
64-Pin Flash Microcontrollers with XLP Technology

High-Performance RISC CPU

- C Compiler Optimized Architecture
- Only 49 Instructions
- Operating Speed:
 - DC – 20 MHz clock input @ 2.5V
 - DC – 16 MHz clock input @ 1.8V
 - DC – 200 ns instruction cycle
- Interrupt Capability with Automatic Context Saving
- 16-Level Deep Hardware Stack with Optional Overflow/Underflow Reset
- Direct, Indirect and Relative Addressing modes:
 - Two full 16-bit File Select Registers (FSRs)
 - FSRs can read program and data memory

Memory

- Up to 28 Kbytes Linear Program Memory Addressing
- Up to 1536 Bytes Linear Data Memory Addressing
- High-Endurance Flash Data Memory (HEF)
 - 128B of nonvolatile data storage
 - 100K erase/write cycles

Flexible Oscillator Structure

- 16 MHz Internal Oscillator Block:
 - Software selectable frequency range from 16 MHz to 31 kHz
- 31 kHz Low-Power Internal Oscillator
- External Oscillator Block with:
 - Four crystal/resonator modes up to 20 MHz
 - Three external clock modes up to 20 MHz
- Fail-Safe Clock Monitor
 - Allows safe shutdown if peripheral clock stops
- Two-Speed Oscillator Start-up
- Oscillator Start-up Timer (OST)

Special Microcontroller Features

- Operating Voltage Range:
 - 1.8V to 3.6V (PIC16LF1526/7)
 - 2.3V to 5.5V (PIC16F1526/7)
- Self-Programmable under Software Control
- Power-on Reset (POR)
- Power-up Timer (PWRT)
- Programmable Low-Power Brown-Out Reset (LPBOR)
- Extended Watch-Dog Timer (WDT):
 - Programmable period from 1 ms to 256s
- Programmable Code Protection
- In-Circuit Serial Programming™ (ICSP™) via two pins
- In-Circuit Debug (ICD) via Two Pins
- Enhanced Low-Voltage Programming (LVP)
- Power-Saving Sleep mode

**Extreme Low-Power Management
PIC16LF1526/7 with XLP**

- Sleep mode: 20 nA @ 1.8V, typical
- Watchdog Timer: 300 nA @ 1.8V, typical
- Secondary Oscillator: 600 nA @ 32 kHz, 1.8V, typical

Analog Features

- Analog-to-Digital Converter (ADC):
 - 10-bit resolution
 - 30 external channels
 - Two internal channels
 - Fixed Voltage Reference (FVR) channel
 - Temperature Indicator channel
 - Auto acquisition capability
 - Conversion available during Sleep
 - Dedicated ADC RC oscillator
 - Fixed Voltage Reference (FVR) as ADC positive reference
- Voltage Reference module:
 - Fixed Voltage Reference (FVR) with 1.024V, 2.048V and 4.096V output levels
 - Low-Power Sleep mode
 - Low-Power BOR (LPBOR)

Peripheral Features

- 53 I/O Pins and One Input-only Pin:
 - High current sink/source 25 mA/25 mA
 - Individually programmable weak pull-ups
 - Individually programmable interrupt-on-change (IOC) pins
- Timer0: 8-Bit Timer/Counter with 8-Bit Programmable Prescaler
- Enhanced Timer1, 3, 5:
 - 16-bit timer/counter with prescaler
 - External Gate Input mode
 - Low-power 32 kHz secondary oscillator driver
- Timer2, 4, 6, 8, 10: 8-Bit Timer/Counter with 8-Bit Period Register, Prescaler and Postscaler
- Ten Capture/Compare/PWM (CCP) modules:
 - 16-bit Capture, 200 ns (max. resolution)
 - 16-bit Compare, 200 ns (max. resolution)
 - 10-bit PWM, 20 kHz @ 10 bits (max. frequency)
- Two Master Synchronous Serial Ports (MSSPs) with SPI and I²C™ with:
 - 7-bit address masking
 - SMBus/PMBus™ compatibility
 - Auto-wake-up on start
- Two Enhanced Universal Synchronous Asynchronous Receiver Transmitters (EUSART):
 - RS-232, RS-485 and LIN compatible
 - Auto-Baud Detect

PIC16(L)F1526/7

PIC16(L)F151X/152X Family Types

| Device | Data Sheet Index | Program Memory Flash (words) | Data SRAM (bytes) | High-Endurance Flash (bytes) | I/O's ⁽²⁾ | ADC | | Timers (8/16-bit) | EUSART | MSSP (I ² C/SPI) | CCP | Debug ⁽¹⁾ | XLP |
|---------------|------------------|------------------------------|-------------------|------------------------------|----------------------|-------------|------------------|-------------------|--------|-----------------------------|-----|----------------------|-----|
| | | | | | | 10-bit (ch) | Advanced Control | | | | | | |
| PIC16(L)F1512 | (1) | 2048 | 128 | 128 | 25 | 17 | Y | 2/1 | 1 | 1 | 2 | I | Y |
| PIC16(L)F1513 | (1) | 4096 | 256 | 128 | 25 | 17 | Y | 2/1 | 1 | 1 | 2 | I | Y |
| PIC16(L)F1516 | (2) | 8192 | 512 | 128 | 25 | 17 | N | 2/1 | 1 | 1 | 2 | I | Y |
| PIC16(L)F1517 | (2) | 8192 | 512 | 128 | 36 | 28 | N | 2/1 | 1 | 1 | 2 | I | Y |
| PIC16(L)F1518 | (2) | 16384 | 1024 | 128 | 25 | 17 | N | 2/1 | 1 | 1 | 2 | I | Y |
| PIC16(L)F1519 | (2) | 16384 | 1024 | 128 | 36 | 28 | N | 2/1 | 1 | 1 | 2 | I | Y |
| PIC16(L)F1526 | (3) | 8192 | 768 | 128 | 54 | 30 | N | 6/3 | 2 | 2 | 10 | I | Y |
| PIC16(L)F1527 | (3) | 16384 | 1536 | 128 | 54 | 30 | N | 6/3 | 2 | 2 | 10 | I | Y |

Note 1: I - Debugging, Integrated on Chip; H - Debugging, available using Debug Header.

2: One pin is input-only.

Data Sheet Index: (Unshaded devices are described in this document.)

1: DS41624 [PIC16\(L\)F1512/13 Data Sheet, 28-Pin Flash, 8-bit Microcontrollers.](#)

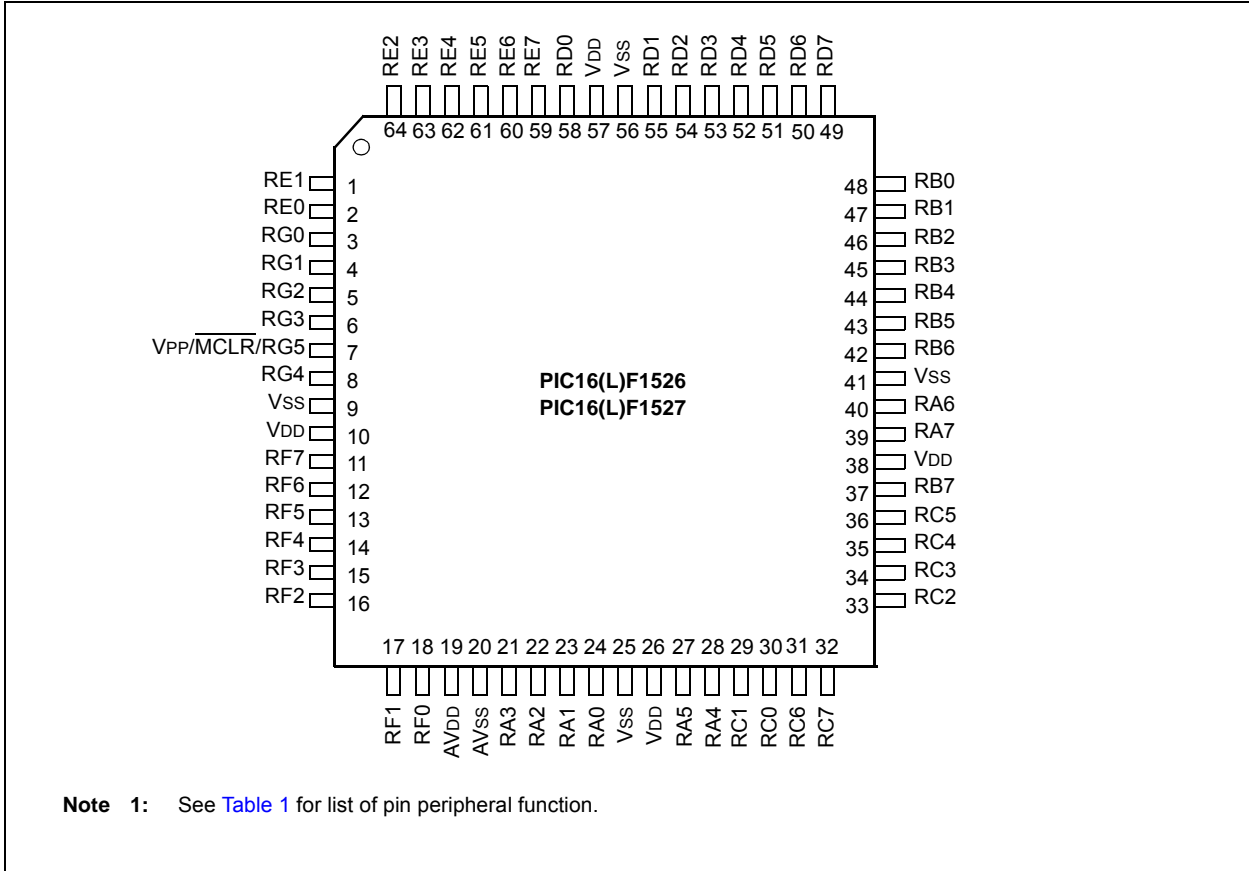
2: DS41452 [PIC16\(L\)F1516/7/8/9 Data Sheet, 28/40/44-Pin Flash, 8-bit MCUs.](#)

3: DS41458 [PIC16\(L\)F1526/7 Data Sheet, 64-Pin Flash, 8-bit MCUs.](#)

Note: For other small form-factor package availability and marking information, please visit <http://www.microchip.com/packaging> or contact your local sales office.

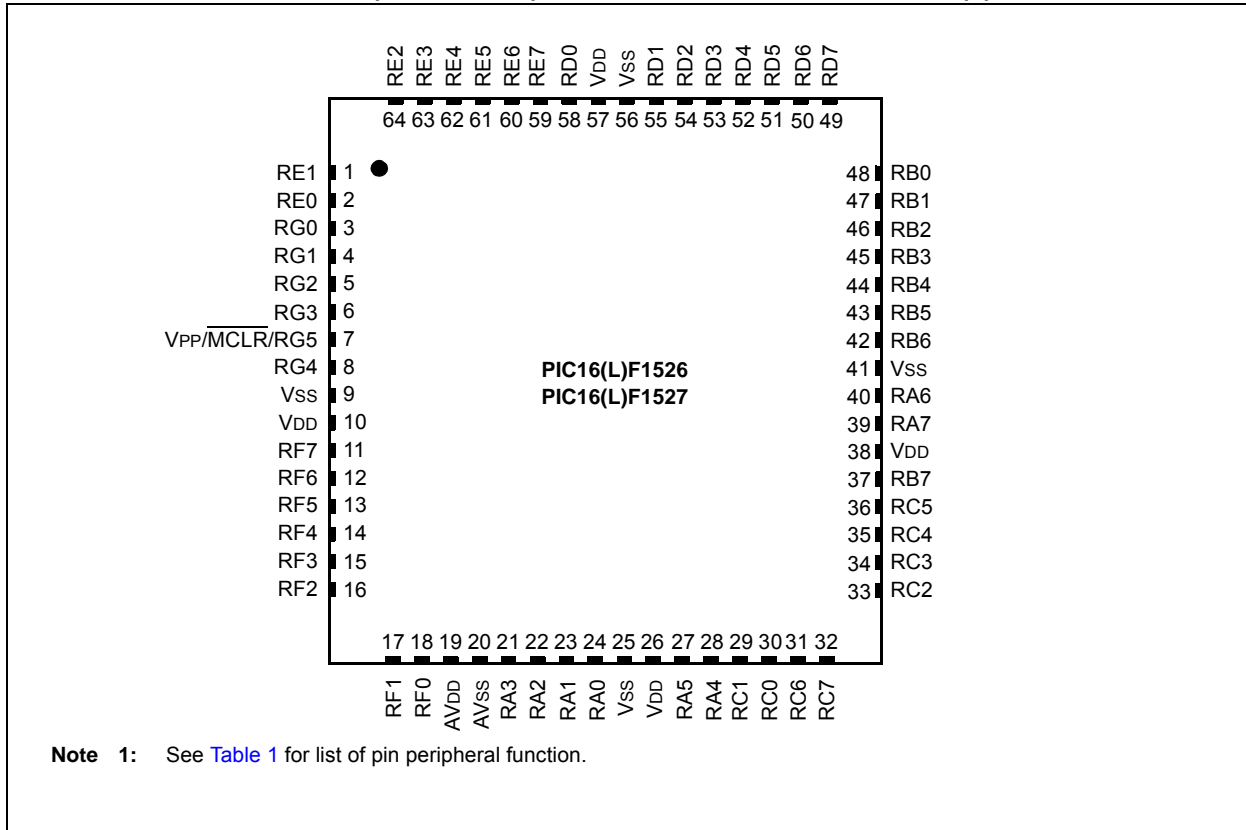
PIC16(L)F1526/7

FIGURE 1: 64-PIN TQFP (10MM X 10MM) PACKAGE DIAGRAM FOR PIC16(L)F1526/7



PIC16(L)F1526/7

FIGURE 2: 64-PIN QFN (9MM X 9MM) PACKAGE DIAGRAM FOR PIC16(L)F1526/7



PIC16(L)F1526/7

TABLE 1: 64-PIN DEVICE ALLOCATION TABLE (PIC16(L)F1526/7)

| I/O | 64-Pin TQFP, QFN | ADC | Timers | CCP | USART | SSP | Interrupt | Pull-up | Basic |
|-----|------------------|------|----------------------|-------|---------|------------|-------------|---------|----------------|
| RA0 | 24 | AN0 | — | — | — | — | — | — | — |
| RA1 | 23 | AN1 | — | — | — | — | — | — | — |
| RA2 | 22 | AN2 | — | — | — | — | — | — | — |
| RA3 | 21 | AN3 | — | — | — | — | — | — | VREF+ |
| RA4 | 28 | — | T0CKI | — | — | — | — | — | — |
| RA5 | 27 | AN4 | T3G | — | — | — | — | — | — |
| RA6 | 40 | — | — | — | — | — | — | — | OSC2/CLKOUT |
| RA7 | 39 | — | — | — | — | — | — | — | OSC1/CLKIN |
| RB0 | 48 | AN17 | — | — | — | — | INT/ IOC | Y | — |
| RB1 | 47 | AN18 | — | — | — | — | IOC | Y | — |
| RB2 | 46 | AN19 | — | — | — | — | IOC | Y | — |
| RB3 | 45 | AN20 | — | — | — | — | IOC | Y | — |
| RB4 | 44 | AN21 | T3CKI ⁽¹⁾ | — | — | — | IOC | Y | — |
| RB5 | 43 | AN22 | T1G/T3CKI | — | — | — | IOC | Y | — |
| RB6 | 42 | — | — | — | — | — | IOC | Y | ICSPCLK/ICDCLK |
| RB7 | 37 | — | — | — | — | — | IOC | Y | ICSPDAT/ICDDAT |
| RC0 | 30 | — | SOSCO/T1CKI | — | — | — | — | — | — |
| RC1 | 29 | — | SOSCI | CCP2 | — | — | — | — | — |
| RC2 | 33 | — | — | CCP1 | — | — | — | — | — |
| RC3 | 34 | — | — | — | — | SCK1/SCL1 | — | — | — |
| RC4 | 35 | — | — | — | — | SDI1/SDA1 | — | — | — |
| RC5 | 36 | — | — | — | — | SDO1 | — | — | — |
| RC6 | 31 | — | — | — | TX1/CK1 | — | — | — | — |
| RC7 | 32 | — | — | — | RX1/DT1 | — | — | — | — |
| RD0 | 58 | AN23 | — | — | — | — | — | Y | — |
| RD1 | 55 | AN24 | T5CKI | — | — | — | — | Y | — |
| RD2 | 54 | AN25 | — | — | — | — | — | Y | — |
| RD3 | 53 | AN26 | — | — | — | — | — | Y | — |
| RD4 | 52 | — | — | — | — | SDO2 | — | Y | — |
| RD5 | 51 | — | — | — | — | SDI2, SDA2 | — | Y | — |
| RD6 | 50 | — | — | — | — | SCK2, SCL2 | — | Y | — |
| RD7 | 49 | — | — | — | — | SS2 | — | Y | — |
| RE0 | 2 | AN27 | — | — | — | — | — | Y | — |
| RE1 | 1 | AN28 | — | — | — | — | — | Y | — |
| RE2 | 64 | AN29 | — | CCP10 | — | — | — | Y | — |
| RE3 | 63 | — | — | CCP9 | — | — | — | Y | — |
| RE4 | 62 | — | — | CCP8 | — | — | — | Y | — |
| RE5 | 61 | — | — | CCP7 | — | — | — | Y | — |
| RE6 | 60 | — | — | CCP6 | — | — | — | Y | — |

Note 1: Alternate pin function selected with the APFCON (Register 12-1) register.

2: Weak pull-up is always enabled when MCLR is enabled, otherwise the pull-up is under user control.

PIC16(L)F1526/7

TABLE 1: 64-PIN DEVICE ALLOCATION TABLE (PIC16(L)F1526/7) (CONTINUED)

| I/O | 64-Pin TQFP, QFN | ADC | Timers | CCP | USART | SSP | Interrupt | Pull-up | Basic |
|------|-------------------|------|--------|---------------------|---------|-----|-----------|------------------|----------|
| RE7 | 59 | — | — | CCP2 ⁽¹⁾ | — | — | — | Y | — |
| RF0 | 18 | AN16 | — | — | — | — | — | — | VCAP |
| RF1 | 17 | AN6 | — | — | — | — | — | — | — |
| RF2 | 16 | AN7 | — | — | — | — | — | — | — |
| RF3 | 15 | AN8 | — | — | — | — | — | — | — |
| RF4 | 14 | AN9 | — | — | — | — | — | — | — |
| RF5 | 13 | AN10 | — | — | — | — | — | — | — |
| RF6 | 12 | AN11 | — | — | — | — | — | — | — |
| RF7 | 11 | AN5 | — | — | — | SS1 | — | — | — |
| RG0 | 3 | — | — | CCP3 | — | — | — | — | — |
| RG1 | 4 | AN15 | — | — | TX2/CK2 | — | — | — | — |
| RG2 | 5 | AN14 | — | — | RX2/DT2 | — | — | — | — |
| RG3 | 6 | AN13 | — | CCP4 | — | — | — | — | — |
| RG4 | 8 | AN12 | T5G | CCP5 | — | — | — | — | — |
| RG5 | 7 | — | — | — | — | — | — | Y ⁽²⁾ | MCLR/VPP |
| VDD | 10, 26, 38, 57 | — | — | — | — | — | — | — | VDD |
| VSS | 9, 25, 41, 56 | — | — | — | — | — | — | — | VSS |
| AVDD | 19 | — | — | — | — | — | — | — | AVDD |
| AVSS | 20 | — | — | — | — | — | — | — | AVSS |

Note 1: Alternate pin function selected with the APFCON (Register 12-1) register.

2: Weak pull-up is always enabled when MCLR is enabled, otherwise the pull-up is under user control.

Table of Contents

| | | |
|------|--|-----|
| 1.0 | Device Overview | 9 |
| 2.0 | Enhanced Mid-Range CPU | 15 |
| 3.0 | Memory Organization | 17 |
| 4.0 | Device Configuration | 42 |
| 5.0 | Oscillator Module (With Fail-Safe Clock Monitor)..... | 48 |
| 6.0 | Resets | 63 |
| 7.0 | Interrupts | 71 |
| 8.0 | Power-Down Mode (Sleep) | 86 |
| 9.0 | Low Dropout (LDO) Voltage Regulator | 90 |
| 10.0 | Watchdog Timer (WDT) | 91 |
| 11.0 | Flash Program Memory Control | 95 |
| 12.0 | I/O Ports | 111 |
| 13.0 | Interrupt-on-Change | 135 |
| 14.0 | Fixed Voltage Reference (FVR) | 139 |
| 15.0 | Temperature Indicator Module | 141 |
| 16.0 | Analog-to-Digital Converter (ADC) Module | 143 |
| 17.0 | Timer0 Module | 156 |
| 18.0 | Timer1/3/5 Modules..... | 159 |
| 19.0 | Timer2/4/6/8/10 Modules..... | 171 |
| 20.0 | Capture/Compare/PWM Module | 175 |
| 21.0 | Master Synchronous Serial Port (MSSP) Module | 193 |
| 22.0 | Enhanced Universal Synchronous Asynchronous Receiver Transmitter (EUSART)..... | 248 |
| 23.0 | In-Circuit Serial Programming™ (ICSP™) | 278 |
| 24.0 | Instruction Set Summary | 280 |
| 25.0 | Electrical Specifications..... | 294 |
| 26.0 | DC and AC Characteristics Graphs and Tables..... | 324 |
| 27.0 | Development Support..... | 358 |
| 28.0 | Packaging Information..... | 362 |
| | Appendix A: Revision History..... | 369 |
| | The Microchip Web Site..... | 371 |
| | Customer Change Notification Service | 371 |
| | Customer Support..... | 371 |
| | Product Identification System | 370 |

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@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)

When contacting a sales office, 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 to receive the most current information on all of our products.

1.0 DEVICE OVERVIEW

The PIC16(L)F1526/7 are described within this data sheet. They are available in 64-pin packages. [Figure 1-1](#) shows a block diagram of the PIC16(L)F1526/7 devices. [Table 1-2](#) shows the pinout descriptions.

Reference [Table 1-1](#) for peripherals available per device.

TABLE 1-1: DEVICE PERIPHERAL SUMMARY

| Peripheral | | PIC16F1526 PIC16LF1526 | PIC16F1527 PIC16LF1527 |
|---------------------------------|---------------------|---------------------------|---------------------------|
| ADC | | • | • |
| EUSART | | • | • |
| Fixed Voltage Reference (FVR) | | • | • |
| Temperature Indicator | | • | • |
| Capture/Compare/PWM Modules | | | |
| | CCP1 | • | • |
| | CCP2 | • | • |
| | CCP3 | • | • |
| | CCP4 | • | • |
| | CCP5 | • | • |
| | CCP6 | • | • |
| | CCP7 | • | • |
| | CCP8 | • | • |
| | CCP9 | • | • |
| | CCP10 | • | • |
| EUSARTs | | | |
| | EUSART1 | • | • |
| | EUSART2 | • | • |
| Master Synchronous Serial Ports | | | |
| | MSSP1 | • | • |
| | MSSP2 | • | • |
| Timers | | | |
| | Timer0 | • | • |
| | Timer1/3/5 | • | • |
| | Timer2/4/6 /8/10 | • | • |

PIC16(L)F1526/7

FIGURE 1-1: PIC16(L)F1526/7 BLOCK DIAGRAM

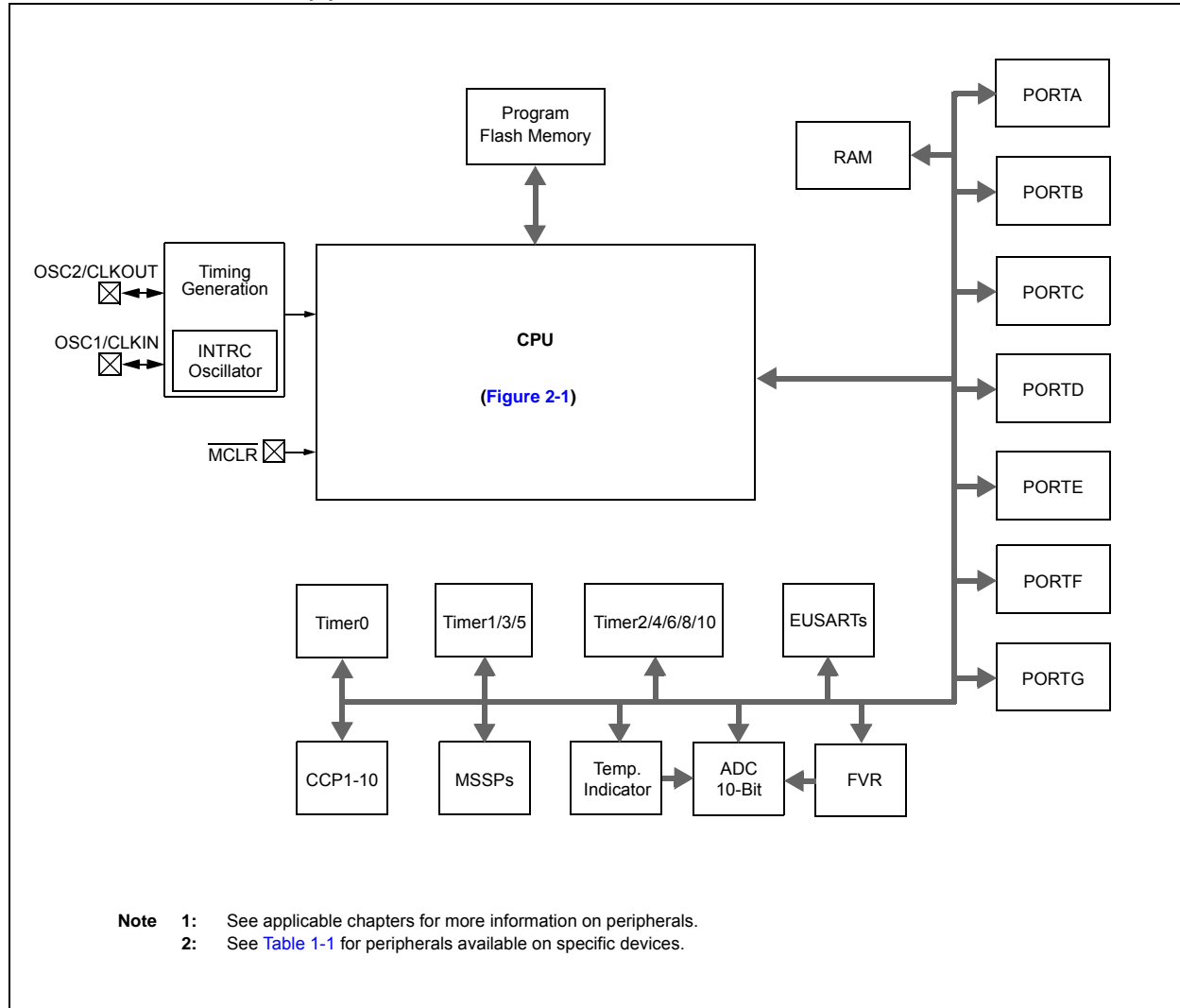


TABLE 1-2: PIC16(L)F1526/7 PINOUT DESCRIPTION

| Name | Function | Input Type | Output Type | Description |
|-------------------------------|----------|------------|-------------|---------------------------------------|
| RA0/AN0 | RA0 | TTL | CMOS | General purpose I/O. |
| | AN0 | AN | — | ADC Channel 0 input. |
| RA1/AN1 | RA1 | TTL | CMOS | General purpose I/O. |
| | AN1 | AN | — | ADC Channel 1 input. |
| RA2/AN2 | RA2 | TTL | CMOS | General purpose I/O. |
| | AN2 | AN | — | ADC Channel 2 input. |
| RA3/AN3/VREF+ | RA3 | TTL | CMOS | General purpose I/O. |
| | AN3 | AN | — | ADC Channel 3 input. |
| | VREF+ | AN | — | ADC Positive Voltage Reference input. |
| RA4/T0CKI | RA4 | TTL | CMOS | General purpose I/O. |
| | T0CKI | ST | — | Timer0 clock input. |
| RA5/AN4/T3G | RA5 | TTL | CMOS | General purpose I/O. |
| | AN4 | AN | — | ADC Channel 4 input. |
| | T3G | ST | — | Timer3 gate input. |
| RA6/OSC2/CLKOUT | RA6 | TTL | CMOS | General purpose I/O. |
| | OSC2 | — | XTAL | Crystal/Resonator (LP, XT, HS modes). |
| | CLKOUT | — | CMOS | Fosc/4 output. |
| RA7/OSC1/CLKIN | RA7 | TTL | CMOS | General purpose I/O. |
| | OSC1 | XTAL | — | Crystal/Resonator (LP, XT, HS modes). |
| | CLKIN | ST | — | External clock input (EC mode). |
| RB0/AN17/INT | RB0 | TTL | CMOS | General purpose I/O with IOC and WPU. |
| | AN17 | AN | — | ADC Channel 17 input. |
| | INT | ST | — | External interrupt. |
| RB1/AN18 | RB1 | TTL | CMOS | General purpose I/O with IOC and WPU. |
| | AN18 | AN | — | ADC Channel 18 input. |
| RB2/AN19 | RB2 | TTL | CMOS | General purpose I/O with IOC and WPU. |
| | AN19 | AN | — | ADC Channel 19 input. |
| RB3/AN20 | RB3 | TTL | CMOS | General purpose I/O with IOC and WPU. |
| | AN20 | AN | — | ADC Channel 20 input. |
| RB4/AN21/T3CKI ⁽¹⁾ | RB4 | TTL | CMOS | General purpose I/O with IOC and WPU. |
| | AN21 | AN | — | ADC Channel 21 input. |
| | T3CKI | ST | — | Timer3 clock input. |
| RB5/AN22/T1G/T3CKI | RB5 | TTL | CMOS | General purpose I/O with IOC and WPU. |
| | AN22 | AN | — | ADC Channel 22 input. |
| | T1G | ST | — | Timer1 gate input. |
| | T3CKI | ST | — | Timer3 clock input. |
| RB6/ICSPCLK/ICDCLK | RB6 | TTL | CMOS | General purpose I/O with IOC and WPU. |
| | ICSPCLK | ST | — | Serial Programming Clock. |
| | ICDCLK | ST | — | In-Circuit Debug Clock. |
| RB7/ICSPDAT/ICDDAT | RB7 | TTL | CMOS | General purpose I/O with IOC and WPU. |
| | ICSPDAT | ST | CMOS | ICSP™ Data I/O. |
| | ICDDAT | ST | CMOS | In-Circuit Data I/O. |

Legend: AN = Analog input or output CMOS = CMOS compatible input or output OD = Open Drain
TTL = TTL compatible input ST = Schmitt Trigger input with CMOS levels I²C = Schmitt Trigger input with I²C levels
HV = High Voltage XTAL = Crystal

Note 1: Alternate pin function selected with the APFCON (Register 12-1) register.
2: RC3, RC4, RD5 and RD6 read the I²C ST input when I²C mode is enabled.

PIC16(L)F1526/7

TABLE 1-2: PIC16(L)F1526/7 PINOUT DESCRIPTION (CONTINUED)

| Name | Function | Input Type | Output Type | Description |
|------------------------------|-----------------|------------------|-------------|-------------------------------------|
| RC0/SOSCO/T1CKI | RC0 | ST | CMOS | General purpose I/O. |
| | SOSCO | XTAL | XTAL | Timer1/3/5 oscillator connection. |
| | T1CKI | ST | — | Timer1/3/5 clock input. |
| RC1/SOSCI/CCP2 | RC1 | ST | CMOS | General purpose I/O. |
| | SOSCI | XTAL | XTAL | Timer1/3/5 oscillator connection. |
| | CCP2 | ST | CMOS | Capture/Compare/PWM2. |
| RC2/CCP1 | RC2 | ST | CMOS | General purpose I/O. |
| | CCP1 | ST | CMOS | Capture/Compare/PWM1. |
| RC3/SCK1/SCL1 ⁽²⁾ | RC3 | ST | CMOS | General purpose I/O. |
| | SCK1 | ST | CMOS | SPI clock. |
| | SCL1 | I ² C | OD | I ² C clock. |
| RC4/SDI1/SDA1 ⁽²⁾ | RC4 | ST | CMOS | General purpose I/O. |
| | SDI1 | ST | — | SPI data input. |
| | SDA1 | I ² C | OD | I ² C data input/output. |
| RC5/SDO1 | RC5 | ST | CMOS | General purpose I/O. |
| | SDO1 | — | CMOS | SPI data output. |
| RC6/TX1/CK1 | RC6 | ST | CMOS | General purpose I/O. |
| | TX1 | — | CMOS | USART1 asynchronous transmit. |
| | CK1 | ST | CMOS | USART1 synchronous clock. |
| RC7/RX1/DT1 | RC7 | ST | CMOS | General purpose I/O. |
| | RX1 | ST | — | USART1 asynchronous input. |
| | DT1 | ST | CMOS | USART1 synchronous data. |
| RDO/AN23 | RD0 | ST | CMOS | General purpose I/O with WPU. |
| | AN23 | AN | — | ADC Channel 23 input. |
| RD1/AN24/T5CKI | RD1 | ST | CMOS | General purpose I/O with WPU. |
| | AN24 | AN | — | ADC Channel 24 input. |
| | T5CKI | ST | — | Timer5 clock input. |
| RD2/AN25 | RD2 | ST | CMOS | General purpose I/O with WPU. |
| | AN25 | AN | — | ADC Channel 25 input. |
| RD3/AN26 | RD3 | ST | CMOS | General purpose I/O with WPU. |
| | AN26 | AN | — | ADC Channel 26 input. |
| RD4/SDO2 | RD4 | ST | CMOS | General purpose I/O with WPU. |
| | SDO2 | — | CMOS | SPI data output. |
| RD5/SDI2/SDA2 ⁽²⁾ | RD5 | ST | CMOS | General purpose I/O with WPU. |
| | SDI2 | ST | — | SPI data input. |
| | SDA2 | I ² C | OD | I ² C data input/output. |
| RD6/SCK2/SCL2 ⁽²⁾ | RD6 | ST | CMOS | General purpose I/O with WPU. |
| | SCK2 | ST | CMOS | SPI clock. |
| | SCL2 | I ² C | OD | I ² C clock. |
| RD7/SS ² | RD7 | ST | CMOS | General purpose I/O with WPU. |
| | SS ² | ST | — | Slave Select input. |
| RE0/AN27 | RE0 | ST | CMOS | General purpose I/O with WPU. |
| | AN27 | AN | — | ADC Channel 27 input. |

Legend: AN = Analog input or output CMOS = CMOS compatible input or output OD = Open Drain
TTL = TTL compatible input ST = Schmitt Trigger input with CMOS levels I²C = Schmitt Trigger input with I²C levels
HV = High Voltage XTAL = Crystal

Note 1: Alternate pin function selected with the APFCON (Register 12-1) register.
Note 2: RC3, RC4, RD5 and RD6 read the I²C ST input when I²C mode is enabled.

TABLE 1-2: PIC16(L)F1526/7 PINOUT DESCRIPTION (CONTINUED)

| Name | Function | Input Type | Output Type | Description |
|---------------------------|------------------|------------|-------------|---|
| RE1/AN28 | RE1 | ST | CMOS | General purpose I/O with WPU. |
| | AN28 | AN | — | ADC Channel 28 input. |
| RE2/AN29/CCP10 | RE2 | ST | CMOS | General purpose I/O with WPU. |
| | AN29 | AN | — | ADC Channel 29 input. |
| | CCP10 | ST | CMOS | Capture/Compare/PWM10. |
| RE3/CCP9 | RE3 | ST | CMOS | General purpose I/O with WPU. |
| | CCP9 | ST | CMOS | Capture/Compare/PWM9. |
| RE4/CCP8 | RE4 | ST | CMOS | General purpose I/O with WPU. |
| | CCP8 | ST | CMOS | Capture/Compare/PWM8. |
| RE5/CCP7 | RE5 | ST | CMOS | General purpose I/O with WPU. |
| | CCP7 | ST | CMOS | Capture/Compare/PWM7. |
| RE6/CCP6 | RE6 | ST | CMOS | General purpose I/O with WPU. |
| | CCP6 | ST | CMOS | Capture/Compare/PWM6. |
| RE7/CCP2 ⁽¹⁾ | RE7 | ST | CMOS | General purpose I/O with WPU. |
| | CCP2 | ST | CMOS | Capture/Compare/PWM2. |
| RF0/AN16/V _{CAP} | RF0 | ST | CMOS | General purpose I/O. |
| | AN16 | AN | — | ADC Channel 16 input. |
| | V _{CAP} | Power | Power | Filter capacitor for Voltage Regulator. |
| RF1/AN6 | RF1 | ST | CMOS | General purpose I/O. |
| | AN6 | AN | — | ADC Channel 6 input. |
| RF2/AN7 | RF2 | ST | CMOS | General purpose I/O. |
| | AN7 | AN | — | ADC Channel 7 input. |
| RF3/AN8 | RF3 | ST | CMOS | General purpose I/O. |
| | AN8 | AN | — | ADC Channel 8 input. |
| RF4/AN9 | RF4 | ST | CMOS | General purpose I/O. |
| | AN9 | AN | — | ADC Channel 9 input. |
| RF5/AN10 | RF5 | ST | CMOS | General purpose I/O. |
| | AN10 | AN | — | ADC Channel 10 input. |
| RF6/AN11 | RF6 | ST | CMOS | General purpose I/O. |
| | AN11 | AN | — | ADC Channel 11 input. |
| RF7/AN5/SS1 | RF7 | ST | CMOS | General purpose I/O. |
| | AN5 | AN | — | ADC Channel 5 input. |
| | SS1 | ST | — | Slave Select input. |
| RG0/CCP3 | RG0 | ST | CMOS | General purpose I/O. |
| | CCP3 | ST | CMOS | Capture/Compare/PWM3. |
| RG1/AN15/TX2/CK2 | RG1 | ST | CMOS | General purpose I/O. |
| | AN15 | AN | — | ADC Channel 15 input. |
| | TX2 | — | CMOS | USART2 asynchronous transmit. |
| | CK2 | ST | CMOS | USART2 synchronous clock. |
| RG2/AN14/RX2/DT2 | RG2 | ST | CMOS | General purpose I/O. |
| | AN14 | AN | — | ADC Channel 14 input. |
| | RX2 | ST | — | USART2 asynchronous input. |
| | DT2 | ST | CMOS | USART2 synchronous data. |

Legend: AN = Analog input or output CMOS = CMOS compatible input or output OD = Open Drain
TTL = TTL compatible input ST = Schmitt Trigger input with CMOS levels I²C = Schmitt Trigger input with I²C levels
HV = High Voltage XTAL = Crystal

Note 1: Alternate pin function selected with the APFCON (Register 12-1) register.
2: RC3, RC4, RD5 and RD6 read the I²C ST input when I²C mode is enabled.

PIC16(L)F1526/7

TABLE 1-2: PIC16(L)F1526/7 PINOUT DESCRIPTION (CONTINUED)

| Name | Function | Input Type | Output Type | Description |
|-------------------|----------|------------|-------------|-------------------------------------|
| RG3/AN13/CCP4 | RG3 | ST | CMOS | General purpose I/O. |
| | AN13 | AN | — | ADC Channel 13 input. |
| | CCP4 | ST | CMOS | Capture/Compare/PWM4. |
| RG4/AN12/T5G/CCP5 | RG4 | ST | — | General purpose input. |
| | AN12 | AN | — | ADC Channel 12 input. |
| | T5G | ST | — | Timer5 gate input. |
| | CCP5 | ST | CMOS | Capture/Compare/PWM5. |
| RG5/MCLR/VPP | RG5 | ST | — | General purpose input with WPU. |
| | MCLR | ST | — | Master Clear with internal pull-up. |
| | VPP | HV | — | Programming voltage. |
| AVDD | AVDD | Power | — | Analog positive supply. |
| AVSS | AVSS | Power | — | Analog ground reference. |
| VDD | VDD | Power | — | Positive supply. |
| VSS | VSS | Power | — | Ground reference. |

Legend: AN = Analog input or output CMOS = CMOS compatible input or output OD = Open Drain
 TTL = TTL compatible input ST = Schmitt Trigger input with CMOS levels I²C = Schmitt Trigger input with I²C levels
 HV = High Voltage XTAL = Crystal

- Note** 1: Alternate pin function selected with the APFCON ([Register 12-1](#)) register.
 2: RC3, RC4, RD5 and RD6 read the I²C ST input when I²C mode is enabled.

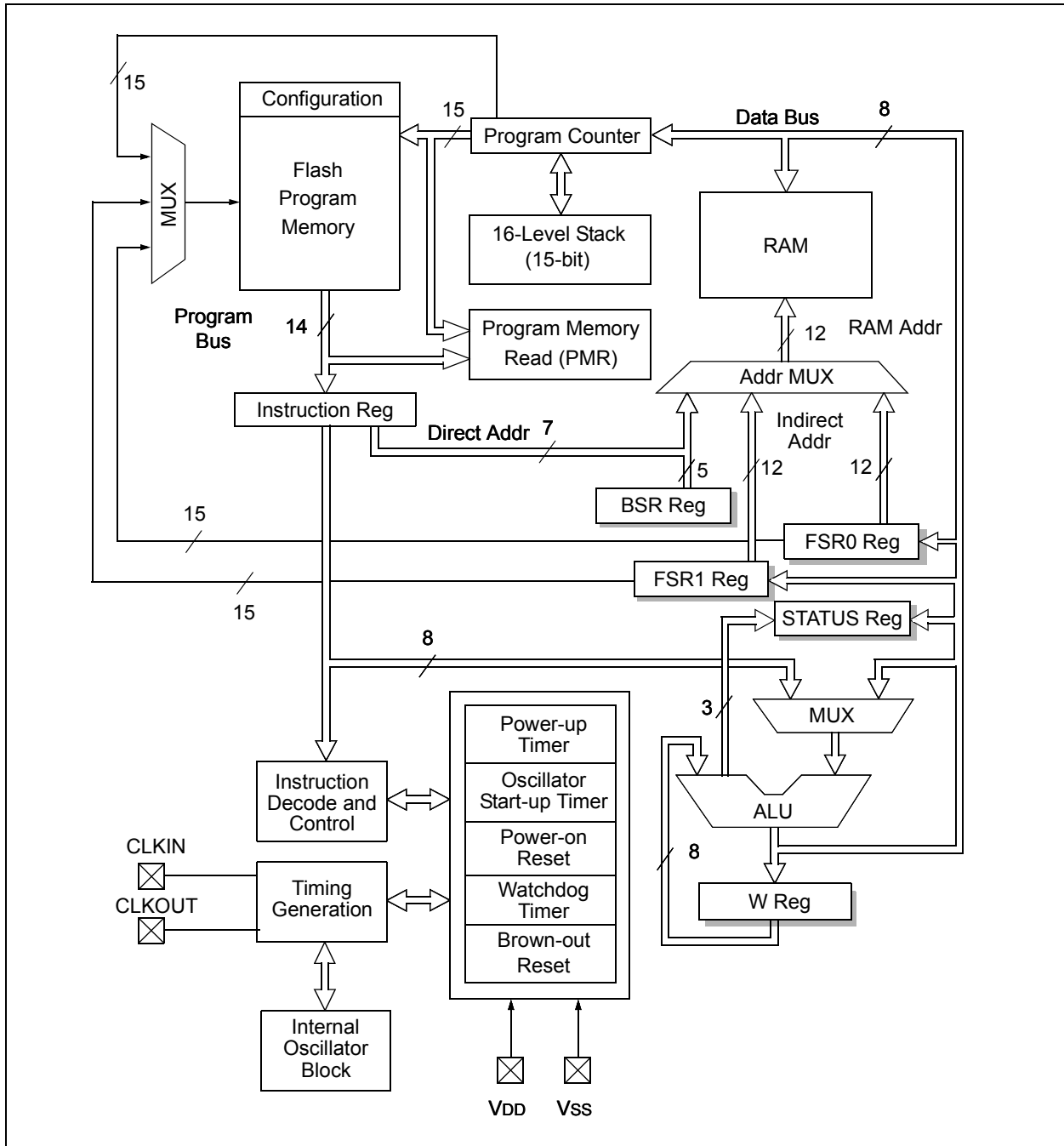
2.0 ENHANCED MID-RANGE CPU

This family of devices contain an enhanced mid-range 8-bit CPU core. The CPU has 49 instructions. Interrupt capability includes automatic context saving. The hardware stack is 16 levels deep and has Overflow and Underflow Reset capability. Direct, Indirect, and

Relative Addressing modes are available. Two File Select Registers (FSRs) provide the ability to read program and data memory.

- Automatic Interrupt Context Saving
- 16-level Stack with Overflow and Underflow
- File Select Registers
- Instruction Set

FIGURE 2-1: CORE BLOCK DIAGRAM



PIC16(L)F1526/7

2.1 Automatic Interrupt Context Saving

During interrupts, certain registers are automatically saved in shadow registers and restored when returning from the interrupt. This saves stack space and user code. See [Section 7.5 “Automatic Context Saving”](#), for more information.

2.2 16-level Stack with Overflow and Underflow

These devices have an external stack memory 15 bits wide and 16 words deep. A Stack Overflow or Underflow will set the appropriate bit (STKOVF or STKUNF) in the PCON register, and if enabled will cause a software Reset. See [Section 3.7 “Stack”](#) for more details.

2.3 File Select Registers

There are two 16-bit File Select Registers (FSR). FSRs can access all file registers and program memory, which allows one Data Pointer for all memory. When an FSR points to program memory, there is one additional instruction cycle in instructions using INDF to allow the data to be fetched. General purpose memory can now also be addressed linearly, providing the ability to access contiguous data larger than 80 bytes. There are also new instructions to support the FSRs. See [Section 3.8 “Indirect Addressing”](#) for more details.

2.4 Instruction Set

There are 49 instructions for the enhanced mid-range CPU to support the features of the CPU. See [Section 24.0 “Instruction Set Summary”](#) for more details.

3.0 MEMORY ORGANIZATION

These devices contain the following types of memory:

- Program Memory
 - Configuration Words
 - Device ID
 - User ID
 - Flash Program Memory
- Data Memory
 - Core Registers
 - Special Function Registers
 - General Purpose RAM
 - Common RAM

The following features are associated with access and control of program memory and data memory:

- PCL and PCLATH
- Stack
- Indirect Addressing

3.1 Program Memory Organization

The enhanced mid-range core has a 15-bit program counter capable of addressing a 32K x 14 program memory space. [Table 3-1](#) shows the memory sizes implemented for the PIC16(L)F1526/7 family. Accessing a location above these boundaries will cause a

wrap-around within the implemented memory space. The Reset vector is at 0000h and the interrupt vector is at 0004h (see [Figure 3-1](#) and [Figure 3-2](#)).

3.2 High Endurance Flash

This device has a 128-byte section of high-endurance Program Flash Memory (PFM) in lieu of data EEPROM. This area is especially well suited for nonvolatile data storage that is expected to be updated frequently over the life of the end product. See [Section 11.2 “Flash Program Memory Overview”](#) for more information on writing data to PFM. Refer to section [Section 3.2.1.2 “Indirect Read with FSR”](#) for more information about using the FSR registers to read byte data stored in PFM.

TABLE 3-1: DEVICE SIZES AND ADDRESSES

| Device | Program Memory Space (Words) | Last Program Memory Address | High-Endurance Flash Memory Address Range ⁽¹⁾ |
|---------------------------|------------------------------|-----------------------------|--|
| PIC16F1526 PIC16LF1526 | 8,192 | 1FFFh | 1F80h-1FFFh |
| PIC16F1527 PIC16LF1527 | 16,384 | 3FFFh | 3F80h-3FFFh |

Note 1: High-endurance Flash applies to the low byte of each address in the range.

PIC16(L)F1526/7

FIGURE 3-1: PROGRAM MEMORY MAP AND STACK FOR PIC16(L)F1526

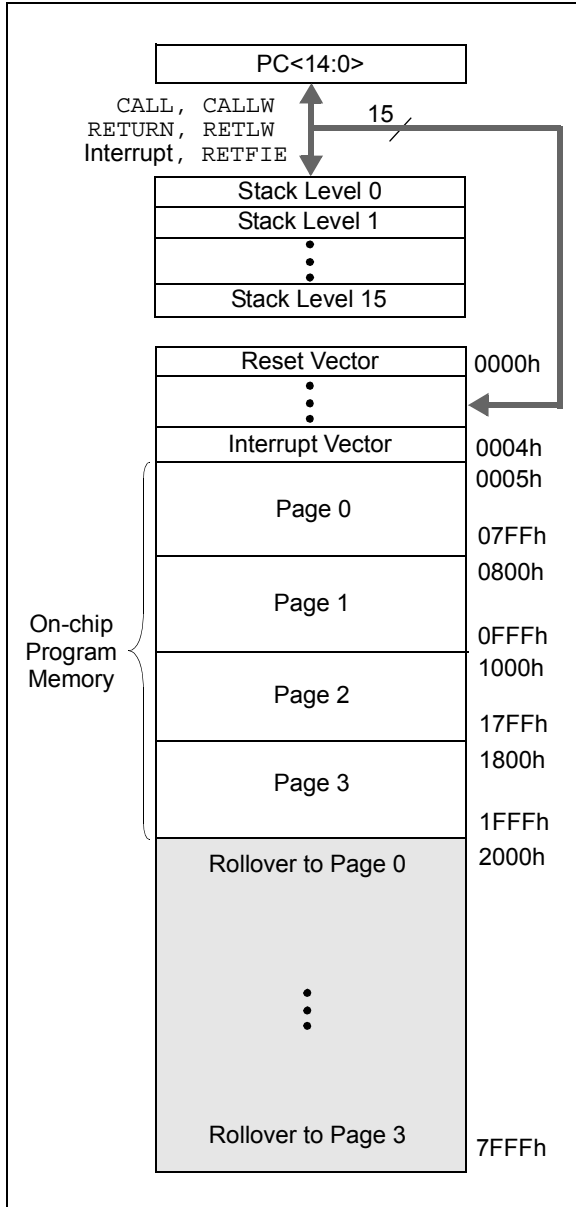
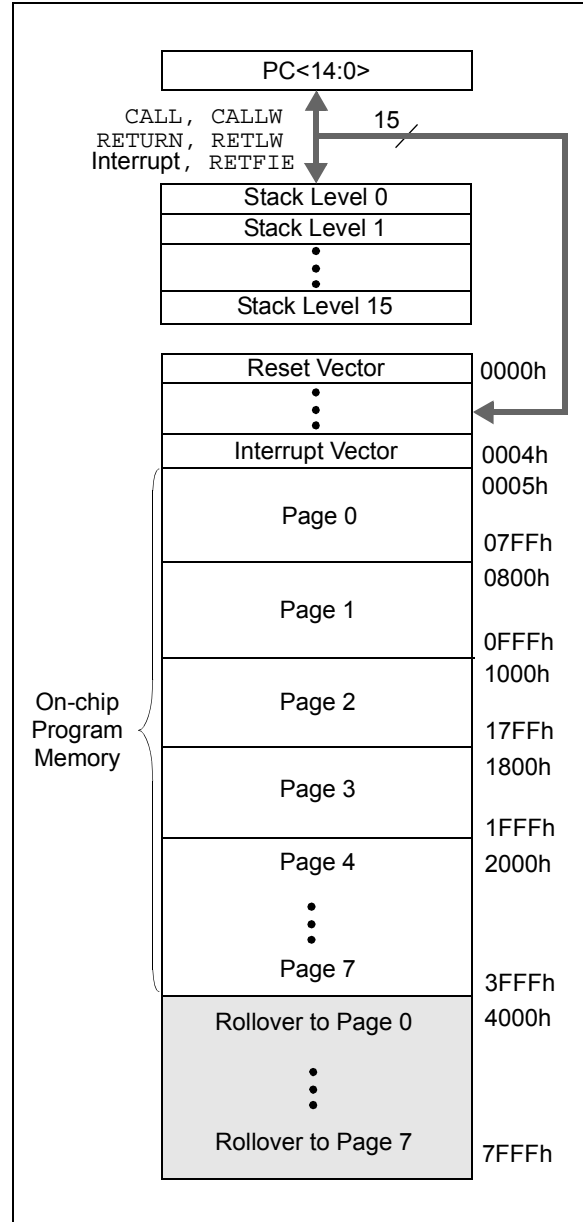


FIGURE 3-2: PROGRAM MEMORY MAP AND STACK FOR PIC16(L)F1527



3.2.1 READING PROGRAM MEMORY AS DATA

There are two methods of accessing constants in program memory. The first method is to use tables of RETLW instructions. The second method is to set an FSR to point to the program memory.

3.2.1.1 RETLW Instruction

The RETLW instruction can be used to provide access to tables of constants. The recommended way to create such a table is shown in [Example 3-1](#).

EXAMPLE 3-1: RETLW INSTRUCTION

```
constants
    BRW          ;Add Index in W to
                ;program counter to
                ;select data
    RETLW DATA0 ;Index0 data
    RETLW DATA1 ;Index1 data
    RETLW DATA2
    RETLW DATA3

my_function
    ;... LOTS OF CODE...
    MOVLW      DATA_INDEX
    CALL constants
    ;... THE CONSTANT IS IN W
```

The BRW instruction makes this type of table very simple to implement. If your code must remain portable with previous generations of microcontrollers, then the BRW instruction is not available so the older table read method must be used.

3.2.1.2 Indirect Read with FSR

The program memory can be accessed as data by setting bit 7 of the FSRxH register and reading the matching INDFx register. The MOVIW instruction will place the lower 8 bits of the addressed word in the W register. Writes to the program memory cannot be performed via the INDF registers. Instructions that access the program memory via the FSR require one extra instruction cycle to complete. [Example 3-2](#) demonstrates accessing the program memory via an FSR.

The high directive will set bit<7> if a label points to a location in program memory.

EXAMPLE 3-2: ACCESSING PROGRAM MEMORY VIA FSR

```
constants
    DW DATA0          ;First constant
    DW DATA1          ;Second constant
    DW DATA2
    DW DATA3

my_function
    ;... LOTS OF CODE...
    MOVLW DATA_INDEX
    ADDLW LOW constants
    MOVWF FSR1L
    MOVLW HIGH constants ;Msb is set
                                automatically
    MOVWF FSR1H
    BTFSC STATUS,C          ;carry from
                                ADDLW?
    INCF FSR1H,f          ;yes
    MOVIW 0[FSR1]
    ;THE PROGRAM MEMORY IS IN W
```

PIC16(L)F1526/7

3.3 Data Memory Organization

The data memory is partitioned in 32 memory banks with 128 bytes in a bank. Each bank consists of (Figure 3-3):

- 12 core registers
- 20 Special Function Registers (SFR)
- Up to 80 bytes of General Purpose RAM (GPR)
- 16 bytes of common RAM

The active bank is selected by writing the bank number into the Bank Select Register (BSR). Unimplemented memory will read as '0'. All data memory can be accessed either directly (via instructions that use the file registers) or indirectly via the two File Select Registers (FSR). See [Section 3.8 “Indirect Addressing”](#) for more information.

Data memory uses a 12-bit address. The upper seven bits of the address define the Bank address and the lower five bits select the registers/RAM in that bank.

3.3.1 CORE REGISTERS

The core registers contain the registers that directly affect the basic operation. The core registers occupy the first 12 addresses of every data memory bank (addresses x00h/x08h through x0Bh/x8Bh). These registers are listed below in [Table 3-2](#). For detailed information, see [Table 3-4](#).

TABLE 3-2: CORE REGISTERS

| Addresses | BANKx |
|--------------|--------|
| x00h or x80h | INDF0 |
| x01h or x81h | INDF1 |
| x02h or x82h | PCL |
| x03h or x83h | STATUS |
| x04h or x84h | FSR0L |
| x05h or x85h | FSR0H |
| x06h or x86h | FSR1L |
| x07h or x87h | FSR1H |
| x08h or x88h | BSR |
| x09h or x89h | WREG |
| x0Ah or x8Ah | PCLATH |
| x0Bh or x8Bh | INTCON |

3.3.1.1 STATUS Register

The STATUS register, shown in [Register 3-1](#), contains:

- the arithmetic status of the ALU
- the Reset status

The STATUS register can be the destination for any instruction, like any other register. If the STATUS register is the destination for an instruction that affects the Z, DC or C bits, then the write to these three bits is disabled. These bits are set or cleared according to the device logic. Furthermore, the $\overline{\text{TO}}$ and $\overline{\text{PD}}$ bits are not writable. Therefore, the result of an instruction with the STATUS register as destination may be different than intended.

For example, `CLRF STATUS` will clear the upper three bits and set the Z bit. This leaves the STATUS register as '000u u1uu' (where u = unchanged).

It is recommended, therefore, that only `BCF`, `BSF`, `SWAPF` and `MOVWF` instructions are used to alter the STATUS register, because these instructions do not affect any Status bits. For other instructions not affecting any Status bits (Refer to [Section 24.0 "Instruction Set Summary"](#)).

Note 1: The C and DC bits operate as Borrow and Digit Borrow out bits, respectively, in subtraction.

3.4 Register Definitions: Status

REGISTER 3-1: STATUS: STATUS REGISTER

| | | | | | | | |
|-------|-----|-----|------------------------|------------------------|---------|-------------------|------------------|
| U-0 | U-0 | U-0 | R-1/q | R-1/q | R/W-0/u | R/W-0/u | R/W-0/u |
| — | — | — | $\overline{\text{TO}}$ | $\overline{\text{PD}}$ | Z | DC ⁽¹⁾ | C ⁽¹⁾ |
| bit 7 | | | | | | | bit 0 |

Legend:

| | | |
|----------------------|----------------------|---|
| R = Readable bit | W = Writable bit | U = Unimplemented bit, read as '0' |
| u = Bit is unchanged | x = Bit is unknown | -n/n = Value at POR and BOR/Value at all other Resets |
| '1' = Bit is set | '0' = Bit is cleared | q = Value depends on condition |

bit 7-5 **Unimplemented:** Read as '0'

bit 4 **TO:** Time-Out bit
 1 = After power-up, `CLRWDT` instruction or `SLEEP` instruction
 0 = A WDT time-out occurred

bit 3 **PD:** Power-Down bit
 1 = After power-up or by the `CLRWDT` instruction
 0 = By execution of the `SLEEP` instruction

bit 2 **Z:** Zero bit
 1 = The result of an arithmetic or logic operation is zero
 0 = The result of an arithmetic or logic operation is not zero

bit 1 **DC:** Digit Carry/Digit Borrow bit (`ADDWF`, `ADDLW`, `SUBLW`, `SUBWF` instructions)⁽¹⁾
 1 = A carry-out from the 4th low-order bit of the result occurred
 0 = No carry-out from the 4th low-order bit of the result

bit 0 **C:** Carry/Borrow bit (`ADDWF`, `ADDLW`, `SUBLW`, `SUBWF` instructions)⁽¹⁾
 1 = A carry-out from the Most Significant bit of the result occurred
 0 = No carry-out from the Most Significant bit of the result occurred

Note 1: For $\overline{\text{Borrow}}$, the polarity is reversed. A subtraction is executed by adding the two's complement of the second operand.

PIC16(L)F1526/7

3.5 Special Function Register

The Special Function Registers are registers used by the application to control the desired operation of peripheral functions in the device. The Special Function Registers occupy the 20 bytes after the core registers of every data memory bank (addresses x0Ch/x8Ch through x1Fh/x9Fh). The registers associated with the operation of the peripherals are described in the appropriate peripheral chapter of this data sheet.

3.5.1 GENERAL PURPOSE RAM

There are up to 80 bytes of GPR in each data memory bank. The Special Function Registers occupy the 20 bytes after the core registers of every data memory bank (addresses x0Ch/x8Ch through x1Fh/x9Fh).

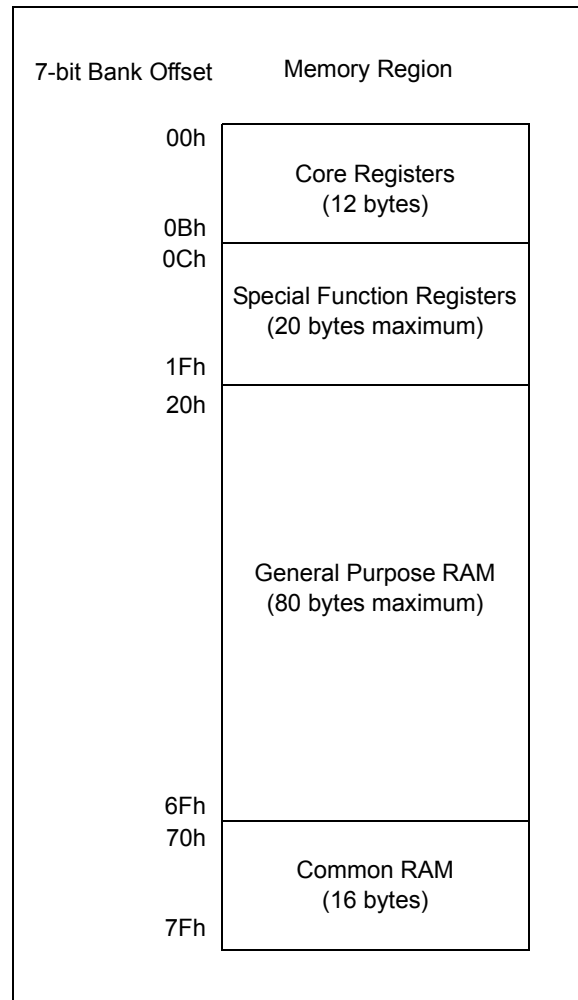
3.5.1.1 Linear Access to GPR

The general purpose RAM can be accessed in a non-banked method via the FSRs. This can simplify access to large memory structures. See [Section 3.8.2 “Linear Data Memory”](#) for more information.

3.5.2 COMMON RAM

There are 16 bytes of common RAM accessible from all banks.

FIGURE 3-3: BANKED MEMORY PARTITIONING



3.5.3 DEVICE MEMORY MAPS

The memory maps for PIC16(L)F1526/7 are shown in [Table 3-3](#).

TABLE 3-3: PIC16(L)F1526/7 MEMORY MAP

| BANK 0 | | BANK 1 | | BANK 2 | | BANK 3 | | BANK 4 | | BANK 5 | | BANK | |
|-------------------------------|-----------------------------------|-------------------------------|-----------------------------------|-------------------------------|-----------------------------------|-------------------------------|-----------------------------------|-------------------------------|-----------------------------------|-------------------------------|-----------------------------------|-------------------------------|-----------------------------------|
| Core Registers (Table 3-2) | | Core Registers (Table 3-2) | | Core Registers (Table 3-2) | | Core Registers (Table 3-2) | | Core Registers (Table 3-2) | | Core Registers (Table 3-2) | | Core Registers (Table 3-2) | |
| 000h | | 080h | | 100h | | 180h | | 200h | | 280h | | 300h | |
| 008h | PORTA | 088h | TRISA | 108h | LATA | 188h | ANSELA | 208h | — | 288h | PORTF | 308h | TRISF |
| 00Ch | PORTB | 08Ch | TRISB | 10Ch | LATB | 18Ch | ANSELB | 20Ch | WPUB | 28Ch | PORTG | 30Ch | TRISG |
| 00Dh | PORTC | 08Dh | TRISC | 10Dh | LATC | 18Dh | — | 20Eh | — | 28Eh | — | 30Eh | — |
| 00Eh | PORTD | 08Eh | TRISD | 10Eh | LATD | 18Eh | ANSEL | 20Fh | WPUD | 28Fh | — | 30Fh | — |
| 00Fh | PORTE | 08Fh | TRISE | 10Fh | LATE | 18Fh | ANSELE | 210h | WPUE | 290h | — | 310h | — |
| 010h | PIR1 | 090h | PIE1 | 110h | — | 190h | PMADRL | 211h | SSP1BUF | 291h | CCPR1L | 311h | CCPR3L |
| 011h | PIR2 | 091h | PIE2 | 111h | — | 191h | PMADRH | 212h | SSP1ADD | 292h | CCPR1H | 312h | CCPR3H |
| 012h | PIR3 | 092h | PIE3 | 112h | — | 192h | PMDATL | 213h | SSP1MSK | 293h | CCP1CON | 313h | CCP3CON |
| 013h | PIR4 | 093h | PIE4 | 113h | — | 193h | PMDATH | 214h | SSP1STAT | 294h | — | 314h | — |
| 014h | TMR0 | 094h | OPTION_REG | 114h | — | 194h | PMCON1 | 215h | SSP1CON1 | 295h | — | 315h | — |
| 015h | TMR1L | 095h | PCON | 115h | BORCON | 195h | PMCON2 | 216h | SSP1CON2 | 296h | — | 316h | — |
| 016h | TMR1H | 096h | WDTCON | 116h | FVRCON | 196h | VREGCON(1) | 217h | SSP1CON3 | 297h | — | 317h | — |
| 017h | T1CON | 097h | — | 117h | — | 197h | — | 218h | — | 298h | CCPR2L | 318h | CCPR4L |
| 018h | T1GCON | 098h | OSCCON | 118h | — | 198h | RC1REG | 219h | SSP2BUF | 299h | CCPR2H | 319h | CCPR4H |
| 019h | TMR2 | 099h | OSCCON | 119h | — | 199h | TX1REG | 21Ah | SSP2ADD | 29Ah | CCP2CON | 31Ah | CCP4CON |
| 01Ah | PR2 | 09Ah | ADRESL | 11Ah | — | 19Ah | SPIBRG | 21Bh | SSP2MSK | 29Bh | — | 31Bh | — |
| 01Bh | T2CON | 09Bh | ADRESH | 11Bh | — | 19Bh | SP1BRGH | 21Ch | SSP2STAT | 29Ch | — | 31Ch | CCPR5L |
| 01Ch | — | 09Ch | ADCON0 | 11Ch | APFCON | 19Ch | RC1STA | 21Dh | SSP2CON1 | 29Dh | CCPTMRS0 | 31Dh | CCPR5H |
| 01Dh | — | 09Dh | ADCON1 | 11Dh | — | 19Dh | TX1STA | 21Eh | SSP2CON2 | 29Eh | CCPTMRS1 | 31Eh | CCP5CON |
| 01Eh | — | 09Eh | — | 11Eh | — | 19Eh | BAUD1CON | 21Fh | SSP2CON3 | 29Fh | CCPTMRS2 | 31Fh | — |
| 01Fh | — | 09Fh | — | 11Fh | — | 19Fh | — | 220h | — | 2A0h | — | 320h | — |
| 020h | General Purpose Register 80 Bytes | 0A0h | General Purpose Register 80 Bytes | 120h | General Purpose Register 80 Bytes | 1A0h | General Purpose Register 80 Bytes | 220h | General Purpose Register 80 Bytes | 2A0h | General Purpose Register 80 Bytes | 320h | General Purpose Register 80 Bytes |
| 06Fh | Common RAM | 0EFh | Common RAM | 16Fh | Common RAM | 1EFh | Common RAM | 26Fh | Common RAM | 2EFh | Common RAM | 36Fh | Common RAM |
| 070h | (Accesses 70h – 7Fh) | 0F0h | (Accesses 70h – 7Fh) | 170h | (Accesses 70h – 7Fh) | 1F0h | (Accesses 70h – 7Fh) | 270h | (Accesses 70h – 7Fh) | 2F0h | (Accesses 70h – 7Fh) | 370h | (Accesses 70h – 7Fh) |
| 07Fh | | 0FFh | | 17Fh | | 1FFh | | 27Fh | | 2FFh | | 37Fh | |

Legend: ■ = Unimplemented data memory locations, read as '0'.

Note 1: PIC16F1526/7 only.

TABLE 3-3: PIC16(L)F1526/7 MEMORY MAP (CONTINUED)

| BANK 8 | | BANK 9 | | BANK 10 | | BANK 11 | | BANK 12 | | BANK 13 | | BANK | |
|--------|---------------------------------------|--------|---------------------------------------|---------|---|---------|---|---------|---|---------|---|------|---------------------------------------|
| 400h | Core Registers (Table 3-2) | 480h | Core Registers (Table 3-2) | 500h | Core Registers (Table 3-2) | 580h | Core Registers (Table 3-2) | 600h | Core Registers (Table 3-2) | 680h | Core Registers (Table 3-2) | 700h | Core Registers (Table 3-2) |
| 40Bh | ANSELF | 48Bh | — | 50Bh | — | 58Bh | — | 60Bh | — | 68Bh | — | 70Bh | — |
| 40Ch | ANSELG | 48Ch | WPUG | 50Ch | — | 58Ch | — | 60Ch | — | 68Ch | — | 70Ch | — |
| 40Dh | — | 48Dh | — | — | — | 58Dh | — | 60Dh | — | — | — | — | — |
| 40Eh | — | 48Eh | — | — | — | 58Eh | — | 60Eh | — | — | — | — | — |
| 40Fh | — | 48Fh | — | — | — | 58Fh | — | 60Fh | — | — | — | — | — |
| 410h | — | 490h | — | — | — | 590h | — | 610h | — | — | — | — | — |
| 411h | TMR3L | 491h | RC2REG | 591h | — | 591h | — | 611h | CCPR6L | — | — | — | — |
| 412h | TMR3H | 492h | TX2REG | 592h | — | 592h | — | 612h | CCPR6H | — | — | — | — |
| 413h | T3CON | 493h | SP2BRG | 593h | — | 593h | — | 613h | CCP6CON | — | — | — | — |
| 414h | T3GCON | 494h | SP2BRGH | 594h | — | 594h | — | 614h | CCPR7L | — | — | — | — |
| 415h | TMR4 | 495h | RC2STA | 595h | — | 595h | TMR8 | 615h | CCPR7H | — | — | — | — |
| 416h | PR4 | 496h | TX2STA | 596h | — | 596h | PR8 | 616h | CCP7CON | — | — | — | — |
| 417h | T4CON | 497h | BAUD2CON | 597h | — | 597h | T8CON | 617h | CCPR8L | — | — | — | — |
| 418h | TMR5L | 498h | — | 598h | — | 598h | — | 618h | CCPR8H | — | — | — | — |
| 419h | TMR5H | 499h | — | 599h | — | 599h | — | 619h | CCP8CON | — | — | — | — |
| 41Ah | T5CON | 49Ah | — | 59Ah | — | 59Ah | — | 61Ah | CCPR9L | — | — | — | — |
| 41Bh | T5GCON | 49Bh | — | 59Bh | — | 59Bh | — | 61Bh | CCPR9H | — | — | — | — |
| 41Ch | TMR6 | 49Ch | — | 59Ch | TMR10 | 59Ch | — | 61Ch | CCP9CON | — | — | — | — |
| 41Dh | PR6 | 49Dh | — | 59Dh | PR10 | 59Dh | — | 61Dh | CCPR10L | — | — | — | — |
| 41Eh | T6CON | 49Eh | — | 59Eh | T10CON | 59Eh | — | 61Eh | CCPR10H | — | — | — | — |
| 41Fh | — | 49Fh | — | 51Fh | — | 59Fh | — | 61Fh | CCP10CON | — | — | — | — |
| 420h | General Purpose Register 80 Bytes | 4A0h | General Purpose Register 32 Bytes | 520h | General Purpose Register 80 Bytes ⁽¹⁾ | 5A0h | General Purpose Register 80 Bytes ⁽¹⁾ | 620h | General Purpose Register 80 Bytes ⁽¹⁾ | 6A0h | General Purpose Register 80 Bytes ⁽¹⁾ | 720h | General Purpose Register 80 Bytes |
| 46Fh | Common RAM (Accesses 70h – 7Fh) | 4EFh | Common RAM (Accesses 70h – 7Fh) | 56Fh | Common RAM (Accesses 70h – 7Fh) | 5EFh | Common RAM (Accesses 70h – 7Fh) | 66Fh | Common RAM (Accesses 70h – 7Fh) | 6EFh | Common RAM (Accesses 70h – 7Fh) | 76Fh | Common RAM (Accesses 70h – 7Fh) |
| 470h | — | 4F0h | — | 570h | — | 5F0h | — | 670h | — | 6F0h | — | 770h | — |
| 47Fh | — | 4Fh | — | 57Fh | — | 5Fh | — | 67Fh | — | 6Fh | — | 77Fh | — |

Legend: ■ = Unimplemented data memory locations, read as '0'.
Note 1: ■ = Unimplemented data memory locations, read as '0'.

TABLE 3-3: PIC16(L)F1526/7 MEMORY MAP (CONTINUED)

| BANK 16 | | BANK 17 | | BANK 18 | | BANK 19 | | BANK 20 | | BANK 21 | | BANK 2 | |
|----------------|---|----------------|---|----------------|---|----------------|---|----------------|---|----------------|---|---------------|---|
| 800h | Core Registers (Table 3-2) | 880h | Core Registers (Table 3-2) | 900h | Core Registers (Table 3-2) | 980h | Core Registers (Table 3-2) | A00h | Core Registers (Table 3-2) | A80h | Core Registers (Table 3-2) | B00h | Core Registers (Table 3-2) |
| 80Bh | Unimplemented Read as '0' | 88Bh | Unimplemented Read as '0' | 90Bh | Unimplemented Read as '0' | 98Bh | Unimplemented Read as '0' | A0Bh | Unimplemented Read as '0' | A8Bh | Unimplemented Read as '0' | B0Bh | Unimplemented Read as '0' |
| 80Ch | General Purpose Register 80 Bytes ⁽¹⁾ | 88Ch | General Purpose Register 80 Bytes ⁽¹⁾ | 90Ch | General Purpose Register 80 Bytes ⁽¹⁾ | 98Ch | General Purpose Register 80 Bytes ⁽¹⁾ | A0Ch | General Purpose Register 80 Bytes ⁽¹⁾ | A8Ch | General Purpose Register 80 Bytes ⁽¹⁾ | B0Ch | General Purpose Register 80 Bytes ⁽¹⁾ |
| 81Fh | Common RAM (Accesses 70h – 7Fh) | 89Fh | Common RAM (Accesses 70h – 7Fh) | 91Fh | Common RAM (Accesses 70h – 7Fh) | 99Fh | Common RAM (Accesses 70h – 7Fh) | A6Fh | Common RAM (Accesses 70h – 7Fh) | AEFh | Common RAM (Accesses 70h – 7Fh) | B6Fh | Common RAM (Accesses 70h – 7Fh) |
| 820h | | 8A0h | | 920h | | 9F0h | | A70h | | AF0h | | B70h | |
| 86Fh | | 8EFh | | 96Fh | | 9EFh | | A7Fh | | AFh | | B7Fh | |
| 870h | | 8F0h | | 970h | | 9F0h | | | | | | | |
| 87Fh | | 8FFh | | 97Fh | | 9FFh | | | | | | | |
| BANK 24 | | BANK 25 | | BANK 26 | | BANK 27 | | BANK 28 | | BANK 29 | | BANK 3 | |
| C00h | Core Registers (Table 3-2) | C80h | Core Registers (Table 3-2) | D00h | Core Registers (Table 3-2) | D80h | Core Registers (Table 3-2) | E00h | Core Registers (Table 3-2) | E80h | Core Registers (Table 3-2) | F00h | Core Registers (Table 3-2) |
| C0Bh | Unimplemented Read as '0' | C8Bh | Unimplemented Read as '0' | D0Bh | Unimplemented Read as '0' | D8Bh | Unimplemented Read as '0' | E0Bh | Unimplemented Read as '0' | E8Bh | Unimplemented Read as '0' | F0Bh | Unimplemented Read as '0' |
| C0Ch | Common RAM (Accesses 70h – 7Fh) | C8Ch | Common RAM (Accesses 70h – 7Fh) | D0Ch | Common RAM (Accesses 70h – 7Fh) | D8Ch | Common RAM (Accesses 70h – 7Fh) | E0Ch | Common RAM (Accesses 70h – 7Fh) | E8Ch | Common RAM (Accesses 70h – 7Fh) | F0Ch | Common RAM (Accesses 70h – 7Fh) |
| C6Fh | | CEFh | | D6Fh | | DEFh | | E6Fh | | EEFh | | F6Fh | |
| C70h | | CF0h | | D70h | | DF0h | | E70h | | EF0h | | F70h | |
| C7Fh | | CFh | | D7Fh | | DFh | | E7Fh | | EFh | | F7Fh | |

Legend: ■ = Unimplemented data memory locations, read as '0'.

Note 1: PIC16(L)F1527 only.

TABLE 3-3: PIC16(L)F1526 MEMORY MAP (CONTINUED)

| Bank 31 | |
|---------|---------------------------------------|
| F80h | Core Registers (Table 3-2) |
| F8Bh | Unimplemented Read as '0' |
| F8Ch | |
| FE3h | |
| FE4h | STATUS_SHAD |
| FE5h | WREG_SHAD |
| FE6h | BSR_SHAD |
| FE7h | PCLATH_SHAD |
| FE8h | FSR0L_SHAD |
| FE9h | FSR0H_SHAD |
| FEAh | FSR1L_SHAD |
| FEBh | FSR1H_SHAD |
| FECh | — |
| FEDh | STKPTR |
| FEEh | TOSL |
| FEFh | TOSH |
| FF0h | Common RAM (Accesses 70h – 7Fh) |
| FFFh | |
| | |

Legend: = Unimplemented data memory locations, read as '0'.

3.5.4 CORE FUNCTION REGISTERS SUMMARY

The Core Function registers listed in [Table 3-4](#) can be addressed from any Bank.

TABLE 3-4: CORE FUNCTION REGISTERS SUMMARY

| Addr | Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Value on POR, BOR | Value on all other Resets | |
|------------------|--------|--|--|--------|-----------------|-----------------|--------|-------|-------|-------------------|---------------------------|-----------|
| Bank 0-31 | | | | | | | | | | | | |
| x00h or x80h | INDF0 | Addressing this location uses contents of FSR0H/FSR0L to address data memory (not a physical register) | | | | | | | | xxxx xxxx | uuuu uuuu | |
| x01h or x81h | INDF1 | Addressing this location uses contents of FSR1H/FSR1L to address data memory (not a physical register) | | | | | | | | xxxx xxxx | uuuu uuuu | |
| x02h or x82h | PCL | Program Counter (PC) Least Significant Byte | | | | | | | | 0000 0000 | 0000 0000 | |
| x03h or x83h | STATUS | — | — | — | \overline{TO} | \overline{PD} | Z | DC | C | ---1 1000 | ---q quuu | |
| x04h or x84h | FSR0L | Indirect Data Memory Address 0 Low Pointer | | | | | | | | 0000 0000 | uuuu uuuu | |
| x05h or x85h | FSR0H | Indirect Data Memory Address 0 High Pointer | | | | | | | | 0000 0000 | 0000 0000 | |
| x06h or x86h | FSR1L | Indirect Data Memory Address 1 Low Pointer | | | | | | | | 0000 0000 | uuuu uuuu | |
| x07h or x87h | FSR1H | Indirect Data Memory Address 1 High Pointer | | | | | | | | 0000 0000 | 0000 0000 | |
| x08h or x88h | BSR | — | — | — | BSR4 | BSR3 | BSR2 | BSR1 | BSR0 | ---0 0000 | ---0 0000 | |
| x09h or x89h | WREG | Working Register | | | | | | | | 0000 0000 | uuuu uuuu | |
| x0Ah or x8Ah | PCLATH | — | Write Buffer for the upper 7 bits of the Program Counter | | | | | | | | -000 0000 | -000 0000 |
| x0Bh or x8Bh | INTCON | GIE | PEIE | TMR0IE | INTE | IOCF | TMR0IF | INTF | IOCF | 0000 0000 | 0000 0000 | |

Legend: x = unknown, u = unchanged, q = value depends on condition, - = unimplemented, read as '0', r = reserved. Shaded locations are unimplemented, read as '0'.

PIC16(L)F1526/7

TABLE 3-2: SPECIAL FUNCTION REGISTER SUMMARY

| Addr | Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Value on POR, BOR | Value on all other Resets |
|---------------|------------|---|--------------|-------------|--------|------------|-------------|-------------|-----------|-------------------|---------------------------|
| Bank 0 | | | | | | | | | | | |
| 00Ch | PORTA | PORTA Data Latch when written: PORTA pins when read | | | | | | | | xxxx xxxx | uuuu uuuu |
| 00Dh | PORTB | PORTB Data Latch when written: PORTB pins when read | | | | | | | | xxxx xxxx | uuuu uuuu |
| 00Eh | PORTC | PORTC Data Latch when written: PORTC pins when read | | | | | | | | xxxx xxxx | uuuu uuuu |
| 00Fh | PORTD | PORTD Data Latch when written: PORTD pins when read | | | | | | | | xxxx xxxx | uuuu uuuu |
| 010h | PORTE | PORTE Data Latch when written: PORTE pins when read | | | | | | | | xxxx xxxx | uuuu uuuu |
| 011h | PIR1 | TMR1GIF | ADIF | RC1IF | TX1IF | SSP1IF | CCP1IF | TMR2IF | TMR1IF | 0000 0000 | 0000 0000 |
| 012h | PIR2 | OSFIF | TMR5GIF | TMR3GIF | — | BCL1IF | TMR10IF | TMR8IF | CCP2IF | 000- 0000 | 000- 0000 |
| 013h | PIR3 | CCP6IF | CCP5IF | CCP4IF | CCP3IF | TMR6IF | TMR5IF | TMR4IF | TMR3IF | 0000 0000 | 0000 0000 |
| 014h | PIR4 | CCP10IF | CCP9IF | RC2IF | TX2IF | CCP8IF | CCP7IF | BCL2IF | SSP2IF | 0000 0000 | 0000 0000 |
| 015h | TMR0 | Timer0 Module Register | | | | | | | | xxxx xxxx | uuuu uuuu |
| 016h | TMR1L | Holding Register for the Least Significant Byte of the 16-bit TMR1 Register | | | | | | | | xxxx xxxx | uuuu uuuu |
| 017h | TMR1H | Holding Register for the Most Significant Byte of the 16-bit TMR1 Register | | | | | | | | xxxx xxxx | uuuu uuuu |
| 018h | T1CON | TMR1CS<1:0> | | T1CKPS<1:0> | | SOSCEN | T1SYNC | — | TMR1ON | 0000 00-0 | uuuu uu-u |
| 019h | T1GCON | TMR1GE | T1GPOL | T1GTM | T1GSPM | T1GGO/DONE | T1GVAL | T1GSS<1:0> | | 0000 0x00 | uuuu uxuu |
| 01Ah | TMR2 | Timer 2 Module Register | | | | | | | | 0000 0000 | 0000 0000 |
| 01Bh | PR2 | Timer 2 Period Register | | | | | | | | 1111 1111 | 1111 1111 |
| 01Ch | T2CON | — | T2OUTPS<3:0> | | | | TMR2ON | T2CKPS<1:0> | | -000 0000 | -000 0000 |
| 01Dh | — | Unimplemented | | | | | | | | — | — |
| 01Eh | — | Unimplemented | | | | | | | | — | — |
| 01Fh | — | Unimplemented | | | | | | | | — | — |
| Bank 1 | | | | | | | | | | | |
| 08Ch | TRISA | PORTA Data Direction Register | | | | | | | | 1111 1111 | 1111 1111 |
| 08Dh | TRISB | PORTB Data Direction Register | | | | | | | | 1111 1111 | 1111 1111 |
| 08Eh | TRISC | PORTC Data Direction Register | | | | | | | | 1111 1111 | 1111 1111 |
| 08Fh | TRISD | PORTD Data Direction Register | | | | | | | | 1111 1111 | 1111 1111 |
| 090h | TRISE | PORTE Data Direction Register | | | | | | | | 1111 1111 | 1111 1111 |
| 091h | PIE1 | TMR1GIE | ADIE | RC1IE | TX1IE | SSP1IE | CCP1IE | TMR2IE | TMR1IE | 0000 0000 | 0000 0000 |
| 092h | PIE2 | OSFIE | TMR5GIE | TMR3GIE | — | BCL1IE | TMR10IE | TMR8IE | CCP2IE | 000- 0000 | 000- 0000 |
| 093h | PIE3 | CCP6IE | CCP5IE | CCP4IE | CCP3IE | TMR6IE | TMR5IE | TMR4IE | TMR3IE | 0000 0000 | 0000 0000 |
| 094h | PIE4 | CCP10IE | CCP9IE | RC2IE | TX2IE | CCP8IE | CCP7IE | BCL2IE | SSP2IE | 0000 0000 | 0000 0000 |
| 095h | OPTION_REG | WPUEN | INTEDG | TMR0CS | TMR0SE | PSA | PS<2:0> | | 1111 1111 | 1111 1111 | |
| 096h | PCON | STKOVF | STKUNF | — | RWDT | RMCLR | RI | POR | BOR | 00-1 11qq | qq-q qquu |
| 097h | WDTCON | — | — | WDTPS<4:0> | | | | SWDTEN | --01 0110 | --01 0110 | |
| 098h | — | Unimplemented | | | | | | | | — | — |
| 099h | OSCCON | — | IRCF<3:0> | | | | — | SCS<1:0> | | -011 1-00 | -011 1-00 |
| 09Ah | OSCSTAT | SOSCR | — | OSTS | HFIOFR | — | — | LFIOFR | HFIOFS | 0-q0 --00 | q-qq --0q |
| 09Bh | ADRESL | ADC Result Register Low | | | | | | | | xxxx xxxx | uuuu uuuu |
| 09Ch | ADRESH | ADC Result Register High | | | | | | | | xxxx xxxx | uuuu uuuu |
| 09Dh | ADCON0 | — | CHS<4:0> | | | | GO/DONE | ADON | -000 0000 | -000 0000 | |
| 09Eh | ADCON1 | ADFM | ADCS<2:0> | | — | — | ADPREF<1:0> | | 0000 --00 | 0000 --00 | |
| 09Fh | — | Unimplemented | | | | | | | | — | — |

Legend: x = unknown, u = unchanged, q = value depends on condition, - = unimplemented, read as '0', r = reserved.
Shaded locations are unimplemented, read as '0'.

- Note 1:** PIC16F1526/7 only.
Note 2: Unimplemented, read as '1'.

PIC16(L)F1526/7

TABLE 3-2: SPECIAL FUNCTION REGISTER SUMMARY (CONTINUED)

| Addr | Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Value on POR, BOR | Value on all other Resets |
|---------------|--------|------------------|--------|-------|-------|-------|-------|------------|---------|-------------------|---------------------------|
| Bank 2 | | | | | | | | | | | |
| 10Ch | LATA | PORTA Data Latch | | | | | | | | xxxx xxxx | uuuu uuuu |
| 10Dh | LATB | PORTB Data Latch | | | | | | | | xxxx xxxx | uuuu uuuu |
| 10Eh | LATC | PORTC Data Latch | | | | | | | | xxxx xxxx | uuuu uuuu |
| 10Fh | LATD | PORTD Data Latch | | | | | | | | xxxx xxxx | uuuu uuuu |
| 110h | LATE | PORTE Data Latch | | | | | | | | xxxx xxxx | uuuu uuuu |
| 111h to 115h | — | Unimplemented | | | | | | | | — | — |
| 116h | BORCON | SBOREN | BORFS | — | — | — | — | — | BORRDY | 10-- ---q | uu-- ---u |
| 117h | FVRCON | FVREN | FVRRDY | TSEN | TSRNG | — | — | ADFVR<1:0> | | 0q00 0000 | 0q00 0000 |
| 118h to 11Ch | — | Unimplemented | | | | | | | | — | — |
| 11Dh | APFCON | — | — | — | — | — | — | T3CKISEL | CCP2SEL | ---- --00 | ---- --00 |
| 11Eh | — | Unimplemented | | | | | | | | — | — |
| 11Fh | — | Unimplemented | | | | | | | | — | — |

Bank 3

| | | | | | | | | | | | | | |
|------|------------------------|--|---|--|-------|-------|-------|--------|----------|-----------|-----------|-----------|-----------|
| 18Ch | ANSELA | — | — | ANSA5 | — | ANSA3 | ANSA2 | ANSA1 | ANSA0 | --1- 1111 | --1- 1111 | | |
| 18Dh | ANSELB | — | — | ANSB5 | ANSB4 | ANSB3 | ANSB2 | ANSB1 | ANSB0 | --11 1111 | --11 1111 | | |
| 18Eh | ANSELC | Unimplemented | | | | | | | | — | — | | |
| 18Fh | ANSELD | — | — | — | — | ANSD3 | ANSD2 | ANSD1 | ANSD0 | ---- 1111 | ---- 1111 | | |
| 190h | ANSELE | — | — | — | — | — | ANSE2 | ANSE1 | ANSE0 | ---- -111 | ---- -111 | | |
| 191h | PMADRL | Program Memory Address Register Low Byte | | | | | | | | 0000 0000 | 0000 0000 | | |
| 192h | PMADRH | — ⁽²⁾ | Program Memory Address Register High Byte | | | | | | | | 1000 0000 | 1000 0000 | |
| 193h | PMDATL | Program Memory Data Register Low Byte | | | | | | | | xxxx xxxx | uuuu uuuu | | |
| 194h | PMDATH | — | — | Program Memory Data Register High Byte | | | | | | | | --xx xxxx | --uu uuuu |
| 195h | PMCON1 | — ⁽²⁾ | CFG5 | LWLO | FREE | WRERR | WREN | WR | RD | 1000 x000 | 1000 q000 | | |
| 196h | PMCON2 | Program Memory control register 2 | | | | | | | | 0000 0000 | 0000 0000 | | |
| 197h | VREGCON ⁽¹⁾ | — | — | — | — | — | — | VREGPM | Reserved | ---- --01 | ---- --01 | | |
| 198h | — | Unimplemented | | | | | | | | — | — | | |
| 199h | RC1REG | USART Receive Data Register | | | | | | | | 0000 0000 | 0000 0000 | | |
| 19Ah | TX1REG | USART Transmit Data Register | | | | | | | | 0000 0000 | 0000 0000 | | |
| 19Bh | SP1BRG | BRG<7:0> | | | | | | | | 0000 0000 | 0000 0000 | | |
| 19Ch | SP1BRGH | BRG<15:8> | | | | | | | | 0000 0000 | 0000 0000 | | |
| 19Dh | RC1STA | SPEN | RX9 | SREN | CREN | ADDEN | FERR | OERR | RX9D | 0000 000x | 0000 000x | | |
| 19Eh | TX1STA | CSRC | TX9 | TXEN | SYNC | SEnDB | BRGH | TRMT | TX9D | 0000 0010 | 0000 0010 | | |
| 19Fh | BAUD1CON | ABDOVF | RCIDL | — | SCKP | BRG16 | — | WUE | ABDEN | 01-0 0-00 | 01-0 0-00 | | |

Legend: x = unknown, u = unchanged, q = value depends on condition, - = unimplemented, read as '0', r = reserved.
Shaded locations are unimplemented, read as '0'.

- Note** 1: PIC16F1526/7 only.
2: Unimplemented, read as '1'.

PIC16(L)F1526/7

TABLE 3-2: SPECIAL FUNCTION REGISTER SUMMARY (CONTINUED)

| Addr | Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Value on POR, BOR | Value on all other Resets |
|---------------|----------|---|---------|-------------|-------|--------------|------------|-------------|-------|-------------------|---------------------------|
| Bank 4 | | | | | | | | | | | |
| 20Ch | — | Unimplemented | | | | | | | | — | — |
| 20Dh | WPUB | WPUB7 | WPUB6 | WPUB5 | WPUB4 | WPUB3 | WPUB2 | WPUB1 | WPUB0 | 1111 1111 | 1111 1111 |
| 20Eh | — | Unimplemented | | | | | | | | — | — |
| 20Fh | WPUD | WPUD7 | WPUD6 | WPUD5 | WPUD4 | WPUD3 | WPUD2 | WPUD1 | WPUD0 | 1111 1111 | 1111 1111 |
| 210h | WPUE | WPUE7 | WPUE6 | WPUE5 | WPUE4 | WPUE3 | WPUE2 | WPUE1 | WPUE0 | 1111 1111 | 1111 1111 |
| 211h | SSP1BUF | Synchronous Serial Port Receive Buffer/Transmit Register | | | | | | | | xxxx xxxx | uuuu uuuu |
| 212h | SSP1ADD | Synchronous Serial Port (I ² C mode) Address Register | | | | | | | | 0000 0000 | 0000 0000 |
| 213h | SSP1MSK | Synchronous Serial Port (I ² C mode) Address Mask Register | | | | | | | | 1111 1111 | 1111 1111 |
| 214h | SSP1STAT | SMP | CKE | D/Ā | P | S | R/W | UA | BF | 0000 0000 | 0000 0000 |
| 215h | SSP1CON1 | WCOL | SSPOV | SSPEN | CKP | SSPM<3:0> | | | | 0000 0000 | 0000 0000 |
| 216h | SSP1CON2 | GCEN | ACKSTAT | ACKDT | ACKEN | RCEN | PEN | RSEN | SEN | 0000 0000 | 0000 0000 |
| 217h | SSP1CON3 | ACKTIM | PCIE | SCIE | BOEN | SDAHT | SBCDE | AHEN | DHEN | 0000 0000 | 0000 0000 |
| 218h | — | Unimplemented | | | | | | | | — | — |
| 219h | SSP2BUF | Synchronous Serial Port Receive Buffer/Transmit Register | | | | | | | | xxxx xxxx | uuuu uuuu |
| 21Ah | SSP2ADD | Synchronous Serial Port (I ² C mode) Address Register | | | | | | | | 0000 0000 | 0000 0000 |
| 21Bh | SSP2MSK | Synchronous Serial Port (I ² C mode) Address Mask Register | | | | | | | | 1111 1111 | 1111 1111 |
| 21Ch | SSP2STAT | SMP | CKE | D/Ā | P | S | R/W | UA | BF | 0000 0000 | 0000 0000 |
| 21Dh | SSP2CON1 | WCOL | SSPOV | SSPEN | CKP | SSPM<3:0> | | | | 0000 0000 | 0000 0000 |
| 21Eh | SSP2CON2 | GCEN | ACKSTAT | ACKDT | ACKEN | RCEN | PEN | RSEN | SEN | 0000 0000 | 0000 0000 |
| 21Fh | SSP2CON3 | ACKTIM | PCIE | SCIE | BOEN | SDAHT | SBCDE | AHEN | DHEN | 0000 0000 | 0000 0000 |
| Bank 5 | | | | | | | | | | | |
| 28Ch | PORTF | RF7 | RF6 | RF5 | RF4 | RF3 | RF2 | RF1 | RF0 | xxxx xxxx | uuuu uuuu |
| 28Dh | PORTG | — | — | RG5 | RG4 | RG3 | RG2 | RG1 | RG0 | --xx xxxx | --uu uuuu |
| 28Eh | — | Unimplemented | | | | | | | | — | — |
| 28Fh | — | Unimplemented | | | | | | | | — | — |
| 290h | — | Unimplemented | | | | | | | | — | — |
| 291h | CCPR1L | Capture/Compare/PWM Register 1 (LSB) | | | | | | | | xxxx xxxx | uuuu uuuu |
| 292h | CCPR1H | Capture/Compare/PWM Register 1 (MSB) | | | | | | | | xxxx xxxx | uuuu uuuu |
| 293h | CCP1CON | — | — | DC1B<1:0> | | | CCP1M<3:0> | | | --00 0000 | --00 0000 |
| 294h | — | Unimplemented | | | | | | | | — | — |
| 295h | — | Unimplemented | | | | | | | | — | — |
| 296h | — | Unimplemented | | | | | | | | — | — |
| 297h | — | Unimplemented | | | | | | | | — | — |
| 298h | CCPR2L | Capture/Compare/PWM Register 2 (LSB) | | | | | | | | xxxx xxxx | uuuu uuuu |
| 299h | CCPR2H | Capture/Compare/PWM Register 2 (MSB) | | | | | | | | xxxx xxxx | uuuu uuuu |
| 29Ah | CCP2CON | — | — | DC2B<1:0> | | | CCP2M<3:0> | | | --00 0000 | --00 0000 |
| 29Bh | — | Unimplemented | | | | | | | | — | — |
| 29Ch | — | Unimplemented | | | | | | | | — | — |
| 29Dh | CCPTMRS0 | C4TSEL<1:0> | | C3TSEL<1:0> | | C2TSEL<1:0> | | C1TSEL<1:0> | | 0000 0000 | 0000 0000 |
| 29Eh | CCPTMRS1 | C8TSEL<1:0> | | C7TSEL<1:0> | | C6TSEL<1:0> | | C5TSEL<1:0> | | 0000 0000 | 0000 0000 |
| 29Fh | CCPTMRS2 | — | — | — | — | C10TSEL<1:0> | | C9TSEL<1:0> | | ---- 0000 | ---- 0000 |

Legend: x = unknown, u = unchanged, q = value depends on condition, - = unimplemented, read as '0', r = reserved.
 Shaded locations are unimplemented, read as '0'.

- Note 1:** PIC16F1526/7 only.
Note 2: Unimplemented, read as '1'.

PIC16(L)F1526/7

TABLE 3-2: SPECIAL FUNCTION REGISTER SUMMARY (CONTINUED)

| Addr | Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Value on POR, BOR | Value on all other Resets |
|---------------|---------|--------------------------------------|--------|------------------|------------|--------|--------|--------|--------|-------------------|---------------------------|
| Bank 6 | | | | | | | | | | | |
| 30Ch | TRISF | TRISF7 | TRISF6 | TRISF5 | TRISF4 | TRISF3 | TRISF2 | TRISF1 | TRISF0 | 1111 1111 | 1111 1111 |
| 30Dh | TRISG | — | — | — ⁽²⁾ | TRISG4 | TRISG3 | TRISG2 | TRISG1 | TRISG0 | --11 1111 | --11 1111 |
| 30Eh | — | Unimplemented | | | | | | | | — | — |
| 30Fh | — | Unimplemented | | | | | | | | — | — |
| 310h | — | Unimplemented | | | | | | | | — | — |
| 311h | CCPR3L | Capture/Compare/PWM Register 3 (LSB) | | | | | | | | xxxx xxxx | uuuu uuuu |
| 312h | CCPR3H | Capture/Compare/PWM Register 3 (MSB) | | | | | | | | xxxx xxxx | uuuu uuuu |
| 313h | CCP3CON | — | — | DC3B<1:0> | CCP3M<3:0> | | | | | --00 0000 | --00 0000 |
| 314h | — | Unimplemented | | | | | | | | — | — |
| 315h | — | Unimplemented | | | | | | | | — | — |
| 316h | — | Unimplemented | | | | | | | | — | — |
| 317h | — | Unimplemented | | | | | | | | — | — |
| 318h | CCPR4L | Capture/Compare/PWM Register 4 (LSB) | | | | | | | | xxxx xxxx | uuuu uuuu |
| 319h | CCPR4H | Capture/Compare/PWM Register 4 (MSB) | | | | | | | | xxxx xxxx | uuuu uuuu |
| 31Ah | CCP4CON | — | — | DC4B<1:0> | CCP4M<3:0> | | | | | --00 0000 | --00 0000 |
| 31Bh | — | Unimplemented | | | | | | | | — | — |
| 31Ch | CCPR5L | Capture/Compare/PWM Register 5 (LSB) | | | | | | | | xxxx xxxx | uuuu uuuu |
| 31Dh | CCPR5H | Capture/Compare/PWM Register 5 (MSB) | | | | | | | | xxxx xxxx | uuuu uuuu |
| 31Eh | CCP5CON | — | — | DC5B<1:0> | CCP5M<3:0> | | | | | --00 0000 | --00 0000 |
| 31Fh | — | Unimplemented | | | | | | | | — | — |
| Bank 7 | | | | | | | | | | | |
| 38Ch | LATF | LATF7 | LATF6 | LATF5 | LATF4 | LATF3 | LATF2 | LATF1 | LATF0 | xxxx xxxx | uuuu uuuu |
| 38Dh | LATG | — | — | — | LATG4 | LATG3 | LATG2 | LATG1 | LATG0 | ---x xxxx | ---u uuuu |
| 38Eh to 393h | — | Unimplemented | | | | | | | | — | — |
| 394h | IOCBP | IOCBP7 | IOCBP6 | IOCBP5 | IOCBP4 | IOCBP3 | IOCBP2 | IOCBP1 | IOCBP0 | 0000 0000 | 0000 0000 |
| 395h | IOCBN | IOCBN7 | IOCBN6 | IOCBN5 | IOCBN4 | IOCBN3 | IOCBN2 | IOCBN1 | IOCBN0 | 0000 0000 | 0000 0000 |
| 396h | IOCBF | IOCBF7 | IOCBF6 | IOCBF5 | IOCBF4 | IOCBF3 | IOCBF2 | IOCBF1 | IOCBF0 | 0000 0000 | 0000 0000 |
| 397h to 39Fh | — | Unimplemented | | | | | | | | — | — |

Legend: x = unknown, u = unchanged, q = value depends on condition, - = unimplemented, read as '0', r = reserved.
Shaded locations are unimplemented, read as '0'.

- Note 1:** PIC16F1526/7 only.
Note 2: Unimplemented, read as '1'.

PIC16(L)F1526/7

TABLE 3-2: SPECIAL FUNCTION REGISTER SUMMARY (CONTINUED)

| Addr | Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Value on POR, BOR | Value on all other Resets |
|---------------|----------|---|--------------|-------------|--------|------------|-----------------|-------------|--------|-------------------|---------------------------|
| Bank 8 | | | | | | | | | | | |
| 40Ch | ANSELF | ANSF7 | ANSF6 | ANSF5 | ANSF4 | ANSF3 | ANSF2 | ANSF1 | ANSF0 | 1111 1111 | 1111 1111 |
| 40Dh | ANSELG | — | — | — | ANSG4 | ANSG3 | ANSG2 | ANSG1 | — | ---1 111- | ---1 111- |
| 40Eh | — | Unimplemented | | | | | | | | — | — |
| 40Fh | — | Unimplemented | | | | | | | | — | — |
| 410h | — | Unimplemented | | | | | | | | — | — |
| 411h | TMR3L | Holding Register for the Least Significant Byte of the 16-bit TMR3 Register | | | | | | | | xxxx xxxx | uuuu uuuu |
| 412h | TMR3H | Holding Register for the Most Significant Byte of the 16-bit TMR3 Register | | | | | | | | xxxx xxxx | uuuu uuuu |
| 413h | T3CON | TMR3CS<1:0> | | T3CKPS<1:0> | | SOSCEN | T3SYN \bar{C} | — | TMR3ON | 0000 00-0 | uuuu uu-u |
| 414h | T3GCON | TMR3GE | T3GPOL | T3GTM | T3GSPM | T3GGO/DONE | T3GVAL | T3GSS<1:0> | | 0000 0x00 | uuuu uxuu |
| 415h | TMR4 | Timer 4 Module Register | | | | | | | | 0000 0000 | 0000 0000 |
| 416h | PR4 | Timer 4 Period Register | | | | | | | | 1111 1111 | 1111 1111 |
| 417h | T4CON | — | T4OUTPS<3:0> | | | | TMR4ON | T4CKPS<1:0> | | -000 0000 | -000 0000 |
| 418h | TMR5L | Holding Register for the Least Significant Byte of the 16-bit TMR5 Register | | | | | | | | xxxx xxxx | uuuu uuuu |
| 419h | TMR5H | Holding Register for the Most Significant Byte of the 16-bit TMR5 Register | | | | | | | | xxxx xxxx | uuuu uuuu |
| 41Ah | T5CON | TMR5CS<1:0> | | T5CKPS<1:0> | | SOSCEN | T5SYN \bar{C} | — | TMR5ON | 0000 00-0 | uuuu uu-u |
| 41Bh | T5GCON | TMR5GE | T5GPOL | T5GTM | T5GSPM | T5GGO/DONE | T5GVAL | T5GSS<1:0> | | 0000 0x00 | uuuu uxuu |
| 41Ch | TMR6 | Timer 6 Module Register | | | | | | | | 0000 0000 | 0000 0000 |
| 41Dh | PR6 | Timer 6 Period Register | | | | | | | | 1111 1111 | 1111 1111 |
| 41Eh | T6CON | — | T6OUTPS<3:0> | | | | TMR6ON | T6CKPS<1:0> | | -000 0000 | -000 0000 |
| 41Fh | — | Unimplemented | | | | | | | | — | — |
| Bank 9 | | | | | | | | | | | |
| 48Ch | — | Unimplemented | | | | | | | | — | — |
| 48Dh | WPUG | — | — | WPUG5 | — | — | — | — | — | --1- ---- | --1- ---- |
| 48Dh to 490h | — | Unimplemented | | | | | | | | — | — |
| 491h | RC2REG | USART Receive Data Register | | | | | | | | 0000 0000 | 0000 0000 |
| 492h | TX2REG | USART Transmit Data Register | | | | | | | | 0000 0000 | 0000 0000 |
| 493h | SP2BRG | BRG<7:0> | | | | | | | | 0000 0000 | 0000 0000 |
| 494h | SP2BRGH | BRG<15:8> | | | | | | | | 0000 0000 | 0000 0000 |
| 495h | RC2STA | SPEN | RX9 | SREN | CREN | ADDEN | FERR | OERR | RX9D | 0000 000x | 0000 000x |
| 496h | TX2STA | CSRC | TX9 | TXEN | SYNC | SENDB | BRGH | TRMT | TX9D | 0000 0010 | 0000 0010 |
| 497h | BAUD2CON | ABDOVF | RCIDL | — | SCKP | BRG16 | — | WUE | ABDEN | 01-0 0-00 | 01-0 0-00 |
| 498h to 49Fh | — | Unimplemented | | | | | | | | — | — |

Legend: x = unknown, u = unchanged, q = value depends on condition, - = unimplemented, read as '0', r = reserved.
Shaded locations are unimplemented, read as '0'.

- Note 1:** PIC16F1526/7 only.
Note 2: Unimplemented, read as '1'.

PIC16(L)F1526/7

TABLE 3-2: SPECIAL FUNCTION REGISTER SUMMARY (CONTINUED)

| Addr | Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Value on POR, BOR | Value on all other Resets | |
|--|----------|---------------------------------------|---------------|------------|-------|-------------|---------|--------------|-----------|-------------------|---------------------------|-----------|
| Bank 10 | | | | | | | | | | | | |
| 50Ch — 51Fh | — | Unimplemented | | | | | | | | — | — | |
| Bank 11 | | | | | | | | | | | | |
| 58Ch — 594h | — | Unimplemented | | | | | | | | — | — | |
| 595h | TMR8 | Timer 8 Module Register | | | | | | | | 0000 0000 | 0000 0000 | |
| 596h | PR8 | Timer 8 Period Register | | | | | | | | 1111 1111 | 1111 1111 | |
| 597h | T8CON | — | T8OUTPS<3:0> | | | | TMR8ON | T8CKPS<1:0> | | | -000 0000 | -000 0000 |
| 598h — 59Bh | — | Unimplemented | | | | | | | | — | — | |
| 59Ch | TMR10 | Timer 10 Module Register | | | | | | | | 0000 0000 | 0000 0000 | |
| 59Dh | PR10 | Timer 10 Period Register | | | | | | | | 1111 1111 | 1111 1111 | |
| 59Eh | T10CON | — | T10OUTPS<3:0> | | | | TMR10ON | T10CKPS<1:0> | | | -000 0000 | -000 0000 |
| 59Fh | — | Unimplemented | | | | | | | | — | — | |
| Bank 12 | | | | | | | | | | | | |
| 60Ch — 610h | — | Unimplemented | | | | | | | | — | — | |
| 611h | CCPR6L | Capture/Compare/PWM Register 6 (LSB) | | | | | | | | xxxx xxxx | uuuu uuuu | |
| 612h | CCPR6H | Capture/Compare/PWM Register 6 (MSB) | | | | | | | | xxxx xxxx | uuuu uuuu | |
| 613h | CCP6CON | — | — | DC6B<1:0> | | CCP6M<3:0> | | | --00 0000 | --00 0000 | | |
| 614h | CCPR7L | Capture/Compare/PWM Register 7 (LSB) | | | | | | | | xxxx xxxx | uuuu uuuu | |
| 615h | CCPR7H | Capture/Compare/PWM Register 7 (MSB) | | | | | | | | xxxx xxxx | uuuu uuuu | |
| 616h | CCP7CON | — | — | DC7B<1:0> | | CCP7M<3:0> | | | --00 0000 | --00 0000 | | |
| 617h | CCPR8L | Capture/Compare/PWM Register 8 (LSB) | | | | | | | | xxxx xxxx | uuuu uuuu | |
| 618h | CCPR8H | Capture/Compare/PWM Register 8 (MSB) | | | | | | | | xxxx xxxx | uuuu uuuu | |
| 619h | CCP8CON | — | — | DC8B<1:0> | | CCP8M<3:0> | | | --00 0000 | --00 0000 | | |
| 61Ah | CCPR9L | Capture/Compare/PWM Register 9 (LSB) | | | | | | | | xxxx xxxx | uuuu uuuu | |
| 61Bh | CCPR9H | Capture/Compare/PWM Register 9 (MSB) | | | | | | | | xxxx xxxx | uuuu uuuu | |
| 61Ch | CCP9CON | — | — | DC9B<1:0> | | CCP9M<3:0> | | | --00 0000 | --00 0000 | | |
| 61Dh | CCPR10L | Capture/Compare/PWM Register 10 (LSB) | | | | | | | | xxxx xxxx | uuuu uuuu | |
| 61Eh | CCPR10H | Capture/Compare/PWM Register 10 (MSB) | | | | | | | | xxxx xxxx | uuuu uuuu | |
| 61Fh | CCP10CON | — | — | DC10B<1:0> | | CCP10M<3:0> | | | --00 0000 | --00 0000 | | |
| Bank 13-30 | | | | | | | | | | | | |
| x0Ch or x8Ch to x1Fh or x9Fh | — | Unimplemented | | | | | | | | — | — | |

Legend: x = unknown, u = unchanged, q = value depends on condition, - = unimplemented, read as '0', r = reserved.
Shaded locations are unimplemented, read as '0'.

- Note 1:** PIC16F1526/7 only.
Note 2: Unimplemented, read as '1'.

PIC16(L)F1526/7

TABLE 3-2: SPECIAL FUNCTION REGISTER SUMMARY (CONTINUED)

| Addr | Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Value on POR, BOR | Value on all other Resets | |
|-------------------|-------------|---|---|-------|--|-------|--------|---------|--------|-------------------|---------------------------|-----------|
| Bank 31 | | | | | | | | | | | | |
| F8Ch — FE3h | — | Unimplemented | | | | | | | | — | — | |
| FE4h | STATUS_SHAD | — | — | — | — | — | Z_SHAD | DC_SHAD | C_SHAD | ---- -xxx | ---- -uuu | |
| FE5h | WREG_SHAD | Working Register Normal (Non-ICD) Shadow | | | | | | | | xxxx xxxx | uuuu uuuu | |
| FE6h | BSR_SHAD | — | — | — | Bank Select Register Normal (Non-ICD) Shadow | | | | | ---x xxxx | ---u uuuu | |
| FE7h | PCLATH_SHAD | — | Program Counter Latch High Register Normal (Non-ICD) Shadow | | | | | | | | -xxx xxxx | uuuu uuuu |
| FE8h | FSR0L_SHAD | Indirect Data Memory Address 0 Low Pointer Normal (Non-ICD) Shadow | | | | | | | | xxxx xxxx | uuuu uuuu | |
| FE9h | FSR0H_SHAD | Indirect Data Memory Address 0 High Pointer Normal (Non-ICD) Shadow | | | | | | | | xxxx xxxx | uuuu uuuu | |
| FEAh | FSR1L_SHAD | Indirect Data Memory Address 1 Low Pointer Normal (Non-ICD) Shadow | | | | | | | | xxxx xxxx | uuuu uuuu | |
| FEBh | FSR1H_SHAD | Indirect Data Memory Address 1 High Pointer Normal (Non-ICD) Shadow | | | | | | | | xxxx xxxx | uuuu uuuu | |
| FECh | — | Unimplemented | | | | | | | | — | — | |
| FEDh | STKPTR | — | — | — | Current Stack Pointer | | | | | ---1 1111 | ---1 1111 | |
| FEEh | TOSL | Top of Stack Low byte | | | | | | | | xxxx xxxx | uuuu uuuu | |
| FEFh | TOSH | — | Top of Stack High byte | | | | | | | | -xxx xxxx | -uuu uuuu |

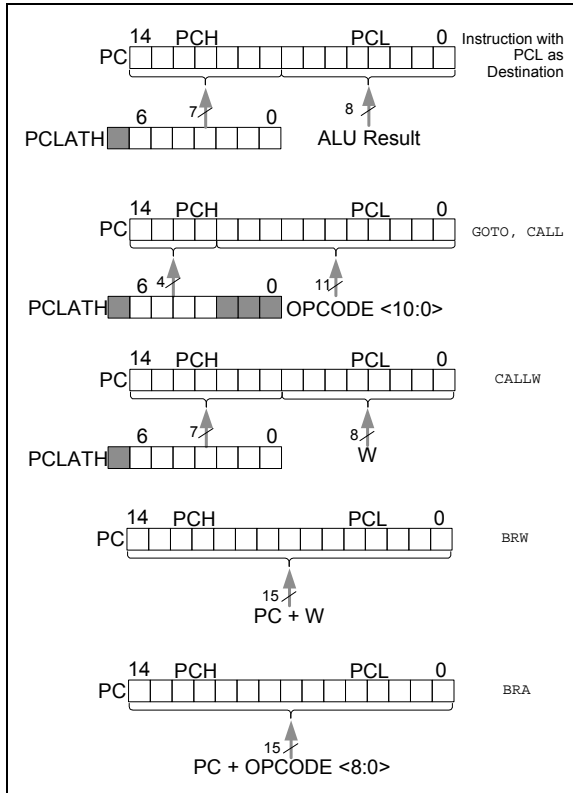
Legend: x = unknown, u = unchanged, q = value depends on condition, - = unimplemented, read as '0', r = reserved.
Shaded locations are unimplemented, read as '0'.

- Note 1:** PIC16F1526/7 only.
Note 2: Unimplemented, read as '1'.

3.6 PCL and PCLATH

The Program Counter (PC) is 15 bits wide. The low byte comes from the PCL register, which is a readable and writable register. The high byte (PC<14:8>) is not directly readable or writable and comes from PCLATH. On any Reset, the PC is cleared. Figure 3-4 shows the five situations for the loading of the PC.

FIGURE 3-4: LOADING OF PC IN DIFFERENT SITUATIONS



3.6.1 MODIFYING PCL

Executing any instruction with the PCL register as the destination simultaneously causes the Program Counter PC<14:8> bits (PCH) to be replaced by the contents of the PCLATH register. This allows the entire contents of the program counter to be changed by writing the desired upper 7 bits to the PCLATH register. When the lower 8 bits are written to the PCL register, all 15 bits of the program counter will change to the values contained in the PCLATH register and those being written to the PCL register.

3.6.2 COMPUTED GOTO

A computed GOTO is accomplished by adding an offset to the program counter (ADDWF PCL). When performing a table read using a computed GOTO method, care should be exercised if the table location crosses a PCL memory boundary (each 256-byte block). Refer to Application Note AN556, "Implementing a Table Read" (DS00556).

3.6.3 COMPUTED FUNCTION CALLS

A computed function CALL allows programs to maintain tables of functions and provide another way to execute state machines or look-up tables. When performing a table read using a computed function CALL, care should be exercised if the table location crosses a PCL memory boundary (each 256-byte block).

If using the CALL instruction, the PCH<2:0> and PCL registers are loaded with the operand of the CALL instruction. PCH<6:3> is loaded with PCLATH<6:3>.

The CALLW instruction enables computed calls by combining PCLATH and W to form the destination address. A computed CALLW is accomplished by loading the W register with the desired address and executing CALLW. The PCL register is loaded with the value of W and PCH is loaded with PCLATH.

3.6.4 BRANCHING

The branching instructions add an offset to the PC. This allows relocatable code and code that crosses page boundaries. There are two forms of branching, BRW and BRA. The PC will have incremented to fetch the next instruction in both cases. When using either branching instruction, a PCL memory boundary may be crossed.

If using BRW, load the W register with the desired unsigned address and execute BRW. The entire PC will be loaded with the address PC + 1 + W.

If using BRA, the entire PC will be loaded with PC + 1 +, the signed value of the operand of the BRA instruction.

PIC16(L)F1526/7

3.7 Stack

All devices have a 16-level x 15-bit wide hardware stack (refer to Figures 3-5 through 3-8). The stack space is not part of either program or data space. The PC is PUSHed onto the stack when `CALL` or `CALLW` instructions are executed or an interrupt causes a branch. The stack is POPed in the event of a `RETURN`, `RETLW` or a `RETFIE` instruction execution. `PCLATH` is not affected by a `PUSH` or `POP` operation.

The stack operates as a circular buffer if the `STVREN` bit is programmed to '0' (Configuration Words). This means that after the stack has been PUSHed sixteen times, the seventeenth PUSH overwrites the value that was stored from the first PUSH. The eighteenth PUSH overwrites the second PUSH (and so on). The `STKOVF` and `STKUNF` flag bits will be set on an Overflow/Underflow, regardless of whether the Reset is enabled.

Note 1: There are no instructions/mnemonics called `PUSH` or `POP`. These are actions that occur from the execution of the `CALL`, `CALLW`, `RETURN`, `RETLW` and `RETFIE` instructions or the vectoring to an interrupt address.

3.7.1 ACCESSING THE STACK

The stack is available through the `TOSH`, `TOSL` and `STKPTR` registers. `STKPTR` is the current value of the Stack Pointer. `TOSH:TOSL` register pair points to the TOP of the stack. Both registers are read/writable. `TOS` is split into `TOSH` and `TOSL` due to the 15-bit size of the PC. To access the stack, adjust the value of `STKPTR`, which will position `TOSH:TOSL`, then read/write to `TOSH:TOSL`. `STKPTR` is 5 bits to allow detection of overflow and underflow.

Note: Care should be taken when modifying the `STKPTR` while interrupts are enabled.

During normal program operation, `CALL`, `CALLW` and Interrupts will increment `STKPTR` while `RETLW`, `RETURN`, and `RETFIE` will decrement `STKPTR`. At any time `STKPTR` can be inspected to see how much stack is left. The `STKPTR` always points at the currently used place on the stack. Therefore, a `CALL` or `CALLW` will increment the `STKPTR` and then write the PC, and a return will unload the PC and then decrement the `STKPTR`.

Reference Figures 3-5 through 3-8 for examples of accessing the stack.

FIGURE 3-5: ACCESSING THE STACK EXAMPLE 1

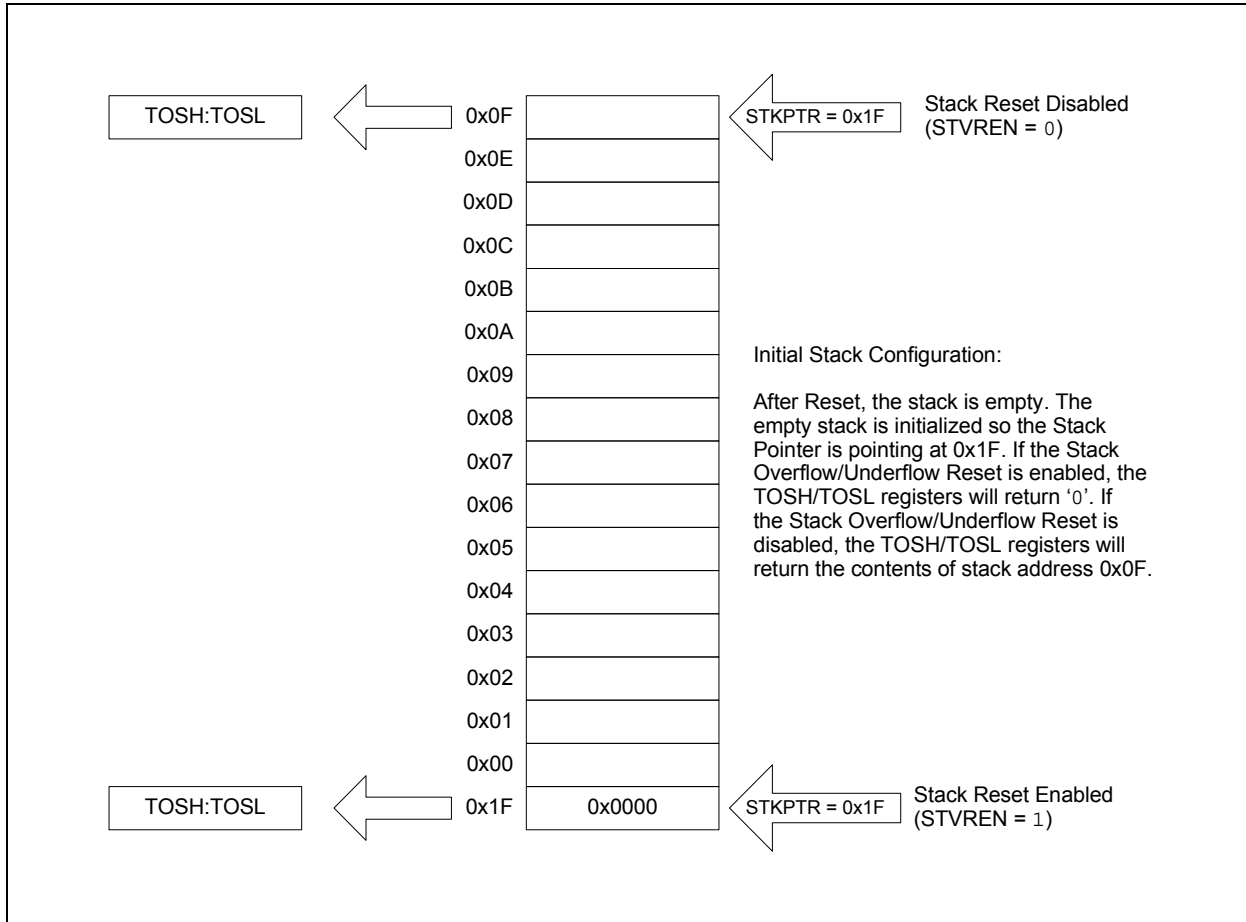


FIGURE 3-6: ACCESSING THE STACK EXAMPLE 2

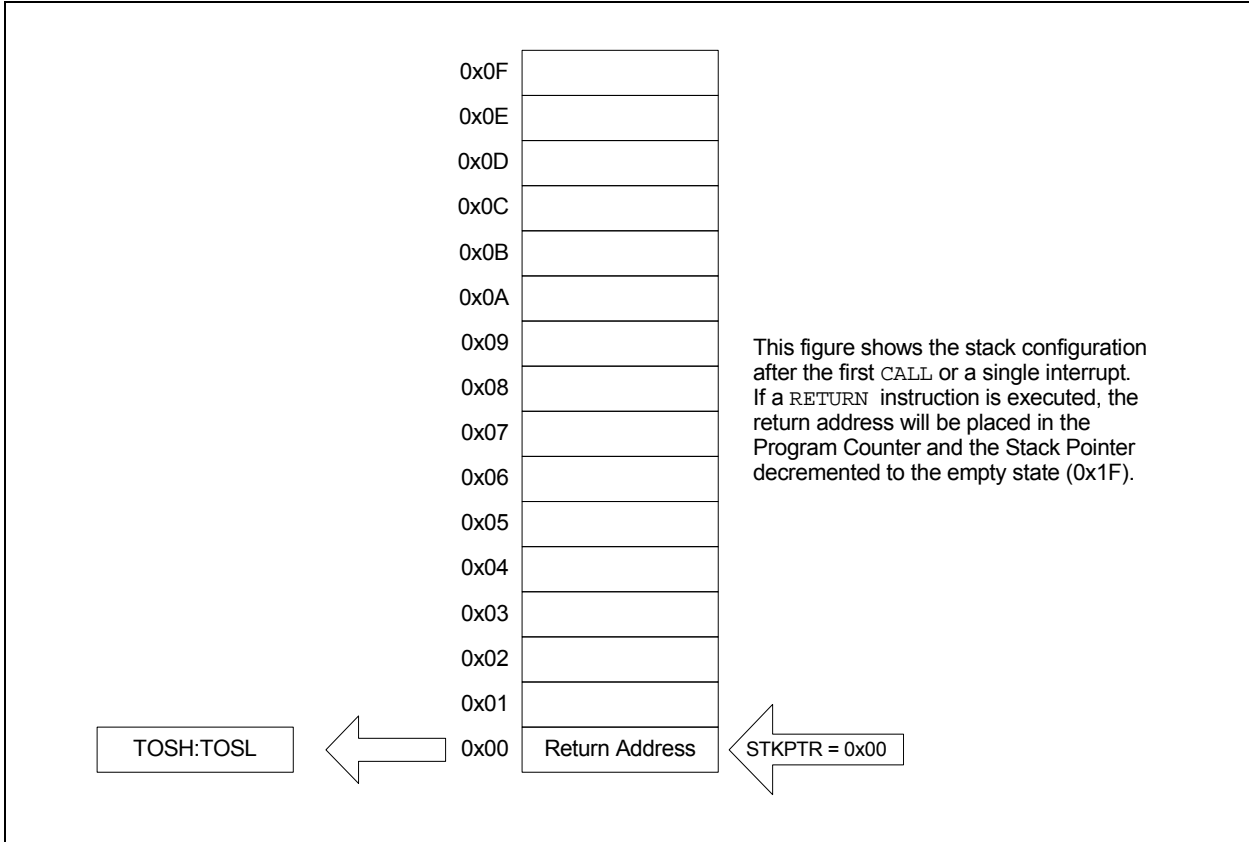
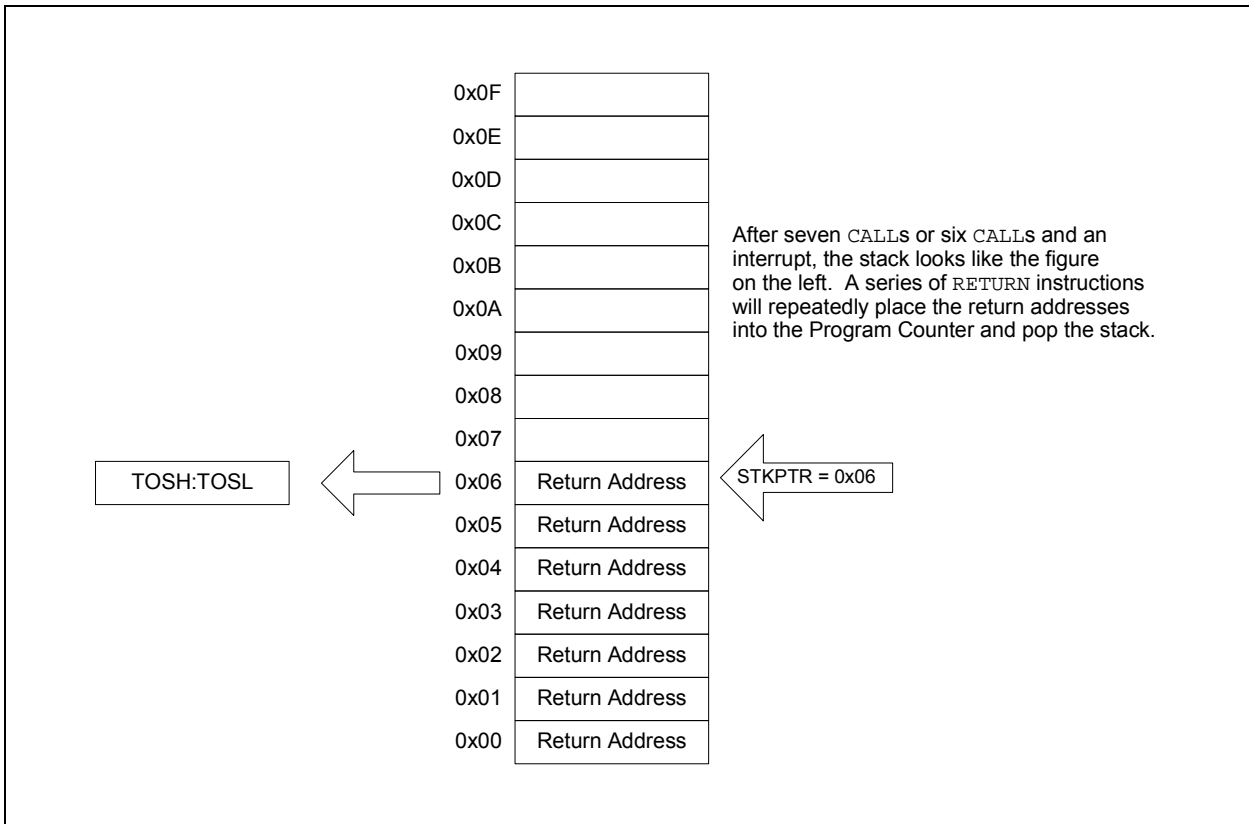
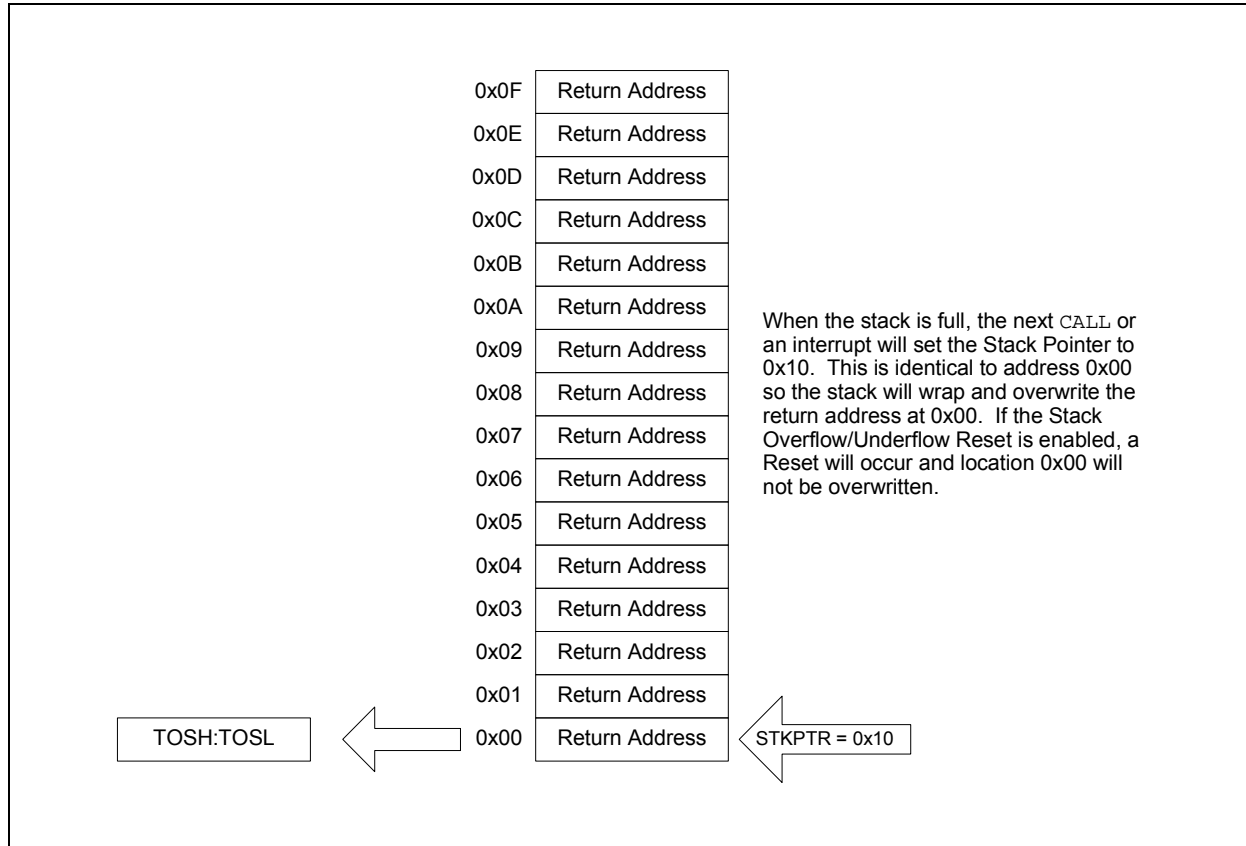


FIGURE 3-7: ACCESSING THE STACK EXAMPLE 3



PIC16(L)F1526/7

FIGURE 3-8: ACCESSING THE STACK EXAMPLE 4



3.7.2 OVERFLOW/UNDERFLOW RESET

If the `STVREN` bit in Configuration Words is programmed to '1', the device will be reset if the stack is PUSHed beyond the sixteenth level or POPed beyond the first level, setting the appropriate bits (`STKOVF` or `STKUNF`, respectively) in the `PCON` register.

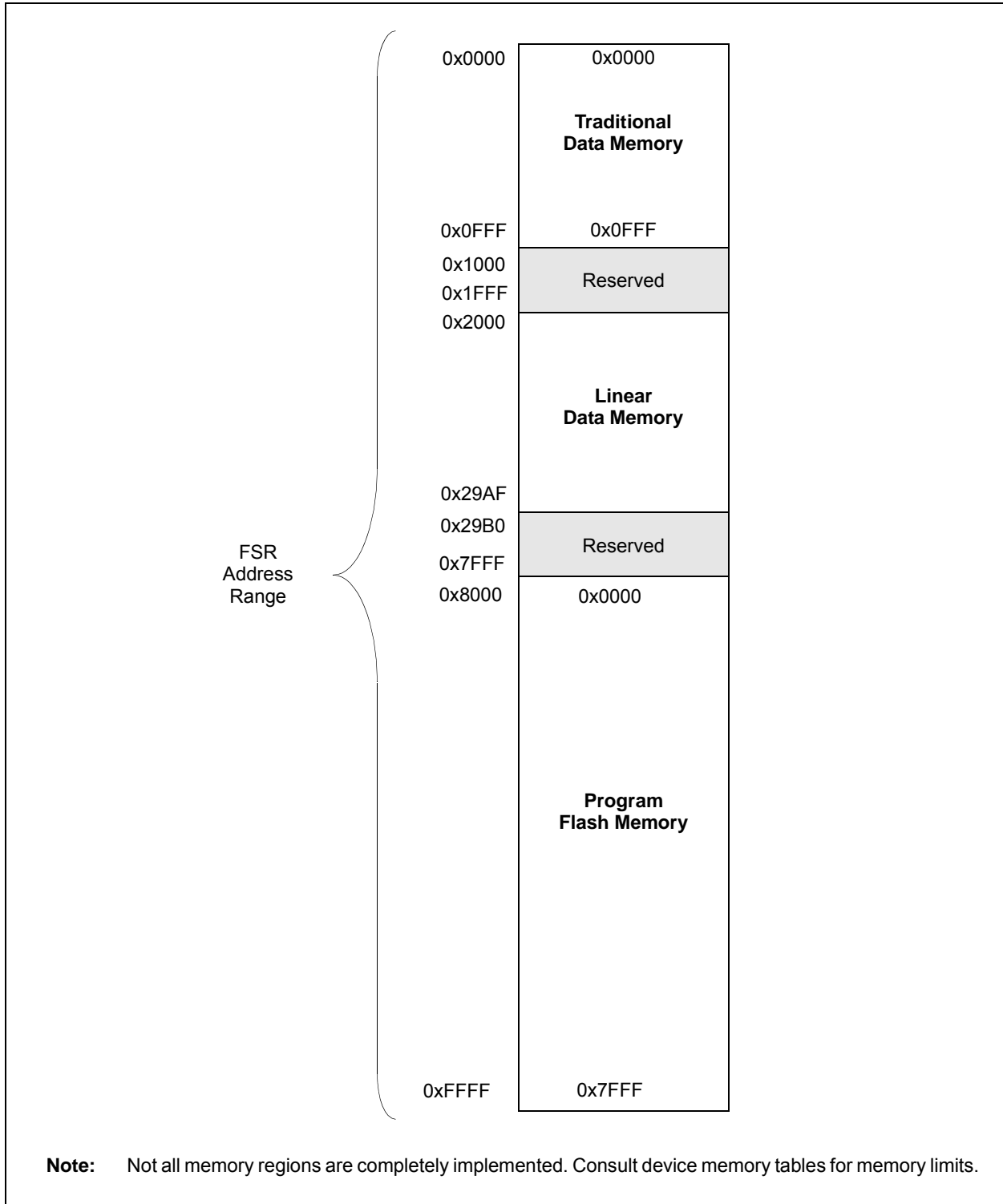
3.8 Indirect Addressing

The `INDFn` registers are not physical registers. Any instruction that accesses an `INDFn` register actually accesses the register at the address specified by the File Select Registers (`FSR`). If the `FSRn` address specifies one of the two `INDFn` registers, the read will return '0' and the write will not occur (though Status bits may be affected). The `FSRn` register value is created by the pair `FSRnH` and `FSRnL`.

The `FSR` registers form a 16-bit address that allows an addressing space with 65536 locations. These locations are divided into three memory regions:

- Traditional Data Memory
- Linear Data Memory
- Program Flash Memory

FIGURE 3-9: INDIRECT ADDRESSING

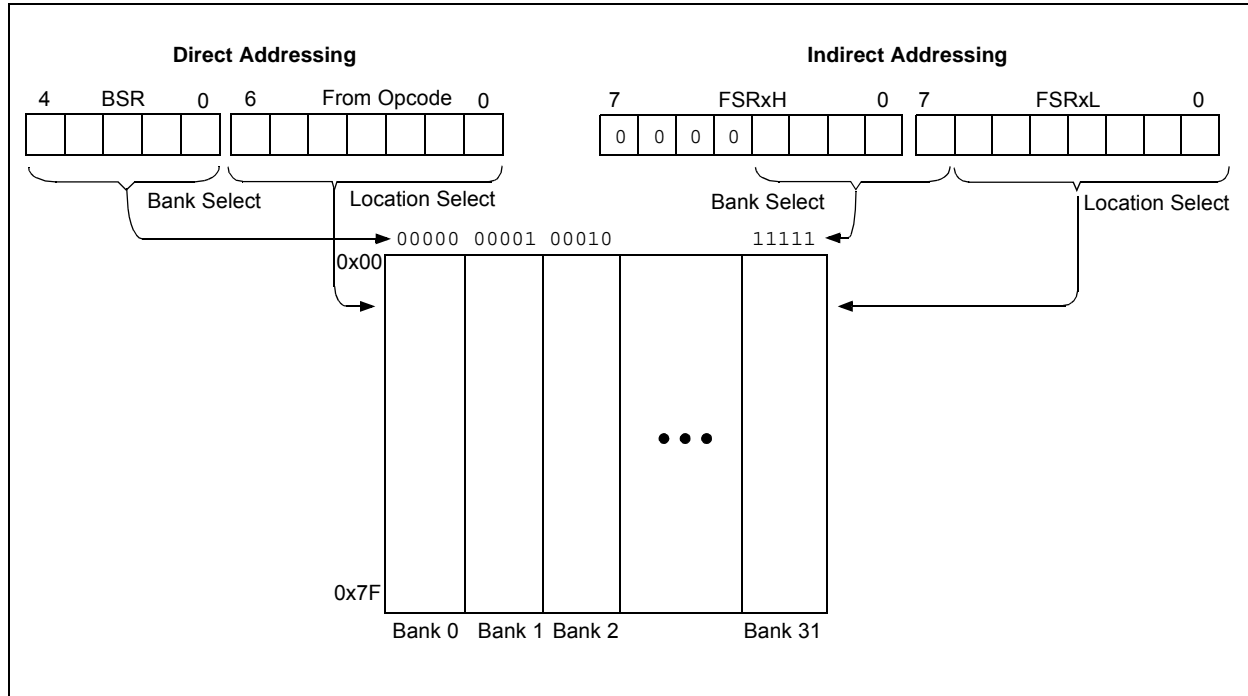


PIC16(L)F1526/7

3.8.1 TRADITIONAL DATA MEMORY

The traditional data memory is a region from FSR address 0x000 to FSR address 0xFFF. The addresses correspond to the absolute addresses of all SFR, GPR and common registers.

FIGURE 3-10: TRADITIONAL DATA MEMORY MAP



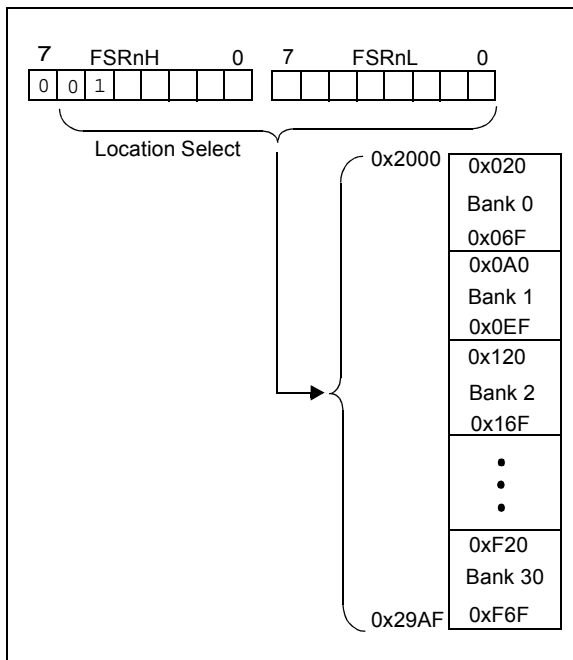
3.8.2 LINEAR DATA MEMORY

The linear data memory is the region from FSR address 0x2000 to FSR address 0x29AF. This region is a virtual region that points back to the 80-byte blocks of GPR memory in all the banks.

Unimplemented memory reads as 0x00. Use of the linear data memory region allows buffers to be larger than 80 bytes because incrementing the FSR beyond one bank will go directly to the GPR memory of the next bank.

The 16 bytes of common memory are not included in the linear data memory region.

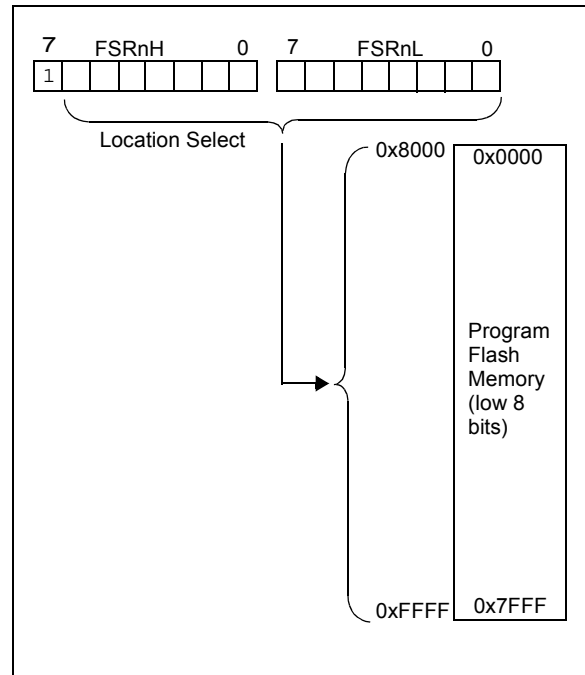
FIGURE 3-11: LINEAR DATA MEMORY MAP



3.8.3 PROGRAM FLASH MEMORY

To make constant data access easier, the entire Program Flash Memory is mapped to the upper half of the FSR address space. When the MSB of FSRnH is set, the lower 15 bits are the address in program memory which will be accessed through INDF. Only the lower 8 bits of each memory location is accessible via INDF. Writing to the Program Flash Memory cannot be accomplished via the FSR/INDF interface. All instructions that access Program Flash Memory via the FSR/INDF interface will require one additional instruction cycle to complete.

FIGURE 3-12: PROGRAM FLASH MEMORY MAP



PIC16(L)F1526/7

4.0 DEVICE CONFIGURATION

Device configuration consists of Configuration Words, Code Protection and Device ID.

4.1 Configuration Words

There are several Configuration Word bits that allow different oscillator and memory protection options. These are implemented as Configuration Word 1 at 8007h and Configuration Word 2 at 8008h.

Note: The `DEBUG` bit in Configuration Word 2 is managed automatically by device development tools including debuggers and programmers. For normal device operation, this bit should be maintained as a '1'.

4.2 Register Definitions: Configuration Words

REGISTER 4-1: CONFIG1: CONFIGURATION WORD 1

| | | | | | |
|--------|-------|----------|------------|-------|-------|
| R/P-1 | R/P-1 | R/P-1 | R/P-1 | R/P-1 | U-1 |
| FCMEN | IESO | CLKOUTEN | BOREN<1:0> | | — |
| bit 13 | | | | | bit 8 |

| | | | | | | | |
|------------------------|----------------------------|-----------------------------|-----------|-------|-----------|-------|-------|
| R/P-1 | R/P-1 | R/P-1 | R/P-1 | R/P-1 | R/P-1 | R/P-1 | R/P-1 |
| $\overline{\text{CP}}$ | MCLR $\overline{\text{E}}$ | PWRTE $\overline{\text{E}}$ | WDTE<1:0> | | FOSC<2:0> | | |
| bit 7 | | | | | | | bit 0 |

Legend:

R = Readable bit P = Programmable bit U = Unimplemented bit, read as '1'
 '0' = Bit is cleared '1' = Bit is set -n = Value when blank or after Bulk Erase

- bit 13 **FCMEN:** Fail-Safe Clock Monitor Enable bit
 1 = Fail-Safe Clock Monitor is enabled
 0 = Fail-Safe Clock Monitor is disabled
- bit 12 **IESO:** Internal External Switchover bit
 1 = Internal/External Switchover mode is enabled
 0 = Internal/External Switchover mode is disabled
- bit 11 **CLKOUTEN:** Clock Out Enable bit
If FOSC configuration bits are set to LP, XT, HS modes:
 This bit is ignored, CLKOUT function is disabled. Oscillator function on the CLKOUT pin.
All other FOSC modes:
 1 = CLKOUT function is disabled. I/O function on the CLKOUT pin.
 0 = CLKOUT function is enabled on the CLKOUT pin
- bit 10-9 **BOREN<1:0>:** Brown-out Reset Enable bits
 11 = BOR enabled
 10 = BOR enabled during operation and disabled in Sleep
 01 = BOR controlled by SBOREN bit of the BORCON register
 00 = BOR disabled
- bit 8 **Unimplemented:** Read as '1'
- bit 7 **CP:** Code Protection bit
 1 = Program memory code protection is disabled
 0 = Program memory code protection is enabled
- bit 6 **MCLR $\overline{\text{E}}$:** MCLR/VPP Pin Function Select bit
If LVP bit = 1:
 This bit is ignored.
If LVP bit = 0:
 1 = MCLR/VPP pin function is MCLR; Weak pull-up enabled.
 0 = MCLR/VPP pin function is digital input; MCLR internally disabled; Weak pull-up under control of WPUG5 bit.
- bit 5 **PWRTE $\overline{\text{E}}$:** Power-up Timer Enable bit
 1 = PWRT disabled
 0 = PWRT enabled
- bit 4-3 **WDTE<1:0>:** Watchdog Timer Enable bit
 11 = WDT enabled
 10 = WDT enabled while running and disabled in Sleep
 01 = WDT controlled by the SWDTEN bit in the WDTCON register
 00 = WDT disabled

PIC16(L)F1526/7

REGISTER 4-1: CONFIG1: CONFIGURATION WORD 1 (CONTINUED)

- bit 2-0 **FOSC<2:0>**: Oscillator Selection bits
- 111 = ECH: External Clock, High-Power mode (4-20 MHz): device clock supplied to CLKIN pin
 - 110 = ECM: External Clock, Medium-Power mode (0.5-4 MHz): device clock supplied to CLKIN pin
 - 101 = ECL: External Clock, Low-Power mode (0-0.5 MHz): device clock supplied to CLKIN pin
 - 100 = INTOSC oscillator: I/O function on CLKIN pin
 - 011 = EXTRC oscillator: External RC circuit connected to CLKIN pin
 - 010 = HS oscillator: High-speed crystal/resonator connected between OSC1 and OSC2 pins
 - 001 = XT oscillator: Crystal/resonator connected between OSC1 and OSC2 pins
 - 000 = LP oscillator: Low-power crystal connected between OSC1 and OSC2 pins

REGISTER 4-2: CONFIG2: CONFIGURATION WORD 2

| | | | | | |
|--------|-------|-------|-------|--------|-------|
| R/P-1 | R/P-1 | R/P-1 | R/P-1 | R/P-1 | U-1 |
| LVP | DEBUG | LPBOR | BORV | STVREN | — |
| bit 13 | | | | | bit 8 |

| | | | | | | | |
|-------|-----|-----|-----------------------|-----|-----|----------|-------|
| U-1 | U-1 | U-1 | R/P-1 | U-1 | U-1 | R/P-1 | R/P-1 |
| — | — | — | VCAPEN ⁽¹⁾ | — | — | WRT<1:0> | |
| bit 7 | | | | | | bit 0 | |

Legend:

R = Readable bit P = Programmable bit U = Unimplemented bit, read as '1'
 '0' = Bit is cleared '1' = Bit is set -n = Value when blank or after Bulk Erase

- bit 13 **LVP:** Low-Voltage Programming Enable bit
 1 = Low-voltage programming enabled
 0 = High-voltage on MCLR must be used for programming
- bit 12 **DEBUG:** In-Circuit Debugger Mode bit
 1 = In-Circuit Debugger disabled, ICSPCLK and ICSPDAT are general purpose I/O pins
 0 = In-Circuit Debugger enabled, ICSPCLK and ICSPDAT are dedicated to the debugger
- bit 11 **LPBOR:** Low-Power BOR bit
 1 = Low-Power BOR is disabled
 0 = Low-Power BOR is enabled
- bit 10 **BORV:** Brown-out Reset Voltage Selection bit⁽²⁾
 1 = Brown-out Reset voltage (*Vbor*), low trip point selected.
 0 = Brown-out Reset voltage (*Vbor*), high trip point selected.
- bit 9 **STVREN:** Stack Overflow/Underflow Reset Enable bit
 1 = Stack Overflow or Underflow will cause a Reset
 0 = Stack Overflow or Underflow will not cause a Reset
- bit 8-5 **Unimplemented:** Read as '1'
- bit 4 **VCAPEN:** Voltage Regulator Capacitor Enable bits⁽¹⁾
 If PIC16LF1526/7 (regulator disabled):
 These bits are ignored. All VCAP pin functions are disabled.
 If PIC16F1526/7 (regulator enabled):
 0 = VCAP functionality is enabled on RF0.
 1 = All VCAP pin functions are disabled
- bit 3-2 **Unimplemented:** Read as '1'
- bit 1-0 **WRT<1:0>:** Flash Memory Self-Write Protection bits
8 kW Flash memory (PIC16(L)F1526 only):
 11 = Write protection off
 10 = 000h to 1FFh write-protected, 200h to 1FFFh may be modified by PMCON control
 01 = 000h to FFFh write-protected, 1000h to 1FFFh may be modified by PMCON control
 00 = 000h to 1FFFh write-protected, no addresses may be modified by PMCON control
16 kW Flash memory (PIC16(L)F1527 only):
 11 = Write protection off
 10 = 000h to 1FFh write-protected, 200h to 3FFFh may be modified by PMCON control
 01 = 000h to 1FFFh write-protected, 2000h to 3FFFh may be modified by PMCON control
 00 = 000h to 3FFFh write-protected, no addresses may be modified by PMCON control

- Note 1:** PIC16F1526/7 only.
Note 2: See *Vbor* parameter for specific trip point voltages.

PIC16(L)F1526/7

4.3 Code Protection

Code protection allows the device to be protected from unauthorized access. Program memory protection is controlled independently. Internal access to the program memory is unaffected by any code protection setting.

4.3.1 PROGRAM MEMORY PROTECTION

The entire program memory space is protected from external reads and writes by the CP bit in Configuration Words. When $\overline{CP} = 0$, external reads and writes of program memory are inhibited and a read will return all '0's. The CPU can continue to read program memory, regardless of the protection bit settings. Writing the program memory is dependent upon the write protection setting. See [Section 4.4 "Write Protection"](#) for more information.

4.4 Write Protection

Write protection allows the device to be protected from unintended self-writes. Applications, such as bootloader software, can be protected while allowing other regions of the program memory to be modified.

The WRT<1:0> bits in Configuration Words define the size of the program memory block that is protected.

4.5 User ID

Four memory locations (8000h-8003h) are designated as ID locations where the user can store checksum or other code identification numbers. These locations are readable and writable during normal execution. See [Section 11.5 "User ID, Device ID and Configuration Word Access"](#) for more information on accessing these memory locations. For more information on checksum calculation, see the *"PIC16F/LF151X/152X Memory Programming Specification"* (DS41422).

4.6 Device ID and Revision ID

The memory location 8006h is where the Device ID and Revision ID are stored. The upper nine bits hold the Device ID. The lower five bits hold the Revision ID. See **Section 11.5 “User ID, Device ID and Configuration Word Access”** for more information on accessing these memory locations.

Development tools, such as device programmers and debuggers, may be used to read the Device ID and Revision ID.

REGISTER 4-3: DEVID: DEVICE ID REGISTER

| | | | | | |
|----------|---|---|-------|---|---|
| R | R | R | R | R | R |
| DEV<8:3> | | | | | |
| bit 13 | | | bit 8 | | |

| | | | | | | | |
|----------|---|---|----------|---|---|---|---|
| R | R | R | R | R | R | R | R |
| DEV<2:0> | | | REV<4:0> | | | | |
| bit 7 | | | bit 0 | | | | |

Legend:

R = Readable bit

'1' = Bit is set

'0' = Bit is cleared

U = Unimplemented bit, read as '1'

-n/n = Value at POR and BOR/Value at all other Resets

bit 13-5 **DEV<8:0>**: Device ID bits

| Device | DEVID<13:0> Values | |
|-------------|--------------------|----------|
| | DEV<8:0> | REV<4:0> |
| PIC16F1526 | 01 0101 100 | x xxxxx |
| PIC16F1527 | 01 0101 101 | x xxxxx |
| PIC16LF1526 | 01 0101 110 | x xxxxx |
| PIC16LF1527 | 01 0101 111 | x xxxxx |

bit 4-0 **REV<4:0>**: Revision ID bits

These bits are used to identify the revision (see Table under DEV<8:0> above).

PIC16(L)F1526/7

5.0 OSCILLATOR MODULE (WITH FAIL-SAFE CLOCK MONITOR)

5.1 Overview

The oscillator module has a wide variety of clock sources and selection features that allow it to be used in a wide range of applications while maximizing performance and minimizing power consumption. [Figure 5-1](#) illustrates a block diagram of the oscillator module.

