
14/20-Pin, 8-Bit Flash Microcontroller

Description

PIC16(L)F1615/9 microcontrollers deliver on-chip features that are unique to the design for embedded control of small motors and general purpose applications in 14/20-pin count packages. Features like 10-bit A/D, CCP, 24-bit SMT and Zero-Cross Detection offer an excellent solution to the variety of applications. The product family also has a CRC+ memory scan and Windowed WDT to support safety-critical systems in home appliances, white goods and other end equipment.

Core Features

- C Compiler Optimized RISC Architecture
- Only 49 Instructions
- Operating Speed:
 - DC – 32 MHz clock input
 - 125 ns minimum instruction cycle
- Interrupt Capability
- 16-Level Deep Hardware Stack
- One 8-Bit Timer
- Four 16-bit Timers
- Low Current Power-on Reset (POR)
- Configurable Power-up Timer (PWRT)
- Brown-out Reset (BOR) with Selectable Trip Point
- Windowed Watchdog Timer (WWDT):
 - Variable prescaler selection
 - Variable window size selection
 - All sources configurable in hardware or software

Memory

- 8 KW Flash Program Memory
- 1024 Bytes Data SRAM
- Direct, Indirect and Relative Addressing modes
- High-Endurance Flash Data Memory (HEF):
 - 128 B of nonvolatile data storage
 - 100K erase/write cycles

Operating Characteristics

- Operating Voltage Range:
 - 1.8V to 3.6V (PIC16LF1615/9)
 - 2.3V to 5.5V (PIC16F1615/9)
- Temperature Range:
 - Industrial: -40°C to 85°C
 - Extended: -40°C to 125°C

eXtreme Low-Power (XLP) Features

- Sleep mode: 50 nA @ 1.8V, typical
- Watchdog Timer: 500 nA @ 1.8V, typical
- Operating Current:
 - 8 uA @ 32 kHz, 1.8V, typical
 - 32 uA/MHz @ 1.8V, typical

Digital Peripherals

- Configurable Logic Cell (CLC):
 - Four CLCs
 - Integrated combinational and sequential logic
- Complementary Waveform Generator (CWG):
 - Rising and falling edge dead-band control
 - Full-bridge, half-bridge, 1-channel drive
 - Multiple signal sources
- Two Capture/Compare/PWM (CCP) modules
- PWM: Two 10-bit Pulse-Width Modulators
- Two Signal Measurement Timers (SMT):
 - 24-bit timer/counter with prescaler
 - Multiple gate and clock inputs
- Angular Timer:
 - Single pulse
 - Multiple pulses with missing pulse recovery
- 8-Bit Timers (TMR2+HLT/4/6):
 - Up to 3 Timer2/4/6 with Hardware Limit Timer (HLT)
 - Monitors Fault Conditions: Stall, Stop, etc.
 - Multiple modes
 - 8-bit timer/counter with prescaler
 - 8-bit period register and postscaler
 - Asynchronous H/W Reset sources
- Math Accelerator with Proportional-Integral-Derivative (PID):
 - Four operation modes
 - Add and multiply
 - Simple multiplier
 - Multiply and Accumulate
 - Programmable PID controller
- Cyclic Redundancy Check with Memory Scan (CRC/SCAN):
 - Software configurable
- Serial Communications:
 - Enhanced USART (EUSART)
 - SPI, I²C, RS-232, RS-485, LIN compatible
 - Auto-Baud Detect, Auto-Wake-up on start

- Up to 17 I/O Pins and One Input-only Pin:
 - Individually programmable pull-ups
 - Slew rate control
 - Interrupt-on-change with edge-select
 - Two High Current Drive pins
- Peripheral Pin Select (PPS):
 - Enables pin mapping of digital I/O

Intelligent Analog Peripherals

- 10-Bit Analog-to-Digital Converter (ADC):
 - Up to 12 external channels
 - Conversion available during Sleep
- Two Comparators (COMP):
 - Low-Power/High-Speed mode
 - Up to three external inverting inputs
 - Fixed Voltage Reference at non-inverting input(s)
 - Comparator outputs externally accessible
- 8-Bit Digital-to-Analog Converter (DAC):
 - 8-bit resolution, rail-to-rail
 - Positive Reference Selection
- Voltage Reference:
 - Fixed Voltage Reference (FVR): 1.024V, 2.048V and 4.096V output levels
- Zero-Cross Detect (ZCD):
 - Detect when AC signal on pin crosses ground
- Two High-Current Drive Pins:
 - 100mA @ 5V

Clocking Structure

- 16 MHz Internal Oscillator:
 - $\pm 1\%$ at calibration
 - Selectable frequency range from 32 MHz to 31 kHz
- 31 kHz Low-Power Internal Oscillator
- 4x Phase-Locked Loop (PLL):
 - For up to 32 MHz internal operation
- External Oscillator Block with:
 - Three external clock modes up to 32 MHz
 - One crystal resonator mode up to 32 MHz
- Fail-Safe Clock Monitor:
 - Allows for safe shutdown if peripheral clock stops
- Two-Speed Oscillator Start-up
- Oscillator Start-up Timer (OST)

TABLE 1: PIC12/16(L)F161X FAMILY TYPES

| Device | Data Sheet Index | Program Memory Flash (W) | Program Memory Flash (kB) | Data SRAM (bytes) | High Endurance Flash (bytes) | I/O Pins | 8-bit Timer with HLT | 16-bit Timer | Angular Timer | Windowed Watchdog Timer | 24-bit SMT | Comparators | 10-bit ADC (ch) | Zero-Cross Detect | CCP/10-bit PWM | CWG | CLC | CRC with Memory Scan | Math Accelerator with PID | High-Current I/O 100mA | PPS | EUSART | I ² C/SPI |
|---------------|------------------|--------------------------|---------------------------|-------------------|------------------------------|----------|----------------------|--------------|---------------|-------------------------|------------|-------------|-----------------|-------------------|----------------|-----|-----|----------------------|---------------------------|------------------------|-----|--------|----------------------|
| PIC12(L)F1612 | (A) | 2048 | 3.5 | 256 | 256 | 6 | 4 | 1 | 0 | Y | 1 | 1 | 4 | 1 | 2/0 | 1 | 0 | Y | 0 | 0 | N | 0 | 0 |
| PIC16(L)F1613 | (A) | 2048 | 3.5 | 256 | 256 | 12 | 4 | 1 | 0 | Y | 2 | 2 | 8 | 1 | 2/0 | 1 | 0 | Y | 0 | 0 | N | 0 | 0 |
| PIC16(L)F1614 | (B) | 4096 | 7 | 512 | 128 | 12 | 4 | 3 | 1 | Y | 2 | 2 | 8 | 1 | 2/2 | 1 | 2 | Y | 1 | 2 | Y | 1 | 1 |
| PIC16(L)F1615 | (C) | 8192 | 14 | 1024 | 1024 | 12 | 4 | 3 | 1 | Y | 2 | 2 | 8 | 1 | 2/2 | 1 | 4 | Y | 1 | 2 | Y | 1 | 1 |
| PIC16(L)F1618 | (B) | 4096 | 7 | 512 | 128 | 18 | 4 | 3 | 1 | Y | 2 | 2 | 12 | 1 | 2/2 | 1 | 2 | Y | 1 | 2 | Y | 1 | 1 |
| PIC16(L)F1619 | (C) | 8192 | 14 | 1024 | 1024 | 18 | 4 | 3 | 1 | Y | 2 | 2 | 12 | 1 | 2/2 | 1 | 4 | Y | 1 | 2 | Y | 1 | 1 |

Note 1: Debugging Methods: (I) – Integrated on Chip; E – using Emulation Product

Data Sheet Index:

- A. DS40001737 [PIC12\(L\)F1612/16\(L\)F1613 Data Sheet, 8/14-Pin, 8-bit Flash Microcontrollers](#)
- B. DS40001769 [PIC16\(L\)F1614/8 Data Sheet, 14/20-Pin, 8-bit Flash Microcontrollers](#)
- C. DS40001770 [PIC16\(L\)F1615/9 Data Sheet, 14/20-Pin, 8-bit Flash Microcontrollers](#)

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

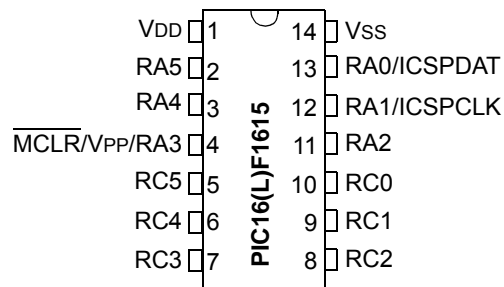
TABLE 2: PACKAGES

| Packages | PDIP | SOIC | DFN | UDFN | TSSOP | QFN | UQFN | SSOP |
|---------------|------|------|-----|------|-------|-----|------|------|
| PIC16(L)F1615 | • | • | | | • | • | | |
| PIC16(L)F1619 | • | • | | | | • | • | • |

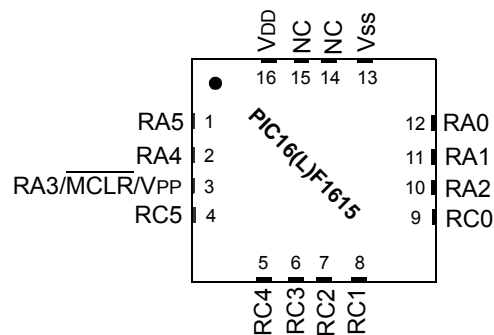
Note: Pin details are subject to change.

PIN DIAGRAMS

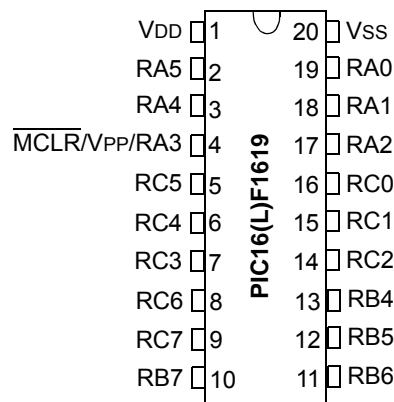
14-pin PDIP, SOIOC, TSSOP



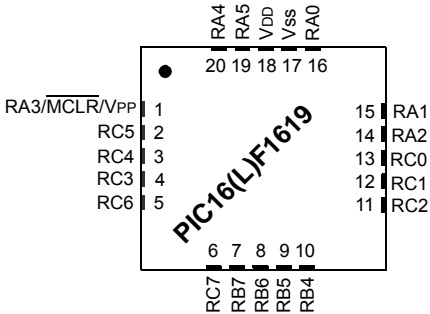
16-pin UQFN



20-pin PDIP, SOIC, SSOP



20-pin QFN, UQFN



PIN ALLOCATION TABLES

TABLE 3: 14/16-PIN ALLOCATION TABLE (PIC16(L)F1615)

| I/O | 14-Pin PDIP, SOIC, TSSOP | 16-Pin UQFN | A/D | Reference | Comparator | Timers | CCP | CWG | ZCD | CLC | EUSART | SMT | Angular Timer | MSSP | PWM | High Current I/O | Interrupt | Pull-up | Basic |
|--------------------|--------------------------|-------------|-----|-----------|------------------|---|---------------------|-----------------------|---------|-----------------------|---------------------|------------------------|---|----------------------|---------|------------------|------------|---------|----------|
| RA0 | 13 | 12 | AN0 | DAC1OUT1 | C1IN+ | — | — | — | — | — | — | — | — | — | — | — | IOC | Y | ICSPDAT |
| RA1 | 12 | 11 | AN1 | VREF+ | C1IN0- C2IN0- | — | — | — | — | — | — | — | — | — | — | — | IOC | Y | ICSPCLK |
| RA2 | 11 | 10 | AN2 | — | — | T0CKI ⁽¹⁾ | — | CWG1IN ⁽¹⁾ | ZCD1IN | — | — | — | — | — | — | — | INT IOC | Y | — |
| RA3 | 4 | 3 | — | — | — | T6IN ⁽¹⁾ | — | — | — | — | — | SMTWIN2 ⁽¹⁾ | — | — | — | — | IOC | Y | MCLR/VPP |
| RA4 | 3 | 2 | AN3 | — | — | T1G ⁽¹⁾ | — | — | — | — | — | SMTSIG1 ⁽¹⁾ | — | — | — | — | IOC | Y | CLKOUT |
| RA5 | 2 | 1 | — | — | — | T1CKI ⁽¹⁾ T2IN ⁽¹⁾ | — | — | — | — | — | SMTWIN1 ⁽¹⁾ | — | — | — | — | IOC | Y | CLKIN |
| RC0 | 10 | 9 | AN4 | — | C2IN+ | T5CKI ⁽¹⁾ | — | — | — | — | — | — | — | SCK ^(1,3) | — | — | IOC | Y | — |
| RC1 | 9 | 8 | AN5 | — | C1IN1- C2IN1- | T4IN ⁽¹⁾ | — | — | — | — | — | SMTSIG2 ⁽¹⁾ | — | SDI ⁽¹⁾ | — | — | IOC | Y | — |
| RC2 | 8 | 7 | AN6 | — | C1IN2- C2IN2- | — | — | — | — | — | — | — | — | — | — | — | IOC | Y | — |
| RC3 | 7 | 6 | AN7 | — | C1IN3- C2IN3- | T5G ⁽¹⁾ | CCP2 ⁽¹⁾ | — | — | CLCIN0 ⁽¹⁾ | — | — | ATCC1 ⁽¹⁾ | SS ⁽¹⁾ | — | — | IOC | Y | — |
| RC4 | 6 | 5 | — | — | — | T3G ⁽¹⁾ | — | — | — | GLCIN1 ⁽¹⁾ | CK ⁽¹⁾ | — | ATCC2 ⁽¹⁾ | — | — | HIC4 | IOC | Y | — |
| RC5 | 5 | 4 | — | — | — | T3CKI ⁽¹⁾ | CCP1 ⁽¹⁾ | — | — | — | RX ^(1,3) | — | ATIN ⁽¹⁾ ATCC3 ⁽¹⁾ | — | — | HIC5 | IOC | Y | — |
| VDD | 1 | 16 | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | VDD |
| VSS | 14 | 13 | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | VSS |
| OUT ⁽²⁾ | — | — | — | — | C1OUT | — | CCP1 | CWG1A | ZCD1OUT | CLC1OUT | DT ⁽³⁾ | — | — | SDO | PWM3OUT | — | — | — | — |
| | — | — | — | — | C2OUT | — | CCP2 | CWG1B | — | CLC2OUT | CK | — | — | SCK ⁽³⁾ | PWM4OUT | — | — | — | — |
| | — | — | — | — | — | — | — | CWG1C | — | — | TX | — | — | — | — | — | — | — | — |
| | — | — | — | — | — | — | — | CWG1D | — | — | — | — | — | — | — | — | — | — | — |

- Note**
- 1: Default peripheral input. Input can be moved to any other pin with the PPS input selection registers.
 - 2: All pin outputs default to PORT latch data. Any pin can be selected as a digital peripheral output with the PPS output selection registers.
 - 3: These peripheral functions are bidirectional. The output pin selections must be the same as the input pin selections.

TABLE 4: 20-PIN ALLOCATION TABLE (PIC16(L)F1619)

| I/O | 20-Pin PDIP, SOIC, SSOP | 20-Pin UQFN | A/D | Reference | Comparator | Timers | CCP | CWG | ZCD | CLC | EUSART | SMT | Angular Timer | MSSP | PWM | High Current I/O | Interrupt | Pull-up | Basic |
|-----|-------------------------|-------------|------|-----------|------------------|---|---------------------|-----------------------|--------|-----------------------|---------------------|------------------------|---------------------|----------------------|-----|------------------|------------|---------|-------------|
| RA0 | 19 | 16 | AN0 | DAC1OUT | C1IN+ | — | — | — | — | — | — | — | — | — | — | — | IOC | Y | ICSPDAT |
| RA1 | 18 | 15 | AN1 | VREF+ | C1IN0- C2IN0- | — | — | — | — | — | — | — | — | — | — | — | IOC | Y | ICSPCLK |
| RA2 | 17 | 14 | AN2 | — | — | T0CKI ⁽¹⁾ | — | CWG1IN ⁽¹⁾ | ZCD1IN | — | — | — | — | — | — | — | INT IOC | Y | — |
| RA3 | 4 | 1 | — | — | — | T6IN ⁽¹⁾ | — | — | — | — | — | SMTWIN2 ⁽¹⁾ | — | — | — | — | IOC | Y | MCLR VPP |
| RA4 | 3 | 20 | AN3 | — | — | T1G ⁽¹⁾ | — | — | — | — | — | SMTSIG1 ⁽¹⁾ | — | — | — | — | IOC | Y | CLKOUT |
| RA5 | 2 | 19 | — | — | — | T1CKI ⁽¹⁾ T2IN ⁽¹⁾ | — | — | — | CLCIN3 ⁽¹⁾ | — | SMTWIN1 ⁽¹⁾ | — | — | — | — | IOC | Y | CLKIN |
| RB4 | 13 | 10 | AN10 | — | — | — | — | — | — | — | — | — | — | SDI ⁽¹⁾ | — | — | IOC | Y | — |
| RB5 | 12 | 9 | AN11 | — | — | — | — | — | — | — | RX ^(1,3) | — | — | — | — | — | IOC | Y | — |
| RB6 | 11 | 8 | — | — | — | — | — | — | — | — | — | — | — | SCK ^(1,3) | — | — | IOC | Y | — |
| RB7 | 10 | 7 | — | — | — | — | — | — | — | — | CK ⁽¹⁾ | — | — | — | — | — | IOC | Y | — |
| RC0 | 16 | 13 | AN4 | — | C2IN+ | T5CKI ⁽¹⁾ | — | — | — | — | — | — | — | — | — | — | IOC | Y | — |
| RC1 | 15 | 12 | AN5 | — | C1IN1- C2IN1- | T4IN ⁽¹⁾ | — | — | — | CLCIN2 ⁽²⁾ | — | SMTSIG2 ⁽¹⁾ | — | — | — | — | IOC | Y | — |
| RC2 | 14 | 11 | AN6 | — | C1IN2- C2IN2- | — | — | — | — | — | — | — | — | — | — | — | IOC | Y | — |
| RC3 | 7 | 4 | AN7 | — | C1IN3- C2IN3- | T5G ⁽¹⁾ | CCP2 ⁽¹⁾ | — | — | CLCIN0 ⁽¹⁾ | — | — | ATCC ⁽¹⁾ | — | — | — | IOC | Y | — |
| RC4 | 6 | 3 | — | — | — | T3G ⁽¹⁾ | — | — | — | CLCIN1 ⁽¹⁾ | — | — | — | — | — | HIC4 | IOC | Y | — |
| RC5 | 5 | 2 | — | — | — | T3CKI ⁽¹⁾ | CCP1 ⁽¹⁾ | — | — | — | — | — | ATIN ⁽¹⁾ | — | — | HIC5 | IOC | Y | — |
| RC6 | 8 | 5 | AN8 | — | — | — | — | — | — | — | — | — | — | SS ⁽¹⁾ | — | — | IOC | Y | — |
| RC7 | 9 | 6 | AN9 | — | — | — | — | — | — | — | — | — | — | — | — | — | IOC | Y | — |
| VDD | 1 | 18 | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| VSS | 20 | 17 | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — |

- Note**
- 1: Default peripheral input. Input can be moved to any other pin with the PPS input selection registers.
 - 2: All pin outputs default to PORT latch data. Any pin can be selected as a digital peripheral output with the PPS output selection registers.
 - 3: These peripheral functions are bidirectional. The output pin selections must be the same as the input pin selections.

TABLE 4: 20-PIN ALLOCATION TABLE (PIC16(L)F1619)

| I/O | 20-Pin PDIP, SOIC, SSOP | 20-Pin UQFN | A/D | Reference | Comparator | Timers | CCP | CWG | ZCD | CLC | EUSART | SMT | Angular Timer | MSSP | PWM | High Current I/O | Interrupt | Pull-up | Basic |
|--------------------|-------------------------|-------------|-----|-----------|------------|--------|------|-------|---------|---------|-------------------|-----|---------------|--------------------|---------|------------------|-----------|---------|-------|
| OUT ⁽²⁾ | — | — | — | — | C1OUT | — | CCP1 | CWG1A | ZCD1OUT | CLC1OUT | DT ⁽³⁾ | — | — | SDO | PWM3OUT | — | — | — | — |
| | — | — | — | — | C2OUT | — | CCP2 | CWG1B | — | CLC2OUT | CK | — | — | SCK ⁽³⁾ | PWM4OUT | — | — | — | — |
| | — | — | — | — | — | — | — | CWG1C | — | CLC3OUT | TX | — | — | — | — | — | — | — | — |
| | — | — | — | — | — | — | — | CWG1D | — | CLC4OUT | — | — | — | — | — | — | — | — | — |

- Note**
- 1: Default peripheral input. Input can be moved to any other pin with the PPS input selection registers.
 - 2: All pin outputs default to PORT latch data. Any pin can be selected as a digital peripheral output with the PPS output selection registers.
 - 3: These peripheral functions are bidirectional. The output pin selections must be the same as the input pin selections.

TABLE OF CONTENTS

| | |
|---|-----|
| Device Overview | 11 |
| Enhanced Mid-Range CPU | 19 |
| Memory Organization | 21 |
| Device Configuration | 66 |
| Oscillator Module | 73 |
| Resets | 84 |
| Interrupts | 92 |
| Power-Down Mode (Sleep) | 109 |
| Windowed Watchdog Timer (WDT) | 112 |
| Flash Program Memory Control | 120 |
| Cyclic Redundancy Check (CRC) Module | 136 |
| I/O Ports | 148 |
| Peripheral Pin Select (PPS) Module | 170 |
| Interrupt-On-Change | 178 |
| Fixed Voltage Reference (FVR) | 184 |
| Temperature Indicator Module | 187 |
| Analog-to-Digital Converter (ADC) Module | 189 |
| 8-bit Digital-to-Analog Converter (DAC1) Module | 203 |
| Comparator Module | 207 |
| Zero-Cross Detection (ZCD) Module | 215 |
| Timer0 Module | 221 |
| Timer1/3/5 Module with Gate Control | 224 |
| Timer2/4/6 Module | 235 |
| Master Synchronous Serial Port (MSSP) Module | 260 |
| Enhanced Universal Synchronous Asynchronous Receiver Transmitter (EUSART) | 313 |
| Capture/Compare/PWM Modules | 345 |
| Pulse-Width Modulation (PWM) Module | 359 |
| Complementary Waveform Generator (CWG) Module | 365 |
| Signal Measurement Timer (SMT) | 391 |
| Configurable Logic Cell (CLC) | 437 |
| Angular Timer (AT) Module | 451 |
| Math Accelerator with Proportional-Integral-Derivative (PID) Module | 480 |
| In-Circuit Serial Programming™ (ICSP™) | 496 |
| Instruction Set Summary | 498 |
| Electrical Specifications | 512 |
| DC and AC Characteristics Graphs and Charts | 536 |
| Development Support | 555 |
| Packaging Information | 559 |
| Data Sheet Revision History | 583 |

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 Website 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 Website; <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 website at www.microchip.com to receive the most current information on all of our products.

1.0 DEVICE OVERVIEW

The PIC16(L)F1615/9 are described within this data sheet. The block diagram of these devices are shown in [Figure 1-1](#), the available peripherals are shown in [Table 1-1](#), and the pin out descriptions are shown in [Tables 1-2](#) and [1-3](#).

TABLE 1-1: DEVICE PERIPHERAL SUMMARY

| Peripheral | | PIC16(L)F1615 | PIC16(L)F1619 |
|---|--------|---------------|---------------|
| Analog-to-Digital Converter (ADC) | | • | • |
| Complementary Wave Generator (CWG) | | • | • |
| Cyclic Redundancy Check (CRC) | | • | • |
| Digital-to-Analog Converter (DAC) | | • | • |
| Enhanced Universal Synchronous/Asynchronous Receiver/Transmitter (EUSART) | | • | • |
| Fixed Voltage Reference (FVR) | | • | • |
| Temperature Indicator | | • | • |
| Windowed Watchdog Timer (WDT) | | • | • |
| Zero-Cross Detection (ZCD) | | • | • |
| Capture/Compare/PWM (CCP) Modules | | | |
| | CCP1 | • | • |
| | CCP2 | • | • |
| Comparators | | | |
| | C1 | • | • |
| | C2 | • | • |
| Configurable Logic Cell (CLC) | | | |
| | CLC1 | • | • |
| | CLC2 | • | • |
| | CLC3 | • | • |
| | CLC4 | • | • |
| Master Synchronous Serial Ports | | | |
| | MSSP1 | • | • |
| Pulse-Width Modulator (PWM) | | | |
| | PWM3 | • | • |
| | PWM4 | • | • |
| Signal Measurement Timer (SMT) | | | |
| | SMT1 | • | • |
| | SMT2 | • | • |
| Timers | | | |
| | Timer0 | • | • |
| | Timer1 | • | • |
| | Timer2 | • | • |
| | Timer3 | • | • |
| | Timer4 | • | • |
| | Timer5 | • | • |
| | Timer6 | • | • |

1.1 Register and Bit Naming Conventions

1.1.1 REGISTER NAMES

When there are multiple instances of the same peripheral in a device, the peripheral control registers will be depicted as the concatenation of a peripheral identifier, peripheral instance, and control identifier. The control registers section will show just one instance of all the register names with an 'x' in the place of the peripheral instance number. This naming convention may also be applied to peripherals when there is only one instance of that peripheral in the device to maintain compatibility with other devices in the family that contain more than one.

1.1.2 BIT NAMES

There are two variants for bit names:

- Short name: Bit function abbreviation
- Long name: Peripheral abbreviation + short name

1.1.2.1 Short Bit Names

Short bit names are an abbreviation for the bit function. For example, some peripherals are enabled with the EN bit. The bit names shown in the registers are the short name variant.

Short bit names are useful when accessing bits in C programs. The general format for accessing bits by the short name is *RegisterName*bits.*ShortName*. For example, the enable bit, EN, in the COG1CON0 register can be set in C programs with the instruction `COG1CON0bits.EN = 1`.

Short names are generally not useful in assembly programs because the same name may be used by different peripherals in different bit positions. When this occurs, during the include file generation, all instances of that short bit name are appended with an underscore plus the name of the register in which the bit resides to avoid naming contentions.

1.1.2.2 Long Bit Names

Long bit names are constructed by adding a peripheral abbreviation prefix to the short name. The prefix is unique to the peripheral, thereby making every long bit name unique. The long bit name for the COG1 enable bit is the COG1 prefix, G1, appended with the enable bit short name, EN, resulting in the unique bit name G1EN.

Long bit names are useful in both C and assembly programs. For example, in C the COG1CON0 enable bit can be set with the `G1EN = 1` instruction. In assembly, this bit can be set with the `BSF COG1CON0,G1EN` instruction.

1.1.2.3 Bit Fields

Bit fields are two or more adjacent bits in the same register. Bit fields adhere only to the short bit naming convention. For example, the three Least Significant bits of the COG1CON0 register contain the mode control bits. The short name for this field is MD. There is no long bit name variant. Bit field access is only possible in C programs. The following example demonstrates a C program instruction for setting the COG1 to the Push-Pull mode:

```
COG1CON0bits.MD = 0x5;
```

Individual bits in a bit field can also be accessed with long and short bit names. Each bit is the field name appended with the number of the bit position within the field. For example, the Most Significant mode bit has the short bit name MD2 and the long bit name is G1MD2. The following two examples demonstrate assembly program sequences for setting the COG1 to Push-Pull mode:

Example 1:

```
MOVLW ~(1<<G1MD1)
ANDWF COG1CON0,F
MOVLW 1<<G1MD2 | 1<<G1MD0
IORWF COG1CON0,F
```

Example 2:

```
BSF COG1CON0,G1MD2
BCF COG1CON0,G1MD1
BSF COG1CON0,G1MD0
```

1.1.3 REGISTER AND BIT NAMING EXCEPTIONS

1.1.3.1 Status, Interrupt, and Mirror Bits

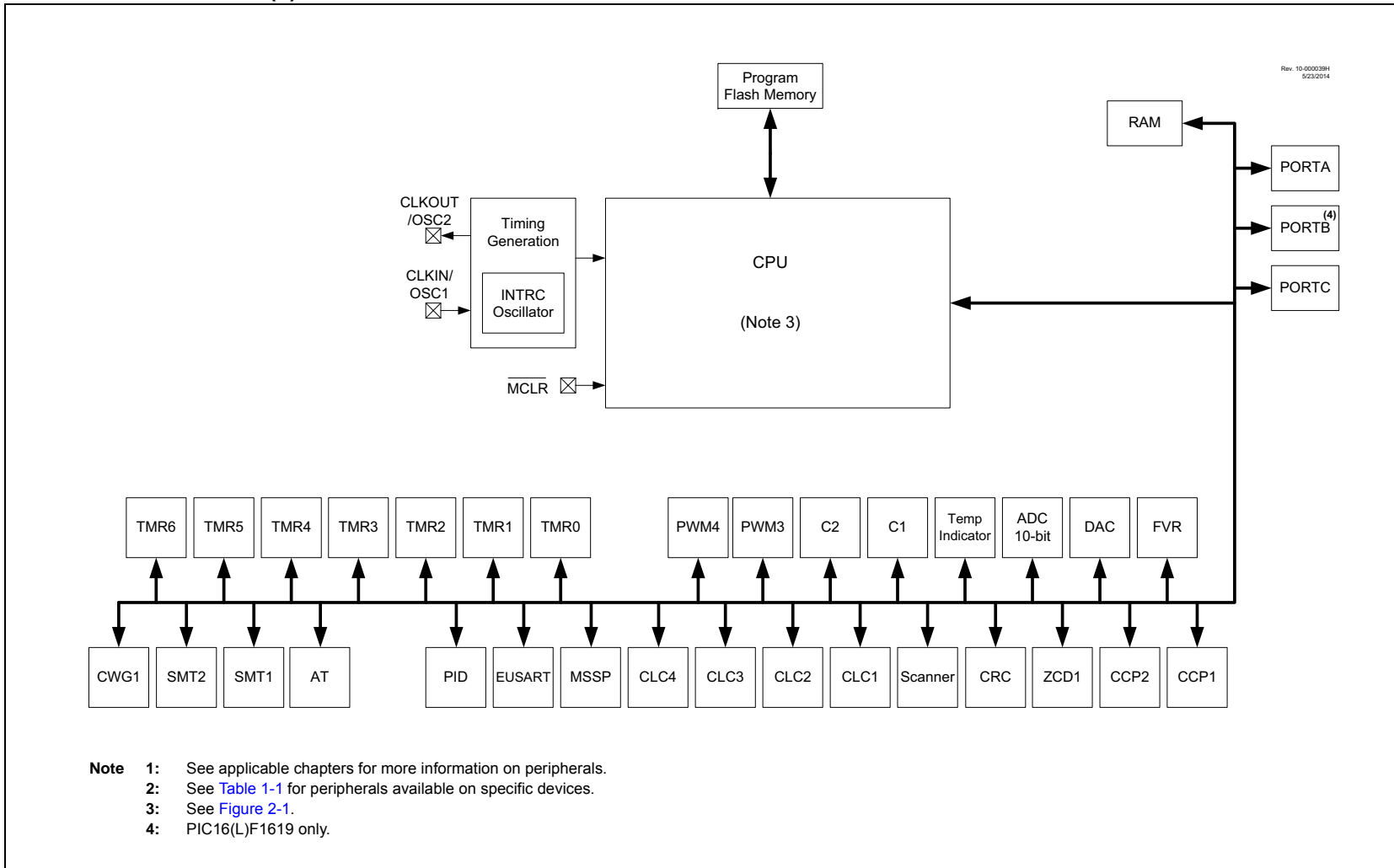
Status, interrupt enables, interrupt flags, and mirror bits are contained in registers that span more than one peripheral. In these cases, the bit name shown is unique so there is no prefix or short name variant.

1.1.3.2 Legacy Peripherals

There are some peripherals that do not strictly adhere to these naming conventions. Peripherals that have existed for many years and are present in almost every device are the exceptions. These exceptions were necessary to limit the adverse impact of the new conventions on legacy code. Peripherals that do adhere to the new convention will include a table in the registers section indicating the long name prefix for each peripheral instance. Peripherals that fall into the exception category will not have this table. These peripherals include, but are not limited to, the following:

- EUSART
- MSSP

FIGURE 1-1: PIC16(L)F1615/9 BLOCK DIAGRAM



- Note**
- 1: See applicable chapters for more information on peripherals.
 - 2: See [Table 1-1](#) for peripherals available on specific devices.
 - 3: See [Figure 2-1](#).
 - 4: PIC16(L)F1619 only.

TABLE 1-2: PIC16(L)F1615 PINOUT DESCRIPTION

| Name | Function | Input Type | Output Type | Description |
|---|----------|------------|-------------|---|
| RA0/AN0/C1IN+/DAC1OUT1/ICSPDAT | RA0 | TTL/ST | CMOS/OD | General purpose I/O. |
| | AN0 | AN | — | ADC Channel input. |
| | C1IN+ | AN | — | Comparator positive input. |
| | DAC1OUT1 | — | AN | Digital-to-Analog Converter output. |
| | ICSPDAT | ST | CMOS | ICSP™ Data I/O. |
| RA1/AN1/VREF+/C1IN0-/C2IN0-/ICSPCLK | RA1 | TTL/ST | CMOS/OD | General purpose I/O. |
| | AN1 | AN | — | ADC Channel input. |
| | VREF+ | AN | — | Voltage Reference input. |
| | C1IN0- | AN | — | Comparator negative input. |
| | C2IN0- | AN | CMOS/OD | Comparator negative input. |
| | ICSPCLK | ST | — | ICSP Programming Clock. |
| RA2/AN2/T0CKI ⁽¹⁾ /CWG1IN ⁽¹⁾ ZCD1IN/INT | RA2 | TTL/ST | CMOS/OD | General purpose I/O. |
| | AN2 | AN | — | ADC Channel input. |
| | T0CKI | TTL/ST | — | Timer0 clock input. |
| | CWG1IN | TTL/ST | — | CWG complementary input. |
| | ZCD1IN | AN | — | Zero-Cross Detect input. |
| | INT | TTL/ST | — | External interrupt. |
| RA3/VPP/T6IN ⁽¹⁾ /SMTWIN2 ⁽¹⁾ / MCLR | RA3 | TTL/ST | — | General purpose input with IOC and WPU. |
| | VPP | HV | — | Programming voltage. |
| | T6IN | TTL/ST | — | Timer6 input. |
| | SMTWIN2 | TTL/ST | — | SMT2 window input. |
| | MCLR | TTL/ST | — | Master Clear with internal pull-up. |
| RA4/AN3/T1G ⁽¹⁾ /SMTSIG1 ⁽¹⁾ / CLKOUT | RA4 | TTL/ST | CMOS/OD | General purpose I/O. |
| | AN3 | AN | — | ADC Channel input. |
| | T1G | TTL/ST | — | Timer1 Gate input. |
| | SMTSIG1 | TTL/ST | — | SMT1 signal input. |
| | CLKOUT | — | CMOS | Fosc/4 output. |
| RA5/CLKIN/T1CKI ⁽¹⁾ /T2IN ⁽¹⁾ / SMTWIN1 ⁽¹⁾ | RA5 | TTL/ST | CMOS/OD | General purpose I/O. |
| | CLKIN | CMOS | — | External clock input (EC mode). |
| | T1CKI | TTL/ST | — | Timer1 clock input. |
| | T2IN | TTL/ST | — | Timer2 input. |
| | SMTWIN1 | TTL/ST | — | SMT1 window input. |
| RC0/AN4/C2IN+/T5CKI ⁽¹⁾ / SCK ⁽¹⁾ | RC0 | TTL/ST | CMOS/OD | General purpose I/O. |
| | AN4 | AN | — | ADC Channel input. |
| | C2IN+ | AN | — | Comparator positive input. |
| | T5CKI | TTL/ST | — | Timer5 clock input. |
| | SCK | ST | CMOS | SPI clock. |

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:** Default peripheral input. Input can be moved to any other pin with the PPS input selection registers.
Note 2: All pin outputs default to PORT latch data. Any pin can be selected as a digital peripheral output with the PPS output selection registers. See [Register 13-1](#).

TABLE 1-2: PIC16(L)F1615 PINOUT DESCRIPTION (CONTINUED)

| Name | Function | Input Type | Output Type | Description |
|---|----------|------------|-------------|---|
| RC1/AN5/C1IN1-/C2IN1-/T4IN ⁽¹⁾ /SMTSIG2 ⁽¹⁾ /SDI ⁽¹⁾ | RC1 | TTL/ST | CMOS/OD | General purpose I/O. |
| | AN5 | AN | — | ADC Channel input. |
| | C1IN1- | AN | — | Comparator negative input. |
| | C2IN1- | AN | — | Comparator negative input. |
| | T4IN | TTL/ST | — | Timer4 input. |
| | SMTSIG2 | TTL/ST | — | SMT2 signal input. |
| | SDI | CMOS | — | SPI data input. |
| RC2/AN6/C1IN2-/C2IN2- | RC2 | TTL/ST | CMOS/OD | General purpose I/O. |
| | AN6 | AN | — | ADC Channel input. |
| | C1IN2- | AN | — | Comparator negative input. |
| | C2IN2- | AN | — | Comparator negative input. |
| RC3/AN7/C1IN3-/C2IN3-/T5G ⁽¹⁾ /CCP2 ⁽¹⁾ /CLCIN0 ⁽¹⁾ /ATCC ⁽¹⁾ /SS | RC3 | TTL/ST | — | General purpose input with IOC and WPU. |
| | AN7 | AN | — | ADC Channel input. |
| | C1IN3- | AN | — | Comparator negative input. |
| | C2IN3- | AN | — | Comparator negative input. |
| | T5G | ST | — | Timer5 Gate input. |
| | CCP2 | TTL/ST | CMOS/OD | Capture/Compare/PWM2. |
| | CLCIN0 | ST | — | Configurable Logic Cell source input. |
| | ATCC | ST | — | Angular Timer Capture/Compare input. |
| | SS | ST | — | Slave Select input. |
| RC4/T3G ⁽¹⁾ /CLCIN1 ⁽¹⁾ /CK ⁽¹⁾ /HIC4 | RC4 | TTL/ST | CMOS/OD | General purpose I/O. |
| | T3G | TTL/ST | — | Timer3 Gate input. |
| | CLCIN1 | ST | — | Configurable Logic Cell source input. |
| | CK | ST | CMOS | EUSART synchronous clock. |
| | HIC4 | TTL | CMOS | High current I/O. |
| RC5/T3CKI ⁽¹⁾ /CCP1 ⁽¹⁾ /RX ⁽¹⁾ /ATIN ⁽¹⁾ /HIC5 | RC5 | TTL/ST | CMOS/OD | General purpose I/O. |
| | T3CKI | TTL/ST | — | Timer3 clock input. |
| | CCP1 | TTL/ST | CMOS/OD | Capture/Compare/PWM1. |
| | RX | ST | — | EUSART asynchronous input. |
| | ATIN | TTL/ST | — | Angular Timer clock input. |
| | HIC5 | TTL | CMOS | High current 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: Default peripheral input. Input can be moved to any other pin with the PPS input selection registers.
Note 2: All pin outputs default to PORT latch data. Any pin can be selected as a digital peripheral output with the PPS output selection registers. See [Register 13-1](#).

TABLE 1-2: PIC16(L)F1615 PINOUT DESCRIPTION (CONTINUED)

| Name | Function | Input Type | Output Type | Description |
|--------------------|----------|------------|--|---|
| OUT ⁽²⁾ | C1OUT | — | CMOS | Comparator output. |
| | C2OUT | — | CMOS | Comparator output. |
| | CCP1 | — | CMOS | Capture/Compare/PWM1 output. |
| | CCP2 | — | CMOS | Capture/Compare/PWM2 output. |
| | PWM3OUT | — | CMOS | PWM3 output. |
| | PWM4OUT | — | CMOS | PWM4 output. |
| | CWG1A | — | CMOS | Complementary Output Generator Output A. |
| | CWG1B | — | CMOS | Complementary Output Generator Output B. |
| | CWG1C | — | CMOS | Complementary Output Generator Output C. |
| | CWG1D | — | CMOS | Complementary Output Generator Output D. |
| | SDO | — | CMOS | SPI data output. |
| | SCK | — | CMOS | SPI clock output. |
| | TX/CK | — | CMOS | EUSART asynchronous TX data/synchronous clock output. |
| | DT | — | CMOS | EUSART synchronous data output.E |
| | CLC1OUT | — | CMOS | Configurable Logic Cell 1 source output. |
| CLC2OUT | — | CMOS | Configurable Logic Cell 2 source output. | |
| ZCD1OUT | — | CMOS | Zero-Cross Detect output. | |
| 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:** Default peripheral input. Input can be moved to any other pin with the PPS input selection registers.
Note 2: All pin outputs default to PORT latch data. Any pin can be selected as a digital peripheral output with the PPS output selection registers. See [Register 13-1](#).

TABLE 1-3: PIC16(L)F1619 PINOUT DESCRIPTION

| Name | Function | Input Type | Output Type | Description |
|---|----------|------------|-------------|---|
| RA0/AN0/C1IN+/DAC1OUT/ICSPDAT | RA0 | TTL/ST | CMOS/OD | General purpose I/O. |
| | AN0 | AN | — | ADC Channel input. |
| | C1IN+ | AN | — | Comparator positive input. |
| | DAC1OUT | — | AN | Digital-to-Analog Converter output. |
| | ICSPDAT | ST | CMOS | ICSP™ Data I/O. |
| RA1/AN1/VREF+/C1IN0-/C2IN0-/ICSPCLK | RA1 | TTL/ST | CMOS/OD | General purpose I/O. |
| | AN1 | AN | — | ADC Channel input. |
| | VREF+ | AN | — | Voltage Reference input. |
| | C1IN0- | AN | — | Comparator negative input. |
| | C2IN0- | AN | CMOS/OD | Comparator negative input. |
| | ICSPCLK | ST | — | ICSP Programming Clock. |
| RA2/AN2/T0CKI ⁽¹⁾ /CWG1IN ⁽¹⁾ /ZCD1IN/INT | RA2 | TTL/ST | CMOS/OD | General purpose I/O. |
| | AN2 | AN | — | ADC Channel input. |
| | T0CKI | TTL/ST | — | Timer0 clock input. |
| | CWG1IN | TTL/ST | — | CWG complementary input. |
| | ZCD1IN | AN | — | Zero-Cross Detect input. |
| | INT | TTL/ST | — | External interrupt. |
| RA3/VPP/T6IN ⁽¹⁾ /SMTWIN2 ⁽¹⁾ /MCLR | RA3 | TTL/ST | — | General purpose input with IOC and WPU. |
| | VPP | HV | — | Programming voltage. |
| | T6IN | TTL/ST | — | Timer6 input. |
| | SMTWIN2 | TTL/ST | — | SMT2 window input. |
| | MCLR | TTL/ST | — | Master Clear with internal pull-up. |
| RA4/AN3/T1G ⁽¹⁾ /SMTSIG1 ⁽¹⁾ /CLKOUT | RA4 | TTL/ST | CMOS/OD | General purpose I/O. |
| | AN3 | AN | — | ADC Channel input. |
| | T1G | TTL/ST | — | Timer1 Gate input. |
| | SMTSIG1 | TTL/ST | — | SMT1 signal input. |
| | CLKOUT | — | CMOS | Fosc/4 output. |
| RA5/CLKIN/T1CKI ⁽¹⁾ /T2IN ⁽¹⁾ /CLCIN3 ⁽¹⁾ /SMTWIN1 | RA5 | TTL/ST | CMOS/OD | General purpose I/O. |
| | CLKIN | CMOS | — | External clock input (EC mode). |
| | T1CKI | TTL/ST | — | Timer1 clock input. |
| | T2IN | TTL/ST | — | Timer2 input. |
| | CLCIN3 | ST | — | Configurable Logic Cell source input. |
| | SMTWIN1 | TTL/ST | — | SMT1 window input. |
| RB4/AN10/SDI ⁽¹⁾ | RB4 | TTL/ST | CMOS/OD | General purpose I/O. |
| | AN10 | AN | — | ADC Channel input. |
| | SDI | CMOS | — | SPI data input. |
| RB5/AN11/RX ^(1, 3) | RB5 | TTL/ST | CMOS/OD | General purpose I/O. |
| | AN11 | AN | — | ADC Channel input. |
| | RX | ST | — | EUSART asynchronous 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: Default peripheral input. Input can be moved to any other pin with the PPS input selection registers.
2: All pin outputs default to PORT latch data. Any pin can be selected as a digital peripheral output with the PPS output selection registers. See [Register 13-1](#).
3: These I²C functions are bidirectional. The output pin selections must be the same as the input pin selections.

TABLE 1-3: PIC16(L)F1619 PINOUT DESCRIPTION (CONTINUED)

| Name | Function | Input Type | Output Type | Description |
|---|----------|------------|-------------|---|
| RB6/SCK ^(1, 3) | RB6 | TTL/ST | CMOS/OD | General purpose I/O. |
| | SCK | ST | CMOS | SPI clock. |
| RB7/CK ⁽¹⁾ | RB7 | TTL/ST | CMOS/OD | General purpose I/O. |
| | CK | ST | CMOS | EUSART synchronous clock. |
| RC0/AN4/C2IN+/T5CKI ⁽¹⁾ | RC0 | TTL/ST | CMOS/OD | General purpose I/O. |
| | AN4 | AN | — | ADC Channel input. |
| | C2IN+ | AN | — | Comparator positive input. |
| | T5CKI | TTL/ST | — | Timer5 clock input. |
| RC1/AN5/C1IN1-/C2IN1-/T4IN ⁽¹⁾ /CLCIN2 ⁽²⁾ /SMTSIG2 ⁽¹⁾ | RC1 | TTL/ST | CMOS/OD | General purpose I/O. |
| | AN5 | AN | — | ADC Channel input. |
| | C1IN1- | AN | — | Comparator negative input. |
| | C2IN1- | AN | — | Comparator negative input. |
| | T4IN | TTL/ST | — | Timer4 input. |
| | CLCIN2 | ST | — | Configurable Logic Cell source input. |
| RC2/AN6/C1IN2-/C2IN2- | RC2 | TTL/ST | CMOS/OD | General purpose I/O. |
| | AN6 | AN | — | ADC Channel input. |
| | C1IN2- | AN | — | Comparator negative input. |
| | C2IN2- | AN | — | Comparator negative input. |
| RC3/AN7/C1IN3-/C2IN3-/T5G ⁽¹⁾ /CCP2 ⁽¹⁾ /CLCIN0 ⁽¹⁾ /ATCC ⁽¹⁾ | RC3 | TTL/ST | — | General purpose input with IOC and WPU. |
| | AN7 | AN | — | ADC Channel input. |
| | C1IN3- | AN | — | Comparator negative input. |
| | C2IN3- | AN | — | Comparator negative input. |
| | T5G | ST | — | Timer5 Gate input. |
| | CCP2 | ST | CMOS | Capture/Compare/PWM2. |
| | CLCIN0 | ST | — | Configurable Logic Cell source input. |
| | ATCC | ST | — | Angular Timer Capture/Compare input. |
| RC4/T3G ⁽¹⁾ /CLCIN1 ⁽¹⁾ /HIC4 | RC4 | TTL/ST | CMOS/OD | General purpose I/O. |
| | T3G | ST | — | Timer3 Gate input. |
| | CLCIN1 | ST | — | Configurable Logic Cell source input. |
| | HIC4 | TTL | CMOS | High current I/O. |
| RC5/T3CKI ⁽¹⁾ /CCP1 ⁽¹⁾ /ATIN ⁽¹⁾ /HIC5 | RC5 | TTL/ST | CMOS/OD | General purpose I/O. |
| | T3CKI | TTL/ST | — | Timer3 clock input. |
| | CCP1 | TTL/ST | CMOS/OD | Capture/Compare/PWM1. |
| | ATIN | TTL/ST | — | Angular Timer clock input. |
| | HIC5 | TTL | CMOS | High current I/O. |
| RC6/AN8/SS ⁽¹⁾ | RC6 | TTL/ST | CMOS/OD | General purpose I/O. |
| | AN8 | AN | — | ADC Channel input. |
| | SS | ST | — | SPI slave select. |
| RC7/AN9 | RC7 | TTL/ST | CMOS/OD | General purpose I/O. |
| | AN9 | AN | — | ADC Channel 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: Default peripheral input. Input can be moved to any other pin with the PPS input selection registers.
2: All pin outputs default to PORT latch data. Any pin can be selected as a digital peripheral output with the PPS output selection registers. See [Register 13-1](#).
3: These I²C functions are bidirectional. The output pin selections must be the same as the input pin selections.

