 | .035 | .045 | 0.64 | 0.89 | 1.14 |
| Mold Draft Angle Top | alpha | 5 | 10 | 15 | 5 | 10 | 15 |
| Mold Draft Angle Bottom | beta | 5 | 10 | 15 | 5 | 10 | 15 |

* Controlling Parameter
 § Significant Characteristic

Notes:

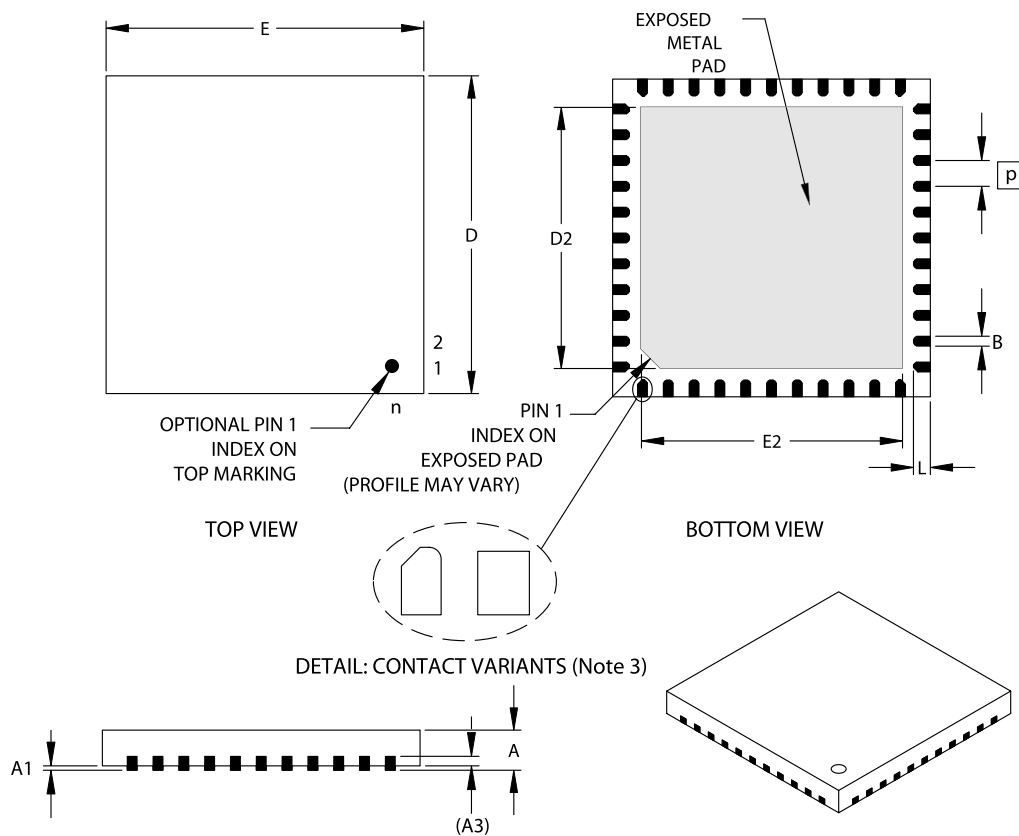
Dimensions D1 and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed .010" (0.254mm) per side.

JEDEC Equivalent: MS-026

Drawing No. C04-076

dsPIC30F

44-Lead Plastic Quad Flat No Lead Package (ML) 8x8 mm Body (QFN)



| Dimension Limits | Units | INCHES | | | MILLIMETERS* | | |
|--------------------|-------------|-----------------------|------|------|-----------------------|------|------|
| | | MIN | NOM | MAX | MIN | NOM | MAX |
| Number of Contacts | n | 44 | | | 44 | | |
| Pitch | \boxed{P} | .026 BSC ¹ | | | 0.65 BSC ¹ | | |
| Overall Height | A | .031 | .035 | .039 | 0.80 | 0.90 | 1.00 |
| Standoff | A1 | .000 | .001 | .002 | 0 | 0.02 | 0.05 |
| Base Thickness | (A3) | .010 REF ² | | | 0.25 REF ² | | |
| Overall Width | E | .309 | .315 | .321 | 7.85 | 8.00 | 8.15 |
| Exposed Pad Width | E2 | .246 | .268 | .274 | 6.25 | 6.80 | 6.95 |
| Overall Length | D | .309 | .315 | .321 | 7.85 | 8.00 | 8.15 |
| Exposed Pad Length | D2 | .246 | .268 | .274 | 6.25 | 6.80 | 6.95 |
| Contact Width | B | .008 | .013 | .013 | 0.20 | 0.33 | 0.35 |
| Contact Length | L | .014 | .016 | .019 | 0.35 | 0.40 | 0.48 |

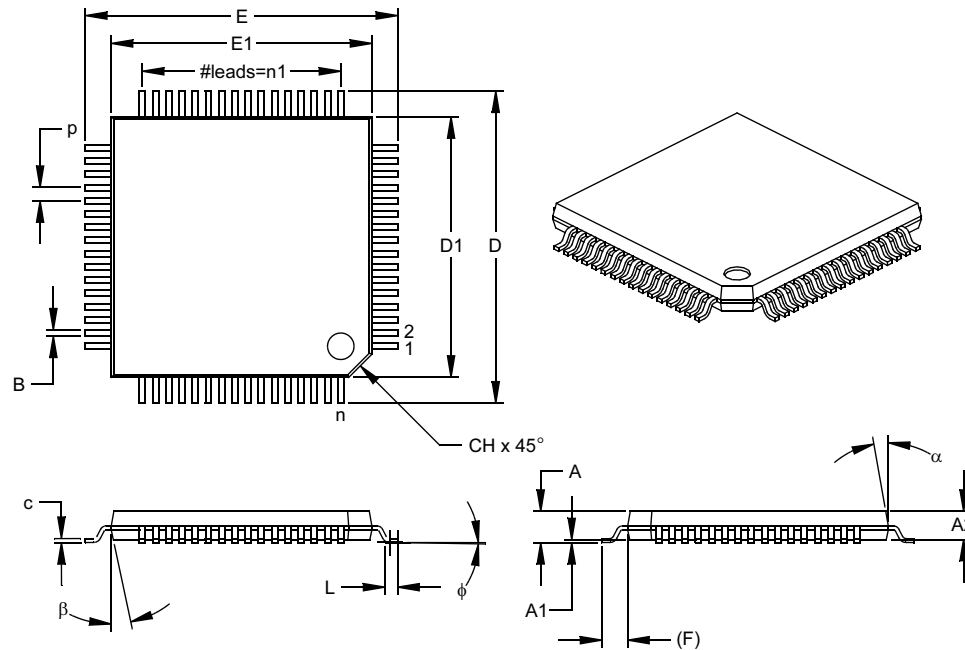
*Controlling Parameter

Notes:

1. BSC: Basic Dimension. Theoretically exact value shown without tolerances. See ASME Y14.5M
2. REF: Reference Dimension, usually without tolerance, for information purposes only. See ASME Y14.5M
3. Contact profiles may vary.