Clock sources can be supplied from external oscillators, quartz crystal resonators, ceramic resonators and Resistor-Capacitor (RC) circuits. In addition, the system clock source can be supplied from one of two internal oscillators, with a choice of speeds selectable via software. Additional clock features include:

- Selectable system clock source between external or internal sources via software.
- Two-Speed Start-up mode, which minimizes latency between external oscillator start-up and code execution.
- Fail-Safe Clock Monitor (FSCM) designed to detect a failure of the external clock source (LP, XT, HS, EC or RC modes) and switch automatically to the internal oscillator.
- Oscillator Start-up Timer (OST) ensures stability of crystal oscillator sources
- Fast start-up oscillator allows internal circuits to power up and stabilize before switching to the 16 MHz HFINTOSC

The oscillator module can be configured in one of eight clock modes.

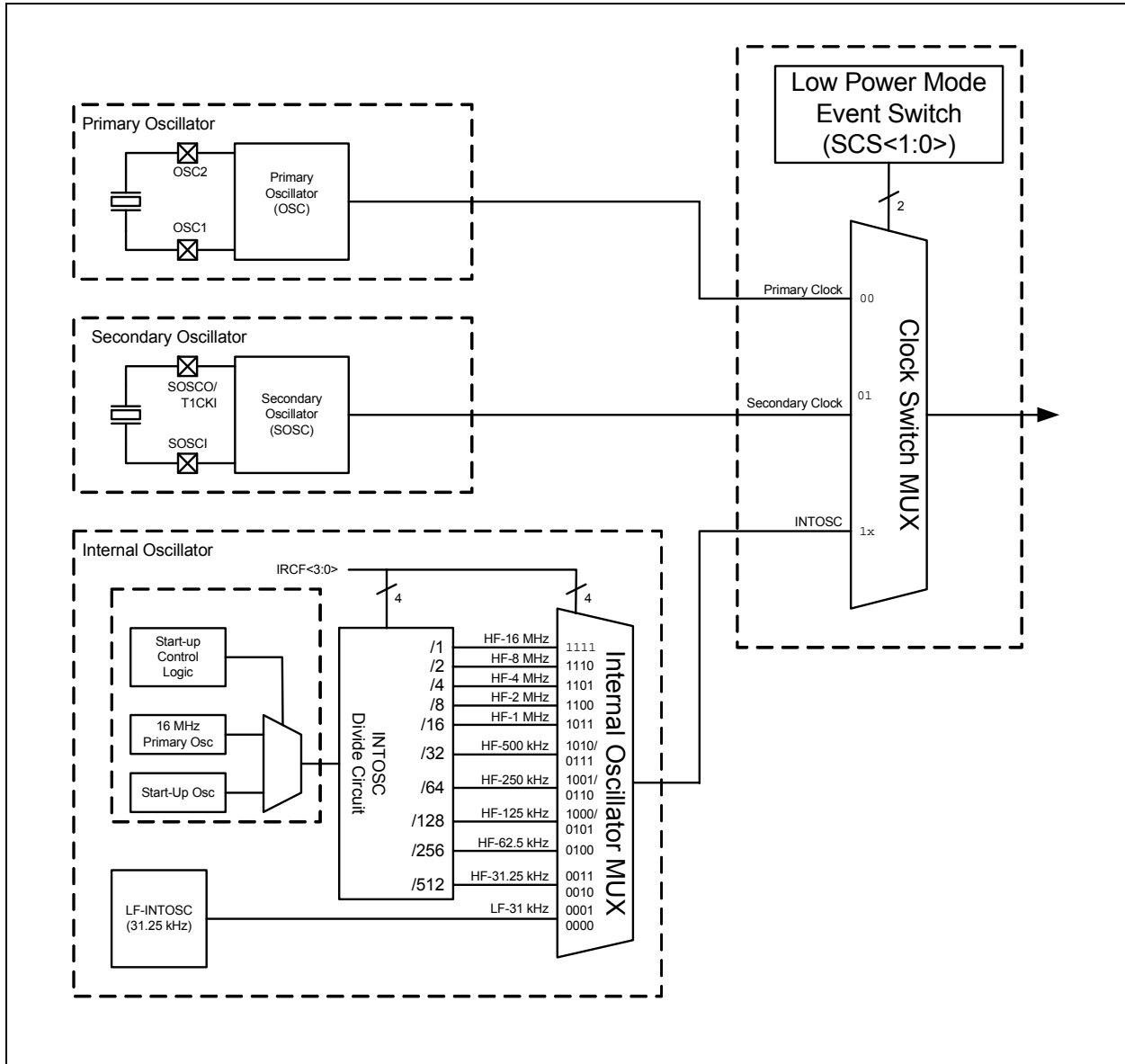
1. ECL – External Clock Low-Power mode (0 MHz to 0.5 MHz)
2. ECM – External Clock Medium-Power mode (0.5 MHz to 4 MHz)
3. ECH – External Clock High-Power mode (4 MHz to 20 MHz)
4. LP – 32 kHz Low-Power Crystal mode.
5. XT – Medium Gain Crystal or Ceramic Resonator Oscillator mode (up to 4 MHz)
6. HS – High Gain Crystal or Ceramic Resonator mode (4 MHz to 20 MHz)
7. RC – External Resistor-Capacitor (RC).
8. INTOSC – Internal oscillator (31 kHz to 16 MHz).

Clock Source modes are selected by the FOSC<2:0> bits in the Configuration Words. The FOSC bits determine the type of oscillator that will be used when the device is first powered.

The EC clock mode relies on an external logic level signal as the device clock source. The LP, XT and HS clock modes require an external crystal or resonator to be connected to the device. Each mode is optimized for a different frequency range. The RC clock mode requires an external resistor and capacitor to set the oscillator frequency.

The INTOSC internal oscillator block produces a low and high-frequency clock source, designated LFINTOSC and HFINTOSC. (see Internal Oscillator Block, [Figure 5-1](#)). A wide selection of device clock frequencies may be derived from these two clock sources.

FIGURE 5-1: SIMPLIFIED PIC® MCU CLOCK SOURCE BLOCK DIAGRAM



PIC16(L)F1526/7

5.2 Clock Source Types

Clock sources can be classified as external or internal.

External clock sources rely on external circuitry for the clock source to function. Examples are: oscillator modules (EC mode), quartz crystal resonators or ceramic resonators (LP, XT and HS modes) and Resistor-Capacitor (RC) mode circuits.

Internal clock sources are contained within the oscillator module. The internal oscillator block has two internal oscillators that are used to generate the internal system clock sources: the 16 MHz High-Frequency Internal Oscillator and the 31 kHz Low-Frequency Internal Oscillator (LFINTOSC).

The system clock can be selected between external or internal clock sources via the System Clock Select (SCS) bits in the OSCCON register. See [Section 5.3 “Clock Switching”](#) for additional information.

5.2.1 EXTERNAL CLOCK SOURCES

An external clock source can be used as the device system clock by performing one of the following actions:

- Program the FOSC<2:0> bits in the Configuration Words to select an external clock source that will be used as the default system clock upon a device Reset.
- Write the SCS<1:0> bits in the OSCCON register to switch the system clock source to:
 - Secondary oscillator during run-time, or
 - An external clock source determined by the value of the FOSC bits.

See [Section 5.3 “Clock Switching”](#) for more information.

5.2.1.1 EC Mode

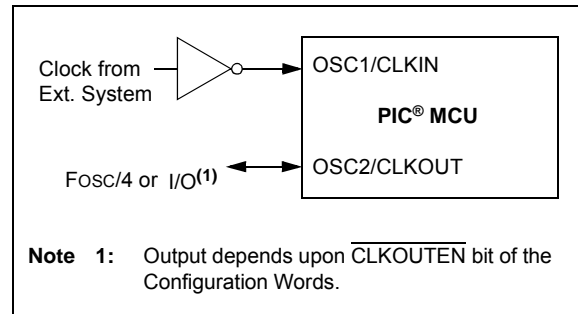
The External Clock (EC) mode allows an externally generated logic level signal to be the system clock source. When operating in this mode, an external clock source is connected to the OSC1 input. OSC2/CLKOUT is available for general purpose I/O or CLKOUT. [Figure 5-2](#) shows the pin connections for EC mode.

EC mode has three power modes to select from through Configuration Words:

- High power, 4-20 MHz (FOSC = 111)
- Medium power, 0.5-4 MHz (FOSC = 110)
- Low power, 0-0.5 MHz (FOSC = 101)

The Oscillator Start-up Timer (OST) is disabled when EC mode is selected. Therefore, there is no delay in operation after a Power-on Reset (POR) or wake-up from Sleep. Because the PIC® MCU design is fully static, stopping the external clock input will have the effect of halting the device while leaving all data intact. Upon restarting the external clock, the device will resume operation as if no time had elapsed.

FIGURE 5-2: EXTERNAL CLOCK (EC) MODE OPERATION



5.2.1.2 LP, XT, HS Modes

The LP, XT and HS modes support the use of quartz crystal resonators or ceramic resonators connected to OSC1 and OSC2 ([Figure 5-3](#)). The three modes select a low, medium or high gain setting of the internal inverter-amplifier to support various resonator types and speed.

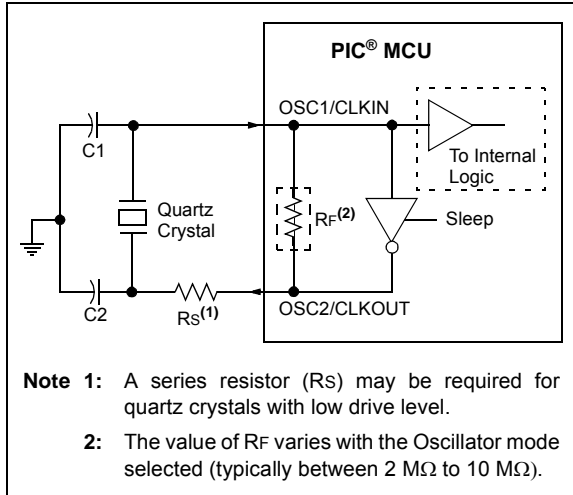
LP Oscillator mode selects the lowest gain setting of the internal inverter-amplifier. LP mode current consumption is the least of the three modes. This mode is designed to drive only 32.768 kHz tuning-fork type crystals (watch crystals).

XT Oscillator mode selects the intermediate gain setting of the internal inverter-amplifier. XT mode current consumption is the medium of the three modes. This mode is best suited to drive resonators with a medium drive level specification.

HS Oscillator mode selects the highest gain setting of the internal inverter-amplifier. HS mode current consumption is the highest of the three modes. This mode is best suited for resonators that require a high drive setting.

[Figure 5-3](#) and [Figure 5-4](#) show typical circuits for quartz crystal and ceramic resonators, respectively.

FIGURE 5-3: QUARTZ CRYSTAL OPERATION (LP, XT OR HS MODE)

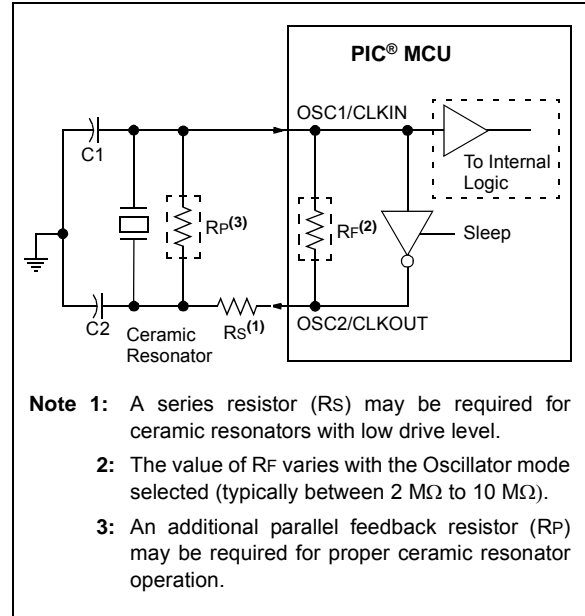


Note 1: Quartz crystal characteristics vary according to type, package and manufacturer. The user should consult the manufacturer data sheets for specifications and recommended application.

- 2:** Always verify oscillator performance over the V_{DD} and temperature range that is expected for the application.
- 3:** For oscillator design assistance, reference the following Microchip Applications Notes:

- AN826, “Crystal Oscillator Basics and Crystal Selection for *rPIC*[®] and *PIC*[®] Devices” (DS00826)
- AN849, “Basic *PIC*[®] Oscillator Design” (DS00849)
- AN943, “Practical *PIC*[®] Oscillator Analysis and Design” (DS00943)
- AN949, “Making Your Oscillator Work” (DS00949)

FIGURE 5-4: CERAMIC RESONATOR OPERATION (XT OR HS MODE)



5.2.1.3 Oscillator Start-up Timer (OST)

If the oscillator module is configured for LP, XT or HS modes, the Oscillator Start-up Timer (OST) counts 1024 oscillations from OSC1. This occurs following a Power-on Reset (POR) and when the Power-up Timer (PWRT) has expired (if configured), or a wake-up from Sleep. During this time, the program counter does not increment and program execution is suspended, unless either FSCM or Two-Speed Start-Up are enabled. The OST ensures that the oscillator circuit, using a quartz crystal resonator or ceramic resonator, has started and is providing a stable system clock to the oscillator module.

In order to minimize latency between external oscillator start-up and code execution, the Two-Speed Clock Start-up mode can be selected (see [Section 5.4 “Two-Speed Clock Start-up Mode”](#)).

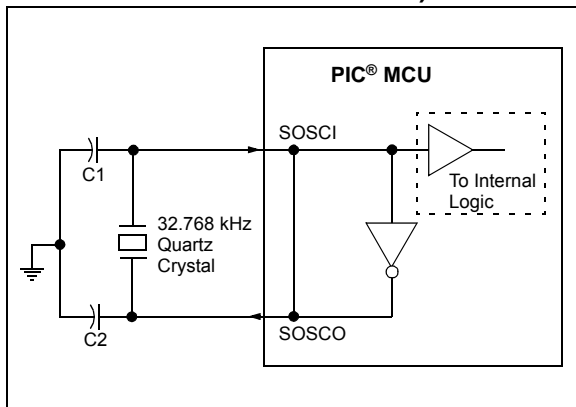
PIC16(L)F1526/7

5.2.1.4 Secondary Oscillator

The secondary oscillator is a separate crystal oscillator that is associated with the Timer1 peripheral. It is optimized for timekeeping operations with a 32.768 kHz crystal connected between the SOSCO and SOSCI device pins.

The secondary oscillator can be used as an alternate system clock source and can be selected during run-time using clock switching. Refer to [Section 5.3 "Clock Switching"](#) for more information.

FIGURE 5-5: QUARTZ CRYSTAL OPERATION (SECONDARY OSCILLATOR)



Note 1: Quartz crystal characteristics vary according to type, package and manufacturer. The user should consult the manufacturer data sheets for specifications and recommended application.

2: Always verify oscillator performance over the VDD and temperature range that is expected for the application.

3: For oscillator design assistance, reference the following Microchip Applications Notes:

- AN826, "Crystal Oscillator Basics and Crystal Selection for rPIC® and PIC® Devices" (DS00826)
- AN849, "Basic PIC® Oscillator Design" (DS00849)
- AN943, "Practical PIC® Oscillator Analysis and Design" (DS00943)
- AN949, "Making Your Oscillator Work" (DS00949)
- TB097, "Interfacing a Micro Crystal MS1V-T1K 32.768 kHz Tuning Fork Crystal to a PIC16F690/SS" (DS91097)
- AN1288, "Design Practices for Low-Power External Oscillators" (DS01288)

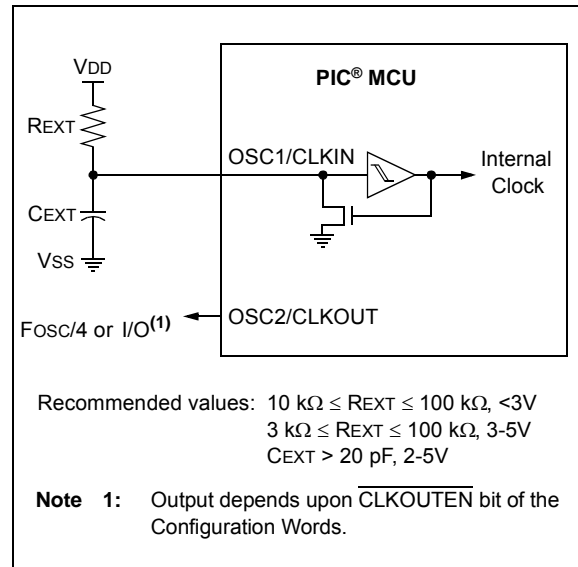
5.2.1.5 External RC Mode

The external Resistor-Capacitor (RC) modes support the use of an external RC circuit. This allows the designer maximum flexibility in frequency choice while keeping costs to a minimum when clock accuracy is not required.

The RC circuit connects to OSC1. OSC2/CLKOUT is available for general purpose I/O or CLKOUT. The function of the OSC2/CLKOUT pin is determined the CLKOUTEN bit in Configuration Words.

[Figure 5-6](#) shows the external RC mode connections.

FIGURE 5-6: EXTERNAL RC MODES



The RC oscillator frequency is a function of the supply voltage, the resistor (R_{EXT}) and capacitor (C_{EXT}) values and the operating temperature. Other factors affecting the oscillator frequency are:

- threshold voltage variation
- component tolerances
- packaging variations in capacitance

The user also needs to take into account variation due to tolerance of the external RC components used.

5.2.2 INTERNAL CLOCK SOURCES

The device may be configured to use the internal oscillator block as the system clock by performing one of the following actions:

- Program the FOSC<2:0> bits in Configuration Words to select the INTOSC clock source, which will be used as the default system clock upon a device Reset.
- Write the SCS<1:0> bits in the OSCCON register to switch the system clock source to the internal oscillator during run-time. See [Section 5.3 “Clock Switching”](#) for more information.

In **INTOSC** mode, OSC1/CLKIN is available for general purpose I/O. OSC2/CLKOUT is available for general purpose I/O or CLKOUT.

The function of the OSC2/CLKOUT pin is determined by the $\overline{\text{CLKOUTEN}}$ bit in Configuration Words.

The internal oscillator block has two independent oscillators that provides the internal system clock source.

1. The **HFINTOSC** (High-Frequency Internal Oscillator) is factory calibrated and operates at 16 MHz.
2. The **LFINTOSC** (Low-Frequency Internal Oscillator) is uncalibrated and operates at 31 kHz.

5.2.2.1 HFINTOSC

The High-Frequency Internal Oscillator (HFINTOSC) is a factory calibrated 16 MHz internal clock source.

The output of the HFINTOSC connects to a postscaler and multiplexer (see [Figure 5-1](#)). The frequency derived from the HFINTOSC can be selected via software using the IRCF<3:0> bits of the OSCCON register. See [Section 5.2.2.4 “Internal Oscillator Clock Switch Timing”](#) for more information.

The HFINTOSC is enabled by:

- Configure the IRCF<3:0> bits of the OSCCON register for the desired HF frequency, and
- FOSC<2:0> = 100, or
- Set the System Clock Source (SCS) bits of the OSCCON register to '1x'.

A fast start-up oscillator allows internal circuits to power-up and stabilize before switching to HFINTOSC.

The High-Frequency Internal Oscillator Ready bit (HFIOFR) of the OSCSTAT register indicates when the HFINTOSC is running.

The High-Frequency Internal Oscillator Stable bit (HFIOFS) of the OSCSTAT register indicates when the HFINTOSC is running within 0.5% of its final value.

5.2.2.2 LFINTOSC

The Low-Frequency Internal Oscillator (LFINTOSC) is an uncalibrated 31 kHz internal clock source.

The output of the LFINTOSC connects to a multiplexer (see [Figure 5-1](#)). Select 31 kHz, via software, using the IRCF<3:0> bits of the OSCCON register. See [Section 5.2.2.4 “Internal Oscillator Clock Switch Timing”](#) for more information. The LFINTOSC is also the frequency for the Power-up Timer (PWRT), Watchdog Timer (WDT) and Fail-Safe Clock Monitor (FSCM).

The LFINTOSC is enabled by selecting 31 kHz (IRCF<3:0> bits of the OSCCON register = 000) as the system clock source (SCS bits of the OSCCON register = 1x), or when any of the following are enabled:

- Configure the IRCF<3:0> bits of the OSCCON register for the desired LF frequency, and
- FOSC<2:0> = 100, or
- Set the System Clock Source (SCS) bits of the OSCCON register to '1x'

Peripherals that use the LFINTOSC are:

- Power-up Timer (PWRT)
- Watchdog Timer (WDT)
- Fail-Safe Clock Monitor (FSCM)

The Low-Frequency Internal Oscillator Ready bit (LFIOFR) of the OSCSTAT register indicates when the LFINTOSC is running.

PIC16(L)F1526/7

5.2.2.3 Internal Oscillator Frequency Selection

The system clock speed can be selected via software using the Internal Oscillator Frequency Select bits IRCF<3:0> of the OSCCON register.

The outputs of the 16 MHz HFINTOSC and LFINTOSC connects to a postscaler and multiplexer (see [Figure 5-1](#)). The Internal Oscillator Frequency Select bits IRCF<3:0> of the OSCCON register select the frequency output of the internal oscillators. One of the following frequencies can be selected via software:

- 16 MHz
- 8 MHz
- 4 MHz
- 2 MHz
- 1 MHz
- 500 kHz (default after Reset)
- 250 kHz
- 125 kHz
- 62.5 kHz
- 31.25 kHz
- 31 kHz (LFINTOSC)

Note: Following any Reset, the IRCF<3:0> bits of the OSCCON register are set to '0111' and the frequency selection is set to 500 kHz. The user can modify the IRCF bits to select a different frequency.

The IRCF<3:0> bits of the OSCCON register allow duplicate selections for some frequencies. These duplicate choices can offer system design trade-offs. Lower power consumption can be obtained when changing oscillator sources for a given frequency. Faster transition times can be obtained between frequency changes that use the same oscillator source.

5.2.2.4 Internal Oscillator Clock Switch Timing

When switching between the HFINTOSC and the LFINTOSC, the new oscillator may already be shut down to save power (see [Figure 5-7](#)). If this is the case, there is a delay after the IRCF<3:0> bits of the OSCCON register are modified before the frequency selection takes place. The OSCSTAT register will reflect the current active status of the HFINTOSC and LFINTOSC oscillators. The sequence of a frequency selection is as follows:

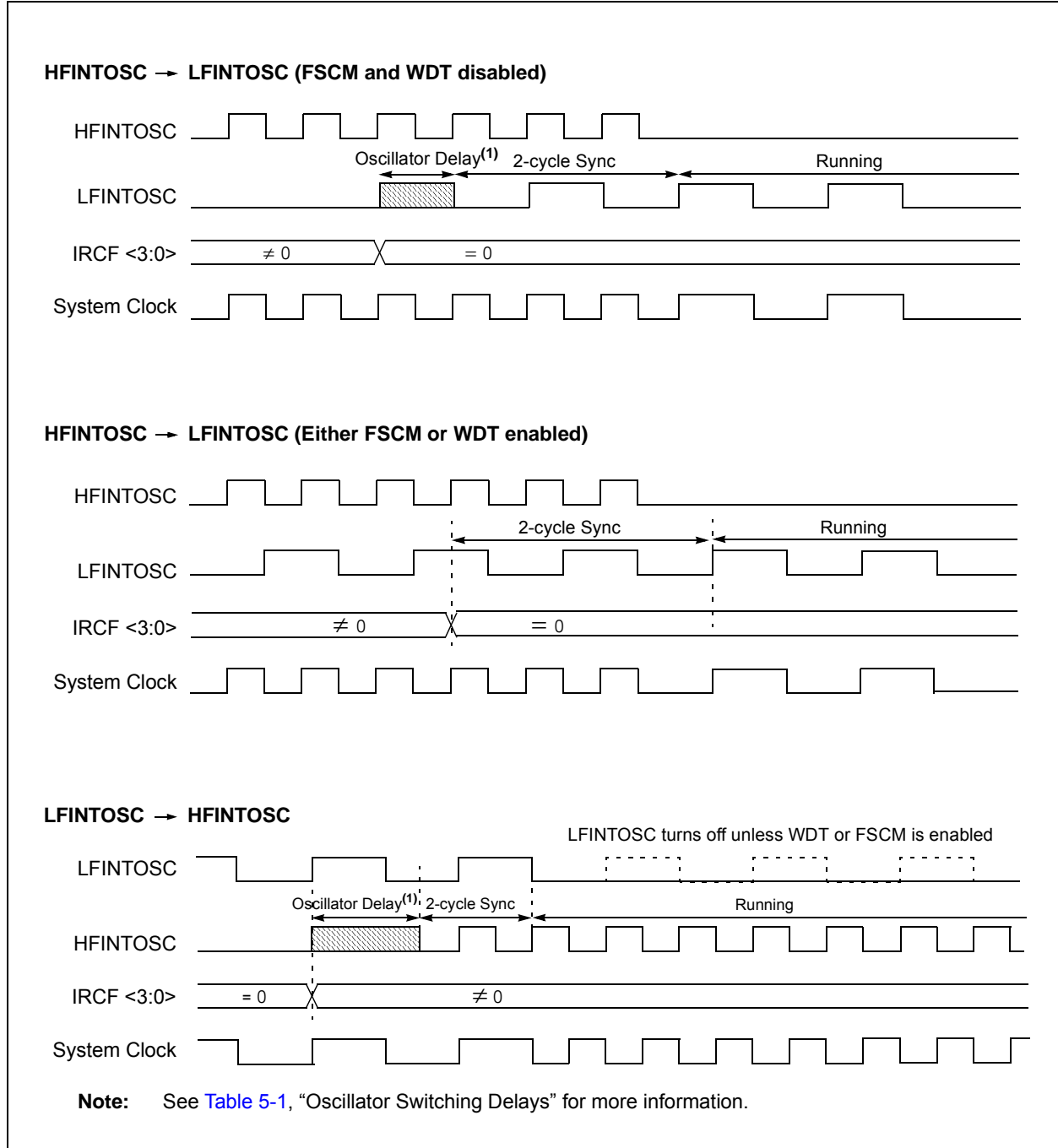
1. IRCF<3:0> bits of the OSCCON register are modified.
2. If the new clock is shut down, a clock start-up delay is started.
3. Clock switch circuitry waits for a falling edge of the current clock.
4. The current clock is held low and the clock switch circuitry waits for a rising edge in the new clock.
5. The new clock is now active.
6. The OSCSTAT register is updated as required.
7. Clock switch is complete.

See [Figure 5-7](#) for more details.

If the internal oscillator speed is switched between two clocks of the same source, there is no start-up delay before the new frequency is selected. Clock switching time delays are shown in [Table 5-1](#).

Start-up delay specifications are located in the oscillator tables of [Section 25.0 "Electrical Specifications"](#)

FIGURE 5-7: INTERNAL OSCILLATOR SWITCH TIMING



PIC16(L)F1526/7

5.3 Clock Switching

The system clock source can be switched between external and internal clock sources via software using the System Clock Select (SCS) bits of the OSCCON register. The following clock sources can be selected using the SCS bits:

- Default system oscillator determined by FOSC bits in Configuration Words
- Secondary oscillator 32 kHz crystal
- Internal Oscillator Block (INTOSC)

5.3.1 SYSTEM CLOCK SELECT (SCS) BITS

The System Clock Select (SCS) bits of the OSCCON register selects the system clock source that is used for the CPU and peripherals.

- When the SCS bits of the OSCCON register = 00, the system clock source is determined by value of the FOSC<2:0> bits in the Configuration Words.
- When the SCS bits of the OSCCON register = 01, the system clock source is the secondary oscillator.
- When the SCS bits of the OSCCON register = 1x, the system clock source is chosen by the internal oscillator frequency selected by the IRCF<3:0> bits of the OSCCON register. After a Reset, the SCS bits of the OSCCON register are always cleared.

Note: Any automatic clock switch, which may occur from Two-Speed Start-up or Fail-Safe Clock Monitor, does not update the SCS bits of the OSCCON register. The user can monitor the OSTS bit of the OSCSTAT register to determine the current system clock source.

When switching between clock sources, a delay is required to allow the new clock to stabilize. These oscillator delays are shown in [Table 5-1](#).

5.3.2 OSCILLATOR START-UP TIMER STATUS (OSTS) BIT

The Oscillator Start-up Timer Status (OSTS) bit of the OSCSTAT register indicates whether the system clock is running from the external clock source, as defined by the FOSC<2:0> bits in the Configuration Words, or from the internal clock source. In particular, OSTS indicates that the Oscillator Start-up Timer (OST) has timed out for LP, XT or HS modes. The OST does not reflect the status of the secondary oscillator.

5.3.3 SECONDARY OSCILLATOR

The secondary oscillator is a separate crystal oscillator associated with the Timer1 peripheral. It is optimized for timekeeping operations with a 32.768 kHz crystal connected between the SOSCO and SOSCI device pins.

The secondary oscillator is enabled using the SOSSEN control bit in the TxCON register. See [Section 18.0 “Timer1/3/5 Module with Gate Control”](#) for more information about the Timer1 peripheral.

5.3.4 SECONDARY OSCILLATOR READY (SOSCR) BIT

The user must ensure that the secondary oscillator is ready to be used before it is selected as a system clock source. The Secondary Oscillator Ready (SOSCR) bit of the OSCSTAT register indicates whether the secondary oscillator is ready to be used. After the SOSCR bit is set, the SCS bits can be configured to select the secondary oscillator.

5.3.5 CLOCK SWITCHING BEFORE SLEEP

When clock switching from an old clock to a new clock, prior to entering Sleep mode, it is necessary to confirm that the switch is complete before the Sleep instruction is executed. Failure to do so may result in an incomplete switch and consequential loss of the system clock altogether. Clock switching is confirmed by monitoring the clock status bits in the OSCSTAT register. Switch confirmation can be accomplished by sensing that the ready bit for the new clock is set or the ready bit for the old clock is cleared. For example, when switching between the internal oscillator with the PLL and the internal oscillator without the PLL, monitor the PLLR bit. When PLLR is set, the switch to 32 MHz operation is complete. Conversely, when PLLR is cleared the switch from the 32 MHz operation to the selected internal clock is complete.

5.4 Two-Speed Clock Start-up Mode

Two-Speed Start-up mode provides additional power savings by minimizing the latency between external oscillator start-up and code execution. In applications that make heavy use of the Sleep mode, Two-Speed Start-up will remove the external oscillator start-up time from the time spent awake and can reduce the overall power consumption of the device. This mode allows the application to wake-up from Sleep, perform a few instructions using the INTOSC internal oscillator block as the clock source and go back to Sleep without waiting for the external oscillator to become stable.

Two-Speed Start-up provides benefits when the oscillator module is configured for LP, XT or HS modes. The Oscillator Start-up Timer (OST) is enabled for these modes and must count 1024 oscillations before the oscillator can be used as the system clock source.

If the oscillator module is configured for any mode other than LP, XT or HS mode, then Two-Speed Start-up is disabled. This is because the external clock oscillator does not require any stabilization time after POR or an exit from Sleep.

If the OST count reaches 1024 before the device enters Sleep mode, the OSTS bit of the OSCSTAT register is set and program execution switches to the external oscillator. However, the system may never operate from the external oscillator if the time spent awake is very short.

Note: Executing a SLEEP instruction will abort the oscillator start-up time and will cause the OSTS bit of the OSCSTAT register to remain clear.

5.4.1 TWO-SPEED START-UP MODE CONFIGURATION

Two-Speed Start-up mode is configured by the following settings:

- IESO (of the Configuration Words) = 1; Internal/External Switchover bit (Two-Speed Start-up mode enabled).
- SCS (of the OSCCON register) = 00.
- FOSC<2:0> bits in the Configuration Words configured for LP, XT or HS mode.

Two-Speed Start-up mode is entered after:

- Power-on Reset (POR) and, if enabled, after Power-up Timer (PWRT) has expired, or
- Wake-up from Sleep.

TABLE 5-1: OSCILLATOR SWITCHING DELAYS

| Switch From | Switch To | Oscillator Delay |
|------------------|----------------------|----------------------------------|
| Any clock source | LFINTOSC | One cycle of each clock source |
| | HFINTOSC | 2 μ s (approx.) |
| | ECH, ECM, ECL, EXTRC | 2 cycles |
| | LP, XT, HS | 1024 Clock Cycles (OST) |
| | Secondary Oscillator | 1024 Secondary Oscillator Cycles |

PIC16(L)F1526/7

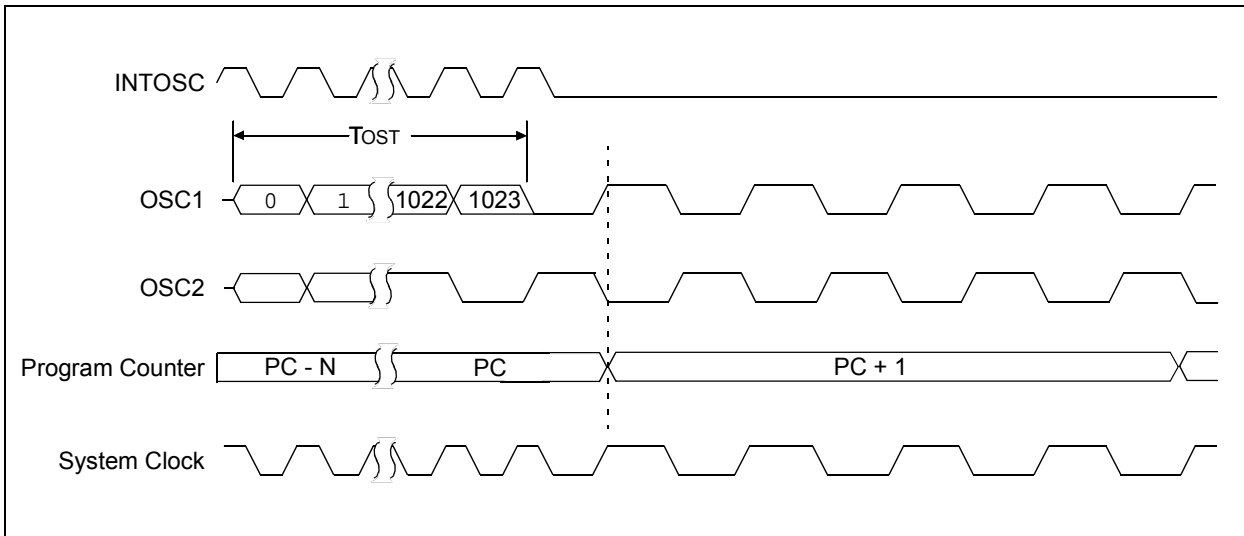
5.4.2 TWO-SPEED START-UP SEQUENCE

1. Wake-up from Power-on Reset or Sleep.
2. Instructions begin execution by the internal oscillator at the frequency set in the IRCF<3:0> bits of the OSCCON register.
3. OST enabled to count 1024 clock cycles.
4. OST timed out, wait for falling edge of the internal oscillator.
5. OSTS is set.
6. System clock held low until the next falling edge of new clock (LP, XT or HS mode).
7. System clock is switched to external clock source.

5.4.3 CHECKING TWO-SPEED CLOCK STATUS

Checking the state of the OSTS bit of the OSCSTAT register will confirm if the microcontroller is running from the external clock source, as defined by the FOSC<2:0> bits in the Configuration Words, or the internal oscillator.

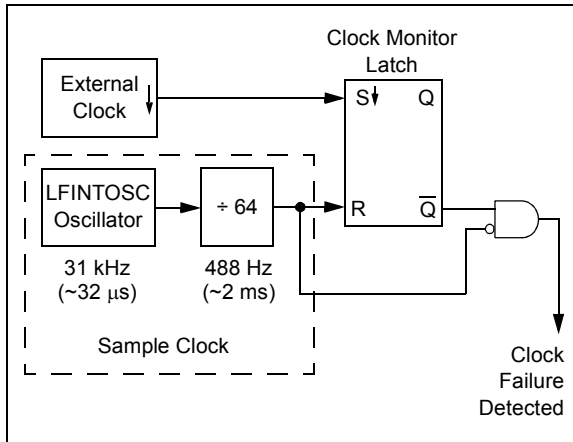
FIGURE 5-8: TWO-SPEED START-UP



5.5 Fail-Safe Clock Monitor

The Fail-Safe Clock Monitor (FSCM) allows the device to continue operating should the external oscillator fail. The FSCM can detect oscillator failure any time after the Oscillator Start-up Timer (OST) has expired. The FSCM is enabled by setting the FCMEN bit in the Configuration Words. The FSCM is applicable to all external Oscillator modes (LP, XT, HS, EC, RC and secondary oscillator).

FIGURE 5-9: FSCM BLOCK DIAGRAM



5.5.1 FAIL-SAFE DETECTION

The FSCM module detects a failed oscillator by comparing the external oscillator to the FSCM sample clock. The sample clock is generated by dividing the LFINTOSC by 64. See [Figure 5-9](#). Inside the fail detector block is a latch. The external clock sets the latch on each falling edge of the external clock. The sample clock clears the latch on each rising edge of the sample clock. A failure is detected when an entire half-cycle of the sample clock elapses before the external clock goes low.

5.5.2 FAIL-SAFE OPERATION

When the external clock fails, the FSCM switches the device clock to an internal clock source and sets the bit flag OSFIF of the PIR2 register. Setting this flag will generate an interrupt if the OSFIE bit of the PIE2 register is also set. The device firmware can then take steps to mitigate the problems that may arise from a failed clock. The system clock will continue to be sourced from the internal clock source until the device firmware successfully restarts the external oscillator and switches back to external operation.

The internal clock source chosen by the FSCM is determined by the IRCF<3:0> bits of the OSCCON register. This allows the internal oscillator to be configured before a failure occurs.

5.5.3 FAIL-SAFE CONDITION CLEARING

The Fail-Safe condition is cleared after a Reset, executing a `SLEEP` instruction or changing the SCS bits of the OSCCON register. When the SCS bits are changed, the OST is restarted. While the OST is running, the device continues to operate from the INTOSC selected in OSCCON. When the OST times out, the Fail-Safe condition is cleared after successfully switching to the external clock source. The OSFIF bit should be cleared prior to switching to the external clock source. If the Fail-Safe condition still exists, the OSFIF flag will again become set by hardware.

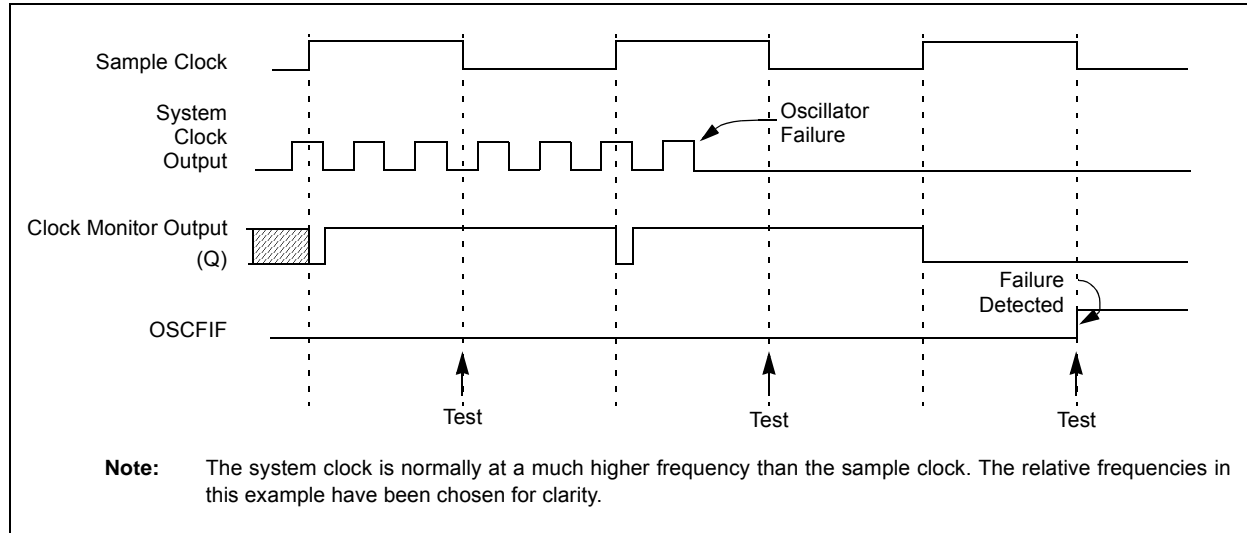
5.5.4 RESET OR WAKE-UP FROM SLEEP

The FSCM is designed to detect an oscillator failure after the Oscillator Start-up Timer (OST) has expired. The OST is used after waking up from Sleep and after any type of Reset. The OST is not used with the EC or RC Clock modes so that the FSCM will be active as soon as the Reset or wake-up has completed. When the FSCM is enabled, the Two-Speed Start-up is also enabled. Therefore, the device will always be executing code while the OST is operating.

Note: Due to the wide range of oscillator start-up times, the Fail-Safe circuit is not active during oscillator start-up (i.e., after exiting Reset or Sleep). After an appropriate amount of time, the user should check the Status bits in the OSCSTAT register to verify the oscillator start-up and that the system clock switchover has successfully completed.

PIC16(L)F1526/7

FIGURE 5-10: FSCM TIMING DIAGRAM



5.6 Register Definitions: Oscillator Control

REGISTER 5-1: OSCCON: OSCILLATOR CONTROL REGISTER

| | | | | | | | |
|-------|-----------|---------|---------|---------|----------|---------|---------|
| U-0 | R/W-0/0 | R/W-1/1 | R/W-1/1 | R/W-1/1 | U-0 | R/W-0/0 | R/W-0/0 |
| — | IRCF<3:0> | | | — | SCS<1:0> | | |
| bit 7 | | | | bit 0 | | | |

Legend:

| | | |
|----------------------|----------------------|---|
| R = Readable bit | W = Writable bit | U = Unimplemented bit, read as '0' |
| u = Bit is unchanged | x = Bit is unknown | -n/n = Value at POR and BOR/Value at all other Resets |
| '1' = Bit is set | '0' = Bit is cleared | |

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

bit 6-3 **IRCF<3:0>:** Internal Oscillator Frequency Select bits

| |
|-------------------------------------|
| 1111 = 16 MHz |
| 1110 = 8 MHz |
| 1101 = 4 MHz |
| 1100 = 2 MHz |
| 1011 = 1 MHz |
| 1010 = 500 kHz ⁽¹⁾ |
| 1001 = 250 kHz ⁽¹⁾ |
| 1000 = 125 kHz ⁽¹⁾ |
| 0111 = 500 kHz (default upon Reset) |
| 0110 = 250 kHz |
| 0101 = 125 kHz |
| 0100 = 62.5 kHz |
| 001x = 31.25 kHz |
| 000x = 31 kHz LF |

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

bit 1-0 **SCS<1:0>:** System Clock Select bits

| |
|--|
| 1x = Internal oscillator block |
| 01 = Secondary oscillator |
| 00 = Clock determined by FOSC<2:0> in Configuration Words. |

Note 1: Duplicate frequency derived from HFINTOSC.

PIC16(L)F1526/7

REGISTER 5-2: OSCSTAT: OSCILLATOR STATUS REGISTER

| | | | | | | | |
|-------|-----|-------|--------|-----|-----|--------|--------|
| R-1/q | U-0 | R-q/q | R-0/q | U-0 | U-0 | R-0/0 | R-0/q |
| SOSCR | — | OSTS | HFIOFR | — | — | LFIOFR | HFIOFS |
| bit 7 | | | | | | | bit 0 |

Legend:

R = Readable bit W = Writable bit U = Unimplemented bit, read as '0'
u = Bit is unchanged x = Bit is unknown -n/n = Value at POR and BOR/Value at all other Resets
'1' = Bit is set '0' = Bit is cleared q = Conditional

- bit 7 **SOSCR:** Secondary Oscillator Ready bit
 If SOSREN = 1:
 1 = Secondary oscillator is ready
 0 = Secondary oscillator is not ready
 If SOSREN = 0:
 1 = Timer1 clock source is always ready
- bit 6 **Unimplemented:** Read as '0'
- bit 5 **OSTS:** Oscillator Start-up Timer Status bit
 1 = Running from the clock defined by the FOSC<2:0> bits of the Configuration Words
 0 = Running from an internal oscillator (FOSC<2:0> = 100)
- bit 4 **HFIOFR:** High-Frequency Internal Oscillator Ready bit
 1 = HFINTOSC is ready
 0 = HFINTOSC is not ready
- bit 3-2 **Unimplemented:** Read as '0'
- bit 1 **LFIOFR:** Low-Frequency Internal Oscillator Ready bit
 1 = LFINTOSC is ready
 0 = LFINTOSC is not ready
- bit 0 **HFIOFS:** High-Frequency Internal Oscillator Stable bit
 1 = HFINTOSC 16 MHz Oscillator is stable and is driving the INTOSC
 0 = HFINTOSC 16 MHz is not stable, the Start-up Oscillator is driving INTOSC

TABLE 5-2: SUMMARY OF REGISTERS ASSOCIATED WITH CLOCK SOURCES

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Register on Page |
|---------|-------------|-----------|-------------|--------|--------|----------|--------|--------|------------------|
| OSCCON | — | IRCF<3:0> | | | — | SCS<1:0> | | | 61 |
| OSCSTAT | SOSCR | — | OSTS | HFIOFR | — | — | LFIOFR | HFIOFS | 62 |
| PIE2 | OSFIE | TMR5GIE | TMR3GIE | — | BCL1IE | TMR10IE | TMR8IE | CCP2IE | 78 |
| PIR2 | OSFIF | TMR5GIF | TMR3GIF | — | BCL1IF | TMR10IF | TMR8IF | CCP2IF | 82 |
| T1CON | TMR1CS<1:0> | | T1CKPS<1:0> | | SOSCEN | T1SYNC | — | TMR1ON | 168 |

Legend: — = unimplemented location, read as '0'. Shaded cells are not used by clock sources.

TABLE 5-3: SUMMARY OF CONFIGURATION WORD WITH CLOCK SOURCES

| Name | Bits | Bit -/7 | Bit -/6 | Bit 13/5 | Bit 12/4 | Bit 11/3 | Bit 10/2 | Bit 9/1 | Bit 8/0 | Register on Page |
|---------|------|---------|---------|----------|-----------|----------|------------|---------|---------|------------------|
| CONFIG1 | 13:8 | — | | FCMEN | IESO | CLKOUTEN | BOREN<1:0> | — | | 43 |
| | 7:0 | CP | MCLRE | PWRTE | WDTE<1:0> | | FOSC<2:0> | | | |

Legend: — = unimplemented location, read as '0'. Shaded cells are not used by clock sources.

6.0 RESETS

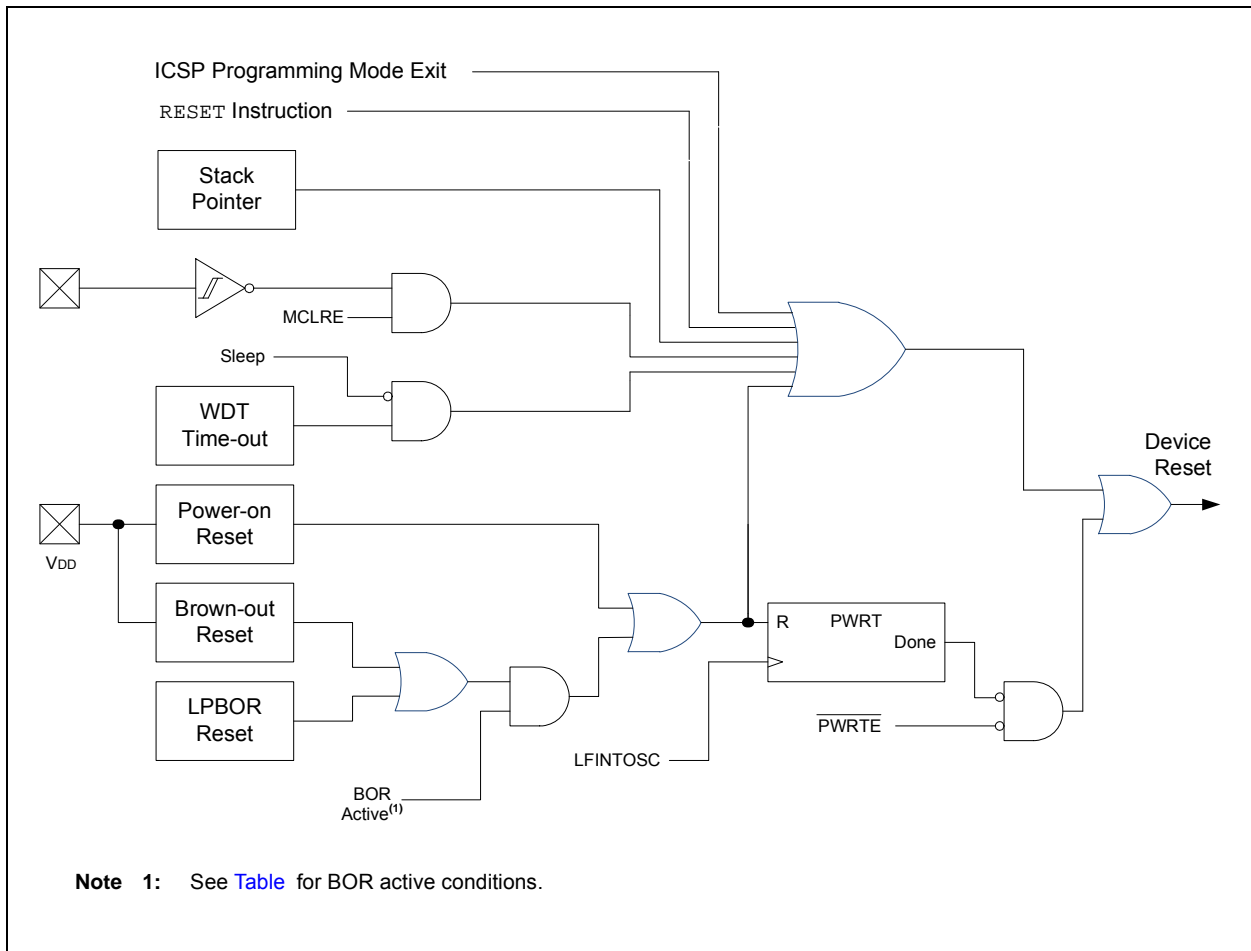
A simplified block diagram of the On-Chip Reset Circuit is shown in [Figure 6-1](#).

There are multiple ways to reset this device:

- Power-On Reset (POR)
- Brown-Out Reset (BOR)
- Low-Power Brown-Out Reset (LPBOR)
- MCLR Reset
- WDT Reset
- RESET instruction
- Stack Overflow
- Stack Underflow
- Programming mode exit

To allow VDD to stabilize, an optional power-up timer can be enabled to extend the Reset time after a BOR or POR event.

FIGURE 6-1: SIMPLIFIED BLOCK DIAGRAM OF ON-CHIP RESET CIRCUIT



PIC16(L)F1526/7

6.1 Power-On Reset (POR)

The POR circuit holds the device in Reset until VDD has reached an acceptable level for minimum operation. Slow rising VDD, fast operating speeds or analog performance may require greater than minimum VDD. The PWRT, BOR or MCLR features can be used to extend the start-up period until all device operation conditions have been met.

6.1.1 POWER-UP TIMER (PWRT)

The Power-up Timer provides a nominal 64 ms time-out on POR or Brown-out Reset.

The device is held in Reset as long as PWRT is active. The PWRT delay allows additional time for the VDD to rise to an acceptable level. The Power-up Timer is enabled by clearing the PWRTE bit in Configuration Words.

The Power-up Timer starts after the release of the POR and BOR.

For additional information, refer to Application Note AN607, "Power-up Trouble Shooting" (DS00607).

6.2 Brown-Out Reset (BOR)

The BOR circuit holds the device in Reset when VDD reaches a selectable minimum level. Between the POR and BOR, complete voltage range coverage for execution protection can be implemented.

The Brown-out Reset module has four operating modes controlled by the BOREN<1:0> bits in Configuration Words. The four operating modes are:

- BOR is always on
- BOR is off when in Sleep
- BOR is controlled by software
- BOR is always off

Refer to [Table](#) for more information.

The Brown-out Reset voltage level is selectable by configuring the BORV bit in Configuration Words.

A VDD noise rejection filter prevents the BOR from triggering on small events. If VDD falls below VBOR for a duration greater than parameter TBORDC, the device will reset. See [Figure 6-2](#) for more information.

TABLE 6-1: BOR OPERATING MODES

| BOREN<1:0> | SBOREN | Device Mode | BOR Mode | Instruction Execution upon: Release of POR or Wake-up from Sleep |
|------------|--------|-------------|----------|---|
| 11 | X | X | Active | Waits for BOR ready ⁽¹⁾ (BORRDY = 1) |
| 10 | X | Awake | Active | Waits for BOR ready (BORRDY = 1) |
| | | Sleep | Disabled | |
| 01 | 1 | X | Active | Waits for BOR ready ⁽¹⁾ (BORRDY = 1) |
| | 0 | X | Disabled | Begins immediately (BORRDY = x) |
| 00 | X | X | Disabled | |

Note 1: In these specific cases, "Release of POR" and "Wake-up from Sleep", there is no delay in start-up. The BOR ready flag, (BORRDY = 1), will be set before the CPU is ready to execute instructions because the BOR circuit is forced on by the BOREN<1:0> bits.

6.2.1 BOR IS ALWAYS ON

When the BOREN bits of Configuration Words are programmed to '11', the BOR is always on. The device start-up will be delayed until the BOR is ready and VDD is higher than the BOR threshold.

BOR protection is active during Sleep. The BOR does not delay wake-up from Sleep.

6.2.2 BOR IS OFF IN SLEEP

When the BOREN bits of Configuration Words are programmed to '10', the BOR is on, except in Sleep. The device start-up will be delayed until the BOR is ready and VDD is higher than the BOR threshold.

BOR protection is not active during Sleep. The device wake-up will be delayed until the BOR is ready.

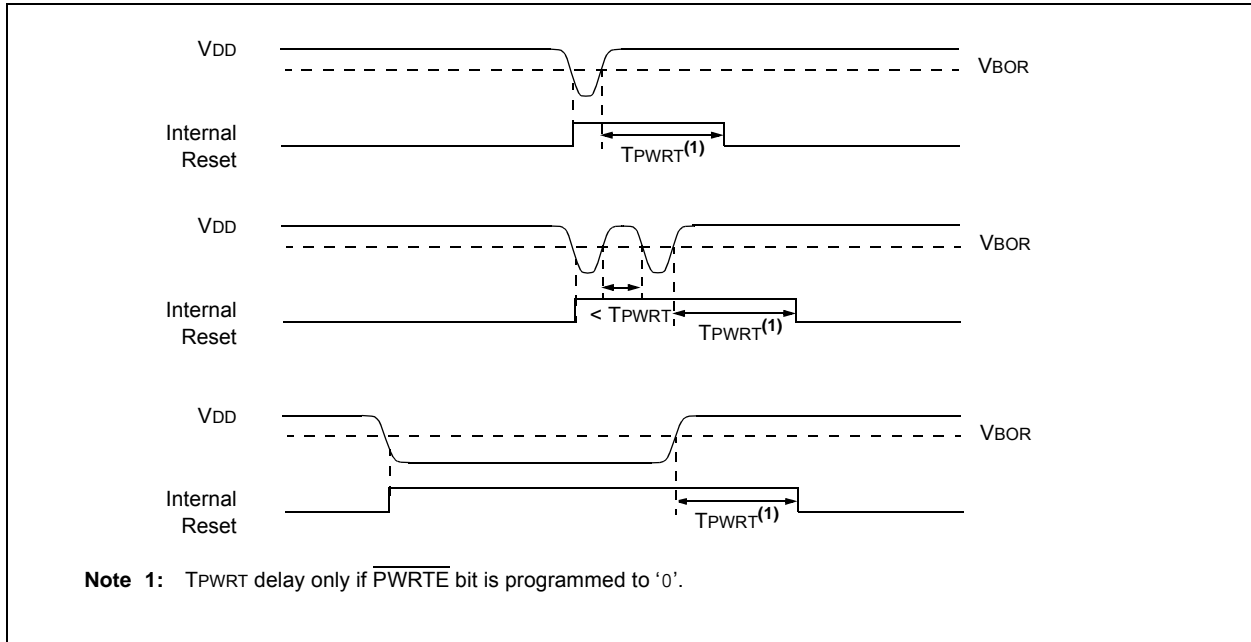
6.2.3 BOR CONTROLLED BY SOFTWARE

When the BOREN bits of Configuration Words are programmed to '01', the BOR is controlled by the SBOREN bit of the BORCON register. The device start-up is not delayed by the BOR ready condition or the VDD level.

BOR protection begins as soon as the BOR circuit is ready. The status of the BOR circuit is reflected in the BORRDY bit of the BORCON register.

BOR protection is unchanged by Sleep.

FIGURE 6-2: BROWN-OUT SITUATIONS



6.3 Register Definitions: BOR Control

REGISTER 6-1: BORCON: BROWN-OUT RESET CONTROL REGISTER

| | | | | | | | |
|---------|---------|-----|-----|-----|-----|-----|--------|
| R/W-1/u | R/W-0/u | U-0 | U-0 | U-0 | U-0 | U-0 | R-q/u |
| SBOREN | BORFS | — | — | — | — | — | BORRDY |
| bit 7 | | | | | | | bit 0 |

Legend:

| | | |
|----------------------|----------------------|---|
| R = Readable bit | W = Writable bit | U = Unimplemented bit, read as '0' |
| u = Bit is unchanged | x = Bit is unknown | -n/n = Value at POR and BOR/Value at all other Resets |
| '1' = Bit is set | '0' = Bit is cleared | q = Value depends on condition |

bit 7 **SBOREN:** Software Brown-out Reset Enable bit⁽¹⁾

If BOREN <1:0> ≠ 01:

SBOREN is read/write, but has no effect on the BOR.

If BOREN <1:0> = 01:

1 = BOR Enabled

0 = BOR Disabled

bit 6 **BORFS:** Brown-out Reset Fast Start bit⁽¹⁾

If BOREN<1:0> = 11 (Always on) or BOREN<1:0> = 00 (Always off)

BORFS is Read/Write, but has no effect.

If BOREN <1:0> = 10 (Disabled in Sleep) or BOREN<1:0> = 01 (Under software control):

1 = Band gap is forced on always (covers sleep/wake-up/operating cases)

0 = Band gap operates normally, and may turn off

bit 5-1 **Unimplemented:** Read as '0'

bit 0 **BORRDY:** Brown-out Reset Circuit Ready Status bit

1 = The Brown-out Reset circuit is active

0 = The Brown-out Reset circuit is inactive

Note 1: BOREN<1:0> bits are located in Configuration Words.

PIC16(L)F1526/7

6.4 Low-Power Brown-Out Reset (LPBOR)

The Low-Power Brown-Out Reset (LPBOR) is an essential part of the Reset subsystem. Refer to [Figure 6-1](#) to see how the BOR interacts with other modules.

The LPBOR is used to monitor the external VDD pin. When too low of a voltage is detected, the device is held in Reset. When this occurs, a register bit ($\overline{\text{BOR}}$) is changed to indicate that a BOR Reset has occurred. The same bit is set for both the BOR and the LPBOR. Refer to [Register 6-2](#).

6.4.1 ENABLING LPBOR

The LPBOR is controlled by the $\overline{\text{LPBOR}}$ bit of Configuration Words. When the device is erased, the LPBOR module defaults to disabled.

6.4.1.1 LPBOR Module Output

The output of the LPBOR module is a signal indicating whether or not a Reset is to be asserted. This signal is OR'd together with the Reset signal of the BOR module to provide the generic $\overline{\text{BOR}}$ signal, which goes to the PCON register and to the power control block.

6.5 $\overline{\text{MCLR}}$

The $\overline{\text{MCLR}}$ is an optional external input that can reset the device. The MCLR function is controlled by the MCLRE bit of Configuration Words and the LVP bit of Configuration Words ([Table 6-2](#)).

TABLE 6-2: $\overline{\text{MCLR}}$ CONFIGURATION

| MCLRE | LVP | $\overline{\text{MCLR}}$ |
|-------|-----|--------------------------|
| 0 | 0 | Disabled |
| 1 | 0 | Enabled |
| x | 1 | Enabled |

6.5.1 $\overline{\text{MCLR}}$ ENABLED

When $\overline{\text{MCLR}}$ is enabled and the pin is held low, the device is held in Reset. The $\overline{\text{MCLR}}$ pin is connected to VDD through an internal weak pull-up.

The device has a noise filter in the $\overline{\text{MCLR}}$ Reset path. The filter will detect and ignore small pulses.

Note: A Reset does not drive the $\overline{\text{MCLR}}$ pin low.

6.5.2 $\overline{\text{MCLR}}$ DISABLED

When $\overline{\text{MCLR}}$ is disabled, the pin functions as a general purpose input and the internal weak pull-up is under software control. See [Section Register 12-19: "PORTE: PORTE Register"](#) for more information.

6.6 Watchdog Timer (WDT) Reset

The Watchdog Timer generates a Reset if the firmware does not issue a CLRWDT instruction within the time-out period. The $\overline{\text{TO}}$ and $\overline{\text{PD}}$ bits in the STATUS register are changed to indicate the WDT Reset. See [Section 10.0 "Watchdog Timer \(WDT\)"](#) for more information.

6.7 RESET Instruction

A RESET instruction will cause a device Reset. The $\overline{\text{R}}$ bit in the PCON register will be set to '0'. See [Table 6-4](#) for default conditions after a RESET instruction has occurred.

6.8 Stack Overflow/Underflow Reset

The device can reset when the Stack Overflows or Underflows. The STKOVF or STKUNF bits of the PCON register indicate the Reset condition. These Resets are enabled by setting the STVREN bit in Configuration Words. See [Section 3.7.2 "Overflow/Underflow Reset"](#) for more information.

6.9 Programming Mode Exit

Upon exit of Programming mode, the device will behave as if a POR had just occurred.

6.10 Power-up Timer

The Power-up Timer optionally delays device execution after a BOR or POR event. This timer is typically used to allow VDD to stabilize before allowing the device to start running.

The Power-up Timer is controlled by the $\overline{\text{PWRT}}E$ bit of Configuration Words.

6.11 Start-up Sequence

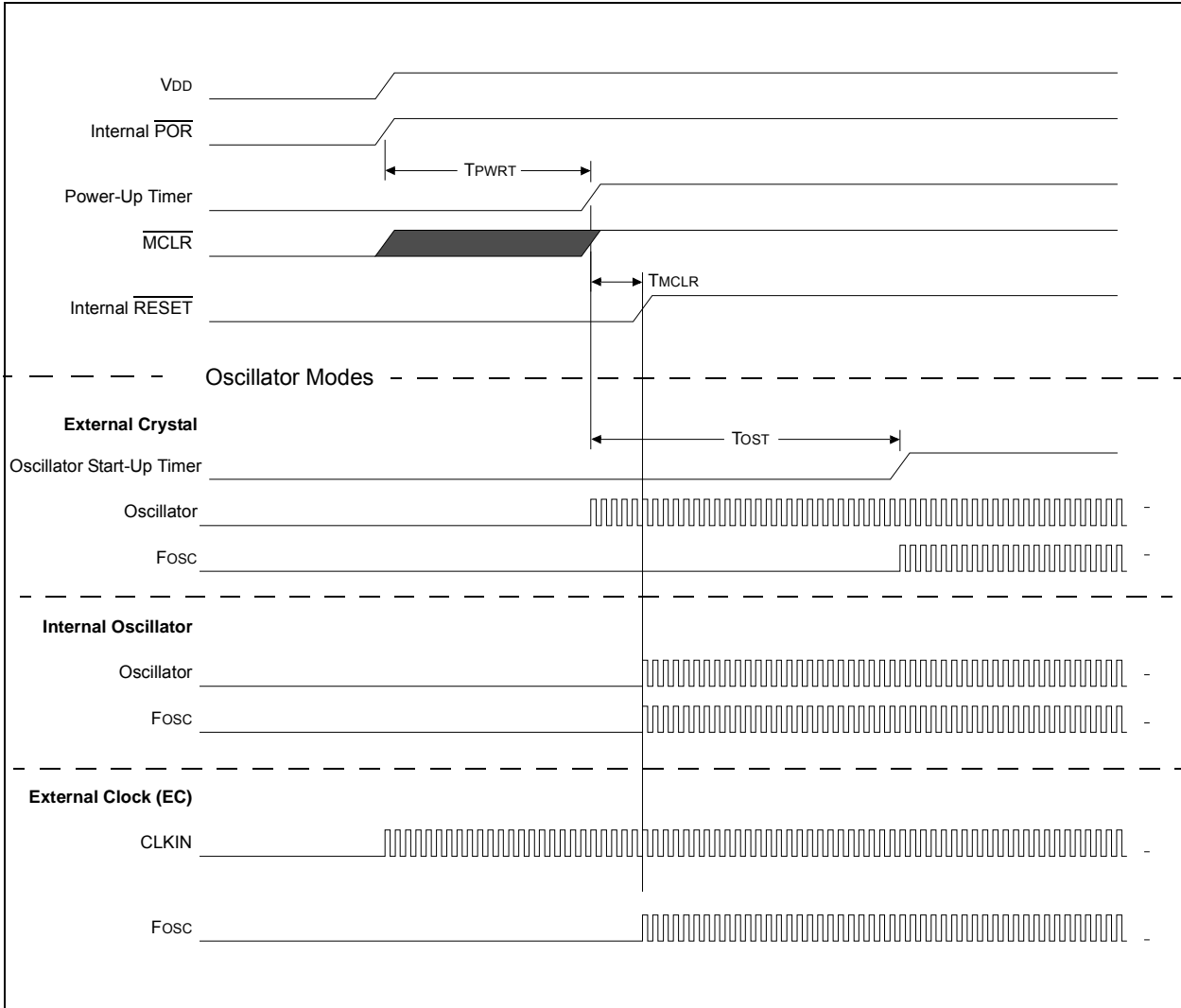
Upon the release of a POR or BOR, the following must occur before the device will begin executing:

1. Power-up Timer runs to completion (if enabled).
2. Oscillator start-up timer runs to completion (if required for oscillator source).
3. $\overline{\text{MCLR}}$ must be released (if enabled).

The total time-out will vary based on oscillator configuration and Power-up Timer configuration. See [Section 5.0 "Oscillator Module \(with Fail-Safe Clock Monitor\)"](#) for more information.

The Power-up Timer and oscillator start-up timer run independently of $\overline{\text{MCLR}}$ Reset. If $\overline{\text{MCLR}}$ is kept low long enough, the Power-up Timer and oscillator start-up timer will expire. Upon bringing $\overline{\text{MCLR}}$ high, the device will begin execution immediately (see [Figure 6-3](#)). This is useful for testing purposes or to synchronize more than one device operating in parallel.

FIGURE 6-3: RESET START-UP SEQUENCE



PIC16(L)F1526/7

6.12 Determining the Cause of a Reset

Upon any Reset, multiple bits in the STATUS and PCON register are updated to indicate the cause of the Reset. Table 6-3 and Table 6-4 show the Reset conditions of these registers.

TABLE 6-3: RESET STATUS BITS AND THEIR SIGNIFICANCE

| STKOVF | STKUNF | RWDT | RMCLR | RI | POR | BOR | TO | PD | Condition |
|--------|--------|------|-------|----|-----|-----|----|----|---|
| 0 | 0 | 1 | 1 | 1 | 0 | x | 1 | 1 | Power-on Reset |
| 0 | 0 | 1 | 1 | 1 | 0 | x | 0 | x | Illegal, \overline{TO} is set on \overline{POR} |
| 0 | 0 | 1 | 1 | 1 | 0 | x | x | 0 | Illegal, \overline{PD} is set on \overline{POR} |
| 0 | 0 | u | 1 | 1 | u | 0 | 1 | 1 | Brown-out Reset |
| u | u | 0 | u | u | u | u | 0 | u | WDT Reset |
| u | u | u | u | u | u | u | 0 | 0 | WDT Wake-up from Sleep |
| u | u | u | u | u | u | u | 1 | 0 | Interrupt Wake-up from Sleep |
| u | u | u | 0 | u | u | u | u | u | \overline{MCLR} Reset during normal operation |
| u | u | u | 0 | u | u | u | 1 | 0 | \overline{MCLR} Reset during Sleep |
| u | u | u | u | 0 | u | u | u | u | RESET Instruction Executed |
| 1 | u | u | u | u | u | u | u | u | Stack Overflow Reset (STVREN = 1) |
| u | 1 | u | u | u | u | u | u | u | Stack Underflow Reset (STVREN = 1) |

TABLE 6-4: RESET CONDITION FOR SPECIAL REGISTERS

| Condition | Program Counter | STATUS Register | PCON Register |
|---|-----------------------|-----------------|---------------|
| Power-on Reset | 0000h | ---1 1000 | 00-1 110x |
| \overline{MCLR} Reset during normal operation | 0000h | ---u uuuu | uu-u 0uuu |
| \overline{MCLR} Reset during Sleep | 0000h | ---1 0uuu | uu-u 0uuu |
| WDT Reset | 0000h | ---0 uuuu | uu-0 uuuu |
| WDT Wake-up from Sleep | PC + 1 | ---0 0uuu | uu-u uuuu |
| Brown-out Reset | 0000h | ---1 1uuu | 00-1 11u0 |
| Interrupt Wake-up from Sleep | PC + 1 ⁽¹⁾ | ---1 0uuu | uu-u uuuu |
| RESET Instruction Executed | 0000h | ---u uuuu | uu-u u0uu |
| Stack Overflow Reset (STVREN = 1) | 0000h | ---u uuuu | 1u-u uuuu |
| Stack Underflow Reset (STVREN = 1) | 0000h | ---u uuuu | u1-u uuuu |

Legend: u = unchanged, x = unknown, - = unimplemented bit, reads as '0'.

Note 1: When the wake-up is due to an interrupt and Global Enable bit (GIE) is set, the return address is pushed on the stack and PC is loaded with the interrupt vector (0004h) after execution of PC + 1.

6.13 Power Control (PCON) Register

The Power Control (PCON) register contains flag bits to differentiate between a:

- Power-on Reset ($\overline{\text{POR}}$)
- Brown-out Reset ($\overline{\text{BOR}}$)
- Reset Instruction Reset ($\overline{\text{RI}}$)
- MCLR Reset ($\overline{\text{RMCLR}}$)
- Watchdog Timer Reset ($\overline{\text{RWDT}}$)
- Stack Underflow Reset (STKUNF)
- Stack Overflow Reset (STKOVF)

The PCON register bits are shown in [Register 6-2](#).

6.14 Register Definitions: Power Control

REGISTER 6-2: PCON: POWER CONTROL REGISTER

| | | | | | | | |
|------------|------------|-----|--------------------------|---------------------------|------------------------|-------------------------|-------------------------|
| R/W/HS-0/q | R/W/HS-0/q | U-0 | R/W/HC-1/q | R/W/HC-1/q | R/W/HC-1/q | R/W/HC-q/u | R/W/HC-q/u |
| STKOVF | STKUNF | — | $\overline{\text{RWDT}}$ | $\overline{\text{RMCLR}}$ | $\overline{\text{RI}}$ | $\overline{\text{POR}}$ | $\overline{\text{BOR}}$ |
| bit 7 | | | | | | bit 0 | |

Legend:

HC = Bit is cleared by hardware

HS = Bit is set by hardware

R = Readable bit

W = Writable bit

U = Unimplemented bit, read as '0'

u = Bit is unchanged

x = Bit is unknown

-m/n = Value at POR and BOR/Value at all other Resets

'1' = Bit is set

'0' = Bit is cleared

q = Value depends on condition

bit 7 **STKOVF:** Stack Overflow Flag bit

1 = A Stack Overflow occurred

0 = A Stack Overflow has not occurred or cleared by firmware

bit 6 **STKUNF:** Stack Underflow Flag bit

1 = A Stack Underflow occurred

0 = A Stack Underflow has not occurred or cleared by firmware

bit 5 **Unimplemented:** Read as '0'

bit 4 **RWDT:** Watchdog Timer Reset Flag bit

1 = A Watchdog Timer Reset has not occurred or set to '1' by firmware

0 = A Watchdog Timer Reset has occurred (cleared by hardware)

bit 3 **RMCLR:** MCLR Reset Flag bit

1 = A $\overline{\text{MCLR}}$ Reset has not occurred or set to '1' by firmware

0 = A $\overline{\text{MCLR}}$ Reset has occurred (cleared by hardware)

bit 2 **RI:** RESET Instruction Flag bit

1 = A RESET instruction has not been executed or set to '1' by firmware

0 = A RESET instruction has been executed (cleared by hardware)

bit 1 **POR:** Power-on Reset Status bit

1 = No Power-on Reset occurred

0 = A Power-on Reset occurred (must be set in software after a Power-on Reset occurs)

bit 0 **BOR:** Brown-out Reset Status bit

1 = No Brown-out Reset occurred

0 = A Brown-out Reset occurred (must be set in software after a Power-on Reset or Brown-out Reset occurs)

PIC16(L)F1526/7

TABLE 6-5: SUMMARY OF REGISTERS ASSOCIATED WITH RESETS

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Register on Page |
|--------|--------|--------|------------|--------------------------|---------------------------|------------------------|-------------------------|-------------------------|------------------|
| BORCON | SBOREN | BORFS | — | — | — | — | — | BORRDY | 65 |
| PCON | STKOVF | STKUNF | — | $\overline{\text{RWDT}}$ | $\overline{\text{RMCLR}}$ | $\overline{\text{RI}}$ | $\overline{\text{POR}}$ | $\overline{\text{BOR}}$ | 69 |
| STATUS | — | — | — | $\overline{\text{TO}}$ | $\overline{\text{PD}}$ | Z | DC | C | 21 |
| WDTCON | — | — | WDTPS<4:0> | | | | | SWDTEN | 93 |

Legend: — = unimplemented bit, read as '0'. Shaded cells are not used by Resets.

Note 1: Other (non Power-up) Resets include MCLR Reset and Watchdog Timer Reset during normal operation.

7.0 INTERRUPTS

The interrupt feature allows certain events to preempt normal program flow. Firmware is used to determine the source of the interrupt and act accordingly. Some interrupts can be configured to wake the MCU from Sleep mode.

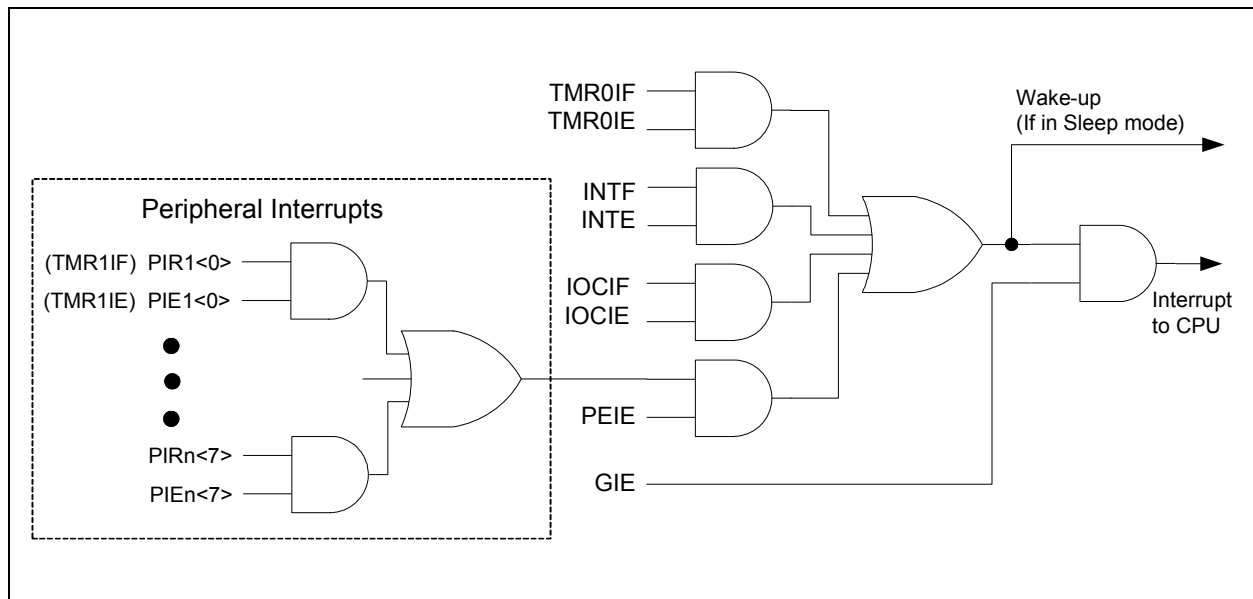
This chapter contains the following information for Interrupts:

- Operation
- Interrupt Latency
- Interrupts During Sleep
- INT Pin
- Automatic Context Saving

Many peripherals produce interrupts. Refer to the corresponding chapters for details.

A block diagram of the interrupt logic is shown in [Figure 7-1](#).

FIGURE 7-1: INTERRUPT LOGIC



PIC16(L)F1526/7

7.1 Operation

Interrupts are disabled upon any device Reset. They are enabled by setting the following bits:

- GIE bit of the INTCON register
- Interrupt Enable bit(s) for the specific interrupt event(s)
- PEIE bit of the INTCON register (if the Interrupt Enable bit of the interrupt event is contained in the PEx register)

The INTCON and PIRx registers record individual interrupts via interrupt flag bits. Interrupt flag bits will be set, regardless of the status of the GIE, PEIE and individual interrupt enable bits.

The following events happen when an interrupt event occurs while the GIE bit is set:

- Current prefetched instruction is flushed
- GIE bit is cleared
- Current Program Counter (PC) is pushed onto the stack
- Critical registers are automatically saved to the shadow registers (See [Section 7.5 “Automatic Context Saving”](#))
- PC is loaded with the interrupt vector 0004h

The firmware within the Interrupt Service Routine (ISR) should determine the source of the interrupt by polling the interrupt flag bits. The interrupt flag bits must be cleared before exiting the ISR to avoid repeated interrupts. Because the GIE bit is cleared, any interrupt that occurs while executing the ISR will be recorded through its interrupt flag, but will not cause the processor to redirect to the interrupt vector.

The `RETFIE` instruction exits the ISR by popping the previous address from the stack, restoring the saved context from the shadow registers and setting the GIE bit.

For additional information on a specific interrupt's operation, refer to its peripheral chapter.

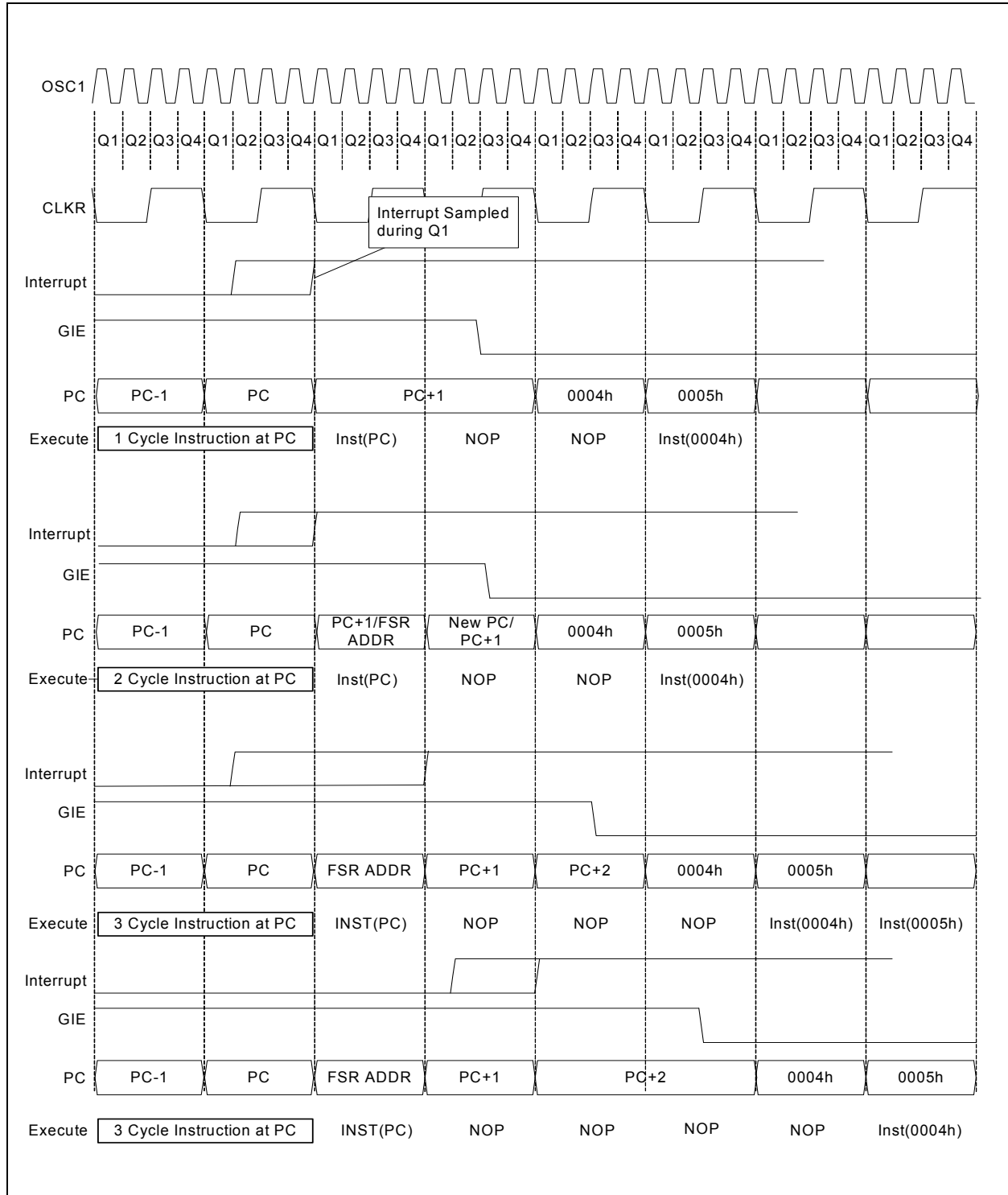
Note 1: Individual interrupt flag bits are set, regardless of the state of any other enable bits.

2: All interrupts will be ignored while the GIE bit is cleared. Any interrupt occurring while the GIE bit is clear will be serviced when the GIE bit is set again.

7.2 Interrupt Latency

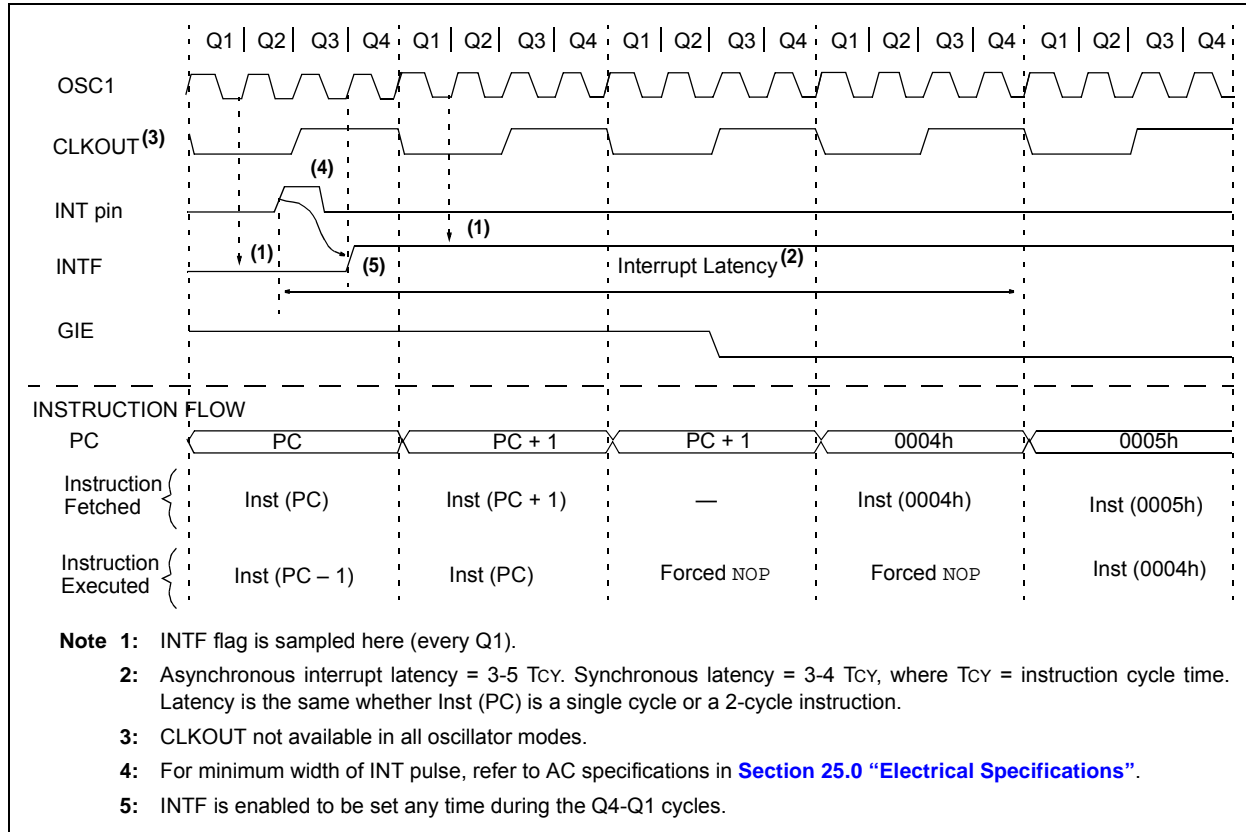
Interrupt latency is defined as the time from when the interrupt event occurs to the time code execution at the interrupt vector begins. The latency for synchronous interrupts is three or four instruction cycles. For asynchronous interrupts, the latency is three to five instruction cycles, depending on when the interrupt occurs. See [Figure 7-2](#) and [Figure 7-3](#) for more details.

FIGURE 7-2: INTERRUPT LATENCY



PIC16(L)F1526/7

FIGURE 7-3: INT PIN INTERRUPT TIMING



7.3 Interrupts During Sleep

Some interrupts can be used to wake from Sleep. To wake from Sleep, the peripheral must be able to operate without the system clock. The interrupt source must have the appropriate Interrupt Enable bit(s) set prior to entering Sleep.

On waking from Sleep, if the GIE bit is also set, the processor will branch to the interrupt vector. Otherwise, the processor will continue executing instructions after the `SLEEP` instruction. The instruction directly after the `SLEEP` instruction will always be executed before branching to the ISR. Refer to the [Section 8.0 “Power-Down Mode \(Sleep\)”](#) for more details.

7.4 INT Pin

The INT pin can be used to generate an asynchronous edge-triggered interrupt. This interrupt is enabled by setting the INTE bit of the INTCON register. The INTEDG bit of the OPTION_REG register determines on which edge the interrupt will occur. When the INTEDG bit is set, the rising edge will cause the interrupt. When the INTEDG bit is clear, the falling edge will cause the interrupt. The INTF bit of the INTCON register will be set when a valid edge appears on the INT pin. If the GIE and INTE bits are also set, the processor will redirect program execution to the interrupt vector.

7.5 Automatic Context Saving

Upon entering an interrupt, the return PC address is saved on the stack. Additionally, the following registers are automatically saved in the shadow registers:

- W register
- STATUS register (except for \overline{TO} and \overline{PD})
- BSR register
- FSR registers
- PCLATH register

Upon exiting the Interrupt Service Routine, these registers are automatically restored. Any modifications to these registers during the ISR will be lost. If modifications to any of these registers are desired, the corresponding shadow register should be modified and the value will be restored when exiting the ISR. The shadow registers are available in Bank 31 and are readable and writable. Depending on the user's application, other registers may also need to be saved.

PIC16(L)F1526/7

7.6 Register Definitions: Interrupt Control

REGISTER 7-1: INTCON: INTERRUPT CONTROL REGISTER

| | | | | | | | |
|---------|---------|---------|---------|---------|---------|---------|-------|
| R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R-0/0 |
| GIE | PEIE | TMR0IE | INTE | IOIE | TMR0IF | INTF | IOCIF |
| bit 7 | | | | | | | bit 0 |

Legend:

R = Readable bit

W = Writable bit

U = Unimplemented bit, read as '0'

u = Bit is unchanged

x = Bit is unknown

-n/n = Value at POR and BOR/Value at all other Resets

'1' = Bit is set

'0' = Bit is cleared

- bit 7 **GIE:** Global Interrupt Enable bit
1 = Enables all active interrupts
0 = Disables all interrupts
- bit 6 **PEIE:** Peripheral Interrupt Enable bit
1 = Enables all active peripheral interrupts
0 = Disables all peripheral interrupts
- bit 5 **TMROIE:** Timer0 Overflow Interrupt Enable bit
1 = Enables the Timer0 interrupt
0 = Disables the Timer0 interrupt
- bit 4 **INTE:** INT External Interrupt Enable bit
1 = Enables the INT external interrupt
0 = Disables the INT external interrupt
- bit 3 **IOIE:** Interrupt-on-Change Interrupt Enable bit
1 = Enables the interrupt-on-change interrupt
0 = Disables the interrupt-on-change interrupt
- bit 2 **TMR0IF:** Timer0 Overflow Interrupt Flag bit
1 = TMR0 register has overflowed
0 = TMR0 register did not overflow
- bit 1 **INTF:** INT External Interrupt Flag bit
1 = The INT external interrupt occurred
0 = The INT external interrupt did not occur
- bit 0 **IOCIF:** Interrupt-on-Change Interrupt Flag bit⁽¹⁾
1 = When at least one of the interrupt-on-change pins changed state
0 = None of the interrupt-on-change pins have changed state

Note 1: The IOCIF flag bit is read-only and cleared when all the Interrupt-on-Change flags in the IOCBF register have been cleared by software.

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

REGISTER 7-2: PIE1: PERIPHERAL INTERRUPT ENABLE REGISTER 1

| | | | | | | | |
|---------|---------|---------|---------|---------|---------|---------|---------|
| R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 |
| TMR1GIE | ADIE | RC1IE | TX1IE | SSP1IE | SSP2IE | TMR2IE | TMR1IE |
| bit 7 | | | | | | | bit 0 |

Legend:

| | | |
|----------------------|----------------------|---|
| R = Readable bit | W = Writable bit | U = Unimplemented bit, read as '0' |
| u = Bit is unchanged | x = Bit is unknown | -n/n = Value at POR and BOR/Value at all other Resets |
| '1' = Bit is set | '0' = Bit is cleared | |

- bit 7 **TMR1GIE:** Timer1 Gate Interrupt Enable bit
1 = Enables the Timer1 Gate Acquisition interrupt
0 = Disables the Timer1 Gate Acquisition interrupt
- bit 6 **ADIE:** Analog-to-Digital Converter (ADC) Interrupt Enable bit
1 = Enables the ADC interrupt
0 = Disables the ADC interrupt
- bit 5 **RC1IE:** USART1 Receive Interrupt Enable bit
1 = Enables the USART1 receive interrupt
0 = Disables the USART1 receive interrupt
- bit 4 **TX1IE:** USART1 Transmit Interrupt Enable bit
1 = Enables the USART1 transmit interrupt
0 = Disables the USART1 transmit interrupt
- bit 3 **SSP1IE:** Synchronous Serial Port (MSSP1) Interrupt Enable bit
1 = Enables the MSSP1 interrupt
0 = Disables the MSSP1 interrupt
- bit 2 **SSP2IE:** Synchronous Serial Port (MSSP2) Interrupt Enable bit
1 = Enables the MSSP2 interrupt
0 = Disables the MSSP2 interrupt
- bit 1 **TMR2IE:** TMR2 to PR2 Match Interrupt Enable bit
1 = Enables the Timer2 to PR2 match interrupt
0 = Disables the Timer2 to PR2 match interrupt
- bit 0 **TMR1IE:** Timer1 Overflow Interrupt Enable bit
1 = Enables the Timer1 overflow interrupt
0 = Disables the Timer1 overflow interrupt

Note: Bit PEIE of the INTCON register must be set to enable any peripheral interrupt.

PIC16(L)F1526/7

REGISTER 7-3: PIE2: PERIPHERAL INTERRUPT ENABLE REGISTER 2

| | | | | | | | |
|-------|---------|-----|-----|---------|---------|---------|-------|
| U-0 | R/W-0/0 | U-0 | U-0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | U-0 |
| | AD2IE | | | BCL1IE | BCL2IE | TMR4IE | |
| bit 7 | | | | | | | bit 0 |

Legend:

R = Readable bit W = Writable bit U = Unimplemented bit, read as '0'
u = Bit is unchanged x = Bit is unknown -n/n = Value at POR and BOR/Value at all other Resets
'1' = Bit is set '0' = Bit is cleared

- bit 7 **Unimplemented:** Read as '0'
- bit 6 **AD2IE:** Analog-to-Digital Converter (ADC2) Interrupt Enable bit
1 = Enables the ADC interrupt
0 = Disables the ADC interrupt
- bit 5-4 **Unimplemented:** Read as '0'
- bit 3 **BCL1IE:** MSSP1 Bus Collision Interrupt Enable bit
1 = Enables the MSSP1 Bus Collision Interrupt
0 = Disables the MSSP1 Bus Collision Interrupt
- bit 2 **BCL2IE:** MSSP2 Bus Collision Interrupt Enable bit
1 = Enables the MSSP2 Bus Collision Interrupt
0 = Disables the MSSP2 Bus Collision Interrupt
- bit 1 **TMR4IE:** TMR4 to PR4 Match Interrupt Enable bit
1 = Enables the Timer8 to PR4 match interrupt
0 = Disables the Timer8 to PR4 match interrupt
- bit 0 **Unimplemented:** Read as '0'

Note: Bit PEIE of the INTCON register must be set to enable any peripheral interrupt.

REGISTER 7-4: PIE3: PERIPHERAL INTERRUPT ENABLE REGISTER 3

| R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 |
|---------|---------|---------|---------|---------|---------|---------|---------|
| CCP6IE | CCP5IE | CCP4IE | CCP3IE | TMR6IE | TMR5IE | TMR4IE | TMR3IE |
| bit 7 | | | | | | | bit 0 |

Legend:

R = Readable bit

W = Writable bit

U = Unimplemented bit, read as '0'

u = Bit is unchanged

x = Bit is unknown

-n/n = Value at POR and BOR/Value at all other Resets

'1' = Bit is set

'0' = Bit is cleared

- bit 7 **CCP6IE:** CCP6 Interrupt Enable bit
1 = Enables the CCP6 interrupt
0 = Disables the CCP6 interrupt
- bit 6 **CCP5IE:** CCP5 Interrupt Enable bit
1 = Enables the CCP5 interrupt
0 = Disables the CCP5 interrupt
- bit 5 **CCP4IE:** CCP4 Interrupt Enable bit
1 = Enables the CCP4 interrupt
0 = Disables the CCP4 interrupt
- bit 4 **CCP3IE:** CCP3 Interrupt Enable bit
1 = Enables the CCP3 interrupt
0 = Disables the CCP3 interrupt
- bit 3 **TMR6IE:** TMR6 to PR6 Match Interrupt Enable bit
1 = Enables the TMR6 to PR6 Match interrupt
0 = Disables the TMR6 to PR6 Match interrupt
- bit 2 **TMR5IE:** Timer5 Overflow Interrupt Enable bit
1 = Enables the Timer5 overflow interrupt
0 = Disables the Timer5 overflow interrupt
- bit 1 **TMR4IE:** TMR4 to PR4 Match Interrupt Enable bit
1 = Enables the TMR4 to PR4 Match interrupt
0 = Disables the TMR4 to PR4 Match interrupt
- bit 0 **TMR3IE:** Timer3 Overflow Interrupt Enable bit
1 = Enables the Timer3 overflow interrupt
0 = Disables the Timer3 overflow interrupt

Note: Bit PEIE of the INTCON register must be set to enable any peripheral interrupt.

PIC16(L)F1526/7

REGISTER 7-5: PIE4: PERIPHERAL INTERRUPT ENABLE REGISTER 4

| | | | | | | | |
|---------|---------|-------|-------|---------|---------|---------|---------|
| R/W-0/0 | R/W-0/0 | R-0/0 | R-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 |
| CCP10IE | CCP9IE | RC2IE | TX2IE | CCP8IE | CCP7IE | BCL2IE | SSP2IE |
| bit 7 | | | | | | | bit 0 |

Legend:

R = Readable bit

W = Writable bit

U = Unimplemented bit, read as '0'

u = Bit is unchanged

x = Bit is unknown

-n/n = Value at POR and BOR/Value at all other Resets

'1' = Bit is set

'0' = Bit is cleared

- bit 7 **CCP10IE:** CCP10 Interrupt Enable bit
1 = Enables the CCP10 interrupt
0 = Disables the CCP10 interrupt
- bit 6 **CCP9IE:** CCP9 Interrupt Enable bit
1 = Enables the CCP9 interrupt
0 = Disables the CCP9 interrupt
- bit 5 **RC2IE:** USART2 Receive Interrupt Enable bit
1 = Enables the USART2 receive interrupt
0 = Disables the USART2 receive interrupt
- bit 4 **TX2IE:** USART2 Transmit Interrupt Enable bit
1 = Enables the USART2 transmit interrupt
0 = Disables the USART2 transmit interrupt
- bit 3 **CCP8IE:** CCP8 Interrupt Enable bit
1 = Enables the CCP8 interrupt
0 = Disables the CCP8 interrupt
- bit 2 **CCP7IE:** CCP7 Interrupt Enable bit
1 = Enables the CCP7 interrupt
0 = Disables the CCP7 interrupt
- bit 1 **BCL2IE:** MSSP2 Bus Collision Interrupt Enable bit
1 = Enables the MSSP2 Bus Collision Interrupt
0 = Disables the MSSP2 Bus Collision Interrupt
- bit 0 **SSP2IE:** Synchronous Serial Port (MSSP2) Interrupt Enable bit
1 = Enables the MSSP2 interrupt
0 = Disables the MSSP2 interrupt

Note: Bit PEIE of the INTCON register must be set to enable any peripheral interrupt.

REGISTER 7-6: PIR1: PERIPHERAL INTERRUPT REQUEST REGISTER 1

| | | | | | | | |
|---------|---------|-------|-------|---------|---------|---------|---------|
| R/W-0/0 | R/W-0/0 | R-0/0 | R-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 |
| TMR1GIF | ADIF | RC1IF | TX1IF | SSP1IF | CCP1IF | TMR2IF | TMR1IF |
| bit 7 | | | | | | | bit 0 |

Legend:

| | | |
|----------------------|----------------------|---|
| R = Readable bit | W = Writable bit | U = Unimplemented bit, read as '0' |
| u = Bit is unchanged | x = Bit is unknown | -n/n = Value at POR and BOR/Value at all other Resets |
| '1' = Bit is set | '0' = Bit is cleared | |

- bit 7 **TMR1GIF:** Timer1 Gate Interrupt Flag bit
1 = Interrupt is pending
0 = Interrupt is not pending
- bit 6 **ADIF:** ADC Converter Interrupt Flag bit
1 = Interrupt is pending
0 = Interrupt is not pending
- bit 5 **RC1IF:** USART1 Receive Interrupt Flag bit
1 = Interrupt is pending
0 = Interrupt is not pending
- bit 4 **TX1IF:** USART1 Transmit Interrupt Flag bit
1 = Interrupt is pending
0 = Interrupt is not pending
- bit 3 **SSP1IF:** Synchronous Serial Port (MSSP1) Interrupt Flag bit
1 = Interrupt is pending
0 = Interrupt is not pending
- bit 2 **CCP1IF:** CCP1 Interrupt Flag bit
1 = Interrupt is pending
0 = Interrupt is not pending
- bit 1 **TMR2IF:** Timer2 to PR2 Interrupt Flag bit
1 = Interrupt is pending
0 = Interrupt is not pending
- bit 0 **TMR1IF:** Timer1 Overflow Interrupt Flag bit
1 = Interrupt is pending
0 = Interrupt is not pending

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

PIC16(L)F1526/7

REGISTER 7-7: PIR2: PERIPHERAL INTERRUPT REQUEST REGISTER 2

| | | | | | | | |
|-------|---------|-----|-----|---------|---------|---------|-------|
| U-0 | R/W-0/0 | U-0 | U-0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | U-0 |
| — | AD2IF | — | — | BCL1IF | BCL2IF | TMR4IF | — |
| bit 7 | | | | | | | bit 0 |

Legend:

R = Readable bit

W = Writable bit

U = Unimplemented bit, read as '0'

u = Bit is unchanged

x = Bit is unknown

-n/n = Value at POR and BOR/Value at all other Resets

'1' = Bit is set

'0' = Bit is cleared

| | |
|---------|---|
| bit 7 | Unimplemented: Read as '0' |
| bit 6 | AD2IF: Timer5 Gate Interrupt Flag bit 1 = Interrupt is pending 0 = Interrupt is not pending |
| bit 5-4 | Unimplemented: Read as '0' |
| bit 3 | BCL1IF: MSSP1 Bus Collision Interrupt Flag bit 1 = Interrupt is pending 0 = Interrupt is not pending |
| bit 2 | BCL2IF: MSSP2 Bus Collision Interrupt Flag bit 1 = Interrupt is pending 0 = Interrupt is not pending |
| bit 1 | TMR4IF: Timer4 to PR4 Interrupt Flag bit 1 = Interrupt is pending 0 = Interrupt is not pending |
| bit 0 | Unimplemented: Read as '0' |

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

REGISTER 7-8: PIR3: PERIPHERAL INTERRUPT REQUEST REGISTER 3

| | | | | | | | |
|---------|---------|---------|---------|---------|---------|---------|---------|
| R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 |
| CCP6IF | CCP5IF | CCP4IF | CCP3IF | TMR6IF | TMR5IF | TMR4IF | TMR3IF |
| bit 7 | | | | | | | bit 0 |

Legend:

| | | |
|----------------------|----------------------|---|
| R = Readable bit | W = Writable bit | U = Unimplemented bit, read as '0' |
| u = Bit is unchanged | x = Bit is unknown | -n/n = Value at POR and BOR/Value at all other Resets |
| '1' = Bit is set | '0' = Bit is cleared | |

| | |
|-------|---|
| bit 7 | CCP6IF: CCP6 Interrupt Flag bit 1 = Interrupt is pending 0 = Interrupt is not pending |
| bit 6 | CCP5IF: CCP5 Interrupt Flag bit 1 = Interrupt is pending 0 = Interrupt is not pending |
| bit 5 | CCP4IF: CCP4 Interrupt Flag bit 1 = Interrupt is pending 0 = Interrupt is not pending |
| bit 4 | CCP3IF: CCP3 Interrupt Flag bit 1 = Interrupt is pending 0 = Interrupt is not pending |
| bit 3 | TMR6IF: TMR6 to PR6 Match Interrupt Flag bit 1 = Interrupt is pending 0 = Interrupt is not pending |
| bit 2 | TMR5IF: Timer5 Overflow Interrupt Flag bit 1 = Interrupt is pending 0 = Interrupt is not pending |
| bit 1 | TMR4IF: TMR4 to PR4 Match Interrupt Flag bit 1 = Interrupt is pending 0 = Interrupt is not pending |
| bit 0 | TMR3IF: Timer3 Overflow Interrupt Flag bit 1 = Interrupt is pending 0 = Interrupt is not pending |

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

PIC16(L)F1526/7

REGISTER 7-9: PIR4: PERIPHERAL INTERRUPT REQUEST REGISTER 4

| | | | | | | | |
|---------|---------|-------|-------|---------|---------|---------|---------|
| R/W-0/0 | R/W-0/0 | R-0/0 | R-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 |
| CCP10IF | CCP9IF | RC2IF | TX2IF | CCP8IF | CCP7IF | BCL2IF | SSP2IF |
| bit 7 | | | | | | | bit 0 |

Legend:

R = Readable bit W = Writable bit U = Unimplemented bit, read as '0'
u = Bit is unchanged x = Bit is unknown -n/n = Value at POR and BOR/Value at all other Resets
'1' = Bit is set '0' = Bit is cleared

- bit 7 **CCP10IF:** CCP10 Interrupt Flag bit
 1 = Interrupt is pending
 0 = Interrupt is not pending
- bit 6 **CCP9IF:** CCP9 Interrupt Flag bit
 1 = Interrupt is pending
 0 = Interrupt is not pending
- bit 5 **RC2IF:** USART2 Receive Interrupt Flag bit
 1 = Interrupt is pending
 0 = Interrupt is not pending
- bit 4 **TX2IF:** USART2 Transmit Interrupt Flag bit
 1 = Interrupt is pending
 0 = Interrupt is not pending
- bit 3 **CCP8IF:** CCP8 Interrupt Flag bit
 1 = Interrupt is pending
 0 = Interrupt is not pending
- bit 2 **CCP7IF:** CCP7 Interrupt Flag bit
 1 = Interrupt is pending
 0 = Interrupt is not pending
- bit 1 **BCL2IF:** MSSP2 Bus Collision Interrupt Flag bit
 1 = Interrupt is pending
 0 = Interrupt is not pending
- bit 0 **SSP2IF:** Synchronous Serial Port (MSSP2) Interrupt Flag bit
 1 = Interrupt is pending
 0 = Interrupt is not pending

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

TABLE 7-1: SUMMARY OF REGISTERS ASSOCIATED WITH INTERRUPTS

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Register on Page |
|------------|---------------------------|--------|--------|--------|--------|---------|--------|--------|------------------|
| INTCON | GIE | PEIE | TMR0IE | INTE | IOCFIE | TMR0IF | INTF | IOCFIF | 76 |
| IOCBF | IOCBF7 | IOCBF6 | IOCBF5 | IOCBF4 | IOCBF3 | IOCBF2 | IOCBF1 | IOCBF0 | 137 |
| IOCBN | IOCBN7 | IOCBN6 | IOCBN5 | IOCBN4 | IOCBN3 | IOCBN2 | IOCBN1 | IOCBN0 | 137 |
| IOCBP | IOCBP7 | IOCBP6 | IOCBP5 | IOCBP4 | IOCBP3 | IOCBP2 | IOCBP1 | IOCBP0 | 137 |
| OPTION_REG | $\overline{\text{WPUEN}}$ | INTEDG | TMR0CS | TMR0SE | PSA | PS<2:0> | | | 158 |
| PIE1 | TMR1GIE | ADIE | RC1IE | TX1IE | SSP1IE | SSP2IE | TMR2IE | TMR1IE | 77 |
| PIE2 | — | AD2IE | — | — | BCL1IE | BCL2IE | TMR4IE | — | 78 |
| PIE3 | CCP6IE | CCP5IE | CCP4IE | CCP3IE | TMR6IE | TMR5IE | TMR4IE | TMR3IE | 79 |
| PIE4 | CCP10IE | CCP9IE | RC2IE | TX2IE | CCP8IE | CCP7IE | BCL2IE | SSP2IE | 80 |
| PIR1 | TMR1GIF | AD1IF | RCIF | TXIF | SSP1IF | SSP2IF | TMR2IF | TMR1IF | 81 |
| PIR2 | — | AD2IF | — | — | BCL1IF | BCL2IF | TMR4IF | — | 82 |
| PIR3 | CCP6IF | CCP5IF | CCP4IF | CCP3IF | TMR6IF | TMR5IF | TMR4IF | TMR3IF | 83 |
| PIR4 | CCP10IF | CCP9IF | RC2IF | TX2IF | CCP8IF | CCP7IF | BCL2IF | SSP2IF | 84 |

Legend: — = unimplemented locations read as '0'. Shaded cells are not used by Interrupts.

PIC16(L)F1526/7

8.0 POWER-DOWN MODE (SLEEP)

The Power-down mode is entered by executing a `SLEEP` instruction.

Upon entering Sleep mode, the following conditions exist:

1. WDT will be cleared but keeps running, if enabled for operation during Sleep.
2. \overline{PD} bit of the STATUS register is cleared.
3. \overline{TO} bit of the STATUS register is set.
4. CPU clock is disabled.
5. 31 kHz LFINTOSC is unaffected and peripherals that operate from it may continue operation in Sleep.
6. Timer1 and peripherals that operate from Timer1 continue operation in Sleep when the Timer1 clock source selected is:
 - LFINTOSC
 - T1CKI
 - Secondary oscillator
7. ADC is unaffected, if the dedicated FRC oscillator is selected.
8. I/O ports maintain the status they had before `SLEEP` was executed (driving high, low or high-impedance).
9. Resets other than WDT are not affected by Sleep mode.

Refer to individual chapters for more details on peripheral operation during Sleep.

To minimize current consumption, the following conditions should be considered:

- I/O pins should not be floating
- External circuitry sinking current from I/O pins
- Internal circuitry sourcing current from I/O pins
- Current draw from pins with internal weak pull-ups
- Modules using 31 kHz LFINTOSC
- Modules using Secondary oscillator

I/O pins that are high-impedance inputs should be pulled to V_{DD} or V_{SS} externally to avoid switching currents caused by floating inputs.

Examples of internal circuitry that might be sourcing current include modules such as the FVR modules. See [Section 14.0 “Fixed Voltage Reference \(FVR\)”](#) for more information on these modules.

8.1 Wake-up from Sleep

The device can wake-up from Sleep through one of the following events:

1. External Reset input on \overline{MCLR} pin, if enabled
2. BOR Reset, if enabled
3. POR Reset
4. Watchdog Timer, if enabled
5. Any external interrupt
6. Interrupts by peripherals capable of running during Sleep (see individual peripheral for more information)

The first three events will cause a device Reset. The last three events are considered a continuation of program execution. To determine whether a device Reset or wake-up event occurred, refer to [Section 6.12 “Determining the Cause of a Reset”](#).

When the `SLEEP` instruction is being executed, the next instruction ($PC + 1$) is prefetched. For the device to wake-up through an interrupt event, the corresponding interrupt enable bit must be enabled. Wake-up will occur regardless of the state of the GIE bit. If the GIE bit is disabled, the device continues execution at the instruction after the `SLEEP` instruction. If the GIE bit is enabled, the device executes the instruction after the `SLEEP` instruction, the device will then call the Interrupt Service Routine. In cases where the execution of the instruction following `SLEEP` is not desirable, the user should have a `NOP` after the `SLEEP` instruction.

The WDT is cleared when the device wakes up from Sleep, regardless of the source of wake-up.

8.1.1 WAKE-UP USING INTERRUPTS

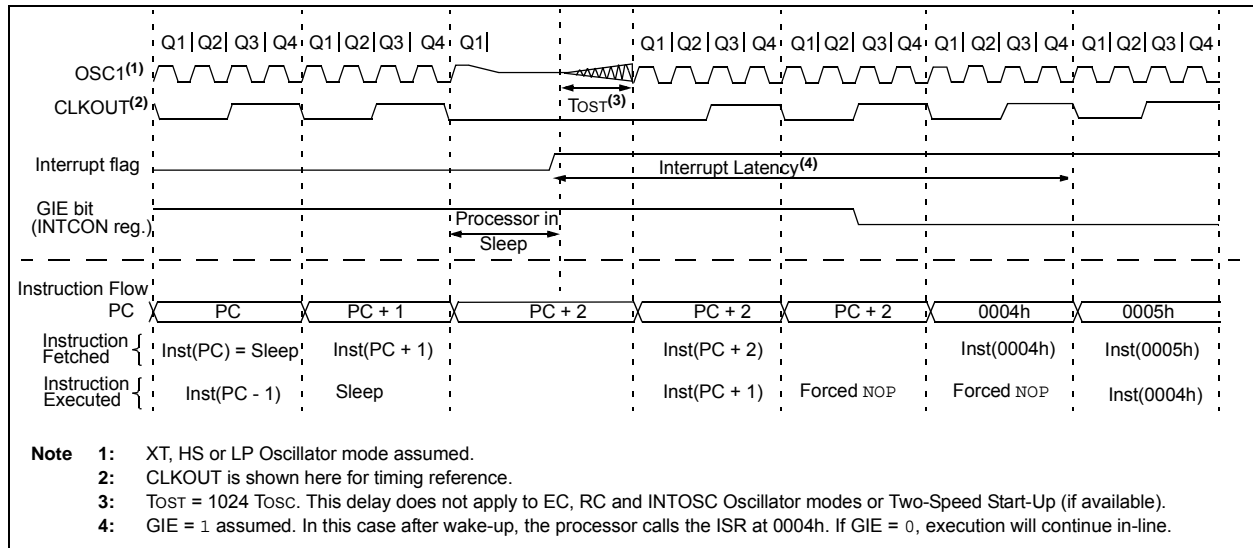
When global interrupts are disabled (GIE cleared) and any interrupt source has both its interrupt enable bit and interrupt flag bit set, one of the following will occur:

- If the interrupt occurs **before** the execution of a `SLEEP` instruction
 - `SLEEP` instruction will execute as a NOP.
 - WDT and WDT prescaler will not be cleared
 - \overline{TO} bit of the STATUS register will not be set
 - \overline{PD} bit of the STATUS register will not be cleared.
- If the interrupt occurs **during or after** the execution of a `SLEEP` instruction

- `SLEEP` instruction will be completely executed
- Device will immediately wake-up from Sleep
- WDT and WDT prescaler will be cleared
- \overline{TO} bit of the STATUS register will be set
- \overline{PD} bit of the STATUS register will be cleared.

Even if the flag bits were checked before executing a `SLEEP` instruction, it may be possible for flag bits to become set before the `SLEEP` instruction completes. To determine whether a `SLEEP` instruction executed, test the \overline{PD} bit. If the \overline{PD} bit is set, the `SLEEP` instruction was executed as a NOP.

FIGURE 8-1: WAKE-UP FROM SLEEP THROUGH INTERRUPT



PIC16(L)F1526/7

8.2 Low-Power Sleep Mode

The PIC16F1526 device contains an internal Low Dropout (LDO) voltage regulator, which allows the device I/O pins to operate at voltages up to 5.5V while the internal device logic operates at a lower voltage. The LDO and its associated reference circuitry must remain active when the device is in Sleep mode. The PIC16F1526 allows the user to optimize the operating current in Sleep, depending on the application requirements.

A Low-Power Sleep mode can be selected by setting the VREGPM bit of the VREGCON register. With this bit set, the LDO and reference circuitry are placed in a low-power state when the device is in Sleep.

8.2.1 SLEEP CURRENT VS. WAKE-UP TIME

In the default operating mode, the LDO and reference circuitry remain in the normal configuration while in Sleep. The device is able to exit Sleep mode quickly since all circuits remain active. In Low-Power Sleep mode, when waking up from Sleep, an extra delay time is required for these circuits to return to the normal configuration and stabilize.

The Low-Power Sleep mode is beneficial for applications that stay in Sleep mode for long periods of time. The normal mode is beneficial for applications that need to wake from Sleep quickly and frequently.

8.2.2 PERIPHERAL USAGE IN SLEEP

Some peripherals that can operate in Sleep mode will not operate properly with the Low-Power Sleep mode selected. The LDO will remain in the normal power mode when those peripherals are enabled. The Low-Power Sleep mode is intended for use with these peripherals:

- Brown-Out Reset (BOR)
- Watchdog Timer (WDT)
- External interrupt pin/Interrupt-on-change pins
- Timer1 (with external clock source)
- CCP (Capture mode)

Note: The PIC16LF1526/7 does not have a configurable Low-Power Sleep mode. PIC16LF1526/7 is an unregulated device and is always in the lowest power state when in Sleep, with no wake-up time penalty. This device has a lower maximum V_{DD} and I/O voltage than the PIC16LF1526/7. See [Section 25.0 “Electrical Specifications”](#) for more information.

8.3 Register Definitions: Voltage Regulator Control

REGISTER 8-1: VREGCON: VOLTAGE REGULATOR CONTROL REGISTER⁽¹⁾

| | | | | | | | |
|-------|-----|-----|-----|-----|-----|---------|----------|
| U-0 | U-0 | U-0 | U-0 | U-0 | U-0 | R/W-0/0 | R/W-1/1 |
| — | — | — | — | — | — | VREGPM | Reserved |
| bit 7 | | | | | | bit 0 | |

Legend:

| | | |
|----------------------|----------------------|---|
| R = Readable bit | W = Writable bit | U = Unimplemented bit, read as '0' |
| u = Bit is unchanged | x = Bit is unknown | -n/n = Value at POR and BOR/Value at all other Resets |
| '1' = Bit is set | '0' = Bit is cleared | |

- bit 7-2 **Unimplemented:** Read as '0'
- bit 1 **VREGPM:** Voltage Regulator Power Mode Selection bit
 1 = Low-Power Sleep mode enabled in Sleep⁽²⁾
 Draws lowest current in Sleep, slower wake-up
 0 = Normal Power mode enabled in Sleep⁽²⁾
 Draws higher current in Sleep, faster wake-up
- bit 0 **Reserved:** Read as '1'. Maintain this bit set.

- Note 1:** PIC16F1526/7 only.
2: See Section 25.0 "Electrical Specifications".

TABLE 8-1: SUMMARY OF REGISTERS ASSOCIATED WITH POWER-DOWN MODE

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Register on Page |
|------------------------|---------|---------|------------|--------|--------|---------|--------|----------|------------------|
| INTCON | GIE | PEIE | TMR0IE | INTE | IOCFIE | TMR0IF | INTF | IOCFIF | 76 |
| IOCBF | IOCBF7 | IOCBF6 | IOCBF5 | IOCBF4 | IOCBF3 | IOCBF2 | IOCBF1 | IOCBF0 | 137 |
| IOCBN | IOCBN7 | IOCBN6 | IOCBN5 | IOCBN4 | IOCBN3 | IOCBN2 | IOCBN1 | IOCBN0 | 137 |
| IOCBP | IOCBP7 | IOCBP6 | IOCBP5 | IOCBP4 | IOCBP3 | IOCBP2 | IOCBP1 | IOCBP0 | 137 |
| PIE1 | TMR1GIE | ADIE | RC1IE | TX1IE | SSP1IE | CCP1IE | TMR2IE | TMR1IE | 77 |
| PIE2 | OSFIE | TMR5GIE | TMR3GIE | — | BCLIE | TMR10IE | TMR8IE | CCP2IE | 78 |
| PIE3 | CCP6IE | CCP5IE | CCP4IE | CCP3IE | TMR6IE | TMR5IE | TMR4IE | TMR3IE | 79 |
| PIE4 | CCP10IE | CCP9IE | RC2IE | TX2IE | CCP8IE | CCP7IE | BCL2IE | SSP2IE | 80 |
| PIR1 | TMR1GIF | ADIF | RC1IF | TX1IF | SSP1IF | CCP1IF | TMR2IF | TMR1IF | 81 |
| PIR2 | OSFIF | TMR5GIF | TMR3GIF | — | BCL1IF | TMR10IF | TMR8IF | CCP2IF | 82 |
| PIR3 | CCP6IF | CCP5IF | CCP4IF | CCP3IF | TMR6IF | TMR5IF | TMR4IF | TMR3IF | 83 |
| PIR4 | CCP10IF | CCP9IF | RC2IF | TX2IF | CCP8IF | CCP7IF | BCL2IF | SSP2IF | 84 |
| STATUS | — | — | — | TO | PD | Z | DC | C | 21 |
| VREGCON ⁽¹⁾ | — | — | — | — | — | — | VREGPM | Reserved | 89 |
| WDTCON | — | — | WDTPS<4:0> | | | | | SWDTEN | 93 |

Legend: — = unimplemented, read as '0'. Shaded cells are not used in Power-down mode.

Note 1: PIC16F1526/7 only.

PIC16(L)F1526/7

9.0 LOW DROPOUT (LDO) VOLTAGE REGULATOR

The PIC16F1526/7 has an internal Low Dropout Regulator (LDO) which provides operation above 3.6V. The LDO regulates a voltage for the internal device logic while permitting the VDD and I/O pins to operate at a higher voltage. There is no user enable/disable control available for the LDO, it is always active. The PIC16LF1526/7 operates at a maximum VDD of 3.6V and does not incorporate an LDO.

A device I/O pin may be configured as the LDO voltage output, identified as the VCAP pin. Although not required, an external low-ESR capacitor may be connected to the VCAP pin for additional regulator stability.

The VCAPEN bit of Configuration Words determines which pin is assigned as the VCAP pin. Refer to [Table 9-1](#).

On power-up, the external capacitor will load the LDO voltage regulator. To prevent erroneous operation, the device is held in Reset while a constant current source charges the external capacitor. After the cap is fully charged, the device is released from Reset. For more information on the constant current rate, refer to the LDO Regulator Characteristics Table in [Section 25.0, Electrical Specifications](#).

TABLE 9-1: VCAPEN SELECT BIT

| VCAPEN | Pin |
|--------|---------|
| 0 | RF0 |
| 1 | No Vcap |

TABLE 9-2: SUMMARY OF CONFIGURATION WORD WITH LDO

| Name | Bits | Bit -/7 | Bit -/6 | Bit 13/5 | Bit 12/4 | Bit 11/3 | Bit 10/2 | Bit 9/1 | Bit 8/0 | Register on Page |
|---------|------|---------|---------|----------|----------------------------------|---------------------------|----------|----------|---------|------------------|
| CONFIG2 | 13:8 | — | | LVP | $\overline{\text{DEBUG}}$ | $\overline{\text{LPBOR}}$ | BORV | STVREN | — | 45 |
| | 7:0 | — | — | — | $\overline{\text{VCAPEN}}^{(1)}$ | — | — | WRT<1:0> | | |

Legend: — = unimplemented locations read as '0'. Shaded cells are not used by LDO.

Note 1: PIC16F1526/7 only.

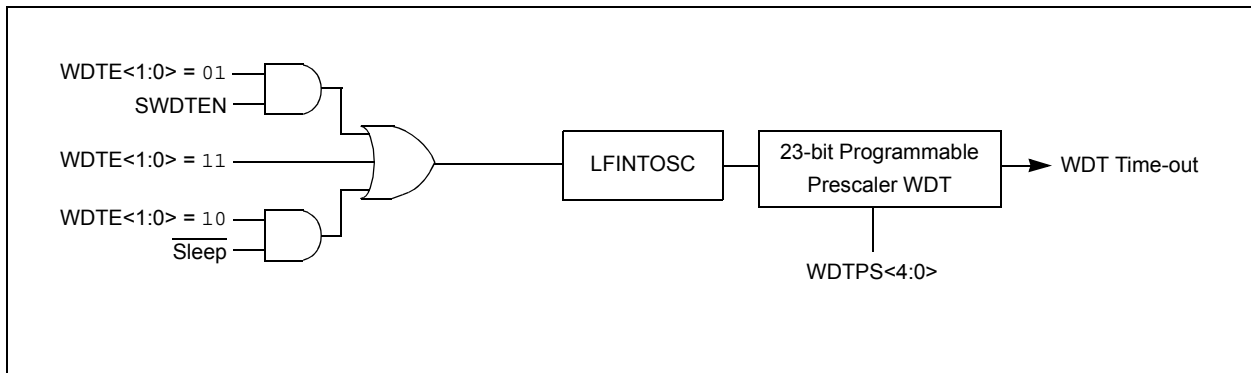
10.0 WATCHDOG TIMER (WDT)

The Watchdog Timer is a system timer that generates a Reset if the firmware does not issue a `CLRWDT` instruction within the time-out period. The Watchdog Timer is typically used to recover the system from unexpected events.

The WDT has the following features:

- Independent clock source
- Multiple operating modes
 - WDT is always on
 - WDT is off when in Sleep
 - WDT is controlled by software
 - WDT is always off
- Configurable time-out period is from 1 ms to 256 seconds (nominal)
- Multiple Reset conditions
- Operation during Sleep

FIGURE 10-1: WATCHDOG TIMER BLOCK DIAGRAM



PIC16(L)F1526/7

10.1 Independent Clock Source

The WDT derives its time base from the 31 kHz LFINTOSC internal oscillator. Time intervals in this chapter are based on a nominal interval of 1 ms. See [Section 25.0 “Electrical Specifications”](#) for the LFINTOSC tolerances.

10.2 WDT Operating Modes

The Watchdog Timer module has four operating modes controlled by the WDTE<1:0> bits in Configuration Words. See [Table 10-1](#).

10.2.1 WDT IS ALWAYS ON

When the WDTE bits of Configuration Words are set to ‘11’, the WDT is always on.

WDT protection is active during Sleep.

10.2.2 WDT IS OFF IN SLEEP

When the WDTE bits of Configuration Words are set to ‘10’, the WDT is on, except in Sleep.

WDT protection is not active during Sleep.

10.2.3 WDT CONTROLLED BY SOFTWARE

When the WDTE bits of Configuration Words are set to ‘01’, the WDT is controlled by the SWDTEN bit of the WDTCON register.

WDT protection is unchanged by Sleep. See [Table 10-1](#) for more details.

TABLE 10-1: WDT OPERATING MODES

| WDTE<1:0> | SWDTEN | Device Mode | WDT Mode |
|-----------|--------|-------------|----------|
| 11 | X | X | Active |
| 10 | X | Awake | Active |
| | | Sleep | Disabled |
| 01 | 1 | X | Active |
| | 0 | X | Disabled |
| 00 | X | X | Disabled |

TABLE 10-2: WDT CLEARING CONDITIONS

| Conditions | WDT |
|---|------------------------------|
| WDTE<1:0> = 00 | Cleared |
| WDTE<1:0> = 01 and SWDTEN = 0 | |
| WDTE<1:0> = 10 and enter Sleep | |
| CLRWDT Command | |
| Oscillator Fail Detected | |
| Exit Sleep + System Clock = SOSC, EXTRC, INTOSC, EXTCLK | |
| Exit Sleep + System Clock = XT, HS, LP | Cleared until the end of OST |
| Change INTOSC divider (IRCF bits) | Unaffected |

10.3 Time-out Period

The WDTPS bits of the WDTCON register set the time-out period from 1 ms to 256 seconds (nominal). After a Reset, the default time-out period is 2 seconds.

10.4 Clearing the WDT

The WDT is cleared when any of the following conditions occur:

- Any Reset
- CLRWDT instruction is executed
- Device enters Sleep
- Device wakes up from Sleep
- Oscillator fail
- WDT is disabled
- Oscillator Start-up Timer (OST) is running

See [Table 10-2](#) for more information.

10.5 Operation During Sleep

When the device enters Sleep, the WDT is cleared. If the WDT is enabled during Sleep, the WDT resumes counting.

When the device exits Sleep, the WDT is cleared again. The WDT remains clear until the OST, if enabled, completes. See [Section 5.0 “Oscillator Module \(with Fail-Safe Clock Monitor\)”](#) for more information on the OST.

When a WDT time-out occurs while the device is in Sleep, no Reset is generated. Instead, the device wakes up and resumes operation. The TO and PD bits in the STATUS register are changed to indicate the event. The RWDT bit in the PCON register can also be used. See [Section 3.0 “Memory Organization”](#) and The STATUS register ([Register 3-1](#)) for more information.

10.6 Register Definitions: Watchdog Control

REGISTER 10-1: WDTCON: WATCHDOG TIMER CONTROL REGISTER

| | | | | | | | |
|-------|-----|------------|---------|---------|---------|---------|---------|
| U-0 | U-0 | R/W-0/0 | R/W-1/1 | R/W-0/0 | R/W-1/1 | R/W-1/1 | R/W-0/0 |
| — | — | WDTPS<4:0> | | | | | SWDTEN |
| bit 7 | | | | | | | bit 0 |

Legend:

| | | |
|----------------------|----------------------|---|
| R = Readable bit | W = Writable bit | U = Unimplemented bit, read as '0' |
| u = Bit is unchanged | x = Bit is unknown | -m/n = Value at POR and BOR/Value at all other Resets |
| '1' = Bit is set | '0' = Bit is cleared | |

bit 7-6 **Unimplemented:** Read as '0'

bit 5-1 **WDTPS<4:0>:** Watchdog Timer Period Select bits⁽¹⁾

Bit Value = Prescale Rate

11111 = Reserved. Results in minimum interval (1:32)

•
•
•

10011 = Reserved. Results in minimum interval (1:32)

10010 = 1:8388608 (2^{23}) (Interval 256s nominal)

10001 = 1:4194304 (2^{22}) (Interval 128s nominal)

10000 = 1:2097152 (2^{21}) (Interval 64s nominal)

01111 = 1:1048576 (2^{20}) (Interval 32s nominal)

01110 = 1:524288 (2^{19}) (Interval 16s nominal)

01101 = 1:262144 (2^{18}) (Interval 8s nominal)

01100 = 1:131072 (2^{17}) (Interval 4s nominal)

01011 = 1:65536 (Interval 2s nominal) (Reset value)

01010 = 1:32768 (Interval 1s nominal)

01001 = 1:16384 (Interval 512 ms nominal)

01000 = 1:8192 (Interval 256 ms nominal)

00111 = 1:4096 (Interval 128 ms nominal)

00110 = 1:2048 (Interval 64 ms nominal)

00101 = 1:1024 (Interval 32 ms nominal)

00100 = 1:512 (Interval 16 ms nominal)

00011 = 1:256 (Interval 8 ms nominal)

00010 = 1:128 (Interval 4 ms nominal)

00001 = 1:64 (Interval 2 ms nominal)

00000 = 1:32 (Interval 1 ms nominal)

bit 0 **SWDTEN:** Software Enable/Disable for Watchdog Timer bit

If WDTE<1:0> = 00:

This bit is ignored.

If WDTE<1:0> = 01:

1 = WDT is turned on

0 = WDT is turned off

If WDTE<1:0> = 1x:

This bit is ignored.

Note 1: Times are approximate. WDT time is based on 31 kHz LFINTOSC.

PIC16(L)F1526/7

TABLE 10-3: SUMMARY OF REGISTERS ASSOCIATED WITH WATCHDOG TIMER

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Register on Page |
|--------|-------|-----------|------------|-----------------|-----------------|-------|----------|--------|------------------|
| OSCCON | — | IRCF<3:0> | | | | — | SCS<1:0> | | 61 |
| STATUS | — | — | — | \overline{TO} | \overline{PD} | Z | DC | C | 21 |
| WDTCON | — | — | WDTPS<4:0> | | | | | SWDTEN | 93 |

Legend: x = unknown, u = unchanged, — = unimplemented locations read as '0'. Shaded cells are not used by Watchdog Timer.

TABLE 10-4: SUMMARY OF CONFIGURATION WORD WITH WATCHDOG TIMER

| Name | Bits | Bit -/7 | Bit -/6 | Bit 13/5 | Bit 12/4 | Bit 11/3 | Bit 10/2 | Bit 9/1 | Bit 8/0 | Register on Page |
|---------|------|-----------------|---------|--------------------|-----------|-----------------------|------------|---------|---------|------------------|
| CONFIG1 | 13:8 | — | | FCMEN | IESO | $\overline{CLKOUTEN}$ | BOREN<1:0> | | — | 43 |
| | 7:0 | \overline{CP} | MCLRE | \overline{PWRTE} | WDTE<1:0> | | FOSC<2:0> | | | |

Legend: — = unimplemented location, read as '0'. Shaded cells are not used by Watchdog Timer.

11.0 FLASH PROGRAM MEMORY CONTROL

The Flash program memory is readable and writable during normal operation over the full VDD range. Program memory is indirectly addressed using Special Function Registers (SFRs). The SFRs used to access program memory are:

- PMCON1
- PMCON2
- PMDATL
- PMDATH
- PMADRL
- PMADRH

When accessing the program memory, the PMDATH:PMDATL register pair forms a 2-byte word that holds the 14-bit data for read/write, and the PMADRH:PMADRL register pair forms a 2-byte word that holds the 15-bit address of the program memory location being read.

The write time is controlled by an on-chip timer. The write/erase voltages are generated by an on-chip charge pump rated to operate over the operating voltage range of the device.

The Flash program memory can be protected in two ways; by code protection (\overline{CP} bit in Configuration Words) and write protection (WRT<1:0> bits in Configuration Words).

Code protection ($\overline{CP} = 0$)⁽¹⁾, disables access, reading and writing, to the Flash program memory via external device programmers. Code protection does not affect the self-write and erase functionality. Code protection can only be reset by a device programmer performing a Bulk Erase to the device, clearing all Flash program memory, Configuration bits and User IDs.

Write protection prohibits self-write and erase to a portion or all of the Flash program memory as defined by the bits WRT<1:0>. Write protection does not affect a device programmers ability to read, write or erase the device.

Note 1: Code protection of the entire Flash program memory array is enabled by clearing the \overline{CP} bit of Configuration Words.

11.1 PMADRL and PMADRH Registers

The PMADRH:PMADRL register pair can address up to a maximum of 32K words of program memory. When selecting a program address value, the MSB of the address is written to the PMADRH register and the LSB is written to the PMADRL register.

11.1.1 PMCON1 AND PMCON2 REGISTERS

PMCON1 is the control register for Flash program memory accesses.

Control bits RD and WR initiate read and write, respectively. These bits cannot be cleared, only set, in software. They are cleared by hardware at completion of the read or write operation. The inability to clear the WR bit in software prevents the accidental, premature termination of a write operation.

The WREN bit, when set, will allow a write operation to occur. On power-up, the WREN bit is clear. The WRERR bit is set when a write operation is interrupted by a Reset during normal operation. In these situations, following Reset, the user can check the WRERR bit and execute the appropriate error handling routine.

The PMCON2 register is a write-only register. Attempting to read the PMCON2 register will return all '0's.

To enable writes to the program memory, a specific pattern (the unlock sequence), must be written to the PMCON2 register. The required unlock sequence prevents inadvertent writes to the program memory write latches and Flash program memory.

11.2 Flash Program Memory Overview

It is important to understand the Flash program memory structure for erase and programming operations. Flash program memory is arranged in rows. A row consists of a fixed number of 14-bit program memory words. A row is the minimum size that can be erased by user software.

After a row has been erased, the user can reprogram all or a portion of this row. Data to be written into the program memory row is written to 14-bit wide data write latches. These write latches are not directly accessible to the user, but may be loaded via sequential writes to the PMDATH:PMDATL register pair.

Note: If the user wants to modify only a portion of a previously programmed row, then the contents of the entire row must be read and saved in RAM prior to the erase. Then, new data and retained data can be written into the write latches to reprogram the row of Flash program memory. However, any unprogrammed locations can be written without first erasing the row. In this case, it is not necessary to save and rewrite the other previously programmed locations.

See [Table 11-1](#) for Erase Row size and the number of write latches for Flash program memory.

PIC16(L)F1526/7

TABLE 11-1: FLASH MEMORY ORGANIZATION BY DEVICE

| Device | Row Erase (words) | Write Latches (words) |
|---------------|-------------------|-----------------------|
| PIC16(L)F1526 | 32 | 32 |
| PIC16(L)F1527 | | |

11.2.1 READING THE FLASH PROGRAM MEMORY

To read a program memory location, the user must:

1. Write the desired address to the PMADRH:PMADRL register pair.
2. Clear the CFGS bit of the PMCON1 register.
3. Then, set control bit RD of the PMCON1 register.

Once the read control bit is set, the program memory Flash controller will use the second instruction cycle to read the data. This causes the second instruction immediately following the "BSF PMCON1, RD" instruction to be ignored. The data is available in the very next cycle, in the PMDATH:PMDATL register pair; therefore, it can be read as two bytes in the following instructions.

PMDATH:PMDATL register pair will hold this value until another read or until it is written to by the user.

Note: The two instructions following a program memory read are required to be NOPs. This prevents the user from executing a two-cycle instruction on the next instruction after the RD bit is set.

FIGURE 11-1: FLASH PROGRAM MEMORY READ FLOWCHART

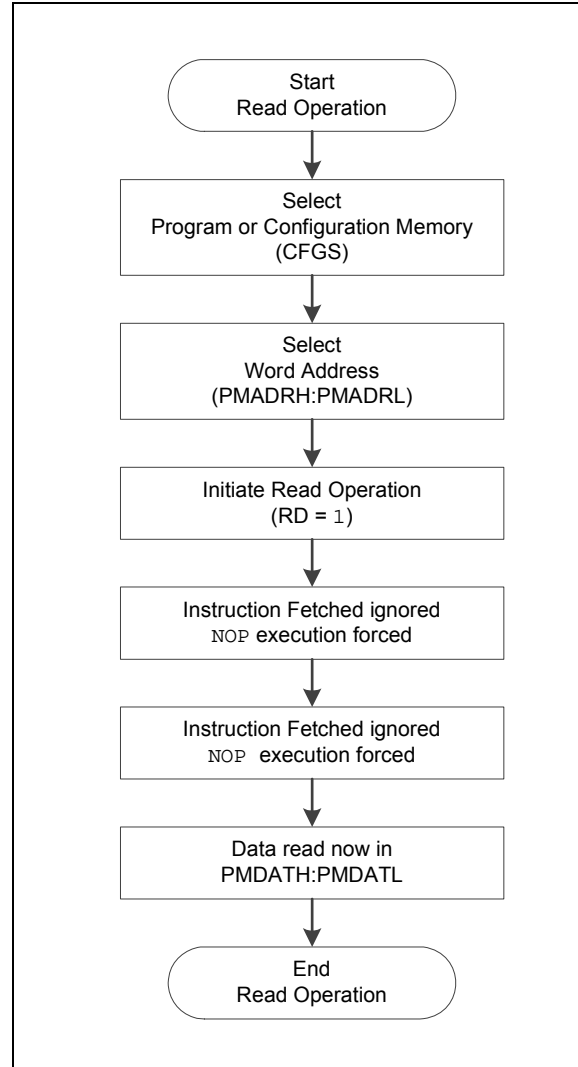
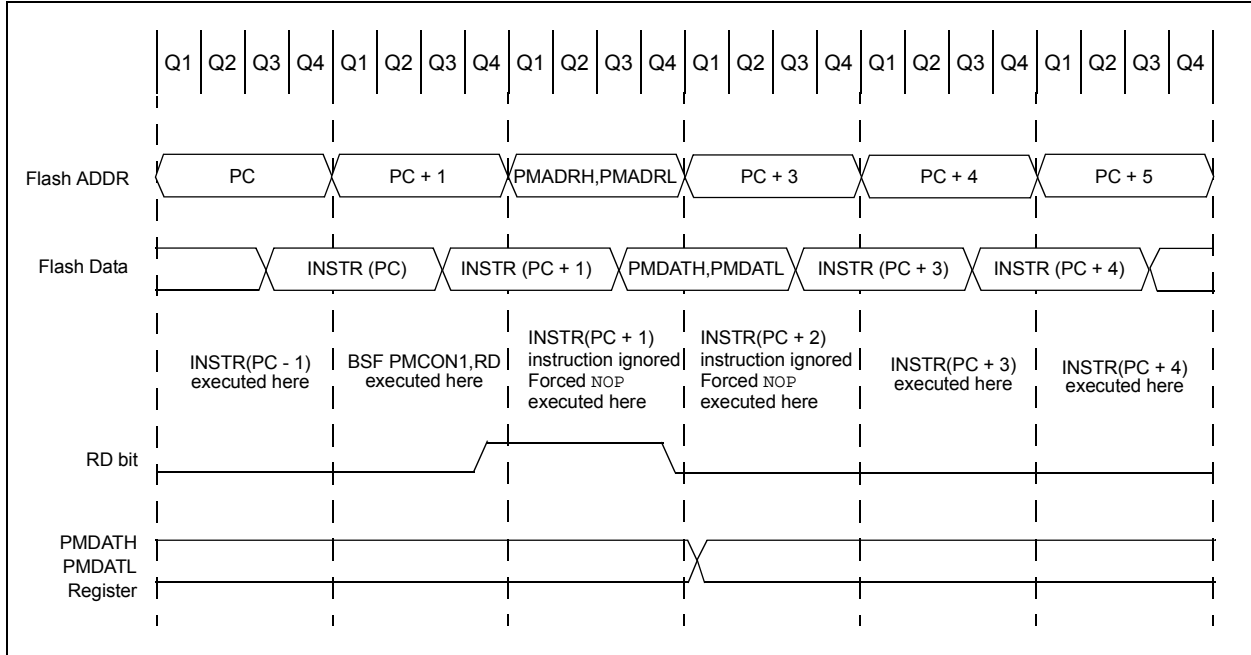


FIGURE 11-2: FLASH PROGRAM MEMORY READ CYCLE EXECUTION



EXAMPLE 11-1: FLASH PROGRAM MEMORY READ

```

* This code block will read 1 word of program
* memory at the memory address:
  PROG_ADDR_HI : PROG_ADDR_LO
* data will be returned in the variables;
*  PROG_DATA_HI, PROG_DATA_LO

  BANKSEL  PMADRL          ; Select Bank for PMCON registers
  MOVLW   PROG_ADDR_LO    ;
  MOVWF   PMADRL          ; Store LSB of address
  MOVLW   PROG_ADDR_HI    ;
  MOVWL   PMADRH          ; Store MSB of address

  BCF     PMCON1,CFGSS    ; Do not select Configuration Space
  BSF     PMCON1,RD       ; Initiate read
  NOP     ; Ignored (Figure 11-2)
  NOP     ; Ignored (Figure 11-2)

  MOVF    PMDATL,W        ; Get LSB of word
  MOVWF   PROG_DATA_LO    ; Store in user location
  MOVF    PMDATH,W        ; Get MSB of word
  MOVWF   PROG_DATA_HI    ; Store in user location

```

PIC16(L)F1526/7

11.2.2 FLASH MEMORY UNLOCK SEQUENCE

The unlock sequence is a mechanism that protects the Flash program memory from unintended self-write programming or erasing. The sequence must be executed and completed without interruption to successfully complete any of the following operations:

- Row Erase
- Load program memory write latches
- Write of program memory write latches to program memory
- Write of program memory write latches to User IDs

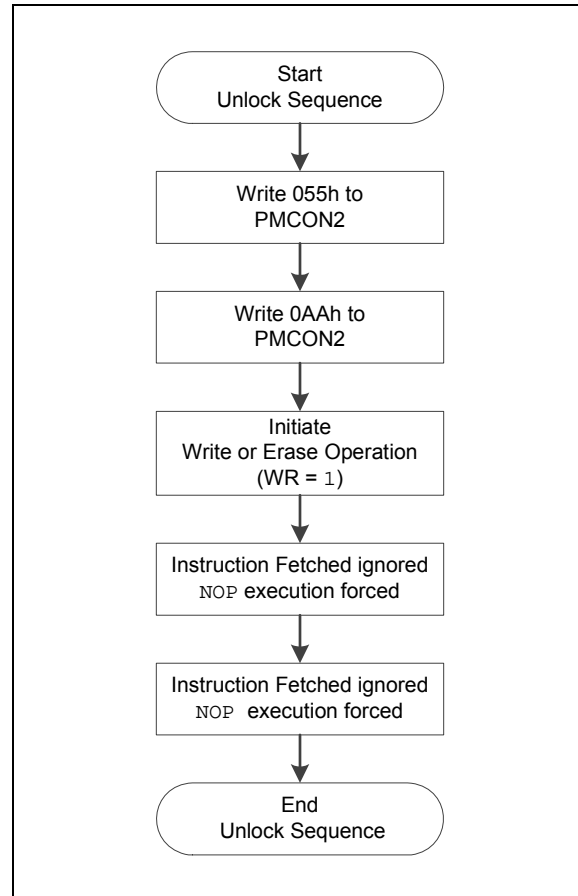
The unlock sequence consists of the following steps:

1. Write 55h to PMCON2
2. Write AAh to PMCON2
3. Set the WR bit in PMCON1
4. NOP instruction
5. NOP instruction

Once the WR bit is set, the processor will always force two NOP instructions. When an Erase Row or Program Row operation is being performed, the processor will stall internal operations (typical 2 ms), until the operation is complete and then resume with the next instruction. When the operation is loading the program memory write latches, the processor will always force the two NOP instructions and continue uninterrupted with the next instruction.

Since the unlock sequence must not be interrupted, global interrupts should be disabled prior to the unlock sequence and re-enabled after the unlock sequence is completed.

FIGURE 11-3: FLASH PROGRAM MEMORY UNLOCK SEQUENCE FLOWCHART



11.2.3 ERASING FLASH PROGRAM MEMORY

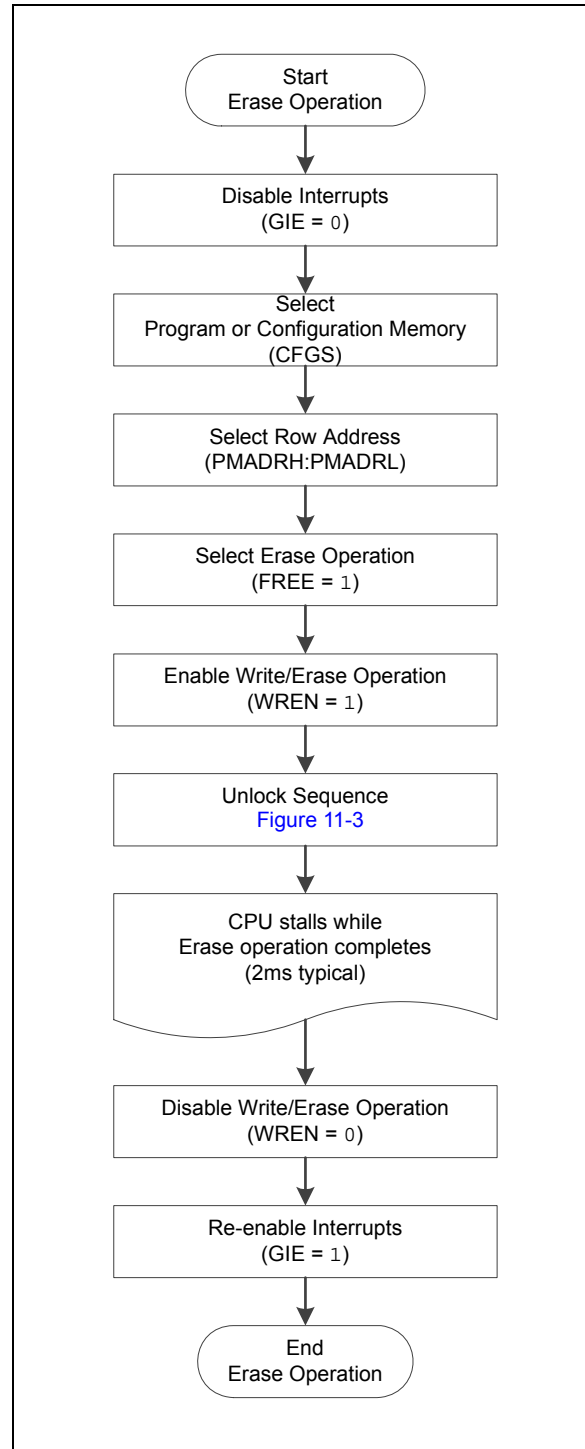
While executing code, program memory can only be erased by rows. To erase a row:

1. Load the PMADRH:PMADRL register pair with any address within the row to be erased.
2. Clear the CFGS bit of the PMCON1 register.
3. Set the FREE and WREN bits of the PMCON1 register.
4. Write 55h, then AAh, to PMCON2 (Flash programming unlock sequence).
5. Set control bit WR of the PMCON1 register to begin the erase operation.

See [Example 11-2](#).

After the “BSF PMCON1, WR” instruction, the processor requires two cycles to set up the erase operation. The user must place two NOP instructions immediately following the WR bit set instruction. The processor will halt internal operations for the typical 2 ms erase time. This is not Sleep mode as the clocks and peripherals will continue to run. After the erase cycle, the processor will resume operation with the third instruction after the PMCON1 WRITE instruction.

FIGURE 11-4: FLASH PROGRAM MEMORY ERASE FLOWCHART



PIC16(L)F1526/7

EXAMPLE 11-2: ERASING ONE ROW OF PROGRAM MEMORY

```
; This row erase routine assumes the following:
; 1. A valid address within the erase row is loaded in ADDRH:ADDRL
; 2. ADDRH and ADDRL are located in shared data memory 0x70 - 0x7F (common RAM)

        BCF      INTCON,GIE      ; Disable ints so required sequences will execute properly
        BANKSEL PMADRL
        MOVF     ADDRL,W         ; Load lower 8 bits of erase address boundary
        MOVWF   PMADRL
        MOVF     ADDRH,W        ; Load upper 6 bits of erase address boundary
        MOVWF   PMADRH
        BCF     PMCON1,CFG5      ; Not configuration space
        BSF     PMCON1,FREE      ; Specify an erase operation
        BSF     PMCON1,WREN      ; Enable writes

        MOVLW   55h             ; Start of required sequence to initiate erase
        MOVWF   PMCON2          ; Write 55h
        MOVLW   0AAh            ;
        MOVWF   PMCON2          ; Write AAh
        BSF     PMCON1,WR       ; Set WR bit to begin erase
        NOP     ; NOP instructions are forced as processor starts
        NOP     ; row erase of program memory.
        ;
        ; The processor stalls until the erase process is complete
        ; after erase processor continues with 3rd instruction

        BCF     PMCON1,WREN      ; Disable writes
        BSF     INTCON,GIE      ; Enable interrupts
```

Required
Sequence

11.2.4 WRITING TO FLASH PROGRAM MEMORY

Program memory is programmed using the following steps:

1. Load the address in PMADRH:PMADRL of the row to be programmed.
2. Load each write latch with data.
3. Initiate a programming operation.
4. Repeat steps 1 through 3 until all data is written.

Before writing to program memory, the word(s) to be written must be erased or previously unwritten. Program memory can only be erased one row at a time. No automatic erase occurs upon the initiation of the write.

Program memory can be written one or more words at a time. The maximum number of words written at one time is equal to the number of write latches. See [Figure 11-5](#) (row writes to program memory with 32 write latches) for more details.

The write latches are aligned to the Flash row address boundary defined by the upper 10-bits of PMADRH:PMADRL, (PMADRH<6:0>:PMADRL<7:5>) with the lower 5-bits of PMADRL, (PMADRL<4:0>) determining the write latch being loaded. Write operations do not cross these boundaries. At the completion of a program memory write operation, the data in the write latches is reset to contain 0x3FFF.

The following steps should be completed to load the write latches and program a row of program memory. These steps are divided into two parts. First, each write latch is loaded with data from the PMDATH:PMDATL using the unlock sequence with LWLO = 1. When the last word to be loaded into the write latch is ready, the LWLO bit is cleared and the unlock sequence executed. This initiates the programming operation, writing all the latches into Flash program memory.

Note: The special unlock sequence is required to load a write latch with data or initiate a Flash programming operation. If the unlock sequence is interrupted, writing to the latches or program memory will not be initiated.

1. Set the WREN bit of the PMCON1 register.
2. Clear the CFGS bit of the PMCON1 register.
3. Set the LWLO bit of the PMCON1 register. When the LWLO bit of the PMCON1 register is '1', the write sequence will only load the write latches and will not initiate the write to Flash program memory.
4. Load the PMADRH:PMADRL register pair with the address of the location to be written.
5. Load the PMDATH:PMDATL register pair with the program memory data to be written.
6. Execute the unlock sequence ([Section 11.2.2 "Flash Memory Unlock Sequence"](#)). The write latch is now loaded.
7. Increment the PMADRH:PMADRL register pair to point to the next location.
8. Repeat steps 5 through 7 until all but the last write latch has been loaded.
9. Clear the LWLO bit of the PMCON1 register. When the LWLO bit of the PMCON1 register is '0', the write sequence will initiate the write to Flash program memory.
10. Load the PMDATH:PMDATL register pair with the program memory data to be written.
11. Execute the unlock sequence ([Section 11.2.2 "Flash Memory Unlock Sequence"](#)). The entire program memory latch content is now written to Flash program memory.

Note: The program memory write latches are reset to the blank state (0x3FFF) at the completion of every write or erase operation. As a result, it is not necessary to load all the program memory write latches. Unloaded latches will remain in the blank state.

An example of the complete write sequence is shown in [Example 11-3](#). The initial address is loaded into the PMADRH:PMADRL register pair; the data is loaded using indirect addressing.