TABLE 1-3: PIC16(L)F1619 PINOUT DESCRIPTION (CONTINUED)

| Name | Function | Input Type | Output Type | Description |
|--------------------|----------|------------|--|---|
| OUT ⁽²⁾ | C1OUT | — | CMOS | Comparator output. |
| | C2OUT | — | CMOS | Comparator output. |
| | CCP1 | — | CMOS | Capture/Compare/PWM1 output. |
| | CCP2 | — | CMOS | Capture/Compare/PWM2 output. |
| | PWM3OUT | — | CMOS | PWM3 output. |
| | PWM4OUT | — | CMOS | PWM4 output. |
| | CWG1A | — | CMOS | Complementary Output Generator Output A. |
| | CWG1B | — | CMOS | Complementary Output Generator Output B. |
| | CWG1C | — | CMOS | Complementary Output Generator Output C. |
| | CWG1D | — | CMOS | Complementary Output Generator Output D. |
| | SDO | — | CMOS | SPI data output. |
| | SCK | — | CMOS | SPI clock output. |
| | TX/CK | — | CMOS | EUSART asynchronous TX data/synchronous clock output. |
| | DT | — | CMOS | EUSART synchronous data output. |
| | CLC1OUT | — | CMOS | Configurable Logic Cell 1 source output. |
| | CLC2OUT | — | CMOS | Configurable Logic Cell 2 source output. |
| | CLC3OUT | — | CMOS | Configurable Logic Cell 3 source output. |
| CLC4OUT | — | CMOS | Configurable Logic Cell 4 source output. | |
| ZCD1OUT | — | CMOS | Zero-Cross Detect output. | |
| 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

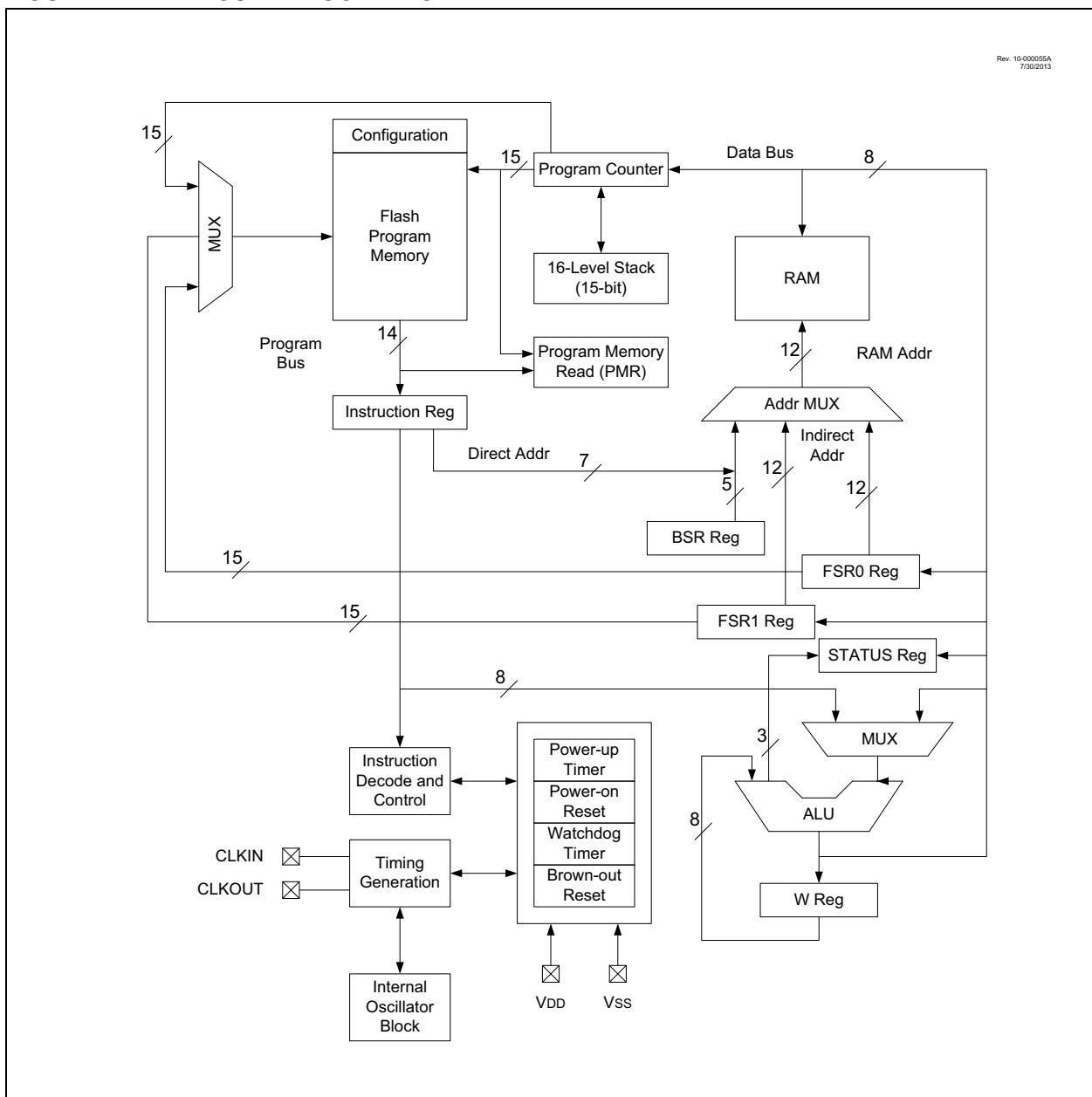
- Note** 1: Default peripheral input. Input can be moved to any other pin with the PPS input selection registers.
2: All pin outputs default to PORT latch data. Any pin can be selected as a digital peripheral output with the PPS output selection registers. See [Register 13-1](#).
3: These I²C functions are bidirectional. The output pin selections must be the same as the input pin selections.

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



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 a hardware 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 [Section 3.5 “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 3.6 “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 34.0 “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. 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](#)).

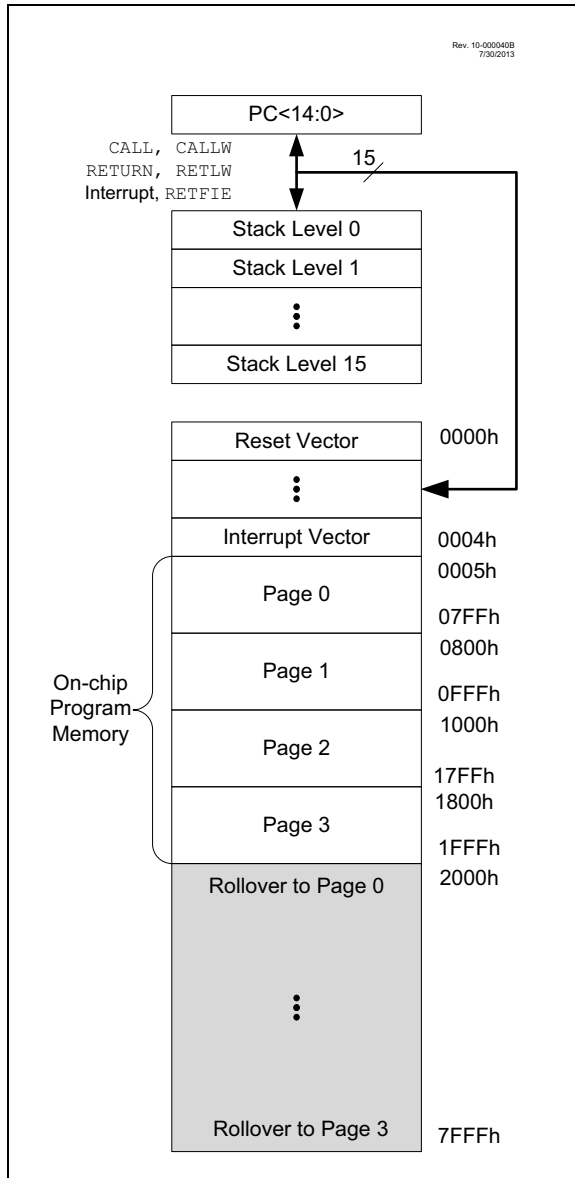
3.2 High-Endurance Flash

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 10.2 “Flash Program Memory Overview”](#) for more information on writing data to PFM. See [Section 3.2.1.2 “Indirect Read with FSR”](#) for more information about using the FSR registers to read byte data stored in PFM.

| Device | Program Memory Space (Words) | Last Program Memory Address | High-Endurance Flash Memory Address Range ⁽¹⁾ |
|-----------------|------------------------------|-----------------------------|--|
| PIC16(L)F1615/9 | 8,192 | 1FFFh | 1F80h-1FFFh |