JEDEC equivalent: M0-220
Drawing No. C04-103

64-Lead Plastic Thin Quad Flatpack (PT) 10x10x1 mm Body, 1.0/0.10 mm Lead Form (TQFP)



| Dimension Limits | Units | INCHES | | | MILLIMETERS* | | |
|--------------------------|-------|--------|------|------|--------------|-------|-------|
| | | MIN | NOM | MAX | MIN | NOM | MAX |
| Number of Pins | n | | 64 | | | 64 | |
| Pitch | p | | .020 | | | 0.50 | |
| Pins per Side | n1 | | 16 | | | 16 | |
| Overall Height | A | .039 | .043 | .047 | 1.00 | 1.10 | 1.20 |
| Molded Package Thickness | A2 | .037 | .039 | .041 | 0.95 | 1.00 | 1.05 |
| Standoff § | A1 | .002 | .006 | .010 | 0.05 | 0.15 | 0.25 |
| Foot Length | L | .018 | .024 | .030 | 0.45 | 0.60 | 0.75 |
| Footprint (Reference) | (F) | | .039 | | | 1.00 | |
| Foot Angle | φ | 0 | 3.5 | 7 | 0 | 3.5 | 7 |
| Overall Width | E | .463 | .472 | .482 | 11.75 | 12.00 | 12.25 |
| Overall Length | D | .463 | .472 | .482 | 11.75 | 12.00 | 12.25 |
| Molded Package Width | E1 | .390 | .394 | .398 | 9.90 | 10.00 | 10.10 |
| Molded Package Length | D1 | .390 | .394 | .398 | 9.90 | 10.00 | 10.10 |
| Lead Thickness | c | .005 | .007 | .009 | 0.13 | 0.18 | 0.23 |
| Lead Width | B | .007 | .009 | .011 | 0.17 | 0.22 | 0.27 |
| Pin 1 Corner Chamfer | CH | .025 | .035 | .045 | 0.64 | 0.89 | 1.14 |
| Mold Draft Angle Top | α | 5 | 10 | 15 | 5 | 10 | 15 |
| Mold Draft Angle Bottom | β | 5 | 10 | 15 | 5 | 10 | 15 |

* Controlling Parameter
 § Significant Characteristic

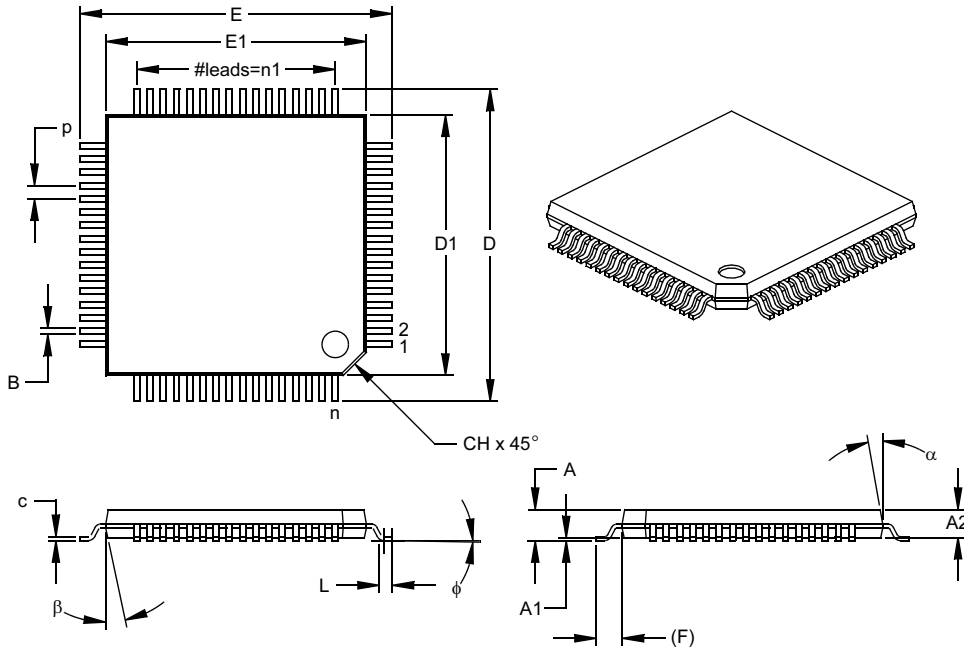
Notes:

Dimensions D1 and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed .010" (0.254mm) per side.

JEDEC Equivalent: MS-026
 Drawing No. C04-085

dsPIC30F

64-Lead Plastic Thin Quad Flatpack (PF) 14x14x1 mm Body, 1.0/0.10 mm Lead Form (TQFP)



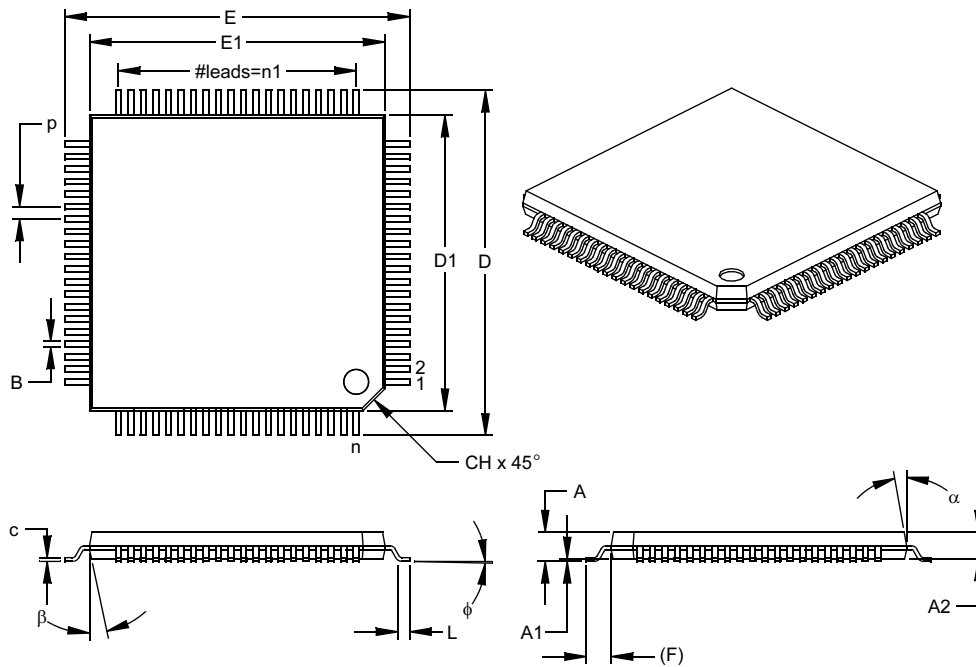
| Dimension Limits | Units | INCHES | | | MILLIMETERS* | | |
|--------------------------|-------|--------|------|------|--------------|-------|------|
| | | MIN | NOM | MAX | MIN | NOM | MAX |
| Number of Pins | n | | 64 | | | 64 | |
| Pitch | p | | .032 | | | 0.80 | |
| Pins per Side | n1 | | 16 | | | 16 | |
| Overall Height | A | | | .047 | | | 1.20 |
| Molded Package Thickness | A2 | .037 | .039 | .041 | 0.95 | 1.00 | 1.05 |
| Standoff § | A1 | .002 | | .006 | 0.05 | | 0.15 |
| Foot Length | L | .018 | .024 | .030 | 0.45 | 0.60 | 0.75 |
| Footprint (Reference) | (F) | | .039 | | | 1.00 | |
| Foot Angle | φ | 0 | | 7 | 0 | | 7 |
| Overall Width | E | | .630 | | | 16.00 | |
| Overall Length | D | | .630 | | | 16.00 | |
| Molded Package Width | E1 | | .551 | | | 14.00 | |
| Molded Package Length | D1 | | .551 | | | 14.00 | |
| Lead Thickness | c | .004 | | .008 | 0.09 | | 0.20 |
| Lead Width | B | .019 | .013 | .018 | 0.30 | 0.32 | 0.45 |
| Pin 1 Corner Chamfer | CH | | | | | | |
| Mold Draft Angle Top | α | 11 | | 13 | 11 | | 13 |
| Mold Draft Angle Bottom | β | 11 | | 13 | 11 | | 13 |