FIGURE 11-5: BLOCK WRITES TO FLASH PROGRAM MEMORY WITH 32 WRITE LATCHES

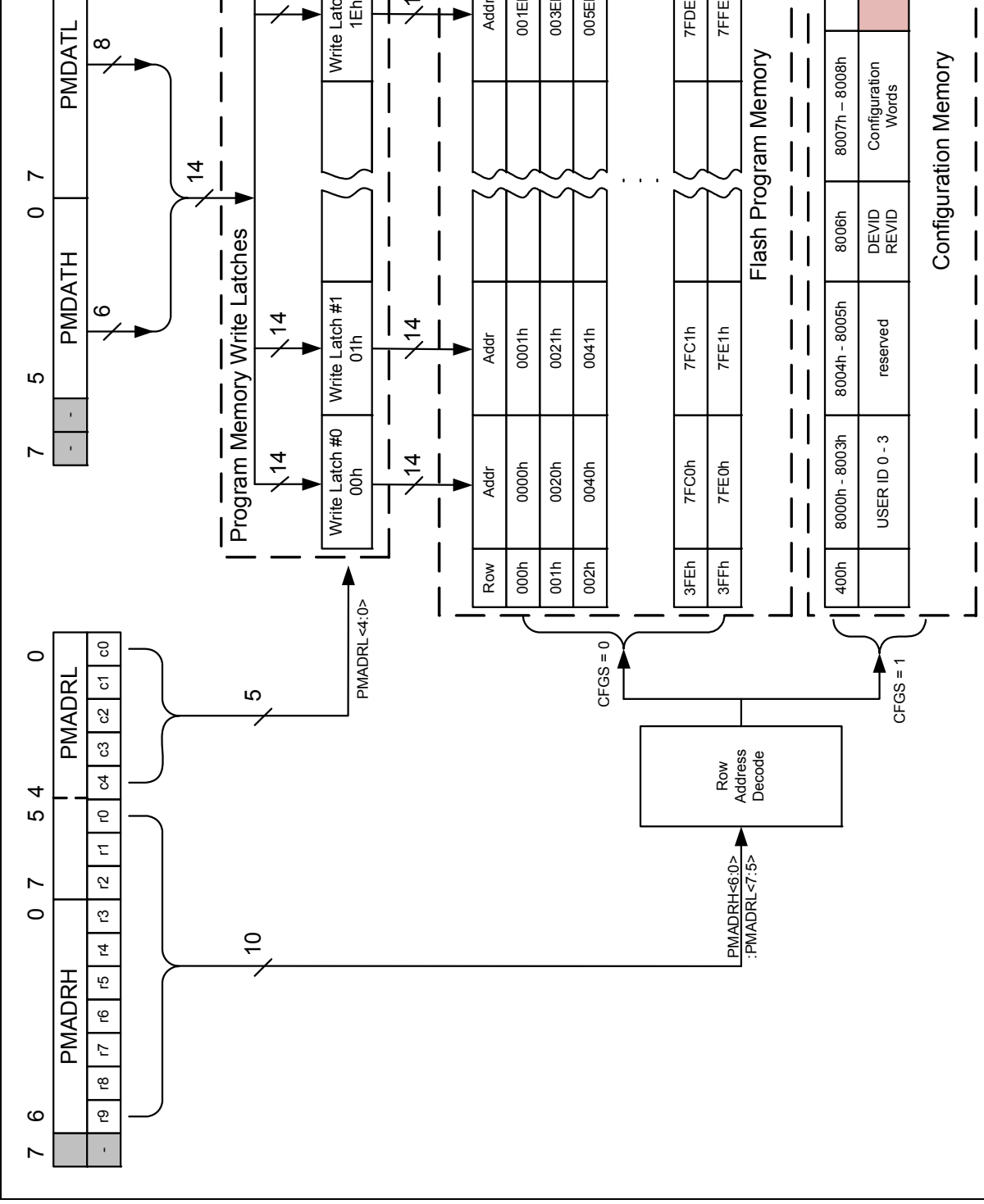
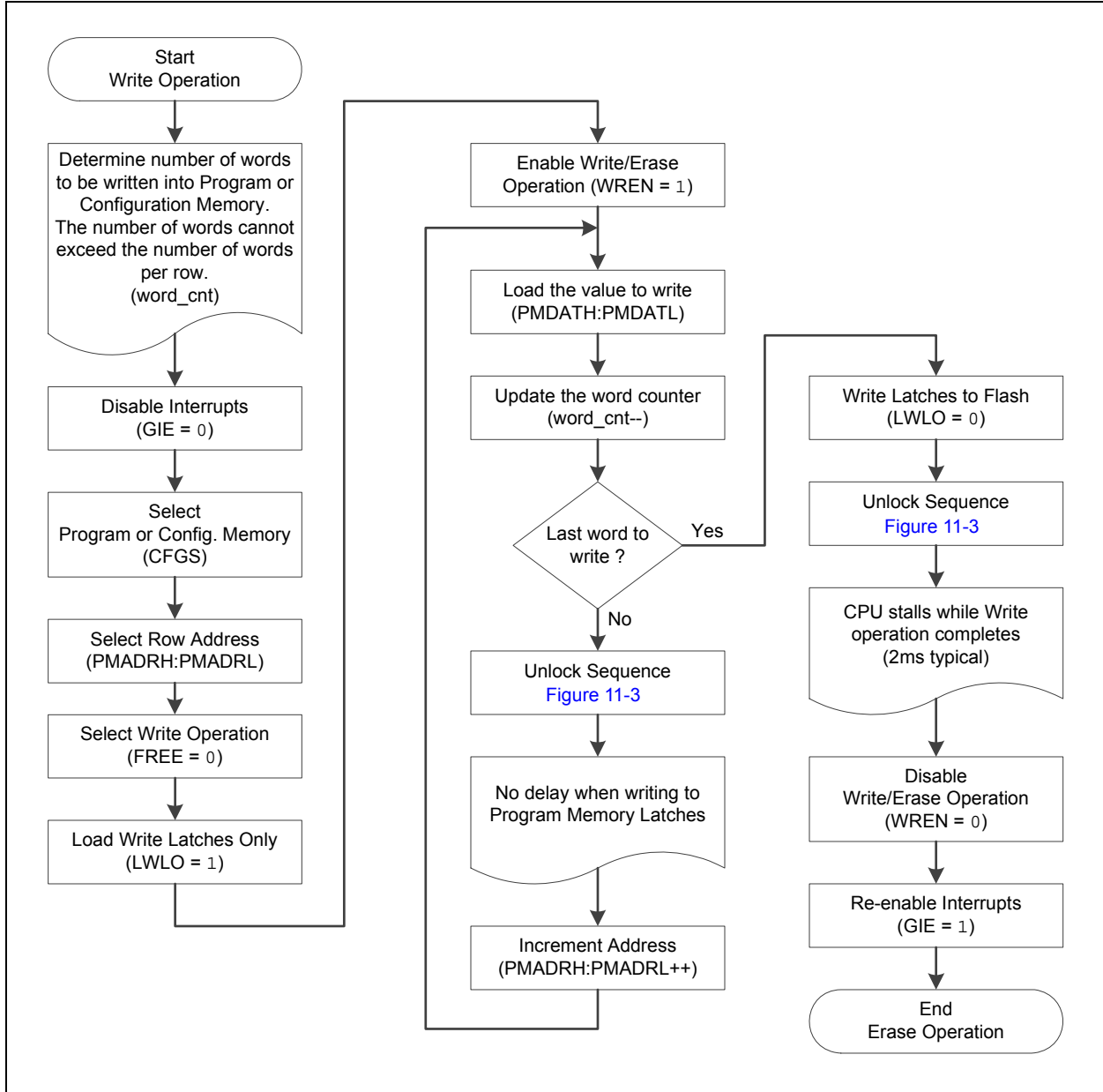


FIGURE 11-6: FLASH PROGRAM MEMORY WRITE FLOWCHART



PIC16(L)F1526/7

EXAMPLE 11-3: WRITING TO FLASH PROGRAM MEMORY

```
; This write routine assumes the following:
; 1. 64 bytes of data are loaded, starting at the address in DATA_ADDR
; 2. Each word of data to be written is made up of two adjacent bytes in DATA_ADDR,
; stored in little endian format
; 3. A valid starting address (the least significant bits = 00000) is loaded in ADDRH:ADDRL
; 4. ADDRH and ADDRL are located in shared data memory 0x70 - 0x7F (common RAM)
;
    BCF      INTCON,GIE      ; Disable ints so required sequences will execute properly
    BANKSEL PMADRH          ; Bank 3
    MOVF    ADDRH,W          ; Load initial address
    MOVWF   PMADRH          ;
    MOVF    ADDRHL,W        ;
    MOVWF   PMADRL         ;
    MOVLW   LOW DATA_ADDR  ; Load initial data address
    MOVWF   FSR0L          ;
    MOVLW   HIGH DATA_ADDR ; Load initial data address
    MOVWF   FSR0H          ;
    BCF     PMCON1,CFGSS    ; Not configuration space
    BSF     PMCON1,WREN     ; Enable writes
    BSF     PMCON1,LWLO     ; Only Load Write Latches

LOOP
    MOVIW   FSR0++         ; Load first data byte into lower
    MOVWF   PMDATH         ;
    MOVIW   FSR0++         ; Load second data byte into upper
    MOVWF   PMDATH         ;

    MOVF    PMADRL,W       ; Check if lower bits of address are '00000'
    XORLW   0x1F           ; Check if we're on the last of 32 addresses
    ANDLW   0x1F           ;
    BTFSC   STATUS,Z       ; Exit if last of 32 words,
    GOTO    START_WRITE    ;

    [ Required Sequence
    MOVLW   55h            ; Start of required write sequence:
    MOVWF   PMCON2         ; Write 55h
    MOVLW   0AAh           ;
    MOVWF   PMCON2         ; Write AAh
    BSF     PMCON1,WR       ; Set WR bit to begin write
    NOP     ; NOP instructions are forced as processor
    ; loads program memory write latches
    NOP     ;

    INCF    PMADRL,F       ; Still loading latches Increment address
    GOTO    LOOP           ; Write next latches

START_WRITE
    BCF     PMCON1,LWLO     ; No more loading latches - Actually start Flash program
    ; memory write

    [ Required Sequence
    MOVLW   55h            ; Start of required write sequence:
    MOVWF   PMCON2         ; Write 55h
    MOVLW   0AAh           ;
    MOVWF   PMCON2         ; Write AAh
    BSF     PMCON1,WR       ; Set WR bit to begin write
    NOP     ; NOP instructions are forced as processor writes
    ; all the program memory write latches simultaneously
    NOP     ; to program memory.
    ; After NOPs, the processor
    ; stalls until the self-write process is complete
    ; after write processor continues with 3rd instruction

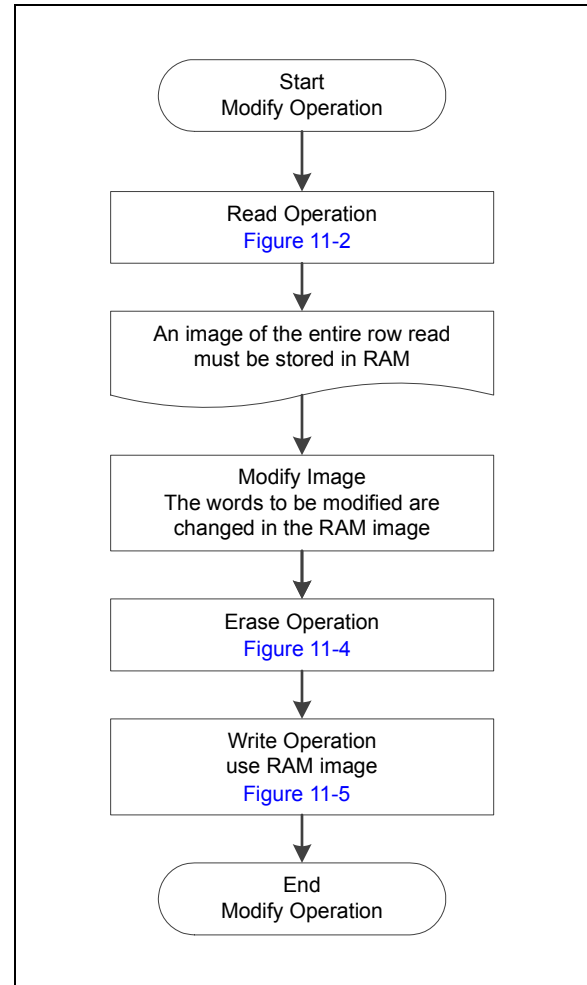
    BCF     PMCON1,WREN     ; Disable writes
    BSF     INTCON,GIE      ; Enable interrupts
```

11.3 Modifying Flash Program Memory

When modifying existing data in a program memory row, and data within that row must be preserved, it must first be read and saved in a RAM image. Program memory is modified using the following steps:

1. Load the starting address of the row to be modified.
2. Read the existing data from the row into a RAM image.
3. Modify the RAM image to contain the new data to be written into program memory.
4. Load the starting address of the row to be rewritten.
5. Erase the program memory row.
6. Load the write latches with data from the RAM image.
7. Initiate a programming operation.

FIGURE 11-7: FLASH PROGRAM MEMORY MODIFY FLOWCHART



PIC16(L)F1526/7

11.4 User ID, Device ID and Configuration Word Access

Instead of accessing program memory, the User ID's, Device ID/Revision ID and Configuration Words can be accessed when $CFG5 = 1$ in the PMCON1 register. This is the region that would be pointed to by $PC<15> = 1$, but not all addresses are accessible. Different access may exist for reads and writes. Refer to [Table 11-2](#).

When read access is initiated on an address outside the parameters listed in [Table 11-2](#), the PMDATH:PMDATL register pair is cleared, reading back '0's.

TABLE 11-2: USER ID, DEVICE ID AND CONFIGURATION WORD ACCESS (CFG5 = 1)

| Address | Function | Read Access | Write Access |
|-------------|-----------------------------|-------------|--------------|
| 8000h-8003h | User IDs | Yes | Yes |
| 8006h | Device ID/Revision ID | Yes | No |
| 8007h-8008h | Configuration Words 1 and 2 | Yes | No |

EXAMPLE 11-4: CONFIGURATION WORD AND DEVICE ID ACCESS

```
* This code block will read 1 word of program memory at the memory address:
*   PROG_ADDR_LO (must be 00h-08h) data will be returned in the variables;
*   PROG_DATA_HI, PROG_DATA_LO

BANKSEL  PMADRL           ; Select correct Bank
MOVLW    PROG_ADDR_LO    ;
MOVWF    PMADRL          ; Store LSB of address
CLRF     PMADRH          ; Clear MSB of address

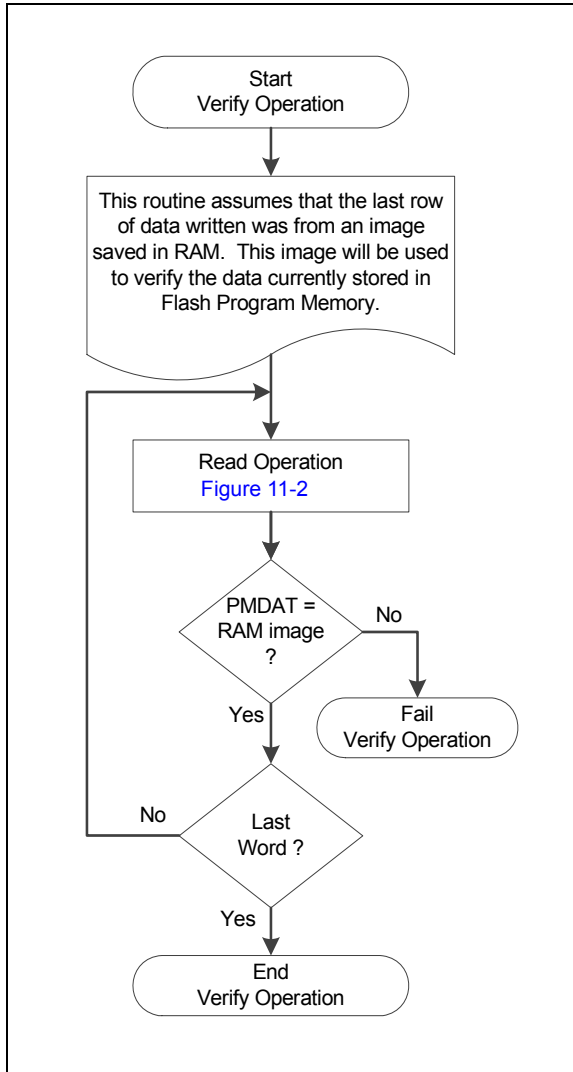
BSF      PMCON1,CFG5     ; Select Configuration Space
BCF      INTCON,GIE      ; Disable interrupts
BSF      PMCON1,RD       ; Initiate read
NOP      ; Executed (See Figure 11-2)
NOP      ; Ignored (See Figure 11-2)
BSF      INTCON,GIE      ; Restore interrupts

MOVF     PMDATL,W        ; Get LSB of word
MOVWF    PROG_DATA_LO    ; Store in user location
MOVF     PMDATH,W        ; Get MSB of word
MOVWF    PROG_DATA_HI    ; Store in user location
```

11.5 Write Verify

It is considered good programming practice to verify that program memory writes agree with the intended value. Since program memory is stored as a full page then the stored program memory contents are compared with the intended data stored in RAM after the last write is complete.

FIGURE 11-8: FLASH PROGRAM MEMORY VERIFY FLOWCHART



PIC16(L)F1526/7

11.6 Register Definitions: Flash Program Memory Control

REGISTER 11-1: PMDATL: PROGRAM MEMORY DATA LOW BYTE REGISTER

| | | | | | | | |
|------------|---------|---------|---------|---------|---------|---------|---------|
| R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u |
| PMDAT<7:0> | | | | | | | |
| bit 7 | | | | | | | bit 0 |

Legend:

| | | |
|----------------------|----------------------|---|
| R = Readable bit | W = Writable bit | U = Unimplemented bit, read as '0' |
| u = Bit is unchanged | x = Bit is unknown | -n/n = Value at POR and BOR/Value at all other Resets |
| '1' = Bit is set | '0' = Bit is cleared | |

bit 7-0 **PMDAT<7:0>**: Read/write value for Least Significant bits of program memory

REGISTER 11-2: PMDATH: PROGRAM MEMORY DATA HIGH BYTE REGISTER

| | | | | | | | |
|-------|-----|-------------|---------|---------|---------|---------|---------|
| U-0 | U-0 | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u |
| — | | PMDAT<13:8> | | | | | |
| bit 7 | | | | | | | bit 0 |

Legend:

| | | |
|----------------------|----------------------|---|
| R = Readable bit | W = Writable bit | U = Unimplemented bit, read as '0' |
| u = Bit is unchanged | x = Bit is unknown | -n/n = Value at POR and BOR/Value at all other Resets |
| '1' = Bit is set | '0' = Bit is cleared | |

bit 7-6 **Unimplemented**: Read as '0'

bit 5-0 **PMDAT<13:8>**: Read/write value for Most Significant bits of program memory

REGISTER 11-3: PMADRL: PROGRAM MEMORY ADDRESS LOW BYTE REGISTER

| | | | | | | | |
|------------|---------|---------|---------|---------|---------|---------|---------|
| R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 |
| PMADR<7:0> | | | | | | | |
| bit 7 | | | | | | | bit 0 |

Legend:

| | | |
|----------------------|----------------------|---|
| R = Readable bit | W = Writable bit | U = Unimplemented bit, read as '0' |
| u = Bit is unchanged | x = Bit is unknown | -n/n = Value at POR and BOR/Value at all other Resets |
| '1' = Bit is set | '0' = Bit is cleared | |

bit 7-0 **PMADR<7:0>**: Specifies the Least Significant bits for program memory address

REGISTER 11-4: PMADRH: PROGRAM MEMORY ADDRESS HIGH BYTE REGISTER

| | | | | | | | |
|-------|---------|-------------|---------|---------|---------|---------|---------|
| U-1 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 |
| —(1) | | PMADR<14:8> | | | | | |
| bit 7 | | | | | | | bit 0 |

Legend:

| | | |
|----------------------|----------------------|---|
| R = Readable bit | W = Writable bit | U = Unimplemented bit, read as '0' |
| u = Bit is unchanged | x = Bit is unknown | -n/n = Value at POR and BOR/Value at all other Resets |
| '1' = Bit is set | '0' = Bit is cleared | |

bit 7 **Unimplemented**: Read as '1'

bit 6-0 **PMADR<14:8>**: Specifies the Most Significant bits for program memory address

Note 1: Unimplemented, read as '1'.

REGISTER 11-5: PMCON1: PROGRAM MEMORY CONTROL 1 REGISTER

| U-1 | R/W-0/0 | R/W-0/0 | R/W/HC-0/0 | R/W/HC-x/q ⁽²⁾ | R/W-0/0 | R/S/HC-0/0 | R/S/HC-0/0 |
|------------------|---------|---------|------------|---------------------------|---------|------------|------------|
| — ⁽¹⁾ | CFGS | LWLO | FREE | WRERR | WREN | WR | RD |
| bit 7 | | | | | | | bit 0 |

Legend:

| | | |
|-------------------------|----------------------|---|
| R = Readable bit | W = Writable bit | U = Unimplemented bit, read as '0' |
| S = Bit can only be set | x = Bit is unknown | -n/n = Value at POR and BOR/Value at all other Resets |
| '1' = Bit is set | '0' = Bit is cleared | HC = Bit is cleared by hardware |

- bit 7 **Unimplemented:** Read as '1'
- bit 6 **CFGS:** Configuration Select bit
 1 = Access Configuration, User ID and Device ID Registers
 0 = Access Flash program memory
- bit 5 **LWLO:** Load Write Latches Only bit⁽³⁾
 1 = Only the addressed program memory write latch is loaded/updated on the next WR command
 0 = The addressed program memory write latch is loaded/updated and a write of all program memory write latches will be initiated on the next WR command
- bit 4 **FREE:** Program Flash Erase Enable bit
 1 = Performs an erase operation on the next WR command (hardware cleared upon completion)
 0 = Performs a write operation on the next WR command
- bit 3 **WRERR:** Program/Erase Error Flag bit
 1 = Condition indicates an improper program or erase sequence attempt or termination (bit is set automatically on any set attempt (write '1') of the WR bit).
 0 = The program or erase operation completed normally.
- bit 2 **WREN:** Program/Erase Enable bit
 1 = Allows program/erase cycles
 0 = Inhibits programming/erasing of program Flash
- bit 1 **WR:** Write Control bit
 1 = Initiates a program Flash program/erase operation.
 The operation is self-timed and the bit is cleared by hardware once operation is complete.
 The WR bit can only be set (not cleared) in software.
 0 = Program/erase operation to the Flash is complete and inactive.
- bit 0 **RD:** Read Control bit
 1 = Initiates a program Flash read. Read takes one cycle. RD is cleared in hardware. The RD bit can only be set (not cleared) in software.
 0 = Does not initiate a program Flash read.

- Note 1:** Unimplemented bit, read as '1'.
- Note 2:** The WRERR bit is automatically set by hardware when a program memory write or erase operation is started (WR = 1).
- Note 3:** The LWLO bit is ignored during a program memory erase operation (FREE = 1).

PIC16(L)F1526/7

REGISTER 11-6: PMCON2: PROGRAM MEMORY CONTROL 2 REGISTER

| | | | | | | | |
|-----------------------------------|-------|-------|-------|-------|-------|-------|-------|
| W-0/0 | W-0/0 | W-0/0 | W-0/0 | W-0/0 | W-0/0 | W-0/0 | W-0/0 |
| Program Memory Control Register 2 | | | | | | | |
| bit 7 | | | | | | | bit 0 |

Legend:

R = Readable bit W = Writable bit U = Unimplemented bit, read as '0'
 S = Bit can only be set x = Bit is unknown -n/n = Value at POR and BOR/Value at all other Resets
 '1' = Bit is set '0' = Bit is cleared

bit 7-0 **Flash Memory Unlock Pattern bits**

To unlock writes, a 55h must be written first, followed by an AAh, before setting the WR bit of the PMCON1 register. The value written to this register is used to unlock the writes. There are specific timing requirements on these writes.

TABLE 11-3: SUMMARY OF REGISTERS ASSOCIATED WITH FLASH PROGRAM MEMORY

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Register on Page |
|--------|-----------------------------------|-------------|-------------|-------|-------|--------|-------|-------|------------------|
| PMCON1 | — ⁽¹⁾ | CFGFS | LWLO | FREE | WRERR | WREN | WR | RD | 109 |
| PMCON2 | Program Memory Control Register 2 | | | | | | | | 110 |
| PMADRL | PMADRL<7:0> | | | | | | | | 108 |
| PMADRH | — ⁽¹⁾ | PMADRH<6:0> | | | | | | | 108 |
| PMDATL | PMDATL<7:0> | | | | | | | | 108 |
| PMDATH | — | — | PMDATH<5:0> | | | | | | 108 |
| INTCON | GIE | PEIE | TMR0IE | INTE | IOCIE | TMR0IF | INTF | IOCIF | 76 |

Legend: — = unimplemented location, read as '0'. Shaded cells are not used by Flash program memory.

Note 1: Unimplemented, read as '1'.

TABLE 11-4: SUMMARY OF CONFIGURATION WORD WITH FLASH PROGRAM MEMORY

| Name | Bits | Bit -/7 | Bit -/6 | Bit 13/5 | Bit 12/4 | Bit 11/3 | Bit 10/2 | Bit 9/1 | Bit 8/0 | Register on Page |
|---------|------|---------|---------|----------|-----------------------|----------|------------|----------|---------|------------------|
| CONFIG1 | 13:8 | — | — | FCMEN | IESO | CLKOUTEN | BOREN<1:0> | | — | 43 |
| | 7:0 | CP | MCLRE | PWRTE | WDTE<1:0> | | FOSC<2:0> | | | |
| CONFIG2 | 13:8 | — | | LVP | DEBUG | LPBOR | BORV | STVREN | — | 45 |
| | 7:0 | — | | — | VCAPEN ⁽¹⁾ | — | — | WRT<1:0> | | |

Legend: — = unimplemented location, read as '0'. Shaded cells are not used by Flash program memory.

12.0 I/O PORTS

In general, when a peripheral is enabled on a port pin, that pin cannot be used as a general purpose output. However, the pin can still be read.

Each port has three standard registers for its operation. These registers are:

- TRISx registers (data direction)
- PORTx registers (reads the levels on the pins of the device)
- LATx registers (output latch)

Some ports may have one or more of the following additional registers. These registers are:

- ANSELx (analog select)
- WPUx (weak pull-up)

TABLE 12-1: PORT AVAILABILITY PER DEVICE

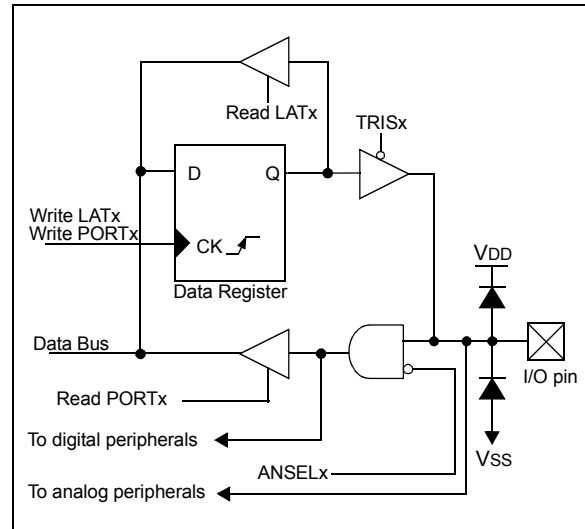
| Device | PORTA | PORTB | PORTC | PORTD | PORTE | PORTF | PORTG |
|---------------|-------|-------|-------|-------|-------|-------|-------|
| PIC16(L)F1526 | • | • | • | • | • | • | • |
| PIC16(L)F1527 | • | • | • | • | • | • | • |

The Data Latch (LATA register) is useful for read-modify-write operations on the value that the I/O pins are driving.

A write operation to the LATA register has the same effect as a write to the corresponding PORTA register. A read of the LATA register reads of the values held in the I/O PORT latches, while a read of the PORTA register reads the actual I/O pin value.

Ports that support analog inputs have an associated ANSELx register. When an ANSEL bit is set, the digital input buffer associated with that bit is disabled. Disabling the input buffer prevents analog signal levels on the pin between a logic high and low from causing excessive current in the logic input circuitry. A simplified model of a generic I/O port, without the interfaces to other peripherals, is shown in [Figure 12-1](#).

FIGURE 12-1: GENERIC I/O PORT OPERATION



PIC16(L)F1526/7

12.1 Alternate Pin Function

The Alternate Pin Function Control (APFCON) registers are used to steer specific peripheral input and output functions between different pins. The APFCON registers are shown in [Register 12-1](#). For this device family, the following functions can be moved between different pins.

- Timer3
- CCP2

These bits have no effect on the values of any TRIS register. PORT and TRIS overrides will be routed to the correct pin. The unselected pin will be unaffected.

12.2 Register Definitions: Alternate Pin Function Control

REGISTER 12-1: APFCON: ALTERNATE PIN FUNCTION CONTROL REGISTER

| | | | | | | | |
|-------|-----|-----|-----|-----|-----|----------|---------|
| U-0 | U-0 | U-0 | U-0 | U-0 | U-0 | R/W-0/0 | R/W-0/0 |
| — | — | — | — | — | — | T3CKISEL | CCP2SEL |
| bit 7 | | | | | | | bit 0 |

Legend:

R = Readable bit

W = Writable bit

U = Unimplemented bit, read as '0'

u = Bit is unchanged

x = Bit is unknown

-n/n = Value at POR and BOR/Value at all other Resets

'1' = Bit is set

'0' = Bit is cleared

bit 7-2 **Unimplemented:** Read as '0'

bit 1 **T3CKISEL:** Timer3 Input Selection bit

1 = T3CKI function is on RB4

0 = T3CKI function is on RB5

bit 0 **CCP2SEL:** Pin Selection bit

1 = CCP2 function is on RE7

0 = CCP2 function is on RC1

12.3 PORTA Registers

12.3.1 DATA REGISTER

PORTA is an 8-bit wide, bidirectional port. The corresponding data direction register is TRISA (Register 12-3). Setting a TRISA bit (= 1) will make the corresponding PORTA pin an input (i.e., disable the output driver). Clearing a TRISA bit (= 0) will make the corresponding PORTA pin an output (i.e., enables output driver and puts the contents of the output latch on the selected pin). Example 12-1 shows how to initialize an I/O port.

Reading the PORTA register (Register 12-2) reads the status of the pins, whereas writing to it will write to the PORT latch. All write operations are read-modify-write operations. Therefore, a write to a port implies that the port pins are read, this value is modified and then written to the PORT data latch (LATA).

12.3.2 DIRECTION CONTROL

The TRISA register (Register 12-3) controls the PORTA pin output drivers, even when they are being used as analog inputs. The user should ensure the bits in the TRISA register are maintained set when using them as analog inputs. I/O pins configured as analog input always read '0'.

12.3.3 ANALOG CONTROL

The ANSELA register (Register 12-5) is used to configure the Input mode of an I/O pin to analog. Setting the appropriate ANSELA bit high will cause all digital reads on the pin to be read as '0' and allow analog functions on the pin to operate correctly.

The state of the ANSELA bits has no effect on digital output functions. A pin with TRIS clear and ANSEL set will still operate as a digital output, but the Input mode will be analog. This can cause unexpected behavior when executing read-modify-write instructions on the affected port.

Note: The ANSELA bits default to the Analog mode after Reset. To use any pins as digital general purpose or peripheral inputs, the corresponding ANSEL bits must be initialized to '0' by user software.

EXAMPLE 12-1: INITIALIZING PORTA

```

; This code example illustrates
; initializing the PORTA register. The
; other ports are initialized in the same
; manner.

BANKSEL PORTA      ;
CLRF PORTA         ;Init PORTA
BANKSEL LATA       ;Data Latch
CLRF LATA          ;
BANKSEL ANSELA     ;
CLRF ANSELA        ;digital I/O
BANKSEL TRISA      ;
MOVLW B'00111000' ;Set RA<5:3> as inputs
MOVWF TRISA        ;and set RA<2:0> as
                  ;outputs
    
```

12.3.4 PORTA FUNCTIONS AND OUTPUT PRIORITIES

Each PORTA pin is multiplexed with other functions. The pins, their combined functions and their output priorities are shown in Table 12-2.

When multiple outputs are enabled, the actual pin control goes to the peripheral with the highest priority.

Analog input functions, such as ADC, are not shown in the priority lists. These inputs are active when the I/O pin is set for Analog mode using the ANSELx registers. Digital output functions may control the pin when it is in Analog mode with the priority shown in the priority list

TABLE 12-2: PORTA OUTPUT PRIORITY

| Pin Name | Function Priority ⁽¹⁾ |
|----------|----------------------------------|
| RA0 | RA0 |
| RA1 | RA1 |
| RA2 | RA2 |
| RA3 | RA3 |
| RA4 | RA4 |
| RA5 | RA5 |
| RA6 | CLKOUT OSC2 RA6 |
| RA7 | RA7 |

Note 1: Priority listed from highest to lowest.

PIC16(L)F1526/7

12.4 Register Definitions: PORTA

REGISTER 12-2: PORTA: PORTA REGISTER

| R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u |
|---------|---------|---------|---------|---------|---------|---------|---------|
| RA7 | RA6 | RA5 | RA4 | RA3 | RA2 | RA1 | RA0 |
| bit 7 | | | | | | | bit 0 |

Legend:

R = Readable bit W = Writable bit U = Unimplemented bit, read as '0'
u = Bit is unchanged x = Bit is unknown -n/n = Value at POR and BOR/Value at all other Resets
'1' = Bit is set '0' = Bit is cleared

bit 7-0 **RA<7:0>**: PORTA I/O Value bits⁽¹⁾
1 = Port pin is $\geq V_{IH}$
0 = Port pin is $\leq V_{IL}$

Note 1: Writes to PORTA are actually written to corresponding LATA register. Reads from PORTA register is return of actual I/O pin values.

REGISTER 12-3: TRISA: PORTA TRI-STATE REGISTER

| R/W-1/1 | R/W-1/1 | R/W-1/1 | R/W-1/1 | R/W-1/1 | R/W-1/1 | R/W-1/1 | R/W-1/1 |
|---------|---------|---------|---------|---------|---------|---------|---------|
| TRISA7 | TRISA6 | TRISA5 | TRISA4 | TRISA3 | TRISA2 | TRISA1 | TRISA0 |
| bit 7 | | | | | | | bit 0 |

Legend:

R = Readable bit W = Writable bit U = Unimplemented bit, read as '0'
u = Bit is unchanged x = Bit is unknown -n/n = Value at POR and BOR/Value at all other Resets
'1' = Bit is set '0' = Bit is cleared

bit 7-0 **TRISA<7:0>**: PORTA Tri-State Control bits
1 = PORTA pin configured as an input (tri-stated)
0 = PORTA pin configured as an output

REGISTER 12-4: LATA: PORTA DATA LATCH REGISTER

| R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u |
|---------|---------|---------|---------|---------|---------|---------|---------|
| LATA7 | LATA6 | LATA5 | LATA4 | LATA3 | LATA2 | LATA1 | LATA0 |
| bit 7 | | | | | | | bit 0 |

Legend:

R = Readable bit W = Writable bit U = Unimplemented bit, read as '0'
u = Bit is unchanged x = Bit is unknown -n/n = Value at POR and BOR/Value at all other Resets
'1' = Bit is set '0' = Bit is cleared

bit 7-0 **LATA<7:0>**: PORTA Output Latch Value bits⁽¹⁾

Note 1: Writes to PORTA are actually written to corresponding LATA register. Reads from PORTA register is return of actual I/O pin values.

REGISTER 12-5: ANSELA: PORTA ANALOG SELECT REGISTER

| | | | | | | | |
|-------|-----|---------|-----|---------|---------|---------|---------|
| U-0 | U-0 | R/W-1/1 | U-0 | R/W-1/1 | R/W-1/1 | R/W-1/1 | R/W-1/1 |
| — | — | ANSA5 | — | ANSA3 | ANSA2 | ANSA1 | ANSA0 |
| bit 7 | | | | | | | bit 0 |

Legend:

| | | |
|----------------------|----------------------|---|
| R = Readable bit | W = Writable bit | U = Unimplemented bit, read as '0' |
| u = Bit is unchanged | x = Bit is unknown | -n/n = Value at POR and BOR/Value at all other Resets |
| '1' = Bit is set | '0' = Bit is cleared | |

bit 7-6 **Unimplemented:** Read as '0'

bit 5 **ANSA5:** Analog Select between Analog or Digital Function on pins RA5, respectively
 1 = Analog input. Pin is assigned as analog input⁽¹⁾. Digital input buffer disabled.
 0 = Digital I/O. Pin is assigned to port or digital special function.

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

bit 3-0 **ANSA<3:0>:** Analog Select between Analog or Digital Function on pins RA<3:0>, respectively
 1 = Analog input. Pin is assigned as analog input⁽¹⁾. Digital input buffer disabled.
 0 = Digital I/O. Pin is assigned to port or digital special function.

Note 1: When setting a pin to an analog input, the corresponding TRIS bit must be set to Input mode in order to allow external control of the voltage on the pin.

TABLE 12-3: SUMMARY OF REGISTERS ASSOCIATED WITH PORTA

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Register on Page |
|------------|--------|--------|--------|--------|--------|---------|----------|---------|------------------|
| ANSELA | — | — | ANSA5 | — | ANSA3 | ANSA2 | ANSA1 | ANSA0 | 115 |
| APFCON | — | — | — | — | — | — | T3CKISEL | CCP2SEL | 112 |
| LATA | LATA7 | LATA6 | LATA5 | LATA4 | LATA3 | LATA2 | LATA1 | LATA0 | 114 |
| OPTION_REG | WPUEN | INTEDG | TMR0CS | TMR0SE | PSA | PS<2:0> | | | 158 |
| PORTA | RA7 | RA6 | RA5 | RA4 | RA3 | RA2 | RA1 | RA0 | 114 |
| TRISA | TRISA7 | TRISA6 | TRISA5 | TRISA4 | TRISA3 | TRISA2 | TRISA1 | TRISA0 | 114 |

Legend: x = unknown, u = unchanged, — = unimplemented locations read as '0'. Shaded cells are not used by PORTA.

TABLE 12-4: SUMMARY OF CONFIGURATION WORD WITH PORTA

| Name | Bits | Bit -7 | Bit -6 | Bit 13/5 | Bit 12/4 | Bit 11/3 | Bit 10/2 | Bit 9/1 | Bit 8/0 | Register on Page |
|---------|------|--------|--------|----------|-----------|----------|------------|---------|---------|------------------|
| CONFIG1 | 13:8 | — | | FCMEN | IESO | CLKOUTEN | BOREN<1:0> | | — | 45 |
| | 7:0 | CP | MCLRE | PWRTE | WDTE<1:0> | | FOSC<2:0> | | | |

Legend: — = unimplemented location, read as '0'. Shaded cells are not used by PORTA.

PIC16(L)F1526/7

12.5 PORTB Registers

12.5.1 DATA REGISTER

PORTB is an 8-bit wide, bidirectional port. The corresponding data direction register is TRISB (Register 12-7). Setting a TRISB bit (= 1) will make the corresponding PORTB pin an input (i.e., put the corresponding output driver in a High-Impedance mode). Clearing a TRISB bit (= 0) will make the corresponding PORTB pin an output (i.e., enable the output driver and put the contents of the output latch on the selected pin). Example 12-1 shows how to initialize an I/O port.

Reading the PORTB register (Register 12-6) reads the status of the pins, whereas writing to it will write to the PORT latch. All write operations are read-modify-write operations. Therefore, a write to a port implies that the port pins are read, this value is modified and then written to the PORT data latch (LATB).

12.5.2 DIRECTION CONTROL

The TRISB register (Register 12-7) controls the PORTB pin output drivers, even when they are being used as analog inputs. The user should ensure the bits in the TRISB register are maintained set when using them as analog inputs. I/O pins configured as analog input always read '0'.

12.5.3 ANALOG CONTROL

The ANSELB register (Register 12-9) is used to configure the Input mode of an I/O pin to analog. Setting the appropriate ANSELB bit high will cause all digital reads on the pin to be read as '0' and allow analog functions on the pin to operate correctly.

The state of the ANSELB bits has no effect on digital output functions. A pin with TRIS clear and ANSELB set will still operate as a digital output, but the Input mode will be analog. This can cause unexpected behavior when executing read-modify-write instructions on the affected port.

Note: The ANSELB bits default to the Analog mode after Reset. To use any pins as digital general purpose or peripheral inputs, the corresponding ANSEL bits must be initialized to '0' by user software.

12.5.4 PORTB FUNCTIONS AND OUTPUT PRIORITIES

Each PORTB pin is multiplexed with other functions. The pins, their combined functions and their output priorities are shown in Table 12-5.

When multiple outputs are enabled, the actual pin control goes to the peripheral with the highest priority.

Analog input and some digital input functions are not included in the list below. These input functions can remain active when the pin is configured as an output. Certain digital input functions override other port functions and are included in Table 12-5.

TABLE 12-5: PORTB OUTPUT PRIORITY

| Pin Name | Function Priority ⁽¹⁾ |
|----------|----------------------------------|
| RB0 | RB0 |
| RB1 | RB1 |
| RB2 | RB2 |
| RB3 | CCP2 RB3 |
| RB4 | RB4 |
| RB5 | RB5 |
| RB6 | ICDCLK RB6 |
| RB7 | ICDDAT RB7 |

Note 1: Priority listed from highest to lowest.

12.6 Register Definitions: PORTB

REGISTER 12-6: PORTB: PORTB REGISTER

| | | | | | | | |
|---------|---------|---------|---------|---------|---------|---------|---------|
| R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u |
| RB7 | RB6 | RB5 | RB4 | RB3 | RB2 | RB1 | RB0 |
| bit 7 | | | | | | | bit 0 |

Legend:

R = Readable bit W = Writable bit U = Unimplemented bit, read as '0'
u = Bit is unchanged x = Bit is unknown -n/n = Value at POR and BOR/Value at all other Resets
'1' = Bit is set '0' = Bit is cleared

bit 7-0 **RB<7:0>**: PORTB I/O Pin bits⁽¹⁾
1 = Port pin is $\geq V_{IH}$
0 = Port pin is $\leq V_{IL}$

Note 1: Writes to PORTB are actually written to corresponding LATB register. Reads from PORTB register is return of actual I/O pin values.

REGISTER 12-7: TRISB: PORTB TRI-STATE REGISTER

| | | | | | | | |
|---------|---------|---------|---------|---------|---------|---------|---------|
| R/W-1/1 | R/W-1/1 | R/W-1/1 | R/W-1/1 | R/W-1/1 | R/W-1/1 | R/W-1/1 | R/W-1/1 |
| TRISB7 | TRISB6 | TRISB5 | TRISB4 | TRISB3 | TRISB2 | TRISB1 | TRISB0 |
| bit 7 | | | | | | | bit 0 |

Legend:

R = Readable bit W = Writable bit U = Unimplemented bit, read as '0'
u = Bit is unchanged x = Bit is unknown -n/n = Value at POR and BOR/Value at all other Resets
'1' = Bit is set '0' = Bit is cleared

bit 7-0 **TRISB<7:0>**: PORTB Tri-State Control bits
1 = PORTB pin configured as an input (tri-stated)
0 = PORTB pin configured as an output

REGISTER 12-8: LATB: PORTB DATA LATCH REGISTER

| | | | | | | | |
|---------|---------|---------|---------|---------|---------|---------|---------|
| R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u |
| LATB7 | LATB6 | LATB5 | LATB4 | LATB3 | LATB2 | LATB1 | LATB0 |
| bit 7 | | | | | | | bit 0 |

Legend:

R = Readable bit W = Writable bit U = Unimplemented bit, read as '0'
u = Bit is unchanged x = Bit is unknown -n/n = Value at POR and BOR/Value at all other Resets
'1' = Bit is set '0' = Bit is cleared

bit 7-0 **LATB<7:0>**: PORTB Output Latch Value bits⁽¹⁾

Note 1: Writes to PORTB are actually written to corresponding LATB register. Reads from PORTB register is return of actual I/O pin values.

PIC16(L)F1526/7

REGISTER 12-9: ANSELB: PORTB ANALOG SELECT REGISTER

| | | | | | | | |
|-------|-----|---------|---------|---------|---------|---------|---------|
| U-0 | U-0 | R/W-1/1 | R/W-1/1 | R/W-1/1 | R/W-1/1 | R/W-1/1 | R/W-1/1 |
| — | — | ANSB5 | ANSB4 | ANSB3 | ANSB2 | ANSB1 | ANSB0 |
| bit 7 | | | | | | | bit 0 |

Legend:

R = Readable bit W = Writable bit U = Unimplemented bit, read as '0'
u = Bit is unchanged x = Bit is unknown -n/n = Value at POR and BOR/Value at all other Resets
'1' = Bit is set '0' = Bit is cleared

bit 7-6 **Unimplemented:** Read as '0'
bit 5-0 **ANSB<5:0>:** Analog Select between Analog or Digital Function on pins RB<5:0>, respectively
1 = Analog input. Pin is assigned as analog input⁽¹⁾. Digital input buffer disabled.
0 = Digital I/O. Pin is assigned to port or digital special function.

Note 1: When setting a pin to an analog input, the corresponding TRIS bit must be set to Input mode in order to allow external control of the voltage on the pin.

REGISTER 12-10: WPUB: WEAK PULL-UP PORTB REGISTER

| | | | | | | | |
|---------|---------|---------|---------|---------|---------|---------|---------|
| R/W-1/1 | R/W-1/1 | R/W-1/1 | R/W-1/1 | R/W-1/1 | R/W-1/1 | R/W-1/1 | R/W-1/1 |
| WPUB7 | WPUB6 | WPUB5 | WPUB4 | WPUB3 | WPUB2 | WPUB1 | WPUB0 |
| bit 7 | | | | | | | bit 0 |

Legend:

R = Readable bit W = Writable bit U = Unimplemented bit, read as '0'
u = Bit is unchanged x = Bit is unknown -n/n = Value at POR and BOR/Value at all other Resets
'1' = Bit is set '0' = Bit is cleared

bit 7-0 **WPUB<7:0>:** Weak Pull-up Register bits
1 = Pull-up enabled
0 = Pull-up disabled

Note 1: Global $\overline{\text{WPUEN}}$ bit of the OPTION_REG register must be cleared for individual pull-ups to be enabled.
Note 2: The weak pull-up device is automatically disabled if the pin is in configured as an output.

TABLE 12-6: SUMMARY OF REGISTERS ASSOCIATED WITH PORTB

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Register on Page |
|--------|--------|--------|--------|--------|--------|--------|----------|---------|------------------|
| APFCON | — | — | — | — | — | — | T3CKISEL | CCP2SEL | 118 |
| ANSELB | — | — | ANSB5 | ANSB4 | ANSB3 | ANSB2 | ANSB1 | ANSB0 | 118 |
| LATB | LATB7 | LATB6 | LATB5 | LATB4 | LATB3 | LATB2 | LATB1 | LATB0 | 117 |
| PORTB | RB7 | RB6 | RB5 | RB4 | RB3 | RB2 | RB1 | RB0 | 117 |
| TRISB | TRISB7 | TRISB6 | TRISB5 | TRISB4 | TRISB3 | TRISB2 | TRISB1 | TRISB0 | 117 |
| WPUB | WPUB7 | WPUB6 | WPUB5 | WPUB4 | WPUB3 | WPUB2 | WPUB1 | WPUB0 | 118 |

Legend: x = unknown, u = unchanged, - = unimplemented locations read as '0'. Shaded cells are not used by PORTB.

12.7 PORTC Registers

12.7.1 DATA REGISTER

PORTC is an 8-bit wide, bidirectional port. The corresponding data direction register is TRISC (Register 12-12). Setting a TRISC bit (= 1) will make the corresponding PORTC pin an input (i.e., put the corresponding output driver in a High-Impedance mode). Clearing a TRISC bit (= 0) will make the corresponding PORTC pin an output (i.e., enable the output driver and put the contents of the output latch on the selected pin). Example 12-1 shows how to initialize an I/O port.

Reading the PORTC register (Register 12-11) reads the status of the pins, whereas writing to it will write to the PORT latch. All write operations are read-modify-write operations. Therefore, a write to a port implies that the port pins are read, this value is modified and then written to the PORT data latch (LATC).

12.7.2 DIRECTION CONTROL

The TRISC register (Register 12-12) controls the PORTC pin output drivers, even when they are being used as analog inputs. The user should ensure the bits in the TRISC register are maintained set when using them as analog inputs. I/O pins configured as analog input always read '0'.

12.7.3 PORTC FUNCTIONS AND OUTPUT PRIORITIES

Each PORTC pin is multiplexed with other functions. The pins, their combined functions and their output priorities are shown in Table 12-7.

When multiple outputs are enabled, the actual pin control goes to the peripheral with the highest priority. Analog input and some digital input functions are not included in the list below. These input functions can remain active when the pin is configured as an output. Certain digital input functions override other port functions and are included in the priority list.

TABLE 12-7: PORTC OUTPUT PRIORITY

| Pin Name | Function Priority ⁽¹⁾ |
|----------|------------------------------------|
| RC0 | SOSCO RC0 |
| RC1 | SOSCI CCP2 RC1 |
| RC2 | CCP1 RC2 |
| RC3 | SCL1 SCK1 RC3 ⁽²⁾ |
| RC4 | SDA1 RC4 ⁽²⁾ |
| RC5 | SDO1 RC5 |
| RC6 | CK1 TX1 RC6 |
| RC7 | DT1 RC7 |

Note 1: Priority listed from highest to lowest.

2: RC3 and RC4 read the I²C ST input when I²C mode is enabled.

PIC16(L)F1526/7

12.8 Register Definitions: PORTC

REGISTER 12-11: PORTC: PORTC REGISTER

| R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u |
|---------|---------|---------|---------|---------|---------|---------|---------|
| RC7 | RC6 | RC5 | RC4 | RC3 | RC2 | RC1 | RC0 |
| bit 7 | | | | | | | bit 0 |

Legend:

R = Readable bit W = Writable bit U = Unimplemented bit, read as '0'
u = Bit is unchanged x = Bit is unknown -n/n = Value at POR and BOR/Value at all other Resets
'1' = Bit is set '0' = Bit is cleared

bit 7-0 **RC<7:0>**: PORTC General Purpose I/O Pin bits⁽¹⁾
1 = Port pin is $\geq V_{IH}$
0 = Port pin is $\leq V_{IL}$

Note 1: Writes to PORTC are actually written to corresponding LATC register. Reads from PORTC register is return of actual I/O pin values.

REGISTER 12-12: TRISC: PORTC TRI-STATE REGISTER

| R/W-1/1 | R/W-1/1 | R/W-1/1 | R/W-1/1 | R/W-1/1 | R/W-1/1 | R/W-1/1 | R/W-1/1 |
|---------|---------|---------|---------|---------|---------|---------|---------|
| TRISC7 | TRISC6 | TRISC5 | TRISC4 | TRISC3 | TRISC2 | TRISC1 | TRISC0 |
| bit 7 | | | | | | | bit 0 |

Legend:

R = Readable bit W = Writable bit U = Unimplemented bit, read as '0'
u = Bit is unchanged x = Bit is unknown -n/n = Value at POR and BOR/Value at all other Resets
'1' = Bit is set '0' = Bit is cleared

bit 7-0 **TRISC<7:0>**: PORTC Tri-State Control bits
1 = PORTC pin configured as an input (tri-stated)
0 = PORTC pin configured as an output

REGISTER 12-13: LATC: PORTC DATA LATCH REGISTER

| R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u |
|---------|---------|---------|---------|---------|---------|---------|---------|
| LATC7 | LATC6 | LATC5 | LATC4 | LATC3 | LATC2 | LATC1 | LATC0 |
| bit 7 | | | | | | | bit 0 |

Legend:

R = Readable bit W = Writable bit U = Unimplemented bit, read as '0'
u = Bit is unchanged x = Bit is unknown -n/n = Value at POR and BOR/Value at all other Resets
'1' = Bit is set '0' = Bit is cleared

bit 7-0 **LATC<7:0>**: PORTC Output Latch Value bits⁽¹⁾

Note 1: Writes to PORTC are actually written to corresponding LATC register. Reads from PORTC register is return of actual I/O pin values.

TABLE 12-8: SUMMARY OF REGISTERS ASSOCIATED WITH PORTC

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Register on Page |
|--------|--------|--------|--------|--------|--------|--------|----------|---------|---------------------|
| APFCON | — | — | — | — | — | — | T3CKISEL | CCP2SEL | 118 |
| LATC | LATC7 | LATC6 | LATC5 | LATC4 | LATC3 | LATC2 | LATC1 | LATC0 | 117 |
| PORTC | RC7 | RC6 | RC5 | RC4 | RC3 | RC2 | RC1 | RC0 | 117 |
| TRISC | TRISC7 | TRISC6 | TRISC5 | TRISC4 | TRISC3 | TRISC2 | TRISC1 | TRISC0 | 120 |

Legend: x = unknown, u = unchanged, - = unimplemented locations read as '0'. Shaded cells are not used by PORTC.

PIC16(L)F1526/7

12.9 PORTD Registers

12.9.1 DATA REGISTER

PORTD is an 8-bit wide, bidirectional port. The corresponding data direction register is TRISD (Register 12-15). Setting a TRISD bit (= 1) will make the corresponding PORTD pin an input (i.e., put the corresponding output driver in a High-Impedance mode). Clearing a TRISD bit (= 0) will make the corresponding PORTD pin an output (i.e., enable the output driver and put the contents of the output latch on the selected pin). Example 12-1 shows how to initialize an I/O port.

Reading the PORTD register (Register 12-14) reads the status of the pins, whereas writing to it will write to the PORT latch. All write operations are read-modify-write operations. Therefore, a write to a port implies that the port pins are read, this value is modified and then written to the PORT data latch (LATD).

12.9.2 DIRECTION CONTROL

The TRISD register (Register 12-15) controls the PORTD pin output drivers, even when they are being used as analog inputs. The user should ensure the bits in the TRISD register are maintained set when using them as analog inputs. I/O pins configured as analog input always read '0'.

12.9.3 ANALOG CONTROL

The ANSEL register (Register 12-17) is used to configure the Input mode of an I/O pin to analog. Setting the appropriate ANSEL bit high will cause all digital reads on the pin to be read as '0' and allow analog functions on the pin to operate correctly.

The state of the ANSEL bits has no effect on digital output functions. A pin with TRIS clear and ANSEL set will still operate as a digital output, but the Input mode will be analog. This can cause unexpected behavior when executing read-modify-write instructions on the affected port.

Note: The ANSEL bits default to the Analog mode after Reset. To use any pins as digital general purpose or peripheral inputs, the corresponding ANSEL bits must be initialized to '0' by user software.

12.9.4 PORTD FUNCTIONS AND OUTPUT PRIORITIES

Each PORTD pin is multiplexed with other functions. The pins, their combined functions and their output priorities are shown in Table 12-9.

When multiple outputs are enabled, the actual pin control goes to the peripheral with the highest priority. Analog input and some digital input functions are not included in the list below. These input functions can remain active when the pin is configured as an output. Certain digital input functions override other port functions and are included in the priority list.

TABLE 12-9: PORTD OUTPUT PRIORITY

| Pin Name | Function Priority ⁽¹⁾ |
|----------|------------------------------------|
| RD0 | RD0 |
| RD1 | RD1 |
| RD2 | RD2 |
| RD3 | RD3 |
| RD4 | SDO2 RD4 |
| RD5 | SDA2 RD5 ⁽²⁾ |
| RD6 | SCL2 SCK2 RD6 ⁽²⁾ |
| RD7 | RD7 |

Note 1: Priority listed from highest to lowest.

2: RD5 and RD6 read the I²C ST input when I²C mode is enabled.

12.10 Register Definitions: PORTD

REGISTER 12-14: PORTD: PORTD REGISTER

| | | | | | | | |
|---------|---------|---------|---------|---------|---------|---------|---------|
| R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u |
| RD7 | RD6 | RD5 | RD4 | RD3 | RD2 | RD1 | RD0 |
| bit 7 | | | | | | | bit 0 |

Legend:

R = Readable bit W = Writable bit U = Unimplemented bit, read as '0'
u = Bit is unchanged x = Bit is unknown -n/n = Value at POR and BOR/Value at all other Resets
'1' = Bit is set '0' = Bit is cleared

bit 7-0 **RD<7:0>**: PORTD General Purpose I/O Pin bits⁽¹⁾
1 = Port pin is $\geq V_{IH}$
0 = Port pin is $\leq V_{IL}$

Note 1: Writes to PORTD are actually written to corresponding LATD register. Reads from PORTD register is return of actual I/O pin values.

REGISTER 12-15: TRISD: PORTD TRI-STATE REGISTER

| | | | | | | | |
|---------|---------|---------|---------|---------|---------|---------|---------|
| R/W-1/1 | R/W-1/1 | R/W-1/1 | R/W-1/1 | R/W-1/1 | R/W-1/1 | R/W-1/1 | R/W-1/1 |
| TRISD7 | TRISD6 | TRISD5 | TRISD4 | TRISD3 | TRISD2 | TRISD1 | TRISD0 |
| bit 7 | | | | | | | bit 0 |

Legend:

R = Readable bit W = Writable bit U = Unimplemented bit, read as '0'
u = Bit is unchanged x = Bit is unknown -n/n = Value at POR and BOR/Value at all other Resets
'1' = Bit is set '0' = Bit is cleared

bit 7-0 **TRISD<7:0>**: PORTD Tri-State Control bits
1 = PORTD pin configured as an input (tri-stated)
0 = PORTD pin configured as an output

REGISTER 12-16: LATD: PORTD DATA LATCH REGISTER

| | | | | | | | |
|---------|---------|---------|---------|---------|---------|---------|---------|
| R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u |
| LATD7 | LATD6 | LATD5 | LATD4 | LATD3 | LATD2 | LATD1 | LATD0 |
| bit 7 | | | | | | | bit 0 |

Legend:

R = Readable bit W = Writable bit U = Unimplemented bit, read as '0'
u = Bit is unchanged x = Bit is unknown -n/n = Value at POR and BOR/Value at all other Resets
'1' = Bit is set '0' = Bit is cleared

bit 7-0 **LATD<7:0>**: PORTD Output Latch Value bits⁽¹⁾

Note 1: Writes to PORTD are actually written to corresponding LATD register. Reads from PORTD register is return of actual I/O pin values.

PIC16(L)F1526/7

REGISTER 12-17: ANSELD: PORTD ANALOG SELECT REGISTER

| | | | | | | | |
|-------|-----|-----|-----|-------|-------|-------|-------|
| U-0 | U-0 | U-0 | U-0 | R/W-1 | R/W-1 | R/W-1 | R/W-1 |
| — | — | — | — | ANSD3 | ANSD2 | ANSD1 | ANSD0 |
| bit 7 | | | | bit 0 | | | |

Legend:

R = Readable bit W = Writable bit U = Unimplemented bit, read as '0'
u = Bit is unchanged x = Bit is unknown -n/n = Value at POR and BOR/Value at all other Resets
'1' = Bit is set '0' = Bit is cleared

bit 7-4 **Unimplemented:** Read as '0'
bit 3-0 **ANSD<3:0>:** Analog Select between Analog or Digital Function on pins RD<3:0>, respectively
1 = Analog input. Pin is assigned as analog input⁽¹⁾. Digital input buffer disabled.
0 = Digital I/O. Pin is assigned to port or digital special function.

Note 1: When setting a pin to an analog input, the corresponding TRIS bit must be set to Input mode in order to allow external control of the voltage on the pin.

REGISTER 12-18: WPUD: WEAK PULL-UP PORTD REGISTER

| | | | | | | | |
|---------|---------|---------|---------|---------|---------|---------|---------|
| R/W-1/1 | R/W-1/1 | R/W-1/1 | R/W-1/1 | R/W-1/1 | R/W-1/1 | R/W-1/1 | R/W-1/1 |
| WPUD7 | WPUD6 | WPUD5 | WPUD4 | WPUD3 | WPUD2 | WPUD1 | WPUD0 |
| bit 7 | | | | | | | bit 0 |

Legend:

R = Readable bit W = Writable bit U = Unimplemented bit, read as '0'
u = Bit is unchanged x = Bit is unknown -n/n = Value at POR and BOR/Value at all other Resets
'1' = Bit is set '0' = Bit is cleared

bit 7-0 **WPUD<7:0>:** Weak Pull-up Register bits
1 = Pull-up enabled
0 = Pull-up disabled

Note 1: Global $\overline{\text{WPUEN}}$ bit of the OPTION_REG register must be cleared for individual pull-ups to be enabled.
2: The weak pull-up device is automatically disabled if the pin is in configured as an output.

TABLE 12-10: SUMMARY OF REGISTERS ASSOCIATED WITH PORTD

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Register on Page |
|--------|--------|--------|--------|--------|--------|--------|--------|--------|------------------|
| ANSELD | — | — | — | — | ANSD3 | ANSD2 | ANSD1 | ANSD0 | 118 |
| LATD | LATD7 | LATD6 | LATD5 | LATD4 | LATD3 | LATD2 | LATD1 | LATD0 | 117 |
| PORTD | RD7 | RD6 | RD5 | RD4 | RD3 | RD2 | RD1 | RD0 | 117 |
| TRISD | TRISD7 | TRISD6 | TRISD5 | TRISD4 | TRISD3 | TRISD2 | TRISD1 | TRISD0 | 123 |
| WPUD | WPUD7 | WPUD6 | WPUD5 | WPUD4 | WPUD3 | WPUD2 | WPUD1 | WPUD0 | 124 |

Legend: x = unknown, u = unchanged, - = unimplemented locations read as '0'. Shaded cells are not used by PORTD.

12.11 PORTE Registers

12.11.1 DATA REGISTER

PORTE is an 8-bit wide, bidirectional port. The corresponding data direction register is TRISE (Register 12-20). Setting a TRISE bit (= 1) will make the corresponding PORTE pin an input (i.e., put the corresponding output driver in a High-Impedance mode). Clearing a TRISE bit (= 0) will make the corresponding PORTE pin an output (i.e., enable the output driver and put the contents of the output latch on the selected pin). Example 12-1 shows how to initialize an I/O port.

Reading the PORTE register (Register 12-19) reads the status of the pins, whereas writing to it will write to the PORT latch. All write operations are read-modify-write operations. Therefore, a write to a port implies that the port pins are read, this value is modified and then written to the PORT data latch (LATE).

12.11.2 DIRECTION CONTROL

The TRISE register (Register 12-20) controls the PORTE pin output drivers, even when they are being used as analog inputs. The user should ensure the bits in the TRISE register are maintained set when using them as analog inputs. I/O pins configured as analog input always read '0'.

12.11.3 ANALOG CONTROL

The ANSELE register (Register 12-22) is used to configure the Input mode of an I/O pin to analog. Setting the appropriate ANSELE bit high will cause all digital reads on the pin to be read as '0' and allow analog functions on the pin to operate correctly.

The state of the ANSELE bits has no effect on digital output functions. A pin with TRIS clear and ANSELE set will still operate as a digital output, but the Input mode will be analog. This can cause unexpected behavior when executing read-modify-write instructions on the affected port.

Note: The ANSELE bits default to the Analog mode after Reset. To use any pins as digital general purpose or peripheral inputs, the corresponding ANSEL bits must be initialized to '0' by user software.

12.11.4 PORTE FUNCTIONS AND OUTPUT PRIORITIES

Each PORTE pin is multiplexed with other functions. The pins, their combined functions and their output priorities are shown in Table 12-11.

When multiple outputs are enabled, the actual pin control goes to the peripheral with the highest priority. Analog input and some digital input functions are not included in the list below. These input functions can remain active when the pin is configured as an output. Certain digital input functions override other port functions and are included in the priority list.

TABLE 12-11: PORTE OUTPUT PRIORITY

| Pin Name | Function Priority ⁽¹⁾ |
|----------|----------------------------------|
| RE0 | RE0 |
| RE1 | RE1 |
| RE2 | CCP10 RE2 |
| RE3 | CCP9 RE3 |
| RE4 | CCP8 RE4 |
| RE5 | CCP7 RE5 |
| RE6 | CCP6 RE6 |
| RE7 | CCP2 RE7 |

Note 1: Priority listed from highest to lowest.

PIC16(L)F1526/7

12.12 Register Definitions: PORTE

REGISTER 12-19: PORTE: PORTE REGISTER

| | | | | | | | |
|---------|---------|---------|---------|---------|---------|---------|---------|
| R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u |
| RE7 | RE6 | RE5 | RE4 | RE3 | RE2 | RE1 | RE0 |
| bit 7 | | | | | | | bit 0 |

Legend:

R = Readable bit W = Writable bit U = Unimplemented bit, read as '0'
u = Bit is unchanged x = Bit is unknown -n/n = Value at POR and BOR/Value at all other Resets
'1' = Bit is set '0' = Bit is cleared

bit 7-0 **RE<7:0>**: PORTE General Purpose I/O Pin bits⁽¹⁾
1 = Port pin is $\geq V_{IH}$
0 = Port pin is $\leq V_{IL}$

Note 1: Writes to PORTE are actually written to corresponding LATE register. Reads from PORTE register is return of actual I/O pin values.

REGISTER 12-20: TRISE: PORTE TRI-STATE REGISTER

| | | | | | | | |
|---------|---------|---------|---------|---------|---------|---------|---------|
| R/W-1/1 | R/W-1/1 | R/W-1/1 | R/W-1/1 | R/W-1/1 | R/W-1/1 | R/W-1/1 | R/W-1/1 |
| TRISE7 | TRISE6 | TRISE5 | TRISE4 | TRISE3 | TRISE2 | TRISE1 | TRISE0 |
| bit 7 | | | | | | | bit 0 |

Legend:

R = Readable bit W = Writable bit U = Unimplemented bit, read as '0'
u = Bit is unchanged x = Bit is unknown -n/n = Value at POR and BOR/Value at all other Resets
'1' = Bit is set '0' = Bit is cleared

bit 7-0 **TRISE<7:0>**: PORTE Tri-State Control bits
1 = PORTE pin configured as an input (tri-stated)
0 = PORTE pin configured as an output

REGISTER 12-21: LATE: PORTE DATA LATCH REGISTER

| | | | | | | | |
|---------|---------|---------|---------|---------|---------|---------|---------|
| R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u |
| LATE7 | LATE6 | LATE5 | LATE4 | LATE3 | LATE2 | LATE1 | LATE0 |
| bit 7 | | | | | | | bit 0 |

Legend:

R = Readable bit W = Writable bit U = Unimplemented bit, read as '0'
u = Bit is unchanged x = Bit is unknown -n/n = Value at POR and BOR/Value at all other Resets
'1' = Bit is set '0' = Bit is cleared

bit 7-0 **LATE<7:0>**: PORTE Output Latch Value bits⁽¹⁾

Note 1: Writes to PORTE are actually written to corresponding LATE register. Reads from PORTE register is return of actual I/O pin values.

REGISTER 12-22: ANSELE: PORTE ANALOG SELECT REGISTER

| | | | | | | | |
|-------|-----|-----|-----|-----|-------|-------|-------|
| U-0 | U-0 | U-0 | U-0 | U-0 | R/W-1 | R/W-1 | R/W-1 |
| — | — | — | — | — | ANSE2 | ANSE1 | ANSE0 |
| bit 7 | | | | | | | bit 0 |

Legend:

| | | |
|----------------------|----------------------|---|
| R = Readable bit | W = Writable bit | U = Unimplemented bit, read as '0' |
| u = Bit is unchanged | x = Bit is unknown | -n/n = Value at POR and BOR/Value at all other Resets |
| '1' = Bit is set | '0' = Bit is cleared | |

bit 7-3 **Unimplemented:** Read as '0'

bit 2-0 **ANSE<2:0>:** Analog Select between Analog or Digital Function on pins RE<2:0>, respectively
 1 = Analog input. Pin is assigned as analog input⁽¹⁾. Digital input buffer disabled.
 0 = Digital I/O. Pin is assigned to port or digital special function.

Note 1: When setting a pin to an analog input, the corresponding TRIS bit must be set to Input mode in order to allow external control of the voltage on the pin.

REGISTER 12-23: WPUE: WEAK PULL-UP PORTE REGISTER

| | | | | | | | |
|---------|---------|---------|---------|---------|---------|---------|---------|
| R/W-1/1 | R/W-1/1 | R/W-1/1 | R/W-1/1 | R/W-1/1 | R/W-1/1 | R/W-1/1 | R/W-1/1 |
| WPUE7 | WPUE6 | WPUE5 | WPUE4 | WPUE3 | WPUE2 | WPUE1 | WPUE0 |
| bit 7 | | | | | | | bit 0 |

Legend:

| | | |
|----------------------|----------------------|---|
| R = Readable bit | W = Writable bit | U = Unimplemented bit, read as '0' |
| u = Bit is unchanged | x = Bit is unknown | -n/n = Value at POR and BOR/Value at all other Resets |
| '1' = Bit is set | '0' = Bit is cleared | |

bit 7-0 **WPUE<7:0>:** Weak Pull-up Register bits
 1 = Pull-up enabled
 0 = Pull-up disabled

Note 1: Global $\overline{\text{WPUEN}}$ bit of the OPTION_REG register must be cleared for individual pull-ups to be enabled.
Note 2: The weak pull-up device is automatically disabled if the pin is configured as an output.

TABLE 12-12: SUMMARY OF REGISTERS ASSOCIATED WITH PORTE

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Register on Page |
|--------|--------|--------|--------|--------|--------|--------|----------|---------|------------------|
| APFCON | — | — | — | — | — | — | T3CKISEL | CCP2SEL | 118 |
| ANSELE | — | — | — | — | — | ANSE2 | ANSE1 | ANSE0 | 127 |
| LATE | LATE7 | LATE6 | LATE5 | LATE4 | LATE3 | LATE2 | LATE1 | LATE0 | 126 |
| PORTE | RE7 | RE6 | RE5 | RE4 | RE3 | RE2 | RE1 | RE0 | 126 |
| TRISE | TRISE7 | TRISE6 | TRISE5 | TRISE4 | TRISE3 | TRISE2 | TRISE1 | TRISE0 | 126 |
| WPUE | WPUE7 | WPUE6 | WPUE5 | WPUE4 | WPUE3 | WPUE2 | WPUE1 | WPUE0 | 127 |

Legend: x = unknown, u = unchanged, - = unimplemented locations read as '0'. Shaded cells are not used by PORTE.

PIC16(L)F1526/7

12.13 PORTF Registers

12.13.1 DATA REGISTER

PORTF is an 8-bit wide, bidirectional port. The corresponding data direction register is TRISF (Register 12-25). Setting a TRISF bit (= 1) will make the corresponding PORTF pin an input (i.e., put the corresponding output driver in a High-Impedance mode). Clearing a TRISF bit (= 0) will make the corresponding PORTF pin an output (i.e., enable the output driver and put the contents of the output latch on the selected pin). Example 12-1 shows how to initialize an I/O port.

Reading the PORTF register (Register 12-24) reads the status of the pins, whereas writing to it will write to the PORT latch. All write operations are read-modify-write operations. Therefore, a write to a port implies that the port pins are read, this value is modified and then written to the PORT data latch (LATF).

12.13.2 DIRECTION CONTROL

The TRISF register (Register 12-25) controls the PORTF pin output drivers, even when they are being used as analog inputs. The user should ensure the bits in the TRISF register are maintained set when using them as analog inputs. I/O pins configured as analog input always read '0'.

12.13.3 ANALOG CONTROL

The ANSELF register (Register 12-27) is used to configure the Input mode of an I/O pin to analog. Setting the appropriate ANSELF bit high will cause all digital reads on the pin to be read as '0' and allow analog functions on the pin to operate correctly.

The state of the ANSELF bits has no effect on digital output functions. A pin with TRIS clear and ANSELF set will still operate as a digital output, but the Input mode will be analog. This can cause unexpected behavior when executing read-modify-write instructions on the affected port.

Note: The ANSELF bits default to the Analog mode after Reset. To use any pins as digital general purpose or peripheral inputs, the corresponding ANSEL bits must be initialized to '0' by user software.

12.13.4 PORTE FUNCTIONS AND OUTPUT PRIORITIES

Each PORTF pin is multiplexed with other functions. The pins, their combined functions and their output priorities are shown in Table 12-13.

When multiple outputs are enabled, the actual pin control goes to the peripheral with the highest priority.

Analog input and some digital input functions are not included in the list below. These input functions can remain active when the pin is configured as an output. Certain digital input functions override other port functions and are included in the priority list.

TABLE 12-13: PORTF OUTPUT PRIORITY

| Pin Name | Function Priority ⁽¹⁾ |
|----------|--|
| RF0 | V _{CAP} ⁽²⁾ RF0 |
| RF1 | RF1 |
| RF2 | RF2 |
| RF3 | RF3 |
| RF4 | RF4 |
| RF5 | RF5 |
| RF6 | RF6 |
| RF7 | RF7 |

Note 1: Priority listed from highest to lowest.
2: PIC16F1526/7 only

12.14 Register Definitions: PORTF

REGISTER 12-24: PORTF: PORTF REGISTER

| | | | | | | | |
|---------|---------|---------|---------|---------|---------|---------|---------|
| R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u |
| RF7 | RF6 | RF5 | RF4 | RF3 | RF2 | RF1 | RF0 |
| bit 7 | | | | | | | bit 0 |

Legend:

R = Readable bit W = Writable bit U = Unimplemented bit, read as '0'
u = Bit is unchanged x = Bit is unknown -n/n = Value at POR and BOR/Value at all other Resets
'1' = Bit is set '0' = Bit is cleared

bit 7-0 **RF<7:0>**: PORTF General Purpose I/O Pin bits⁽¹⁾
1 = Port pin is $\geq V_{IH}$
0 = Port pin is $\leq V_{IL}$

Note 1: Writes to PORTF are actually written to corresponding LATF register. Reads from PORTF register is return of actual I/O pin values.

REGISTER 12-25: TRISF: PORTF TRI-STATE REGISTER

| | | | | | | | |
|---------|---------|---------|---------|---------|---------|---------|---------|
| R/W-1/1 | R/W-1/1 | R/W-1/1 | R/W-1/1 | R/W-1/1 | R/W-1/1 | R/W-1/1 | R/W-1/1 |
| TRISF7 | TRISF6 | TRISF5 | TRISF4 | TRISF3 | TRISF2 | TRISF1 | TRISF0 |
| bit 7 | | | | | | | bit 0 |

Legend:

R = Readable bit W = Writable bit U = Unimplemented bit, read as '0'
u = Bit is unchanged x = Bit is unknown -n/n = Value at POR and BOR/Value at all other Resets
'1' = Bit is set '0' = Bit is cleared

bit 7-0 **TRISF<7:0>**: PORTF Tri-State Control bits
1 = PORTF pin configured as an input (tri-stated)
0 = PORTF pin configured as an output

REGISTER 12-26: LATF: PORTF DATA LATCH REGISTER

| | | | | | | | |
|---------|---------|---------|---------|---------|---------|---------|---------|
| R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u |
| LATF7 | LATF6 | LATF5 | LATF4 | LATF3 | LATF2 | LATF1 | LATF0 |
| bit 7 | | | | | | | bit 0 |

Legend:

R = Readable bit W = Writable bit U = Unimplemented bit, read as '0'
u = Bit is unchanged x = Bit is unknown -n/n = Value at POR and BOR/Value at all other Resets
'1' = Bit is set '0' = Bit is cleared

bit 7-0 **LATF<7:0>**: PORTF Output Latch Value bits⁽¹⁾

Note 1: Writes to PORTF are actually written to corresponding LATF register. Reads from PORTF register is return of actual I/O pin values.

PIC16(L)F1526/7

REGISTER 12-27: ANSELF: PORTF ANALOG SELECT REGISTER

| | | | | | | | |
|-------|-------|-------|-------|-------|-------|-------|-------|
| R/W-1 | R/W-1 | R/W-1 | R/W-1 | R/W-1 | R/W-1 | R/W-1 | R/W-1 |
| ANSF7 | ANSF6 | ANSF5 | ANSF4 | ANSF3 | ANSF2 | ANSF1 | ANSF0 |
| bit 7 | | | | | | | bit 0 |

Legend:

R = Readable bit W = Writable bit U = Unimplemented bit, read as '0'
u = Bit is unchanged x = Bit is unknown -n/n = Value at POR and BOR/Value at all other Resets
'1' = Bit is set '0' = Bit is cleared

bit 7-0 **ANSF<7:0>**: Analog Select between Analog or Digital Function on pins RF<7:0>, respectively
1 = Analog input. Pin is assigned as analog input⁽¹⁾. Digital input buffer disabled.
0 = Digital I/O. Pin is assigned to port or digital special function.

Note 1: When setting a pin to an analog input, the corresponding TRIS bit must be set to Input mode in order to allow external control of the voltage on the pin.

TABLE 12-14: SUMMARY OF REGISTERS ASSOCIATED WITH PORTF

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Register on Page |
|--------|--------|--------|--------|--------|--------|--------|--------|--------|------------------|
| ANSELF | ANSF7 | ANSF6 | ANSF5 | ANSF4 | ANSF3 | ANSF2 | ANSF1 | ANSF0 | 130 |
| LATF | LATF7 | LATF6 | LATF5 | LATF4 | LATF3 | LATF2 | LATF1 | LATF0 | 129 |
| PORTF | RF7 | RF6 | RF5 | RF4 | RF3 | RF2 | RF1 | RF0 | 129 |
| TRISF | TRISF7 | TRISF6 | TRISF5 | TRISF4 | TRISF3 | TRISF2 | TRISF1 | TRISF0 | 129 |

Legend: x = unknown, u = unchanged, - = unimplemented locations read as '0'. Shaded cells are not used by PORTF.

TABLE 12-15: SUMMARY OF CONFIGURATION WORD WITH PORTF

| Name | Bits | Bit -/7 | Bit -/6 | Bit 13/5 | Bit 12/4 | Bit 11/3 | Bit 10/2 | Bit 9/1 | Bit 8/0 | Register on Page |
|---------|------|---------|---------|----------|-----------------------|----------|----------|----------|---------|------------------|
| CONFIG2 | 13:8 | — | — | LVP | DEBUG | LPBOR | BORV | STVREN | — | 45 |
| | 7:0 | — | — | — | VCAPEN ⁽¹⁾ | — | — | WRT<1:0> | — | |

Legend: — = unimplemented locations read as '0'. Shaded cells are not used by PORTF.

Note 1: PIC16F1526/7 only.

12.15 PORTG Registers

12.15.1 DATA REGISTER

PORTG is a 6-bit wide, bidirectional port. The corresponding data direction register is TRISG (Register 12-29). Setting a TRISG bit (= 1) will make the corresponding PORTG pin an input (i.e., disable the output driver). Clearing a TRISG bit (= 0) will make the corresponding PORTG pin an output (i.e., enables output driver and puts the contents of the output latch on the selected pin). The exception is RG5, which is input only and its TRIS bit will always read as '1'. Example 12-1 shows how to initialize an I/O port.

Reading the PORTG register (Register 12-28) reads the status of the pins, whereas writing to it will write to the PORT latch. All write operations are read-modify-write operations. Therefore, a write to a port implies that the port pins are read, this value is modified and then written to the PORT data latch (LATG).

12.15.2 DIRECTION CONTROL

The TRISG register (Register 12-29) controls the PORTG pin output drivers, even when they are being used as analog inputs. The user should ensure the bits in the TRISG register are maintained set when using them as analog inputs. I/O pins configured as analog input always read '0'.

12.15.3 ANALOG CONTROL

The ANSELG register (Register 12-31) is used to configure the Input mode of an I/O pin to analog. Setting the appropriate ANSELG bit high will cause all digital reads on the pin to be read as '0' and allow analog functions on the pin to operate correctly.

The state of the ANSELG bits has no effect on digital output functions. A pin with TRIS clear and ANSEL set will still operate as a digital output, but the Input mode will be analog. This can cause unexpected behavior when executing read-modify-write instructions on the affected port.

Note: The ANSELG bits default to the Analog mode after Reset. To use any pins as digital general purpose or peripheral inputs, the corresponding ANSEL bits must be initialized to '0' by user software.

12.15.4 PORTG FUNCTIONS AND OUTPUT PRIORITIES

Each PORTG pin is multiplexed with other functions. The pins, their combined functions and their output priorities are shown in Table 12-16.

When multiple outputs are enabled, the actual pin control goes to the peripheral with the highest priority.

Analog input functions, such as ADC, are not shown in the priority lists. These inputs are active when the I/O pin is set for Analog mode using the ANSELx registers. Digital output functions may control the pin when it is in Analog mode with the priority list.

TABLE 12-16: PORTG OUTPUT PRIORITY

| Pin Name | Function Priority ⁽¹⁾ |
|----------|----------------------------------|
| RG0 | CCP3 RG0 |
| RG1 | CK2 TX2 RG1 |
| RG2 | DT2 RG2 |
| RG3 | CCP4 RG3 |
| RG4 | CCP5 RG4 |
| RG5 | Input only pin |

Note 1: Priority listed from highest to lowest.

PIC16(L)F1526/7

12.16 Register Definitions: PORTG

REGISTER 12-28: PORTG: PORTG REGISTER

| | | | | | | | |
|-------|-----|-------|---------|---------|---------|---------|---------|
| U-0 | U-0 | R-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u |
| — | — | RG5 | RG4 | RG3 | RG2 | RG1 | RG0 |
| bit 7 | | | | | | | bit 0 |

Legend:

R = Readable bit W = Writable bit U = Unimplemented bit, read as '0'
u = Bit is unchanged x = Bit is unknown -n/n = Value at POR and BOR/Value at all other Resets
'1' = Bit is set '0' = Bit is cleared

bit 7-6 **Unimplemented:** Read as '0'
bit 5-0 **RG<5:0>:** PORTG I/O Pin bits⁽¹⁾
 1 = Port pin is > V_{IH}
 0 = Port pin is < V_{IL}

Note 1: Writes to PORTG are actually written to corresponding LATG register. Reads from PORTG register is return of actual I/O pin values.

REGISTER 12-29: TRISG: PORTG TRI-STATE REGISTER

| | | | | | | | |
|-------|-----|------------------|--------|--------|--------|--------|--------|
| U-0 | U-0 | U-1 | R/W-1 | R/W-1 | R/W-1 | R/W-1 | R/W-1 |
| — | — | — ⁽¹⁾ | TRISG4 | TRISG3 | TRISG2 | TRISG1 | TRISG0 |
| bit 7 | | | | | | | bit 0 |

Legend:

R = Readable bit W = Writable bit U = Unimplemented bit, read as '0'
u = bit is unchanged x = Bit is unknown -n/n = Value at POR and BOR/Value at all other Resets
'1' = Bit is set '0' = Bit is cleared

bit 7-6 **Unimplemented:** Read as '0'
bit 5 **Unimplemented:** Read as '1'
bit 4-0 **TRISG<4:0>:** RG<4:0> Tri-State Control bits⁽¹⁾
 1 = PORTG pin configured as an input (tri-stated)
 0 = PORTG pin configured as an output

Note 1: Unimplemented, read as '1'.

REGISTER 12-30: LATG: PORTG DATA LATCH REGISTER

| | | | | | | | |
|-------|-----|-----|---------|---------|---------|---------|---------|
| U-0 | U-0 | U-0 | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u |
| — | — | — | LATG4 | LATG3 | LATG2 | LATG1 | LATG0 |
| bit 7 | | | | | | | bit 0 |

Legend:

| | | |
|----------------------|----------------------|---|
| R = Readable bit | W = Writable bit | U = Unimplemented bit, read as '0' |
| u = Bit is unchanged | x = Bit is unknown | -n/n = Value at POR and BOR/Value at all other Resets |
| '1' = Bit is set | '0' = Bit is cleared | |

bit 7-5 **Unimplemented:** Read as '0'

bit 4-0 **LATG<4:0>:** PORTG Output Latch Value bits⁽¹⁾

Note 1: Writes to PORTG are actually written to corresponding LATG register. Reads from PORTG register is return of actual I/O pin values.

REGISTER 12-31: ANSELG: PORTG ANALOG SELECT REGISTER

| | | | | | | | |
|-------|-----|-----|-------|-------|-------|-------|-------|
| U-0 | U-0 | U-0 | R/W-1 | R/W-1 | R/W-1 | R/W-1 | U-0 |
| — | — | — | ANSG4 | ANSG3 | ANSG2 | ANSG1 | — |
| bit 7 | | | | | | | bit 0 |

Legend:

| | | |
|----------------------|----------------------|---|
| R = Readable bit | W = Writable bit | U = Unimplemented bit, read as '0' |
| u = bit is unchanged | x = Bit is unknown | -n/n = Value at POR and BOR/Value at all other Resets |
| '1' = Bit is set | '0' = Bit is cleared | |

bit 7-5 **Unimplemented:** Read as '0'

bit 4-1 **ANSG<4:1>:** Analog Select between Analog or Digital Function on Pins RG<4:1>, respectively
 1 = Analog input. Pin is assigned as analog input⁽¹⁾. Digital input buffer disabled.
 0 = Digital I/O. Pin is assigned to port or digital special function.

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

Note 1: When setting a pin to an analog input, the corresponding TRIS bit must be set to Input mode in order to allow external control of the voltage on the pin.

PIC16(L)F1526/7

REGISTER 12-32: WPUG: WEAK PULL-UP PORTG REGISTER

| | | | | | | | | |
|-------|-----|---------|-----|-----|-----|-----|-------|--|
| U-0 | U-0 | R/W-1/1 | U-0 | U-0 | U-0 | U-0 | U-0 | |
| — | — | WPUG5 | — | — | — | — | — | |
| bit 7 | | | | | | | bit 0 | |

Legend:

| | | |
|----------------------|----------------------|---|
| R = Readable bit | W = Writable bit | U = Unimplemented bit, read as '0' |
| u = Bit is unchanged | x = Bit is unknown | -n/n = Value at POR and BOR/Value at all other Resets |
| '1' = Bit is set | '0' = Bit is cleared | |

bit 7-6 **Unimplemented:** Read as '0'

bit 5 **WPUG5:** Weak Pull-up Register bit
 1 = Pull-up enabled
 0 = Pull-up disabled

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

Note 1: Global WPUEN bit of the OPTION_REG register must be cleared for individual pull-ups to be enabled.

2: The weak pull-up device is automatically disabled if the pin is in configured as an output.

TABLE 12-17: SUMMARY OF REGISTERS ASSOCIATED WITH PORTG

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Register on Page |
|--------|-------|-------|------------------|--------|--------|--------|--------|--------|------------------|
| ANSELG | — | — | — | ANSG4 | ANSG3 | ANSG2 | ANSG1 | — | 133 |
| LATG | — | — | — | LATG4 | LATG3 | LATG2 | LATG1 | LATG0 | 133 |
| PORTG | — | — | RG5 | RG4 | RG3 | RG2 | RG1 | RG0 | 132 |
| TRISG | — | — | — ⁽¹⁾ | TRISG4 | TRISG3 | TRISG2 | TRISG1 | TRISG0 | 132 |
| WPUG | — | — | WPUG5 | — | — | — | — | — | 134 |

Legend: x = unknown, u = unchanged, — = unimplemented locations read as '0'. Shaded cells are not used by PORTG.

Note 1: Unimplemented, read as '1'.

TABLE 12-18: SUMMARY OF CONFIGURATION WORD WITH PORTG

| Name | Bits | Bit -/7 | Bit -/6 | Bit 13/5 | Bit 12/4 | Bit 11/3 | Bit 10/2 | Bit 9/1 | Bit 8/0 | Register on Page |
|---------|------|---------|---------|----------|-----------|----------|------------|---------|---------|------------------|
| CONFIG1 | 13:8 | — | — | FCMEN | IESO | CLKOUTEN | BOREN<1:0> | — | — | 45 |
| | 7:0 | CP | MCLRE | PWRTE | WDTE<1:0> | — | FOSC<2:0> | — | — | |

Legend: — = unimplemented location, read as '0'. Shaded cells are not used by PORTG.

13.0 INTERRUPT-ON-CHANGE

The PORTB pins can be configured to operate as Interrupt-On-Change (IOC) pins. An interrupt can be generated by detecting a signal that has either a rising edge or a falling edge. Any individual PORTB pin, or combination of PORTB pins, can be configured to generate an interrupt. The interrupt-on-change module has the following features:

- Interrupt-on-Change enable (Master Switch)
- Individual pin configuration
- Rising and falling edge detection
- Individual pin interrupt flags

Figure 13-1 is a block diagram of the IOC module.

13.1 Enabling the Module

To allow individual PORTB pins to generate an interrupt, the IOCIE bit of the INTCON register must be set. If the IOCIE bit is disabled, the edge detection on the pin will still occur, but an interrupt will not be generated.

13.2 Individual Pin Configuration

For each PORTB pin, a rising edge detector and a falling edge detector are present. To enable a pin to detect a rising edge, the associated IOCBP_x bit of the IOCBP register is set. To enable a pin to detect a falling edge, the associated IOCBN_x bit of the IOCBN register is set.

A pin can be configured to detect rising and falling edges simultaneously by setting both the IOCBP_x bit and the IOCBN_x bit of the IOCBP and IOCBN registers, respectively.

13.3 Interrupt Flags

The IOCBF_x bits located in the IOCBF register are status flags that correspond to the Interrupt-on-change pins of PORTB. If an expected edge is detected on an appropriately enabled pin, then the status flag for that pin will be set, and an interrupt will be generated if the IOCIE bit is set. The IOCIF bit of the INTCON register reflects the status of all IOCBF_x bits.

13.4 Clearing Interrupt Flags

The individual status flags, (IOCBF_x bits), can be cleared by resetting them to zero. If another edge is detected during this clearing operation, the associated status flag will be set at the end of the sequence, regardless of the value actually being written.

In order to ensure that no detected edge is lost while clearing flags, only AND operations masking out known changed bits should be performed. The following sequence is an example of what should be performed.

EXAMPLE 13-1: CLEARING INTERRUPT FLAGS (PORTA EXAMPLE)

```
MOVLW 0xff
XORWF IOCAF, W
ANDWF IOCAF, F
```

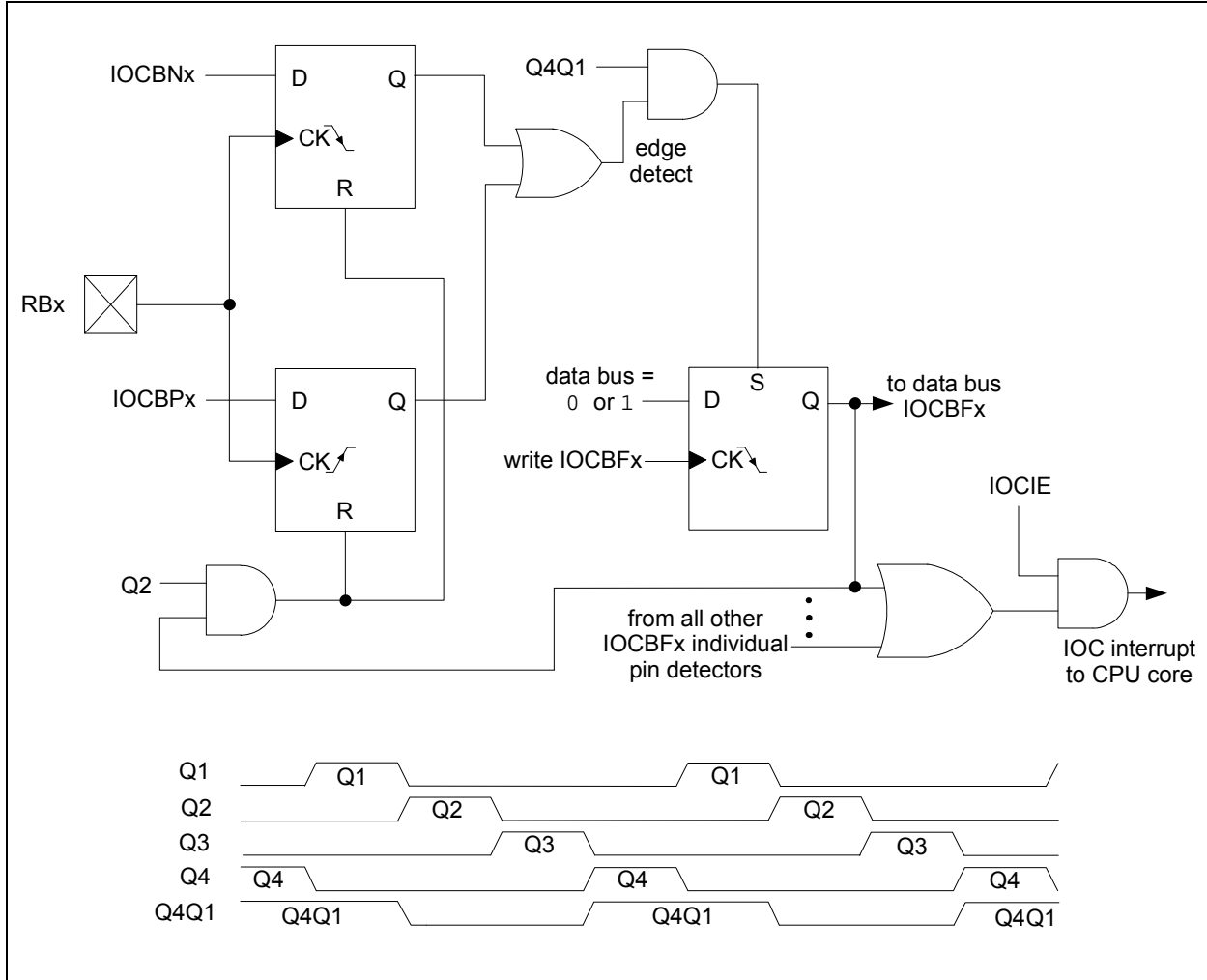
13.5 Operation in Sleep

The interrupt-on-change interrupt sequence will wake the device from Sleep mode, if the IOCIE bit is set.

If an edge is detected while in Sleep mode, the IOCBF register will be updated prior to the first instruction executed out of Sleep.

PIC16(L)F1526/7

FIGURE 13-1: INTERRUPT-ON-CHANGE BLOCK DIAGRAM



13.6 Register Definitions: Interrupt-on-change Control

REGISTER 13-1: IOCBP: INTERRUPT-ON-CHANGE PORTB POSITIVE EDGE REGISTER

| | | | | | | | |
|---------|---------|---------|---------|---------|---------|---------|---------|
| R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 |
| IOCBP7 | IOCBP6 | IOCBP5 | IOCBP4 | IOCBP3 | IOCBP2 | IOCBP1 | IOCBP0 |
| bit 7 | | | | | | | bit 0 |

Legend:

R = Readable bit W = Writable bit U = Unimplemented bit, read as '0'
u = Bit is unchanged x = Bit is unknown -n/n = Value at POR and BOR/Value at all other Resets
'1' = Bit is set '0' = Bit is cleared

bit 7-0 **IOCBP<7:0>**: Interrupt-on-Change PORTB Positive Edge Enable bits
1 = Interrupt-on-Change enabled on the pin for a positive going edge. IOCBFx bit and IOCIF flag will be set upon detecting an edge.
0 = Interrupt-on-Change disabled for the associated pin

REGISTER 13-2: IOCBN: INTERRUPT-ON-CHANGE PORTB NEGATIVE EDGE REGISTER

| | | | | | | | |
|---------|---------|---------|---------|---------|---------|---------|---------|
| R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 |
| IOCBN7 | IOCBN6 | IOCBN5 | IOCBN4 | IOCBN3 | IOCBN2 | IOCBN1 | IOCBN0 |
| bit 7 | | | | | | | bit 0 |

Legend:

R = Readable bit W = Writable bit U = Unimplemented bit, read as '0'
u = Bit is unchanged x = Bit is unknown -n/n = Value at POR and BOR/Value at all other Resets
'1' = Bit is set '0' = Bit is cleared

bit 7-0 **IOCBN<7:0>**: Interrupt-on-Change PORTB Negative Edge Enable bits
1 = Interrupt-on-Change enabled on the pin for a negative going edge. IOCBFx bit and IOCIF flag will be set upon detecting an edge.
0 = Interrupt-on-Change disabled for the associated pin

REGISTER 13-3: IOCBF: INTERRUPT-ON-CHANGE PORTB FLAG REGISTER

| | | | | | | | |
|------------|------------|------------|------------|------------|------------|------------|------------|
| R/W/HS-0/0 | R/W/HS-0/0 | R/W/HS-0/0 | R/W/HS-0/0 | R/W/HS-0/0 | R/W/HS-0/0 | R/W/HS-0/0 | R/W/HS-0/0 |
| IOCBF7 | IOCBF6 | IOCBF5 | IOCBF4 | IOCBF3 | IOCBF2 | IOCBF1 | IOCBF0 |
| bit 7 | | | | | | | bit 0 |

Legend:

R = Readable bit W = Writable bit U = Unimplemented bit, read as '0'
u = Bit is unchanged x = Bit is unknown -n/n = Value at POR and BOR/Value at all other Resets
'1' = Bit is set '0' = Bit is cleared HS - Bit is set in hardware

bit 7-0 **IOCBF<7:0>**: Interrupt-on-Change PORTB Flag bits
1 = An enabled change was detected on the associated pin.
Set when IOCBPx = 1 and a rising edge was detected on RBx, or when IOCBNx = 1 and a falling edge was detected on RBx.
0 = No change was detected, or the user cleared the detected change.

PIC16(L)F1526/7

TABLE 13-1: SUMMARY OF REGISTERS ASSOCIATED WITH INTERRUPT-ON-CHANGE

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Register on Page |
|--------|--------|--------|--------|--------|--------|--------|--------|--------|------------------|
| ANSELB | — | — | ANSB5 | ANSB4 | ANSB3 | ANSB2 | ANSB1 | ANSB0 | 118 |
| INTCON | GIE | PEIE | TMR0IE | INTE | IOCIE | TMR0IF | INTF | IOCIF | 76 |
| IOCBP | IOCBP7 | IOCBP6 | IOCBP5 | IOCBP4 | IOCBP3 | IOCBP2 | IOCBP1 | IOCBP0 | 137 |
| IOCBN | IOCBN7 | IOCBN6 | IOCBN5 | IOCBN4 | IOCBN3 | IOCBN2 | IOCBN1 | IOCBN0 | 137 |
| IOCBF | IOCBF7 | IOCBF6 | IOCBF5 | IOCBF4 | IOCBF3 | IOCBF2 | IOCBF1 | IOCBF0 | 137 |
| TRISB | TRISB7 | TRISB6 | TRISB5 | TRISB4 | TRISB3 | TRISB2 | TRISB1 | TRISB0 | 117 |

Legend: — = unimplemented location, read as '0'. Shaded cells are not used by Interrupt-on-Change.

14.0 FIXED VOLTAGE REFERENCE (FVR)

The Fixed Voltage Reference, or FVR, is a stable voltage reference, independent of V_{DD} , with 1.024V, 2.048V or 4.096V selectable output levels. The output of the FVR can be configured to supply a reference voltage to the following:

- ADC input channel
- ADC positive reference

The FVR can be enabled by setting the FVREN bit of the FVRCON register.

14.1 Independent Gain Amplifiers

The output of the FVR supplied to the ADC module is routed through two independent programmable gain amplifiers. Each amplifier can be configured to amplify the reference voltage by 1x, 2x or 4x, to produce the three possible voltage levels.

The ADFVR<1:0> bits of the FVRCON register are used to enable and configure the gain amplifier settings for the reference supplied to the ADC module. Reference [Section 16.0 “Analog-to-Digital Converter \(ADC\) Module”](#) for additional information.

14.2 FVR Stabilization Period

When the Fixed Voltage Reference module is enabled, it requires time for the reference and amplifier circuits to stabilize. Once the circuits stabilize and are ready for use, the FVRRDY bit of the FVRCON register will be set. See [Section 25.0 “Electrical Specifications”](#) for the minimum delay requirement.

FIGURE 14-1: VOLTAGE REFERENCE BLOCK DIAGRAM

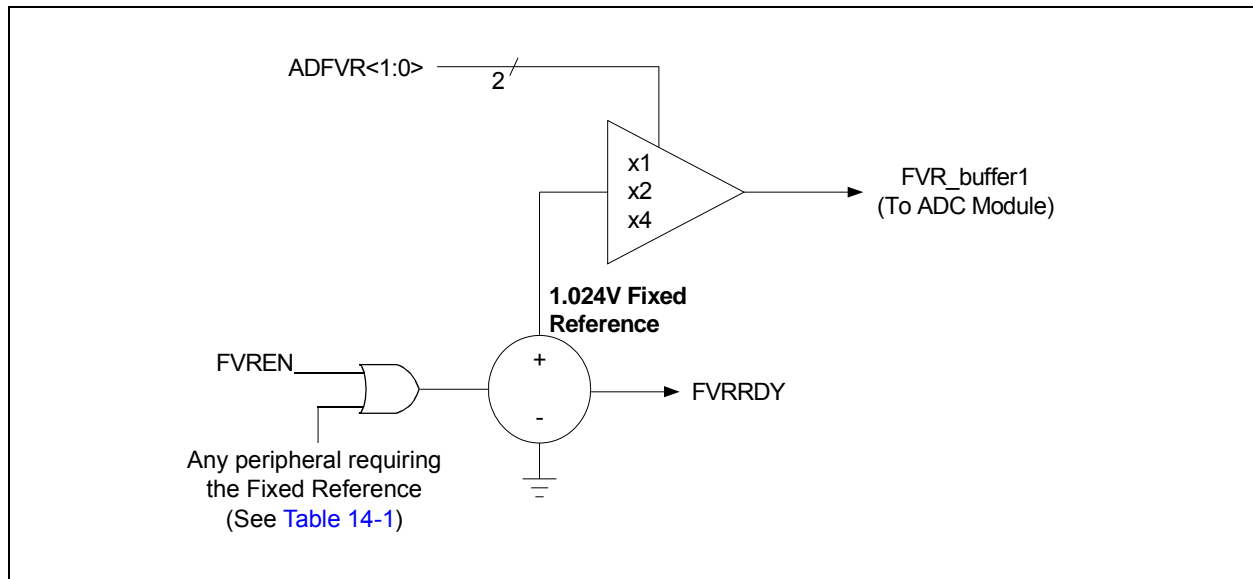


TABLE 14-1: PERIPHERALS REQUIRING THE FIXED VOLTAGE REFERENCE (FVR)

| Peripheral | Conditions | Description |
|------------|---|--|
| HFINTOSC | FOSC<2:0> = 100 and IRCF<3:0> = 000x | INTOSC is active and device is not in Sleep. |
| BOR | BOREN<1:0> = 11 | BOR always enabled. |
| | BOREN<1:0> = 10 and BORFS = 1 | BOR disabled in Sleep mode, BOR Fast Start enabled. |
| | BOREN<1:0> = 01 and BORFS = 1 | BOR under software control, BOR Fast Start enabled. |
| LDO | All PIC16F1526/7 devices, when VREGPM = 1 and not in Sleep | The device runs off of the low-power regulator when in Sleep mode. |

PIC16(L)F1526/7

14.3 Register Definitions: FVR Control

REGISTER 14-1: FVRCON: FIXED VOLTAGE REFERENCE CONTROL REGISTER

| | | | | | | | |
|---------|-----------------------|---------|---------|-----|-----|------------|---------|
| R/W-0/0 | R-q/q | R/W-0/0 | R/W-0/0 | U-0 | U-0 | R/W-0/0 | R/W-0/0 |
| RVREN | FVRRDY ⁽¹⁾ | TSEN | TSRNG | — | — | ADFVR<1:0> | |
| bit 7 | | | | | | bit 0 | |

Legend:

| | | |
|----------------------|----------------------|---|
| R = Readable bit | W = Writable bit | U = Unimplemented bit, read as '0' |
| u = Bit is unchanged | x = Bit is unknown | -n/n = Value at POR and BOR/Value at all other Resets |
| '1' = Bit is set | '0' = Bit is cleared | q = Value depends on condition |

- bit 7 **RVREN:** Fixed Voltage Reference Enable bit
1 = Fixed Voltage Reference is enabled
0 = Fixed Voltage Reference is disabled
- bit 6 **FVRRDY:** Fixed Voltage Reference Ready Flag bit⁽¹⁾
1 = Fixed Voltage Reference output is ready for use
0 = Fixed Voltage Reference output is not ready or not enabled
- bit 5 **TSEN:** Temperature Indicator Enable bit
1 = Temperature Indicator is enabled
0 = Temperature Indicator is disabled
- bit 4 **TSRNG:** Temperature Indicator Range Selection bit
1 = V_{OUT} = V_{DD} - 4V_T (High Range)
0 = V_{OUT} = V_{DD} - 2V_T (Low Range)
- bit 3-2 **Unimplemented:** Read as '0'
- bit 1-0 **ADFVR<1:0>:** ADC Fixed Voltage Reference Selection bits
11 = ADC Fixed Voltage Reference Peripheral output is 4x (4.096V)⁽²⁾
10 = ADC Fixed Voltage Reference Peripheral output is 2x (2.048V)⁽²⁾
01 = ADC Fixed Voltage Reference Peripheral output is 1x (1.024V)
00 = ADC Fixed Voltage Reference Peripheral output is off

- Note 1:** FVRRDY is always '1' on PIC16F1526/7 only.
Note 2: Fixed Voltage Reference output cannot exceed V_{DD}.

TABLE 14-2: SUMMARY OF REGISTERS ASSOCIATED WITH FIXED VOLTAGE REFERENCE

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Register on page |
|--------|-------|--------|-------|-------|-------|-------|------------|-------|------------------|
| FVRCON | RVREN | FVRRDY | TSEN | TSRNG | — | — | ADFVR<1:0> | | 140 |

Legend: Shaded cells are unused by the Fixed Voltage Reference.

15.0 TEMPERATURE INDICATOR MODULE

This family of devices is equipped with a temperature circuit designed to measure the operating temperature of the silicon die. The circuit's range of operating temperature falls between -40°C and +85°C. The output is a voltage that is proportional to the device temperature. The output of the temperature indicator is internally connected to the device ADC.

The circuit may be used as a temperature threshold detector or a more accurate temperature indicator, depending on the level of calibration performed. A one-point calibration allows the circuit to indicate a temperature closely surrounding that point. A two-point calibration allows the circuit to sense the entire range of temperature more accurately. Reference Application Note AN1333, *Use and Calibration of the Internal Temperature Indicator (DS01333)* for more details regarding the calibration process.

15.1 Circuit Operation

Figure 15-1 shows a simplified block diagram of the temperature circuit. The proportional voltage output is achieved by measuring the forward voltage drop across multiple silicon junctions.

Equation 15-1 describes the output characteristics of the temperature indicator.

EQUATION 15-1: V_{OUT} RANGES

High Range: $V_{OUT} = V_{DD} - 4V_T$

Low Range: $V_{OUT} = V_{DD} - 2V_T$

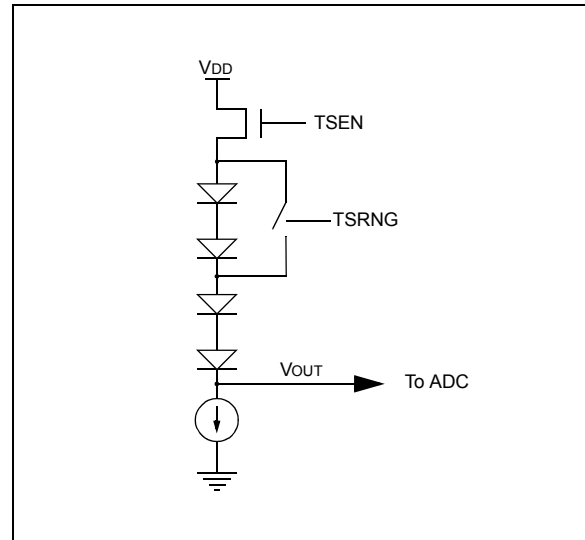
The temperature sense circuit is integrated with the Fixed Voltage Reference (FVR) module. See Section 14.0 "Fixed Voltage Reference (FVR)" for more information.

The circuit is enabled by setting the TSEN bit of the FVRCON register. When disabled, the circuit draws no current.

The circuit operates in either high or low range. The high range, selected by setting the TSRNG bit of the FVRCON register, provides a wider output voltage. This provides more resolution over the temperature range, but may be less consistent from part to part. This range requires a higher bias voltage to operate and thus, a higher V_{DD} is needed.

The low range is selected by clearing the TSRNG bit of the FVRCON register. The low range generates a lower voltage drop and thus, a lower bias voltage is needed to operate the circuit. The low range is provided for low voltage operation.

FIGURE 15-1: TEMPERATURE CIRCUIT DIAGRAM



15.2 Minimum Operating V_{DD}

When the temperature circuit is operated in low range, the device may be operated at any operating voltage that is within specifications.

When the temperature circuit is operated in high range, the device operating voltage, V_{DD}, must be high enough to ensure that the temperature circuit is correctly biased.

Table 15-1 shows the recommended minimum V_{DD} vs. range setting.

TABLE 15-1: RECOMMENDED V_{DD} VS. RANGE

| Min. V _{DD} , TSRNG = 1 | Min. V _{DD} , TSRNG = 0 |
|----------------------------------|----------------------------------|
| 3.6V | 1.8V |

15.3 Temperature Output

The output of the circuit is measured using the internal Analog-to-Digital Converter. A channel is reserved for the temperature circuit output. Refer to Section 16.0 "Analog-to-Digital Converter (ADC) Module" for detailed information.

Note: Every time the ADC MUX is changed to the temperature indicator output selection (CHS bit in the ADCCON0 register), wait 500 μsec for the sampling capacitor to fully charge before sampling the temperature indicator output.

PIC16(L)F1526/7

15.4 ADC Acquisition Time

To ensure accurate temperature measurements, the user must wait at least 200 μ s after the ADC input multiplexer is connected to the temperature indicator output before the conversion is performed. In addition, the user must wait 200 μ s between sequential conversions of the temperature indicator output.

TABLE 15-2: SUMMARY OF REGISTERS ASSOCIATED WITH THE TEMPERATURE INDICATOR

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Register on page |
|--------|-------|--------|-------|-------|-------|-------|------------|-------|---------------------|
| FVRCON | FVREN | FVRRDY | TSEN | TSRNG | — | — | ADFVR<1:0> | | 140 |

Legend: Shaded cells are unused by the temperature indicator module.

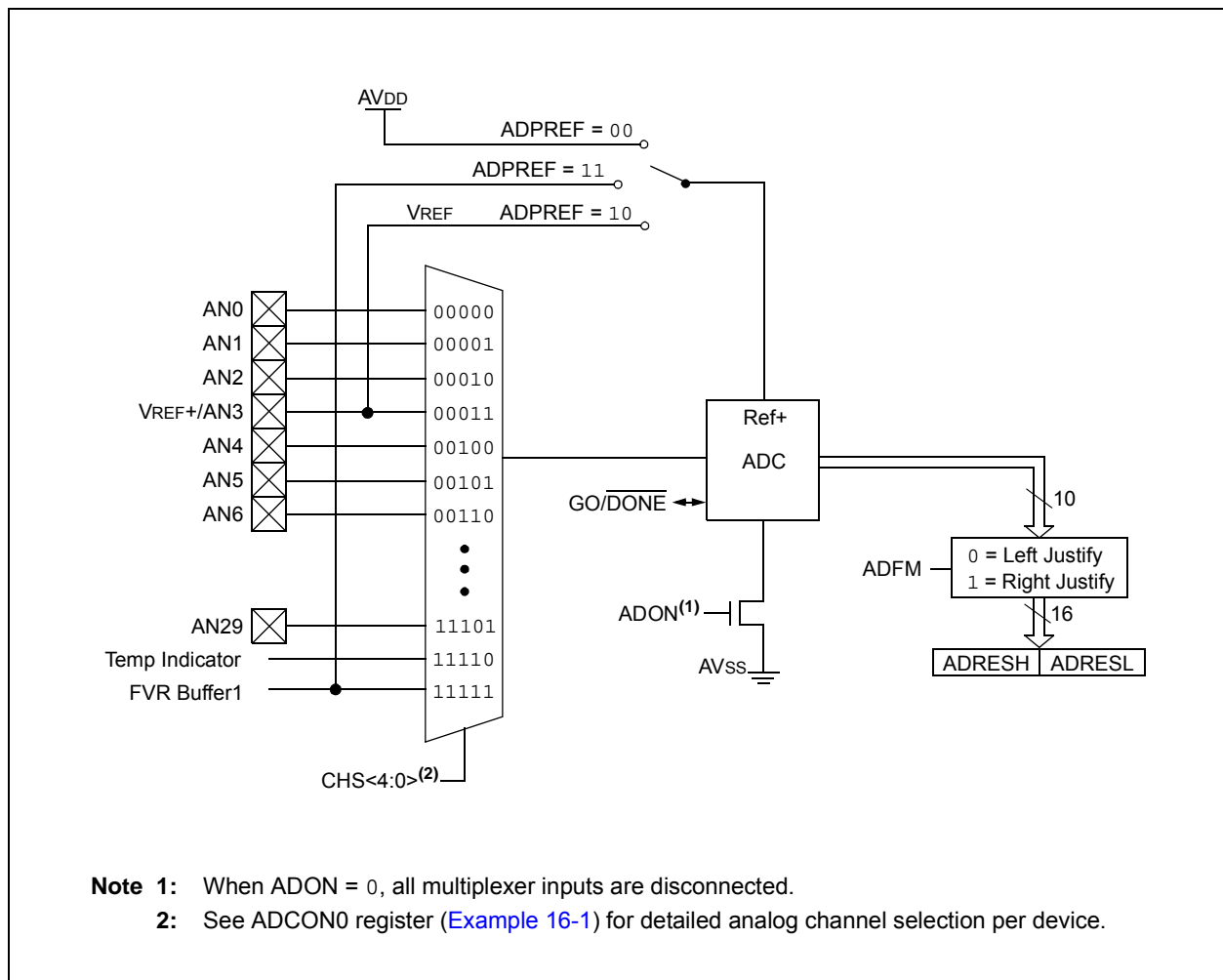
16.0 ANALOG-TO-DIGITAL CONVERTER (ADC) MODULE

The Analog-to-Digital Converter (ADC) allows conversion of an analog input signal to a 10-bit binary representation of that signal. This device uses analog inputs, which are multiplexed into a single sample and hold circuit. The output of the sample and hold is connected to the input of the converter. The converter generates a 10-bit binary result via successive approximation and stores the conversion result into the ADC result registers (ADRESH:ADRESL register pair). Figure 16-1 shows the block diagram of the ADC.

The ADC voltage reference is software selectable to be either internally generated or externally supplied.

The ADC can generate an interrupt upon completion of a conversion. This interrupt can be used to wake-up the device from Sleep.

FIGURE 16-1: ADC BLOCK DIAGRAM



- Note 1:** When ADON = 0, all multiplexer inputs are disconnected.
Note 2: See ADCON0 register (Example 16-1) for detailed analog channel selection per device.

PIC16(L)F1526/7

16.1 ADC Configuration

When configuring and using the ADC the following functions must be considered:

- Port configuration
- Channel selection
- ADC voltage reference selection
- ADC conversion clock source
- Interrupt control
- Result formatting

16.1.1 PORT CONFIGURATION

The ADC can be used to convert both analog and digital signals. When converting analog signals, the I/O pin should be configured for analog by setting the associated TRIS and ANSEL bits. Refer to [Section 12.0 “I/O Ports”](#) for more information.

Note: Analog voltages on any pin that is defined as a digital input may cause the input buffer to conduct excess current.

16.1.2 CHANNEL SELECTION

There are 32 channel selections available:

- AN<29:0> pins
- Temperature Indicator
- FVR (Fixed Voltage Reference) Output

Refer to [Section 14.0 “Fixed Voltage Reference \(FVR\)”](#) and [Section 15.0 “Temperature Indicator Module”](#) for more information on these channel selections.

The CHS bits of the ADCON0 register determine which channel is connected to the sample and hold circuit.

When changing channels, a delay is required before starting the next conversion. Refer to [Section 16.2 “ADC Operation”](#) for more information.

16.1.3 ADC VOLTAGE REFERENCE

The ADPREF bits of the ADCON1 register provides control of the positive voltage reference. The positive voltage reference can be:

- VREF+ pin
- VDD
- FVR 2.048V
- FVR 4.096V (Not available on LF devices)

See [Section 14.0 “Fixed Voltage Reference \(FVR\)”](#) for more details on the Fixed Voltage Reference.

16.1.4 CONVERSION CLOCK

The source of the conversion clock is software selectable via the ADCS bits of the ADCON1 register. There are seven possible clock options:

- FOSC/2
- FOSC/4
- FOSC/8
- FOSC/16
- FOSC/32
- FOSC/64
- FRC (dedicated internal FRC oscillator)

The time to complete one bit conversion is defined as TAD. One full 10-bit conversion requires 11.5 TAD periods as shown in [Figure 16-2](#).

For correct conversion, the appropriate TAD specification must be met. Refer to the ADC conversion requirements in [Section 25.0 “Electrical Specifications”](#) for more information. [Table 16-1](#) gives examples of appropriate ADC clock selections.

Note: Unless using the FRC, any changes in the system clock frequency will change the ADC clock frequency, which may adversely affect the ADC result.

TABLE 16-1: ADC CLOCK PERIOD (TAD) Vs. DEVICE OPERATING FREQUENCIES

| ADC Clock Period (TAD) | | Device Frequency (Fosc) | | | | |
|------------------------|-----------|-----------------------------|-----------------------------|-----------------------------|-----------------------------|-----------------------------|
| ADC Clock Source | ADCS<2:0> | 20 MHz | 16 MHz | 8 MHz | 4 MHz | 1 MHz |
| Fosc/2 | 000 | 100 ns ⁽²⁾ | 125 ns ⁽²⁾ | 250 ns ⁽²⁾ | 500 ns ⁽²⁾ | 2.0 μs |
| Fosc/4 | 100 | 200 ns ⁽²⁾ | 250 ns ⁽²⁾ | 500 ns ⁽²⁾ | 1.0 μs | 4.0 μs |
| Fosc/8 | 001 | 400 ns ⁽²⁾ | 0.5 μs ⁽²⁾ | 1.0 μs | 2.0 μs | 8.0 μs ⁽³⁾ |
| Fosc/16 | 101 | 800 ns | 1.0 μs | 2.0 μs | 4.0 μs | 16.0 μs ⁽³⁾ |
| Fosc/32 | 010 | 1.6 μs | 2.0 μs | 4.0 μs | 8.0 μs ⁽³⁾ | 32.0 μs ⁽³⁾ |
| Fosc/64 | 110 | 3.2 μs | 4.0 μs | 8.0 μs ⁽³⁾ | 16.0 μs ⁽³⁾ | 64.0 μs ⁽³⁾ |
| FRC | x11 | 1.0-6.0 μs ^(1,4) | 1.0-6.0 μs ^(1,4) | 1.0-6.0 μs ^(1,4) | 1.0-6.0 μs ^(1,4) | 1.0-6.0 μs ^(1,4) |

Legend: Shaded cells are outside of recommended range.

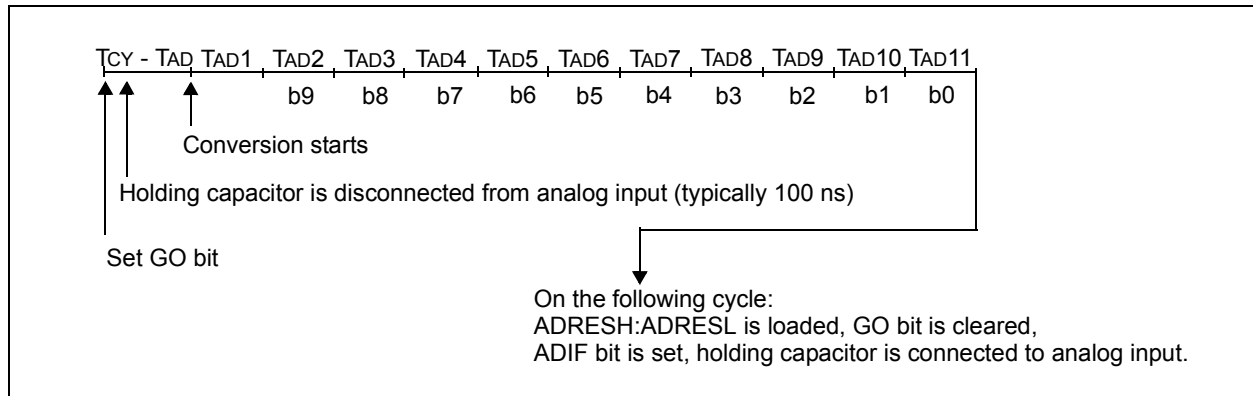
Note 1: The FRC source has a typical TAD time of 1.6 μs for VDD.

2: These values violate the minimum required TAD time.

3: For faster conversion times, the selection of another clock source is recommended.

4: The ADC clock period (TAD) and total ADC conversion time can be minimized when the ADC clock is derived from the system clock FOSC. However, the FRC clock source must be used when conversions are to be performed with the device in Sleep mode.

FIGURE 16-2: ANALOG-TO-DIGITAL CONVERSION TAD CYCLES



PIC16(L)F1526/7

16.1.5 INTERRUPTS

The ADC module allows for the ability to generate an interrupt upon completion of an Analog-to-Digital conversion. The ADC Interrupt Flag is the ADIF bit in the PIR1 register. The ADC Interrupt Enable is the ADIE bit in the PIE1 register. The ADIF bit must be cleared in software.

Note 1: The ADIF bit is set at the completion of every conversion, regardless of whether or not the ADC interrupt is enabled.

2: The ADC operates during Sleep only when the FRC oscillator is selected.

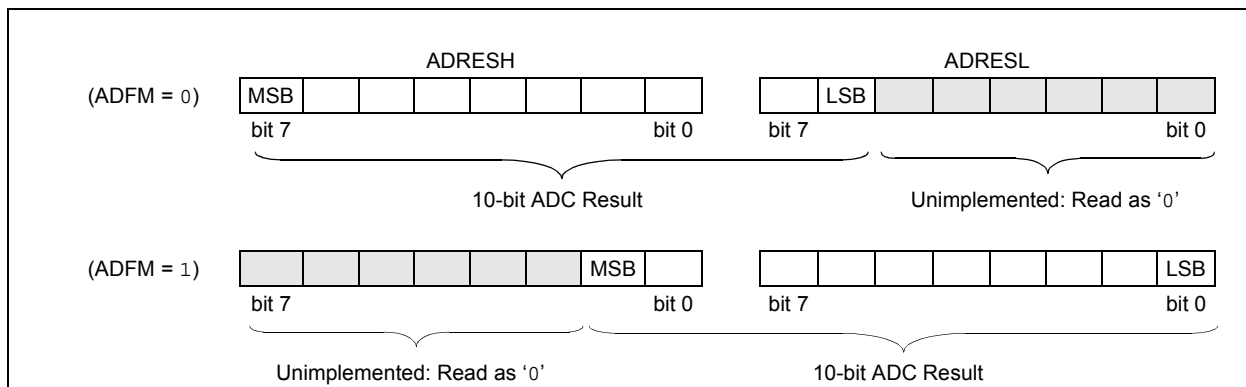
This interrupt can be generated while the device is operating or while in Sleep. If the device is in Sleep, the interrupt will wake-up the device. Upon waking from Sleep, the next instruction following the `SLEEP` instruction is always executed. If the user is attempting to wake-up from Sleep and resume in-line code execution, the GIE and PEIE bits of the INTCON register must be disabled. If the GIE and PEIE bits of the INTCON register are enabled, execution will switch to the Interrupt Service Routine.

16.1.6 RESULT FORMATTING

The 10-bit ADC conversion result can be supplied in two formats, left justified or right justified. The ADFM bit of the ADCON1 register controls the output format.

Figure 16-3 shows the two output formats.

FIGURE 16-3: 10-BIT ADC CONVERSION RESULT FORMAT



16.2 ADC Operation

16.2.1 STARTING A CONVERSION

To enable the ADC module, the ADON bit of the ADCON0 register must be set to a '1'. Setting the GO/DONE bit of the ADCON0 register to a '1' will start the Analog-to-Digital conversion.

Note: The GO/DONE bit should not be set in the same instruction that turns on the ADC. Refer to [Section 16.2.6 “ADC Conversion Procedure”](#).

16.2.2 COMPLETION OF A CONVERSION

When the conversion is complete, the ADC module will:

- Clear the GO/DONE bit
- Set the ADIF Interrupt Flag bit
- Update the ADRESH and ADRESL registers with new conversion result

16.2.3 TERMINATING A CONVERSION

If a conversion must be terminated before completion, the GO/DONE bit can be cleared in software. The ADRESH and ADRESL registers will be updated with the partially complete Analog-to-Digital conversion sample. Incomplete bits will match the last bit converted.

Note: A device Reset forces all registers to their Reset state. Thus, the ADC module is turned off and any pending conversion is terminated.

16.2.4 ADC OPERATION DURING SLEEP

The ADC module can operate during Sleep. This requires the ADC clock source to be set to the FRC option. When the FRC oscillator source is selected, the ADC waits one additional instruction before starting the conversion. This allows the SLEEP instruction to be executed, which can reduce system noise during the conversion. If the ADC interrupt is enabled, the device will wake-up from Sleep when the conversion completes. If the ADC interrupt is disabled, the ADC module is turned off after the conversion completes, although the ADON bit remains set.

When the ADC clock source is something other than FRC, a SLEEP instruction causes the present conversion to be aborted and the ADC module is turned off, although the ADON bit remains set.

16.2.5 SPECIAL EVENT TRIGGER

The Special Event Trigger of the CCPx module allows periodic ADC measurements without software intervention. When this trigger occurs, the GO/DONE bit is set by hardware and the Timer1 counter resets to zero.

TABLE 16-2: SPECIAL EVENT TRIGGER

| Device | CCP |
|-----------------|-------|
| PIC16(L)F1526/7 | CCP10 |

Using the Special Event Trigger does not assure proper ADC timing. It is the user's responsibility to ensure that the ADC timing requirements are met.

Refer to [Section 20.0 “Capture/Compare/PWM Modules”](#) for more information.

PIC16(L)F1526/7

16.2.6 ADC CONVERSION PROCEDURE

This is an example procedure for using the ADC to perform an Analog-to-Digital conversion:

1. Configure Port:
 - Disable pin output driver (Refer to the TRIS register)
 - Configure pin as analog (Refer to the ANSEL register)
 - Disable weak pull-ups either globally (Refer to the OPTION_REG register) or individually (Refer to the appropriate WPUx register)
2. Configure the ADC module:
 - Select ADC conversion clock
 - Configure voltage reference
 - Select ADC input channel
 - Turn on ADC module
3. Configure ADC interrupt (optional):
 - Clear ADC interrupt flag
 - Enable ADC interrupt
 - Enable peripheral interrupt
 - Enable global interrupt⁽¹⁾
4. Wait the required acquisition time⁽²⁾.
5. Start conversion by setting the GO/DONE bit.
6. Wait for ADC conversion to complete by one of the following:
 - Polling the GO/DONE bit
 - Waiting for the ADC interrupt (interrupts enabled)
7. Read ADC Result.
8. Clear the ADC interrupt flag (required if interrupt is enabled).

Note 1: The global interrupt can be disabled if the user is attempting to wake-up from Sleep and resume in-line code execution.

2: Refer to [Section 16.4](#) “[ADC Acquisition Requirements](#)”.

EXAMPLE 16-1: ADC CONVERSION

```
;This code block configures the ADC
;for polling, VDD and VSS references, Frc
;clock and AN0 input.
;
;Conversion start & polling for completion
; are included.
;
BANKSEL    ADCON1    ;
MOVLW     B'11110000' ;Right justify, Frc
;clock
MOVWF     ADCON1    ;Vdd and Vss Vref
BANKSEL    TRISA     ;
BSF       TRISA,0   ;Set RA0 to input
BANKSEL    ANSEL     ;
BSF       ANSEL,0   ;Set RA0 to analog
BANKSEL    WPUA     ;
BCF       WPUA,0    ;Disable weak
;pull-up on RA0
BANKSEL    ADCON0    ;
MOVLW     B'00000001' ;Select channel AN0
MOVWF     ADCON0    ;Turn ADC On
CALL     SampleTime ;Acquisiton delay
BSF      ADCON0,ADGO ;Start conversion
BTFSC    ADCON0,ADGO ;Is conversion done?
GOTO     $-1        ;No, test again
BANKSEL    ADRESH    ;
MOVF     ADRESH,W   ;Read upper 2 bits
MOVWF    RESULTHI   ;store in GPR space
```

16.3 Register Definitions: ADC Control

REGISTER 16-1: ADCON0: ADC CONTROL REGISTER 0

| | | | | | | | |
|-------|----------|---------|---------|---------|---------|---------|---------|
| U-0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 |
| — | CHS<4:0> | | | | | GO/DONE | ADON |
| bit 7 | | | | | | | bit 0 |

Legend:

| | | |
|----------------------|----------------------|---|
| R = Readable bit | W = Writable bit | U = Unimplemented bit, read as '0' |
| u = Bit is unchanged | x = Bit is unknown | -n/n = Value at POR and BOR/Value at all other Resets |
| '1' = Bit is set | '0' = Bit is cleared | |

- bit 7 **Unimplemented:** Read as '0'
- bit 6-2 **CHS<4:0>:** Analog Channel Select bits
 - 11111 = FVR (Fixed Voltage Reference) Buffer 1 Output⁽¹⁾
 - 11110 = Temperature Indicator⁽²⁾.
 - 11101 = AN29
 -
 -
 -
 - 00110 = AN6
 - 00101 = AN5
 - 00100 = AN4
 - 00011 = AN3
 - 00010 = AN2
 - 00001 = AN1
 - 00000 = AN0
- bit 1 **GO/DONE:** ADC Conversion Status bit
 - 1 = ADC conversion cycle in progress. Setting this bit starts an ADC conversion cycle.
This bit is automatically cleared by hardware when the ADC conversion has completed.
 - 0 = ADC conversion completed/not in progress
- bit 0 **ADON:** ADC Enable bit
 - 1 = ADC is enabled
 - 0 = ADC is disabled and consumes no operating current

- Note 1:** See [Section 14.0 “Fixed Voltage Reference \(FVR\)”](#) for more information.
Note 2: See [Section 15.0 “Temperature Indicator Module”](#) for more information.

PIC16(L)F1526/7

REGISTER 16-2: ADCON1: ADC CONTROL REGISTER 1

| | | | | | | | |
|---------|-----------|---------|---------|-----|-----|-------------|---------|
| R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | U-0 | U-0 | R/W-0/0 | R/W-0/0 |
| ADFM | ADCS<2:0> | | | — | — | ADPREF<1:0> | |
| bit 7 | | | | | | | bit 0 |

Legend:

| | | |
|----------------------|----------------------|---|
| R = Readable bit | W = Writable bit | U = Unimplemented bit, read as '0' |
| u = Bit is unchanged | x = Bit is unknown | -n/n = Value at POR and BOR/Value at all other Resets |
| '1' = Bit is set | '0' = Bit is cleared | |

- bit 7 **ADFM:** ADC Result Format Select bit
 1 = Right justified. Six Most Significant bits of ADRESH are set to '0' when the conversion result is loaded.
 0 = Left justified. Six Least Significant bits of ADRESL are set to '0' when the conversion result is loaded.
- bit 6-4 **ADCS<2:0>:** ADC Conversion Clock Select bits
 111 = FRC (clock supplied from a dedicated FRC oscillator)
 110 = FOSC/64
 101 = FOSC/16
 100 = FOSC/4
 011 = FRC (clock supplied from a dedicated FRC oscillator)
 010 = FOSC/32
 001 = FOSC/8
 000 = FOSC/2
- bit 3-2 **Unimplemented:** Read as '0'
- bit 1-0 **ADPREF<1:0>:** ADC Positive Voltage Reference Configuration bits
 11 = VREF+ is connected to internal Fixed Voltage Reference (FVR) module⁽¹⁾
 10 = VREF+ is connected to external VREF+ pin⁽¹⁾
 01 = Reserved
 00 = VREF+ is connected to VDD

Note 1: When selecting the FVR or the VREF+ pin as the source of the positive reference, be aware that a minimum voltage specification exists. See [Section 25.0 "Electrical Specifications"](#) for details.

REGISTER 16-3: ADRESH: ADC RESULT REGISTER HIGH (ADRESH) ADFM = 0

| | | | | | | | |
|------------|---------|---------|---------|---------|---------|---------|---------|
| R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u |
| ADRES<9:2> | | | | | | | |
| bit 7 | | | | | | | bit 0 |

Legend:

| | | |
|----------------------|----------------------|---|
| R = Readable bit | W = Writable bit | U = Unimplemented bit, read as '0' |
| u = Bit is unchanged | x = Bit is unknown | -n/n = Value at POR and BOR/Value at all other Resets |
| '1' = Bit is set | '0' = Bit is cleared | |

bit 7-0 **ADRES<9:2>**: ADC Result Register bits
Upper 8 bits of 10-bit conversion result

REGISTER 16-4: ADRESL: ADC RESULT REGISTER LOW (ADRESL) ADFM = 0

| | | | | | | | |
|------------|---------|---------|---------|---------|---------|---------|---------|
| R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u |
| ADRES<1:0> | | — | — | — | — | — | — |
| bit 7 | | | | | | | bit 0 |

Legend:

| | | |
|----------------------|----------------------|---|
| R = Readable bit | W = Writable bit | U = Unimplemented bit, read as '0' |
| u = Bit is unchanged | x = Bit is unknown | -n/n = Value at POR and BOR/Value at all other Resets |
| '1' = Bit is set | '0' = Bit is cleared | |

bit 7-6 **ADRES<1:0>**: ADC Result Register bits
Lower 2 bits of 10-bit conversion result

bit 5-0 **Reserved**: Do not use.

PIC16(L)F1526/7

REGISTER 16-5: ADRESH: ADC RESULT REGISTER HIGH (ADRESH) ADFM = 1

| | | | | | | | |
|---------|---------|---------|---------|---------|---------|------------|---------|
| R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u |
| — | — | — | — | — | — | ADRES<9:8> | |
| bit 7 | | | | | | | bit 0 |

Legend:

| | | |
|----------------------|----------------------|---|
| R = Readable bit | W = Writable bit | U = Unimplemented bit, read as '0' |
| u = Bit is unchanged | x = Bit is unknown | -n/n = Value at POR and BOR/Value at all other Resets |
| '1' = Bit is set | '0' = Bit is cleared | |

bit 7-2 **Reserved:** Do not use.
bit 1-0 **ADRES<9:8>**: ADC Result Register bits
Upper 2 bits of 10-bit conversion result

REGISTER 16-6: ADRESL: ADC RESULT REGISTER LOW (ADRESL) ADFM = 1

| | | | | | | | |
|------------|---------|---------|---------|---------|---------|---------|---------|
| R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u |
| ADRES<7:0> | | | | | | | |
| bit 7 | | | | | | | bit 0 |

Legend:

| | | |
|----------------------|----------------------|---|
| R = Readable bit | W = Writable bit | U = Unimplemented bit, read as '0' |
| u = Bit is unchanged | x = Bit is unknown | -n/n = Value at POR and BOR/Value at all other Resets |
| '1' = Bit is set | '0' = Bit is cleared | |

bit 7-0 **ADRES<7:0>**: ADC Result Register bits
Lower 8 bits of 10-bit conversion result

16.4 ADC Acquisition Requirements

For the ADC to meet its specified accuracy, the charge holding capacitor (CHOLD) must be allowed to fully charge to the input channel voltage level. The Analog Input model is shown in Figure 16-4. The source impedance (RS) and the internal sampling switch (RSS) impedance directly affect the time required to charge the capacitor CHOLD. The sampling switch (RSS) impedance varies over the device voltage (VDD), refer to Figure 16-4. **The maximum recommended impedance for analog sources is 10 kΩ.** As the

source impedance is decreased, the acquisition time may be decreased. After the analog input channel is selected (or changed), an ADC acquisition must be done before the conversion can be started. To calculate the minimum acquisition time, Equation 16-1 may be used. This equation assumes that 1/2 LSB error is used (1,024 steps for the ADC). The 1/2 LSB error is the maximum error allowed for the ADC to meet its specified resolution.

EQUATION 16-1: ACQUISITION TIME EXAMPLE

Assumptions: Temperature = 50°C and external impedance of 10kΩ 5.0V VDD

$$\begin{aligned} T_{ACQ} &= \text{Amplifier Settling Time} + \text{Hold Capacitor Charging Time} + \text{Temperature Coefficient} \\ &= T_{AMP} + T_C + T_{COFF} \\ &= 2\mu s + T_C + [(Temperature - 25^\circ C)(0.05\mu s/^\circ C)] \end{aligned}$$

The value for TC can be approximated with the following equations:

$$V_{APPLIED} \left(1 - \frac{1}{(2^{n+1}) - 1} \right) = V_{CHOLD} \quad ;[1] \text{ } V_{CHOLD} \text{ charged to within } 1/2 \text{ lsb}$$

$$V_{APPLIED} \left(1 - e^{-\frac{T_C}{RC}} \right) = V_{CHOLD} \quad ;[2] \text{ } V_{CHOLD} \text{ charge response to } V_{APPLIED}$$

$$V_{APPLIED} \left(1 - e^{-\frac{T_C}{RC}} \right) = V_{APPLIED} \left(1 - \frac{1}{(2^{n+1}) - 1} \right) \quad ;\text{combining [1] and [2]}$$

Note: Where n = number of bits of the ADC.

Solving for TC:

$$\begin{aligned} T_C &= -CHOLD(RIC + RSS + RS) \ln(1/2047) \\ &= -10pF(1k\Omega + 7k\Omega + 10k\Omega) \ln(0.000488) \\ &= 1.37\mu s \end{aligned}$$

Therefore:

$$\begin{aligned} T_{ACQ} &= 2\mu s + 1.37\mu s + [(50^\circ C - 25^\circ C)(0.05\mu s/^\circ C)] \\ &= 4.62\mu s \end{aligned}$$

Note 1: The reference voltage (VREF) has no effect on the equation, since it cancels itself out.

2: The charge holding capacitor (CHOLD) is not discharged after each conversion.

3: The maximum recommended impedance for analog sources is 10 kΩ. This is required to meet the pin leakage specification.

PIC16(L)F1526/7

FIGURE 16-4: ANALOG INPUT MODEL

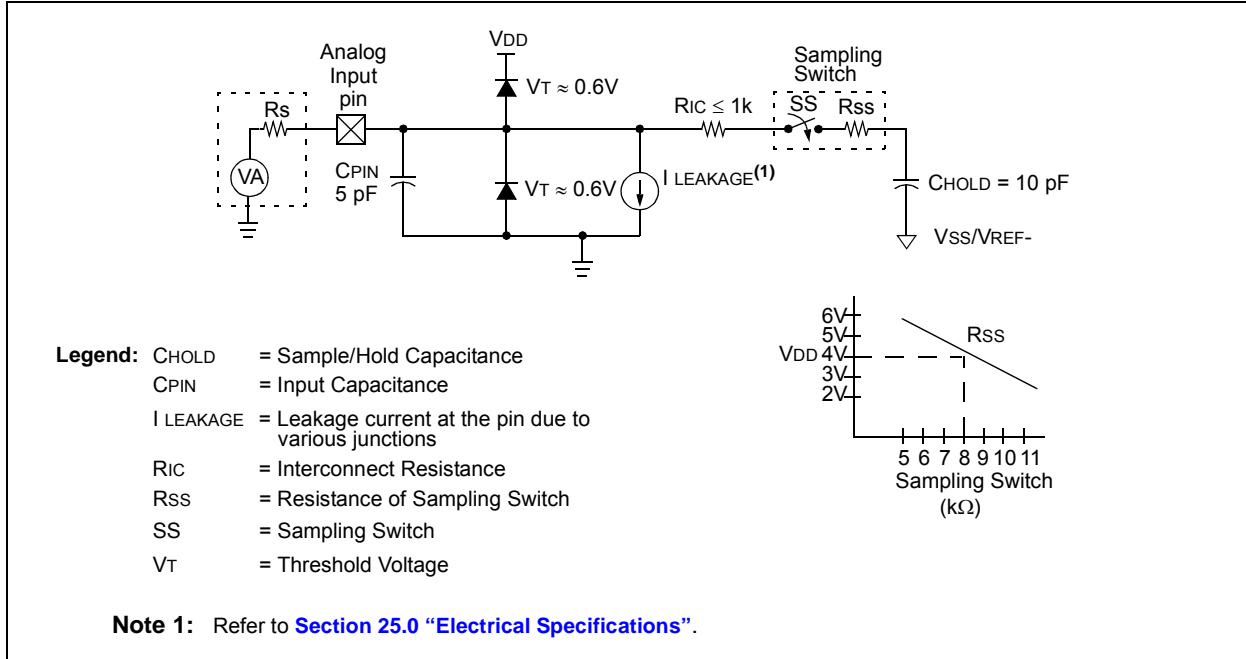


FIGURE 16-5: ADC TRANSFER FUNCTION

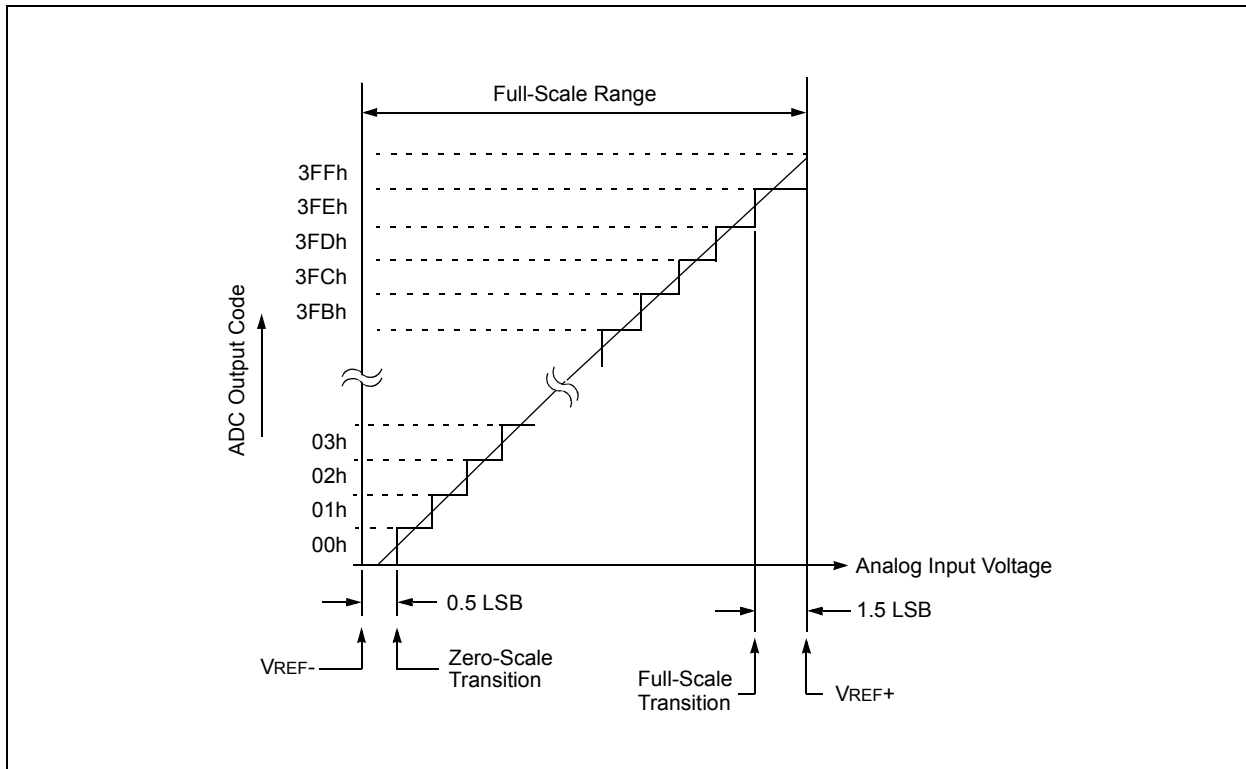


TABLE 16-3: SUMMARY OF REGISTERS ASSOCIATED WITH ADC

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Register on Page |
|----------|--------------------------|-----------|------------------|--------|-------------|--------|-------------|--------|------------------|
| ADCON0 | — | CHS<4:0> | | | | — | GO/DONE | ADON | 149 |
| ADCON1 | ADFM | ADCS<2:0> | | | — | — | ADPREF<1:0> | | 150 |
| ADRESH | ADC Result Register High | | | | | | | | 151, 152 |
| ADRESL | ADC Result Register Low | | | | | | | | 151, 152 |
| ANSELA | — | — | ANSA5 | — | ANSA3 | ANSA2 | ANSA1 | ANSA0 | 115 |
| ANSELB | — | — | ANSB5 | ANSB4 | ANSB3 | ANSB2 | ANSB1 | ANSB0 | 118 |
| ANSELD | — | — | — | — | ANSD3 | ANSD2 | ANSD1 | ANSD0 | 124 |
| ANSELE | — | — | — | — | — | ANSE2 | ANSE1 | ANSE0 | 127 |
| ANSELF | ANSF7 | ANSF6 | ANSF5 | ANSF4 | ANSF3 | ANSF2 | ANSF1 | ANSF0 | 130 |
| ANSELG | — | — | — | ANSG4 | ANSG3 | ANSG2 | ANSG1 | — | 133 |
| CCP1CON | — | — | DC1B<1:0> | | CCP1M<3:0> | | | | 189 |
| CCP2CON | — | — | DC2B<1:0> | | CCP2M<3:0> | | | | 189 |
| CCP3CON | — | — | DC3B<1:0> | | CCP3M<3:0> | | | | 189 |
| CCP4CON | — | — | DC4B<1:0> | | CCP4M<3:0> | | | | 189 |
| CCP5CON | — | — | DC5B<1:0> | | CCP5M<3:0> | | | | 189 |
| CCP6CON | — | — | DC6B<1:0> | | CCP6M<3:0> | | | | 189 |
| CCP7CON | — | — | DC7B<1:0> | | CCP7M<3:0> | | | | 189 |
| CCP8CON | — | — | DC8B<1:0> | | CCP8M<3:0> | | | | 189 |
| CCP9CON | — | — | DC9B<1:0> | | CCP9M<3:0> | | | | 189 |
| CCP10CON | — | — | DC10B<1:0> | | CCP10M<3:0> | | | | 189 |
| FVRCON | FVREN | FVRRDY | TSEN | TSRNG | CDAFVR<1:0> | | ADFVR<1:0> | | 140 |
| INTCON | GIE | PEIE | TMR0IE | INTE | IOCIE | TMR0IF | INTF | IOCIF | 76 |
| PIE1 | TMR1GIE | ADIE | RC1IE | TX1IE | SSP1IE | CCP1IE | TMR2IE | TMR1IE | 77 |
| PIR1 | TMR1GIF | ADIF | RC1IF | TX1IF | SSP1IF | CCP1IF | TMR2IF | TMR1IF | 81 |
| TRISA | TRISA7 | TRISA6 | TRISA5 | TRISA4 | TRISA3 | TRISA2 | TRISA1 | TRISA0 | 114 |
| TRISB | TRISB7 | TRISB6 | TRISB5 | TRISB4 | TRISB3 | TRISB2 | TRISB1 | TRISB0 | 117 |
| TRISD | TRISD7 | TRISD6 | TRISD5 | TRISD4 | TRISD3 | TRISD2 | TRISD1 | TRISD0 | 123 |
| TRISE | TRISE7 | TRISE6 | TRISE5 | TRISE4 | TRISE3 | TRISE2 | TRISE1 | TRISE0 | 126 |
| TRISF | TRISF7 | TRISF6 | TRISF5 | TRISF4 | TRISF3 | TRISF2 | TRISF1 | TRISF0 | 129 |
| TRISG | — | — | — ⁽¹⁾ | TRISG4 | TRISG3 | TRISG2 | TRISG1 | TRISG0 | 132 |

Legend: — = unimplemented read as '0'. Shaded cells are not used for ADC module.

Note 1: Unimplemented, read as '1'.

PIC16(L)F1526/7

17.0 TIMER0 MODULE

The Timer0 module is an 8-bit timer/counter with the following features:

- 8-bit timer/counter register (TMR0)
- 8-bit prescaler (independent of Watchdog Timer)
- Programmable internal or external clock source
- Programmable external clock edge selection
- Interrupt on overflow
- TMR0 can be used to gate Timer1/3/5

Figure 17-1 is a block diagram of the Timer0 module.

17.1 Timer0 Operation

The Timer0 module can be used as either an 8-bit timer or an 8-bit counter.

17.1.1 8-BIT TIMER MODE

The Timer0 module will increment every instruction cycle, if used without a prescaler. 8-bit Timer mode is selected by clearing the TMR0CS bit of the OPTION_REG register.

When TMR0 is written, the increment is inhibited for two instruction cycles immediately following the write.

Note: The value written to the TMR0 register can be adjusted, in order to account for the two instruction cycle delay when TMR0 is written.

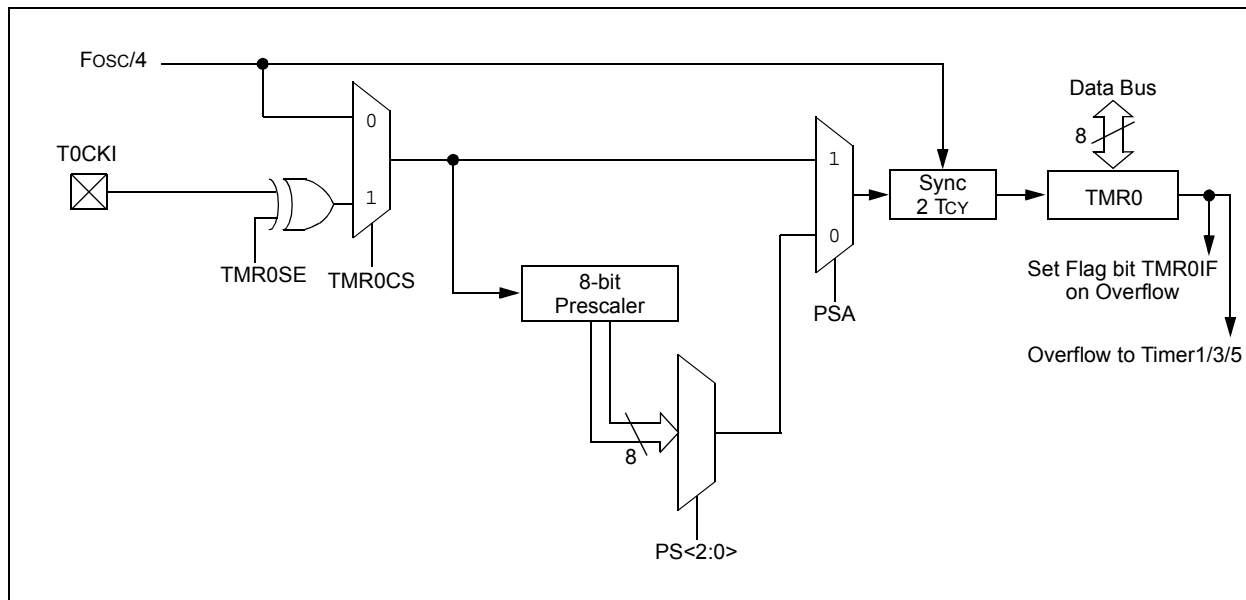
17.1.2 8-BIT COUNTER MODE

In 8-Bit Counter mode, the Timer0 module will increment on either the rising or falling edge of the T0CKI pin.

The 8-bit Counter mode using the T0CKI pin is selected by setting the TMR0CS bit in the OPTION_REG register to '1'.

The rising or falling transition of the incrementing edge for either input source is determined by the TMR0SE bit in the OPTION_REG register.

FIGURE 17-1: BLOCK DIAGRAM OF THE TIMER0



17.1.3 SOFTWARE PROGRAMMABLE PRESCALER

A software programmable prescaler is available for exclusive use with Timer0. The prescaler is enabled by clearing the PSA bit of the OPTION_REG register.

Note: The Watchdog Timer (WDT) uses its own independent prescaler.

There are 8 prescaler options for the Timer0 module ranging from 1:2 to 1:256. The prescale values are selectable via the PS<2:0> bits of the OPTION_REG register. In order to have a 1:1 prescaler value for the Timer0 module, the prescaler must be disabled by setting the PSA bit of the OPTION_REG register.

The prescaler is not readable or writable. All instructions writing to the TMR0 register will clear the prescaler.

17.1.4 TIMER0 INTERRUPT

Timer0 will generate an interrupt when the TMR0 register overflows from FFh to 00h. The TMR0IF interrupt flag bit of the INTCON register is set every time the TMR0 register overflows, regardless of whether or not the Timer0 interrupt is enabled. The TMR0IF bit can only be cleared in software. The Timer0 interrupt enable is the TMR0IE bit of the INTCON register.

Note: The Timer0 interrupt cannot wake the processor from Sleep since the timer is frozen during Sleep.

17.1.5 8-BIT COUNTER MODE SYNCHRONIZATION

When in 8-bit Counter mode, the incrementing edge on the T0CKI pin must be synchronized to the instruction clock. Synchronization can be accomplished by sampling the prescaler output on the Q2 and Q4 cycles of the instruction clock. The high and low periods of the external clocking source must meet the timing requirements as shown in [Section 25.0 “Electrical Specifications”](#).

17.1.6 OPERATION DURING SLEEP

Timer0 cannot operate while the processor is in Sleep mode. The contents of the TMR0 register will remain unchanged while the processor is in Sleep mode.

PIC16(L)F1526/7

17.2 Register Definitions: Option Register

REGISTER 17-1: OPTION_REG: OPTION REGISTER

| | | | | | | | |
|---------------------------|---------|---------|---------|---------|---------|---------|---------|
| R/W-1/1 | R/W-1/1 | R/W-1/1 | R/W-1/1 | R/W-1/1 | R/W-1/1 | R/W-1/1 | R/W-1/1 |
| $\overline{\text{WPUEN}}$ | INTEDG | TMR0CS | TMR0SE | PSA | PS<2:0> | | |
| bit 7 | | | | | | | bit 0 |

Legend:

R = Readable bit W = Writable bit U = Unimplemented bit, read as '0'
u = Bit is unchanged x = Bit is unknown -n/n = Value at POR and BOR/Value at all other Resets
'1' = Bit is set '0' = Bit is cleared

- bit 7 **$\overline{\text{WPUEN}}$** : Weak Pull-Up Enable bit
1 = All weak pull-ups are disabled (except $\overline{\text{MCLR}}$, if it is enabled)
0 = Weak pull-ups are enabled by individual WPUA latch values
- bit 6 **INTEDG**: Interrupt Edge Select bit
1 = Interrupt on rising edge of INT pin
0 = Interrupt on falling edge of INT pin
- bit 5 **TMR0CS**: Timer0 Clock Source Select bit
1 = Transition on T0CKI pin
0 = Internal instruction cycle clock (FOSC/4)
- bit 4 **TMR0SE**: Timer0 Source Edge Select bit
1 = Increment on high-to-low transition on T0CKI pin
0 = Increment on low-to-high transition on T0CKI pin
- bit 3 **PSA**: Prescaler Assignment bit
1 = Prescaler is not assigned to the Timer0 module
0 = Prescaler is assigned to the Timer0 module
- bit 2-0 **PS<2:0>**: Prescaler Rate Select bits

| Bit Value | Timer0 Rate |
|-----------|-------------|
| 000 | 1 : 2 |
| 001 | 1 : 4 |
| 010 | 1 : 8 |
| 011 | 1 : 16 |
| 100 | 1 : 32 |
| 101 | 1 : 64 |
| 110 | 1 : 128 |
| 111 | 1 : 256 |

TABLE 17-1: SUMMARY OF REGISTERS ASSOCIATED WITH TIMER0

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Register on Page |
|------------|---------------------------|--------|--------|--------|--------|---------|--------|--------|------------------|
| INTCON | GIE | PEIE | TMR0IE | INTE | IOCIE | TMR0IF | INTF | IOCIF | 76 |
| OPTION_REG | $\overline{\text{WPUEN}}$ | INTEDG | TMR0CS | TMR0SE | PSA | PS<2:0> | | | 158 |
| TMR0 | Timer0 Module Register | | | | | | | | 156* |
| TRISA | TRISA7 | TRISA6 | TRISA5 | TRISA4 | TRISA3 | TRISA2 | TRISA1 | TRISA0 | 114 |

Legend: — = Unimplemented locations, read as '0'. Shaded cells are not used by the Timer0 module.

* Page provides register information.