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

FIGURE 3-1: PROGRAM MEMORY MAP AND STACK FOR PIC16(L)F1615/9



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 eight 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` operator 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 sets
                        automatically
  MOVWF FSR1H
  BTFSC STATUS, C      ;carry from ADDLW?
  INCF FSR1h, f        ;yes
  MOVIW 0[FSR1]
;THE PROGRAM MEMORY IS IN W
```


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 3-2):

- 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 3.6 “Indirect Addressing” for more information.

Data memory uses a 12-bit address. The upper five bits of the address define the Bank address and the lower seven 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/x80h through x0Bh/x8Bh). These registers are listed below in Table 3-1. For detailed

TABLE 3-1: 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 TO and 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 34.0 "Instruction Set Summary"](#)).

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

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 |
| — | — | — | <u>TO</u> | <u>PD</u> | Z | <u>DC</u> ⁽¹⁾ | <u>C</u> ⁽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⁽¹⁾ (`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 Borrow, the polarity is reversed. A subtraction is executed by adding the two's complement of the second operand. For rotate (`RRF`, `RLF`) instructions, this bit is loaded with either the high-order or low-order bit of the source register.

3.3.2 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.

3.3.3 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).

3.3.3.1 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 3.6.2 “Linear Data Memory”](#) for more information.

3.3.4 COMMON RAM

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

3.3.5 DEVICE MEMORY MAPS

The memory maps are shown in [Table 3-2](#) through [Table 3-12](#).

FIGURE 3-2: BANKED MEMORY PARTITIONING

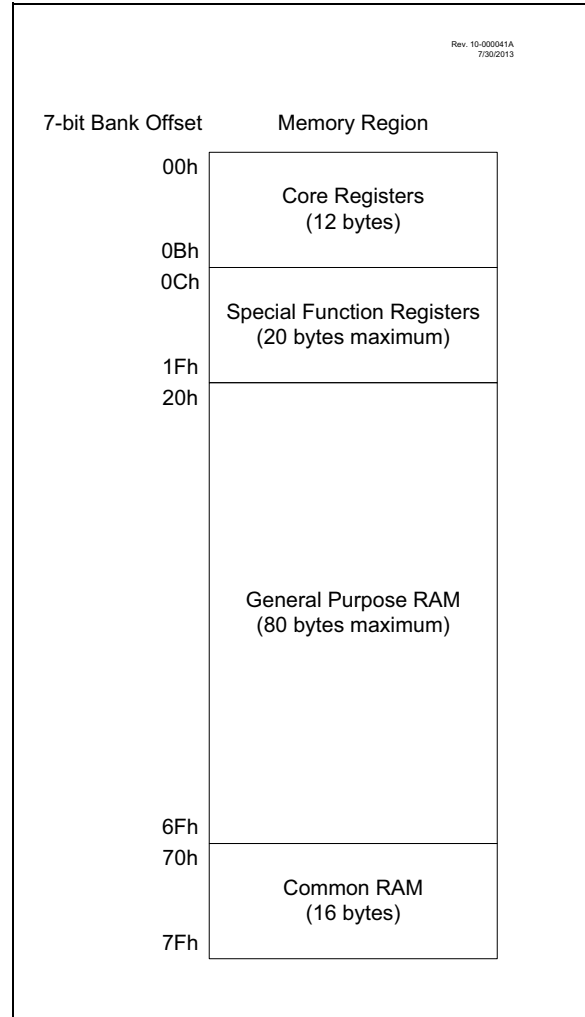


TABLE 3-2: PIC16(L)F1615 MEMORY MAP, BANK 0-7

| BANK 0 | | BANK 1 | | BANK 2 | | BANK 3 | | BANK 4 | | BANK 5 | | BANK 6 | | BANK 7 | |
|--------|--------------------------------------|--------|--------------------------------------|--------|--------------------------------------|--------|--------------------------------------|--------|--------------------------------------|--------|--------------------------------------|--------|--------------------------------------|--------|--------------------------------------|
| 000h | Core Registers (Table 3-1) | 080h | Core Registers (Table 3-1) | 100h | Core Registers (Table 3-1) | 180h | Core Registers (Table 3-1) | 200h | Core Registers (Table 3-1) | 280h | Core Registers (Table 3-1) | 300h | Core Registers (Table 3-1) | 380h | Core Registers (Table 3-1) |
| 00Bh | | 08Bh | | 10Bh | | 18Bh | | 20Bh | | 28Bh | | 30Bh | | 38Bh | |
| 00Ch | PORTA | 08Ch | TRISA | 10Ch | LATA | 18Ch | ANSELA | 20Ch | WPUA | 28Ch | ODCONA | 30Ch | SLRCONA | 38Ch | INLVLA |
| 00Dh | — | 08Dh | — | 10Dh | — | 18Dh | — | 20Dh | — | 28Dh | — | 30Dh | — | 38Dh | — |
| 00Eh | PORTC | 08Eh | TRISC | 10Eh | LATC | 18Eh | ANSELC | 20Eh | WPUC | 28Eh | ODCONC | 30Eh | SLRCONC | 38Eh | INLVLC |
| 00Fh | — | 08Fh | — | 10Fh | — | 18Fh | — | 20Fh | — | 28Fh | — | 30Fh | — | 38Fh | — |
| 010h | PIR1 | 090h | PIE1 | 110h | — | 190h | — | 210h | — | 290h | — | 310h | — | 390h | — |
| 011h | PIR2 | 091h | PIE2 | 111h | CM1CON0 | 191h | PMADRL | 211h | SSP1BUF | 291h | CCPR1L | 311h | — | 391h | IOCAP |
| 012h | PIR3 | 092h | PIE3 | 112h | CM1CON1 | 192h | PMADRH | 212h | SSP1ADD | 292h | CCPR1H | 312h | — | 392h | IOCAN |
| 013h | PIR4 | 093h | PIE4 | 113h | CM2CON0 | 193h | PMDATL | 213h | SSP1MSK | 293h | CCP1CON | 313h | — | 393h | IOCAF |
| 014h | PIR5 | 094h | PIE5 | 114h | CM2CON1 | 194h | PMDATH | 214h | SSP1STAT | 294h | CCP1CAP | 314h | — | 394h | — |
| 015h | TMR0 | 095h | OPTION_REG | 115h | CMOUT | 195h | PMCON1 | 215h | SSP1CON | 295h | — | 315h | — | 395h | — |
| 016h | TMR1L | 096h | PCON | 116h | BORCON | 196h | PMCON2 | 216h | SSP1CON2 | 296h | — | 316h | — | 396h | — |
| 017h | TMR1H | 097h | — | 117h | FVRCON | 197h | VREGCON | 217h | SSP1CON3 | 297h | — | 317h | — | 397h | IOCCP |
| 018h | T1CON | 098h | OSCTUNE | 118h | DAC1CON0 | 198h | — | 218h | — | 298h | CCPR2L | 318h | — | 398h | IOCCN |
| 019h | T1GCON | 099h | OSCCON | 119h | DAC1CON1 | 199h | RC1REG | 219h | — | 299h | CCPR2H | 319h | — | 399h | IOCCF |
| 01Ah | TMR2 | 09Ah | OSCSTAT | 11Ah | — | 19Ah | TX1REG | 21Ah | — | 29Ah | CCP2CON | 31Ah | — | 39Ah | — |
| 01Bh | PR2 | 09Bh | ADRESL | 11Bh | — | 19Bh | SP1BRGL | 21Bh | — | 29Bh | CCP2CAP | 31Bh | — | 39Bh | — |
| 01Ch | T2CON | 09Ch | ADRESH | 11Ch | ZCD1CON | 19Ch | SP1BRGH | 21Ch | — | 29Ch | — | 31Ch | — | 39Ch | — |
| 01Dh | T2HLT | 09Dh | ADCON0 | 11Dh | — | 19Dh | RC1STA | 21Dh | — | 29Dh | — | 31Dh | — | 39Dh | — |
| 01Eh | T2CLKCON | 09Eh | ADCON1 | 11Eh | — | 19Eh | TX1STA | 21Eh | — | 29Eh | CCPTMRS | 31Eh | — | 39Eh | — |
| 01Fh | T2RST | 09Fh | ADCON2 | 11Fh | — | 19Fh | BAUD1CON | 21Fh | — | 29Fh | — | 31Fh | — | 39Fh | — |
| 020h | General Purpose Register 96 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 | 3A0h | General Purpose Register 80 Bytes |
| | | 0EFh | Common RAM (Accesses 70h – 7Fh) | 16Fh | Common RAM (Accesses 70h – 7Fh) | 1EFh | Common RAM (Accesses 70h – 7Fh) | 26Fh | Common RAM (Accesses 70h – 7Fh) | 2EFh | Common RAM (Accesses 70h – 7Fh) | 36Fh | Common RAM (Accesses 70h – 7Fh) | 3EFh | Common RAM (Accesses 70h – 7Fh) |
| | | 0F0h | | 170h | | 1F0h | | 270h | | 2F0h | | 370h | | 3F0h | |
| 07Fh | | 0FFh | | 17Fh | | 1FFh | | 27Fh | | 2FFh | | 37Fh | | 3FFh | |

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

TABLE 3-3: PIC16(L)F1619 MEMORY MAP, BANK 0-7

| BANK 0 | | BANK 1 | | BANK 2 | | BANK 3 | | BANK 4 | | BANK 5 | | BANK 6 | | BANK 7 | |
|--------|--------------------------------------|--------|---------------------------------------|--------|---------------------------------------|--------|---------------------------------------|--------|---------------------------------------|--------|---------------------------------------|--------|---------------------------------------|--------|---------------------------------------|
| 000h | Core Registers (Table 3-1) | 080h | Core Registers (Table 3-1) | 100h | Core Registers (Table 3-1) | 180h | Core Registers (Table 3-1) | 200h | Core Registers (Table 3-1) | 280h | Core Registers (Table 3-1) | 300h | Core Registers (Table 3-1) | 380h | Core Registers (Table 3-1) |
| 00Bh | | 08Bh | | 10Bh | | 18Bh | | 20Bh | | 28Bh | | 30Bh | | 38Bh | |
| 00Ch | PORTA | 08Ch | TRISA | 10Ch | LATA | 18Ch | ANSELA | 20Ch | WPUA | 28Ch | ODCONA | 30Ch | SLRCONA | 38Ch | INLVLA |
| 00Dh | PORTB | 08Dh | TRISB | 10Dh | LATB | 18Dh | ANSELB | 20Dh | WPUB | 28Dh | ODCONB | 30Dh | SLRCONB | 38Dh | INLVLB |
| 00Eh | PORTC | 08Eh | TRISC | 10Eh | LATC | 18Eh | ANSELC | 20Eh | WPUC | 28Eh | ODCONC | 30Eh | SLRCONC | 38Eh | INLVLC |
| 00Fh | — | 08Fh | — | 10Fh | — | 18Fh | — | 20Fh | — | 28Fh | — | 30Fh | — | 38Fh | — |
| 010h | PIR1 | 090h | PIE1 | 110h | — | 190h | — | 210h | — | 290h | — | 310h | — | 390h | — |
| 011h | PIR2 | 091h | PIE2 | 111h | CM1CON0 | 191h | PMADRL | 211h | SSP1BUF | 291h | CCP1RL | 311h | — | 391h | IOCAP |
| 012h | PIR3 | 092h | PIE3 | 112h | CM1CON1 | 192h | PMADRH | 212h | SSP1ADD | 292h | CCP1RH | 312h | — | 392h | IOCAN |
| 013h | PIR4 | 093h | PIE4 | 113h | CM2CON0 | 193h | PMDATL | 213h | SSP1MSK | 293h | CCP1CON | 313h | — | 393h | IOCAF |
| 014h | PIR5 | 094h | PIE5 | 114h | CM2CON1 | 194h | PMDATH | 214h | SSP1STAT | 294h | CCP1CAP | 314h | — | 394h | IOCBP |
| 015h | TMR0 | 095h | OPTION_REG | 115h | CMOUT | 195h | PMCON1 | 215h | SSP1CON | 295h | — | 315h | — | 395h | IOCBN |
| 016h | TMR1L | 096h | PCON | 116h | BORCON | 196h | PMCON2 | 216h | SSP1CON2 | 296h | — | 316h | — | 396h | IOCBF |
| 017h | TMR1H | 097h | — | 117h | FVRCON | 197h | VREGCON | 217h | SSP1CON3 | 297h | — | 317h | — | 397h | IOCCP |
| 018h | T1CON | 098h | OSCTUNE | 118h | DAC1CON0 | 198h | — | 218h | — | 298h | CCP2RL | 318h | — | 398h | IOCCN |
| 019h | T1GCON | 099h | OSCCON | 119h | DAC1CON1 | 199h | RC1REG | 219h | — | 299h | CCP2RH | 319h | — | 399h | IOCCF |
| 01Ah | TMR2 | 09Ah | OSCSTAT | 11Ah | — | 19Ah | TX1REG | 21Ah | — | 29Ah | CCP2CON | 31Ah | — | 39Ah | — |
| 01Bh | PR2 | 09Bh | ADRESL | 11Bh | — | 19Bh | SP1BRGL | 21Bh | — | 29Bh | CCP2CAP | 31Bh | — | 39Bh | — |
| 01Ch | T2CON | 09Ch | ADRESH | 11Ch | ZCD1CON | 19Ch | SP1BRGH | 21Ch | — | 29Ch | — | 31Ch | — | 39Ch | — |
| 01Dh | T2HLT | 09Dh | ADCON0 | 11Dh | — | 19Dh | RC1STA | 21Dh | — | 29Dh | — | 31Dh | — | 39Dh | — |
| 01Eh | T2CLKCON | 09Eh | ADCON1 | 11Eh | — | 19Eh | TX1STA | 21Eh | — | 29Eh | CCPTMRS | 31Eh | — | 39Eh | — |
| 01Fh | T2RST | 09Fh | ADCON2 | 11Fh | — | 19Fh | BAUD1CON | 21Fh | — | 29Fh | — | 31Fh | — | 39Fh | — |
| 020h | General Purpose Register 96 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 | 3A0h | General Purpose Register 80 Bytes |
| | | 0EFh | | 16Fh | | 1EFh | | 26Fh | | 2EFh | | 36Fh | | 3EFh | |
| | | 0F0h | Common RAM (Accesses 70h – 7Fh) | 170h | Common RAM (Accesses 70h – 7Fh) | 1F0h | Common RAM (Accesses 70h – 7Fh) | 270h | Common RAM (Accesses 70h – 7Fh) | 2F0h | Common RAM (Accesses 70h – 7Fh) | 370h | Common RAM (Accesses 70h – 7Fh) | 3F0h | Common RAM (Accesses 70h – 7Fh) |
| 07Fh | | 0FFh | | 17Fh | | 1FFh | | 27Fh | | 2FFh | | 37Fh | | 3FFh | |

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

TABLE 3-4: PIC16(L)F1615/9 MEMORY MAP, BANK 8-15

| BANK 8 | | BANK 9 | | BANK 10 | | BANK 11 | | BANK 12 | | BANK 13 | | BANK 14 | | BANK 15 | |
|--------|--------------------------------------|--------|--------------------------------------|---------|--------------------------------------|---------|--------------------------------------|---------|--------------------------------------|---------|-------------------------------|---------|-------------------------------|---------|-------------------------------|
| 400h | Core Registers (Table 3-1) | 480h | Core Registers (Table 3-1) | 500h | Core Registers (Table 3-1) | 580h | Core Registers (Table 3-1) | 600h | Core Registers (Table 3-1) | 680h | Core Registers (Table 3-1) | 700h | Core Registers (Table 3-1) | 780h | Core Registers (Table 3-1) |
| 40Bh | — | 48Bh | — | 50Bh | — | 58Bh | PID1SETL | 60Bh | PID1Z2L | 68Bh | — | 70Bh | — | 78Bh | — |
| 40Ch | — | 48Ch | — | 50Ch | — | 58Ch | PID1SETH | 60Ch | PID1Z2H | 68Ch | — | 70Ch | — | 78Ch | — |
| 40Dh | — | 48Dh | — | 50Dh | — | 58Dh | — | 60Dh | PID1Z2H | 68Dh | — | 70Dh | — | 78Dh | — |
| 40Eh | HDRVENC | 48Eh | — | 50Eh | — | 58Eh | PID1INL | 60Eh | PID1Z2U | 68Eh | — | 70Eh | — | 78Eh | — |
| 40Fh | — | 48Fh | — | 50Fh | — | 58Fh | PID1INH | 60Fh | PID1ACCLL | 68Fh | — | 70Fh | — | 78Fh | — |
| 410h | — | 490h | — | 510h | — | 590h | PID1K1L | 610h | PID1ACCLH | 690h | — | 710h | — | 790h | — |
| 411h | — | 491h | — | 511h | — | 591h | PID1K1H | 611h | PID1ACCHL | 691h | CWG1DBR | 711h | WDTCON0 | 791h | CRCDATL |
| 412h | — | 492h | — | 512h | — | 592h | PID1K2L | 612h | PID1ACCHH | 692h | CWG1DBF | 712h | WDTCON1 | 792h | CRCDATH |
| 413h | TMR4 | 493h | TMR3L | 513h | — | 593h | PID1K2H | 613h | PID1ACCUL | 693h | CWG1AS0 | 713h | WDTPSL | 793h | CRCACCL |
| 414h | PR4 | 494h | TMR3H | 514h | — | 594h | PID1K3L | 614h | PID1CON | 694h | CWG1AS1 | 714h | WDTPSH | 794h | CRCACCH |
| 415h | T4CON | 495h | T3CON | 515h | — | 595h | PID1K3H | 615h | — | 695h | CWG1OCON0 | 715h | WDTTMR | 795h | CRCSHIFTL |
| 416h | T4HLT | 496h | T3GCON | 516h | — | 596h | PID1OUTLL | 616h | — | 696h | CWG1CON0 | 716h | — | 796h | CRCSHIFTH |
| 417h | T4CLKCON | 497h | — | 517h | — | 597h | PID1OUTLH | 617h | PWM3DCL | 697h | CWG1CON1 | 717h | — | 797h | CRCXORL |
| 418h | T4RST | 498h | — | 518h | — | 598h | PID1OUTHL | 618h | PWM3DCH | 698h | — | 718h | SCANLADR | 798h | CRCXORH |
| 419h | — | 499h | — | 519h | — | 599h | PID1OUTH | 619h | PWM3CON | 699h | CWG1CLKCON | 719h | SCANLADR | 799h | CRCCON0 |
| 41Ah | TMR6 | 49Ah | TMR5L | 51Ah | — | 59Ah | PID1OUTUL | 61Ah | PWM4DCL | 69Ah | CWG1ISM | 71Ah | SCANHADRL | 79Ah | CRCCON1 |
| 41Bh | PR6 | 49Bh | TMR5H | 51Bh | — | 59Bh | PID1Z1L | 61Bh | PWM4DCH | 69Bh | — | 71Bh | SCANHADRH | 79Bh | — |
| 41Ch | T6CON | 49Ch | T5CON | 51Ch | — | 59Ch | PID1Z1H | 61Ch | PWM4CON | 69Ch | — | 71Ch | SCANCON0 | 79Ch | — |
| 41Dh | T6HLT | 49Dh | T5GCON | 51Dh | — | 59Dh | PID1Z1U | 61Dh | — | 69Dh | — | 71Dh | SCANTRIG | 79Dh | — |
| 41Eh | T6CLKCON | 49Eh | — | 51Eh | — | 59Eh | — | 61Eh | — | 69Eh | — | 71Eh | — | 79Eh | — |
| 41Fh | T6RST | 49Fh | — | 51Fh | — | 59Fh | — | 61Fh | — | 69Fh | — | 71Fh | — | 79Fh | — |
| 420h | General Purpose Register 80 Bytes | 4A0h | General Purpose Register 80 Bytes | 520h | General Purpose Register 80 Bytes | 5A0h | General Purpose Register 80 Bytes | 620h | General Purpose Register 48 Bytes | 6A0h | Unimplemented Read as '0' | 720h | Unimplemented Read as '0' | 7A0h | Unimplemented Read as '0' |
| 64Fh | — | 64Fh | — | 64Fh | — | 64Fh | — | 64Fh | Unimplemented Read as '0' | 64Fh | — | 64Fh | — | 64Fh | — |
| 46Fh | — | 4EFh | — | 56Fh | — | 5EFh | — | 66Fh | — | 66Fh | — | 76Fh | — | 7EFh | — |
| 470h | Accesses 70h – 7Fh | 4F0h | Accesses 70h – 7Fh | 570h | Accesses 70h – 7Fh | 5F0h | Accesses 70h – 7Fh | 670h | Accesses 70h – 7Fh | 6F0h | Accesses 70h – 7Fh | 770h | Accesses 70h – 7Fh | 7F0h | Accesses 70h – 7Fh |
| 47Fh | — | 4FFh | — | 57Fh | — | 5FFh | — | 67Fh | — | 6FFh | — | 77Fh | — | 7FFh | — |

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

TABLE 3-5: PIC16(L)F1615/9 MEMORY MAP, BANK 16-23

| BANK 16 | | BANK 17 | | BANK 18 | | BANK 19 | | BANK 20 | | BANK 21 | | BANK 22 | | BANK 23 | |
|---------|-------------------------------|---------|-------------------------------|---------|-------------------------------|---------|-------------------------------|---------|-------------------------------|---------|-------------------------------|---------|-------------------------------|---------|-------------------------------|
| 800h | Core Registers (Table 3-1) | 880h | Core Registers (Table 3-1) | 900h | Core Registers (Table 3-1) | 980h | Core Registers (Table 3-1) | A00h | Core Registers (Table 3-1) | A80h | Core Registers (Table 3-1) | B00h | Core Registers (Table 3-1) | B80h | Core Registers (Table 3-1) |
| 80Bh | | 88Bh | | 90Bh | | 98Bh | | A0Bh | | A8Bh | | B0Bh | | B8Bh | |
| 80Ch | AT1RESL | 88Ch | AT1CLK | 90Ch | Unimplemented Read as '0' | 98Ch | Unimplemented Read as '0' | A0Ch | Unimplemented Read as '0' | A8Ch | Unimplemented Read as '0' | B0Ch | Unimplemented Read as '0' | B8Ch | Unimplemented Read as '0' |
| 80Dh | AT1RESH | 88Dh | AT1SIG | | | | | | | | | | | | |
| 80Eh | AT1MISSL | 88Eh | AT1CSEL1 | | | | | | | | | | | | |
| 80Fh | AT1MISSH | 88Fh | AT1CC1L | | | | | | | | | | | | |
| 810h | AT1PERL | 890h | AT1CC1H | | | | | | | | | | | | |
| 811h | AT1PERH | 891h | AT1CCON1 | | | | | | | | | | | | |
| 812h | AT1PHSL | 892h | AT1CSEL2 | | | | | | | | | | | | |
| 813h | AT1PHSH | 893h | AT1CC2L | | | | | | | | | | | | |
| 814h | AT1CON0 | 894h | AT1CC2H | | | | | | | | | | | | |
| 815h | AT1CON1 | 895h | AT1CCON2 | | | | | | | | | | | | |
| 816h | AT1IRO | 896h | AT1CSEL2 | | | | | | | | | | | | |
| 817h | AT1IE0 | 897h | AT1CC3L | | | | | | | | | | | | |
| 818h | AT1IR1 | 898h | AT1CC3H | | | | | | | | | | | | |
| 819h | AT1IE1 | 899h | AT1CCON3 | | | | | | | | | | | | |
| 81Ah | AT1STPTL | 89Ah | | | | | | | | | | | | | |
| 81Bh | AT1STPTH | | | | | | | | | | | | | | |
| 81Ch | AT1ERRL | | | | | | | | | | | | | | |
| 81Dh | AT1ERRH | | | | | | | | | | | | | | |
| 86Fh | | 8EFh | | 96Fh | | 9EFh | | A6Fh | | AEFh | | B6Fh | | BEFh | |
| 870h | Accesses 70h – 7Fh | 8F0h | Accesses 70h – 7Fh | 970h | Accesses 70h – 7Fh | 9F0h | Accesses 70h – 7Fh | A70h | Accesses 70h – 7Fh | AF0h | Accesses 70h – 7Fh | B70h | Accesses 70h – 7Fh | BF0h | Accesses 70h – 7Fh |
| 87Fh | | 8FFh | | 97Fh | | 9FFh | | A7Fh | | AFh | | B7Fh | | BFh | |

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

TABLE 3-6: PIC16(L)F1615/9 MEMORY MAP, BANK 24-31

| BANK 24 | | BANK 25 | | BANK 26 | | BANK 27 | | BANK 28 | | BANK 29 | | BANK 30 | | BANK 31 | |
|---------|-------------------------------|---------|-------------------------------|---------|-------------------------------|---------|-------------------------------|---------|-------------------------------|---------|-------------------------------|---------|-------------------------------|---------|-------------------------------|
| C00h | Core Registers (Table 3-1) | C80h | Core Registers (Table 3-1) | D00h | Core Registers (Table 3-1) | D80h | Core Registers (Table 3-1) | E00h | Core Registers (Table 3-1) | E80h | Core Registers (Table 3-1) | F00h | Core Registers (Table 3-1) | F80h | Core Registers (Table 3-1) |
| C0Bh | — | C8Bh | — | D0Bh | — | D8Bh | — | E0Bh | — | E8Bh | — | F0Bh | — | F8Bh | — |
| C0Ch | — | C8Ch | — | D0Ch | — | D8Ch | — | E0Ch | — | E8Ch | — | F0Ch | — | F8Ch | — |
| C0Dh | — | C8Dh | — | D0Dh | — | D8Dh | — | E0Dh | — | E8Dh | — | F0Dh | — | F8Dh | — |
| C0Eh | — | C8Eh | — | D0Eh | — | D8Eh | — | E0Eh | — | E8Eh | — | F0Eh | — | F8Eh | — |
| C0Fh | — | C8Fh | — | D0Fh | — | D8Fh | — | E0Fh | — | E8Fh | — | F0Fh | — | F8Fh | — |
| C10h | — | C90h | — | D10h | — | D90h | — | E10h | — | E90h | — | F10h | — | F90h | — |
| C11h | — | C91h | — | D11h | — | D91h | — | E11h | — | E91h | — | F11h | — | F91h | — |
| C12h | — | C92h | — | D12h | — | D92h | — | E12h | — | E92h | — | F12h | — | F92h | — |
| C13h | — | C93h | — | D13h | — | D93h | — | E13h | — | E93h | — | F13h | — | F93h | — |
| C14h | — | C94h | — | D14h | — | D94h | — | E14h | — | E94h | — | F14h | — | F94h | — |
| C15h | — | C95h | — | D15h | — | D95h | — | E15h | — | E95h | — | F15h | — | F95h | — |
| C16h | — | C96h | — | D16h | — | D96h | — | E16h | — | E96h | — | F16h | — | F96h | — |
| C17h | — | C97h | — | D17h | — | D97h | — | E17h | — | E97h | — | F17h | — | F97h | — |
| C18h | — | C98h | — | D18h | — | D98h | — | E18h | — | E98h | — | F18h | — | F98h | — |
| C19h | — | C99h | — | D19h | — | D99h | — | E19h | — | E99h | — | F19h | — | F99h | — |
| C1Ah | — | C9Ah | — | D1Ah | — | D9Ah | — | E1Ah | — | E9Ah | — | F1Ah | — | F9Ah | — |
| C1Bh | — | C9Bh | — | D1Bh | — | D9Bh | — | E1Bh | — | E9Bh | — | F1Bh | — | F9Bh | — |
| C1Ch | — | C9Ch | — | D1Ch | — | D9Ch | — | E1Ch | — | E9Ch | — | F1Ch | — | F9Ch | — |
| C1Dh | — | C9Dh | — | D1Dh | — | D9Dh | — | E1Dh | — | E9Dh | — | F1Dh | — | F9Dh | — |
| C1Eh | — | C9Eh | — | D1Eh | — | D9Eh | — | E1Eh | — | E9Eh | — | F1Eh | — | F9Eh | — |
| C1Fh | — | C9Fh | — | D1Fh | — | D9Fh | — | E1Fh | — | E9Fh | — | F1Fh | — | F9Fh | — |
| C20h | — | CA0h | — | D20h | — | DA0h | — | E20h | — | EA0h | — | F20h | — | FA0h | — |
| | Unimplemented Read as '0' | | Unimplemented Read as '0' | | Unimplemented Read as '0' | | | | | | | | | | |
| C6Fh | — | CEFh | — | D6Fh | — | DEFh | — | E6Fh | — | EEFh | — | F6Fh | — | FEFh | — |
| C70h | — | CF0h | — | D70h | — | DF0h | — | E70h | — | EF0h | — | F70h | — | FF0h | — |
| | Accesses 70h – 7Fh | | Accesses 70h – 7Fh | | Accesses 70h – 7Fh | | Accesses 70h – 7Fh | | Accesses 70h – 7Fh | | Accesses 70h – 7Fh | | Accesses 70h – 7Fh | | Accesses 70h – 7Fh |
| CFFh | — | CFh | — | D7Fh | — | DFh | — | E7Fh | — | EFh | — | F7Fh | — | FFh | — |

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

TABLE 3-7: PIC16(L)F1615/9 MEMORY MAP, BANK 27

| Bank 27 | |
|---------|----------|
| D8Ch | SMT1TMRL |
| D8Dh | SMT1TMRH |
| D8Eh | SMT1TMRU |
| D8Fh | SMT1CPRL |
| D90h | SMT1CPRH |
| D91h | SMT1CPRU |
| D92h | SMT1CPWL |
| D93h | SMT1CPWH |
| D94h | SMT1CPWU |
| D95h | SMT1PRL |
| D96h | SMT1PRH |
| D97h | SMT1PRU |
| D98h | SMT1CON0 |
| D99h | SMT1CON1 |
| D9Ah | SMT1STAT |
| D9Bh | SMT1CLK |
| D9Ch | SMT1SIG |
| D9Dh | SMT1WIN |
| D9Eh | SMT2TMRL |
| D9Fh | SMT2TMRH |
| DA0h | SMT2TMRU |
| DA1h | SMT2CPRL |
| DA2h | SMT2CPRH |
| DA3h | SMT2CPRU |
| DA4h | SMT2CPWL |
| DA5h | SMT2CPWH |
| DA6h | SMT2CPWU |
| DA7h | SMT2PRL |
| DA8h | SMT2PRH |
| DA9h | SMT2PRU |
| DAAh | SMT2CON0 |
| DABh | SMT2CON1 |
| DACH | SMT2STAT |
| DADh | SMT2CLK |
| DAEh | SMT2SIG |
| DAFh | SMT2WIN |
| DB0h | — |
| DEFh | — |

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

TABLE 3-8: PIC16(L)F1615/9 MEMORY MAP, BANK 28

| Bank 28 | |
|---------|------------|
| E0Ch | --- |
| E0Dh | --- |
| E0Eh | --- |
| E0Fh | PPSLOCK |
| E10h | INTPPS |
| E11h | T0CKIPPS |
| E12h | T1CKIPPS |
| E13h | T1GPPS |
| E14h | CCP1PPS |
| E15h | CCP2PPS |
| E16h | ATINPPS |
| E17h | CWGINPPS |
| E18h | T2PPS |
| E19h | T3CKIPPS |
| E1Ah | T3GPPS |
| E1Bh | T4PPS |
| E1Ch | T5CKIPPS |
| E1Dh | T5GPPS |
| E1Eh | T6PPS |
| E1Fh | ATCC1PPS |
| E20h | SSPCLKPPS |
| E21h | SSPDATPPS |
| E22h | SSPSSPPS |
| E23h | ATCC2PPS |
| E24h | RXPPS |
| E25h | CKPPS |
| E26h | SMT1SIGPPS |
| E27h | SMT1WINPPS |
| E28h | CLCIN0PPS |
| E29h | CLCIN1PPS |
| E2Ah | CLCIN2PPS |
| E2Bh | CLCIN3PPS |
| E2Ch | SMT2SIGPPS |
| E2Dh | SMT2WINPPS |
| E2Eh | ATCC3PPS |
| E2Fh | — |
| E6Fh | — |

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

TABLE 3-9: PIC16(L)F1615 MEMORY MAP, BANK 29

| Bank 29 | |
|---------|--------|
| E8Ch | --- |
| E8Dh | --- |
| E8Eh | --- |
| E8Fh | --- |
| E90h | RA0PPS |
| E91h | RA1PPS |
| E92h | RA2PPS |
| E93h | --- |
| E94h | RA4PPS |
| E95h | RA5PPS |
| E96h | --- |
| E97h | --- |
| E98h | --- |
| E99h | --- |
| E9Ah | --- |
| E9Bh | --- |
| E9Ch | --- |
| E9Dh | --- |
| E9Eh | --- |
| E9Fh | --- |
| EA0h | RC0PPS |
| EA1h | RC1PPS |
| EA2h | RC2PPS |
| EA3h | RC3PPS |
| EA4h | RC4PPS |
| EA5h | RC5PPS |
| EA6h | --- |
| EEFh | --- |

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

TABLE 3-10: PIC16(L)F1619 MEMORY MAP, BANK 29

| Bank 29 | |
|---------|--------|
| E8Ch | --- |
| E8Dh | --- |
| E8Eh | --- |
| E8Fh | --- |
| E90h | RA0PPS |
| E91h | RA1PPS |
| E92h | RA2PPS |
| E93h | --- |
| E94h | RA4PPS |
| E95h | RA5PPS |
| E96h | --- |
| E97h | --- |
| E98h | --- |
| E99h | --- |
| E9Ah | --- |
| E9Bh | --- |
| E9Ch | RB4PPS |
| E9Dh | RB5PPS |
| E9Eh | RB6PPS |
| E9Fh | RB7PPS |
| EA0h | RC0PPS |
| EA1h | RC1PPS |
| EA2h | RC2PPS |
| EA3h | RC3PPS |
| EA4h | RC4PPS |
| EA5h | RC5PPS |
| EA6h | RC6PPS |
| EA7h | RC7PPS |
| EA8h | --- |
| EEFh | --- |

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

TABLE 3-11: PIC16(L)F1615/9 MEMORY MAP, BANK 30

| Bank 30 | |
|---------|----------|
| F0Ch | --- |
| F0Dh | --- |
| F0Eh | --- |
| F0Fh | CLCDATA |
| F10h | CLC1CON |
| F11h | CLC1POL |
| F12h | CLC1SEL0 |
| F13h | CLC1SEL1 |
| F14h | CLC1SEL2 |
| F15h | CLC1SEL3 |
| F16h | CLC1GLS0 |
| F17h | CLC1GLS1 |
| F18h | CLC1GLS2 |
| F19h | CLC1GLS3 |
| F1Ah | CLC2CON |
| F1Bh | CLC2POL |
| F1Ch | CLC2SEL0 |
| F1Dh | CLC2SEL1 |
| F1Eh | CLC2SEL2 |
| F1Fh | CLC2SEL3 |
| F20h | CLC2GLS0 |
| F21h | CLC2GLS1 |
| F22h | CLC2GLS2 |
| F23h | CLC2GLS3 |
| F24h | CLC3CON |
| F25h | CLC3POL |
| F26h | CLC3SEL0 |
| F27h | CLC3SEL1 |
| F28h | CLC3SEL2 |
| F29h | CLC3SEL3 |
| F2Ah | CLC3GLS0 |
| F2Bh | CLC3GLS1 |
| F2Ch | CLC3GLS2 |
| F2Dh | CLC3GLS3 |
| F2Eh | CLC4CON |
| F2Fh | CLC4POL |
| F30h | CLC4SEL0 |
| F31h | CLC4SEL1 |
| F32h | CLC4SEL2 |
| F33h | CLC4SEL3 |
| F34h | CLC4GLS0 |
| F35h | CLC4GLS1 |
| F36h | CLC4GLS2 |
| F37h | CLC4GLS3 |
| F38h | --- |
| F6Fh | --- |

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

TABLE 3-12: PIC16(L)F1615/9 MEMORY MAP, BANK 31

| Bank 31 | |
|---------|------------------------------|
| F8Ch | Unimplemented Read as '0' |
| 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 |

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

3.3.6 CORE FUNCTION REGISTERS SUMMARY

The Core Function registers listed in [Table 3-13](#) can be addressed from any Bank.

TABLE 3-13: 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 ruuu | |
| 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 | — | — | — | BSR<4:0> | | | | --- | 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 | IOCFIE | TMR0IF | INTF | IOCFIF | 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'.

TABLE 3-14: 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 | — | — | RA5 | RA4 | RA3 | RA2 | RA1 | RA0 | --xx xxxx | --xx xxxx |
| 00Dh | PORTB ⁽⁴⁾ | RB7 | RB6 | RB5 | RB4 | — | — | — | — | xxxx ---- | xxxx ---- |
| 00Eh | PORTC | RC7 ⁽⁴⁾ | RC6 ⁽⁴⁾ | RC5 | RC4 | RC3 | RC2 | RC1 | RC0 | xxxx xxxx | xxxx xxxx |
| 00Fh | — | Unimplemented | | | | | | | | — | — |
| 010h | PIR1 | TMR1GIF | ADIF | RCIF | TXIF | SSP1IF | CCP1IF | TMR2IF | TMR1IF | 0000 0000 | 0000 0000 |
| 011h | PIR2 | OSFIF | C2IF | C1IF | — | BCL1IF | TMR6IF | TMR4IF | CCP2IF | -00- 0000 | -00- 0000 |
| 012h | PIR3 | — | — | CWGIF | ZCDIF | CLC4IF | CLC3IF | CLC2IF | CLC1IF | --00 0000 | --00 0000 |
| 013h | PIR4 | SCANIF | CRCIF | SMT2PWAIF | SMT2PRAIF | SMT2IF | SMT1PWAIF | SMT1PRAIF | SMT1IF | 0000 0000 | 0000 0000 |
| 014h | PIR5 | TMR3GIF | TMR3IF | TMR5GIF | TMR5IF | — | AT1IF | PID1EIF | PID1DIF | 0000 -000 | 0000 -000 |
| 015h | TMR0 | Holding Register for the 8-bit Timer0 Count | | | | | | | | xxxx xxxx | uuuu uuuu |
| 016h | TMR1L | Holding Register for the Least Significant Byte of the 16-bit TMR1 Count | | | | | | | | xxxx xxxx | uuuu uuuu |
| 017h | TMR1H | Holding Register for the Most Significant Byte of the 16-bit TMR1 Count | | | | | | | | xxxx xxxx | uuuu uuuu |
| 018h | T1CON | TMR1CS<1:0> | | T1CKPS<1:0> | | — | T1SYN \bar{C} | — | TMR1ON | 0000 -0-0 | uuuu -u-u |
| 019h | T1GCON | TMR1GE | T1GPOL | T1GTM | T1GSPM | T1GGO/ DONE | T1GVAL | T1GSS<1:0> | | 0000 0x00 | uuuu uxuu |
| 01Ah | TMR2 | Timer2 Module Register | | | | | | | | 0000 0000 | 0000 0000 |
| 01Bh | PR2 | Timer2 Period Register | | | | | | | | 1111 1111 | 1111 1111 |
| 01Ch | T2CON | ON | CKPS<2:0> | | | OUTPS<3:0> | | | 0000 0000 | 0000 0000 | |
| 01Dh | T2HLT | PSYNC | CKPOL | CKSYNC | MODE<4:0> | | | 0000 0000 | 0000 0000 | | |
| 01Eh | T2CLKCON | — | — | — | — | CS<3:0> | | | ---- 0000 | ---- 0000 | |
| 01Fh | T2RST | — | — | — | — | RSEL<3:0> | | | ---- 0000 | ---- 0000 | |

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

- Note** 1: PIC16F1615/9 only.
 2: Unimplemented, read as '1'.
 3: PIC16(L)F1615 only.
 4: PIC16(L)F1619 only.

TABLE 3-14: 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 1 | | | | | | | | | | | | |
| 08Ch | TRISA | — | — | TRISA5 | TRISA4 | — ⁽²⁾ | TRISA2 | TRISA1 | TRISA0 | --11 1111 | --11 1111 | |
| 08Dh | TRISB ⁽⁴⁾ | TRISB7 | TRISB6 | TRISB5 | TRISB4 | — | — | — | — | 1111 ---- | 1111 ---- | |
| 08Eh | TRISC | TRISC7 ⁽⁴⁾ | TRISC6 ⁽⁴⁾ | TRISC5 | TRISC4 | TRISC3 | TRISC2 | TRISC1 | TRISC0 | 1111 1111 | 1111 1111 | |
| 08Fh | — | Unimplemented | | | | | | | | — | — | |
| 090h | — | Unimplemented | | | | | | | | — | — | |
| 090h | PIE1 | TMR1GIE | ADIE | RCIE | TXIE | SSP1IE | CCP1IE | TMR2IE | TMR1IE | 0000 0000 | 0000 0000 | |
| 091h | PIE2 | OSFIE | C2IE | C1IE | — | BCL1IE | TMR6IE | TMR4IE | CCP2IE | -00- 0000 | -00- 0000 | |
| 092h | PIE3 | — | — | CWGIE | ZCDIE | CLC4IE | CLC3IE | CLC2IE | CLC1IE | --00 0000 | --00 0000 | |
| 093h | PIE4 | SCANIE | CRCIE | SMT2PWAIE | SMT2PRAIE | SMT2IE | SMT1PWAIE | SMT1PRAIE | SMT1IE | 0000 0000 | 0000 0000 | |
| 094h | PIE5 | TMR3GIE | TMR3IE | TMR5GIE | TMR5IE | — | AT1IE | PID1EIE | PID1DIE | 0000 -000 | 0000 -000 | |
| 095h | OPTION_REG | WPUEN | INTEDG | TMR0CS | TMR0SE | PSA | PS<2:0> | | | 1111 1111 | 1111 1111 | |
| 096h | PCON | STKOVF | STKUNF | WDTWV | RWDT | RMCLR | R \bar{I} | POR | BOR | 00-1 11qq | qq-b qquu | |
| 097h | — | Unimplemented | | | | | | | | — | — | |
| 098h | OSCTUNE | — | — | TUN<5:0> | | | | | — | — | --00 0000 | --00 0000 |
| 099h | OSCCON | SPLLEN | IRCF<3:0> | | | | — | SCS<1:0> | | | 0011 1-00 | 0011 1-00 |
| 09Ah | OSCSTAT | — | PLL | OSTS | HFIOFR | HFIOFL | MFIOFR | LFIOFR | HFIOFS | -000 0000 | -qqq qqbb | |
| 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 | ADCON2 | TRIGSEL<4:0> | | | | | — | — | — | — | 0000 0--- | 0000 0--- |

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

- Note** 1: PIC16F1615/9 only.
 2: Unimplemented, read as '1'.
 3: PIC16(L)F1615 only.
 4: PIC16(L)F1619 only.

TABLE 3-14: 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 | — | — | LATA5 | LATA4 | LATA3 | LATA2 | LATA1 | LATA0 | --xx xxxx | --uu uuuu | |
| 10Dh | LATB ⁽⁴⁾ | LATB7 | LATB6 | LATB5 | LATB4 | — | — | — | — | xxxx ---- | uuuu ---- | |
| 10Eh | LATC | LATC7 ⁽⁴⁾ | LATC6 ⁽⁴⁾ | LATC5 | LATC4 | LATC3 | LATC2 | LATC1 | LATC0 | xxxx xxxx | uuuu uuuu | |
| 10Fh | — | Unimplemented | | | | | | | | | — | — |
| 110h | — | Unimplemented | | | | | | | | | — | — |
| 111h | CM1CON0 | C1ON | C1OUT | — | C1POL | — | C1SP | C1HYS | C1SYNC | 00-0 -100 | 00-0 -100 | |
| 112h | CM1CON1 | C1INTP | C1INTN | C1PCH<1:0> | | — | C1NCH<2:0> | | | 0000 -000 | 0000 -000 | |
| 113h | CM2CON0 ⁽⁴⁾ | C2ON | C2OUT | — | C2POL | — | C2SP | C2HYS | C2SYNC | 00-0 -100 | 00-0 -100 | |
| 114h | CM2CON1 ⁽⁴⁾ | C2INTP | C2INTN | C2PCH<1:0> | | — | C2NCH<2:0> | | | 0000 -000 | 0000 -000 | |
| 115h | CMOUT | — | — | — | — | — | — | MC2OUT | MC1OUT | ---- --00 | ---- --00 | |
| 116h | BORCON | SBOREN | BORFS | — | — | — | — | — | BORRDY | 10-- ---q | uu-- ---u | |
| 117h | FVRCON | FVREN | FVRRDY | TSEN | TSRNG | CDAFVR<1:0> | | ADFVR<1:0> | | 0q00 0000 | 0q00 0000 | |
| 118h | DAC1CON0 | DAC1EN | — | DAC1OE1 | — | DAC1PSS<1:0> | | — | — | 0-0- 00-- | 0-0- 00-- | |
| 119h | DAC1CON1 | DAC1R<7:0> | | | | | | | | 0000 0000 | 0000 0000 | |
| 11Ah | — | Unimplemented | | | | | | | | | — | — |
| 11Bh | — | Unimplemented | | | | | | | | | — | — |
| 11Ch | ZCD1CON | ZCD1EN | — | ZCD1OUT | ZCD1POL | — | — | ZCD1INTP | ZCD1INTN | 0-00 --00 | 0-00 --00 | |
| 11Dh | — | Unimplemented | | | | | | | | | — | — |
| 11Eh | — | Unimplemented | | | | | | | | | — | — |
| 11Fh | — | Unimplemented | | | | | | | | | — | — |

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

- Note 1:** PIC16F1615/9 only.
Note 2: Unimplemented, read as '1'.
Note 3: PIC16(L)F1615 only.
Note 4: PIC16(L)F1619 only.

TABLE 3-14: 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 3 | | | | | | | | | | | | |
| 18Ch | ANSELA | — | — | — | ANSA4 | — | ANSA2 | ANSA1 | ANSA0 | ---1 -111 | ---1 -111 | |
| 18Dh | ANSELB ⁽⁴⁾ | — | — | ANSB5 | ANSB4 | — | — | — | — | --11 ---- | --11 ---- | |
| 18Eh | ANSELC | ANSC7 ⁽⁴⁾ | ANSC6 ⁽⁴⁾ | — | — | ANSC3 | ANSC2 | ANSC1 | ANSC0 | 11-- 1111 | 11-- 1111 | |
| 18Fh | — | Unimplemented | | | | | | | | | — | — |
| 190h | — | Unimplemented | | | | | | | | | — | — |
| 191h | PMADRL | Flash Program Memory Address Register Low Byte | | | | | | | | 0000 0000 | 0000 0000 | |
| 192h | PMADRH | — ⁽²⁾ | Flash Program Memory Address Register High Byte | | | | | | | | 1000 0000 | 1000 0000 |
| 193h | PMDATL | Flash Program Memory Read Data Register Low Byte | | | | | | | | xxxx xxxx | uuuu uuuu | |
| 194h | PMDATH | — | — | Flash Program Memory Read Data Register High Byte | | | | | | --xx xxxx | --uu uuuu | |
| 195h | PMCON1 | — ⁽²⁾ | CFGS | LWLO | FREE | WRERR | WREN | WR | RD | 1000 x000 | 1000 q000 | |
| 196h | PMCON2 | Flash Program Memory Control Register 2 | | | | | | | | 0000 0000 | 0000 0000 | |
| 197h | VREGCON ⁽¹⁾ | — | — | — | — | — | — | VREGPM | Reserved | ---- --01 | ---- --01 | |
| 198h | — | Unimplemented | | | | | | | | | — | — |
| 199h | RC1REG | EUSART Receive Data Register | | | | | | | | 0000 0000 | 0000 0000 | |
| 19Ah | TX1REG | EUSART Transmit Data Register | | | | | | | | 0000 0000 | 0000 0000 | |
| 19Bh | SP1BRGL | Baud Rate Generator Data Register Low | | | | | | | | 0000 0000 | 0000 0000 | |
| 19Ch | SP1BRGH | Baud Rate Generator Data Register High | | | | | | | | 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, r = reserved. Shaded locations are unimplemented, read as '0'.

- Note 1:** PIC16F1615/9 only.
Note 2: Unimplemented, read as '1'.
Note 3: PIC16(L)F1615 only.
Note 4: PIC16(L)F1619 only.

TABLE 3-14: 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 | WPUA | — | — | WPUA5 | WPUA4 | WPUA3 | WPUA2 | WPUA1 | WPUA0 | --11 1111 | --11 1111 | |
| 20Dh | WPUB ⁽⁴⁾ | WPUB7 | WPUB6 | WPUB5 | WPUB4 | — | — | — | — | 1111 ---- | 1111 ---- | |
| 20Eh | WPUC | WPUC7 ⁽⁴⁾ | WPUC6 ⁽⁴⁾ | WPUC5 | WPUC4 | WPUC3 | WPUC2 | WPUC1 | WPUC0 | 1111 1111 | 111 1111 | |
| 20Fh | — | Unimplemented | | | | | | | | | — | — |
| 210h | — | Unimplemented | | | | | | | | | — | — |
| 211h | SSP1BUF | Synchronous Serial Port Receive Buffer/Transmit Register | | | | | | | | xxxx xxxx | xxxx xxxx | |
| 212h | SSP1ADD | ADD<7:0> | | | | | | | | 0000 0000 | 0000 0000 | |
| 213h | SSP1MSK | MSK<7:0> | | | | | | | | 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 to 21Fh | — | Unimplemented | | | | | | | | | — | — |

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

- Note** 1: PIC16F1615/9 only.
 2: Unimplemented, read as '1'.
 3: PIC16(L)F1615 only.
 4: PIC16(L)F1619 only.

TABLE 3-14: 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 5 | | | | | | | | | | | |