* Controlling Parameter
 § Significant Characteristic

Notes:

Dimensions D1 and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed .010" (0.254mm) per side.
 JEDEC Equivalent: MS-026
 Drawing No. C04-085

80-Lead Plastic Thin Quad Flatpack (PT) 12x12x1 mm Body, 1.0/0.10 mm Lead Form (TQFP)



| Dimension Limits | Units | INCHES | | | MILLIMETERS* | | |
|--------------------------|-------|--------|------|------|--------------|-------|-------|
| | | MIN | NOM | MAX | MIN | NOM | MAX |
| Number of Pins | n | | 80 | | | 80 | |
| Pitch | p | | .020 | | | 0.50 | |
| Pins per Side | n1 | | 20 | | | 20 | |
| Overall Height | A | .039 | .043 | .047 | 1.00 | 1.10 | 1.20 |
| Molded Package Thickness | A2 | .037 | .039 | .041 | 0.95 | 1.00 | 1.05 |
| Standoff § | A1 | .002 | .004 | .006 | 0.05 | 0.10 | 0.15 |
| Foot Length | L | .018 | .024 | .030 | 0.45 | 0.60 | 0.75 |
| Footprint (Reference) | (F) | | .039 | | | 1.00 | |
| Foot Angle | φ | 0 | 3.5 | 7 | 0 | 3.5 | 7 |
| Overall Width | E | .541 | .551 | .561 | 13.75 | 14.00 | 14.25 |
| Overall Length | D | .541 | .551 | .561 | 13.75 | 14.00 | 14.25 |
| Molded Package Width | E1 | .463 | .472 | .482 | 11.75 | 12.00 | 12.25 |
| Molded Package Length | D1 | .463 | .472 | .482 | 11.75 | 12.00 | 12.25 |
| Lead Thickness | c | .004 | .006 | .008 | 0.09 | 0.15 | 0.20 |
| Lead Width | B | .007 | .009 | .011 | 0.17 | 0.22 | 0.27 |
| Pin 1 Corner Chamfer | CH | .025 | .035 | .045 | 0.64 | 0.89 | 1.14 |
| Mold Draft Angle Top | α | 5 | 10 | 15 | 5 | 10 | 15 |
| Mold Draft Angle Bottom | β | 5 | 10 | 15 | 5 | 10 | 15 |

* Controlling Parameter
 § Significant Characteristic

Notes:

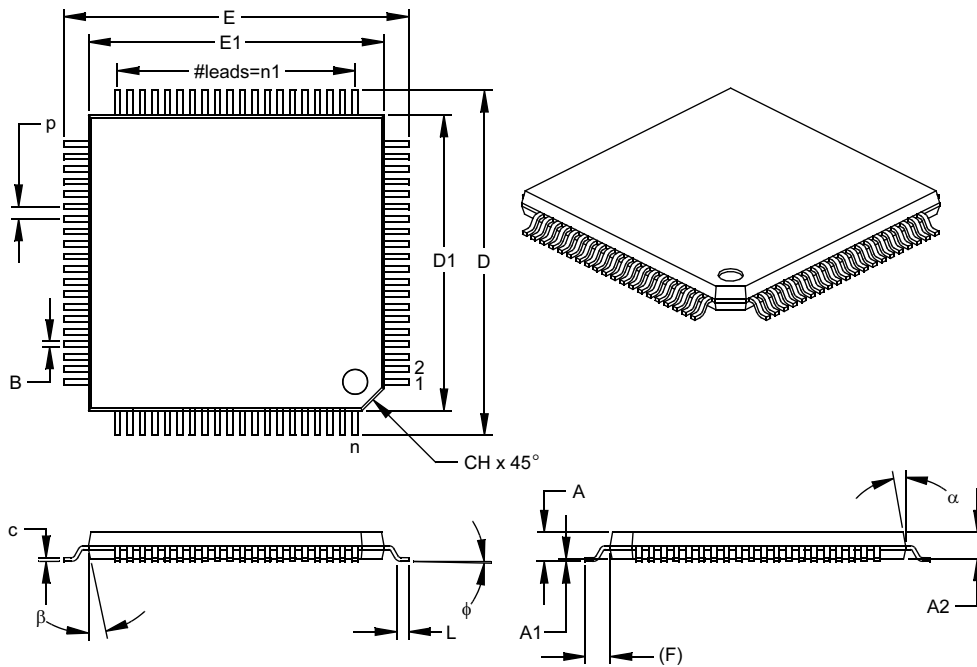
Dimensions D1 and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed .010" (0.254mm) per side.

JEDEC Equivalent: MS-026

Drawing No. C04-092

dsPIC30F

80-Lead Plastic Thin Quad Flatpack (PF) 14x14x1 mm Body, 1.0/0.10 mm Lead Form (TQFP)



| Dimension Limits | Units | INCHES | | | MILLIMETERS* | | |
|--------------------------|-------|--------|------|------|--------------|-------|------|
| | | MIN | NOM | MAX | MIN | NOM | MAX |
| Number of Pins | n | | 80 | | | 80 | |
| Pitch | p | | .026 | | | 0.65 | |
| Pins per Side | n1 | | 20 | | | 20 | |
| Overall Height | A | | | .047 | | | 1.20 |
| Molded Package Thickness | A2 | .037 | .039 | .041 | 0.95 | 1.00 | 1.05 |
| Standoff § | A1 | .002 | | .006 | 0.05 | | 0.15 |
| Foot Length | L | .018 | .024 | .030 | 0.45 | 0.60 | 0.75 |
| Footprint (Reference) | (F) | | .039 | | | 1.00 | |
| Foot Angle | φ | 0 | | 7 | 0 | | 7 |
| Overall Width | E | | .630 | | | 16.00 | |
| Overall Length | D | | .630 | | | 16.00 | |
| Molded Package Width | E1 | | .551 | | | 14.00 | |
| Molded Package Length | D1 | | .551 | | | 14.00 | |
| Lead Thickness | c | .004 | | .008 | 0.09 | | 0.20 |
| Lead Width | B | .009 | .013 | .015 | 0.22 | 0.32 | 0.38 |
| Pin 1 Corner Chamfer | CH | | | | | | |
| Mold Draft Angle Top | α | 11 | | 13 | 11 | | 13 |
| Mold Draft Angle Bottom | β | 11 | | 13 | 11 | | 13 |

* Controlling Parameter
 § Significant Characteristic

Notes:

Dimensions D1 and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed .010" (0.254mm) per side.