18.0 TIMER1/3/5 MODULE WITH GATE CONTROL

The Timer1/3/5 module is a 16-bit timer/counter with the following features:

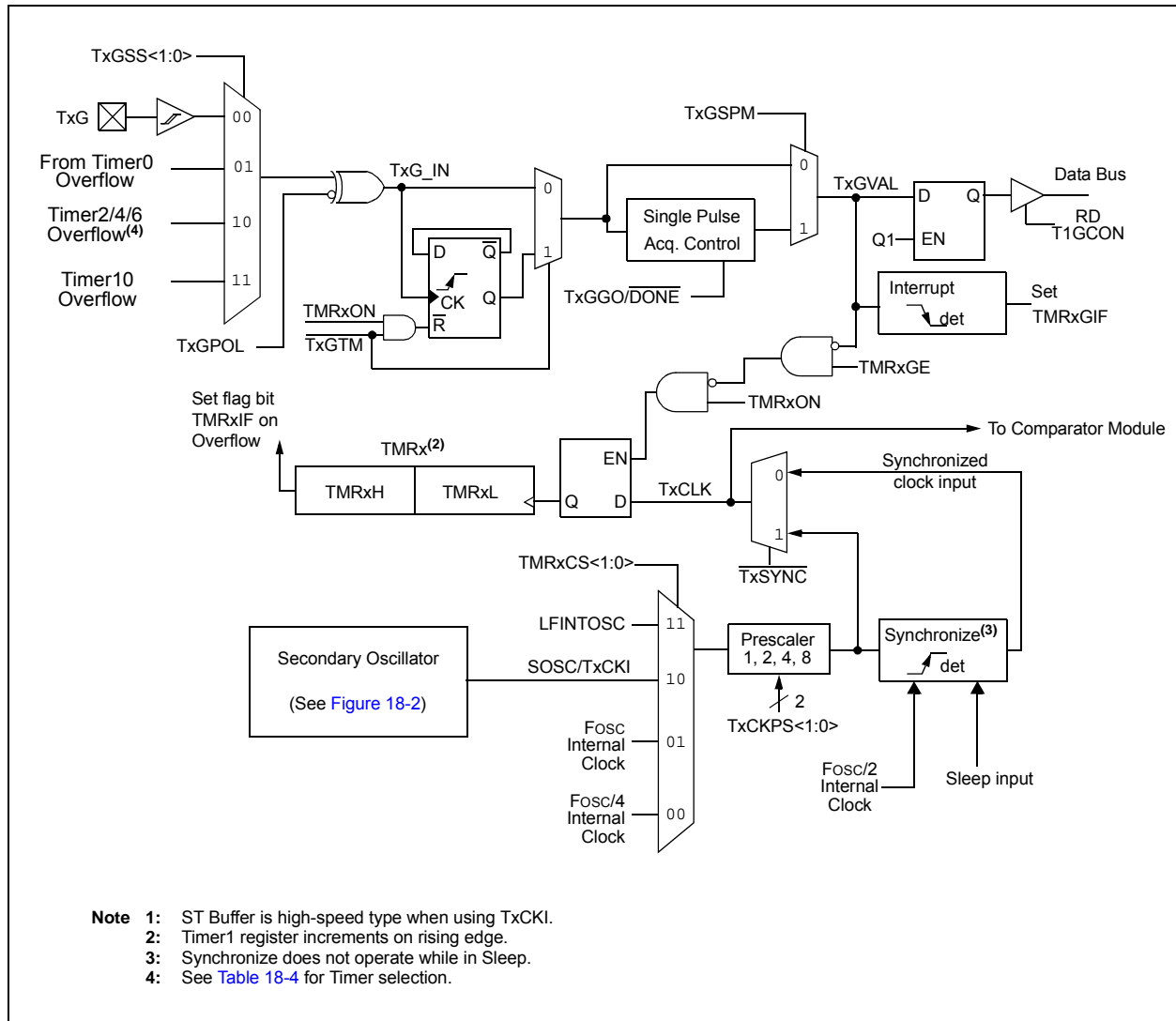
- 16-bit timer/counter register pair (TMRxH:TMRxL)
- Programmable internal or external clock source
- 2-bit prescaler
- Dedicated 32 kHz oscillator circuit
- Optionally synchronized comparator out
- Multiple Timer1/3/5 gate (count enable) sources
- Interrupt on overflow
- Wake-up on overflow (external clock, Asynchronous mode only)
- Time base for the Capture/Compare function
- Auto-conversion Trigger (with CCP)
- Selectable Gate Source Polarity

- Gate Toggle mode
- Gate Single-pulse mode
- Gate Value Status
- Gate Event Interrupt

Figure 18-1 is a block diagram of the Timer1/3/5 module.

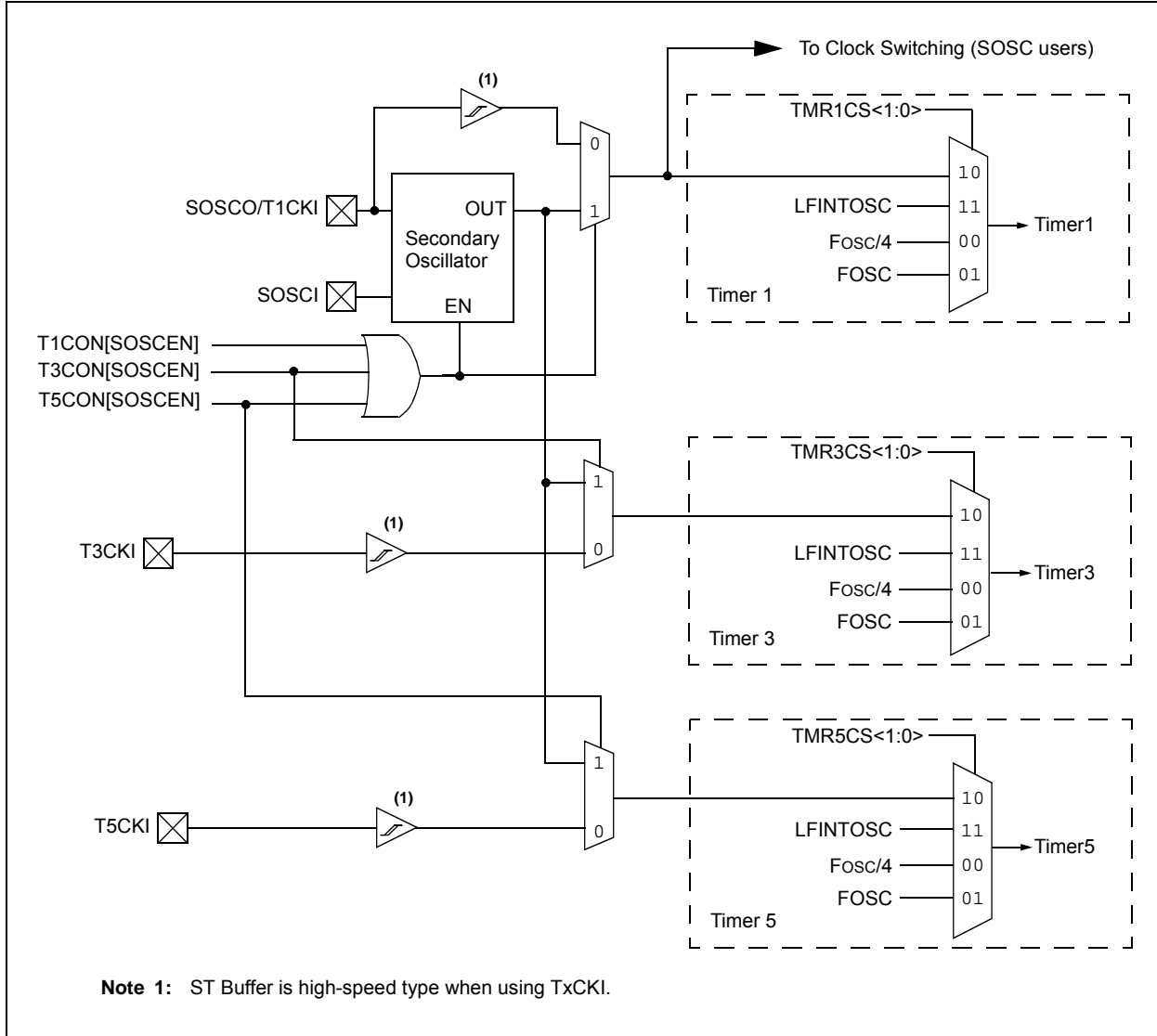
Note: The 'x' variable used in this section is used to designate Timer1, Timer3 or Timer5. For example, TxCON references T1CON, T3CON or T5CON. PRx references PR1, PR3 or PR5.

FIGURE 18-1: TIMER1/3/5 BLOCK DIAGRAM



PIC16(L)F1526/7

FIGURE 18-2: TIMER1/3/5 CLOCK SOURCE DIAGRAM



18.1 Timer1/3/5 Operation

The Timer1/3/5 module is a 16-bit incrementing counter which is accessed through the TMRxH:TMRxL register pair. Writes to TMRxH or TMRxL directly update the counter.

When used with an internal clock source, the module is a timer and increments on every instruction cycle. When used with an external clock source, the module can be used as either a timer or counter and increments on every selected edge of the external source.

Timer1/3/5 is enabled by configuring the TMRxON and TMRxGE bits in the TxCON and TxGCON registers, respectively. Table 18-1 displays the Timer1/3/5 enable selections.

TABLE 18-1: TIMER1/3/5 ENABLE SELECTIONS

| TMRxON | TMRxGE | Timer1/3/5 Operation |
|--------|--------|----------------------|
| 0 | 0 | Off |
| 0 | 1 | Off |
| 1 | 0 | Always On |
| 1 | 1 | Count Enabled |

18.2 Clock Source Selection

The TMRxCS<1:0> and SOSSEN bits of the TxCON register are used to select the clock source for Timer1/3/5. Table 18-2 displays the clock source selections.

18.2.1 INTERNAL CLOCK SOURCE

When the internal clock source is selected, the TMRxH:TMRxL register pair will increment on multiples of FOSC as determined by the Timer1/3/5 prescaler.

When the FOSC internal clock source is selected, the Timer1/3/5 register value will increment by four counts every instruction clock cycle. Due to this condition, a 2 LSB error in resolution will occur when reading the Timer1/3/5 value. To utilize the full resolution of Timer1/3/5, an asynchronous input signal must be used to gate the Timer1/3/5 clock input.

The following asynchronous sources may be used:

- Asynchronous event on the TxG pin to Timer1/3/5 gate

18.2.2 EXTERNAL CLOCK SOURCE

When the external clock source is selected, the Timer1/3/5 module may work as a timer or a counter.

When enabled to count, Timer1/3/5 is incremented on the rising edge of the external clock input TxCKI. These external clock inputs (TxCKI) can be synchronized to the microcontroller system clock or they can run asynchronously.

When used as a timer with a clock oscillator, an external 32.768 kHz crystal can be used in conjunction with the dedicated internal oscillator circuit.

Note: In Counter mode, a falling edge must be registered by the counter prior to the first incrementing rising edge after any one or more of the following conditions:

- Timer1/3/5 enabled after POR
- Write to TMRxH or TMRxL
- Timer1/3/5 is disabled
- Timer1/3/5 is disabled (TMRxON = 0) when TxCKI is high then Timer1/3/5 is enabled (TMRxON = 1) when TxCKI is low.

TABLE 18-2: CLOCK SOURCE SELECTIONS

| TMRxCS<1:0> | SOSSEN | Clock Source |
|-------------|--------|--|
| 00 | x | Instruction Clock (FOSC/4) |
| 01 | x | System Clock (FOSC) |
| 10 | 0 | External Clocking on TxCKI Pin |
| | 1 | Secondary Oscillator Circuit on SOSCI/SOSCO Pins |
| 11 | x | LFINTOSC |

PIC16(L)F1526/7

18.3 Timer1/3/5 Prescaler

Timer1/3/5 has four prescaler options allowing 1, 2, 4 or 8 divisions of the clock input. The TxCKPS bits of the TxCON register control the prescale counter. The prescale counter is not directly readable or writable; however, the prescaler counter is cleared upon a write to TMRxH or TMRxL.

18.4 Timer1/3/5 Oscillator

A dedicated low-power 32.768 kHz oscillator circuit is built-in between pins SOSC1 (input) and SOSCO (amplifier output). This internal circuit is to be used in conjunction with an external 32.768 kHz crystal.

The oscillator circuit is enabled by setting the SOSSEN bit of the TxCON register. The oscillator will continue to run during Sleep.

Note: The oscillator requires a start-up and stabilization time before use. Thus, SOSSEN should be set and a suitable delay observed prior to enabling Timer1/3/5.

18.5 Timer1/3/5 Operation in Asynchronous Counter Mode

If control bit $\overline{\text{TxSYNC}}$ of the TxCON register is set, the external clock input is not synchronized. The timer increments asynchronously to the internal phase clocks. If the external clock source is selected then the timer will continue to run during Sleep and can generate an interrupt on overflow, which will wake-up the processor. However, special precautions in software are needed to read/write the timer (see [Section 18.5.1 “Reading and Writing Timer1/3/5 in Asynchronous Counter Mode”](#)).

Note: When switching from synchronous to asynchronous operation, it is possible to skip an increment. When switching from asynchronous to synchronous operation, it is possible to produce an additional increment.

18.5.1 READING AND WRITING TIMER1/3/5 IN ASYNCHRONOUS COUNTER MODE

Reading TMRxH or TMRxL while the timer is running from an external asynchronous clock will ensure a valid read (taken care of in hardware). However, the user should keep in mind that reading the 16-bit timer in two 8-bit values itself, poses certain problems, since the timer may overflow between the reads.

For writes, it is recommended that the user simply stop the timer and write the desired values. A write contention may occur by writing to the timer registers, while the register is incrementing. This may produce an unpredictable value in the TMRxH:TMRxL register pair.

18.6 Timer1/3/5 Gate

Timer1/3/5 can be configured to count freely or the count can be enabled and disabled using Timer1/3/5 gate circuitry. This is also referred to as Timer1/3/5 Gate Enable.

Timer1/3/5 gate can also be driven by multiple selectable sources.

18.6.1 TIMER1/3/5 GATE ENABLE

The Timer1/3/5 Gate Enable mode is enabled by setting the TMRxGE bit of the TxGCON register. The polarity of the Timer1/3/5 Gate Enable mode is configured using the TxGPOL bit of the TxGCON register.

When Timer1/3/5 Gate Enable mode is enabled, Timer1/3/5 will increment on the rising edge of the Timer1/3/5 clock source. When Timer1/3/5 Gate Enable mode is disabled, no incrementing will occur and Timer1/3/5 will hold the current count. See [Figure 18-4](#) for timing details.

TABLE 18-3: TIMER1/3/5 GATE ENABLE SELECTIONS

| TxCLK | TxGPOL | TxG | Timer1/3/5 Operation |
|-------|--------|-----|----------------------|
| ↑ | 0 | 0 | Counts |
| ↑ | 0 | 1 | Holds Count |
| ↑ | 1 | 0 | Holds Count |
| ↑ | 1 | 1 | Counts |

18.6.2 TIMER1/3/5 GATE SOURCE SELECTION

The Timer1/3/5 gate source can be selected from one of four different sources. Source selection is controlled by the TxGSS bits of the TxGCON register. The polarity for each available source is also selectable. Polarity selection is controlled by the TxGPOL bit of the TxGCON register.

TABLE 18-4: TIMER1/3/5 GATE SOURCES

| T1GSS | Timer1 Gate Source | Timer3 Gate Source | Timer5 Gate Source |
|-------|---|--------------------|--------------------|
| 00 | T1G Pin | T3G Pin | T5G Pin |
| 01 | Overflow of Timer0 (TMR0 increments from FFh to 00h) | | |
| 10 | Timer2 match PR2 (TMR2 increments to match PR2) | Timer4 match PR4 | Timer6 match PR6 |
| 11 | Timer10 match PR10 | | |

18.6.2.1 TxG Pin Gate Operation

The TxG pin is one source for Timer1/3/5 gate control. It can be used to supply an external source to the Timer1/3/5 gate circuitry.

18.6.2.2 Timer0 Overflow Gate Operation

When Timer0 increments from FFh to 00h, a low-to-high pulse will automatically be generated and internally supplied to the Timer1/3/5 gate circuitry.

18.6.3 TIMER1/3/5 GATE TOGGLE MODE

When Timer1/3/5 Gate Toggle mode is enabled, it is possible to measure the full-cycle length of a Timer1/3/5 gate signal, as opposed to the duration of a single level pulse.

The Timer1/3/5 gate source is routed through a flip-flop that changes state on every incrementing edge of the signal. See [Figure 18-5](#) for timing details.

Timer1/3/5 Gate Toggle mode is enabled by setting the TxGTM bit of the TxGCON register. When the TxGTM bit is cleared, the flip-flop is cleared and held clear. This is necessary in order to control which edge is measured.

Note: Enabling Toggle mode at the same time as changing the gate polarity may result in indeterminate operation.

18.6.4 TIMER1/3/5 GATE SINGLE-PULSE MODE

When Timer1/3/5 Gate Single-Pulse mode is enabled, it is possible to capture a single-pulse gate event. Timer1/3/5 Gate Single-Pulse mode is first enabled by setting the TxGSPM bit in the TxGCON register. Next, the TxGGO/DONE bit in the TxGCON register must be set. The Timer1/3/5 will be fully enabled on the next incrementing edge. On the next trailing edge of the pulse, the TxGGO/DONE bit will automatically be cleared. No other gate events will be allowed to increment Timer1/3/5 until the TxGGO/DONE bit is once again set in software. See [Figure 18-6](#) for timing details.

If the Single-Pulse Gate mode is disabled by clearing the TxGSPM bit in the TxGCON register, the TxGGO/DONE bit should also be cleared.

Enabling the Toggle mode and the Single-Pulse mode simultaneously will permit both sections to work together. This allows the cycle times on the Timer1/3/5 gate source to be measured. See [Figure 18-7](#) for timing details.

18.6.5 TIMER1/3/5 GATE VALUE STATUS

When Timer1/3/5 Gate Value Status is utilized, it is possible to read the most current level of the gate control value. The value is stored in the TxGVAL bit in the TxGCON register. The TxGVAL bit is valid even when the Timer1/3/5 gate is not enabled (TMRxGE bit is cleared).

PIC16(L)F1526/7

18.6.6 TIMER1/3/5 GATE EVENT INTERRUPT

When Timer1/3/5 Gate Event Interrupt is enabled, it is possible to generate an interrupt upon the completion of a gate event. When the falling edge of TxGVAL occurs, the TMRxGIF flag bit in the PIR1 register will be set. If the TMRxGIE bit in the PIE1 register is set, then an interrupt will be recognized.

The TMRxGIF flag bit operates even when the Timer1/3/5 gate is not enabled (TMRxGE bit is cleared).

18.7 Timer1/3/5 Interrupt

The Timer1/3/5 register pair (TMRxH:TMRxL) increments to FFFFh and rolls over to 0000h. When Timer1/3/5 rolls over, the Timer1/3/5 interrupt flag bit of the PIR1 register is set. To enable the interrupt on rollover, you must set these bits:

- TMRxON bit of the TxCON register
- TMRxIE bit of the PIE1 register
- PEIE bit of the INTCON register
- GIE bit of the INTCON register

The interrupt is cleared by clearing the TMRxIF bit in the Interrupt Service Routine.

| |
|---|
| Note: The TMRxH:TMRxL register pair and the TMRxIF bit should be cleared before enabling interrupts. |
|---|

18.8 Timer1/3/5 Operation During Sleep

Timer1/3/5 can only operate during Sleep when setup in Asynchronous Counter mode. In this mode, an external crystal or clock source can be used to increment the counter. To set up the timer to wake the device:

- TMRxON bit of the TxCON register must be set
- TMRxIE bit of the PIE1 register must be set
- PEIE bit of the INTCON register must be set
- TxSYNC bit of the TxCON register must be set
- TMRxCS bits of the TxCON register must be configured
- SOSSEN bit of the TxCON register must be configured

The device will wake-up on an overflow and execute the next instructions. If the GIE bit of the INTCON register is set, the device will call the Interrupt Service Routine.

Timer1/3/5 oscillator will continue to operate in Sleep regardless of the TxSYNC bit setting.

18.9 ECCP/CCP Capture/Compare Time Base

The CCP module uses the TMRxH:TMRxL register pair as the time base when operating in Capture or Compare mode.

In Capture mode, the value in the TMRxH:TMRxL register pair is copied into the CCPR1H:CCPR1L register pair on a configured event.

In Compare mode, an event is triggered when the value CCPR1H:CCPR1L register pair matches the value in the TMRxH:TMRxL register pair. This event can be a Special Event Trigger.

For more information, see [Section 20.0 “Capture/Compare/PWM Modules”](#).

18.10 ECCP/CCP Special Event Trigger

When the CCP is configured to trigger a special event, the trigger will clear the TMRxH:TMRxL register pair. This special event does not cause a Timer1/3/5 interrupt. The CCP module may still be configured to generate a CCP interrupt.

In this mode of operation, the CCPR1H:CCPR1L register pair becomes the period register for Timer1/3/5.

Timer1/3/5 should be synchronized and Fosc/4 should be selected as the clock source in order to utilize the Special Event Trigger. Asynchronous operation of Timer1/3/5 can cause a Special Event Trigger to be missed.

In the event that a write to TMRxH or TMRxL coincides with a Special Event Trigger from the CCP, the write will take precedence.

For more information, see [Section 16.2.5 “Special Event Trigger”](#).

FIGURE 18-3: TIMER1/3/5 INCREMENTING EDGE

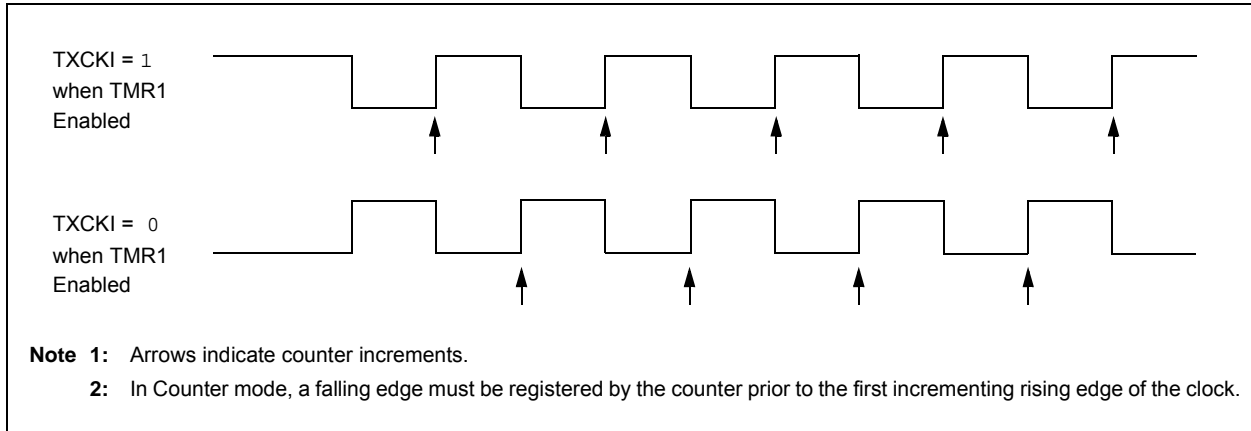
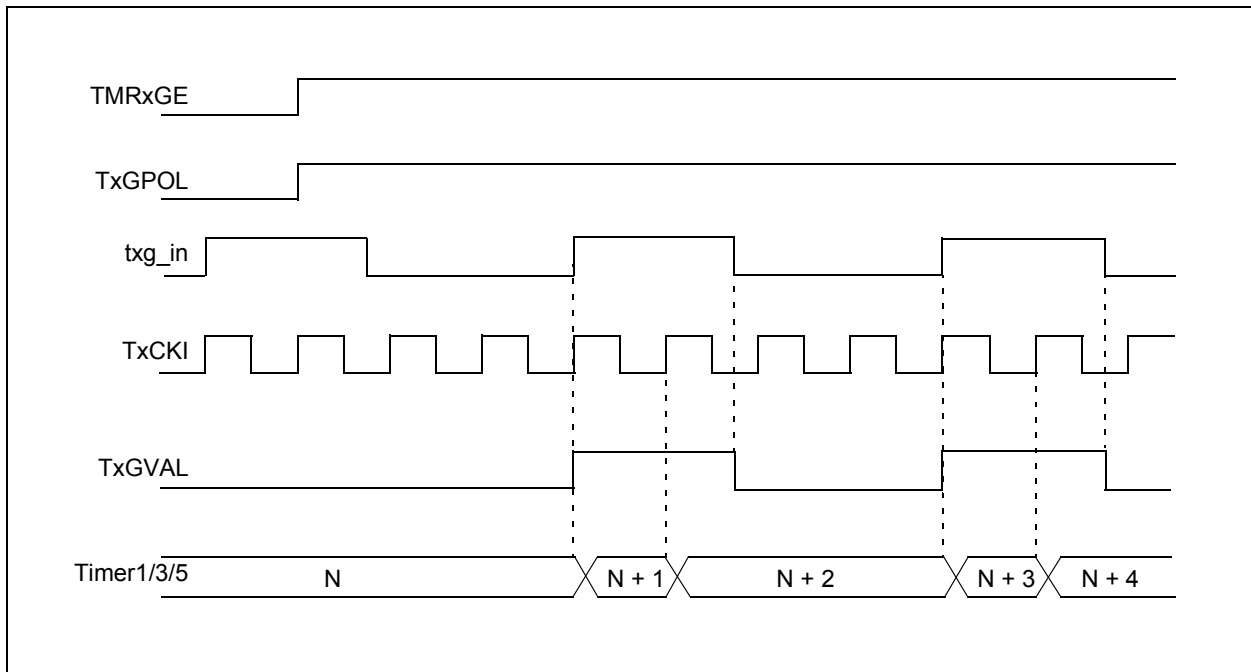


FIGURE 18-4: TIMER1/3/5 GATE ENABLE MODE



PIC16(L)F1526/7

FIGURE 18-5: TIMER1/3/5 GATE TOGGLE MODE

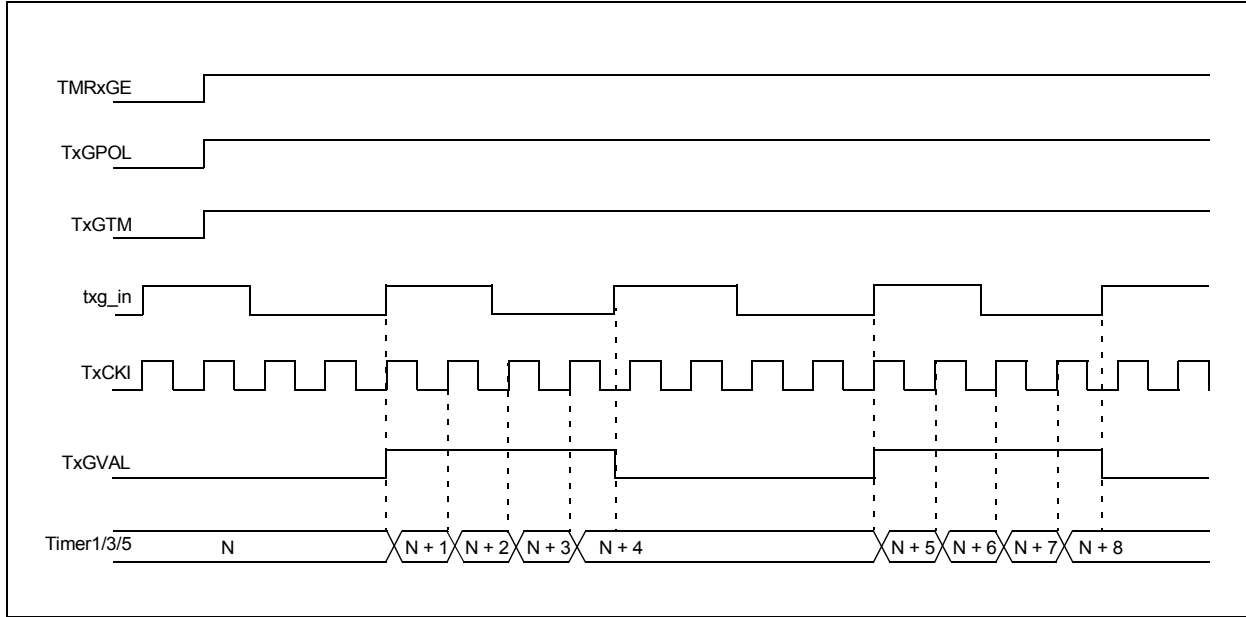


FIGURE 18-6: TIMER1/3/5 GATE SINGLE-PULSE MODE

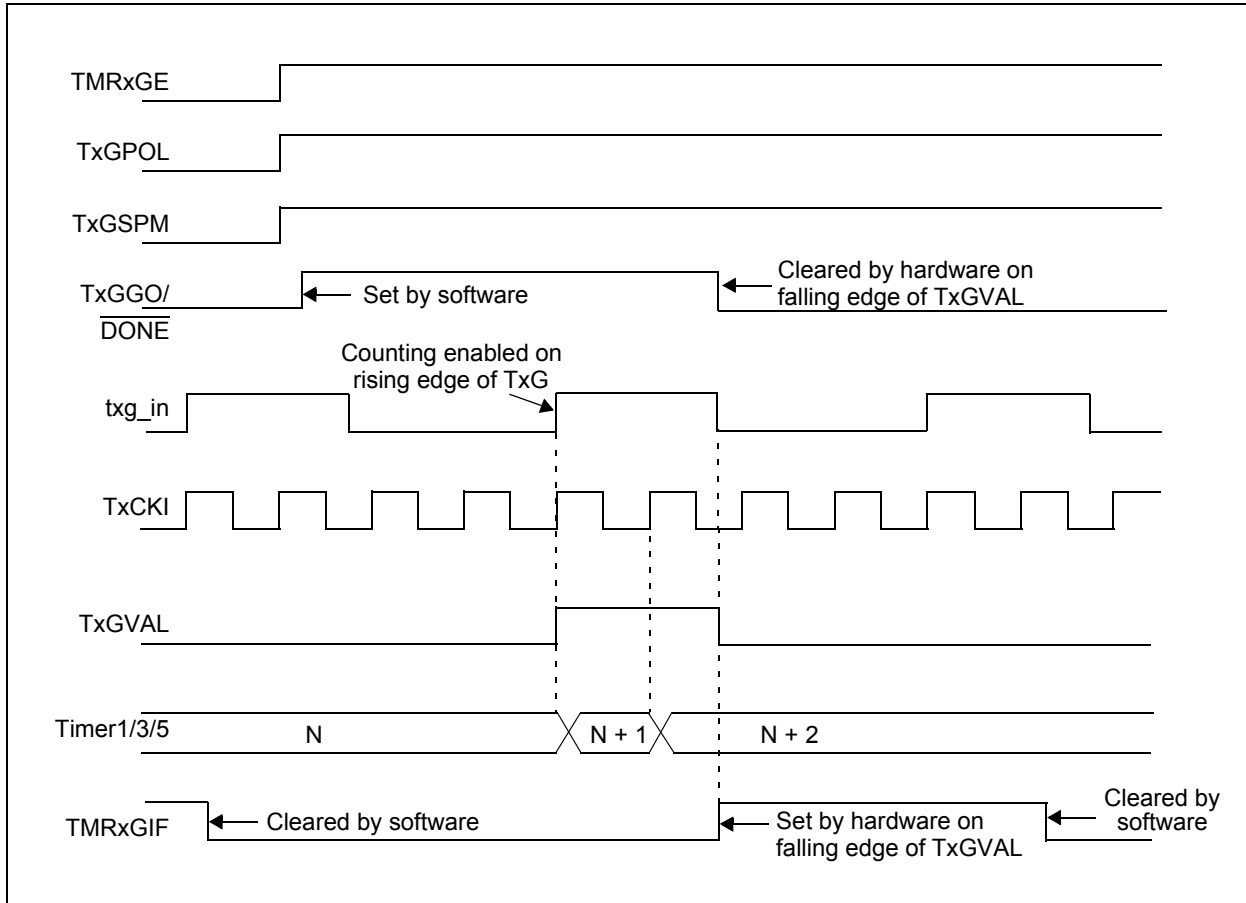
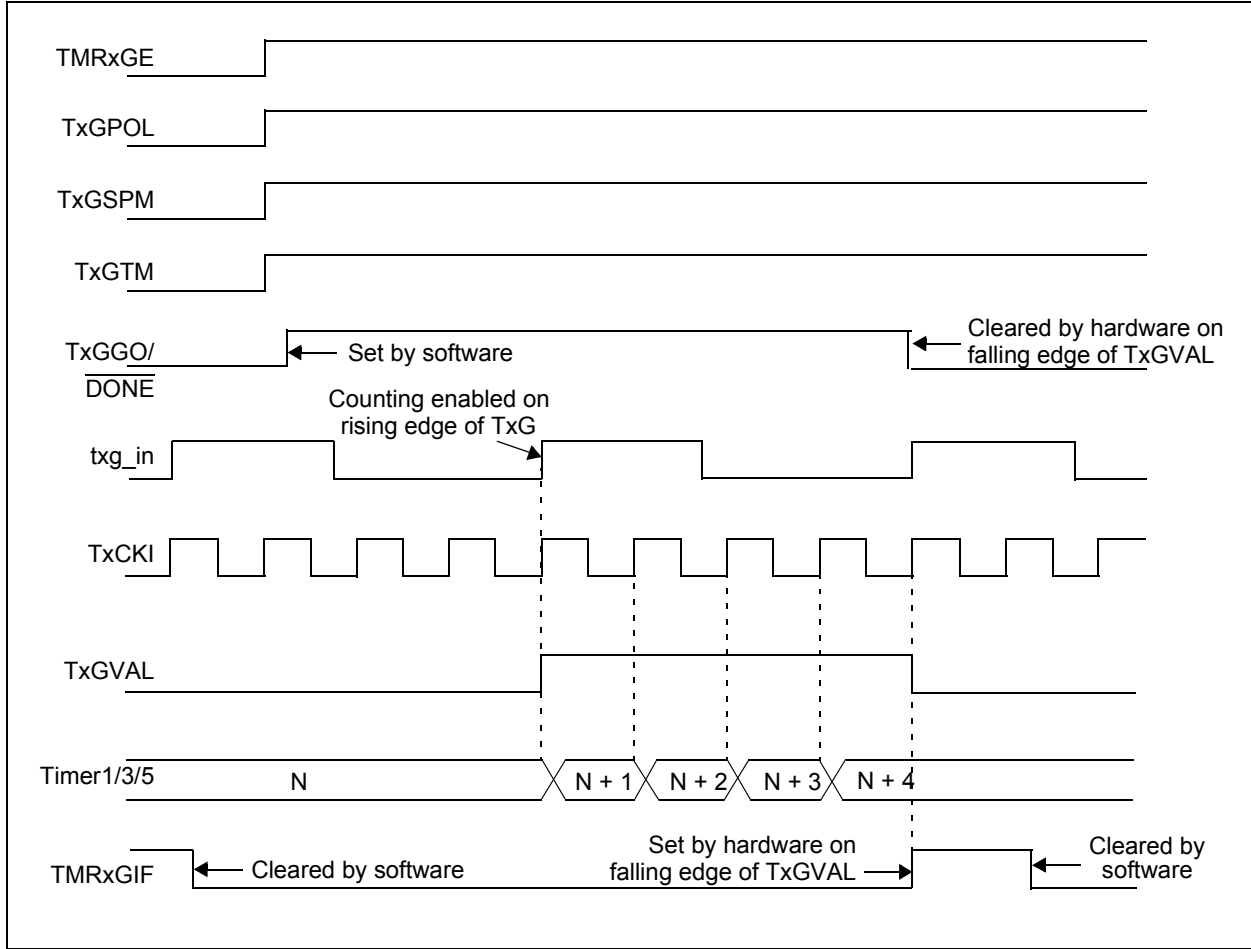


FIGURE 18-7: TIMER1/3/5 GATE SINGLE-PULSE AND TOGGLE COMBINED MODE



PIC16(L)F1526/7

18.11 Register Definitions: Timer1/3/5 Control

REGISTER 18-1: TxCON: TIMER1/3/5 CONTROL REGISTER

| | | | | | | | |
|-------------|---------|-------------|---------|---------|---------|-----|---------|
| R/W-0/u | R/W-0/u | R/W-0/u | R/W-0/u | R/W-0/u | R/W-0/u | U-0 | R/W-0/u |
| TMRxCS<1:0> | | TxCKPS<1:0> | | SOSCEN | TxSYNC | — | TMRxON |
| bit 7 | | | | | | | bit 0 |

Legend:

| | | |
|----------------------|----------------------|---|
| R = Readable bit | W = Writable bit | U = Unimplemented bit, read as '0' |
| u = Bit is unchanged | x = Bit is unknown | -n/n = Value at POR and BOR/Value at all other Resets |
| '1' = Bit is set | '0' = Bit is cleared | |

bit 7-6 **TMRxCS<1:0>**: Timer1/3/5 Clock Source Select bits

- 11 = Timer1/3/5 clock source is LFINTOSC
- 10 = Timer1/3/5 clock source is pin or oscillator:
 - If SOSCEN = 0:
External clock from TxCKI pin (on the rising edge)
 - If SOSCEN = 1:
Crystal oscillator on SOSCI/SOSCO pins
- 01 = Timer1/3/5 clock source is system clock (FOSC)
- 00 = Timer1/3/5 clock source is instruction clock (FOSC/4)

bit 5-4 **TxCKPS<1:0>**: Timer1/3/5 Input Clock Prescale Select bits

- 11 = 1:8 Prescale value
- 10 = 1:4 Prescale value
- 01 = 1:2 Prescale value
- 00 = 1:1 Prescale value

bit 3 **SOSCEN**: LP Oscillator Enable Control bit

- 1 = Dedicated secondary oscillator circuit enabled
- 0 = Dedicated secondary oscillator circuit disabled

bit 2 **TxSYNC**: Timer1/3/5 External Clock Input Synchronization Control bit

- TMRxCS<1:0> = 1X:
- 1 = Do not synchronize external clock input
- 0 = Synchronize external clock input with system clock (FOSC)

TMRxCS<1:0> = 0X:
This bit is ignored.

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

bit 0 **TMRxON**: Timer1/3/5 On bit

- 1 = Enables Timer1/3/5
- 0 = Stops Timer1/3/5
Clears Timer1/3/5 gate flip-flop

18.12 Register Definitions: Timer1/3/5 Gate Control

REGISTER 18-2: TxGCON: TIMER1/3/5 GATE CONTROL REGISTER

| | | | | | | | |
|---------|---------|---------|---------|-----------------|--------|------------|---------|
| R/W-0/u | R/W-0/u | R/W-0/u | R/W-0/u | R/W/HC-0/u | R-x/x | R/W-0/u | R/W-0/u |
| TMRxGE | TxGPOL | TxGTM | TxGSPM | TxGGGO/ DONE | TxGVAL | TxGSS<1:0> | |
| bit 7 | | | | | | bit 0 | |

Legend:

| | | |
|----------------------|----------------------|---|
| R = Readable bit | W = Writable bit | U = Unimplemented bit, read as '0' |
| u = Bit is unchanged | x = Bit is unknown | -n/n = Value at POR and BOR/Value at all other Resets |
| '1' = Bit is set | '0' = Bit is cleared | HC = Bit is cleared by hardware |

- bit 7 **TMRxGE:** Timer1/3/5 Gate Enable bit
If TMRxON = 0:
This bit is ignored
If TMRxON = 1:
1 = Timer1/3/5 counting is controlled by the Timer1/3/5 gate function
0 = Timer1/3/5 counts regardless of Timer1/3/5 gate function
- bit 6 **TxGPOL:** Timer1/3/5 Gate Polarity bit
1 = Timer1/3/5 gate is active-high (Timer1/3/5 counts when gate is high)
0 = Timer1/3/5 gate is active-low (Timer1/3/5 counts when gate is low)
- bit 5 **TxGTM:** Timer1/3/5 Gate Toggle Mode bit
1 = Timer1/3/5 Gate Toggle mode is enabled
0 = Timer1/3/5 Gate Toggle mode is disabled and toggle flip-flop is cleared
Timer1/3/5 gate flip-flop toggles on every rising edge.
- bit 4 **TxGSPM:** Timer1/3/5 Gate Single-Pulse Mode bit
1 = Timer1/3/5 Gate Single-Pulse mode is enabled and is controlling Timer1/3/5 gate
0 = Timer1/3/5 Gate Single-Pulse mode is disabled
- bit 3 **TxGGGO/DONE:** Timer1/3/5 Gate Single-Pulse Acquisition Status bit
1 = Timer1/3/5 gate single-pulse acquisition is ready, waiting for an edge
0 = Timer1/3/5 gate single-pulse acquisition has completed or has not been started
- bit 2 **TxGVAL:** Timer1/3/5 Gate Current State bit
Indicates the current state of the Timer1/3/5 gate that could be provided to TMRxH:TMRxL.
Unaffected by Timer1/3/5 Gate Enable (TMRxGE).
- bit 1-0 **TxGSS<1:0>:** Timer1/3/5 Gate Source Select bits
11 = Timer10 match PR10
10 = Timer2/4/6/8 match PR2/PR4/PR6/PR8⁽¹⁾
01 = Timer0 overflow output
00 = Timer1/3/5 gate pin

Note 1: See [Table 18-4](#) for Timer selection.

PIC16(L)F1526/7

18.12.1 ALTERNATE PIN LOCATIONS

This module incorporates I/O pins that can be moved to other locations with the use of the alternate pin function register, APFCON. To determine which pins can be moved and what their default locations are upon a Reset, see [Section 12.1 “Alternate Pin Function”](#) for more information.

TABLE 18-5: SUMMARY OF REGISTERS ASSOCIATED WITH TIMER1/3/5

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Register on Page |
|----------|---|---------|-------------|--------|----------------|---------|------------|---------|------------------|
| ANSELA | — | — | ANSA5 | — | ANSA3 | ANSA2 | ANSA1 | ANSA0 | 115 |
| APFCON | — | — | — | — | — | — | T3CKISEL | CCP2SEL | 112 |
| CCP1CON | — | — | DC1B<1:0> | | CCP1M<3:0> | | | | 189 |
| CCP2CON | — | — | DC2B<1:0> | | CCP2M<3:0> | | | | 189 |
| CCP3CON | — | — | DC3B<1:0> | | CCP3M<3:0> | | | | 189 |
| CCP4CON | — | — | DC4B<1:0> | | CCP4M<3:0> | | | | 189 |
| CCP5CON | — | — | DC5B<1:0> | | CCP5M<3:0> | | | | 189 |
| CCP6CON | — | — | DC6B<1:0> | | CCP6M<3:0> | | | | 189 |
| CCP7CON | — | — | DC7B<1:0> | | CCP7M<3:0> | | | | 189 |
| CCP8CON | — | — | DC8B<1:0> | | CCP8M<3:0> | | | | 189 |
| CCP9CON | — | — | DC9B<1:0> | | CCP9M<3:0> | | | | 189 |
| CCP10CON | — | — | DC10B<1:0> | | CCP10M<3:0> | | | | 189 |
| INTCON | GIE | PEIE | TMR0IE | INTE | IOCIE | TMR0IF | INTF | IOCIF | 76 |
| PIE1 | TMR1GIE | ADIE | RC1IE | TX1IE | SSP1IE | CCP1IE | TMR2IE | TMR1IE | 77 |
| PIE2 | OSFIE | TMR5GIE | TMR3GIE | — | BCL1IE | TMR10IE | TMR8IE | CCP2IE | 78 |
| PIE3 | CCP6IE | CCP5IE | CCP4IE | CCP3IE | TMR6IE | TMR5IE | TMR4IE | TMR3IE | 79 |
| PIR1 | TMR1GIF | ADIF | RC1IF | TX1IF | SSP1IF | CCP1IF | TMR2IF | TMR1IF | 81 |
| PIR2 | OSFIF | TMR5GIF | TMR3GIF | — | BCL1IF | TMR10IF | TMR8IF | CCP2IF | 82 |
| PIR3 | CCP6IF | CCP5IF | CCP4IF | CCP3IF | TMR6IF | TMR5IF | TMR4IF | TMR3IF | 83 |
| TMR1H | Holding Register for the Most Significant Byte of the 16-bit TMR1 Register | | | | | | | | 164* |
| TMR3H | Holding Register for the Most Significant Byte of the 16-bit TMR3 Register | | | | | | | | 164* |
| TMR5H | Holding Register for the Most Significant Byte of the 16-bit TMR5 Register | | | | | | | | 164* |
| TMR1L | Holding Register for the Least Significant Byte of the 16-bit TMR1 Register | | | | | | | | 164* |
| TMR3L | Holding Register for the Least Significant Byte of the 16-bit TMR3 Register | | | | | | | | 164* |
| TMR5L | Holding Register for the Least Significant Byte of the 16-bit TMR5 Register | | | | | | | | 164* |
| TRISA | TRISA7 | TRISA6 | TRISA5 | TRISA4 | TRISA3 | TRISA2 | TRISA1 | TRISA0 | 114 |
| T1CON | TMR1CS<1:0> | | T1CKPS<1:0> | | SOSCEN | T1SYNC | — | TMR1ON | 168 |
| T3CON | TMR3CS<1:0> | | T3CKPS<1:0> | | SOSCEN | T3SYNC | — | TMR3ON | 168 |
| T5CON | TMR5CS<1:0> | | T5CKPS<1:0> | | SOSCEN | T5SYNC | — | TMR5ON | 168 |
| T1GCON | TMR1GE | T1GPOL | T1GTM | T1GSPM | T1GGO/ DONE | T1GVAL | T1GSS<1:0> | | 169 |
| T3GCON | TMR3GE | T3GPOL | T3GTM | T3GSPM | T3GGO/ DONE | T3GVAL | T3GSS<1:0> | | 169 |
| T5GCON | TMR5GE | T5GPOL | T5GTM | T5GSPM | T5GGO/ DONE | T5GVAL | T5GSS<1:0> | | 169 |

Legend: — = unimplemented, read as '0'. Shaded cells are not used by the Timer1/3/5 module.

* Page provides register information.

19.0 TIMER2/4/6/8/10 MODULES

There are up to five identical Timer2-type modules available. To maintain pre-existing naming conventions, the Timers are called Timer2, Timer4, Timer6, Timer8 and Timer10 (also Timer2/4/6/8/10).

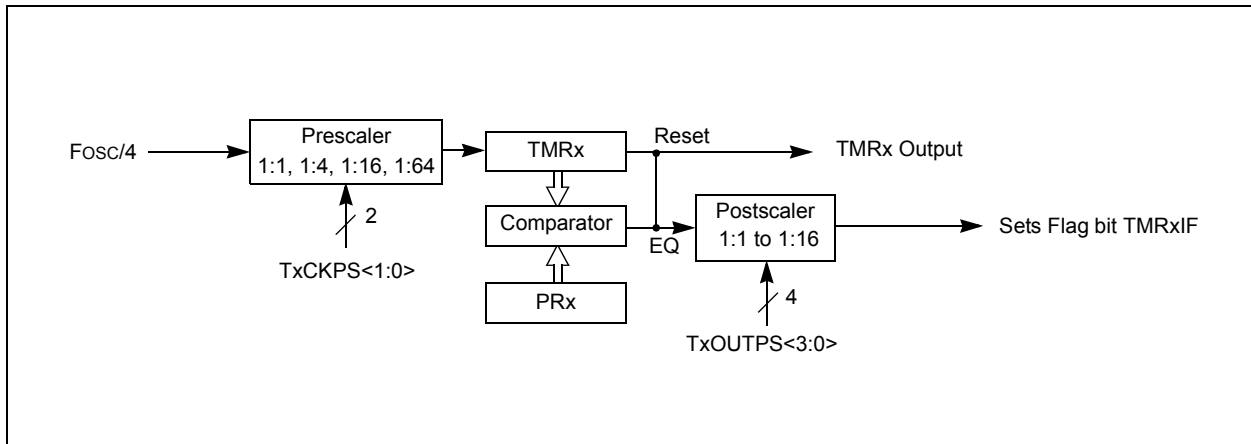
Note: The 'x' variable used in this section is used to designate Timer2, Timer4, Timer6, Timer8 or Timer10. For example, TxCON references T2CON, T4CON, T6CON, T8CON or T10CON. PRx references PR2, PR4, PR6, PR8 or PR10.

The Timer2/4/6/8/10 modules incorporate the following features:

- 8-bit Timer and Period registers (TMR2/4/6/8/10 and PR2/4/6/8/10, respectively)
- Readable and writable (both registers)
- Software programmable prescaler (1:1, 1:4, 1:16, and 1:64)
- Software programmable postscaler (1:1 to 1:16)
- Interrupt on TMR2/4/6/8/10 match with PR2/4/6/8/10, respectively
- Optional use as the shift clock for the MSSPx modules (Timer2 only)

See [Figure 19-1](#) for a block diagram of Timer2/4/6/8/10.

FIGURE 19-1: TIMER2/4/6/8/10 BLOCK DIAGRAM



PIC16(L)F1526/7

19.1 Timer2/4/6/8/10 Operation

The clock input to the Timer2/4/6/8/10 modules is the system instruction clock (Fosc/4).

TMR2/4/6/8/10 increments from 00h on each clock edge.

A 4-bit counter/prescaler on the clock input allows direct input, divide-by-4 and divide-by-16 prescale options. These options are selected by the prescaler control bits, TxCKPS<1:0> of the TxCON register. The value of TMR2/4/6/8/10 is compared to that of the Period register, PR2/4/6/8/10, on each clock cycle. When the two values match, the comparator generates a match signal as the timer output. This signal also resets the value of TMR2/4/6/8/10 to 00h on the next cycle and drives the output counter/postscaler (see [Section 19.2 “Timer2/4/6/8/10 Interrupt”](#)).

The TMR2/4/6/8/10 and PR2/4/6/8/10 registers are both directly readable and writable. The TMR2/4/6/8/10 register is cleared on any device Reset, whereas the PR2/4/6/8/10 register initializes to FFh. Both the prescaler and postscaler counters are cleared on the following events:

- a write to the TMR2/4/6/8/10 register
- a write to the TxCON register
- Power-on Reset (POR)
- Brown-out Reset (BOR)
- MCLR Reset
- Watchdog Timer (WDT) Reset
- Stack Overflow Reset
- Stack Underflow Reset
- RESET Instruction

| |
|--|
| Note: TMR2/4/6/8/10 is not cleared when TxCON is written. |
|--|

19.2 Timer2/4/6/8/10 Interrupt

Timer2/4/6/8/10 can also generate an optional device interrupt. The Timer2/4/6/8/10 output signal (TMRx-to-PRx match) provides the input for the 4-bit counter/postscaler. This counter generates the TMRx match interrupt flag which is latched in TMRxIF of the PIRx register. The interrupt is enabled by setting the TMR2/4/6/8/10 Match Interrupt Enable bit, TMRxIE of the PIRx register.

A range of 16 postscale options (from 1:1 through 1:16 inclusive) can be selected with the postscaler control bits, TxOUTPS<3:0>, of the TxCON register.

19.3 Timer2/4/6/8/10 Output

The unscaled output of TMR2/4/6/8/10 is available primarily to the CCP modules, where it is used as a time base for operations in PWM mode.

Timer2 can be optionally used as the shift clock source for the MSSPx modules operating in SPI mode. Additional information is provided in [Section 21.1 “Master SSPx \(MSSPx\) Module Overview”](#)

19.4 Timer2/4/6/8/10 Operation During Sleep

The Timer2/4/6/8/10 timers cannot be operated while the processor is in Sleep mode. The contents of the TMR2/4/6/8/10 and PR2/4/6/8/10 registers will remain unchanged while the processor is in Sleep mode.

19.5 Register Definitions: Timer2/4/6/8/10 Control

REGISTER 19-1: TxCON: TIMER2/TIMER4/TIMER6/TIMER8/TIMER10 CONTROL REGISTER

| | | | | | | | |
|-------|--------------|---------|---------|---------|-------------|---------|---------|
| U-0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 |
| — | TxOUTPS<3:0> | | | TMRxON | TxCKPS<1:0> | | |
| bit 7 | | | | | | | bit 0 |

Legend:

| | | |
|----------------------|----------------------|---|
| R = Readable bit | W = Writable bit | U = Unimplemented bit, read as '0' |
| u = Bit is unchanged | x = Bit is unknown | -n/n = Value at POR and BOR/Value at all other Resets |
| '1' = Bit is set | '0' = Bit is cleared | |

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

bit 6-3 **TxOUTPS<3:0>:** Timerx Output Postscaler Select bits

1111 = 1:16 Postscaler
 1110 = 1:15 Postscaler
 1101 = 1:14 Postscaler
 1100 = 1:13 Postscaler
 1011 = 1:12 Postscaler
 1010 = 1:11 Postscaler
 1001 = 1:10 Postscaler
 1000 = 1:9 Postscaler
 0111 = 1:8 Postscaler
 0110 = 1:7 Postscaler
 0101 = 1:6 Postscaler
 0100 = 1:5 Postscaler
 0011 = 1:4 Postscaler
 0010 = 1:3 Postscaler
 0001 = 1:2 Postscaler
 0000 = 1:1 Postscaler

bit 2 **TMRxON:** Timerx On bit

1 = Timer2/4/6 is on
 0 = Timer2/4/6 is off

bit 1-0 **TxCKPS<1:0>:** Timer2-type Clock Prescale Select bits

11 = Prescaler is 64
 10 = Prescaler is 16
 01 = Prescaler is 4
 00 = Prescaler is 1

PIC16(L)F1526/7

TABLE 19-1: SUMMARY OF REGISTERS ASSOCIATED WITH TIMER2/4/6/8/10

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Register on Page |
|----------|--|---------------|------------|--------|-------------|---------|----------|----------|------------------|
| CCP1CON | — | — | DC1B<1:0> | | CCP1M<3:0> | | | | 189 |
| CCP2CON | — | — | DC2B<1:0> | | CCP2M<3:0> | | | | 189 |
| CCP3CON | — | — | DC3B<1:0> | | CCP3M<3:0> | | | | 189 |
| CCP4CON | — | — | DC4B<1:0> | | CCP4M<3:0> | | | | 189 |
| CCP5CON | — | — | DC5B<1:0> | | CCP5M<3:0> | | | | 189 |
| CCP6CON | — | — | DC6B<1:0> | | CCP6M<3:0> | | | | 189 |
| CCP7CON | — | — | DC7B<1:0> | | CCP7M<3:0> | | | | 189 |
| CCP8CON | — | — | DC8B<1:0> | | CCP8M<3:0> | | | | 189 |
| CCP9CON | — | — | DC9B<1:0> | | CCP9M<3:0> | | | | 189 |
| CCP10CON | — | — | DC10B<1:0> | | CCP10M<3:0> | | | | 189 |
| INTCON | GIE | PEIE | TMR0IE | INTE | IOCFIE | TMR0IF | INTF | IOCF | 76 |
| PIE1 | TMR1GIE | ADIE | RC1IE | TX1IE | SSP1IE | CCP1IE | TMR2IE | TMR1IE | 77 |
| PIE2 | OSFIE | TMR5GIE | TMR3GIE | — | BCL1IE | TMR10IE | TMR8IE | CCP2IE | 78 |
| PIE3 | CCP6IE | CCP5IE | CCP4IE | CCP3IE | TMR6IE | TMR5IE | TMR4IE | TMR3IE | 79 |
| PIR1 | TMR1GIF | ADIF | RC1IF | TX1IF | SSP1IF | CCP1IF | TMR2IF | TMR1IF | 81 |
| PIR2 | OSFIF | TMR5GIF | TMR3GIF | — | BCL1IF | TMR10IF | TMR8IF | CCP2IF | 82 |
| PIR3 | CCP6IF | CCP5IF | CCP4IF | CCP3IF | TMR6IF | TMR5IF | TMR4IF | TMR3IF | 83 |
| PR2 | Timer2 Module Period Register | | | | | | | | 213* |
| PR4 | Timer4 Module Period Register | | | | | | | | 213* |
| PR6 | Timer6 Module Period Register | | | | | | | | 213* |
| PR8 | Timer8 Module Period Register | | | | | | | | 213* |
| PR10 | Timer10 Module Period Register | | | | | | | | 213* |
| T2CON | — | T2OUTPS<3:0> | | | | TMR2ON | T2CKPS1 | T2CKPS0 | 215 |
| T4CON | — | T4OUTPS<3:0> | | | | TMR4ON | T4CKPS1 | T4CKPS0 | 215 |
| T6CON | — | T6OUTPS<3:0> | | | | TMR6ON | T6CKPS1 | T6CKPS0 | 215 |
| T8CON | — | T8OUTPS<3:0> | | | | TMR8ON | T8CKPS1 | T8CKPS0 | 215 |
| T10CON | — | T10OUTPS<3:0> | | | | TMR10ON | T10CKPS1 | T10CKPS0 | 215 |
| TMR2 | Holding Register for the 8-bit TMR2 Register | | | | | | | | 213* |
| TMR4 | Holding Register for the 8-bit TMR4 Register ⁽¹⁾ | | | | | | | | 213* |
| TMR6 | Holding Register for the 8-bit TMR6 Register ⁽¹⁾ | | | | | | | | 213* |
| TMR8 | Holding Register for the 8-bit TMR8 Register ⁽¹⁾ | | | | | | | | 213* |
| TMR10 | Holding Register for the 8-bit TMR10 Register ⁽¹⁾ | | | | | | | | 213* |

Legend: — = unimplemented location, read as '0'. Shaded cells are not used for Timer2/4/6/8/10 module.

* Page provides register information.

20.0 CAPTURE/COMPARE/PWM MODULES

The Capture/Compare/PWM module is a peripheral which allows the user to time and control different events, and to generate Pulse-Width Modulation (PWM) signals. In Capture mode, the peripheral allows the timing of the duration of an event. The Compare mode allows the user to trigger an external event when a predetermined amount of time has expired. The PWM mode can generate Pulse-Width Modulated signals of varying frequency and duty cycle.

This device contains ten standard Capture/Compare/PWM modules (CCP1 through CCP10).

The capture and compare functions are identical for all CCP modules.

Note 1: In devices with more than one CCP module, it is very important to pay close attention to the register names used. A number placed after the module acronym is used to distinguish between separate modules. For example, the CCP1CON and CCP2CON control the same operational aspects of two completely different CCP modules.

2: Throughout this section, generic references to a CCP module in any of its operating modes may be interpreted as being equally applicable to any CCPx module. Register names, module signals, I/O pins, and bit names may use the generic designator 'x' to indicate the use of a numeral to distinguish a particular module, when required.

PIC16(L)F1526/7

20.1 Capture Mode

The Capture mode function described in this section is available and identical for CCP modules.

Capture mode makes use of the 16-bit Timer1/3/5 resource. When an event occurs on the CCPx pin, the 16-bit CCPRxH:CCPRxL register pair captures and stores the 16-bit value of the TMRxH:TMRxL register pair, respectively. An event is defined as one of the following and is configured by the CCPxM<3:0> bits of the CCPxCON register:

- Every falling edge
- Every rising edge
- Every 4th rising edge
- Every 16th rising edge

When a capture is made, the Interrupt Request Flag bit CCPxIF of the PIRx register is set. The interrupt flag must be cleared in software. If another capture occurs before the value in the CCPRxH, CCPRxL register pair is read, the old captured value is overwritten by the new captured value.

Figure 20-1 shows a simplified diagram of the Capture operation.

20.1.1 CCP PIN CONFIGURATION

In Capture mode, the CCPx pin should be configured as an input by setting the associated TRIS control bit.

Also, the CCP2x pin function can be moved to alternative pins using the APFCON register. Refer to Section 12.1 “Alternate Pin Function” for more details.

Note: If the CCPx pin is configured as an output, a write to the port can cause a capture condition.

20.1.2 TIMER1/3/5 MODE RESOURCE

Timer1/3/5 must be running in Timer mode or Synchronized Counter mode for the CCP module to use the capture feature. In Asynchronous Counter mode, the capture operation may not work.

See Section 18.0 “Timer1/3/5 Module with Gate Control” for more information on configuring Timer1/3/5.

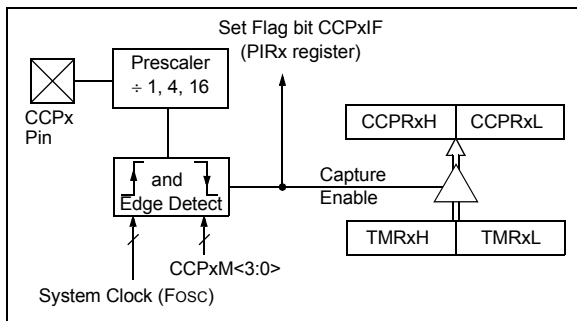
TABLE 20-1: CCPx CAPTURE TIMER1/3/5 RESOURCES

| CCP | TMR1 | TMR3 | TMR5 |
|-------|------|------|------|
| CCP1 | • | • | |
| CCP2 | • | • | |
| CCP3 | • | • | |
| CCP4 | • | • | |
| CCP5 | • | • | |
| CCP6 | • | | • |
| CCP7 | • | | • |
| CCP8 | • | | • |
| CCP9 | • | | • |
| CCP10 | • | | • |

20.1.3 SOFTWARE INTERRUPT MODE

When the Capture mode is changed, a false capture interrupt may be generated. The user should keep the CCPxIE interrupt enable bit of the PIRx register clear to avoid false interrupts. Additionally, the user should clear the CCPxIF interrupt flag bit of the PIRx register following any change in Operating mode.

FIGURE 20-1: CAPTURE MODE OPERATION BLOCK DIAGRAM



20.1.4 CCP PRESCALER

There are four prescaler settings specified by the CCPxM<3:0> bits of the CCPxCON register. Whenever the CCP module is turned off, or the CCP module is not in Capture mode, the prescaler counter is cleared. Any Reset will clear the prescaler counter.

Switching from one capture prescaler to another does not clear the prescaler and may generate a false interrupt. To avoid this unexpected operation, turn the module off by clearing the CCPxCON register before changing the prescaler. [Equation 20-1](#) demonstrates the code to perform this function.

EXAMPLE 20-1: CHANGING BETWEEN CAPTURE PRESCALERS

```
BANKSEL CCPxCON    ;Set Bank bits to point
                   ;to CCPxCON
CLRF    CCPxCON    ;Turn CCP module off
MOVLW  NEW_CAPT_PS ;Load the W reg with
                   ;the new prescaler
                   ;move value and CCP ON
MOVWF  CCPxCON    ;Load CCPxCON with this
                   ;value
```

20.1.5 CAPTURE DURING SLEEP

Capture mode depends upon the Timer1/3/5 module for proper operation. There are two options for driving the Timer1/3/5 module in Capture mode. It can be driven by the instruction clock (FOSC/4), or by an external clock source.

When Timer1/3/5 is clocked by FOSC/4, Timer1/3/5 will not increment during Sleep. When the device wakes from Sleep, Timer1/3/5 will continue from its previous state.

Capture mode will operate during Sleep when Timer1/3/5 is clocked by an external clock source.

20.1.6 ALTERNATE PIN LOCATIONS

This module incorporates I/O pins that can be moved to other locations with the use of the alternate pin function register, APFCON. To determine which pins can be moved and what their default locations are upon a reset, see [Section 12.1 "Alternate Pin Function"](#) for more information.

PIC16(L)F1526/7

TABLE 20-2: SUMMARY OF REGISTERS ASSOCIATED WITH CAPTURE

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Register on Page |
|----------|---|---------|-------------|--------|-------------|---------|------------|---------|------------------|
| APFCON | — | — | — | — | — | — | T3CKISEL | CCP2SEL | 112 |
| CCP1CON | — | — | DC1B<1:0> | | CCP1M<3:0> | | | | 189 |
| CCP2CON | — | — | DC2B<1:0> | | CCP2M<3:0> | | | | 189 |
| CCP3CON | — | — | DC3B<1:0> | | CCP3M<3:0> | | | | 189 |
| CCP4CON | — | — | DC4B<1:0> | | CCP4M<3:0> | | | | 189 |
| CCP5CON | — | — | DC5B<1:0> | | CCP5M<3:0> | | | | 189 |
| CCP6CON | — | — | DC6B<1:0> | | CCP6M<3:0> | | | | 189 |
| CCP7CON | — | — | DC7B<1:0> | | CCP7M<3:0> | | | | 189 |
| CCP8CON | — | — | DC8B<1:0> | | CCP8M<3:0> | | | | 189 |
| CCP9CON | — | — | DC9B<1:0> | | CCP9M<3:0> | | | | 189 |
| CCP10CON | — | — | DC10B<1:0> | | CCP10M<3:0> | | | | 189 |
| CCPR1L | Capture/Compare/PWM Register 1 Low Byte (LSB) | | | | | | | | 176* |
| CCPR2L | Capture/Compare/PWM Register 2 Low Byte (LSB) | | | | | | | | 176* |
| CCPR3L | Capture/Compare/PWM Register 3 Low Byte (LSB) | | | | | | | | 176* |
| CCPR4L | Capture/Compare/PWM Register 4 Low Byte (LSB) | | | | | | | | 176* |
| CCPR5L | Capture/Compare/PWM Register 5 Low Byte (LSB) | | | | | | | | 176* |
| CCPR6L | Capture/Compare/PWM Register 6 Low Byte (LSB) | | | | | | | | 176* |
| CCPR7L | Capture/Compare/PWM Register 7 Low Byte (LSB) | | | | | | | | 176* |
| CCPR8L | Capture/Compare/PWM Register 8 Low Byte (LSB) | | | | | | | | 176* |
| CCPR9L | Capture/Compare/PWM Register 9 Low Byte (LSB) | | | | | | | | 176* |
| CCPR10L | Capture/Compare/PWM Register 10 Low Byte (LSB) | | | | | | | | 176* |
| CCPR1H | Capture/Compare/PWM Register 1 High Byte (MSB) | | | | | | | | 176* |
| CCPR2H | Capture/Compare/PWM Register 2 High Byte (MSB) | | | | | | | | 176* |
| CCPR3H | Capture/Compare/PWM Register 3 High Byte (MSB) | | | | | | | | 176* |
| CCPR4H | Capture/Compare/PWM Register 4 High Byte (MSB) | | | | | | | | 176* |
| CCPR5H | Capture/Compare/PWM Register 5 High Byte (MSB) | | | | | | | | 176* |
| CCPR6H | Capture/Compare/PWM Register 6 High Byte (MSB) | | | | | | | | 176* |
| CCPR7H | Capture/Compare/PWM Register 7 High Byte (MSB) | | | | | | | | 176* |
| CCPR8H | Capture/Compare/PWM Register 8 High Byte (MSB) | | | | | | | | 176* |
| CCPR9H | Capture/Compare/PWM Register 9 High Byte (MSB) | | | | | | | | 176* |
| CCPR10H | Capture/Compare/PWM Register 10 High Byte (MSB) | | | | | | | | 176* |
| INTCON | GIE | PEIE | TMR0IE | INTE | IOCF | TMR0IF | INTF | IOCF | 76 |
| PIE1 | TMR1GIE | ADIE | RC1IE | TX1IE | SSP1IE | CCP1IE | TMR2IE | TMR1IE | 77 |
| PIE2 | OSFIE | TMR5GIE | TMR3GIE | — | BCL1IE | TMR10IE | TMR8IE | CCP2IE | 78 |
| PIE3 | CCP6IE | CCP5IE | CCP4IE | CCP3IE | TMR6IE | TMR5IE | TMR4IE | TMR3IE | 79 |
| PIE4 | CCP10IE | CCP9IE | RC2IE | TX2IE | CCP8IE | CCP7IE | BCL2IE | SSP2IE | 80 |
| PIR1 | TMR1GIF | ADIF | RC1IF | TX1IF | SSP1IF | CCP1IF | TMR2IF | TMR1IF | 81 |
| PIR2 | OSFIF | TMR5GIF | TMR3GIF | — | BCL1IF | TMR10IF | TMR8IF | CCP2IF | 82 |
| PIR3 | CCP6IF | CCP5IF | CCP4IF | CCP3IF | TMR6IF | TMR5IF | TMR4IF | TMR3IF | 83 |
| PIR4 | CCP10IF | CCP9IF | RC2IF | TX2IF | CCP8IF | CCP7IF | BCL2IF | SSP2IF | 84 |
| T1CON | TMR1CS<1:0> | | T1CKPS<1:0> | | SOSCEN | T1SYNC | — | TMR1ON | 168 |
| T3CON | TMR3CS<1:0> | | T3CKPS<1:0> | | SOSCEN | T3SYNC | — | TMR3ON | 168 |
| T5CON | TMR5CS<1:0> | | T5CKPS<1:0> | | SOSCEN | T5SYNC | — | TMR5ON | 168 |
| T1GCON | TMR1GE | T1GPOL | T1GTM | T1GSPM | T1GGO/DONE | T1GVAL | T1GSS<1:0> | | 169 |
| T3GCON | TMR3GE | T3GPOL | T3GTM | T3GSPM | T3GGO/DONE | T3GVAL | T3GSS<1:0> | | 169 |

Legend: — = Unimplemented location, read as '0'. Shaded cells are not used by Capture mode.
 * Page provides register information.

TABLE 20-2: SUMMARY OF REGISTERS ASSOCIATED WITH CAPTURE (CONTINUED)

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Register on Page |
|--------|---|--------|--------|--------|------------|--------|------------|--------|------------------|
| T5GCON | TMR5GE | T5GPOL | T5GTM | T5GSPM | T5GGO/DONE | T5GVAL | T5GSS<1:0> | | 169 |
| TMR1L | Holding Register for the Least Significant Byte of the 16-bit TMR1 Register | | | | | | | | 164* |
| TMR3L | Holding Register for the Least Significant Byte of the 16-bit TMR3 Register | | | | | | | | 164* |
| TMR6L | Holding Register for the Least Significant Byte of the 16-bit TMR6 Register | | | | | | | | 164* |
| TMR1H | Holding Register for the Most Significant Byte of the 16-bit TMR1 Register | | | | | | | | 164* |
| TMR3H | Holding Register for the Most Significant Byte of the 16-bit TMR3 Register | | | | | | | | 164* |
| TMR6H | Holding Register for the Most Significant Byte of the 16-bit TMR6 Register | | | | | | | | 164* |
| TRISA | TRISA7 | TRISA6 | TRISA5 | TRISA4 | TRISA3 | TRISA2 | TRISA1 | TRISA0 | 114 |

Legend: — = Unimplemented location, read as '0'. Shaded cells are not used by Capture mode.

* Page provides register information.

PIC16(L)F1526/7

20.2 Compare Mode

The Compare mode function described in this section is available and identical for CCP modules.

Compare mode makes use of the 16-bit Timer1/3/5 resource. The 16-bit value of the CCPRxH:CCPRxL register pair is constantly compared against the 16-bit value of the TMRxH:TMRxL register pair. When a match occurs, one of the following events can occur:

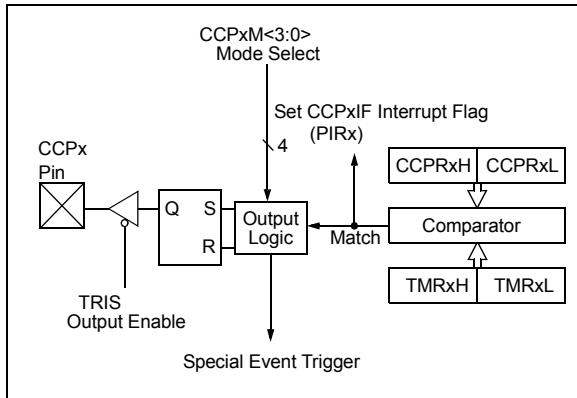
- Toggle the CCPx output
- Set the CCPx output
- Clear the CCPx output
- Generate a Special Event Trigger
- Generate a Software Interrupt

The action on the pin is based on the value of the CCPxM<3:0> control bits of the CCPxCON register. At the same time, the interrupt flag CCPxIF bit is set.

All Compare modes can generate an interrupt.

Figure 20-2 shows a simplified diagram of the Compare operation.

FIGURE 20-2: COMPARE MODE OPERATION BLOCK DIAGRAM



20.2.1 CCP PIN CONFIGURATION

The user must configure the CCPx pin as an output by clearing the associated TRIS bit.

Also, the CCPx pin function can be moved to alternative pins using the APFCON register. Refer to [Section 12.1 “Alternate Pin Function”](#) for more details.

Note: Clearing the CCPxCON register will force the CCPx compare output latch to the default low level. This is not the PORT I/O data latch.

20.2.2 TIMER1/3/5 MODE RESOURCE

In Compare mode, Timer1/3/5 must be running in either Timer mode or Synchronized Counter mode. The compare operation may not work in Asynchronous Counter mode.

TABLE 20-3: CCPx COMPARE TIMER1/3/5 RESOURCES

| CCP | TMR1 | TMR3 | TMR5 |
|-------|------|------|------|
| CCP1 | • | • | |
| CCP2 | • | • | |
| CCP3 | • | • | |
| CCP4 | • | • | |
| CCP5 | • | • | |
| CCP6 | • | | • |
| CCP7 | • | | • |
| CCP8 | • | | • |
| CCP9 | • | | • |
| CCP10 | • | | • |

See [Section 18.0 “Timer1/3/5 Module with Gate Control”](#) for more information on configuring Timer1/3/5.

Note: Clocking Timer1/3/5 from the system clock (FOSC) should not be used in Compare mode. In order for Compare mode to recognize the trigger event on the CCPx pin, Timer1/3/5 must be clocked from the instruction clock (FOSC/4) or from an external clock source.

20.2.3 SOFTWARE INTERRUPT MODE

When Generate Software Interrupt mode is chosen (CCPxM<3:0> = 1010), the CCPx module does not assert control of the CCPx pin (see the CCPxCON register).

20.2.4 SPECIAL EVENT TRIGGER

When Special Event Trigger mode is chosen (CCPxM<3:0> = 1011), the CCPx module does the following:

- Resets Timer1/3/5
- Starts an ADC conversion if ADC is enabled

The CCPx module does not assert control of the CCPx pin in this mode.

The Special Event Trigger output of the CCP occurs immediately upon a match between the TMRxH, TMRxL register pair and the CCPRxH, CCPRxL register pair. The TMRxH, TMRxL register pair is not reset until the next rising edge of the Timer1/3/5 clock. The Special Event Trigger output starts an ADC conversion (if the ADC module is enabled). This allows the CCPRxH, CCPRxL register pair to effectively provide a 16-bit programmable period register for Timer1/3/5.

TABLE 20-4: SPECIAL EVENT TRIGGER

| Device | CCPx |
|-----------------|-------|
| PIC16(L)F1526/7 | CCP10 |

Refer to [Section 16.2.5 “Special Event Trigger”](#) for more information.

Note 1: The Special Event Trigger from the CCP module does not set interrupt flag bit TMRxIF of the PIRx register.

2: Removing the match condition by changing the contents of the CCPRxH and CCPRxL register pair, between the clock edge that generates the Special Event Trigger and the clock edge that generates the Timer1/3/5 Reset, will preclude the Reset from occurring.

20.2.5 COMPARE DURING SLEEP

The Compare mode is dependent upon the system clock (Fosc) for proper operation. Since Fosc is shut down during Sleep mode, the Compare mode will not function properly during Sleep.

20.2.6 ALTERNATE PIN LOCATIONS

This module incorporates I/O pins that can be moved to other locations with the use of the alternate pin function register, APFCON. To determine which pins can be moved and what their default locations are upon a reset, see [Section 12.1 “Alternate Pin Function”](#) for more information.

PIC16(L)F1526/7

TABLE 20-5: SUMMARY OF REGISTERS ASSOCIATED WITH COMPARE

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Register on Page |
|----------|---|---------|-------------|--------|-------------|---------|------------|---------|------------------|
| APFCON | — | — | — | — | — | — | T3CKISEL | CCP2SEL | 112 |
| CCP1CON | — | — | DC1B<1:0> | | CCP1M<3:0> | | | | 189 |
| CCP2CON | — | — | DC2B<1:0> | | CCP2M<3:0> | | | | 189 |
| CCP3CON | — | — | DC3B<1:0> | | CCP3M<3:0> | | | | 189 |
| CCP4CON | — | — | DC4B<1:0> | | CCP4M<3:0> | | | | 189 |
| CCP5CON | — | — | DC5B<1:0> | | CCP5M<3:0> | | | | 189 |
| CCP6CON | — | — | DC6B<1:0> | | CCP6M<3:0> | | | | 189 |
| CCP7CON | — | — | DC7B<1:0> | | CCP7M<3:0> | | | | 189 |
| CCP8CON | — | — | DC8B<1:0> | | CCP8M<3:0> | | | | 189 |
| CCP9CON | — | — | DC9B<1:0> | | CCP9M<3:0> | | | | 189 |
| CCP10CON | — | — | DC10B<1:0> | | CCP10M<3:0> | | | | 189 |
| CCPR1L | Capture/Compare/PWM Register 1 Low Byte (LSB) | | | | | | | | 176* |
| CCPR2L | Capture/Compare/PWM Register 2 Low Byte (LSB) | | | | | | | | 176* |
| CCPR3L | Capture/Compare/PWM Register 3 Low Byte (LSB) | | | | | | | | 176* |
| CCPR4L | Capture/Compare/PWM Register 4 Low Byte (LSB) | | | | | | | | 176* |
| CCPR5L | Capture/Compare/PWM Register 5 Low Byte (LSB) | | | | | | | | 176* |
| CCPR6L | Capture/Compare/PWM Register 6 Low Byte (LSB) | | | | | | | | 176* |
| CCPR7L | Capture/Compare/PWM Register 7 Low Byte (LSB) | | | | | | | | 176* |
| CCPR8L | Capture/Compare/PWM Register 8 Low Byte (LSB) | | | | | | | | 176* |
| CCPR9L | Capture/Compare/PWM Register 9 Low Byte (LSB) | | | | | | | | 176* |
| CCPR10L | Capture/Compare/PWM Register 10 Low Byte (LSB) | | | | | | | | 176* |
| CCPR1H | Capture/Compare/PWM Register 1 High Byte (MSB) | | | | | | | | 176* |
| CCPR2H | Capture/Compare/PWM Register 2 High Byte (MSB) | | | | | | | | 176* |
| CCPR3H | Capture/Compare/PWM Register 3 High Byte (MSB) | | | | | | | | 176* |
| CCPR4H | Capture/Compare/PWM Register 4 High Byte (MSB) | | | | | | | | 176* |
| CCPR5H | Capture/Compare/PWM Register 5 High Byte (MSB) | | | | | | | | 176* |
| CCPR6H | Capture/Compare/PWM Register 6 High Byte (MSB) | | | | | | | | 176* |
| CCPR7H | Capture/Compare/PWM Register 7 High Byte (MSB) | | | | | | | | 176* |
| CCPR8H | Capture/Compare/PWM Register 8 High Byte (MSB) | | | | | | | | 176* |
| CCPR9H | Capture/Compare/PWM Register 9 High Byte (MSB) | | | | | | | | 176* |
| CCPR10H | Capture/Compare/PWM Register 10 High Byte (MSB) | | | | | | | | 176* |
| INTCON | GIE | PEIE | TMR0IE | INTE | IOCF | TMR0IF | INTF | IOCF | 76 |
| PIE1 | TMR1GIE | ADIE | RC1IE | TX1IE | SSP1IE | CCP1IE | TMR2IE | TMR1IE | 77 |
| PIE2 | OSFIE | TMR5GIE | TMR3GIE | — | BCL1IE | TMR10IE | TMR8IE | CCP2IE | 78 |
| PIE3 | CCP6IE | CCP5IE | CCP4IE | CCP3IE | TMR6IE | TMR5IE | TMR4IE | TMR3IE | 79 |
| PIE4 | CCP10IE | CCP9IE | RC2IE | TX2IE | CCP8IE | CCP7IE | BCL2IE | SSP2IE | 80 |
| PIR1 | TMR1GIF | ADIF | RC1IF | TX1IF | SSP1IF | CCP1IF | TMR2IF | TMR1IF | 81 |
| PIR2 | OSFIF | TMR5GIF | TMR3GIF | — | BCL1IF | TMR10IF | TMR8IF | CCP2IF | 82 |
| PIR3 | CCP6IF | CCP5IF | CCP4IF | CCP3IF | TMR6IF | TMR5IF | TMR4IF | TMR3IF | 83 |
| PIR4 | CCP10IF | CCP9IF | RC2IF | TX2IF | CCP8IF | CCP7IF | BCL2IF | SSP2IF | 84 |
| T1CON | TMR1CS<1:0> | | T1CKPS<1:0> | | SOSCEN | T1SYNC | — | TMR1ON | 168 |
| T3CON | TMR3CS<1:0> | | T3CKPS<1:0> | | SOSCEN | T3SYNC | — | TMR3ON | 168 |
| T5CON | TMR5CS<1:0> | | T5CKPS<1:0> | | SOSCEN | T5SYNC | — | TMR5ON | 168 |
| T1GCON | TMR1GE | T1GPOL | T1GTM | T1GSPM | T1GGO/DONE | T1GVAL | T1GSS<1:0> | | 169 |
| T3GCON | TMR3GE | T3GPOL | T3GTM | T3GSPM | T3GGO/DONE | T3GVAL | T3GSS<1:0> | | 169 |

Legend: — = Unimplemented location, read as '0'. Shaded cells are not used by Compare mode.
 * Page provides register information.

TABLE 20-5: SUMMARY OF REGISTERS ASSOCIATED WITH COMPARE (CONTINUED)

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Register on Page |
|--------|---|--------|--------|--------|------------|--------|------------|--------|------------------|
| T5GCON | TMR5GE | T5GPOL | T5GTM | T5GSPM | T5GGO/DONE | T5GVAL | T5GSS<1:0> | | 169 |
| TMR1L | Holding Register for the Least Significant Byte of the 16-bit TMR1 Register | | | | | | | | 164* |
| TMR3L | Holding Register for the Least Significant Byte of the 16-bit TMR3 Register | | | | | | | | 164* |
| TMR6L | Holding Register for the Least Significant Byte of the 16-bit TMR6 Register | | | | | | | | 164* |
| TMR1H | Holding Register for the Most Significant Byte of the 16-bit TMR1 Register | | | | | | | | 164* |
| TMR3H | Holding Register for the Most Significant Byte of the 16-bit TMR3 Register | | | | | | | | 164* |
| TMR6H | Holding Register for the Most Significant Byte of the 16-bit TMR6 Register | | | | | | | | 164* |
| TRISA | TRISA7 | TRISA6 | TRISA5 | TRISA4 | TRISA3 | TRISA2 | TRISA1 | TRISA0 | 114 |

Legend: — = Unimplemented location, read as '0'. Shaded cells are not used by Compare mode.

* Page provides register information.

PIC16(L)F1526/7

20.3 PWM Overview

Pulse-Width Modulation (PWM) is a scheme that provides power to a load by switching quickly between fully on and fully off states. The PWM signal resembles a square wave where the high portion of the signal is considered the on state and the low portion of the signal is considered the off state. The high portion, also known as the pulse width, can vary in time and is defined in steps. A larger number of steps applied, which lengthens the pulse width, also supplies more power to the load. Lowering the number of steps applied, which shortens the pulse width, supplies less power. The PWM period is defined as the duration of one complete cycle or the total amount of on and off time combined.

PWM resolution defines the maximum number of steps that can be present in a single PWM period. A higher resolution allows for more precise control of the pulse width time and in turn the power that is applied to the load.

The term duty cycle describes the proportion of the on time to the off time and is expressed in percentages, where 0% is fully off and 100% is fully on. A lower duty cycle corresponds to less power applied and a higher duty cycle corresponds to more power applied.

Figure 20-3 shows a typical waveform of the PWM signal.

20.3.1 STANDARD PWM OPERATION

The standard PWM function described in this section is available and identical for CCP modules.

The standard PWM mode generates a Pulse-Width Modulation (PWM) signal on the CCPx pin with up to 10 bits of resolution. The period, duty cycle, and resolution are controlled by the following registers:

- PRx registers
- TxCON registers
- CCPxL registers
- CCPxCON registers

Figure 20-4 shows a simplified block diagram of PWM operation.

Note 1: The corresponding TRIS bit must be cleared to enable the PWM output on the CCPx pin.

2: Clearing the CCPxCON register will relinquish control of the CCPx pin.

FIGURE 20-3: CCP PWM OUTPUT SIGNAL

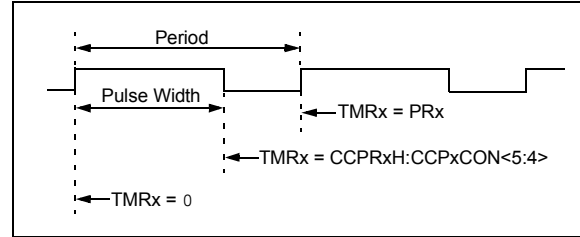


FIGURE 20-4: SIMPLIFIED PWM BLOCK DIAGRAM

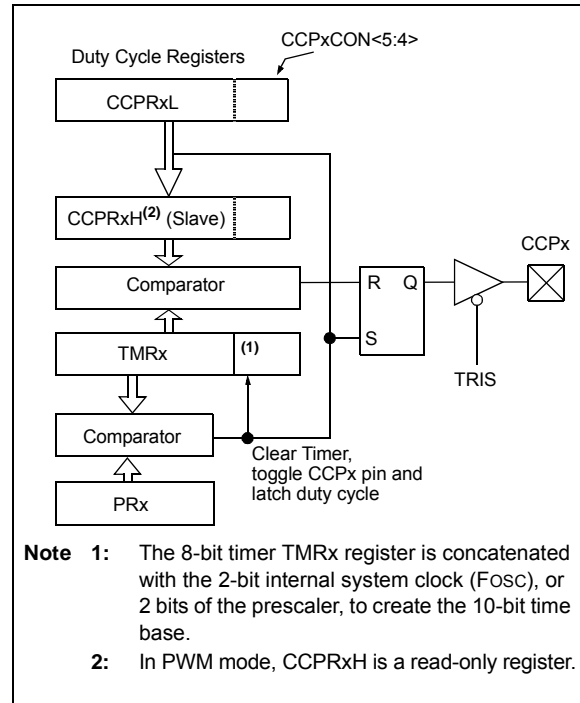


TABLE 20-6: CCPx PWM TIMER2/4/6/8/10 RESOURCES

| CCP | TMR2 | TMR4 | TMR6 | TMR8 | TMR10 |
|-------|------|------|------|------|-------|
| CCP1 | • | • | • | | |
| CCP2 | • | • | • | | |
| CCP3 | • | • | • | | |
| CCP4 | • | • | • | | |
| CCP5 | • | | • | • | |
| CCP6 | • | | • | • | |
| CCP7 | • | | • | • | |
| CCP8 | • | | | • | • |
| CCP9 | • | | | • | • |
| CCP10 | • | | | • | • |

20.3.2 SETUP FOR PWM OPERATION

The following steps should be taken when configuring the CCP module for standard PWM operation:

1. Disable the CCPx pin output driver by setting the associated TRIS bit.
2. Load the PRx register with the PWM period value.
3. Configure the CCP module for the PWM mode by loading the CCPxCON register with the appropriate values.
4. Load the CCPRxL register and the DCxBx bits of the CCPxCON register, with the PWM duty cycle value.
5. Configure and start Timer2/4/6/8/10:
 - Select the Timer2/4/6/8/10 resource to be used for PWM generation by setting the CxTSEL<1:0> bits in the CCPTMRSx register.
 - Clear the TMRxIF interrupt flag bit of the PIRx register. See Note below.
 - Configure the TxCKPS bits of the TxCON register with the Timer prescale value.
 - Enable the Timer by setting the TMRxON bit of the TxCON register.
6. Enable PWM output pin:
 - Wait until the Timer overflows and the TMRxIF bit of the PIRx register is set. See Note below.
 - Enable the CCPx pin output driver by clearing the associated TRIS bit.

Note: In order to send a complete duty cycle and period on the first PWM output, the above steps must be included in the setup sequence. If it is not critical to start with a complete PWM signal on the first output, then step 6 may be ignored.

20.3.3 TIMER2/4/6/8/10 TIMER RESOURCE

The PWM standard mode makes use of one of the 8-bit Timer2/4/6/8/10 timer resources to specify the PWM period.

Configuring the CxTSEL<1:0> bits in the CCPTMRSx register selects which Timer2/4/6/8/10 timer is used.

See [Table 20-6](#) for CCPx PWM Timer2/4/6/8/10 resources.

20.3.4 PWM PERIOD

The PWM period is specified by the PRx register of Timer2/4/6/8/10. The PWM period can be calculated using the formula of [Equation 20-1](#).

EQUATION 20-1: PWM PERIOD

$$PWM\ Period = [(PRx) + 1] \cdot 4 \cdot TOSC \cdot (TMRx\ Prescale\ Value)$$

Note 1: TOSC = 1/FOSC

When TMRx is equal to PRx, the following three events occur on the next increment cycle:

- TMRx is cleared
- The CCPx pin is set. (Exception: If the PWM duty cycle = 0%, the pin will not be set.)
- The PWM duty cycle is latched from CCPRxL into CCPRxH.

Note: The Timer postscaler (see [Section 19.1 “Timer2/4/6/8/10 Operation”](#)) is not used in the determination of the PWM frequency.

PIC16(L)F1526/7

20.3.5 PWM DUTY CYCLE

The PWM duty cycle is specified by writing a 10-bit value to multiple registers: CCPRxL register and DCxB<1:0> bits of the CCPxCON register. The CCPRxL contains the eight MSBs and the DCxB<1:0> bits of the CCPxCON register contain the two LSBs. CCPRxL and DCxB<1:0> bits of the CCPxCON register can be written to at any time. The duty cycle value is not latched into CCPRxH until after the period completes (i.e., a match between PRx and TMRx registers occurs). While using the PWM, the CCPRxH register is read-only.

Equation 20-2 is used to calculate the PWM pulse width.

Equation 20-3 is used to calculate the PWM duty cycle ratio.

EQUATION 20-2: PULSE WIDTH

$$\text{Pulse Width} = (\text{CCPRxL:CCPxCON}\langle 5:4 \rangle) \cdot T_{\text{OSC}} \cdot (\text{TMRx Prescale Value})$$

EQUATION 20-3: DUTY CYCLE RATIO

$$\text{Duty Cycle Ratio} = \frac{(\text{CCPRxL:CCPxCON}\langle 5:4 \rangle)}{4(\text{PRx} + 1)}$$

The CCPRxH register and a 2-bit internal latch are used to double buffer the PWM duty cycle. This double buffering is essential for glitchless PWM operation.

The 8-bit timer TMRx register is concatenated with either the 2-bit internal system clock (Fosc), or 2 bits of the prescaler, to create the 10-bit time base. The system clock is used if the Timer2/4/6/8/10 prescaler is set to 1:1.

When the 10-bit time base matches the CCPRxH and 2-bit latch, then the CCPx pin is cleared (see Figure 20-4).

20.3.6 PWM RESOLUTION

The resolution determines the number of available duty cycles for a given period. For example, a 10-bit resolution will result in 1024 discrete duty cycles, whereas an 8-bit resolution will result in 256 discrete duty cycles.

The maximum PWM resolution is 10 bits when PRx is 255. The resolution is a function of the PRx register value as shown by Equation 20-4.

EQUATION 20-4: PWM RESOLUTION

$$\text{Resolution} = \frac{\log[4(\text{PRx} + 1)]}{\log(2)} \text{ bits}$$

Note: If the pulse width value is greater than the period the assigned PWM pin(s) will remain unchanged.

TABLE 20-7: EXAMPLE PWM FREQUENCIES AND RESOLUTIONS (Fosc = 20 MHz)

| PWM Frequency | 1.22 kHz | 4.88 kHz | 19.53 kHz | 78.12 kHz | 156.3 kHz | 208.3 kHz |
|---------------------------|----------|----------|-----------|-----------|-----------|-----------|
| Timer Prescale (1, 4, 16) | 16 | 4 | 1 | 1 | 1 | 1 |
| PRx Value | 0xFF | 0xFF | 0xFF | 0x3F | 0x1F | 0x17 |
| Maximum Resolution (bits) | 10 | 10 | 10 | 8 | 7 | 6.6 |

TABLE 20-8: EXAMPLE PWM FREQUENCIES AND RESOLUTIONS (Fosc = 8 MHz)

| PWM Frequency | 1.22 kHz | 4.90 kHz | 19.61 kHz | 76.92 kHz | 153.85 kHz | 200.0 kHz |
|---------------------------|----------|----------|-----------|-----------|------------|-----------|
| Timer Prescale (1, 4, 16) | 16 | 4 | 1 | 1 | 1 | 1 |
| PRx Value | 0x65 | 0x65 | 0x65 | 0x19 | 0x0C | 0x09 |
| Maximum Resolution (bits) | 8 | 8 | 8 | 6 | 5 | 5 |

20.3.7 OPERATION IN SLEEP MODE

In Sleep mode, the TMRx register will not increment and the state of the module will not change. If the CCPx pin is driving a value, it will continue to drive that value. When the device wakes up, TMRx will continue from its previous state.

20.3.8 CHANGES IN SYSTEM CLOCK FREQUENCY

The PWM frequency is derived from the system clock frequency. Any changes in the system clock frequency will result in changes to the PWM frequency. See [Section 5.0 “Oscillator Module \(with Fail-Safe Clock Monitor\)”](#) for additional details.

20.3.9 EFFECTS OF RESET

Any Reset will force all ports to Input mode and the CCP registers to their Reset states.

20.3.10 ALTERNATE PIN LOCATIONS

This module incorporates I/O pins that can be moved to other locations with the use of the alternate pin function register, APFCON. To determine which pins can be moved and what their default locations are upon a reset, see [Section 12.1 “Alternate Pin Function”](#) for more information.

TABLE 20-9: SUMMARY OF REGISTERS ASSOCIATED WITH STANDARD PWM

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Register on Page |
|----------|-------------------------|--------------|------------|--------|-------------|-------------|----------|---------|------------------|
| APFCON | — | — | — | — | — | — | T3CKISEL | CCP2SEL | 112 |
| CCP1CON | — | — | DC1B<1:0> | | CCP1M<3:0> | | | | 189 |
| CCP2CON | — | — | DC2B<1:0> | | CCP2M<3:0> | | | | 189 |
| CCP3CON | — | — | DC3B<1:0> | | CCP3M<3:0> | | | | 189 |
| CCP4CON | — | — | DC4B<1:0> | | CCP4M<3:0> | | | | 189 |
| CCP5CON | — | — | DC5B<1:0> | | CCP5M<3:0> | | | | 189 |
| CCP6CON | — | — | DC6B<1:0> | | CCP6M<3:0> | | | | 189 |
| CCP7CON | — | — | DC7B<1:0> | | CCP7M<3:0> | | | | 189 |
| CCP8CON | — | — | DC8B<1:0> | | CCP8M<3:0> | | | | 189 |
| CCP9CON | — | — | DC9B<1:0> | | CCP9M<3:0> | | | | 189 |
| CCP10CON | — | — | DC10B<1:0> | | CCP10M<3:0> | | | | 189 |
| INTCON | GIE | PEIE | TMR0IE | INTE | IOCIE | TMR0IF | INTF | IOCIF | 76 |
| PIE1 | TMR1GIE | ADIE | RC1IE | TX1IE | SSP1IE | CCP1IE | TMR2IE | TMR1IE | 77 |
| PIE2 | OSFIE | TMR5GIE | TMR3GIE | — | BCL1IE | TMR10IE | TMR8IE | CCP2IE | 78 |
| PIE3 | CCP6IE | CCP5IE | CCP4IE | CCP3IE | TMR6IE | TMR5IE | TMR4IE | TMR3IE | 79 |
| PIE4 | CCP10IE | CCP9IE | RC2IE | TX2IE | CCP8IE | CCP7IE | BCL2IE | SSP2IE | 80 |
| PIR1 | TMR1GIF | ADIF | RC1IF | TX1IF | SSP1IF | CCP1IF | TMR2IF | TMR1IF | 81 |
| PIR2 | OSFIF | TMR5GIF | TMR3GIF | — | BCL1IF | TMR10IF | TMR8IF | CCP2IF | 82 |
| PIR3 | CCP6IF | CCP5IF | CCP4IF | CCP3IF | TMR6IF | TMR5IF | TMR4IF | TMR3IF | 83 |
| PIR4 | CCP10IF | CCP9IF | RC2IF | TX2IF | CCP8IF | CCP7IF | BCL2IF | SSP2IF | 84 |
| PR2 | Timer2 Period Register | | | | | | | | 171* |
| PR4 | Timer4 Period Register | | | | | | | | 171* |
| PR6 | Timer6 Period Register | | | | | | | | 171* |
| PR8 | Timer8 Period Register | | | | | | | | 171* |
| PR10 | Timer10 Period Register | | | | | | | | 171* |
| T2CON | — | T2OUTPS<3:0> | | | TMR2ON | T2CKPS<:0>1 | | | 168 |
| T4CON | — | T4OUTPS<3:0> | | | TMR4ON | T4CKPS<:0>1 | | | 168 |
| T6CON | — | T6OUTPS<3:0> | | | TMR6ON | T6CKPS<:0>1 | | | 168 |

Legend: — = Unimplemented location, read as '0'. Shaded cells are not used by the PWM.

* Page provides register information.

PIC16(L)F1526/7

TABLE 20-10: SUMMARY OF REGISTERS ASSOCIATED WITH STANDARD PWM (CONTINUED)

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Register on Page |
|--------|-------------------------|---------------|--------|--------|--------|---------|--------------|--------|------------------|
| T8CON | — | T8OUTPS<3:0> | | | | TMR8ON | T8CKPS<:0>1 | | 168 |
| T10CON | — | T10OUTPS<3:0> | | | | TMR10ON | T10CKPS<:0>1 | | 168 |
| TMR2 | Timer2 Module Register | | | | | | | | 171* |
| TMR4 | Timer4 Module Register | | | | | | | | 171* |
| TMR6 | Timer6 Module Register | | | | | | | | 171* |
| TMR8 | Timer8 Module Register | | | | | | | | 171* |
| TMR10 | Timer10 Module Register | | | | | | | | 171* |
| TRISA | TRISA7 | TRISA6 | TRISA5 | TRISA4 | TRISA3 | TRISA2 | TRISA1 | TRISA0 | 114 |

Legend: — = Unimplemented location, read as '0'. Shaded cells are not used by the PWM.

* Page provides register information.

20.4 Register Definitions: ECCP Control

REGISTER 20-1: CCPxCON: CCPx CONTROL REGISTER

| U-0 | U-0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | |
|-------|-----|-----------|---------|------------|---------|---------|---------|-------|
| — | — | DCxB<1:0> | | CCPxM<3:0> | | | | |
| bit 7 | | | | | | | | bit 0 |

Legend:

| | | |
|----------------------|----------------------|--|
| R = Readable bit | W = Writable bit | U = Unimplemented bit, read as '0' |
| u = Bit is unchanged | x = Bit is unknown | -n/n = Value at POR and BOR/Value at all other Reset |
| '1' = Bit is set | '0' = Bit is cleared | |

bit 7-6 **Unimplemented:** Read as '0'

bit 5-4 **DCxB<1:0>:** PWM Duty Cycle Least Significant bits

Capture mode:

Unused

Compare mode:

Unused

PWM mode:

These bits are the two LSBs of the PWM duty cycle. The eight MSBs are found in CCPxL.

bit 3-0 **CCPxM<3:0>:** CCPx Mode Select bits

11xx = PWM mode

1011 = Compare mode: Special Event Trigger (sets CCP10IF bit (CCP10), starts ADC conversion if ADC module is enabled)⁽¹⁾

1010 = Compare mode: generate software interrupt only

1001 = Compare mode: clear output on compare match (set CCPxIF)

1000 = Compare mode: set output on compare match (set CCPxIF)

0111 = Capture mode: every 16th rising edge

0110 = Capture mode: every 4th rising edge

0101 = Capture mode: every rising edge

0100 = Capture mode: every falling edge

0011 = Reserved

0010 = Compare mode: toggle output on match

0001 = Reserved

0000 = Capture/Compare/PWM off (resets CCPx module)

PIC16(L)F1526/7

REGISTER 20-2: CCPTMRS0: CCP TIMER SELECTION CONTROL REGISTER 0

| | | | | | | | |
|-------------|---------|-------------|---------|-------------|---------|-------------|---------|
| R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 |
| C4TSEL<1:0> | | C3TSEL<1:0> | | C2TSEL<1:0> | | C1TSEL<1:0> | |
| bit 7 | | | | | | | bit 0 |

Legend:

| | | |
|----------------------|----------------------|---|
| R = Readable bit | W = Writable bit | U = Unimplemented bit, read as '0' |
| u = Bit is unchanged | x = Bit is unknown | -n/n = Value at POR and BOR/Value at all other Resets |
| '1' = Bit is set | '0' = Bit is cleared | |

bit 7-6 **C4TSEL<1:0>**: CCP4 Timer Selection bits
When in Capture/Compare mode:
 x1 = CCP4 is based off Timer3 in Capture/Compare mode
 x0 = CCP4 is based off Timer1 in Capture/Compare mode
When in PWM mode:
 11 = Reserved
 10 = CCP4 is based off Timer6 in PWM mode
 01 = CCP4 is based off Timer4 in PWM mode
 00 = CCP4 is based off Timer2 in PWM mode

bit 5-4 **C3TSEL<1:0>**: CCP3 Timer Selection bits
When in Capture/Compare mode:
 x1 = CCP3 is based off Timer3 in Capture/Compare mode
 x0 = CCP3 is based off Timer1 in Capture/Compare mode
When in PWM mode:
 11 = Reserved
 10 = CCP3 is based off Timer6 in PWM mode
 01 = CCP3 is based off Timer4 in PWM mode
 00 = CCP3 is based off Timer2 in PWM mode

bit 3-2 **C2TSEL<1:0>**: CCP2 Timer Selection bits
When in Capture/Compare mode:
 x1 = CCP2 is based off Timer3 in Capture/Compare mode
 x0 = CCP2 is based off Timer1 in Capture/Compare mode
When in PWM mode:
 11 = Reserved
 10 = CCP2 is based off Timer6 in PWM mode
 01 = CCP2 is based off Timer4 in PWM mode
 00 = CCP2 is based off Timer2 in PWM mode

bit 1-0 **C1TSEL<1:0>**: CCP1 Timer Selection bits
When in Capture/Compare mode:
 x1 = CCP1 is based off Timer3 in Capture/Compare mode
 x0 = CCP1 is based off Timer1 in Capture/Compare mode
When in PWM mode:
 11 = Reserved
 10 = CCP1 is based off Timer6 in PWM mode
 01 = CCP1 is based off Timer4 in PWM mode
 00 = CCP1 is based off Timer2 in PWM mode

REGISTER 20-3: CCPTMRS1: CCP TIMER SELECTION CONTROL REGISTER 1

| R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 |
|-------------|---------|-------------|---------|-------------|---------|-------------|---------|
| C8TSEL<1:0> | | C7TSEL<1:0> | | C6TSEL<1:0> | | C5TSEL<1:0> | |
| bit 7 | | | | | | | bit 0 |

Legend:

| | | |
|----------------------|----------------------|---|
| R = Readable bit | W = Writable bit | U = Unimplemented bit, read as '0' |
| u = Bit is unchanged | x = Bit is unknown | -n/n = Value at POR and BOR/Value at all other Resets |
| '1' = Bit is set | '0' = Bit is cleared | |

- bit 7-6 **C8TSEL<1:0>**: CCP8 Timer Selection bits
When in Capture/Compare mode:
 x1 = CCP8 is based off Timer5 in Capture/Compare mode
 x0 = CCP8 is based off Timer1 in Capture/Compare mode
When in PWM mode:
 11 = Reserved
 10 = CCP8 is based off Timer10 in PWM mode
 01 = CCP8 is based off Timer8 in PWM mode
 00 = CCP8 is based off Timer2 in PWM mode
- bit 5-4 **C7TSEL<1:0>**: CCP7 Timer Selection bits
When in Capture/Compare mode:
 x1 = CCP7 is based off Timer5 in Capture/Compare mode
 x0 = CCP7 is based off Timer1 in Capture/Compare mode
When in PWM mode:
 11 = Reserved
 10 = CCP7 is based off Timer8 in PWM mode
 01 = CCP7 is based off Timer6 in PWM mode
 00 = CCP7 is based off Timer2 in PWM mode
- bit 3-2 **C6TSEL<1:0>**: CCP6 Timer Selection bits
When in Capture/Compare mode:
 x1 = CCP6 is based off Timer5 in Capture/Compare mode
 x0 = CCP6 is based off Timer1 in Capture/Compare mode
When in PWM mode:
 11 = Reserved
 10 = CCP6 is based off Timer8 in PWM mode
 01 = CCP6 is based off Timer6 in PWM mode
 00 = CCP6 is based off Timer2 in PWM mode
- bit 1-0 **C5TSEL<1:0>**: CCP5 Timer Selection bits
When in Capture/Compare mode:
 x1 = CCP5 is based off Timer5 in Capture/Compare mode
 x0 = CCP5 is based off Timer1 in Capture/Compare mode
When in PWM mode:
 11 = Reserved
 10 = CCP5 is based off Timer8 in PWM mode
 01 = CCP5 is based off Timer6 in PWM mode
 00 = CCP5 is based off Timer2 in PWM mode

PIC16(L)F1526/7

REGISTER 20-4: CCPTMRS2: CCP TIMER SELECTION CONTROL REGISTER 2

| | | | | | | | |
|-------|-----|-----|-----|--------------|---------|-------------|---------|
| U-0 | U-0 | U-0 | U-0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 |
| — | — | — | — | C10TSEL<1:0> | | C9TSEL<1:0> | |
| bit 7 | | | | | | | bit 0 |

Legend:

| | | |
|----------------------|----------------------|---|
| R = Readable bit | W = Writable bit | U = Unimplemented bit, read as '0' |
| u = Bit is unchanged | x = Bit is unknown | -n/n = Value at POR and BOR/Value at all other Resets |
| '1' = Bit is set | '0' = Bit is cleared | |

bit 7-4 **Unimplemented:** Read as '0'

bit 3-2 **C10TSEL<1:0>:** CCP10 Timer Selection bits

When in Capture/Compare mode:

x1 = CCP10 is based off Timer5 in Capture/Compare mode

x0 = CCP10 is based off Timer1 in Capture/Compare mode

When in PWM mode:

11 = Reserved

10 = CCP10 is based off Timer10 in PWM mode

01 = CCP10 is based off Timer8 in PWM mode

00 = CCP10 is based off Timer2 in PWM mode

bit 1-0 **C9TSEL<1:0>:** CCP9 Timer Selection bits

When in Capture/Compare mode:

x1 = CCP9 is based off Timer5 in Capture/Compare mode

x0 = CCP9 is based off Timer1 in Capture/Compare mode

When in PWM mode:

11 = Reserved

10 = CCP9 is based off Timer10 in PWM mode

01 = CCP9 is based off Timer8 in PWM mode

00 = CCP9 is based off Timer2 in PWM mode

PIC16(L)F1526/7

The I²C interface supports the following modes and features:

- Master mode
- Slave mode
- Byte NACKing (Slave mode)
- Limited Multi-master support
- 7-bit and 10-bit addressing
- Start and Stop interrupts
- Interrupt masking
- Clock stretching
- Bus collision detection
- General call address matching
- Address masking
- Address Hold and Data Hold modes
- Selectable SDAx hold times

Figure 21-2 is a block diagram of the I²C Interface module in Master mode. Figure 21-3 is a diagram of the I²C interface module in Slave mode.

The PIC16F1527 has two MSSP modules, MSSP1 and MSSP2, each module operating independently from the other.

Note 1: In devices with more than one MSSP module, it is very important to pay close attention to SSPxCONx register names. SSP1CON1 and SSP1CON2 registers control different operational aspects of the same module, while SSP1CON1 and SSP2CON1 control the same features for two different modules.

2: Throughout this section, generic references to an MSSP module in any of its operating modes may be interpreted as being equally applicable to MSSP1 or MSSP2. Register names, module I/O signals, and bit names may use the generic designator 'x' to indicate the use of a numeral to distinguish a particular module when required.

FIGURE 21-2: MSSPX BLOCK DIAGRAM (I²C MASTER MODE)

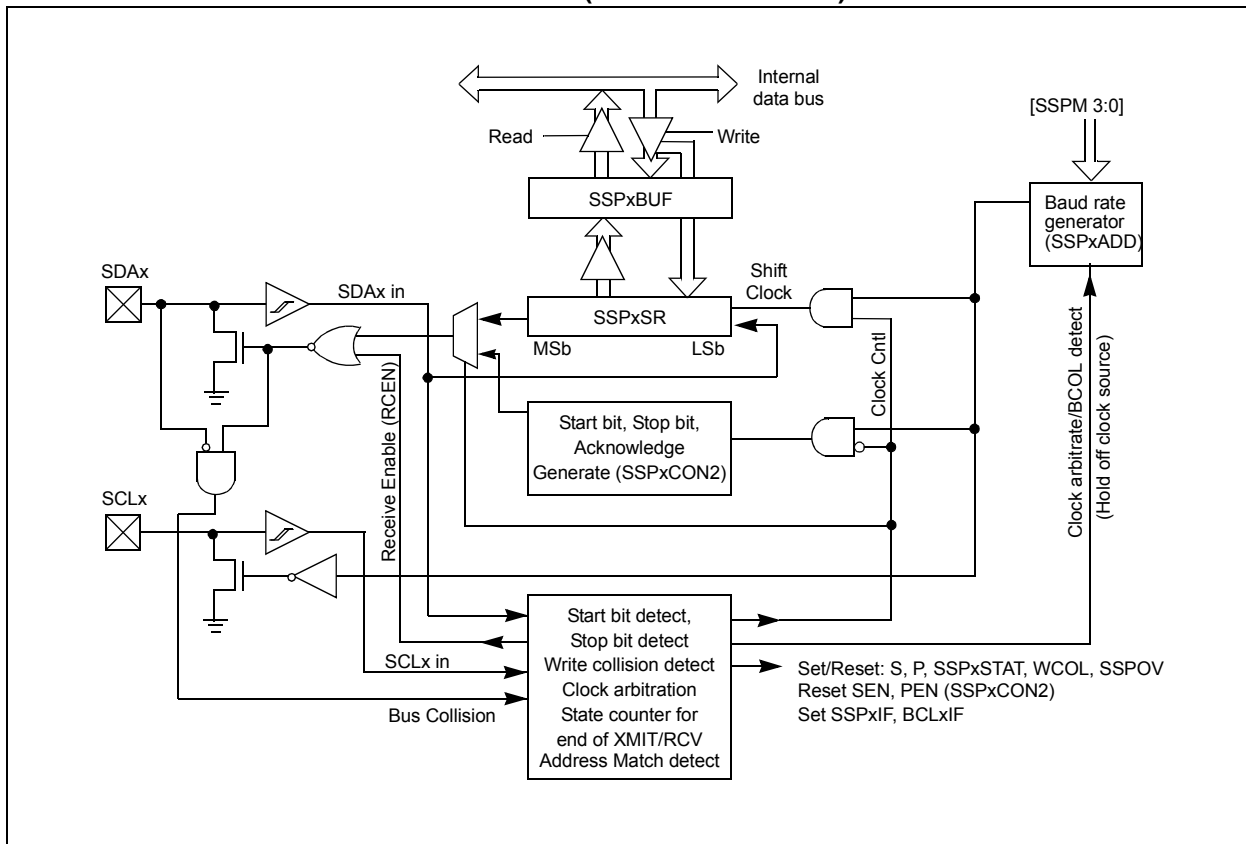
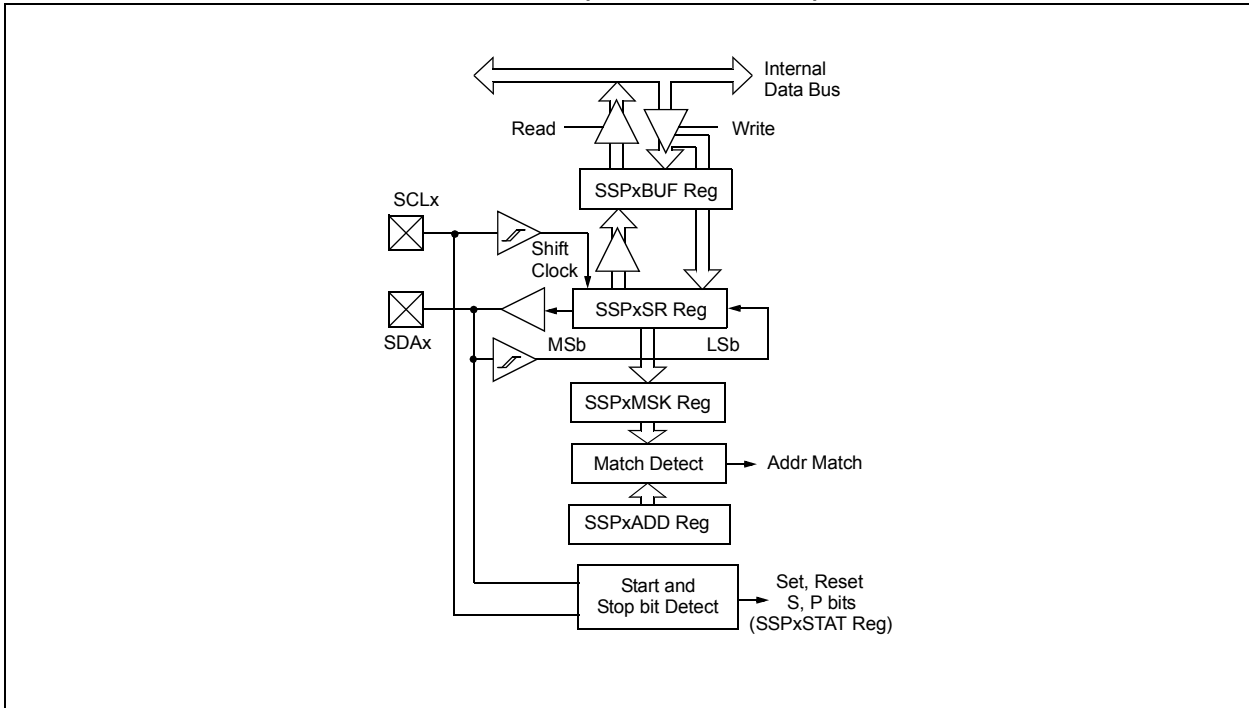


FIGURE 21-3: MSSPX BLOCK DIAGRAM (I²C SLAVE MODE)



21.2 SPI Mode Overview

The Serial Peripheral Interface (SPI) bus is a synchronous serial data communication bus that operates in Full-Duplex mode. Devices communicate in a master/slave environment where the master device initiates the communication. A slave device is controlled through a chip select known as Slave Select.

The SPI bus specifies four signal connections:

- Serial Clock (SCKx)
- Serial Data Out (SDOx)
- Serial Data In (SDIx)
- Slave Select (\overline{SSx})

Figure 21-1 shows the block diagram of the MSSPx module when operating in SPI mode.

The SPI bus operates with a single master device and one or more slave devices. When multiple slave devices are used, an independent Slave Select connection is required from the master device to each slave device.

Figure 21-4 shows a typical connection between a master device and multiple slave devices.

The master selects only one slave at a time. Most slave devices have tri-state outputs so their output signal appears disconnected from the bus when they are not selected.

Transmissions involve two shift registers, eight bits in size, one in the master and one in the slave. With either the master or the slave device, data is always shifted out one bit at a time, with the Most Significant bit (MSb) shifted out first. At the same time, a new Least Significant bit (LSb) is shifted into the same register.

Figure 21-5 shows a typical connection between two processors configured as master and slave devices.

Data is shifted out of both shift registers on the programmed clock edge and latched on the opposite edge of the clock.

The master device transmits information out on its SDOx output pin which is connected to, and received by, the slave's SDIx input pin. The slave device transmits information out on its SDOx output pin, which is connected to, and received by, the master's SDIx input pin.

To begin communication, the master device first sends out the clock signal. Both the master and the slave devices should be configured for the same clock polarity.

The master device starts a transmission by sending out the MSb from its shift register. The slave device reads this bit from that same line and saves it into the LSb position of its shift register.

During each SPI clock cycle, a full-duplex data transmission occurs. This means that while the master device is sending out the MSb from its shift register (on

its SDOx pin) and the slave device is reading this bit and saving it as the LSb of its shift register, that the slave device is also sending out the MSb from its shift register (on its SDOx pin) and the master device is reading this bit and saving it as the LSb of its shift register.

After 8 bits have been shifted out, the master and slave have exchanged register values.

If there is more data to exchange, the shift registers are loaded with new data and the process repeats itself.

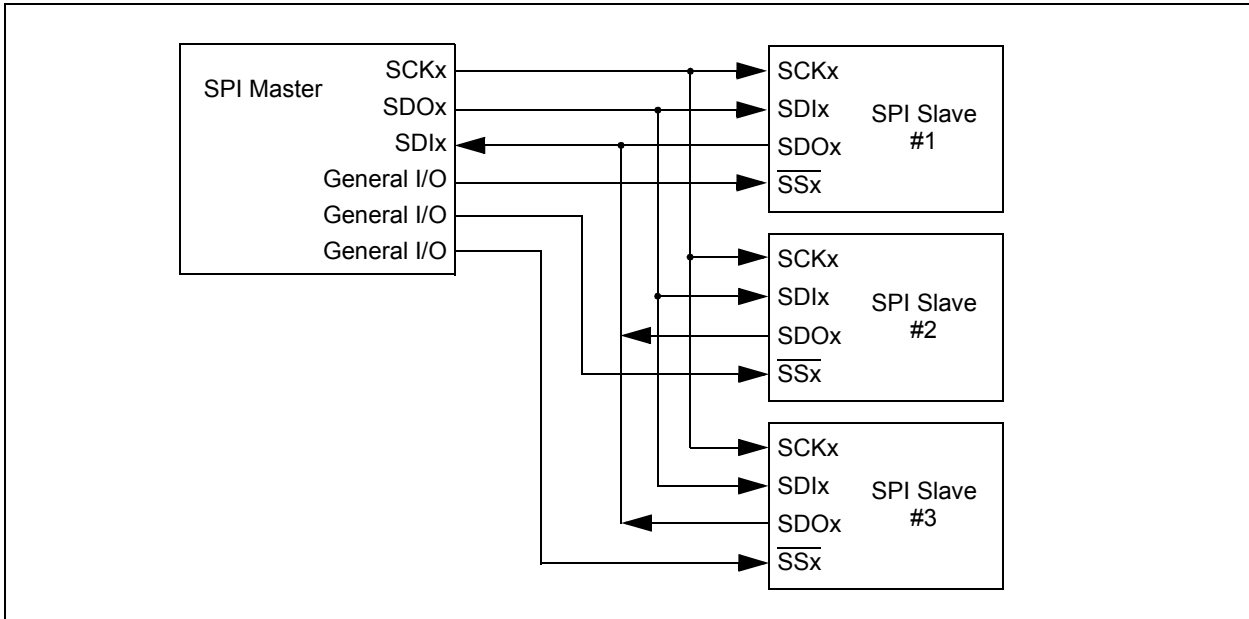
Whether the data is meaningful or not (dummy data), depends on the application software. This leads to three scenarios for data transmission:

- Master sends useful data and slave sends dummy data.
- Master sends useful data and slave sends useful data.
- Master sends dummy data and slave sends useful data.

Transmissions may involve any number of clock cycles. When there is no more data to be transmitted, the master stops sending the clock signal and it deselects the slave.

Every slave device connected to the bus that has not been selected through its slave select line must disregard the clock and transmission signals and must not transmit out any data of its own.

FIGURE 21-4: SPI MASTER AND MULTIPLE SLAVE CONNECTION



21.2.1 SPI MODE REGISTERS

The MSSPx module has five registers for SPI mode operation. These are:

- MSSPx STATUS register (SSPxSTAT)
- MSSPx Control Register 1 (SSPxCON1)
- MSSPx Control Register 3 (SSPxCON3)
- MSSPx Data Buffer register (SSPxBUF)
- MSSPx Address register (SSPxADD)
- MSSPx Shift register (SSPxSR)
(Not directly accessible)

SSPxCON1 and SSPxSTAT are the control and STATUS registers in SPI mode operation. The SSPxCON1 register is readable and writable. The lower 6 bits of the SSPxSTAT are read-only. The upper two bits of the SSPxSTAT are read/write.

In one SPI master mode, SSPxADD can be loaded with a value used in the Baud Rate Generator. More information on the Baud Rate Generator is available in [Section 21.7 “Baud Rate Generator”](#).

SSPxSR is the shift register used for shifting data in and out. SSPxBUF provides indirect access to the SSPxSR register. SSPxBUF is the buffer register to which data bytes are written, and from which data bytes are read.

In receive operations, SSPxSR and SSPxBUF together create a buffered receiver. When SSPxSR receives a complete byte, it is transferred to SSPxBUF and the SSPxIF interrupt is set.

During transmission, the SSPxBUF is not buffered. A write to SSPxBUF will write to both SSPxBUF and SSPxSR.

PIC16(L)F1526/7

21.2.2 SPI MODE OPERATION

When initializing the SPI, several options need to be specified. This is done by programming the appropriate control bits (SSPxCON1<5:0> and SSPxSTAT<7:6>). These control bits allow the following to be specified:

- Master mode (SCKx is the clock output)
- Slave mode (SCKx is the clock input)
- Clock Polarity (Idle state of SCKx)
- Data Input Sample Phase (middle or end of data output time)
- Clock Edge (output data on rising/falling edge of SCKx)
- Clock Rate (Master mode only)
- Slave Select mode (Slave mode only)

To enable the serial port, SSPx Enable bit, SSPEN of the SSPxCON1 register, must be set. To reset or reconfigure SPI mode, clear the SSPEN bit, re-initialize the SSPxCONx registers and then set the SSPEN bit. This configures the SDIx, SDOx, SCKx and \overline{SSx} pins as serial port pins. For the pins to behave as the serial port function, some must have their data direction bits (in the TRIS register) appropriately programmed as follows:

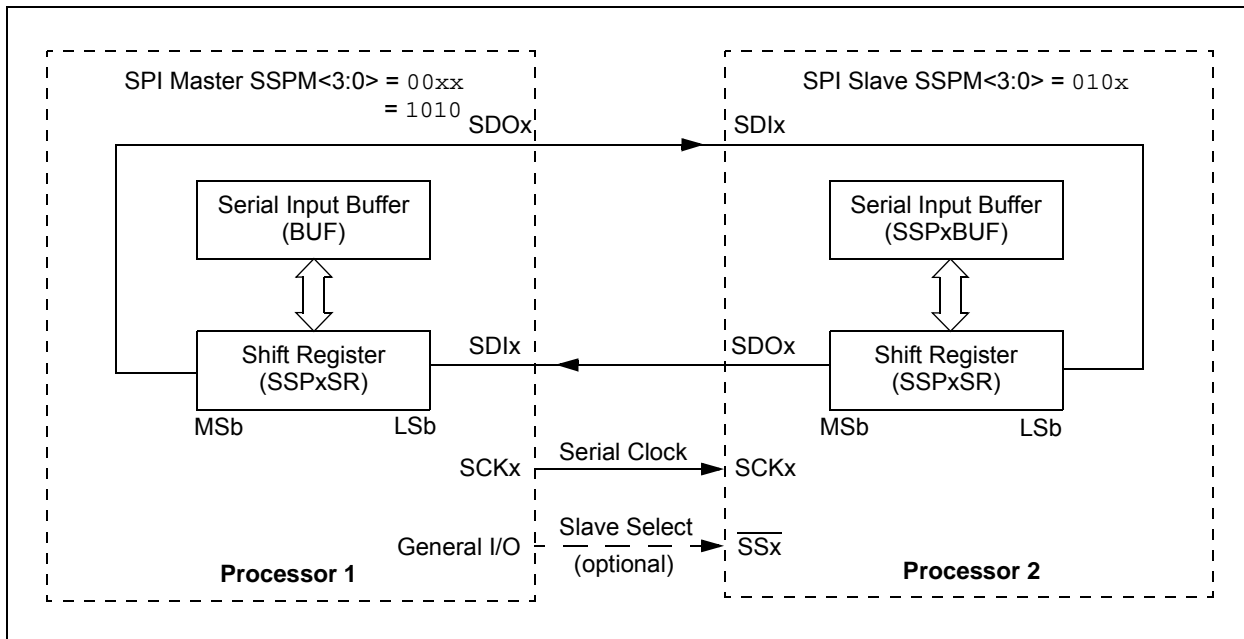
- SDIx must have corresponding TRIS bit set
- SDOx must have corresponding TRIS bit cleared
- SCKx (Master mode) must have corresponding TRIS bit cleared
- SCKx (Slave mode) must have corresponding TRIS bit set
- \overline{SSx} must have corresponding TRIS bit set

Any serial port function that is not desired may be overridden by programming the corresponding data direction (TRIS) register to the opposite value.

The MSSPx consists of a transmit/receive shift register (SSPxSR) and a buffer register (SSPxBUF). The SSPxSR shifts the data in and out of the device, MSb first. The SSPxBUF holds the data that was written to the SSPxSR until the received data is ready. Once the 8 bits of data have been received, that byte is moved to the SSPxBUF register. Then, the Buffer Full Detect bit, BF of the SSPxSTAT register, and the interrupt flag bit, SSPxIF, are set. This double-buffering of the received data (SSPxBUF) allows the next byte to start reception before reading the data that was just received. Any write to the SSPxBUF register during transmission/reception of data will be ignored and the write collision detect bit WCOL of the SSPxCON1 register, will be set. User software must clear the WCOL bit to allow the following write(s) to the SSPxBUF register to complete successfully.

When the application software is expecting to receive valid data, the SSPxBUF should be read before the next byte of data to transfer is written to the SSPxBUF. The Buffer Full bit, BF of the SSPxSTAT register, indicates when SSPxBUF has been loaded with the received data (transmission is complete). When the SSPxBUF is read, the BF bit is cleared. This data may be irrelevant if the SPI is only a transmitter. Generally, the MSSPx interrupt is used to determine when the transmission/reception has completed. If the interrupt method is not going to be used, then software polling can be done to ensure that a write collision does not occur.

FIGURE 21-5: SPI MASTER/SLAVE CONNECTION



21.2.3 SPI MASTER MODE

The master can initiate the data transfer at any time because it controls the SCKx line. The master determines when the slave (Processor 2, [Figure 21-5](#)) is to broadcast data by the software protocol.

In Master mode, the data is transmitted/received as soon as the SSPxBUF register is written to. If the SPI is only going to receive, the SDOx output could be disabled (programmed as an input). The SSPxSR register will continue to shift in the signal present on the SDIx pin at the programmed clock rate. As each byte is received, it will be loaded into the SSPxBUF register as if a normal received byte (interrupts and Status bits appropriately set).

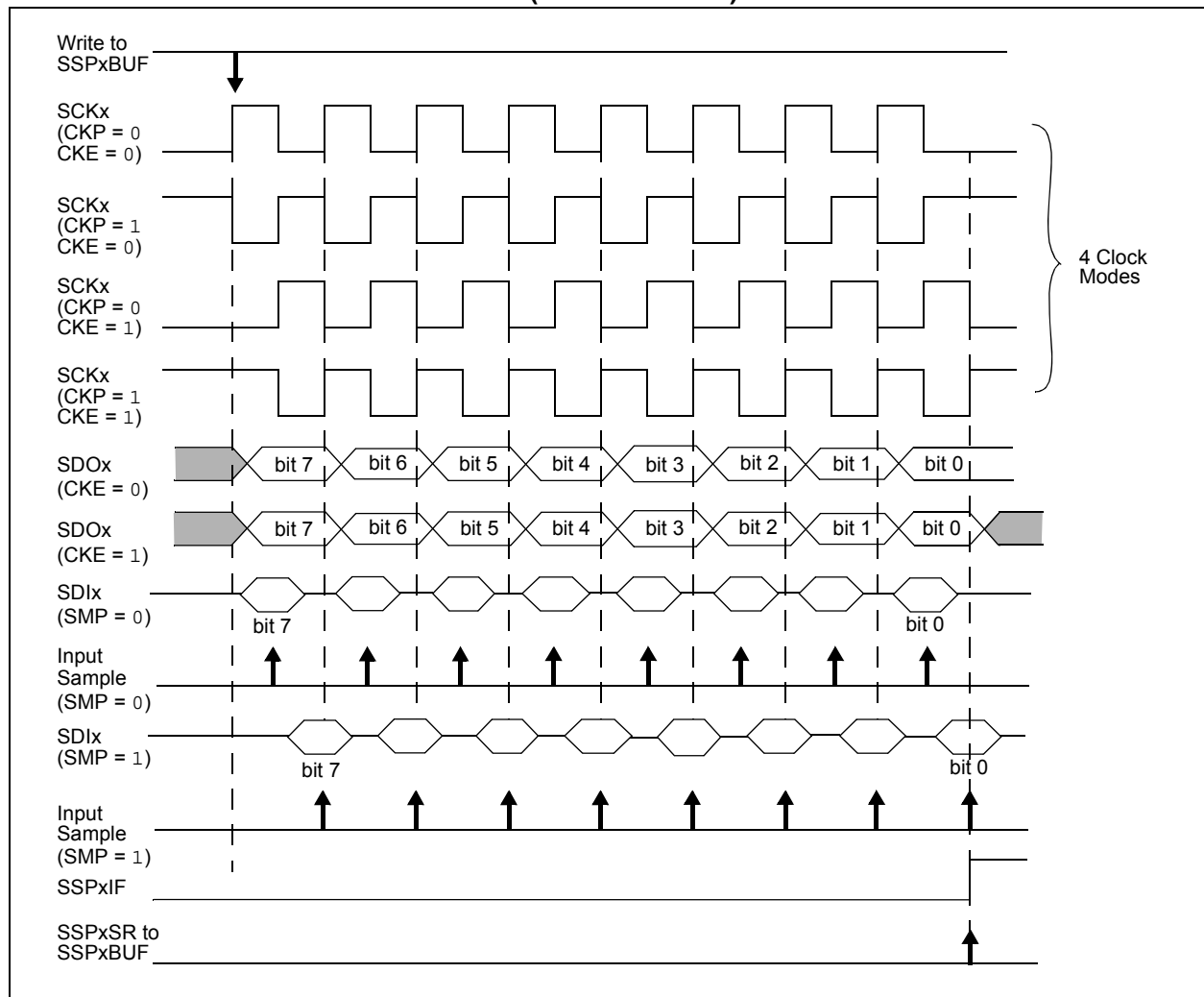
The clock polarity is selected by appropriately programming the CKP bit of the SSPxCON1 register and the CKE bit of the SSPxSTAT register. This then, would give waveforms for SPI communication as shown in [Figure 21-6](#), [Figure 21-8](#), [Figure 21-9](#) and [Figure 21-10](#), where the MSb is transmitted first. In Master mode, the SPI clock rate (bit rate) is user programmable to be one of the following:

- $F_{osc}/4$ (or T_{CY})
- $F_{osc}/16$ (or $4 * T_{CY}$)
- $F_{osc}/64$ (or $16 * T_{CY}$)
- Timer2 output/2
- $F_{osc}/(4 * (SSPxADD + 1))$

[Figure 21-6](#) shows the waveforms for Master mode.

When the CKE bit is set, the SDOx data is valid before there is a clock edge on SCKx. The change of the input sample is shown based on the state of the SMP bit. The time when the SSPxBUF is loaded with the received data is shown.

FIGURE 21-6: SPI MODE WAVEFORM (MASTER MODE)



PIC16(L)F1526/7

21.2.4 SPI SLAVE MODE

In Slave mode, the data is transmitted and received as external clock pulses appear on SCKx. When the last bit is latched, the SSPxIF interrupt flag bit is set.

Before enabling the module in SPI Slave mode, the clock line must match the proper Idle state. The clock line can be observed by reading the SCKx pin. The Idle state is determined by the CKP bit of the SSPxCON1 register.

While in Slave mode, the external clock is supplied by the external clock source on the SCKx pin. This external clock must meet the minimum high and low times as specified in the electrical specifications.

While in Sleep mode, the slave can transmit/receive data. The shift register is clocked from the SCKx pin input and when a byte is received, the device will generate an interrupt. If enabled, the device will wake-up from Sleep.

21.2.4.1 Daisy-Chain Configuration

The SPI bus can sometimes be connected in a daisy-chain configuration. The first slave output is connected to the second slave input, the second slave output is connected to the third slave input, and so on. The final slave output is connected to the master input. Each slave sends out, during a second group of clock pulses, an exact copy of what was received during the first group of clock pulses. The whole chain acts as one large communication shift register. The daisy-chain feature only requires a single Slave Select line from the master device.

Figure 21-7 shows the block diagram of a typical daisy-chain connection when operating in SPI mode.

In a daisy-chain configuration, only the most recent byte on the bus is required by the slave. Setting the BOEN bit of the SSPxCON3 register will enable writes to the SSPxBUF register, even if the previous byte has not been read. This allows the software to ignore data that may not apply to it.

21.2.5 SLAVE SELECT SYNCHRONIZATION

The Slave Select can also be used to synchronize communication. The Slave Select line is held high until the master device is ready to communicate. When the Slave Select line is pulled low, the slave knows that a new transmission is starting.

If the slave fails to receive the communication properly, it will be reset at the end of the transmission, when the Slave Select line returns to a high state. The slave is then ready to receive a new transmission when the Slave Select line is pulled low again. If the Slave Select line is not used, there is a risk that the slave will eventually become out of sync with the master. If the slave misses a bit, it will always be one bit off in future transmissions. Use of the Slave Select line allows the slave and master to align themselves at the beginning of each transmission.

The \overline{SSx} pin allows a Synchronous Slave mode. The SPI must be in Slave mode with \overline{SSx} pin control enabled (SSPxCON1<3:0> = 0100).

When the \overline{SSx} pin is low, transmission and reception are enabled and the SDOx pin is driven.

When the \overline{SSx} pin goes high, the SDOx pin is no longer driven, even if in the middle of a transmitted byte and becomes a floating output. External pull-up/pull-down resistors may be desirable depending on the application.

- | |
|--|
| <p>Note 1: When the SPI is in Slave mode with \overline{SSx} pin control enabled (SSPxCON1<3:0> = 0100), the SPI module will reset if the \overline{SSx} pin is set to VDD.</p> <p>2: When the SPI is used in Slave mode with CKE set; the user must enable \overline{SSx} pin control.</p> <p>3: While operated in SPI Slave mode the SMP bit of the SSPxSTAT register must remain clear.</p> |
|--|

When the SPI module resets, the bit counter is forced to '0'. This can be done by either forcing the \overline{SSx} pin to a high level or clearing the SSPEN bit.

FIGURE 21-7: SPI DAISY-CHAIN CONNECTION

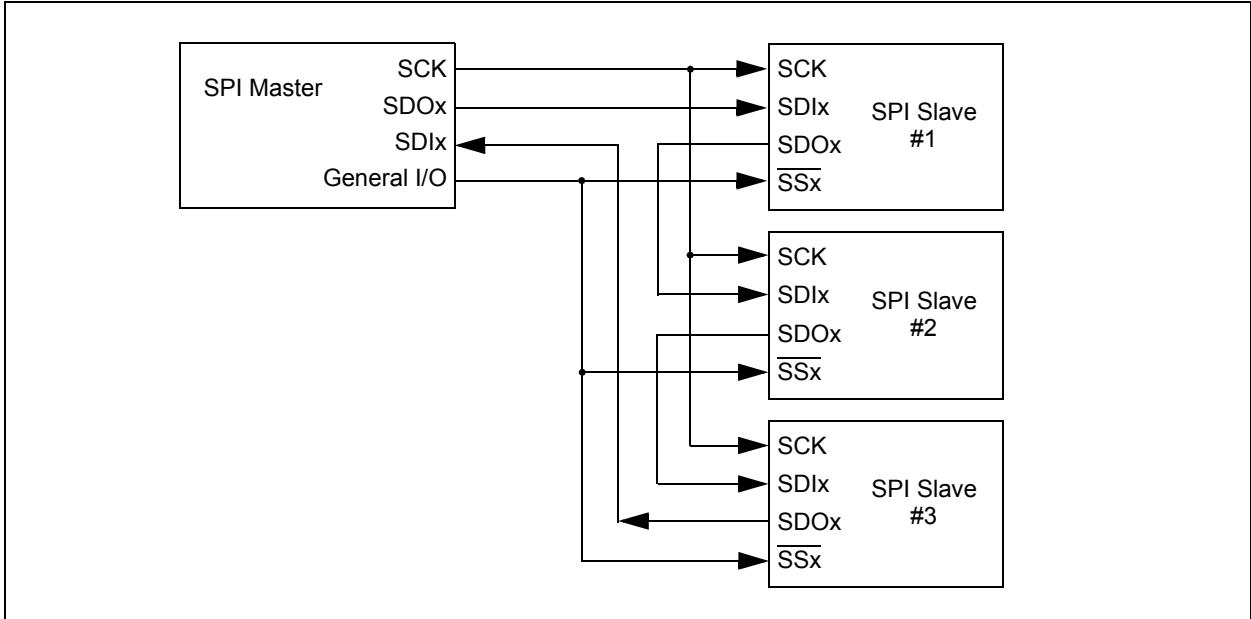
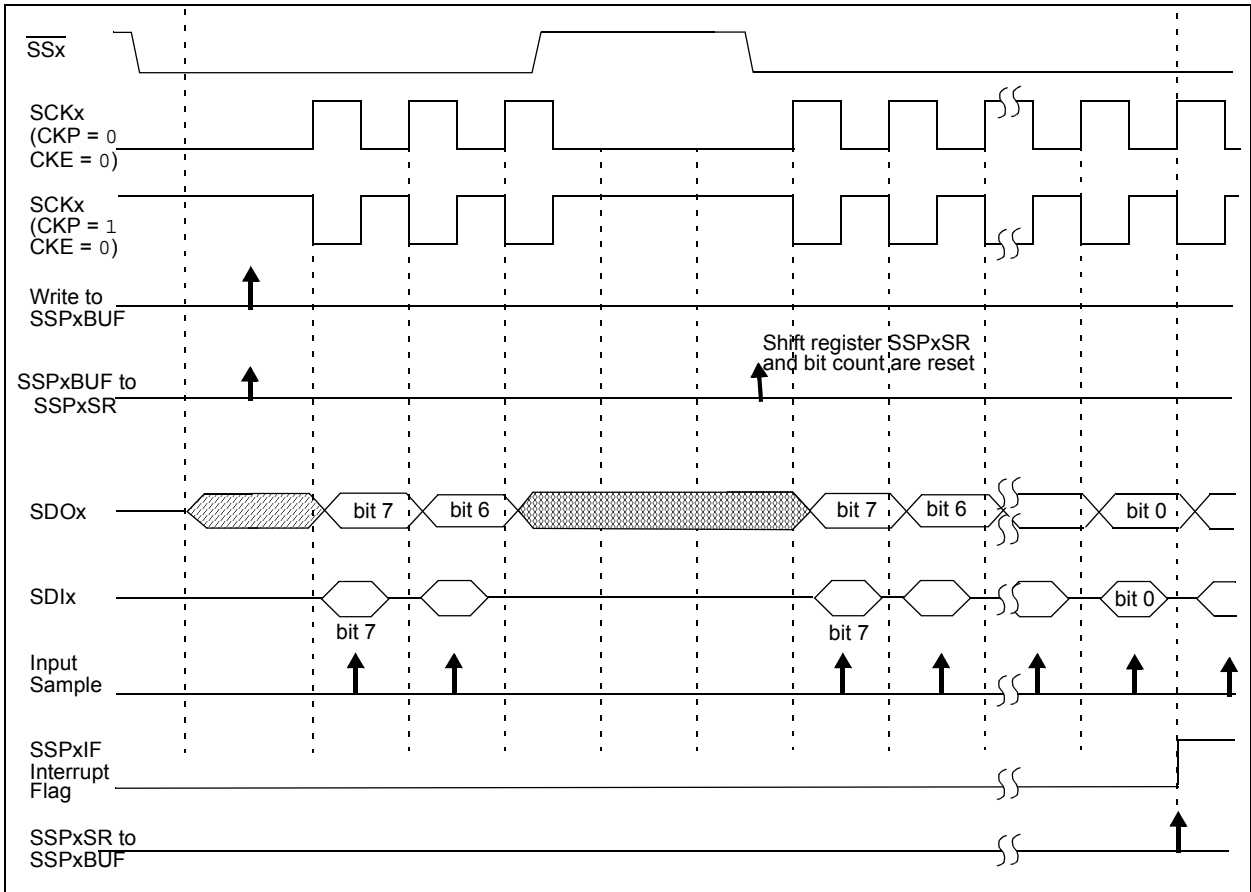


FIGURE 21-8: SLAVE SELECT SYNCHRONOUS WAVEFORM



PIC16(L)F1526/7

FIGURE 21-9: SPI MODE WAVEFORM (SLAVE MODE WITH CKE = 0)

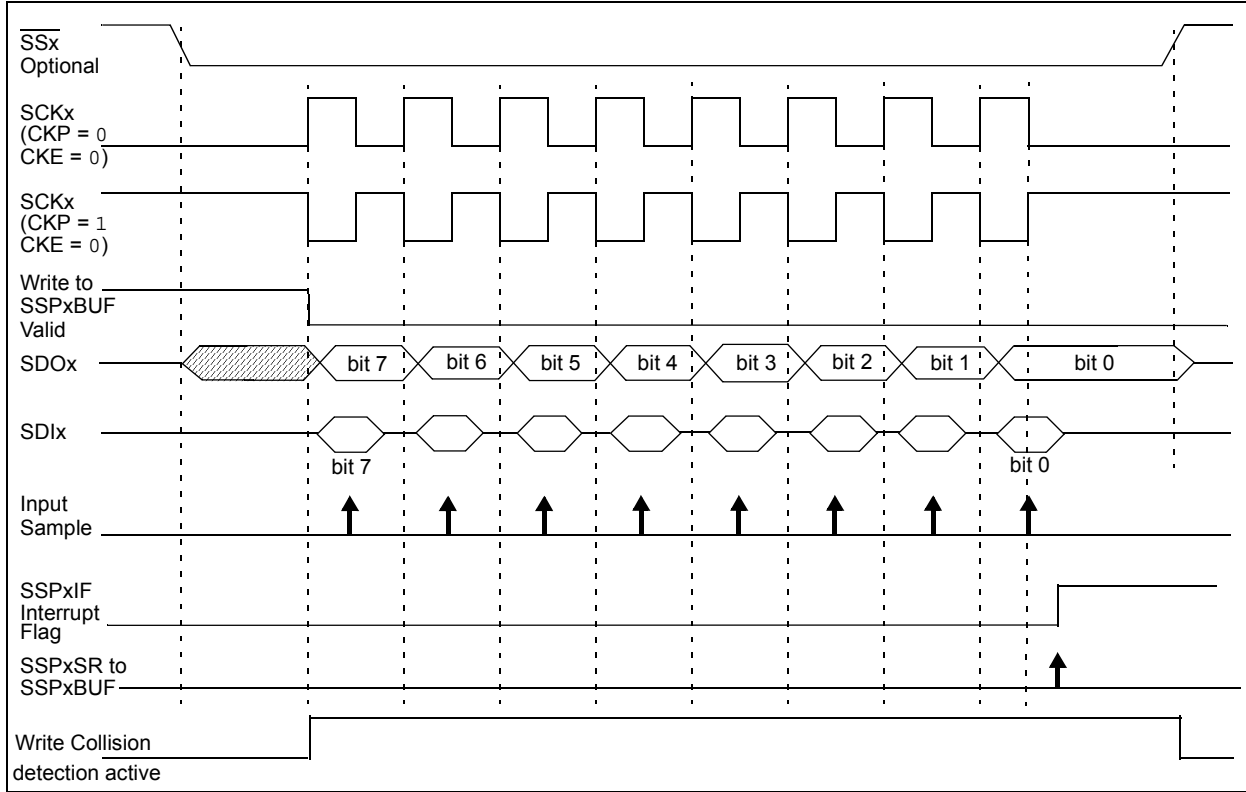
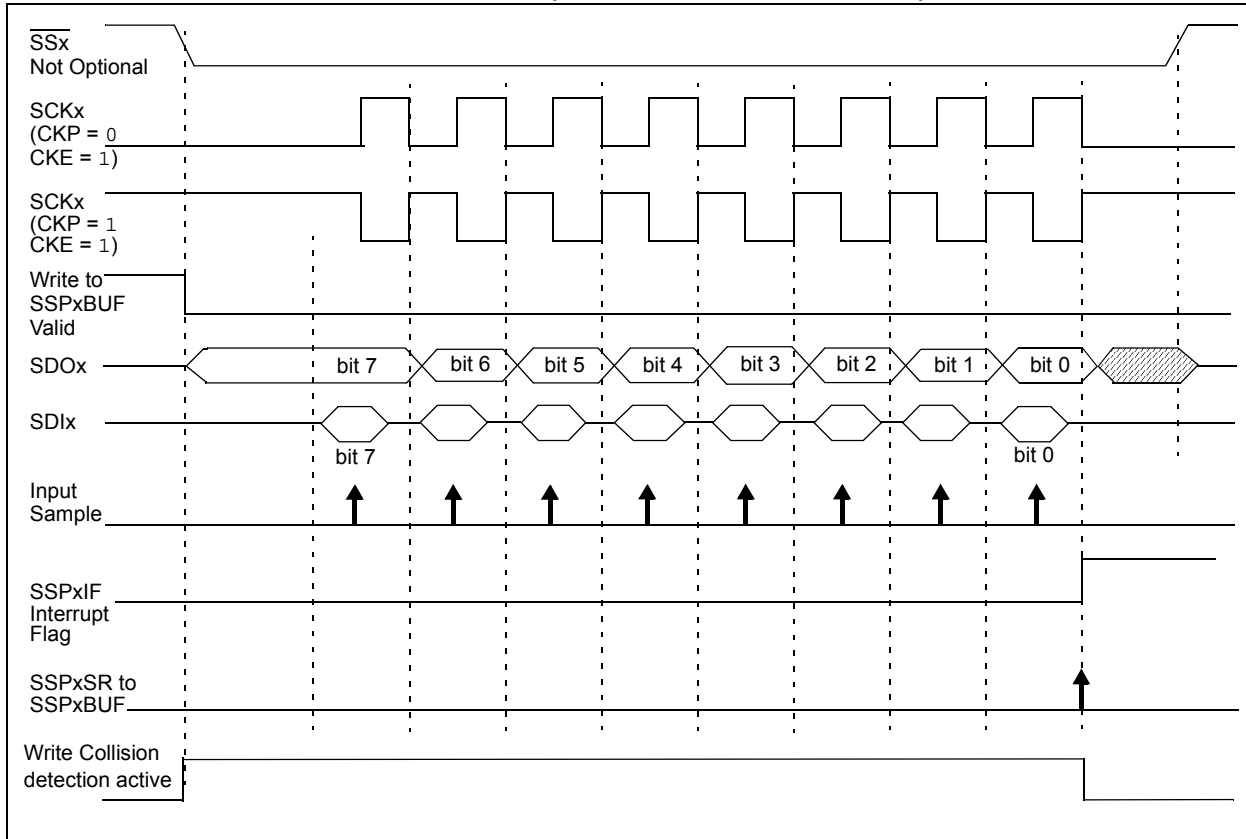


FIGURE 21-10: SPI MODE WAVEFORM (SLAVE MODE WITH CKE = 1)



21.2.6 SPI OPERATION IN SLEEP MODE

In SPI Master mode, module clocks may be operating at a different speed than when in Full-Power mode; in the case of the Sleep mode, all clocks are halted.

Special care must be taken by the user when the MSSPx clock is much faster than the system clock.

In Slave mode, when MSSPx interrupts are enabled, after the master completes sending data, an MSSPx interrupt will wake the controller from Sleep.

If an exit from Sleep mode is not desired, MSSPx interrupts should be disabled.

In SPI Master mode, when the Sleep mode is selected, all module clocks are halted and the transmission/reception will remain in that state until the device wakes. After the device returns to Run mode, the module will resume transmitting and receiving data.

In SPI Slave mode, the SPI Transmit/Receive Shift register operates asynchronously to the device. This allows the device to be placed in Sleep mode and data to be shifted into the SPI Transmit/Receive Shift register. When all 8 bits have been received, the MSSPx interrupt flag bit will be set and if enabled, will wake the device.

TABLE 21-1: SUMMARY OF REGISTERS ASSOCIATED WITH SPI OPERATION

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Register on Page |
|----------|--|--------|--------|--------|-----------|--------|--------|--------|------------------|
| ANSELF | ANSF7 | ANSF6 | ANSF5 | ANSF4 | ANSF3 | ANSF2 | ANSF1 | ANSF0 | 130 |
| INTCON | GIE | PEIE | TMR0IE | INTE | IOCFIE | TMR0IF | INTF | IOCF | 76 |
| PIE1 | TMR1GIE | ADIE | RC1IE | TX1IE | SSP1IE | CCP1IE | TMR2IE | TMR1IE | 77 |
| PIE4 | CCP10IE | CCP9IE | RC2IE | TX2IE | CCP8IE | CCP7IE | BCL2IE | SSP2IE | 80 |
| PIR1 | TMR1GIF | ADIF | RC1IF | TX1IF | SSP1IF | CCP1IF | TMR2IF | TMR1IF | 81 |
| PIR4 | CCP10IF | CCP9IF | RC2IF | TX2IF | CCP8IF | CCP7IF | BCL2IF | SSP2IF | 84 |
| SSP1BUF | MSSPx Receive Buffer/Transmit Register | | | | | | | | 197* |
| SSP2BUF | MSSPx Receive Buffer/Transmit Register | | | | | | | | 197* |
| SSP1CON1 | WCOL | SSPOV | SSPEN | CKP | SSPM<3:0> | | | | 244 |
| SSP2CON1 | WCOL | SSPOV | SSPEN | CKP | SSPM<3:0> | | | | 244 |
| SSP1CON3 | ACKTIM | PCIE | SCIE | BOEN | SDAHT | SBCDE | AHEN | DHEN | 246 |
| SSP2CON3 | ACKTIM | PCIE | SCIE | BOEN | SDAHT | SBCDE | AHEN | DHEN | 246 |
| SSP1STAT | SMP | CKE | D/Ā | P | S | R/Ā | UA | BF | 242 |
| SSP2STAT | SMP | CKE | D/Ā | P | S | R/Ā | UA | BF | 242 |
| TRISC | TRISC7 | TRISC6 | TRISC5 | TRISC4 | TRISC3 | TRISC2 | TRISC1 | TRISC0 | 120 |
| TRISD | TRISD7 | TRISD6 | TRISD5 | TRISD4 | TRISD3 | TRISD2 | TRISD1 | TRISD0 | 123 |
| TRISF | TRISF7 | TRISF6 | TRISF5 | TRISF4 | TRISF3 | TRISF2 | TRISF1 | TRISF0 | 129 |

Legend: — = Unimplemented location, read as '0'. Shaded cells are not used by the MSSPx in SPI mode.

* Page provides register information.

PIC16(L)F1526/7

21.3 I²C MODE OVERVIEW

The Inter-Integrated Circuit Bus (I²C) is a multi-master serial data communication bus. Devices communicate in a master/slave environment where the master devices initiate the communication. A slave device is controlled through addressing.

The I²C bus specifies two signal connections:

- Serial Clock (SCLx)
- Serial Data (SDAx)

Figure 21-2 and Figure 21-3 shows the block diagram of the MSSPx module when operating in I²C mode.

Both the SCLx and SDAx connections are bidirectional open-drain lines, each requiring pull-up resistors for the supply voltage. Pulling the line to ground is considered a logical zero and letting the line float is considered a logical one.

Figure 21-11 shows a typical connection between two processors configured as master and slave devices.

The I²C bus can operate with one or more master devices and one or more slave devices.

There are four potential modes of operation for a given device:

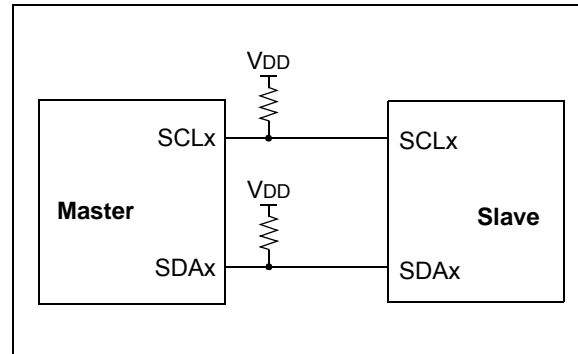
- Master Transmit mode
(master is transmitting data to a slave)
- Master Receive mode
(master is receiving data from a slave)
- Slave Transmit mode
(slave is transmitting data to a master)
- Slave Receive mode
(slave is receiving data from the master)

To begin communication, a master device starts out in Master Transmit mode. The master device sends out a Start bit followed by the address byte of the slave it intends to communicate with. This is followed by a single Read/Write bit, which determines whether the master intends to transmit to or receive data from the slave device.

If the requested slave exists on the bus, it will respond with an Acknowledge bit, otherwise known as an ACK. The master then continues in either Transmit mode or Receive mode and the slave continues in the complement, either in Receive mode or Transmit mode, respectively.

A Start bit is indicated by a high-to-low transition of the SDAx line while the SCLx line is held high. Address and data bytes are sent out, Most Significant bit (MSb) first. The Read/Write bit is sent out as a logical one when the master intends to read data from the slave, and is sent out as a logical zero when it intends to write data to the slave.

FIGURE 21-11: I²C MASTER/SLAVE CONNECTION



The Acknowledge bit ($\overline{\text{ACK}}$) is an active-low signal, which holds the SDAx line low to indicate to the transmitter that the slave device has received the transmitted data and is ready to receive more.

The transition of a data bit is always performed while the SCLx line is held low. Transitions that occur while the SCLx line is held high are used to indicate Start and Stop bits.

If the master intends to write to the slave, then it repeatedly sends out a byte of data, with the slave responding after each byte with an $\overline{\text{ACK}}$ bit. In this example, the master device is in Master Transmit mode and the slave is in Slave Receive mode.

If the master intends to read from the slave, then it repeatedly receives a byte of data from the slave, and responds after each byte with an $\overline{\text{ACK}}$ bit. In this example, the master device is in Master Receive mode and the slave is Slave Transmit mode.

On the last byte of data communicated, the master device may end the transmission by sending a Stop bit. If the master device is in Receive mode, it sends the Stop bit in place of the last ACK bit. A Stop bit is indicated by a low-to-high transition of the SDAx line while the SCLx line is held high.

In some cases, the master may want to maintain control of the bus and re-initiate another transmission. If so, the master device may send another Start bit in place of the Stop bit or last $\overline{\text{ACK}}$ bit when it is in receive mode.

The I²C bus specifies three message protocols;

- Single message where a master writes data to a slave.
- Single message where a master reads data from a slave.
- Combined message where a master initiates a minimum of two writes, or two reads, or a combination of writes and reads, to one or more slaves.

When one device is transmitting a logical one, or letting the line float, and a second device is transmitting a logical zero, or holding the line low, the first device can detect that the line is not a logical one. This detection, when used on the SCLx line, is called clock stretching. Clock stretching gives slave devices a mechanism to control the flow of data. When this detection is used on the SDAx line, it is called arbitration. Arbitration ensures that there is only one master device communicating at any single time.