| 28Ch | ODCONA | — | — | ODA5 | ODA4 | — | ODA2 | ODA1 | ODA0 | --00 -000 | --00 -000 |
| 28Dh | ODCONB ⁽⁴⁾ | ODB7 | ODB6 | ODB5 | ODB4 | — | — | — | — | 0000 ---- | 0000 ---- |
| 28Eh | ODCONC | ODC7 ⁽⁴⁾ | ODC6 ⁽⁴⁾ | ODC5 | ODC4 | ODC3 | ODC2 | ODC1 | ODC0 | 0000 0000 | 0000 0000 |
| 28Fh | — | Unimplemented | | | | | | | | — | — |
| 290h | — | Unimplemented | | | | | | | | — | — |
| 291h | CCP1RL | Capture/Compare/PWM 1 Register (LSB) | | | | | | | | xxxx xxxx | uuuu uuuu |
| 292h | CCP1RH | Capture/Compare/PWM 1 Register (MSB) | | | | | | | | xxxx xxxx | uuuu uuuu |
| 293h | CCP1CON | EN | — | OUT | FMT | MODE<3:0> | | | 0000 0000 | 0000 0000 | |
| 294h | CCP1CAP | — | — | — | — | — | CTS<2:0> | | | ---- -000 | ---- -000 |
| 295h — 297h | — | Unimplemented | | | | | | | | — | — |
| 298h | CCP2RL | Capture/Compare/PWM 2 Register (LSB) | | | | | | | | xxxx xxxx | uuuu uuuu |
| 299h | CCP2RH | Capture/Compare/PWM 2 Register (MSB) | | | | | | | | xxxx xxxx | uuuu uuuu |
| 29Ah | CCP2CON | EN | — | OUT | FMT | MODE<3:0> | | | 0000 0000 | 0000 0000 | |
| 29Bh | CCP2CAP | — | — | — | — | — | CTS<2:0> | | | ---- -000 | ---- -000 |
| 29Ch | — | Unimplemented | | | | | | | | — | — |
| 29Dh | — | Unimplemented | | | | | | | | — | — |
| 29Eh | CCPTMRS | P4TSEL<1:0> | | P3TSEL<1:0> | | C2TSEL<1:0> | | C1TSEL<1:0> | | 0000 0000 | 0000 0000 |
| 29Fh | — | Unimplemented | | | | | | | | — | — |
| Bank 6 | | | | | | | | | | | |
| 30Ch | SLRCONA | — | — | SLRA5 | SLRA4 | — | SLRA2 | SLRA1 | SLRA0 | --11 -111 | --11 -111 |
| 30Dh | SLRCONB ⁽⁴⁾ | SLRB7 | SLRB6 | SLRB5 | SLRB4 | — | — | — | — | 1111 ---- | 1111 ---- |
| 30Eh | SLRCONC | SLRC7 ⁽⁴⁾ | SLRC6 ⁽⁴⁾ | SLRC5 | SLRC4 | SLRC3 | SLRC2 | SLRC1 | SLRC0 | 1111 1111 | 1111 1111 |
| 30Fh — 31Fh | — | Unimplemented | | | | | | | | — | — |

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

- Note** 1: PIC16F1615/9 only.
 2: Unimplemented, read as '1'.
 3: PIC16(L)F1615 only.
 4: PIC16(L)F1619 only.

TABLE 3-14: 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 7 | | | | | | | | | | | |
| 38Ch | INLVLA | — | — | INLVLA5 | INLVLA4 | INLVLA3 | INLVLA2 | INLVLA1 | INLVLA0 | --11 1111 | --11 1111 |
| 38Dh | INLVLB ⁽⁴⁾ | INLVLB7 | INLVLB6 | INLVLB5 | INLVLB4 | — | — | — | — | 1111 ---- | 1111 ---- |
| 38Eh | INLVLC | INLVLC7 ⁽⁴⁾ | INLVLC6 ⁽⁴⁾ | INLVLC5 | INLVLC4 | INLVLC3 | INLVLC2 | INLVLC1 | INLVLC0 | 1111 1111 | 1111 1111 |
| 30Fh | — | Unimplemented | | | | | | | | — | — |
| 390h | — | Unimplemented | | | | | | | | — | — |
| 391h | IOCAP | — | — | IOCAP5 | IOCAP4 | IOCAP3 | IOCAP2 | IOCAP1 | IOCAP0 | --00 0000 | --00 0000 |
| 392h | IOCAN | — | — | IOCAN5 | IOCAN4 | IOCAN3 | IOCAN2 | IOCAN1 | IOCAN0 | --00 0000 | --00 0000 |
| 393h | IOCAF | — | — | IOCAF5 | IOCAF4 | IOCAF3 | IOCAF2 | IOCAF1 | IOCAF0 | --00 0000 | --00 0000 |
| 394h | IOCBP ⁽⁴⁾ | IOCBP7 | IOCBP6 | IOCBP5 | IOCBP4 | — | — | — | — | 0000 ---- | 0000 ---- |
| 395h | IOCBN ⁽⁴⁾ | IOCBN7 | IOCBN6 | IOCBN5 | IOCBN4 | — | — | — | — | 0000 ---- | 0000 ---- |
| 396h | IOCBF ⁽⁴⁾ | IOCBF7 | IOCBF6 | IOCBF5 | IOCBF4 | — | — | — | — | 0000 ---- | 0000 ---- |
| 397h | IOCCP | IOCCP7 ⁽⁴⁾ | IOCCP6 ⁽⁴⁾ | IOCCP5 | IOCCP4 | IOCCP3 | IOCCP2 | IOCCP1 | IOCCP0 | 0000 0000 | 0000 0000 |
| 398h | IOCCN | IOCCN7 ⁽⁴⁾ | IOCCN6 ⁽⁴⁾ | IOCCN5 | IOCCN4 | IOCCN3 | IOCCN2 | IOCCN1 | IOCCN0 | 0000 0000 | 0000 0000 |
| 399h | IOCCF | IOCCF7 ⁽⁴⁾ | IOCCF6 ⁽⁴⁾ | IOCCF5 | IOCCF4 | IOCCF3 | IOCCF2 | IOCCF1 | IOCCF0 | 0000 0000 | 0000 0000 |
| 39Ah to 39Fh | — | Unimplemented | | | | | | | | — | — |

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

- Note** 1: PIC16F1615/9 only.
 2: Unimplemented, read as '1'.
 3: PIC16(L)F1615 only.
 4: PIC16(L)F1619 only.

TABLE 3-14: 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 | — | Unimplemented | | | | | | | | | — | — |
| 40Dh | — | Unimplemented | | | | | | | | | — | — |
| 40Eh | HIDRVC | — | — | HIDC5 | HIDC4 | — | — | — | — | --00 ---- | --00 ---- | |
| 40Fh to 412h | — | Unimplemented | | | | | | | | | — | — |
| 413h | TMR4 | Timer4 Module Register | | | | | | | | | 0000 0000 | 0000 0000 |
| 414h | PR4 | Timer4 Period Register | | | | | | | | | 1111 1111 | 1111 1111 |
| 415h | T4CON | ON | CKPS<2:0> | | | OUTPS<3:0> | | | 0000 0000 | | | |
| 416h | T4HLT | PSYNC | CKPOL | CKSYNC | MODE<4:0> | | | 0000 0000 | | | | |
| 417h | T4CLKCON | — | — | — | — | CS<3:0> | | | ---- 0000 | | | |
| 418h | T4RST | — | — | — | — | RSEL<3:0> | | | ---- 0000 | | | |
| 419h | — | Unimplemented | | | | | | | | | — | — |
| 41Ah | TMR6 | Timer6 Module Register | | | | | | | | | 0000 0000 | 0000 0000 |
| 41Bh | PR6 | Timer6 Period Register | | | | | | | | | 1111 1111 | 1111 1111 |
| 41Ch | T6CON | ON | CKPS<2:0> | | | OUTPS<3:0> | | | 0000 0000 | | | |
| 41Dh | T6HLT | PSYNC | CKPOL | CKSYNC | MODE<4:0> | | | 0000 0000 | | | | |
| 41Eh | T6CLKCON | — | — | — | — | CS<3:0> | | | ---- 0000 | | | |
| 41Fh | T6RST | — | — | — | — | RSEL<3:0> | | | ---- 0000 | | | |

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

- Note** 1: PIC16F1615/9 only.
 2: Unimplemented, read as '1'.
 3: PIC16(L)F1615 only.
 4: PIC16(L)F1619 only.

TABLE 3-14: 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 9 | | | | | | | | | | | |
| 48Ch to 492h | — | Unimplemented | | | | | | | | — | — |
| 493h | TMR3L | Timer3 Module Register | | | | | | | | xxxx xxxx | xxxx xxxx |
| 494h | TMR3H | Timer3 Module Register | | | | | | | | xxxx xxxx | xxxx xxxx |
| 495h | T3CON | TMR3CS<1:0> | | T3CKPS<1:0> | | — | T3SYNC | — | TMR3CON | xxxx -x-x | xxxx -x-x |
| 496h | T3GCON | TMR3GE | T3GPOL | T3GTM | T3GSPM | T3GGO/ DONE | T3GVAL | T3GSS<1:0> | | xxxx xxxx | xxxx xxxx |
| 497h to 499h | — | Unimplemented | | | | | | | | — | — |
| 49Ah | TMR5L | Timer5 Module Register | | | | | | | | xxxx xxxx | xxxx xxxx |
| 49Bh | TMR5H | Timer5 Module Register | | | | | | | | xxxx xxxx | xxxx xxxx |
| 49Ch | T5CON | TMR5CS<1:0> | | T5CKPS<1:0> | | — | T5SYNC | — | TMR5CON | xxxx -x-x | xxxx -x-x |
| 49Dh | T5GCON | TMR5GE | T5GPOL | T5GTM | T5GSPM | T5GGO/ DONE | T5GVAL | T5GSS<1:0> | | xxxx xxxx | xxxx xxxx |
| 49Eh | — | Unimplemented | | | | | | | | — | — |
| 49Fh | — | Unimplemented | | | | | | | | — | — |
| Bank 10 | | | | | | | | | | | |
| 50Ch to 51Fh | — | Unimplemented | | | | | | | | — | — |

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

- Note** 1: PIC16F1615/9 only.
 2: Unimplemented, read as '1'.
 3: PIC16(L)F1615 only.
 4: PIC16(L)F1619 only.

TABLE 3-14: 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 11 | | | | | | | | | | | |
| 58Ch | PID1SELT | | | | | | | | | xxxx xxxx | xxxx xxxx |
| 58Dh | PID1SETH | | | | | | | | | xxxx xxxx | xxxx xxxx |
| 58Eh | PID1INL | | | | | | | | | 0000 0000 | 0000 0000 |
| 58Fh | PID1INH | | | | | | | | | 0000 0000 | 0000 0000 |
| 590h | PID1K1L | | | | | | | | | xxxx xxxx | xxxx xxxx |
| 591h | PID1K1H | | | | | | | | | xxxx xxxx | xxxx xxxx |
| 592h | PID1K2L | | | | | | | | | xxxx xxxx | xxxx xxxx |
| 593h | PID1K2H | | | | | | | | | xxxx xxxx | xxxx xxxx |
| 594h | PID1K3L | | | | | | | | | xxxx xxxx | xxxx xxxx |
| 595h | PID1K3H | | | | | | | | | xxxx xxxx | xxxx xxxx |
| 596h | PID1OUTLL | | | | | | | | | 0000 0000 | 0000 0000 |
| 597h | PID1OUTLH | | | | | | | | | 0000 0000 | 0000 0000 |
| 598h | PID1OUTH | | | | | | | | | 0000 0000 | 0000 0000 |
| 599h | PID1OUTH | | | | | | | | | 0000 0000 | 0000 0000 |
| 59Ah | PID1OUTU | — | — | — | — | | | | | ---- 0000 | ---- 0000 |
| 59Bh | PID1Z1L | | | | | | | | | 0000 0000 | 0000 0000 |
| 59Ch | PID1Z1H | | | | | | | | | 0000 0000 | 0000 0000 |
| 59Dh | PID1Z1U | — | — | — | — | — | — | — | Z116 | ---- ---0 | ---- ---0 |
| 59Eh | — | Unimplemented | | | | | | | | — | — |
| 59Fh | — | Unimplemented | | | | | | | | — | — |

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

- Note 1:** PIC16F1615/9 only.
Note 2: Unimplemented, read as '1'.
Note 3: PIC16(L)F1615 only.
Note 4: PIC16(L)F1619 only.

TABLE 3-14: 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 12 | | | | | | | | | | | |
| 60Ch | PID1Z2L | Z2<7:0> | | | | | | | | 0000 0000 | 0000 0000 |
| 60Dh | PID1Z2H | Z2<15:8> | | | | | | | | 0000 0000 | 0000 0000 |
| 60Eh | PID1Z2U | — | — | — | — | — | — | — | Z216 | ---- -00 | ---- -00 |
| 60Fh | PID1ACLL | ACC<7:0> | | | | | | | | 0000 0000 | 0000 0000 |
| 610h | PID1ACCLH | ACC<15:8> | | | | | | | | 0000 0000 | 0000 0000 |
| 611h | PID1ACCHL | ACC<23:16> | | | | | | | | 0000 0000 | 0000 0000 |
| 612h | PID1ACCHH | ACC<31:24> | | | | | | | | 0000 0000 | 0000 0000 |
| 613h | PID1ACCU | — | — | — | — | — | ACC<34:32> | | | ---- -000 | ---- -000 |
| 614h | PID1CON | EN | BUSY | — | — | — | MODE<2:0> | | | 00-- 0000 | 00-- 0000 |
| 615h | — | Unimplemented | | | | | | | | — | — |
| 616h | — | Unimplemented | | | | | | | | — | — |
| 617h | PWM3DCL | DC<1:0> | | — | — | — | — | — | — | xx-- ---- | xx-- ---- |
| 618h | PWM3DCH | DC<9:2> | | | | | | | | xxxx xxxx | xxxx xxxx |
| 619h | PWM3CON | EN | — | OUT | POL | — | — | — | — | 0-x0 ---- | 0-x0 ---- |
| 61Ah | PWM4DCL | DC<1:0> | | — | — | — | — | — | — | xx-- ---- | xx-- ---- |
| 61Bh | PWM4DCH | DC<9:2> | | | | | | | | xxxx xxxx | xxxx xxxx |
| 61Ch | PWM4CON | EN | — | OUT | POL | — | — | — | — | 0-x0 ---- | 0-x0 ---- |
| 61Dh to 61Fh | — | Unimplemented | | | | | | | | — | — |

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

- Note** 1: PIC16F1615/9 only.
 2: Unimplemented, read as '1'.
 3: PIC16(L)F1615 only.
 4: PIC16(L)F1619 only.

TABLE 3-14: 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 13 | | | | | | | | | | | |
| 68Ch to 690h | — | Unimplemented | | | | | | | | — | — |
| 691h | CWG1DBR | — | — | DBR<5:0> | | | | — | — | --00 0000 | --00 0000 |
| 692h | CWG1DBF | — | — | DBF<5:0> | | | | — | — | --xx xxxx | --xx xxxx |
| 693h | CWG1AS0 | SHUTDOWN | REN | LSBD<1:0> | | LSAC<1:0> | | — | — | 0001 01-- | 00001 01-- |
| 694h | CWG1AS1 | — | TMR6AS | TMR4AS | TMR2AS | — | C2AS ⁽⁴⁾ | C1AS | INAS | -000 -000 | -000 -000 |
| 695h | CWG1OCON0 | OVRD | OVRC | OVRB | OVRA | STRD | STRC | STRB | STRA | 0000 0000 | 0000 0000 |
| 696h | CWG1CON0 | EN | LD | — | — | — | MODE<2:0> | | | 00-- -000 | 00-- -000 |
| 697h | CWG1CON1 | — | — | IN | — | POLD | POLC | POLB | POLA | --x- 0000 | --x- 0000 |
| 698h | — | Unimplemented | | | | | | | | — | — |
| 699h | CWG1CLKCON | — | — | — | — | — | — | — | CS | ---- ---0 | ---- ---0 |
| 69Ah | CWG1ISM | — | — | — | — | IS<3:0> | | | | ---- 0000 | ---- 0000 |
| 69Bh to 6EFh | — | Unimplemented | | | | | | | | — | — |

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

- Note** 1: PIC16F1615/9 only.
 2: Unimplemented, read as '1'.
 3: PIC16(L)F1615 only.
 4: PIC16(L)F1619 only.

TABLE 3-14: 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 14 | | | | | | | | | | | |
| 70Ch to 710h | — | Unimplemented | | | | | | | | — | — |
| 711h | WDTCON0 | — | — | WDTPS<4:0> | | | | SEN | — | — | — |
| 712h | WDTCON1 | — | WDTCS<2:0> | | | — | WINDOW<2:0> | | | — | — |
| 713h | WDTPSL | PSCNT<7:0> | | | | | | | | 0000 0000 | 0000 0000 |
| 714h | WDTPSH | PSCNT<15:8> | | | | | | | | 0000 0000 | 0000 0000 |
| 715h | WDTTMR | WDTTMR<4:0> | | | | STATE | PSCNT<17:16> | | | 0000 0000 | 0000 0000 |
| 716h | — | Unimplemented | | | | | | | | — | — |
| 717h | — | Unimplemented | | | | | | | | — | — |
| 718h | SCANLADRL | LADR<7:0> | | | | | | | | 0000 0000 | 0000 0000 |
| 719h | SCANLADRH | LADR<15:8> | | | | | | | | 0000 0000 | 0000 0000 |
| 71Ah | SCANHADRL | HADR<7:0> | | | | | | | | 1111 1111 | 1111 1111 |
| 71Bh | SCANHADRH | HADR<15:8> | | | | | | | | 1111 1111 | 1111 1111 |
| 71Ch | SCANCON0 | EN | SCANGO | BUSY | INVALID | INTM | — | MODE<1:0> | | 0000 0-00 | 0000 0-00 |
| 71Dh | SCANTRIG | TSEL<3:0> | | | | | | | | ---- 0000 | ---- 0000 |
| 71Eh | — | Unimplemented | | | | | | | | — | — |
| 71Fh | — | Unimplemented | | | | | | | | — | — |

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

- Note**
- 1: PIC16F1615/9 only.
 - 2: Unimplemented, read as '1'.
 - 3: PIC16(L)F1615 only.
 - 4: PIC16(L)F1619 only.

TABLE 3-14: 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 |
|-----------------|-----------|---------------|-------|-------|-------|-----------|-------|--------|-------|-------------------|---------------------------|
| Banks 15 | | | | | | | | | | | |
| 78Ch to 790h | — | Unimplemented | | | | | | | | — | — |
| 791h | CRCDATL | DAT<7:0> | | | | | | | | xxxx xxxx | xxxx xxxx |
| 792h | CRCDATH | DAT<15:8> | | | | | | | | xxxx xxxx | xxxx xxxx |
| 793h | CRCACCL | ACC<7:0> | | | | | | | | 0000 0000 | 0000 0000 |
| 794h | CRCACCH | ACC<15:8> | | | | | | | | 0000 0000 | 0000 0000 |
| 795h | CRCSHIFTL | SHIFT<7:0> | | | | | | | | 0000 0000 | 0000 0000 |
| 796h | CRCSHIFTH | SHIFT<15:8> | | | | | | | | 0000 0000 | 0000 0000 |
| 797h | CRCXORL | XOR<7:1> | | | | | | | — | xxxx xxx- | xxxx xxx- |
| 798h | CRCXORH | XOR<15:8> | | | | | | | | xxxx xxxX | xxxx xxxX |
| 799h | CRCCON0 | EN | CRCGO | BUSY | ACCM | — | — | SHIFTM | FULL | 0000 --00 | 0000 -00 |
| 79Ah | CRCCON1 | DLEN<3:0> | | | | PLEN<3:0> | | | | 0000 0000 | 0000 0000 |
| 79Bh to 79Fh | — | Unimplemented | | | | | | | | — | — |

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

- Note** 1: PIC16F1615/9 only.
 2: Unimplemented, read as '1'.
 3: PIC16(L)F1615 only.
 4: PIC16(L)F1619 only.

TABLE 3-14: 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 16 | | | | | | | | | | | | |
| 80Ch | AT1RESL | RES<7:0> | | | | | | | | xxxx xxxx | xxxx xxxx | |
| 80Dh | AT1RESH | — | — | — | — | — | — | RES<9:8> | | ---- -xx | ---- -xx | |
| 80Eh | AT1MISSL | MISS<7:0> | | | | | | | | xxxx xxxx | xxxx xxxx | |
| 80Fh | AT1MISSH | MISS<15:8> | | | | | | | | xxxx xxxx | xxxx xxxx | |
| 810h | AT1PERL | PER<7:0> | | | | | | | | xxxx xxxx | xxxx xxxx | |
| 811h | AT1PERH | POV | PER<14:8> | | | | | | | | xxxx xxxx | xxxx xxxx |
| 812h | AT1PHSL | PHS<7:0> | | | | | | | | xxxx xxxx | xxxx xxxx | |
| 813h | AT1PHSH | — | — | — | — | — | — | PHS<9:8> | | ---- -xx | ---- -xx | |
| 814h | AT1CON0 | EN | PREC | PS<1:0> | | POL | — | APMOD | MODE | 0x00 --00 | 0x00 -00 | |
| 815h | AT1CON1 | — | PHP | — | PRP | — | MPP | ACCS | VALID | 0000 0000 | 0000 0000 | |
| 816h | AT1IR0 | — | — | — | — | — | PHSIF | MISSIF | PERIF | ----000 | ----000 | |
| 817h | AT1IE0 | — | — | — | — | — | PHSIE | MISSIE | PERIE | ----000 | ----000 | |
| 818h | AT1IR1 | — | — | — | — | — | CC3IF | CC2IF | CC1IF | ----000 | ----000 | |
| 819h | AT1IE1 | — | — | — | — | — | CC3IE | CC2IE | CC1IE | ----000 | ----000 | |
| 81Ah | AT1STPTL | STPT<7:0> | | | | | | | | xxxx xxxx | xxxx xxxx | |
| 81Bh | AT1STPTH | — | STPT<14:8> | | | | | | | | -xxx xxxx | -xxx xxxx |
| 81Ch | AT1ERRL | ERR<7:0> | | | | | | | | xxxx xxxx | xxxx xxxx | |
| 81Dh | AT1ERRH | ERR<15:8> | | | | | | | | xxxx xxxx | xxxx xxxx | |
| 81Eh | — | Unimplemented | | | | | | | | — | — | |
| 81Fh | — | Unimplemented | | | | | | | | — | — | |

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

- Note** 1: PIC16F1615/9 only.
 2: Unimplemented, read as '1'.
 3: PIC16(L)F1615 only.
 4: PIC16(L)F1619 only.

TABLE 3-14: 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 17 | | | | | | | | | | | |
| 88Ch | AT1CLK | — | — | — | — | — | — | — | CS0 | ---- -00 | ---- -00 |
| 88Dh | AT1SIG | — | — | — | — | — | SSEL<2:0> | | | ---- -000 | ---- -000 |
| 88Eh | AT1CSEL1 | — | — | — | — | — | CP1S<2:0> | | | ---- -000 | ---- -000 |
| 88Fh | AT1CC1L | CC1<7:0> | | | | | | | | 0000 0000 | 0000 0000 |
| 890h | AT1CC1H | — | — | — | — | — | CC1<9:8> | | | ---- -000 | ---- -000 |
| 891h | AT1CCON1 | CC1EN | — | — | CC1POL | CAP1P | — | — | CC1MODE | 0--0 0--0 | 0--0 0--0 |
| 892h | AT1CSEL2 | — | — | — | — | — | CP2S<2:0> | | | ---- -000 | ---- -000 |
| 893h | AT1CC2L | CC2<7:0> | | | | | | | | 0000 0000 | 0000 0000 |
| 894h | AT1CC2H | — | — | — | — | — | CC2<9:8> | | | ---- -000 | ---- -000 |
| 895h | AT1CCON2 | CC2EN | — | — | CC2POL | CAP2P | — | — | CC2MODE | 0--0 0--0 | 0--0 0--0 |
| 896h | AT1CSEL3 | — | — | — | — | — | CP3S<2:0> | | | ---- -000 | ---- -000 |
| 897h | AT1CC1L | CC3<7:0> | | | | | | | | 0000 0000 | 0000 0000 |
| 898h | AT1CC1H | — | — | — | — | — | CC3<9:8> | | | ---- -000 | ---- -000 |
| 899h | AT1CCON1 | CC3EN | — | — | CC3POL | CAP3P | — | — | CC3MODE | 0--0 0--0 | 0--0 0--0 |
| 89Ah to 89Fh | — | Unimplemented | | | | | | | | — | — |
| Bank 18-26 | | | | | | | | | | | |
| x0Ch/ x8Ch — x1Fh/ x9Fh | — | Unimplemented | | | | | | | | — | — |

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

- Note**
- 1: PIC16F1615/9 only.
 - 2: Unimplemented, read as '1'.
 - 3: PIC16(L)F1615 only.
 - 4: PIC16(L)F1619 only.

TABLE 3-14: 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 | |
|-----------------|----------|---------------|--------|-------|-----------|----------------|-----------|-------------|-----------|-------------------|---------------------------|-----------|
| Banks 27 | | | | | | | | | | | | |
| D80h to D8Bh | — | Unimplemented | | | | | | | | — | — | |
| D8Ch | SMT1TMRL | | | | | SMT1TMR<7:0> | | | | | 0000 0000 | 0000 0000 |
| D8Dh | SMT1TMRH | | | | | SMT1TMR<15:8> | | | | | 0000 0000 | 0000 0000 |
| D8Eh | SMT1TMRU | | | | | SMT1TMR<23:16> | | | | | 0000 0000 | 0000 0000 |
| D8Fh | SMT1CPRL | | | | | SMT1CPR<7:0> | | | | | xxxx xxxx | xxxx xxxx |
| D90h | SMT1CPRH | | | | | SMT1CPR<15:8> | | | | | xxxx xxxx | xxxx xxxx |
| D91h | SMT1CPRU | | | | | SMT1CPR<23:16> | | | | | xxxx xxxx | xxxx xxxx |
| D92h | SMT1CPWL | | | | | SMT1CPW<7:0> | | | | | xxxx xxxx | xxxx xxxx |
| D93h | SMT1CPWH | | | | | SMT1CPW<15:8> | | | | | xxxx xxxx | xxxx xxxx |
| D94h | SMT1CPWU | | | | | SMT1CPW<23:16> | | | | | xxxx xxxx | xxxx xxxx |
| D95h | SMT1PRL | | | | | SMT1PR<7:0> | | | | | xxxx xxxx | xxxx xxxx |
| D96h | SMT1PRH | | | | | SMT1PR<15:8> | | | | | xxxx xxxx | xxxx xxxx |
| D97h | SMT1PRU | | | | | SMT1PR<23:16> | | | | | xxxx xxxx | xxxx xxxx |
| D98h | SMT1CON0 | EN | — | STP | WPOL | SPOL | CPOL | SMT1PS<1:0> | | 0-00 0000 | 0-00 0000 | |
| D99h | SMT1CON1 | SMT1GO | REPEAT | — | — | MODE<3:0> | | | 00-- 0000 | | 00-- 0000 | |
| D9Ah | SMT1STAT | CPRUP | CPWUP | RST | — | — | TS | WS | AS | 000- -000 | 000- -000 | |
| D9Bh | SMT1CLK | — | — | — | — | — | CSEL<2:0> | | ---- -000 | | ---- -000 | |
| D9Ch | SMT1SIG | — | — | — | SSEL<4:0> | | | ---0 0000 | | ---0 0000 | | |
| D9Dh | SMT1WIN | — | — | — | WSEL<4:0> | | | ---0 0000 | | ---0 0000 | | |
| D9Eh | SMT2TMRL | | | | | SMT2TMR<7:0> | | | | | 0000 0000 | 0000 0000 |
| D9Fh | SMT2TMRH | | | | | SMT2TMR<15:8> | | | | | 0000 0000 | 0000 0000 |
| DA0h | SMT2TMRU | | | | | SMT2TMR<23:16> | | | | | 0000 0000 | 0000 0000 |
| DA1h | SMT2CPRL | | | | | SMT2CPR<7:0> | | | | | xxxx xxxx | xxxx xxxx |
| DA2h | SMT2CPRH | | | | | SMT2CPR<15:8> | | | | | xxxx xxxx | xxxx xxxx |
| DA3h | SMT2CPRU | | | | | SMT2CPR<23:16> | | | | | xxxx xxxx | xxxx xxxx |
| DA4h | SMT2CPWL | | | | | SMT2CPW<7:0> | | | | | xxxx xxxx | xxxx xxxx |

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

- Note** 1: PIC16F1615/9 only.
 2: Unimplemented, read as '1'.
 3: PIC16(L)F1615 only.
 4: PIC16(L)F1619 only.

TABLE 3-14: 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 27 (Continued) | | | | | | | | | | | |
| DA5h | SMT2CPWH | SMT2CPW<15:8> | | | | | | | | xxxx xxxx | xxxx xxxx |
| DA6h | SMT2CPWU | SMT2CPW<23:16> | | | | | | | | xxxx xxxx | xxxx xxxx |
| DA7h | SMT2PRL | SMT2PR<7:0> | | | | | | | | xxxx xxxx | xxxx xxxx |
| DA8h | SMT2PRH | SMT2PR<15:8> | | | | | | | | xxxx xxxx | xxxx xxxx |
| DA9h | SMT2PRU | SMT2PR<23:16> | | | | | | | | xxxx xxxx | xxxx xxxx |
| DAAh | SMT2CON0 | EN | — | STP | WPOL | SPOL | CPOL | SMT2PS<1:0> | | 0-00 0000 | 0-00 0000 |
| DABh | SMT2CON1 | SMT2GO | REPEAT | — | — | MODE<3:0> | | | 00-- 0000 | 00-- 0000 | |
| DACH | SMT2STAT | CPRUP | CPWUP | RST | — | — | TS | WS | AS | 000- -000 | 000- -000 |
| DADh | SMT2CLK | — | — | — | — | — | CSEL<2:0> | | | ---- -000 | ---- -000 |
| DAEh | SMT2SIG | — | — | — | SSEL<4:0> | | | ----0 0000 | ----0 0000 | | |
| DAFh | SMT2WIN | — | — | — | WSEL<4:0> | | | ----0 0000 | ----0 0000 | | |

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

- Note**
- 1: PIC16F1615/9 only.
 - 2: Unimplemented, read as '1'.
 - 3: PIC16(L)F1615 only.
 - 4: PIC16(L)F1619 only.

TABLE 3-14: 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 | |
|-----------------|--------------------------|---------------|-------|-------|----------------|-------|-------|-------|-----------|-------------------|---------------------------|------|
| Banks 28 | | | | | | | | | | | | |
| E0Ch to E0Eh | — | Unimplemented | | | | | | | | — | — | |
| E0Fh | PPSLOCK | — | — | — | — | — | — | — | PPSLOCKED | ---- -00 | ---- -00 | |
| E10h | INTPPS | — | — | — | INTPPS<4:0> | | | | --- | 0010 | --- | 0010 |
| E11h | T0CKIPPS | — | — | — | T0CKIPPS<4:0> | | | | --- | 0010 | --- | 0010 |
| E12h | T1CKIPPS | — | — | — | T1CKIPPS<4:0> | | | | --- | 0101 | --- | 0101 |
| E13h | T1GPPS | — | — | — | T1GPPS<4:0> | | | | --- | 0100 | --- | 0100 |
| E14h | CCP1PPS | — | — | — | CCP1PPS<4:0> | | | | --- | 0101 | --- | 0101 |
| E15h | CCP2PPS | — | — | — | CCP2PPS<4:0> | | | | --- | 0011 | --- | 0011 |
| E16h | ATINPPS | — | — | — | ATINPPS<4:0> | | | | --- | 0101 | --- | 0101 |
| E17h | CWGINPPS | — | — | — | CWGINPPS<4:0> | | | | --- | 0010 | --- | 0010 |
| E18h | T2PPS | — | — | — | T2PPS<4:0> | | | | --- | 0101 | --- | 0101 |
| E19h | T3CKIPPS | — | — | — | T3CKIPPS<4:0> | | | | --- | 0101 | --- | 0101 |
| E1Ah | T3GPPS | — | — | — | T3GPPS<4:0> | | | | --- | 0100 | --- | 0100 |
| E1Bh | T4PPS | — | — | — | T4PPS<4:0> | | | | --- | 0001 | --- | 0001 |
| E1Ch | T5CKIPPS | — | — | — | T5CKIPPS<4:0> | | | | --- | 0000 | --- | 0000 |
| E1Dh | T5GPPS | — | — | — | T5GPPS<4:0> | | | | --- | 0011 | --- | 0011 |
| E1Eh | T6PPS | — | — | — | T6PPS<4:0> | | | | --- | 0011 | --- | 0011 |
| E1Fh | ATCC1PPS | — | — | — | ATCC1PPS<4:0> | | | | --- | 0011 | --- | 0011 |
| E20h | SSPCLKPPS ⁽³⁾ | — | — | — | SSPCLKPPS<4:0> | | | | --- | 0000 | --- | 0000 |
| E20h | SSPCLKPPS ⁽⁴⁾ | — | — | — | SSPCLKPPS<4:0> | | | | --- | 0000 | --- | 1110 |
| E21h | SSPDATPPS ⁽³⁾ | — | — | — | SSPDATPPS<4:0> | | | | --- | 0001 | --- | 0001 |
| E21h | SSPDATPPS ⁽⁴⁾ | — | — | — | SSPDATPPS<4:0> | | | | --- | 0001 | --- | 1100 |
| E22h | SSPSSPPS ⁽³⁾ | — | — | — | SSPSSPPS<4:0> | | | | --- | 0011 | --- | 0011 |
| E22h | SSPSSPPS ⁽⁴⁾ | — | — | — | SSPSSPPS<4:0> | | | | --- | 0110 | --- | 0110 |
| E23h | ATCC2PPS | — | — | — | ATCC2PPS<4:0> | | | | --- | 0100 | --- | 0100 |

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

- Note** 1: PIC16F1615/9 only.
 2: Unimplemented, read as '1'.
 3: PIC16(L)F1615 only.
 4: PIC16(L)F1619 only.

TABLE 3-14: 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 |
|-----------------------------|----------------------|---------------|-------|-------|-------|-------|-----------------|-------|-------|-------------------|---------------------------|
| Banks 28 (Continued) | | | | | | | | | | | |
| E24h | RXPPS ⁽³⁾ | — | — | — | | | RXPPS<4:0> | | | ---1 0101 | ---1 0101 |
| E24h | RXPPS ⁽⁴⁾ | — | — | — | | | RXPPS<4:0> | | | ---0 1101 | ---0 1101 |
| E25h | CKPPS ⁽³⁾ | — | — | — | | | CKPPS<4:0> | | | ---1 0100 | ---1 0100 |
| E25h | CKPPS ⁽⁴⁾ | — | — | — | | | CKPPS<4:0> | | | ---0 1111 | ---0 1111 |
| E26h | SMT1SIGPPS | — | — | — | | | SMT1SIGPPS<4:0> | | | ---0 0100 | ---0 0100 |
| E27h | SMT1WINPPS | — | — | — | | | SMT1WINPPS<4:0> | | | ---0 0101 | ---0 0101 |
| E28h | CLCIN0PPS | — | — | — | | | CLCIN0PPS<4:0> | | | ---1 0011 | ---1 0011 |
| E29h | CLCIN1PPS | — | — | — | | | CLCIN1PPS<4:0> | | | ---1 0100 | ---1 0100 |
| E2Ah | CLCIN2PPS | — | — | — | | | CLCIN2PPS<4:0> | | | ---1 0001 | ---1 0001 |
| E2Bh | CLCIN3PPS | — | — | — | | | CLCIN3PPS<4:0> | | | ---0 0101 | ---0 0101 |
| E2Ch | SMT2SIGPPS | — | — | — | | | SMT2SIGPPS<4:0> | | | ---1 0001 | ---1 0001 |
| E2Dh | SMT2WINPPS | — | — | — | | | SMT2WINPPS<4:0> | | | ---0 0011 | ---0 0011 |
| E2Eh | ATCC3PPS | — | — | — | | | ATCC3PPS<4:0> | | | ---1 0101 | ---1 0101 |
| E2Fh to E6Fh | — | Unimplemented | | | | | | | | — | — |

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

- Note 1:** PIC16F1615/9 only.
Note 2: Unimplemented, read as '1'.
Note 3: PIC16(L)F1615 only.
Note 4: PIC16(L)F1619 only.

TABLE 3-14: 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 | |
|-----------------|-----------------------|---------------|-------|-------|-------|-------|-------|-------|-------|-------------------|---------------------------|-----------|
| Banks 29 | | | | | | | | | | | | |
| E8Ch to E8Fh | — | Unimplemented | | | | | | | | — | — | |
| E90h | RA0PPS | — | — | — | — | — | — | — | — | RA0PPS<4:0> | ---0 0000 | ---0 0000 |
| E91h | RA1PPS | — | — | — | — | — | — | — | — | RA1PPS<4:0> | ---0 0000 | ---0 0000 |
| E92h | RA2PPS | — | — | — | — | — | — | — | — | RA2PPS<4:0> | ---0 0000 | ---0 0000 |
| E93h | — | Unimplemented | | | | | | | | — | — | |
| E94h | RA4PPS | — | — | — | — | — | — | — | — | RA4PPS<4:0> | ---0 0000 | ---0 0000 |
| E95h | RA5PPS | — | — | — | — | — | — | — | — | RA5PPS<4:0> | ---0 0000 | ---0 0000 |
| E96h to E9Bh | — | Unimplemented | | | | | | | | — | — | |
| E9Ch | RB4PPS ⁽⁴⁾ | — | — | — | — | — | — | — | — | RB4PPS<4:0> | ---0 0000 | ---0 0000 |
| E9Dh | RB5PPS ⁽⁴⁾ | — | — | — | — | — | — | — | — | RB5PPS<4:0> | ---0 0000 | ---0 0000 |
| E9Eh | RB6PPS ⁽⁴⁾ | — | — | — | — | — | — | — | — | RB6PPS<4:0> | ---0 0000 | ---0 0000 |
| E9Fh | RB7PPS ⁽⁴⁾ | — | — | — | — | — | — | — | — | RB7PPS<4:0> | ---0 0000 | ---0 0000 |
| EA0h | RC0PPS | — | — | — | — | — | — | — | — | RC0PPS<4:0> | ---0 0000 | ---0 0000 |
| EA1h | RC1PPS | — | — | — | — | — | — | — | — | RC1PPS<4:0> | ---0 0000 | ---0 0000 |
| EA2h | RC2PPS | — | — | — | — | — | — | — | — | RC2PPS<4:0> | ---0 0000 | ---0 0000 |
| EA3h | RC3PPS | — | — | — | — | — | — | — | — | RC3PPS<4:0> | ---0 0000 | ---0 0000 |
| EA4h | RC4PPS | — | — | — | — | — | — | — | — | RC4PPS<4:0> | ---0 0000 | ---0 0000 |
| EA5h | RC5PPS | — | — | — | — | — | — | — | — | RC5PPS<4:0> | ---0 0000 | ---0 0000 |
| EA6h | RC6PPS ⁽⁴⁾ | — | — | — | — | — | — | — | — | RC6PPS<4:0> | ---0 0000 | ---0 0000 |
| EA7h | RC7PPS ⁽⁴⁾ | — | — | — | — | — | — | — | — | RC7PPS<4:0> | ---0 0000 | ---0 0000 |
| EA8h to EEfh | — | Unimplemented | | | | | | | | — | — | |

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

- Note** 1: PIC16F1615/9 only.
 2: Unimplemented, read as '1'.
 3: PIC16(L)F1615 only.
 4: PIC16(L)F1619 only.

TABLE 3-14: 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 |
|-----------------|----------|---------------|----------|-------------|----------|----------|--------------|----------|----------|-------------------|---------------------------|
| Banks 30 | | | | | | | | | | | |
| F0Ch to F0Eh | — | Unimplemented | | | | | | | | — | — |
| F0Fh | CLCDATA | — | — | — | — | MLC4OUT | MLC3OUT | MLC2OUT | MLC1OUT | ---- 0000 | ---- 0000 |
| F10h | CLC1CON | LC1EN | — | LC1OUT | LC1INTP | LC1INTN | LC1MODE<2:0> | | | 0-x0 0000 | 0-x0 0000 |
| F11h | CLC1POL | LC1POL | — | — | — | LC1G4POL | LC1G3POL | LC1G2POL | LC1G1POL | x--- xxxx | x--- xxxx |
| F12h | CLC1SEL0 | — | — | LC1D1S<5:0> | | | | | | --xx xxxx | --xx xxxx |
| F13h | CLC1SEL1 | — | — | LC1D2S<5:0> | | | | | | --xx xxxx | --xx xxxx |
| F14h | CLC1SEL2 | — | — | LC1D3S<5:0> | | | | | | --xx xxxx | --xx xxxx |
| F15h | CLC1SEL3 | — | — | LC1D4S<5:0> | | | | | | --xx xxxx | --xx xxxx |
| F16h | CLC1GLS0 | LC1G1D4T | LC1G1D4N | LC1G1D3T | LC1G1D3N | LC1G1D2T | LC1G1D2N | LC1G1D1T | LC1G1D1N | xxxx xxxx | xxxx xxxx |
| F17h | CLC1GLS1 | LC1G2D4T | LC1G2D4N | LC1G2D3T | LC1G2D3N | LC1G2D2T | LC1G2D2N | LC1G2D1T | LC1G2D1N | xxxx xxxx | xxxx xxxx |
| F18h | CLC1GLS2 | LC1G3D4T | LC1G3D4N | LC1G3D3T | LC1G3D3N | LC1G3D2T | LC1G3D2N | LC1G3D1T | LC1G3D1N | xxxx xxxx | xxxx xxxx |
| F19h | CLC1GLS3 | LC1G4D4T | LC1G4D4N | LC1G4D3T | LC1G4D3N | LC1G4D2T | LC1G4D2N | LC1G4D1T | LC1G4D1N | xxxx xxxx | xxxx xxxx |
| F1Ah | CLC2CON | LC2EN | — | LC2OUT | LC2INTP | LC2INTN | LC2MODE<2:0> | | | 0-x0 0000 | 0-x0 0000 |
| F1Bh | CLC2POL | LC2POL | — | — | — | LC2G4POL | LC2G3POL | LC2G2POL | LC2G1POL | x--- xxxx | x--- xxxx |
| F1Ch | CLC2SEL0 | — | — | LC2D1S<5:0> | | | | | | --xx xxxx | --xx xxxx |
| F1Dh | CLC2SEL1 | — | — | LC2D2S<5:0> | | | | | | --xx xxxx | --xx xxxx |
| F1Eh | CLC2SEL2 | — | — | LC2D3S<5:0> | | | | | | --xx xxxx | --xx xxxx |
| F1Fh | CLC2SEL3 | — | — | LC2D4S<5:0> | | | | | | --xx xxxx | --xx xxxx |
| F20h | CLC2GLS0 | LC2G1D4T | LC2G1D4N | LC2G1D3T | LC2G1D3N | LC2G1D2T | LC2G1D2N | LC2G1D1T | LC2G1D1N | xxxx xxxx | xxxx xxxx |
| F21h | CLC2GLS1 | LC2G2D4T | LC2G2D4N | LC2G2D3T | LC2G2D3N | LC2G2D2T | LC2G2D2N | LC2G2D1T | LC2G2D1N | xxxx xxxx | xxxx xxxx |
| F22h | CLC2GLS2 | LC2G3D4T | LC2G3D4N | LC2G3D3T | LC2G3D3N | LC2G3D2T | LC2G3D2N | LC2G3D1T | LC2G3D1N | xxxx xxxx | xxxx xxxx |
| F23h | CLC2GLS3 | LC2G4D4T | LC2G4D4N | LC2G4D3T | LC2G4D3N | LC2G4D2T | LC2G4D2N | LC2G4D1T | LC2G4D1N | xxxx xxxx | xxxx xxxx |

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

- Note**
- 1: PIC16F1615/9 only.
 - 2: Unimplemented, read as '1'.
 - 3: PIC16(L)F1615 only.
 - 4: PIC16(L)F1619 only.

TABLE 3-14: 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 |
|-----------------------------|----------|---------------|----------|-------------|----------|----------|--------------|----------|----------|-------------------|---------------------------|
| Banks 30 (Continued) | | | | | | | | | | | |
| F24h | CLC3CON | LC3EN | — | LC3OUT | LC3INTP | LC3INTN | LC3MODE<2:0> | | | 0-x0 0000 | 0-x0 0000 |
| F25h | CLC3POL | LC3POL | — | — | — | LC3G4POL | LC3G3POL | LC3G2POL | LC3G1POL | x--- xxxx | x--- xxxx |
| F26h | CLC3SEL0 | — | — | LC3D1S<5:0> | | | | | | --xx xxxx | --xx xxxx |
| F27h | CLC3SEL1 | — | — | LC3D2S<5:0> | | | | | | --xx xxxx | --xx xxxx |
| F28h | CLC3SEL2 | — | — | LC3D3S<5:0> | | | | | | --xx xxxx | --xx xxxx |
| F29h | CLC3SEL3 | — | — | LC3D4S<5:0> | | | | | | --xx xxxx | --xx xxxx |
| F2Ah | CLC3GLS0 | LC3G1D4T | LC3G1D4N | LC3G1D3T | LC3G1D3N | LC3G1D2T | LC3G1D2N | LC3G1D1T | LC3G1D1N | xxxx xxxx | xxxx xxxx |
| F2Bh | CLC3GLS1 | LC3G2D4T | LC3G2D4N | LC3G2D3T | LC3G2D3N | LC3G2D2T | LC3G2D2N | LC3G2D1T | LC3G2D1N | xxxx xxxx | xxxx xxxx |
| F2Ch | CLC3GLS2 | LC3G3D4T | LC3G3D4N | LC3G3D3T | LC3G3D3N | LC3G3D2T | LC3G3D2N | LC3G3D1T | LC3G3D1N | xxxx xxxx | xxxx xxxx |
| F2Dh | CLC3GLS3 | LC3G4D4T | LC3G4D4N | LC3G4D3T | LC3G4D3N | LC3G4D2T | LC3G4D2N | LC3G4D1T | LC3G4D1N | xxxx xxxx | xxxx xxxx |
| F2Eh | CLC4CON | LC4EN | — | LC4OUT | LC4INTP | LC4INTN | LC4MODE<2:0> | | | 0-x0 0000 | 0-x0 0000 |
| F2Fh | CLC4POL | LC4POL | — | — | — | LC4G4POL | LC4G3POL | LC4G2POL | LC4G1POL | x--- xxxx | x--- xxxx |
| F30h | CLC4SEL0 | — | — | LC4D1S<5:0> | | | | | | --xx xxxx | --xx xxxx |
| F31h | CLC4SEL1 | — | — | LC4D2S<5:0> | | | | | | --xx xxxx | --xx xxxx |
| F32h | CLC4SEL2 | — | — | LC4D3S<5:0> | | | | | | --xx xxxx | --xx xxxx |
| F33h | CLC4SEL3 | — | — | LC4D4S<5:0> | | | | | | --xx xxxx | --xx xxxx |
| F34h | CLC4GLS0 | LC4G1D4T | LC4G1D4N | LC4G1D3T | LC4G1D3N | LC4G1D2T | LC4G1D2N | LC4G1D1T | LC4G1D1N | xxxx xxxx | xxxx xxxx |
| F35h | CLC4GLS1 | LC4G2D4T | LC4G2D4N | LC4G2D3T | LC4G2D3N | LC4G2D2T | LC4G2D2N | LC4G2D1T | LC4G2D1N | xxxx xxxx | xxxx xxxx |
| F36h | CLC4GLS2 | LC4G3D4T | LC4G3D4N | LC4G3D3T | LC4G3D3N | LC4G3D2T | LC4G3D2N | LC4G3D1T | LC4G3D1N | xxxx xxxx | xxxx xxxx |
| F37h | CLC4GLS3 | LC4G4D4T | LC4G4D4N | LC4G4D3T | LC4G4D3N | LC4G4D2T | LC4G4D2N | LC4G4D1T | LC4G4D1N | xxxx xxxx | xxxx xxxx |
| F38h to F6Fh | — | Unimplemented | | | | | | | | — | — |

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

- Note**
- 1: PIC16F1615/9 only.
 - 2: Unimplemented, read as '1'.
 - 3: PIC16(L)F1615 only.
 - 4: PIC16(L)F1619 only.

PIC16(L)F1615/9

TABLE 3-14: SPECIAL FUNCTION REGISTER SUMMARY (CONTINUED)

| Addr | Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|-------------------|-----------------|--|--|-------|-----------------------------|-------|--------|---------|-------|
| Bank 31 | | | | | | | | | |
| F8Ch — FE3h | — | Unimplemented | | | | | | | |
| FE4h | STATUS_ SHAD | — | — | — | — | — | Z_SHAD | DC_SHAD | C_ |
| FE5h | WREG_ SHAD | Working Register Shadow | | | | | | | |
| FE6h | BSR_ SHAD | — | — | — | Bank Select Register Shadow | | | | |
| FE7h | PCLATH_ SHAD | — | Program Counter Latch High Register Shadow | | | | | | |
| FE8h | FSR0L_ SHAD | Indirect Data Memory Address 0 Low Pointer Shadow | | | | | | | |
| FE9h | FSR0H_ SHAD | Indirect Data Memory Address 0 High Pointer Shadow | | | | | | | |
| FEAh | FSR1L_ SHAD | Indirect Data Memory Address 1 Low Pointer Shadow | | | | | | | |
| FEBh | FSR1H_ SHAD | Indirect Data Memory Address 1 High Pointer Shadow | | | | | | | |
| FECh | — | Unimplemented | | | | | | | |
| FEDh | STKPTR | — | — | — | Current Stack Pointer | | | | |
| FEEh | TOSL | Top-of-Stack Low byte | | | | | | | |
| FEFh | TOSH | — | Top-of-Stack High byte | | | | | | |

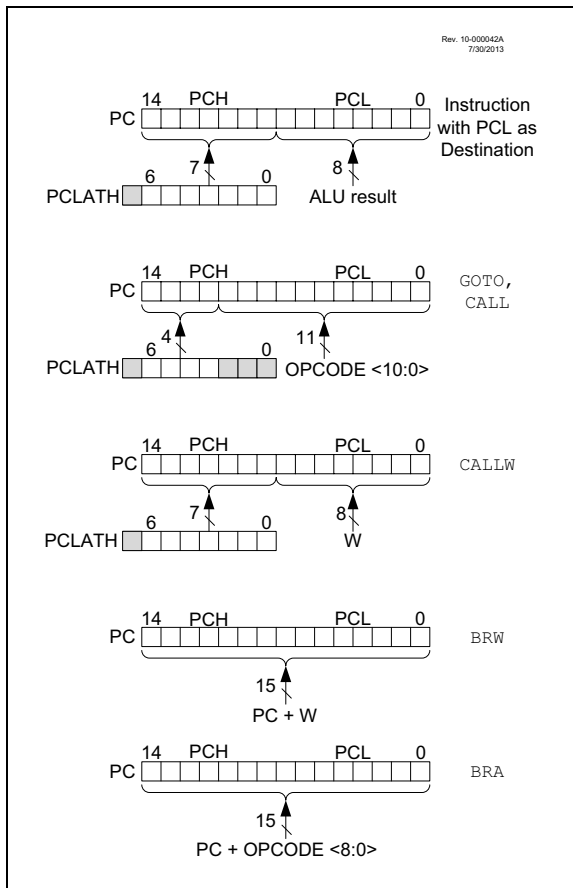
Legend: x = unknown, u = unchanged, q = value depends on condition, - = unimplemented, r = reserved. Shaded locations are unimplemented.