JEDEC Equivalent: MS-026

Drawing No. C04-092

INDEX

Numerics

12-bit Analog-to-Digital Converter (A/D) Module 145

A

A/D 145

Aborting a Conversion 147

 ADC_{CHS} Register 145

ADCON1 Register 145

ADCON2 Register 145

ADCON3 Register 145

ADCSSL Register 145

ADPCFG Register 145

Configuring Analog Port Pins 76, 150

Connection Considerations 150

Conversion Operation 146

Effects of a Reset 149

Operation During CPU Idle Mode 149

Operation During CPU Sleep Mode 149

Output Formats 149

Power-down Modes 149

Programming the Sample Trigger 147

Register Map 151

Result Buffer 146

Sampling Requirements 148

Selecting the Conversion Clock 147

Selecting the Conversion Sequence 146

AC Characteristics 195

Load Conditions 195

AC Temperature and Voltage Specifications 195

AC-Link Mode Operation 142

16-bit Mode 142

20-bit Mode 142

Address Generator Units 47

Alternate Vector Table 59

Analog-to-Digital Converter. *See* A/D.

Assembler

MPASM Assembler 177

Automatic Clock Stretch 110

During 10-bit Addressing (STREN = 1) 110

During 7-bit Addressing (STREN = 1) 110

Receive Mode 110

Transmit Mode 110

B

Bandgap Start-up Time

Requirements 200

Timing Characteristics 200

Barrel Shifter 33

Bit Reversed Addressing

Example 54

Implementation 53

Bit-Reversed Addressing 53

Modifier Values Table 54

Sequence Table (16-Entry) 54

Block Diagrams

12-bit A/D Functional 145

16-bit Timer1 Module 81

16-bit Timer2 87

16-bit Timer3 87

16-bit Timer4 92

16-bit Timer5 92

32-bit Timer2/3 86

32-bit Timer4/5 91

CAN Buffers and Protocol Engine 124

DCI Module 136

Dedicated Port Structure 75

DSP Engine 29

dsPIC30F5013/6013/6014 18

External Power-on Reset Circuit 162

I²C 108

Input Capture Mode 95

Oscillator System 155

Output Compare Mode 99

Reset System 159

Shared Port Structure 76

SPI 104

SPI Master/Slave Connection 104

UART Receiver 116

UART Transmitter 115

BOR Characteristics 194

BOR. *See* Brown-out Reset.

Brown-out Reset

Characteristics 193

Timing Requirements 200

C

C Compilers

MPLAB C17 178

MPLAB C18 178

MPLAB C30 178

CAN Module 123

Baud Rate Setting 128

CAN1 Register Map 130

CAN2 Register Map 132

Frame Types 123

I/O Timing Characteristics 217

Message Reception 126

Message Transmission 127

Modes of Operation 125

Overview 123

CLKOUT and I/O Timing

Characteristics 198

Requirements 198

Code Examples

Data EEPROM Block Erase 70

Data EEPROM Block Write 72

Data EEPROM Read 69

Data EEPROM Word Erase 70

Data EEPROM Word Write 71

Erasing a Row of Program Memory 65

Initiating a Programming Sequence 66

Loading Write Latches 66

Code Protection 153

Core Architecture 21

Overview 21

D

Data Accumulators and Adder/Subtractor 31

Data Space Write Saturation 33

Overflow and Saturation 31

Round Logic 32

Write Back 32

Data Address Space 39

Alignment 40

Alignment (Figure) 40

Effect of Invalid Memory Accesses (Table) 40

MCU and DSP (MAC Class) Instructions Example 43

Memory Map 40

Near Data Space 41

dsPIC30F

| | | | |
|---|-----|---|-----|
| Sample Memory Map | 42 | Timing Requirements | |
| Software Stack | 41 | AC-Link Mode | 207 |
| Spaces | 39 | Multichannel, I ² S Modes | 206 |
| Width | 40 | Transmit Slot Enable Bits | 139 |
| Data Converter Interface (DCI) Module | 135 | Transmit Status Bits | 141 |
| Data EEPROM Memory | 69 | Transmit/Receive Shift Register | 135 |
| Erasing | 70 | Underflow Mode Control Bit | 141 |
| Erasing, Block | 70 | Word Size Selection Bits | 137 |
| Erasing, Word | 70 | Demonstration Boards | |
| Protection Against Spurious Write | 73 | PICDEM 1 | 180 |
| Reading | 69 | PICDEM 17 | 180 |
| Write Verify | 73 | PICDEM 18R PIC18C601/801 | 181 |
| Writing | 71 | PICDEM 2 Plus | 180 |
| Writing, Block | 72 | PICDEM 3 PIC16C92X | 180 |
| Writing, Word | 71 | PICDEM 4 | 180 |
| Data Space Organization | 47 | PICDEM LIN PIC16C43X | 181 |
| DC Characteristics | 184 | PICDEM USB PIC16C7X5 | 181 |
| BOR | 194 | PICDEM.net Internet/Ethernet | 180 |
| Brown-out Reset | 193 | Development Support | 177 |
| I/O Pin Input Specifications | 191 | Device Configuration | |
| I/O Pin Output Specifications | 192 | Register Map | 168 |
| Idle Current (IDLE) | 187 | Device Configuration Registers | |
| Low-Voltage Detect | 192 | FBORPOR | 166 |
| LVDL | 193 | FGS | 166 |
| Operating Current (IDD) | 185 | FOSC | 166 |
| Power-Down Current (IPD) | 189 | FWDT | 166 |
| Program and EEPROM | 194 | Device Overview | 17 |
| Temperature and Voltage Specifications | 184 | Disabling the UART | 117 |
| DCI Module | | Divide Support | 27 |
| Bit Clock Generator | 139 | Instructions (Table) | 27 |
| Buffer Alignment with Data Frames | 140 | DSP Engine | 28 |
| Buffer Control | 135 | Multiplier | 30 |
| Buffer Data Alignment | 135 | Dual Output Compare Match Mode | 100 |
| Buffer Length Control | 140 | Continuous Pulse Mode | 100 |
| COFS Pin | 135 | Single Pulse Mode | 100 |
| CSCK Pin | 135 | E | |
| CSDI Pin | 135 | Electrical Characteristics | 183 |
| CSDO Mode Bit | 141 | AC | 195 |
| CSDO Pin | 135 | DC | 184 |
| Data Justification Control Bit | 139 | Enabling and Setting Up UART | |
| Device Frequencies for Common Codec CSCK Frequen- | | Alternate I/O | 117 |
| cies (Table) | 139 | Setting Up Data, Parity and Stop Bit Selections | 117 |
| Digital Loopback Mode | 141 | Enabling the UART | 117 |
| Enable | 137 | Equations | |
| Frame Sync Generator | 137 | A/D Conversion Clock | 147 |
| Frame Sync Mode Control Bits | 137 | Baud Rate | 119 |
| I/O Pins | 135 | Bit Clock Frequency | 139 |
| Interrupts | 141 | COFSG Period | 137 |
| Introduction | 135 | Serial Clock Rate | 112 |
| Master Frame Sync Operation | 137 | Time Quantum for Clock Generation | 129 |
| Operation | 137 | Errata | 16 |
| Operation During CPU Idle Mode | 142 | Evaluation and Programming Tools | 181 |
| Operation During CPU Sleep Mode | 142 | Exception Processing | 55 |
| Receive Slot Enable Bits | 140 | External Clock Timing Characteristics | |
| Receive Status Bits | 141 | Type A, B and C Timer | 201 |
| Register Map | 143 | External Clock Timing Requirements | 196 |
| Sample Clock Edge Control Bit | 139 | Type A Timer | 201 |
| Slave Frame Sync Operation | 138 | Type B Timer | 202 |
| Slot Enable Bits Operation with Frame Sync | 140 | Type C Timer | 202 |
| Slot Status Bits | 141 | External Interrupt Requests | 60 |
| Synchronous Data Transfers | 140 | | |
| Timing Characteristics | | | |
| AC-Link Mode | 207 | | |
| Multichannel, I ² S Modes | 205 | | |