21.3.1 CLOCK STRETCHING

When a slave device has not completed processing data, it can delay the transfer of more data through the process of clock stretching. An addressed slave device may hold the SCLx clock line low after receiving or sending a bit, indicating that it is not yet ready to continue. The master that is communicating with the slave will attempt to raise the SCLx line in order to transfer the next bit, but will detect that the clock line has not yet been released. Because the SCLx connection is open-drain, the slave has the ability to hold that line low until it is ready to continue communicating.

Clock stretching allows receivers that cannot keep up with a transmitter to control the flow of incoming data.

21.3.2 ARBITRATION

Each master device must monitor the bus for Start and Stop bits. If the device detects that the bus is busy, it cannot begin a new message until the bus returns to an Idle state.

However, two master devices may try to initiate a transmission on or about the same time. When this occurs, the process of arbitration begins. Each transmitter checks the level of the SDAx data line and compares it to the level that it expects to find. The first transmitter to observe that the two levels do not match, loses arbitration, and must stop transmitting on the SDAx line.

For example, if one transmitter holds the SDAx line to a logical one (lets it float) and a second transmitter holds it to a logical zero (pulls it low), the result is that the SDAx line will be low. The first transmitter then observes that the level of the line is different than expected and concludes that another transmitter is communicating.

The first transmitter to notice this difference is the one that loses arbitration and must stop driving the SDAx line. If this transmitter is also a master device, it also must stop driving the SCLx line. It then can monitor the lines for a Stop condition before trying to reissue its transmission. In the meantime, the other device that has not noticed any difference between the expected and actual levels on the SDAx line continues with its original transmission. It can do so without any complications, because so far, the transmission appears exactly as expected with no other transmitter disturbing the message.

Slave Transmit mode can also be arbitrated, when a master addresses multiple slaves, but this is less common.

If two master devices are sending a message to two different slave devices at the address stage, the master sending the lower slave address always wins arbitration. When two master devices send messages to the same slave address, and addresses can sometimes refer to multiple slaves, the arbitration process must continue into the data stage.

Arbitration usually occurs very rarely, but it is a necessary process for proper multi-master support.

PIC16(L)F1526/7

21.4 I²C MODE OPERATION

All MSSPx I²C communication is byte oriented and shifted out MSb first. Six SFR registers and 2 interrupt flags interface the module with the PIC[®] microcontroller and user software. Two pins, SDAx and SCLx, are exercised by the module to communicate with other external I²C devices.

21.4.1 BYTE FORMAT

All communication in I²C is done in 9-bit segments. A byte is sent from a master to a slave or vice-versa, followed by an Acknowledge bit sent back. After the 8th falling edge of the SCLx line, the device outputting data on the SDAx changes that pin to an input and reads in an acknowledge value on the next clock pulse.

The clock signal, SCLx, is provided by the master. Data is valid to change while the SCLx signal is low, and sampled on the rising edge of the clock. Changes on the SDAx line while the SCLx line is high define special conditions on the bus, explained below.

21.4.2 DEFINITION OF I²C TERMINOLOGY

There is language and terminology in the description of I²C communication that have definitions specific to I²C. That word usage is defined below and may be used in the rest of this document without explanation. This table was adapted from the Philips I²C specification.

21.4.3 SDAX AND SCLX PINS

Selection of any I²C mode with the SSPEN bit set, forces the SCLx and SDAx pins to be open-drain. These pins should be set by the user to inputs by setting the appropriate TRIS bits.

Note: Data is tied to output zero when an I²C mode is enabled.

21.4.4 SDAX HOLD TIME

The hold time of the SDAx pin is selected by the SDAHT bit of the SSPxCON3 register. Hold time is the time SDAx is held valid after the falling edge of SCLx. Setting the SDAHT bit selects a longer 300 ns minimum hold time and may help on buses with large capacitance.

TABLE 21-2: I²C BUS TERMS

| TERM | Description |
|------------------|---|
| Transmitter | The device which shifts data out onto the bus. |
| Receiver | The device which shifts data in from the bus. |
| Master | The device that initiates a transfer, generates clock signals and terminates a transfer. |
| Slave | The device addressed by the master. |
| Multi-master | A bus with more than one device that can initiate data transfers. |
| Arbitration | Procedure to ensure that only one master at a time controls the bus. Winning arbitration ensures that the message is not corrupted. |
| Synchronization | Procedure to synchronize the clocks of two or more devices on the bus. |
| Idle | No master is controlling the bus, and both SDAx and SCLx lines are high. |
| Active | Any time one or more master devices are controlling the bus. |
| Addressed Slave | Slave device that has received a matching address and is actively being clocked by a master. |
| Matching Address | Address byte that is clocked into a slave that matches the value stored in SSPxADD. |
| Write Request | Slave receives a matching address with R/W bit clear, and is ready to clock in data. |
| Read Request | Master sends an address byte with the R/W bit set, indicating that it wishes to clock data out of the Slave. This data is the next and all following bytes until a Restart or Stop. |
| Clock Stretching | When a device on the bus hold SCLx low to stall communication. |
| Bus Collision | Any time the SDAx line is sampled low by the module while it is outputting and expected high state. |

21.4.5 START CONDITION

The I²C specification defines a Start condition as a transition of SDAx from a high to a low state while SCLx line is high. A Start condition is always generated by the master and signifies the transition of the bus from an Idle to an Active state. Figure 21-12 shows wave forms for Start and Stop conditions.

A bus collision can occur on a Start condition if the module samples the SDAx line low before asserting it low. This does not conform to the I²C Specification that states no bus collision can occur on a Start.

21.4.6 STOP CONDITION

A Stop condition is a transition of the SDAx line from low-to-high state while the SCLx line is high.

Note: At least one SCLx low time must appear before a Stop is valid, therefore, if the SDAx line goes low then high again while the SCLx line stays high, only the Start condition is detected.

21.4.7 RESTART CONDITION

A Restart is valid any time that a Stop would be valid. A master can issue a Restart if it wishes to hold the bus after terminating the current transfer. A Restart has the same effect on the slave that a Start would, resetting all slave logic and preparing it to clock in an address. The master may want to address the same or another slave. Figure 21-13 shows the wave form for a Restart condition.

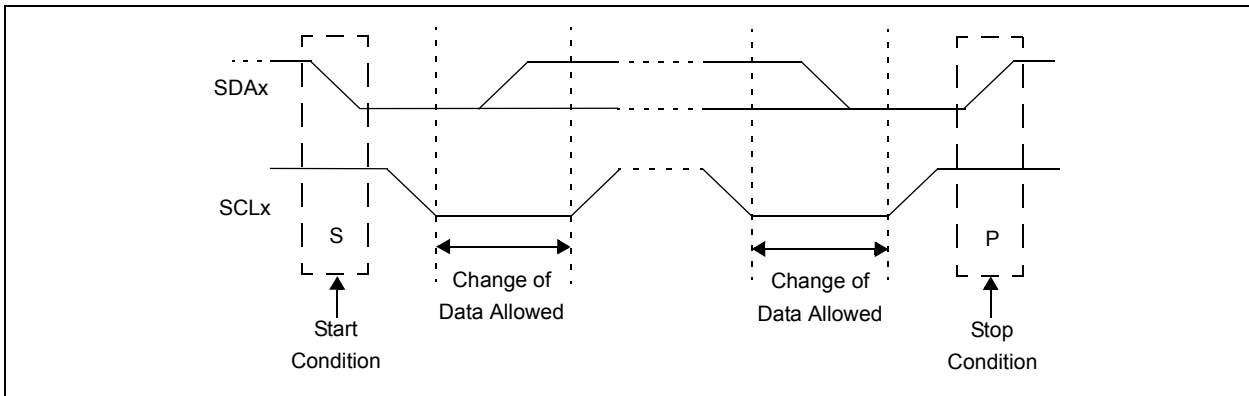
In 10-bit Addressing Slave mode a Restart is required for the master to clock data out of the addressed slave. Once a slave has been fully addressed, matching both high and low address bytes, the master can issue a Restart and the high address byte with the R/W bit set. The slave logic will then hold the clock and prepare to clock out data.

After a full match with R/W clear in 10-bit mode, a prior match flag is set and maintained. Until a Stop condition, a high address with R/W clear, or high address match fails.

21.4.8 START/STOP CONDITION INTERRUPT MASKING

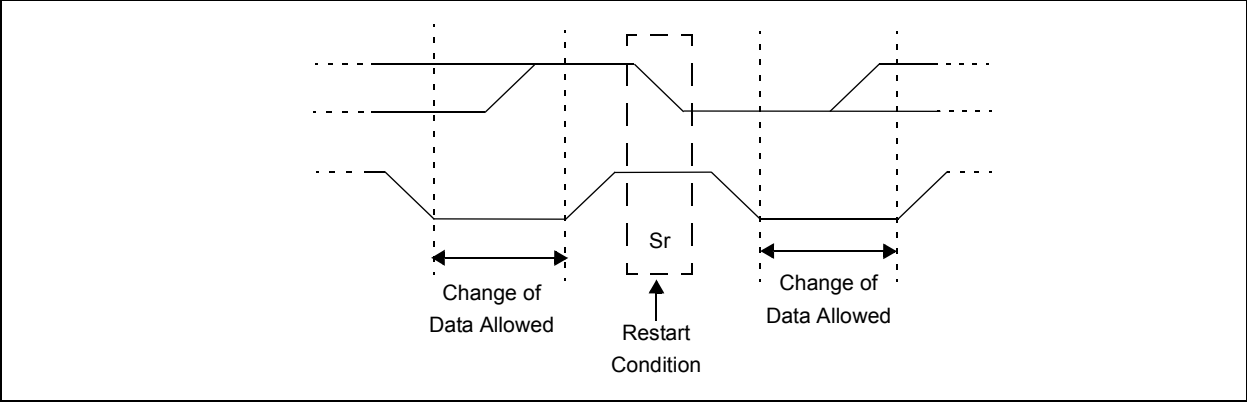
The SCIE and PCIE bits of the SSPxCON3 register can enable the generation of an interrupt in Slave modes that do not typically support this function. Slave modes where interrupt on Start and Stop detect are already enabled, these bits will have no effect.

FIGURE 21-12: I²C START AND STOP CONDITIONS



PIC16(L)F1526/7

FIGURE 21-13: I²C RESTART CONDITION



21.4.9 ACKNOWLEDGE SEQUENCE

The 9th SCLx pulse for any transferred byte in I²C is dedicated as an Acknowledge. It allows receiving devices to respond back to the transmitter by pulling the SDAx line low. The transmitter must release control of the line during this time to shift in the response. The Acknowledge (ACK) is an active-low signal, pulling the SDAx line low indicated to the transmitter that the device has received the transmitted data and is ready to receive more.

The result of an $\overline{\text{ACK}}$ is placed in the ACKSTAT bit of the SSPxCON2 register.

Slave software, when the AHEN and DHEN bits are set, allow the user to set the $\overline{\text{ACK}}$ value sent back to the transmitter. The ACKDT bit of the SSPxCON2 register is set/cleared to determine the response.

Slave hardware will generate an $\overline{\text{ACK}}$ response if the AHEN and DHEN bits of the SSPxCON3 register are clear.

There are certain conditions where an $\overline{\text{ACK}}$ will not be sent by the slave. If the BF bit of the SSPxSTAT register or the SSPOV bit of the SSPxCON1 register are set when a byte is received.

When the module is addressed, after the 8th falling edge of SCLx on the bus, the ACKTIM bit of the SSPxCON3 register is set. The ACKTIM bit indicates the acknowledge time of the active bus. The ACKTIM Status bit is only active when the AHEN bit or DHEN bit is enabled.

21.5 I²C SLAVE MODE OPERATION

The MSSPx Slave mode operates in one of four modes selected in the SSPM bits of SSPxCON1 register. The modes can be divided into 7-bit and 10-bit Addressing mode. 10-bit Addressing modes operate the same as 7-bit with some additional overhead for handling the larger addresses.

Modes with Start and Stop bit interrupts operated the same as the other modes with SSPxIF additionally getting set upon detection of a Start, Restart, or Stop condition.

21.5.1 SLAVE MODE ADDRESSES

The SSPxADD register ([Register 21-6](#)) contains the Slave mode address. The first byte received after a Start or Restart condition is compared against the value stored in this register. If the byte matches, the value is loaded into the SSPxBUF register and an interrupt is generated. If the value does not match, the module goes idle and no indication is given to the software that anything happened.

The SSPx Mask register ([Register 21-5](#)) affects the address matching process. See [Section 21.5.9 “SSPx Mask Register”](#) for more information.

21.5.1.1 I²C Slave 7-bit Addressing Mode

In 7-bit Addressing mode, the LSB of the received data byte is ignored when determining if there is an address match.

21.5.1.2 I²C Slave 10-bit Addressing Mode

In 10-bit Addressing mode, the first received byte is compared to the binary value of '1 1 1 0 A9 A8 0'. A9 and A8 are the two MSb's of the 10-bit address and stored in bits 2 and 1 of the SSPxADD register.

After the acknowledge of the high byte the UA bit is set and SCLx is held low until the user updates SSPxADD with the low address. The low address byte is clocked in and all 8 bits are compared to the low address value in SSPxADD. Even if there is not an address match; SSPxIF and UA are set, and SCLx is held low until SSPxADD is updated to receive a high byte again. When SSPxADD is updated the UA bit is cleared. This ensures the module is ready to receive the high address byte on the next communication.

A high and low address match as a write request is required at the start of all 10-bit addressing communication. A transmission can be initiated by issuing a Restart once the slave is addressed, and clocking in the high address with the R/W bit set. The slave hardware will then acknowledge the read request and prepare to clock out data. This is only valid for a slave after it has received a complete high and low address byte match.

PIC16(L)F1526/7

21.5.2 SLAVE RECEPTION

When the $\overline{R/W}$ bit of a matching received address byte is clear, the $\overline{R/W}$ bit of the SSPxSTAT register is cleared. The received address is loaded into the SSPxBUF register and acknowledged.

When the overflow condition exists for a received address, then not Acknowledge is given. An overflow condition is defined as either bit BF of the SSPxSTAT register is set, or bit SSPOV of the SSPxCON1 register is set. The BOEN bit of the SSPxCON3 register modifies this operation. For more information see [Register 21-4](#).

An MSSPx interrupt is generated for each transferred data byte. Flag bit, SSPxIF, must be cleared by software.

When the SEN bit of the SSPxCON2 register is set, SCLx will be held low (clock stretch) following each received byte. The clock must be released by setting the CKP bit of the SSPxCON1 register, except sometimes in 10-bit mode. See [Section 21.2.3 “SPI Master Mode”](#) for more detail.

21.5.2.1 7-bit Addressing Reception

This section describes a standard sequence of events for the MSSPx module configured as an I²C slave in 7-bit Addressing mode. [Figure 21-14](#) and [Figure 21-15](#) is used as a visual reference for this description.

This is a step by step process of what typically must be done to accomplish I²C communication.

1. Start bit detected.
2. S bit of SSPxSTAT is set; SSPxIF is set if interrupt on Start detect is enabled.
3. Matching address with $\overline{R/W}$ bit clear is received.
4. The slave pulls SDAx low sending an \overline{ACK} to the master, and sets SSPxIF bit.
5. Software clears the SSPxIF bit.
6. Software reads received address from SSPxBUF clearing the BF flag.
7. If SEN = 1; Slave software sets CKP bit to release the SCLx line.
8. The master clocks out a data byte.
9. Slave drives SDAx low sending an \overline{ACK} to the master, and sets SSPxIF bit.
10. Software clears SSPxIF.
11. Software reads the received byte from SSPxBUF clearing BF.
12. Steps 8-12 are repeated for all received bytes from the master.
13. Master sends Stop condition, setting P bit of SSPxSTAT, and the bus goes idle.

21.5.2.2 7-bit Reception with AHEN and DHEN

Slave device reception with AHEN and DHEN set operate the same as without these options with extra interrupts and clock stretching added after the 8th falling edge of SCLx. These additional interrupts allow the slave software to decide whether it wants to ACK the receive address or data byte, rather than the hardware. This functionality adds support for PMBus™ that was not present on previous versions of this module.

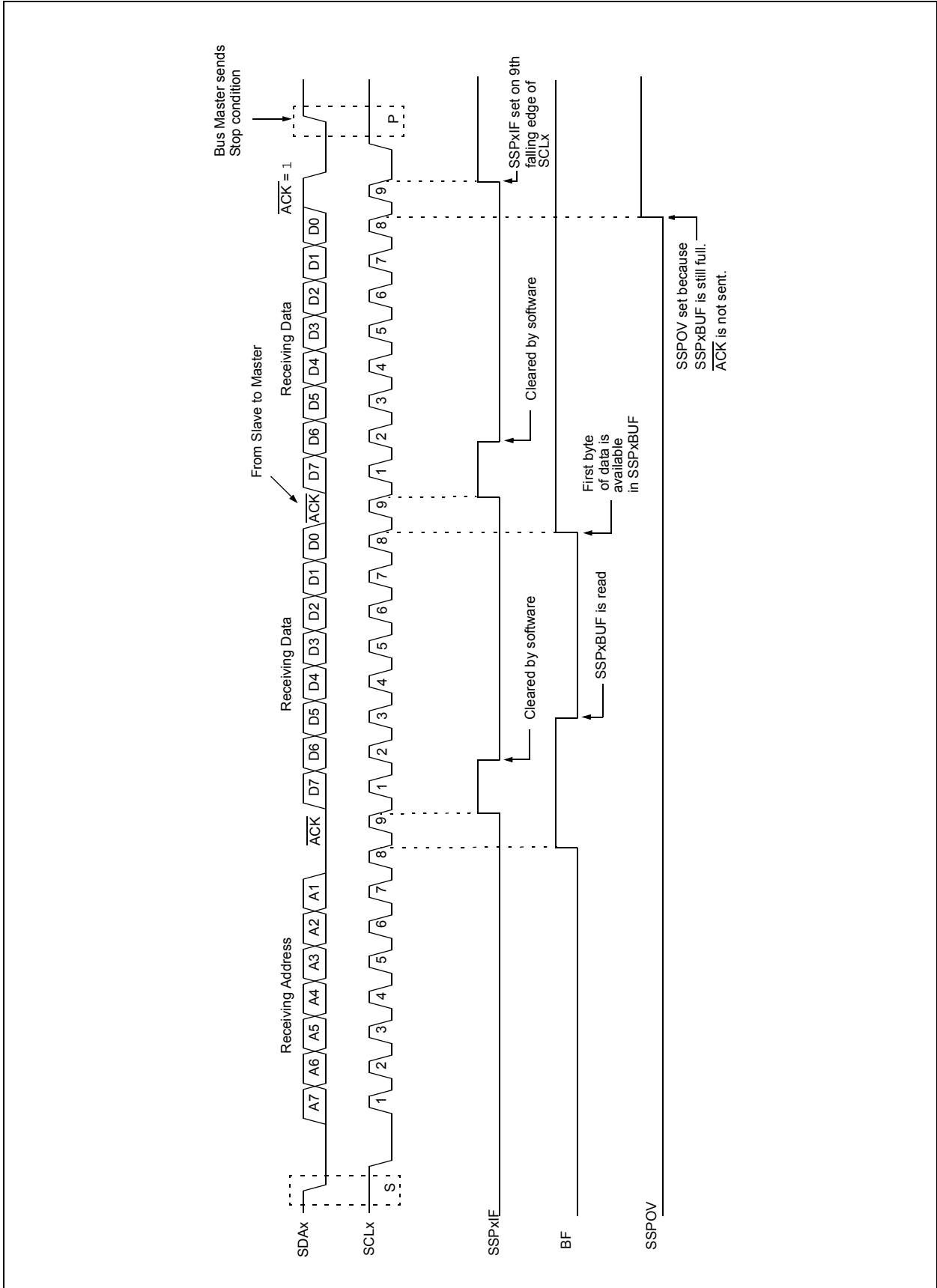
This list describes the steps that need to be taken by slave software to use these options for I²C communication. [Figure 21-16](#) displays a module using both address and data holding. [Figure 21-17](#) includes the operation with the SEN bit of the SSPxCON2 register set.

1. S bit of SSPxSTAT is set; SSPxIF is set if interrupt on Start detect is enabled.
2. Matching address with $\overline{R/W}$ bit clear is clocked in. SSPxIF is set and CKP cleared after the 8th falling edge of SCLx.
3. Slave clears the SSPxIF.
4. Slave can look at the ACKTIM bit of the SSPxCON3 register to determine if the SSPxIF was after or before the \overline{ACK} .
5. Slave reads the address value from SSPxBUF, clearing the BF flag.
6. Slave sets \overline{ACK} value clocked out to the master by setting ACKDT.
7. Slave releases the clock by setting CKP.
8. SSPxIF is set after an \overline{ACK} , not after a NACK.
9. If SEN = 1 the slave hardware will stretch the clock after the \overline{ACK} .
10. Slave clears SSPxIF.

Note: SSPxIF is still set after the 9th falling edge of SCLx even if there is no clock stretching and BF has been cleared. Only if NACK is sent to master is SSPxIF not set

11. SSPxIF set and CKP cleared after 8th falling edge of SCLx for a received data byte.
12. Slave looks at ACKTIM bit of SSPxCON3 to determine the source of the interrupt.
13. Slave reads the received data from SSPxBUF clearing BF.
14. Steps 7-14 are the same for each received data byte.
15. Communication is ended by either the slave sending an $\overline{ACK} = 1$, or the master sending a Stop condition. If a Stop is sent and Interrupt on Stop Detect is disabled, the slave will only know by polling the P bit of the SSTAT register.

FIGURE 21-14: I²C SLAVE, 7-BIT ADDRESS, RECEPTION (SEN = 0, AHEN = 0, DHEN = 0)



PIC16(L)F1526/7

FIGURE 21-15: I²C SLAVE, 7-BIT ADDRESS, RECEPTION (SEN = 1, AHEN = 0, DHEN = 0)

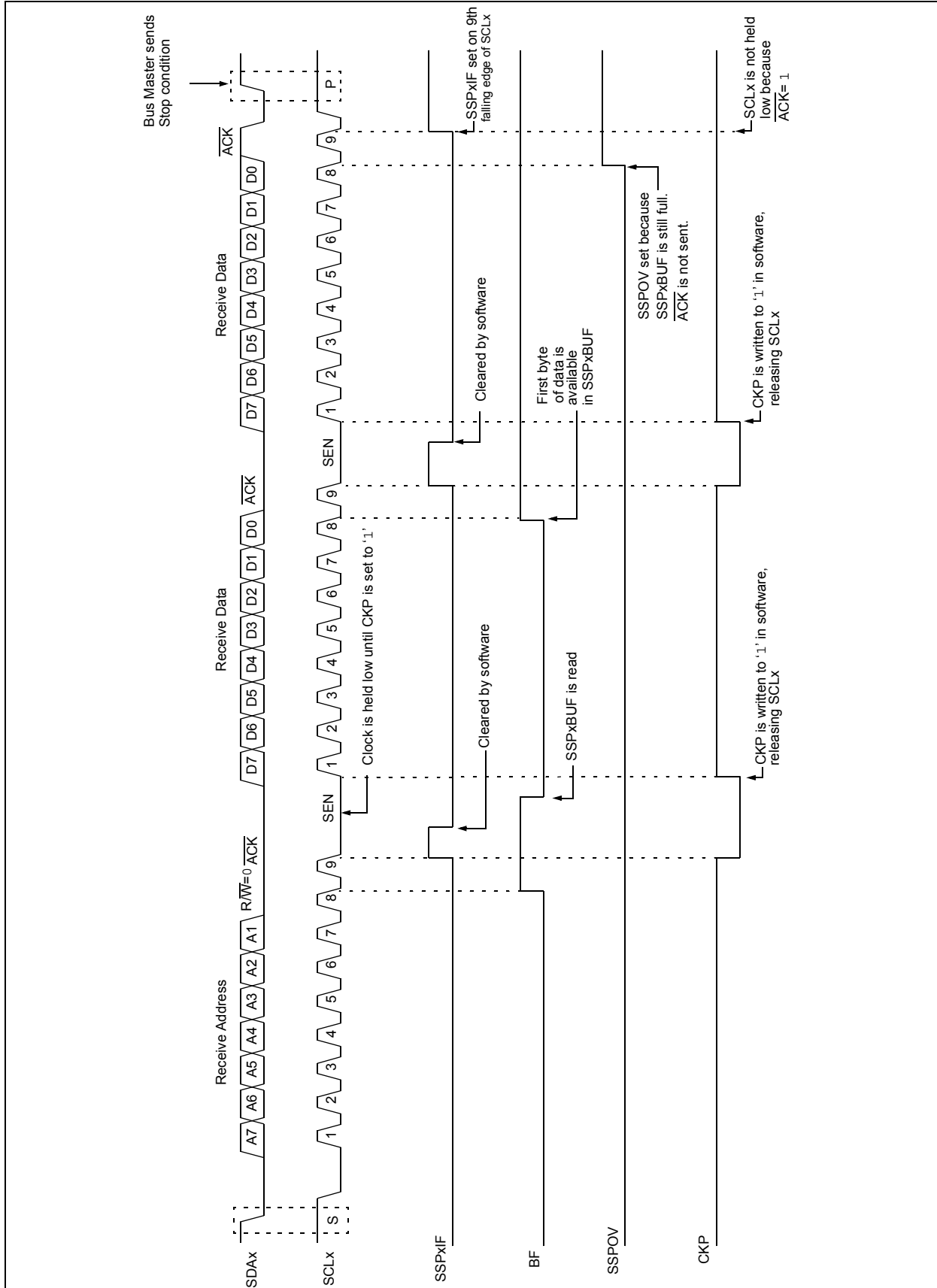
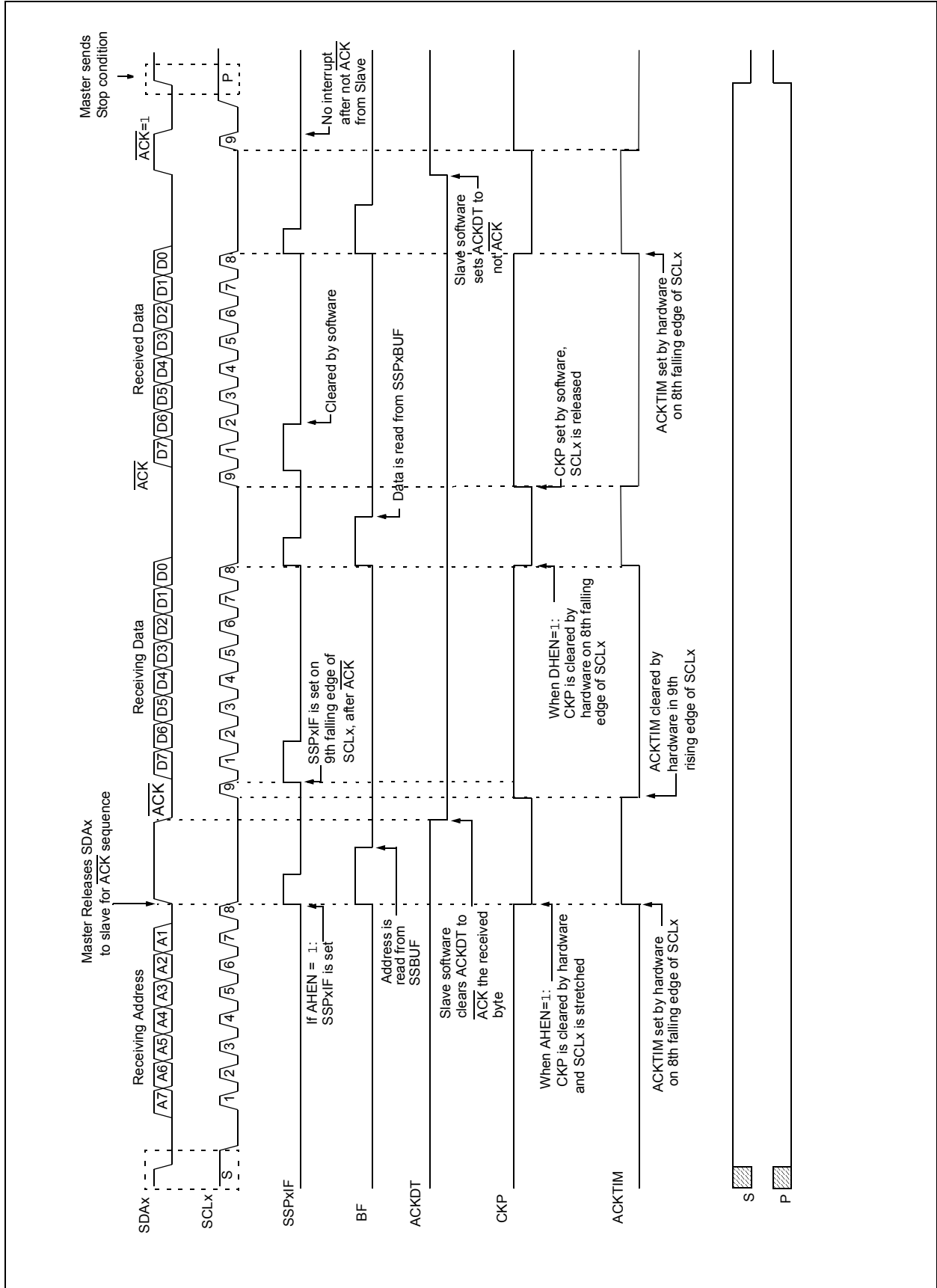
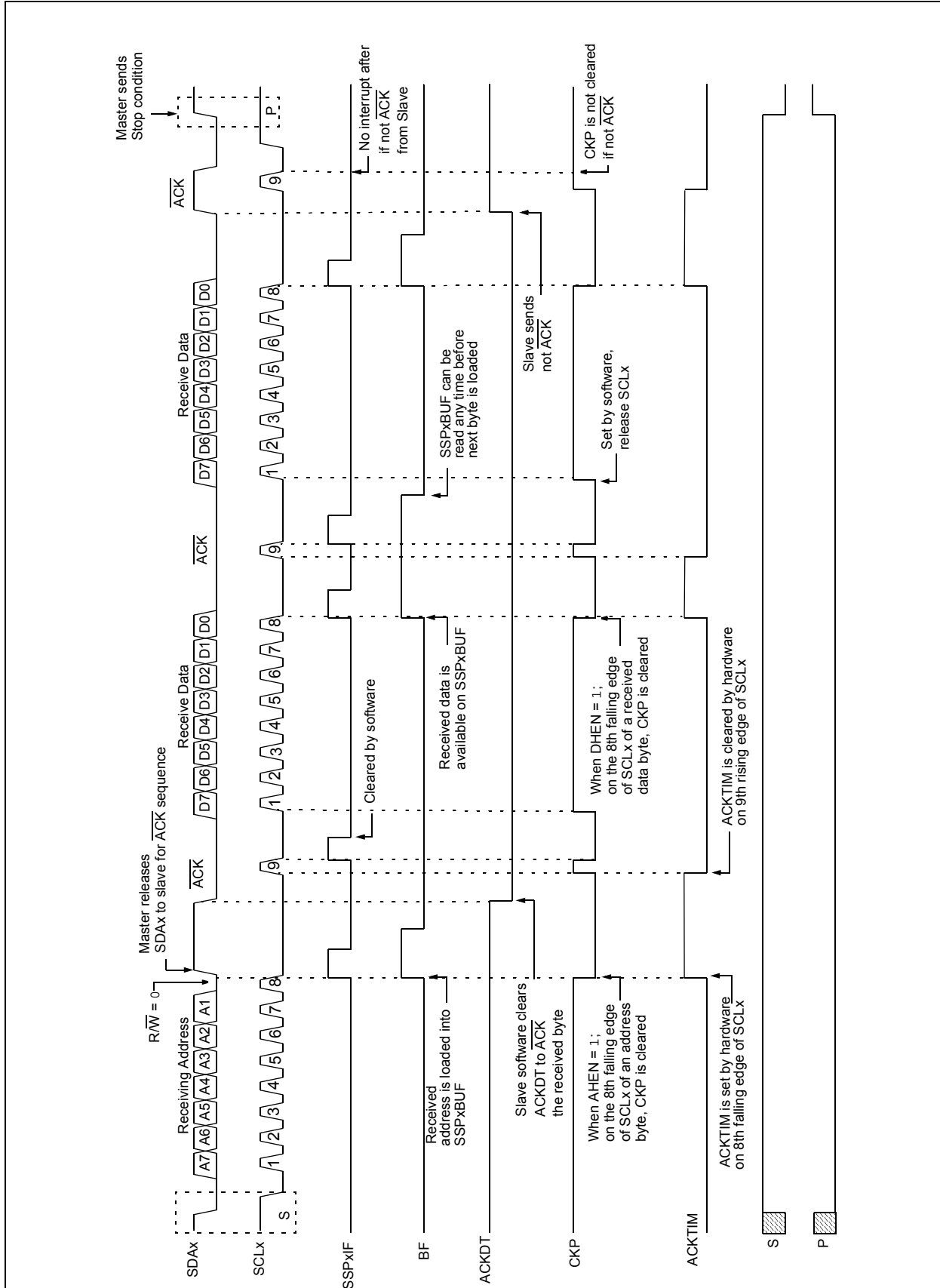


FIGURE 21-16: I²C SLAVE, 7-BIT ADDRESS, RECEPTION (SEN = 0, AHEN = 1, DHEN = 1)



PIC16(L)F1526/7

FIGURE 21-17: I²C SLAVE, 7-BIT ADDRESS, RECEPTION (SEN = 1, AHEN = 1, DHEN = 1)



21.5.3 SLAVE TRANSMISSION

When the $\overline{R/W}$ bit of the incoming address byte is set and an address match occurs, the $\overline{R/W}$ bit of the SSPxSTAT register is set. The received address is loaded into the SSPxBUF register, and an \overline{ACK} pulse is sent by the slave on the ninth bit.

Following the \overline{ACK} , slave hardware clears the CKP bit and the SCLx pin is held low (see [Section 21.5.6 "Clock Stretching"](#) for more detail). By stretching the clock, the master will be unable to assert another clock pulse until the slave is done preparing the transmit data.

The transmit data must be loaded into the SSPxBUF register which also loads the SSPxSR register. Then the SCLx pin should be released by setting the CKP bit of the SSPxCON1 register. The eight data bits are shifted out on the falling edge of the SCLx input. This ensures that the SDAx signal is valid during the SCLx high time.

The \overline{ACK} pulse from the master-receiver is latched on the rising edge of the ninth SCLx input pulse. This \overline{ACK} value is copied to the ACKSTAT bit of the SSPxCON2 register. If ACKSTAT is set (not \overline{ACK}), then the data transfer is complete. In this case, when the not \overline{ACK} is latched by the slave, the slave goes idle and waits for another occurrence of the Start bit. If the SDAx line was low (\overline{ACK}), the next transmit data must be loaded into the SSPxBUF register. Again, the SCLx pin must be released by setting bit CKP.

An MSSPx interrupt is generated for each data transfer byte. The SSPxIF bit must be cleared by software and the SSPxSTAT register is used to determine the status of the byte. The SSPxIF bit is set on the falling edge of the ninth clock pulse.

21.5.3.1 Slave Mode Bus Collision

A slave receives a Read request and begins shifting data out on the SDAx line. If a bus collision is detected and the SBCDE bit of the SSPxCON3 register is set, the BCLxIF bit of the PIRx register is set. Once a bus collision is detected, the slave goes Idle and waits to be addressed again. User software can use the BCLxIF bit to handle a slave bus collision.

21.5.3.2 7-bit Transmission

A master device can transmit a read request to a slave, and then clock data out of the slave. The list below outlines what software for a slave will need to do to accomplish a standard transmission. [Figure 21-18](#) can be used as a reference to this list.

1. Master sends a Start condition on SDAx and SCLx.
2. S bit of SSPxSTAT is set; SSPxIF is set if interrupt on Start detect is enabled.
3. Matching address with $\overline{R/W}$ bit set is received by the slave setting SSPxIF bit.
4. Slave hardware generates an \overline{ACK} and sets SSPxIF.
5. SSPxIF bit is cleared by user.
6. Software reads the received address from SSPxBUF, clearing BF.
7. $\overline{R/W}$ is set so CKP was automatically cleared after the \overline{ACK} .
8. The slave software loads the transmit data into SSPxBUF.
9. CKP bit is set releasing SCLx, allowing the master to clock the data out of the slave.
10. SSPxIF is set after the \overline{ACK} response from the master is loaded into the ACKSTAT register.
11. SSPxIF bit is cleared.
12. The slave software checks the ACKSTAT bit to see if the master wants to clock out more data.

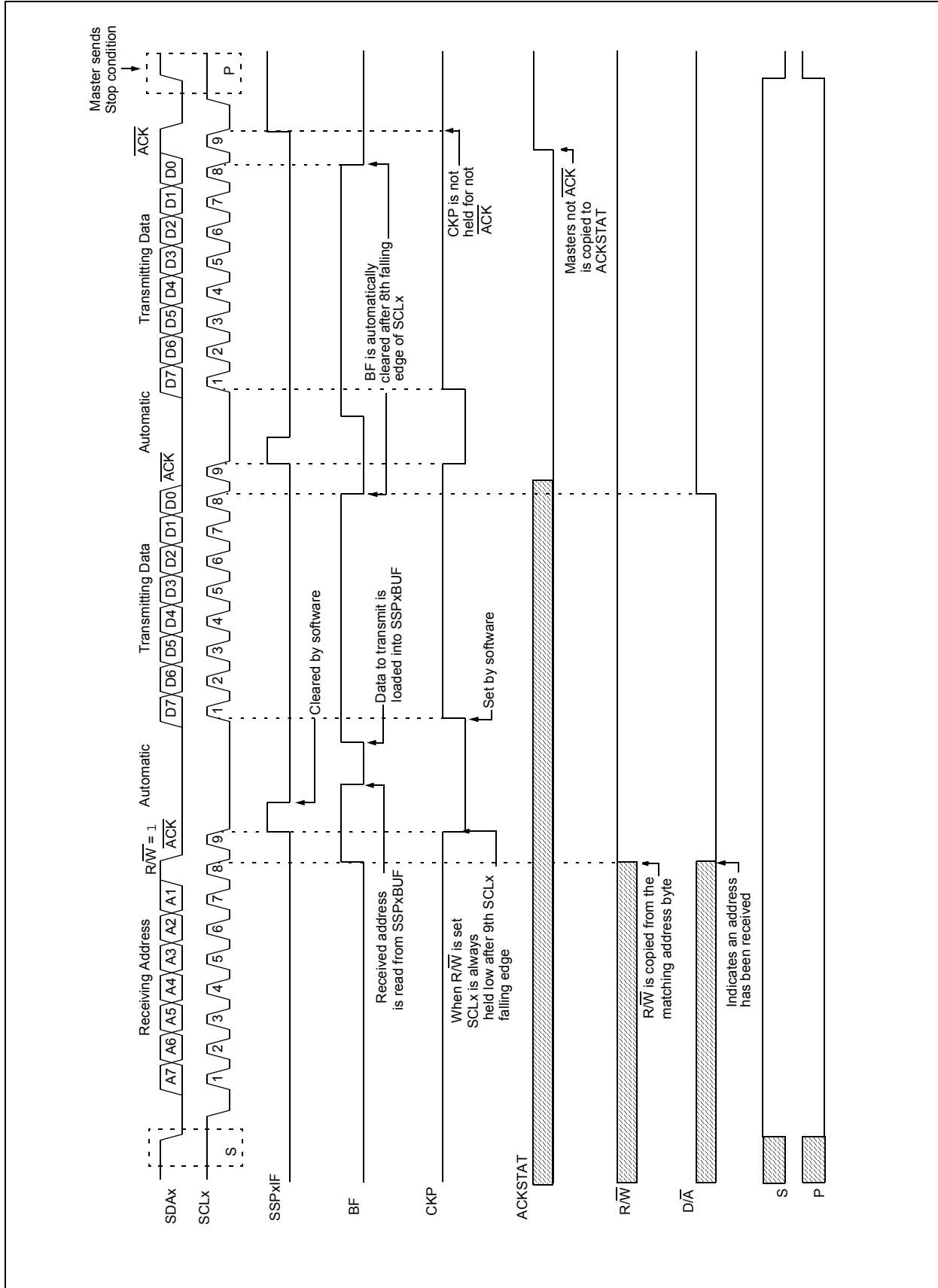
Note 1: If the master \overline{ACK} s the clock will be stretched.

2: ACKSTAT is the only bit updated on the rising edge of SCLx (9th) rather than the falling.

13. Steps 9-13 are repeated for each transmitted byte.
14. If the master sends a not \overline{ACK} ; the clock is not held, but SSPxIF is still set.
15. The master sends a Restart condition or a Stop.
16. The slave is no longer addressed.

PIC16(L)F1526/7

FIGURE 21-18: I²C SLAVE, 7-BIT ADDRESS, TRANSMISSION (AHEN = 0)



21.5.3.3 7-bit Transmission with Address Hold Enabled

Setting the AHEN bit of the SSPxCON3 register enables additional clock stretching and interrupt generation after the 8th falling edge of a received matching address. Once a matching address has been clocked in, CKP is cleared and the SSPxIF interrupt is set.

Figure 21-19 displays a standard waveform of a 7-bit Address Slave Transmission with AHEN enabled.

1. Bus starts Idle.
2. Master sends Start condition; the S bit of SSPxSTAT is set; SSPxIF is set if interrupt on Start detect is enabled.
3. Master sends matching address with $\overline{R/W}$ bit set. After the 8th falling edge of the SCLx line the CKP bit is cleared and SSPxIF interrupt is generated.
4. Slave software clears SSPxIF.
5. Slave software reads ACKTIM bit of SSPxCON3 register, and $\overline{R/W}$ and $\overline{D/A}$ of the SSPxSTAT register to determine the source of the interrupt.
6. Slave reads the address value from the SSPxBUF register clearing the BF bit.
7. Slave software decides from this information if it wishes to ACK or not ACK and sets the ACKDT bit of the SSPxCON2 register accordingly.
8. Slave sets the CKP bit releasing SCLx.
9. Master clocks in the \overline{ACK} value from the slave.
10. Slave hardware automatically clears the CKP bit and sets SSPxIF after the \overline{ACK} if the $\overline{R/W}$ bit is set.
11. Slave software clears SSPxIF.
12. Slave loads value to transmit to the master into SSPxBUF setting the BF bit.

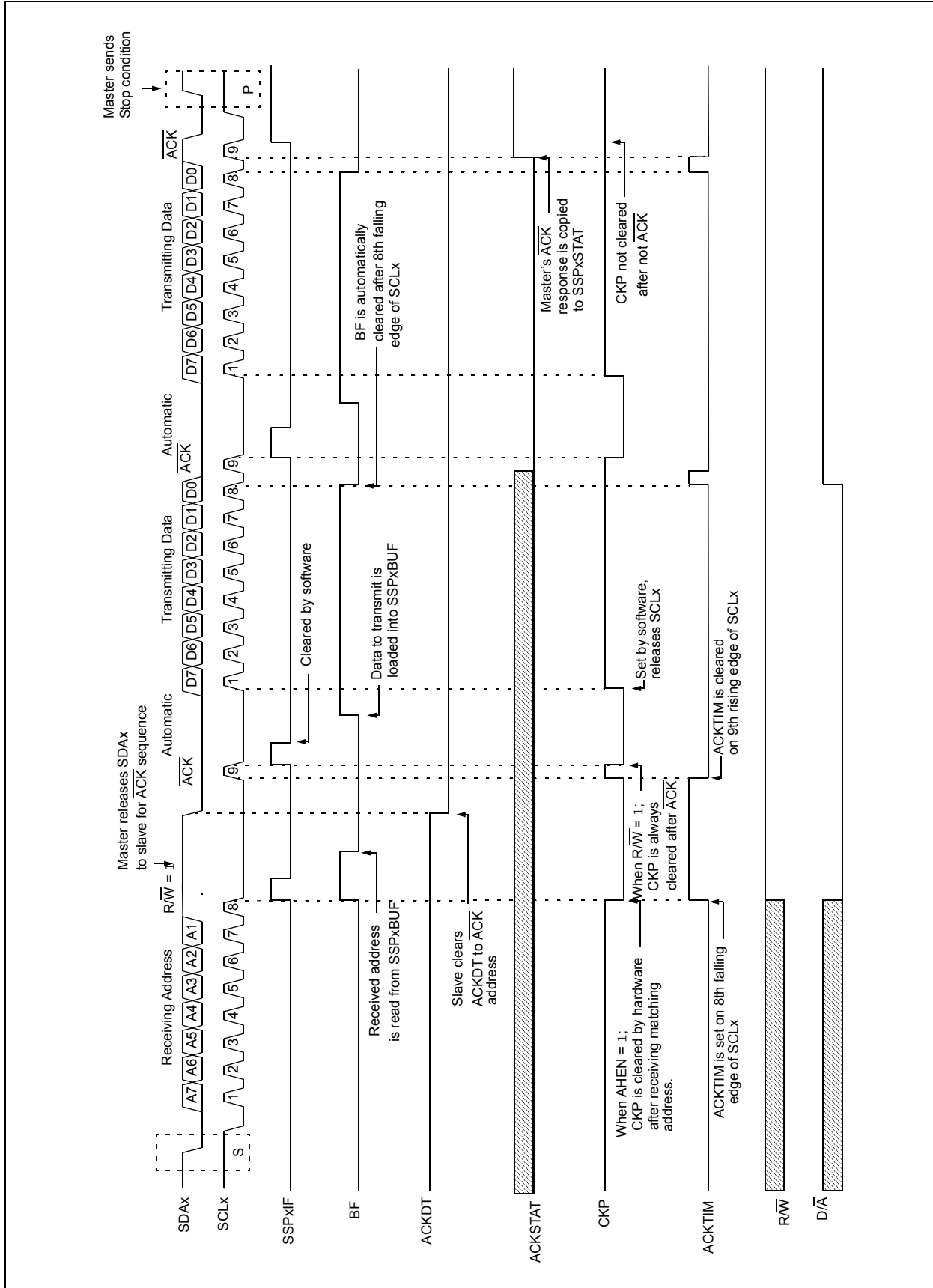
Note: SSPxBUF cannot be loaded until after the \overline{ACK} .

13. Slave sets CKP bit releasing the clock.
14. Master clocks out the data from the slave and sends an \overline{ACK} value on the 9th SCLx pulse.
15. Slave hardware copies the \overline{ACK} value into the ACKSTAT bit of the SSPxCON2 register.
16. Steps 10-15 are repeated for each byte transmitted to the master from the slave.
17. If the master sends a not \overline{ACK} the slave releases the bus allowing the master to send a Stop and end the communication.

Note: Master must send a not \overline{ACK} on the last byte to ensure that the slave releases the SCLx line to receive a Stop.

PIC16(L)F1526/7

FIGURE 21-19: I²C SLAVE, 7-BIT ADDRESS, TRANSMISSION (AHEN = 1)



21.5.4 SLAVE MODE 10-BIT ADDRESS RECEPTION

This section describes a standard sequence of events for the MSSPx module configured as an I²C slave in 10-bit Addressing mode.

Figure 21-20 is used as a visual reference for this description.

This is a step by step process of what must be done by slave software to accomplish I²C communication.

1. Bus starts Idle.
2. Master sends Start condition; S bit of SSPxSTAT is set; SSPxIF is set if interrupt on Start detect is enabled.
3. Master sends matching high address with $\overline{R/W}$ bit clear; UA bit of the SSPxSTAT register is set.
4. Slave sends \overline{ACK} and SSPxIF is set.
5. Software clears the SSPxIF bit.
6. Software reads received address from SSPxBUF clearing the BF flag.
7. Slave loads low address into SSPxADD, releasing SCLx.
8. Master sends matching low address byte to the slave; UA bit is set.

Note: Updates to the SSPxADD register are not allowed until after the ACK sequence.

9. Slave sends \overline{ACK} and SSPxIF is set.

Note: If the low address does not match, SSPxIF and UA are still set so that the slave software can set SSPxADD back to the high address. BF is not set because there is no match. CKP is unaffected.

10. Slave clears SSPxIF.
11. Slave reads the received matching address from SSPxBUF clearing BF.
12. Slave loads high address into SSPxADD.
13. Master clocks a data byte to the slave and clocks out the slaves ACK on the 9th SCLx pulse; SSPxIF is set.
14. If SEN bit of SSPxCON2 is set, CKP is cleared by hardware and the clock is stretched.
15. Slave clears SSPxIF.
16. Slave reads the received byte from SSPxBUF clearing BF.
17. If SEN is set the slave sets CKP to release the SCLx.
18. Steps 13-17 repeat for each received byte.
19. Master sends Stop to end the transmission.

21.5.5 10-BIT ADDRESSING WITH ADDRESS OR DATA HOLD

Reception using 10-bit addressing with AHEN or DHEN set is the same as with 7-bit modes. The only difference is the need to update the SSPxADD register using the UA bit. All functionality, specifically when the CKP bit is cleared and SCLx line is held low are the same. Figure 21-21 can be used as a reference of a slave in 10-bit addressing with AHEN set.

Figure 21-22 shows a standard waveform for a slave transmitter in 10-bit Addressing mode.

PIC16(L)F1526/7

FIGURE 21-20: I²C SLAVE, 10-BIT ADDRESS, RECEPTION (SEN = 1, AHEN = 0, DHEN = 0)

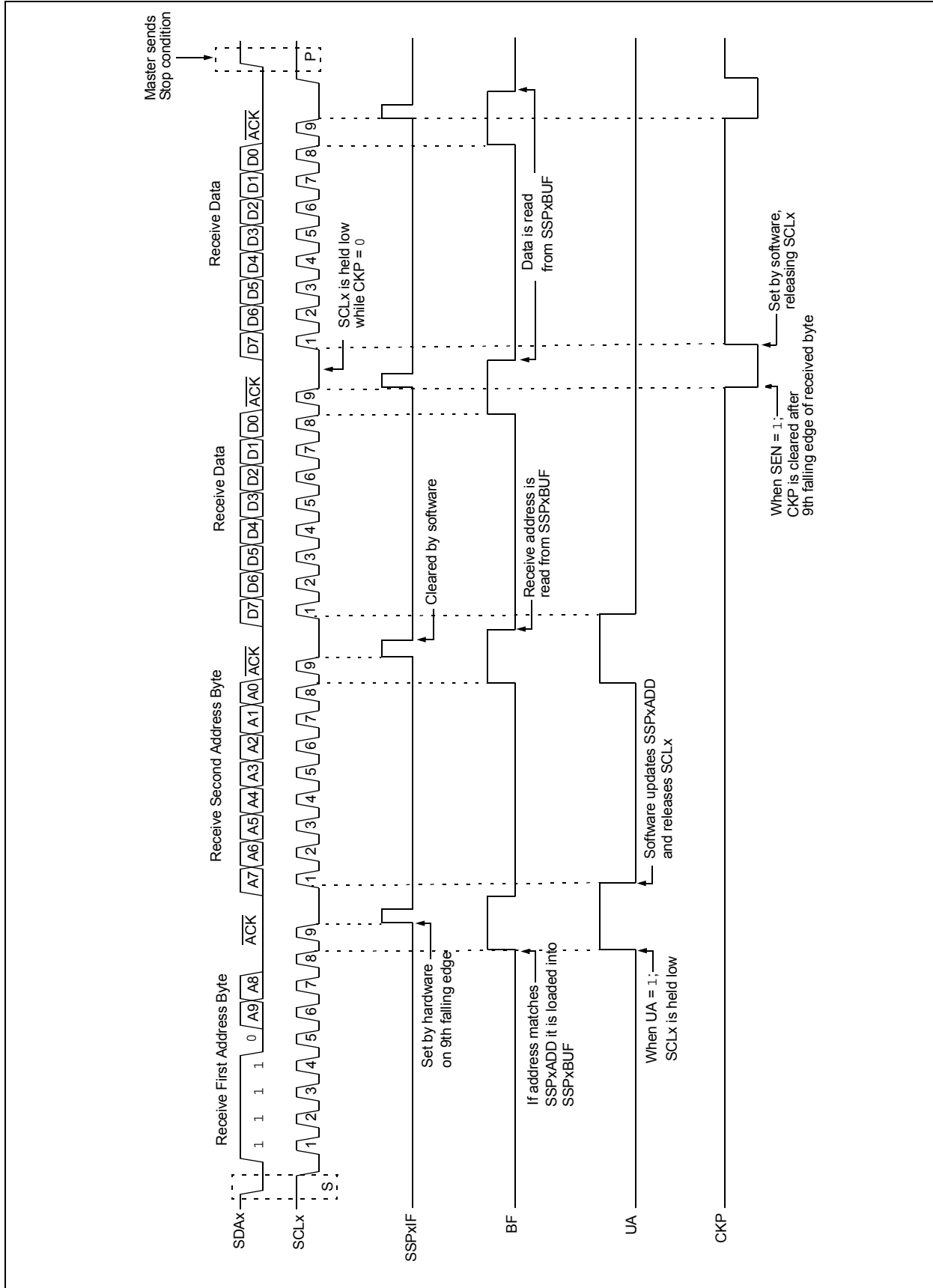
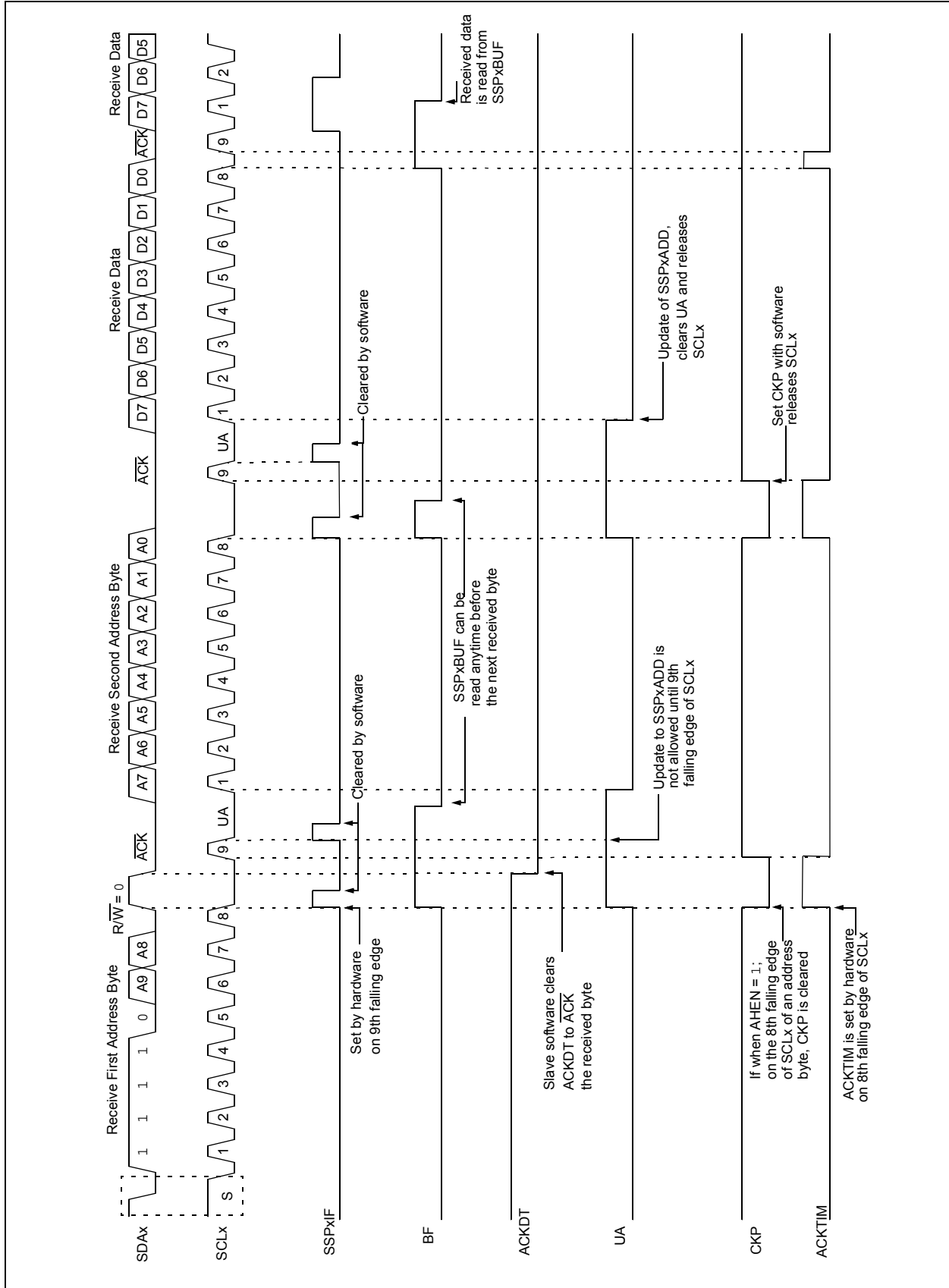
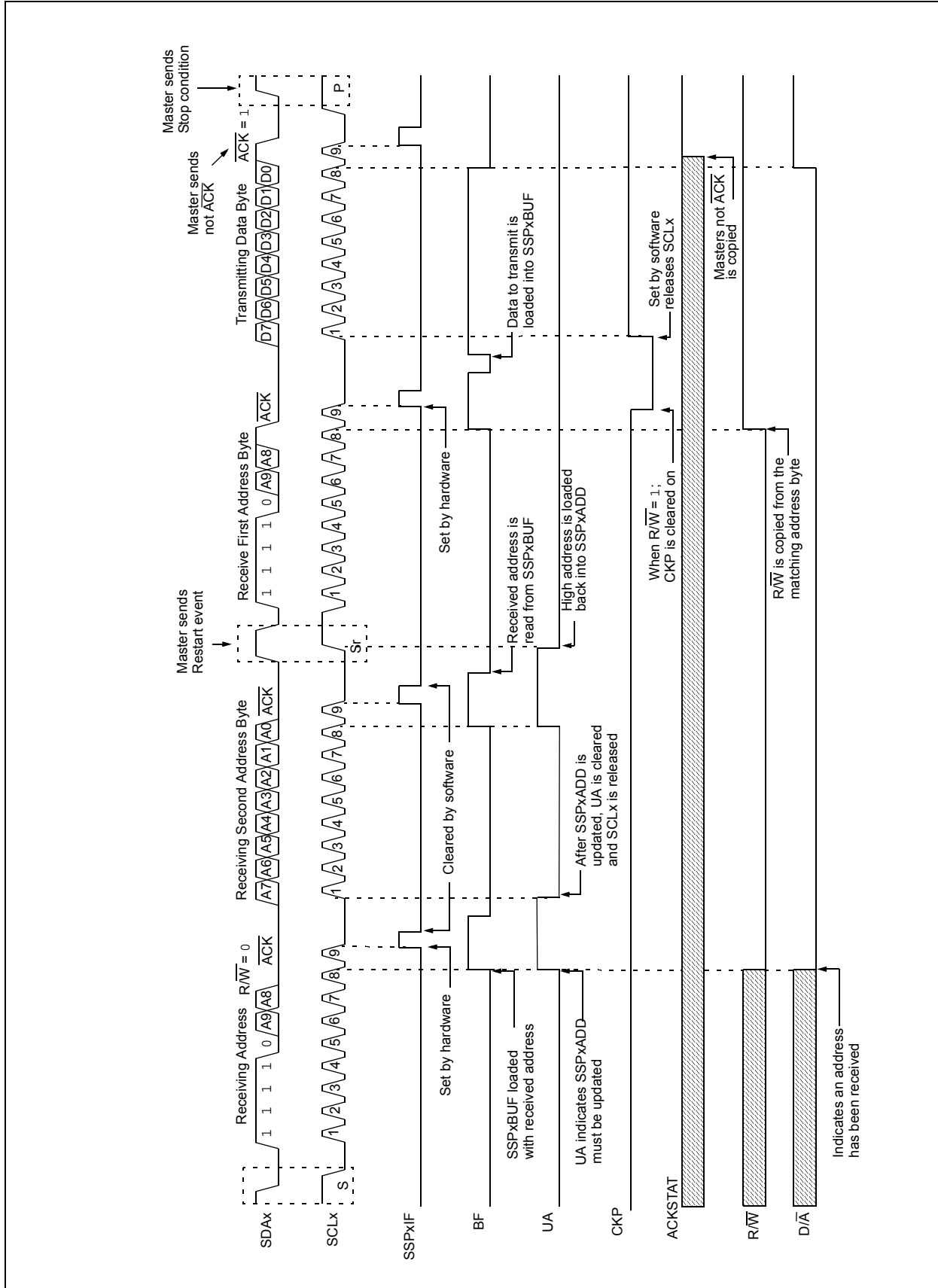


FIGURE 21-21: I²C SLAVE, 10-BIT ADDRESS, RECEPTION (SEN = 0, AHEN = 1, DHEN = 0)



PIC16(L)F1526/7

FIGURE 21-22: I²C SLAVE, 10-BIT ADDRESS, TRANSMISSION (SEN = 0, AHEN = 0, DHEN = 0)



21.5.6 CLOCK STRETCHING

Clock stretching occurs when a device on the bus holds the SCLx line low effectively pausing communication. The slave may stretch the clock to allow more time to handle data or prepare a response for the master device. A master device is not concerned with stretching as anytime it is active on the bus and not transferring data it is stretching. Any stretching done by a slave is invisible to the master software and handled by the hardware that generates SCLx.

The CKP bit of the SSPxCON1 register is used to control stretching in software. Any time the CKP bit is cleared, the module will wait for the SCLx line to go low and then hold it. Setting CKP will release SCLx and allow more communication.

21.5.6.1 Normal Clock Stretching

Following an $\overline{\text{ACK}}$ if the $\overline{\text{R/W}}$ bit of SSPxSTAT is set, a read request, the slave hardware will clear CKP. This allows the slave time to update SSPxBUF with data to transfer to the master. If the SEN bit of SSPxCON2 is set, the slave hardware will always stretch the clock after the $\overline{\text{ACK}}$ sequence. Once the slave is ready; CKP is set by software and communication resumes.

Note 1: The BF bit has no effect on if the clock will be stretched or not. This is different than previous versions of the module that would not stretch the clock, clear CKP, if SSPxBUF was read before the 9th falling edge of SCLx.

2: Previous versions of the module did not stretch the clock for a transmission if SSPxBUF was loaded before the 9th falling edge of SCLx. It is now always cleared for read requests.

21.5.6.2 10-bit Addressing Mode

In 10-bit Addressing mode, when the UA bit is set, the clock is always stretched. This is the only time the SCLx is stretched without CKP being cleared. SCLx is released immediately after a write to SSPxADD.

Note: Previous versions of the module did not stretch the clock if the second address byte did not match.

21.5.6.3 Byte NACKing

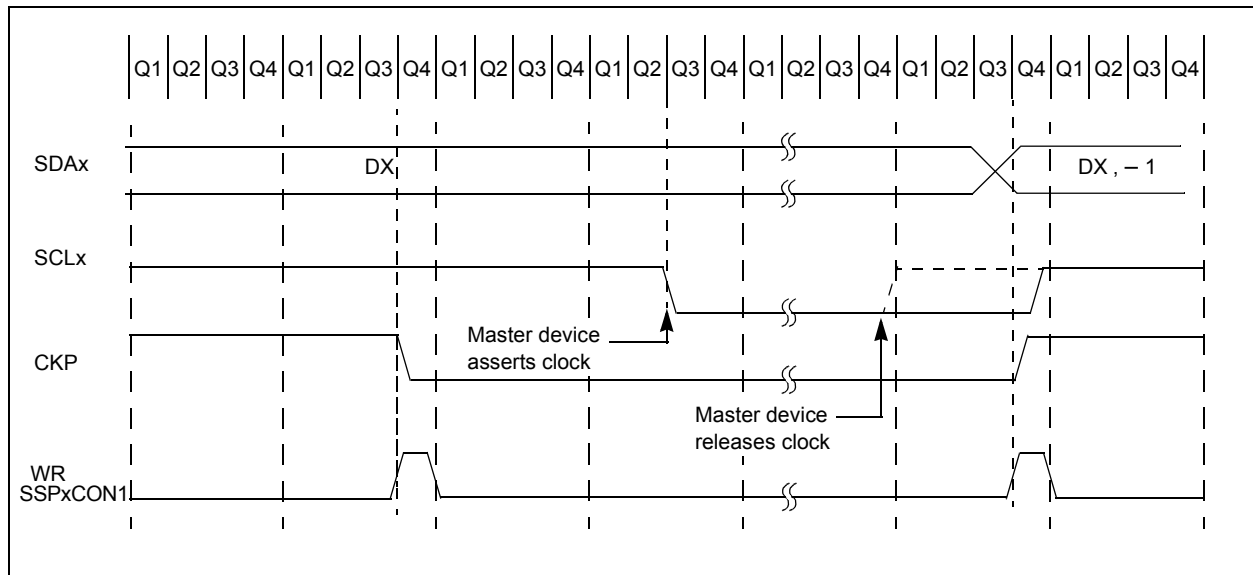
When AHEN bit of SSPxCON3 is set; CKP is cleared by hardware after the 8th falling edge of SCLx for a received matching address byte. When DHEN bit of SSPxCON3 is set; CKP is cleared after the 8th falling edge of SCLx for received data.

Stretching after the 8th falling edge of SCLx allows the slave to look at the received address or data and decide if it wants to ACK the received data.

21.5.7 CLOCK SYNCHRONIZATION AND THE CKP BIT

Any time the CKP bit is cleared, the module will wait for the SCLx line to go low and then hold it. However, clearing the CKP bit will not assert the SCLx output low until the SCLx output is already sampled low. Therefore, the CKP bit will not assert the SCLx line until an external I²C master device has already asserted the SCLx line. The SCLx output will remain low until the CKP bit is set and all other devices on the I²C bus have released SCLx. This ensures that a write to the CKP bit will not violate the minimum high time requirement for SCLx (see Figure 21-23).

FIGURE 21-23: CLOCK SYNCHRONIZATION TIMING



PIC16(L)F1526/7

21.5.8 GENERAL CALL ADDRESS SUPPORT

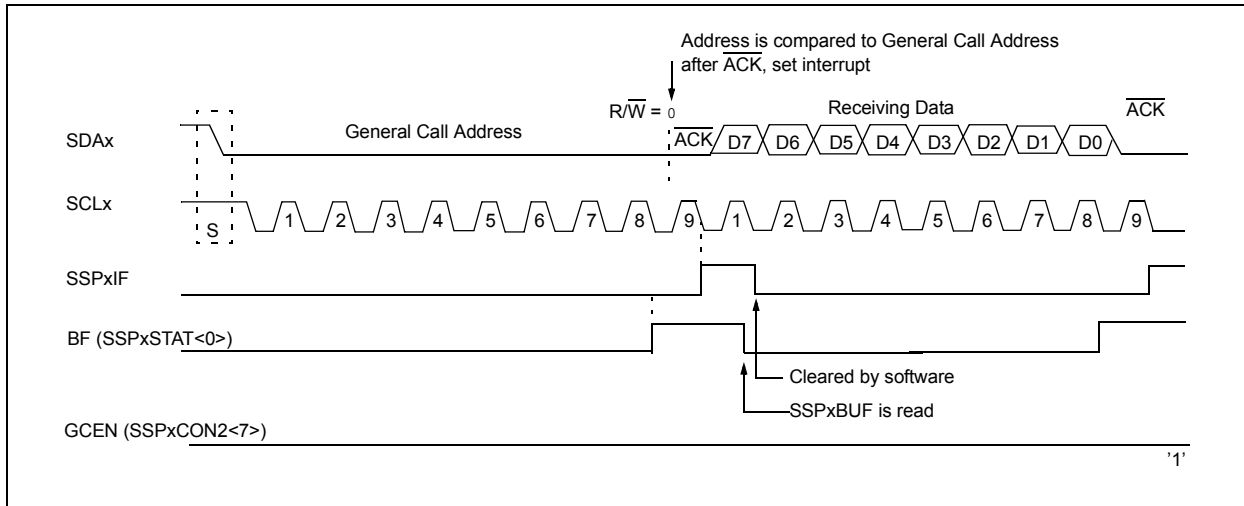
The addressing procedure for the I²C bus is such that the first byte after the Start condition usually determines which device will be the slave addressed by the master device. The exception is the general call address which can address all devices. When this address is used, all devices should, in theory, respond with an acknowledge.

The general call address is a reserved address in the I²C protocol, defined as address 0x00. When the GCEN bit of the SSPxCON2 register is set, the slave module will automatically ACK the reception of this address regardless of the value stored in SSPxADD. After the slave clocks in an address of all zeros with the R/W bit clear, an interrupt is generated and slave software can read SSPxBUF and respond. Figure 21-24 shows a general call reception sequence.

In 10-bit Address mode, the UA bit will not be set on the reception of the general call address. The slave will prepare to receive the second byte as data, just as it would in 7-bit mode.

If the AHEN bit of the SSPxCON3 register is set, just as with any other address reception, the slave hardware will stretch the clock after the 8th falling edge of SCLx. The slave must then set its ACKDT value and release the clock with communication progressing as it would normally.

FIGURE 21-24: SLAVE MODE GENERAL CALL ADDRESS SEQUENCE



21.5.9 SSPX MASK REGISTER

An SSPx Mask (SSPxMSK) register (Register 21-5) is available in I²C Slave mode as a mask for the value held in the SSPxSR register during an address comparison operation. A zero ('0') bit in the SSPxMSK register has the effect of making the corresponding bit of the received address a "don't care".

This register is reset to all '1's upon any Reset condition and, therefore, has no effect on standard SSPx operation until written with a mask value.

The SSPx Mask register is active during:

- 7-bit Address mode: address compare of A<7:1>.
- 10-bit Address mode: address compare of A<7:0> only. The SSPx mask has no effect during the reception of the first (high) byte of the address.

21.6 I²C MASTER MODE

Master mode is enabled by setting and clearing the appropriate SSPM bits in the SSPxCON1 register and by setting the SSPEN bit. In Master mode, the SDA and SCK pins must be configured as inputs. The MSSP peripheral hardware will override the output driver TRIS controls when necessary to drive the pins low.

Master mode of operation is supported by interrupt generation on the detection of the Start and Stop conditions. The Stop (P) and Start (S) bits are cleared from a Reset or when the MSSPx module is disabled. Control of the I²C bus may be taken when the P bit is set, or the bus is Idle.

In Firmware Controlled Master mode, user code conducts all I²C bus operations based on Start and Stop bit condition detection. Start and Stop condition detection is the only active circuitry in this mode. All other communication is done by the user software directly manipulating the SDAX and SCLx lines.

The following events will cause the SSPx Interrupt Flag bit, SSPxIF, to be set (SSPx interrupt, if enabled):

- Start condition detected
- Stop condition detected
- Data transfer byte transmitted/received
- Acknowledge transmitted/received
- Repeated Start generated

Note 1: The MSSPx module, when configured in I²C Master mode, does not allow queuing of events. For instance, the user is not allowed to initiate a Start condition and immediately write the SSPxBUF register to initiate transmission before the Start condition is complete. In this case, the SSPxBUF will not be written to and the WCOL bit will be set, indicating that a write to the SSPxBUF did not occur

2: When in Master mode, Start/Stop detection is masked and an interrupt is generated when the SEN/PEN bit is cleared and the generation is complete.

21.6.1 I²C MASTER MODE 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 SDAX, while SCLx outputs the serial clock. The first byte transmitted contains the slave address of the receiving device (7 bits) and the Read/Write (R/W) bit. In this case, the R/W bit will be logic '0'. Serial data is transmitted 8 bits at a time. After each byte is transmitted, an Acknowledge 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 R/W bit. In this case, the R/W bit will be logic '1'. Thus, the first byte transmitted is a 7-bit slave address followed by a '1' to indicate the receive bit. Serial data is received via SDAX, while SCLx outputs the serial clock. Serial data is received 8 bits at a time. After each byte is received, an Acknowledge bit is transmitted. Start and Stop conditions indicate the beginning and end of transmission.

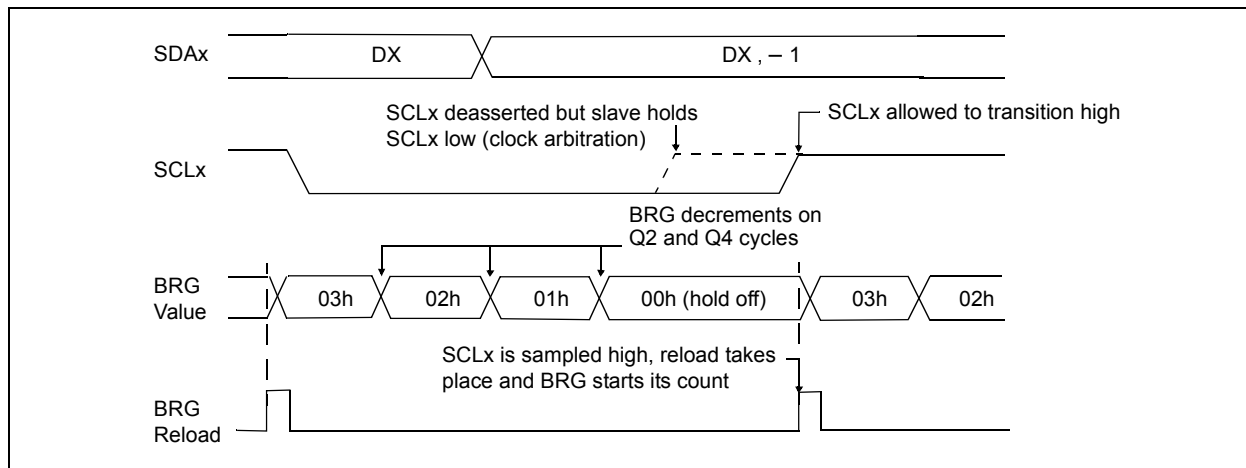
A Baud Rate Generator is used to set the clock frequency output on SCLx. See [Section 21.7 "Baud Rate Generator"](#) for more detail.

PIC16(L)F1526/7

21.6.2 CLOCK ARBITRATION

Clock arbitration occurs when the master, during any receive, transmit or Repeated Start/Stop condition, releases the SCLx pin (SCLx allowed to float high). When the SCLx pin is allowed to float high, the Baud Rate Generator (BRG) is suspended from counting until the SCLx pin is actually sampled high. When the SCLx pin is sampled high, the Baud Rate Generator is reloaded with the contents of SSPxADD<7:0> and begins counting. This ensures that the SCLx high time will always be at least one BRG rollover count in the event that the clock is held low by an external device (Figure 21-25).

FIGURE 21-25: BAUD RATE GENERATOR TIMING WITH CLOCK ARBITRATION



21.6.3 WCOL STATUS FLAG

If the user writes the SSPxBUF when a Start, Restart, Stop, Receive or Transmit sequence is in progress, the WCOL is set and the contents of the buffer are unchanged (the write does not occur). Any time the WCOL bit is set it indicates that an action on SSPxBUF was attempted while the module was not Idle.

Note: Because queueing of events is not allowed, writing to the lower 5 bits of SSPxCON2 is disabled until the Start condition is complete.

21.6.4 I²C MASTER MODE START CONDITION TIMING

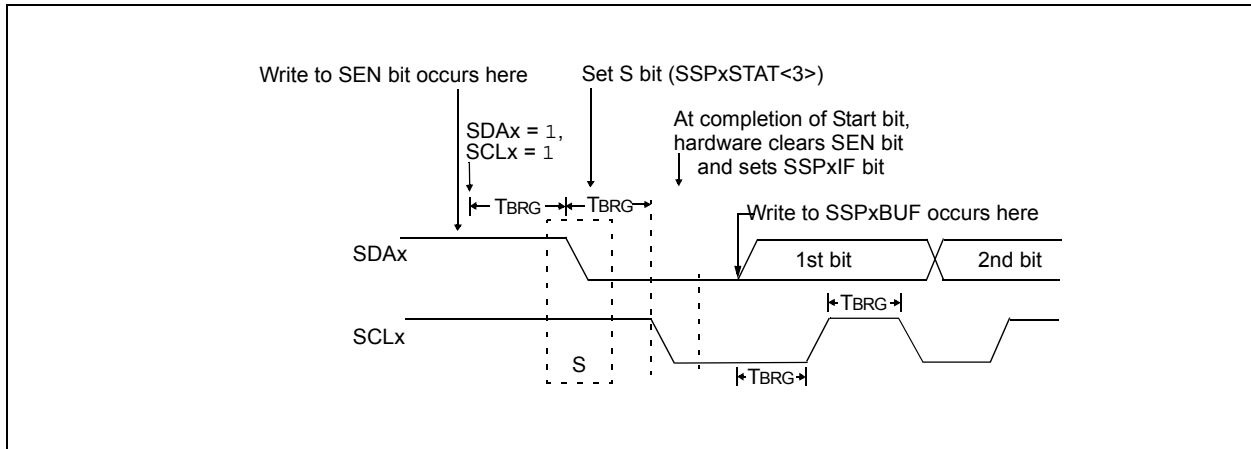
To initiate a Start condition (Figure 21-26), the user sets the Start Enable bit, SEN bit of the SSPxCON2 register. If the SDAx and SCLx pins are sampled high, the Baud Rate Generator is reloaded with the contents of SSPxADD<7:0> and starts its count. If SCLx and SDAx are both sampled high when the Baud Rate Generator times out (TBRG), the SDAx pin is driven low. The action of the SDAx being driven low while SCLx is high is the Start condition and causes the S bit of the SSPxSTAT1 register to be set. Following this, the Baud Rate Generator is reloaded with the contents of SSPxADD<7:0> and resumes its count. When the Baud Rate Generator times out (TBRG), the SEN bit of the SSPxCON2 register will be automatically cleared

by hardware; the Baud Rate Generator is suspended, leaving the SDAx line held low and the Start condition is complete.

Note 1: If at the beginning of the Start condition, the SDAx and SCLx pins are already sampled low, or if during the Start condition, the SCLx line is sampled low before the SDAx line is driven low, a bus collision occurs, the Bus Collision Interrupt Flag, BCLxIF, is set, the Start condition is aborted and the I²C module is reset into its Idle state.

2: The Philips I²C Specification states that a bus collision cannot occur on a Start.

FIGURE 21-26: FIRST START BIT TIMING



21.6.6 I²C MASTER MODE TRANSMISSION

Transmission of a data byte, a 7-bit address or the other half of a 10-bit address is accomplished by simply writing a value to the SSPxBUF register. This action will set the Buffer Full flag bit, BF 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 SDAx pin after the falling edge of SCLx is asserted. SCLx is held low for one Baud Rate Generator rollover count (TBRG). Data should be valid before SCLx is released high. When the SCLx pin is released high, it is held that way for TBRG. The data on the SDAx pin must remain stable for that duration and some hold time after the next falling edge of SCLx. After the eighth bit is shifted out (the falling edge of the eighth clock), the BF flag is cleared and the master releases SDAx. This allows the slave device being addressed to respond with an ACK bit during the ninth bit time if an address match occurred, or if data was received properly. The status of $\overline{\text{ACK}}$ is written into the ACKSTAT bit on the rising edge of the ninth clock. If the master receives an Acknowledge, the Acknowledge Status bit, ACKSTAT, is cleared. If not, the bit is set. After the ninth clock, the SSPxIF bit is set and the master clock (Baud Rate Generator) is suspended until the next data byte is loaded into the SSPxBUF, leaving SCLx low and SDAx unchanged (Figure 21-28).

After the write to the SSPxBUF, each bit of the address will be shifted out on the falling edge of SCLx until all seven address bits and the R/W bit are completed. On the falling edge of the eighth clock, the master will release the SDAx pin, allowing the slave to respond with an Acknowledge. On the falling edge of the ninth clock, the master will sample the SDAx pin to see if the address was recognized by a slave. The status of the $\overline{\text{ACK}}$ bit is loaded into the ACKSTAT Status bit of the SSPxCON2 register. Following the falling edge of the ninth clock transmission of the address, the SSPxIF is set, the BF flag is cleared and the Baud Rate Generator is turned off until another write to the SSPxBUF takes place, holding SCLx low and allowing SDAx to float.

21.6.6.1 BF Status Flag

In Transmit mode, the BF bit of the SSPxSTAT register is set when the CPU writes to SSPxBUF and is cleared when all 8 bits are shifted out.

21.6.6.2 WCOL Status Flag

If the user writes the SSPxBUF when a transmit is already in progress (i.e., SSPxSR is still shifting out a data byte), the WCOL bit is set and the contents of the buffer are unchanged (the write does not occur).

WCOL must be cleared by software before the next transmission.

21.6.6.3 ACKSTAT Status Flag

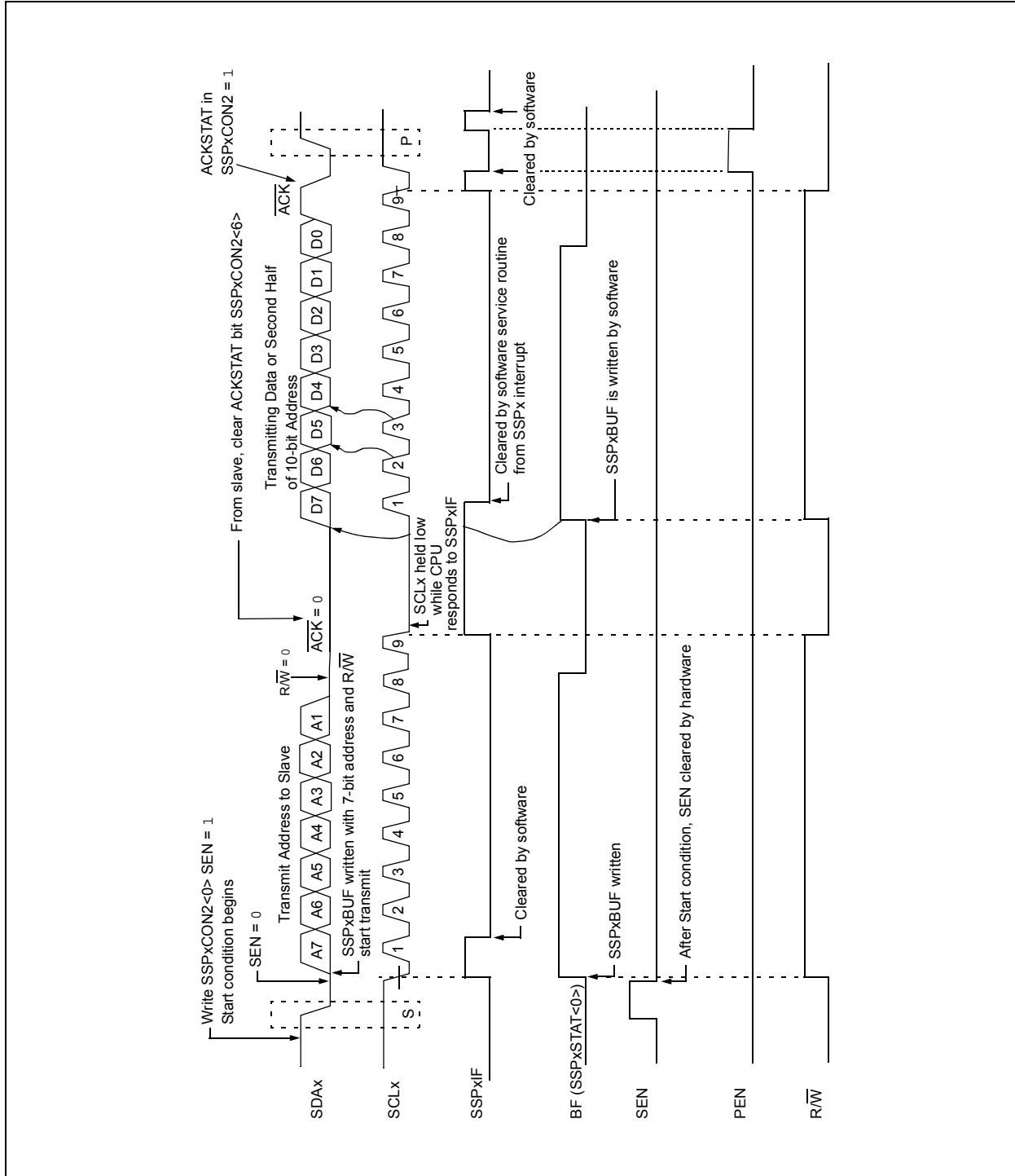
In Transmit mode, the ACKSTAT bit of the SSPxCON2 register is cleared when the slave has sent an Acknowledge ($\text{ACK} = 0$) and is set when the slave does not Acknowledge ($\text{ACK} = 1$). A slave sends an Acknowledge when it has recognized its address (including a general call), or when the slave has properly received its data.

21.6.6.4 Typical transmit sequence:

1. The user generates a Start condition by setting the SEN bit of the SSPxCON2 register.
2. SSPxIF is set by hardware on completion of the Start.
3. SSPxIF is cleared by software.
4. The MSSPx module will wait the required start time before any other operation takes place.
5. The user loads the SSPxBUF with the slave address to transmit.
6. Address is shifted out the SDAx pin until all 8 bits are transmitted. Transmission begins as soon as SSPxBUF is written to.
7. The MSSPx module shifts in the $\overline{\text{ACK}}$ bit from the slave device and writes its value into the ACKSTAT bit of the SSPxCON2 register.
8. The MSSPx module generates an interrupt at the end of the ninth clock cycle by setting the SSPxIF bit.
9. The user loads the SSPxBUF with eight bits of data.
10. Data is shifted out the SDAx pin until all 8 bits are transmitted.
11. The MSSPx module shifts in the $\overline{\text{ACK}}$ bit from the slave device and writes its value into the ACKSTAT bit of the SSPxCON2 register.
12. Steps 8-11 are repeated for all transmitted data bytes.
13. The user generates a Stop or Restart condition by setting the PEN or RSEN bits of the SSPxCON2 register. Interrupt is generated once the Stop/Restart condition is complete.

PIC16(L)F1526/7

FIGURE 21-28: I²C MASTER MODE WAVEFORM (TRANSMISSION, 7 OR 10-BIT ADDRESS)



21.6.7 I²C MASTER MODE RECEPTION

Master mode reception (Figure 21-29) is enabled by programming the Receive Enable bit, RCEN bit of the SSPxCON2 register.

Note: The MSSPx module must be in an Idle state before the RCEN bit is set or the RCEN bit will be disregarded.

The Baud Rate Generator begins counting and on each rollover, the state of the SCLx pin changes (high-to-low/low-to-high) and data is shifted into the SSPxSR. After the falling edge of the eighth clock, the receive enable flag is automatically cleared, the contents of the SSPxSR are loaded into the SSPxBUF, the BF flag bit is set, the SSPxIF flag bit is set and the Baud Rate Generator is suspended from counting, holding SCLx low. The MSSPx is now in Idle state awaiting the next command. When the buffer is read by the CPU, the BF flag bit is automatically cleared. The user can then send an Acknowledge bit at the end of reception by setting the Acknowledge Sequence Enable, ACKEN bit of the SSPxCON2 register.

21.6.7.1 BF Status Flag

In receive operation, the BF bit is set when an address or data byte is loaded into SSPxBUF from SSPxSR. It is cleared when the SSPxBUF register is read.

21.6.7.2 SSPOV Status Flag

In receive operation, the SSPOV bit is set when 8 bits are received into the SSPxSR and the BF flag bit is already set from a previous reception.

21.6.7.3 WCOL Status Flag

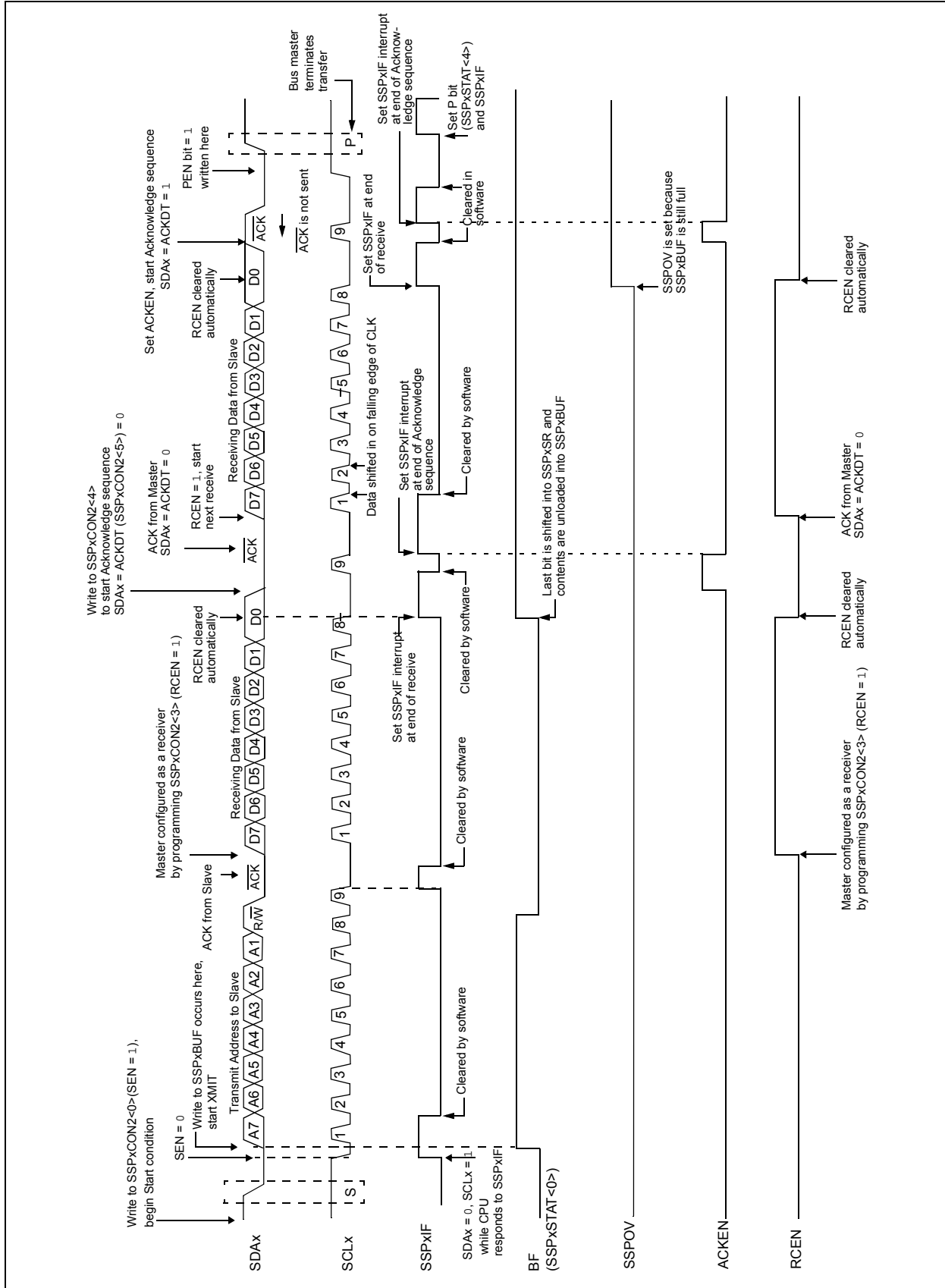
If the user writes the SSPxBUF when a receive is already in progress (i.e., SSPxSR is still shifting in a data byte), the WCOL bit is set and the contents of the buffer are unchanged (the write does not occur).

21.6.7.4 Typical Receive Sequence:

1. The user generates a Start condition by setting the SEN bit of the SSPxCON2 register.
2. SSPxIF is set by hardware on completion of the Start.
3. SSPxIF is cleared by software.
4. User writes SSPxBUF with the slave address to transmit and the R/W bit set.
5. Address is shifted out the SDAx pin until all 8 bits are transmitted. Transmission begins as soon as SSPxBUF is written to.
6. The MSSPx module shifts in the \overline{ACK} bit from the slave device and writes its value into the ACKSTAT bit of the SSPxCON2 register.
7. The MSSPx module generates an interrupt at the end of the ninth clock cycle by setting the SSPxIF bit.
8. User sets the RCEN bit of the SSPxCON2 register and the master clocks in a byte from the slave.
9. After the 8th falling edge of SCLx, SSPxIF and BF are set.
10. Master clears SSPxIF and reads the received byte from SSPxUF, clears BF.
11. Master sets \overline{ACK} value sent to slave in ACKDT bit of the SSPxCON2 register and initiates the \overline{ACK} by setting the ACKEN bit.
12. Masters \overline{ACK} is clocked out to the slave and SSPxIF is set.
13. User clears SSPxIF.
14. Steps 8-13 are repeated for each received byte from the slave.
15. Master sends a not \overline{ACK} or Stop to end communication.

PIC16(L)F1526/7

FIGURE 21-29: I²C MASTER MODE WAVEFORM (RECEPTION, 7-BIT ADDRESS)



21.6.8 ACKNOWLEDGE SEQUENCE TIMING

An Acknowledge sequence is enabled by setting the Acknowledge Sequence Enable bit, ACKEN bit of the SSPxCON2 register. When this bit is set, the SCLx pin is pulled low and the contents of the Acknowledge data bit are presented on the SDAx pin. If the user wishes to generate an Acknowledge, then the ACKDT bit should be cleared. If not, the user should set the ACKDT bit before starting an Acknowledge sequence. The Baud Rate Generator then counts for one rollover period (TBRG) and the SCLx pin is deasserted (pulled high). When the SCLx pin is sampled high (clock arbitration), the Baud Rate Generator counts for TBRG. The SCLx pin is then pulled low. Following this, the ACKEN bit is automatically cleared, the Baud Rate Generator is turned off and the MSSPx module then goes into Idle mode (Figure 21-30).

21.6.8.1 WCOL Status Flag

If the user writes the SSPxBUF when an Acknowledge sequence is in progress, then the WCOL bit is set and the contents of the buffer are unchanged (the write does not occur).

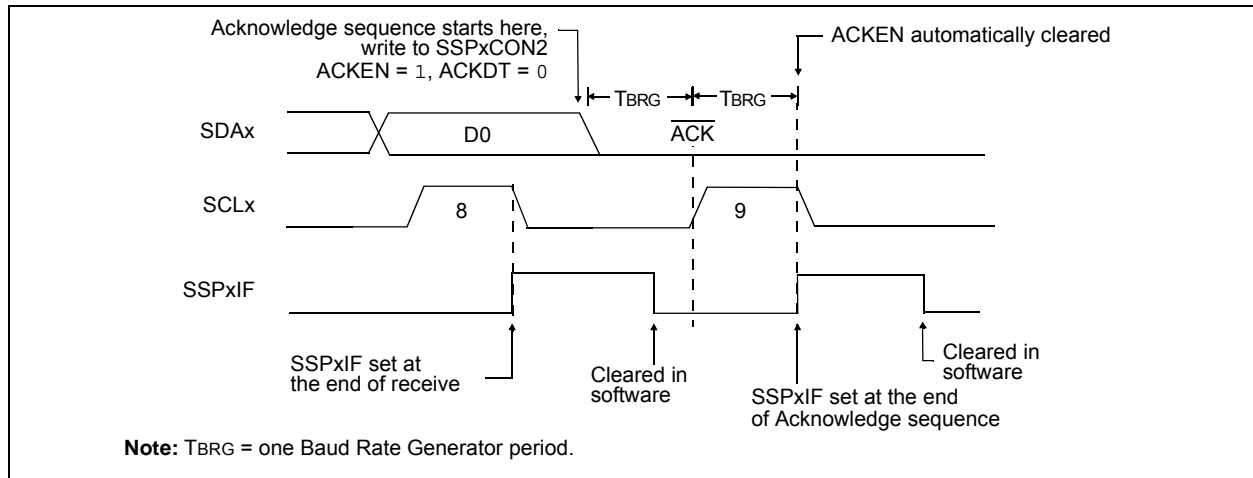
21.6.9 STOP CONDITION TIMING

A Stop bit is asserted on the SDAx pin at the end of a receive/transmit by setting the Stop Sequence Enable bit, PEN bit of the SSPxCON2 register. At the end of a receive/transmit, the SCLx line is held low after the falling edge of the ninth clock. When the PEN bit is set, the master will assert the SDAx line low. When the SDAx line is sampled low, the Baud Rate Generator is reloaded and counts down to '0'. When the Baud Rate Generator times out, the SCLx pin will be brought high and one TBRG (Baud Rate Generator rollover count) later, the SDAx pin will be deasserted. When the SDAx pin is sampled high while SCLx is high, the P bit of the SSPxSTAT register is set. A TBRG later, the PEN bit is cleared and the SSPxIF bit is set (Figure 21-31).

21.6.9.1 WCOL Status Flag

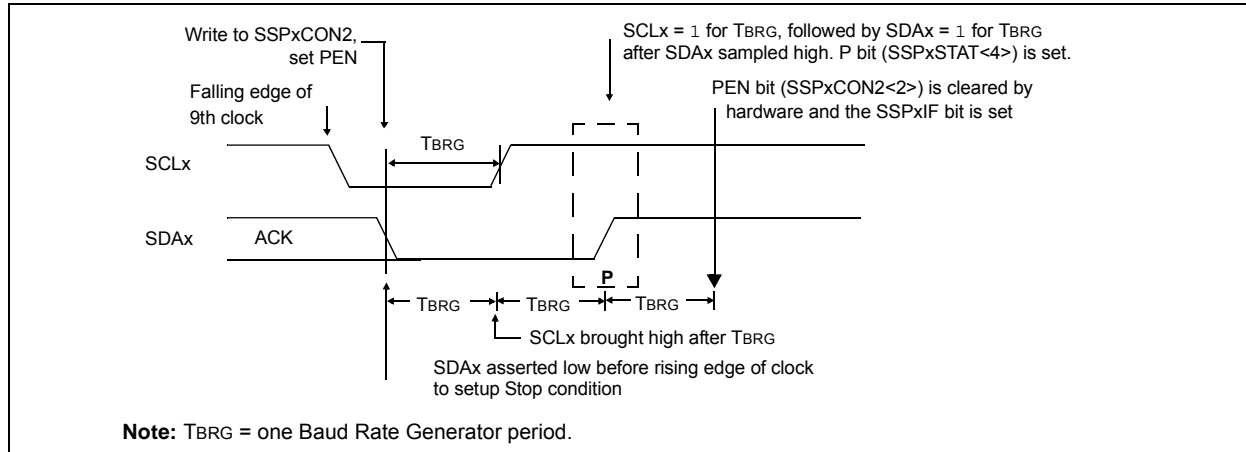
If the user writes the SSPxBUF when a Stop sequence is in progress, then the WCOL bit is set and the contents of the buffer are unchanged (the write does not occur).

FIGURE 21-30: ACKNOWLEDGE SEQUENCE WAVEFORM



PIC16(L)F1526/7

FIGURE 21-31: STOP CONDITION RECEIVE OR TRANSMIT MODE



21.6.10 SLEEP OPERATION

While in Sleep mode, the I²C slave module can receive addresses or data and when an address match or complete byte transfer occurs, wake the processor from Sleep (if the MSSPx interrupt is enabled).

21.6.11 EFFECTS OF A RESET

A Reset disables the MSSPx module and terminates the current transfer.

21.6.12 MULTI-MASTER MODE

In Multi-Master mode, the interrupt generation on the detection of the Start and Stop conditions allows the determination of when the bus is free. The Stop (P) and Start (S) bits are cleared from a Reset or when the MSSPx module is disabled. Control of the I²C bus may be taken when the P bit of the SSPxSTAT register is set, or the bus is Idle, with both the S and P bits clear. When the bus is busy, enabling the SSPx interrupt will generate the interrupt when the Stop condition occurs.

In multi-master operation, the SDAx line must be monitored for arbitration to see if the signal level is the expected output level. This check is performed by hardware with the result placed in the BCLxIF bit.

The states where arbitration can be lost are:

- Address Transfer
- Data Transfer
- A Start Condition
- A Repeated Start Condition
- An Acknowledge Condition

21.6.13 MULTI-MASTER COMMUNICATION, BUS COLLISION AND BUS ARBITRATION

Multi-Master mode support is achieved by bus arbitration. When the master outputs address/data bits onto the SDAx pin, arbitration takes place when the master outputs a '1' on SDAx, by letting SDAx float high and another master asserts a '0'. When the SCLx pin floats high, data should be stable. If the expected data on SDAx is a '1' and the data sampled on the SDAx pin is '0', then a bus collision has taken place. The master will set the Bus Collision Interrupt Flag, BCLxIF and reset the I²C port to its Idle state (Figure 21-32).

If a transmit was in progress when the bus collision occurred, the transmission is halted, the BF flag is cleared, the SDAx and SCLx lines are deasserted and the SSPxBUF can be written to. 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.

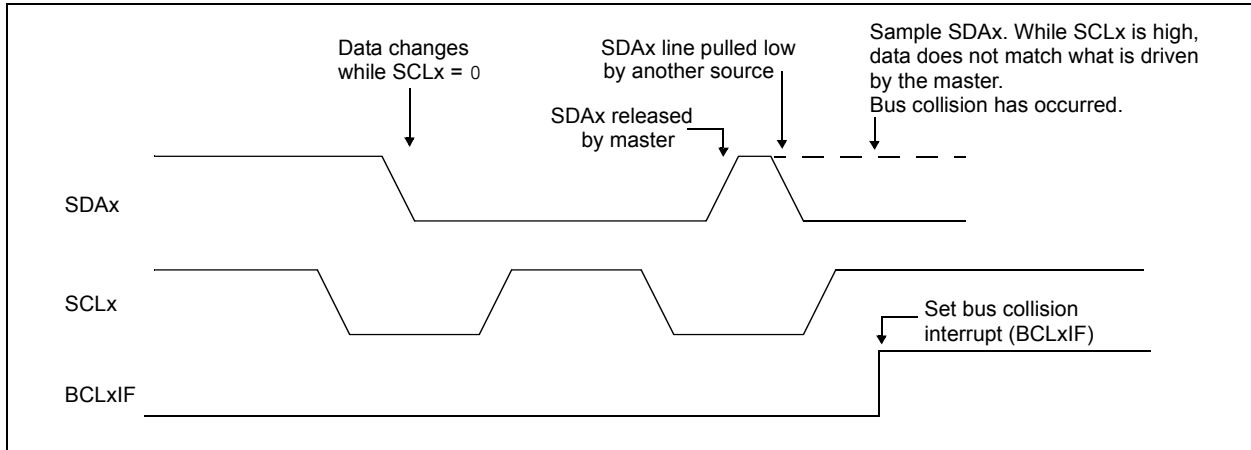
If a Start, Repeated Start, Stop or Acknowledge condition was in progress when the bus collision occurred, the condition is aborted, the SDAx and SCLx lines are deasserted and the respective control bits in the SSPxCON2 register are cleared. When the user services the bus collision Interrupt Service 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 SDAx and SCLx pins. If a Stop condition occurs, the SSPxIF bit will be set.

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

In Multi-Master mode, 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 SSPxSTAT register, or the bus is Idle and the S and P bits are cleared.

FIGURE 21-32: BUS COLLISION TIMING FOR TRANSMIT AND ACKNOWLEDGE



PIC16(L)F1526/7

21.6.13.1 Bus Collision During a Start Condition

During a Start condition, a bus collision occurs if:

- SDAx or SCLx are sampled low at the beginning of the Start condition (Figure 21-33).
- SCLx is sampled low before SDAx is asserted low (Figure 21-34).

During a Start condition, both the SDAx and the SCLx pins are monitored.

If the SDAx pin is already low, or the SCLx pin is already low, then all of the following occur:

- the Start condition is aborted,
- the BCLxIF flag is set and
- the MSSPx module is reset to its Idle state (Figure 21-33).

The Start condition begins with the SDAx and SCLx pins deasserted. When the SDAx pin is sampled high, the Baud Rate Generator is loaded and counts down. If the SCLx pin is sampled low while SDAx is high, a bus collision occurs because it is assumed that another master is attempting to drive a data '1' during the Start condition.

If the SDAx pin is sampled low during this count, the BRG is reset and the SDAx line is asserted early (Figure 21-35). If, however, a '1' is sampled on the

SDAx pin, the SDAx pin is asserted low at the end of the BRG count. The Baud Rate Generator is then reloaded and counts down to zero; if the SCLx pin is sampled as '0' during this time, a bus collision does not occur. At the end of the BRG count, the SCLx pin is asserted low.

Note: The reason that bus collision is not a factor during a Start condition is that no two bus masters can assert a Start condition at the exact same time. Therefore, one master will always assert SDAx before the other. This condition does not cause a bus collision because the two masters must be allowed to arbitrate the first address following the Start condition. If the address is the same, arbitration must be allowed to continue into the data portion, Repeated Start or Stop conditions.

FIGURE 21-33: BUS COLLISION DURING START CONDITION (SDAx ONLY)

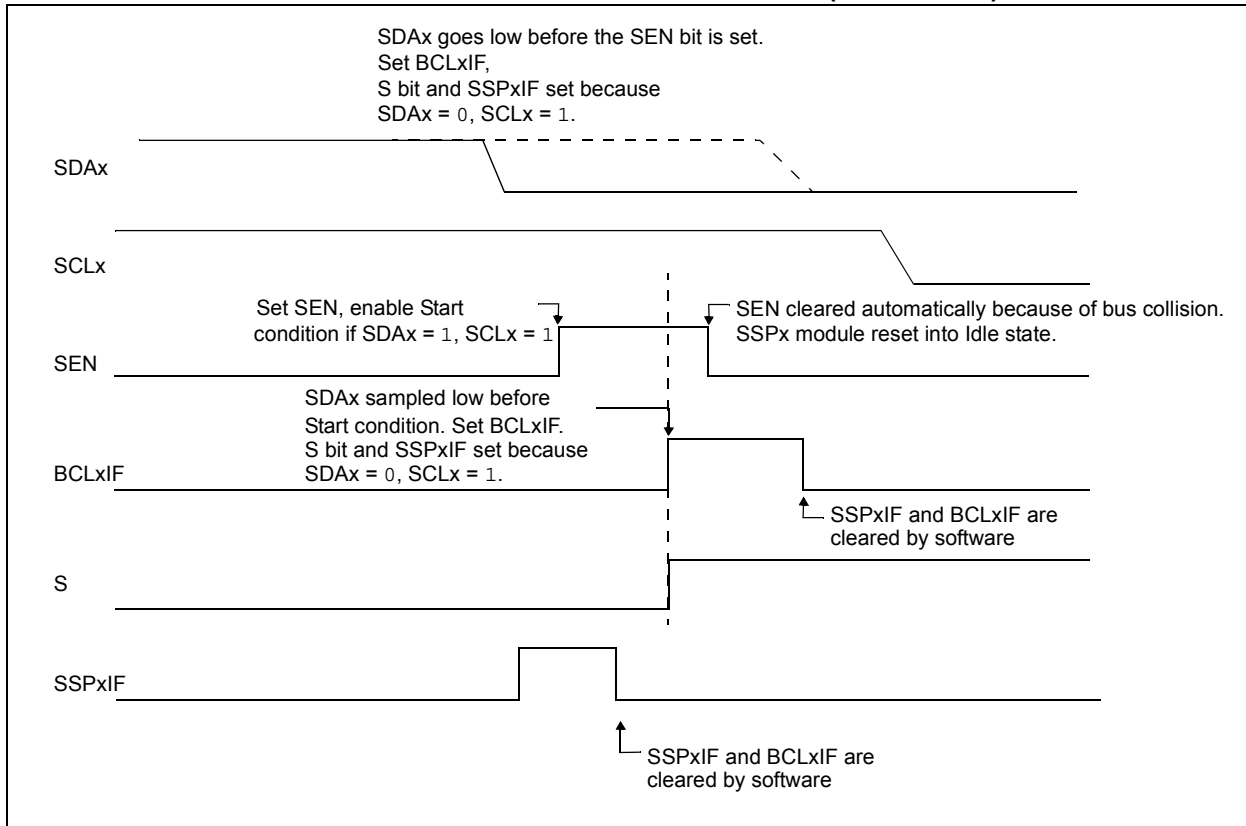


FIGURE 21-34: BUS COLLISION DURING START CONDITION (SCLX = 0)

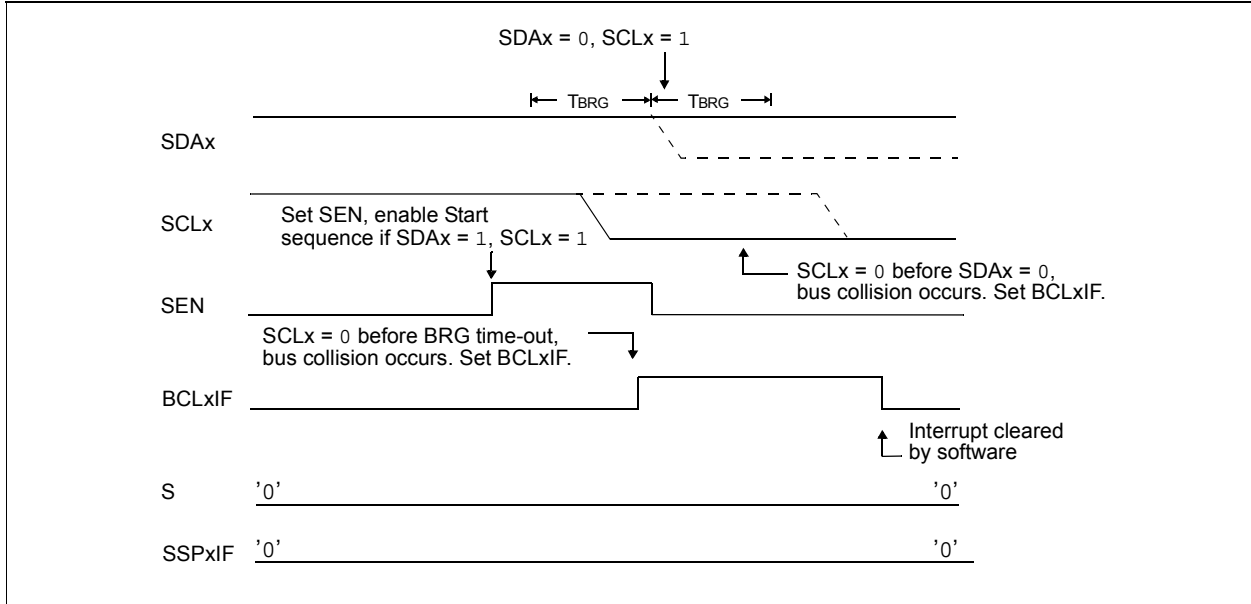
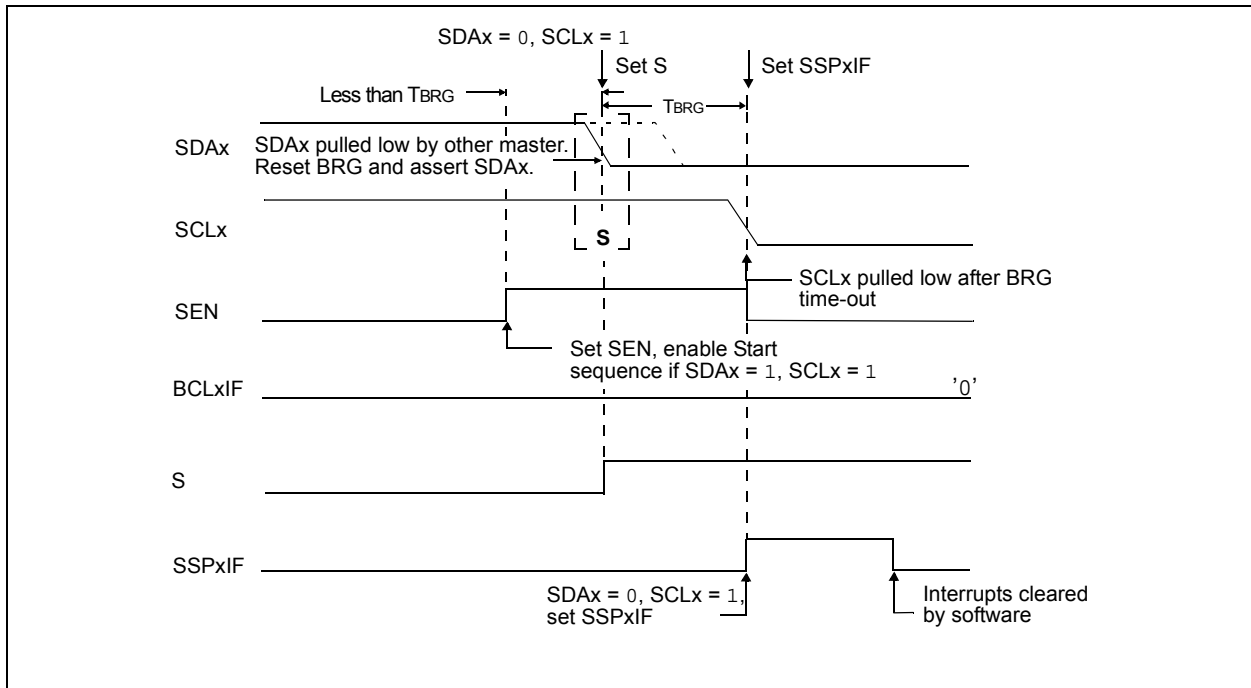


FIGURE 21-35: BRG RESET DUE TO SDA ARBITRATION DURING START CONDITION



PIC16(L)F1526/7

21.6.13.2 Bus Collision During a Repeated Start Condition

During a Repeated Start condition, a bus collision occurs if:

- A low level is sampled on SDAx when SCLx goes from low level to high level (Case 1).
- SCLx goes low before SDAx is asserted low, indicating that another master is attempting to transmit a data '1' (Case 2).

When the user releases SDAx and the pin is allowed to float high, the BRG is loaded with SSPxADD and counts down to zero. The SCLx pin is then deasserted and when sampled high, the SDAx pin is sampled.

If SDAx is low, a bus collision has occurred (i.e., another master is attempting to transmit a data '0', [Figure 21-36](#)). If SDAx is sampled high, the BRG is reloaded and begins counting. If SDAx goes from high-to-low before the BRG times out, no bus collision occurs because no two masters can assert SDAx at exactly the same time.

If SCLx goes from high-to-low before the BRG times out and SDAx has not already been asserted, a bus collision occurs. In this case, another master is attempting to transmit a data '1' during the Repeated Start condition, see [Figure 21-37](#).

If, at the end of the BRG time-out, both SCLx and SDAx are still high, the SDAx pin is driven low and the BRG is reloaded and begins counting. At the end of the count, regardless of the status of the SCLx pin, the SCLx pin is driven low and the Repeated Start condition is complete.

FIGURE 21-36: BUS COLLISION DURING A REPEATED START CONDITION (CASE 1)

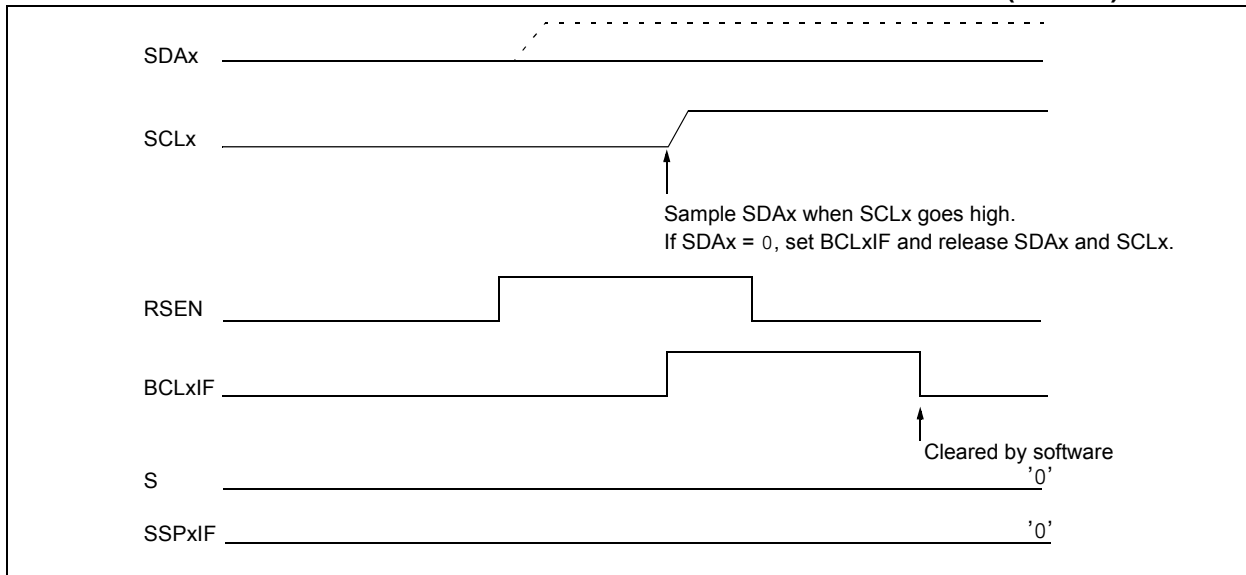
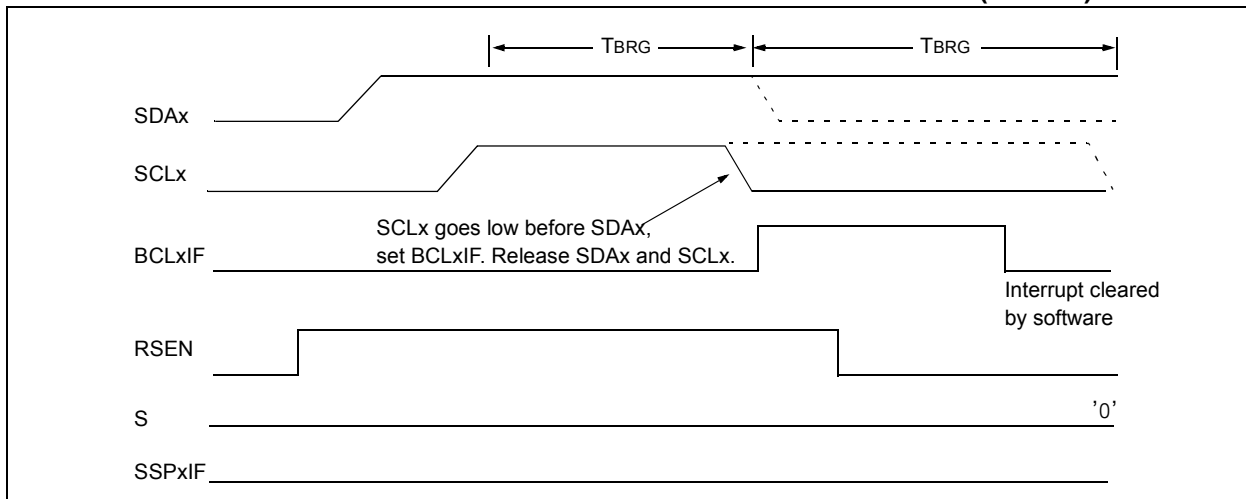


FIGURE 21-37: BUS COLLISION DURING REPEATED START CONDITION (CASE 2)



21.6.13.3 Bus Collision During a Stop Condition

Bus collision occurs during a Stop condition if:

- After the SDAx pin has been deasserted and allowed to float high, SDAx is sampled low after the BRG has timed out (Case 1).
- After the SCLx pin is deasserted, SCLx is sampled low before SDAx goes high (Case 2).

The Stop condition begins with SDAx asserted low. When SDAx is sampled low, the SCLx pin is allowed to float. When the pin is sampled high (clock arbitration), the Baud Rate Generator is loaded with SSPxADD and counts down to 0. After the BRG times out, SDAx is sampled. If SDAx is sampled low, a bus collision has occurred. This is due to another master attempting to drive a data '0' (Figure 21-38). If the SCLx pin is sampled low before SDAx is allowed to float high, a bus collision occurs. This is another case of another master attempting to drive a data '0' (Figure 21-39).

FIGURE 21-38: BUS COLLISION DURING A STOP CONDITION (CASE 1)

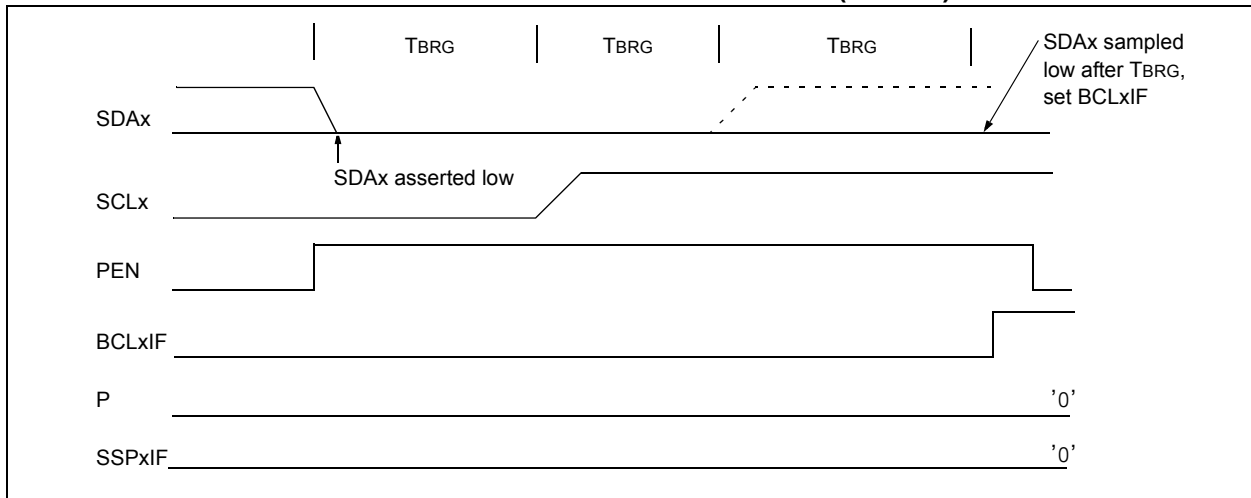
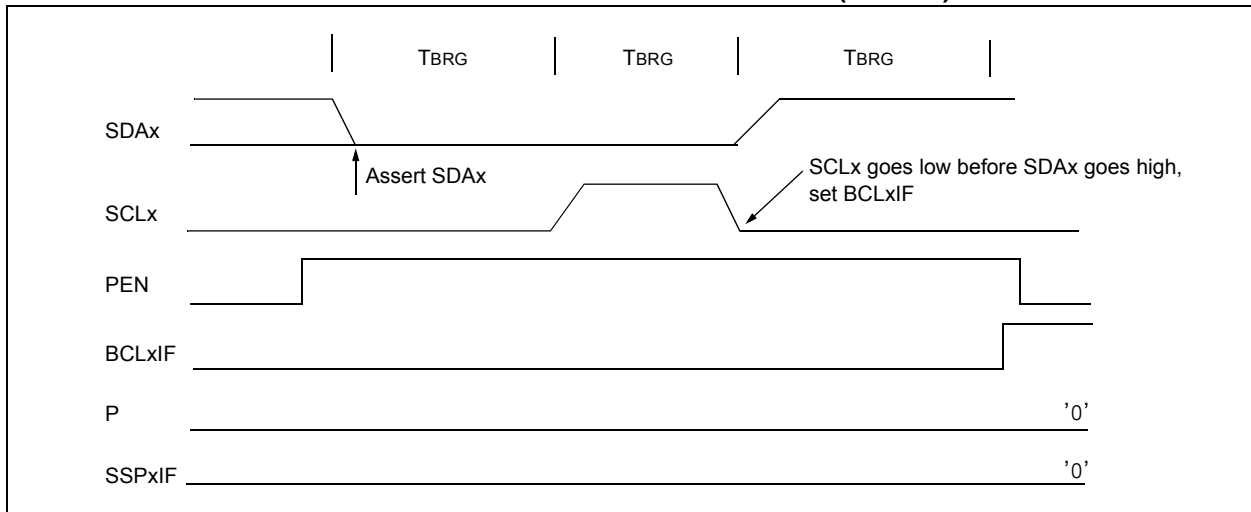


FIGURE 21-39: BUS COLLISION DURING A STOP CONDITION (CASE 2)



PIC16(L)F1526/7

TABLE 21-3: SUMMARY OF REGISTERS ASSOCIATED WITH I²C OPERATION

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Reset Values on Page: |
|----------|--|---------|-------------|--------|-----------|-------------|--------|--------|-----------------------|
| INTCON | GIE | PEIE | TMR0IE | INTE | IOCFIE | TMR0IF | INTF | IOCFIF | 76 |
| PIE1 | TMR1GIE | ADIE | RC1IE | TX1IE | SSP1IE | CCP1IE | TMR2IE | TMR1IE | 77 |
| PIE2 | OSFIE | TMR5GIE | TMR3GIE | — | BCL1IE | TMR10IE | TMR8IE | CCP2IE | 78 |
| PIE4 | CCP10IE | CCP9IE | RC2IE | TX2IE | CCP8IE | CCP7IE | BCL2IE | SSP2IE | 80 |
| PIR1 | TMR1GIF | ADIF | RC1IF | TX1IF | SSP1IF | CCP1IF | TMR2IF | TMR1IF | 81 |
| PIR2 | OSFIF | TMR5GIF | TMR3GIF | — | BCL1IF | TMR10IF | TMR8IF | CCP2IF | 82 |
| PIR4 | CCP10IF | CCP9IF | RC2IF | TX2IF | CCP8IF | CCP7IF | BCL2IF | SSP2IF | 84 |
| SSP1ADD | ADD<7:0> | | | | | | | | 247 |
| SSP2ADD | ADD<7:0> | | | | | | | | 247 |
| SSP1BUF | MSSPx Receive Buffer/Transmit Register | | | | | | | | 197* |
| SSP2BUF | MSSPx Receive Buffer/Transmit Register | | | | | | | | 197* |
| SSP1CON1 | WCOL | SSPOV | SSPEN | CKP | SSPM<3:0> | | | | 244 |
| SSP2CON1 | WCOL | SSPOV | SSPEN | CKP | SSPM<3:0> | | | | 244 |
| SSP1CON2 | GCEN | ACKSTAT | ACKDT | ACKEN | RCEN | PEN | RSEN | SEN | 245 |
| SSP2CON2 | GCEN | ACKSTAT | ACKDT | ACKEN | RCEN | PEN | RSEN | SEN | 245 |
| SSP1CON3 | ACKTIM | PCIE | SCIE | BOEN | SDAHT | SBCDE | AHEN | DHEN | 246 |
| SSP2CON3 | ACKTIM | PCIE | SCIE | BOEN | SDAHT | SBCDE | AHEN | DHEN | 246 |
| SSP1MSK | MSK<7:0> | | | | | | | | 247 |
| SSP2MSK | MSK<7:0> | | | | | | | | 247 |
| SSP1STAT | SMP | CKE | D \bar{A} | P | S | R \bar{W} | UA | BF | 242 |
| SSP2STAT | SMP | CKE | D \bar{A} | P | S | R \bar{W} | UA | BF | 242 |
| TRISC | TRISC7 | TRISC6 | TRISC5 | TRISC4 | TRISC3 | TRISC2 | TRISC1 | TRISC0 | 120 |
| TRISD | TRISD7 | TRISD6 | TRISD5 | TRISD4 | TRISD3 | TRISD2 | TRISD1 | TRISD0 | 123 |

Legend: — = unimplemented location, read as '0'. Shaded cells are not used by the MSSP module in I²C mode.

* Page provides register information.

Note 1: PIC16(L)F1527 only.

21.7 BAUD RATE GENERATOR

The MSSPx module has a Baud Rate Generator available for clock generation in both I²C and SPI Master modes. The Baud Rate Generator (BRG) reload value is placed in the SSPxADD register (Register 21-6). When a write occurs to SSPxBUF, the Baud Rate Generator will automatically begin counting down.

Once the given operation is complete, the internal clock will automatically stop counting and the clock pin will remain in its last state.

An internal signal “Reload” in Figure 21-40 triggers the value from SSPxADD to be loaded into the BRG counter. This occurs twice for each oscillation of the

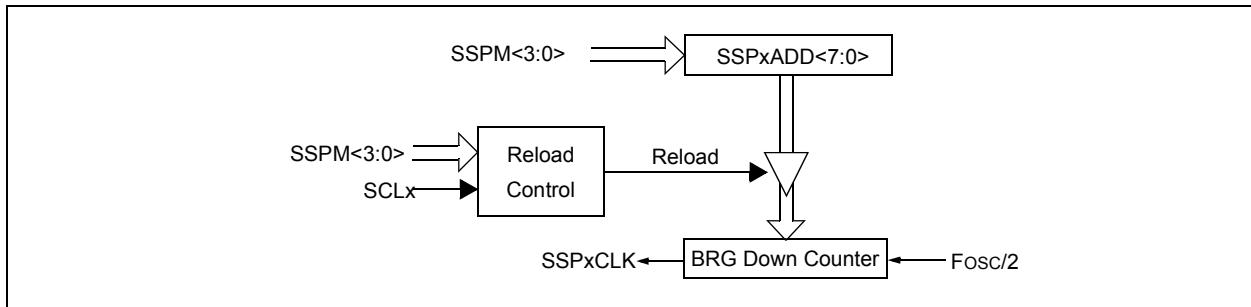
module clock line. The logic dictating when the reload signal is asserted depends on the mode the MSSPx is being operated in.

Table 21-4 demonstrates clock rates based on instruction cycles and the BRG value loaded into SSPxADD.

EQUATION 21-1:

$$F_{CLOCK} = \frac{F_{OSC}}{(SSPxADD + 1)(4)}$$

FIGURE 21-40: BAUD RATE GENERATOR BLOCK DIAGRAM



Note: Values of 0x00, 0x01 and 0x02 are not valid for SSPxADD when used as a Baud Rate Generator for I²C. This is an implementation limitation.

TABLE 21-4: MSSPX CLOCK RATE W/BRG

| Fosc | Fcy | BRG Value | FCLOCK (2 Rollovers of BRG) |
|--------|-------|-----------|--------------------------------|
| 16 MHz | 4 MHz | 09h | 400 kHz ⁽¹⁾ |
| 16 MHz | 4 MHz | 0Ch | 308 kHz |
| 16 MHz | 4 MHz | 27h | 100 kHz |
| 4 MHz | 1 MHz | 09h | 100 kHz |

Note 1: Refer to I/O port electrical and timing specifications in Table 25-3 and Figure 25-7 to ensure the system is designed to support the I/O timing requirements.

PIC16(L)F1526/7

21.8 Register Definitions: MSSP Control

REGISTER 21-1: SSPxSTAT: SSPx STATUS REGISTER

| | | | | | | | |
|---------|---------|-------------|-------|-------|-------------|-------|-------|
| R/W-0/0 | R/W-0/0 | R-0/0 | R-0/0 | R-0/0 | R-0/0 | R-0/0 | R-0/0 |
| SMP | CKE | D \bar{A} | P | S | R \bar{W} | UA | BF |
| bit 7 | | | | | | | bit 0 |

Legend:

| | | |
|----------------------|----------------------|---|
| R = Readable bit | W = Writable bit | U = Unimplemented bit, read as '0' |
| u = Bit is unchanged | x = Bit is unknown | -n/n = Value at POR and BOR/Value at all other Resets |
| '1' = Bit is set | '0' = Bit is cleared | |

- bit 7 **SMP:** SPI Data Input Sample bit
SPI Master mode:
 1 = Input data sampled at end of data output time
 0 = Input data sampled at middle of data output time
SPI Slave mode:
 SMP must be cleared when SPI is used in Slave mode
In I²C Master or Slave mode:
 1 = Slew rate control disabled for standard speed mode (100 kHz and 1 MHz)
 0 = Slew rate control enabled for high speed mode (400 kHz)
- bit 6 **CKE:** SPI Clock Edge Select bit (SPI mode only)
In SPI Master or Slave mode:
 1 = Transmit occurs on transition from active to Idle clock state
 0 = Transmit occurs on transition from Idle to active clock state
In I²C™ mode only:
 1 = Enable input logic so that thresholds are compliant with SMBus specification
 0 = Disable SMBus specific inputs
- bit 5 **D \bar{A} :** Data/Address bit (I²C mode only)
 1 = Indicates that the last byte received or transmitted was data
 0 = Indicates that the last byte received or transmitted was address
- bit 4 **P:** Stop bit
 (I²C mode only. This bit is cleared when the MSSPx module is disabled, SSPEN is cleared.)
 1 = Indicates that a Stop bit has been detected last (this bit is '0' on Reset)
 0 = Stop bit was not detected last
- bit 3 **S:** Start bit
 (I²C mode only. This bit is cleared when the MSSPx module is disabled, SSPEN is cleared.)
 1 = Indicates that a Start bit has been detected last (this bit is '0' on Reset)
 0 = Start bit was not detected last
- bit 2 **R \bar{W} :** Read/Write bit information (I²C mode only)
 This bit holds the R \bar{W} bit information following the last address match. This bit is only valid from the address match to the next Start bit, Stop bit, or not ACK bit.
In I²C Slave mode:
 1 = Read
 0 = Write
In I²C Master mode:
 1 = Transmit is in progress
 0 = Transmit is not in progress
 OR-ing this bit with SEN, RSEN, PEN, RCEN or ACKEN will indicate if the MSSPx is in Idle mode.
- bit 1 **UA:** Update Address bit (10-bit I²C mode only)
 1 = Indicates that the user needs to update the address in the SSPxADD register
 0 = Address does not need to be updated

REGISTER 21-1: SSPxSTAT: SSPx STATUS REGISTER (CONTINUED)

bit 0

BF: Buffer Full Status bit

Receive (SPI and I²C modes):

1 = Receive complete, SSPxBUF is full

0 = Receive not complete, SSPxBUF is empty

Transmit (I²C mode only):

1 = Data transmit in progress (does not include the ACK and Stop bits), SSPxBUF is full

0 = Data transmit complete (does not include the ACK and Stop bits), SSPxBUF is empty

PIC16(L)F1526/7

REGISTER 21-2: SSPxCON1: SSPx CONTROL REGISTER 1

| R/C/HS-0/0 | R/C/HS-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 |
|------------|------------|---------|---------|-----------|---------|---------|---------|
| WCOL | SSPxOV | SSPEN | CKP | SSPM<3:0> | | | |
| bit 7 | | | | | | | bit 0 |

Legend:

| | | |
|----------------------|----------------------|---|
| R = Readable bit | W = Writable bit | U = Unimplemented bit, read as '0' |
| u = Bit is unchanged | x = Bit is unknown | -n/n = Value at POR and BOR/Value at all other Resets |
| '1' = Bit is set | '0' = Bit is cleared | HS = Bit is set by hardware |
| | | C = User cleared |

| | |
|---------|---|
| bit 7 | <p>WCOL: Write Collision Detect bit</p> <p><u>Master mode:</u></p> <p>1 = A write to the SSPxBUF register was attempted while the I²C conditions were not valid for a transmission to be started</p> <p>0 = No collision</p> <p><u>Slave mode:</u></p> <p>1 = The SSPxBUF register is written while it is still transmitting the previous word (must be cleared in software)</p> <p>0 = No collision</p> |
| bit 6 | <p>SSPxOV: Receive Overflow Indicator bit⁽¹⁾</p> <p><u>In SPI mode:</u></p> <p>1 = A new byte is received while the SSPxBUF register is still holding the previous data. In case of overflow, the data in SSPxSR is lost. Overflow can only occur in Slave mode. In Slave mode, the user must read the SSPxBUF, even if only transmitting data, to avoid setting overflow. In Master mode, the overflow bit is not set since each new reception (and transmission) is initiated by writing to the SSPxBUF register (must be cleared in software).</p> <p>0 = No overflow</p> <p><u>In I²C mode:</u></p> <p>1 = A byte is received while the SSPxBUF register is still holding the previous byte. SSPxOV is a "don't care" in Transmit mode (must be cleared in software).</p> <p>0 = No overflow</p> |
| bit 5 | <p>SSPEN: Synchronous Serial Port Enable bit</p> <p>In both modes, when enabled, these pins must be properly configured as input or output</p> <p><u>In SPI mode:</u></p> <p>1 = Enables serial port and configures SCKx, SDOx, SDIx and SSx as the source of the serial port pins⁽²⁾</p> <p>0 = Disables serial port and configures these pins as I/O port pins</p> <p><u>In I²C mode:</u></p> <p>1 = Enables the serial port and configures the SDAx and SCLx pins as the source of the serial port pins⁽³⁾</p> <p>0 = Disables serial port and configures these pins as I/O port pins</p> |
| bit 4 | <p>CKP: Clock Polarity Select bit</p> <p><u>In SPI mode:</u></p> <p>1 = Idle state for clock is a high level</p> <p>0 = Idle state for clock is a low level</p> <p><u>In I²C Slave mode:</u></p> <p>SCLx release control</p> <p>1 = Enable clock</p> <p>0 = Holds clock low (clock stretch). (Used to ensure data setup time.)</p> <p><u>In I²C Master mode:</u></p> <p>Unused in this mode</p> |
| bit 3-0 | <p>SSPM<3:0>: Synchronous Serial Port Mode Select bits</p> <p>1111 = I²C Slave mode, 10-bit address with Start and Stop bit interrupts enabled</p> <p>1110 = I²C Slave mode, 7-bit address with Start and Stop bit interrupts enabled</p> <p>1101 = Reserved</p> <p>1100 = Reserved</p> <p>1011 = I²C firmware controlled Master mode (Slave idle)</p> <p>1010 = SPI Master mode, clock = Fosc/(4 * (SSPxADD+1))⁽⁵⁾</p> <p>1001 = Reserved</p> <p>1000 = I²C Master mode, clock = Fosc/(4 * (SSPxADD+1))⁽⁴⁾</p> <p>0111 = I²C Slave mode, 10-bit address</p> <p>0110 = I²C Slave mode, 7-bit address</p> <p>0101 = SPI Slave mode, clock = SCKx pin, SSx pin control disabled, SSx can be used as I/O pin</p> <p>0100 = SPI Slave mode, clock = SCKx pin, SSx pin control enabled</p> <p>0011 = SPI Master mode, clock = TMR2 output/2</p> <p>0010 = SPI Master mode, clock = Fosc/64</p> <p>0001 = SPI Master mode, clock = Fosc/16</p> <p>0000 = SPI Master mode, clock = Fosc/4</p> |

- Note**
- 1: In Master mode, the overflow bit is not set since each new reception (and transmission) is initiated by writing to the SSPxBUF register.
 - 2: When enabled, these pins must be properly configured as input or output.
 - 3: When enabled, the SDAx and SCLx pins must be configured as inputs.
 - 4: SSPxADD values of 0, 1 or 2 are not supported for I²C mode.
 - 5: SSPxADD value of '0' is not supported. Use SSPM = 0000 instead.

REGISTER 21-3: SSPxCON2: SSPx CONTROL REGISTER 2

| R/W-0/0 | R-0/0 | R/W-0/0 | R/S/HS-0/0 | R/S/HS-0/0 | R/S/HS-0/0 | R/S/HS-0/0 | R/W/HS-0/0 |
|---------|---------|---------|------------|------------|------------|------------|------------|
| GCEN | ACKSTAT | ACKDT | ACKEN | RCEN | PEN | RSEN | SEN |
| bit 7 | | | | | | | bit 0 |

Legend:

| | | |
|----------------------|----------------------|---|
| R = Readable bit | W = Writable bit | U = Unimplemented bit, read as '0' |
| u = Bit is unchanged | x = Bit is unknown | -n/n = Value at POR and BOR/Value at all other Resets |
| '1' = Bit is set | '0' = Bit is cleared | HC = Cleared by hardware S = User set |

- bit 7 **GCEN:** General Call Enable bit (in I²C Slave mode only)
 1 = Enable interrupt when a general call address (0x00 or 00h) is received in the SSPxSR
 0 = General call address disabled
- bit 6 **ACKSTAT:** Acknowledge Status bit (in I²C mode only)
 1 = Acknowledge was not received
 0 = Acknowledge was received
- bit 5 **ACKDT:** Acknowledge Data bit (in I²C mode only)
In Receive mode:
 Value transmitted when the user initiates an Acknowledge sequence at the end of a receive
 1 = Not Acknowledge
 0 = Acknowledge
- bit 4 **ACKEN:** Acknowledge Sequence Enable bit (in I²C Master mode only)
In Master Receive mode:
 1 = Initiate Acknowledge sequence on SDAx and SCLx pins, and transmit ACKDT data bit.
 Automatically cleared by hardware.
 0 = Acknowledge sequence idle
- bit 3 **RCEN:** Receive Enable bit (in I²C Master mode only)
 1 = Enables Receive mode for I²C
 0 = Receive idle
- bit 2 **PEN:** Stop Condition Enable bit (in I²C Master mode only)
SCKx Release Control:
 1 = Initiate Stop condition on SDAx and SCLx pins. Automatically cleared by hardware.
 0 = Stop condition idle
- bit 1 **RSEN:** Repeated Start Condition Enabled bit (in I²C Master mode only)
 1 = Initiate Repeated Start condition on SDAx and SCLx pins. Automatically cleared by hardware.
 0 = Repeated Start condition idle
- bit 0 **SEN:** Start Condition Enable/Stretch Enable bit
In Master mode:
 1 = Initiate Start condition on SDAx and SCLx pins. Automatically cleared by hardware.
 0 = Start condition idle
In Slave mode:
 1 = Clock stretching is enabled for both slave transmit and slave receive (stretch enabled)
 0 = Clock stretching is disabled

Note 1: For bits ACKEN, RCEN, PEN, RSEN, SEN: If the I²C module is not in the Idle mode, this bit may not be set (no spooling) and the SSPxBUF may not be written (or writes to the SSPxBUF are disabled).

PIC16(L)F1526/7

REGISTER 21-4: SSPxCON3: SSPx CONTROL REGISTER 3

| | | | | | | | |
|--------|---------|---------|---------|---------|---------|---------|---------|
| R-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 |
| ACKTIM | PCIE | SCIE | BOEN | SDAHT | SBCDE | AHEN | DHEN |
| bit 7 | | | | | | | bit 0 |

Legend:

| | | |
|----------------------|----------------------|---|
| R = Readable bit | W = Writable bit | U = Unimplemented bit, read as '0' |
| u = Bit is unchanged | x = Bit is unknown | -n/n = Value at POR and BOR/Value at all other Resets |
| '1' = Bit is set | '0' = Bit is cleared | |

- bit 7 **ACKTIM:** Acknowledge Time Status bit (I²C mode only)⁽³⁾
 1 = Indicates the I²C bus is in an Acknowledge sequence, set on 8TH falling edge of SCLx clock
 0 = Not an Acknowledge sequence, cleared on 9TH rising edge of SCLx clock
- bit 6 **PCIE:** Stop Condition Interrupt Enable bit (I²C mode only)
 1 = Enable interrupt on detection of Stop condition
 0 = Stop detection interrupts are disabled⁽²⁾
- bit 5 **SCIE:** Start Condition Interrupt Enable bit (I²C mode only)
 1 = Enable interrupt on detection of Start or Restart conditions
 0 = Start detection interrupts are disabled⁽²⁾
- bit 4 **BOEN:** Buffer Overwrite Enable bit
In SPI Slave mode:⁽¹⁾
 1 = SSPxBUF updates every time that a new data byte is shifted in ignoring the BF bit
 0 = If new byte is received with BF bit of the SSPxSTAT register already set, SSPOV bit of the SSPxCON1 register is set, and the buffer is not updated
In I²C Master mode and SPI Master mode:
 This bit is ignored.
In I²C Slave mode:
 1 = SSPxBUF is updated and \overline{ACK} is generated for a received address/data byte, ignoring the state of the SSPOV bit only if the BF bit = 0.
 0 = SSPxBUF is only updated when SSPOV is clear
- bit 3 **SDAHT:** SDAx Hold Time Selection bit (I²C mode only)
 1 = Minimum of 300 ns hold time on SDAx after the falling edge of SCLx
 0 = Minimum of 100 ns hold time on SDAx after the falling edge of SCLx
- bit 2 **SBCDE:** Slave Mode Bus Collision Detect Enable bit (I²C Slave mode only)
 If on the rising edge of SCLx, SDAx is sampled low when the module is outputting a high state, the BCLxIF bit of the PIR2 register is set, and bus goes idle
 1 = Enable slave bus collision interrupts
 0 = Slave bus collision interrupts are disabled
- bit 1 **AHEN:** Address Hold Enable bit (I²C Slave mode only)
 1 = Following the 8th falling edge of SCLx for a matching received address byte; CKP bit of the SSPxCON1 register will be cleared and the SCLx will be held low.
 0 = Address holding is disabled
- bit 0 **DHEN:** Data Hold Enable bit (I²C Slave mode only)
 1 = Following the 8th falling edge of SCLx for a received data byte; slave hardware clears the CKP bit of the SSPxCON1 register and SCLx is held low.
 0 = Data holding is disabled

- Note 1:** For daisy-chained SPI operation; allows the user to ignore all but the last received byte. SSPOV is still set when a new byte is received and BF = 1, but hardware continues to write the most recent byte to SSPxBUF.
- 2:** This bit has no effect in Slave modes that Start and Stop condition detection is explicitly listed as enabled.
- 3:** The ACKTIM Status bit is only active when the AHEN bit or DHEN bit is set.

REGISTER 21-5: SSPxMSK: SSPx MASK REGISTER

| | | | | | | | |
|----------|---------|---------|---------|---------|---------|---------|---------|
| R/W-1/1 | R/W-1/1 | R/W-1/1 | R/W-1/1 | R/W-1/1 | R/W-1/1 | R/W-1/1 | R/W-1/1 |
| MSK<7:0> | | | | | | | |
| bit 7 | | | | | | | bit 0 |

Legend:

| | | |
|----------------------|----------------------|---|
| R = Readable bit | W = Writable bit | U = Unimplemented bit, read as '0' |
| u = Bit is unchanged | x = Bit is unknown | -n/n = Value at POR and BOR/Value at all other Resets |
| '1' = Bit is set | '0' = Bit is cleared | |

- bit 7-1 **MSK<7:1>**: Mask bits
 1 = The received address bit n is compared to SSPxADD<n> to detect I²C address match
 0 = The received address bit n is not used to detect I²C address match
- bit 0 **MSK<0>**: Mask bit for I²C Slave mode, 10-bit Address
 I²C Slave mode, 10-bit address (SSPM<3:0> = 0111 or 1111):
 1 = The received address bit 0 is compared to SSPxADD<0> to detect I²C address match
 0 = The received address bit 0 is not used to detect I²C address match
 I²C Slave mode, 7-bit address, the bit is ignored

REGISTER 21-6: SSPxADD: MSSPx ADDRESS AND BAUD RATE REGISTER (I²C MODE)

| | | | | | | | |
|----------|---------|---------|---------|---------|---------|---------|---------|
| R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 |
| ADD<7:0> | | | | | | | |
| bit 7 | | | | | | | bit 0 |

Legend:

| | | |
|----------------------|----------------------|---|
| R = Readable bit | W = Writable bit | U = Unimplemented bit, read as '0' |
| u = Bit is unchanged | x = Bit is unknown | -n/n = Value at POR and BOR/Value at all other Resets |
| '1' = Bit is set | '0' = Bit is cleared | |

Master mode:

- bit 7-0 **ADD<7:0>**: Baud Rate Clock Divider bits
 SCLx pin clock period = ((ADD<7:0> + 1) * 4) / Fosc

10-Bit Slave mode — Most Significant Address byte:

- bit 7-3 **Not used:** Unused for Most Significant Address byte. Bit state of this register is a “don't care”. Bit pattern sent by master is fixed by I²C specification and must be equal to '11110'. However, those bits are compared by hardware and are not affected by the value in this register.
- bit 2-1 **ADD<2:1>**: Two Most Significant bits of 10-bit address
- bit 0 **Not used:** Unused in this mode. Bit state is a “don't care”.

10-Bit Slave mode — Least Significant Address byte:

- bit 7-0 **ADD<7:0>**: Eight Least Significant bits of 10-bit address

7-Bit Slave mode:

- bit 7-1 **ADD<7:1>**: 7-bit address
- bit 0 **Not used:** Unused in this mode. Bit state is a “don't care”.

PIC16(L)F1526/7

22.0 ENHANCED UNIVERSAL SYNCHRONOUS ASYNCHRONOUS RECEIVER TRANSMITTER (EUSART)

Note: The PIC16(L)F1526/7 devices have two EUSARTs. Therefore, all information in this section refers to both EUSART 1 and EUSART 2.

The Enhanced Universal Synchronous Asynchronous Receiver Transmitter (EUSART) module is a serial I/O communications peripheral. It contains all the clock generators, shift registers and data buffers necessary to perform an input or output serial data transfer independent of device program execution. The EUSART, also known as a Serial Communications Interface (SCI), can be configured as a full-duplex asynchronous system or half-duplex synchronous system. Full-Duplex mode is useful for communications with peripheral systems, such as CRT terminals and personal computers. Half-Duplex Synchronous mode is intended for communications with peripheral devices, such as A/D or D/A integrated circuits, serial EEPROMs or other microcontrollers.

These devices typically do not have internal clocks for baud rate generation and require the external clock signal provided by a master synchronous device.

The EUSART module includes the following capabilities:

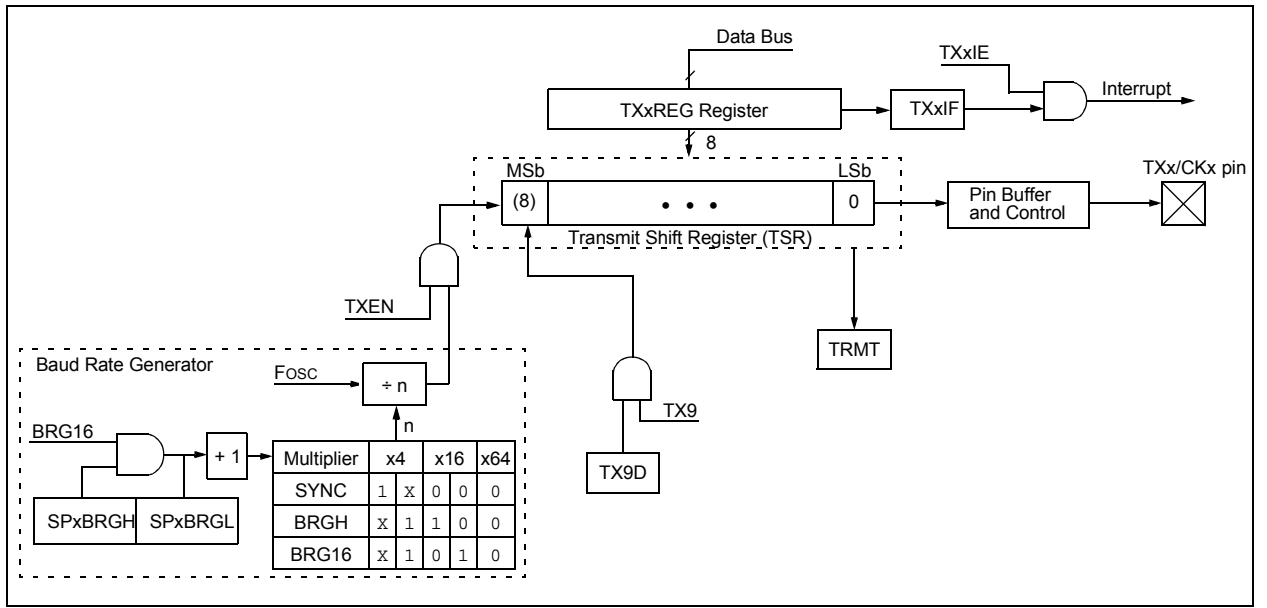
- Full-duplex asynchronous transmit and receive
- Two-character input buffer
- One-character output buffer
- Programmable 8-bit or 9-bit character length
- Address detection in 9-bit mode
- Input buffer overrun error detection
- Received character framing error detection
- Half-duplex synchronous master
- Half-duplex synchronous slave
- Programmable clock and data polarity

The EUSART module implements the following additional features, making it ideally suited for use in Local Interconnect Network (LIN) bus systems:

- Automatic detection and calibration of the baud rate
- Wake-up on Break reception
- 13-bit Break character transmit

Block diagrams of the EUSART transmitter and receiver are shown in [Figure 22-1](#) and [Figure 22-2](#).

FIGURE 22-1: EUSART TRANSMIT BLOCK DIAGRAM



PIC16(L)F1526/7

22.1 EUSART Asynchronous Mode

The EUSART transmits and receives data using the standard non-return-to-zero (NRZ) format. NRZ is implemented with two levels: a V_{OH} mark state which represents a '1' data bit, and a V_{OL} space state which represents a '0' data bit. NRZ refers to the fact that consecutively transmitted data bits of the same value stay at the output level of that bit without returning to a neutral level between each bit transmission. An NRZ transmission port idles in the mark state. Each character transmission consists of one Start bit followed by eight or nine data bits and is always terminated by one or more Stop bits. The Start bit is always a space and the Stop bits are always marks. The most common data format is 8 bits. Each transmitted bit persists for a period of 1/(Baud Rate). An on-chip dedicated 8-bit/16-bit Baud Rate Generator is used to derive standard baud rate frequencies from the system oscillator. See [Table 22-5](#) for examples of baud rate configurations.

The EUSART transmits and receives the LSB first. The EUSART's transmitter and receiver are functionally independent, but share the same data format and baud rate. Parity is not supported by the hardware, but can be implemented in software and stored as the ninth data bit.

22.1.1 EUSART ASYNCHRONOUS TRANSMITTER

The EUSART transmitter block diagram is shown in [Figure 22-1](#). The heart of the transmitter is the serial Transmit Shift Register (TSR), which is not directly accessible by software. The TSR obtains its data from the transmit buffer, which is the TXxREG register.

22.1.1.1 Enabling the Transmitter

The EUSART transmitter is enabled for asynchronous operations by configuring the following three control bits:

- TXEN = 1
- SYNC = 0
- SPEN = 1

All other EUSART control bits are assumed to be in their default state.

Setting the TXEN bit of the TXxSTA register enables the transmitter circuitry of the EUSART. Clearing the SYNC bit of the TXxSTA register configures the EUSART for asynchronous operation. Setting the SPEN bit of the RCxSTA register enables the EUSART. The programmer must set the corresponding TRIS bit to configure the TXx/CKx I/O pin as an output. If the TXx/CKx pin is shared with an analog peripheral, the analog I/O function must be disabled by clearing the corresponding ANSEL bit.

Note: The TXxIF transmitter interrupt flag is set when the TXEN enable bit is set.

22.1.1.2 Transmitting Data

A transmission is initiated by writing a character to the TXxREG register. If this is the first character, or the previous character has been completely flushed from the TSR, the data in the TXxREG is immediately transferred to the TSR register. If the TSR still contains all or part of a previous character, the new character data is held in the TXxREG until the Stop bit of the previous character has been transmitted. The pending character in the TXxREG is then transferred to the TSR in one T_{cy} immediately following the Stop bit transmission. The transmission of the Start bit, data bits and Stop bit sequence commences immediately following the transfer of the data to the TSR from the TXxREG.