- Note**
- 1: PIC16F1615/9 only.
 - 2: Unimplemented, read as '1'.
 - 3: PIC16(L)F1615 only.
 - 4: PIC16(L)F1619 only.

3.4 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 3-3 shows the five situations for the loading of the PC.

FIGURE 3-3: LOADING OF PC IN DIFFERENT SITUATIONS



3.4.1 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 seven bits to the PCLATH register. When the lower eight 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.

3.4.2 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).

3.4.3 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.

3.4.4 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.

3.5 Stack

All devices have a 16-level x 15-bit wide hardware stack (refer to Figures 3-4 through 3-7). 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.

3.5.1 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 five 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 Figure 3-4 through Figure 3-7 for examples of accessing the stack.

FIGURE 3-4: ACCESSING THE STACK EXAMPLE 1

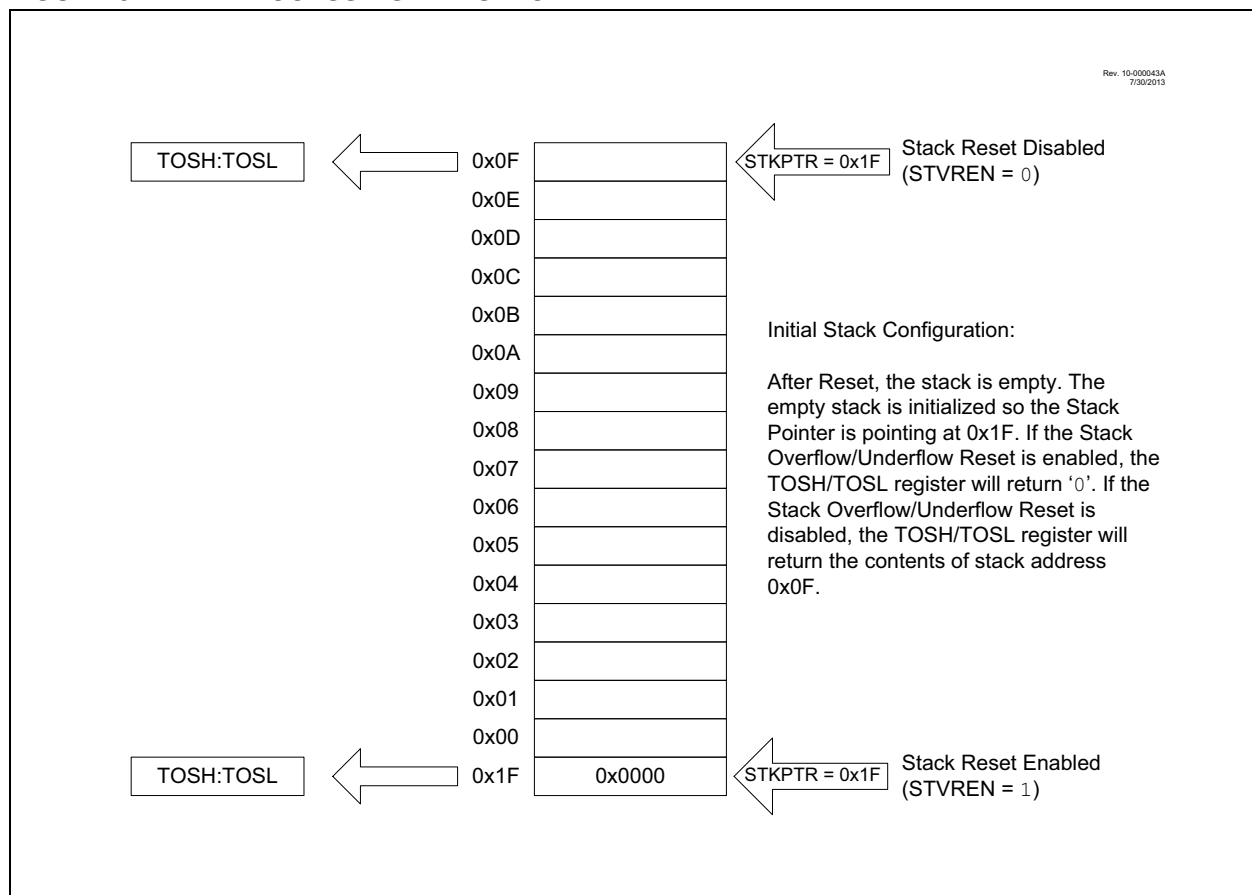


FIGURE 3-5: ACCESSING THE STACK EXAMPLE 2

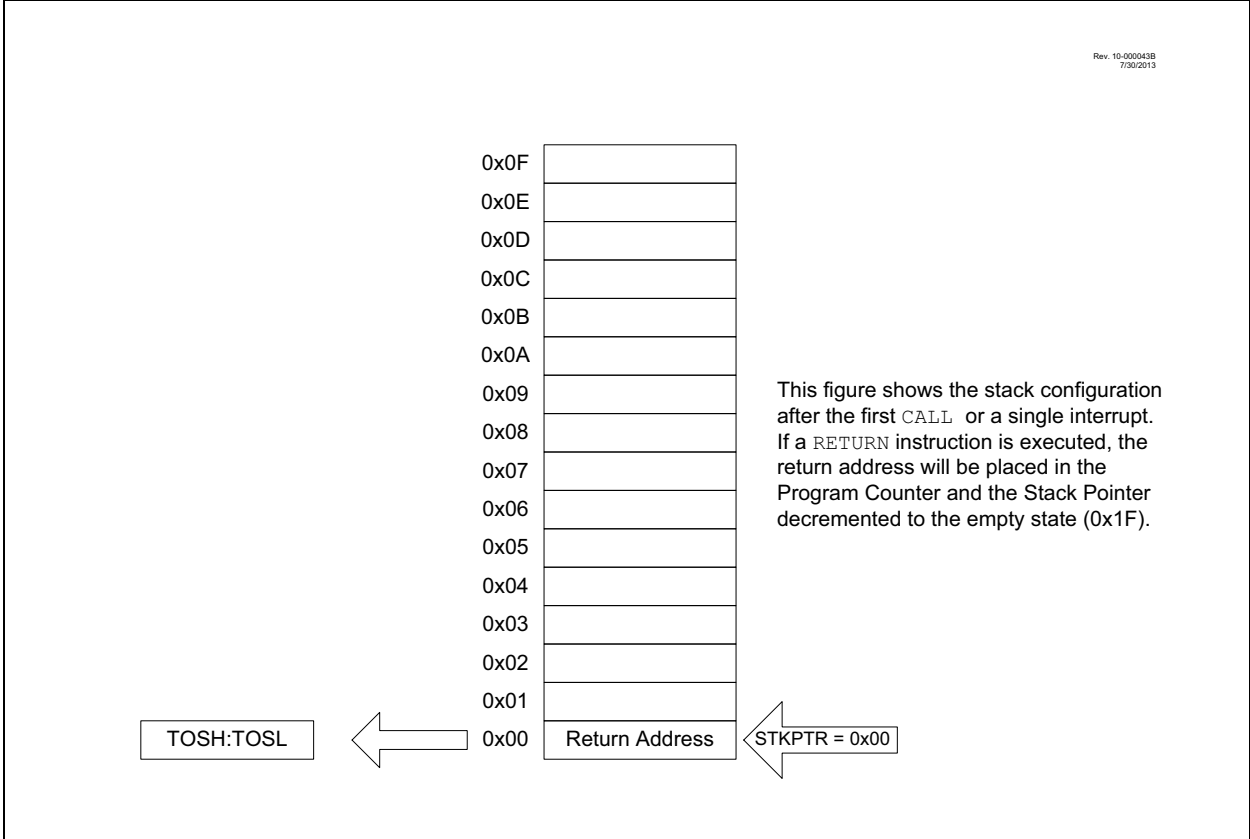


FIGURE 3-6: ACCESSING THE STACK EXAMPLE 3

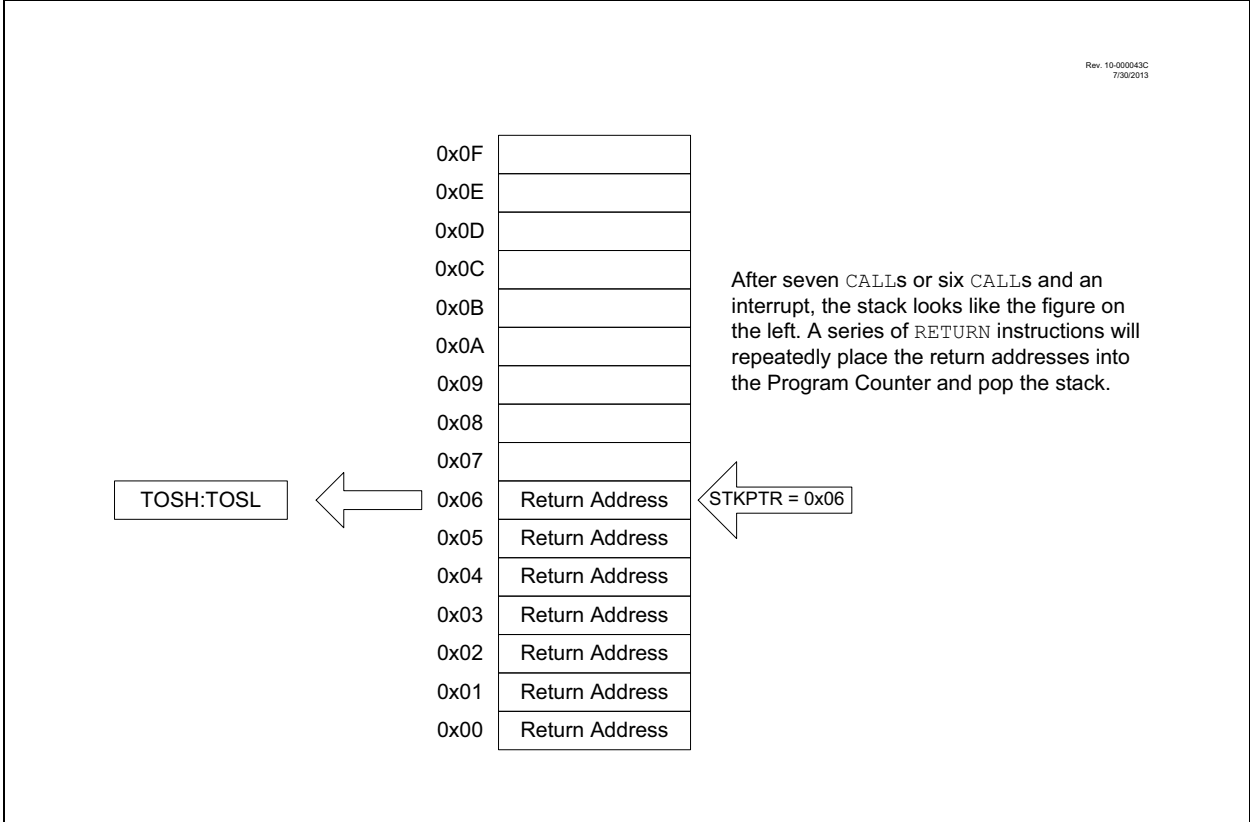
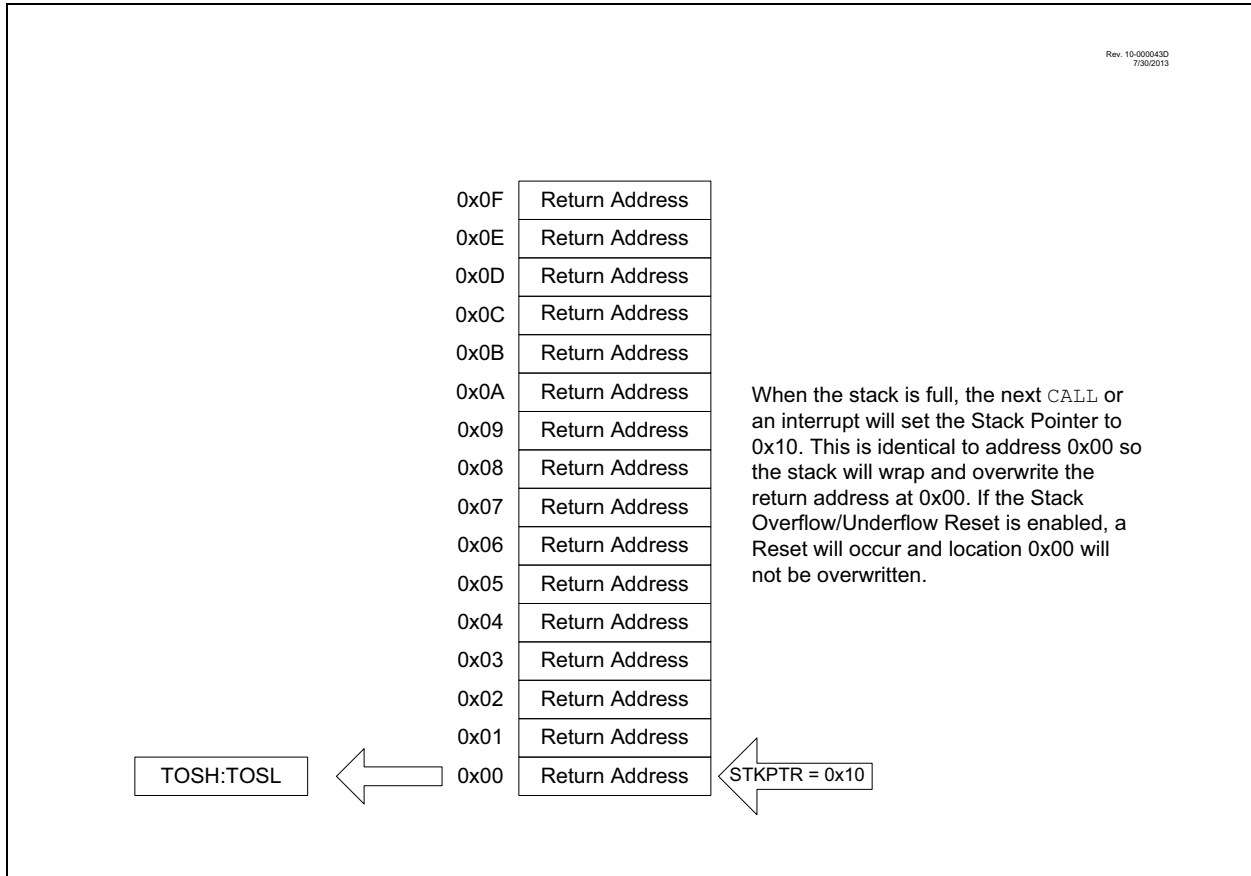


FIGURE 3-7: ACCESSING THE STACK EXAMPLE 4



3.5.2 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.

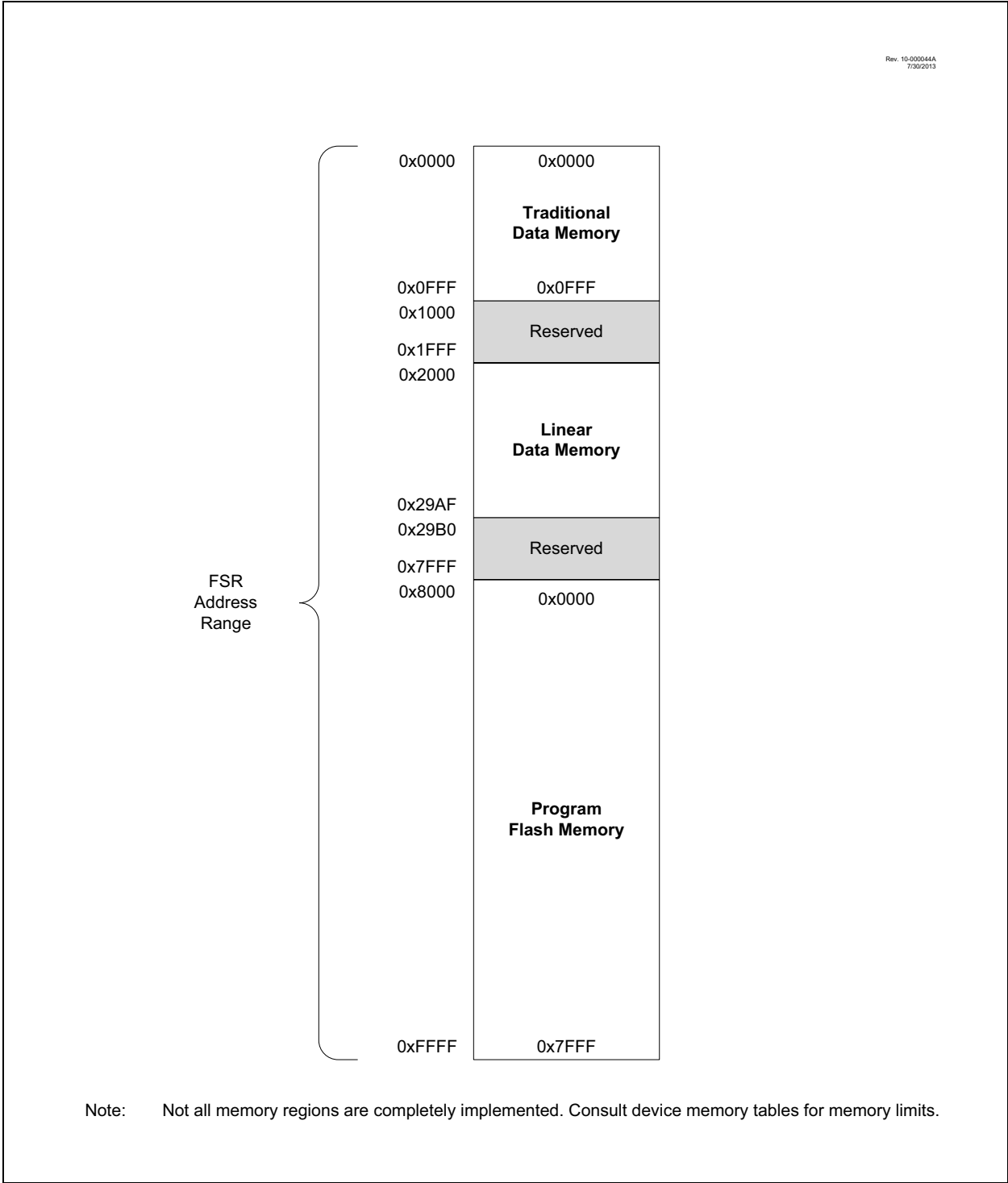
3.6 Indirect Addressing

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

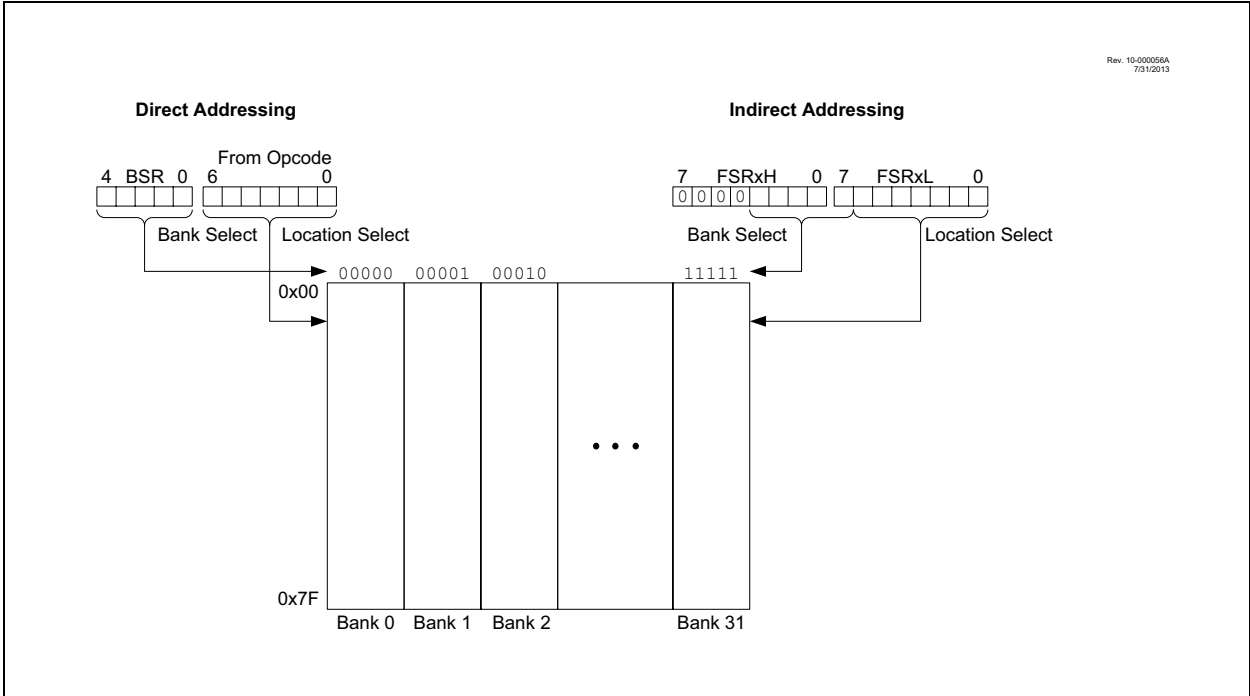
FIGURE 3-8: INDIRECT ADDRESSING



3.6.1 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 3-9: TRADITIONAL DATA MEMORY MAP



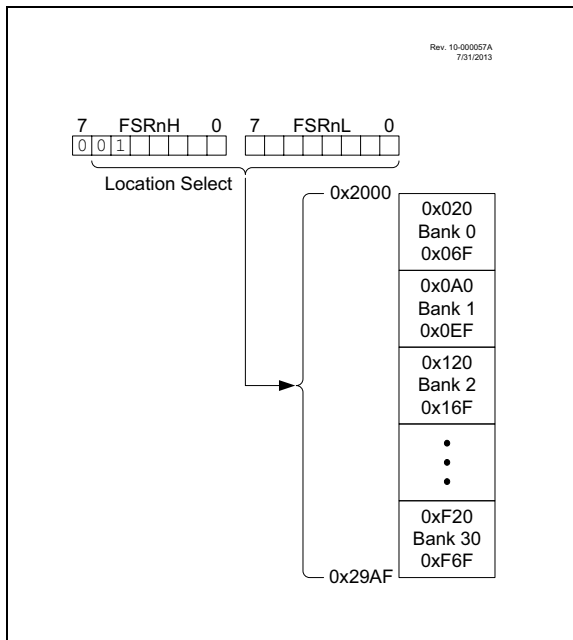
3.6.2 LINEAR DATA MEMORY

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.

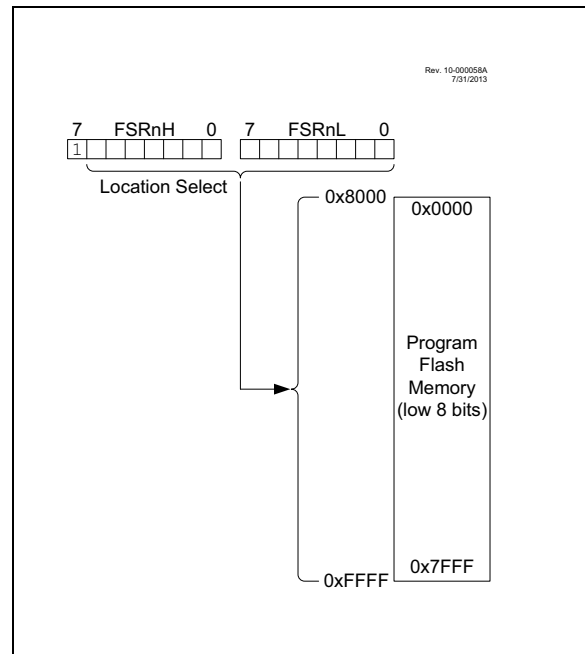
FIGURE 3-10: LINEAR DATA MEMORY MAP



3.6.3 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 eight 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 3-11: PROGRAM FLASH MEMORY MAP



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, Configuration Word 2 at 8008h, and Configuration 3 at 8009h.

| |
|--|
| <p>Note: The $\overline{\text{DEBUG}}$ bit in Configuration Words is managed automatically by device development tools including debuggers and programmers. For normal device operation, this bit should be maintained as a '1'.</p> |
|--|

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> ⁽¹⁾ | | — |
| bit 13 | | | | | bit 8 |

| | | | | | | | |
|-------------------|-------|-------|-----|-----|-----------|-------|-------|
| R/P-1 | R/P-1 | R/P-1 | U-1 | U-1 | R/P-1 | R/P-1 | R/P-1 |
| CP ⁽²⁾ | MCLRE | PWRTE | — | — | 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 Clock Monitor Enable bit

1 = Fail-Safe Clock Monitor is enabled

0 = Fail-Safe Clock Monitor is disabled

bit 12 **IESO:** Internal/External Switch Over bit

1 = Internal/External Switch Over mode is enabled

0 = Internal/External Switch Over mode is disabled

bit 11 **CLKOUTEN:** Clock Out Enable bit

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⁽¹⁾

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⁽²⁾

1 = Program memory code protection is disabled

0 = Program memory code protection is enabled

bit 6 **MCLRE:** 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 WPUA3 bit.

bit 5 **PWRTE:** Power-Up Timer Enable bit

1 = PWRT disabled

0 = PWRT enabled

bit 4-3 **Unimplemented:** Read as '1'

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

bit 2-0 **FOSC<2:0>**: Oscillator Selection bits

- 111 =ECH: External clock, High-Power mode: on CLKIN pin
- 110 =ECM: External clock, Medium-Power mode: on CLKIN pin
- 101 =ECL: External clock, Low-Power mode: on CLKIN pin
- 100 =INTOSC oscillator: I/O function on CLKIN pin
- 011 =Reserved
- 010 =HS: HS oscillator, high-speed crystal/resonator connected between OSC1 and OSC2 pins
- 001 =Reserved
- 000 =Reserved

- Note 1:** Enabling Brown-out Reset does not automatically enable Power-up Timer.
2: Once enabled, code-protect can only be disabled by bulk erasing the device.

REGISTER 4-2: CONFIG2: CONFIGURATION WORD 2

| | | | | | |
|--------------------|----------------------|-------|---------------------|--------|-------|
| R/P-1 | R/P-1 | R/P-1 | R/P-1 | R/P-1 | R/P-1 |
| LVP ⁽¹⁾ | DEBUG ⁽³⁾ | LPBOR | BORV ⁽²⁾ | STVREN | PLLEN |
| bit 13 | | | | | bit 8 |

| | | | | | | | |
|-------|-----|-----|-----|-----|---------|----------|-------|
| R/P-1 | U-1 | U-1 | U-1 | U-1 | R/P-1 | R/P-1 | R/P-1 |
| ZCD | — | — | — | — | PPS1WAY | 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⁾
 1 = Low-voltage programming enabled
 0 = High-voltage on MCLR must be used for programming
- bit 12 **DEBUG:** In-Circuit Debugger Mode bit⁽³⁾
 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 Enable bit
 1 = Low-Power Brown-out Reset is disabled
 0 = Low-Power Brown-out Reset 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 **PLLEN:** PLL Enable bit
 1 = 4xPLL enabled
 0 = 4xPLL disabled
- bit 7 **ZCD:** ZCD Disable bit
 1 = ZCD disabled. ZCD can be enabled by setting the ZCD1EN bit of ZCD1CON
 0 = ZCD always enabled
- bit 6-3 **Unimplemented:** Read as '1'
- bit 2 **PPS1WAY:** PPSLOCK Bit One-Way Set Enable bit
 1 = The PPSLOCK bit can only be set once after an unlocking sequence is executed; once PPSLOCK is set, all future changes to PPS registers are prevented
 0 = The PPSLOCK bit can be set and cleared as needed (provided an unlocking sequence is executed)
- bit 1-0 **WRT<1:0>:** Flash Memory Self-Write Protection bits
8 kW Flash memory (PIC16(L)F1615/9):
 11 = OFF - Write protection off
 10 = BOOT - 000h to 01FFh write-protected, 200h to 1FFFh may be modified by PMCON control
 01 = HALF - 000h to 0FFFh write-protected, 1000h to 1FFFh may be modified by PMCON control
 00 = ALL - 000h to 1FFFh write-protected, no addresses may be modified by PMCON control

- Note 1:** The LVP bit cannot be programmed to '0' when Programming mode is entered via LVP.
2: See VBOR parameter for specific trip point voltages.
3: The DEBUG bit in Configuration Words is managed automatically by device development tools including debuggers and programmers. For normal device operation, this bit should be maintained as a '1'.

REGISTER 4-3: CONFIG3: CONFIGURATION WORD 3

| | | | | | |
|-------------|-------|-------|-------------|-------|-------|
| R/P-0 | R/P-0 | R/P-1 | R/P-1 | R/P-1 | R/P-1 |
| WDTCCS<2:0> | | | WDTCWS<2:0> | | |
| bit 13 | | | bit 8 | | |

| | | | | | | | |
|-------|-----------|-------|-------------|-------|-------|-------|-------|
| U-1 | R/P-1 | R/P-1 | R/P-1 | R/P-1 | R/P-1 | R/P-1 | R/P-1 |
| — | WDTE<1:0> | | WDTCPS<4:0> | | | | |
| 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 13-11 **WDTCCS<2:0>**: WDT Configuration Clock Select bits
 111 =Software Control; WDT clock selected by CS<2:0>
 110 =Reserved
 .
 .
 .
 010 =Reserved
 001 =WDT reference clock is MFINTOSC, 31.25 kHz (default value)
 000 =WDT reference clock is LFINTOSC, 31.00 kHz output

bit 10-8 **WDTCWS<2:0>**: WDT Configuration Window Select bits.

| WDTCWS <2:0> | WINDOW at POR | | | Software control of WINDOW? | Keyed access required? |
|-----------------|---------------|---------------------------------|-----------------------------------|-----------------------------------|------------------------------|
| | Value | Window delay Percent of time | Window opening Percent of time | | |
| 111 | 111 | n/a | 100 | Yes | No |
| 110 | 111 | n/a | 100 | No | Yes |
| 101 | 101 | 25 | 75 | | |
| 100 | 100 | 37.5 | 62.5 | | |
| 011 | 011 | 50 | 50 | | |
| 010 | 010 | 62.5 | 37.5 | | |
| 001 | 001 | 75 | 25 | | |
| 000 | 000 | 87.5 | 12.5 ⁽¹⁾ | | |

Default fuse = 111

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

bit 6-5 **WDTE<1:0>**: Watchdog Timer Enable bits
 11 =WDT enabled in all modes, the SEN bit in the WDTCON0 register is ignored
 10 =WDT enabled while running and disabled in Sleep
 01 =WDT controlled by the SEN bit in the WDTCON0 register
 00 = WDT disabled

REGISTER 4-3: CONFIG3: CONFIGURATION WORD 3 (CONTINUED)

bit 4-0 **WDTCP3<4:0>**: WDT Configuration Period Select bits

| WDTCP3 <4:0> | WDTPS at POR | | | Software control of WDTPS | |
|-----------------------|-----------------------|---------------|---|---------------------------------|-----|
| | Value | Divider Ratio | Typical time out (F _{IN} = 31 kHz) | | |
| 11111 | 01011 | 1:65536 | 2 ¹⁶ | 2 s | Yes |
| 10011 ... 11110 | 10011 ... 11110 | 1:32 | 2 ⁵ | 1 ms | |
| 10010 | 10010 | 1:8388608 | 2 ²³ | 256 s | No |
| 10001 | 10001 | 1:4194304 | 2 ²² | 128 s | |
| 10000 | 10000 | 1:2097152 | 2 ²¹ | 64 s | |
| 01111 | 01111 | 1:1048576 | 2 ²⁰ | 32 s | |
| 01110 | 01110 | 1:524299 | 2 ¹⁹ | 16 s | |
| 01101 | 01101 | 1:262144 | 2 ¹⁸ | 8 s | |
| 01100 | 01100 | 1:131072 | 2 ¹⁷ | 4 s | |
| 01011 | 01011 | 1:65536 | 2 ¹⁶ | 2 s | |
| 01010 | 01010 | 1:32768 | 2 ¹⁵ | 1 s | |
| 01001 | 01001 | 1:16384 | 2 ¹⁴ | 512 ms | |
| 01000 | 01000 | 1:8192 | 2 ¹³ | 256 ms | |
| 00111 | 00111 | 1:4096 | 2 ¹² | 128 ms | |
| 00110 | 00110 | 1:2048 | 2 ¹¹ | 64 ms | |
| 00101 | 00101 | 1:1024 | 2 ¹⁰ | 32 ms | |
| 00100 | 00100 | 1:512 | 2 ⁹ | 16 ms | |
| 00011 | 00011 | 1:256 | 2 ⁸ | 8 ms | |
| 00010 | 00010 | 1:128 | 2 ⁷ | 4 ms | |
| 00001 | 00001 | 1:64 | 2 ⁶ | 2 ms | |
| 00000 | 00000 | 1:32 | 2 ⁵ | 1 ms | |

Default
fuse = 11111

Note 1: A window delay of 12.5% is only available in Software Control mode via the WDTCON1 register.

4.3 Code Protection

Code protection allows the device to be protected from unauthorized access. 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 \overline{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 boot loader 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 10.4 "User ID, Device ID and Configuration Word Access"](#) for more information on accessing these memory locations. For more information on checksum calculation, see the "*PIC12(L)F1612/16(L)F161X Memory Programming Specification*" (DS40001720).

4.6 Device ID and Revision ID

The 14-bit Device ID word is located at 8006h and the 14-bit Revision ID is located at 8005h. These locations are read-only and cannot be erased or modified. See [Section 10.4 “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.

4.7 Register Definitions: Device ID

REGISTER 4-4: DEVID: DEVICE ID REGISTER

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

| | | | | | | | |
|----------|---|---|---|-------|---|---|---|
| R | R | R | R | R | R | R | R |
| DEV<7:0> | | | | | | | |
| bit 7 | | | | bit 0 | | | |

Legend:

R = Readable bit

'1' = Bit is set

'0' = Bit is cleared

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

| Device | DEVID<13:0> Values |
|-------------|---------------------------|
| PIC16F1615 | 11 0000 0111 1100 (307Ch) |
| PIC16LF1615 | 11 0000 0111 1110 (307Eh) |
| PIC16F1619 | 11 0000 0111 1101 (307Dh) |
| PIC16LF1619 | 11 0000 0111 1111 (307Fh) |

REGISTER 4-5: REVID: REVISION ID REGISTER

| | | | | | | | |
|-----------|---|---|---|-------|---|---|---|
| R | R | R | R | R | R | R | R |
| REV<13:8> | | | | | | | |
| bit 13 | | | | bit 8 | | | |

| | | | | | | | |
|----------|---|---|---|-------|---|---|---|
| R | R | R | R | R | R | R | R |
| REV<7:0> | | | | | | | |
| bit 7 | | | | bit 0 | | | |

Legend:

R = Readable bit

'1' = Bit is set

'0' = Bit is cleared

bit 13-0 **REV<13:0>**: Revision ID bits

5.0 OSCILLATOR MODULE

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. In addition, the system clock source can be supplied from one of two internal oscillators and PLL circuits, 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 (HS, ECH, ECM, or ECL modes) and switch automatically to the internal oscillator.
- Oscillator Start-up Timer (OST) ensures stability of crystal oscillator sources

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

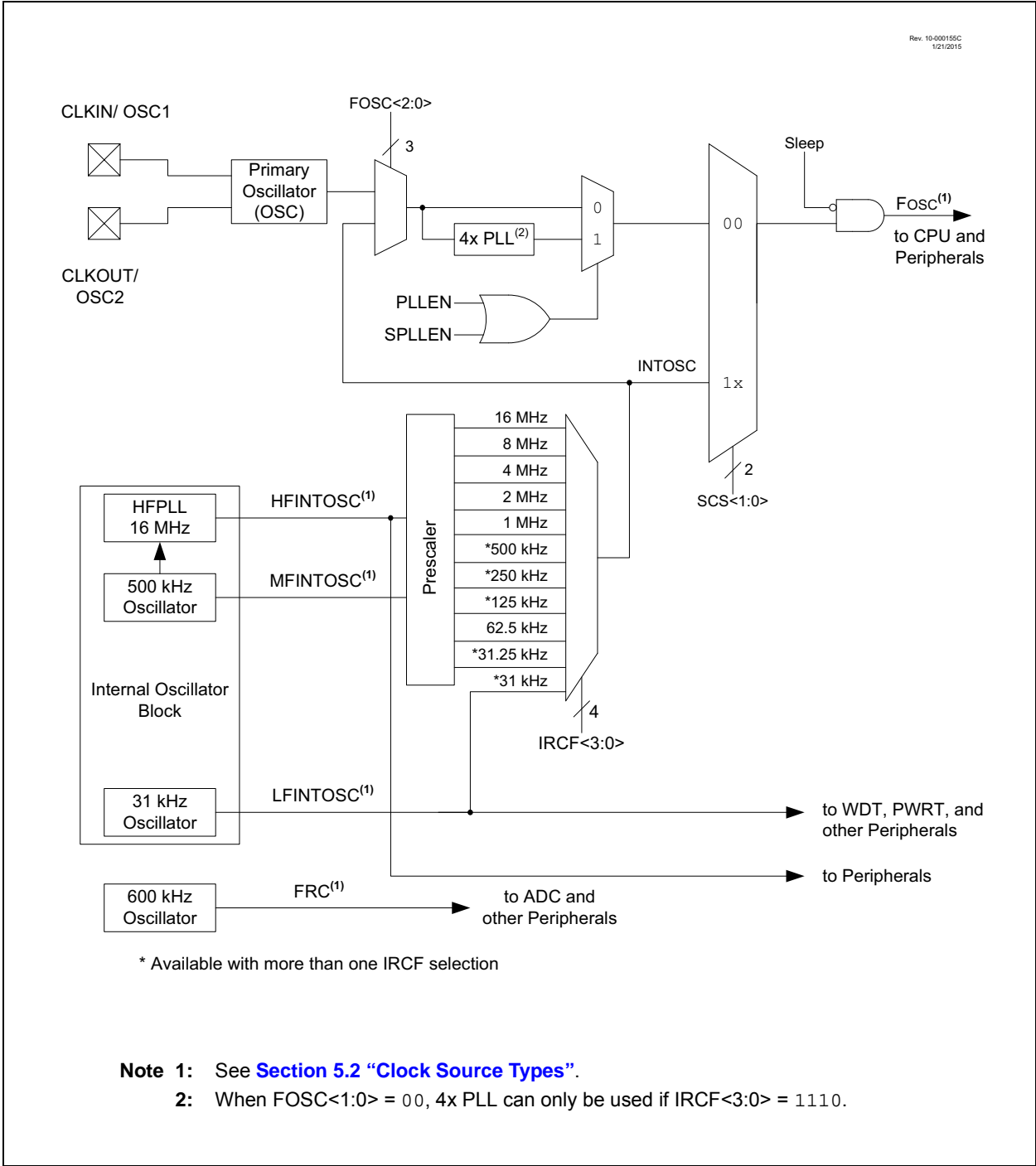
1. ECL – External Clock Low-Power mode (0 MHz to 0.5 MHz)
2. ECM – External Clock Medium-Power mode (0.5 MHz to 4 MHz)
3. ECH – External Clock High-Power mode (4 MHz to 32 MHz)
4. INTOSC – Internal oscillator (31 kHz to 32 MHz).
5. HS – High Gain Crystal or Ceramic Resonator mode (4 MHz to 10 MHz)

Clock Source modes are selected by the FOSC<1:0> bits in the Configuration Words. The FOSC bits determine the type of oscillator that will be used when the device is first powered.

The ECH, ECM, and ECL Clock modes rely on an external logic level signal as the device clock source. The HS Clock mode requires an external crystal or resonator to be connected to the device.

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

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



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 (ECH, ECM, ECL modes) and quartz crystal resonators or ceramic resonators (HS mode).

Internal clock sources are contained within the oscillator module. The internal oscillator block has two internal oscillators and a dedicated Phase Lock Loop (HFPLL) that are used to generate three internal system clock sources: the 16 MHz High-Frequency Internal Oscillator (HFINTOSC), 500 kHz (MFINTOSC) 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<1:0> 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:
 - 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

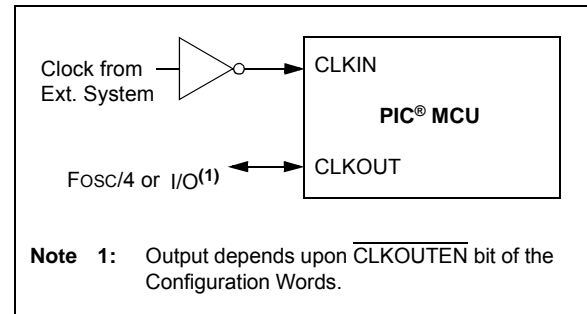
The External Clock (EC) mode allows an externally generated logic level signal to be the system clock source. When operating in this mode, an external clock source is connected to the CLKIN input. CLKOUT is available for general purpose I/O or CLKOUT. [Figure 5-2](#) shows the pin connections for EC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

EC mode has three power modes to select from through the Fosc bits in the Configuration Words:

- ECH – High power, 4-20 MHz
- ECM – Medium power, 0.5-4 MHz
- ECL – Low power, 0-0.5 MHz

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 limiting 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 HS Mode

The HS mode supports the use of quartz crystal resonators or ceramic resonators connected to OSC1 and OSC2. [Figure 5-3](#) and [Figure 5-4](#) show typical circuits for quartz crystal and ceramic resonators, respectively.

FIGURE 5-3: QUARTZ CRYSTAL OPERATION (HS MODE)

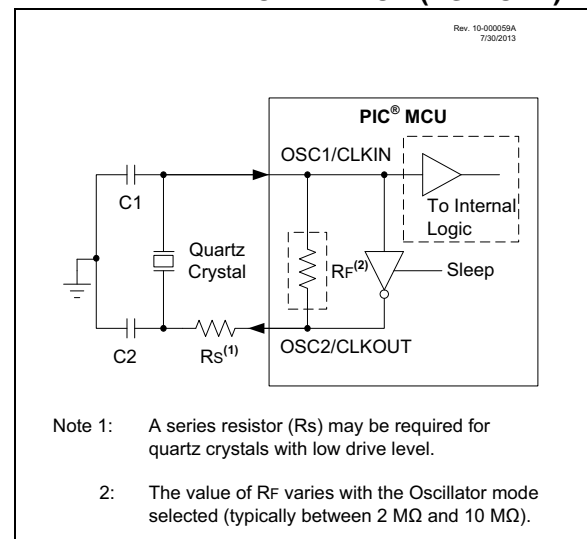
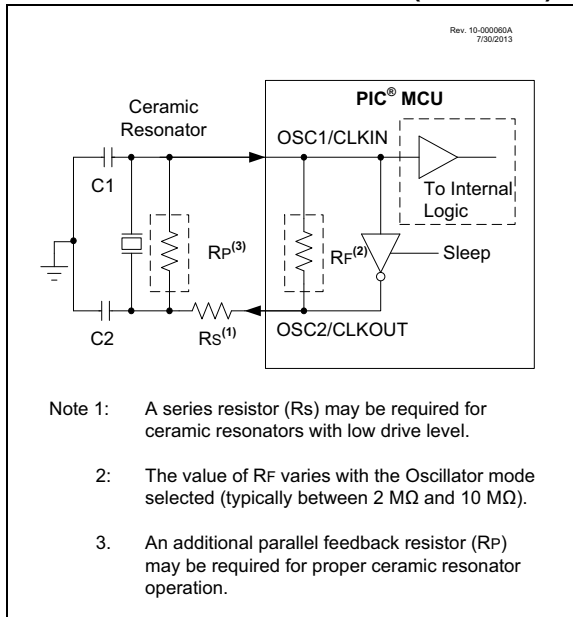


FIGURE 5-4: CERAMIC RESONATOR OPERATION (HS MODE)



5.2.1.3 Oscillator Start-up Timer (OST)

If the oscillator module is configured for 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. 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”](#)).

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<1:0> 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, CLKIN is available for general purpose I/O. 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 and a dedicated Phase Lock Loop, HFPLL that can produce one of three internal system clock sources.

1. The **HFINTOSC** (High-Frequency Internal Oscillator) is factory calibrated and operates at 16 MHz. The HFINTOSC source is generated from the 500 kHz MFINTOSC source and the dedicated Phase Lock Loop, HFPLL. The frequency of the HFINTOSC can be user-adjusted via software using the OSCTUNE register ([Register 5-3](#)).
2. The **MFINTOSC** (Medium-Frequency Internal Oscillator) is factory calibrated and operates at 500 kHz. The frequency of the MFINTOSC can be user-adjusted via software using the OSCTUNE register ([Register 5-3](#)).
3. 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 frequency of the HFINTOSC can be altered via software using the OSCTUNE register ([Register 5-3](#)).

The output of the HFINTOSC connects to a postscaler and multiplexer (see [Figure 5-1](#)). One of multiple frequencies derived from the HFINTOSC can be selected via software using the IRCF<3:0> bits of the OSCCON register. See [Section 5.2.2.8 “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<1:0> = 00, 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 Status Locked bit (HFIOFL) of the OSCSTAT register indicates when the HFINTOSC is running within 2% of its final value.

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 MFINTOSC

The Medium-Frequency Internal Oscillator (MFINTOSC) is a factory calibrated 500 kHz internal clock source. The frequency of the MFINTOSC can be altered via software using the OSCTUNE register ([Register 5-3](#)).

The output of the MFINTOSC connects to a postscaler and multiplexer (see [Figure 5-1](#)). One of nine frequencies derived from the MFINTOSC can be selected via software using the IRCF<3:0> bits of the OSCCON register. See [Section 5.2.2.8 “Internal Oscillator Clock Switch Timing”](#) for more information.

The MFINTOSC is enabled by:

- Configure the IRCF<3:0> bits of the OSCCON register for the desired HF frequency, and
- FOSC<1:0> = 00, or
- Set the System Clock Source (SCS) bits of the OSCCON register to ‘1x’.

The Medium-Frequency Internal Oscillator Ready bit (MFIOFR) of the OSCSTAT register indicates when the MFINTOSC is running.

5.2.2.3 Internal Oscillator Frequency Adjustment

The 500 kHz internal oscillator is factory calibrated. This internal oscillator can be adjusted in software by writing to the OSCTUNE register (Register 5-3). Since the HFINTOSC and MFINTOSC clock sources are derived from the 500 kHz internal oscillator a change in the OSCTUNE register value will apply to both.

The default value of the OSCTUNE register is '0'. The value is a 6-bit two's complement number. A value of 1Fh will provide an adjustment to the maximum frequency. A value of 20h will provide an adjustment to the minimum frequency.

When the OSCTUNE register is modified, the oscillator frequency will begin shifting to the new frequency. Code execution continues during this shift. There is no indication that the shift has occurred.

OSCTUNE does not affect the LFINTOSC frequency. Operation of features that depend on the LFINTOSC clock source frequency, such as the Power-up Timer (PWRT), Watchdog Timer (WDT), and peripherals, are *not* affected by the change in frequency.

5.2.2.4 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 5.2.2.8 "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<1:0> = 00, 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.

5.2.2.5 FRC

The FRC clock is an uncalibrated, nominal 600 kHz peripheral clock source.

The FRC is automatically turned on by the peripherals requesting the FRC clock.

The FRC clock will continue to run during Sleep.

5.2.2.6 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 postscaler outputs of the 16 MHz HFINTOSC, 500 kHz MFINTOSC, and 31 kHz LFINTOSC output connect to a 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.

5.2.2.7 32 MHz Internal Oscillator Frequency Selection

The Internal Oscillator Block can be used with the 4x PLL associated with the External Oscillator Block to produce a 32 MHz internal system clock source. Either the 8 or 16 MHz internal oscillator settings can be used, with the 16 MHz being divided by two before being input into the PLL. The following settings are required to use the 32 MHz internal clock source:

- The FOSC bits in Configuration Words must be set to use the INTOSC source as the device system clock (FOSC<1:0> = 00).
- The SCS bits in the OSCCON register must be cleared to use the clock determined by FOSC<1:0> in Configuration Words (SCS<1:0> = 00).
- The IRCF bits in the OSCCON register must be set to either the 16 MHz (IRCF<3:0> = 1111) or the 8 MHz HFINTOSC (IRCF<3:0> = 1110).
- The SPLLEN bit in the OSCCON register must be set to enable the 4x PLL, or the PLEN bit of the Configuration Words must be programmed to a '1'.

Note: When using the PLEN bit of the Configuration Words, the 4x PLL cannot be disabled by software and the 8/16 MHz HFINTOSC option will no longer be available.

The 4x PLL is not available for use with the internal oscillator when the SCS bits of the OSCCON register are set to '1x'. The SCS bits must be set to '00' to use the 4x PLL with the internal oscillator.

5.2.2.8 Internal Oscillator Clock Switch Timing

When switching between the HFINTOSC, MFINTOSC and the LFINTOSC, the new oscillator may already be shut down to save power (see [Figure 5-3](#)). 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, MFINTOSC and LFINTOSC oscillators. The sequence of a frequency selection is as follows:

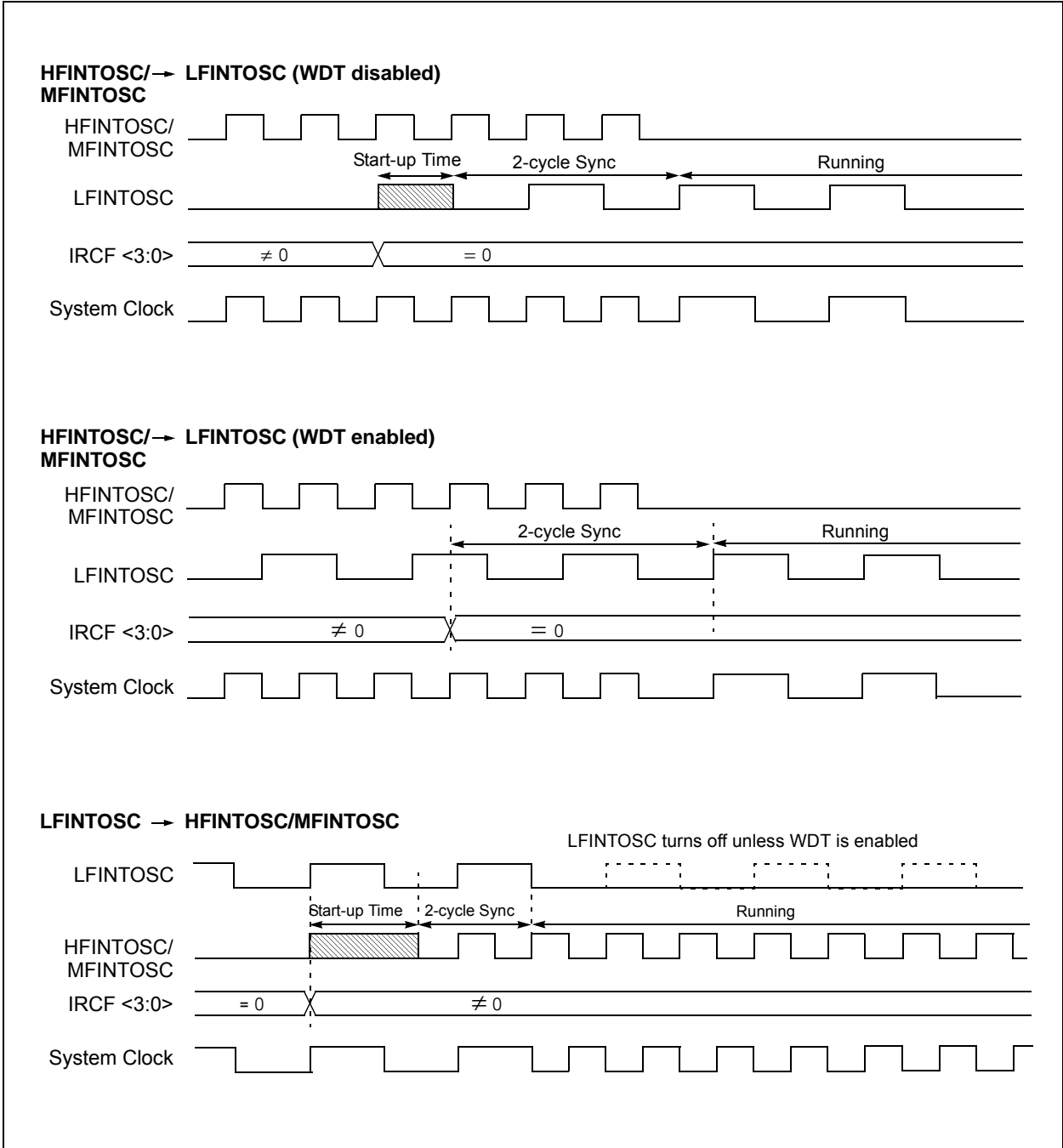
1. IRCF<3:0> bits of the OSCCON register are modified.
2. If the new clock is shut down, a clock start-up delay is started.
3. Clock switch circuitry waits for a falling edge of the current clock.
4. The current clock is held low and the clock switch circuitry waits for a rising edge in the new clock.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

See [Figure 5-3](#) 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 35.0 "Electrical Specifications"](#).

FIGURE 5-5: INTERNAL OSCILLATOR SWITCH TIMING



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
- 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<1:0> bits in the Configuration Words.
- 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.

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 CLOCK SWITCHING BEFORE SLEEP

When clock switching from an old clock to a new clock is requested just 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 32 MHz operation to the selected internal clock is complete.

TABLE 5-1: OSCILLATOR SWITCHING DELAYS

| Switch From | Switch To | Frequency | Oscillator Delay |
|------------------|---|---|---|
| Sleep | LFINTOSC ⁽¹⁾ MFINTOSC ⁽¹⁾ HFINTOSC ⁽¹⁾ | 31 kHz 31.25 kHz-500 kHz 31.25 kHz-16 MHz | Oscillator Warm-up Delay (TWARM) ⁽²⁾ |
| Sleep/POR | EC ⁽¹⁾ | DC – 32 MHz | 2 cycles |
| LFINTOSC | EC ⁽¹⁾ | DC – 32 MHz | 1 cycle of each |
| Any clock source | MFINTOSC ⁽¹⁾ HFINTOSC ⁽¹⁾ | 31.25 kHz-500 kHz 31.25 kHz-16 MHz | 2 μs (approx.) |
| Any clock source | LFINTOSC ⁽¹⁾ | 31 kHz | 1 cycle of each |
| PLL inactive | PLL active | 16-32 MHz | 2 ms (approx.) |

Note 1: PLL inactive.

2: See [Section 35.0, Electrical Specifications](#).

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 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 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 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.

Note: When FSCM is enabled, Two-Speed Start-up will automatically be enabled.

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 (HS mode).
7. System clock is switched to external clock source.

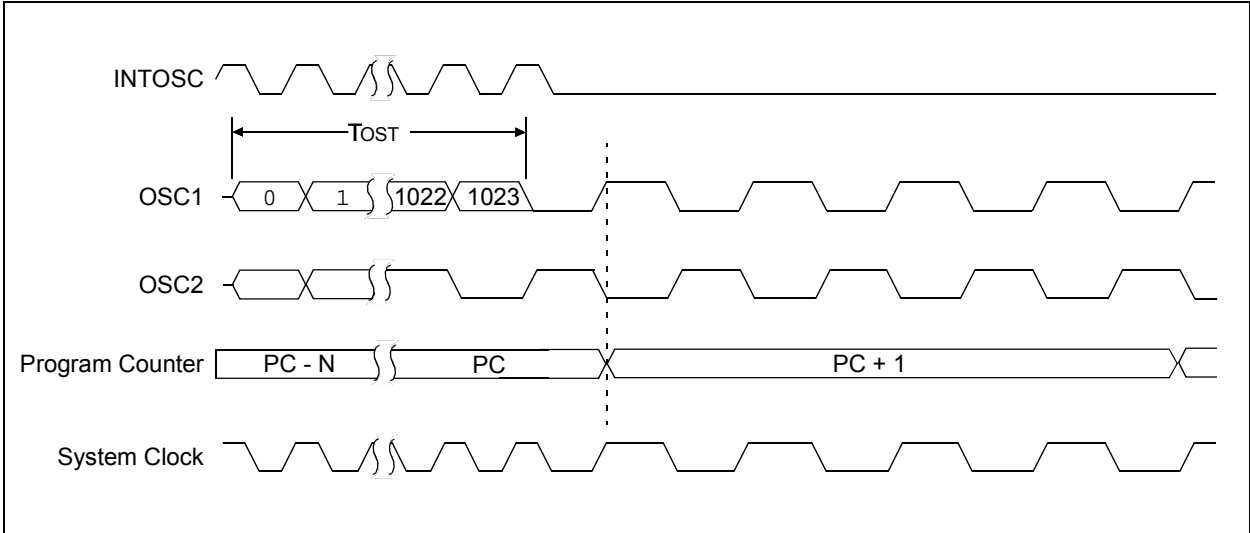
5.4.3 CHECKING TWO-SPEED CLOCK STATUS

Checking the state of the OSTS bit of the OSCSTAT register will confirm if the CPU is running from the external clock source, as defined by the FOSC<2:0> bits in the Configuration Words, or the internal oscillator. See [Table 5-2](#).

TABLE 5-2: OSCILLATOR SWITCHING DELAYS

| Switch From | Switch To | Oscillator Delay |
|------------------|----------------------|----------------------------------|
| Any clock source | LFINTOSC | 1 cycle of each clock source |
| | HFINTOSC | 2 μ s (approx.) |
| | ECH, ECM, ECL, EXTRC | 2 cycles |
| | HS | 1024 Clock Cycles (OST) |
| | Secondary Oscillator | 1024 Secondary Oscillator Cycles |

FIGURE 5-6: TWO-SPEED START-UP

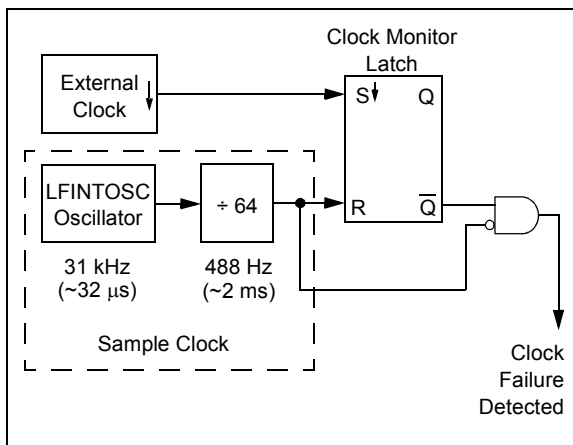


5.5 Fail-Safe Clock Monitor

The Fail-Safe Clock Monitor (FSCM) allows the device to continue operating should the external oscillator or external clock fail. If an oscillator mode is selected, the FSCM can detect oscillator failure any time after the Oscillator Start-up Timer (OST) has expired. When an external clock mode is selected, the FSCM can detect failure as soon as the device is released from Reset.

FSCM is enabled by setting the FCMEN bit in the Configuration Words. The FSCM is applicable to external oscillator modes (HS) and external clock modes (ECH, ECM, ECL, EXTRC) and the Secondary Oscillator (SOSC).

FIGURE 5-7: FSCM BLOCK DIAGRAM



5.5.1 FAIL-SAFE DETECTION

The FSCM module detects a failed oscillator by monitoring falling clock edges and using LFINTOSC as a time base. See Figure 5-7. Detection of a failed oscillator will take 32 to 96 cycles of the LFINTOSC. Figure 5-10 shows a timing diagram of the FSCM module.

5.5.2 FAIL-SAFE OPERATION

When the external clock fails, the FSCM switches the CPU clock to an internal clock source and sets the OSFIF bit of the PIR2 register. The internal clock source is determined by the IRCF<3:0> bits in the OSCCON register.

When the OSFIF bit is set, an interrupt will be generated, if the OSFIE bit in the PIE2 register is enabled. The user's firmware in the Interrupt Service Routine (ISR) can then take steps to mitigate the problems that may arise from the failed clock.

The system clock will continue to be sourced from the internal clock source until the Fail-Safe condition has been cleared, see Section 5.5.3 "Fail-Safe Condition Clearing".

5.5.3 FAIL-SAFE CONDITION CLEARING

When a Fail-Safe condition exists, the user must take the following actions to clear the condition before returning to normal operation with the external source.

The next sections describe how to clear the Fail-Safe condition for specific clock selections (FOSC bits) and clock switching modes (SCS bit settings).

5.5.3.1 External Oscillator with SCS<1:0> = 00

When a Fail-Safe condition occurs with the FOSC bits selecting external oscillator (FOSC<2:0> = HS) and the clock switch has been selected to run from the FOSC selection (SCS<1:0> = 00), the condition is cleared by performing the following procedure.

When SCS<1:0> = 00 (Running from FOSC selection)

SCS<1:0> = 1x:

Change the SCS bits in the OSCCON register to select the internal oscillator block. This resets the OST timer and allows it to operate again.

OSFIF = 0:

Clear the OSFIF bit in the PIR2 register.

SCS<1:0> = 00:

Change the SCS bits in the OSCCON register to select the FOSC Configuration Word clock selection. This will start the OST. The CPU will continue to operate from the internal oscillator until the OST expires, the clock module will switch to the external oscillator and the Fail-Safe condition will be cleared.

If the Fail-Safe condition still exists, the OSFIF bit will again be set by hardware.

5.5.3.2 External Clock with SCS<1:0> = 00

When a Fail-Safe condition occurs with the FOSC bits selecting external clock (FOSC<2:0> = ECH, ECM, ECL, EXTRC) and the clock switch has selected to run from the FOSC selection (SCS<1:0> = 00), the condition is cleared by performing the following procedure.

When SCS<1:0> = 00 (Running from FOSC selection)

SCS<1:0> = 1x:

Change the SCS bits in the OSCCON register to select the internal oscillator block. This resets the OST timer and allows it to operate again.

OSFIF = 0:

Clear the OSFIF bit in the PIR2 register.

SCS<1:0> = 00:

Change the SCS bits in the OSCCON register to select the FOSC Configuration Word clock selection. Since the OST is not applicable with external clocks, the clock module will immediately switch to the external clock, and the Fail-Safe condition will be cleared.

If the Fail-Safe condition still exists, the OSFIF bit will again be set by hardware.

5.5.4 RESET OR WAKE-UP FROM SLEEP

The FSCM is designed to detect external oscillator or external clock failures.

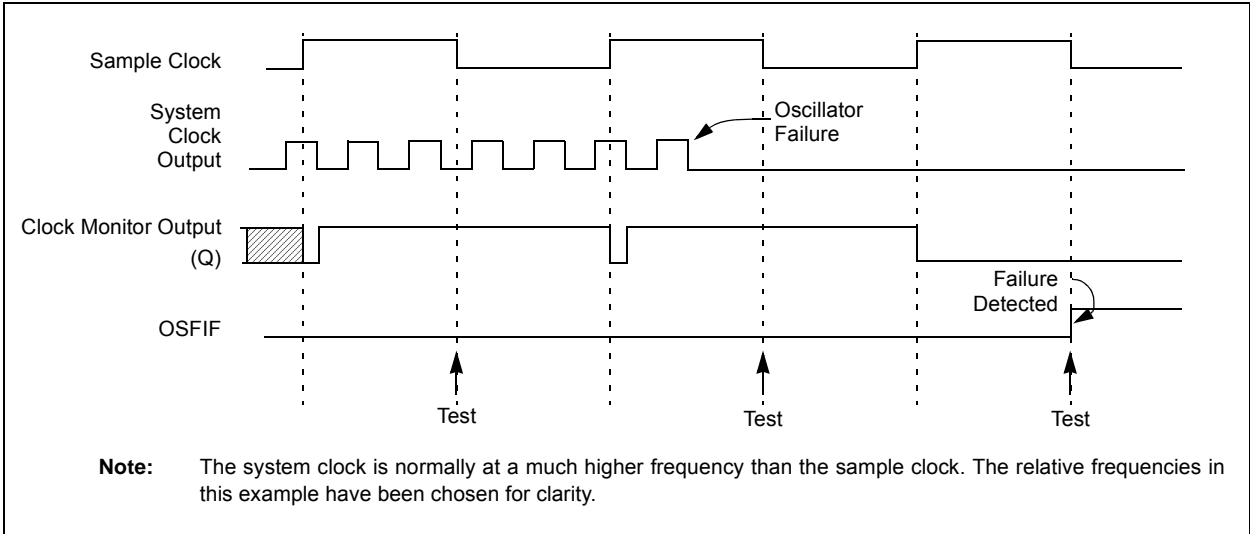
When FSCM is used with an external oscillator, the Oscillator Start-up Timer (OST) count must expire before the FSCM becomes active. The OST is used after waking up from Sleep and after any type of Reset.

When the FSCM is used with external clocks, the OST is not used and 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).

FIGURE 5-8: FSCM TIMING DIAGRAM



5.6 Register Definitions: Oscillator Control

REGISTER 5-1: OSCCON: OSCILLATOR CONTROL REGISTER

| | | | | | | | |
|---------|-----------|---------|---------|---------|----------|---------|---------|
| R/W-0/0 | 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 |
| SPLLEN | 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 **SPLLEN:** Software PLL Enable bit
If PLEN in Configuration Words = 1:
 SPLLEN bit is ignored. 4x PLL is always enabled (subject to oscillator requirements)
If PLEN in Configuration Words = 0:
 1 = 4x PLL is enabled
 0 = 4x PLL is disabled
- bit 6-3 **IRCF<3:0>:** Internal Oscillator Frequency Select bits
 1111 =16 MHz HF
 1110 =8 MHz HF
 1101 =4 MHz HF
 1100 =2 MHz HF
 1011 =1 MHz HF
 1010 =500 kHz HF⁽¹⁾
 1001 =250 kHz HF⁽¹⁾
 1000 =125 kHz HF⁽¹⁾
 0111 =500 kHz MF (default upon Reset)
 0110 =250 kHz MF
 0101 =125 kHz MF
 0100 =62.5 kHz MF
 0011 =31.25 kHz HF⁽¹⁾
 0010 =31.25 kHz MF
 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 = Reserved (defaults to internal oscillator block)
 00 = Clock determined by FOSC<1:0> in Configuration Words.

Note 1: Duplicate frequency derived from HFINTOSC.

REGISTER 5-2: OSCSTAT: OSCILLATOR STATUS REGISTER

| | | | | | | | |
|-------|-------|------|--------|--------|--------|--------|--------|
| U-0 | R-0/q | U-0 | R-0/q | R-0/q | R-q/q | R-0/q | R-0/q |
| — | PLLr | OSTS | HFIOfR | HFIOfL | MFIOFR | 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 | Unimplemented: Read as '0' |
| bit 6 | PLLr: 4x PLL Ready bit 1 = 4x PLL is ready 0 = 4x PLL is not ready |
| 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 | HFIOfL: High-Frequency Internal Oscillator Locked bit 1 = HFINTOSC is at least 2% accurate 0 = HFINTOSC is not 2% accurate |
| bit 2 | MFIOFR: Medium-Frequency Internal Oscillator Ready bit 1 = MFINTOSC is ready 0 = MFINTOSC is not ready |
| 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 is stable 0 = HFINTOSC is not stable |

REGISTER 5-3: OSCTUNE: OSCILLATOR TUNING 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 |
| — | — | TUN<5:0> | | | | | |
| 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 **Unimplemented:** Read as '0'

bit 5-0 **TUN<5:0>:** Frequency Tuning bits

100000 = Minimum frequency

•

•

•

111111 =

000000 = Oscillator module is running at the factory-calibrated frequency.

000001 =

•

•

•

011110 =

011111 = Maximum frequency

TABLE 5-3: 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 | SPLLEN | IRCF<3:0> | | | | — | SCS<1:0> | | 81 |
| OSCSTAT | — | PLLRC | OSTS | HFIOFR | HFIOFL | MFIOFR | LFIOFR | HFIOFS | 82 |
| OSCTUNE | — | — | TUN<5:0> | | | | | | 83 |

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

TABLE 5-4: 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> | | — | 69 |
| | 7:0 | CP | MCLRE | PWRTE | — | — | FOSC<2:0> | | | |

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

6.0 RESETS

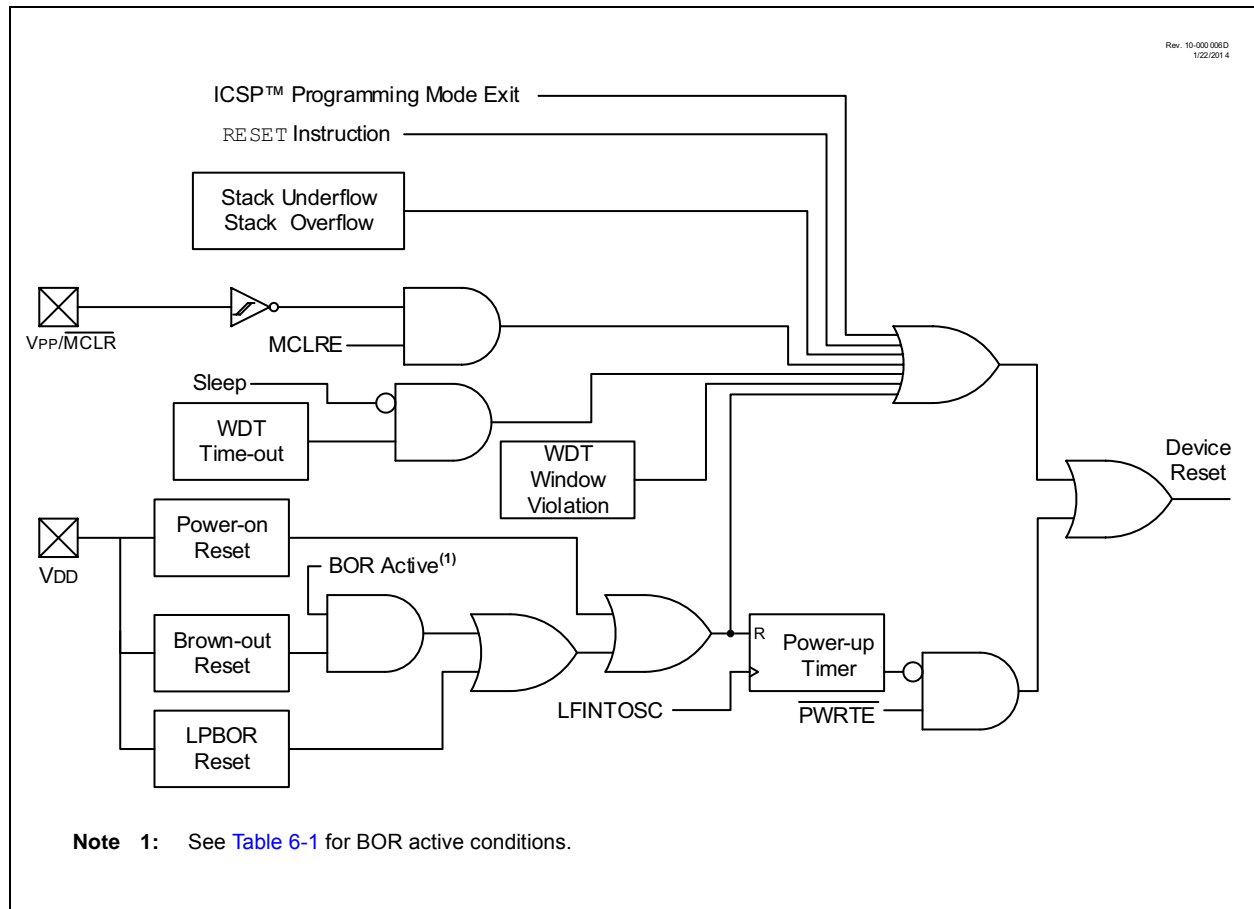
There are multiple ways to reset this device:

- Power-On Reset (POR)
- Brown-Out Reset (BOR)
- Low-Power Brown-Out Reset (LPBOR)
- $\overline{\text{MCLR}}$ 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.

A simplified block diagram of the On-chip Reset Circuit is shown in Figure 6-1.

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



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 PWRT 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 6-1](#) 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.

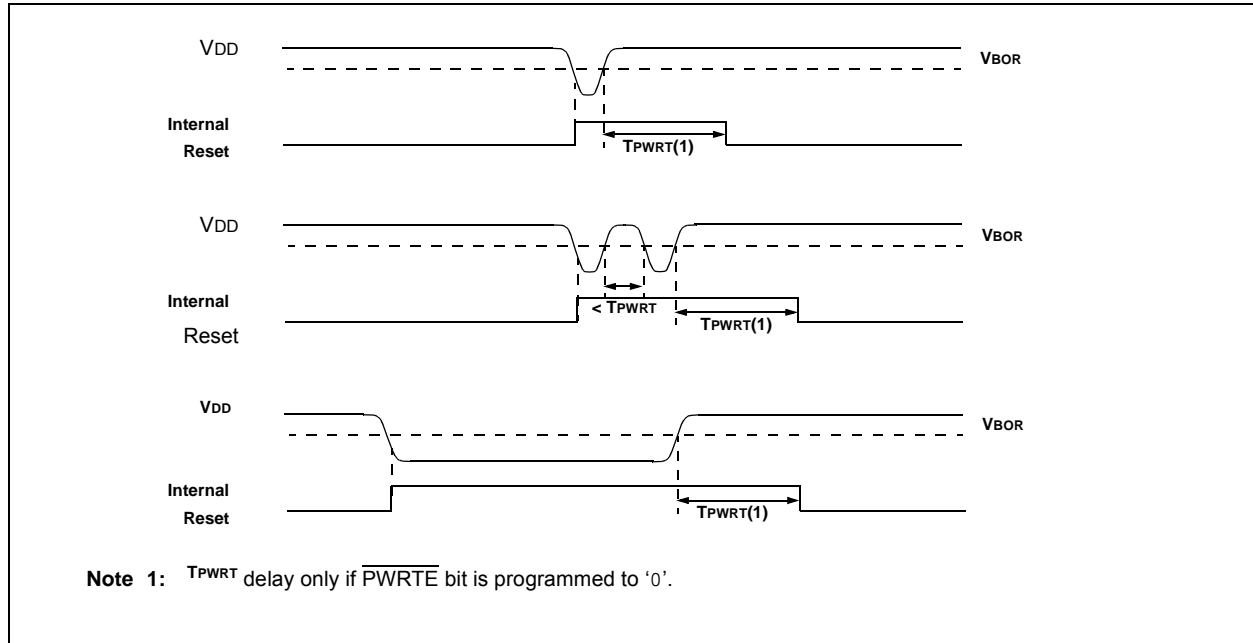
6.2.3 BOR CONTROLLED BY SOFTWARE

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
If BOREN <1:0> in Configuration Words = 01:
 1 = BOR Enabled
 0 = BOR Disabled
If BOREN <1:0> in Configuration Words ≠ 01:
 SBOREN is read/write, but has no effect on the BOR
- bit 6 **BORFS:** Brown-Out Reset Fast Start bit⁽¹⁾
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
If BOREN <1:0> = 11 (Always on) or BOREN <1:0> = 00 (Always off)
 BORFS is Read/Write, but has no effect.
- 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.

6.4 Low-Power Brown-Out Reset (LPBOR)

The Low-Power Brown-Out Reset (LPBOR) operates like the BOR to detect low voltage conditions on the VDD pin. When too low of a voltage is detected, the device is held in Reset. When this occurs, a register bit (BOR) is changed to indicate that a BOR Reset has occurred. The $\overline{\text{BOR}}$ bit in PCON is used for both BOR and the LPBOR. Refer to [Register 6-2](#).

The LPBOR voltage threshold (V_{LPBOR}) has a wider tolerance than the BOR (V_{BOR}), but requires much less current (LPBOR current) to operate. The LPBOR is intended for use when the BOR is configured as disabled ($\text{BOREN} = 00$) or disabled in Sleep mode ($\text{BOREN} = 10$).

Refer to [Figure 6-1](#) to see how the LPBOR interacts with other modules.

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.

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

The $\overline{\text{MCLR}}$ is an optional external input that can reset the device. The $\overline{\text{MCLR}}$ function is controlled by the $\overline{\text{MCLRE}}$ 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 12.1 “PORTA Registers”](#) for more information.

6.6 Watchdog Timer (WDT) Reset