F

| | |
|----------------------------|----|
| Fast Context Saving | 60 |
| Flash Program Memory | 63 |
| Control Registers | 64 |
| NVMADR | 64 |
| NVMADRU | 64 |
| NVMCON | 64 |
| NVMKEY | 64 |

I

| | |
|--|---------|
| I/O Pin Specifications | |
| Input | 191 |
| Output | 192 |
| I/O Ports | 75 |
| Parallel (PIO) | 75 |
| I ² C | 107 |
| I ² C 10-bit Slave Mode Operation | 109 |
| Reception | 110 |
| Transmission | 109 |
| I ² C 7-bit Slave Mode Operation | 109 |
| Reception | 109 |
| Transmission | 109 |
| I ² C Master Mode Operation | 111 |
| Baud Rate Generator | 112 |
| Clock Arbitration | 112 |
| Multi-Master Communication, Bus Collision and Bus Arbitration | 112 |
| Reception | 111 |
| Transmission | 111 |
| I ² C Master Mode Support | 111 |
| I ² C Module | 107 |
| Addresses | 109 |
| Bus Data Timing Characteristics | |
| Master Mode | 213 |
| Slave Mode | 215 |
| Bus Data Timing Requirements | |
| Master Mode | 214 |
| Slave Mode | 216 |
| Bus Start/Stop Bits Timing Characteristics | |
| Master Mode | 213 |
| Slave Mode | 215 |
| General Call Address Support | 111 |
| Interrupts | 110 |
| IPMI Support | 111 |
| Operating Function Description | 107 |
| Operation During CPU Sleep and Idle Modes | 112 |
| Pin Configuration | 107 |
| Programmer's Model | 107 |
| Register Map | 113 |
| Registers | 107 |
| Slope Control | 111 |
| Software Controlled Clock Stretching (STREN = 1) | 110 |
| Various Modes | 107 |
| I ² S Mode Operation | 142 |
| Data Justification | 142 |
| Frame and Data Word Length Selection | 142 |
| Idle Current (IDLE) | 187 |
| In-Circuit Serial Programming (ICSP) | 63, 153 |
| Input Capture (CAPX) Timing Characteristics | 203 |
| Input Capture Module | 95 |
| Interrupts | 97 |
| Register Map | 98 |
| Input Capture Operation During Sleep and Idle Modes | 97 |
| CPU Idle Mode | 97 |
| CPU Sleep Mode | 97 |

| | |
|---|-----|
| Input Capture Timing Requirements | 203 |
| Input Change Notification Module | 79 |
| Register Map (Bits 15-8) | 79 |
| Register Map (Bits 7-0) | 79 |
| Instruction Addressing Modes | 47 |
| File Register Instructions | 48 |
| Fundamental Modes Supported | 47 |
| MAC Instructions | 48 |
| MCU Instructions | 48 |
| Move and Accumulator Instructions | 48 |
| Other Instructions | 48 |
| Instruction Flow | 24 |
| Pipeline | |
| 1-Word, 1-Cycle (Figure) | 24 |
| 1-Word, 2-Cycle (Figure) | 24 |
| 1-Word, 2-Cycle MOV.D Operations (Figure) | 25 |
| 1-Word, 2-Cycle Table Operations (Figure) | 25 |
| 1-Word, 2-Cycle with Instruction Stall (Figure) | 26 |
| 2-Word, 2-Cycle DO, DOW (Figure) | 26 |
| 2-Word, 2-Cycle GOTO, CALL (Figure) | 25 |
| Instruction Set | |
| Overview | 172 |
| Summary | 169 |
| Instruction Stalls | 49 |
| Introduction | 49 |
| Raw Dependency Detection | 49 |
| Inter-Integrated Circuit. See I ² C. | |
| Internal Clock Timing Examples | 197 |
| Interrupt Controller | |
| Register Map | 61 |
| Interrupt Priority | 56 |
| Interrupt Sequence | 59 |
| Interrupt Stack Frame | 59 |
| Interrupts | 55 |

L

| | |
|--|-----|
| Load Conditions | 195 |
| Low Voltage Detect (LVD) | 165 |
| Low-Voltage Detect Characteristics | 192 |
| LVDL Characteristics | 193 |

M

| | |
|--|-----|
| Memory Organization | 17 |
| Core Register Map | 43 |
| Modes of Operation | |
| Disable | 125 |
| Initialization | 125 |
| Listen All Messages | 125 |
| Listen Only | 125 |
| Loopback | 125 |
| Normal Operation | 125 |
| Modulo Addressing | 50 |
| Applicability | 53 |
| Decrementing Buffer Operation Example | 52 |
| Incrementing Buffer Operation Example | 51 |
| Restrictions | 53 |
| Start and End Address | 50 |
| W Address Register Selection | 51 |
| MPLAB ASM30 Assembler, Linker, Librarian | 178 |
| MPLAB ICD 2 In-Circuit Debugger | 179 |
| MPLAB ICE 2000 High Performance Universal In-Circuit Emulator | 179 |
| MPLAB ICE 4000 High Performance Universal In-Circuit Emulator | 179 |