22.1.1.3 Transmit Data Polarity

The polarity of the transmit data can be controlled with the SCKP bit of the BAUDxCON register. The default state of this bit is '0' which selects high true transmit idle and data bits. Setting the SCKP bit to '1' will invert the transmit data resulting in low true idle and data bits. The SCKP bit controls transmit data polarity only in Asynchronous mode. In Synchronous mode the SCKP bit has a different function.

22.1.1.4 Transmit Interrupt Flag

The TXxIF interrupt flag bit of the PIR1/PIR4 register is set whenever the EUSART transmitter is enabled and no character is being held for transmission in the TXxREG. In other words, the TXxIF bit is only clear when the TSR is busy with a character and a new character has been queued for transmission in the TXxREG. The TXxIF flag bit is not cleared immediately upon writing TXxREG. TXxIF becomes valid in the second instruction cycle following the write execution. Polling TXxIF immediately following the TXxREG write will return invalid results. The TXxIF bit is read-only, it cannot be set or cleared by software.

The TXxIF interrupt can be enabled by setting the TXxIE interrupt enable bit of the PIE1/PIE4 register. However, the TXxIF flag bit will be set whenever the TXxREG is empty, regardless of the state of TXxIE enable bit.

To use interrupts when transmitting data, set the TXxIE bit only when there is more data to send. Clear the TXxIE interrupt enable bit upon writing the last character of the transmission to the TXxREG.

22.1.1.5 TSR Status

The TRMT bit of the TXxSTA register indicates the status of the TSR register. This is a read-only bit. The TRMT bit is set when the TSR register is empty and is cleared when a character is transferred to the TSR register from the TXxREG. The TRMT bit remains clear until all bits have been shifted out of the TSR register. No interrupt logic is tied to this bit, so the user needs to poll this bit to determine the TSR status.

Note: The TSR register is not mapped in data memory, so it is not available to the user.

22.1.1.6 Transmitting 9-Bit Characters

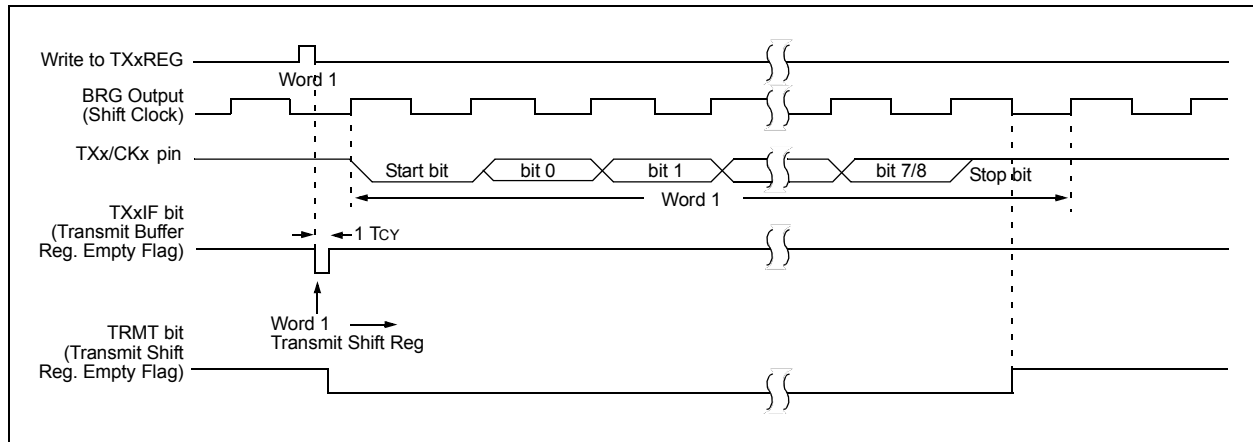
The EUSART supports 9-bit character transmissions. When the TX9 bit of the TXxSTA register is set the EUSART will shift 9 bits out for each character transmitted. The TX9D bit of the TXxSTA register is the ninth, and Most Significant, data bit. When transmitting 9-bit data, the TX9D data bit must be written before writing the 8 Least Significant bits into the TXxREG. All nine bits of data will be transferred to the TSR shift register immediately after the TXxREG is written.

A special 9-bit Address mode is available for use with multiple receivers. See [Section 22.1.2.7 “Address Detection”](#) for more information on the Address mode.

22.1.1.7 Asynchronous Transmission Set-up:

1. Initialize the SPxBRGH:SPxBRGL register pair and the BRGH and BRG16 bits to achieve the desired baud rate (see [Section 22.4 “EUSART Baud Rate Generator \(BRG\)”](#)).
2. Set the RXx/DTx and TXx/CKx TRIS controls to ‘1’.
3. Enable the asynchronous serial port by clearing the SYNC bit and setting the SPEN bit.
4. If 9-bit transmission is desired, set the TX9 control bit. A set ninth data bit will indicate that the 8 Least Significant data bits are an address when the receiver is set for address detection.
5. Set the SCKP control bit if inverted transmit data polarity is desired.
6. Enable the transmission by setting the TXEN control bit. This will cause the TXxIF interrupt bit to be set.
7. If interrupts are desired, set the TXxIE interrupt enable bit. An interrupt will occur immediately provided that the GIE and PEIE bits of the INTCON register are also set.
8. If 9-bit transmission is selected, the ninth bit should be loaded into the TX9D data bit.
9. Load 8-bit data into the TXxREG register. This will start the transmission.

FIGURE 22-3: ASYNCHRONOUS TRANSMISSION



PIC16(L)F1526/7

FIGURE 22-4: ASYNCHRONOUS TRANSMISSION (BACK-TO-BACK)

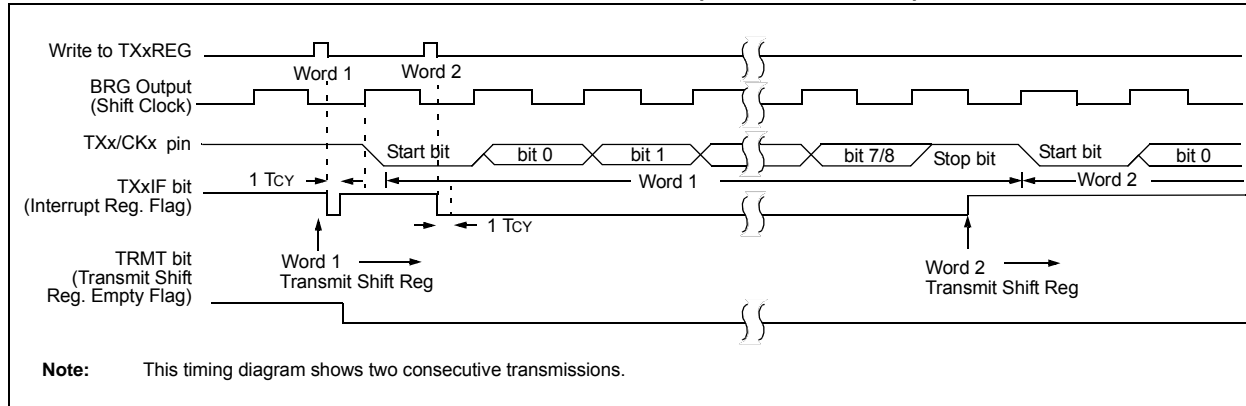


TABLE 22-1: SUMMARY OF REGISTERS ASSOCIATED WITH ASYNCHRONOUS TRANSMISSION

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Register on page |
|----------|--|--------|--------|-------|--------|--------|--------|--------|------------------|
| BAUD1CON | ABDOVF | RCIDL | — | SCKP | BRG16 | — | WUE | ABDEN | 260 |
| BAUD2CON | ABDOVF | RCIDL | — | SCKP | BRG16 | — | WUE | ABDEN | 260 |
| INTCON | GIE | PEIE | TMR0IE | INTE | IOCIE | TMR0IF | INTF | IOCIF | 76 |
| PIE1 | TMR1GIE | ADIE | RC1IE | TX1IE | SSP1IE | CCP1IE | TMR2IE | TMR1IE | 77 |
| PIE4 | CCP10IE | CCP9IE | RC2IE | TX2IE | CCP8IE | CCP7IE | BCL2IE | SSP2IE | 80 |
| PIR1 | TMR1GIF | ADIF | RC1IF | TX1IF | SSP1IF | CCP1IF | TMR2IF | TMR1IF | 81 |
| PIR4 | CCP10IF | CCP9IF | RC2IF | TX2IF | CCP8IF | CCP7IF | BCL2IF | SSP2IF | 80 |
| RC1STA | SPEN | RX9 | SREN | CREN | ADDEN | FERR | OERR | RX9D | 259 |
| RC2STA | SPEN | RX9 | SREN | CREN | ADDEN | FERR | OERR | RX9D | 259 |
| SP1BRGL | EUSART1 Baud Rate Generator, Low Byte | | | | | | | | 261* |
| SP1BRGH | EUSART1 Baud Rate Generator, High Byte | | | | | | | | 261* |
| SP2BRGL | EUSART2 Baud Rate Generator, Low Byte | | | | | | | | 261* |
| SP2BRGH | EUSART2 Baud Rate Generator, High Byte | | | | | | | | 261* |
| TX1REG | EUSART1 Transmit Register | | | | | | | | 250* |
| TX1STA | CSRC | TX9 | TXEN | SYNC | SENDB | BRGH | TRMT | TX9D | 258 |
| TX2REG | EUSART2 Transmit Register | | | | | | | | 250* |
| TX2STA | CSRC | TX9 | TXEN | SYNC | SENDB | BRGH | TRMT | TX9D | 258 |

Legend: — = unimplemented locations, read as '0'. Shaded bits are not used for asynchronous transmission.

* Page provides register information.

22.1.2 EUSART ASYNCHRONOUS RECEIVER

The Asynchronous mode would typically be used in RS-232 systems. The receiver block diagram is shown in [Figure 22-2](#). The data is received on the RXx/DTx pin and drives the data recovery block. The data recovery block is actually a high-speed shifter operating at 16 times the baud rate, whereas the serial Receive Shift Register (RSR) operates at the bit rate. When all 8 or 9 bits of the character have been shifted in, they are immediately transferred to a two character First-In-First-Out (FIFO) memory. The FIFO buffering allows reception of two complete characters and the start of a third character before software must start servicing the EUSART receiver. The FIFO and RSR registers are not directly accessible by software. Access to the received data is via the RCxREG register.

22.1.2.1 Enabling the Receiver

The EUSART receiver is enabled for asynchronous operation by configuring the following three control bits:

- CREN = 1
- SYNC = 0
- SPEN = 1

All other EUSART control bits are assumed to be in their default state.

Setting the CREN bit of the RCxSTA register enables the receiver circuitry of the EUSART. Clearing the SYNC bit of the TXxSTA register configures the EUSART for asynchronous operation. Setting the SPEN bit of the RCxSTA register enables the EUSART. The programmer must set the corresponding TRIS bit to configure the RXx/DTx I/O pin as an input.

Note 1: If the RX/DT function is on an analog pin, the corresponding ANSEL bit must be cleared for the receiver to function.

If the RXx/DTx pin is shared with an analog peripheral the analog I/O function must be disabled by clearing the corresponding ANSEL bit.

22.1.2.2 Receiving Data

The receiver data recovery circuit initiates character reception on the falling edge of the first bit. The first bit, also known as the Start bit, is always a zero. The data recovery circuit counts one-half bit time to the center of the Start bit and verifies that the bit is still a zero. If it is not a zero then the data recovery circuit aborts character reception, without generating an error, and resumes looking for the falling edge of the Start bit. If the Start bit zero verification succeeds then the data recovery circuit counts a full bit time to the center of the next bit. The bit is then sampled by a majority detect circuit and the resulting '0' or '1' is shifted into the RSR. This repeats until all data bits have been sampled and shifted into the RSR. One final bit time is measured and the level sampled. This is the Stop bit, which is always a '1'. If the data recovery circuit samples a '0' in the Stop bit position then a framing error is set for this character, otherwise the framing error is cleared for this character. See [Section 22.1.2.4 "Receive Framing Error"](#) for more information on framing errors.

Immediately after all data bits and the Stop bit have been received, the character in the RSR is transferred to the EUSART receive FIFO and the RCxIF interrupt flag bit of the PIR1/PIR4 register is set. The top character in the FIFO is transferred out of the FIFO by reading the RCxREG register.

Note: If the receive FIFO is overrun, no additional characters will be received until the overrun condition is cleared. See [Section 22.1.2.5 "Receive Overrun Error"](#) for more information on overrun errors.

PIC16(L)F1526/7

22.1.2.3 Receive Interrupts

The RCxIF interrupt flag bit of the PIR1/PIR4 register is set whenever the EUSART receiver is enabled and there is an unread character in the receive FIFO. The RCxIF interrupt flag bit is read-only, it cannot be set or cleared by software.

RCxIF interrupts are enabled by setting the following bits:

- RCxIE interrupt enable bit of the PIE1/PIE4 register
- PEIE peripheral interrupt enable bit of the INTCON register
- GIE global interrupt enable bit of the INTCON register

The RCxIF interrupt flag bit will be set when there is an unread character in the FIFO, regardless of the state of interrupt enable bits.

22.1.2.4 Receive Framing Error

Each character in the receive FIFO buffer has a corresponding framing error Status bit. A framing error indicates that a Stop bit was not seen at the expected time. The framing error status is accessed via the FERR bit of the RCxSTA register. The FERR bit represents the status of the top unread character in the receive FIFO. Therefore, the FERR bit must be read before reading the RCxREG.

The FERR bit is read-only and only applies to the top unread character in the receive FIFO. A framing error (FERR = 1) does not preclude reception of additional characters. It is not necessary to clear the FERR bit. Reading the next character from the FIFO buffer will advance the FIFO to the next character and the next corresponding framing error.

The FERR bit can be forced clear by clearing the SPEN bit of the RCxSTA register which resets the EUSART. Clearing the CREN bit of the RCxSTA register does not affect the FERR bit. A framing error by itself does not generate an interrupt.

| |
|--|
| <p>Note: If all receive characters in the receive FIFO have framing errors, repeated reads of the RCxREG will not clear the FERR bit.</p> |
|--|

22.1.2.5 Receive Overrun Error

The receive FIFO buffer can hold two characters. An overrun error will be generated if a third character, in its entirety, is received before the FIFO is accessed. When this happens the OERR bit of the RCxSTA register is set. The characters already in the FIFO buffer can be read but no additional characters will be received until the error is cleared. The error must be cleared by either clearing the CREN bit of the RCxSTA register or by resetting the EUSART by clearing the SPEN bit of the RCxSTA register.

22.1.2.6 Receiving 9-bit Characters

The EUSART supports 9-bit character reception. When the RX9 bit of the RCxSTA register is set, the EUSART will shift 9 bits into the RSR for each character received. The RX9D bit of the RCxSTA register is the ninth and Most Significant data bit of the top unread character in the receive FIFO. When reading 9-bit data from the receive FIFO buffer, the RX9D data bit must be read before reading the 8 Least Significant bits from the RCxREG.

22.1.2.7 Address Detection

A special Address Detection mode is available for use when multiple receivers share the same transmission line, such as in RS-485 systems. Address detection is enabled by setting the ADDEN bit of the RCxSTA register.

Address detection requires 9-bit character reception. When address detection is enabled, only characters with the ninth data bit set will be transferred to the receive FIFO buffer, thereby setting the RCxIF interrupt bit. All other characters will be ignored.

Upon receiving an address character, user software determines if the address matches its own. Upon address match, user software must disable address detection by clearing the ADDEN bit before the next Stop bit occurs. When user software detects the end of the message, determined by the message protocol used, software places the receiver back into the Address Detection mode by setting the ADDEN bit.

22.1.2.8 Asynchronous Reception Set-up:

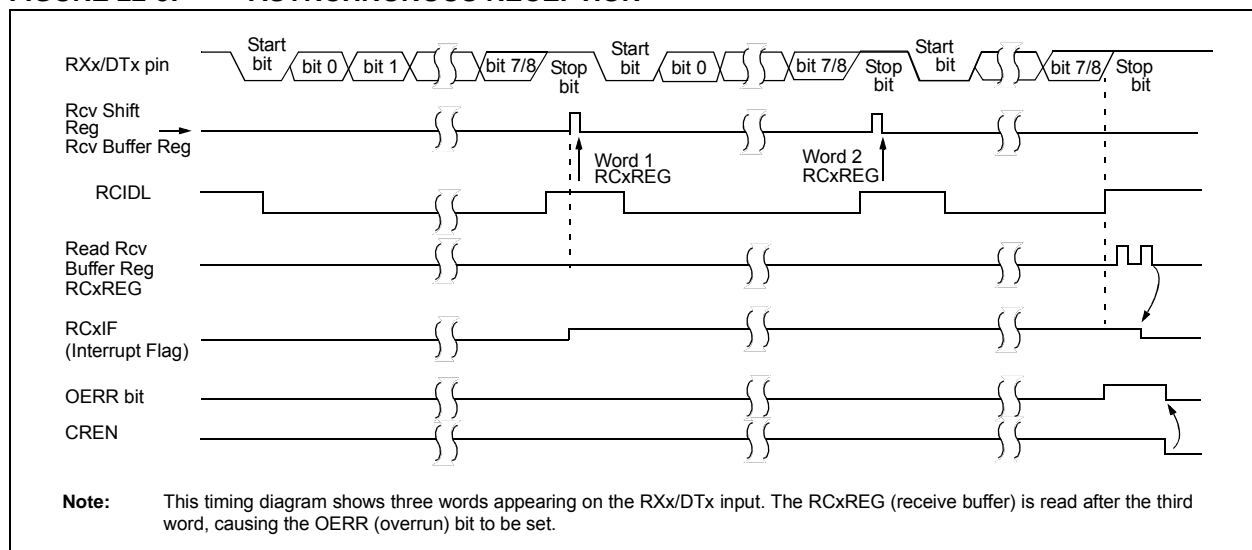
1. Initialize the SPxBRGH:SPxBRGL register pair and the BRGH and BRG16 bits to achieve the desired baud rate (see [Section 22.4 “EUSART Baud Rate Generator \(BRG\)”](#)).
2. Set the RXx/DTx and TXx/CKx TRIS controls to '1'.
3. Enable the serial port by setting the SPEN bit and the RXx/DTx pin TRIS bit. The SYNC bit must be clear for asynchronous operation.
4. If interrupts are desired, set the RCxIE interrupt enable bit and set the GIE and PEIE bits of the INTCON register.
5. If 9-bit reception is desired, set the RX9 bit.
6. Enable reception by setting the CREN bit.
7. The RCxIF interrupt flag bit will be set when a character is transferred from the RSR to the receive buffer. An interrupt will be generated if the RCxIE interrupt enable bit was also set.
8. Read the RCxSTA register to get the error flags and, if 9-bit data reception is enabled, the ninth data bit.
9. Get the received 8 Least Significant data bits from the receive buffer by reading the RCxREG register.
10. If an overrun occurred, clear the OERR flag by clearing the CREN receiver enable bit.

22.1.2.9 9-bit Address Detection Mode Set-up

This mode would typically be used in RS-485 systems. To set up an Asynchronous Reception with Address Detect Enable:

1. Initialize the SPxBRGH, SPxBRGL register pair and the BRGH and BRG16 bits to achieve the desired baud rate (see [Section 22.4 “EUSART Baud Rate Generator \(BRG\)”](#)).
2. Set the RXx/DTx and TXx/CKx TRIS controls to '1'.
3. Enable the serial port by setting the SPEN bit. The SYNC bit must be clear for asynchronous operation.
4. If interrupts are desired, set the RCxIE interrupt enable bit and set the GIE and PEIE bits of the INTCON register.
5. Enable 9-bit reception by setting the RX9 bit.
6. Enable address detection by setting the ADDEN bit.
7. Enable reception by setting the CREN bit.
8. The RCxIF interrupt flag bit will be set when a character with the ninth bit set is transferred from the RSR to the receive buffer. An interrupt will be generated if the RCxIE interrupt enable bit was also set.
9. Read the RCxSTA register to get the error flags. The ninth data bit will always be set.
10. Get the received 8 Least Significant data bits from the receive buffer by reading the RCxREG register. Software determines if this is the device's address.
11. If an overrun occurred, clear the OERR flag by clearing the CREN receiver enable bit.
12. If the device has been addressed, clear the ADDEN bit to allow all received data into the receive buffer and generate interrupts.

FIGURE 22-5: ASYNCHRONOUS RECEPTION



PIC16(L)F1526/7

TABLE 22-2: SUMMARY OF REGISTERS ASSOCIATED WITH ASYNCHRONOUS RECEPTION

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Register on Page |
|----------|--|--------|--------|--------|--------|--------|--------|--------|------------------|
| BAUD1CON | ABDOVF | RCIDL | — | SCKP | BRG16 | — | WUE | ABDEN | 260 |
| BAUD2CON | ABDOVF | RCIDL | — | SCKP | BRG16 | — | WUE | ABDEN | 260 |
| INTCON | GIE | PEIE | TMR0IE | INTE | IOCIE | TMR0IF | INTF | IOCIF | 76 |
| PIE1 | TMR1GIE | ADIE | RC1IE | TX1IE | SSP1IE | CCP1IE | TMR2IE | TMR1IE | 77 |
| PIE4 | CCP10IE | CCP9IE | RC2IE | TX2IE | CCP8IE | CCP7IE | BCL2IE | SSP2IE | 80 |
| PIR1 | TMR1GIF | ADIF | RC1IF | TX1IF | SSP1IF | CCP1IF | TMR2IF | TMR1IF | 81 |
| PIR4 | CCP10IF | CCP9IF | RC2IF | TX2IF | CCP8IF | CCP7IF | BCL2IF | SSP2IF | 84 |
| RC1REG | EUSART1 Receive Register | | | | | | | | 253* |
| RC1STA | SPEN | RX9 | SREN | CREN | ADDEN | FERR | OERR | RX9D | 259 |
| RC2REG | EUSART2 Receive Register | | | | | | | | 253* |
| RC2STA | SPEN | RX9 | SREN | CREN | ADDEN | FERR | OERR | RX9D | 259 |
| SP1BRGL | EUSART1 Baud Rate Generator, Low Byte | | | | | | | | 261* |
| SP1BRGH | EUSART1 Baud Rate Generator, High Byte | | | | | | | | 261* |
| SP2BRGL | EUSART2 Baud Rate Generator, Low Byte | | | | | | | | 261* |
| SP2BRGH | EUSART2 Baud Rate Generator, High Byte | | | | | | | | 261* |
| TRISC | TRISC7 | TRISC6 | TRISC5 | TRISC4 | TRISC3 | TRISC2 | TRISC1 | TRISC0 | 120 |
| TX1STA | CSRC | TX9 | TXEN | SYNC | SENDB | BRGH | TRMT | TX9D | 258 |
| TX2STA | CSRC | TX9 | TXEN | SYNC | SENDB | BRGH | TRMT | TX9D | 258 |

Legend: — = unimplemented locations, read as '0'. Shaded bits are not used for asynchronous reception.

* Page provides register information.

22.2 Clock Accuracy with Asynchronous Operation

The factory calibrates the internal oscillator block output (HFINTOSC). However, the HFINTOSC frequency may drift as VDD or temperature changes, and this directly affects the asynchronous baud rate.

The Auto-Baud Detect feature (refer to section [Section 22.4.1, Auto-Baud Detect](#)) can be used to compensate for changes in the INTOSC frequency.

There may not be a fine enough resolution when adjusting the Baud Rate Generator to compensate for a gradual change in the peripheral clock frequency.

The first (preferred) method uses the OSCTUNE register to adjust the HFINTOSC output. Adjusting the value in the OSCTUNE register allows for fine resolution changes to the system clock source. See [Section 5.2 “Clock Source Types”](#) for more information.

The other method adjusts the value in the Baud Rate Generator. This can be done automatically with the Auto-Baud Detect feature (see [Section 22.4.1 “Auto-Baud Detect”](#)). There may not be fine enough resolution when adjusting the Baud Rate Generator to compensate for a gradual change in the peripheral clock frequency.

PIC16(L)F1526/7

22.3 Register Definitions: EUSART Control

REGISTER 22-1: TXxSTA: TRANSMIT STATUS AND CONTROL REGISTER

| R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R-1 | R/W-0 |
|-------|-------|---------------------|-------|-------|-------|------|-------|
| CSRC | TX9 | TXEN ⁽¹⁾ | SYNC | SENDB | BRGH | TRMT | TX9D |
| bit 7 | | | | | | | bit 0 |

Legend:

R = Readable bit

W = Writable bit

U = Unimplemented bit, read as '0'

-n = Value at POR

'1' = Bit is set

'0' = Bit is cleared

x = Bit is unknown

- bit 7 **CSRC:** Clock Source Select bit
Asynchronous mode:
Don't care
Synchronous mode:
1 = Master mode (clock generated internally from BRG)
0 = Slave mode (clock from external source)
- bit 6 **TX9:** 9-bit Transmit Enable bit
1 = Selects 9-bit transmission
0 = Selects 8-bit transmission
- bit 5 **TXEN:** Transmit Enable bit⁽¹⁾
1 = Transmit enabled
0 = Transmit disabled
- bit 4 **SYNC:** EUSART Mode Select bit
1 = Synchronous mode
0 = Asynchronous mode
- bit 3 **SENDB:** Send Break Character bit
Asynchronous mode:
1 = Send Sync Break on next transmission (cleared by hardware upon completion)
0 = Sync Break transmission completed
Synchronous mode:
Don't care
- bit 2 **BRGH:** High Baud Rate Select bit
Asynchronous mode:
1 = High speed
0 = Low speed
Synchronous mode:
Unused in this mode
- bit 1 **TRMT:** Transmit Shift Register Status bit
1 = TSR empty
0 = TSR full
- bit 0 **TX9D:** Ninth bit of Transmit Data
Can be address/data bit or a parity bit.

Note 1: SREN/CREN overrides TXEN in Sync mode.

REGISTER 22-2: RCxSTA: RECEIVE STATUS AND CONTROL REGISTER

| R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R-0 | R-0 | R-x |
|-------|-------|-------|-------|-------|------|------|-------|
| SPEN | RX9 | SREN | CREN | ADDEN | FERR | OERR | RX9D |
| bit 7 | | | | | | | bit 0 |

Legend:

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

- bit 7 **SPEN:** Serial Port Enable bit
 1 = Serial port enabled (configures RXx/DTx and TXx/CKx pins as serial port pins)
 0 = Serial port disabled (held in Reset)
- bit 6 **RX9:** 9-bit Receive Enable bit
 1 = Selects 9-bit reception
 0 = Selects 8-bit reception
- bit 5 **SREN:** Single Receive Enable bit
Asynchronous mode:
 Don't care
Synchronous mode – Master:
 1 = Enables single receive
 0 = Disables single receive
 This bit is cleared after reception is complete.
Synchronous mode – Slave
 Don't care
- bit 4 **CREN:** Continuous Receive Enable bit
Asynchronous mode:
 1 = Enables receiver
 0 = Disables receiver
Synchronous mode:
 1 = Enables continuous receive until enable bit CREN is cleared (CREN overrides SREN)
 0 = Disables continuous receive
- bit 3 **ADDEN:** Address Detect Enable bit
Asynchronous mode 9-bit (RX9 = 1):
 1 = Enables address detection, enable interrupt and load the receive buffer when RSR<8> is set
 0 = Disables address detection, all bytes are received and ninth bit can be used as parity bit
Asynchronous mode 8-bit (RX9 = 0):
 Don't care
- bit 2 **FERR:** Framing Error bit
 1 = Framing error (can be updated by reading RCxREG register and receive next valid byte)
 0 = No framing error
- bit 1 **OERR:** Overrun Error bit
 1 = Overrun error (can be cleared by clearing bit CREN)
 0 = No overrun error
- bit 0 **RX9D:** Ninth bit of Received Data
 This can be address/data bit or a parity bit and must be calculated by user firmware.

PIC16(L)F1526/7

REGISTER 22-3: BAUDxCON: BAUD RATE CONTROL REGISTER

| R-0/0 | R-1/1 | U-0 | R/W-0/0 | R/W-0/0 | U-0 | R/W-0/0 | R/W-0/0 |
|--------|-------|-----|---------|---------|-----|---------|---------|
| ABDOVF | RCIDL | — | SCKP | BRG16 | — | WUE | ABDEN |
| bit 7 | | | | | | | bit 0 |

Legend:

| | | |
|----------------------|----------------------|---|
| R = Readable bit | W = Writable bit | U = Unimplemented bit, read as '0' |
| u = Bit is unchanged | x = Bit is unknown | -n/n = Value at POR and BOR/Value at all other Resets |
| '1' = Bit is set | '0' = Bit is cleared | |

- bit 7 **ABDOVF:** Auto-Baud Detect Overflow bit
Asynchronous mode:
 1 = Auto-baud timer overflowed
 0 = Auto-baud timer did not overflow
Synchronous mode:
 Don't care
- bit 6 **RCIDL:** Receive Idle Flag bit
Asynchronous mode:
 1 = Receiver is Idle
 0 = Start bit has been received and the receiver is receiving
Synchronous mode:
 Don't care
- bit 5 **Unimplemented:** Read as '0'
- bit 4 **SCKP:** Synchronous Clock Polarity Select bit
Asynchronous mode:
 1 = Transmit inverted data to the TXx/CKx pin
 0 = Transmit non-inverted data to the TXx/CKx pin
Synchronous mode:
 1 = Data is clocked on rising edge of the clock
 0 = Data is clocked on falling edge of the clock
- bit 3 **BRG16:** 16-bit Baud Rate Generator bit
 1 = 16-bit Baud Rate Generator is used
 0 = 8-bit Baud Rate Generator is used
- bit 2 **Unimplemented:** Read as '0'
- bit 1 **WUE:** Wake-up Enable bit
Asynchronous mode:
 1 = Receiver is waiting for a falling edge. No character will be received, byte RCIF will be set. WUE will automatically clear after RCIF is set.
 0 = Receiver is operating normally
Synchronous mode:
 Don't care
- bit 0 **ABDEN:** Auto-Baud Detect Enable bit
Asynchronous mode:
 1 = Auto-Baud Detect mode is enabled (clears when auto-baud is complete)
 0 = Auto-Baud Detect mode is disabled
Synchronous mode:
 Don't care

22.4 EUSART Baud Rate Generator (BRG)

The Baud Rate Generator (BRG) is an 8-bit or 16-bit timer that is dedicated to the support of both the asynchronous and synchronous EUSART operation. By default, the BRG operates in 8-bit mode. Setting the BRG16 bit of the BAUDxCON register selects 16-bit mode.

The SPxBRGH:SPxBRGL register pair determines the period of the free running baud rate timer. In Asynchronous mode the multiplier of the baud rate period is determined by both the BRGH bit of the TXxSTA register and the BRG16 bit of the BAUDxCON register. In Synchronous mode, the BRGH bit is ignored.

Example 22-1 provides a sample calculation for determining the desired baud rate, actual baud rate, and baud rate % error.

Typical baud rates and error values for various asynchronous modes have been computed for your convenience and are shown in [Table 22-5](#). It may be advantageous to use the high baud rate (BRGH = 1), or the 16-bit BRG (BRG16 = 1) to reduce the baud rate error. The 16-bit BRG mode is used to achieve slow baud rates for fast oscillator frequencies.

Writing a new value to the SPxBRGH, SPxBRGL register pair causes the BRG timer to be reset (or cleared). This ensures that the BRG does not wait for a timer overflow before outputting the new baud rate.

If the system clock is changed during an active receive operation, a receive error or data loss may result. To avoid this problem, check the status of the RCIDL bit to make sure that the receive operation is Idle before changing the system clock.

EXAMPLE 22-1: CALCULATING BAUD RATE ERROR

For a device with Fosc of 16 MHz, desired baud rate of 9600, Asynchronous mode, 8-bit BRG:

$$\text{Desired Baud Rate} = \frac{F_{OSC}}{64([SPxBRGH:SPxBRG] + 1)}$$

Solving for SPxBRGH:SPxBRGL:

$$SPxBRGH:SPxBRGL = \frac{F_{OSC}}{\text{Desired Baud Rate}} - 1$$

$$= \frac{16000000}{9600} - 1$$

$$= [25.042] = 25$$

$$\text{ActualBaudRate} = \frac{16000000}{64(25 + 1)}$$

$$= 9615$$

$$\text{Baud Rate \% Error} = \frac{\text{Calc. Baud Rate} - \text{Desired Baud Rate}}{\text{Desired Baud Rate}}$$

$$= \frac{(9615 - 9600)}{9600} = 0.16\%$$

TABLE 22-3: BAUD RATE FORMULAS

| Configuration Bits | | | BRG/EUSART Mode | Baud Rate Formula |
|--------------------|-------|------|---------------------|-------------------|
| SYNC | BRG16 | BRGH | | |
| 0 | 0 | 0 | 8-bit/Asynchronous | Fosc/[64 (n+1)] |
| 0 | 0 | 1 | 8-bit/Asynchronous | Fosc/[16 (n+1)] |
| 0 | 1 | 0 | 16-bit/Asynchronous | |
| 0 | 1 | 1 | 16-bit/Asynchronous | Fosc/[4 (n+1)] |
| 1 | 0 | x | 8-bit/Synchronous | |
| 1 | 1 | x | 16-bit/Synchronous | |

Legend: x = Don't care, n = value of SPxBRGH, SPxBRGL register pair

PIC16(L)F1526/7

TABLE 22-4: SUMMARY OF REGISTERS ASSOCIATED WITH THE BAUD RATE GENERATOR

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Reset Values on page |
|----------|--|-------|-------|-------|-------|-------|-------|-------|----------------------|
| BAUD1CON | ABDOVF | RCIDL | — | SCKP | BRG16 | — | WUE | ABDEN | 260 |
| BAUD2CON | ABDOVF | RCIDL | — | SCKP | BRG16 | — | WUE | ABDEN | 260 |
| RC1STA | SPEN | RX9 | SREN | CREN | ADDEN | FERR | OERR | RX9D | 259 |
| RC2STA | SPEN | RX9 | SREN | CREN | ADDEN | FERR | OERR | RX9D | 259 |
| SP1BRGL | EUSART1 Baud Rate Generator, Low Byte | | | | | | | | 261* |
| SP1BRGH | EUSART1 Baud Rate Generator, High Byte | | | | | | | | 261* |
| SP2BRGL | EUSART2 Baud Rate Generator, Low Byte | | | | | | | | 261* |
| SP2BRGH | EUSART2 Baud Rate Generator, High Byte | | | | | | | | 261* |
| TX1STA | CSRC | TX9 | TXEN | SYNC | SENCB | BRGH | TRMT | TX9D | 258 |
| TX2STA | CSRC | TX9 | TXEN | SYNC | SENCB | BRGH | TRMT | TX9D | 258 |

Legend: — = unimplemented, read as '0'. Shaded bits are not used by the BRG.

* Page provides register information.

TABLE 22-5: BAUD RATES FOR ASYNCHRONOUS MODES

| BAUD RATE | SYNC = 0, BRGH = 0, BRG16 = 0 | | | | | | | | | | | |
|-----------|-------------------------------|---------|-----------------------|-------------------|---------|-----------------------|-------------------|---------|-----------------------|--------------------|---------|-----------------------|
| | Fosc = 20.000 MHz | | | Fosc = 18.432 MHz | | | Fosc = 16.000 MHz | | | Fosc = 11.0592 MHz | | |
| | Actual Rate | % Error | SPBRG value (decimal) | Actual Rate | % Error | SPBRG value (decimal) | Actual Rate | % Error | SPBRG value (decimal) | Actual Rate | % Error | SPBRG value (decimal) |
| 300 | — | — | — | — | — | — | — | — | — | — | — | — |
| 1200 | 1221 | 1.73 | 255 | 1200 | 0.00 | 239 | 1202 | 0.16 | 207 | 1200 | 0.00 | 143 |
| 2400 | 2404 | 0.16 | 129 | 2400 | 0.00 | 119 | 2404 | 0.16 | 103 | 2400 | 0.00 | 71 |
| 9600 | 9470 | -1.36 | 32 | 9600 | 0.00 | 29 | 9615 | 0.16 | 25 | 9600 | 0.00 | 17 |
| 10417 | 10417 | 0.00 | 29 | 10286 | -1.26 | 27 | 10417 | 0.00 | 23 | 10165 | -2.42 | 16 |
| 19.2k | 19.53k | 1.73 | 15 | 19.20k | 0.00 | 14 | 19.23k | 0.16 | 12 | 19.20k | 0.00 | 8 |
| 57.6k | — | — | — | 57.60k | 0.00 | 7 | — | — | — | 57.60k | 0.00 | 2 |
| 115.2k | — | — | — | — | — | — | — | — | — | — | — | — |

| BAUD RATE | SYNC = 0, BRGH = 0, BRG16 = 0 | | | | | | | | | | | |
|-----------|-------------------------------|---------|-----------------------|------------------|---------|-----------------------|-------------------|---------|-----------------------|------------------|---------|-----------------------|
| | Fosc = 8.000 MHz | | | Fosc = 4.000 MHz | | | Fosc = 3.6864 MHz | | | Fosc = 1.000 MHz | | |
| | Actual Rate | % Error | SPBRG value (decimal) | Actual Rate | % Error | SPBRG value (decimal) | Actual Rate | % Error | SPBRG value (decimal) | Actual Rate | % Error | SPBRG value (decimal) |
| 300 | — | — | — | 300 | 0.16 | 207 | 300 | 0.00 | 191 | 300 | 0.16 | 51 |
| 1200 | 1202 | 0.16 | 103 | 1202 | 0.16 | 51 | 1200 | 0.00 | 47 | 1202 | 0.16 | 12 |
| 2400 | 2404 | 0.16 | 51 | 2404 | 0.16 | 25 | 2400 | 0.00 | 23 | — | — | — |
| 9600 | 9615 | 0.16 | 12 | — | — | — | 9600 | 0.00 | 5 | — | — | — |
| 10417 | 10417 | 0.00 | 11 | 10417 | 0.00 | 5 | — | — | — | — | — | — |
| 19.2k | — | — | — | — | — | — | 19.20k | 0.00 | 2 | — | — | — |
| 57.6k | — | — | — | — | — | — | 57.60k | 0.00 | 0 | — | — | — |
| 115.2k | — | — | — | — | — | — | — | — | — | — | — | — |

| BAUD RATE | SYNC = 0, BRGH = 1, BRG16 = 0 | | | | | | | | | | | |
|-----------|-------------------------------|---------|-----------------------|-------------------|---------|-----------------------|-------------------|---------|-----------------------|--------------------|---------|-----------------------|
| | Fosc = 20.000 MHz | | | Fosc = 18.432 MHz | | | Fosc = 16.000 MHz | | | Fosc = 11.0592 MHz | | |
| | Actual Rate | % Error | SPBRG value (decimal) | Actual Rate | % Error | SPBRG value (decimal) | Actual Rate | % Error | SPBRG value (decimal) | Actual Rate | % Error | SPBRG value (decimal) |
| 300 | — | — | — | — | — | — | — | — | — | — | — | — |
| 1200 | — | — | — | — | — | — | — | — | — | — | — | — |
| 2400 | — | — | — | — | — | — | — | — | — | — | — | — |
| 9600 | 9615 | 0.16 | 129 | 9600 | 0.00 | 119 | 9615 | 0.16 | 103 | 9600 | 0.00 | 71 |
| 10417 | 10417 | 0.00 | 119 | 10378 | -0.37 | 110 | 10417 | 0.00 | 95 | 10473 | 0.53 | 65 |
| 19.2k | 19.23k | 0.16 | 64 | 19.20k | 0.00 | 59 | 19.23k | 0.16 | 51 | 19.20k | 0.00 | 35 |
| 57.6k | 56.82k | -1.36 | 21 | 57.60k | 0.00 | 19 | 58.82k | 2.12 | 16 | 57.60k | 0.00 | 11 |
| 115.2k | 113.64k | -1.36 | 10 | 115.2k | 0.00 | 9 | 111.1k | -3.55 | 8 | 115.2k | 0.00 | 5 |

PIC16(L)F1526/7

TABLE 22-5: BAUD RATES FOR ASYNCHRONOUS MODES (CONTINUED)

| BAUD RATE | SYNC = 0, BRGH = 1, BRG16 = 0 | | | | | | | | | | | |
|-----------|-------------------------------|---------|-----------------------|------------------|---------|-----------------------|-------------------|---------|-----------------------|------------------|---------|-----------------------|
| | Fosc = 8.000 MHz | | | Fosc = 4.000 MHz | | | Fosc = 3.6864 MHz | | | Fosc = 1.000 MHz | | |
| | Actual Rate | % Error | SPBRG value (decimal) | Actual Rate | % Error | SPBRG value (decimal) | Actual Rate | % Error | SPBRG value (decimal) | Actual Rate | % Error | SPBRG value (decimal) |
| 300 | — | — | — | — | — | — | — | — | — | 300 | 0.16 | 207 |
| 1200 | — | — | — | 1202 | 0.16 | 207 | 1200 | 0.00 | 191 | 1202 | 0.16 | 51 |
| 2400 | 2404 | 0.16 | 207 | 2404 | 0.16 | 103 | 2400 | 0.00 | 95 | 2404 | 0.16 | 25 |
| 9600 | 9615 | 0.16 | 51 | 9615 | 0.16 | 25 | 9600 | 0.00 | 23 | — | — | — |
| 10417 | 10417 | 0.00 | 47 | 10417 | 0.00 | 23 | 10473 | 0.53 | 21 | 10417 | 0.00 | 5 |
| 19.2k | 19231 | 0.16 | 25 | 19.23k | 0.16 | 12 | 19.2k | 0.00 | 11 | — | — | — |
| 57.6k | 55556 | -3.55 | 8 | — | — | — | 57.60k | 0.00 | 3 | — | — | — |
| 115.2k | — | — | — | — | — | — | 115.2k | 0.00 | 1 | — | — | — |

| BAUD RATE | SYNC = 0, BRGH = 0, BRG16 = 1 | | | | | | | | | | | |
|-----------|-------------------------------|---------|-----------------------|-------------------|---------|-----------------------|-------------------|---------|-----------------------|--------------------|---------|-----------------------|
| | Fosc = 20.000 MHz | | | Fosc = 18.432 MHz | | | Fosc = 16.000 MHz | | | Fosc = 11.0592 MHz | | |
| | Actual Rate | % Error | SPBRG value (decimal) | Actual Rate | % Error | SPBRG value (decimal) | Actual Rate | % Error | SPBRG value (decimal) | Actual Rate | % Error | SPBRG value (decimal) |
| 300 | 300.0 | -0.01 | 4166 | 300.0 | 0.00 | 3839 | 300.03 | 0.01 | 3332 | 300.0 | 0.00 | 2303 |
| 1200 | 1200 | -0.03 | 1041 | 1200 | 0.00 | 959 | 1200.5 | 0.04 | 832 | 1200 | 0.00 | 575 |
| 2400 | 2399 | -0.03 | 520 | 2400 | 0.00 | 479 | 2398 | -0.08 | 416 | 2400 | 0.00 | 287 |
| 9600 | 9615 | 0.16 | 129 | 9600 | 0.00 | 119 | 9615 | 0.16 | 103 | 9600 | 0.00 | 71 |
| 10417 | 10417 | 0.00 | 119 | 10378 | -0.37 | 110 | 10417 | 0.00 | 95 | 10473 | 0.53 | 65 |
| 19.2k | 19.23k | 0.16 | 64 | 19.20k | 0.00 | 59 | 19.23k | 0.16 | 51 | 19.20k | 0.00 | 35 |
| 57.6k | 56.818 | -1.36 | 21 | 57.60k | 0.00 | 19 | 58.82k | 2.12 | 16 | 57.60k | 0.00 | 11 |
| 115.2k | 113.636 | -1.36 | 10 | 115.2k | 0.00 | 9 | 111.11k | -3.55 | 8 | 115.2k | 0.00 | 5 |

| BAUD RATE | SYNC = 0, BRGH = 0, BRG16 = 1 | | | | | | | | | | | |
|-----------|-------------------------------|---------|-----------------------|------------------|---------|-----------------------|-------------------|---------|-----------------------|------------------|---------|-----------------------|
| | Fosc = 8.000 MHz | | | Fosc = 4.000 MHz | | | Fosc = 3.6864 MHz | | | Fosc = 1.000 MHz | | |
| | Actual Rate | % Error | SPBRG value (decimal) | Actual Rate | % Error | SPBRG value (decimal) | Actual Rate | % Error | SPBRG value (decimal) | Actual Rate | % Error | SPBRG value (decimal) |
| 300 | 299.9 | -0.02 | 1666 | 300.1 | 0.04 | 832 | 300.0 | 0.00 | 767 | 300.5 | 0.16 | 207 |
| 1200 | 1199 | -0.08 | 416 | 1202 | 0.16 | 207 | 1200 | 0.00 | 191 | 1202 | 0.16 | 51 |
| 2400 | 2404 | 0.16 | 207 | 2404 | 0.16 | 103 | 2400 | 0.00 | 95 | 2404 | 0.16 | 25 |
| 9600 | 9615 | 0.16 | 51 | 9615 | 0.16 | 25 | 9600 | 0.00 | 23 | — | — | — |
| 10417 | 10417 | 0.00 | 47 | 10417 | 0.00 | 23 | 10473 | 0.53 | 21 | 10417 | 0.00 | 5 |
| 19.2k | 19.23k | 0.16 | 25 | 19.23k | 0.16 | 12 | 19.20k | 0.00 | 11 | — | — | — |
| 57.6k | 55556 | -3.55 | 8 | — | — | — | 57.60k | 0.00 | 3 | — | — | — |
| 115.2k | — | — | — | — | — | — | 115.2k | 0.00 | 1 | — | — | — |

TABLE 22-5: BAUD RATES FOR ASYNCHRONOUS MODES (CONTINUED)

| BAUD RATE | SYNC = 0, BRGH = 1, BRG16 = 1 or SYNC = 1, BRG16 = 1 | | | | | | | | | | | |
|-----------|--|---------|-----------------------|-------------------|---------|-----------------------|-------------------|---------|-----------------------|--------------------|---------|-----------------------|
| | Fosc = 20.000 MHz | | | Fosc = 18.432 MHz | | | Fosc = 16.000 MHz | | | Fosc = 11.0592 MHz | | |
| | Actual Rate | % Error | SPBRG value (decimal) | Actual Rate | % Error | SPBRG value (decimal) | Actual Rate | % Error | SPBRG value (decimal) | Actual Rate | % Error | SPBRG value (decimal) |
| 300 | 300.0 | 0.00 | 16665 | 300.0 | 0.00 | 15359 | 300.0 | 0.00 | 13332 | 300.0 | 0.00 | 9215 |
| 1200 | 1200 | -0.01 | 4166 | 1200 | 0.00 | 3839 | 1200.1 | 0.01 | 3332 | 1200 | 0.00 | 2303 |
| 2400 | 2400 | 0.02 | 2082 | 2400 | 0.00 | 1919 | 2399.5 | -0.02 | 1666 | 2400 | 0.00 | 1151 |
| 9600 | 9597 | -0.03 | 520 | 9600 | 0.00 | 479 | 9592 | -0.08 | 416 | 9600 | 0.00 | 287 |
| 10417 | 10417 | 0.00 | 479 | 10425 | 0.08 | 441 | 10417 | 0.00 | 383 | 10433 | 0.16 | 264 |
| 19.2k | 19.23k | 0.16 | 259 | 19.20k | 0.00 | 239 | 19.23k | 0.16 | 207 | 19.20k | 0.00 | 143 |
| 57.6k | 57.47k | -0.22 | 86 | 57.60k | 0.00 | 79 | 57.97k | 0.64 | 68 | 57.60k | 0.00 | 47 |
| 115.2k | 116.3k | 0.94 | 42 | 115.2k | 0.00 | 39 | 114.29k | -0.79 | 34 | 115.2k | 0.00 | 23 |

| BAUD RATE | SYNC = 0, BRGH = 1, BRG16 = 1 or SYNC = 1, BRG16 = 1 | | | | | | | | | | | |
|-----------|--|---------|-----------------------|------------------|---------|-----------------------|-------------------|---------|-----------------------|------------------|---------|-----------------------|
| | Fosc = 8.000 MHz | | | Fosc = 4.000 MHz | | | Fosc = 3.6864 MHz | | | Fosc = 1.000 MHz | | |
| | Actual Rate | % Error | SPBRG value (decimal) | Actual Rate | % Error | SPBRG value (decimal) | Actual Rate | % Error | SPBRG value (decimal) | Actual Rate | % Error | SPBRG value (decimal) |
| 300 | 300.0 | 0.00 | 6666 | 300.0 | 0.01 | 3332 | 300.0 | 0.00 | 3071 | 300.1 | 0.04 | 832 |
| 1200 | 1200 | -0.02 | 1666 | 1200 | 0.04 | 832 | 1200 | 0.00 | 767 | 1202 | 0.16 | 207 |
| 2400 | 2401 | 0.04 | 832 | 2398 | 0.08 | 416 | 2400 | 0.00 | 383 | 2404 | 0.16 | 103 |
| 9600 | 9615 | 0.16 | 207 | 9615 | 0.16 | 103 | 9600 | 0.00 | 95 | 9615 | 0.16 | 25 |
| 10417 | 10417 | 0 | 191 | 10417 | 0.00 | 95 | 10473 | 0.53 | 87 | 10417 | 0.00 | 23 |
| 19.2k | 19.23k | 0.16 | 103 | 19.23k | 0.16 | 51 | 19.20k | 0.00 | 47 | 19.23k | 0.16 | 12 |
| 57.6k | 57.14k | -0.79 | 34 | 58.82k | 2.12 | 16 | 57.60k | 0.00 | 15 | — | — | — |
| 115.2k | 117.6k | 2.12 | 16 | 111.1k | -3.55 | 8 | 115.2k | 0.00 | 7 | — | — | — |

PIC16(L)F1526/7

22.4.1 AUTO-BAUD DETECT

The EUSART module supports automatic detection and calibration of the baud rate.

In the Auto-Baud Detect (ABD) mode, the clock to the BRG is reversed. Rather than the BRG clocking the incoming RXx signal, the RXx signal is timing the BRG. The Baud Rate Generator is used to time the period of a received 55h (ASCII “U”) which is the Sync character for the LIN bus. The unique feature of this character is that it has five rising edges including the Stop bit edge.

Setting the ABDEN bit of the BAUDxCON register starts the auto-baud calibration sequence (Figure 22.4.2). While the ABD sequence takes place, the EUSART state machine is held in Idle. On the first rising edge of the receive line, after the Start bit, the SPxBRGL begins counting up using the BRG counter clock as shown in Table 22-6. The fifth rising edge will occur on the RXx/DTx pin at the end of the eighth bit period. At that time, an accumulated value totaling the proper BRG period is left in the SPxBRGH:SPxBRGL register pair, the ABDEN bit is automatically cleared, and the RCxIF interrupt flag is set. A read operation on the RCxREG needs to be performed to clear the RCxIF interrupt. RCxREG content should be discarded. When calibrating for modes that do not use the SPxBRGH register the user can verify that the SPxBRGL register did not overflow by checking for 00h in the SPxBRGH register.

The BRG auto-baud clock is determined by the BRG16 and BRGH bits as shown in Table 22-6. During ABD, both the SPxBRGH and SPxBRGL registers are used as a 16-bit counter, independent of the BRG16 bit setting. While calibrating the baud rate period, the SPxBRGH and SPxBRGL registers are clocked at

1/8th the BRG base clock rate. The resulting byte measurement is the average bit time when clocked at full speed.

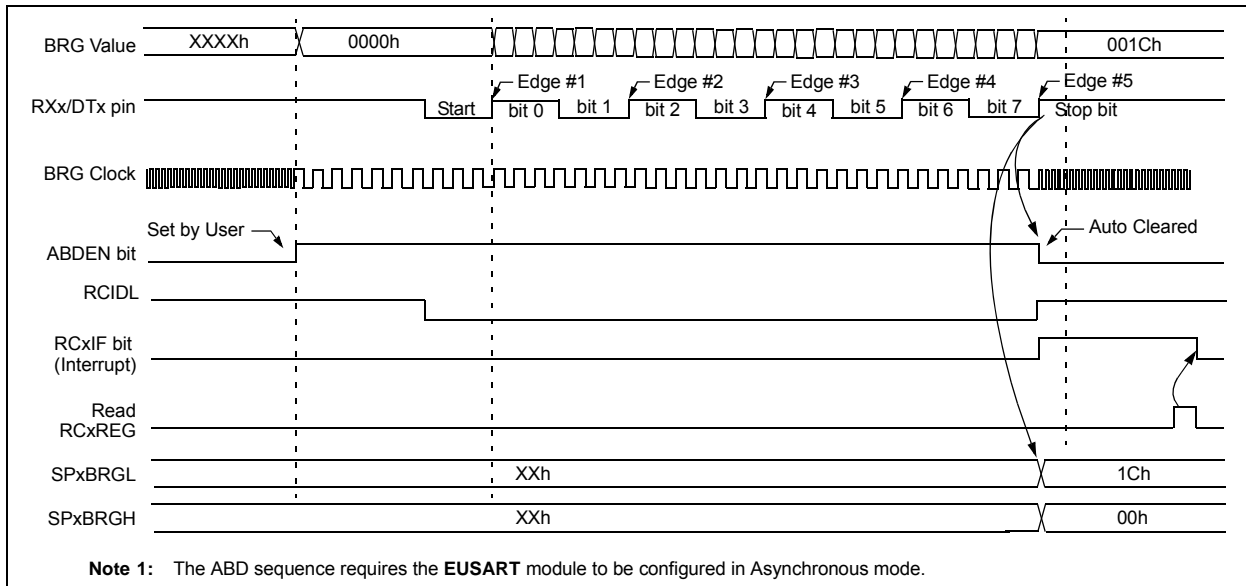
- Note 1:** If the WUE bit is set with the ABDEN bit, auto-baud detection will occur on the byte following the Break character (see Section 22.4.3 “Auto-Wake-up on Break”).
- 2:** It is up to the user to determine that the incoming character baud rate is within the range of the selected BRG clock source. Some combinations of oscillator frequency and EUSART baud rates are not possible.
- 3:** During the auto-baud process, the auto-baud counter starts counting at 1. Upon completion of the auto-baud sequence, to achieve maximum accuracy, subtract 1 from the SPxBRGH:SPxBRGL register pair.

TABLE 22-6: BRG COUNTER CLOCK RATES

| BRG16 | BRGH | BRG Base Clock | BRG ABD Clock |
|-------|------|----------------|---------------|
| 0 | 0 | Fosc/64 | Fosc/512 |
| 0 | 1 | Fosc/16 | Fosc/128 |
| 1 | 0 | Fosc/16 | Fosc/128 |
| 1 | 1 | Fosc/4 | Fosc/32 |

Note: During the ABD sequence, SPxBRGL and SPxBRGH registers are both used as a 16-bit counter, independent of BRG16 setting.

FIGURE 22-6: AUTOMATIC BAUD RATE CALIBRATION



22.4.2 AUTO-BAUD OVERFLOW

During the course of automatic baud detection, the ABDOVF bit of the BAUDxCON register will be set if the baud rate counter overflows before the fifth rising edge is detected on the RX pin. The ABDOVF bit indicates that the counter has exceeded the maximum count that can fit in the 16 bits of the SPxBRGH:SPxBRGL register pair. The overflow condition will set the RCIF flag. The counter continues to count until the fifth rising edge is detected on the RX pin. The RCIDL bit will remain false ('0') until the fifth rising edge at which time the RCIDL bit will be set. If the RCREG is read after the overflow occurs but before the fifth rising edge, then the fifth rising edge will set the RCIF again.

Terminating the auto-baud process early to clear an overflow condition will prevent proper detection of the sync character fifth rising edge. If any falling edges of the sync character have not yet occurred when the ABDEN bit is cleared then those will be falsely detected as Start bits. The following steps are recommended to clear the overflow condition:

1. Read RCREG to clear RCIF
2. If RCIDL is zero then wait for RCIF and repeat step 1.
3. Clear the ABDOVF bit.

22.4.3 AUTO-WAKE-UP ON BREAK

During Sleep mode, all clocks to the EUSART are suspended. Because of this, the Baud Rate Generator is inactive and a proper character reception cannot be performed. The Auto-Wake-up feature allows the controller to wake-up due to activity on the RXx/DTx line. This feature is available only in Asynchronous mode.

The Auto-Wake-up feature is enabled by setting the WUE bit of the BAUDxCON register. Once set, the normal receive sequence on RXx/DTx is disabled, and the EUSART remains in an Idle state, monitoring for a wake-up event independent of the CPU mode. A wake-up event consists of a high-to-low transition on the RXx/DTx line. (This coincides with the start of a Sync Break or a wake-up signal character for the LIN protocol.)

The EUSART module generates an RCxIF interrupt coincident with the wake-up event. The interrupt is generated synchronously to the Q clocks in normal CPU operating modes (Figure 22-7), and asynchronously if the device is in Sleep mode (Figure 22-8). The interrupt condition is cleared by reading the RCxREG register.

The WUE bit is automatically cleared by the low-to-high transition on the RXx line at the end of the Break. This signals to the user that the Break event is over. At this point, the EUSART module is in Idle mode waiting to receive the next character.

22.4.3.1 Special Considerations

Break Character

To avoid character errors or character fragments during a wake-up event, the wake-up character must be all zeros.

When the wake-up is enabled the function works independent of the low time on the data stream. If the WUE bit is set and a valid non-zero character is received, the low time from the Start bit to the first rising edge will be interpreted as the wake-up event. The remaining bits in the character will be received as a fragmented character and subsequent characters can result in framing or overrun errors.

Therefore, the initial character in the transmission must be all '0's. This must be 10 or more bit times, 13-bit times recommended for LIN bus, or any number of bit times for standard RS-232 devices.

Oscillator Startup Time

Oscillator start-up time must be considered, especially in applications using oscillators with longer start-up intervals (i.e., LP, XT or HS mode). The Sync Break (or wake-up signal) character must be of sufficient length, and be followed by a sufficient interval, to allow enough time for the selected oscillator to start and provide proper initialization of the EUSART.

WUE Bit

The wake-up event causes a receive interrupt by setting the RCxIF bit. The WUE bit is cleared by hardware by a rising edge on RXx/DTx. The interrupt condition is then cleared by software by reading the RCxREG register and discarding its contents.

To ensure that no actual data is lost, check the RCIDL bit to verify that a receive operation is not in process before setting the WUE bit. If a receive operation is not occurring, the WUE bit may then be set just prior to entering the Sleep mode.

PIC16(L)F1526/7

FIGURE 22-7: AUTO-WAKE-UP BIT (WUE) TIMING DURING NORMAL OPERATION

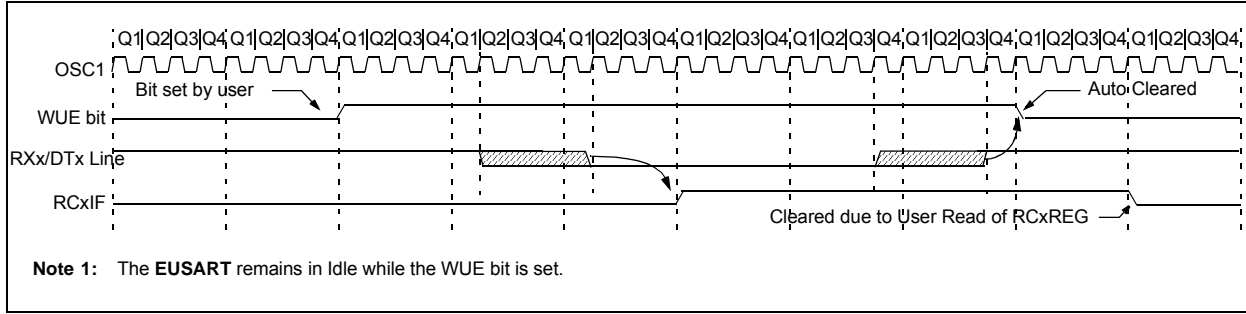
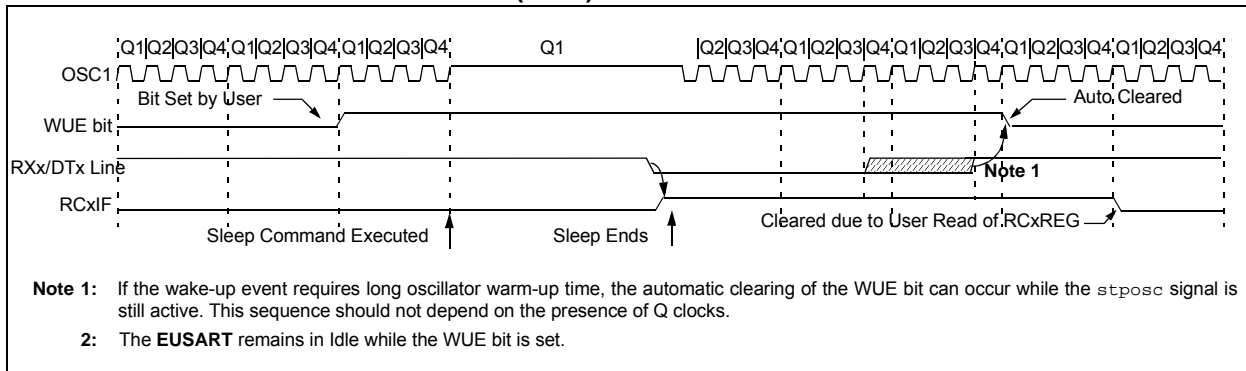


FIGURE 22-8: AUTO-WAKE-UP BIT (WUE) TIMINGS DURING SLEEP



22.4.4 BREAK CHARACTER SEQUENCE

The EUSART module has the capability of sending the special Break character sequences that are required by the LIN bus standard. A Break character consists of a Start bit, followed by 12 '0' bits and a Stop bit.

To send a Break character, set the SENDB and TXEN bits of the TXxSTA register. The Break character transmission is then initiated by a write to the TXxREG. The value of data written to TXxREG will be ignored and all '0's will be transmitted.

The SENDB bit is automatically reset by hardware after the corresponding Stop bit is sent. This allows the user to preload the transmit FIFO with the next transmit byte following the Break character (typically, the Sync character in the LIN specification).

The TRMT bit of the TXxSTA register indicates when the transmit operation is active or Idle, just as it does during normal transmission. See Figure 22-9 for the timing of the Break character sequence.

22.4.4.1 Break and Sync Transmit Sequence

The following sequence will start a message frame header made up of a Break, followed by an auto-baud Sync byte. This sequence is typical of a LIN bus master.

1. Configure the EUSART for the desired mode.
2. Set the TXEN and SENDB bits to enable the Break sequence.
3. Load the TXxREG with a dummy character to initiate transmission (the value is ignored).
4. Write '55h' to TXxREG to load the Sync character into the transmit FIFO buffer.
5. After the Break has been sent, the SENDB bit is reset by hardware and the Sync character is then transmitted.

When the TXxREG becomes empty, as indicated by the TXxIF, the next data byte can be written to TXxREG.

22.4.5 RECEIVING A BREAK CHARACTER

The Enhanced EUSART module can receive a Break character in two ways.

The first method to detect a Break character uses the FERR bit of the RCxSTA register and the Received data as indicated by RCxREG. The Baud Rate Generator is assumed to have been initialized to the expected baud rate.

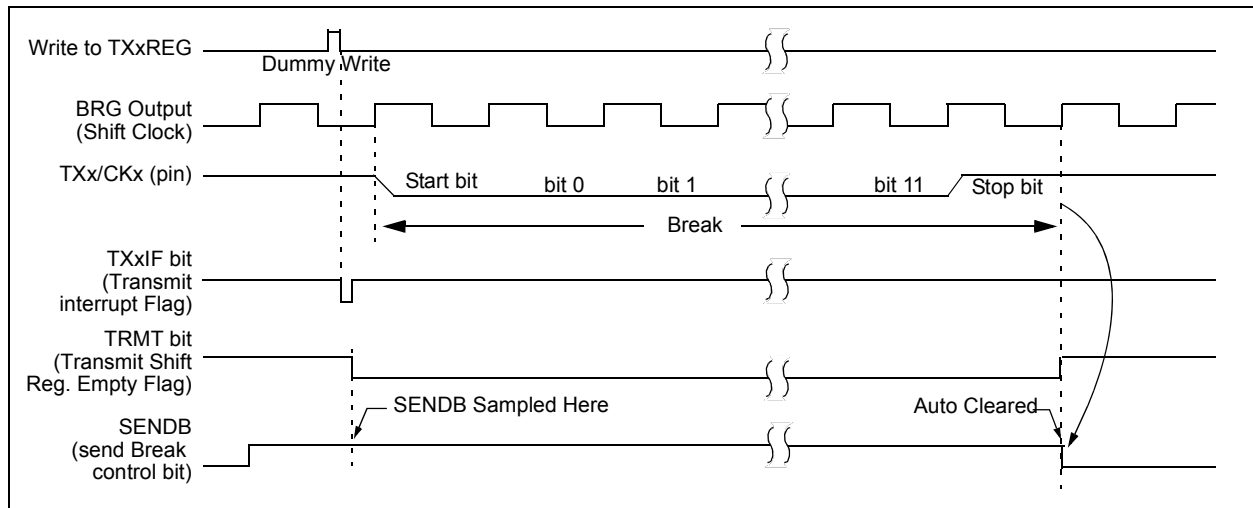
A Break character has been received when;

- RCxIF bit is set
- FERR bit is set
- RCxREG = 00h

The second method uses the Auto-Wake-up feature described in Section 22.4.3 "Auto-Wake-up on Break". By enabling this feature, the EUSART will sample the next two transitions on RXx/DTx, cause an RCxIF interrupt, and receive the next data byte followed by another interrupt.

Note that following a Break character, the user will typically want to enable the Auto-Baud Detect feature. For both methods, the user can set the ABDEN bit of the BAUDxCON register before placing the EUSART in Sleep mode.

FIGURE 22-9: SEND BREAK CHARACTER SEQUENCE



PIC16(L)F1526/7

22.5 EUSART Synchronous Mode

Synchronous serial communications are typically used in systems with a single master and one or more slaves. The master device contains the necessary circuitry for baud rate generation and supplies the clock for all devices in the system. Slave devices can take advantage of the master clock by eliminating the internal clock generation circuitry.

There are two signal lines in Synchronous mode: a bidirectional data line and a clock line. Slaves use the external clock supplied by the master to shift the serial data into and out of their respective receive and transmit shift registers. Since the data line is bidirectional, synchronous operation is half-duplex only. Half-duplex refers to the fact that master and slave devices can receive and transmit data but not both simultaneously. The EUSART can operate as either a master or slave device.

Start and Stop bits are not used in synchronous transmissions.

22.5.1 SYNCHRONOUS MASTER MODE

The following bits are used to configure the EUSART for Synchronous Master operation:

- SYNC = 1
- CSRC = 1
- SREN = 0 (for transmit); SREN = 1 (for receive)
- CREN = 0 (for transmit); CREN = 1 (for receive)
- SPEN = 1

Setting the SYNC bit of the TXxSTA register configures the device for synchronous operation. Setting the CSRC bit of the TXxSTA register configures the device as a master. Clearing the SREN and CREN bits of the RCxSTA register ensures that the device is in the Transmit mode, otherwise the device will be configured to receive. Setting the SPEN bit of the RCxSTA register enables the EUSART. If the RXx/DTx or TXx/CKx pins are shared with an analog peripheral the analog I/O functions must be disabled by clearing the corresponding ANSEL bits.

The TRIS bits corresponding to the RXx/DTx and TXx/CKx pins should be set.

22.5.1.1 Master Clock

Synchronous data transfers use a separate clock line, which is synchronous with the data. A device configured as a master transmits the clock on the TXx/CKx line. The TXx/CKx pin output driver is automatically enabled when the EUSART is configured for synchronous transmit or receive operation. Serial data bits change on the leading edge to ensure they are valid at the trailing edge of each clock. One clock cycle is generated for each data bit. Only as many clock cycles are generated as there are data bits.

22.5.1.2 Clock Polarity

A clock polarity option is provided for Microwire compatibility. Clock polarity is selected with the SCKP bit of the BAUDxCON register. Setting the SCKP bit sets the clock Idle state as high. When the SCKP bit is set, the data changes on the falling edge of each clock and is sampled on the rising edge of each clock. Clearing the SCKP bit sets the Idle state as low. When the SCKP bit is cleared, the data changes on the rising edge of each clock and is sampled on the falling edge of each clock.

22.5.1.3 Synchronous Master Transmission

Data is transferred out of the device on the RXx/DTx pin. The RXx/DTx and TXx/CKx pin output drivers are automatically enabled when the EUSART is configured for synchronous master transmit operation.

A transmission is initiated by writing a character to the TXxREG register. If the TSR still contains all or part of a previous character the new character data is held in the TXxREG until the last bit of the previous character has been transmitted. If this is the first character, or the previous character has been completely flushed from the TSR, the data in the TXxREG is immediately transferred to the TSR. The transmission of the character commences immediately following the transfer of the data to the TSR from the TXxREG.

Each data bit changes on the leading edge of the master clock and remains valid until the subsequent leading clock edge.

Note: The TSR register is not mapped in data memory, so it is not available to the user.

22.5.1.4 Synchronous Master Transmission Set-up:

1. Initialize the SPxBRGH, SPxBRGL register pair and the BRGH and BRG16 bits to achieve the desired baud rate (see [Section 22.4 “EUSART Baud Rate Generator \(BRG\)”](#)).
2. Set the RXx/DTx and TXx/CKx TRIS controls to '1'.
3. Enable the synchronous master serial port by setting bits SYNC, SPEN and CSRC. Set the TRIS bits corresponding to the RXx/DTx and TXx/CKx I/O pins.
4. Disable Receive mode by clearing bits SREN and CREN.
5. Enable Transmit mode by setting the TXEN bit.
6. If 9-bit transmission is desired, set the TX9 bit.
7. If interrupts are desired, set the TXxIE, GIE and PEIE interrupt enable bits.
8. If 9-bit transmission is selected, the ninth bit should be loaded in the TX9D bit.
9. Start transmission by loading data to the TXx-REG register.

FIGURE 22-10: SYNCHRONOUS TRANSMISSION

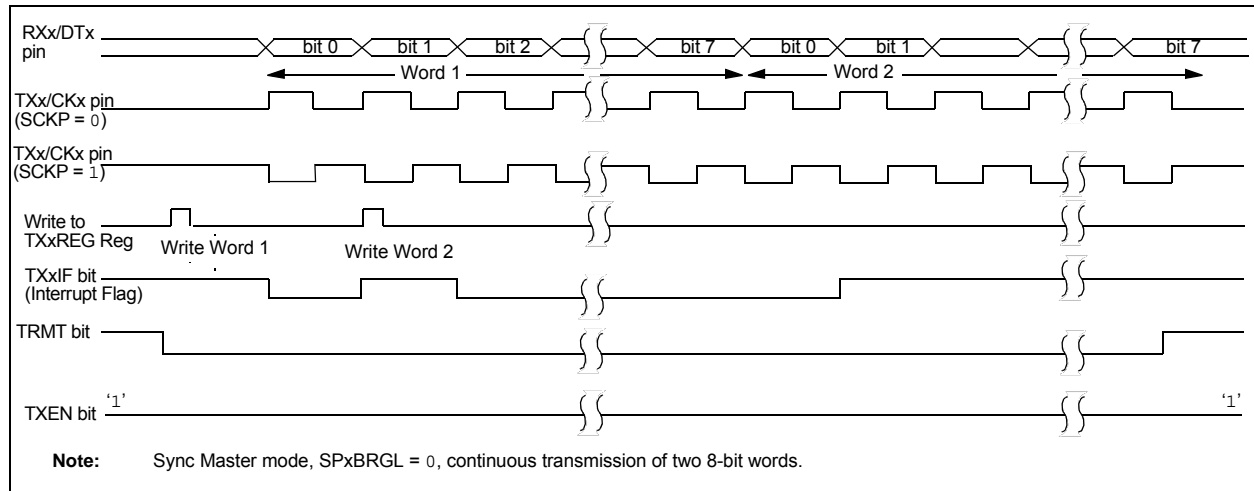
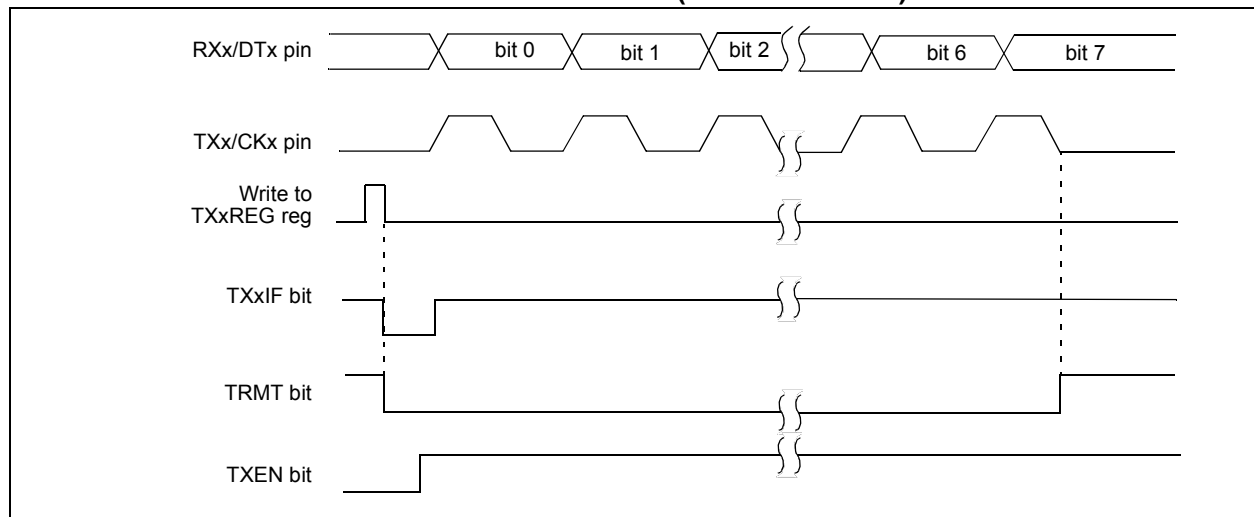


FIGURE 22-11: SYNCHRONOUS TRANSMISSION (THROUGH TXEN)



PIC16(L)F1526/7

TABLE 22-7: SUMMARY OF REGISTERS ASSOCIATED WITH SYNCHRONOUS MASTER TRANSMISSION

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Register on Page |
|----------|--|--------|------------------|--------|--------|--------|--------|--------|------------------|
| BAUD1CON | ABDOVF | RCIDL | — | SCKP | BRG16 | — | WUE | ABDEN | 260 |
| BAUD2CON | ABDOVF | RCIDL | — | SCKP | BRG16 | — | WUE | ABDEN | 260 |
| INTCON | GIE | PEIE | TMR0IE | INTE | IOCFE | TMR0IF | INTF | IOCFIF | 76 |
| PIE1 | TMR1GIE | ADIE | RC1IE | TX1IE | SSP1IE | CCP1IE | TMR2IE | TMR1IE | 76 |
| PIE4 | CCP10IE | CCP9IE | RC2IE | TX2IE | CCP8IE | CCP7IE | BCL2IE | SSP2IE | 80 |
| PIR1 | TMR1GIF | ADIF | RC1IF | TX1IF | SSP1IF | CCP1IF | TMR2IF | TMR1IF | 81 |
| PIR4 | CCP10IF | CCP9IF | RC2IF | TX2IF | CCP8IF | CCP7IF | BCL2IF | SSP2IF | 84 |
| RC1STA | SPEN | RX9 | SREN | CREN | ADDEN | FERR | OERR | RX9D | 259 |
| RC2STA | SPEN | RX9 | SREN | CREN | ADDEN | FERR | OERR | RX9D | 259 |
| SP1BRGL | EUSART1 Baud Rate Generator, Low Byte | | | | | | | | 261* |
| SP1BRGH | EUSART1 Baud Rate Generator, High Byte | | | | | | | | 261* |
| SP2BRGL | EUSART2 Baud Rate Generator, Low Byte | | | | | | | | 261* |
| SP2BRGH | EUSART2 Baud Rate Generator, High Byte | | | | | | | | 261* |
| TRISC | TRISC7 | TRISC6 | TRISC5 | TRISC4 | TRISC3 | TRISC2 | TRISC1 | TRISC0 | 120 |
| TRISG | — | — | — ⁽¹⁾ | TRISG4 | TRISG3 | TRISG2 | TRISG1 | TRISG0 | 132 |
| TX1REG | EUSART1 Transmit Register | | | | | | | | 250* |
| TX1STA | CSRC | TX9 | TXEN | SYNC | SENCB | BRGH | TRMT | TX9D | 258 |
| TX2REG | EUSART2 Transmit Register | | | | | | | | 250* |
| TX2STA | CSRC | TX9 | TXEN | SYNC | SENCB | BRGH | TRMT | TX9D | 258 |

Legend: — = unimplemented locations, read as '0'. Shaded bits are not used for synchronous master transmission.

* Page provides register information

Note 1: Unimplemented, read as '1'.

22.5.1.5 Synchronous Master Reception

Data is received at the RXx/DTx pin. The RXx/DTx pin output driver must be disabled by setting the corresponding TRIS bits when the EUSART is configured for synchronous master receive operation.

In Synchronous mode, reception is enabled by setting either the Single Receive Enable bit (SREN of the RCxSTA register) or the Continuous Receive Enable bit (CREN of the RCxSTA register).

When SREN is set and CREN is clear, only as many clock cycles are generated as there are data bits in a single character. The SREN bit is automatically cleared at the completion of one character. When CREN is set, clocks are continuously generated until CREN is cleared. If CREN is cleared in the middle of a character the CK clock stops immediately and the partial character is discarded. If SREN and CREN are both set, then SREN is cleared at the completion of the first character and CREN takes precedence.

To initiate reception, set either SREN or CREN. Data is sampled at the RXx/DTx pin on the trailing edge of the TXx/CKx clock pin and is shifted into the Receive Shift Register (RSR). When a complete character is received into the RSR, the RCxIF bit is set and the character is automatically transferred to the two character receive FIFO. The Least Significant eight bits of the top character in the receive FIFO are available in RCxREG. The RCxIF bit remains set as long as there are un-read characters in the receive FIFO.

22.5.1.6 Slave Clock

Synchronous data transfers use a separate clock line, which is synchronous with the data. A device configured as a slave receives the clock on the TXx/CKx line. The TXx/CKx pin output driver must be disabled by setting the associated TRIS bit when the device is configured for synchronous slave transmit or receive operation. Serial data bits change on the leading edge to ensure they are valid at the trailing edge of each clock. One data bit is transferred for each clock cycle. Only as many clock cycles should be received as there are data bits.

22.5.1.7 Receive Overrun Error

The receive FIFO buffer can hold two characters. An overrun error will be generated if a third character, in its entirety, is received before RCxREG is read to access the FIFO. When this happens the OERR bit of the RCxSTA register is set. Previous data in the FIFO will not be overwritten. The two characters in the FIFO buffer can be read, however, no additional characters will be received until the error is cleared. The OERR bit can only be cleared by clearing the overrun condition. If the overrun error occurred when the SREN bit is set and CREN is clear then the error is cleared by reading RCxREG.

If the overrun occurred when the CREN bit is set then the error condition is cleared by either clearing the CREN bit of the RCxSTA register or by clearing the SPEN bit which resets the EUSART.

22.5.1.8 Receiving 9-bit Characters

The EUSART supports 9-bit character reception. When the RX9 bit of the RCxSTA register is set the EUSART will shift 9-bits into the RSR for each character received. The RX9D bit of the RCxSTA register is the ninth, and Most Significant, data bit of the top unread character in the receive FIFO. When reading 9-bit data from the receive FIFO buffer, the RX9D data bit must be read before reading the 8 Least Significant bits from the RCxREG.

22.5.1.9 Synchronous Master Reception Set-up:

1. Initialize the SPxBRGH, SPxBRGL register pair for the appropriate baud rate. Set or clear the BRGH and BRG16 bits, as required, to achieve the desired baud rate.
2. Set the RXx/DTx and TXx/CKx TRIS controls to '1'.
3. Enable the synchronous master serial port by setting bits SYNC, SPEN and CSRC. Disable RXx/DTx and TXx/CKx output drivers by setting the corresponding TRIS bits.
4. Ensure bits CREN and SREN are clear.
5. If using interrupts, set the GIE and PEIE bits of the INTCON register and set RCxIE.
6. If 9-bit reception is desired, set bit RX9.
7. Start reception by setting the SREN bit or for continuous reception, set the CREN bit.
8. Interrupt flag bit RCxIF will be set when reception of a character is complete. An interrupt will be generated if the enable bit RCxIE was set.
9. Read the RCxSTA register to get the ninth bit (if enabled) and determine if any error occurred during reception.
10. Read the 8-bit received data by reading the RCxREG register.
11. If an overrun error occurs, clear the error by either clearing the CREN bit of the RCxSTA register or by clearing the SPEN bit which resets the EUSART.

PIC16(L)F1526/7

FIGURE 22-12: SYNCHRONOUS RECEPTION (MASTER MODE, SREN)

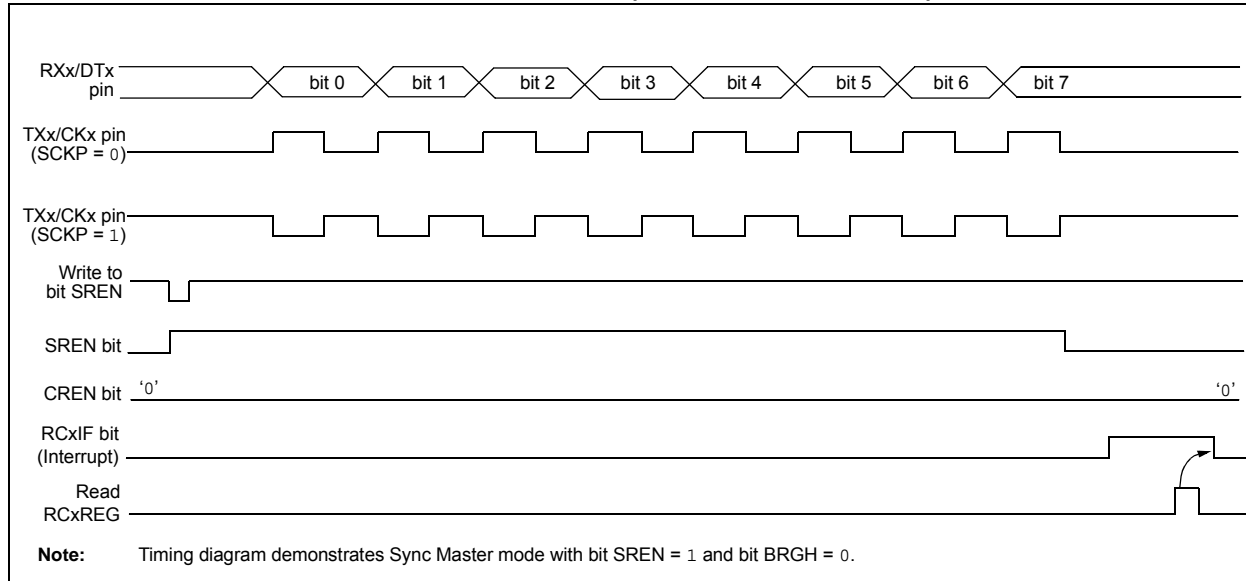


TABLE 22-8: SUMMARY OF REGISTERS ASSOCIATED WITH SYNCHRONOUS MASTER RECEPTION

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Register on Page |
|----------|--|--------|--------|-------|--------|--------|--------|--------|------------------|
| BAUD1CON | ABDOVF | RCIDL | — | SCKP | BRG16 | — | WUE | ABDEN | 260 |
| BAUD2CON | ABDOVF | RCIDL | — | SCKP | BRG16 | — | WUE | ABDEN | 260 |
| INTCON | GIE | PEIE | TMR0IE | INTE | IOCF | TMR0IF | INTF | IOCF | 76 |
| PIE1 | TMR1GIE | ADIE | RC1IE | TX1IE | SSP1IE | CCP1IE | TMR2IE | TMR1IE | 77 |
| PIE4 | CCP10IE | CCP9IE | RC2IE | TX2IE | CCP8IE | CCP7IE | BCL2IE | SSP2IE | 80 |
| PIR1 | TMR1GIF | ADIF | RC1IF | TX1IF | SSP1IF | CCP1IF | TMR2IF | TMR1IF | 81 |
| PIR4 | CCP10IF | CCP9IF | RC2IF | TX2IF | CCP8IF | CCP7IF | BCL2IF | SSP2IF | 84 |
| RC1REG | EUSART1 Receive Register | | | | | | | | 253* |
| RC1STA | SPEN | RX9 | SREN | CREN | ADDEN | FERR | OERR | RX9D | 259 |
| RC2REG | EUSART2 Receive Register | | | | | | | | 253* |
| RC2STA | SPEN | RX9 | SREN | CREN | ADDEN | FERR | OERR | RX9D | 259 |
| SP1BRGL | EUSART1 Baud Rate Generator, Low Byte | | | | | | | | 261* |
| SP1BRGH | EUSART1 Baud Rate Generator, High Byte | | | | | | | | 261* |
| SP2BRGL | EUSART2 Baud Rate Generator, Low Byte | | | | | | | | 261* |
| SP2BRGH | EUSART2 Baud Rate Generator, High Byte | | | | | | | | 261* |
| TX1STA | CSRC | TX9 | TXEN | SYNC | SENDB | BRGH | TRMT | TX9D | 258 |
| TX2STA | CSRC | TX9 | TXEN | SYNC | SENDB | BRGH | TRMT | TX9D | 258 |

Legend: — = unimplemented locations, read as '0'. Shaded bits are not used for synchronous master reception.

* Page provides register information.

22.5.2 SYNCHRONOUS SLAVE MODE

The following bits are used to configure the EUSART for Synchronous slave operation:

- SYNC = 1
- CSRC = 0
- SREN = 0 (for transmit); SREN = 1 (for receive)
- CREN = 0 (for transmit); CREN = 1 (for receive)
- SPEN = 1

Setting the SYNC bit of the TXxSTA register configures the device for synchronous operation. Clearing the CSRC bit of the TXxSTA register configures the device as a slave. Clearing the SREN and CREN bits of the RCxSTA register ensures that the device is in the Transmit mode, otherwise the device will be configured to receive. Setting the SPEN bit of the RCxSTA register enables the EUSART. If the RXx/DTx or TXx/CKx pins are shared with an analog peripheral the analog I/O functions must be disabled by clearing the corresponding ANSEL bits.

RXx/DTx and TXx/CKx pin output drivers must be disabled by setting the corresponding TRIS bits.

22.5.2.1 EUSART Synchronous Slave Transmit

The operation of the Synchronous Master and Slave modes are identical (see [Section 22.5.1.3 “Synchronous Master Transmission”](#)), except in the case of the Sleep mode.

If two words are written to the TXxREG and then the SLEEP instruction is executed, the following will occur:

1. The first character will immediately transfer to the TSR register and transmit.
2. The second word will remain in TXxREG register.
3. The TXxIF bit will not be set.
4. After the first character has been shifted out of TSR, the TXxREG register will transfer the second character to the TSR and the TXxIF bit will now be set.
5. If the PEIE and TXxIE bits are set, the interrupt will wake the device from Sleep and execute the next instruction. If the GIE bit is also set, the program will call the Interrupt Service Routine.

22.5.2.2 Synchronous Slave Transmission Set-up:

1. Set the SYNC and SPEN bits and clear the CSRC bit.
2. Set the RXx/DTx and TXx/CKx TRIS controls to '1'.
3. Clear the CREN and SREN bits.
4. If using interrupts, ensure that the GIE and PEIE bits of the INTCON register are set and set the TXxIE bit.
5. If 9-bit transmission is desired, set the TX9 bit.
6. Enable transmission by setting the TXEN bit.
7. If 9-bit transmission is selected, insert the Most Significant bit into the TX9D bit.
8. Start transmission by writing the Least Significant 8 bits to the TXxREG register.

PIC16(L)F1526/7

TABLE 22-9: SUMMARY OF REGISTERS ASSOCIATED WITH SYNCHRONOUS SLAVE TRANSMISSION

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Register on Page |
|----------|--|--------|--------|--------|--------|--------|--------|--------|------------------|
| BAUD1CON | ABDOVF | RCIDL | — | SCKP | BRG16 | — | WUE | ABDEN | 260 |
| BAUD2CON | ABDOVF | RCIDL | — | SCKP | BRG16 | — | WUE | ABDEN | 260 |
| INTCON | GIE | PEIE | TMR0IE | INTE | IOCFIE | TMR0IF | INTF | IOCFIF | 76 |
| PIE1 | TMR1GIE | ADIE | RC1IE | TX1IE | SSP1IE | CCP1IE | TMR2IE | TMR1IE | 77 |
| PIE4 | CCP10IE | CCP9IE | RC2IE | TX2IE | CCP8IE | CCP7IE | BCL2IE | SSP2IE | 80 |
| PIR1 | TMR1GIF | ADIF | RC1IF | TX1IF | SSP1IF | CCP1IF | TMR2IF | TMR1IF | 81 |
| PIR4 | CCP10IF | CCP9IF | RC2IF | TX2IF | CCP8IF | CCP7IF | BCL2IF | SSP2IF | 84 |
| RC1STA | SPEN | RX9 | SREN | CREN | ADDEN | FERR | OERR | RX9D | 259 |
| RC2STA | SPEN | RX9 | SREN | CREN | ADDEN | FERR | OERR | RX9D | 259 |
| SP1BRGL | EUSART1 Baud Rate Generator, Low Byte | | | | | | | | 261* |
| SP1BRGH | EUSART1 Baud Rate Generator, High Byte | | | | | | | | 261* |
| SP2BRGL | EUSART2 Baud Rate Generator, Low Byte | | | | | | | | 261* |
| SP2BRGH | EUSART2 Baud Rate Generator, High Byte | | | | | | | | 261* |
| TRISC | TRISC7 | TRISC6 | TRISC5 | TRISC4 | TRISC3 | TRISC2 | TRISC1 | TRISC0 | 120 |
| TX1REG | EUSART1 Transmit Register | | | | | | | | 250* |
| TX1STA | CSRC | TX9 | TXEN | SYNC | SENDB | BRGH | TRMT | TX9D | 258 |
| TX2REG | EUSART2 Transmit Register | | | | | | | | 250* |
| TX2STA | CSRC | TX9 | TXEN | SYNC | SENDB | BRGH | TRMT | TX9D | 258 |

Legend: — = unimplemented locations, read as '0'. Shaded bits are not used for synchronous slave transmission.

* Page provides register information.

22.5.2.3 EUSART Synchronous Slave Reception

The operation of the Synchronous Master and Slave modes is identical ([Section 22.5.1.5 “Synchronous Master Reception”](#)), with the following exceptions:

- Sleep
- CREN bit is always set, therefore the receiver is never Idle
- SREN bit, which is a “don’t care” in Slave mode

A character may be received while in Sleep mode by setting the CREN bit prior to entering Sleep. Once the word is received, the RSR register will transfer the data to the RCxREG register. If the RCxIE enable bit is set, the interrupt generated will wake the device from Sleep and execute the next instruction. If the GIE bit is also set, the program will branch to the interrupt vector.

22.5.2.4 Synchronous Slave Reception Set-up:

1. Set the SYNC and SPEN bits and clear the CSRC bit.
2. Set the RXx/DTx and TXx/CKx TRIS controls to ‘1’.
3. If using interrupts, ensure that the GIE and PEIE bits of the INTCON register are set and set the RCxIE bit.
4. If 9-bit reception is desired, set the RX9 bit.
5. Set the CREN bit to enable reception.
6. The RCxIF bit will be set when reception is complete. An interrupt will be generated if the RCxIE bit was set.
7. If 9-bit mode is enabled, retrieve the Most Significant bit from the RX9D bit of the RCxSTA register.
8. Retrieve the 8 Least Significant bits from the receive FIFO by reading the RCxREG register.
9. If an overrun error occurs, clear the error by either clearing the CREN bit of the RCxSTA register or by clearing the SPEN bit which resets the EUSART.

TABLE 22-10: REGISTERS ASSOCIATED WITH SYNCHRONOUS SLAVE RECEPTION

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Register on Page |
|----------|--|--------|--------|-------|--------|--------|--------|--------|------------------|
| BAUD1CON | ABDOVF | RCIDL | — | SCKP | BRG16 | — | WUE | ABDEN | 260 |
| BAUD2CON | ABDOVF | RCIDL | — | SCKP | BRG16 | — | WUE | ABDEN | 260 |
| INTCON | GIE | PEIE | TMR0IE | INTE | IOCF | TMR0IF | INTF | IOCF | 76 |
| PIE1 | TMR1GIE | ADIE | RC1IE | TX1IE | SSP1IE | CCP1IE | TMR2IE | TMR1IE | 77 |
| PIE4 | CCP10IE | CCP9IE | RC2IE | TX2IE | CCP8IE | CCP7IE | BCL2IE | SSP2IE | 80 |
| PIR1 | TMR1GIF | ADIF | RC1IF | TX1IF | SSP1IF | CCP1IF | TMR2IF | TMR1IF | 81 |
| PIR4 | CCP10IF | CCP9IF | RC2IF | TX2IF | CCP8IF | CCP7IF | BCL2IF | SSP2IF | 84 |
| RC1REG | EUSART1 Receive Register | | | | | | | | 253* |
| RC1STA | SPEN | RX9 | SREN | CREN | ADDEN | FERR | OERR | RX9D | 259 |
| RC2REG | EUSART2 Receive Register | | | | | | | | 253* |
| RC2STA | SPEN | RX9 | SREN | CREN | ADDEN | FERR | OERR | RX9D | 259 |
| SP1BRGL | EUSART1 Baud Rate Generator, Low Byte | | | | | | | | 261* |
| SP1BRGH | EUSART1 Baud Rate Generator, High Byte | | | | | | | | 261* |
| SP2BRGL | EUSART2 Baud Rate Generator, Low Byte | | | | | | | | 261* |
| SP2BRGH | EUSART2 Baud Rate Generator, High Byte | | | | | | | | 261* |
| TX1STA | CSRC | TX9 | TXEN | SYNC | SENDB | BRGH | TRMT | TX9D | 258 |
| TX2STA | CSRC | TX9 | TXEN | SYNC | SENDB | BRGH | TRMT | TX9D | 258 |

Legend: — = unimplemented locations, read as ‘0’. Shaded bits are not used for synchronous slave reception.

* Page provides register information.

PIC16(L)F1526/7

23.0 IN-CIRCUIT SERIAL PROGRAMMING™ (ICSP™)

ICSP™ programming allows customers to manufacture circuit boards with unprogrammed devices. Programming can be done after the assembly process, allowing the device to be programmed with the most recent firmware or a custom firmware. Five pins are needed for ICSP™ programming:

- ICSPCLK
- ICSPDAT
- MCLR/VPP
- VDD
- VSS

In Program/Verify mode the program memory, user IDs and the Configuration Words are programmed through serial communications. The ICSPDAT pin is a bidirectional I/O used for transferring the serial data and the ICSPCLK pin is the clock input. For more information on ICSP™ refer to the “PIC16F/LF151X/152X Memory Programming Specification” (DS41422).

23.1 High-Voltage Programming Entry Mode

The device is placed into High-Voltage Programming Entry mode by holding the ICSPCLK and ICSPDAT pins low then raising the voltage on MCLR/VPP to VIH.

23.2 Low-Voltage Programming Entry Mode

The Low-Voltage Programming Entry mode allows the PIC® Flash MCUs devices to be programmed using VDD only, without high voltage. When the LVP bit of Configuration Words is set to ‘1’, the low-voltage ICSP programming entry is enabled. To disable the Low-Voltage ICSP mode, the LVP bit must be programmed to ‘0’.

Entry into the Low-Voltage Programming Entry mode requires the following steps:

1. MCLR is brought to VIL.
2. A 32-bit key sequence is presented on ICSPDAT, while clocking ICSPCLK.

Once the key sequence is complete, MCLR must be held at VIL for as long as Program/Verify mode is to be maintained.

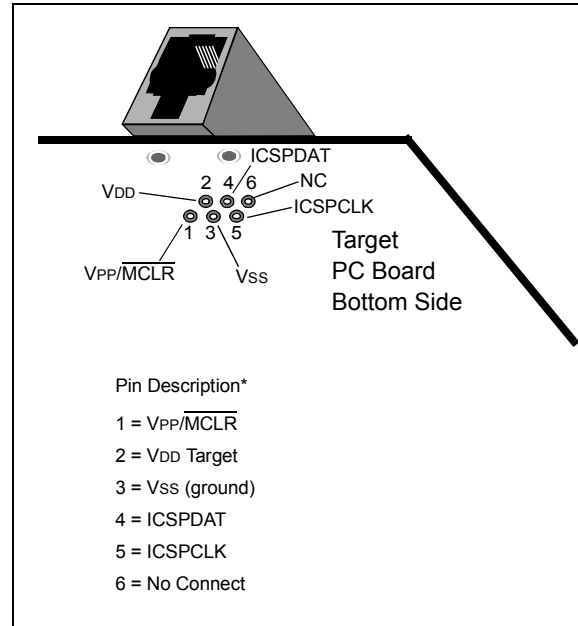
If low-voltage programming is enabled (LVP = 1), the MCLR Reset function is automatically enabled and cannot be disabled. See [Section 6.4 “Low-Power Brown-Out Reset \(LPBOR\)”](#) for more information.

The LVP bit can only be reprogrammed to ‘0’ by using the High-Voltage Programming mode.

23.3 Common Programming Interfaces

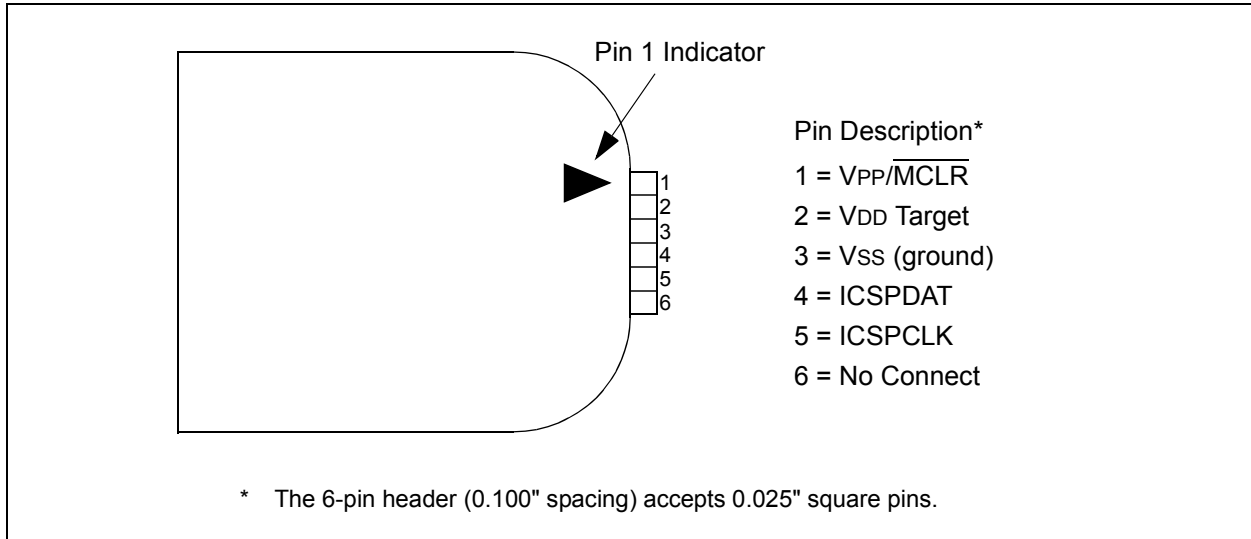
Connection to a target device is typically done through an ICSP™ header. A commonly found connector on development tools is the RJ-11 in the 6P6C (6 pin, 6 connector) configuration. See [Figure 23-1](#).

FIGURE 23-1: ICD RJ-11 STYLE CONNECTOR INTERFACE



Another connector often found in use with the PICKIT™ programmers is a standard 6-pin header with 0.1 inch spacing. Refer to [Figure 23-2](#).

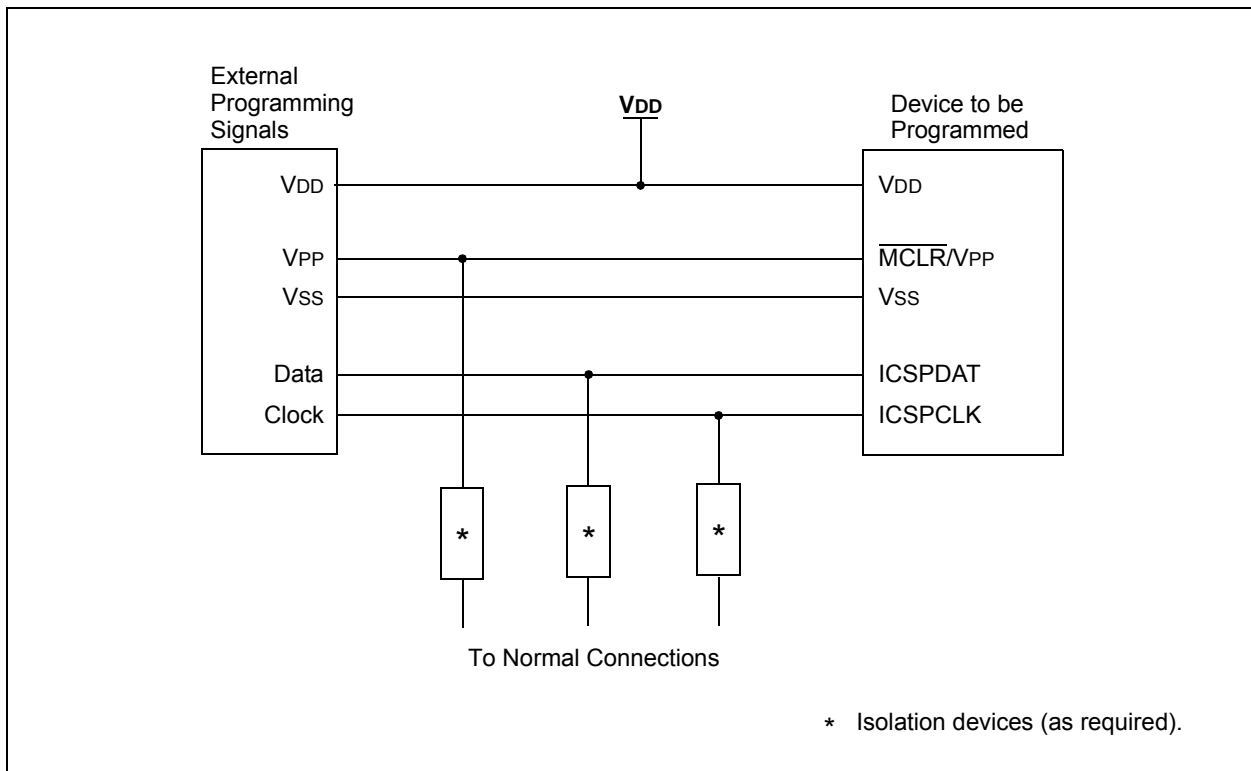
FIGURE 23-2: PICKIT™ PROGRAMMER STYLE CONNECTOR INTERFACE



For additional interface recommendations, refer to your specific device programmer manual prior to PCB design.

It is recommended that isolation devices be used to separate the programming pins from other circuitry. The type of isolation is highly dependent on the specific application and may include devices such as resistors, diodes, or even jumpers. See [Figure 23-3](#) for more information.

FIGURE 23-3: TYPICAL CONNECTION FOR ICSP™ PROGRAMMING



PIC16(L)F1526/7

24.0 INSTRUCTION SET SUMMARY

Each instruction is a 14-bit word containing the operation code (opcode) and all required operands. The opcodes are broken into three broad categories.

- Byte Oriented
- Bit Oriented
- Literal and Control

The literal and control category contains the most varied instruction word format.

Table 24-3 lists the instructions recognized by the MPASM™ assembler.

All instructions are executed within a single instruction cycle, with the following exceptions, which may take two or three cycles:

- Subroutine takes two cycles (CALL, CALLW)
- Returns from interrupts or subroutines take two cycles (RETURN, RETLW, RETFIE)
- Program branching takes two cycles (GOTO, BRA, BRW, BTFSS, BTFSC, DECFSZ, INCSFZ)
- One additional instruction cycle will be used when any instruction references an indirect file register and the file select register is pointing to program memory.

One instruction cycle consists of 4 oscillator cycles; for an oscillator frequency of 4 MHz, this gives a nominal instruction execution rate of 1 MHz.

All instruction examples use the format '0xhh' to represent a hexadecimal number, where 'h' signifies a hexadecimal digit.

24.1 Read-Modify-Write Operations

Any instruction that specifies a file register as part of the instruction performs a Read-Modify-Write (R-M-W) operation. The register is read, the data is modified, and the result is stored according to either the instruction, or the destination designator 'd'. A read operation is performed on a register even if the instruction writes to that register.

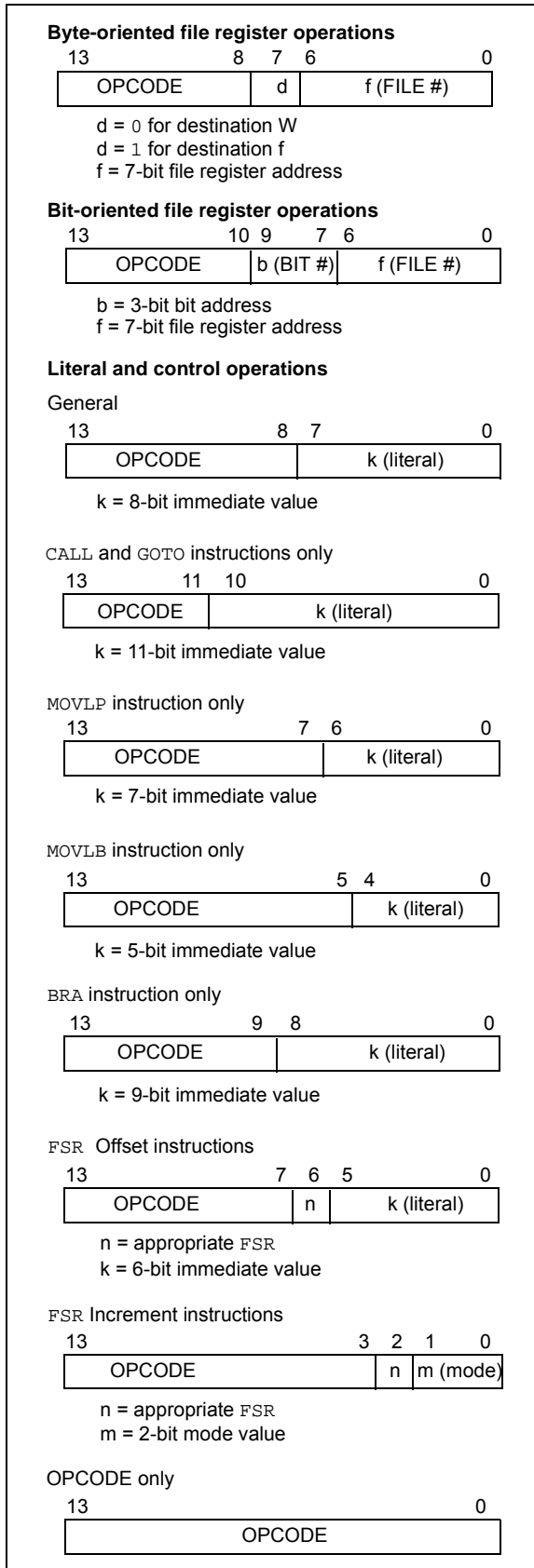
TABLE 24-1: OPCODE FIELD DESCRIPTIONS

| Field | Description |
|-------|---|
| f | Register file address (0x00 to 0x7F) |
| W | Working register (accumulator) |
| b | Bit address within an 8-bit file register |
| k | Literal field, constant data or label |
| x | Don't care location (= 0 or 1). The assembler will generate code with x = 0. It is the recommended form of use for compatibility with all Microchip software tools. |
| d | Destination select; d = 0: store result in W, d = 1: store result in file register f. Default is d = 1. |
| n | FSR or INDF number. (0-1) |
| mm | Pre-post increment-decrement mode selection |

TABLE 24-2: ABBREVIATION DESCRIPTIONS

| Field | Description |
|-----------------|-----------------|
| PC | Program Counter |
| \overline{TO} | Time-out bit |
| C | Carry bit |
| DC | Digit carry bit |
| Z | Zero bit |
| \overline{PD} | Power-down bit |

FIGURE 24-1: GENERAL FORMAT FOR INSTRUCTIONS



PIC16(L)F1526/7

TABLE 24-3: INSTRUCTION SET

| Mnemonic, Operands | Description | Cycles | 14-Bit Opcode | | | | Status Affected | Notes | |
|---|-------------|-------------------------------|---------------|-----|------|------|--------------------|----------|------|
| | | | MSb | LSb | | | | | |
| BYTE-ORIENTED FILE REGISTER OPERATIONS | | | | | | | | | |
| ADDWF | f, d | Add W and f | 1 | 00 | 0111 | dfff | ffff | C, DC, Z | 2 |
| ADDWFC | f, d | Add with Carry W and f | 1 | 11 | 1101 | dfff | ffff | C, DC, Z | 2 |
| ANDWF | f, d | AND W with f | 1 | 00 | 0101 | dfff | ffff | Z | 2 |
| ASRF | f, d | Arithmetic Right Shift | 1 | 11 | 0111 | dfff | ffff | C, Z | 2 |
| LSLF | f, d | Logical Left Shift | 1 | 11 | 0101 | dfff | ffff | C, Z | 2 |
| LSRF | f, d | Logical Right Shift | 1 | 11 | 0110 | dfff | ffff | C, Z | 2 |
| CLRF | f | Clear f | 1 | 00 | 0001 | 1fff | ffff | Z | 2 |
| CLRWF | – | Clear W | 1 | 00 | 0001 | 0000 | 00xx | Z | 2 |
| COMF | f, d | Complement f | 1 | 00 | 1001 | dfff | ffff | Z | 2 |
| DECF | f, d | Decrement f | 1 | 00 | 0011 | dfff | ffff | Z | 2 |
| INCF | f, d | Increment f | 1 | 00 | 1010 | dfff | ffff | Z | 2 |
| IORWF | f, d | Inclusive OR W with f | 1 | 00 | 0100 | dfff | ffff | Z | 2 |
| MOVF | f, d | Move f | 1 | 00 | 1000 | dfff | ffff | Z | 2 |
| MOVWF | f | Move W to f | 1 | 00 | 0000 | 1fff | ffff | | 2 |
| RLF | f, d | Rotate Left f through Carry | 1 | 00 | 1101 | dfff | ffff | C | 2 |
| RRF | f, d | Rotate Right f through Carry | 1 | 00 | 1100 | dfff | ffff | C | 2 |
| SUBWF | f, d | Subtract W from f | 1 | 00 | 0010 | dfff | ffff | C, DC, Z | 2 |
| SUBWFB | f, d | Subtract with Borrow W from f | 1 | 11 | 1011 | dfff | ffff | C, DC, Z | 2 |
| SWAPF | f, d | Swap nibbles in f | 1 | 00 | 1110 | dfff | ffff | | 2 |
| XORWF | f, d | Exclusive OR W with f | 1 | 00 | 0110 | dfff | ffff | Z | 2 |
| BYTE ORIENTED SKIP OPERATIONS | | | | | | | | | |
| DECFSZ | f, d | Decrement f, Skip if 0 | 1(2) | 00 | 1011 | dfff | ffff | | 1, 2 |
| INCFSZ | f, d | Increment f, Skip if 0 | 1(2) | 00 | 1111 | dfff | ffff | | 1, 2 |
| BIT-ORIENTED FILE REGISTER OPERATIONS | | | | | | | | | |
| BCF | f, b | Bit Clear f | 1 | 01 | 00bb | bfff | ffff | | 2 |
| BSF | f, b | Bit Set f | 1 | 01 | 01bb | bfff | ffff | | 2 |
| BIT-ORIENTED SKIP OPERATIONS | | | | | | | | | |
| BTFSC | f, b | Bit Test f, Skip if Clear | 1 (2) | 01 | 10bb | bfff | ffff | | 1, 2 |
| BTFSS | f, b | Bit Test f, Skip if Set | 1 (2) | 01 | 11bb | bfff | ffff | | 1, 2 |
| LITERAL OPERATIONS | | | | | | | | | |
| ADDLW | k | Add literal and W | 1 | 11 | 1110 | kkkk | kkkk | C, DC, Z | |
| ANDLW | k | AND literal with W | 1 | 11 | 1001 | kkkk | kkkk | Z | |
| IORLW | k | Inclusive OR literal with W | 1 | 11 | 1000 | kkkk | kkkk | Z | |
| MOVLB | k | Move literal to BSR | 1 | 00 | 0000 | 001k | kkkk | | |
| MOVLW | k | Move literal to PCLATH | 1 | 11 | 0001 | 1kkk | kkkk | | |
| MOVLW | k | Move literal to W | 1 | 11 | 0000 | kkkk | kkkk | | |
| SUBLW | k | Subtract W from literal | 1 | 11 | 1100 | kkkk | kkkk | C, DC, Z | |
| XORLW | k | Exclusive OR literal with W | 1 | 11 | 1010 | kkkk | kkkk | Z | |

Note 1: If the Program Counter (PC) is modified, or a conditional test is true, the instruction requires two cycles. The second cycle is executed as a NOP.

2: If this instruction addresses an INDF register and the MSb of the corresponding FSR is set, this instruction will require one additional instruction cycle.

TABLE 24-3: INSTRUCTION SET (CONTINUED)

| Mnemonic, Operands | Description | Cycles | 14-Bit Opcode | | | | Status Affected | Notes |
|-----------------------------|-------------|--|---------------|-----|------|------|--------------------|--------------------------------|
| | | | MSb | LSb | | | | |
| CONTROL OPERATIONS | | | | | | | | |
| BRA | k | Relative Branch | 2 | 11 | 001k | kkkk | kkkk | |
| BRW | – | Relative Branch with W | 2 | 00 | 0000 | 0000 | 1011 | |
| CALL | k | Call Subroutine | 2 | 10 | 0kkk | kkkk | kkkk | |
| CALLW | – | Call Subroutine with W | 2 | 00 | 0000 | 0000 | 1010 | |
| GOTO | k | Go to address | 2 | 10 | 1kkk | kkkk | kkkk | |
| RETFIE | k | Return from interrupt | 2 | 00 | 0000 | 0000 | 1001 | |
| RETLW | k | Return with literal in W | 2 | 11 | 0100 | kkkk | kkkk | |
| RETURN | – | Return from Subroutine | 2 | 00 | 0000 | 0000 | 1000 | |
| INHERENT OPERATIONS | | | | | | | | |
| CLRWDT | – | Clear Watchdog Timer | 1 | 00 | 0000 | 0110 | 0100 | $\overline{TO}, \overline{PD}$ |
| NOP | – | No Operation | 1 | 00 | 0000 | 0000 | 0000 | |
| OPTION | – | Load OPTION_REG register with W | 1 | 00 | 0000 | 0110 | 0010 | |
| RESET | – | Software device Reset | 1 | 00 | 0000 | 0000 | 0001 | |
| SLEEP | – | Go into Standby mode | 1 | 00 | 0000 | 0110 | 0011 | $\overline{TO}, \overline{PD}$ |
| TRIS | f | Load TRIS register with W | 1 | 00 | 0000 | 0110 | 0fff | |
| C-COMPILER OPTIMIZED | | | | | | | | |
| ADDFSR | n, k | Add Literal k to FSRn | 1 | 11 | 0001 | 0nkk | kkkk | |
| MOVIW | n mm | Move Indirect FSRn to W with pre/post inc/dec modifier, mm | 1 | 00 | 0000 | 0001 | 0nmm | Z |
| | k[n] | Move INDFn to W, Indexed Indirect. | 1 | 11 | 1111 | 0nkk | kkkk | Z |
| MOVWI | n mm | Move W to Indirect FSRn with pre/post inc/dec modifier, mm | 1 | 00 | 0000 | 0001 | 1nmm | |
| | k[n] | Move W to INDFn, Indexed Indirect. | 1 | 11 | 1111 | 1nkk | kkkk | |

Note 1: If the Program Counter (PC) is modified, or a conditional test is true, the instruction requires two cycles. The second cycle is executed as a NOP.

2: If this instruction addresses an INDF register and the MSb of the corresponding FSR is set, this instruction will require one additional instruction cycle.

3: See Table in the MOVIW and MOVWI instruction descriptions.

PIC16(L)F1526/7

24.2 Instruction Descriptions

ADDFSR **Add Literal to FSRn**

Syntax: [*label*] ADDFSR FSRn, k
Operands: $-32 \leq k \leq 31$
 $n \in [0, 1]$
Operation: $\text{FSR}(n) + k \rightarrow \text{FSR}(n)$
Status Affected: None
Description: The signed 6-bit literal 'k' is added to the contents of the FSRnH:FSRnL register pair.

FSRn is limited to the range 0000h - FFFFh. Moving beyond these bounds will cause the FSR to wrap-around.

ADDLW **Add literal and W**

Syntax: [*label*] ADDLW k
Operands: $0 \leq k \leq 255$
Operation: $(W) + k \rightarrow (W)$
Status Affected: C, DC, Z
Description: The contents of the W register are added to the 8-bit literal 'k' and the result is placed in the W register.

ADDWF **Add W and f**

Syntax: [*label*] ADDWF f,d
Operands: $0 \leq f \leq 127$
 $d \in [0,1]$
Operation: $(W) + (f) \rightarrow (\text{destination})$
Status Affected: C, DC, Z
Description: Add the contents of the W register with register 'f'. If 'd' is '0', the result is stored in the W register. If 'd' is '1', the result is stored back in register 'f'.

ADDWFC **ADD W and CARRY bit to f**

Syntax: [*label*] ADDWFC f {,d}
Operands: $0 \leq f \leq 127$
 $d \in [0,1]$
Operation: $(W) + (f) + (C) \rightarrow \text{dest}$
Status Affected: C, DC, Z
Description: Add W, the Carry flag and data memory location 'f'. If 'd' is '0', the result is placed in W. If 'd' is '1', the result is placed in data memory location 'f'.

ANDLW **AND literal with W**

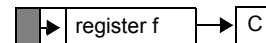
Syntax: [*label*] ANDLW k
Operands: $0 \leq k \leq 255$
Operation: $(W) .\text{AND}. (k) \rightarrow (W)$
Status Affected: Z
Description: The contents of W register are AND'ed with the 8-bit literal 'k'. The result is placed in the W register.

ANDWF **AND W with f**

Syntax: [*label*] ANDWF f,d
Operands: $0 \leq f \leq 127$
 $d \in [0,1]$
Operation: $(W) .\text{AND}. (f) \rightarrow (\text{destination})$
Status Affected: Z
Description: AND the W register with register 'f'. If 'd' is '0', the result is stored in the W register. If 'd' is '1', the result is stored back in register 'f'.

ASRF **Arithmetic Right Shift**

Syntax: [*label*] ASRF f {,d}
Operands: $0 \leq f \leq 127$
 $d \in [0,1]$
Operation: $(f<7>) \rightarrow \text{dest}<7>$
 $(f<7:1>) \rightarrow \text{dest}<6:0>$,
 $(f<0>) \rightarrow C$,
Status Affected: C, Z
Description: The contents of register 'f' are shifted one bit to the right through the Carry flag. The MSb remains unchanged. If 'd' is '0', the result is placed in W. If 'd' is '1', the result is stored back in register 'f'.



| BCF | Bit Clear f |
|------------------|--|
| Syntax: | [<i>label</i>] BCF f,b |
| Operands: | $0 \leq f \leq 127$ $0 \leq b \leq 7$ |
| Operation: | $0 \rightarrow (f)$ |
| Status Affected: | None |
| Description: | Bit 'b' in register 'f' is cleared. |

| BTFSC | Bit Test f, Skip if Clear |
|------------------|---|
| Syntax: | [<i>label</i>] BTFSC f,b |
| Operands: | $0 \leq f \leq 127$ $0 \leq b \leq 7$ |
| Operation: | skip if (f) = 0 |
| Status Affected: | None |
| Description: | If bit 'b' in register 'f' is '1', the next instruction is executed. If bit 'b', in register 'f', is '0', the next instruction is discarded, and a NOP is executed instead, making this a 2-cycle instruction. |

| BRA | Relative Branch |
|------------------|---|
| Syntax: | [<i>label</i>] BRA label [<i>label</i>] BRA \$+k |
| Operands: | $-256 \leq \text{label} - \text{PC} + 1 \leq 255$ $-256 \leq k \leq 255$ |
| Operation: | $(\text{PC}) + 1 + k \rightarrow \text{PC}$ |
| Status Affected: | None |
| Description: | Add the signed 9-bit literal 'k' to the PC. Since the PC will have incremented to fetch the next instruction, the new address will be $\text{PC} + 1 + k$. This instruction is a 2-cycle instruction. This branch has a limited range. |

| BTFSS | Bit Test f, Skip if Set |
|------------------|---|
| Syntax: | [<i>label</i>] BTFSS f,b |
| Operands: | $0 \leq f \leq 127$ $0 \leq b < 7$ |
| Operation: | skip if (f) = 1 |
| Status Affected: | None |
| Description: | If bit 'b' in register 'f' is '0', the next instruction is executed. If bit 'b' is '1', then the next instruction is discarded and a NOP is executed instead, making this a 2-cycle instruction. |

| BRW | Relative Branch with W |
|------------------|--|
| Syntax: | [<i>label</i>] BRW |
| Operands: | None |
| Operation: | $(\text{PC}) + (W) \rightarrow \text{PC}$ |
| Status Affected: | None |
| Description: | Add the contents of W (unsigned) to the PC. Since the PC will have incremented to fetch the next instruction, the new address will be $\text{PC} + 1 + (W)$. This instruction is a 2-cycle instruction. |

| BSF | Bit Set f |
|------------------|--|
| Syntax: | [<i>label</i>] BSF f,b |
| Operands: | $0 \leq f \leq 127$ $0 \leq b \leq 7$ |
| Operation: | $1 \rightarrow (f)$ |
| Status Affected: | None |
| Description: | Bit 'b' in register 'f' is set. |

PIC16(L)F1526/7

CALL Call Subroutine

Syntax: [*label*] CALL k
Operands: $0 \leq k \leq 2047$
Operation: (PC)+ 1 → TOS,
k → PC<10:0>,
(PCLATH<6:3>) → PC<14:11>
Status Affected: None
Description: Call Subroutine. First, return address (PC + 1) is pushed onto the stack. The eleven-bit immediate address is loaded into PC bits <10:0>. The upper bits of the PC are loaded from PCLATH. CALL is a 2-cycle instruction.

CALLW Subroutine Call With W

Syntax: [*label*] CALLW
Operands: None
Operation: (PC) + 1 → TOS,
(W) → PC<7:0>,
(PCLATH<6:0>) → PC<14:8>
Status Affected: None
Description: Subroutine call with W. First, the return address (PC + 1) is pushed onto the return stack. Then, the contents of W is loaded into PC<7:0>, and the contents of PCLATH into PC<14:8>. CALLW is a 2-cycle instruction.

CLRF Clear f

Syntax: [*label*] CLRF f
Operands: $0 \leq f \leq 127$
Operation: 00h → (f)
1 → Z
Status Affected: Z
Description: The contents of register 'f' are cleared and the Z bit is set.

CLRW Clear W

Syntax: [*label*] CLRW
Operands: None
Operation: 00h → (W)
1 → Z
Status Affected: Z
Description: W register is cleared. Zero bit (Z) is set.

CLRWDTClear Watchdog Timer

Syntax: [*label*] CLRWDTClear Watchdog Timer
Operands: None
Operation: 00h → WDT
0 → WDT prescaler,
1 → \overline{TO}
1 → \overline{PD}
Status Affected: \overline{TO} , \overline{PD}
Description: CLRWDTClear Watchdog Timer instruction resets the Watchdog Timer. It also resets the prescaler of the WDT. Status bits \overline{TO} and \overline{PD} are set.

COMF Complement f

Syntax: [*label*] COMF f,d
Operands: $0 \leq f \leq 127$
d ∈ [0,1]
Operation: (\bar{f}) → (destination)
Status Affected: Z
Description: The contents of register 'f' are complemented. If 'd' is '0', the result is stored in W. If 'd' is '1', the result is stored back in register 'f'.

DECF Decrement f

Syntax: [*label*] DECF f,d
Operands: $0 \leq f \leq 127$
d ∈ [0,1]
Operation: (f) - 1 → (destination)
Status Affected: Z
Description: Decrement register 'f'. If 'd' is '0', the result is stored in the W register. If 'd' is '1', the result is stored back in register 'f'.

DECFSZ Decrement f, Skip if 0

Syntax: [*label*] DECFSZ f,d

Operands: $0 \leq f \leq 127$
 $d \in [0,1]$

Operation: $(f) - 1 \rightarrow (\text{destination})$;
 skip if result = 0

Status Affected: None

Description: The contents of register 'f' are decremented. If 'd' is '0', the result is placed in the W register. If 'd' is '1', the result is placed back in register 'f'. If the result is '1', the next instruction is executed. If the result is '0', then a NOP is executed instead, making it a 2-cycle instruction.

INCFSZ Increment f, Skip if 0

Syntax: [*label*] INCFSZ f,d

Operands: $0 \leq f \leq 127$
 $d \in [0,1]$

Operation: $(f) + 1 \rightarrow (\text{destination})$,
 skip if result = 0

Status Affected: None

Description: The contents of register 'f' are incremented. If 'd' is '0', the result is placed in the W register. If 'd' is '1', the result is placed back in register 'f'. If the result is '1', the next instruction is executed. If the result is '0', a NOP is executed instead, making it a 2-cycle instruction.

GOTO Unconditional Branch

Syntax: [*label*] GOTO k

Operands: $0 \leq k \leq 2047$

Operation: $k \rightarrow \text{PC}\langle 10:0 \rangle$
 $\text{PCLATH}\langle 6:3 \rangle \rightarrow \text{PC}\langle 14:11 \rangle$

Status Affected: None

Description: GOTO is an unconditional branch. The eleven-bit immediate value is loaded into PC bits $\langle 10:0 \rangle$. The upper bits of PC are loaded from PCLATH $\langle 4:3 \rangle$. GOTO is a 2-cycle instruction.

IORLW Inclusive OR literal with W

Syntax: [*label*] IORLW k

Operands: $0 \leq k \leq 255$

Operation: $(W) .\text{OR. } k \rightarrow (W)$

Status Affected: Z

Description: The contents of the W register are OR'ed with the 8-bit literal 'k'. The result is placed in the W register.

INCF Increment f

Syntax: [*label*] INCF f,d

Operands: $0 \leq f \leq 127$
 $d \in [0,1]$

Operation: $(f) + 1 \rightarrow (\text{destination})$

Status Affected: Z

Description: The contents of register 'f' are incremented. If 'd' is '0', the result is placed in the W register. If 'd' is '1', the result is placed back in register 'f'.

IORWF Inclusive OR W with f

Syntax: [*label*] IORWF f,d

Operands: $0 \leq f \leq 127$
 $d \in [0,1]$

Operation: $(W) .\text{OR. } (f) \rightarrow (\text{destination})$

Status Affected: Z

Description: Inclusive OR the W register with register 'f'. If 'd' is '0', the result is placed in the W register. If 'd' is '1', the result is placed back in register 'f'.

PIC16(L)F1526/7

LSLF Logical Left Shift

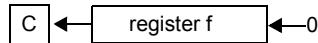
Syntax: [label] LSLF f{,d}

Operands: $0 \leq f \leq 127$
 $d \in [0,1]$

Operation: $(f<7>) \rightarrow C$
 $(f<6:0>) \rightarrow \text{dest}<7:1>$
 $0 \rightarrow \text{dest}<0>$

Status Affected: C, Z

Description: The contents of register 'f' are shifted one bit to the left through the Carry flag. A '0' is shifted into the LSB. If 'd' is '0', the result is placed in W. If 'd' is '1', the result is stored back in register 'f'.



LSRF Logical Right Shift

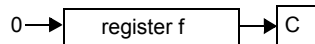
Syntax: [label] LSRF f{,d}

Operands: $0 \leq f \leq 127$
 $d \in [0,1]$

Operation: $0 \rightarrow \text{dest}<7>$
 $(f<7:1>) \rightarrow \text{dest}<6:0>$,
 $(f<0>) \rightarrow C$,

Status Affected: C, Z

Description: The contents of register 'f' are shifted one bit to the right through the Carry flag. A '0' is shifted into the MSb. If 'd' is '0', the result is placed in W. If 'd' is '1', the result is stored back in register 'f'.



MOVF Move f

Syntax: [label] MOVF f,d

Operands: $0 \leq f \leq 127$
 $d \in [0,1]$

Operation: $(f) \rightarrow (\text{dest})$

Status Affected: Z

Description: The contents of register f is moved to a destination dependent upon the status of d. If $d = 0$, destination is W register. If $d = 1$, the destination is file register f itself. $d = 1$ is useful to test a file register since status flag Z is affected.

Words: 1

Cycles: 1

Example: MOVF FSR, 0

After Instruction

W = value in FSR register

Z = 1

MOVIW **Move INDFn to W**

Syntax: [*label*] MOVIW ++FSRn
 [*label*] MOVIW --FSRn
 [*label*] MOVIW FSRn++
 [*label*] MOVIW FSRn--
 [*label*] MOVIW k[FSRn]

Operands: n ∈ [0,1]
 mm ∈ [00,01, 10, 11]
 -32 ≤ k ≤ 31

Operation: INDFn → W
 Effective address is determined by

- FSR + 1 (preincrement)
- FSR - 1 (predecrement)
- FSR + k (relative offset)

After the Move, the FSR value will be either:

- FSR + 1 (all increments)
- FSR - 1 (all decrements)
- Unchanged

Status Affected: Z

| Mode | Syntax | mm |
|---------------|--------|----|
| Preincrement | ++FSRn | 00 |
| Predecrement | --FSRn | 01 |
| Postincrement | FSRn++ | 10 |
| Postdecrement | FSRn-- | 11 |

Description: This instruction is used to move data between W and one of the indirect registers (INDFn). Before/after this move, the pointer (FSRn) is updated by pre/post incrementing/decrementing it.

Note: The INDFn registers are not physical registers. Any instruction that accesses an INDFn register actually accesses the register at the address specified by the FSRn.

FSRn is limited to the range 0000h - FFFFh. Incrementing/decrementing it beyond these bounds will cause it to wrap-around.

MOVLB **Move literal to BSR**

Syntax: [*label*] MOVLB k

Operands: 0 ≤ k ≤ 31

Operation: k → BSR

Status Affected: None

Description: The 5-bit literal 'k' is loaded into the Bank Select Register (BSR).

MOVLP **Move literal to PCLATH**

Syntax: [*label*] MOVLP k

Operands: 0 ≤ k ≤ 127

Operation: k → PCLATH

Status Affected: None

Description: The 7-bit literal 'k' is loaded into the PCLATH register.

MOVLW **Move literal to W**

Syntax: [*label*] MOVLW k

Operands: 0 ≤ k ≤ 255

Operation: k → (W)

Status Affected: None

Description: The 8-bit literal 'k' is loaded into W register. The "don't cares" will assemble as '0's.

Words: 1

Cycles: 1

Example: MOVLW 0x5A
 After Instruction
 W = 0x5A

MOVWF **Move W to f**

Syntax: [*label*] MOVWF f

Operands: 0 ≤ f ≤ 127

Operation: (W) → (f)

Status Affected: None

Description: Move data from W register to register 'f'.

Words: 1

Cycles: 1

Example: MOVWF OPTION_REG
 Before Instruction
 OPTION_REG = 0xFF
 W = 0x4F
 After Instruction
 OPTION_REG = 0x4F
 W = 0x4F

PIC16(L)F1526/7

MOVWI Move W to INDFn

Syntax: [*label*] MOVWI ++FSRn
 [*label*] MOVWI --FSRn
 [*label*] MOVWI FSRn++
 [*label*] MOVWI FSRn--
 [*label*] MOVWI k[FSRn]

Operands: $n \in [0,1]$
 $mm \in [00,01, 10, 11]$
 $-32 \leq k \leq 31$

Operation: $W \rightarrow \text{INDFn}$
 Effective address is determined by

- FSR + 1 (preincrement)
- FSR - 1 (predecrement)
- FSR + k (relative offset)

After the Move, the FSR value will be either:

- FSR + 1 (all increments)
- FSR - 1 (all decrements)

Unchanged

Status Affected: None

| Mode | Syntax | mm |
|---------------|--------|----|
| Preincrement | ++FSRn | 00 |
| Predecrement | --FSRn | 01 |
| Postincrement | FSRn++ | 10 |
| Postdecrement | FSRn-- | 11 |

Description: This instruction is used to move data between W and one of the indirect registers (INDFn). Before/after this move, the pointer (FSRn) is updated by pre/post incrementing/decrementing it.

Note: The INDFn registers are not physical registers. Any instruction that accesses an INDFn register actually accesses the register at the address specified by the FSRn.

FSRn is limited to the range 0000h - FFFFh. Incrementing/decrementing it beyond these bounds will cause it to wrap-around.

The increment/decrement operation on FSRn WILL NOT affect any Status bits.

NOP No Operation

Syntax: [*label*] NOP

Operands: None

Operation: No operation

Status Affected: None

Description: No operation.

Words: 1

Cycles: 1

Example: NOP

OPTION Load OPTION_REG Register with W

Syntax: [*label*] OPTION

Operands: None

Operation: (W) \rightarrow OPTION_REG

Status Affected: None

Description: Move data from W register to OPTION_REG register.

Words: 1

Cycles: 1

Example: OPTION

Before Instruction
 OPTION_REG = 0xFF
 W = 0x4F

After Instruction
 OPTION_REG = 0x4F
 W = 0x4F

RESET Software Reset

Syntax: [*label*] RESET

Operands: None

Operation: Execute a device Reset. Resets the $\overline{\text{RI}}$ flag of the PCON register.

Status Affected: None

Description: This instruction provides a way to execute a hardware Reset by software.

RETFIE **Return from Interrupt**

Syntax: `[label] RETFIE`

Operands: None

Operation: TOS → PC,
 1 → GIE

Status Affected: None

Description: Return from Interrupt. Stack is POPed and Top-of-Stack (TOS) is loaded in the PC. Interrupts are enabled by setting Global Interrupt Enable bit, GIE (INTCON<7>). This is a 2-cycle instruction.

Words: 1

Cycles: 2

Example: `RETFIE`

 After Interrupt

 PC = TOS

 GIE = 1

RETURN **Return from Subroutine**

Syntax: `[label] RETURN`

Operands: None

Operation: TOS → PC

Status Affected: None

Description: Return from subroutine. The stack is POPed and the top of the stack (TOS) is loaded into the program counter. This is a 2-cycle instruction.

RETLW **Return with literal in W**

Syntax: `[label] RETLW k`

Operands: $0 \leq k \leq 255$

Operation: $k \rightarrow (W)$;
 TOS → PC

Status Affected: None

Description: The W register is loaded with the 8-bit literal 'k'. The program counter is loaded from the top of the stack (the return address). This is a 2-cycle instruction.

Words: 1

Cycles: 2

Example: `CALL TABLE;W contains table`
 `;offset value`

- `;W now has table value`
-
-

`ADDWF PC ;W = offset`
 `RETLW k1 ;Begin table`
 `RETLW k2 ;`

-
-
-

`RETLW kn ; End of table`

TABLE

Before Instruction

 W = 0x07

After Instruction

 W = value of k8

RLF **Rotate Left f through Carry**

Syntax: `[label] RLF f,d`

Operands: $0 \leq f \leq 127$
 $d \in [0,1]$

Operation: See description below

Status Affected: C

Description: The contents of register 'f' are rotated one bit to the left through the Carry flag. If 'd' is '0', the result is placed in the W register. If 'd' is '1', the result is stored back in register 'f'.

Words: 1

Cycles: 1

Example: `RLF REG1,0`

Before Instruction

 REG1 = 1110 0110

 C = 0

After Instruction

 REG1 = 1110 0110

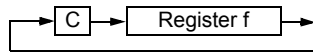
 W = 1100 1100

 C = 1

PIC16(L)F1526/7

RRF Rotate Right f through Carry

Syntax: [*label*] RRF f,d
 Operands: $0 \leq f \leq 127$
 $d \in [0,1]$
 Operation: See description below
 Status Affected: C
 Description: The contents of register 'f' are rotated one bit to the right through the Carry flag. If 'd' is '0', the result is placed in the W register. If 'd' is '1', the result is placed back in register 'f'.



SLEEP Enter Sleep mode

Syntax: [*label*] SLEEP
 Operands: None
 Operation: 00h → WDT,
 0 → WDT prescaler,
 1 → \overline{TO} ,
 0 → \overline{PD}
 Status Affected: \overline{TO} , \overline{PD}
 Description: The power-down Status bit, \overline{PD} is cleared. Time-out Status bit, \overline{TO} is set. Watchdog Timer and its prescaler are cleared.
 The processor is put into Sleep mode with the oscillator stopped.

SUBLW Subtract W from literal

Syntax: [*label*] SUBLW k
 Operands: $0 \leq k \leq 255$
 Operation: $k - (W) \rightarrow (W)$
 Status Affected: C, DC, Z
 Description: The W register is subtracted (2's complement method) from the 8-bit literal 'k'. The result is placed in the W register.

| | |
|--------|--|
| C = 0 | $W > k$ |
| C = 1 | $W \leq k$ |
| DC = 0 | $W\langle 3:0 \rangle > k\langle 3:0 \rangle$ |
| DC = 1 | $W\langle 3:0 \rangle \leq k\langle 3:0 \rangle$ |

SUBWF Subtract W from f

Syntax: [*label*] SUBWF f,d
 Operands: $0 \leq f \leq 127$
 $d \in [0,1]$
 Operation: $(f) - (W) \rightarrow (\text{destination})$
 Status Affected: C, DC, Z
 Description: Subtract (2's complement method) W register from register 'f'. If 'd' is '0', the result is stored in the W register. If 'd' is '1', the result is stored back in register 'f'.

| | |
|--------|--|
| C = 0 | $W > f$ |
| C = 1 | $W \leq f$ |
| DC = 0 | $W\langle 3:0 \rangle > f\langle 3:0 \rangle$ |
| DC = 1 | $W\langle 3:0 \rangle \leq f\langle 3:0 \rangle$ |

SUBWFB Subtract W from f with Borrow

Syntax: SUBWFB f{,d}
 Operands: $0 \leq f \leq 127$
 $d \in [0,1]$
 Operation: $(f) - (W) - (\overline{B}) \rightarrow \text{dest}$
 Status Affected: C, DC, Z
 Description: Subtract W and the BORROW flag (CARRY) from register 'f' (2's complement method). If 'd' is '0', the result is stored in W. If 'd' is '1', the result is stored back in register 'f'.

SWAPF **Swap Nibbles in f**

Syntax: [*label*] SWAPF f,d

Operands: $0 \leq f \leq 127$
 $d \in [0,1]$

Operation: ($f<3:0>$) \rightarrow (destination $<7:4>$),
 ($f<7:4>$) \rightarrow (destination $<3:0>$)

Status Affected: None

Description: The upper and lower nibbles of register 'f' are exchanged. If 'd' is '0', the result is placed in the W register. If 'd' is '1', the result is placed in register 'f'.

XORLW **Exclusive OR literal with W**

Syntax: [*label*] XORLW k

Operands: $0 \leq k \leq 255$

Operation: (W) .XOR. k \rightarrow (W)

Status Affected: Z

Description: The contents of the W register are XOR'ed with the 8-bit literal 'k'. The result is placed in the W register.

TRIS **Load TRIS Register with W**

Syntax: [*label*] TRIS f

Operands: $5 \leq f \leq 7$

Operation: (W) \rightarrow TRIS register 'f'

Status Affected: None

Description: Move data from W register to TRIS register.
 When 'f' = 5, TRISA is loaded.
 When 'f' = 6, TRISB is loaded.
 When 'f' = 7, TRISC is loaded.

XORWF **Exclusive OR W with f**

Syntax: [*label*] XORWF f,d

Operands: $0 \leq f \leq 127$
 $d \in [0,1]$

Operation: (W) .XOR. (f) \rightarrow (destination)

Status Affected: Z

Description: Exclusive OR the contents of the W register with register 'f'. If 'd' is '0', the result is stored in the W register. If 'd' is '1', the result is stored back in register 'f'.

PIC16(L)F1526/7

25.0 ELECTRICAL SPECIFICATIONS

Absolute Maximum Ratings^(†)

| | |
|---|-----------------------|
| Ambient temperature under bias | -40°C to +125°C |
| Storage temperature | -65°C to +150°C |
| Voltage on VDD with respect to VSS, PIC16F1526/7 | -0.3V to +6.5V |
| Voltage on VCAP with respect to VSS, PIC16F1526/7 | -0.3V to +4.0V |
| Voltage on VDD with respect to VSS, PIC16LF1526/7 | -0.3V to +4.0V |
| Voltage on $\overline{\text{MCLR}}$ with respect to VSS | -0.3V to +9.0V |
| Voltage on all other pins with respect to VSS | -0.3V to (VDD + 0.3V) |
| Total power dissipation ⁽¹⁾ | 800 mW |
| Maximum current out of VSS pin, -40°C ≤ TA ≤ +85°C for industrial | 350 mA |
| Maximum current out of VSS pin, -40°C ≤ TA ≤ +125°C for extended | 140 mA |
| Maximum current into VDD pin, -40°C ≤ TA ≤ +85°C for industrial | 350 mA |
| Maximum current into VDD pin, -40°C ≤ TA ≤ +125°C for extended | 140 mA |
| Clamp current, IK (VPIN < 0 or VPIN > VDD)..... | ± 20 mA |
| Maximum output current sunk by any I/O pin..... | 50 mA |
| Maximum output current sourced by any I/O pin..... | 50 mA |

Note 1: Power dissipation is calculated as follows: $P_{DIS} = V_{DD} \times \{I_{DD} - \sum I_{OH}\} + \sum \{(V_{DD} - V_{OH}) \times I_{OH}\} + \sum (V_{OL} \times I_{OL})$.

† **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.

FIGURE 25-1: PIC16F1526/7 VOLTAGE FREQUENCY GRAPH, $-40^{\circ}\text{C} \leq T_A \leq +125^{\circ}\text{C}$

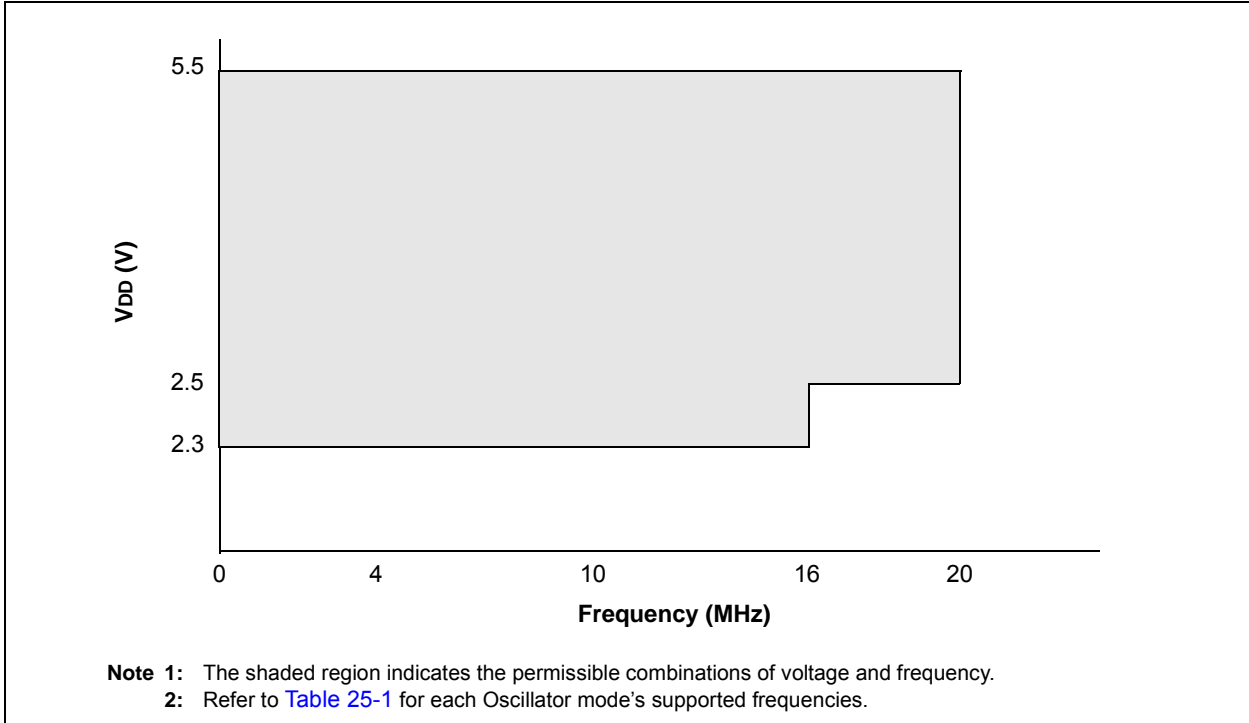
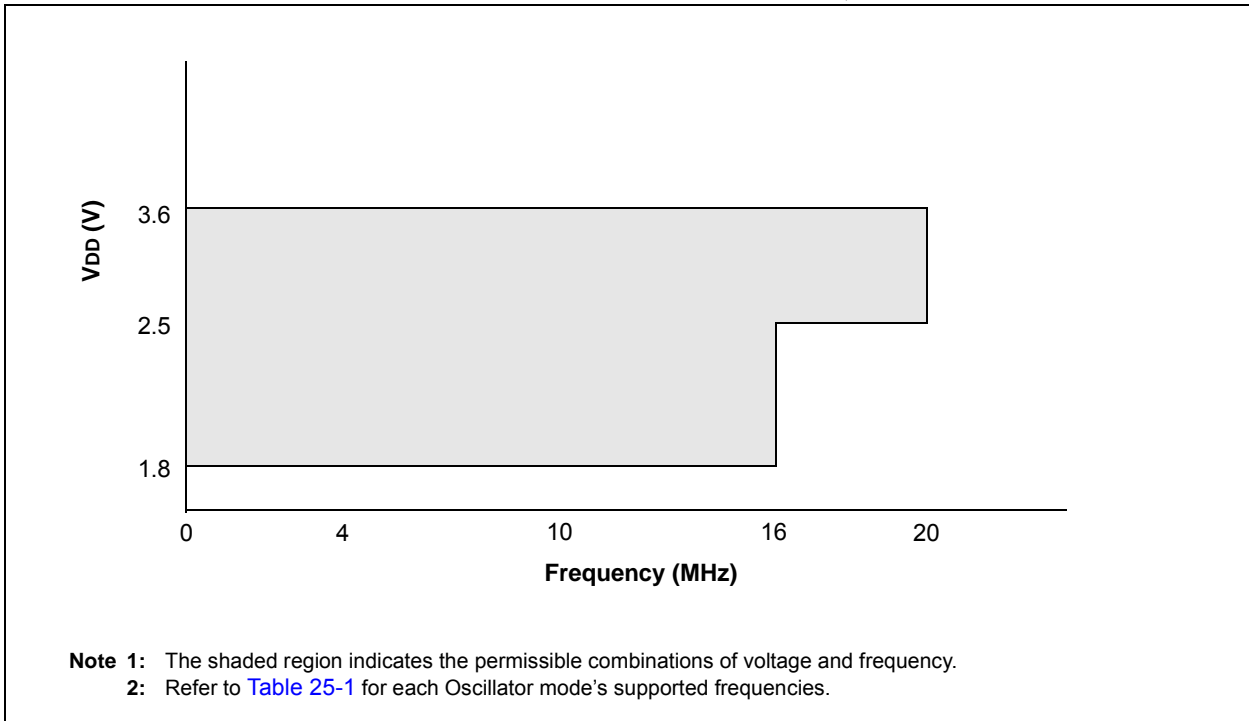
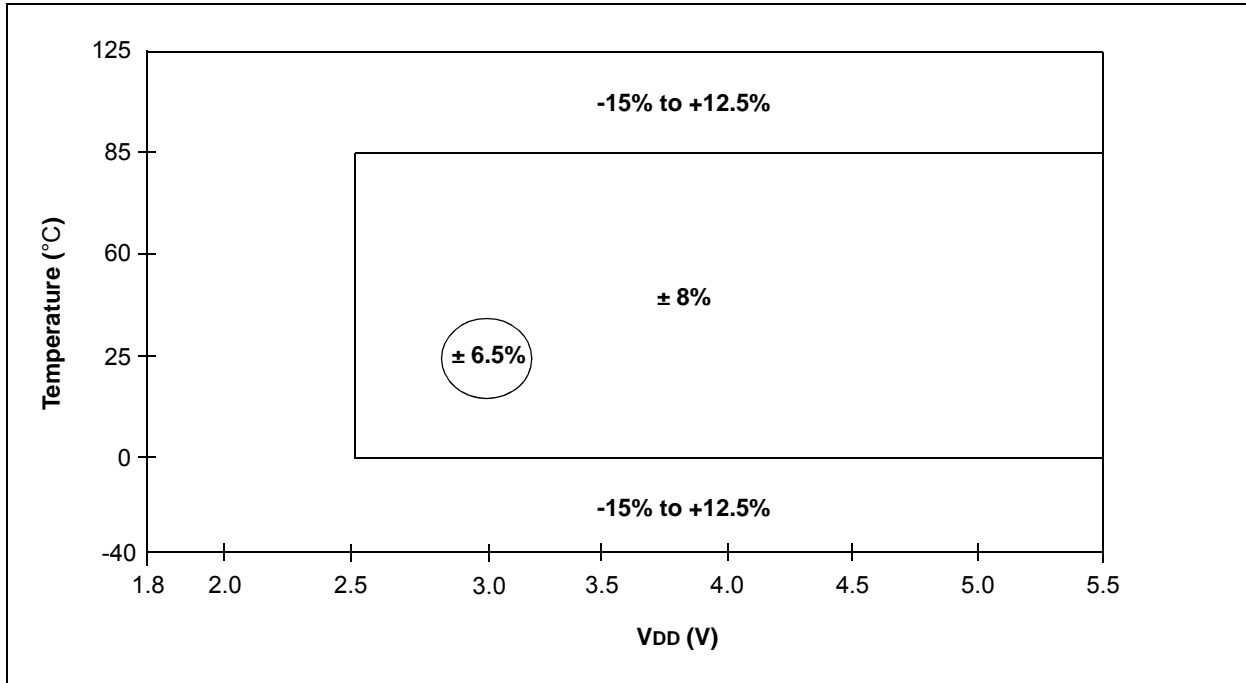


FIGURE 25-2: PIC16LF1526/7 VOLTAGE FREQUENCY GRAPH, $-40^{\circ}\text{C} \leq T_A \leq +125^{\circ}\text{C}$



PIC16(L)F1526/7

FIGURE 25-3: HFINTOSC FREQUENCY ACCURACY OVER DEVICE V_{DD} AND TEMPERATURE



25.1 DC Characteristics: Supply Voltage

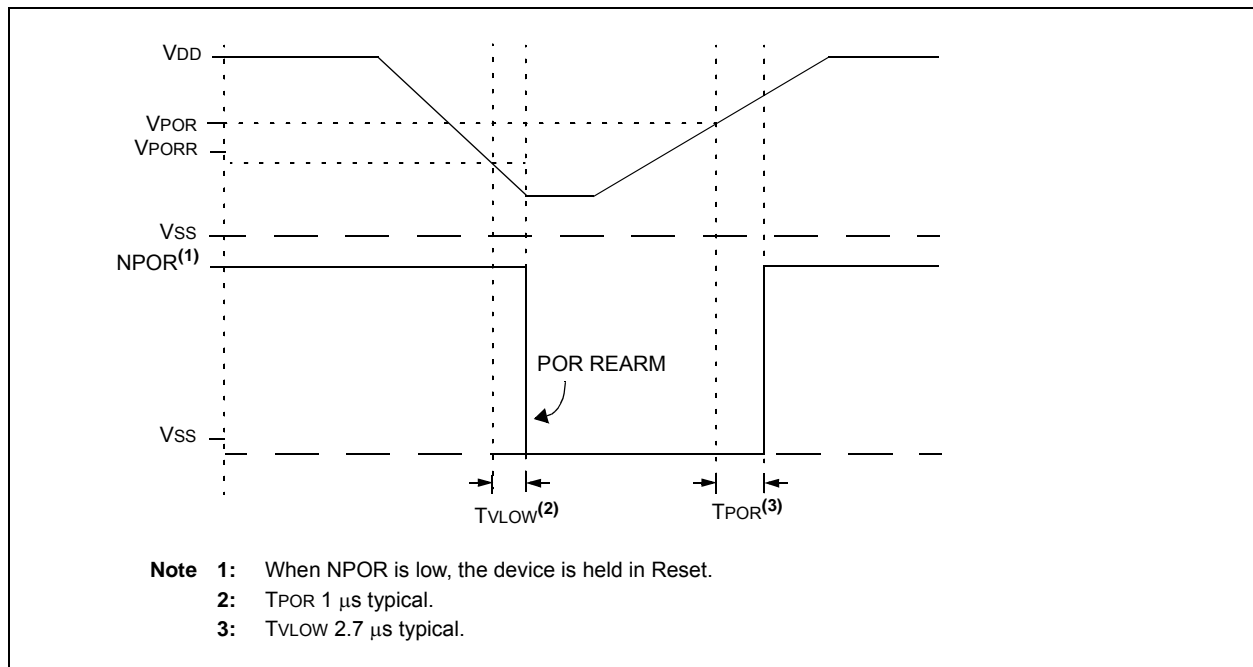
| PIC16LF1526/7 | | Standard Operating Conditions (unless otherwise stated) | | | | | |
|---|--------|---|------|------|------|-------|---|
| | | Operating temperature -40°C ≤ TA ≤ +85°C for industrial -40°C ≤ TA ≤ +125°C for extended | | | | | |
| PIC16F1526/7 | | Standard Operating Conditions (unless otherwise stated) | | | | | |
| | | Operating temperature -40°C ≤ TA ≤ +85°C for industrial -40°C ≤ TA ≤ +125°C for extended | | | | | |
| Param. No. | Sym. | Characteristic | Min. | Typ† | Max. | Units | Conditions |
| Supply Voltage (VDDMIN, VDDMAX) | | | | | | | |
| D001 | VDD | | 1.8 | — | 3.6 | V | Fosc ≤ 16 MHz |
| | | | 2.5 | — | 3.6 | V | Fosc ≤ 20 MHz |
| D001 | | | 2.3 | — | 5.5 | V | Fosc ≤ 16 MHz |
| | | | 2.5 | — | 5.5 | V | Fosc ≤ 20 MHz |
| RAM Data Retention Voltage⁽¹⁾ | | | | | | | |
| D002* | VDR | | 1.5 | — | — | V | Device in Sleep mode |
| D002* | | | 1.7 | — | — | V | Device in Sleep mode |
| D002A* | VPOR* | Power-on Reset Release Voltage | — | 1.6 | — | V | |
| Power-on Reset Rearm Voltage | | | | | | | |
| D002B* | VPORR* | | — | 0.8 | — | V | |
| D002B* | | | — | 1.5 | — | V | |
| D003 | VADFVR | Fixed Voltage Reference Voltage for ADC | -8 | | 6 | % | 1.024V, VDD ≥ 2.5V 2.048V, VDD ≥ 2.5V 4.096V, VDD ≥ 4.75V |
| D004* | SVDD | VDD Rise Rate to ensure internal Power-on Reset signal | 0.05 | — | — | V/ms | See Section 6.1 “Power-On Reset (POR)” for details. |

* These parameters are characterized but not tested.