The Watchdog Timer generates a Reset if the firmware does not issue a $\overline{\text{CLRWDT}}$ instruction within the time-out period and the window is open. The $\overline{\text{TO}}$ and $\overline{\text{PD}}$ bits in the STATUS register are changed to indicate a WDT Reset caused by the timer overflowing, and $\overline{\text{WDTWV}}$ bit in the PCON register is changed to indicate a WDT Reset caused by a window violation. See [Section 9.0 “Windowed Watchdog Timer \(WDT\)”](#) for more information.

6.7 RESET Instruction

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

6.8 Stack Overflow/Underflow Reset

The device can reset when the Stack Overflows or Underflows. The $\overline{\text{STKOVF}}$ or $\overline{\text{STKUNF}}$ bits of the PCON register indicate the Reset condition. These Resets are enabled by setting the $\overline{\text{STVREN}}$ bit in Configuration Words. See [Section 3.5.2 “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 $\overline{\text{PWRTÉ}}$ bit of Configuration Words.

6.11 Start-up Sequence

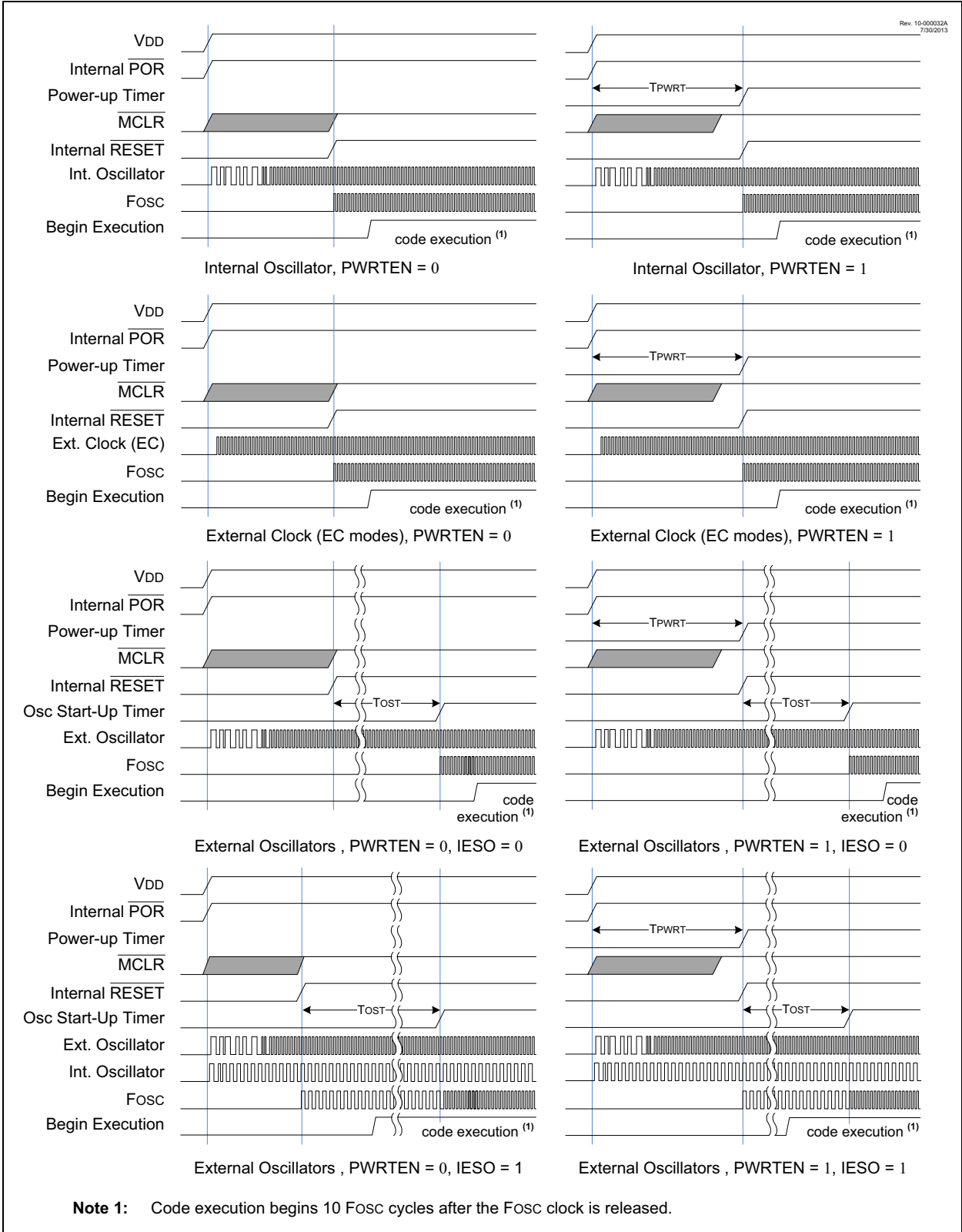
Upon the release of a POR or BOR, the following must occur before the device will begin executing:

1. Power-up Timer runs to completion (if enabled).
2. $\overline{\text{MCLR}}$ 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”](#) for more information.

The Power-up Timer runs independently of $\overline{\text{MCLR}}$ Reset. If $\overline{\text{MCLR}}$ is kept low long enough, the Power-up Timer will expire. Upon bringing $\overline{\text{MCLR}}$ high, the device will begin execution after 10 F_{osc} cycles (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



6.12 Determining the Cause of a Reset

Upon any Reset, multiple bits in the STATUS and PCON registers 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 | 0011 110x |
| \overline{MCLR} Reset during normal operation | 0000h | ---u uuuu | uuuu 0uuu |
| \overline{MCLR} Reset during Sleep | 0000h | ---1 0uuu | uuuu 0uuu |
| WDT Reset | 0000h | ---0 uuuu | uuu0 uuuu |
| WDT Wake-up from Sleep | PC + 1 | ---0 0uuu | uuuu uuuu |
| Brown-out Reset | 0000h | ---1 1uuu | 00uu 11u0 |
| Interrupt Wake-up from Sleep | PC + 1 ⁽¹⁾ | ---1 0uuu | uuuu uuuu |
| RESET Instruction Executed | 0000h | ---u uuuu | uuuu u0uu |
| Stack Overflow Reset (STVREN = 1) | 0000h | ---u uuuu | 1uuu uuuu |
| Stack Underflow Reset (STVREN = 1) | 0000h | ---u uuuu | u1uu uuuu |
| WDT Window Violation | 0000h | ---1 uuuu | uu0u uuuu |

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

Note 1: When the wake-up is due to an interrupt and the Global Interrupt 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 | R/W/HC-1/q | 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 | $\overline{\text{WDTWV}}$ | $\overline{\text{RWDT}}$ | $\overline{\text{RMCLR}}$ | $\overline{\text{RI}}$ | $\overline{\text{POR}}$ | $\overline{\text{BOR}}$ |
| bit 7 | | | | | | bit 0 | |

Legend:

| | |
|---------------------------------|---|
| HC = Bit is cleared by hardware | HS = Bit is set by hardware |
| R = Readable bit | W = Writable bit |
| u = Bit is unchanged | x = Bit is unknown |
| '1' = Bit is set | '0' = Bit is cleared |
| | U = Unimplemented bit, read as '0' |
| | -n/n = Value at POR and BOR/Value at all other Resets |
| | 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 | $\overline{\text{WDTWV}}$: WDT Window Violation Flag bit 1 = A WDT Window Violation Reset has not occurred or set by firmware 0 = A WDT Window Violation Reset has occurred (a CLRWDT instruction was executed either without arming the window or outside the window (cleared by hardware)) |
| bit 4 | $\overline{\text{RWDT}}$: Watchdog Timer Reset Flag bit 1 = A Watchdog Timer Reset has not occurred or set by firmware 0 = A Watchdog Timer Reset has occurred (cleared by hardware) |
| bit 3 | $\overline{\text{RMCLR}}$: MCLR Reset Flag bit 1 = A MCLR Reset has not occurred or set by firmware 0 = A MCLR Reset has occurred (cleared by hardware) |
| bit 2 | $\overline{\text{RI}}$: RESET Instruction Flag bit 1 = A RESET instruction has not been executed or set by firmware 0 = A RESET instruction has been executed (cleared by hardware) |
| bit 1 | $\overline{\text{POR}}$: 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 | $\overline{\text{BOR}}$: 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) |

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 | 86 |
| PCON | STKOVF | STKUNF | WDTWV | RWD \bar{T} | RMCLR | R \bar{I} | POR | BOR | 90 |
| STATUS | — | — | — | T \bar{O} | P \bar{D} | Z | DC | C | 25 |
| WDTCON0 | — | — | WDTPS<4:0> | | | | | SEN | 116 |

Legend: — = unimplemented bit, reads 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.

TABLE 6-6: SUMMARY OF CONFIGURATION WORD WITH RESETS

| 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> | | — | 69 |
| | 7:0 | C \bar{P} | MCLRE | PWRTE | — | — | FOSC<2:0> | | | |
| CONFIG2 | 13:8 | — | — | LVP | DEBUG | LPBOR | BORV | STVREN | PLLEN | 68 |
| | 7:0 | ZCD | — | — | — | — | PPS1WAY | WRT<1:0> | | |
| CONFIG3 | 13:8 | — | — | WDTCCS<2:0> | | | WDTCWS<2:0> | | | 69 |
| | 7:0 | — | WDTE<1:0> | WDTCP5<4:0> | | | | | | |

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

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.

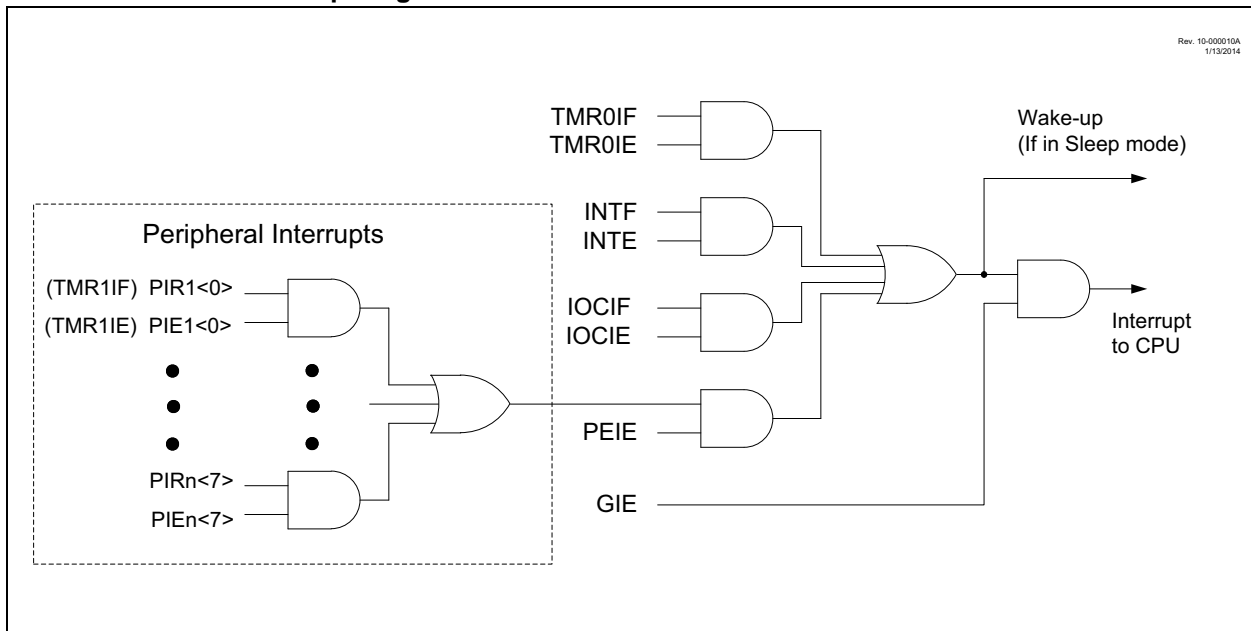
This chapter contains the following information for Interrupts:

- 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



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 PIE1, PIE2 and PIE3 registers)

The INTCON, PIR1, PIR2 and PIR3 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

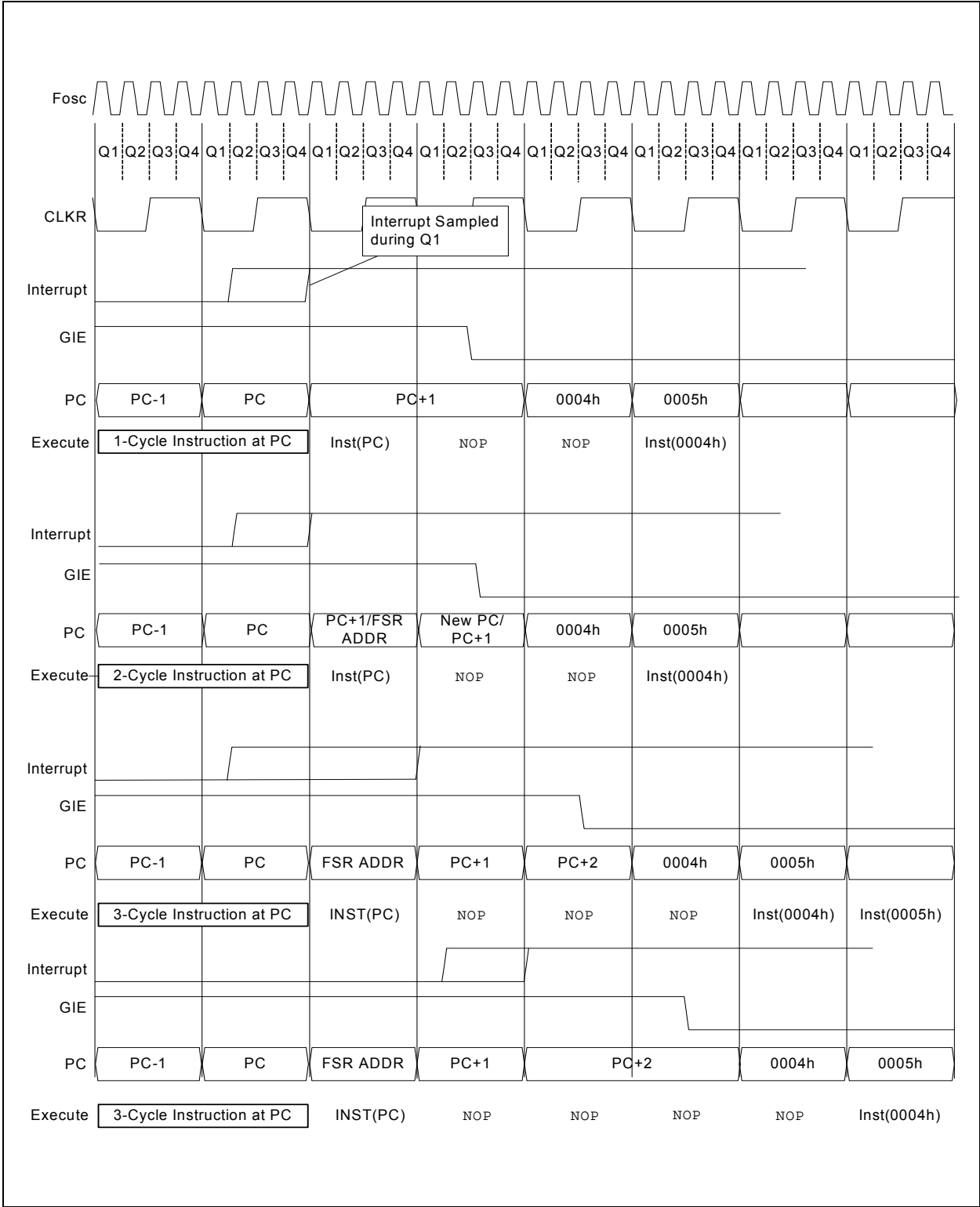
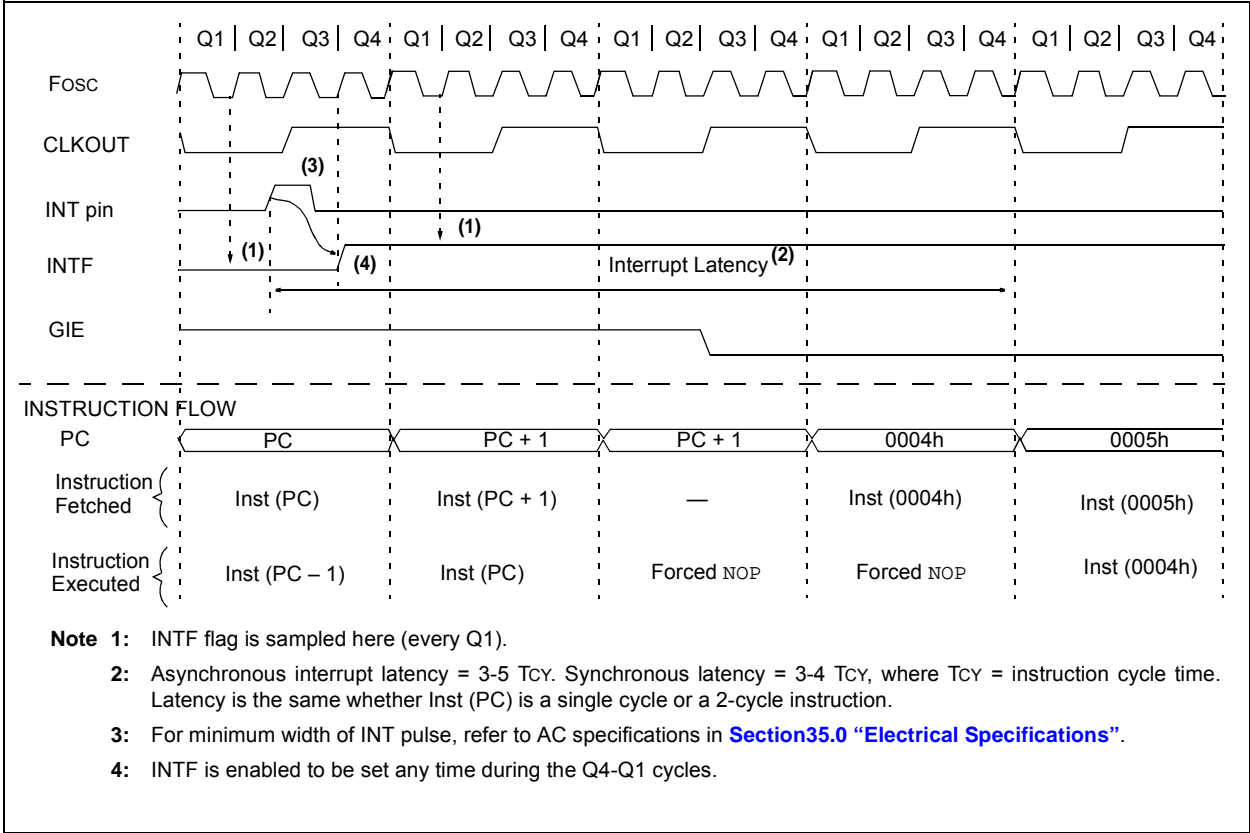


FIGURE 7-3: INT PIN INTERRUPT TIMING



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 [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.

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 ⁽¹⁾ | PEIE ⁽²⁾ | TMR0IE | INTE | IOCIE | TMR0IF | INTF | IOCIF ⁽³⁾ |
| 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⁾
 1 = Enables all active interrupts
 0 = Disables all interrupts
- bit 6 **PEIE:** Peripheral Interrupt Enable bit⁽²⁾
 1 = Enables all active peripheral interrupts
 0 = Disables all peripheral interrupts
- bit 5 **TMR0IE:** 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 **IOCIE:** Interrupt-on-Change Enable bit
 1 = Enables the interrupt-on-change
 0 = Disables the interrupt-on-change
- 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⁽³⁾
 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: Interrupt flag bits are set when an interrupt condition occurs, regardless of the state of its corresponding enable bit or the Global Interrupt Enable bit, GIE of the INTCON register. User software should ensure the appropriate interrupt flag bits are clear prior to enabling an interrupt.

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

3: The IOCIF Flag bit is read-only and cleared when all the interrupt-on-change flags in the IOCxF registers have been cleared by software.

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 | RCIE | TXIE | SSP1IE | CCP1IE | 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 **RCIE:** EUSART Receive Interrupt Enable bit
1 = Enables the EUSART receive interrupt
0 = Disables the EUSART receive interrupt
- bit 4 **TXIE:** EUSART Transmit Interrupt Enable bit
1 = Enables the EUSART transmit interrupt
0 = Disables the EUSART transmit interrupt
- bit 3 **SSP1IE:** Synchronous Serial Port (MSSP) Interrupt Enable bit
1 = Enables the MSSP interrupt
0 = Disables the MSSP interrupt
- bit 2 **CCP1IE:** CCP1 Interrupt Enable bit
1 = Enables the CCP1 interrupt
0 = Disables the CCP1 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.

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

| | | | | | | | |
|---------|---------|---------|-----|---------|---------|---------|---------|
| R/W-0/0 | R/W-0/0 | R/W-0/0 | U-0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 |
| OSFIE | C2IE | C1IE | — | BCL1IE | TMR6IE | TMR4IE | CCP2IE |
| 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 **OSFIE:** Oscillator Fail Interrupt Enable bit
1 = Enables the Oscillator Fail interrupt
0 = Disables the Oscillator Fail interrupt
- bit 6 **C2IE:** Comparator C2 Interrupt Enable bit
1 = Enables the Comparator C2 interrupt
0 = Disables the Comparator C2 interrupt
- bit 5 **C1IE:** Comparator C1 Interrupt Enable bit
1 = Enables the Comparator C1 interrupt
0 = Disables the Comparator C1 interrupt
- bit 4 **Unimplemented:** Read as '0'
- bit 3 **BCL1IE:** MSSP Bus Collision Interrupt Enable bit
1 = Enables the MSSP Bus Collision Interrupt
0 = Disables the MSSP Bus Collision Interrupt
- bit 2 **TMR6IE:** TMR6 to PR6 Match Interrupt Enable bit
1 = Enables the Timer6 to PR6 match interrupt
0 = Disables the Timer6 to PR6 match interrupt
- bit 1 **TMR4IE:** TMR4 to PR4 Match Interrupt Enable bit
1 = Enables the Timer4 to PR4 match interrupt
0 = Disables the Timer4 to PR4 match interrupt
- bit 0 **CCP2IE:** CCP2 Interrupt Enable bit
1 = The CCP2 interrupt is enabled
0 = The CCP2 interrupt is not enabled

Note 1: Bit PEIE of the INTCON register must be set to enable any peripheral interrupt.

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

| | | | | | | | |
|-------|-----|---------|---------|---------|---------|---------|---------|
| 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 |
| — | — | CWGIE | ZCDIE | CLC4IE | CLC3IE | CLC2IE | CLC1IE |
| 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 **CWGIE:** Complementary Waveform Generator (CWG) Interrupt Enable bit
 - 1 = Enables the CWG interrupt
 - 0 = Disables the CWG interrupt
- bit 4 **ZCDIE:** Zero-Cross Detection (ZCD) Interrupt Enable bit
 - 1 = Enables the ZCD interrupt
 - 0 = Disables the ZCD interrupt
- bit 3 **CLC4IE:** Configurable Logic Block 4 Interrupt Enable bit
 - 1 = Enables the CLC 4 interrupt
 - 0 = Disables the CLC 4 interrupt
- bit 2 **CLC3IE:** Configurable Logic Block 3 Interrupt Enable bit
 - 1 = Enables the CLC 3 interrupt
 - 0 = Disables the CLC 3 interrupt
- bit 1 **CLC2IE:** Configurable Logic Block 2 Interrupt Enable bit
 - 1 = Enables the CLC 2 interrupt
 - 0 = Disables the CLC 2 interrupt
- bit 0 **CLC1IE:** Configurable Logic Block 1 Interrupt Enable bit
 - 1 = Enables the CLC 1 interrupt
 - 0 = Disables the CLC 1 interrupt

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

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

| 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 |
|---------|---------|-----------|-----------|---------|-----------|-----------|---------|
| SCANIE | CRCIE | SMT2PWAIE | SMT2PRAIE | SMT2IE | SMT1PWAIE | SMT1PRAIE | SMT1IE |
| 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 **SCANIE:** Scanner Interrupt Enable bit
 1 = Enables the scanner interrupt
 0 = Disables the scanner interrupt
- bit 6 **CRCIE:** CRC Interrupt Enable bit
 1 = Enables the CRC interrupt
 0 = Disables the CRC interrupt
- bit 5 **SMT2PWAIE:** SMT2 Pulse Width Acquisition Interrupt Enable bit
 1 = Enables the SMT2 acquisition interrupt
 0 = Disables the SMT2 acquisition interrupt
- bit 4 **SMT2PRAIE:** SMT2 Period Acquisition Interrupt Enable bit
 1 = Enables the SMT2 acquisition interrupt
 0 = Disables the SMT2 acquisition interrupt
- bit 3 **SMT2IE:** SMT2 Match Interrupt Enable bit
 1 = Enables the SMT2 period match interrupt
 0 = Disables the SMT2 period match interrupt
- bit 2 **SMT1PWAIE:** SMT1 Pulse Width Acquisition Interrupt Enable bit
 1 = Enables the SMT1 acquisition interrupt
 0 = Disables the SMT1 acquisition interrupt
- bit 1 **SMT1PRAIE:** SMT1 Period Acquisition Interrupt Enable bit
 1 = Enables the SMT1 acquisition interrupt
 0 = Disables the SMT1 acquisition interrupt
- bit 0 **SMT1IE:** SMT1 Match Interrupt Enable bit
 1 = Enables the SMT1 period match interrupt
 0 = Disables the SMT1 period match interrupt

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

REGISTER 7-6: PIE5: PERIPHERAL INTERRUPT ENABLE REGISTER 5

| R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | U-0 | R/HS-0/0 | R/W/HS-0/0 | R/W/HS-0/0 |
|---------|---------|---------|---------|-----|----------|------------|------------|
| TMR3GIE | TMR3IE | TMR5GIE | TMR5IE | — | AT1IE | PID1EIE | PID1DIE |
| 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 **TMR3GIE:** Timer3 Gate Interrupt Enable bit
1 = Enables the Timer3 Gate interrupt
0 = Disables the Timer3 Gate interrupt
- bit 6 **TMR3IE:** Timer3 Overflow Interrupt Enable bit
1 = Enables the Timer3 overflow interrupt
0 = Disables the Timer3 overflow interrupt
- bit 5 **TMR5GIE:** Timer5 Gate Interrupt Enable bit
1 = Enables the Timer5 Gate interrupt
0 = Disables the Timer5 Gate interrupt
- bit 4 **TMR5IE:** Timer5 Overflow Interrupt Enable bit
1 = Enables the Timer5 overflow interrupt
0 = Disables the Timer5 overflow interrupt
- bit 3 **Unimplemented:** Read as '0'
- bit 2 **AT1IE:** Angular Timer 1 Interrupt Enable bit
1 = Enables the Angular Timer 1 interrupt
0 = Disables the Angular Timer 1 interrupt
- bit 1 **PID1EIE:** PID Error Interrupt Enable bit
1 = Enables the PID error interrupt
0 = Disables the PID error interrupt
- bit 0 **PID1DIE:** PID Interrupt Enable bit
1 = Enables the PID interrupt
0 = Disables the PID interrupt

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

REGISTER 7-7: PIR1: PERIPHERAL INTERRUPT REQUEST 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 | ADIF | RCIF | TXIF | 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 Interrupt Flag bit
1 = Interrupt is pending
0 = Interrupt is not pending
- bit 5 **RCIF:** EUSART Receive Interrupt Flag bit
1 = Interrupt is pending
0 = Interrupt is not pending
- bit 4 **TXIF:** EUSART Transmit Interrupt Flag bit
1 = Interrupt is pending
0 = Interrupt is not pending
- bit 3 **SSP1IF:** Synchronous Serial Port (MSSP) 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 Interrupt 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: PIR2: PERIPHERAL INTERRUPT REQUEST REGISTER 2

| | | | | | | | |
|---------|---------|---------|-----|---------|---------|---------|---------|
| R/W-0/0 | R/W-0/0 | R/W-0/0 | U-0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 |
| OSFIF | C2IF | C1IF | — | BCL1IF | TMR6IF | TMR4IF | CCP2IF |
| 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 **OSFIF:** Oscillator Fail Interrupt Flag bit
 1 = Interrupt is pending
 0 = Interrupt is not pending
- bit 6 **C2IF:** Comparator C2 Interrupt Flag bit
 1 = Interrupt is pending
 0 = Interrupt is not pending
- bit 5 **C1IF:** Comparator C1 Interrupt Flag bit
 1 = Interrupt is pending
 0 = Interrupt is not pending
- bit 4 **Unimplemented:** Read as '0'
- bit 3 **BCL1IF:** MSSP Bus Collision Interrupt Flag bit
 1 = Interrupt is pending
 0 = Interrupt is not pending
- bit 2 **TMR6IF:** Timer6 to PR6 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 **CCP2IF:** CCP2 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.

REGISTER 7-9: PIR3: PERIPHERAL INTERRUPT REQUEST REGISTER 3

| | | | | | | | |
|-------|-----|---------|---------|---------|---------|---------|---------|
| 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 |
| — | — | CWGIF | ZCDIF | CLC4IF | CLC3IF | CLC2IF | CLC1IF |
| 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 **CWGIF:** CWG Interrupt Flag bit
 1 = Interrupt is pending
 0 = Interrupt is not pending
- bit 4 **ZCDIF:** ZCD Interrupt Flag bit
 1 = Interrupt is pending
 0 = Interrupt is not pending
- bit 3 **CLC4IF:** Configurable Logic Block 4 Interrupt Flag bit
 1 = Interrupt is pending
 0 = Interrupt is not pending
- bit 2 **CLC3IF:** Configurable Logic Block 3 Interrupt Flag bit
 1 = Interrupt is pending
 0 = Interrupt is not pending
- bit 1 **CLC2IF:** Configurable Logic Block 2 Interrupt Flag bit
 1 = Interrupt is pending
 0 = Interrupt is not pending
- bit 0 **CLC1IF:** Configurable Logic Block 1 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.

REGISTER 7-10: PIR4: PERIPHERAL INTERRUPT REQUEST REGISTER 4

| | | | | | | | |
|---------|---------|-----------|-----------|---------|-----------|-----------|---------|
| 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 |
| SCANIF | CRCIF | SMT2PWAIF | SMT2PRAIF | SMT2IF | SMT1PWAIF | SMT1PRAIF | SMT1IF |
| 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 **SCANIF:** Scanner Interrupt Flag bit
1 = Interrupt is pending
0 = Interrupt is not pending
- bit 6 **CRCIF:** CRC Interrupt Flag bit
1 = Interrupt is pending
0 = Interrupt is not pending
- bit 5 **SMT2PWAIF:** SMT2 Pulse Width Acquisition Interrupt Flag bit
1 = Interrupt is pending
0 = Interrupt is not pending
- bit 4 **SMT2PRAIF:** SMT2 Period Acquisition Interrupt Flag bit
1 = Interrupt is pending
0 = Interrupt is not pending
- bit 3 **SMT2IF:** SMT2 Match Interrupt Flag bit
1 = Interrupt is pending
0 = Interrupt is not pending
- bit 2 **SMT1PWAIF:** SMT1 Pulse Width Acquisition Interrupt Flag bit
1 = Interrupt is pending
0 = Interrupt is not pending
- bit 1 **SMT1PRAIF:** SMT1 Period Acquisition Interrupt Flag bit
1 = Interrupt is pending
0 = Interrupt is not pending
- bit 0 **SMT1IF:** SMT1 Match 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.

REGISTER 7-11: PIR5: PERIPHERAL INTERRUPT REQUEST REGISTER 5

| R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | U-0 | R/HS-0/0 | R/W/HS-0/0 | R/W/HS-0/0 |
|---------|---------|---------|---------|-----|----------|------------|------------|
| TMR3GIF | TMR3IF | TMR5GIF | TMR5IF | — | AT1IF | PID1EIF | PID1DIF |
| 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 **TMR3GIF:** Timer3 Gate Interrupt Flag bit
1 = Interrupt is pending
0 = Interrupt is not pending
- bit 6 **TMR3IF:** Timer3 Overflow Interrupt Flag bit
1 = Interrupt is pending
0 = Interrupt is not pending
- bit 5 **TMR5GIF:** Timer5 Gate Interrupt Flag bit
1 = Interrupt is pending
0 = Interrupt is not pending
- bit 4 **TMR5IF:** Timer5 Overflow Interrupt Flag bit
1 = Interrupt is pending
0 = Interrupt is not pending
- bit 3 Unimplemented: Read as '0'
- bit 2 **AT1IF:** Angular Timer 1 Interrupt Flag bit
1 = Interrupt is pending
0 = Interrupt is not pending
- bit 1 **PID1EIF:** PID Error Interrupt Flag bit
1 = Interrupt is pending
0 = Interrupt is not pending
- bit 0 **PID1DIF:** PID 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 Interrupt 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 | IOCF | 97 |
| OPTION_REG | WPUEN | INTEDG | TMR0CS | TMR0SE | PSA | PS<2:0> | | | 223 |
| PIE1 | TMR1GIE | ADIE | RCIE | TXIE | SSP1IE | CCP1IE | TMR2IE | TMR1IE | 98 |
| PIE2 | OSFIE | C2IE | C1IE | — | BCL1IE | TMR6IE | TMR4IE | CCP2IE | 107 |
| PIE3 | — | — | CWGIE | ZCDIE | CLC4IE | CLC3IE | CLC2IE | CLC1IE | 108 |
| PIE4 | SCANIE | CRCIE | SMT2PWAIE | SMT2PRAIE | SMT2IE | SMT1PWAIE | SMT1PRAIE | SMT1IF | 101 |
| PIE5 | TMR3GIE | TMR3IE | TMR5GIE | TMR5IE | — | AT1IE | PID1EIE | PID1DIE | 102 |
| PIR1 | TMR1GIF | ADIF | RCIF | TXIF | SSP1IF | CCP1IF | TMR2IF | TMR1IF | 103 |
| PIR2 | OSFIF | C2IF | C1IF | — | BCL1IF | TMR6IF | TMR4IF | CCP2IF | 112 |
| PIR3 | — | — | CWGFIF | ZCDFIF | CLC4IF | CLC3IF | CLC2IF | CLC1IF | 113 |
| PIR4 | SCANIF | CRCIF | SMT2PWAIF | SMT2PRAIF | SMT2IF | SMT1PWAIF | SMT1PRAIF | SMT1IF | 106 |
| PIR5 | TMR3GIF | TMR3IF | TMR5GIF | TMR5IF | — | AT1IF | PID1EIF | PID1DIF | 107 |

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

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
 - Timer1 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
- CWG modules using HFINTOSC

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 the FVR module. See [Section 15.0 “Fixed Voltage Reference \(FVR\)”](#) for more information on this module.

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.

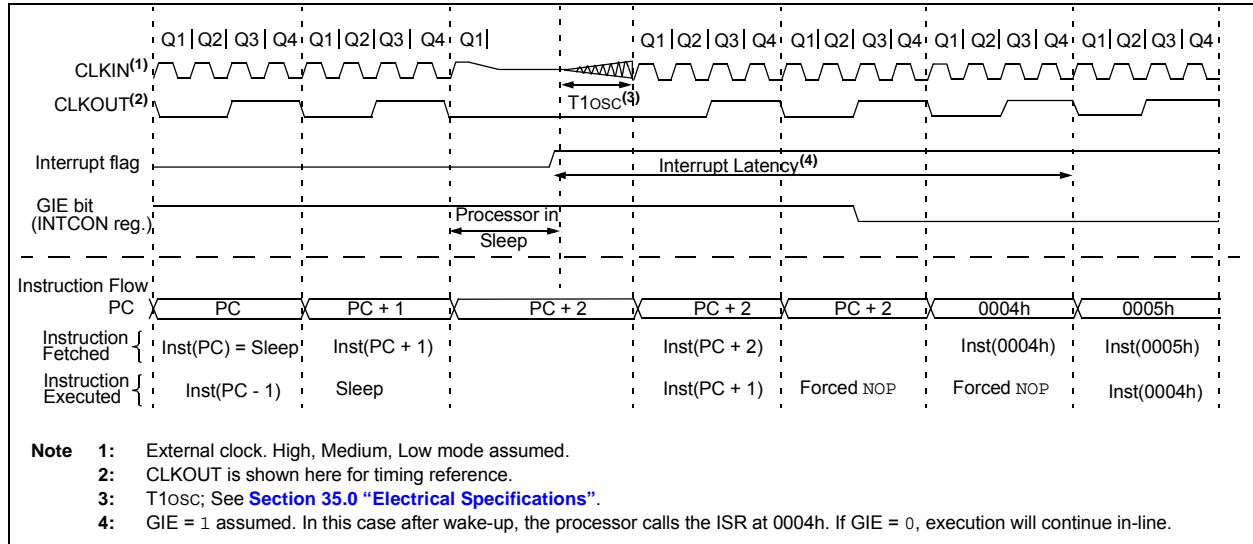
8.1.1 WAKE-UP USING INTERRUPTS

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



8.2 Low-Power Sleep Mode

This 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.

Low-Power Sleep mode allows the user to optimize the operating current in Sleep. Low-Power Sleep mode can be selected by setting the VREGPM bit of the VREGCON register, putting the LDO and reference circuitry in a low-power state whenever 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)

The Complementary Waveform Generator (CWG) can utilize the HFINTOSC oscillator as either a clock source or as an input source. Under certain conditions, when the HFINTOSC is selected for use with the CWG modules, the HFINTOSC will remain active during Sleep. This will have a direct effect on the Sleep mode current.

Please refer to sections [Section 28.11 "Operation During Sleep"](#) for more information.

Note: The PIC16LF1615/9 does not have a configurable Low-Power Sleep mode. PIC16LF1615/9 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 VDD and I/O voltage than the PIC16F1615/9. See [Section 35.0 "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: PIC16F1615/9 only.

2: See [Section 35.0 “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 | 97 |
| IOCAF | — | — | IOCAF5 | IOCAF4 | IOCAF3 | IOCAF2 | IOCAF1 | IOCAF0 | 180 |
| IOCAN | — | — | IOCAN5 | IOCAN4 | IOCAN3 | IOCAN2 | IOCAN1 | IOCAN0 | 180 |
| IOCAP | — | — | IOCAP5 | IOCAP4 | IOCAP3 | IOCAP2 | IOCAP1 | IOCAP0 | 180 |
| IOCCP | IOCCP7 ⁽¹⁾ | IOCCP6 ⁽¹⁾ | IOCCP5 | IOCCP4 | IOCCP3 | IOCCP2 | IOCCP1 | IOCCP0 | 180 |
| IOCCN | IOCCN7 ⁽¹⁾ | IOCCN6 ⁽¹⁾ | IOCCN5 | IOCCN4 | IOCCN3 | IOCCN2 | IOCCN1 | IOCCN0 | 180 |
| IOCCF | IOCCF7 ⁽¹⁾ | IOCCF6 ⁽¹⁾ | IOCCF5 | IOCCF4 | IOCCF3 | IOCCF2 | IOCCF1 | IOCCF0 | 180 |
| PIE1 | TMR1GIE | ADIE | RCIE | TXIE | SSP1IE | CCP1IE | TMR2IE | TMR1IE | 98 |
| PIE2 | OSFIE | C2IE | C1IE | — | BCL1IE | TMR6IE | TMR4IE | CCP2IE | 107 |
| PIE3 | — | — | CWGIE | ZCDIE | CLC4IE | CLC3IE | CLC2IE | CLC1IE | 108 |
| PIE4 | SCANIE | CRCIE | SMT2PWAIE | SMT2PRAIE | SMT2IE | SMT1PWAIE | SMT1PRAIE | SMT1IF | 101 |
| PIE5 | TMR3GIE | TMR3IE | TMR5GIE | TMR5IE | — | AT1IE | PID1EIE | PID1DIE | 102 |
| PIR1 | TMR1GIF | ADIF | RCIF | TXIF | SSP1IF | CCP1IF | TMR2IF | TMR1IF | 103 |
| PIR2 | OSFIF | C2IF | C1IF | — | BCL1IF | TMR6IF | TMR4IF | CCP2IF | 112 |
| PIR3 | — | — | CWGIF | ZCDIF | CLC4IF | CLC3IF | CLC2IF | CLC1IF | 113 |
| PIR4 | SCANIF | CRCIF | SMT2PWAIF | SMT2PRAIF | SMT2IF | SMT1PWAIF | SMT1PRAIF | SMT1IF | 106 |
| PIR5 | TMR3GIF | TMR3IF | TMR5GIF | TMR5IF | — | AT1IF | PID1EIF | PID1DIF | 107 |
| STATUS | — | — | — | \overline{TO} | \overline{PD} | Z | DC | C | 25 |
| WDTCON0 | — | — | WDTPS<4:0> | | | | | SEN | 116 |

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

Note 1: PIC16(L)F1619 only.

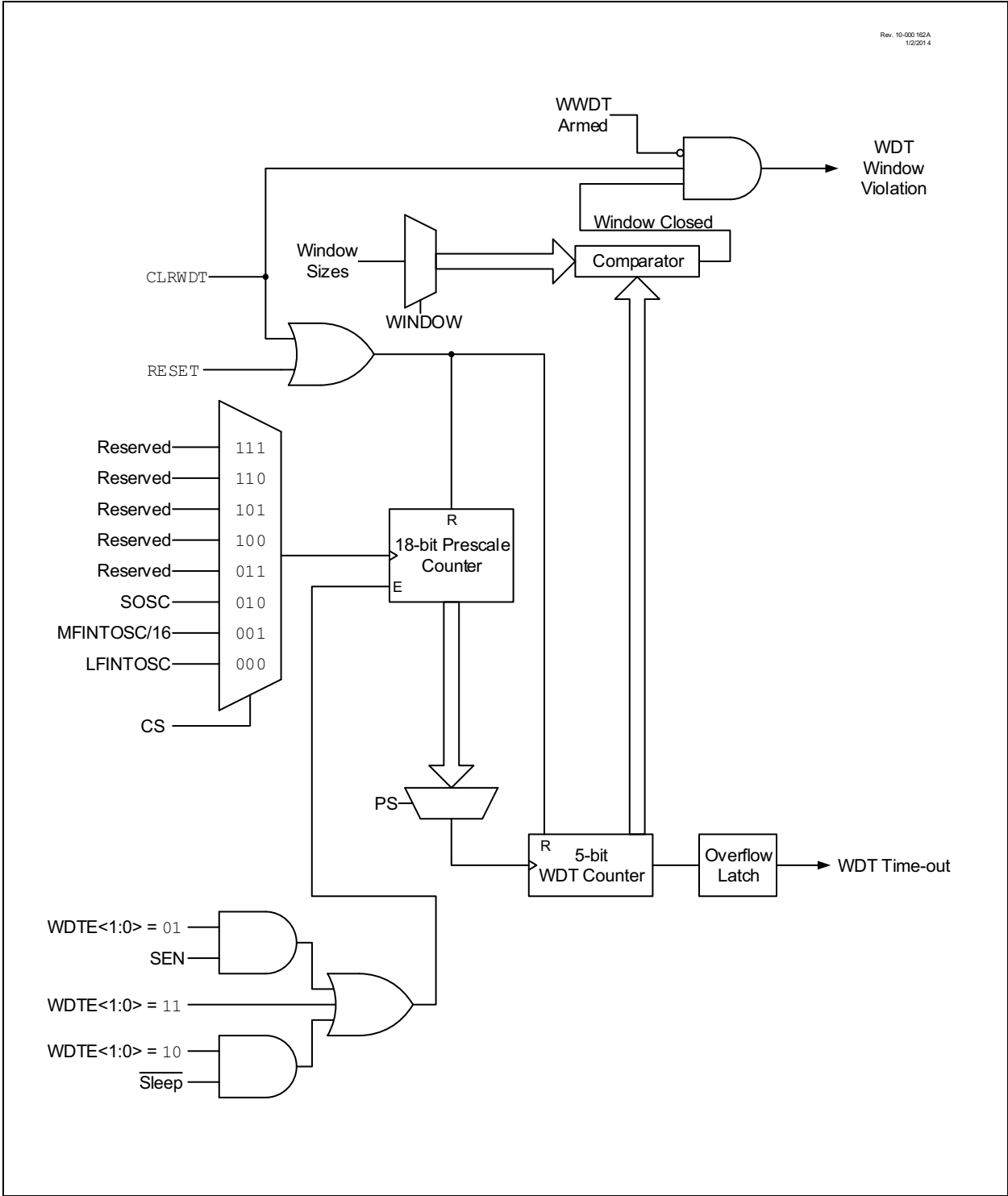
9.0 WINDOWED WATCHDOG TIMER (WDT)

The Watchdog Timer (WDT) 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 Windowed Watchdog Timer (WDT) differs in that CLRWDT instructions are only accepted when they are performed within a specific window during the time-out period.

The WDT has the following features:

- Selectable 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)
- Configurable window size from 12.5 to 100 percent of the time-out period
- Multiple Reset conditions
- Operation during Sleep

FIGURE 9-1: WATCHDOG TIMER BLOCK DIAGRAM



9.1 Independent Clock Source

The WDT can derive its time base from either the 31 kHz LFINTOSC or 31.25 kHz MFINTOSC internal oscillators, depending on the value of either the WDTCCS<2:0> configuration bits or the WDTCS<2:0> bits of WDTCON1. Time intervals in this chapter are based on a minimum nominal interval of 1 ms. See [Section 35.0 “Electrical Specifications”](#) for LFINTOSC and MFINTOSC tolerances.

9.2 WDT Operating Modes

The Watchdog Timer module has four operating modes controlled by the WDTE<1:0> bits in Configuration Words. See [Table 9-1](#).

9.2.1 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.

9.2.2 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.

9.2.3 WDT CONTROLLED BY SOFTWARE

When the WDTE bits of Configuration Words are set to ‘01’, the WDT is controlled by the SEN bit of the WDTCON0 register.

WDT protection is unchanged by Sleep. See [Table 9-1](#) for more details.

TABLE 9-1: WDT OPERATING MODES

| WDTE<1:0> | SEN | 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 |

9.3 Time-Out Period

The WDTPS bits of the WDTCON0 register set the time-out period from 1 ms to 256 seconds (nominal). After a Reset, the default time-out period is two seconds.

9.4 Watchdog Window

The Watchdog Timer has an optional Windowed mode that is controlled by the WDTCWS<2:0> Configuration bits and WINDOW<2:0> bits of the WDTCON1 register. In the Windowed mode, the CLRWDT instruction must occur within the allowed window of the WDT period. Any CLRWDT instruction that occurs outside of this window will trigger a window violation and will cause a WDT Reset, similar to a WDT time out. See [Figure 9-2](#) for an example.

The window size is controlled by the WDTCWS<2:0> Configuration bits, or the WINDOW<2:0> bits of WDTCON1, if WDTCWS<2:0> = 111.

In the event of a window violation, a Reset will be generated and the WDTWV bit of the PCON register will be cleared. This bit is set by a POR or can be set in firmware.

9.5 Clearing the WDT

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

- Any Reset
- Valid CLRWDT instruction is executed
- Device enters Sleep
- Device wakes up from Sleep
- WDT is disabled
- Oscillator Start-up Timer (OST) is running
- Any write to the WDTCON0 or WDTCON1 registers

9.5.1 CLRWDT CONSIDERATIONS (WINDOWED MODE)

When in Windowed mode, the WDT must be armed before a CLRWDT instruction will clear the timer. This is performed by reading the WDTCON0 register. Executing a CLRWDT instruction without performing such an arming action will trigger a window violation.

See [Table 9-2](#) for more information.

9.6 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.

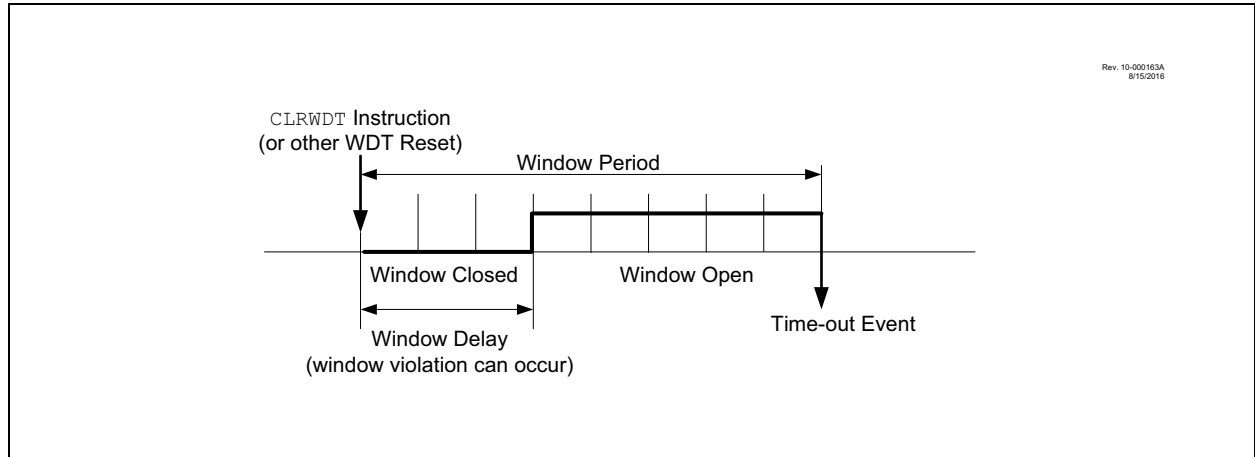
The WDT remains clear until the OST, if enabled, completes. See [Section 5.0 “Oscillator Module”](#) for more information on the OST.

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”](#) for more information.

TABLE 9-2: WDT CLEARING CONDITIONS

| Conditions | WDT |
|--|------------|
| WDTE<1:0> = 00 | Cleared |
| WDTE<1:0> = 01 and SEN = 0 | |
| WDTE<1:0> = 10 and enter Sleep | |
| CLRWDT Command | |
| Oscillator Fail Detected | |
| Exit Sleep + System Clock = T1OSC, EXTRC, INTOSC, EXTCLK | |
| Change INTOSC divider (IRCF bits) | Unaffected |

FIGURE 9-2: WINDOW PERIOD AND DELAY



9.7 Register Definitions: Windowed Watchdog Timer Control

REGISTER 9-1: WDTCON0: WATCHDOG TIMER CONTROL REGISTER 0

| | | | | | | | |
|-------|-----|--|--|--|--|--|---------|
| U-0 | U-0 | R/W ⁽³⁾ -q/q ⁽²⁾ | R/W ⁽³⁾ -q/q ⁽²⁾ | R/W ⁽³⁾ -q/q ⁽²⁾ | R/W ⁽³⁾ -q/q ⁽²⁾ | R/W ⁽³⁾ -q/q ⁽²⁾ | R/W-0/0 |
| — | — | WDTPS<4:0> ⁽¹⁾ | | | | | 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 | q = Value depends on condition |

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

bit 5-1 **WDTPS<4:0>:** Watchdog Timer Prescale 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 (2²³) (Interval 256s nominal)

10001 = 1:4194304 (2²²) (Interval 128s nominal)

10000 = 1:2097152 (2²¹) (Interval 64s nominal)

01111 = 1:1048576 (2²⁰) (Interval 32s nominal)

01110 = 1:524288 (2¹⁹) (Interval 16s nominal)

01101 = 1:262144 (2¹⁸) (Interval 8s nominal)

01100 = 1:131072 (2¹⁷) (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 **SEN:** Software Enable/Disable for Watchdog Timer bit

If WDTE<1:0> = 1x:

This bit is ignored.

If WDTE<1:0> = 01:

1 = WDT is turned on

0 = WDT is turned off

If WDTE<1:0> = 00:

This bit is ignored.

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

2: When WDTCPSS <4:0> in CONFIG3 = 11111, the Reset value of WDTPS<4:0> is 01011. Otherwise, the Reset value of WDTPS<4:0> is equal to WDTCPSS<4:0> in CONFIG3.

3: When WDTCPSS <4:0> in CONFIG3 ≠ 11111, these bits are read-only.

REGISTER 9-2: WDTCON1: WATCHDOG TIMER CONTROL REGISTER 1

| | | | | | | | |
|-------|--|--|--|-----|--|--|--|
| U-0 | R/W ⁽³⁾ -q/q ⁽¹⁾ | R/W ⁽³⁾ -q/q ⁽¹⁾ | R/W ⁽³⁾ -q/q ⁽¹⁾ | U-0 | R/W ⁽⁴⁾ -q/q ⁽²⁾ | R/W ⁽⁴⁾ -q/q ⁽²⁾ | R/W ⁽⁴⁾ -q/q ⁽²⁾ |
| — | WDTCS<2:0> | | | — | WINDOW<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 | q = Value depends on condition |

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

bit 6-4 **WDTCS<2:0>:** Watchdog Timer Clock Select bits

 111 = Reserved

 •

 •

 •

 010 = Reserved

 001 = MFINTOSC 31.25 kHz

 000 = LFINTOSC 31 kHz

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

bit 2-0 **WINDOW<2:0>:** Watchdog Timer Window Select bits

| WINDOW<2:0> | Window delay Percent of time | Window opening Percent of time |
|-------------|---------------------------------|-----------------------------------|
| 111 | N/A | 100 |
| 110 | 12.5 | 87.5 |
| 101 | 25 | 75 |
| 100 | 37.5 | 62.5 |
| 011 | 50 | 50 |
| 010 | 62.5 | 37.5 |
| 001 | 75 | 25 |
| 000 | 87.5 | 12.5 |

- Note 1:** If WDTCCS <2:0> in CONFIG3 = 111, the Reset value of WDTCS<2:0> is 000.
- 2:** The Reset value of WINDOW<2:0> is determined by the value of WDTCWS<2:0> in the CONFIG3 register.
- 3:** If WDTCCS<2:0> in CONFIG3 ≠ 111, these bits are read-only.
- 4:** If WDTCWS<2:0> in CONFIG3 ≠ 111, these bits are read-only.

REGISTER 9-3: WDTPSL: WDT PRESCALE SELECT LOW BYTE REGISTER (READ ONLY)

| | | | | | | | |
|---------------------------|-------|-------|-------|-------|-------|-------|-------|
| R-0/0 | R-0/0 | R-0/0 | R-0/0 | R-0/0 | R-0/0 | R-0/0 | R-0/0 |
| PSCNT<7:0> ⁽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 | |

bit 7-0 **PSCNT<7:0>**: Prescale Select Low Byte bits⁽¹⁾

Note 1: The 18-bit WDT prescale value, PSCNT<17:0> includes the WDTPSL, WDTPSH and the lower bits of the WDTTMR registers. PSCNT<17:0> is intended for debug operations and should be read during normal operation.

REGISTER 9-4: WDTPSH: WDT PRESCALE SELECT HIGH BYTE REGISTER (READ ONLY)

| | | | | | | | |
|----------------------------|-------|-------|-------|-------|-------|-------|-------|
| R-0/0 | R-0/0 | R-0/0 | R-0/0 | R-0/0 | R-0/0 | R-0/0 | R-0/0 |
| PSCNT<15:8> ⁽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 | |

bit 7-0 **PSCNT<15:8>**: Prescale Select High Byte bits⁽¹⁾

Note 1: The 18-bit WDT prescale value, PSCNT<17:0> includes the WDTPSL, WDTPSH and the lower bits of the WDTTMR registers. PSCNT<17:0> is intended for debug operations and should be read during normal operation.

REGISTER 9-5: WDTTMR: WDT TIMER REGISTER (READ ONLY)

| | | | | | | | |
|-------------|-------|-------|-------|-------|-----------------------------|-------|-------|
| R-0/0 | R-0/0 | R-0/0 | R-0/0 | R-0/0 | R-0/0 | R-0/0 | R-0/0 |
| WDTTMR<3:0> | | | | STATE | PSCNT<17:16> ⁽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 | |

bit 7-3 **WDTTMR<4:0>**: Watchdog Timer Value

bit 2 **STATE**: WDT Armed Status bit
 1 = WDT is armed
 0 = WDT is not armed

bit 1-0 **PSCNT<17:16>**: Prescale Select Upper Byte bits⁽¹⁾

Note 1: The 18-bit WDT prescale value, PSCNT<17:0> includes the WDTPSL, WDTPSH and the lower bits of the WDTTMR registers. PSCNT<17:0> is intended for debug operations and should be read during normal operation.

TABLE 9-3: 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 | SPLLEN | IRCF<3:0> | | | — | | SCS<1:0> | | 81 |
| PCON | STKOVF | STKUNF | WDTWV | RWD \bar{T} | RMCLR | R \bar{I} | POR | BOR | 90 |
| STATUS | — | — | — | T \bar{O} | P \bar{D} | Z | DC | C | 25 |
| WDTCON0 | — | — | WDTPS<4:0> | | | | SEN | | 116 |
| WDTCON1 | — | WDTCS<2:0> | | | — | WINDOW<2:0> | | | 116 |
| WDTPSL | PSCNT<7:0> | | | | | | | | 116 |
| WDTPSH | PSCNT<15:8> | | | | | | | | 116 |
| WDTTMR | — | WDTTMR<4:0> | | | | STATE | PSCNT<17:16> | | 116 |

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

TABLE 9-4: 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 | CLKOUTEN | BOREN<1:0> | | — | 69 |
| | 7:0 | C \bar{P} | MCLRE | PWRTE | — | — | FOSC<2:0> | | — | |
| CONFIG3 | 13:8 | — | — | WDTCCS<2:0> | | | WDTCWS<2:0> | | — | 69 |
| | 7:0 | — | WDTE<1:0> | WDTCPSS<4:0> | | | | — | | |

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

10.0 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.

10.1 PMADRL and PMADRH Registers

The PMADRH:PMADRL register pair can address up to a maximum of 16K 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.

10.1.1 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.

10.2 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 10-1](#) for Erase Row size and the number of write latches for Flash program memory.

TABLE 10-1: FLASH MEMORY ORGANIZATION BY DEVICE

| Device | Row Erase (words) | Write Latches (words) |
|---------------|-------------------|-----------------------|
| PIC16(L)F1615 | 32 | 32 |
| PIC16(L)F1619 | | |

10.2.1 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 2-cycle instruction on the next instruction after the RD bit is set.

FIGURE 10-1: FLASH PROGRAM MEMORY READ FLOWCHART

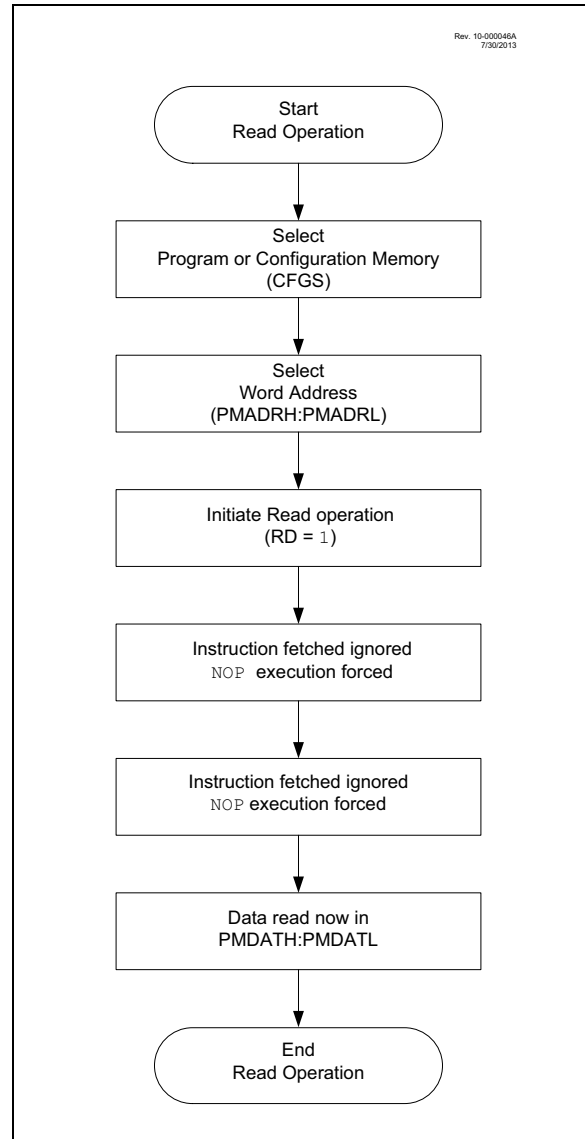


FIGURE 10-2: FLASH PROGRAM MEMORY READ CYCLE EXECUTION



EXAMPLE 10-1: 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    ;
  MOVWF   PMADRH          ; Store MSB of address

  BCF     PMCON1,CFG5      ; Do not select Configuration Space
  BSF     PMCON1,RD        ; Initiate read
  NOP     ; Ignored (Figure 10-2)
  NOP     ; Ignored (Figure 10-2)

  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
  