dsPIC30F

| | | | |
|--|-----|--|----------|
| MPLAB Integrated Development Environment Software .. | 177 | Data Access from, Address Generation | 35 |
| MPLINK Object Linker/MPLIB Object Librarian | 178 | Data Space Window into Operation..... | 38 |
| Multiplier | | Data Table Access (LS Word) | 36 |
| 16-bit Integer and Fractional Modes Example | 30 | Data Table Access (MS Byte)..... | 37 |
| N | | Memory Map..... | 39 |
| NVM | | Table Instructions | |
| Register Map..... | 67 | TBLRDH | 36 |
| O | | TBLRDL | 36 |
| OC/PWM Module Timing Characteristics..... | 204 | TBLWTH | 36 |
| Operating Current (IDD)..... | 185 | TBLWTL | 36 |
| Operating Frequency vs Voltage | | Visibility from Data Space..... | 37 |
| dsPIC30FXXXX-20 (Extended)..... | 184 | Program and EEPROM Characteristics..... | 194 |
| Oscillator | | Program Counter | 22 |
| Configurations | 156 | Programmable | 153 |
| Fail-Safe Clock Monitor..... | 158 | Programmer's Model | 22 |
| Fast RC (FRC)..... | 157 | Diagram | 23 |
| Initial Clock Source Selection | 156 | Programming Operations..... | 65 |
| Low Power RC (LPRC)..... | 158 | Algorithm for Program Flash..... | 65 |
| LP Oscillator Control | 157 | Erasing a Row of Program Memory..... | 65 |
| Phase Locked Loop (PLL) | 157 | Initiating the Programming Sequence..... | 66 |
| Start-up Timer (OST) | 157 | Loading Write Latches | 66 |
| Operating Modes (Table) | 154 | Protection Against Accidental Writes to OSCCON | 159 |
| System Overview | 153 | R | |
| Oscillator Selection | 153 | Reset | 153, 159 |
| Oscillator Start-up Timer | | BOR, Programmable | 162 |
| Timing Characteristics | 199 | Brown-out Reset (BOR)..... | 153 |
| Timing Requirements | 200 | Oscillator Start-up Timer (OST)..... | 153 |
| Output Compare Interrupts | 101 | POR | |
| Output Compare Module..... | 99 | Operating without FSCM and PWRT..... | 162 |
| Register Map..... | 102 | With Long Crystal Start-up Time | 162 |
| Timing Characteristics | 203 | POR (Power-on Reset)..... | 160 |
| Timing Requirements | 203 | Power-on Reset (POR)..... | 153 |
| Output Compare Operation During CPU Idle Mode..... | 101 | Power-up Timer (PWRT) | 153 |
| Output Compare Sleep Mode Operation..... | 101 | Reset Sequence | 57 |
| P | | Reset Sources | 57 |
| Packaging Information | 223 | Reset Sources | |
| Marking | 223 | Brown-out Reset (BOR)..... | 57 |
| Peripheral Module Disable (PMD) Registers | 167 | Illegal Instruction Trap | 57 |
| PICkit 1 Flash Starter Kit..... | 181 | Trap Lockout..... | 57 |
| PICSTART Plus Development Programmer | 179 | Uninitialized W Register Trap | 57 |
| Pinout Descriptions | 19 | Watchdog Time-out | 57 |
| PLL Clock Timing Specifications..... | 197 | Reset Timing Characteristics..... | 199 |
| POR. See Power-on Reset. | | Reset Timing Requirements | 200 |
| Port Write/Read Example..... | 76 | RTSP Operation | 64 |
| PORTA Register Map | 77 | Run-Time Self-Programming (RTSP)..... | 63 |
| PORTB Register Map | 77 | S | |
| PORTC Register Map | 77 | Serial Peripheral Interface. See SPI. | |
| PORTD Register Map | 77 | Simple Capture Event Mode..... | 96 |
| PORTF Register Map..... | 78 | Buffer Operation | 96 |
| PORTG Register Map..... | 78 | Hall Sensor Mode | 96 |
| Power Saving Modes | 165 | Prescaler | 96 |
| Idle | 166 | Timer2 and Timer3 Selection Mode..... | 96 |
| Sleep | 165 | Simple OC/PWM Mode Timing Requirements | 204 |
| Sleep and Idle | 153 | Simple Output Compare Match Mode | 100 |
| Power-Down Current (IPD) | 189 | Simple PWM Mode | 100 |
| Power-up Timer | | Input Pin Fault Protection | 100 |
| Timing Characteristics | 199 | Period | 101 |
| Timing Requirements | 200 | Software Simulator (MPLAB SIM) | 178 |
| PRO MATE II Universal Device Programmer | 179 | Software Simulator (MPLAB SIM30) | 178 |
| Program Address Space | 35 | Software Stack Pointer, Frame Pointer | 22 |
| Alignment and Data Access Using Table | | CALL Stack Frame | 41 |
| Instructions..... | 36 | SPI | 103 |
| Construction | 35 | SPI Module | 103 |
| | | Framed SPI Support..... | 103 |

| | | | |
|--|----------|---|-----|
| Operating Function Description | 103 | External Clock | 195 |
| Operation During CPU Idle Mode | 105 | I ² C Bus Data | |
| Operation During CPU Sleep Mode | 105 | Master Mode | 213 |
| SDOx Disable | 103 | Slave Mode | 215 |
| Slave Select Synchronization | 105 | I ² C Bus Start/Stop Bits | |
| SPI1 Register Map | 106 | Master Mode | 213 |
| SPI2 Register Map | 106 | Slave Mode | 215 |
| Timing Characteristics | | Input Capture (CAPX) | 203 |
| Master Mode (CKE = 0) | 208 | OC/PWM Module | 204 |
| Master Mode (CKE = 1) | 209 | Oscillator Start-up Timer | 199 |
| Slave Mode (CKE = 1) | 210, 211 | Output Compare Module | 203 |
| Timing Requirements | | Power-up Timer | 199 |
| Master Mode (CKE = 0) | 208 | Reset | 199 |
| Master Mode (CKE = 1) | 209 | SPI Module | |
| Slave Mode (CKE = 0) | 210 | Master Mode (CKE = 0) | 208 |
| Slave Mode (CKE = 1) | 212 | Master Mode (CKE = 1) | 209 |
| Word and Byte Communication | 103 | Slave Mode (CKE = 0) | 210 |
| Status Bits, Their Significance and the Initialization | | Slave Mode (CKE = 1) | 211 |
| Condition for RCON Register, Case 1 | 163 | Type A, B and C Timer External Clock | 201 |
| Status Bits, Their Significance and the Initialization | | Watchdog Timer | 199 |
| Condition for RCON Register, Case 2 | 164 | Timing Diagrams | |
| Status Register | 22 | CAN Bit | 128 |
| Z Status Bit | 22 | Frame Sync, AC-Link Start of Frame | 138 |
| Symbols Used in Opcode Descriptions | 170 | Frame Sync, Multi-Channel Mode | 138 |
| System Integration | 153 | I ² S Interface Frame Sync | 138 |
| Register Map | 168 | PWM Output | 101 |
| T | | Time-out Sequence on Power-up | |
| Table Instruction Operation Summary | 63 | (MCLR Not Tied to VDD), Case 1 | 160 |
| Temperature and Voltage Specifications | | Time-out Sequence on Power-up | |
| AC | 195 | (MCLR Not Tied to VDD), Case 2 | 161 |
| DC | 184 | Time-out Sequence on Power-up | |
| Timer1 Module | 81 | (MCLR Tied to VDD) | 160 |
| 16-bit Asynchronous Counter Mode | 81 | Timing Diagrams and Specifications | |
| 16-bit Synchronous Counter Mode | 81 | DC Characteristics - Internal RC Accuracy | 197 |
| 16-bit Timer Mode | 81 | Timing Diagrams. See Timing Characteristics | |
| Gate Operation | 82 | Timing Requirements | |
| Interrupt | 82 | A/D Conversion | |
| Operation During Sleep Mode | 82 | 10-Bit | 221 |
| Prescaler | 82 | Bandgap Start-up Time | 200 |
| Real-Time Clock | 82 | Brown-out Reset | 200 |
| Interrupts | 83 | CLKOUT and I/O | 198 |
| Oscillator Operation | 83 | DCI Module | |
| Register Map | 84 | AC-Link Mode | 207 |
| Timer2 and Timer3 Selection Mode | 100 | Multichannel, I ² S Modes | 206 |
| Timer2/3 Module | 85 | External Clock | 196 |
| 16-bit Timer Mode | 85 | I ² C Bus Data (Master Mode) | 214 |
| 32-bit Synchronous Counter Mode | 85 | I ² C Bus Data (Slave Mode) | 216 |
| 32-bit Timer Mode | 85 | Input Capture | 203 |
| ADC Event Trigger | 88 | Oscillator Start-up Timer | 200 |
| Gate Operation | 88 | Output Compare Module | 203 |
| Interrupt | 88 | Power-up Timer | 200 |
| Operation During Sleep Mode | 88 | Reset | 200 |
| Register Map | 89 | Simple OC/PWM Mode | 204 |
| Timer Prescaler | 88 | SPI Module | |
| Timer4/5 Module | 91 | Master Mode (CKE = 0) | 208 |
| Register Map | 93 | Master Mode (CKE = 1) | 209 |
| Timing Characteristics | | Slave Mode (CKE = 0) | 210 |
| A/D Conversion | | Slave Mode (CKE = 1) | 212 |
| 10-Bit (ASAM = 0, SSRC = 000) | 220 | Type A Timer External Clock | 201 |
| Bandgap Start-up Time | 200 | Type B Timer External Clock | 202 |
| CAN Module I/O | 217 | Type C Timer External Clock | 202 |
| CLKOUT and I/O | 198 | Watchdog Timer | 200 |
| DCI Module | | Timing Specifications | |
| AC-Link Mode | 207 | PLL Clock | 197 |
| Multichannel, I ² S Modes | 205 | | |