† Data in “Typ” column is at 3.0V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

Note 1: This is the limit to which VDD can be lowered in Sleep mode without losing RAM data.

FIGURE 25-4: POR AND POR REARM WITH SLOW RISING VDD



PIC16(L)F1526/7

25.2 DC Characteristics: Supply Current (IDD)

| PIC16LF1526/7 | | Standard Operating Conditions (unless otherwise stated) | | | | | |
|---|------------------------|---|------|------|-------|------------|---|
| | | Operating temperature -40°C ≤ TA ≤ +85°C for industrial -40°C ≤ TA ≤ +125°C for extended | | | | | |
| PIC16F1526/7 | | Standard Operating Conditions (unless otherwise stated) | | | | | |
| | | Operating temperature -40°C ≤ TA ≤ +85°C for industrial -40°C ≤ TA ≤ +125°C for extended | | | | | |
| Param No. | Device Characteristics | Min. | Typ† | Max. | Units | Conditions | |
| | | | | | | VDD | Note |
| Supply Current (IDD)^(1, 2, 3) | | | | | | | |
| D009 | LDO Regulator | — | 350 | — | μA | | Device operating at 8 MHz |
| | | — | 13 | — | μA | | Sleep VREGPM = 0 |
| | | — | 0.3 | — | μA | | Sleep VREGPM = 1 |
| D010 | | — | 10 | 20 | μA | 1.8 | FOSC = 32 kHz |
| | | — | 15 | 35 | μA | 3.0 | LP Oscillator -40°C ≤ TA ≤ +85°C |
| D010 | | — | 20 | 35 | μA | 2.3 | FOSC = 32 kHz |
| | | — | 30 | 45 | μA | 3.0 | LP Oscillator |
| | | — | 40 | 50 | μA | 5.0 | -40°C ≤ TA ≤ +85°C |
| D011 | | — | 70 | 100 | μA | 1.8 | FOSC = 1 MHz |
| | | — | 130 | 200 | μA | 3.0 | XT Oscillator |
| D011 | | — | 120 | 180 | μA | 2.3 | FOSC = 1 MHz |
| | | — | 160 | 240 | μA | 3.0 | XT Oscillator |
| | | — | 240 | 360 | μA | 5.0 | |
| D012 | | — | 170 | 245 | μA | 1.8 | FOSC = 4 MHz |
| | | — | 300 | 440 | μA | 3.0 | XT Oscillator |
| D012 | | — | 290 | 475 | μA | 2.3 | FOSC = 4 MHz |
| | | — | 380 | 525 | μA | 3.0 | XT Oscillator |
| | | — | 460 | 675 | μA | 5.0 | |
| D013 | | — | 25 | 35 | μA | 1.8 | FOSC = 500 kHz |
| | | — | 42 | 60 | μA | 3.0 | External Clock (ECL), Low-Power mode |
| D013 | | — | 50 | 65 | μA | 2.3 | FOSC = 500 kHz |
| | | — | 60 | 80 | μA | 3.0 | External Clock (ECL), Low-Power mode |
| | | — | 70 | 85 | μA | 5.0 | |

* These parameters are characterized but not tested.

† Data in "Typ" column is at 3.0V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

- Note** 1: The test conditions for all IDD measurements in active operation mode are: OSC1 = external square wave, from rail-to-rail; all I/O pins tri-stated, pulled to VDD; MCLR = VDD; WDT disabled.
- 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.
- 3: 0.1 μF capacitor on VCAP pin (PIC16F1526/7).
- 4: For RC oscillator configurations, current through REXT is not included. The current through the resistor can be extended by the formula $I_R = V_{DD}/2R_{EXT}$ (mA) with REXT in kΩ.

25.2 DC Characteristics: Supply Current (IDD) (Continued)

| PIC16LF1526/7 | | Standard Operating Conditions (unless otherwise stated) | | | | | |
|---|------------------------|---|------|------|-------|------------|---|
| | | Operating temperature -40°C ≤ TA ≤ +85°C for industrial -40°C ≤ TA ≤ +125°C for extended | | | | | |
| PIC16F1526/7 | | Standard Operating Conditions (unless otherwise stated) | | | | | |
| | | Operating temperature -40°C ≤ TA ≤ +85°C for industrial -40°C ≤ TA ≤ +125°C for extended | | | | | |
| Param No. | Device Characteristics | Min. | Typ† | Max. | Units | Conditions | |
| | | | | | | VDD | Note |
| Supply Current (IDD)^(1, 2, 3) | | | | | | | |
| D014 | | — | 150 | 225 | μA | 1.8 | Fosc = 4 MHz External Clock (ECM) Medium-Power mode |
| | | — | 280 | 400 | μA | 3.0 | |
| D014 | | — | 240 | 325 | μA | 2.3 | Fosc = 4 MHz External Clock (ECM) Medium-Power mode |
| | | — | 325 | 450 | μA | 3.0 | |
| | | — | 410 | 550 | μA | 5.0 | |
| D014A | | — | 1.4 | 1.8 | mA | 3.0 | Fosc = 20 MHz External Clock (ECH) High-Power mode |
| | | — | 1.6 | 2.3 | mA | 3.6 | |
| D014A | | — | 1.45 | 1.9 | mA | 3.0 | Fosc = 20 MHz External Clock (ECH) High-Power mode |
| | | — | 1.7 | 2.4 | mA | 5.0 | |
| D015 | | — | 6.0 | 15 | μA | 1.8 | Fosc = 31 kHz LFINTOSC -40°C ≤ TA ≤ +85°C |
| | | — | 15.0 | 35 | μA | 3.0 | |
| D015 | | — | 18 | 28 | μA | 2.3 | Fosc = 31 kHz LFINTOSC -40°C ≤ TA ≤ +85°C |
| | | — | 24 | 40 | μA | 3.0 | |
| | | — | 26 | 45 | μA | 5.0 | |
| D016 | | — | 245 | 400 | μA | 1.8 | Fosc = 500 kHz HFINTOSC |
| | | — | 320 | 425 | μA | 3.0 | |
| D016 | | — | 300 | 340 | μA | 2.3 | Fosc = 500 kHz HFINTOSC |
| | | — | 340 | 370 | μA | 3.0 | |
| | | — | 380 | 450 | μA | 5.0 | |
| D017* | | — | 0.6 | 0.9 | mA | 1.8 | Fosc = 8 MHz HFINTOSC |
| | | — | 0.9 | 1.1 | mA | 3.0 | |
| D017* | | — | 0.7 | 1.0 | mA | 2.3 | Fosc = 8 MHz HFINTOSC |
| | | — | 0.9 | 1.2 | mA | 3.0 | |
| | | — | 1.1 | 1.3 | mA | 5.0 | |

* These parameters are characterized but not tested.

† Data in "Typ" column is at 3.0V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

- Note**
- 1: The test conditions for all IDD measurements in active operation mode are: OSC1 = external square wave, from rail-to-rail; all I/O pins tri-stated, pulled to VDD; MCLR = VDD; WDT disabled.
 - 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.
 - 3: 0.1 μF capacitor on VCAP pin (PIC16F1526/7).
 - 4: For RC oscillator configurations, current through REXT is not included. The current through the resistor can be extended by the formula $I_R = V_{DD}/2R_{EXT}$ (mA) with REXT in kΩ.

PIC16(L)F1526/7

25.2 DC Characteristics: Supply Current (IDD) (Continued)

| PIC16LF1526/7 | | Standard Operating Conditions (unless otherwise stated) Operating temperature -40°C ≤ TA ≤ +85°C for industrial -40°C ≤ TA ≤ +125°C for extended | | | | | |
|---|------------------------|--|------|------|-------|------------|---------------------------------------|
| PIC16F1526/7 | | Standard Operating Conditions (unless otherwise stated) Operating temperature -40°C ≤ TA ≤ +85°C for industrial -40°C ≤ TA ≤ +125°C for extended | | | | | |
| Param No. | Device Characteristics | Min. | Typ† | Max. | Units | Conditions | |
| | | | | | | VDD | Note |
| Supply Current (IDD)^(1, 2, 3) | | | | | | | |
| D018 | | — | 0.9 | 1.4 | mA | 1.8 | Fosc = 16 MHz HFINTOSC |
| | | — | 1.5 | 1.8 | mA | 3.0 | |
| D018 | | — | 1.0 | 1.5 | mA | 2.3 | Fosc = 16 MHz HFINTOSC |
| | | — | 1.5 | 1.8 | mA | 3.0 | |
| | | — | 1.7 | 1.9 | mA | 5.0 | |
| D020 | | — | 1.7 | 2.0 | mA | 3.0 | Fosc = 20 MHz HS Oscillator |
| | | — | 2.1 | 2.5 | mA | 3.6 | |
| D020 | | — | 1.8 | 2.1 | mA | 3.0 | Fosc = 20 MHz HS Oscillator |
| | | — | 2.2 | 2.7 | mA | 5.0 | |
| D021 | | — | 190 | 240 | μA | 1.8 | Fosc = 4 MHz EXTRC (Note 4) |
| | | — | 340 | 400 | μA | 3.0 | |
| D021 | | — | 250 | 350 | μA | 2.3 | Fosc = 4 MHz EXTRC (Note 4) |
| | | — | 340 | 440 | μA | 3.0 | |
| | | — | 425 | 525 | μA | 5.0 | |

* These parameters are characterized but not tested.

† Data in "Typ" column is at 3.0V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

- Note 1:** The test conditions for all IDD measurements in active operation mode are: OSC1 = external square wave, from rail-to-rail; all I/O pins tri-stated, pulled to VDD; MCLR = VDD; WDT disabled.
- Note 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.
- Note 3:** 0.1 μF capacitor on VCAP pin (PIC16F1526/7).
- Note 4:** For RC oscillator configurations, current through REXT is not included. The current through the resistor can be extended by the formula $I_R = V_{DD}/2R_{EXT}$ (mA) with REXT in kΩ.

25.3 DC Characteristics: Power-Down Currents (IPD)

| PIC16LF1526/7 | | Standard Operating Conditions (unless otherwise stated) | | | | | | |
|--|------------------------|---|------|------------|-------------|-------|------------|---|
| | | Operating temperature -40°C ≤ TA ≤ +85°C for industrial -40°C ≤ TA ≤ +125°C for extended | | | | | | |
| PIC16F1526/7 | | Standard Operating Conditions (unless otherwise stated) | | | | | | |
| | | Operating temperature -40°C ≤ TA ≤ +85°C for industrial -40°C ≤ TA ≤ +125°C for extended | | | | | | |
| Param No. | Device Characteristics | Min. | Typ† | Max. +85°C | Max. +125°C | Units | Conditions | |
| | | | | | | | VDD | Note |
| Power-down Currents (IPD)⁽²⁾ | | | | | | | | |
| D022 | Base IPD | — | 0.02 | 1.0 | 8.0 | μA | 1.8 | WDT, BOR, FVR and SOSC disabled, all peripherals inactive |
| | | — | 0.03 | 2.0 | 9.0 | μA | 3.0 | |
| D022 | Base IPD | — | 0.20 | 3.0 | 10 | μA | 2.3 | WDT, BOR, FVR and SOSC disabled, all peripherals inactive, Low-power regulator active |
| | | — | 0.30 | 4.0 | 12 | μA | 3.0 | |
| | | — | 0.47 | 6.0 | 15 | μA | 5.0 | |
| D023 | | — | 0.50 | 6.0 | 14 | μA | 1.8 | WDT Current (Note 1) |
| | | — | 0.80 | 7.0 | 17 | μA | 3.0 | |
| D023 | | — | 0.50 | 6.0 | 15 | μA | 2.3 | WDT Current (Note 1) VREGPM = 1 |
| | | — | 0.77 | 7.0 | 20 | μA | 3.0 | |
| | | — | 0.85 | 8.0 | 22 | μA | 5.0 | |
| D023A | | — | 8.5 | 24 | 27 | μA | 3.0 | FVR current (Note 1) |
| D023A | | — | 19 | 27 | 37 | μA | 3.0 | FVR current (Note 1) VREGPM = 1 |
| | | — | 20 | 29 | 45 | μA | 5.0 | |
| D024 | | — | 8.0 | 17 | 20 | μA | 3.0 | BOR Current (Note 1) |
| D024 | | — | 8.0 | 17 | 30 | μA | 3.0 | BOR Current (Note 1) VREGPM = 1 |
| | | — | 9.0 | 20 | 40 | μA | 5.0 | |
| D024A | | — | 0.30 | 4.0 | 8.0 | μA | 3.0 | LPBOR Current (Note 1) |
| D024A | | — | 0.30 | 4.0 | 14 | μA | 3.0 | LPBOR Current (Note 1) VREGPM = 1 |
| | | — | 0.45 | 8.0 | 17 | μA | 5.0 | |
| D025 | | — | 0.3 | 5.0 | 9.0 | μA | 1.8 | SOSC Current (Note 1) |
| | | — | 0.5 | 8.5 | 12 | μA | 3.0 | |
| D025 | | — | 1.1 | 6.0 | 10 | μA | 2.3 | SOSC Current (Note 1) VREGPM = 1 |
| | | — | 1.3 | 8.5 | 20 | μA | 3.0 | |
| | | — | 1.4 | 10 | 25 | μA | 5.0 | |
| D026* | | — | 0.10 | 1.0 | 9.0 | μA | 1.8 | ADC Current (Note 1, 3), No conversion in progress |
| | | — | 0.10 | 2.0 | 10 | μA | 3.0 | |
| D026* | | — | 0.16 | 3.0 | 10 | μA | 2.3 | ADC Current (Note 1, 3), No conversion in progress VREGPM = 1 |
| | | — | 0.40 | 4.0 | 11 | μA | 3.0 | |
| | | — | 0.50 | 6.0 | 16 | μA | 5.0 | |

* These parameters are characterized but not tested.

† Data in "Typ" column is at 3.0V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

- Note 1:** The peripheral Δ current can be determined by subtracting the base IPD current from this limit. Max. values should be used when calculating total current consumption.
- Note 2:** The power-down current in Sleep mode does not depend on the oscillator type. Power-down current is measured with the part in Sleep mode, with all I/O pins in high-impedance state and tied to VDD.
- Note 3:** ADC clock source is FRC.

PIC16(L)F1526/7

25.3 DC Characteristics: Power-Down Currents (IPD) (Continued)

| PIC16LF1526/7 | | Standard Operating Conditions (unless otherwise stated) Operating temperature -40°C ≤ TA ≤ +85°C for industrial -40°C ≤ TA ≤ +125°C for extended | | | | | | |
|--|------------------------|--|------|------------|-------------|-------|------------|---|
| PIC16F1526/7 | | Standard Operating Conditions (unless otherwise stated) Operating temperature -40°C ≤ TA ≤ +85°C for industrial -40°C ≤ TA ≤ +125°C for extended | | | | | | |
| Param No. | Device Characteristics | Min. | Typ† | Max. +85°C | Max. +125°C | Units | Conditions | |
| | | | | | | | VDD | Note |
| Power-down Base Current (IPD) ⁽²⁾ | | | | | | | | |
| D026A* | | — | 250 | — | — | μA | 1.8 | ADC Current (Note 1, 3), Conversion in progress |
| | | — | 250 | — | — | μA | 3.0 | |
| D026A* | | — | 280 | — | — | μA | 2.3 | ADC Current (Note 1, 3), Conversion in progress VREGPM = 1 |
| | | — | 280 | — | — | μA | 3.0 | |
| | | — | 280 | — | — | μA | 5.0 | |

* These parameters are characterized but not tested.

† Data in "Typ" column is at 3.0V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

- Note 1:** The peripheral Δ current can be determined by subtracting the base IPD current from this limit. Max. values should be used when calculating total current consumption.
- Note 2:** The power-down current in Sleep mode does not depend on the oscillator type. Power-down current is measured with the part in Sleep mode, with all I/O pins in high-impedance state and tied to VDD.
- Note 3:** ADC clock source is FRC.

25.4 DC Characteristics: I/O Ports

| DC CHARACTERISTICS | | | Standard Operating Conditions (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. | Sym. | Characteristic | Min. | Typ† | Max. | Units | Conditions |
| D030 D030A D031 D032 D033 | V _{IL} | Input Low Voltage | | | | | |
| | | I/O PORT: | | | | | |
| | | with TTL buffer | — | — | 0.8 | V | $4.5\text{V} \leq V_{DD} \leq 5.5\text{V}$ |
| | | with Schmitt Trigger buffer | — | — | $0.15 V_{DD}$ | V | $1.8\text{V} \leq V_{DD} \leq 4.5\text{V}$ |
| | | with I ² C levels | — | — | $0.2 V_{DD}$ | V | $2.0\text{V} \leq V_{DD} \leq 5.5\text{V}$ |
| | | with SMBus levels | — | — | 0.8 | V | $2.7\text{V} \leq V_{DD} \leq 5.5\text{V}$ |
| | | MCLR, OSC1 (RC mode) | — | — | $0.2 V_{DD}$ | V | (Note 1) |
| D040 D040A D041 D042 D043A D043B | V _{IH} | Input High Voltage | | | | | |
| I/O PORT: | | | | | | | |
| with TTL buffer | | 2.0 | — | — | V | $4.5\text{V} \leq V_{DD} \leq 5.5\text{V}$ | |
| with Schmitt Trigger buffer | | $0.25 V_{DD} + 0.8$ | — | — | V | $1.8\text{V} \leq V_{DD} \leq 4.5\text{V}$ | |
| with I ² C levels | | $0.8 V_{DD}$ | — | — | V | $2.0\text{V} \leq V_{DD} \leq 5.5\text{V}$ | |
| with SMBus levels | | $0.7 V_{DD}$ | — | — | V | | |
| MCLR | | 2.1 | — | — | V | $2.7\text{V} \leq V_{DD} \leq 5.5\text{V}$ | |
| D060 D061 | I _{IL} | Input Leakage Current⁽²⁾ | | | | | |
| I/O Ports | | — | ± 5 | ± 125 | nA | $V_{SS} \leq V_{PIN} \leq V_{DD}$, Pin at high impedance, 85°C | |
| MCLR ⁽³⁾ | | — | ± 50 | ± 200 | nA | $V_{SS} \leq V_{PIN} \leq V_{DD}$ Pin at high impedance, 85°C | |
| D070* | I _{PUR} | Weak Pull-up Current | | | | | |
| | | 25 25 | 100 140 | 200 300 | μA μA | $V_{DD} = 3.3\text{V}, V_{PIN} = V_{SS}$ $V_{DD} = 5.0\text{V}, V_{PIN} = V_{SS}$ | |
| D080 | V _{OL} | Output Low Voltage⁽⁴⁾ | | | | | |
| | | I/O Ports | — | — | 0.6 | V | I _{OL} = 8 mA, V _{DD} = 5V I _{OL} = 6 mA, V _{DD} = 3.3V I _{OL} = 1.8 mA, V _{DD} = 1.8V |
| D090 | V _{OH} | Output High Voltage⁽⁴⁾ | | | | | |
| | | I/O Ports | $V_{DD} - 0.7$ | — | — | V | I _{OH} = 3.5 mA, V _{DD} = 5V I _{OH} = 3 mA, V _{DD} = 3.3V I _{OH} = 1 mA, V _{DD} = 1.8V |

* These parameters are characterized but not tested.

† Data in "Typ" column is at 3.0V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

Note 1: In RC oscillator configuration, the OSC1/CLKIN pin is a Schmitt Trigger input. It is not recommended to use an external clock in RC mode.

2: Negative current is defined as current sourced by the pin.

3: The leakage current on the 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.

4: Including OSC2 in CLKOUT mode.

PIC16(L)F1526/7

25.4 DC Characteristics: I/O Ports (Continued)

| DC CHARACTERISTICS | | | Standard Operating Conditions (unless otherwise stated) | | | | |
|---|-------|--|---|------|------|-------|--|
| | | | Operating temperature -40°C ≤ TA ≤ +85°C for industrial -40°C ≤ TA ≤ +125°C for extended | | | | |
| Param No. | Sym. | Characteristic | Min. | Typ† | Max. | Units | Conditions |
| Capacitive Loading Specs on I/O Pins | | | | | | | |
| D101* | COSC2 | OSC2 pin | — | — | 15 | pF | In XT, HS and LP modes when external clock is used to drive OSC1 |
| D101A* | Cio | All I/O pins | — | — | 50 | pF | |
| D102 | | V _{CAP} Capacitor Charging Current | — | 200 | — | μA | |
| D102A | | Source/Sink capability when charging is complete | — | 0.0 | — | mA | |

* These parameters are characterized but not tested.

† Data in "Typ" column is at 3.0V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

- Note 1:** In RC oscillator configuration, the OSC1/CLKIN pin is a Schmitt Trigger input. It is not recommended to use an external clock in RC mode.
- 2:** Negative current is defined as current sourced by the pin.
- 3:** 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.
- 4:** Including OSC2 in CLKOUT mode.

25.5 Memory Programming Requirements

| DC CHARACTERISTICS | | | Standard Operating Conditions (unless otherwise stated) Operating temperature -40°C ≤ TA ≤ +85°C for industrial -40°C ≤ TA ≤ +125°C for extended | | | | |
|--|--------|---|--|------|--------|-------|---|
| Param No. | Sym. | Characteristic | Min. | Typ† | Max. | Units | Conditions |
| Program Memory Programming Specifications | | | | | | | |
| D110 | VIHH | Voltage on $\overline{\text{MCLR}}/\text{VPP}$ pin | 8.0 | — | 9.0 | V | (Note 2) |
| D111 | IDDP | Supply Current during Programming | — | — | 10 | mA | |
| D112 | VBE | VDD for Bulk Erase | 2.7 | — | VDDMAX | V | |
| D113 | VPEW | VDD for Write or Row Erase | VDDMIN | — | VDDMAX | V | |
| D114 | IPPPGM | Current on $\overline{\text{MCLR}}/\text{VPP}$ during Erase/Write | — | 1.0 | — | mA | |
| D115 | IDDPGM | Current on VDD during Erase/Write | — | 5.0 | — | mA | |
| Program Flash Memory | | | | | | | |
| D121 | EP | Cell Endurance | 10K | — | — | E/W | -40°C to +85°C (Note 1) |
| D122 | VPRW | VDD for Read/Write | VDDMIN | — | VDDMAX | V | |
| D123 | TIW | Self-timed Write Cycle Time | — | 2 | 2.5 | ms | Provided no other specifications are violated |
| D124 | TRETD | Characteristic Retention | — | 40 | — | Year | |
| D125 | EHEFC | High-Endurance Flash Cell | 100K | — | — | E/W | 0°C to +60°C lower byte Last 128 addresses |

† Data in “Typ” column is at 3.0V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

Note 1: Self-write and Block Erase.

Note 2: Required only if single-supply programming is disabled.

PIC16(L)F1526/7

25.6 Thermal Considerations

| Standard Operating Conditions (unless otherwise stated) | | | | | |
|--|----------------|--|------|-----------------------------|--|
| Operating temperature $-40^{\circ}\text{C} \leq T_A \leq +125^{\circ}\text{C}$ | | | | | |
| Param No. | Sym. | Characteristic | Typ. | Units | Conditions |
| TH01 | θ_{JA} | Thermal Resistance Junction to Ambient | 48.3 | $^{\circ}\text{C}/\text{W}$ | 64-pin TQFP (10x10 mm) package |
| | | | 28.0 | $^{\circ}\text{C}/\text{W}$ | 64-pin QFN (9x9 mm) package |
| TH02 | θ_{JC} | Thermal Resistance Junction to Case | 26.1 | $^{\circ}\text{C}/\text{W}$ | 64-pin TQFP (10x10 mm) package |
| | | | 1.2 | $^{\circ}\text{C}/\text{W}$ | 64-pin QFN (9x9 mm) package |
| TH03 | T_{JMAX} | Maximum Junction Temperature | 150 | $^{\circ}\text{C}$ | |
| TH04 | PD | Power Dissipation | — | W | $PD = P_{INTERNAL} + P_{I/O}$ |
| TH05 | $P_{INTERNAL}$ | Internal Power Dissipation | — | W | $P_{INTERNAL} = I_{DD} \times V_{DD}^{(1)}$ |
| TH06 | $P_{I/O}$ | I/O Power Dissipation | — | W | $P_{I/O} = \sum (I_{OL} \times V_{OL}) + \sum (I_{OH} \times (V_{DD} - V_{OH}))$ |
| TH07 | P_{DER} | Derated Power | — | W | $P_{DER} = PD_{MAX} (T_J - T_A) / \theta_{JA}^{(2)}$ |

Note 1: I_{DD} is current to run the chip alone without driving any load on the output pins.

2: T_A = Ambient Temperature; T_J = Junction Temperature.

25.7 Timing Parameter Symbology

The timing parameter symbols have been created with one of the following formats:

1. TppS2ppS
2. TppS

| | | | |
|----------|-----------|---|------|
| T | | | |
| F | Frequency | T | Time |

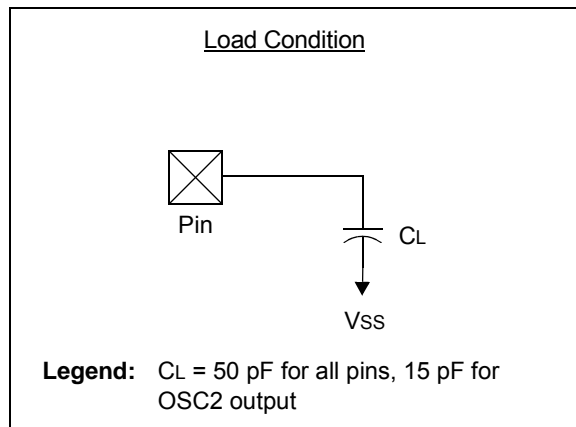
Lowercase letters (pp) and their meanings:

| | | | |
|-----------|-----------------|-----|------------------------------------|
| pp | | | |
| cc | CCP1 | osc | OSC1 |
| ck | CLKOUT | rd | \overline{RD} |
| cs | \overline{CS} | rw | \overline{RD} or \overline{WR} |
| di | SDIx | sc | SCKx |
| do | SDO | ss | \overline{SS} |
| dt | Data in | t0 | T0CKI |
| io | I/O PORT | t1 | T1CKI |
| mc | MCLR | wr | \overline{WR} |

Uppercase letters and their meanings:

| | | | |
|----------|--------------------------|---|----------------|
| S | | | |
| F | Fall | P | Period |
| H | High | R | Rise |
| I | Invalid (High-impedance) | V | Valid |
| L | Low | Z | High-impedance |

FIGURE 25-5: LOAD CONDITIONS



PIC16(L)F1526/7

25.8 AC Characteristics: PIC16(L)F1526/7-I/E

FIGURE 25-6: CLOCK TIMING

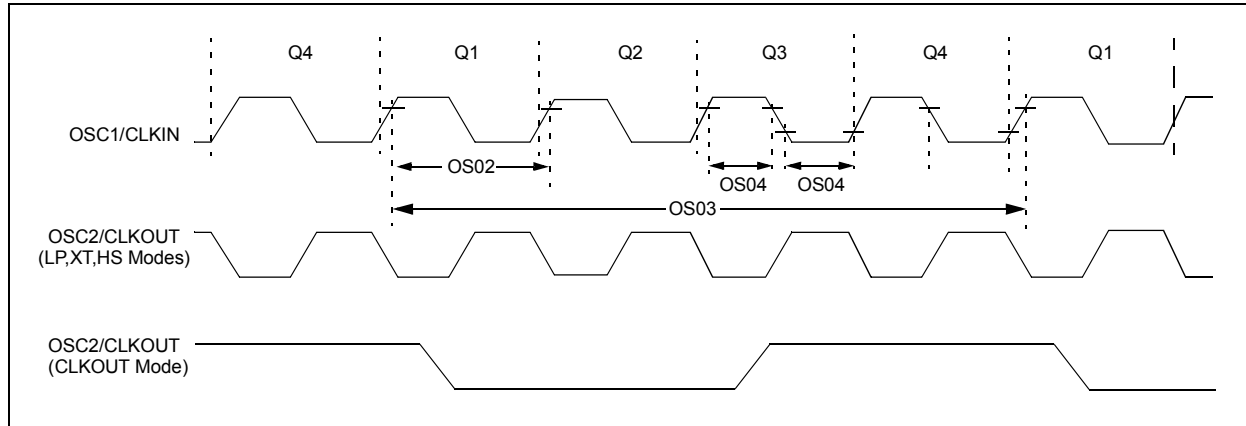


TABLE 25-1: CLOCK OSCILLATOR TIMING REQUIREMENTS

| Standard Operating Conditions (unless otherwise stated) | | | | | | | |
|--|-------------------------------------|---|--------|---------------|---------------------------------------|---------------|----------------------|
| Operating temperature $-40^{\circ}\text{C} \leq T_A \leq +125^{\circ}\text{C}$ | | | | | | | |
| Param No. | Sym. | Characteristic | Min. | Typ† | Max. | Units | Conditions |
| OS01 | Fosc | External CLKIN Frequency ⁽¹⁾ | DC | — | 0.5 | MHz | External Clock (ECL) |
| | | | DC | — | 4 | MHz | External Clock (ECM) |
| | | | DC | — | 20 | MHz | External Clock (ECH) |
| | Oscillator Frequency ⁽¹⁾ | — | 32.768 | — | kHz | LP Oscillator | |
| | | 0.1 | — | 4 | MHz | XT Oscillator | |
| 1 | | — | 4 | MHz | HS Oscillator | | |
| 1 | | — | 20 | MHz | HS Oscillator, $V_{DD} > 2.7\text{V}$ | | |
| OS02 | Tosc | External CLKIN Period ⁽¹⁾ | 27 | — | ∞ | μs | LP Oscillator |
| | | | 250 | — | ∞ | ns | XT Oscillator |
| | | | 50 | — | ∞ | ns | HS Oscillator |
| | | | 50 | — | ∞ | ns | External Clock (EC) |
| | Oscillator Period ⁽¹⁾ | — | 30.5 | — | μs | LP Oscillator | |
| 250 | — | 10,000 | ns | XT Oscillator | | | |
| 50 | — | 1,000 | ns | HS Oscillator | | | |
| 250 | — | — | ns | RC Oscillator | | | |
| OS03 | Tcy | Instruction Cycle Time ⁽¹⁾ | 200 | Tcy | DC | ns | $T_{CY} = 4/F_{OSC}$ |
| OS04* | TosH, TosL | External CLKIN High, External CLKIN Low | 2 | — | — | μs | LP oscillator |
| | | | 100 | — | — | ns | XT oscillator |
| | | | 20 | — | — | ns | HS oscillator |
| OS05* | TosR, TosF | External CLKIN Rise, External CLKIN Fall | 0 | — | — | ns | LP oscillator |
| | | | 0 | — | — | ns | XT oscillator |
| | | | 0 | — | — | ns | HS oscillator |

* These parameters are characterized but not tested.

† Data in "Typ" column is at 3.0V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

Note 1: 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 OSC1 pin. When an external clock input is used, the "max" cycle time limit is "DC" (no clock) for all devices.

TABLE 25-2: OSCILLATOR PARAMETERS

| Standard Operating Conditions (unless otherwise stated) | | | | | | | | |
|--|----------|--|-----------------|------|------|------|---------------|--|
| Operating Temperature $-40^{\circ}\text{C} \leq T_A \leq +125^{\circ}\text{C}$ | | | | | | | | |
| Param No. | Sym. | Characteristic | Freq. Tolerance | Min. | Typ† | Max. | Units | Conditions |
| OS08 | HFosc | Internal Calibrated HFINTOSC Frequency (Note 1) | $\pm 6.5\%$ | — | 16.0 | — | MHz | $V_{DD} = 3.0\text{V}$ at 25°C (Note 2) |
| OS09 | LFosc | Internal LFINTOSC Frequency | — | — | 31 | — | kHz | (Note 3) |
| OS10* | Tiosc ST | HFINTOSC Wake-up from Sleep Start-up Time | — | — | 5 | 15 | μs | VREGPM = 0 |

* These parameters are characterized but not tested.

† Data in “Typ” column is at 3.0V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

- Note 1:** To ensure these oscillator frequency tolerances, V_{DD} and V_{SS} must be capacitively decoupled as close to the device as possible. 0.1 μF and 0.01 μF values in parallel are recommended.
- Note 2:** See [Figure 25-3](#), HFINTOSC Frequency Accuracy over V_{DD} and Temperature.
- Note 3:** See [Figure 26-60](#) and [Figure 26-61](#), LFINTOSC Frequency Characteristics over V_{DD} and Temperature.

PIC16(L)F1526/7

FIGURE 25-7: CLKOUT AND I/O TIMING

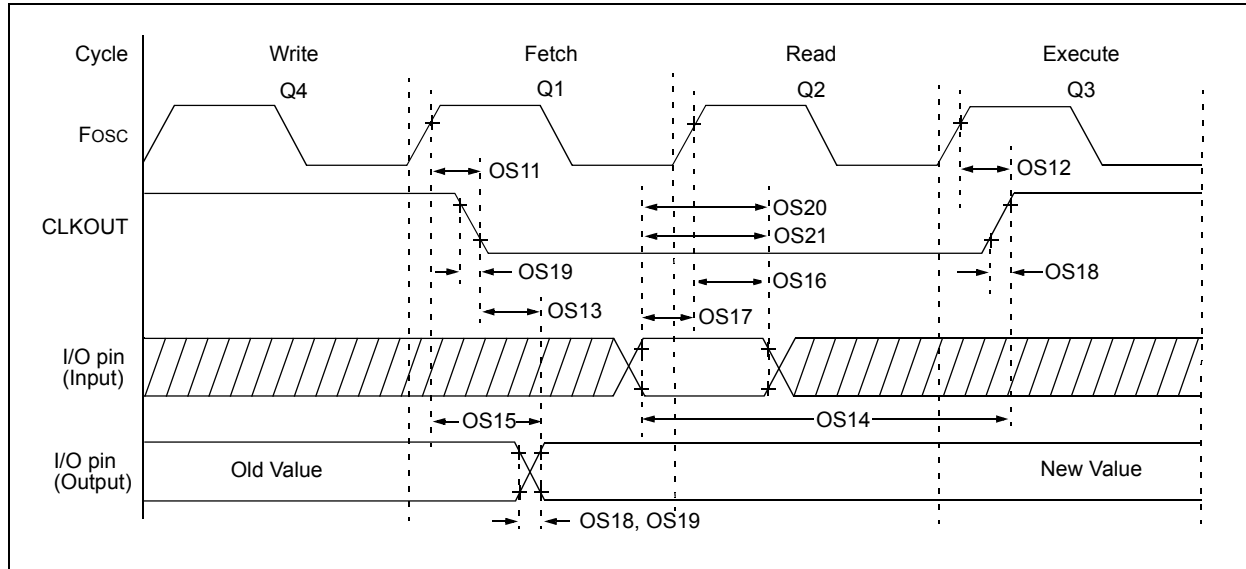


TABLE 25-3: CLKOUT AND I/O TIMING PARAMETERS

| Standard Operating Conditions (unless otherwise stated) | | | | | | | |
|--|----------|--|---------------|----------|----------|-------|------------------------------|
| Operating Temperature $-40^{\circ}\text{C} \leq T_A \leq +125^{\circ}\text{C}$ | | | | | | | |
| Param No. | Sym. | Characteristic | Min. | Typ† | Max. | Units | Conditions |
| OS11 | TosH2ckL | Fosc \uparrow to CLKOUT \downarrow ⁽¹⁾ | — | — | 70 | ns | VDD = 3.3-5.0V |
| OS12 | TosH2ckH | Fosc \uparrow to CLKOUT \uparrow ⁽¹⁾ | — | — | 72 | ns | VDD = 3.3-5.0V |
| OS13 | TckL2ioV | CLKOUT \downarrow to Port out valid ⁽¹⁾ | — | — | 20 | ns | |
| OS14 | TioV2ckH | Port input valid before CLKOUT \uparrow ⁽¹⁾ | Tosc + 200 ns | — | — | ns | |
| OS15 | TosH2ioV | Fosc \uparrow (Q1 cycle) to Port out valid | — | 50 | 70* | ns | VDD = 3.3-5.0V |
| OS16 | TosH2ioL | Fosc \uparrow (Q2 cycle) to Port input invalid (I/O in setup time) | 50 | — | — | ns | VDD = 3.3-5.0V |
| OS17 | TioV2osH | Port input valid to Fosc \uparrow (Q2 cycle) (I/O in setup time) | 20 | — | — | ns | |
| OS18* | TioR | Port output rise time | — | 40 15 | 72 32 | ns | VDD = 1.8V VDD = 3.3-5.0V |
| OS19* | TioF | Port output fall time | — | 28 15 | 55 30 | ns | VDD = 1.8V VDD = 3.3-5.0V |
| OS20* | Tinp | INT pin input high or low time | 25 | — | — | ns | |
| OS21* | Tioc | Interrupt-on-change new input level time | 25 | — | — | ns | |

* These parameters are characterized but not tested.

† Data in "Typ" column is at 3.0V, 25°C unless otherwise stated.

Note 1: Measurements are taken in RC mode where CLKOUT output is 4 x Tosc.

FIGURE 25-8: RESET, WATCHDOG TIMER, OSCILLATOR START-UP TIMER AND POWER-UP TIMER TIMING

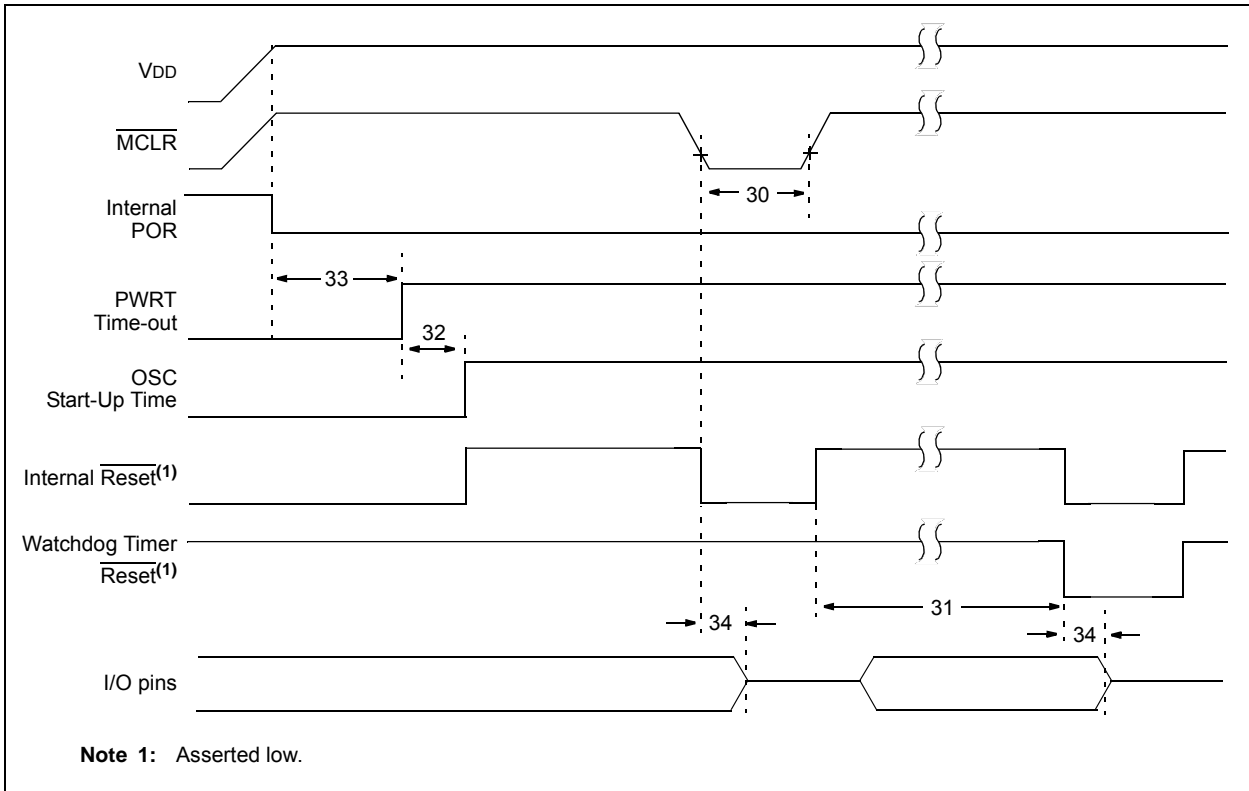
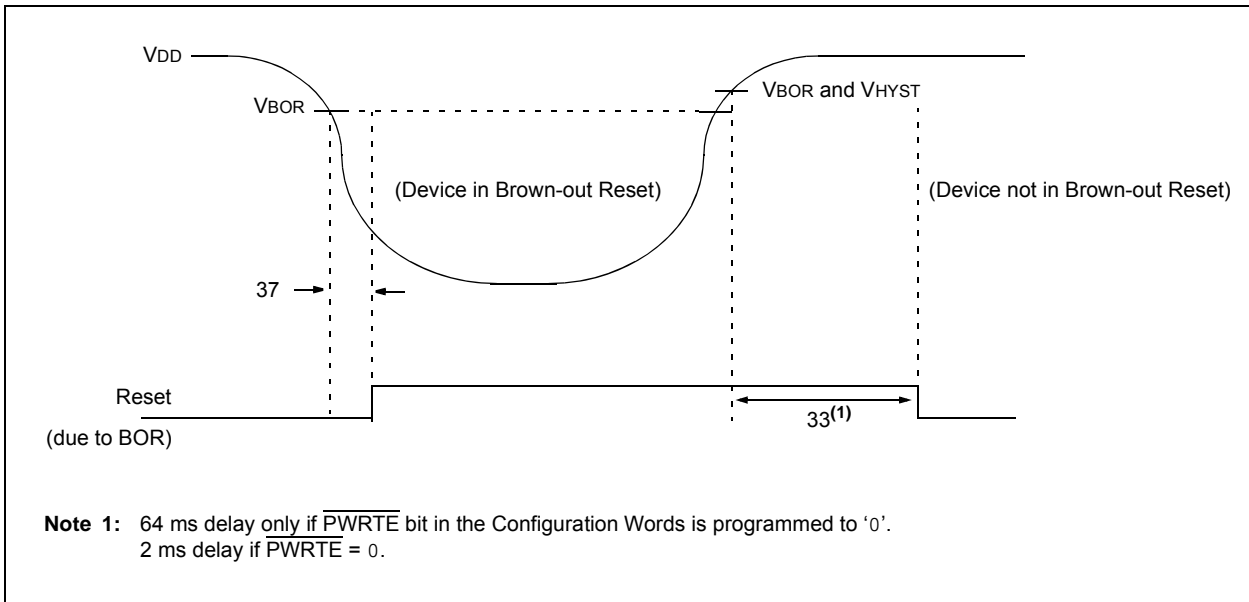


FIGURE 25-9: BROWN-OUT RESET TIMING AND CHARACTERISTICS



PIC16(L)F1526/7

TABLE 25-4: RESET, WATCHDOG TIMER, OSCILLATOR START-UP TIMER, POWER-UP TIMER AND BROWN-OUT RESET PARAMETERS

| Standard Operating Conditions (unless otherwise stated) | | | | | | | |
|--|--------|--|------|------|------|---------------|---|
| Operating Temperature $-40^{\circ}\text{C} \leq T_A \leq +125^{\circ}\text{C}$ | | | | | | | |
| Param No. | Sym. | Characteristic | Min. | Typ† | Max. | Units | Conditions |
| 30 | TMCL | MCLR Pulse Width (low) | 2 | — | — | μs | |
| 30A | TMCLR | | — | — | — | — | |
| 31 | TWDTLP | Low-Power Watchdog Timer Time-out Period | 10 | 16 | 27 | ms | $V_{DD} = 3.3\text{V}-5\text{V}$, 1:512 prescaler used |
| 32 | TOST | Oscillator Start-up Timer Period ⁽¹⁾ | — | 1024 | — | Tosc | (Note 3) |
| 33* | TPWRT | Power-up Timer Period, $\overline{\text{PWRTE}} = 0$ | 40 | 65 | 140 | ms | |
| 34* | TIOZ | I/O high impedance from MCLR Low or Watchdog Timer Reset | — | — | 2.0 | μs | |
| 35 | VBOR | Brown-out Reset Voltage ⁽²⁾ | 2.55 | 2.70 | 2.85 | V | BORV = 0, PIC16(L)F1526/7 |
| | | | 2.35 | 2.45 | 2.58 | V | BORV = 1, PIC16F1526/7 |
| | | | 1.80 | 1.90 | 2.00 | V | BORV = 1, PIC16LF1526/7 |
| 36* | VHYST | Brown-out Reset Hysteresis | 0 | 25 | 60 | mV | -40°C to $+85^{\circ}\text{C}$ |
| 37* | TBORDC | Brown-out Reset DC Response Time | 1 | 3 | 35 | μs | $V_{DD} \leq V_{BOR}$ |
| 38 | VLPBOR | Low-Power Brown-out Reset Voltage | 1.8 | 2.1 | 2.5 | V | LPBOR = 1 |

* These parameters are characterized but not tested.

† Data in "Typ" column is at 3.0V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

Note 1: By design, the Oscillator Start-up Timer (OST) counts the first 1024 cycles, independent of frequency.

2: To ensure these voltage tolerances, V_{DD} and V_{SS} must be capacitively decoupled as close to the device as possible. 0.1 μF and 0.01 μF values in parallel are recommended.

FIGURE 25-10: TIMER0 AND TIMER1 EXTERNAL CLOCK TIMINGS

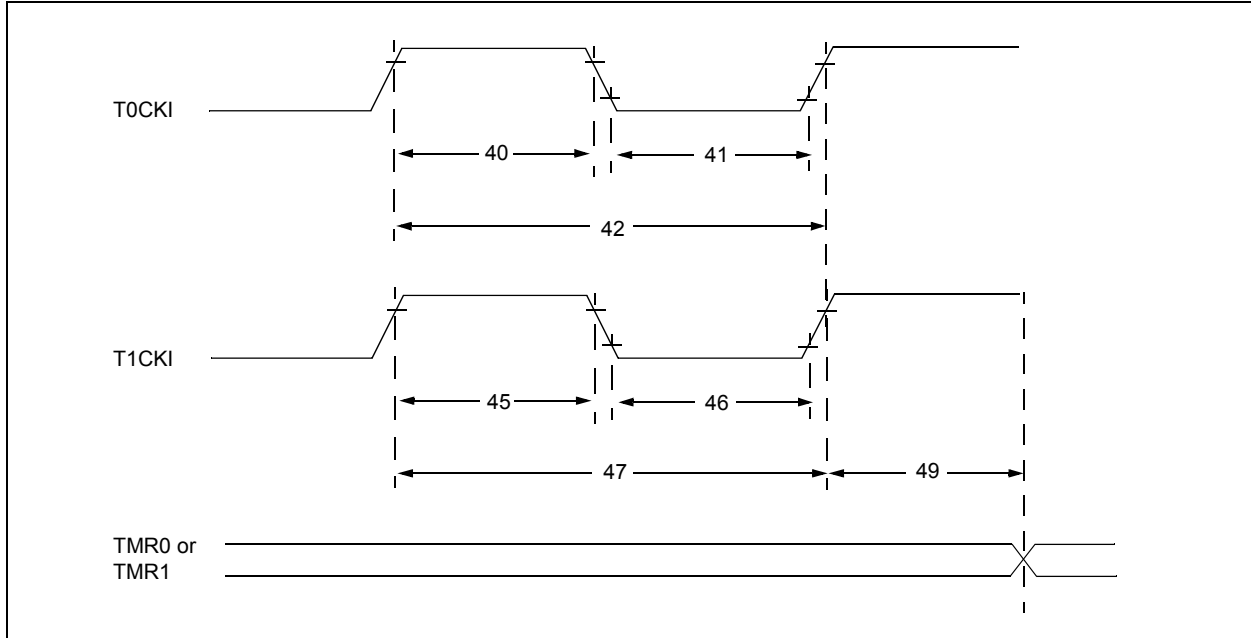


TABLE 25-5: TIMER0 AND TIMER1 EXTERNAL CLOCK REQUIREMENTS

| Standard Operating Conditions (unless otherwise stated) | | | | | | | | |
|--|-----------|--|-----------------------------|--|--------|-------------|-------|---------------------|
| Operating Temperature $-40^{\circ}\text{C} \leq T_A \leq +125^{\circ}\text{C}$ | | | | | | | | |
| Param No. | Sym. | Characteristic | | Min. | Typ† | Max. | Units | Conditions |
| 40* | Tt0H | T0CKI High Pulse Width | No Prescaler | $0.5 T_{CY} + 20$ | — | — | ns | |
| | | | With Prescaler | 10 | — | — | ns | |
| 41* | Tt0L | T0CKI Low Pulse Width | No Prescaler | $0.5 T_{CY} + 20$ | — | — | ns | |
| | | | With Prescaler | 10 | — | — | ns | |
| 42* | Tt0P | T0CKI Period | | Greater of: 20 or $\frac{T_{CY} + 40}{N}$ | — | — | ns | N = prescale value |
| 45* | Tt1H | T1CKI High Time | Synchronous, No Prescaler | $0.5 T_{CY} + 20$ | — | — | ns | |
| | | | Synchronous, with Prescaler | 15 | — | — | ns | |
| | | | Asynchronous | 30 | — | — | ns | |
| 46* | Tt1L | T1CKI Low Time | Synchronous, No Prescaler | $0.5 T_{CY} + 20$ | — | — | ns | |
| | | | Synchronous, with Prescaler | 15 | — | — | ns | |
| | | | Asynchronous | 30 | — | — | ns | |
| 47* | Tt1P | T1CKI Input Period | Synchronous | Greater of: 30 or $\frac{T_{CY} + 40}{N}$ | — | — | ns | N = prescale value |
| | | | Asynchronous | 60 | — | — | ns | |
| 48 | Ft1 | Timer1 Oscillator Input Frequency Range (oscillator enabled by setting bit SOSCEN) | | 32.4 | 32.768 | 33.1 | kHz | |
| 49* | TCKEZTMR1 | Delay from External Clock Edge to Timer Increment | | $2 T_{osc}$ | — | $7 T_{osc}$ | — | Timers in Sync mode |

* These parameters are characterized but not tested.

† Data in "Typ" column is at 3.0V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

PIC16(L)F1526/7

FIGURE 25-11: CAPTURE/COMPARE/PWM TIMINGS (CCP)

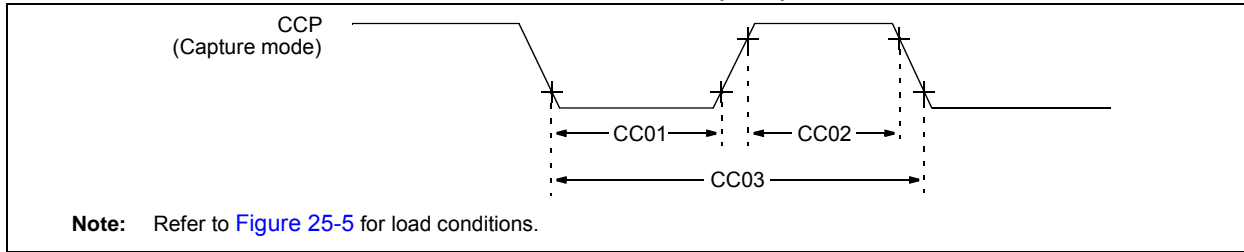


TABLE 25-6: CAPTURE/COMPARE/PWM REQUIREMENTS (CCP)

| Standard Operating Conditions (unless otherwise stated) | | | | | | | | |
|--|------|---------------------|----------------|--------------------------|------|------|-------|--------------------|
| Operating Temperature $-40^{\circ}\text{C} \leq T_A \leq +125^{\circ}\text{C}$ | | | | | | | | |
| Param No. | Sym. | Characteristic | | Min. | Typ† | Max. | Units | Conditions |
| CC01* | TccL | CCP Input Low Time | No Prescaler | $0.5T_{CY} + 20$ | — | — | ns | |
| | | | With Prescaler | 20 | — | — | ns | |
| CC02* | TccH | CCP Input High Time | No Prescaler | $0.5T_{CY} + 20$ | — | — | ns | |
| | | | With Prescaler | 20 | — | — | ns | |
| CC03* | TccP | CCP Input Period | | $\frac{3T_{CY} + 40}{N}$ | — | — | ns | N = prescale value |

* These parameters are characterized but not tested.

† Data in "Typ" column is at 3.0V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

TABLE 25-7: ANALOG-TO-DIGITAL CONVERTER (ADC) CHARACTERISTICS^(1,2,3)

| Standard Operating Conditions (unless otherwise stated) | | | | | | | |
|---|------|--|------|------|------|-------|---|
| Operating Temperature Tested at 25°C | | | | | | | |
| Param No. | Sym. | Characteristic | Min. | Typ† | Max. | Units | Conditions |
| AD01 | NR | Resolution | — | — | 10 | bit | |
| AD02 | EIL | Integral Error | — | ±1 | ±1.7 | LSb | VREF = 3.0V |
| AD03 | EDL | Differential Error | — | ±1 | ±1 | LSb | No missing codes VREF = 3.0V |
| AD04 | EOFF | Offset Error | — | ±1 | ±2.5 | LSb | VREF = 3.0V |
| AD05 | EGN | Gain Error | — | ±1 | ±2.0 | LSb | VREF = 3.0V |
| AD06 | VREF | Reference Voltage ⁽⁴⁾ | 1.8 | — | VDD | V | VREF = (VREF+ minus VREF-) |
| AD07 | VAIN | Full-Scale Range | VSS | — | VREF | V | |
| AD08 | ZAIN | Recommended Impedance of Analog Voltage Source | — | — | 10 | kΩ | Can go higher if external 0.01μF capacitor is present on input pin. |

† Data in “Typ” column is at 3.0V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

- Note 1:** Total Absolute Error includes integral, differential, offset and gain errors.
Note 2: The ADC conversion result never decreases with an increase in the input voltage and has no missing codes.
Note 3: ADC VREF is from external VREF, VDD pin or FVR, whichever is selected as reference input.
Note 4: ADC Reference Voltage (Ref+) is the selected reference input, VREF+ pin, VDD pin or the FVR Buffer1. When the FVR is selected as the reference input, the FVR Buffer1 output selection must be 2.048V or 4.096V (ADFVR<1:0> = 1x).

TABLE 25-8: ADC CONVERSION REQUIREMENTS

| Standard Operating Conditions (unless otherwise stated) | | | | | | | |
|---|------|---|------|--|------|-------|--|
| Operating Temperature -40°C ≤ TA ≤ +125°C | | | | | | | |
| Param No. | Sym. | Characteristic | Min. | Typ† | Max. | Units | Conditions |
| AD130* | TAD | ADC Clock Period | 1.0 | — | 9.0 | μs | FOSC-based |
| | | ADC Internal RC Oscillator Period | 1.0 | 2.0 | 6.0 | μs | ADCS<2:0> = x11 (ADC FRC mode) |
| AD131 | TcNV | Conversion Time (not including Acquisition Time) ⁽¹⁾ | — | 11 | — | TAD | Set GO/DONE bit to conversion complete |
| AD132* | TACQ | Acquisition Time | — | 5.0 | — | μs | |
| AD133 | THCD | Holding Capacitor Disconnect | — | 0.5*TAD + 40 ns (0.5*TAD + 40 ns) to | — | | ADCS<2:0> ≠ x11 (FOSC-based) |
| | | | — | (1.5*TAD + 40 ns) | — | | ADCS<2:0> = x11 (ADC FRC mode) |

* These parameters are characterized but not tested.

† Data in “Typ” column is at 3.0V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

- Note 1:** The ADRES register may be read on the following TcY cycle.

PIC16(L)F1526/7

FIGURE 25-12: ADC CONVERSION TIMING (NORMAL MODE)

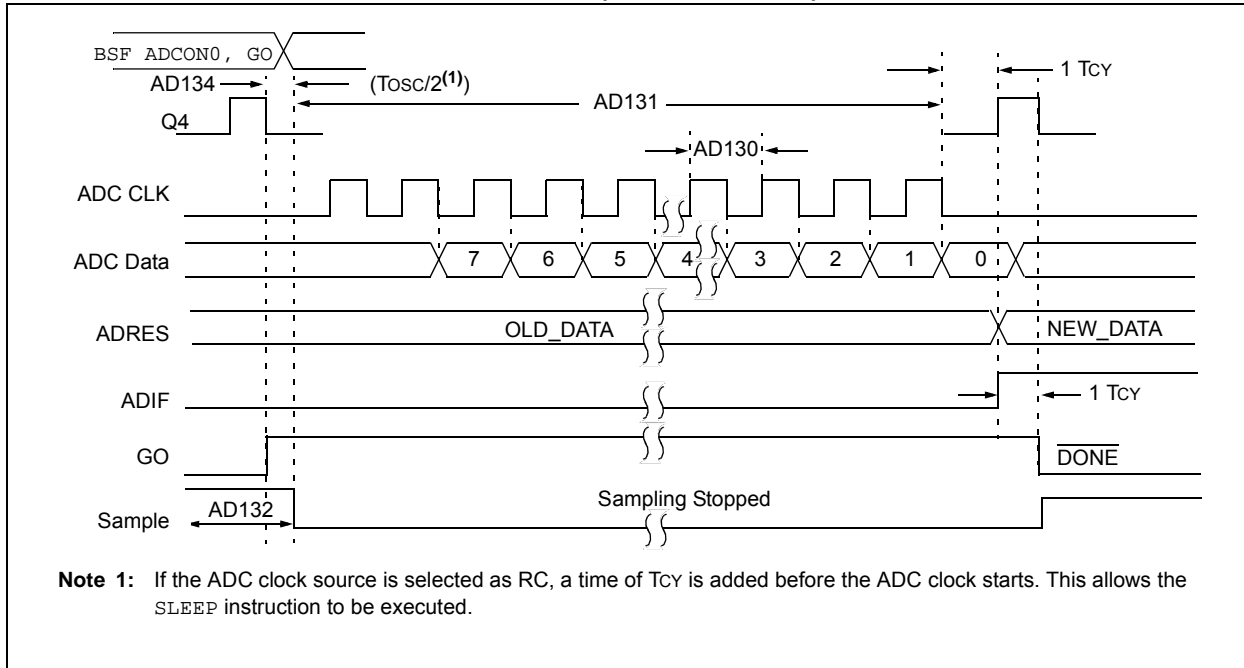


FIGURE 25-13: ADC CONVERSION TIMING (SLEEP MODE)

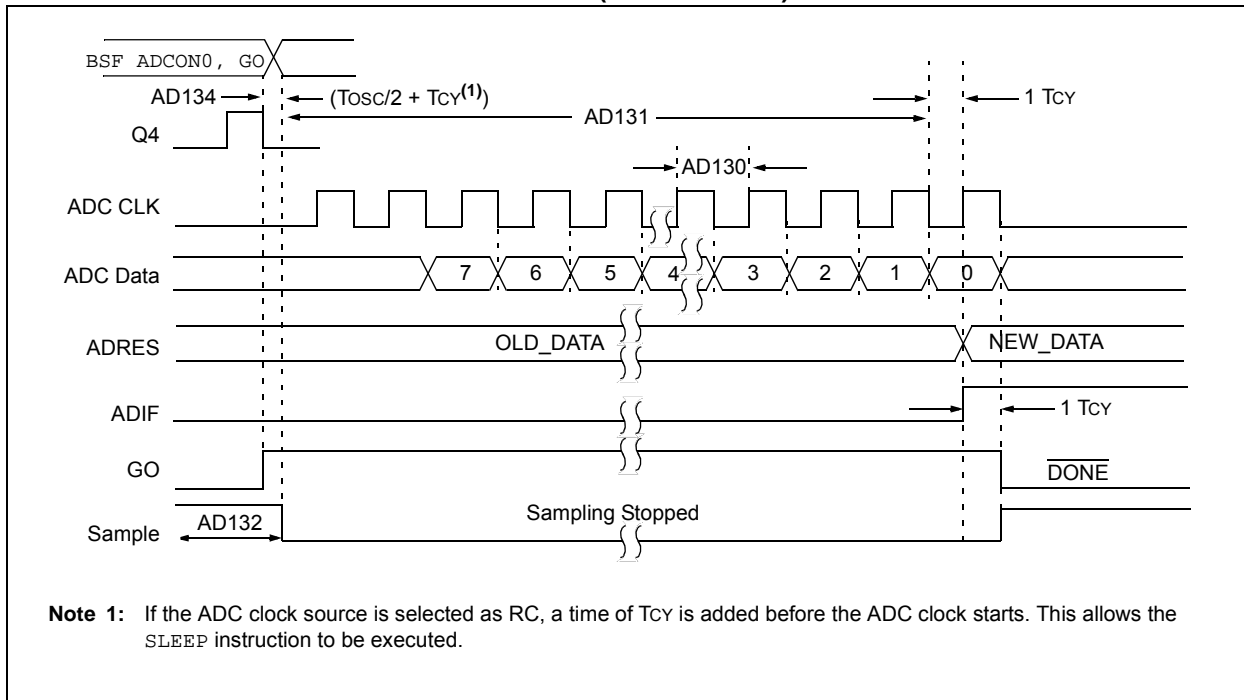


TABLE 25-9: LOW DROPOUT (LDO) REGULATOR CHARACTERISTICS

| Standard Operating Conditions (unless otherwise stated) | | | | | | | |
|--|------|------------------------|------|------|------|---------------|------------|
| Operating Temperature $-40^{\circ}\text{C} \leq T_A \leq +125^{\circ}\text{C}$ | | | | | | | |
| Param No. | Sym. | Characteristic | Min. | Typ† | Max. | Units | Conditions |
| LDO01 | | LDO Regulation Voltage | — | 3.0 | — | V | |
| LDO02 | | LDO External Capacitor | 0.1 | — | 1 | μF | |

† Data in "Typ" column is at 3.0V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

FIGURE 25-14: USART SYNCHRONOUS TRANSMISSION (MASTER/SLAVE) TIMING

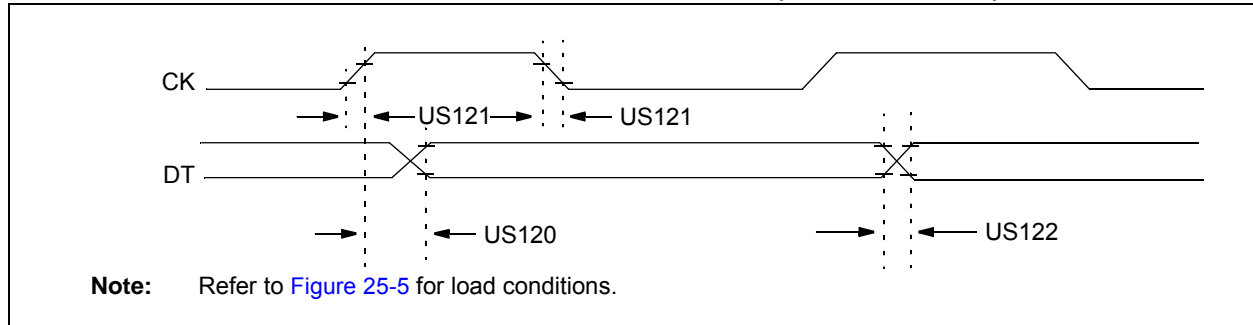


TABLE 25-10: USART SYNCHRONOUS TRANSMISSION REQUIREMENTS

| Standard Operating Conditions (unless otherwise stated) | | | | | | | |
|--|----------|--|----------|------|-------|------------|--|
| Operating Temperature $-40^{\circ}\text{C} \leq T_A \leq +125^{\circ}\text{C}$ | | | | | | | |
| Param. No. | Symbol | Characteristic | Min. | Max. | Units | Conditions | |
| US120 | TckH2DTV | SYNC XMIT (Master and Slave) Clock high to data-out valid | 3.0-5.5V | — | 80 | ns | |
| | | | 1.8-5.5V | — | 100 | ns | |
| US121 | TCKRF | Clock out rise time and fall time (Master mode) | 3.0-5.5V | — | 45 | ns | |
| | | | 1.8-5.5V | — | 50 | ns | |
| US122 | TDTRF | Data-out rise time and fall time | 3.0-5.5V | — | 45 | ns | |
| | | | 1.8-5.5V | — | 50 | ns | |

PIC16(L)F1526/7

FIGURE 25-15: USART SYNCHRONOUS RECEIVE (MASTER/SLAVE) TIMING

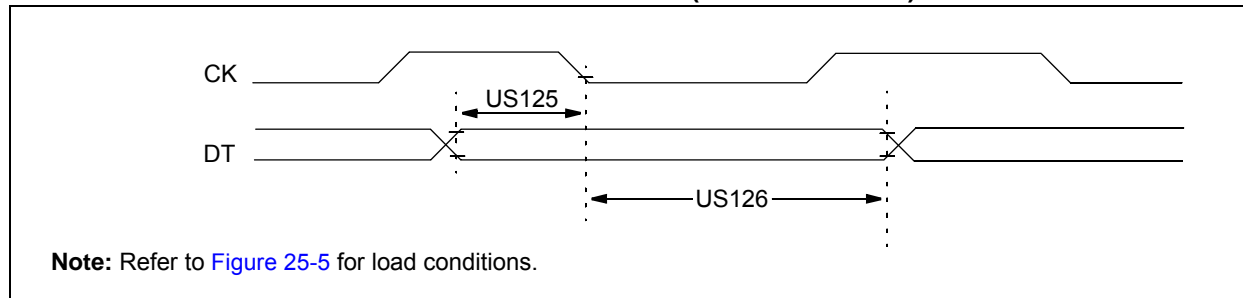


TABLE 25-11: USART SYNCHRONOUS RECEIVE REQUIREMENTS

| Standard Operating Conditions (unless otherwise stated) | | | | | | |
|--|----------|---|------|------|-------|------------|
| Operating Temperature $-40^{\circ}\text{C} \leq T_A \leq +125^{\circ}\text{C}$ | | | | | | |
| Param. No. | Symbol | Characteristic | Min. | Max. | Units | Conditions |
| US125 | TdtV2ckL | SYNC RCV (Master and Slave) Data-hold before CK ↓ (DT hold time) | 10 | — | ns | |
| US126 | TckL2DTL | Data-hold after CK ↓ (DT hold time) | 15 | — | ns | |

FIGURE 25-16: SPI MASTER MODE TIMING (CKE = 0, SMP = 0)

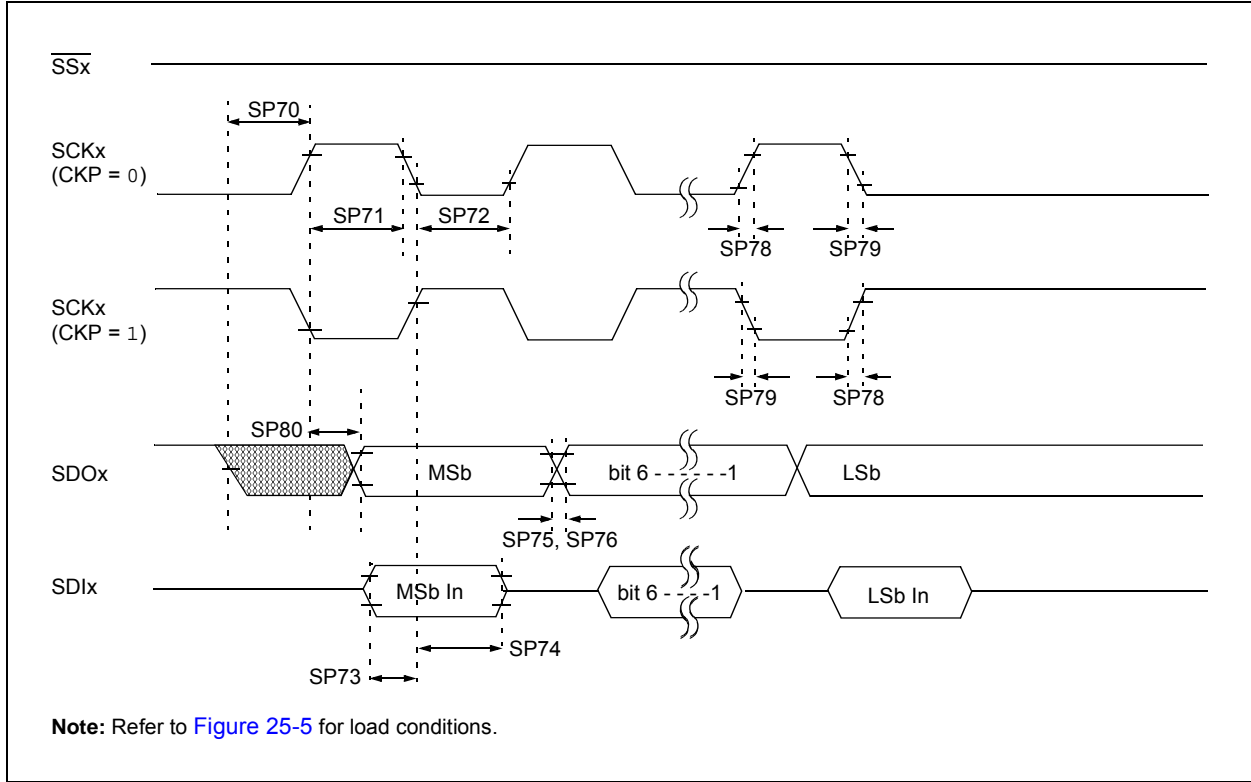
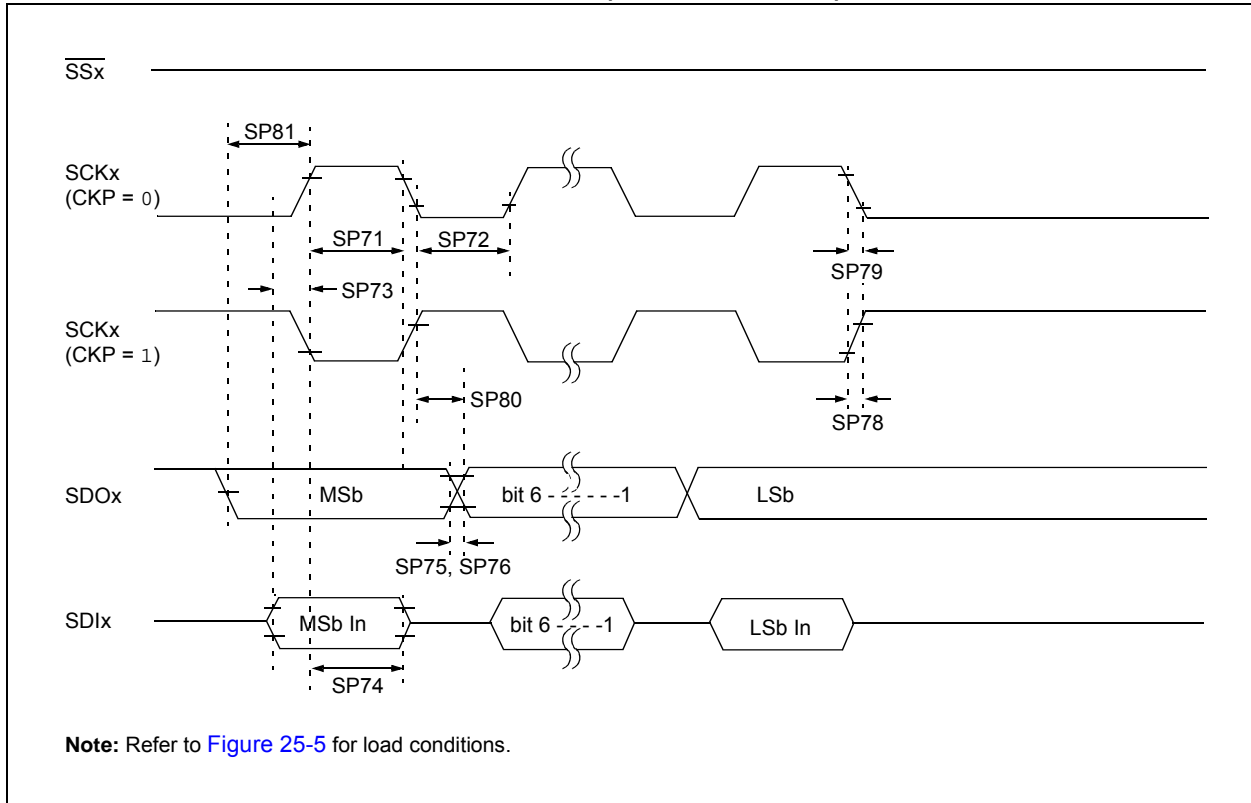


FIGURE 25-17: SPI MASTER MODE TIMING (CKE = 1, SMP = 1)



PIC16(L)F1526/7

FIGURE 25-18: SPI SLAVE MODE TIMING (CKE = 0)

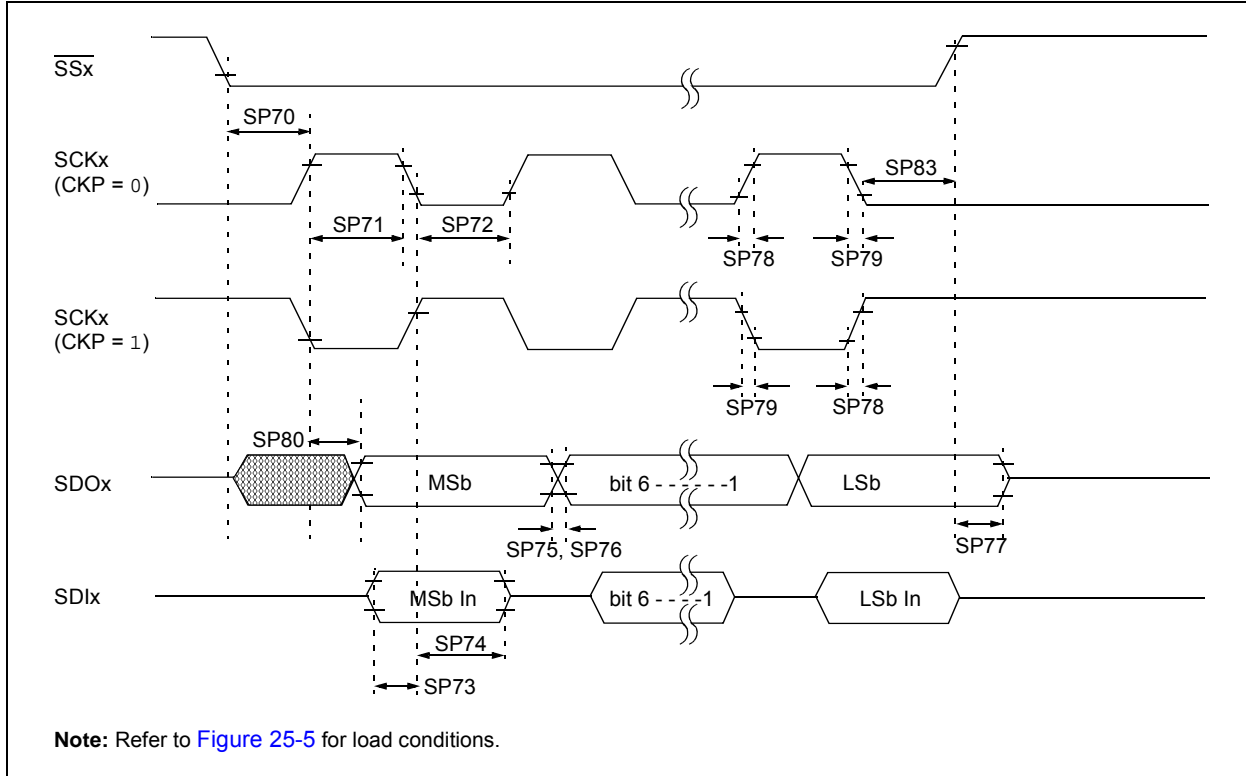


FIGURE 25-19: SPI SLAVE MODE TIMING (CKE = 1)

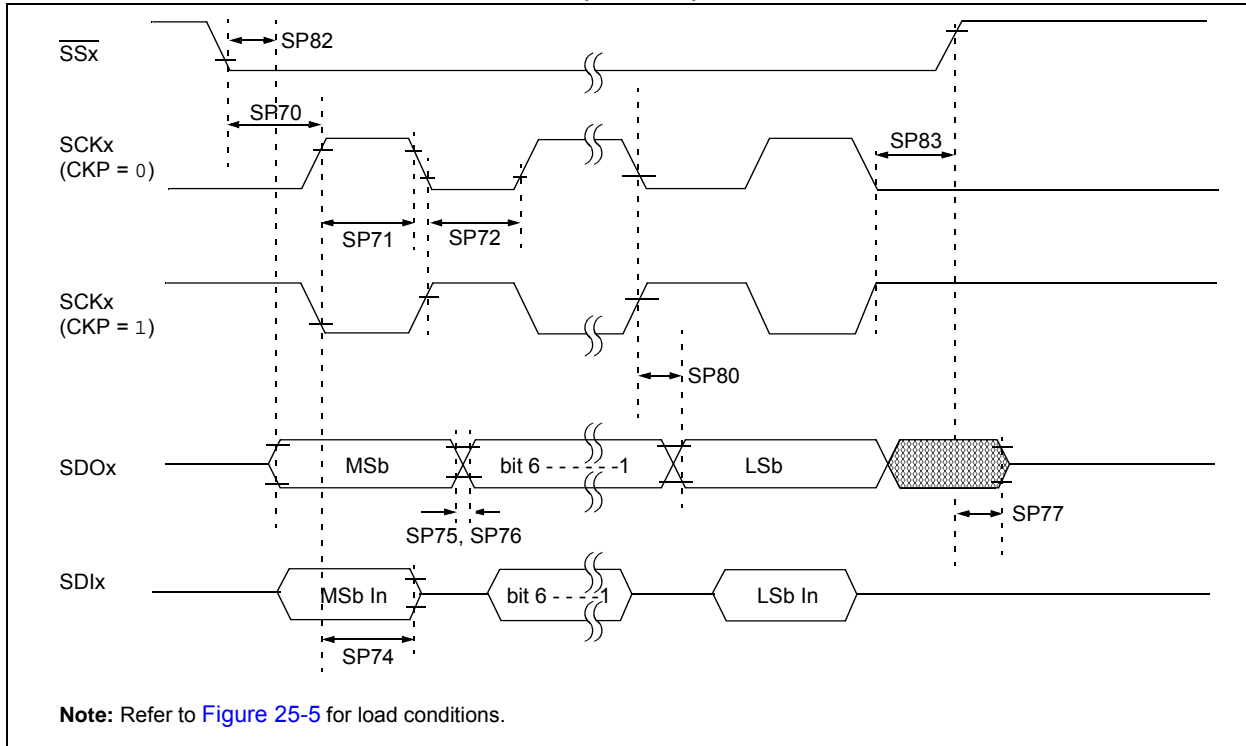


TABLE 25-12: SPI MODE REQUIREMENTS

| Standard Operating Conditions (unless otherwise stated) | | | | | | | |
|--|-----------------------|--|-------------|------|------|-------|------------|
| Operating Temperature $-40^{\circ}\text{C} \leq T_A \leq +125^{\circ}\text{C}$ | | | | | | | |
| Param No. | Symbol | Characteristic | Min. | Typ† | Max. | Units | Conditions |
| SP70* | TssL2sCH, TssL2sCL | $\overline{\text{SS}}\downarrow$ to SCK \downarrow or SCK \uparrow input | 2.25 Tcy | — | — | ns | |
| SP71* | Tsch | SCK input high time (Slave mode) | Tcy + 20 | — | — | ns | |
| SP72* | Tscl | SCK input low time (Slave mode) | Tcy + 20 | — | — | ns | |
| SP73* | TdIV2sCH, TdIV2sCL | Setup time of SDI data input to SCK edge | 100 | — | — | ns | |
| SP74* | Tsch2dIL, Tscl2dIL | Hold time of SDI data input to SCK edge | 100 | — | — | ns | |
| SP75* | TdoR | SDO data output rise time | 3.0-5.5V | — | 10 | 25 | ns |
| | | | 1.8-5.5V | — | 25 | 50 | ns |
| SP76* | TdoF | SDO data output fall time | — | 10 | 25 | ns | |
| SP77* | TssH2doZ | $\overline{\text{SS}}\uparrow$ to SDO output high-impedance | 10 | — | 50 | ns | |
| SP78* | Tscr | SCK output rise time (Master mode) | 3.0-5.5V | — | 10 | 25 | ns |
| | | | 1.8-5.5V | — | 25 | 50 | ns |
| SP79* | Tscf | SCK output fall time (Master mode) | — | 10 | 25 | ns | |
| SP80* | Tsch2doV, Tscl2doV | SDO data output valid after SCK edge | 3.0-5.5V | — | — | 50 | ns |
| | | | 1.8-5.5V | — | — | 145 | ns |
| SP81* | TdoV2sCH, TdoV2sCL | SDO data output setup to SCK edge | Tcy | — | — | ns | |
| SP83* | Tsch2ssH, Tscl2ssH | $\overline{\text{SS}}\uparrow$ after SCK edge | 1.5Tcy + 40 | — | — | ns | |

* These parameters are characterized but not tested.

† Data in "Typ" column is at 3.0V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

PIC16(L)F1526/7

FIGURE 25-20: I²C BUS START/STOP BITS TIMING

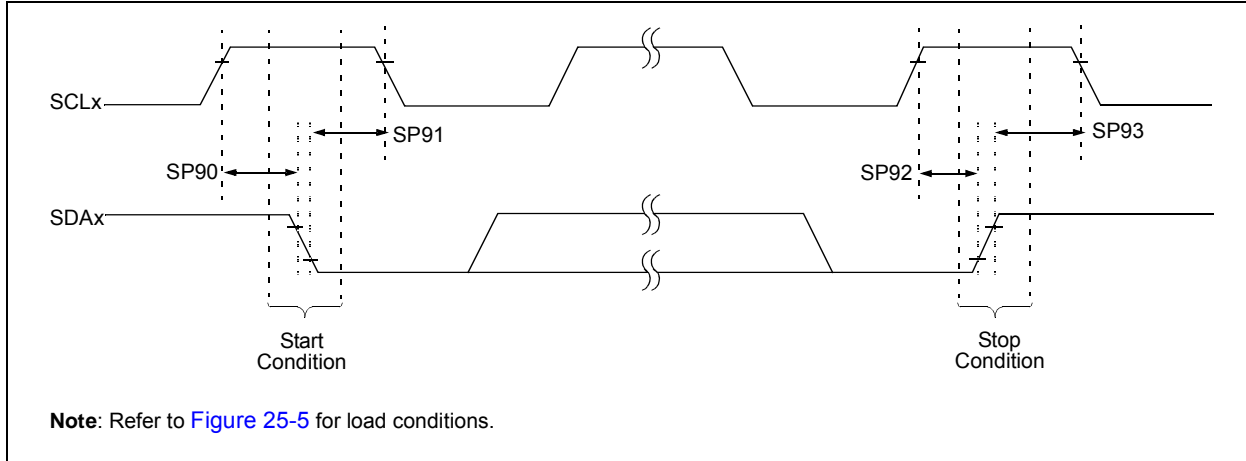


TABLE 25-13: I²C BUS START/STOP BITS REQUIREMENTS

| Standard Operating Conditions (unless otherwise stated) | | | | | | | | |
|---|---------|-----------------------|--------------|---------------------------------|------|------|-------|---|
| | | Operating Temperature | | -40°C ≤ T _A ≤ +125°C | | | | |
| Param. No. | Symbol | Characteristic | | Min. | Typ. | Max. | Units | Conditions |
| SP90* | TSU:STA | Start condition | 100 kHz mode | 4700 | — | — | ns | Only relevant for repeated Start condition |
| | | Setup time | 400 kHz mode | 600 | — | — | | |
| SP91* | THD:STA | Start condition | 100 kHz mode | 4000 | — | — | ns | After this period, the first clock pulse is generated |
| | | Hold time | 400 kHz mode | 600 | — | — | | |
| SP92* | TSU:STO | Stop condition | 100 kHz mode | 4700 | — | — | ns | |
| | | Setup time | 400 kHz mode | 600 | — | — | | |
| SP93 | THD:STO | Stop condition | 100 kHz mode | 4000 | — | — | ns | |
| | | Hold time | 400 kHz mode | 600 | — | — | | |

* These parameters are characterized but not tested.

FIGURE 25-21: I²C BUS DATA TIMING

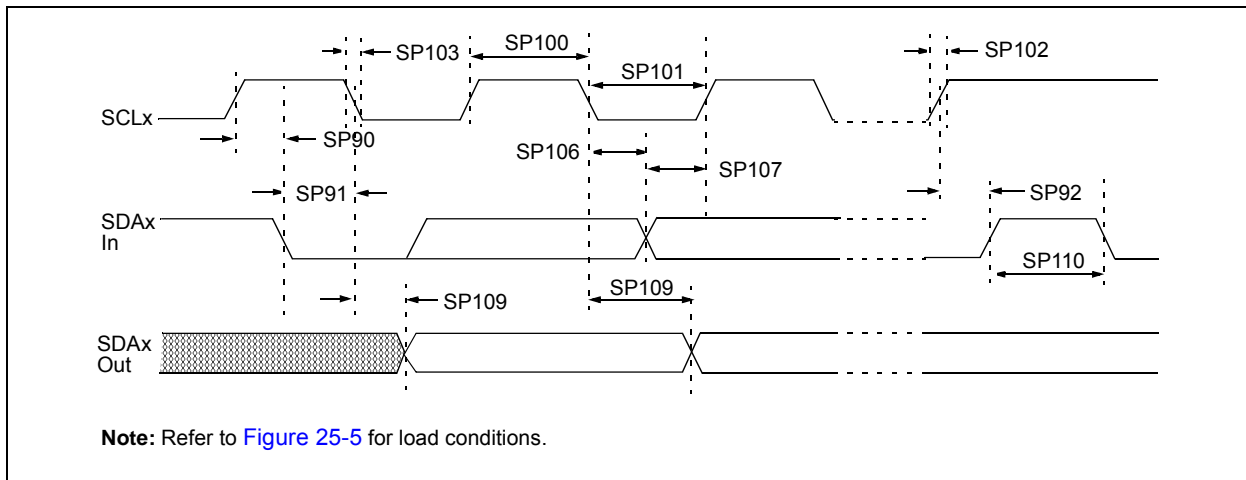


TABLE 25-14: I²C BUS DATA REQUIREMENTS

| Standard Operating Conditions (unless otherwise stated) | | | | | | | |
|--|---------------------|-------------------------|--------------------|------------------------|------|-------|---|
| Operating Temperature $-40^{\circ}\text{C} \leq T_A \leq +125^{\circ}\text{C}$ | | | | | | | |
| Param. No. | Symbol | Characteristic | | Min. | Max. | Units | Conditions |
| SP100* | T _{HIGH} | 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 |
| | | SSP module | 1.5T _{CY} | — | — | | |
| SP101* | T _{LOW} | 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 |
| | | SSP module | 1.5T _{CY} | — | — | | |
| SP102* | T _R | SDA and SCL rise time | 100 kHz mode | — | 1000 | ns | |
| | | | 400 kHz mode | 20 + 0.1C _B | 300 | ns | C _B is specified to be from 10-400 pF |
| SP103* | T _F | SDA and SCL fall time | 100 kHz mode | — | 250 | ns | |
| | | | 400 kHz mode | 20 + 0.1C _B | 250 | ns | C _B is specified to be from 10-400 pF |
| SP106* | T _{HD:DAT} | Data input hold time | 100 kHz mode | 0 | — | ns | |
| | | | 400 kHz mode | 0 | 0.9 | μs | |
| SP107* | T _{SU:DAT} | Data input setup time | 100 kHz mode | 250 | — | ns | (Note 2) |
| | | | 400 kHz mode | 100 | — | ns | |
| SP109* | T _{AA} | Output valid from clock | 100 kHz mode | — | 3500 | ns | (Note 1) |
| | | | 400 kHz mode | — | — | ns | |
| SP110* | T _{BUF} | 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 | |
| SP111 | C _B | Bus capacitive loading | | — | 400 | pF | |

* These parameters are characterized but not tested.

Note 1: As a transmitter, the device must provide this internal minimum delay time to bridge the undefined region (min. 300 ns) of the falling edge of SCL to avoid unintended generation of Start or Stop conditions.

2: A Fast mode (400 kHz) I²C bus device can be used in a Standard mode (100 kHz) I²C bus system, but the requirement T_{SU:DAT} ≥ 250 ns must then be met. This will automatically be the case if the device does not stretch the low period of the SCL signal. If such a device does stretch the low period of the SCL signal, it must output the next data bit to the SDA line T_R max. + T_{SU:DAT} = 1000 + 250 = 1250 ns (according to the Standard mode I²C bus specification), before the SCL line is released.

PIC16(L)F1526/7

26.0 DC AND AC CHARACTERISTICS GRAPHS AND CHARTS

The graphs and tables provided in this section are for **design guidance** and are **not tested**.

In some graphs or tables, the data presented are **outside specified operating range** (i.e., outside specified VDD range). This is for **information only** and devices are ensured to operate properly only within the specified range.

| |
|--|
| <p>Note: The graphs and tables provided following this note are a statistical summary based on a limited number of samples and are provided for informational purposes only. The performance characteristics listed herein are not tested or guaranteed. In some graphs or tables, the data presented may be outside the specified operating range (e.g., outside specified power supply range) and therefore, outside the warranted range.</p> |
|--|

“Typical” represents the mean of the distribution at 25°C. **“MAXIMUM”**, **“Max.”**, **“MINIMUM”** or **“Min.”** represents $(\text{mean} + 3\sigma)$ or $(\text{mean} - 3\sigma)$ respectively, where σ is a standard deviation, over each temperature range.

FIGURE 26-1: I_{DD}, LP OSCILLATOR, F_{osc} = 32 kHz, PIC16LF1526 ONLY

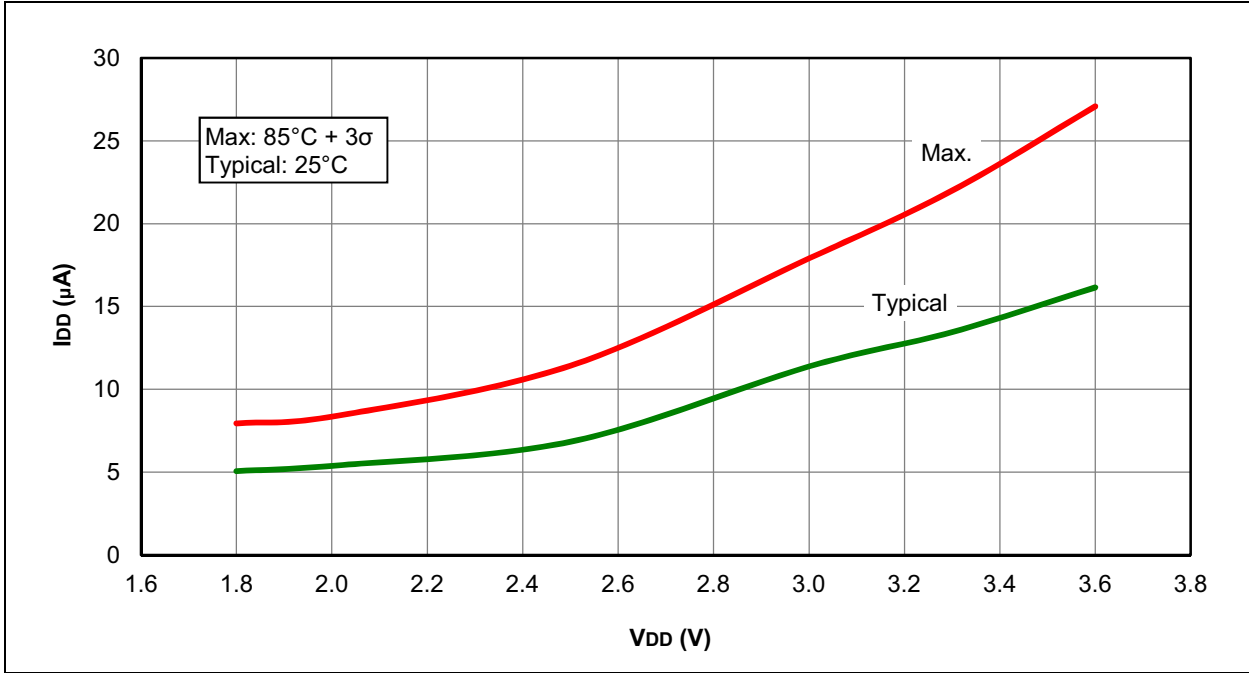
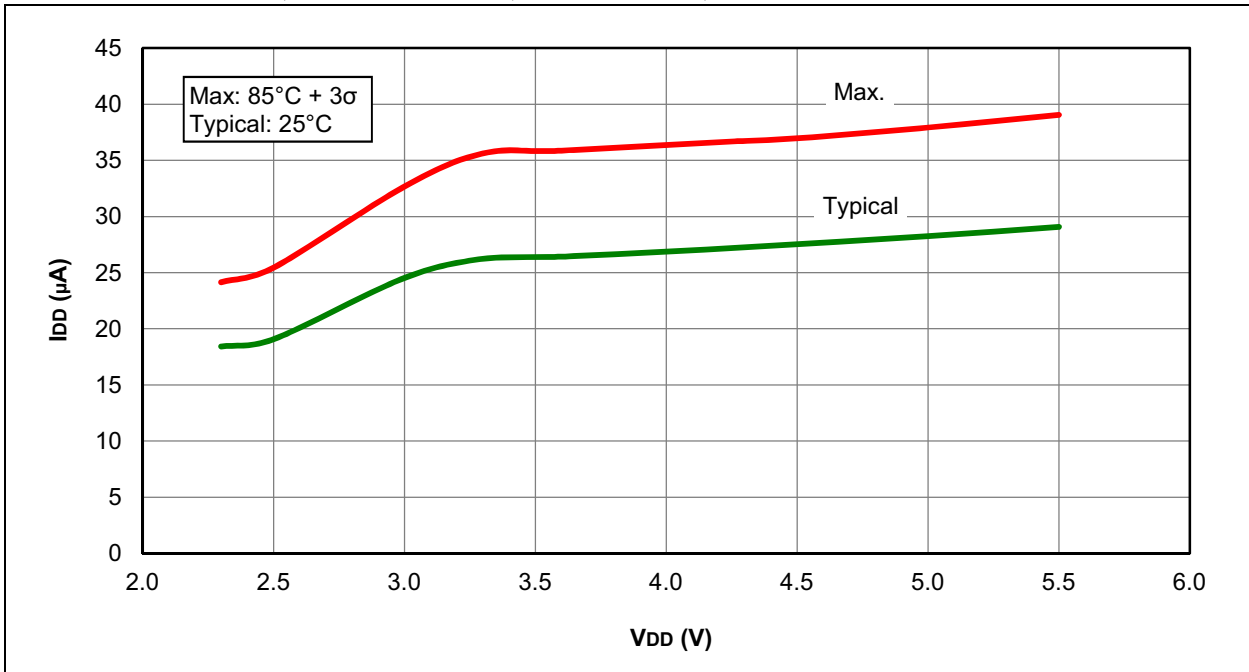


FIGURE 26-2: I_{DD}, LP OSCILLATOR, F_{osc} = 32 kHz, PIC16F1526/7 ONLY



PIC16(L)F1526/7

FIGURE 26-3: I_{DD} TYPICAL, XT AND EXTRC OSCILLATOR, PIC16LF1526 ONLY

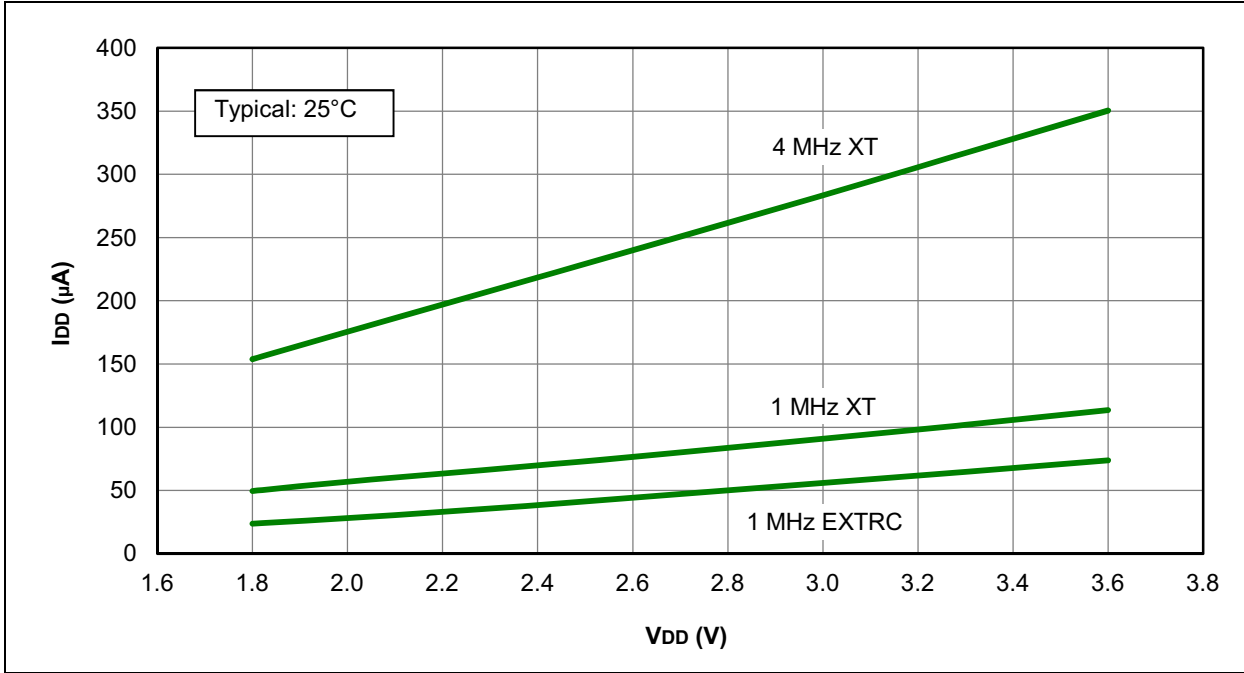


FIGURE 26-4: I_{DD} MAXIMUM, XT AND EXTRC OSCILLATOR, PIC16LF1526 ONLY

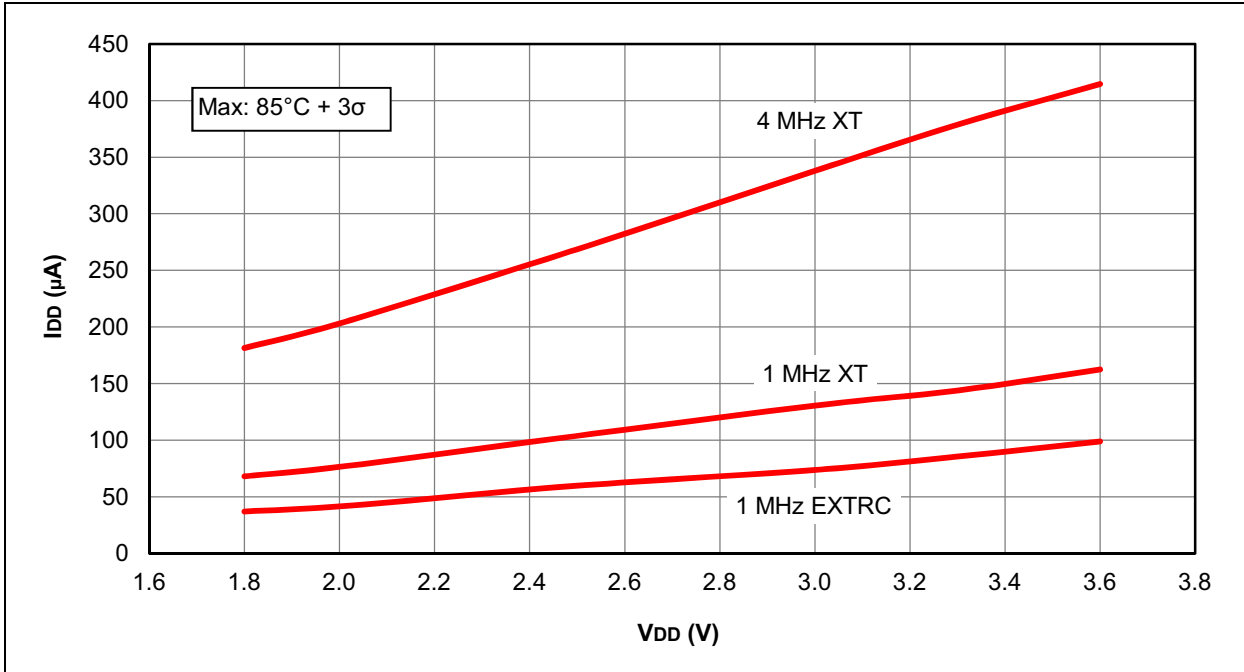


FIGURE 26-5: I_{DD} TYPICAL, XT AND EXTRC OSCILLATOR, PIC16F1526/7 ONLY

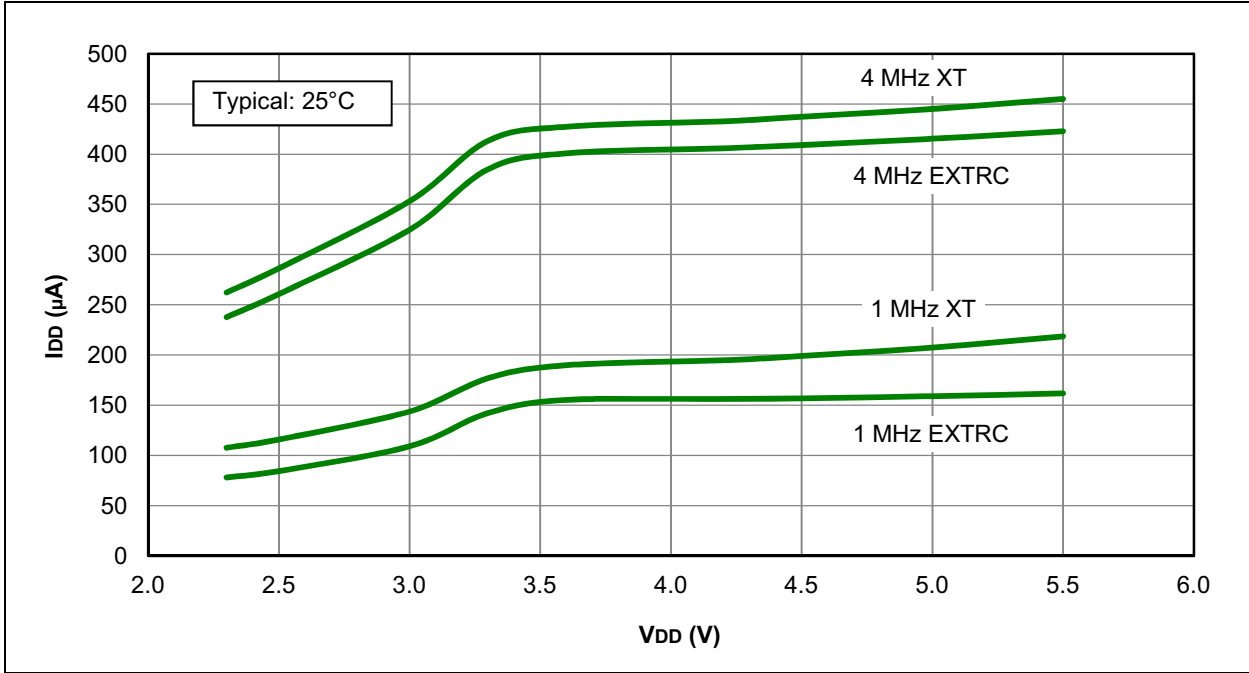
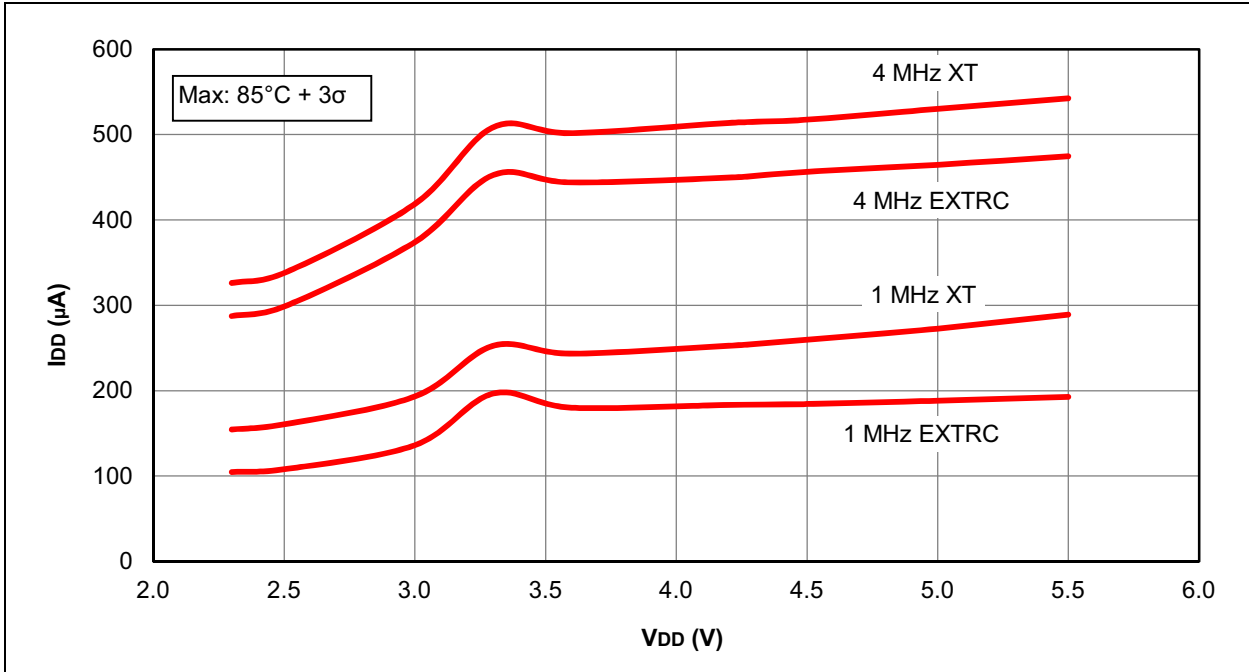


FIGURE 26-6: I_{DD} MAXIMUM, XT AND EXTRC OSCILLATOR, PIC16F1526/7 ONLY



PIC16(L)F1526/7

FIGURE 26-7: I_{DD} , EXTERNAL CLOCK (ECL), LOW-POWER MODE, $F_{osc} = 32$ kHz, PIC16LF1526 ONLY

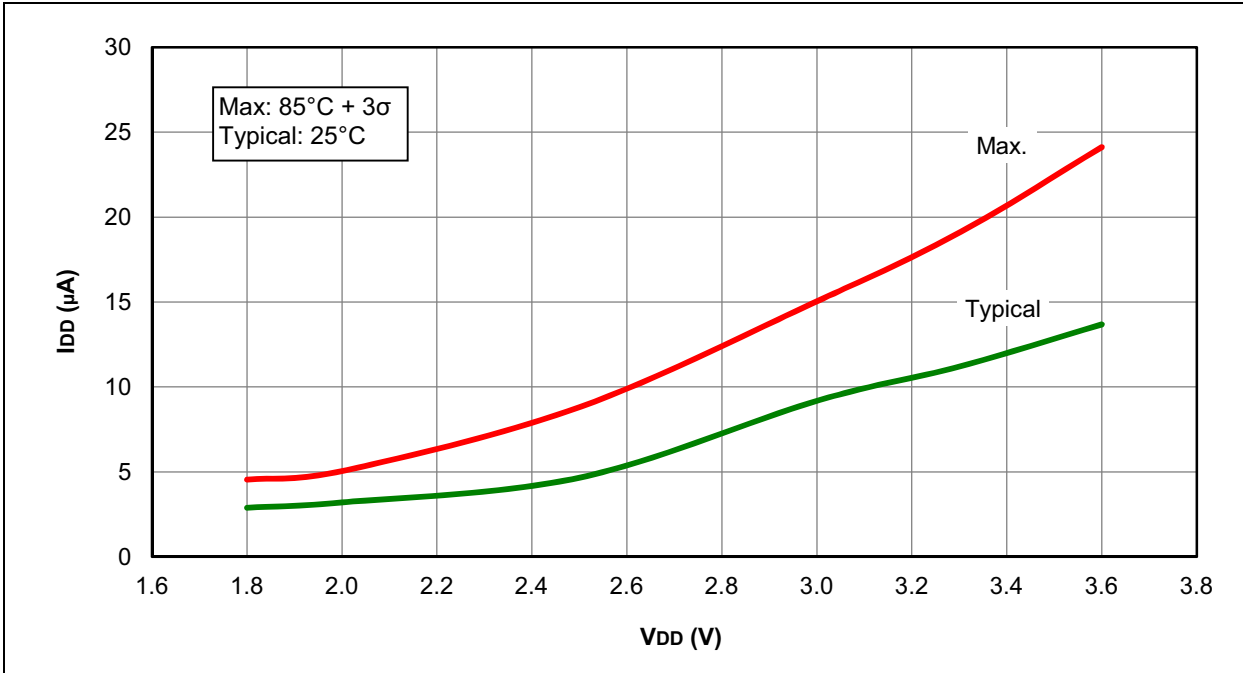


FIGURE 26-8: I_{DD} , EXTERNAL CLOCK (ECL), LOW-POWER MODE, $F_{osc} = 32$ kHz, PIC16F1526/7 ONLY

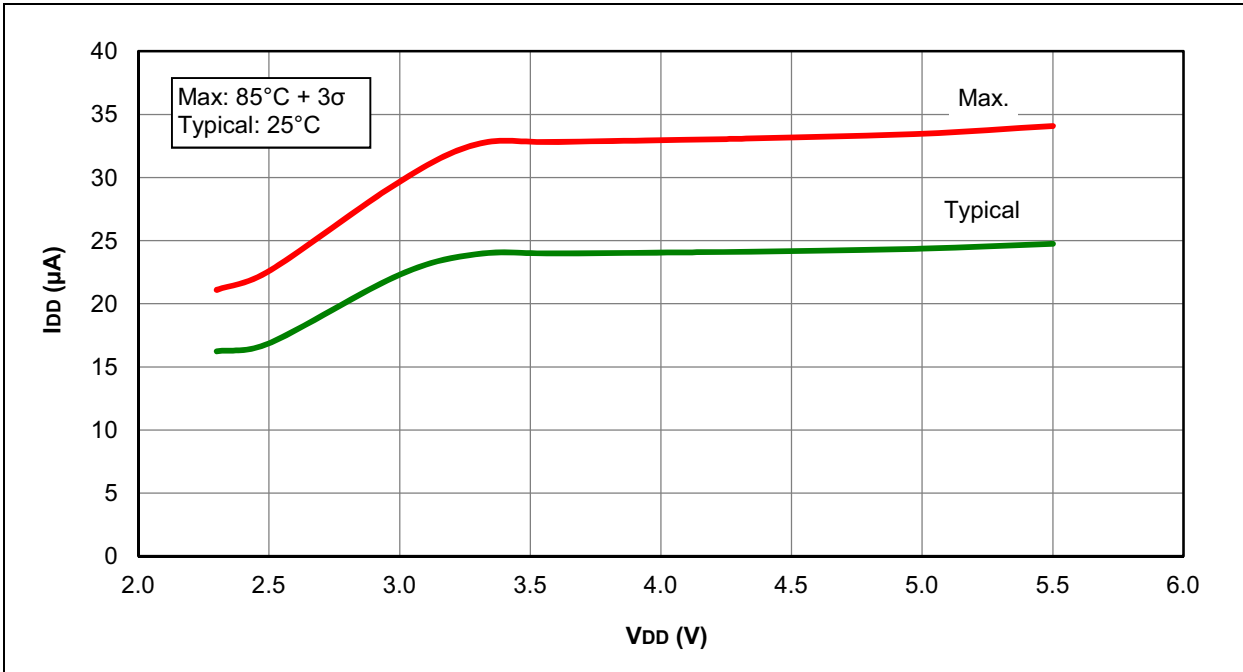


FIGURE 26-9: I_{DD}, EXTERNAL CLOCK (ECL), LOW-POWER MODE, F_{osc} = 500 kHz, PIC16LF1526 ONLY

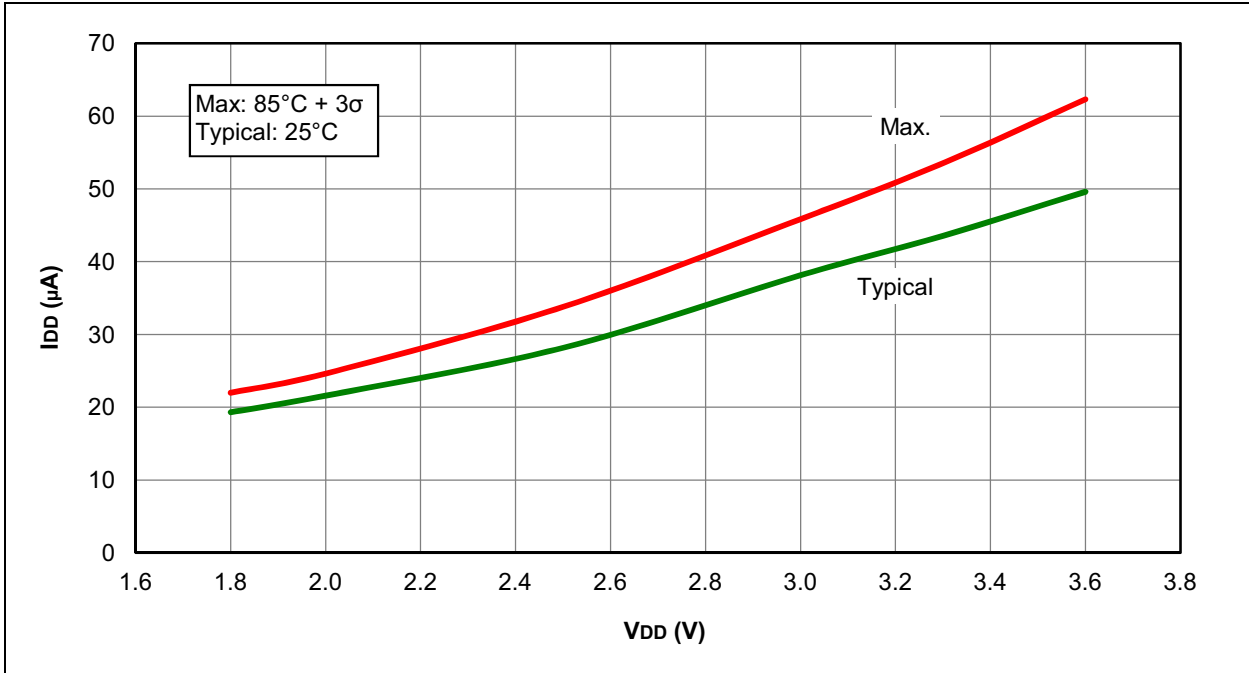
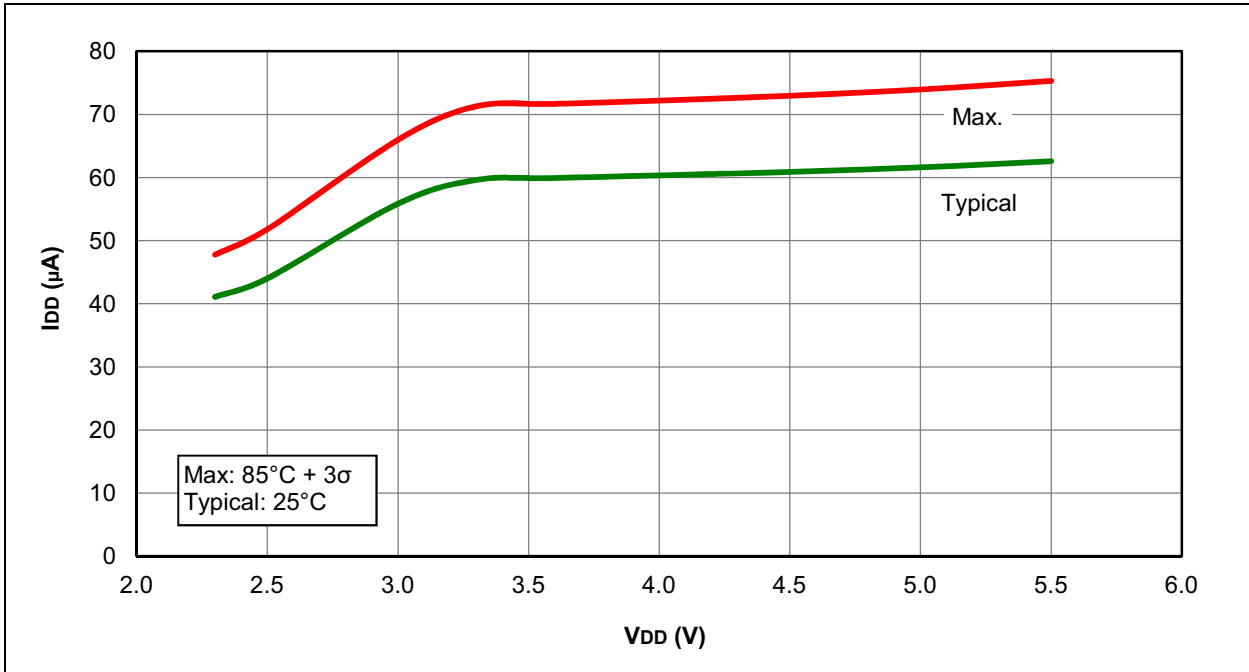


FIGURE 26-10: I_{DD}, EXTERNAL CLOCK (ECL), LOW-POWER MODE, F_{osc} = 500 kHz, PIC16F1526/7 ONLY



PIC16(L)F1526/7

FIGURE 26-11: I_{DD} TYPICAL, EXTERNAL CLOCK (ECM), MEDIUM-POWER MODE, PIC16LF1526 ONLY

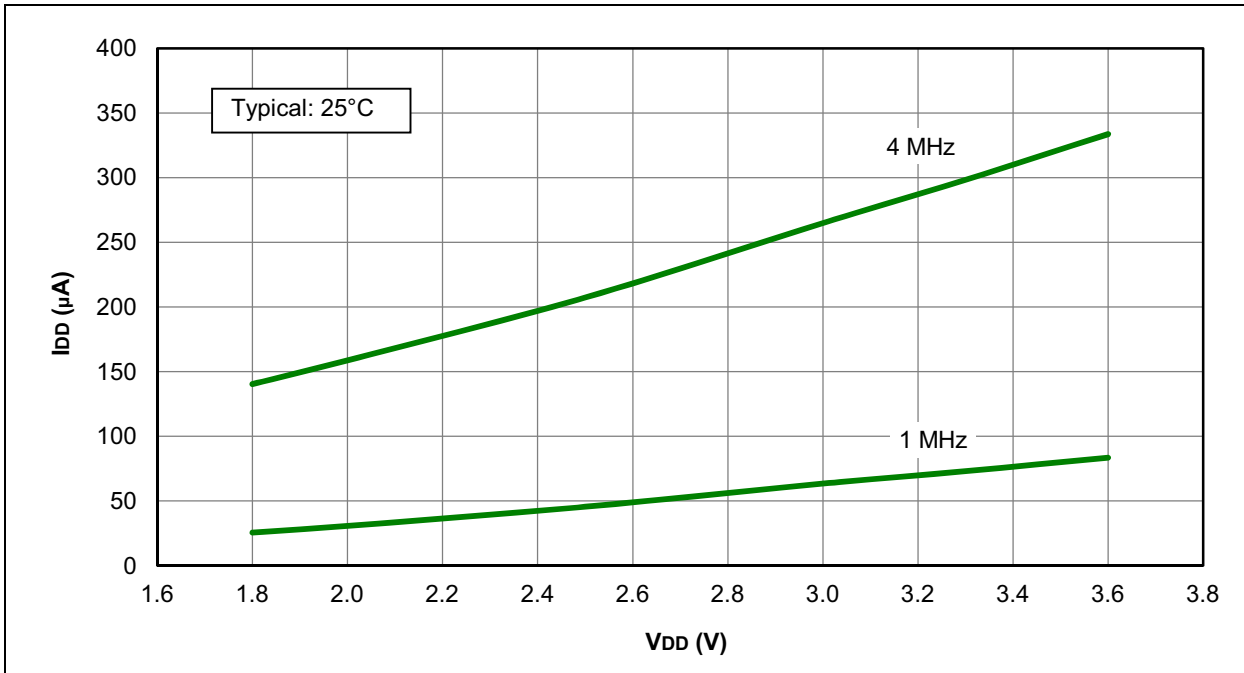


FIGURE 26-12: I_{DD} MAXIMUM, EXTERNAL CLOCK (ECM), MEDIUM-POWER MODE, PIC16LF1526 ONLY

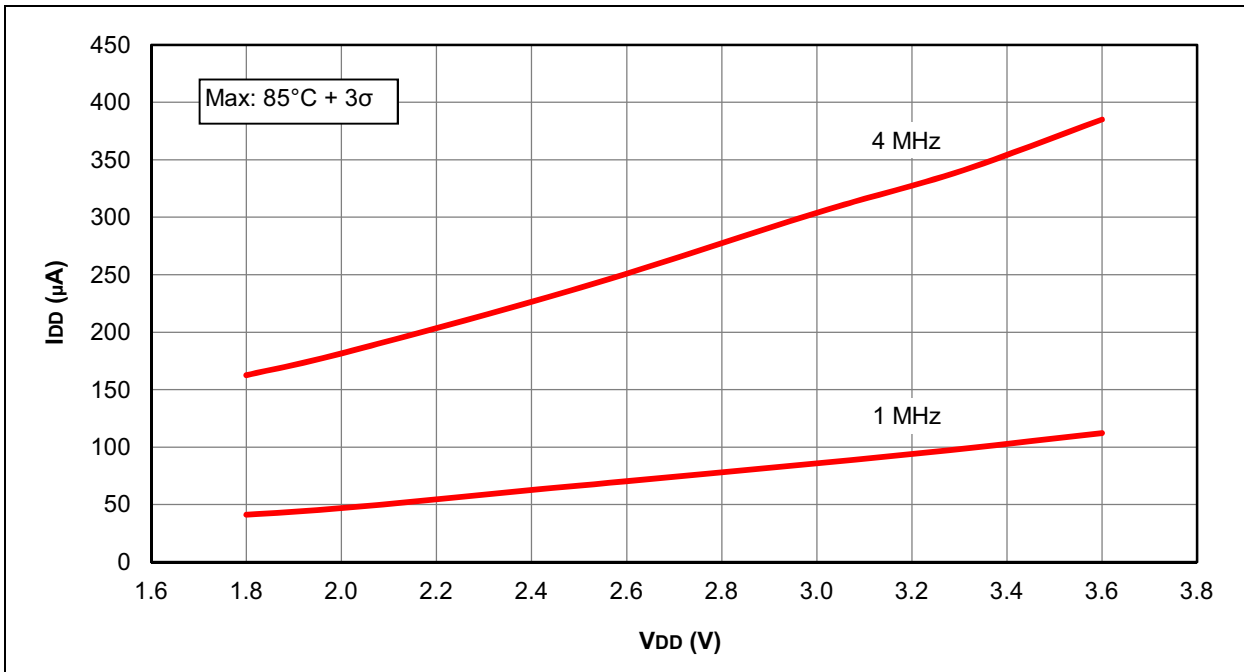


FIGURE 26-13: I_{DD} TYPICAL, EXTERNAL CLOCK (ECM), MEDIUM-POWER MODE, PIC16F1526/7 ONLY

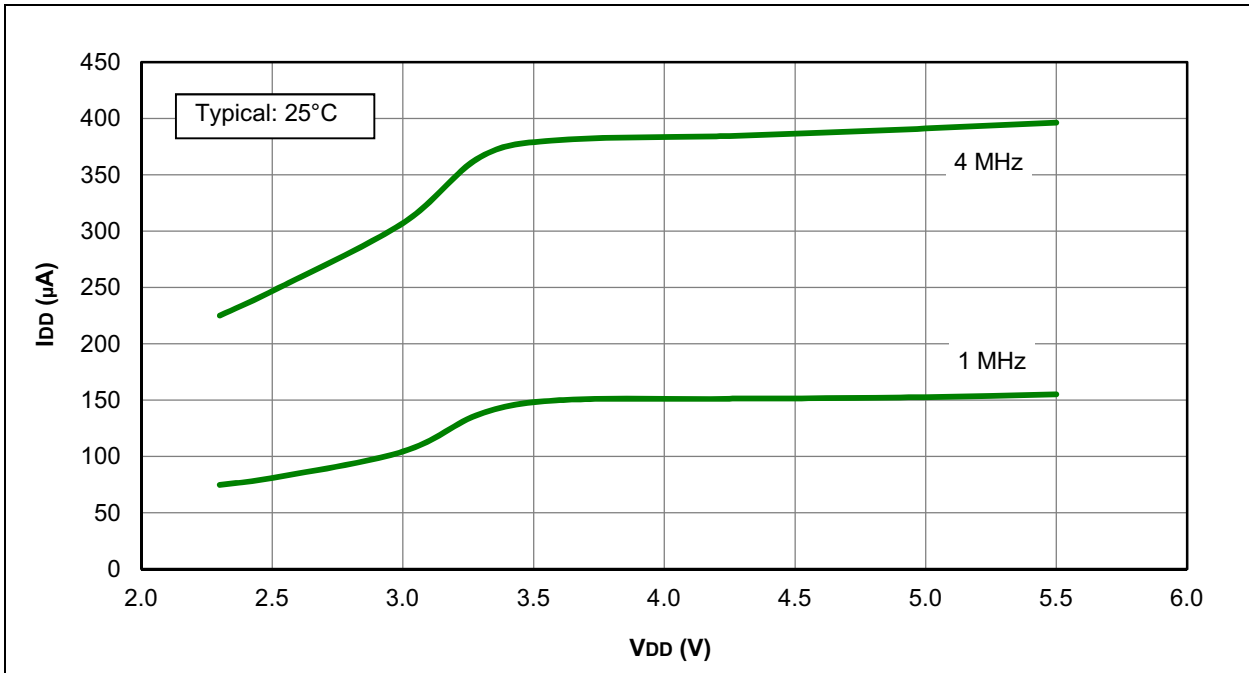
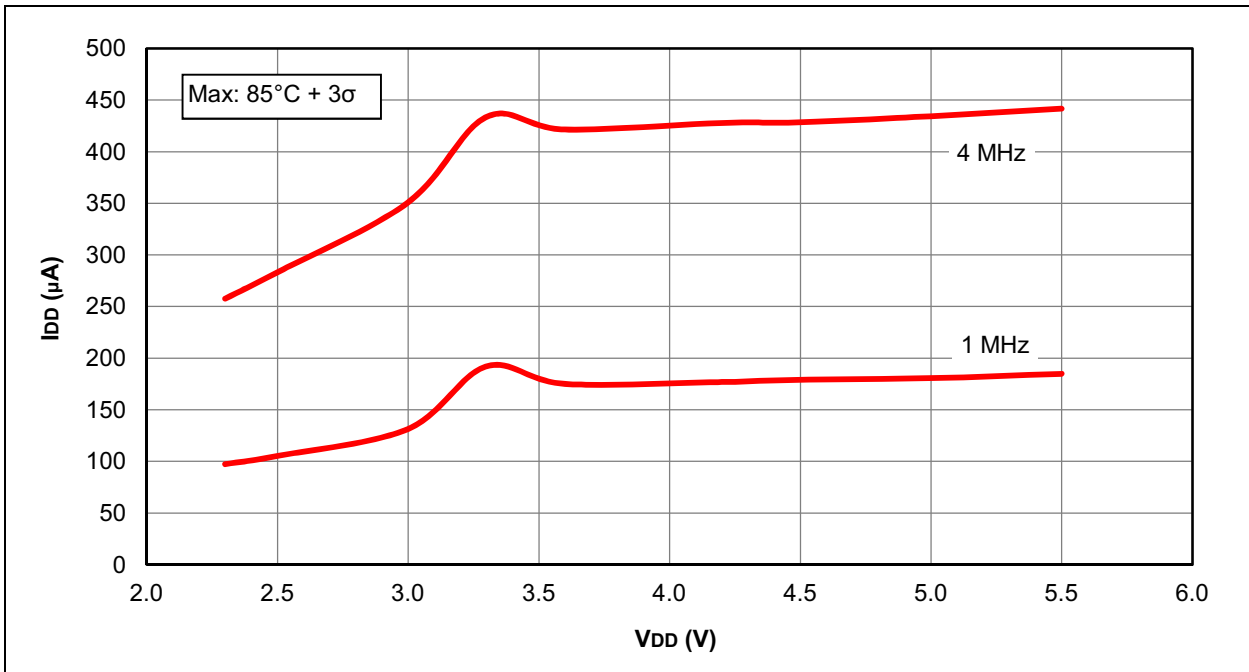


FIGURE 26-14: I_{DD} MAXIMUM, EXTERNAL CLOCK (ECM), MEDIUM-POWER MODE, PIC16F1526/7 ONLY



PIC16(L)F1526/7

FIGURE 26-15: I_{DD} TYPICAL, EXTERNAL CLOCK (ECH), HIGH-POWER MODE, PIC16LF1526 ONLY

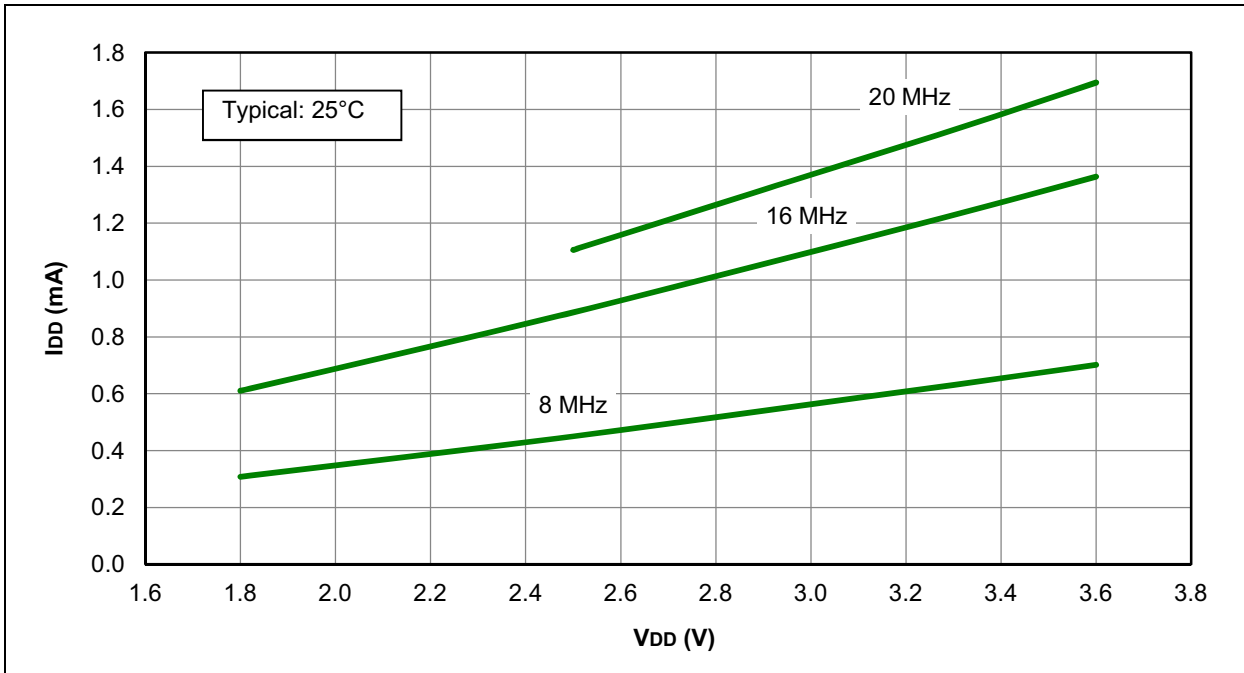


FIGURE 26-16: I_{DD} MAXIMUM, EXTERNAL CLOCK (ECH), HIGH-POWER MODE, PIC16LF1526 ONLY

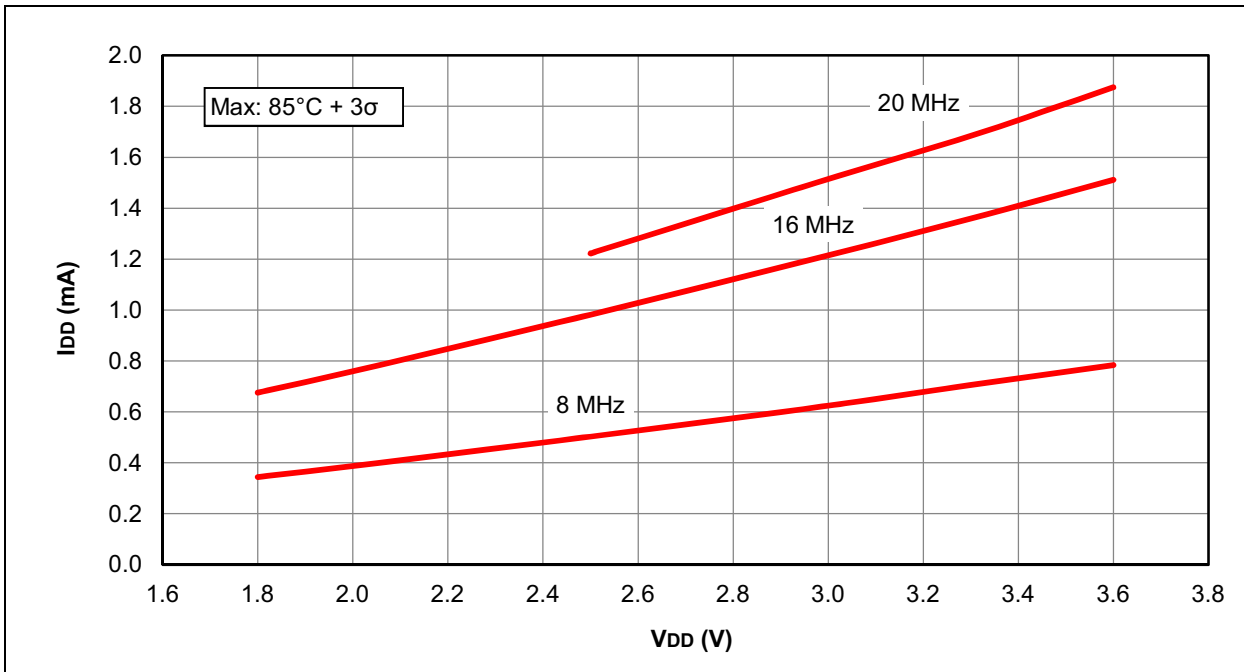


FIGURE 26-17: I_{DD} TYPICAL, EXTERNAL CLOCK (ECH), HIGH-POWER MODE, PIC16F1526/7 ONLY

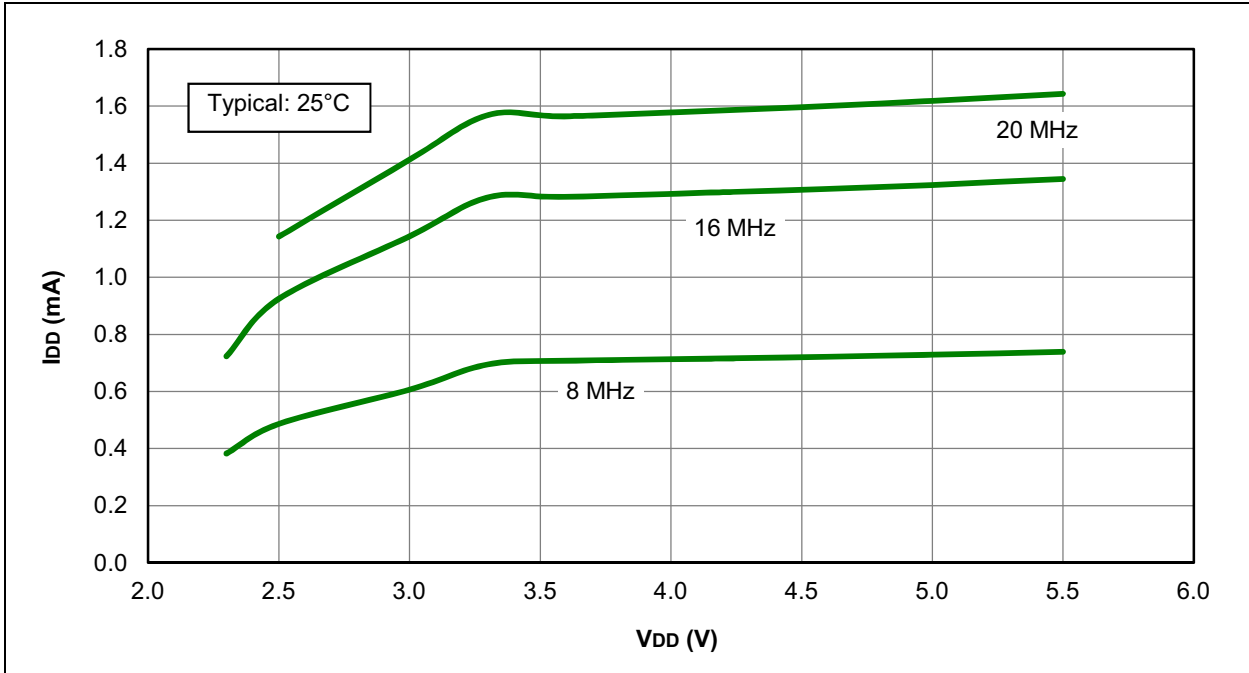
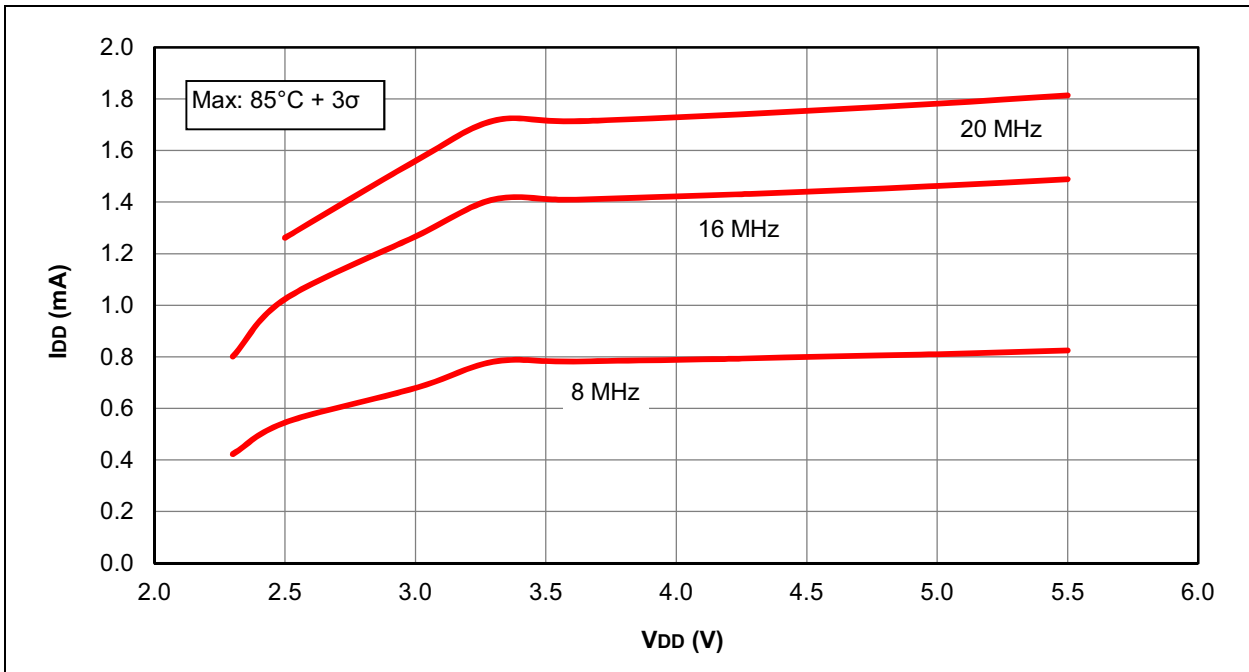


FIGURE 26-18: I_{DD} MAXIMUM, EXTERNAL CLOCK (ECH), HIGH-POWER MODE, PIC16F1526/7 ONLY



PIC16(L)F1526/7

FIGURE 26-19: I_{DD} , LFINTOSC, $F_{osc} = 31$ kHz, PIC16LF1526 ONLY

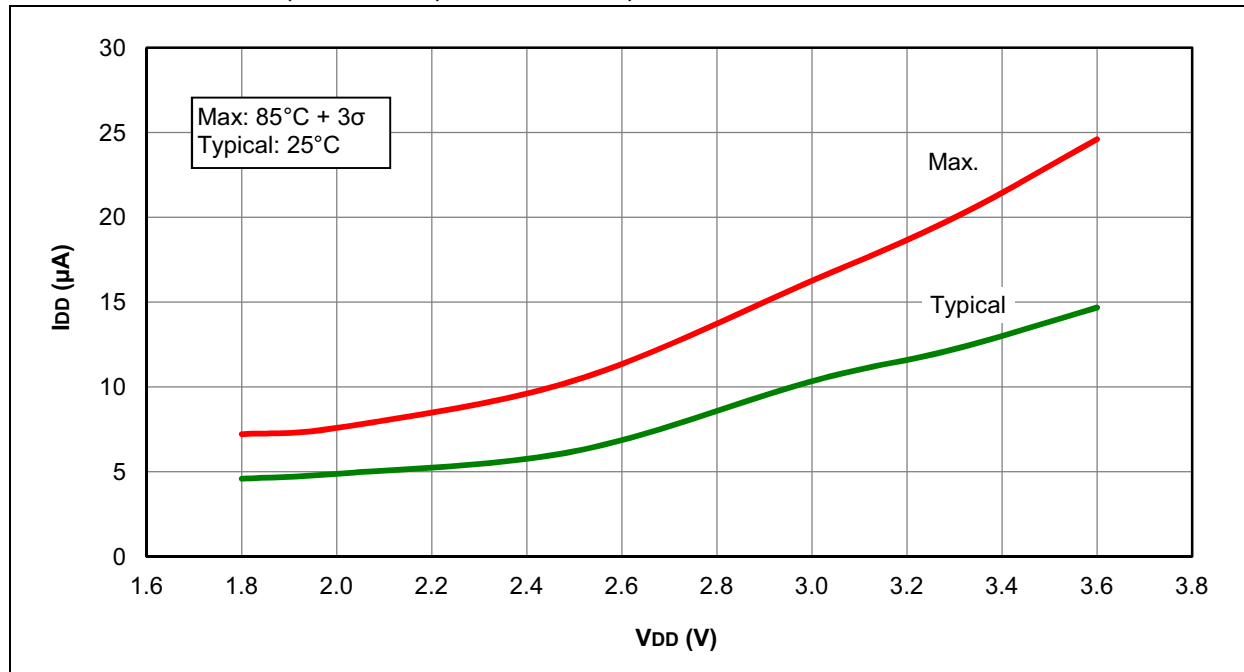


FIGURE 26-20: I_{DD} , LFINTOSC, $F_{osc} = 31$ kHz, PIC16F1526/7 ONLY

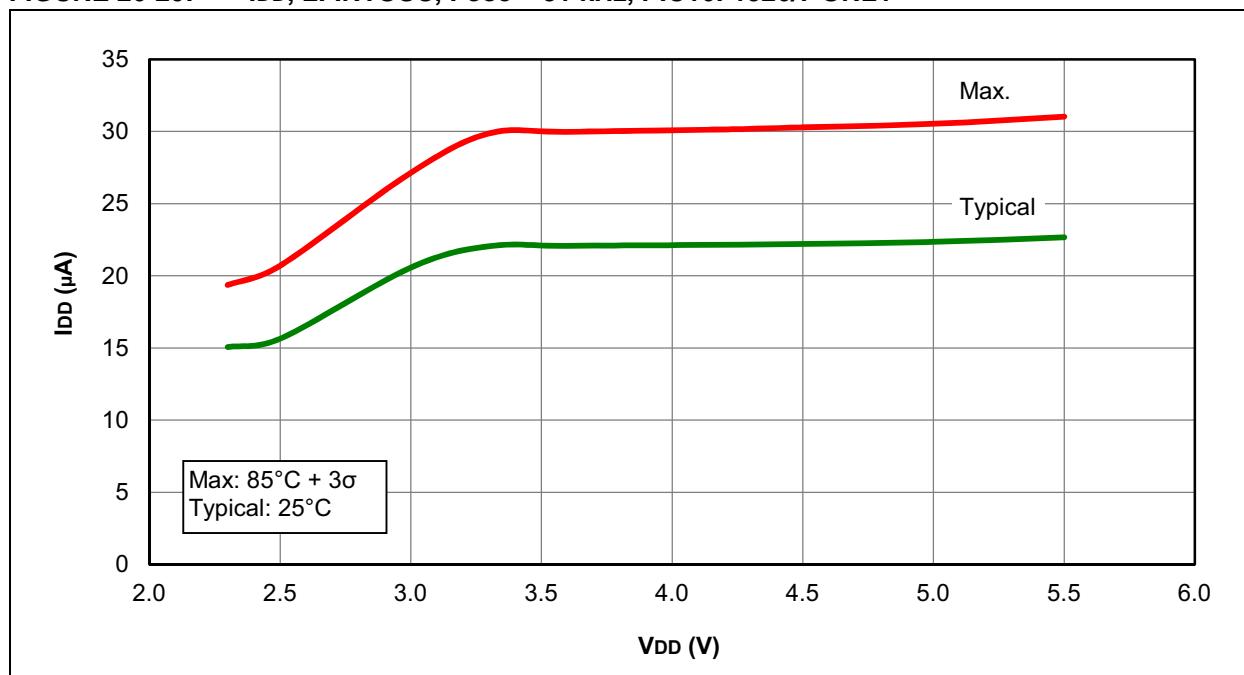


FIGURE 26-21: I_{DD}, MFINTOSC, F_{osc} = 500 kHz, PIC16LF1526 ONLY

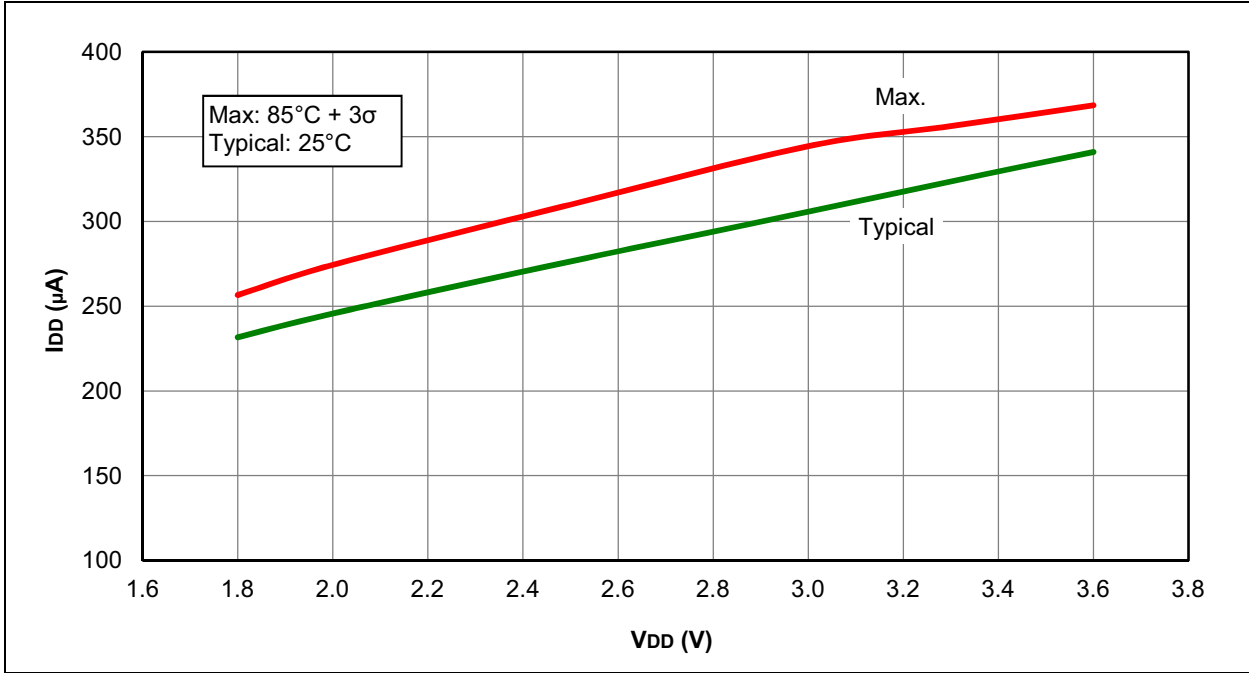
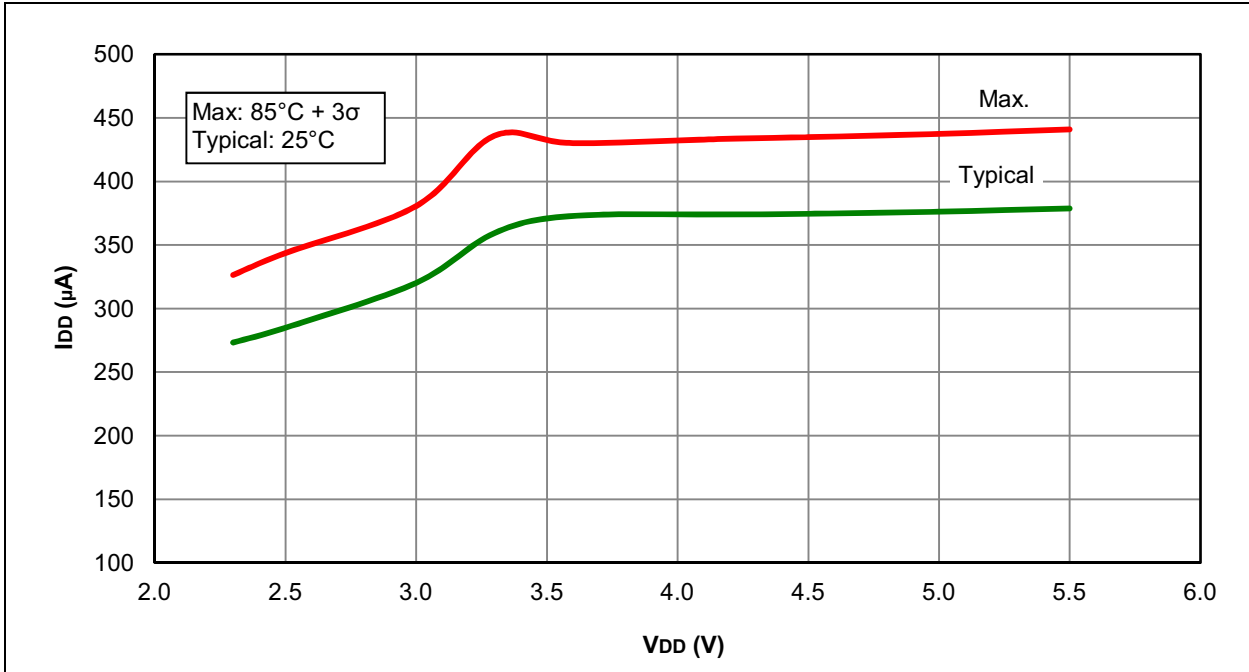