```

10.2.2 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

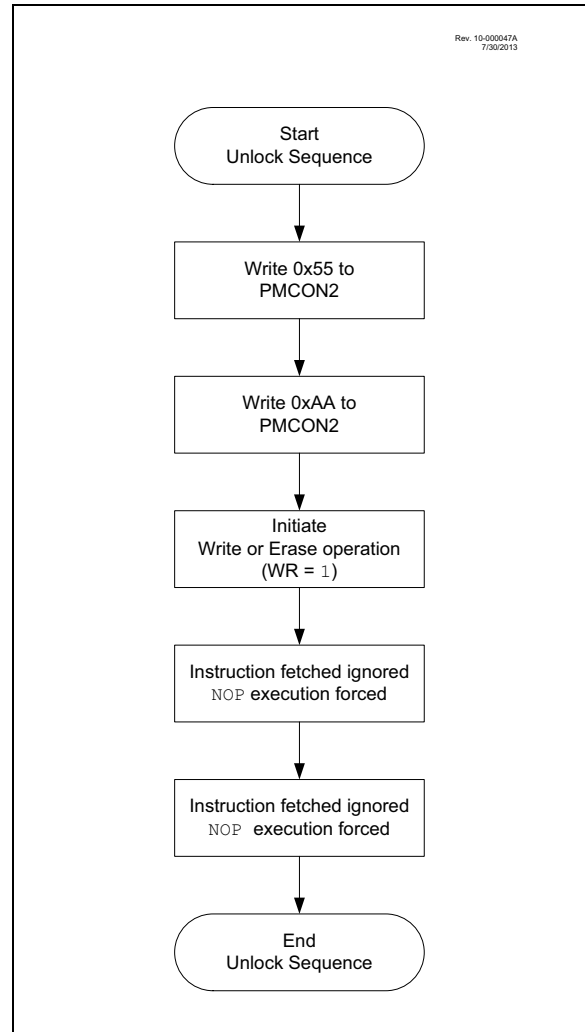
The unlock sequence consists of the following steps:

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 10-3: FLASH PROGRAM MEMORY UNLOCK SEQUENCE FLOWCHART



10.2.3 ERASING FLASH PROGRAM MEMORY

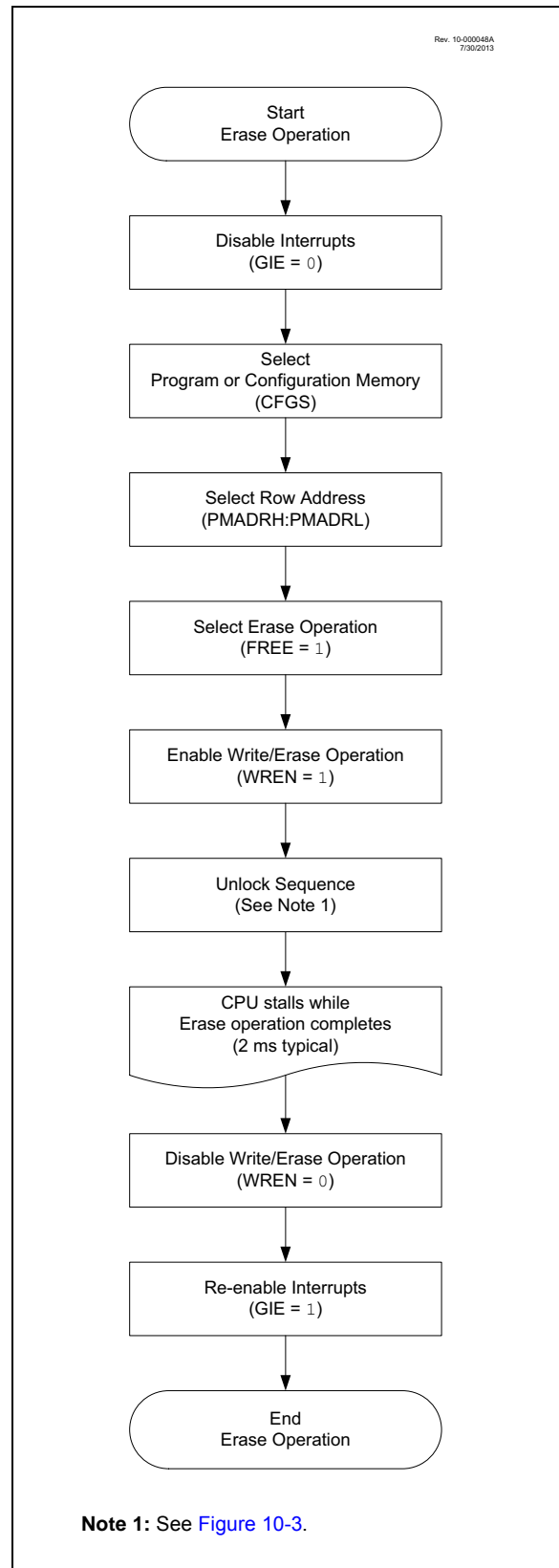
While executing code, program memory can only be erased by rows. To erase a row:

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 10-2](#).

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 10-4: FLASH PROGRAM MEMORY ERASE FLOWCHART



EXAMPLE 10-2: 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,CFGS      ; 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  AAh              ;
        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

10.2.4 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 10-5](#) (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 11 bits of PMADRH:PMADRL, (PMADRH<6:0>:PMADRL<7:4>) with the lower four bits of PMADRL, (PMADRL<3:0>) 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 10.2.2 "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 10.2.2 "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 10-3](#). The initial address is loaded into the PMADRH:PMADRL register pair; the data is loaded using indirect addressing.

FIGURE 10-5: BLOCK WRITES TO FLASH PROGRAM MEMORY WITH 32 WRITE LATCHES

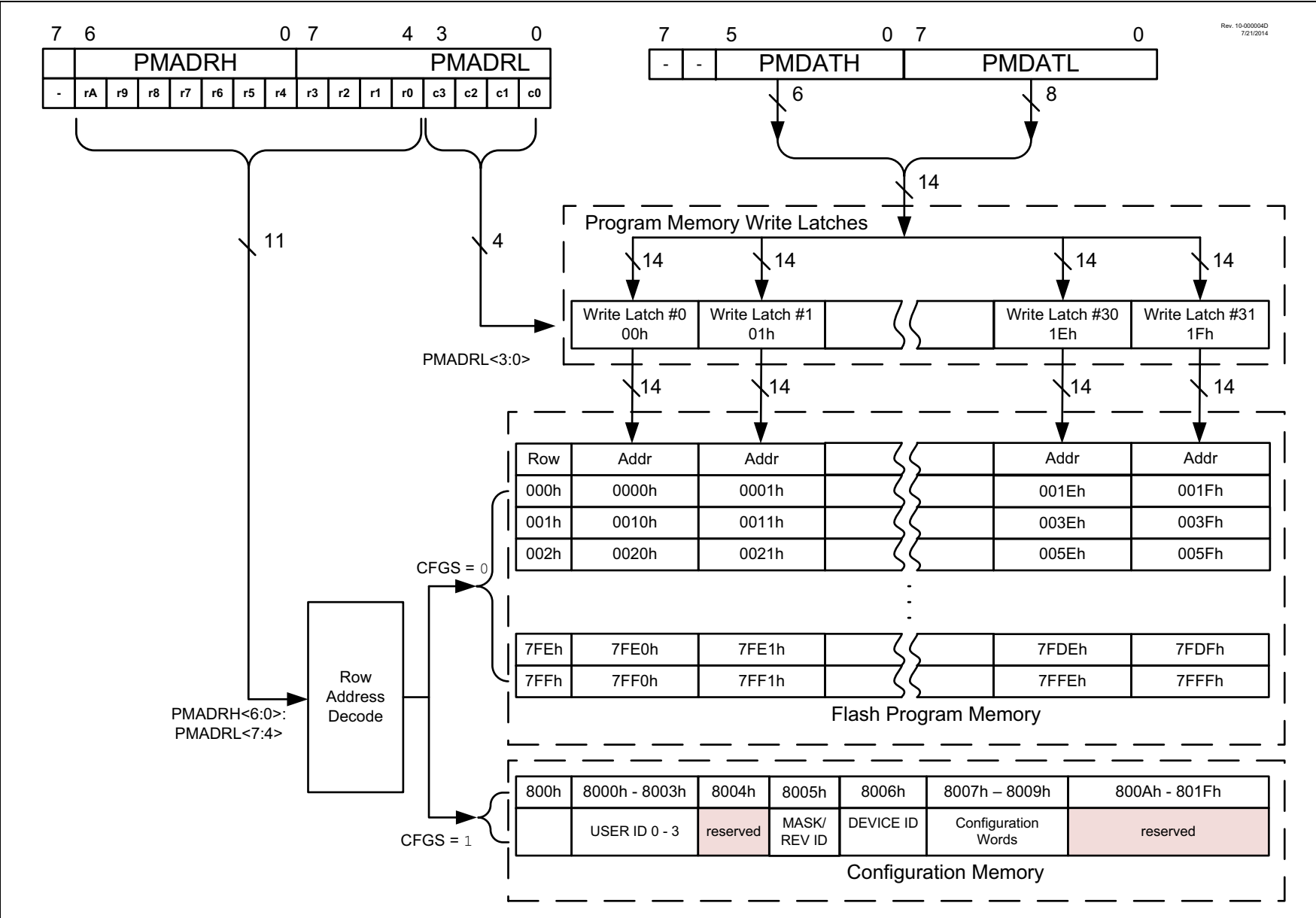
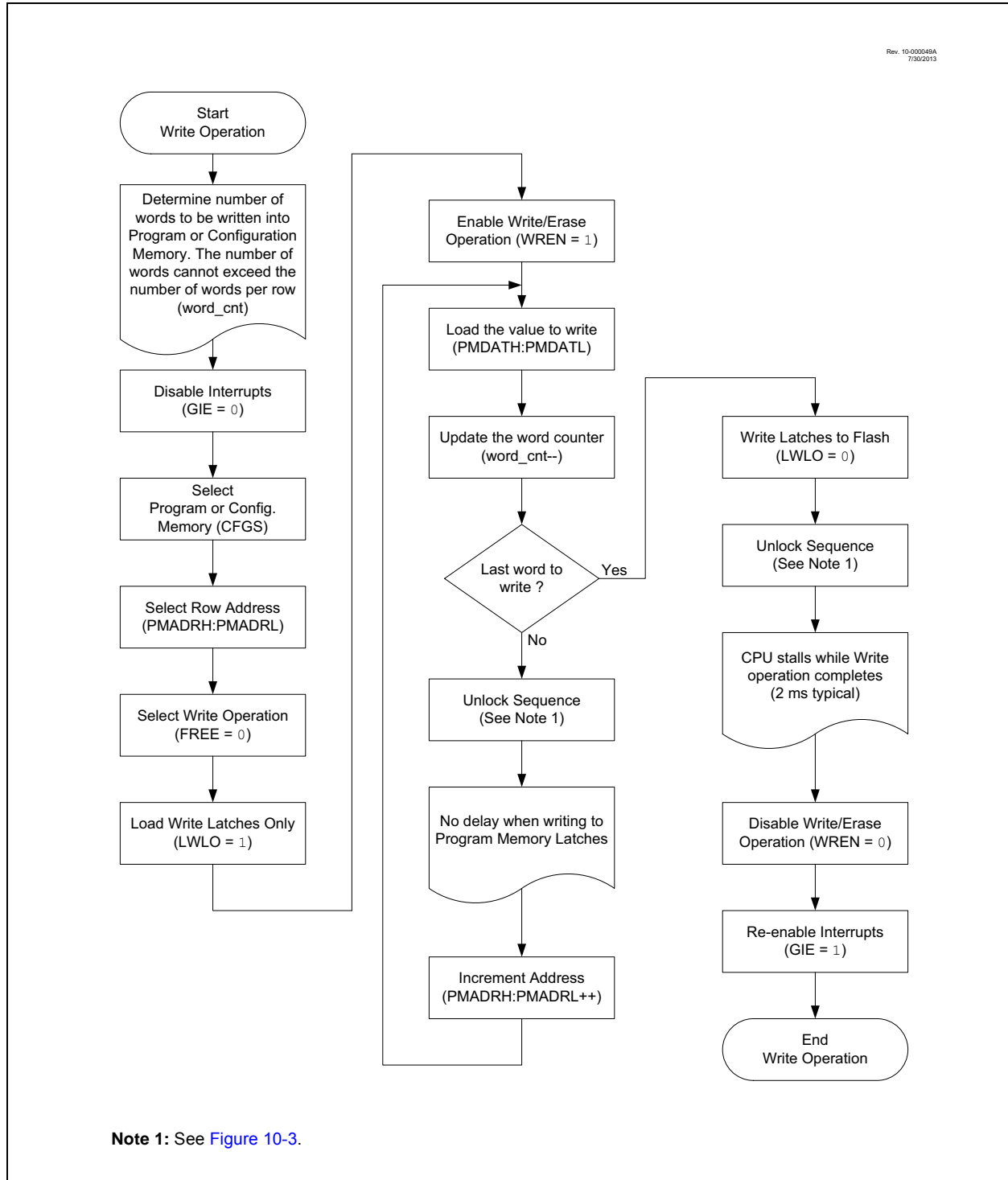


FIGURE 10-6: FLASH PROGRAM MEMORY WRITE FLOWCHART



EXAMPLE 10-3: WRITING TO FLASH PROGRAM MEMORY (32 WRITE LATCHES)

```

; 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    ADDRL,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,CFG5      ; 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   PMDATL          ;
    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   AAh              ;
    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   AAh              ;
    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

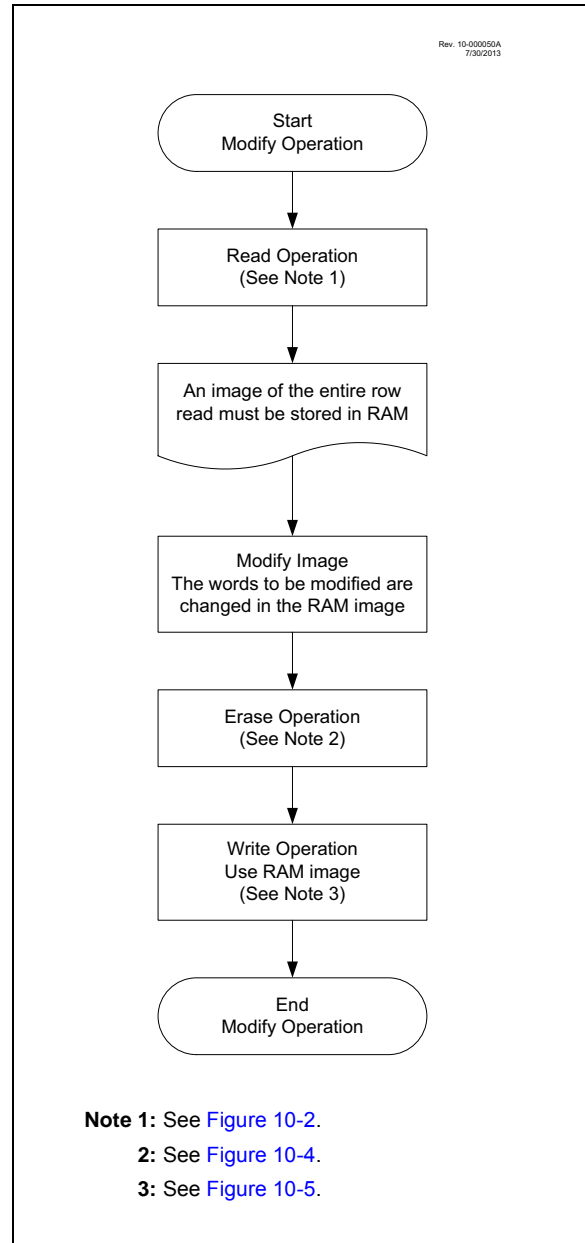
```

10.3 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 10-7: FLASH PROGRAM MEMORY MODIFY FLOWCHART



10.4 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 10-2](#).

When read access is initiated on an address outside the parameters listed in [Table 10-2](#), the PMDATH:PMDATL register pair is cleared, reading back '0's.

TABLE 10-2: USER ID, DEVICE ID AND CONFIGURATION WORD ACCESS ($CFG5 = 1$)

| Address | Function | Read Access | Write Access |
|-------------|---------------------------------|-------------|--------------|
| 8000h-8003h | User IDs | Yes | Yes |
| 8006h/8005h | Device ID/Revision ID | Yes | No |
| 8007h-8009h | Configuration Words 1, 2, and 3 | Yes | No |

EXAMPLE 10-4: 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
    NOP      ; Executed (See Figure 10-2)
    NOP      ; Ignored (See Figure 10-2)
    BSF      INTCON,GIE        ; Restore interrupts