dsPIC30F

| | |
|----------------------------|----|
| Traps | 57 |
| Hard and Soft | 58 |
| Sources | 57 |
| Address Error Trap | 58 |
| Math Error Trap | 57 |
| Oscillator Fail Trap | 58 |
| Stack Error Trap | 58 |

U

| | |
|---|-----|
| UART Module | |
| Address Detect Mode | 119 |
| Auto Baud Support | 120 |
| Baud Rate Generator | 119 |
| Enabling and Setting Up | 117 |
| Framing Error (FERR) | 119 |
| Idle Status | 119 |
| Loopback Mode | 119 |
| Operation During CPU Sleep and Idle Modes | 120 |
| Overview | 115 |
| Parity Error (PERR) | 119 |
| Receive Break | 119 |
| Receive Buffer (UxRXB) | 118 |
| Receive Buffer Overrun Error (OERR Bit) | 118 |
| Receive Interrupt | 118 |
| Receiving Data | 118 |
| Receiving in 8-bit or 9-bit Data Mode | 118 |
| Reception Error Handling | 118 |

| | |
|---------------------------------------|-----|
| Transmit Break | 118 |
| Transmit Buffer (UxTXB) | 117 |
| Transmit Interrupt | 118 |
| Transmitting Data | 117 |
| Transmitting in 8-bit Data Mode | 117 |
| Transmitting in 9-bit Data Mode | 117 |
| UART1 Register Map | 121 |
| UART2 Register Map | 121 |

UART Operation

| | |
|---|-----|
| Idle Mode | 120 |
| Sleep Mode | 120 |
| Unit ID Locations | 153 |
| Universal Asynchronous Receiver Transmitter. <i>See</i> UART. | |

W

| | |
|-----------------------------------|----------|
| Wake-up from Sleep | 153 |
| Wake-up from Sleep and Idle | 60 |
| Watchdog Timer | |
| Timing Characteristics | 199 |
| Timing Requirements | 200 |
| Watchdog Timer (WDT) | 153, 165 |
| Enabling and Disabling | 165 |
| Operation | 165 |
| WWW, On-Line Support | 16 |

ON-LINE SUPPORT

Microchip provides on-line support on the Microchip World Wide Web site.

The web site is used by Microchip as a means to make files and information easily available to customers. To view the site, the user must have access to the Internet and a web browser, such as Netscape® or Microsoft® Internet Explorer. Files are also available for FTP download from our FTP site.

Connecting to the Microchip Internet Web Site

The Microchip web site is available at the following URL:

www.microchip.com

The file transfer site is available by using an FTP service to connect to:

<ftp://ftp.microchip.com>

The web site and file transfer site provide a variety of services. Users may download files for the latest Development Tools, Data Sheets, Application Notes, User's Guides, Articles and Sample Programs. A variety of Microchip specific business information is also available, including listings of Microchip sales offices, distributors and factory representatives. Other data available for consideration is:

- Latest Microchip Press Releases
- Technical Support Section with Frequently Asked Questions
- Design Tips
- Device Errata
- Job Postings
- Microchip Consultant Program Member Listing
- Links to other useful web sites related to Microchip Products
- Conferences for products, Development Systems, technical information and more
- Listing of seminars and events

SYSTEMS INFORMATION AND UPGRADE HOT LINE

The Systems Information and Upgrade Line provides system users a listing of the latest versions of all of Microchip's development systems software products. Plus, this line provides information on how customers can receive the most current upgrade kits. The Hot Line Numbers are:

1-800-755-2345 for U.S. and most of Canada, and

1-480-792-7302 for the rest of the world.

042003

dsPIC30F

READER RESPONSE

It is our intention to provide you with the best documentation possible to ensure successful use of your Microchip product. If you wish to provide your comments on organization, clarity, subject matter, and ways in which our documentation can better serve you, please FAX your comments to the Technical Publications Manager at (480) 792-4150.

Please list the following information, and use this outline to provide us with your comments about this document.