FIGURE 26-22: I_{DD}, MFINTOSC, F_{osc} = 500 kHz, PIC16F1526/7 ONLY



PIC16(L)F1526/7

FIGURE 26-23: I_{DD} TYPICAL, HFINTOSC, PIC16LF1526 ONLY

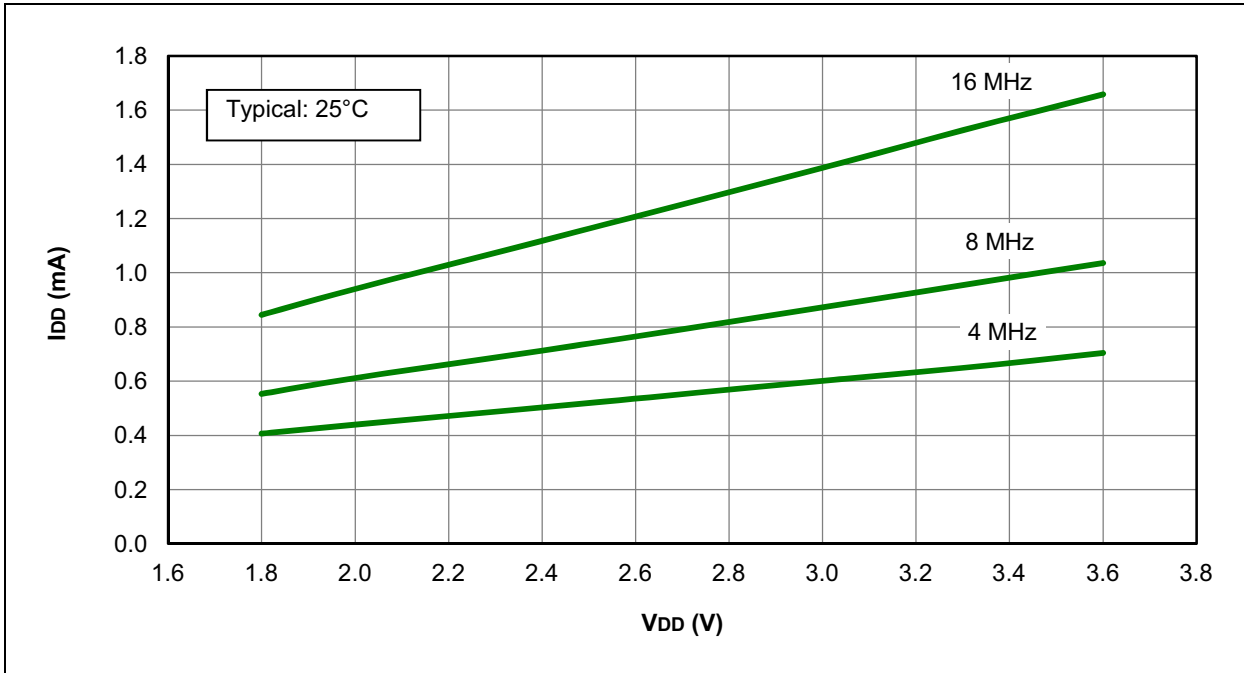


FIGURE 26-24: I_{DD} MAXIMUM, HFINTOSC, PIC16LF1526 ONLY

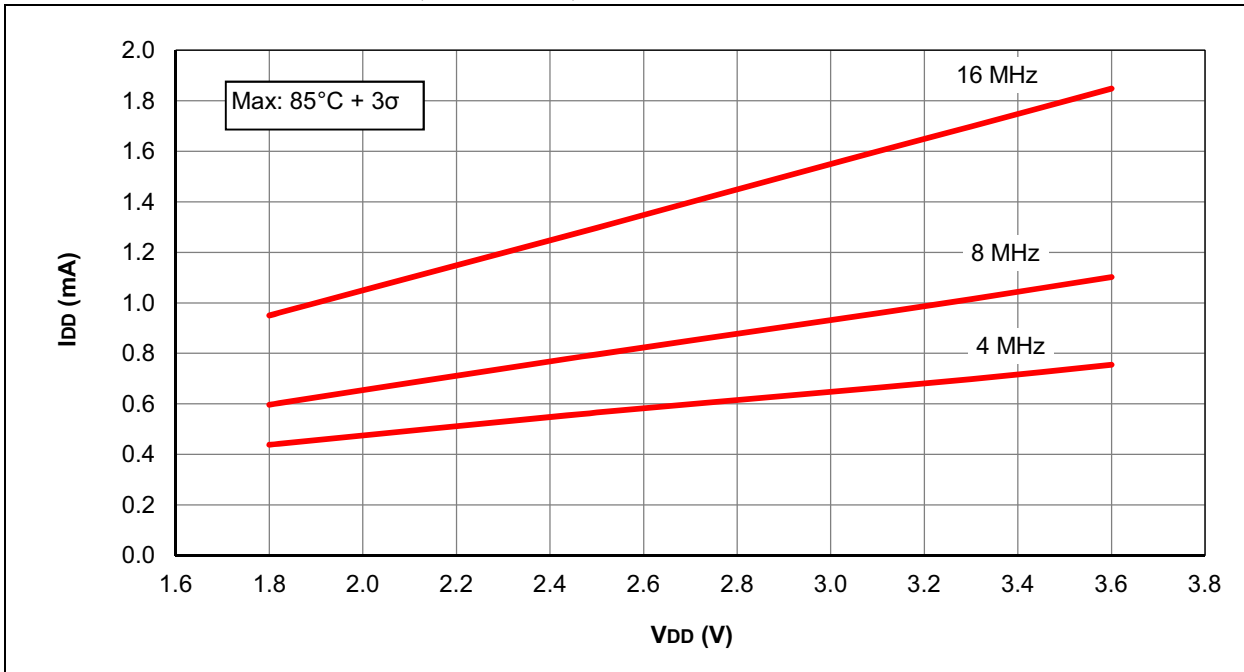


FIGURE 26-25: I_{DD} TYPICAL, HFINTOSC, PIC16F1526/7 ONLY

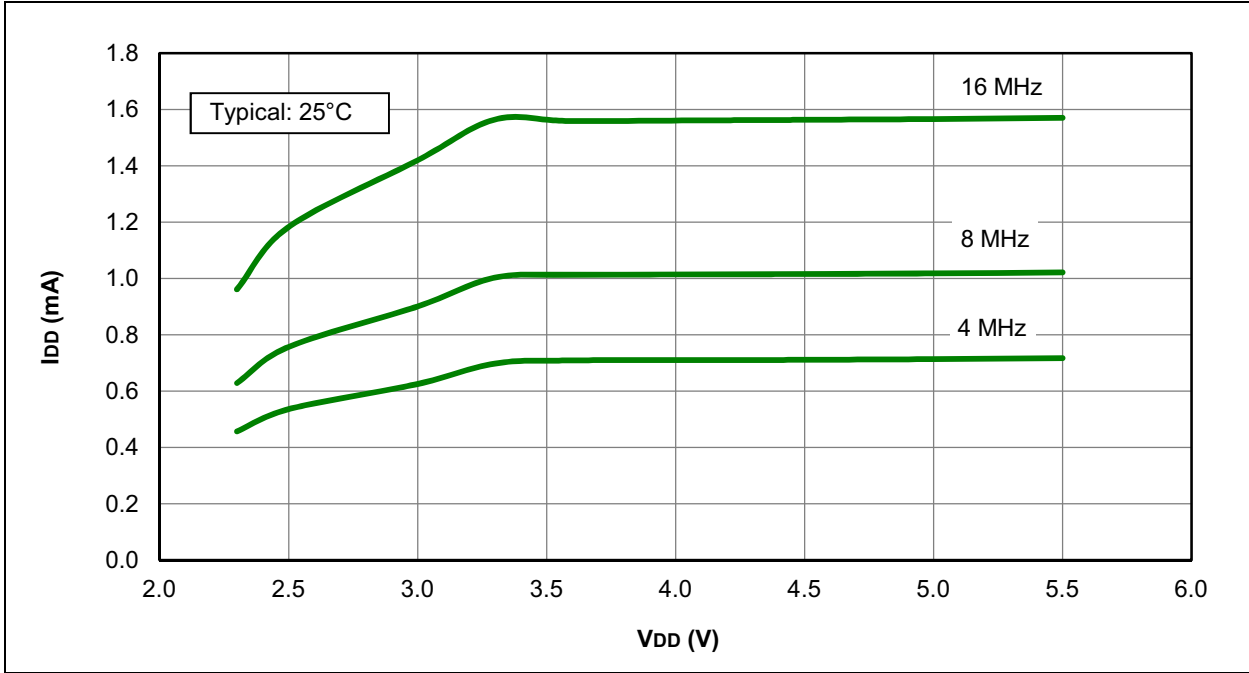
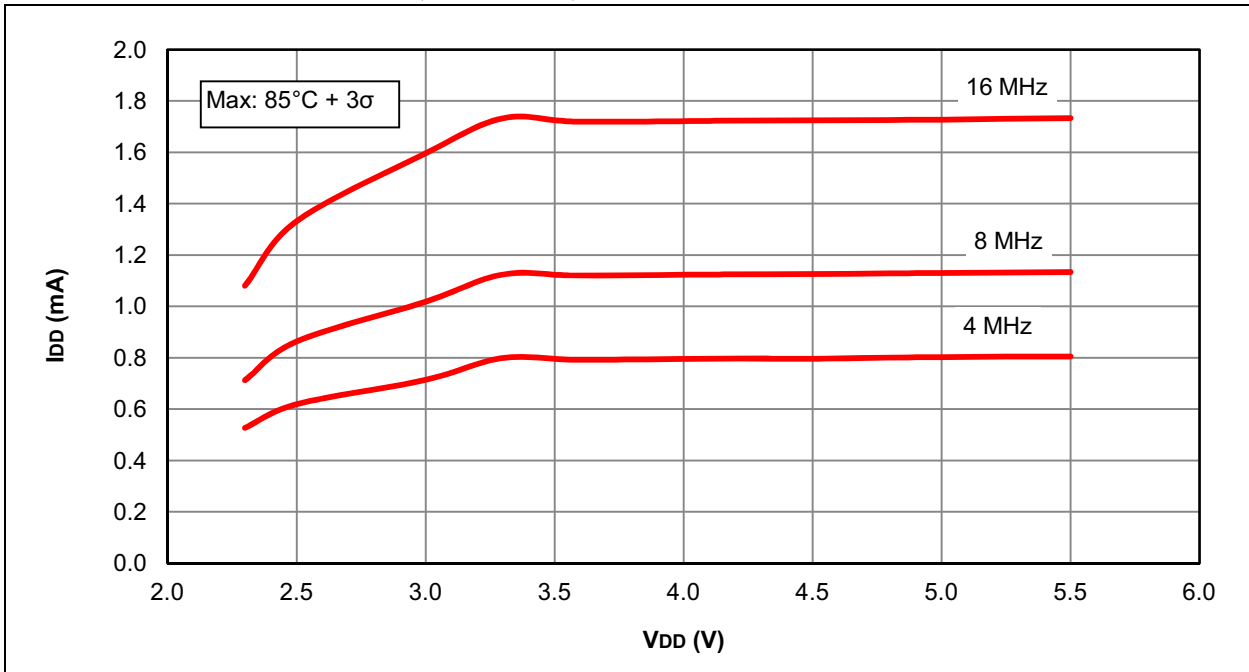


FIGURE 26-26: I_{DD} MAXIMUM, HFINTOSC, PIC16F1526/7 ONLY



PIC16(L)F1526/7

FIGURE 26-27: I_{DD} TYPICAL, HS OSCILLATOR, PIC16LF1526 ONLY

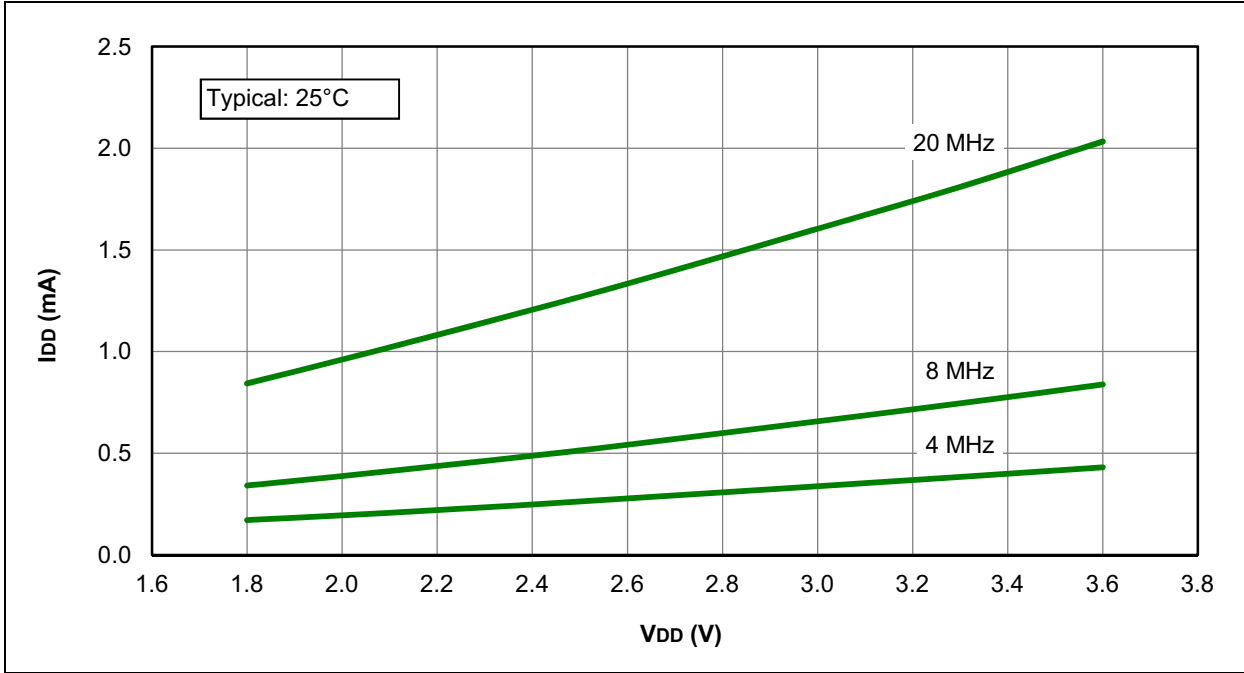


FIGURE 26-28: I_{DD} MAXIMUM, HS OSCILLATOR, PIC16LF1526 ONLY

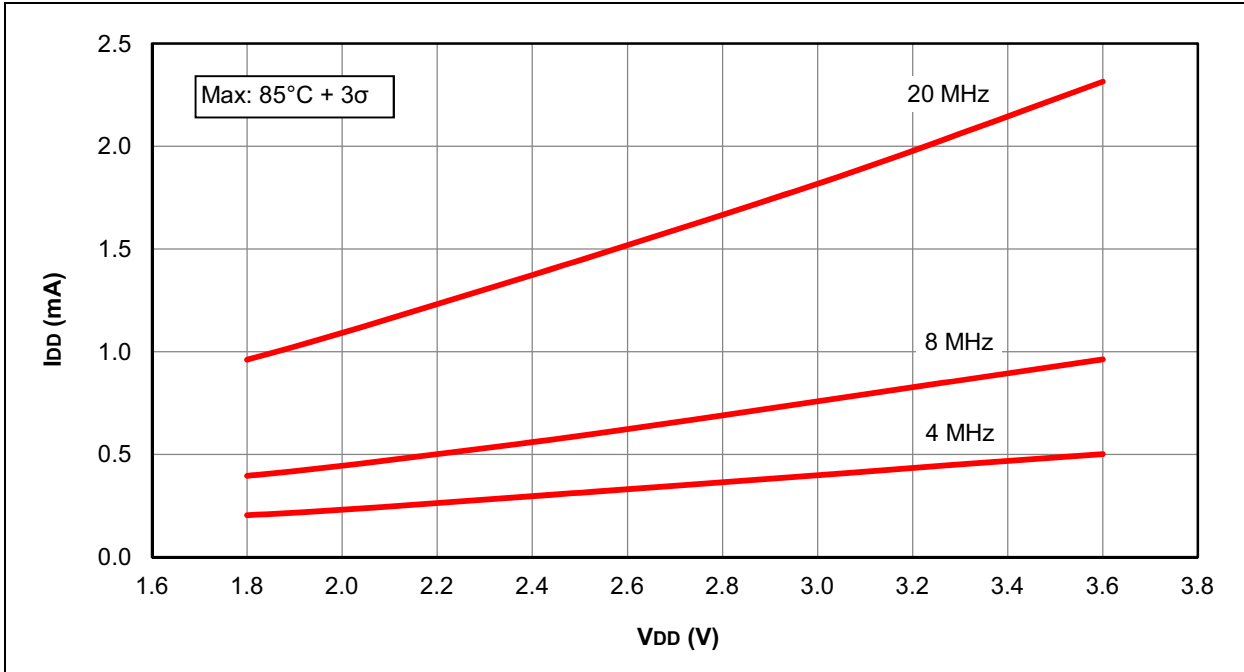


FIGURE 26-29: I_{DD} TYPICAL, HS OSCILLATOR, PIC16F1526/7 ONLY

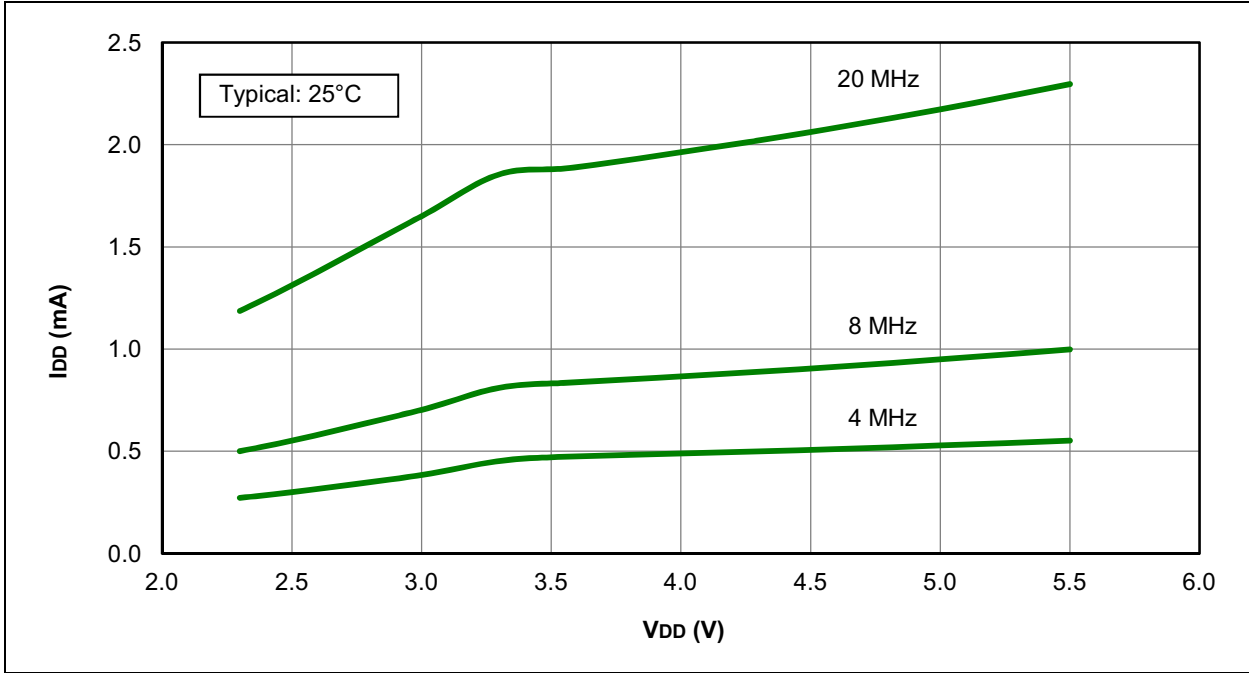
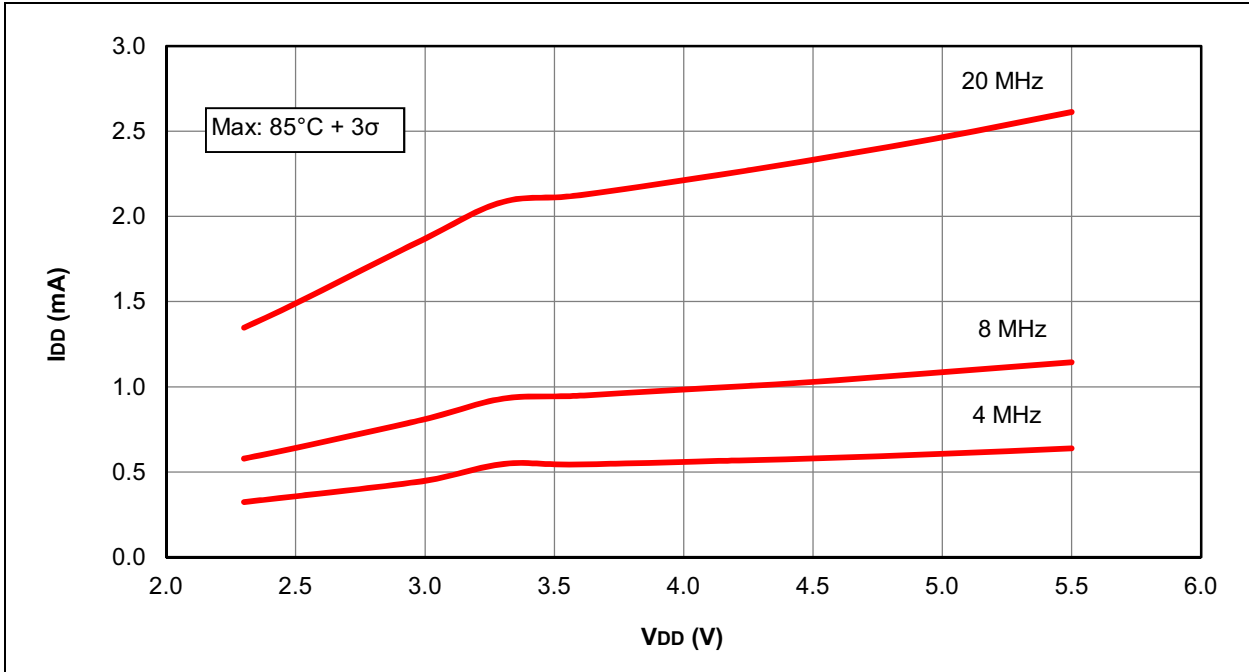


FIGURE 26-30: I_{DD} MAXIMUM, HS OSCILLATOR, PIC16F1526/7 ONLY



PIC16(L)F1526/7

FIGURE 26-31: I_{PD} BASE, SLEEP MODE, PIC16LF1526 ONLY

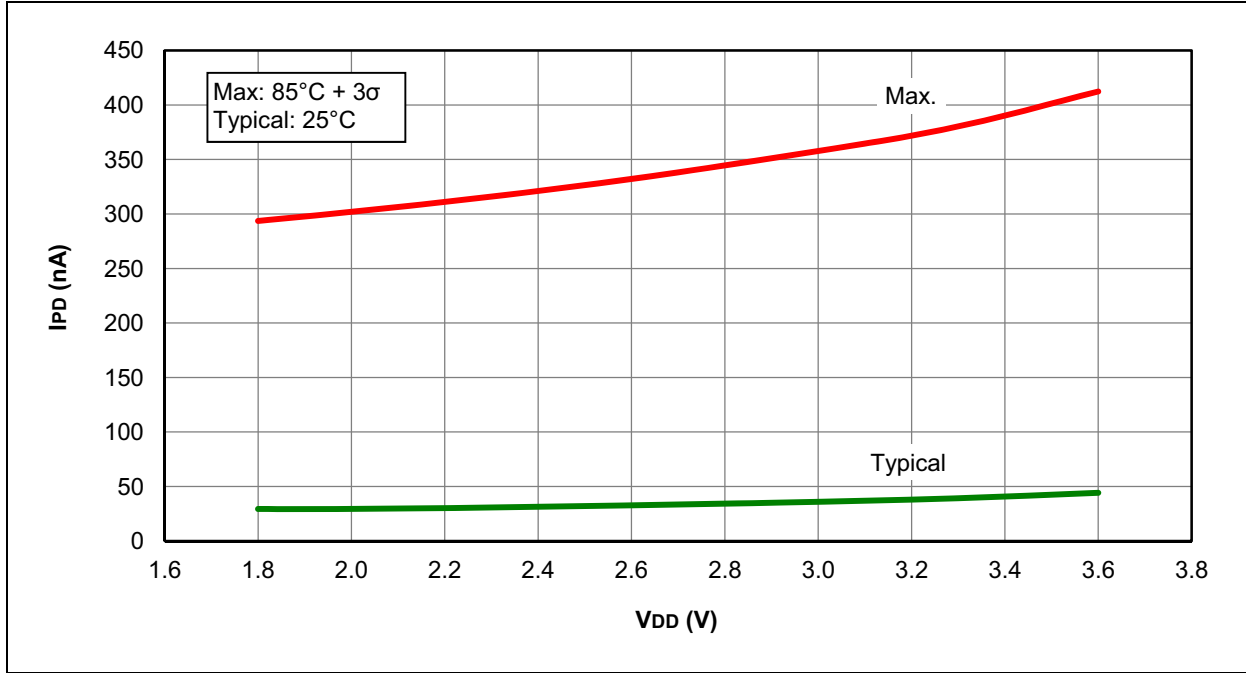


FIGURE 26-32: I_{PD} BASE, LOW-POWER SLEEP MODE, V_{REGPM} = 1, PIC16F1526/7 ONLY

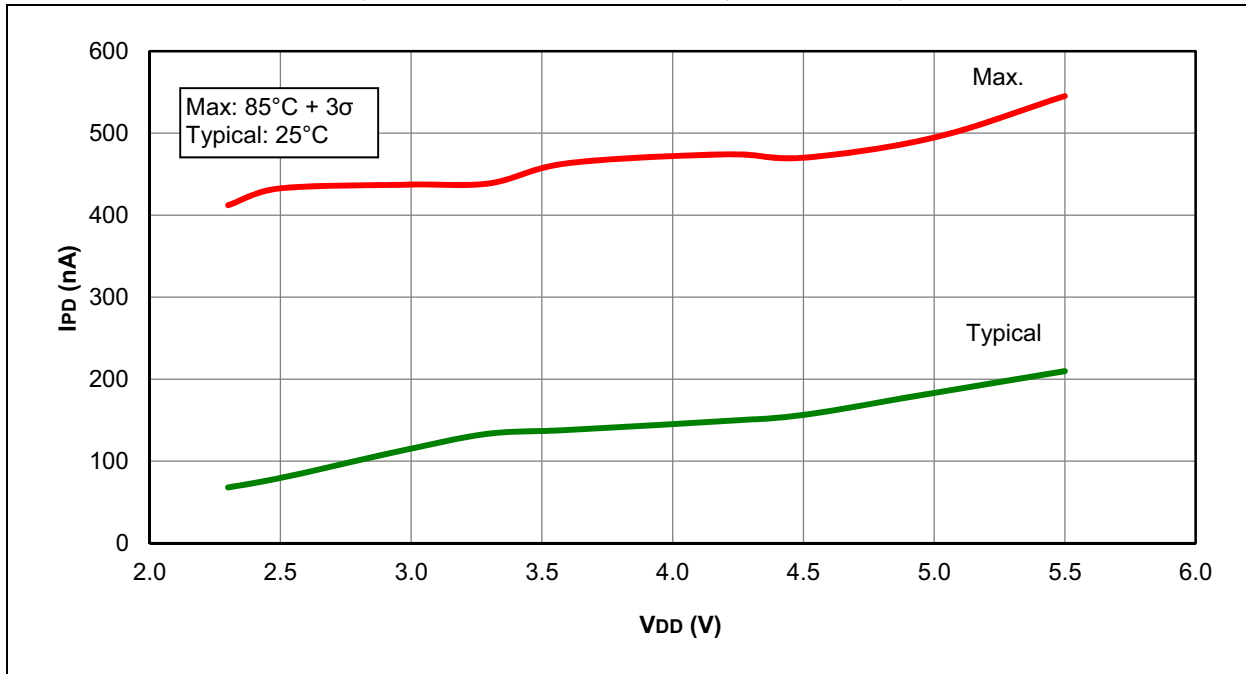


FIGURE 26-33: I_{PD}, WATCHDOG TIMER (WDT), PIC16LF1526 ONLY

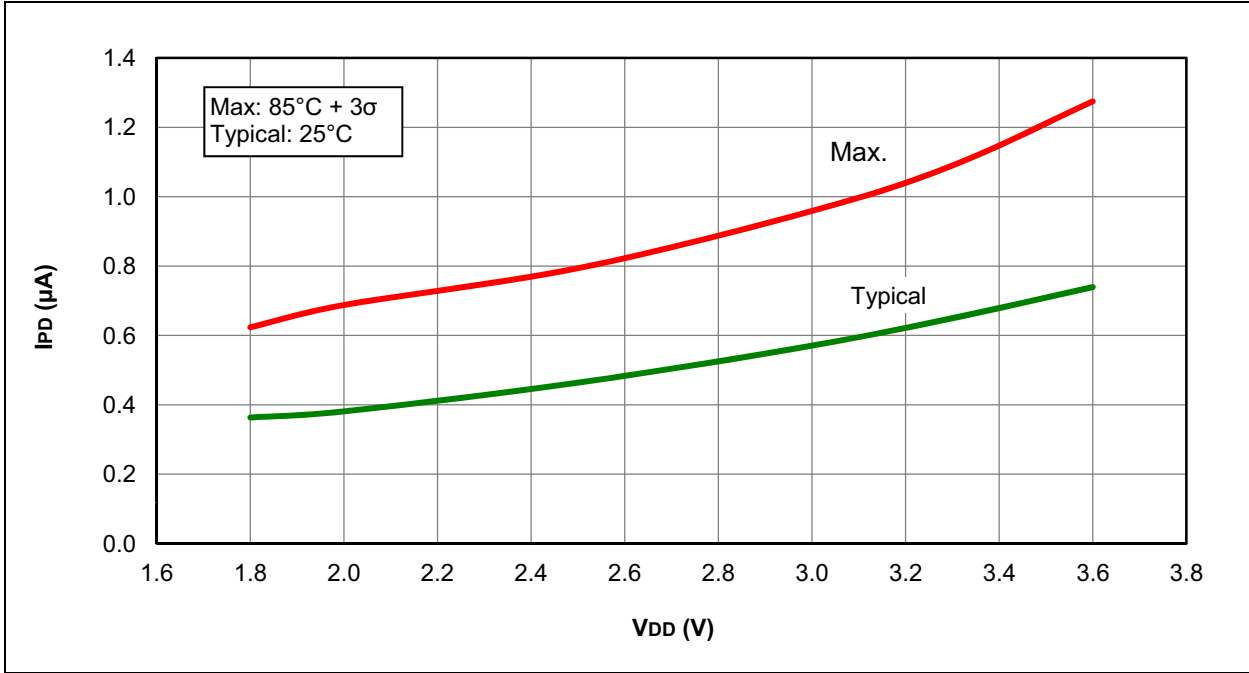
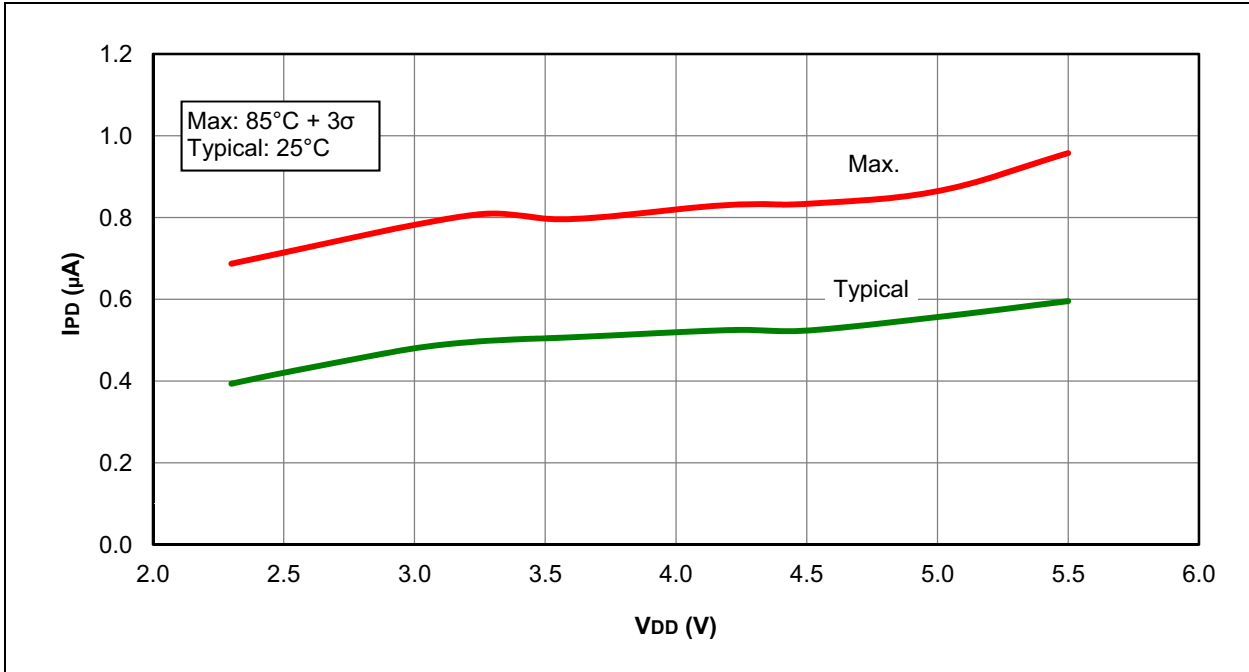


FIGURE 26-34: I_{PD}, WATCHDOG TIMER (WDT), PIC16F1526/7 ONLY



PIC16(L)F1526/7

FIGURE 26-35: I_{PD}, FIXED VOLTAGE REFERENCE (FVR), PIC16LF1526 ONLY

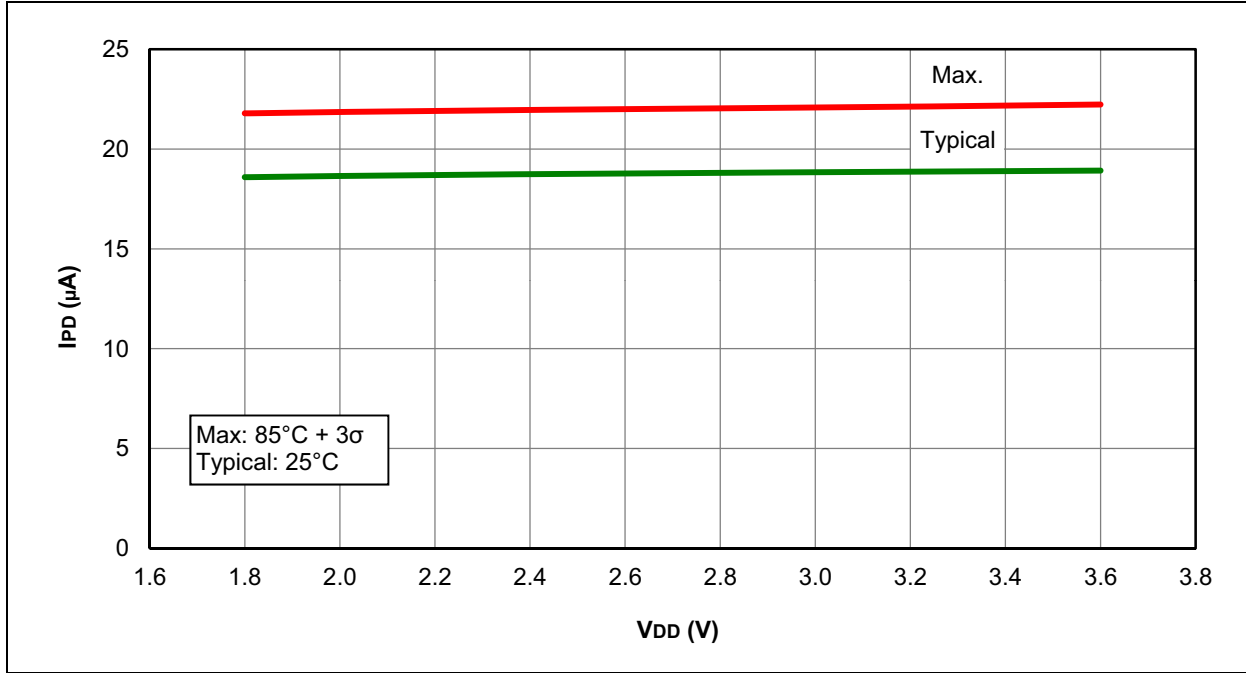


FIGURE 26-36: I_{PD}, FIXED VOLTAGE REFERENCE (FVR), PIC16F1526/7 ONLY

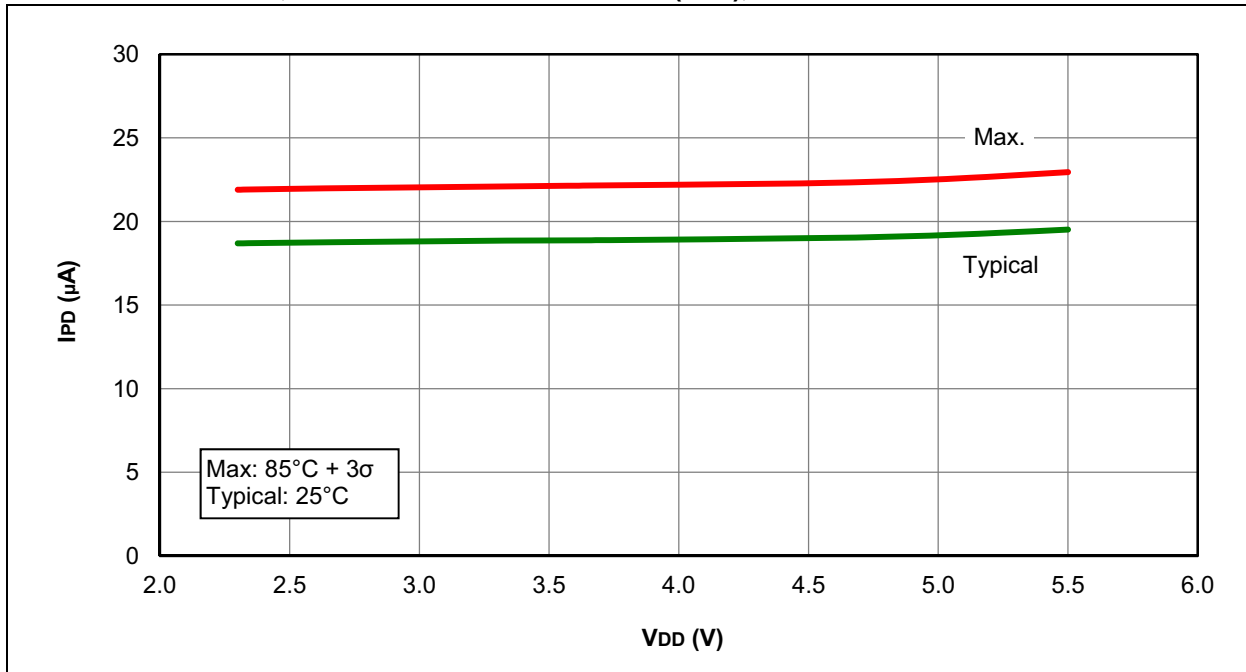


FIGURE 26-37: I_{PD}, BROWN-OUT RESET (BOR), BORV = 1, PIC16LF1526 ONLY

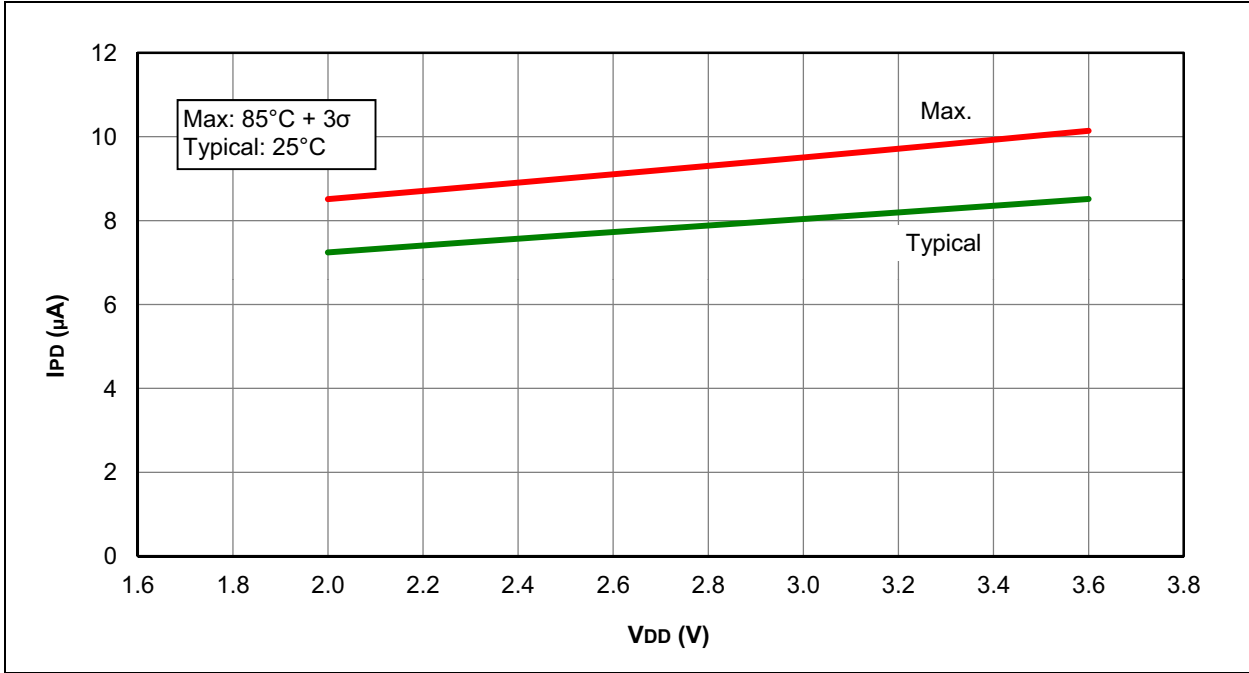
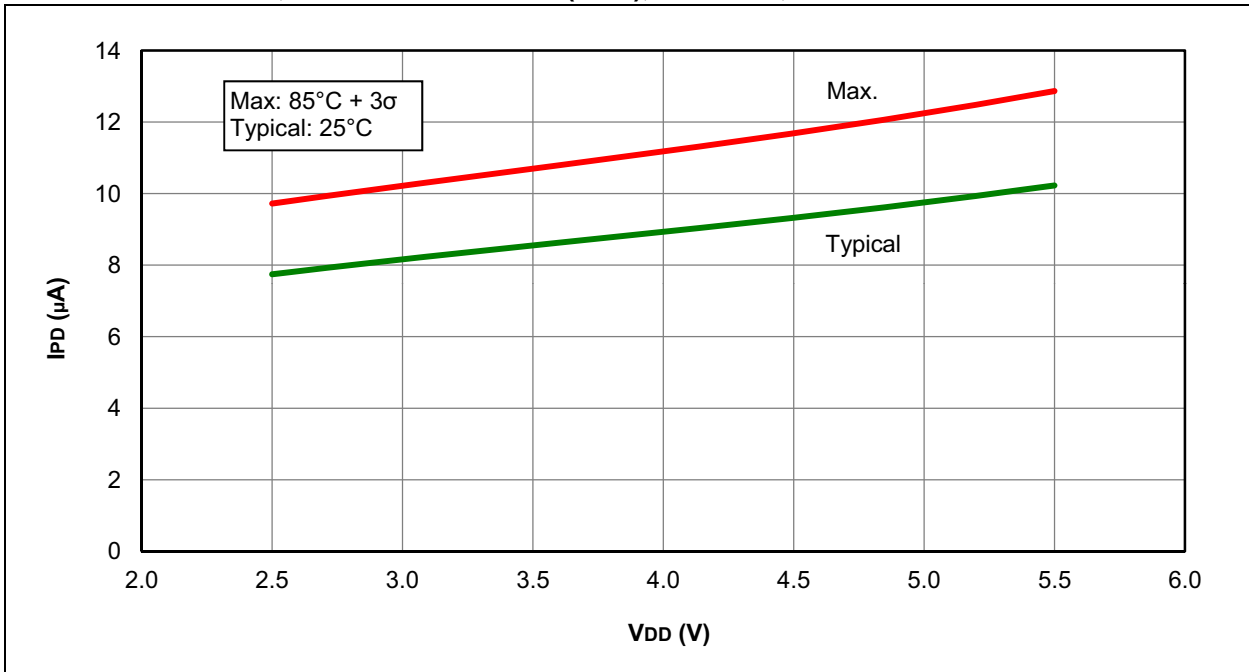


FIGURE 26-38: I_{PD}, BROWN-OUT RESET (BOR), BORV = 1, PIC16F1526/7 ONLY



PIC16(L)F1526/7

FIGURE 26-39: I_{PD} , SECONDARY OSCILLATOR, $F_{osc} = 32$ kHz, PIC16LF1526 ONLY

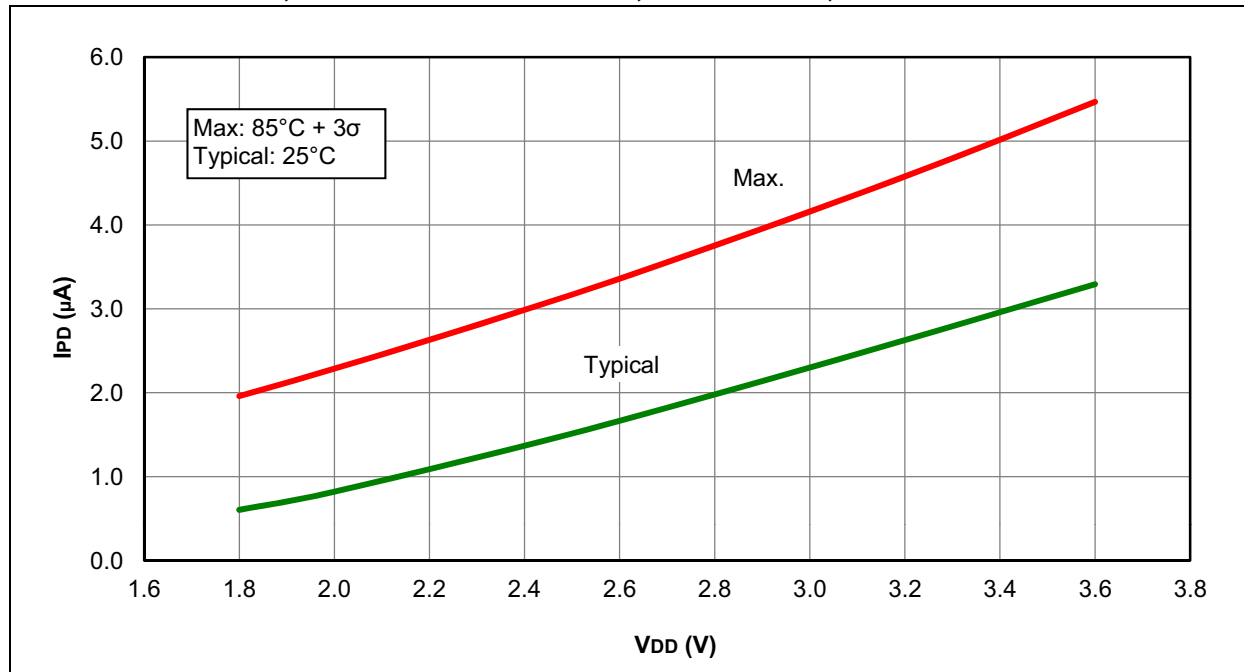


FIGURE 26-40: I_{PD} , SECONDARY OSCILLATOR, $F_{osc} = 32$ kHz, PIC16F1526/7 ONLY

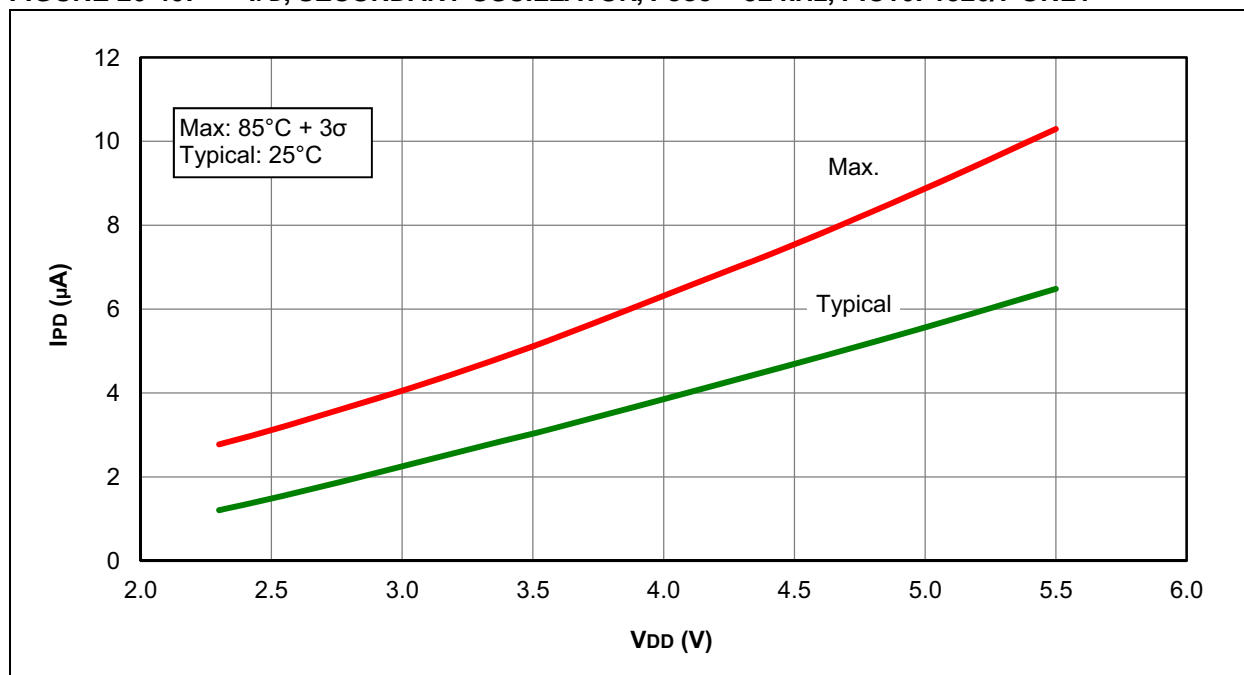


FIGURE 26-41: V_{OH} vs. I_{OH} OVER TEMPERATURE, $V_{DD} = 5.5V$, PIC16F1526/7 ONLY

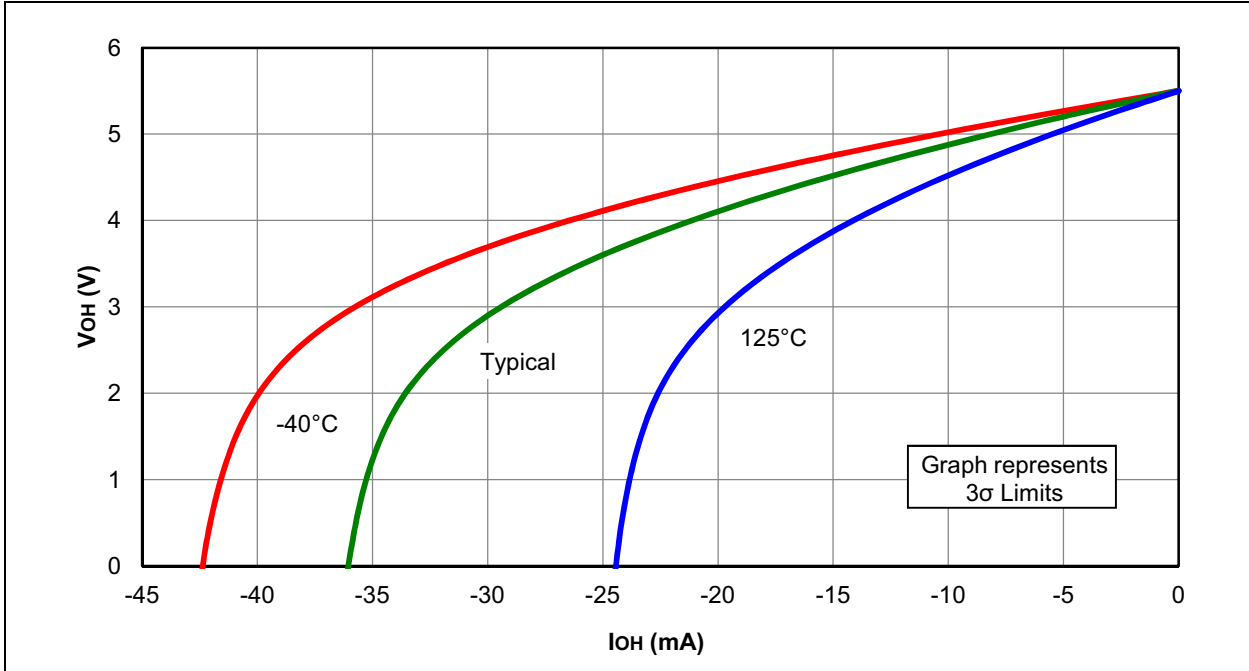
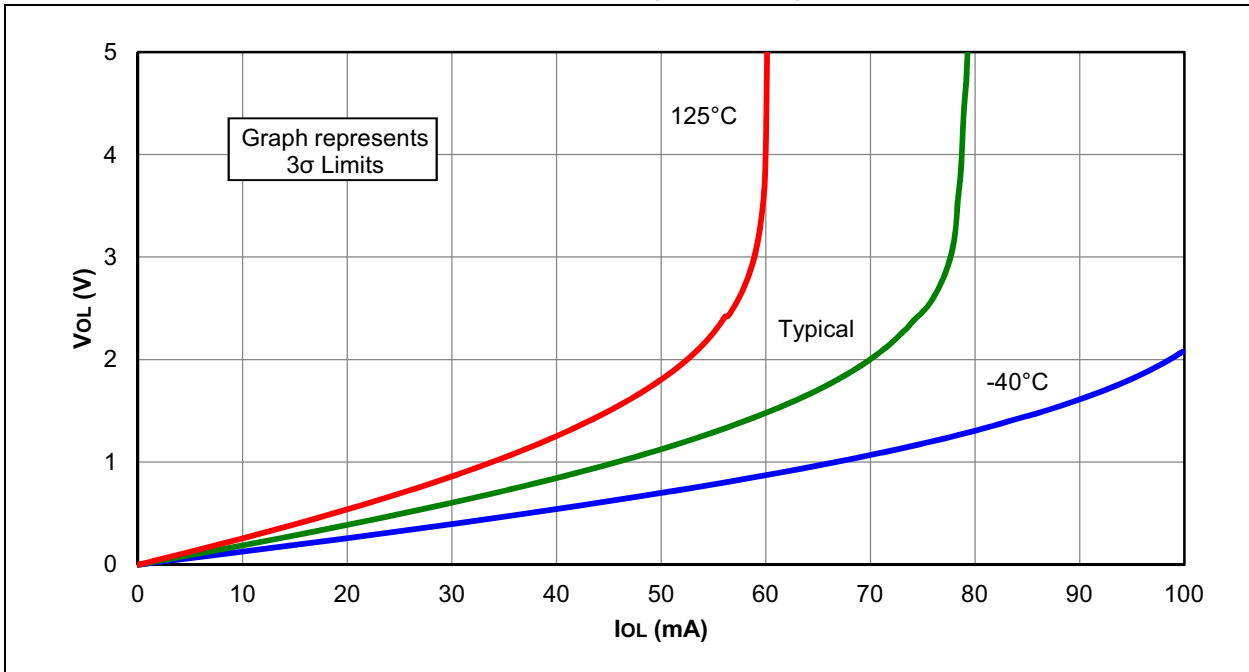


FIGURE 26-42: V_{OL} vs. I_{OL} OVER TEMPERATURE, $V_{DD} = 5.5V$, PIC16F1526/7 ONLY



PIC16(L)F1526/7

FIGURE 26-43: V_{OH} vs. I_{OH} OVER TEMPERATURE, $V_{DD} = 3.0V$

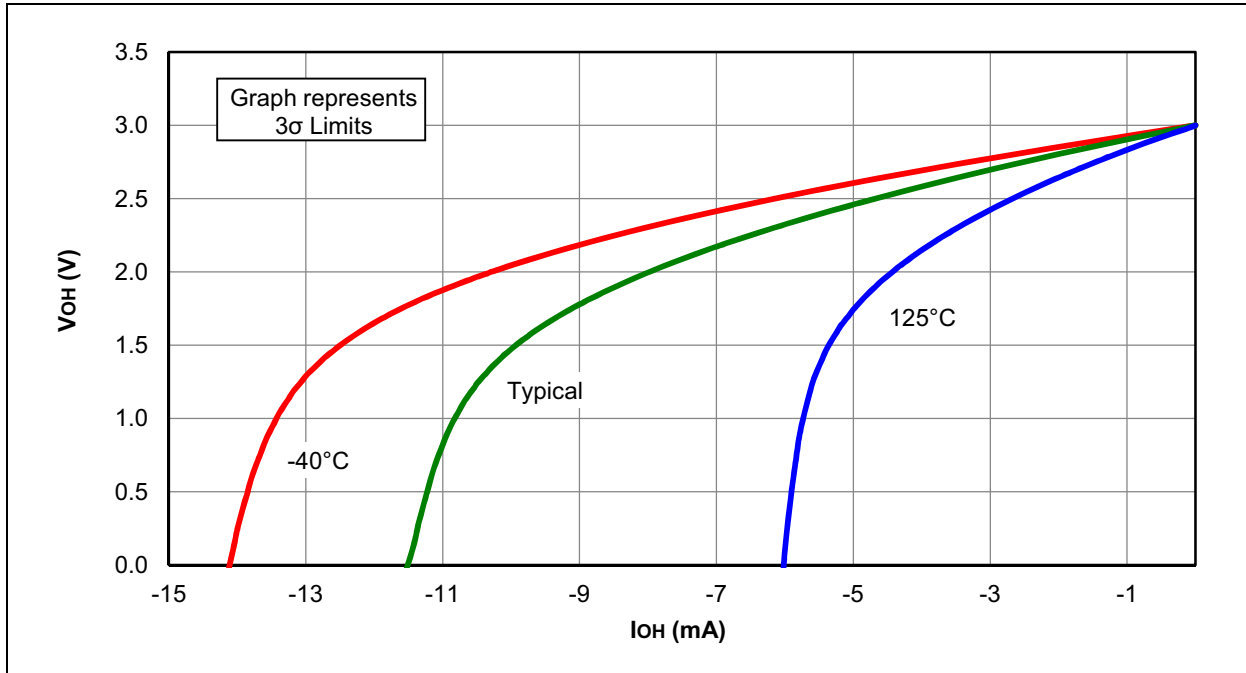


FIGURE 26-44: V_{OL} vs. I_{OL} OVER TEMPERATURE, $V_{DD} = 3.0V$

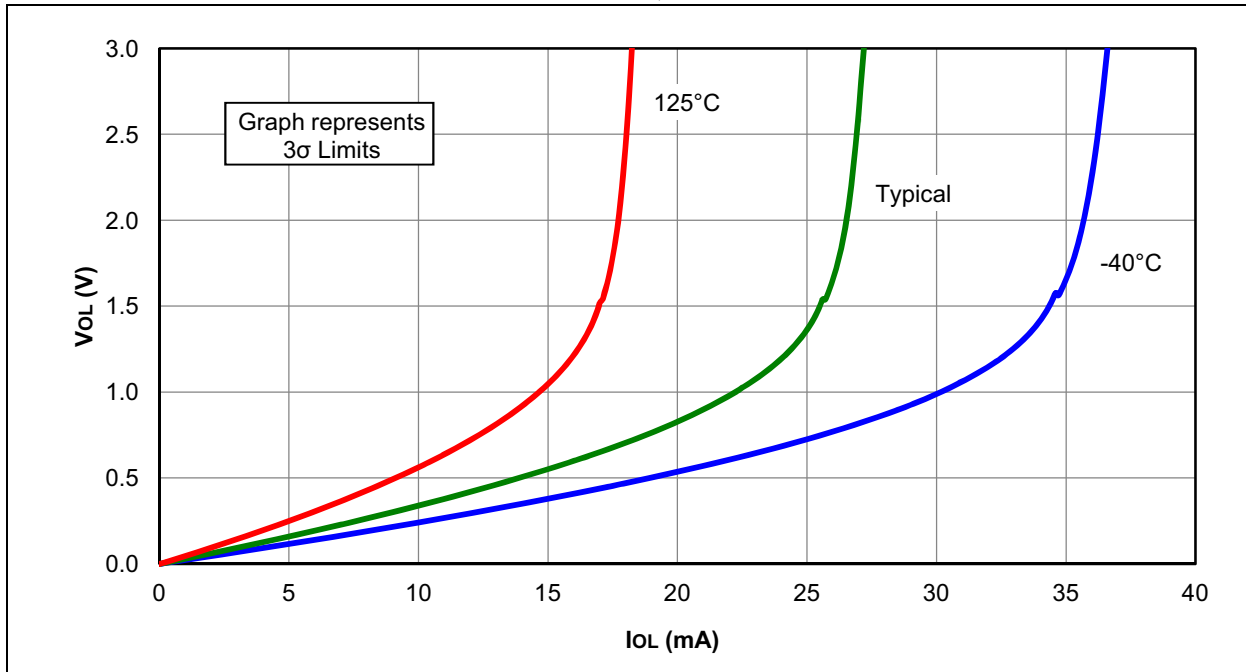


FIGURE 26-45: V_{OH} vs. I_{OH} OVER TEMPERATURE, $V_{DD} = 1.8V$, PIC16LF1526 ONLY

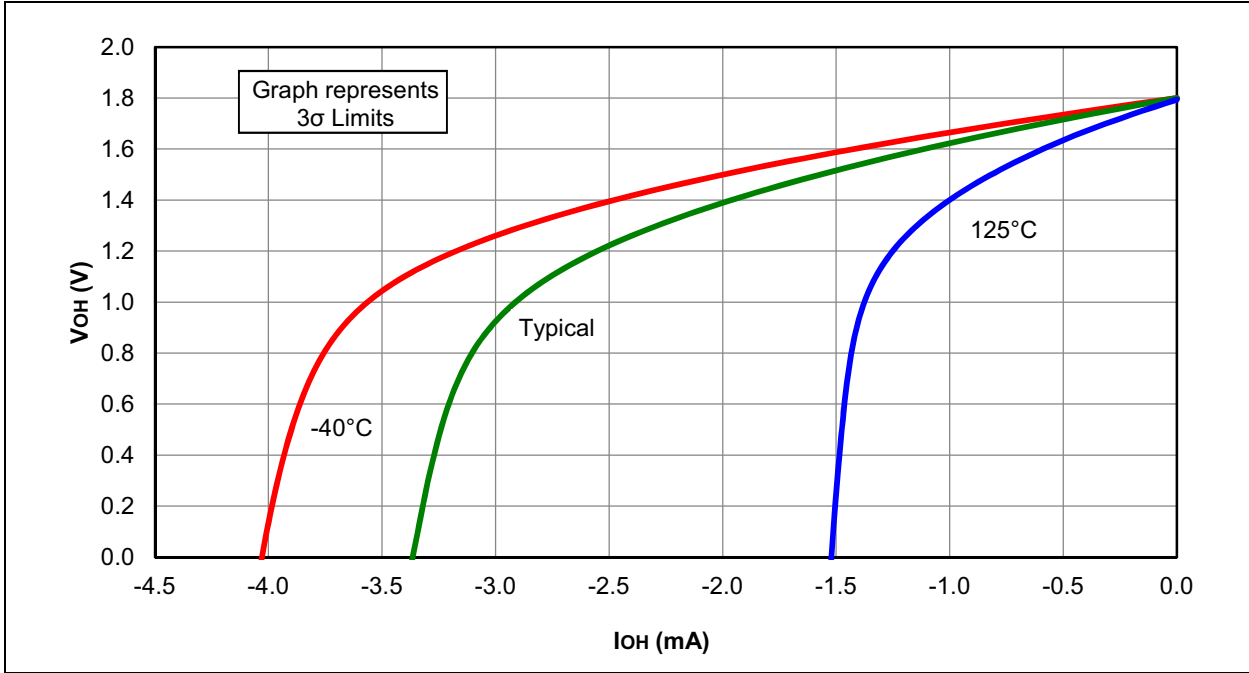
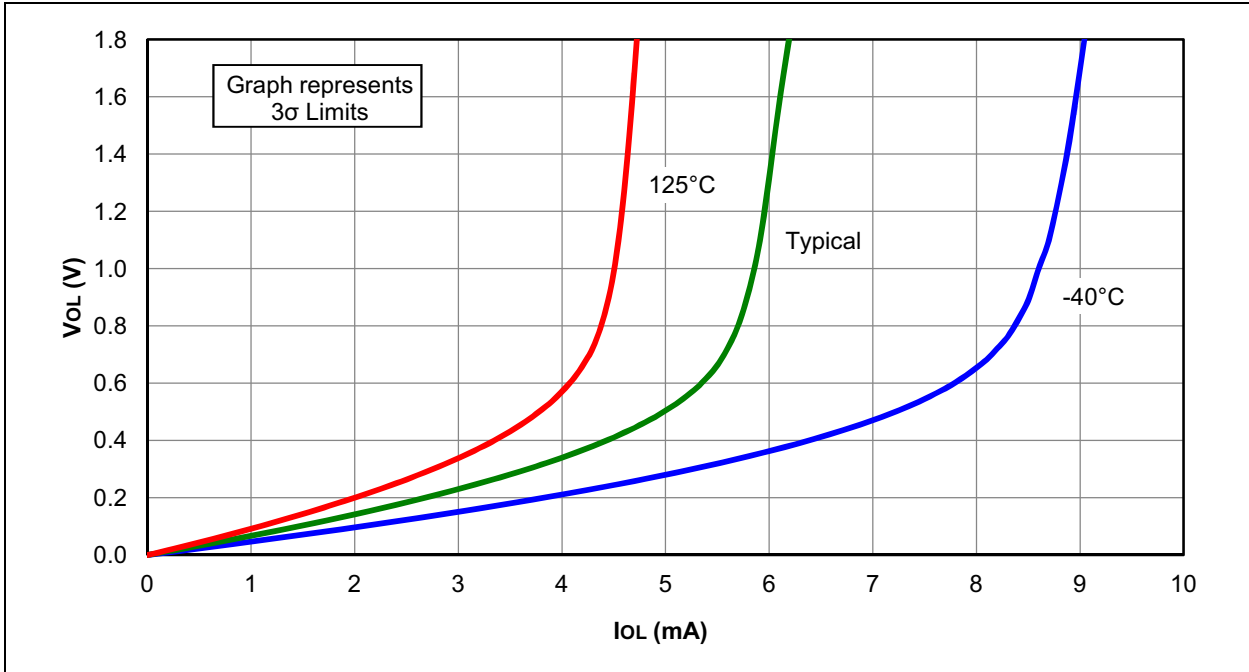


FIGURE 26-46: V_{OL} vs. I_{OL} OVER TEMPERATURE, $V_{DD} = 1.8V$, PIC16LF1526 ONLY



PIC16(L)F1526/7

FIGURE 26-47: POR RELEASE VOLTAGE

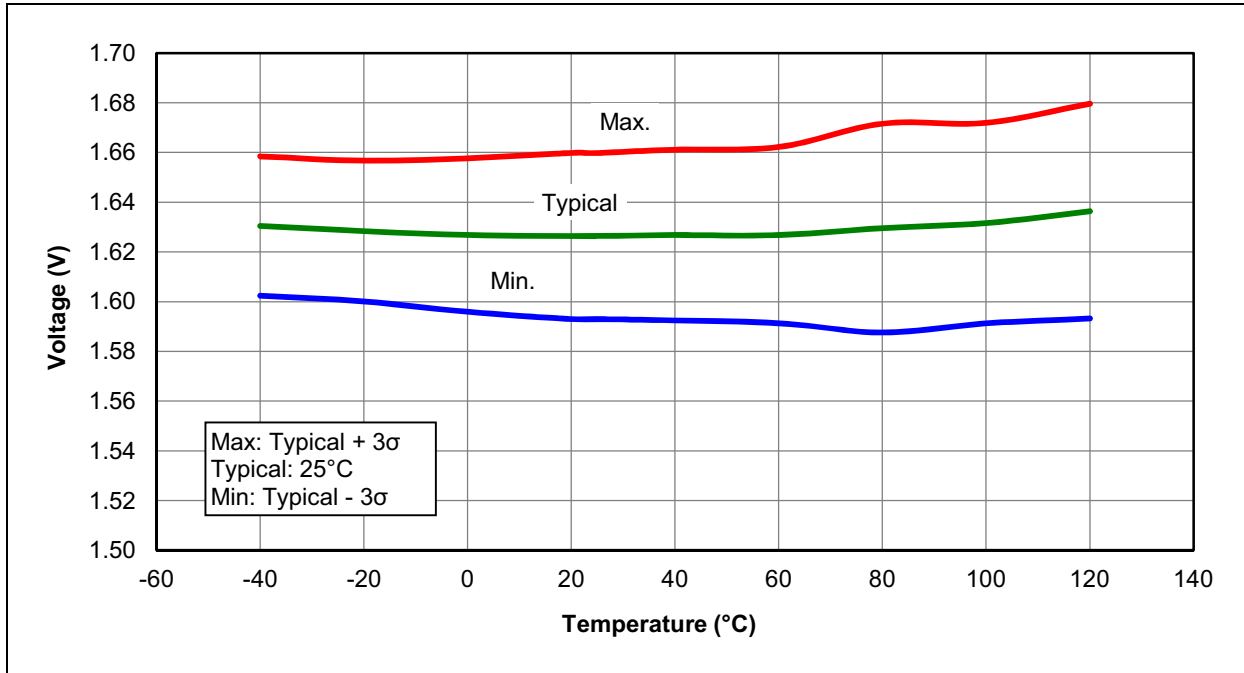


FIGURE 26-48: POR REARM VOLTAGE, PIC16F1526/7 ONLY

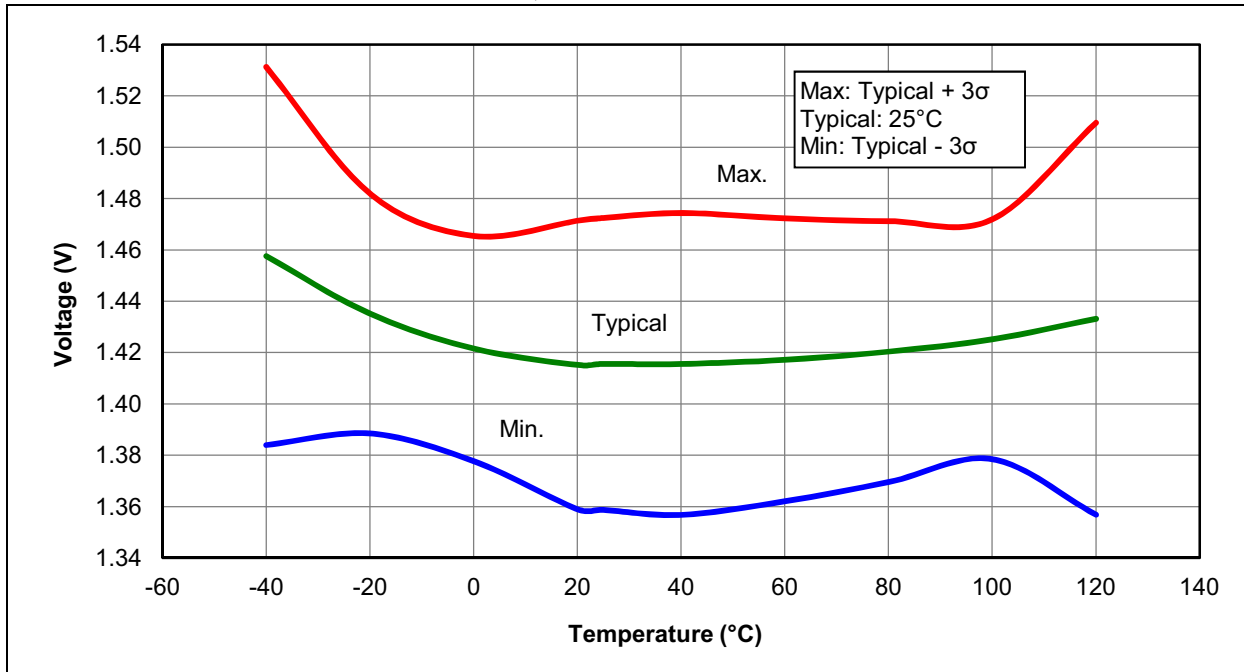


FIGURE 26-49: BROWN-OUT RESET VOLTAGE, BORV = 1, PIC16LF1526 ONLY

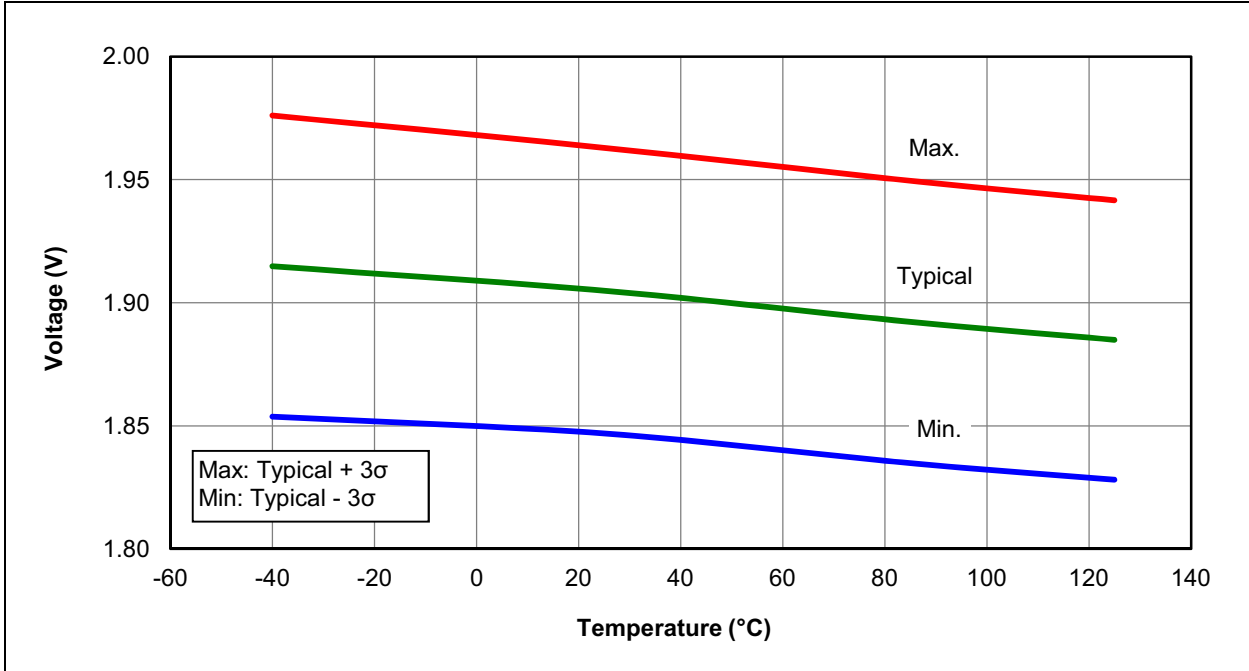
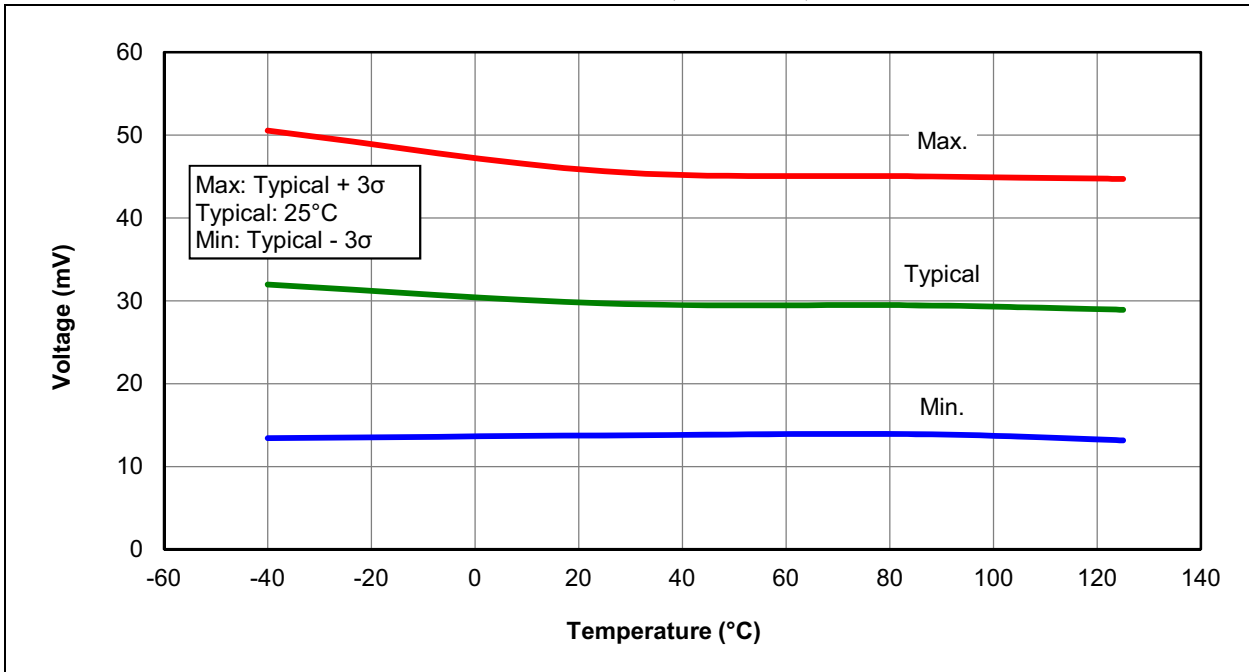


FIGURE 26-50: BROWN-OUT RESET HYSTERESIS, BORV = 1, PIC16LF1526 ONLY



PIC16(L)F1526/7

FIGURE 26-51: BROWN-OUT RESET VOLTAGE, BORV = 1, PIC16F1526/7 ONLY

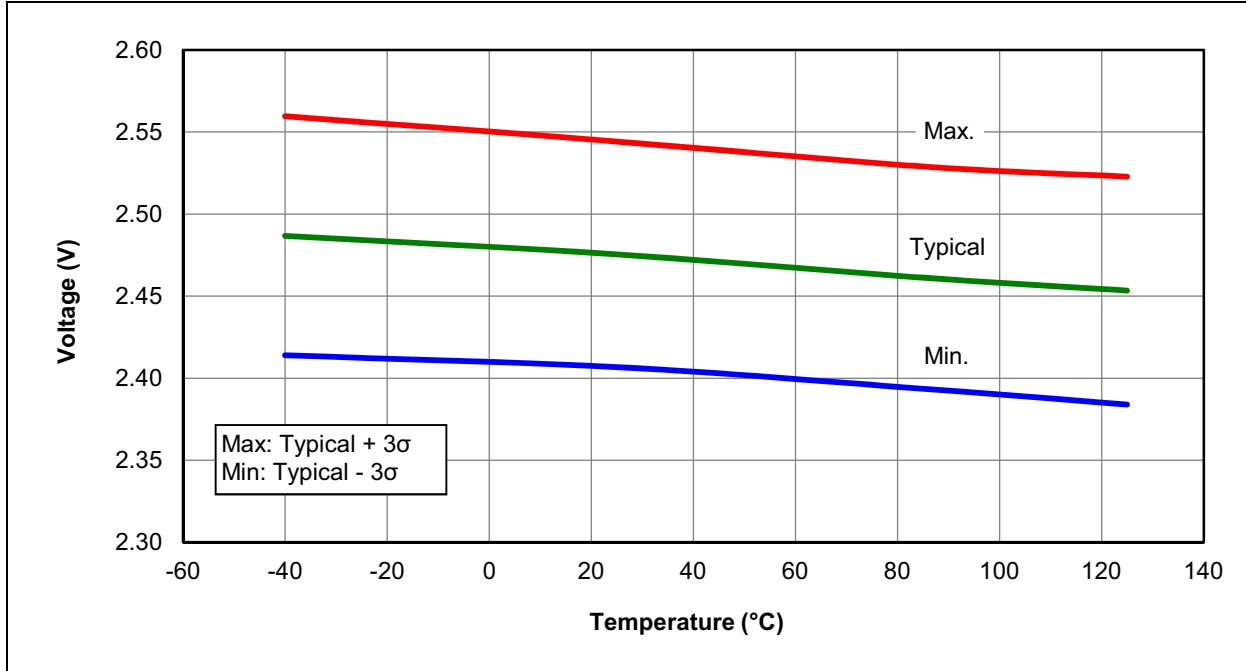


FIGURE 26-52: BROWN-OUT RESET HYSTERESIS, BORV = 1, PIC16F1526/7 ONLY

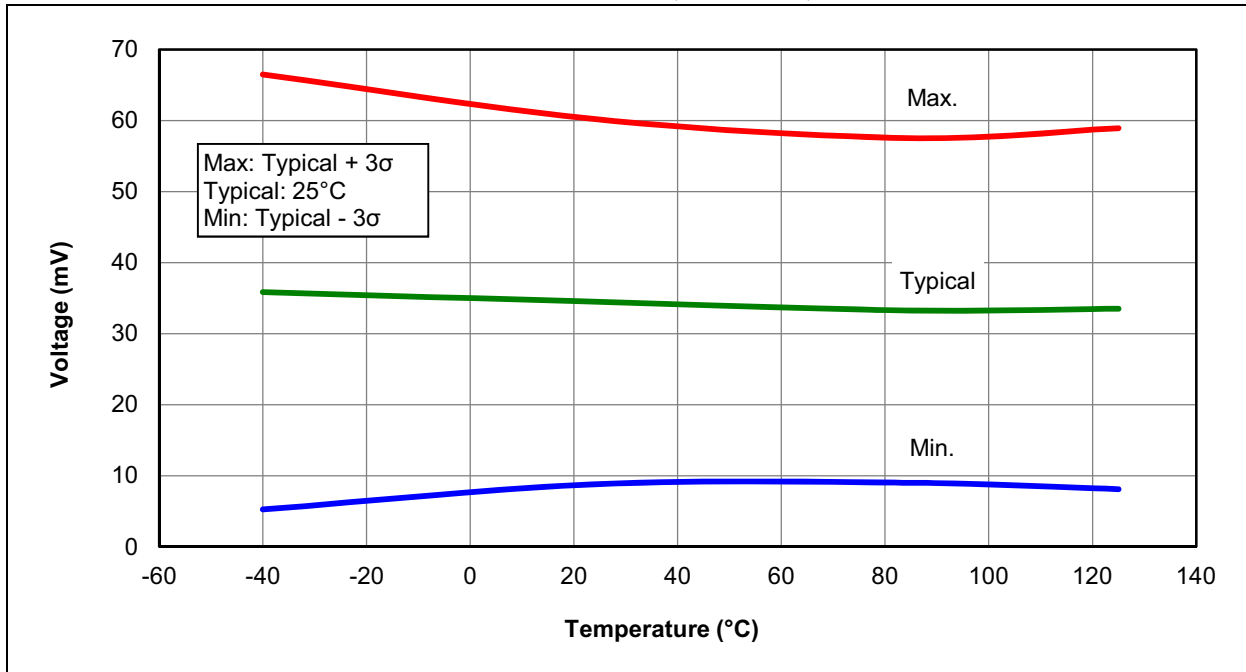


FIGURE 26-53: BROWN-OUT RESET VOLTAGE, BORV = 0

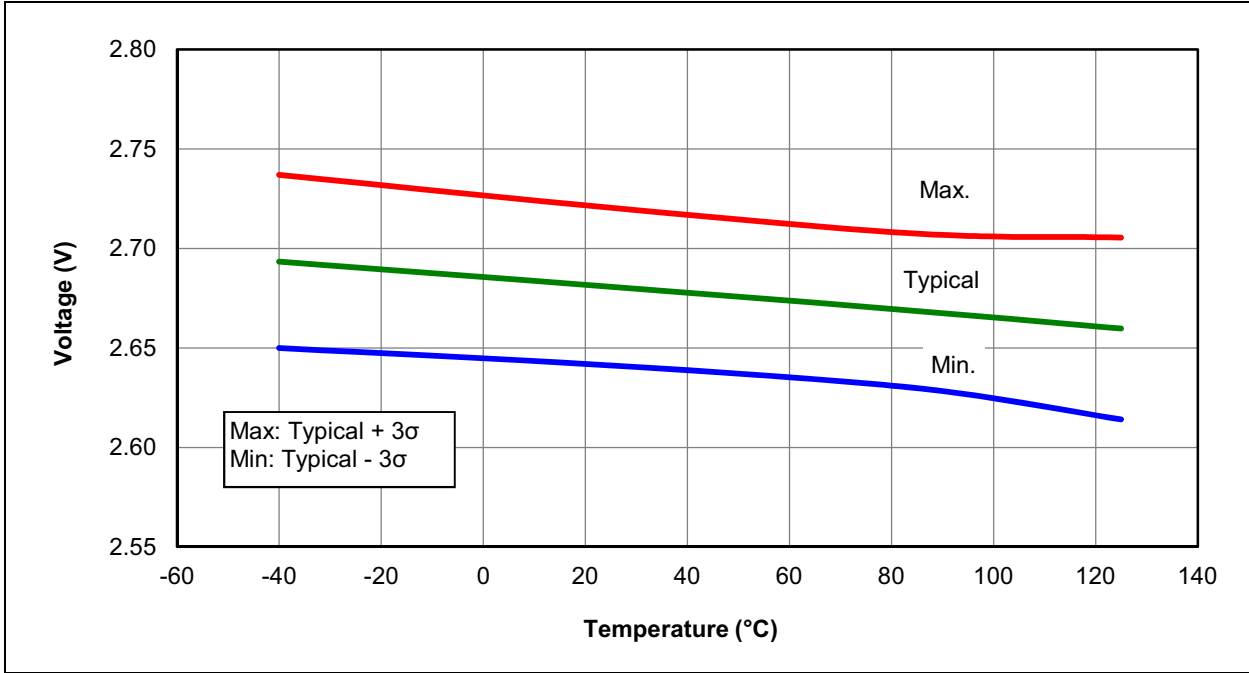
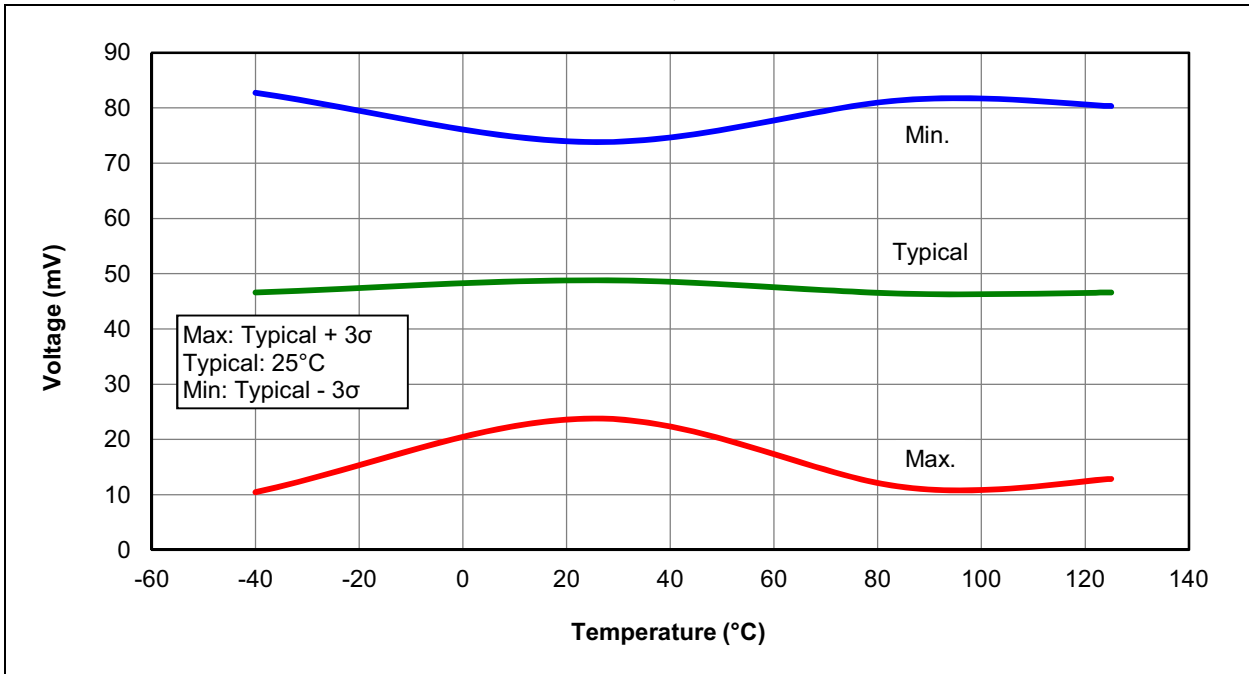


FIGURE 26-54: BROWN-OUT RESET HYSTERESIS, BORV = 0



PIC16(L)F1526/7

FIGURE 26-55: LOW-POWER BROWN-OUT RESET VOLTAGE, LPBOR = 0

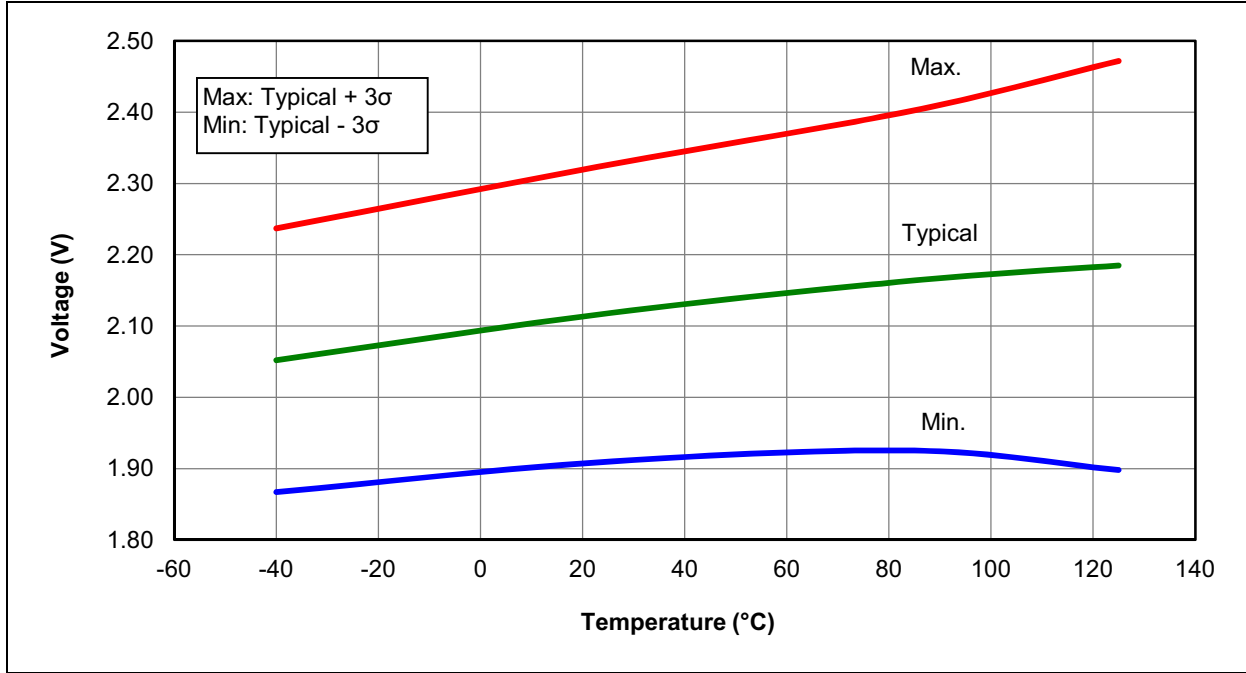


FIGURE 26-56: LOW-POWER BROWN-OUT RESET HYSTERESIS, LPBOR = 0

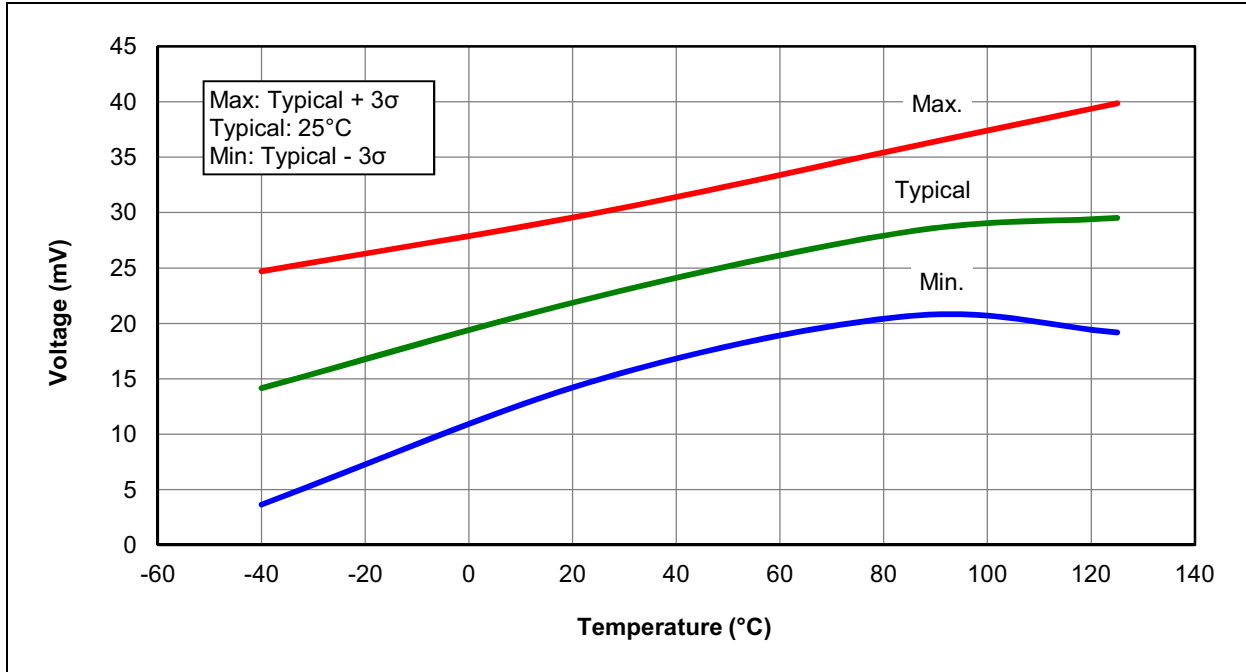


FIGURE 26-57: WDT TIME-OUT PERIOD

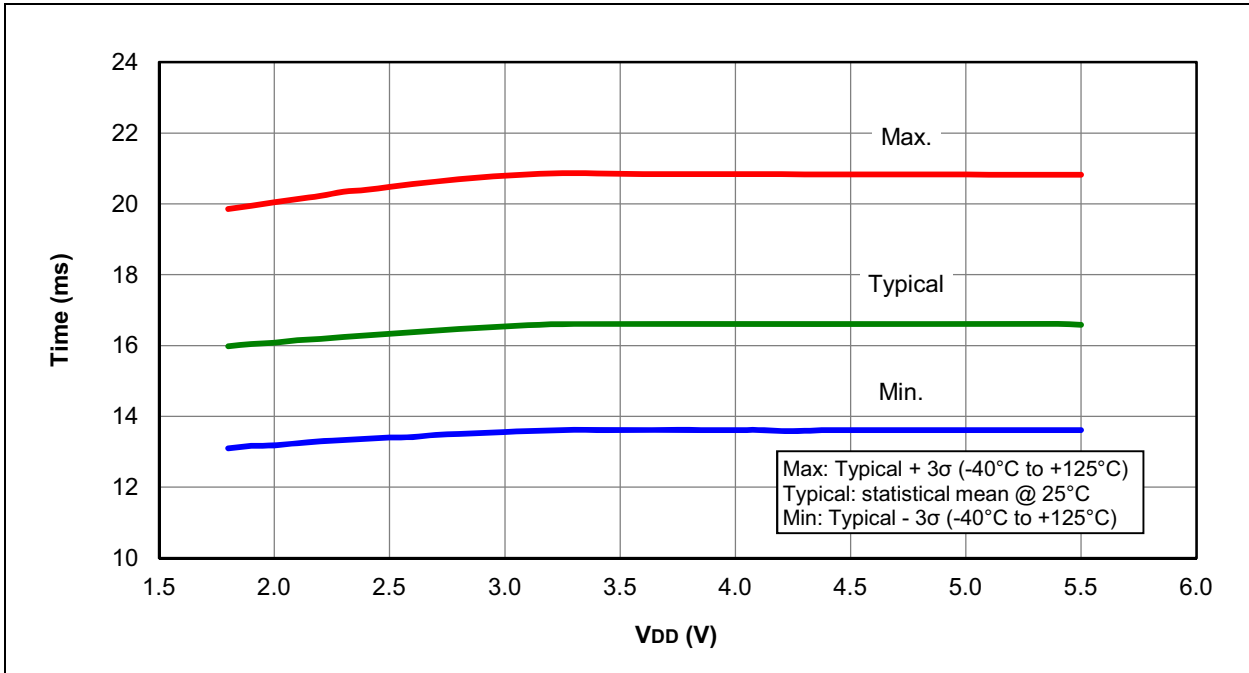
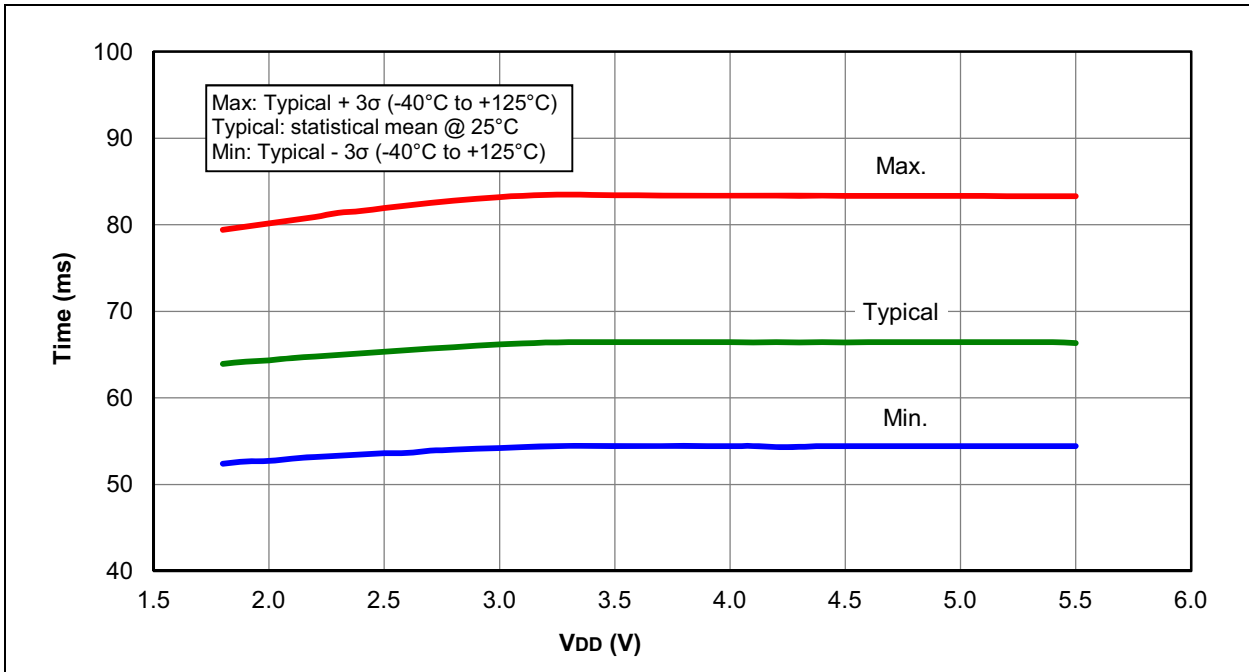


FIGURE 26-58: PWRT PERIOD



PIC16(L)F1526/7

FIGURE 26-59: FVR STABILIZATION PERIOD

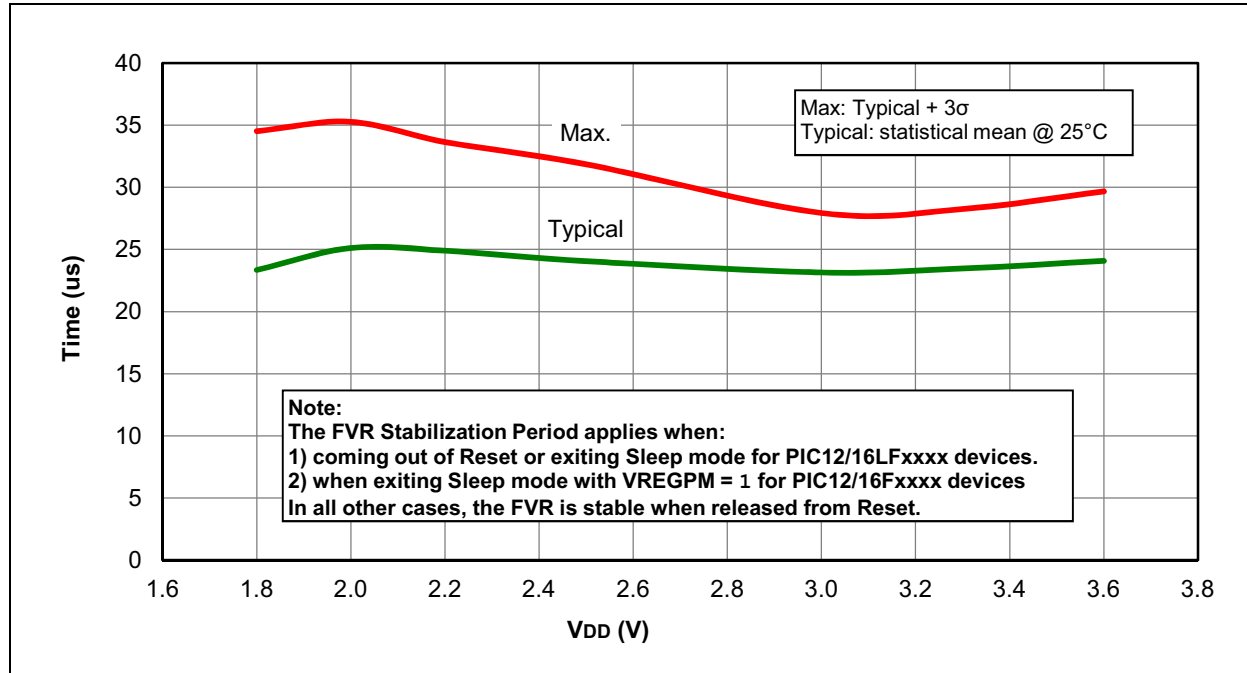


FIGURE 26-60: LFINTOSC FREQUENCY OVER V_{DD} AND TEMPERATURE, PIC16LF1526 ONLY

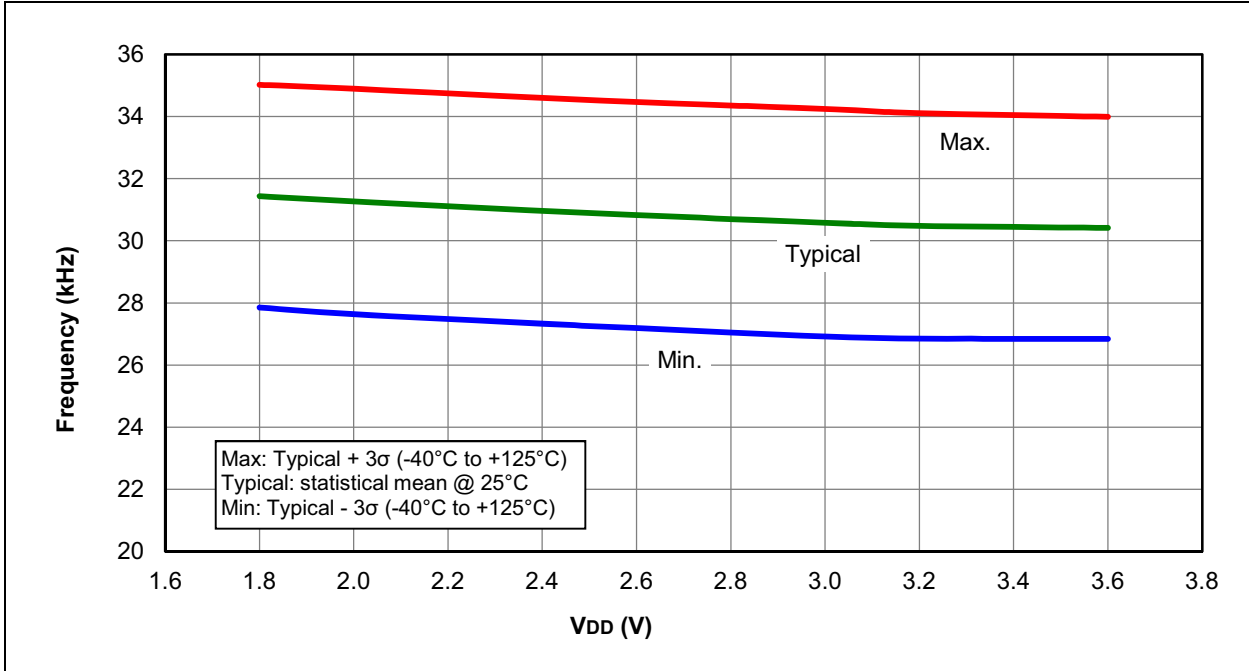
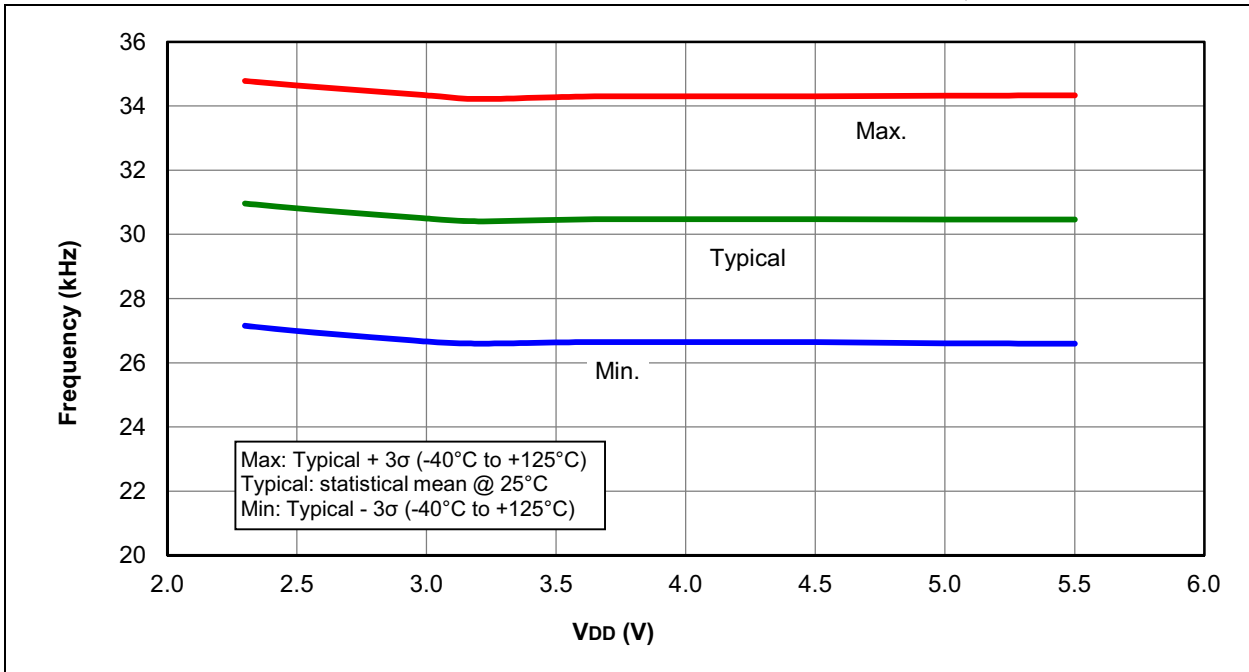


FIGURE 26-61: LFINTOSC FREQUENCY OVER V_{DD} AND TEMPERATURE, PIC16F1526/7 ONLY



PIC16(L)F1526/7

FIGURE 26-62: SLEEP MODE, WAKE PERIOD WITH HFINTOSC SOURCE, PIC16LF1526/7 ONLY

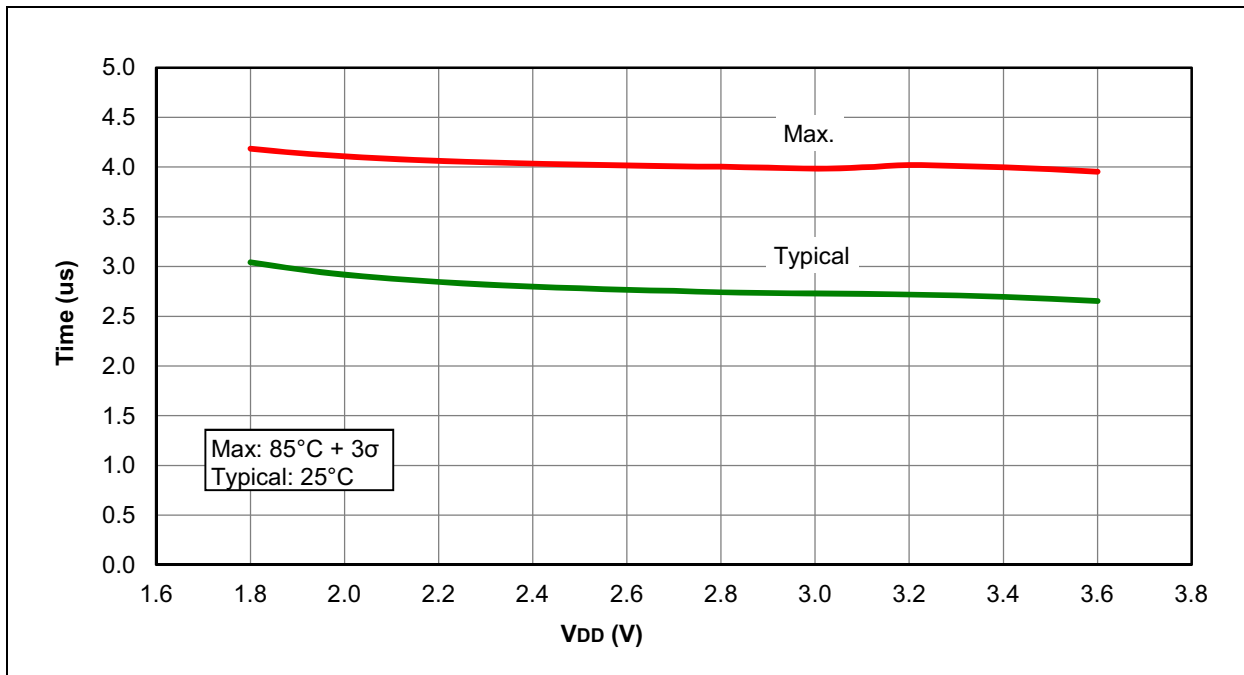


FIGURE 26-63: LOW-POWER SLEEP MODE, WAKE PERIOD WITH HFINTOSC SOURCE, VREGPM = 1, PIC16F1526/7 ONLY

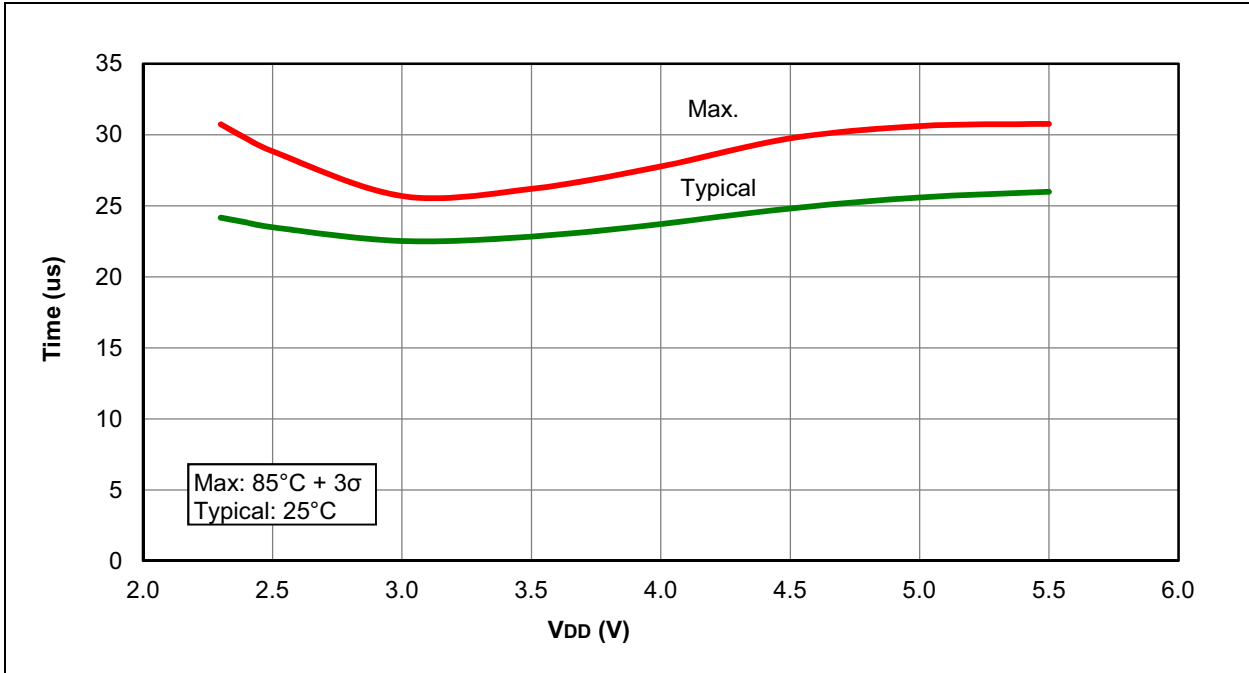
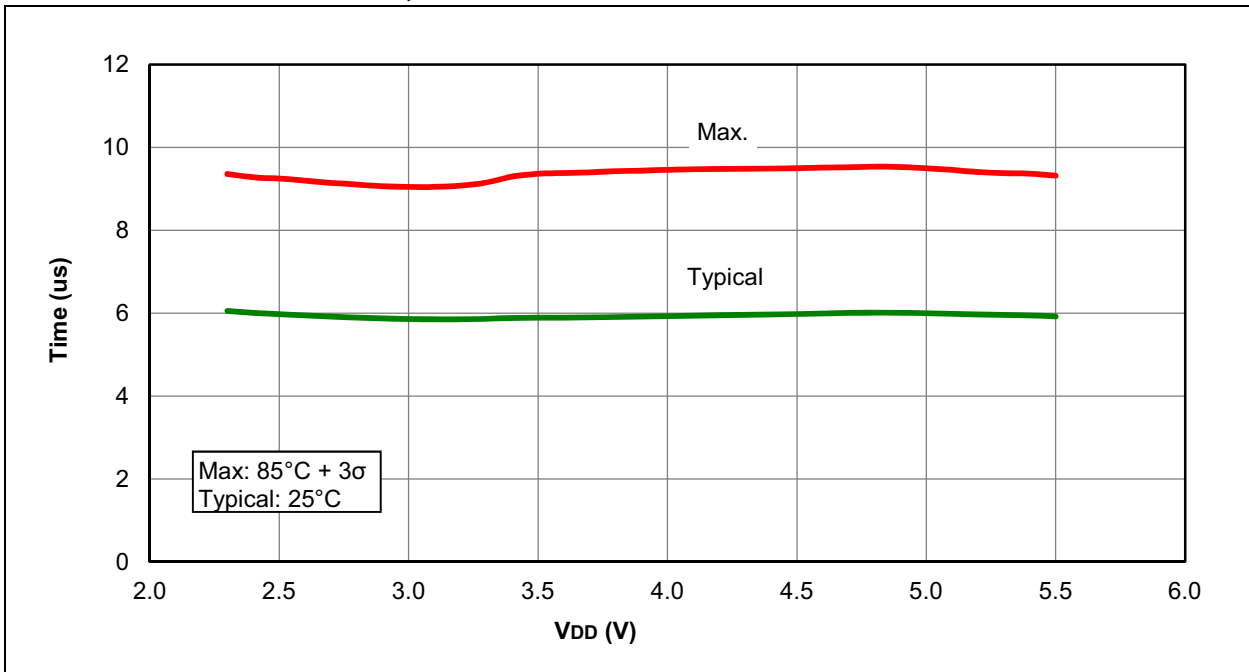


FIGURE 26-64: SLEEP MODE, WAKE PERIOD WITH HFINTOSC SOURCE, VREGPM = 0, PIC16F1526/7 ONLY



27.0 DEVELOPMENT SUPPORT

The PIC[®] microcontrollers (MCU) and dsPIC[®] digital signal controllers (DSC) are supported with a full range of software and hardware development tools:

- Integrated Development Environment
 - MPLAB[®] X IDE Software
- Compilers/Assemblers/Linkers
 - MPLAB XC Compiler
 - MPASM[™] Assembler
 - MPLINK[™] Object Linker/
MPLIB[™] Object Librarian
 - MPLAB Assembler/Linker/Librarian for
Various Device Families
- Simulators
 - MPLAB X SIM Software Simulator
- Emulators
 - MPLAB REAL ICE[™] In-Circuit Emulator
- In-Circuit Debuggers/Programmers
 - MPLAB ICD 3
 - PICKit[™] 3
- Device Programmers
 - MPLAB PM3 Device Programmer
- Low-Cost Demonstration/Development Boards,
Evaluation Kits and Starter Kits
- Third-party development tools

27.1 MPLAB X Integrated Development Environment Software

The MPLAB X IDE is a single, unified graphical user interface for Microchip and third-party software, and hardware development tool that runs on Windows[®], Linux and Mac OS[®] X. Based on the NetBeans IDE, MPLAB X IDE is an entirely new IDE with a host of free software components and plug-ins for high-performance application development and debugging. Moving between tools and upgrading from software simulators to hardware debugging and programming tools is simple with the seamless user interface.

With complete project management, visual call graphs, a configurable watch window and a feature-rich editor that includes code completion and context menus, MPLAB X IDE is flexible and friendly enough for new users. With the ability to support multiple tools on multiple projects with simultaneous debugging, MPLAB X IDE is also suitable for the needs of experienced users.

Feature-Rich Editor:

- Color syntax highlighting
- Smart code completion makes suggestions and provides hints as you type
- Automatic code formatting based on user-defined rules
- Live parsing

User-Friendly, Customizable Interface:

- Fully customizable interface: toolbars, toolbar buttons, windows, window placement, etc.
- Call graph window

Project-Based Workspaces:

- Multiple projects
- Multiple tools
- Multiple configurations
- Simultaneous debugging sessions

File History and Bug Tracking:

- Local file history feature
- Built-in support for Bugzilla issue tracker

27.2 MPLAB XC Compilers

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

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

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

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

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

27.3 MPASM Assembler

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

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

The MPASM Assembler features include:

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

27.4 MPLINK Object Linker/ MPLIB Object Librarian

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

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

The object linker/librarian 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

27.5 MPLAB Assembler, Linker and Librarian for Various Device Families

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

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

27.6 MPLAB X SIM Software Simulator

The MPLAB X SIM Software Simulator allows code development in a PC-hosted environment by simulating the PIC MCUs and dsPIC DSCs on an instruction level. On any given instruction, the data areas can be examined or modified and stimuli can be applied from a comprehensive stimulus controller. Registers can be logged to files for further run-time analysis. The trace buffer and logic analyzer display extend the power of the simulator to record and track program execution, actions on I/O, most peripherals and internal registers.

The MPLAB X SIM Software Simulator fully supports symbolic debugging using the MPLAB XC Compilers, and the MPASM and MPLAB Assemblers. The software simulator offers the flexibility to develop and debug code outside of the hardware laboratory environment, making it an excellent, economical software development tool.

27.7 MPLAB REAL ICE In-Circuit Emulator System

The MPLAB REAL ICE In-Circuit Emulator System is Microchip's next generation high-speed emulator for Microchip Flash DSC and MCU devices. It debugs and programs all 8, 16 and 32-bit MCU, and DSC devices with the easy-to-use, powerful graphical user interface of the MPLAB X IDE.

The emulator is connected to the design engineer's PC using a high-speed USB 2.0 interface and is connected to the target with either a connector compatible with in-circuit debugger systems (RJ-11) or with the new high-speed, noise tolerant, Low-Voltage Differential Signal (LVDS) interconnection (CAT5).

The emulator is field upgradable through future firmware downloads in MPLAB X IDE. MPLAB REAL ICE offers significant advantages over competitive emulators including full-speed emulation, run-time variable watches, trace analysis, complex breakpoints, logic probes, a ruggedized probe interface and long (up to three meters) interconnection cables.

27.8 MPLAB ICD 3 In-Circuit Debugger System

The MPLAB ICD 3 In-Circuit Debugger System is Microchip's most cost-effective, high-speed hardware debugger/programmer for Microchip Flash DSC and MCU devices. It debugs and programs PIC Flash microcontrollers and dsPIC DSCs with the powerful, yet easy-to-use graphical user interface of the MPLAB IDE.

The MPLAB ICD 3 In-Circuit Debugger probe is connected to the design engineer's PC using a high-speed USB 2.0 interface and is connected to the target with a connector compatible with the MPLAB ICD 2 or MPLAB REAL ICE systems (RJ-11). MPLAB ICD 3 supports all MPLAB ICD 2 headers.

27.9 PICkit 3 In-Circuit Debugger/Programmer

The MPLAB PICkit 3 allows debugging and programming of PIC and dsPIC Flash microcontrollers at a most affordable price point using the powerful graphical user interface of the MPLAB IDE. The MPLAB PICkit 3 is connected to the design engineer's PC using a full-speed USB interface and can be connected to the target via a Microchip debug (RJ-11) connector (compatible with MPLAB ICD 3 and MPLAB REAL ICE). The connector uses two device I/O pins and the Reset line to implement in-circuit debugging and In-Circuit Serial Programming™ (ICSP™).

27.10 MPLAB PM3 Device Programmer

The MPLAB PM3 Device Programmer is a universal, CE compliant device programmer with programmable voltage verification at VDDMIN and VDDMAX for maximum reliability. It features a large LCD display (128 x 64) for menus and error messages, and a modular, detachable socket assembly to support various package types. The ICSP cable assembly is included as a standard item. In Stand-Alone mode, the MPLAB PM3 Device Programmer can read, verify and program PIC devices without a PC connection. It can also set code protection in this mode. The MPLAB PM3 connects to the host PC via an RS-232 or USB cable. The MPLAB PM3 has high-speed communications and optimized algorithms for quick programming of large memory devices, and incorporates an MMC card for file storage and data applications.

27.11 Demonstration/Development Boards, Evaluation Kits, and Starter Kits

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

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

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

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

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

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

27.12 Third-Party Development Tools

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

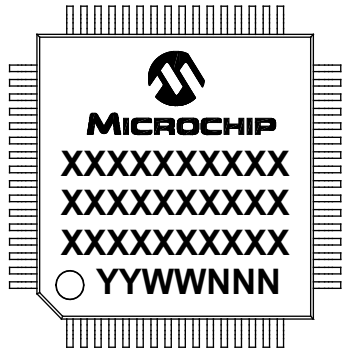
- Device Programmers and Gang Programmers from companies, such as SoftLog and CCS
- Software Tools from companies, such as Gimpel and Trace Systems
- Protocol Analyzers from companies, such as Saleae and Total Phase
- Demonstration Boards from companies, such as MikroElektronika, Digilent® and Olimex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

PIC16(L)F1526/7

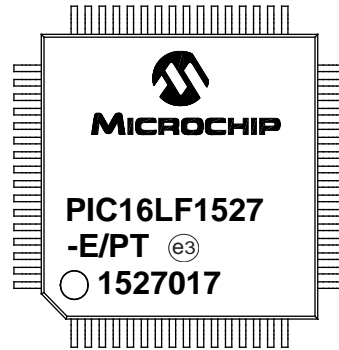
28.0 PACKAGING INFORMATION

28.1 Package Marking Information

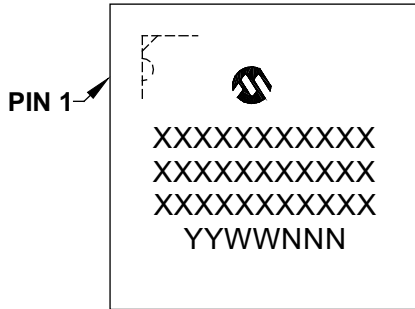
64-Lead TQFP (10x10x1 mm)



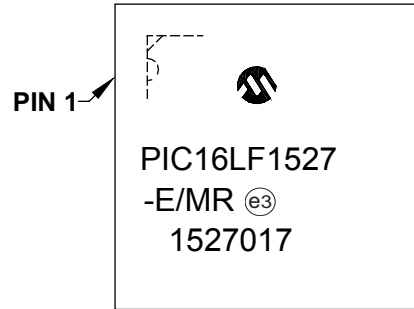
Example



64-Lead QFN (9x9x0.9 mm)



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 |
| | (e3) | Pb-free JEDEC® designator for Matte Tin (Sn) |
| | * | This package is Pb-free. The Pb-free JEDEC® designator (e3) can be found on the outer packaging for this package. |

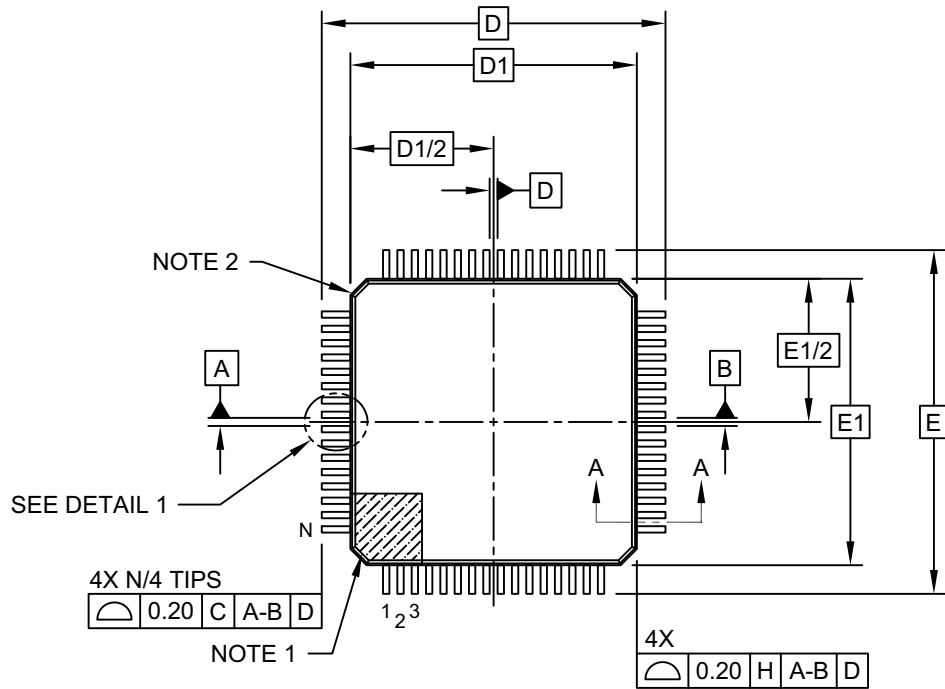
Note: In the event the full Microchip part number cannot be marked on one line, it will be carried over to the next line, thus limiting the number of available characters for customer-specific information.

28.2 Package Details

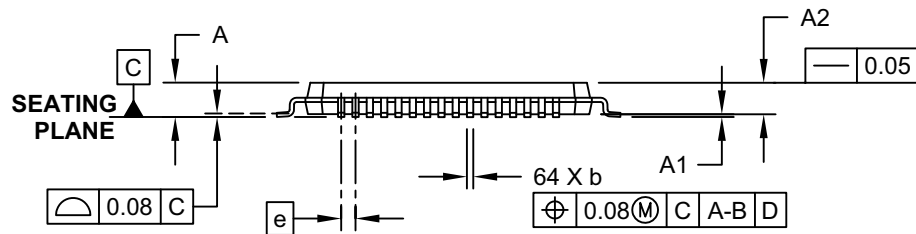
The following sections give the technical details of the packages.

64-Lead Plastic Thin Quad Flatpack (PT)-10x10x1 mm Body, 2.00 mm Footprint [TQFP]

Note: For the most current package drawings, please see the Microchip Packaging Specification located at <http://www.microchip.com/packaging>



TOP VIEW

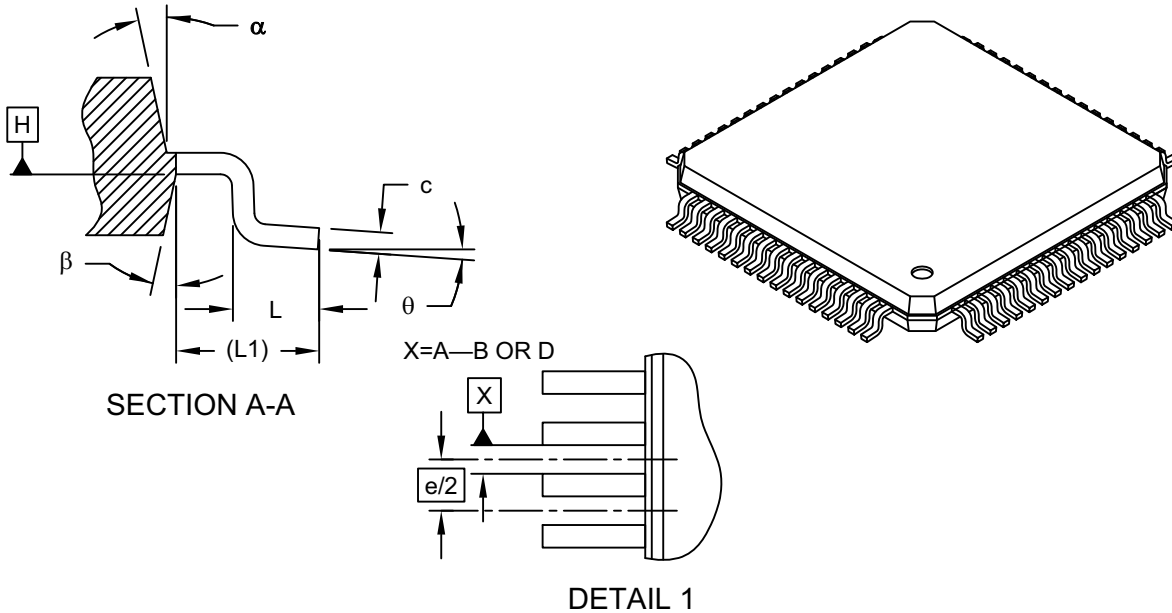


SIDE VIEW

PIC16(L)F1526/7

64-Lead Plastic Thin Quad Flatpack (PT)-10x10x1 mm Body, 2.00 mm Footprint [TQFP]

Note: For the most current package drawings, please see the Microchip Packaging Specification located at <http://www.microchip.com/packaging>



| Dimension Limits | Units | MILLIMETERS | | |
|--------------------------|----------|-------------|------|------|
| | | MIN | NOM | MAX |
| Number of Leads | N | 64 | | |
| Lead Pitch | e | 0.50 BSC | | |
| Overall Height | A | - | - | 1.20 |
| Molded Package Thickness | A2 | 0.95 | 1.00 | 1.05 |
| Standoff | A1 | 0.05 | - | 0.15 |
| Foot Length | L | 0.45 | 0.60 | 0.75 |
| Footprint | L1 | 1.00 REF | | |
| Foot Angle | ϕ | 0° | 3.5° | 7° |
| Overall Width | E | 12.00 BSC | | |
| Overall Length | D | 12.00 BSC | | |
| Molded Package Width | E1 | 10.00 BSC | | |
| Molded Package Length | D1 | 10.00 BSC | | |
| Lead Thickness | c | 0.09 | - | 0.20 |
| Lead Width | b | 0.17 | 0.22 | 0.27 |
| Mold Draft Angle Top | α | 11° | 12° | 13° |
| Mold Draft Angle Bottom | β | 11° | 12° | 13° |

Notes:

- Pin 1 visual index feature may vary, but must be located within the hatched area.
- Chamfers at corners are optional; size may vary.
- Dimensions D1 and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed 0.25mm per side.
- Dimensioning and tolerancing per ASME Y14.5M

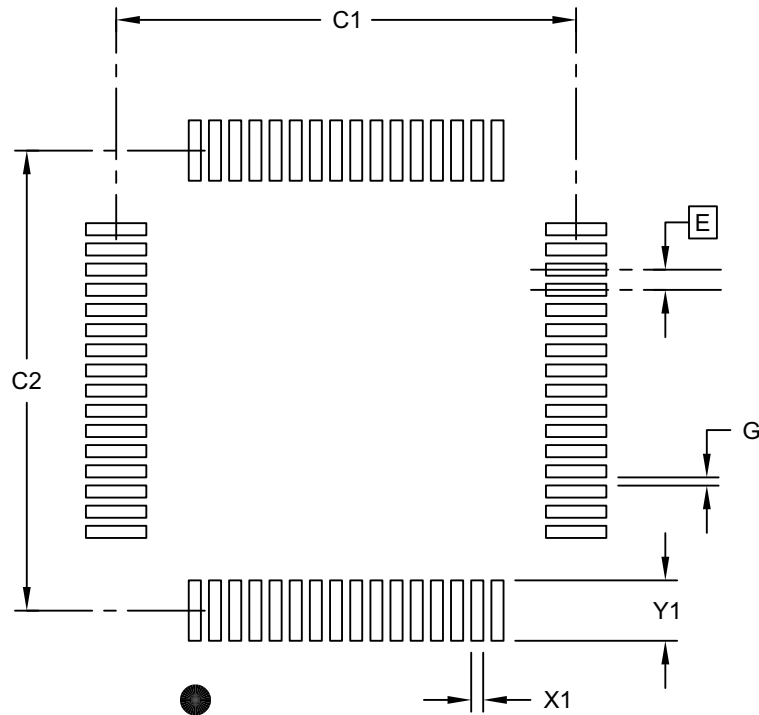
BSC: Basic Dimension. Theoretically exact value shown without tolerances.

REF: Reference Dimension, usually without tolerance, for information purposes only.

Microchip Technology Drawing C04-085C Sheet 2 of 2

64-Lead Plastic Thin Quad Flatpack (PT)-10x10x1 mm Body, 2.00 mm Footprint [TQFP]

Note: For the most current package drawings, please see the Microchip Packaging Specification located at <http://www.microchip.com/packaging>



RECOMMENDED LAND PATTERN

| Dimension Limits | Units | MILLIMETERS | | |
|--------------------------|-------|-------------|-------|------|
| | | MIN | NOM | MAX |
| Contact Pitch | E | 0.50 BSC | | |
| Contact Pad Spacing | C1 | | 11.40 | |
| Contact Pad Spacing | C2 | | 11.40 | |
| Contact Pad Width (X28) | X1 | | | 0.30 |
| Contact Pad Length (X28) | Y1 | | | 1.50 |
| Distance Between Pads | G | 0.20 | | |

Notes:

1. Dimensioning and tolerancing per ASME Y14.5M

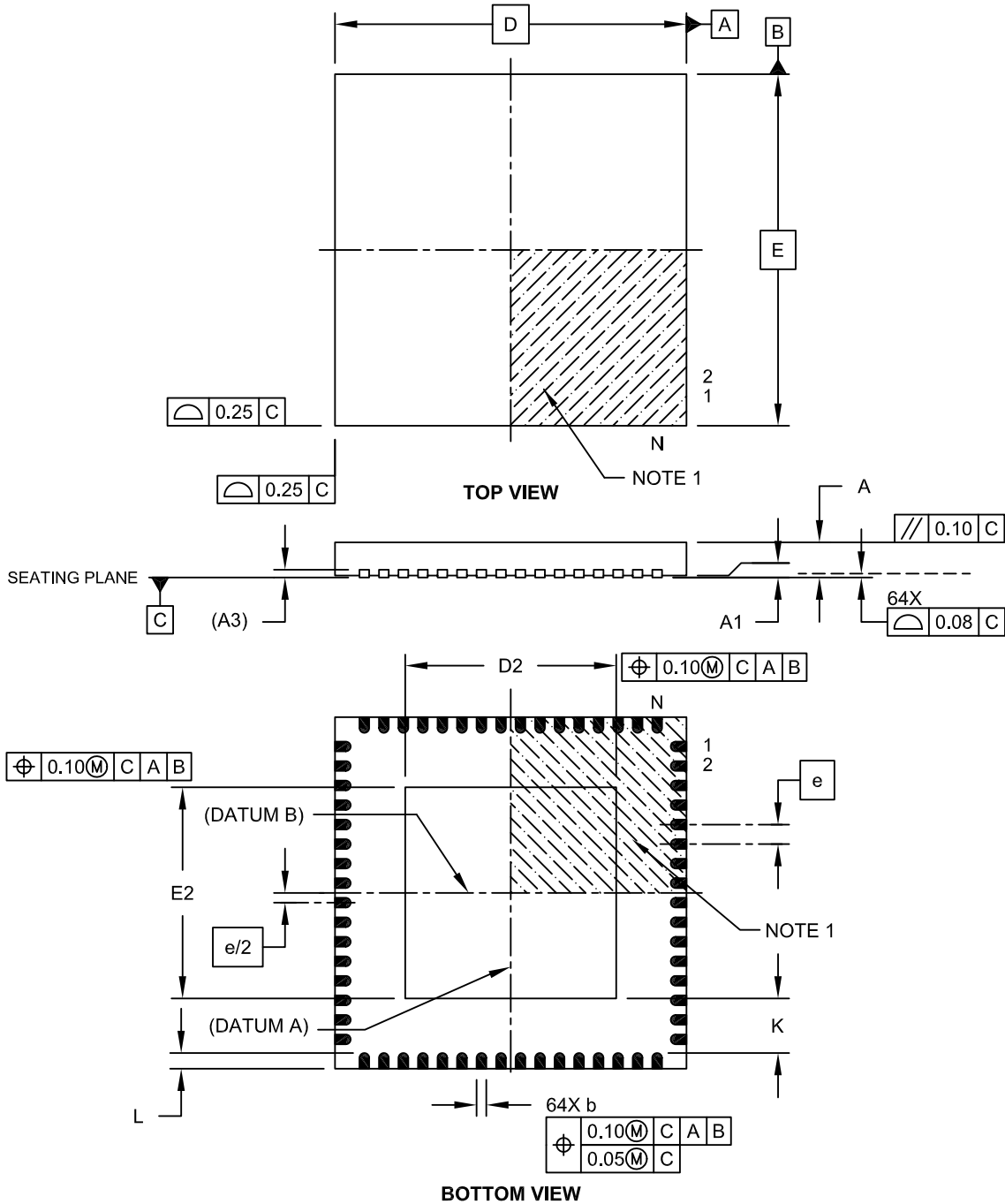
BSC: Basic Dimension. Theoretically exact value shown without tolerances.

Microchip Technology Drawing C04-2085B Sheet 1 of 1

PIC16(L)F1526/7

64-Lead Plastic Quad Flat, No Lead Package (MR) – 9x9x0.9 mm Body with 5.40 x 5.40 Exposed Pad [QFN]

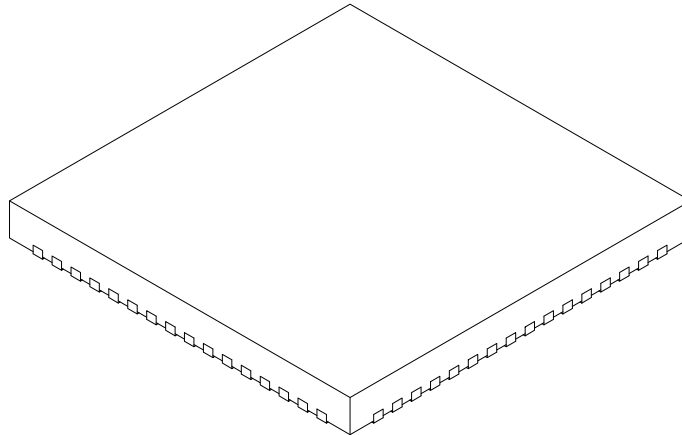
Note: For the most current package drawings, please see the Microchip Packaging Specification located at <http://www.microchip.com/packaging>



Microchip Technology Drawing C04-154A Sheet 1 of 2

64-Lead Plastic Quad Flat, No Lead Package (MR) – 9x9x0.9 mm Body with 5.40 x 5.40 Exposed Pad [QFN]

Note: For the most current package drawings, please see the Microchip Packaging Specification located at <http://www.microchip.com/packaging>



| | | Units | MILLIMETERS | | |
|------------------------|----|----------|-------------|------|-----|
| Dimension Limits | | | MIN | NOM | MAX |
| Number of Pins | N | | 64 | | |
| Pitch | e | | 0.50 BSC | | |
| Overall Height | A | 0.80 | 0.90 | 1.00 | |
| Standoff | A1 | 0.00 | 0.02 | 0.05 | |
| Contact Thickness | A3 | 0.20 REF | | | |
| Overall Width | E | 9.00 BSC | | | |
| Exposed Pad Width | E2 | 5.30 | 5.40 | 5.50 | |
| Overall Length | D | 9.00 BSC | | | |
| Exposed Pad Length | D2 | 5.30 | 5.40 | 5.50 | |
| Contact Width | b | 0.20 | 0.25 | 0.30 | |
| Contact Length | L | 0.30 | 0.40 | 0.50 | |
| Contact-to-Exposed Pad | K | 0.20 | - | - | |

Notes:

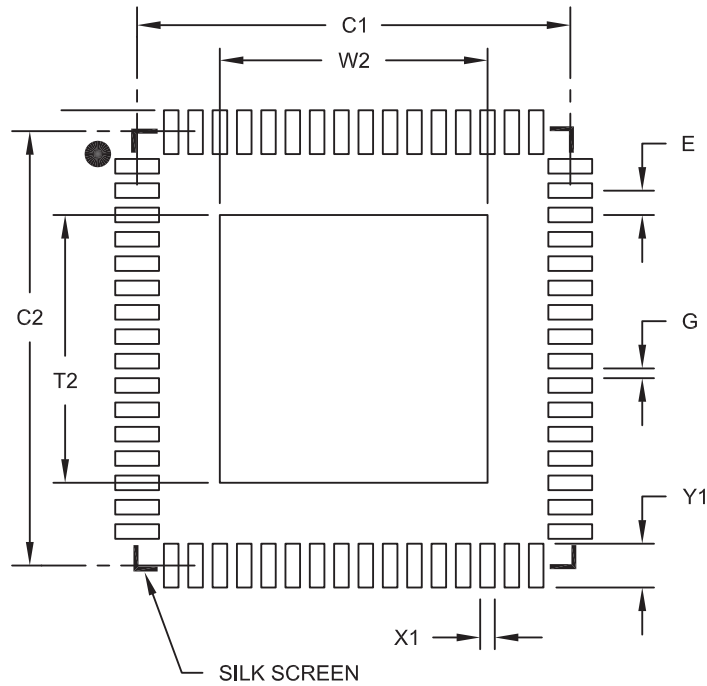
- Pin 1 visual index feature may vary, but must be located within the hatched area.
- Package is saw singulated.
- Dimensioning and tolerancing per ASME Y14.5M.
 BSC: Basic Dimension. Theoretically exact value shown without tolerances.
 REF: Reference Dimension, usually without tolerance, for information purposes only.

Microchip Technology Drawing C04-154A Sheet 2 of 2

PIC16(L)F1526/7

64-Lead Plastic Quad Flat, No Lead Package (MR) – 9x9x0.9 mm Body [QFN]
 With 0.40 mm Contact Length and 5.40x5.40mm Exposed Pad

Note: For the most current package drawings, please see the Microchip Packaging Specification located at <http://www.microchip.com/packaging>



RECOMMENDED LAND PATTERN

| Dimension Limits | Units | MILLIMETERS | | |
|----------------------------|-------|-------------|------|------|
| | | MIN | NOM | MAX |
| Contact Pitch | E | 0.50 BSC | | |
| Optional Center Pad Width | W2 | | | 5.50 |
| Optional Center Pad Length | T2 | | | 5.50 |
| Contact Pad Spacing | C1 | | 8.90 | |
| Contact Pad Spacing | C2 | | 8.90 | |
| Contact Pad Width (X64) | X1 | | | 0.30 |
| Contact Pad Length (X64) | Y1 | | | 0.85 |
| Distance Between Pads | G | 0.20 | | |

Notes:

1. Dimensioning and tolerancing per ASME Y14.5M

BSC: Basic Dimension. Theoretically exact value shown without tolerances.

Microchip Technology Drawing No. C04-2154A

APPENDIX A: DATA SHEET REVISION HISTORY

Revision A (01/2011)

Original release.

Revision B (05/2011)

Electrical Spec updates.

Revision C (01/2013)

Updated Electrical Spec and added Characterization Data Graphs.

Revision D (09/2015)

Updated chapters High-Performance RISC CPU, Device Overview, Memory Organization, Device Configuration, Enhanced Universal Synchronous Asynchronous Receiver Transmitter (EUSART), Packaging Information. Other minor corrections.

PIC16(L)F1526/7

PRODUCT IDENTIFICATION SYSTEM

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

| PART NO. | | [X]⁽¹⁾ | - | X | /XX | XXX |
|-------------------------------|--|---|----------|--------------------------|----------------|----------------|
| Device | | Tape and Reel Option | | Temperature Range | Package | Pattern |
| Device: | | PIC16F1526, PIC16LF1526 PIC16F1527, PIC16LF1527 | | | | |
| Tape and Reel Option: | | Blank = Standard packaging (tube or tray) T = Tape and Reel ⁽¹⁾ | | | | |
| Temperature Range: | | I = -40°C to +85°C (Industrial) E = -40°C to +125°C (Extended) | | | | |
| Package:⁽²⁾ | | MR = Plastic Quad Flat, no lead (QFN) PT = Plastic Thin Quad Flatpack (TQFP) | | | | |
| Pattern: | | QTP, SQTP, Code or Special Requirements (blank otherwise) | | | | |

Examples:

- a) PIC16F1526T - I/MR 301
Tape and Reel,
Industrial temperature,
QFN package,
QTP pattern #301
- b) PIC16F1527 - I/PT
Industrial temperature
TQFP package
- c) PIC16F1527 - E/MR
Extended temperature,
QFN package

Note 1: Tape and Reel identifier only appears in the catalog part number description. This identifier is used for ordering purposes and is not printed on the device package. Check with your Microchip Sales Office for package availability with the Tape and Reel option.

2: Small form-factor packaging options may be available. Please check www.microchip.com/packaging for small-form factor package availability, or contact your local Sales Office.

THE MICROCHIP WEB SITE

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

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

CUSTOMER CHANGE NOTIFICATION SERVICE

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

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

CUSTOMER SUPPORT

Users of Microchip products can receive assistance through several channels:

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

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

Technical support is available through the web site at: <http://www.microchip.com/support>

Note the following details of the code protection feature on Microchip devices:

- Microchip products meet the specification contained in their particular Microchip Data Sheet.
- Microchip believes that its family of products is one of the most secure families of its kind on the market today, when used in the intended manner and under normal conditions.
- There are dishonest and possibly illegal methods used to breach the code protection feature. All of these methods, to our knowledge, require using the Microchip products in a manner outside the operating specifications contained in Microchip's Data Sheets. Most likely, the person doing so is engaged in theft of intellectual property.
- Microchip is willing to work with the customer who is concerned about the integrity of their code.
- Neither Microchip nor any other semiconductor manufacturer can guarantee the security of their code. Code protection does not mean that we are guaranteeing the product as “unbreakable.”

Code protection is constantly evolving. We at Microchip are committed to continuously improving the code protection features of our products. Attempts to break Microchip's code protection feature may be a violation of the Digital Millennium Copyright Act. If such acts allow unauthorized access to your software or other copyrighted work, you may have a right to sue for relief under that Act.

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

Trademarks

The Microchip name and logo, the Microchip logo, dsPIC, FlashFlex, flexPWR, JukeBlox, KEELOQ, KEELOQ logo, Klear, LANCheck, MediaLB, MOST, MOST logo, MPLAB, OptoLyzer, PIC, PICSTART, PIC³² logo, RightTouch, SpyNIC, SST, SST Logo, SuperFlash and UNI/O are registered trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

The Embedded Control Solutions Company and mTouch are registered trademarks of Microchip Technology Incorporated in the U.S.A.

Analog-for-the-Digital Age, BodyCom, chipKIT, chipKIT logo, CodeGuard, dsPICDEM, dsPICDEM.net, ECAN, In-Circuit Serial Programming, ICSP, Inter-Chip Connectivity, KlearNet, KlearNet logo, MiWi, motorBench, MPASM, MPF, MPLAB Certified logo, MPLIB, MPLINK, MultiTRAK, NetDetach, Omniscient Code Generation, PICDEM, PICDEM.net, PICKit, PICTail, RightTouch logo, REAL ICE, SQI, Serial Quad I/O, Total Endurance, TSHARC, USBCheck, VariSense, ViewSpan, WiperLock, Wireless DNA, and ZENA are trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

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

Silicon Storage Technology is a registered trademark of Microchip Technology Inc. in other countries.

GestIC is a registered trademark of Microchip Technology Germany II GmbH & Co. KG, a subsidiary of Microchip Technology Inc., in other countries.

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

© 2011-2015, Microchip Technology Incorporated, Printed in the U.S.A., All Rights Reserved.

ISBN: 978-1-63277-823-9

QUALITY MANAGEMENT SYSTEM
CERTIFIED BY DNV
== ISO/TS 16949 ==

Microchip received ISO/TS-16949:2009 certification for its worldwide headquarters, design and wafer fabrication facilities in Chandler and Tempe, Arizona; Gresham, Oregon and design centers in California and India. The Company's quality system processes and procedures are for its PIC[®] MCUs and dsPIC[®] DSCs, KEELOQ[®] code hopping devices, Serial EEPROMs, microperipherals, nonvolatile memory and analog products. In addition, Microchip's quality system for the design and manufacture of development systems is ISO 9001:2000 certified.



MICROCHIP

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:

<http://www.microchip.com/support>

Web Address:

www.microchip.com

Atlanta

Duluth, GA

Tel: 678-957-9614

Fax: 678-957-1455

Austin, TX

Tel: 512-257-3370

Boston

Westborough, MA

Tel: 774-760-0087

Fax: 774-760-0088

Chicago

Itasca, IL

Tel: 630-285-0071

Fax: 630-285-0075

Cleveland

Independence, OH

Tel: 216-447-0464

Fax: 216-447-0643

Dallas

Addison, TX

Tel: 972-818-7423

Fax: 972-818-2924

Detroit

Novi, MI

Tel: 248-848-4000

Houston, TX

Tel: 281-894-5983

Indianapolis

Noblesville, IN

Tel: 317-773-8323

Fax: 317-773-5453

Los Angeles

Mission Viejo, CA

Tel: 949-462-9523

Fax: 949-462-9608

New York, NY

Tel: 631-435-6000

San Jose, CA

Tel: 408-735-9110

Canada - Toronto

Tel: 905-673-0699

Fax: 905-673-6509

ASIA/PACIFIC

Asia Pacific Office

Suites 3707-14, 37th Floor

Tower 6, The Gateway

Harbour City, Kowloon

Hong Kong

Tel: 852-2943-5100

Fax: 852-2401-3431

Australia - Sydney

Tel: 61-2-9868-6733

Fax: 61-2-9868-6755

China - Beijing

Tel: 86-10-8569-7000

Fax: 86-10-8528-2104

China - Chengdu

Tel: 86-28-8665-5511

Fax: 86-28-8665-7889

China - Chongqing

Tel: 86-23-8980-9588

Fax: 86-23-8980-9500

China - Dongguan

Tel: 86-769-8702-9880

China - Hangzhou

Tel: 86-571-8792-8115

Fax: 86-571-8792-8116

China - Hong Kong SAR

Tel: 852-2943-5100

Fax: 852-2401-3431

China - Nanjing

Tel: 86-25-8473-2460

Fax: 86-25-8473-2470

China - Qingdao

Tel: 86-532-8502-7355

Fax: 86-532-8502-7205

China - Shanghai

Tel: 86-21-5407-5533

Fax: 86-21-5407-5066

China - Shenyang

Tel: 86-24-2334-2829

Fax: 86-24-2334-2393

China - Shenzhen

Tel: 86-755-8864-2200

Fax: 86-755-8203-1760

China - Wuhan

Tel: 86-27-5980-5300

Fax: 86-27-5980-5118

China - Xian

Tel: 86-29-8833-7252

Fax: 86-29-8833-7256

ASIA/PACIFIC

China - Xiamen

Tel: 86-592-2388138

Fax: 86-592-2388130

China - Zhuhai

Tel: 86-756-3210040

Fax: 86-756-3210049

India - Bangalore

Tel: 91-80-3090-4444

Fax: 91-80-3090-4123

India - New Delhi

Tel: 91-11-4160-8631

Fax: 91-11-4160-8632

India - Pune

Tel: 91-20-3019-1500

Japan - Osaka

Tel: 81-6-6152-7160

Fax: 81-6-6152-9310

Japan - Tokyo

Tel: 81-3-6880-3770

Fax: 81-3-6880-3771

Korea - Daegu

Tel: 82-53-744-4301

Fax: 82-53-744-4302

Korea - Seoul

Tel: 82-2-554-7200

Fax: 82-2-558-5932 or

82-2-558-5934

Malaysia - Kuala Lumpur

Tel: 60-3-6201-9857

Fax: 60-3-6201-9859

Malaysia - Penang

Tel: 60-4-227-8870

Fax: 60-4-227-4068

Philippines - Manila

Tel: 63-2-634-9065

Fax: 63-2-634-9069

Singapore

Tel: 65-6334-8870

Fax: 65-6334-8850

Taiwan - Hsin Chu

Tel: 886-3-5778-366

Fax: 886-3-5770-955

Taiwan - Kaohsiung

Tel: 886-7-213-7828

Taiwan - Taipei

Tel: 886-2-2508-8600

Fax: 886-2-2508-0102

Thailand - Bangkok

Tel: 66-2-694-1351

Fax: 66-2-694-1350

EUROPE

Austria - Wels

Tel: 43-7242-2244-39

Fax: 43-7242-2244-393

Denmark - Copenhagen

Tel: 45-4450-2828

Fax: 45-4485-2829

France - Paris

Tel: 33-1-69-53-63-20

Fax: 33-1-69-30-90-79

Germany - Dusseldorf

Tel: 49-2129-3766400

Germany - Karlsruhe

Tel: 49-721-625370

Germany - Munich

Tel: 49-89-627-144-0

Fax: 49-89-627-144-44

Italy - Milan

Tel: 39-0331-742611

Fax: 39-0331-466781

Italy - Venice

Tel: 39-049-7625286

Netherlands - Drunen

Tel: 31-416-690399

Fax: 31-416-690340

Poland - Warsaw

Tel: 48-22-3325737

Spain - Madrid

Tel: 34-91-708-08-90

Fax: 34-91-708-08-91

Sweden - Stockholm

Tel: 46-8-5090-4654

UK - Wokingham



Tel: 44-118-921-5800

Fax: 44-118-921-5820

07/14/15

Looking for pricing, stock, or lifecycle information?

Click below to explore more details on WIN SOURCE:

-  [View PIC16LF1527-I/PT on WIN SOURCE](#)
-  [Microchip Technology](#) Information

Optimize Your Supply Chain with WIN SOURCE Solutions

-  Global Sourcing Solution
-  Obsolete Management
-  Cost Control Management
-  Shortage Management
-  Alternative Solution
-  Excess Inventory Management