To: Technical Publications Manager
RE: Reader Response
From: Name _____
Company _____
Address _____
City / State / ZIP / Country _____
Telephone: (_____) _____ - _____ FAX: (_____) _____ - _____

Application (optional):

Would you like a reply? ___Y ___N

Device: dsPIC30F

Literature Number: DS70083G

Questions:

1. What are the best features of this document?

2. How does this document meet your hardware and software development needs?

3. Do you find the organization of this document easy to follow? If not, why?

4. What additions to the document do you think would enhance the structure and subject?

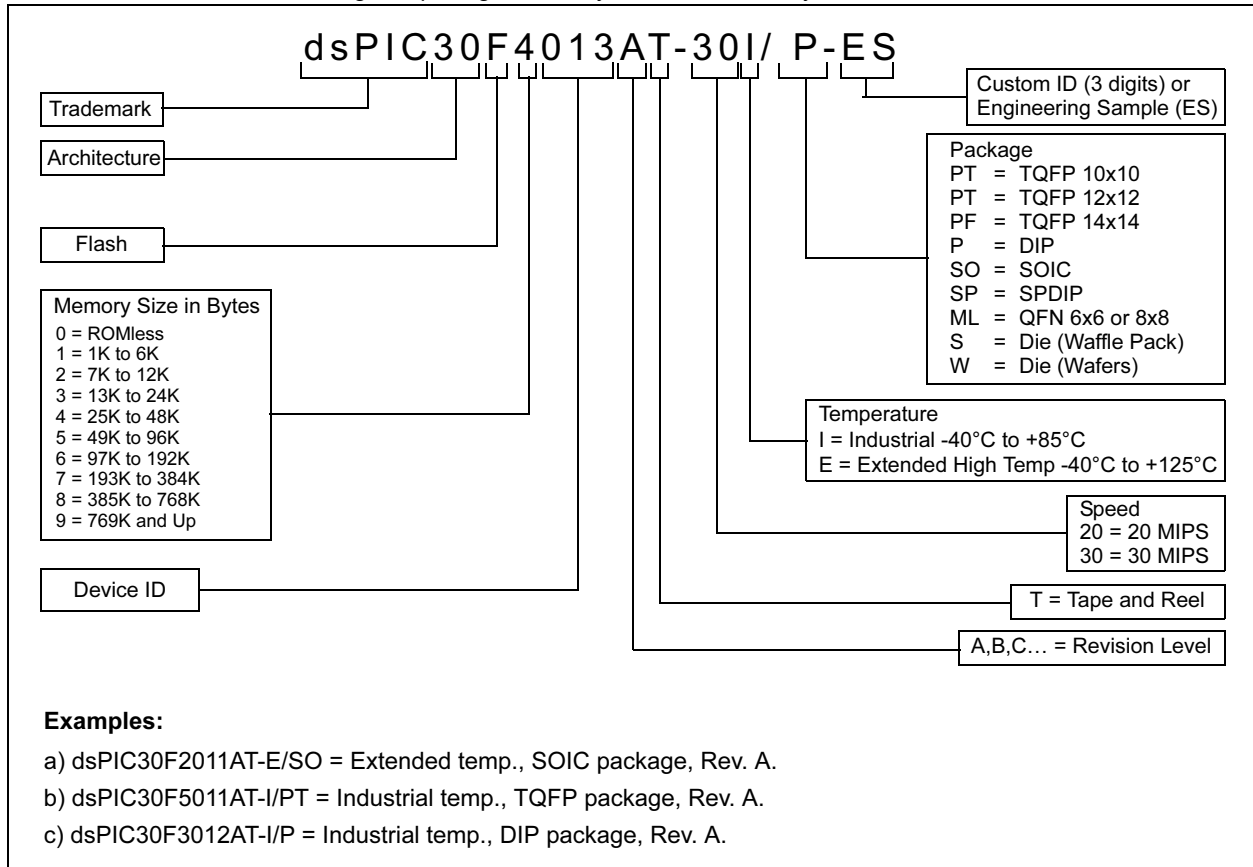
5. What deletions from the document could be made without affecting the overall usefulness?

6. Is there any incorrect or misleading information (what and where)?

7. How would you improve this document?

PRODUCT IDENTIFICATION SYSTEM

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





WORLDWIDE SALES AND SERVICE

AMERICAS

Corporate Office
2355 West Chandler Blvd.
Chandler, AZ 85224-6199
Tel: 480-792-7200
Fax: 480-792-7277
Technical Support:
480-792-7627
Web Address:
www.microchip.com

Atlanta
Alpharetta, GA
Tel: 770-640-0034
Fax: 770-640-0307

Boston
Westford, MA
Tel: 978-692-3848
Fax: 978-692-3821

Chicago
Itasca, IL
Tel: 630-285-0071
Fax: 630-285-0075

Dallas
Addison, TX
Tel: 972-818-7423
Fax: 972-818-2924

Detroit
Farmington Hills, MI
Tel: 248-538-2250
Fax: 248-538-2260

Kokomo
Kokomo, IN
Tel: 765-864-8360
Fax: 765-864-8387

Los Angeles
Mission Viejo, CA
Tel: 949-462-9523
Fax: 949-462-9608

San Jose
Mountain View, CA
Tel: 650-215-1444
Fax: 650-961-0286

Toronto
Mississauga, Ontario,
Canada
Tel: 905-673-0699
Fax: 905-673-6509

ASIA/PACIFIC

Australia - Sydney
Tel: 61-2-9868-6733
Fax: 61-2-9868-6755

China - Beijing
Tel: 86-10-8528-2100
Fax: 86-10-8528-2104

China - Chengdu
Tel: 86-28-8676-6200
Fax: 86-28-8676-6599

China - Fuzhou
Tel: 86-591-750-3506
Fax: 86-591-750-3521

China - Hong Kong SAR
Tel: 852-2401-1200
Fax: 852-2401-3431

China - Shanghai
Tel: 86-21-6275-5700
Fax: 86-21-6275-5060

China - Shenzhen
Tel: 86-755-8290-1380
Fax: 86-755-8295-1393

China - Shunde
Tel: 86-757-2839-5507
Fax: 86-757-2839-5571

China - Qingdao
Tel: 86-532-502-7355
Fax: 86-532-502-7205

ASIA/PACIFIC

India - Bangalore
Tel: 91-80-2229-0061
Fax: 91-80-2229-0062

India - New Delhi
Tel: 91-11-5160-8632
Fax: 91-11-5160-8632

Japan - Kanagawa
Tel: 81-45-471-6166
Fax: 81-45-471-6122

Korea - Seoul
Tel: 82-2-554-7200
Fax: 82-2-558-5932 or
82-2-558-5934

Singapore
Tel: 65-6334-8870
Fax: 65-6334-8850

Taiwan - Kaohsiung
Tel: 886-7-536-4816
Fax: 886-7-536-4817

Taiwan - Taipei
Tel: 886-2-2500-6610
Fax: 886-2-2508-0102

Taiwan - Hsinchu
Tel: 886-3-572-9526
Fax: 886-3-572-6459

EUROPE

Austria - Weis
Tel: 43-7242-2244-399
Fax: 43-7242-2244-393

Denmark - Ballerup
Tel: 45-4420-9895
Fax: 45-4420-9910

France - Massy
Tel: 33-1-69-53-63-20
Fax: 33-1-69-30-90-79

Germany - Ismaning
Tel: 49-89-627-144-0
Fax: 49-89-627-144-44

Italy - Milan
Tel: 39-0331-742611
Fax: 39-0331-466781

Netherlands - Drunen
Tel: 31-416-690399
Fax: 31-416-690340

England - Berkshire
Tel: 44-118-921-5869
Fax: 44-118-921-5820

08/24/04