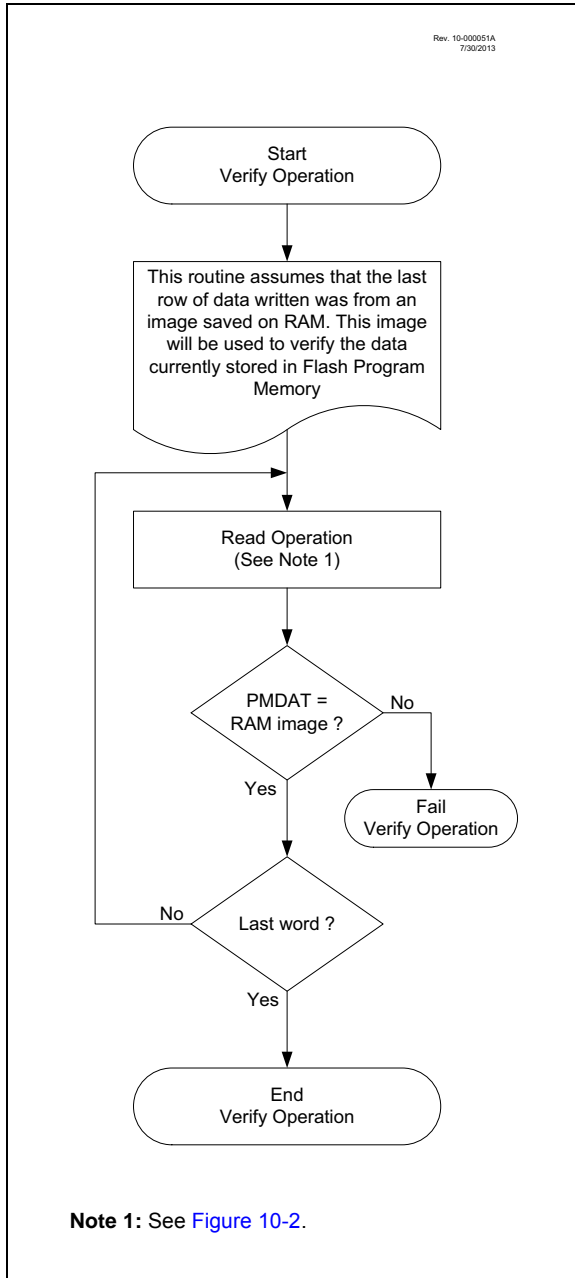
    MOVF     PMDATL,W          ; Get LSB of word
    MOVWF    PROG_DATA_LO     ; Store in user location
    MOVF     PMDATH,W          ; Get MSB of word
    MOVWF    PROG_DATA_HI     ; Store in user location

```

10.5 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 10-8: FLASH PROGRAM MEMORY VERIFY FLOWCHART



10.6 Register Definitions: Flash Program Memory Control

REGISTER 10-1: 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 10-2: 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 10-3: 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 10-4: 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 10-5: 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'.
 2: The WRERR bit is automatically set by hardware when a program memory write or erase operation is started (WR = 1).
 3: The LWLO bit is ignored during a program memory erase operation (FREE = 1).

REGISTER 10-6: 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 10-3: 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 |
|--------|-----------------------------------|-------------|-------------|-------|-------|--------|-------|-------|------------------|
| INTCON | GIE | PEIE | TMR0IE | INTE | IOCIE | TMR0IF | INTF | IOCIF | 97 |
| PMCON1 | —(1) | CFGS | LWLO | FREE | WRERR | WREN | WR | RD | 134 |
| PMCON2 | Program Memory Control Register 2 | | | | | | | | 135 |
| PMADRL | PMADRL<7:0> | | | | | | | | 133 |
| PMADRH | —(1) | PMADRH<6:0> | | | | | | | 133 |
| PMDATL | PMDATL<7:0> | | | | | | | | 133 |
| PMDATH | — | — | PMDATH<5:0> | | | | | 133 | |

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

Note 1: Unimplemented, read as '1'.

TABLE 10-4: 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> | | — | 69 |
| | 7:0 | CP | MCLRE | PWRTE | — | — | FOSC<2:0> | | | |
| CONFIG2 | 13:8 | — | — | LVP | DEBUG | LPBOR | BORV | STVREN | PLLEN | 68 |
| | 7:0 | ZCD | — | — | — | — | PPS1WAY | WRT<1:0> | | |
| CONFIG3 | 13:8 | — | — | WDTCCS<2:0> | | | WDTCWS<2:0> | | | 69 |
| | 7:0 | — | WDTE<1:0> | | WDTCPSC<4:0> | | | | | |

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

11.0 CYCLIC REDUNDANCY CHECK (CRC) MODULE

The Cyclic Redundancy Check (CRC) module provides a software-configurable hardware-implemented CRC checksum generator. This module includes the following features:

- Any standard CRC up to 16 bits can be used
- Configurable Polynomial
- Any seed value up to 16 bits can be used
- Standard and reversed bit order available
- Augmented zeros can be added automatically or by the user
- Memory scanner for fast CRC calculations on program memory user data
- Software loadable data registers for calculating CRC values not from the memory scanner

11.1 CRC Module Overview

The CRC module provides a means for calculating a check value of program memory. The CRC module is coupled with a memory scanner for faster CRC calculations. The memory scanner can automatically provide data to the CRC module. The CRC module can also be operated by directly writing data to SFRs, without using the scanner.

11.2 CRC Functional Overview

The CRC module can be used to detect bit errors in the Flash memory using the built-in memory scanner or through user input RAM. The CRC module can accept up to a 16-bit polynomial with up to a 16-bit seed value. A CRC calculated check value (or checksum) will then be generated into the CRCACC<15:0> registers for user storage. The CRC module uses an XOR shift register implementation to perform the polynomial division required for the CRC calculation.

EXAMPLE 11-1:

Rev. 10-000206A
1/8/2014

CRC-16-ANSI

$x^{16} + x^{15} + x^2 + 1$ (17 bits)

Standard 16-bit representation = 0x8005

CRCXORH = 0b10000000
CRCXORL = 0b00000010- (1)

Data Sequence:
0x55, 0x66, 0x77, 0x88

DLEN = 0b0111
PLEN = 0b1111

Data entered into the CRC:
SHIFTM = 0:
01010101 01100110 01110111 10001000

SHIFTM = 1:
10101010 01100110 11101110 00010001

Check Value (ACCM = 1):

SHIFTM = 0: 0x32D6
CRCACCH = 0b00110010
CRCACCL = 0b11010110

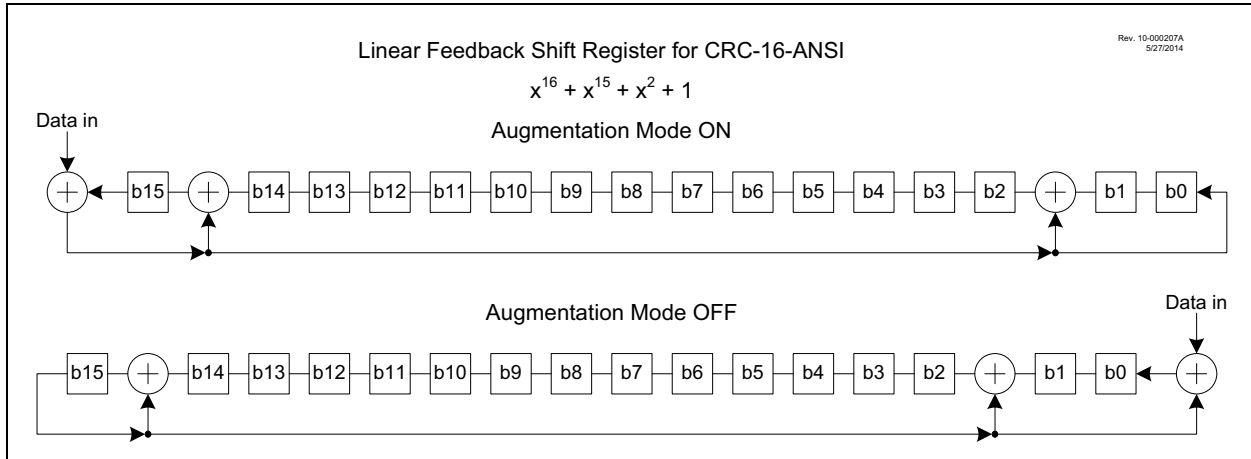
SHIFTM = 1: 0x6BA2
CRCACCH = 0b01101011
CRCACCL = 0b10100010

Note 1: Bit 0 is unimplemented. The LSb of any CRC polynomial is always '1' and will always be treated as a '1' by the CRC for calculating the CRC check value. This bit will be read in software as a '0'.

11.3 CRC Polynomial Implementation

Any polynomial can be used. The polynomial and accumulator sizes are determined by the PLEN<3:0> bits. For an n-bit accumulator, PLEN = n-1 and the corresponding polynomial is n+1 bits. Therefore, the accumulator can be any size up to 16 bits with a corresponding polynomial up to 17 bits. The MSb and LSb of the polynomial are always '1', which is forced by hardware. All polynomial bits between the MSb and LSb are specified by the CRCXOR registers. For example, when using CRC-16-ANSI, the polynomial is defined as $X^{16} + X^{15} + X^2 + 1$. The X^{16} and $X^0 = 1$ terms are the MSb and LSb controlled by hardware. The X^{15} and X^2 terms are specified by setting the corresponding CRCXOR<15:0> bits with the value of 0x8004. The actual value is 0x8005 because the hardware sets the LSb to 1. However, the LSb of the CRCXORL register is unimplemented and always reads as '0'. Please refer to [Example 11-1](#).

EXAMPLE 11-2: CRC LFSR EXAMPLE



11.4 CRC Data Sources

Data can be input to the CRC module in two ways:

- User data using the CRCDAT registers
- Flash using the Program Memory Scanner

To set the number of bits of data, up to 16 bits, the DLEN bits of CRCCON1 must be set accordingly. Only data bits in CRCDATA registers up to DLEN will be used, other data bits in CRCDATA registers will be ignored.

Data is moved into the CRCSHIFT as an intermediate to calculate the check value located in the CRCACC registers.

The SHIFTM bit is used to determine the bit order of the data being shifted into the accumulator. If SHIFTM is not set, the data will be shifted in MSb first. The value of DLEN will determine the MSb. If SHIFTM bit is set, the data will be shifted into the accumulator in reversed order, LSb first.

The CRC module can be seeded with an initial value by setting the CRCACC<15:0> registers to the appropriate value before beginning the CRC.

11.4.1 CRC FROM USER DATA

To use the CRC module on data input from the user, the user must write the data to the CRCDAT registers. The data from the CRCDAT registers will be latched into the shift registers on any write to the CRCDATL register.

11.4.2 CRC FROM FLASH

To use the CRC module on data located in Flash memory, the user can initialize the Program Memory Scanner as defined in [Section 11.8, Program Memory Scan Configuration](#).

11.5 CRC Check Value

The CRC check value will be located in the CRCACC registers after the CRC calculation has finished. The check value will depend on two mode settings of the CRCCON register: ACCM and SHIFTM. When the ACCM bit is set, the CRC module augments the data with a number of zeros equal to the length of the polynomial to align the final check value. When the ACCM bit is not set, the CRC will stop at the end of the data. A number of zeros equal to the length of the polynomial can then be entered into CRCDAT to find the same check value as Augmented mode. Alternatively, the expected check value can be entered at this point to make the final result equal 0.

When the CRC check value is computed with the SHIFTM bit set (selecting LSb first), and the ACCM bit is set, then the final value in the CRCACC registers will be reversed such that the LSb will be in the MSb position and vice versa. This is the expected check value in bit reversed form. If you are creating a check value to be appended to a data stream, a bit reversal must be performed on the final value to achieve the correct checksum. The CRC can be used to do this reversal by the following method:

- Save the CRCACC value in user RAM space
- Clear the CRCACC registers
- Clear the CRCXOR registers
- Write the saved CRCACC value to the CRCDAT input

The properly oriented check value will be in the CRCACC registers as the result.

11.6 CRC Interrupt

The CRC will generate an interrupt when the BUSY bit transitions from 1 to 0. The CRCIF interrupt flag bit of the PIR4 register is set every time the BUSY bit transitions, regardless of whether or not the CRC interrupt is enabled. The CRCIF bit can only be cleared in software. The CRC interrupt enable is the CRCIE bit of the PIE4 register.

11.7 Configuring the CRC

The following steps illustrate how to properly configure the CRC.

1. Determine if the automatic Program Memory Scan will be used with the Scanner or manual calculation through the SFR interface and perform the actions specified in [Section 11.4 “CRC Data Sources”](#), depending on which decision was made.
2. If desired, seed a starting CRC value into the CRCACCH/L registers.
3. Program the CRCXORH/L registers with the desired generator polynomial.
4. Program the DLEN<3:0> bits of the CRCCON1 register with the length of the data word - 1 (refer to Example 11-1). This determines how many times the shifter will shift into the accumulator for each data word.
5. Program the PLEN<3:0> bits of the CRCCON1 register with the length of the polynomial - 2 (refer to Example 11-1).
6. Determine whether shifting in trailing zeros is desired and set the ACCM bit of CRCCON0 register appropriately.
7. Likewise, determine whether the MSb or LSb should be shifted first and write the SHIFTM bit of CRCCON0 register appropriately.
8. Write the CRCGO bit of the CRCCON0 register to begin the shifting process.
- 9a. If manual SFR entry is used, monitor the FULL bit of CRCCON0 register. When FULL = 0, another word of data can be written to the CRCDATH/L registers, keeping in mind that CRCDATH should be written first if the data has >8 bits, as the shifter will begin upon the CRCDATL register being written.
- 9b. If the scanner is used, the scanner will automatically stuff words into the CRCDATH/L registers as needed, as long as the SCANGO bit is set.
- 10a. If using the Flash memory scanner, monitor the SCANIF (or the SCANGO bit) for the scanner to finish pushing information into the CRCDATA registers. After the scanner is completed, monitor the CRCIF (or the BUSY bit) to determine that the CRC has been completed and the check value can be read from the CRCACC registers. If both the interrupt flags are set (or both BUSY and SCANGO bits are cleared), the completed CRC calculation can be read from the CRCACCH/L registers.
- 10b. If manual entry is used, monitor the CRCIF (or BUSY bit) to determine when the CRCACC registers will hold the check value.

11.8 Program Memory Scan Configuration

If desired, the Program Memory Scan module may be used in conjunction with the CRC module to perform a CRC calculation over a range of program memory addresses. In order to set up the Scanner to work with the CRC, perform the following steps:

1. Set the EN bit to enable the module. This can be performed at any point preceding the setting of the SCANGO bit, but if it gets disabled, all internal states of the Scanner are reset (registers are unaffected).
2. Choose which memory access mode is to be used (see [Section 11.10 “Scanning Modes”](#)) and set the MODE bits of the SCANCON0 register appropriately.
3. Based on the memory access mode, set the INTM bits of the SCANCON0 register to the appropriate interrupt mode (see [Section 11.10.5 “Interrupt Interaction”](#)).
4. Set the SCANLADRL/H and SCANHADRL/H registers with the beginning and ending locations in memory that are to be scanned.
5. Begin the scan by setting the SCANGO bit in the SCANCON0 register. The scanner will wait (CRCGO must be set) for the signal from the CRC that it is ready for the first Flash memory location, then begin loading data into the CRC. It will continue to do so until it either hits the configured end address or an address that is unimplemented on the device, at which point the SCANGO bit will clear, Scanner functions will cease, and the SCANIF interrupt will be triggered. Alternately, the SCANGO bit can be cleared in software if desired.

11.9 Scanner Interrupt

The scanner will trigger an interrupt when the SCANGO bit transitions from 1 to 0. The SCANIF interrupt flag of PIR4 is set when the last memory location is reached and the data is entered into the CRCDATA registers. The SCANIF bit can only be cleared in software. The SCAN interrupt enable is the SCANIE bit of the PIE4 register.

11.10 Scanning Modes

The memory scanner can scan in four modes: Burst, Peek, Concurrent, and Triggered. These modes are controlled by the MODE bits of the SCANCON0 register. The four modes are summarized in [Table 11-1](#).

11.10.1 BURST MODE

When MODE = 01, the scanner is in Burst mode. In Burst mode, CPU operation is stalled beginning with the operation after the one that sets the SCANGO bit, and the scan begins, using the instruction clock to execute.

The CPU is held until the scan stops. Note that because the CPU is not executing instructions, the SCANGO bit cannot be cleared in software, so the CPU will remain stalled until one of the hardware end-conditions occurs. Burst mode has the highest throughput for the scanner, but has the cost of stalling other execution while it occurs.

11.10.2 CONCURRENT MODE

When MODE = 00, the scanner is in Concurrent mode. Concurrent mode, like Burst mode, stalls the CPU while performing accesses of memory. However, while Burst mode stalls until all accesses are complete, Concurrent mode allows the CPU to execute in between access cycles.

11.10.3 TRIGGERED MODE

When MODE = 11, the scanner is in Triggered mode. Triggered mode behaves identically to Concurrent mode, except instead of beginning the scan immedi-

ately upon the SCANGO bit being set, it waits for a rising edge from a separate trigger clock, the source of which is determined by the SCANTRIG register.

11.10.4 PEEK MODE

When MODE = 10, the scanner is in Peek mode. Peek mode waits for an instruction cycle in which the CPU does not need to access the NVM (such as a branch instruction) and uses that cycle to do its own NVM access. This results in the lowest throughput for the NVM access (and can take a much longer time to complete a scan than the other modes), but does so without any impact on execution times, unlike the other modes.

TABLE 11-1: SUMMARY OF SCANNER MODES

| MODE<1:0> | | Description | | |
|-----------|------------|---|---------------------------|---|
| | | First Scan Access | CPU Operation | |
| 11 | Triggered | As soon as possible following a trigger | Stalled during NVM access | CPU resumes execution following each access |
| 10 | Peek | At the first dead cycle | Timing is unaffected | CPU continues execution following each access |
| 01 | Burst | As soon as possible | Stalled during NVM access | CPU suspended until scan completes |
| 00 | Concurrent | | | CPU resumes execution following each access |

11.10.5 INTERRUPT INTERACTION

The INTM bit of the SCANCON0 register controls the scanner's response to interrupts depending on which mode the NVM scanner is in, as described in [Table 11-2](#).

TABLE 11-2: SCAN INTERRUPT MODES

| INTM | MODE<1:0> | |
|------|---|--|
| | MODE == Burst | MODE != Burst |
| 1 | Interrupt overrides SCANGO to pause the burst and the interrupt handler executes at full speed; Scanner Burst resumes when interrupt completes. | Scanner suspended during interrupt response; interrupt executes at full speed and scan resumes when the interrupt is complete. |
| 0 | Interrupts do not override SCANGO, and the scan (burst) operation will continue; interrupt response will be delayed until scan completes (latency will be increased). | Scanner accesses NVM during interrupt response. If MODE != Peak the interrupt handler execution speed will be affected. |

In general, if INTM = 0, the scanner will take precedence over the interrupt, resulting in decreased interrupt processing speed and/or increased interrupt

response latency. If INTM = 1, the interrupt will take precedence and have a better speed, delaying the memory scan.

11.10.6 WDT INTERACTION

Operation of the WDT is not affected by scanner activity. Hence, it is possible that long scans, particularly in Burst mode, may exceed the WDT time-out period and result in an undesired device Reset. This should be considered when performing memory scans with an application that also utilizes WDT.

11.10.7 IN-CIRCUIT DEBUG (ICD) INTERACTION

The scanner freezes when an ICD halt occurs, and remains frozen until user-mode operation resumes. The debugger may inspect the SCANCON0 and SCANLADR registers to determine the state of the scan.

The ICD interaction with each operating mode is summarized in [Table 11-3](#).

TABLE 11-3: ICD AND SCANNER INTERACTIONS

| ICD Halt | Scanner Operating Mode | | |
|------------------|--|---|--|
| | Peek | Concurrent Triggered | Burst |
| External Halt | If Scanner would peek an instruction that is not executed (because of ICD entry), the peek will occur after ICD exit, when the instruction executes. | If external halt is asserted during a scan cycle, the instruction (delayed by scan) may or may not execute before ICD entry, depending on external halt timing. | If external halt is asserted during the BSF (SCANCON.GO), ICD entry occurs, and the burst is delayed until ICD exit. Otherwise, the current NVM-access cycle will complete, and then the scanner will be interrupted for ICD entry. |
| | | If external halt is asserted during the cycle immediately prior to the scan cycle, both scan and instruction execution happen after the ICD exits. | If external halt is asserted during the burst, the burst is suspended and will resume with ICD exit. |
| PC Breakpoint | | Scan cycle occurs before ICD entry and instruction execution happens after the ICD exits. | If PCPB (or single step) is on BSF (SCANCON.GO), the ICD is entered before execution; execution of the burst will occur at ICD exit, and the burst will run to completion. |
| Data Breakpoint | | The instruction with the dataBP executes and ICD entry occurs immediately after. If scan is requested during that cycle, the scan cycle is postponed until the ICD exits. | |
| Single Step | | If a scan cycle is ready after the debug instruction is executed, the scan will read PFM and then the ICD is re-entered. | |
| SWBP and ICDINST | | | If scan would stall a SWBP, the scan cycle occurs and the ICD is entered. |

11.11 Register Definitions: CRC and Scanner Control

REGISTER 11-1: CRCCON0: CRC CONTROL REGISTER 0

| | | | | | | | |
|---------|---------|------|---------|-----|-----|---------|-------|
| R/W-0/0 | R/W-0/0 | R-0 | R/W-0/0 | U-0 | U-0 | R/W-0/0 | R-0 |
| EN | CRCGO | BUSY | ACCM | — | — | SHIFTM | FULL |
| 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 **EN:** CRC Enable bit
1 = CRC module is released from Reset
0 = CRC is disabled and consumes no operating current
- bit 6 **CRCGO:** CRC Start bit
1 = Start CRC serial shifter
0 = CRC serial shifter turned off
- bit 5 **BUSY:** CRC Busy bit
1 = Shifting in progress or pending
0 = All valid bits in shifter have been shifted into accumulator and EMPTY = 1
- bit 4 **ACCM:** Accumulator Mode bit
1 = Data is augmented with zeros
0 = Data is not augmented with zeros
- bit 3-2 **Unimplemented:** Read as '0'
- bit 1 **SHIFTM:** Shift Mode bit
1 = Shift right (LSb)
0 = Shift left (MSb)
- bit 0 **FULL:** Data Path Full Indicator bit
1 = CRCDATH/L registers are full
0 = CRCDATH/L registers have shifted their data into the shifter

REGISTER 11-2: CRCCON1: CRC 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 |
| DLEN<3:0> | | | | PLEN<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

- bit 7-4 **DLEN<3:0>:** Data Length bits
Denotes the length of the data word -1 (See Example 11-1)
- bit 3-0 **PLEN<3:0>:** Polynomial Length bits
Denotes the length of the polynomial -1 (See Example 11-1)

REGISTER 11-3: CRCDATH: CRC DATA HIGH BYTE REGISTER

| | | | | | | | |
|-----------|---------|---------|---------|---------|---------|---------|---------|
| R/W-x/x | R/W-x/x | R/W-x/x | R/W-x/x | R/W-x/x | R/W-x/x | R/W-x/x | R/W-x/x |
| DAT<15:8> | | | | | | | |
| 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 7-0 **DAT<15:8>**: CRC Input/Output Data bits

REGISTER 11-4: CRCDATL: CRC DATA LOW BYTE REGISTER

| | | | | | | | |
|----------|---------|---------|---------|---------|---------|---------|---------|
| R/W-x/x | R/W-x/x | R/W-x/x | R/W-x/x | R/W-x/x | R/W-x/x | R/W-x/x | R/W-x/x |
| DAT<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 **DAT<7:0>**: CRC Input/Output Data bits
Writing to this register fills the shifter.

REGISTER 11-5: CRCACCH: CRC ACCUMULATOR HIGH 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 |
| ACC<15:8> | | | | | | | |
| 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 7-0 **ACC<15:8>**: CRC Accumulator Register bits
Writing to this register writes to the CRC accumulator register. Reading from this register reads the CRC accumulator.

REGISTER 11-6: CRCACCL: CRC ACCUMULATOR 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 |
| ACC<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 **ACC<7:0>**: CRC Accumulator Register bits
Writing to this register writes to the CRC accumulator register through the CRC write bus. Reading from this register reads the CRC accumulator.

REGISTER 11-7: CRCSHIFTH: CRC SHIFT HIGH BYTE REGISTER

| | | | | | | | |
|-------------|-----|-----|-----|-----|-----|-----|-------|
| R-0 | R-0 | R-0 | R-0 | R-0 | R-0 | R-0 | R-0 |
| SHIFT<15:8> | | | | | | | |
| 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 7-0 **SHIFT<15:8>**: CRC Shifter Register bits
 Reading from this register reads the CRC Shifter.

REGISTER 11-8: CRCSHIFTL: CRC SHIFT LOW BYTE REGISTER

| | | | | | | | |
|------------|-----|-----|-----|-----|-----|-----|-------|
| R-0 | R-0 | R-0 | R-0 | R-0 | R-0 | R-0 | R-0 |
| SHIFT<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 **SHIFT<7:0>**: CRC Shifter Register bits
 Reading from this register reads the CRC Shifter.

REGISTER 11-9: CRCXORH: CRC XOR HIGH BYTE REGISTER

| | | | | | | | |
|-----------|-----|-----|-----|-----|-----|-----|-------|
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| XOR<15:8> | | | | | | | |
| 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 7-0 **XOR<15:8>**: XOR of Polynomial Term X_N Enable bits

REGISTER 11-10: CRCXORL: CRC XOR LOW BYTE REGISTER

| | | | | | | | |
|----------|---------|---------|---------|---------|---------|---------|-------|
| R/W-x/x | R/W-x/x | R/W-x/x | R/W-x/x | R/W-x/x | R/W-x/x | R/W-x/x | U-0 |
| XOR<7:1> | | | | | | | — |
| 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 | |

bit 7-1 **XOR<7:1>**: XOR of Polynomial Term X_N Enable bits
 bit 0 **Unimplemented**: Read as '0'

REGISTER 11-11: SCANCON0: SCANNER ACCESS CONTROL REGISTER 0

| | | | | | | | |
|-------------------|--------------------------|---------------------|---------|---------|-----|--------------------------|---------|
| R/W-0/0 | R/W/HC-0/0 | R-0 | R-0 | R/W-0/0 | U-0 | R/W-0/0 | R/W-0/0 |
| EN ⁽¹⁾ | SCANGO ^(2, 3) | BUSY ⁽⁴⁾ | INVALID | INTM | — | MODE<1:0> ⁽⁵⁾ | |
| 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 **EN:** Scanner Enable bit⁽¹⁾
 1 = Scanner is enabled
 0 = Scanner is disabled, internal states are reset
- bit 6 **SCANGO:** Scanner GO bit^(2, 3)
 1 = When the CRC sends a ready signal, NVM will be accessed according to MDx and data passed to the client peripheral.
 0 = Scanner operations will not occur
- bit 5 **BUSY:** Scanner Busy Indicator bit⁽⁴⁾
 1 = Scanner cycle is in process
 0 = Scanner cycle is complete (or never started)
- bit 4 **INVALID:** Scanner Abort signal bit
 1 = SCANLADRL/H has incremented or contains an invalid address⁽⁶⁾
 0 = SCANLADRL/H points to a valid address
- bit 3 **INTM:** NVM Scanner Interrupt Management Mode Select bit
If MODE = 10:
 This bit is ignored
If MODE = 01 (CPU is stalled until all data is transferred):
 1 = SCANGO is overridden (to zero) during interrupt operation; scanner resumes after returning from interrupt
 0 = SCANGO is not affected by interrupts, the interrupt response will be affected
If MODE = 00 or 11:
 1 = SCANGO is overridden (to zero) during interrupt operation; scan operations resume after returning from interrupt
 0 = Interrupts do not prevent NVM access
- bit 2 **Unimplemented:** Read as '0'
- bit 1-0 **MODE<1:0>:** Memory Access Mode bits⁽⁵⁾
 11 = Triggered mode
 10 = Peek mode
 01 = Burst mode
 00 = Concurrent mode

- Note 1:** Setting EN = 0 (SCANCON0 register) does not affect any other register content.
- 2:** This bit is cleared when LADR > HADR (and a data cycle is not occurring).
- 3:** If INTM = 1, this bit is overridden (to zero, but not cleared) during an interrupt response.
- 4:** BUSY = 1 when the NVM is being accessed, or when the CRC sends a ready signal.
- 5:** See [Table 11-1](#) for more detailed information.
- 6:** An invalid address happens when the entire range of the PFM is scanned and completed, i.e., device memory is 0x4000 and SCANHADR = 0x3FFF, after the last scan SCANLADR increments to 0x4000, the address is invalid.

REGISTER 11-12: SCANLADRH: SCAN LOW ADDRESS HIGH 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 |
| LADR<15:8> ^(1, 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 **LADR<15:8>**: Scan Start/Current Address bits^(1, 2)
 Most Significant bits of the current address to be fetched from, value increments on each fetch of memory.

- Note 1:** Registers SCANLADRH/L form a 16-bit value, but are not guarded for atomic or asynchronous access; registers should only be read or written while SCANGO = 0 (SCANCON0 register).
2: While SCANGO = 1 (SCANCON0 register), writing to this register is ignored.

REGISTER 11-13: SCANLADRL: SCAN LOW 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 |
| LADR<7:0> ^(1, 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 **LADR<7:0>**: Scan Start/Current Address bits^(1, 2)
 Least Significant bits of the current address to be fetched from, value increments on each fetch of memory

- Note 1:** Registers SCANLADRH/L form a 16-bit value, but are not guarded for atomic or asynchronous access; registers should only be read or written while SCANGO = 0 (SCANCON0 register).
2: While SCANGO = 1 (SCANCON0 register), writing to this register is ignored.

REGISTER 11-14: SCANHADR_H: SCAN HIGH ADDRESS HIGH 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 |
| HADR<15:8> ^(1, 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 **HADR<15:8>**: Scan End Address bits^(1, 2)
 Most Significant bits of the address at the end of the designated scan

- Note 1:** Registers SCANHADR_H/L form a 16-bit value, but are not guarded for atomic or asynchronous access; registers should only be read or written while SCANGO = 0 (SCANCON0 register).
- 2:** While SCANGO = 1 (SCANCON0 register), writing to this register is ignored.

REGISTER 11-15: SCANHADRL: SCAN HIGH 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 |
| HADR<7:0> ^(1, 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 **HADR<7:0>**: Scan End Address bits^(1, 2)
 Least Significant bits of the address at the end of the designated scan

- Note 1:** Registers SCANHADR_H/L form a 16-bit value, but are not guarded for atomic or asynchronous access; registers should only be read or written while SCANGO = 0 (SCANCON0 register).
- 2:** While SCANGO = 1 (SCANCON0 register), writing to this register is ignored.

REGISTER 11-16: SCANTRIG: SCAN TRIGGER SELECTION REGISTER

| | | | | | | | |
|-------|-----|-----|-----|-----------|-----|---------|---------|
| U-0 | U-0 | U-0 | U-0 | U-0 | U-0 | R/W-0/0 | R/W-0/0 |
| — | — | — | — | TSEL<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 | |

bit 7-4 **Unimplemented:** Read as '0'

bit 3-0 **TSEL<3:0>:** Scanner Data Trigger Input Selection bits
 1111-1010 = Reserved
 1001 = SMT2_Match
 1000 = SMT1_Match
 0111 = TMR0_Overflow
 0110 = TMR5_Overflow
 0101 = TMR3_Overflow
 0100 = TMR1_Overflow
 0011 = TMR6_postscaled
 0010 = TMR4_postscaled
 0001 = TMR2_postscaled
 0000 = LFINTOSC

TABLE 11-4: SUMMARY OF REGISTERS ASSOCIATED WITH CRC

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Register on Page |
|-----------|-------------|--------|-----------|-----------|-----------|-----------|-----------|--------|------------------|
| CRCACCH | ACC<15:8> | | | | | | | | 142 |
| CRCACCL | ACC<7:0> | | | | | | | | 142 |
| CRCCON0 | EN | CRCGO | BUSY | ACCM | — | — | SHIFTM | FULL | 141 |
| CRCCON1 | DLEN<3:0> | | | | PLEN<3:0> | | | | 141 |
| CRCDATH | DAT<15:8> | | | | | | | | 142 |
| CRCDATL | DAT<7:0> | | | | | | | | 142 |
| CRCSHIFTH | SHIFT<15:8> | | | | | | | | 143 |
| CRCSHIFTL | SHIFT<7:0> | | | | | | | | 143 |
| CRCXORH | XOR<15:8> | | | | | | | | 143 |
| CRCXORL | XOR<7:1> | | | | | | | | 143 |
| INTCON | GIE | PEIE | TMR0IE | INTE | IOIE | TMR0IF | INTF | IOCIF | 97 |
| PIR4 | SCANIF | CRCIF | SMT2PWAIF | SMT2PRAIF | SMT2IF | SMT1PWAIF | SMT1PRAIF | SMT1IF | 106 |
| PIE4 | SCANIE | CRCIE | SMT2PWAIE | SMT2PRAIE | SMT2IE | SMT1PWAIE | SMT1PRAIE | SMT1IE | 101 |
| SCANCON0 | EN | SCANGO | BUSY | INVALID | INTM | — | MODE<1:0> | | 144 |
| SCANHADRH | HADR<15:8> | | | | | | | | 146 |
| SCANHADRL | HADR<7:0> | | | | | | | | 146 |
| SCANLADRH | LADR<15:8> | | | | | | | | 145 |
| SCANLADRL | LADR<7:0> | | | | | | | | 145 |
| SCANTRIG | TSEL<3:0> | | | | | | | | 147 |

Legend: — = unimplemented location, read as '0'. Shaded cells are not used for the CRC module.

* Page provides register information.

12.0 I/O PORTS

Each port has six 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)
- INLVLx (input level control)
- ODCONx registers (open-drain)
- SLRCONx registers (slew rate)

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

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

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.

TABLE 12-1: PORT AVAILABILITY PER DEVICE

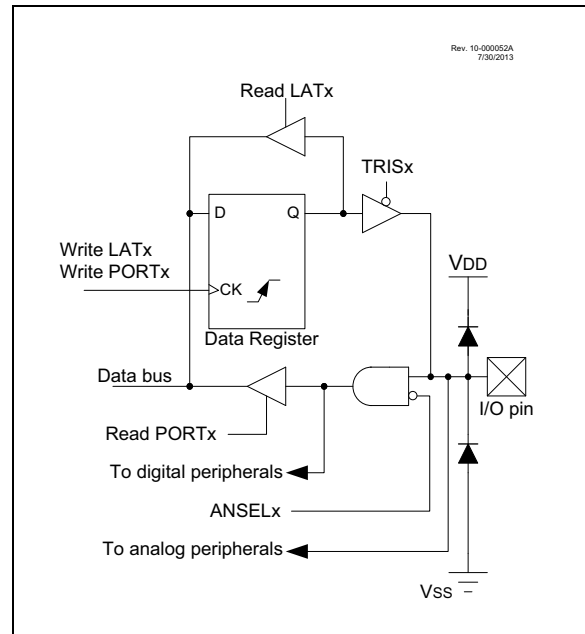
| Device | PORTA | PORTB | PORTC |
|---------------|-------|-------|-------|
| PIC16(L)F1619 | • | • | • |
| PIC16(L)F1615 | • | | • |

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

A write operation to the LATx register has the same effect as a write to the corresponding PORTx register. A read of the LATx register reads of the values held in the I/O PORT latches, while a read of the PORTx 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



12.1 PORTA Registers

12.1.1 DATA REGISTER

PORTA is a 6-bit wide, bidirectional port. The corresponding data direction register is TRISA (Register 12-2). 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). The exception is RA3, 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 PORTA register (Register 12-1) 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).

12.1.2 DIRECTION CONTROL

The TRISA register (Register 12-2) 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'.

12.1.3 OPEN-DRAIN CONTROL

The ODCONA register (Register 12-6) controls the open-drain feature of the port. Open-drain operation is independently selected for each pin. When an ODCONA bit is set, the corresponding port output becomes an open-drain driver capable of sinking current only. When an ODCONA bit is cleared, the corresponding port output pin is the standard push-pull drive capable of sourcing and sinking current.

12.1.4 SLEW RATE CONTROL

The SLRCONA register (Register 12-7) controls the slew rate option for each port pin. Slew rate control is independently selectable for each port pin. When an SLRCONA bit is set, the corresponding port pin drive is slew rate limited. When an SLRCONA bit is cleared, The corresponding port pin drive slews at the maximum rate possible.

12.1.5 INPUT THRESHOLD CONTROL

The INLVLA register (Register 12-8) controls the input voltage threshold for each of the available PORTA input pins. A selection between the Schmitt Trigger CMOS or the TTL Compatible thresholds is available. The input threshold is important in determining the value of a read of the PORTA register and also the level at which an interrupt-on-change occurs, if that feature is enabled. See 35.3 "DC Characteristics" for more information on threshold levels.

Note: Changing the input threshold selection should be performed while all peripheral modules are disabled. Changing the threshold level during the time a module is active may inadvertently generate a transition associated with an input pin, regardless of the actual voltage level on that pin.

12.1.6 ANALOG CONTROL

The ANSELA register (Register 12-4) 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

```
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
```

12.1.7 PORTA FUNCTIONS AND OUTPUT PRIORITIES

Each pin defaults to the PORT latch data after Reset. Other functions are selected with the peripheral pin select logic. See [Section 13.0 “Peripheral Pin Select \(PPS\) Module”](#) for more information. Analog input functions, such as ADC inputs, are not shown in the peripheral pin select lists. These inputs are active when the I/O pin is set for Analog mode using the ANSELA register. Digital output functions may continue to control the pin when in Analog mode.

12.2 Register Definitions: PORTA

REGISTER 12-1: PORTA: PORTA REGISTER

| | | | | | | | |
|-------|-----|---------|---------|-------|---------|---------|---------|
| U-0 | U-0 | R/W-x/x | R/W-x/x | R-x/x | R/W-x/x | R/W-x/x | R/W-x/x |
| — | — | 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 7-6 **Unimplemented:** Read as '0'
bit 5-0 **RA<5:0>:** 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 12-2: TRISA: PORTA TRI-STATE REGISTER

| | | | | | | | |
|-------|-----|---------|---------|------------------|---------|---------|---------|
| U-0 | U-0 | R/W-1/1 | R/W-1/1 | U-1 | R/W-1/1 | R/W-1/1 | R/W-1/1 |
| — | — | TRISA5 | TRISA4 | — ⁽¹⁾ | 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 7-6 **Unimplemented:** Read as '0'
bit 5-4 **TRISA<5:4>:** PORTA Tri-State Control bit
 1 = PORTA pin configured as an input (tri-stated)
 0 = PORTA pin configured as an output
bit 3 **Unimplemented:** Read as '1'
bit 2-0 **TRISA<2:0>:** PORTA Tri-State Control bit
 1 = PORTA pin configured as an input (tri-stated)
 0 = PORTA pin configured as an output

Note 1: Unimplemented, read as '1'.

REGISTER 12-3: LATA: PORTA DATA LATCH 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 |
| — | — | 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 7-6 **Unimplemented:** Read as '0'
bit 5-0 **LATA<5:0>:** RA<5:0> 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 12-4: ANSA: PORTA ANALOG SELECT REGISTER

| | | | | | | | |
|-------|-----|-----|---------|-----|---------|---------|---------|
| U-0 | U-0 | U-0 | R/W-1/1 | U-0 | R/W-1/1 | R/W-1/1 | R/W-1/1 |
| — | — | — | ANSA4 | — | 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 7-5 **Unimplemented:** Read as '0'
bit 4 **ANSA4:** Analog Select between Analog or Digital Function on Pins RA4, 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 3 **Unimplemented:** Read as '0'
bit 2-0 **ANSA<2:0>:** Analog Select between Analog or Digital Function on Pins RA<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 12-5: WPUA: WEAK PULL-UP PORTA 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 |
| — | — | WPUA5 | WPUA4 | WPUA3 | WPUA2 | WPUA1 | WPUA0 |
| 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 **WPUA<5:0>:** Weak Pull-up Register bits⁽³⁾
 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.
Note 3: For the WPUA3 bit, when MCLRE = 1, weak pull-up is internally enabled, but not reported here.

REGISTER 12-6: ODCONA: PORTA OPEN-DRAIN CONTROL REGISTER

| | | | | | | | |
|-------|-----|---------|---------|-----|---------|---------|---------|
| U-0 | U-0 | R/W-0/0 | R/W-0/0 | U-0 | R/W-0/0 | R/W-0/0 | R/W-0/0 |
| — | — | ODA5 | ODA4 | — | ODA2 | ODA1 | ODA0 |
| 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 5-4 **ODA<5:4>:** PORTA Open-Drain Enable bits
 For RA<5:4> pins, respectively
 1 = Port pin operates as open-drain drive (sink current only)
 0 = Port pin operates as standard push-pull drive (source and sink current)

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

bit 2-0 **ODA<2:0>:** PORTA Open-Drain Enable bits
 For RA<2:0> pins, respectively
 1 = Port pin operates as open-drain drive (sink current only)
 0 = Port pin operates as standard push-pull drive (source and sink current)

REGISTER 12-7: SLRCONA: PORTA SLEW RATE CONTROL REGISTER

| | | | | | | | |
|-------|-----|---------|---------|-----|---------|---------|---------|
| U-0 | U-0 | R/W-1/1 | R/W-1/1 | U-0 | R/W-1/1 | R/W-1/1 | R/W-1/1 |
| — | — | SLRA5 | SLRA4 | — | SLRA2 | SLRA1 | SLRA0 |
| 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 5-4 **SLRA<5:4>:** PORTA Slew Rate Enable bits
 For RA<5:4> pins, respectively
 1 = Port pin slew rate is limited
 0 = Port pin slews at maximum rate
- bit 3 **Unimplemented:** Read as '0'
- bit 2-0 **SLRA<2:0>:** PORTA Slew Rate Enable bits
 For RA<2:0> pins, respectively
 1 = Port pin slew rate is limited
 0 = Port pin slews at maximum rate

REGISTER 12-8: INLVLA: PORTA INPUT LEVEL CONTROL 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 |
| — | — | INLVLA5 | INLVLA4 | INLVLA3 | INLVLA2 | INLVLA1 | INLVLA0 |
| 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 **INLVLA<5:0>:** PORTA Input Level Select bits
 For RA<5:0> pins, respectively
 1 = ST input used for PORT reads and interrupt-on-change
 0 = TTL input used for PORT reads and interrupt-on-change

TABLE 12-2: 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 | — | — | — | ANSA4 | — | ANSA2 | ANSA1 | ANSA0 | 152 |
| INLVLA | — | — | INLVLA5 | INLVLA4 | INLVLA3 | INLVLA2 | INLVLA1 | INLVLA0 | 154 |
| LATA | — | — | LATA5 | LATA4 | LATA3 | LATA2 | LATA1 | LATA0 | 152 |
| ODCONA | — | — | ODA5 | ODA4 | — | ODA2 | ODA1 | ODA0 | 153 |
| OPTION_REG | WPUEN | INTEDG | TMR0CS | TMR0SE | PSA | PS<2:0> | | | 223 |
| PORTA | — | — | RA5 | RA4 | RA3 | RA2 | RA1 | RA0 | 151 |
| SLRCONA | — | — | SLRA5 | SLRA4 | — | SLRA2 | SLRA1 | SLRA0 | 154 |
| TRISA | — | — | TRISA5 | TRISA4 | — ⁽¹⁾ | TRISA2 | TRISA1 | TRISA0 | 151 |
| WPUA | — | — | WPUA5 | WPUA4 | WPUA3 | WPUA2 | WPUA1 | WPUA0 | 153 |

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

Note 1: Unimplemented, read as '1'.

TABLE 12-3: 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> | | — | 69 |
| | 7:0 | CP | MCLRE | PWRTE | — | — | FOSC<2:0> | | | |

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

12.3 PORTB Registers (PIC16(L)F1619 Only)

12.3.1 DATA REGISTER

PORTB is a 4-bit wide, bidirectional port. The corresponding data direction register is TRISB (Register 12-10). Setting a TRISB bit (= 1) will make the corresponding PORTB pin an input (i.e., disable the output driver). Clearing a TRISB bit (= 0) will make the corresponding PORTB 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 PORTB register (Register 12-9) 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).

12.3.2 DIRECTION CONTROL

The TRISB register (Register 12-10) 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'.

12.3.3 OPEN-DRAIN CONTROL

The ODCONB register (Register 12-14) controls the open-drain feature of the port. Open-drain operation is independently selected for each pin. When an ODCONB bit is set, the corresponding port output becomes an open-drain driver capable of sinking current only. When an ODCONB bit is cleared, the corresponding port output pin is the standard push-pull drive capable of sourcing and sinking current.

12.3.4 SLEW RATE CONTROL

The SLRCONB register (Register 12-15) controls the slew rate option for each port pin. Slew rate control is independently selectable for each port pin. When an SLRCONB bit is set, the corresponding port pin drive is slew rate limited. When an SLRCONB bit is cleared, The corresponding port pin drive slews at the maximum rate possible.

12.3.5 INPUT THRESHOLD CONTROL

The INLVLB register (Register 12-16) controls the input voltage threshold for each of the available PORTB input pins. A selection between the Schmitt Trigger CMOS or the TTL Compatible thresholds is available. The input threshold is important in determining the value of a read of the PORTB register and also the level at which an interrupt-on-change occurs, if that feature is enabled. See 35.3 "DC Characteristics" for more information on threshold levels.

Note: Changing the input threshold selection should be performed while all peripheral modules are disabled. Changing the threshold level during the time a module is active may inadvertently generate a transition associated with an input pin, regardless of the actual voltage level on that pin.

12.3.6 ANALOG CONTROL

The ANSELB register (Register 12-12) 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 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 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.

12.3.7 PORTB FUNCTIONS AND OUTPUT PRIORITIES

Each pin defaults to the PORT latch data after Reset. Other functions are selected with the peripheral pin select logic. See [Section 13.0 “Peripheral Pin Select \(PPS\) Module”](#) for more information. Analog input functions, such as ADC inputs, are not shown in the peripheral pin select lists. These inputs are active when the I/O pin is set for Analog mode using the ANSELB register. Digital output functions continue to may continue to control the pin when it is in Analog mode.

12.4 Register Definitions: PORTB

REGISTER 12-9: PORTB: PORTB REGISTER

| | | | | | | | |
|---------|---------|---------|---------|-----|-----|-----|-------|
| R/W-x/x | R/W-x/x | R/W-x/x | R/W-x/x | U-0 | U-0 | U-0 | U-0 |
| RB7 | RB6 | RB5 | RB4 | — | — | — | — |
| 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 **RB<7:4>**: PORTB I/O Value bits⁽¹⁾
1 = Port pin is \geq VIH
0 = Port pin is \leq VIL

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

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 12-10: TRISB: PORTB TRI-STATE REGISTER

| | | | | | | | |
|---------|---------|---------|---------|-----|-----|-----|-------|
| R/W-1/1 | R/W-1/1 | R/W-1/1 | R/W-1/1 | U-0 | U-0 | U-0 | U-0 |
| TRISB7 | TRISB6 | TRISB5 | TRISB4 | — | — | — | — |
| 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 **TRISB<7:4>**: PORTB Tri-State Control bits
1 = PORTB pin configured as an input (tri-stated)
0 = PORTB pin configured as an output

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

REGISTER 12-11: LATB: PORTB DATA LATCH REGISTER

| | | | | | | | |
|---------|---------|---------|---------|-----|-----|-----|-------|
| R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | U-0 | U-0 | U-0 | U-0 |
| LATB7 | LATB6 | LATB5 | LATB4 | — | — | — | — |
| 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 **LATB<7:4>**: RB<7:4> Output Latch Value bits⁽¹⁾

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

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 12-12: ANSELB: PORTB ANALOG SELECT REGISTER

| | | | | | | | |
|-------|-----|---------|---------|-----|-----|-----|-------|
| U-0 | U-0 | R/W-1/1 | R/W-1/1 | U-0 | U-0 | U-0 | U-0 |
| — | — | ANSB5 | ANSB4 | — | — | — | — |
| 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 5-4 **ANSB<5:4>**: Analog Select between Analog or Digital Function on Pins RB<5:4>, 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 3-0 **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.

REGISTER 12-13: WPUB: WEAK PULL-UP PORTB REGISTER^{(1),(2)}

| | | | | | | | |
|---------|---------|---------|---------|-----|-----|-----|-------|
| R/W-1/1 | R/W-1/1 | R/W-1/1 | R/W-1/1 | U-0 | U-0 | U-0 | U-0 |
| WPUB7 | WPUB6 | WPUB5 | WPUB4 | — | — | — | — |
| 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 **WPUB<7:4>**: Weak Pull-up Register bits
 1 = Pull-up enabled
 0 = Pull-up disabled

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

- 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.

REGISTER 12-14: ODCONB: PORTB OPEN-DRAIN CONTROL REGISTER

| | | | | | | | |
|---------|---------|---------|---------|-----|-----|-----|-------|
| R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | U-0 | U-0 | U-0 | U-0 |
| ODB7 | ODB6 | ODB5 | ODB4 | — | — | — | — |
| 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 **ODB<7:4>**: PORTB Open-Drain Enable bits
 For RB<7:4> pins, respectively
 1 = Port pin operates as open-drain drive (sink current only)
 0 = Port pin operates as standard push-pull drive (source and sink current)

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

REGISTER 12-15: SLRCONB: PORTB SLEW RATE CONTROL REGISTER

| | | | | | | | |
|---------|---------|---------|---------|-----|-----|-----|-------|
| R/W-1/1 | R/W-1/1 | R/W-1/1 | R/W-1/1 | U-0 | U-0 | U-0 | U-0 |
| SLRB7 | SLRB6 | SLRB5 | SLRB4 | — | — | — | — |
| 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 **SLRB<7:4>**: PORTA Slew Rate Enable bits
For RB<7:4> pins, respectively
1 = Port pin slew rate is limited
0 = Port pin slews at maximum rate

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

REGISTER 12-16: INLVLB: PORTB INPUT LEVEL CONTROL REGISTER

| | | | | | | | |
|---------|---------|---------|---------|-----|-----|-----|-------|
| R/W-1/1 | R/W-1/1 | R/W-1/1 | R/W-1/1 | U-0 | U-0 | U-0 | U-0 |
| INLVLB7 | INLVLB6 | INLVLB5 | INLVLB4 | — | — | — | — |
| 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 **INLVLB<7:4>**: PORTB Input Level Select bits
For RB<7:4> pins, respectively
1 = ST input used for PORT reads and interrupt-on-change
0 = TTL input used for PORT reads and interrupt-on-change

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

TABLE 12-4: 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 |
|------------|---------|---------|---------|---------|-------|---------|-------|-------|------------------|
| ANSELB | — | — | ANSB5 | ANSB4 | — | — | — | — | 159 |
| INLVLB | INLVLB7 | INLVLB6 | INLVLB5 | INLVLB4 | — | — | — | — | 161 |
| LATB | LATB7 | LATB6 | LATB5 | LATB4 | — | — | — | — | 159 |
| ODCONB | ODB7 | ODB6 | ODB5 | ODB4 | — | — | — | — | 160 |
| OPTION_REG | WPUEN | INTEDG | TMR0CS | TMR0SE | PSA | PS<2:0> | | | 223 |
| PORTB | RB7 | RB6 | RB5 | RB4 | — | — | — | — | 158 |
| SLRCONB | SLRB7 | SLRB6 | SLRB5 | SLRB4 | — | — | — | — | 161 |
| TRISB | TRISB7 | TRISB6 | TRISB5 | TRISB4 | — | — | — | — | 158 |
| WPUB | WPUB7 | WPUB6 | WPUB5 | WPUB4 | — | — | — | — | 160 |

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

Note 1: Unimplemented, read as '1'.

TABLE 12-5: SUMMARY OF CONFIGURATION WORD WITH PORTB

| 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> | — | — | 69 |
| | 7:0 | $\overline{\text{CP}}$ | MCLRE | $\overline{\text{PWRTE}}$ | WDTE<1:0> | | FOSC<2:0> | | — | |

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

12.5 PORTC Registers

12.5.1 DATA REGISTER

PORTC is an 8-bit wide, bidirectional port. The corresponding data direction register is TRISC (Register 12-18). Setting a TRISC bit (= 1) will make the corresponding PORTC pin an input (i.e., disable the output driver). 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 12-17) 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).

12.5.2 DIRECTION CONTROL

The TRISC register (Register 12-18) 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'.

12.5.3 OPEN-DRAIN CONTROL

The ODCONC register (Register 12-22) controls the open-drain feature of the port. Open-drain operation is independently selected for each pin. When an ODCONC bit is set, the corresponding port output becomes an open-drain driver capable of sinking current only. When an ODCONC bit is cleared, the corresponding port output pin is the standard push-pull drive capable of sourcing and sinking current.

12.5.4 SLEW RATE CONTROL

The SLRCONC register (Register 12-23) controls the slew rate option for each port pin. Slew rate control is independently selectable for each port pin. When an SLRCONC bit is set, the corresponding port pin drive is slew rate limited. When an SLRCONC bit is cleared, The corresponding port pin drive slews at the maximum rate possible.

12.5.5 INPUT THRESHOLD CONTROL

The INLVLC register (Register 12-24) controls the input voltage threshold for each of the available PORTC input pins. A selection between the Schmitt Trigger CMOS or the TTL Compatible thresholds is available. The input threshold is important in determining the value of a read of the PORTC register and also the level at which an interrupt-on-change occurs, if that feature is enabled. See 35.3 "DC Characteristics" for more information on threshold levels.

Note: Changing the input threshold selection should be performed while all peripheral modules are disabled. Changing the threshold level during the time a module is active may inadvertently generate a transition associated with an input pin, regardless of the actual voltage level on that pin.

12.5.6 ANALOG CONTROL

The ANSEL register (Register 12-20) 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.

12.5.7 HIGH DRIVE STRENGTH PINS

The HIDRVC register (Register 12-25) controls the high drive options on the RC4 and RC5. When a HIDRVC bit is cleared, the pin has normal drive strengths. When a HIDRVC bit is set, its respective pin can sink or source currents up to 100mA.

12.5.8 PORTC FUNCTIONS AND OUTPUT PRIORITIES

Each pin defaults to the PORT latch data after Reset. Other functions are selected with the peripheral pin select logic. See [Section 13.0 “Peripheral Pin Select \(PPS\) Module”](#) for more information. Analog input functions, such as ADC inputs, are not shown in the peripheral pin select lists. These inputs are active when the I/O pin is set for Analog mode using the ANSEL register. Digital output functions continue to may continue to control the pin when it is in Analog mode.

12.6 Register Definitions: PORTC

REGISTER 12-17: 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 ⁽¹⁾ | RC6 ⁽¹⁾ | 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 I/O Value bits^(1, 2)
 1 = Port pin is $\geq V_{IH}$
 0 = Port pin is $\leq V_{IL}$

- Note 1:** RC<7:6> on PIC16(L)F1619 only.
Note 2: Writes to PORTC are actually written to corresponding LATC register. Reads from PORTC register is return of actual I/O pin values.

REGISTER 12-18: 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 ⁽¹⁾ | TRISC6 ⁽¹⁾ | 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⁾
 1 = PORTC pin configured as an input (tri-stated)
 0 = PORTC pin configured as an output

- Note 1:** TRISC<7:6> on PIC16(L)F1619 only.

REGISTER 12-19: 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 ⁽¹⁾ | LATC6 ⁽¹⁾ | 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>**: RC<7:0> Output Latch Value bits⁽¹⁾
 1 = PORTC pin configured as an input (tri-stated)
 0 = PORTC pin configured as an output

- Note 1:** LATC<7:6> on PIC16(L)F1619 only.
2: Writes to PORTC are actually written to corresponding LATC register. Reads from PORTC register is return of actual I/O pin values.

REGISTER 12-20: ANSEL: PORTC ANALOG SELECT REGISTER

| | | | | | | | |
|----------------------|----------------------|-----|-----|---------|---------|---------|---------|
| R/W-1/1 | R/W-1/1 | U-0 | U-0 | R/W-1/1 | R/W-1/1 | R/W-1/1 | R/W-1/1 |
| ANSC7 ⁽¹⁾ | ANSC6 ⁽¹⁾ | — | — | ANSC3 | ANSC2 | ANSC1 | ANSC0 |
| 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 **ANSC<7:6>**: Analog Select between Analog or Digital Function on Pins RC<7:6>, respectively⁽¹⁾
 1 = Analog input. Pin is assigned as analog input⁽²⁾. Digital input buffer disabled.
 0 = Digital I/O. Pin is assigned to port or digital special function.
bit 5-4 **Unimplemented:** Read as '0'
bit 3-0 **ANSC<3:0>**: Analog Select between Analog or Digital Function on Pins RC<3:0>, respectively
 1 = Analog input. Pin is assigned as analog input⁽²⁾. Digital input buffer disabled.
 0 = Digital I/O. Pin is assigned to port or digital special function.

- Note 1:** ANSC<7:6> on PIC16(L)F1619 only.
2: 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 12-21: WPUC: WEAK PULL-UP PORTC REGISTER^{(2),(3)}

| | | | | | | | |
|----------------------|----------------------|---------|---------|---------|---------|---------|---------|
| 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 |
| WPUC7 ⁽¹⁾ | WPUC6 ⁽¹⁾ | WPUC5 | WPUC4 | WPUC3 | WPUC2 | WPUC1 | WPUC0 |
| 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 **WPUC<7:0>**: Weak Pull-up Register bits⁽¹⁾
 1 = Pull-up enabled
 0 = Pull-up disabled

- Note 1:** WPUC<7:6> on PIC16(L)F1619 only.
2: Global WPUEN bit of the OPTION_REG register must be cleared for individual pull-ups to be enabled.
3: The weak pull-up device is automatically disabled if the pin is configured as an output.

REGISTER 12-22: ODCONC: PORTC OPEN-DRAIN 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/W-0/0 |
| ODC7 ⁽¹⁾ | ODC6 ⁽¹⁾ | ODC5 | ODC4 | ODC3 | ODC2 | ODC1 | ODC0 |
| 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 **ODC<7:0>**: PORTC Open-Drain Enable bits⁽¹⁾
 For RC<7:0> pins, respectively
 1 = Port pin operates as open-drain drive (sink current only)
 0 = Port pin operates as standard push-pull drive (source and sink current)

- Note 1:** ODC<7:6> on PIC16(L)F1619 only.

REGISTER 12-23: SLRCONC: PORTC SLEW RATE CONTROL 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 |
| SLRC7 ⁽¹⁾ | SLRC6 ⁽¹⁾ | SLRC5 | SLRC4 | SLRC3 | SLRC2 | SLRC1 | SLRC0 |
| 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 **SLRC<7:0>**: PORTC Slew Rate Enable bits⁽¹⁾
 For RC<7:0> pins, respectively
 1 = Port pin slew rate is limited
 0 = Port pin slews at maximum rate

Note 1: SLRC<7:6> on PIC16(L)F1619 only.

REGISTER 12-24: INLVLC: PORTC INPUT LEVEL CONTROL 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 |
| INLVLC7 ⁽¹⁾ | INLVLC6 ⁽¹⁾ | INLVLC5 | INLVLC4 | INLVLC3 | INLVLC2 | INLVLC1 | INLVLC0 |
| 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 **INLVLC<7:0>**: PORTC Input Level Select bits⁽¹⁾
 For RC<7:0> pins, respectively
 1 = ST input used for PORT reads and interrupt-on-change
 0 = TTL input used for PORT reads and interrupt-on-change

Note 1: INLVLC<7:6> on PIC16(L)F1619 only.

REGISTER 12-25: HIDRVC: PORTC HIGH DRIVE STRENGTH REGISTER

| | | | | | | | |
|-------|-----|---------|---------|-----|-----|-----|-------|
| U-0 | U-0 | R/W-0/0 | R/W-0/0 | U-0 | U-0 | U-0 | U-0 |
| — | — | HIDC5 | HIDC4 | — | — | — | — |
| 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 | HIDC5: High Current Drive Enable on Port C5 1 = High Drive enabled 0 = High Drive disabled |
| bit 4 | HIDC4: High Current Drive Enable on Port C4 1 = High Drive enabled 0 = High Drive disabled |
| bit 3-0 | Unimplemented: Read as '0' |

TABLE 12-6: 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 |
|---------|------------------------|------------------------|---------|---------|---------|---------|---------|---------|------------------|
| ANSEL | ANSC7 ⁽¹⁾ | ANSC6 ⁽¹⁾ | — | — | ANSC3 | ANSC2 | ANSC1 | ANSC0 | 166 |
| HIDRVC | — | — | HIDC5 | HIDC4 | — | — | — | — | 169 |
| INLVLC | INLVLC7 ⁽¹⁾ | INLVLC6 ⁽¹⁾ | INLVLC5 | INLVLC4 | INLVLC3 | INLVLC2 | INLVLC1 | INLVLC0 | 168 |
| LATC | LATC7 ⁽¹⁾ | LATC6 ⁽¹⁾ | LATC5 | LATC4 | LATC3 | LATC2 | LATC1 | LATC0 | 166 |
| ODCONC | ODC7 ⁽¹⁾ | ODC6 ⁽¹⁾ | ODC5 | ODC4 | ODC3 | ODC2 | ODC1 | ODC0 | 167 |
| PORTC | RC7 ⁽¹⁾ | RC6 ⁽¹⁾ | RC5 | RC4 | RC3 | RC2 | RC1 | RC0 | 165 |
| SLRCONC | SLRC7 ⁽¹⁾ | SLRC6 ⁽¹⁾ | SLRC5 | SLRC4 | SLRC3 | SLRC2 | SLRC1 | SLRC0 | 168 |
| TRISC | TRISC7 ⁽¹⁾ | TRISC6 ⁽¹⁾ | TRISC5 | TRISC4 | TRISC3 | TRISC2 | TRISC1 | TRISC0 | 165 |
| WPUC | WPUC7 ⁽¹⁾ | WPUC6 ⁽¹⁾ | WPUC5 | WPUC4 | WPUC3 | WPUC2 | WPUC1 | WPUC0 | 167 |

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

Note 1: PIC16(L)F1619 only

13.0 PERIPHERAL PIN SELECT (PPS) MODULE

The Peripheral Pin Select (PPS) module connects peripheral inputs and outputs to the device I/O pins. Only digital signals are included in the selections. All analog inputs and outputs remain fixed to their assigned pins. Input and output selections are independent as shown in the simplified block diagram [Figure 13-1](#).

13.1 PPS Inputs

Each peripheral has a PPS register with which the inputs to the peripheral are selected. Inputs include the device pins.

Multiple peripherals can operate from the same source simultaneously. Port reads always return the pin level regardless of peripheral PPS selection. If a pin also has associated analog functions, the ANSEL bit for that pin must be cleared to enable the digital input buffer.

Although every peripheral has its own PPS input selection register, the selections are identical for every peripheral as shown in [Register 13-1](#).

Note: The notation “xxx” in the register name is a place holder for the peripheral identifier. For example, CLC1PPS.

13.2 PPS Outputs

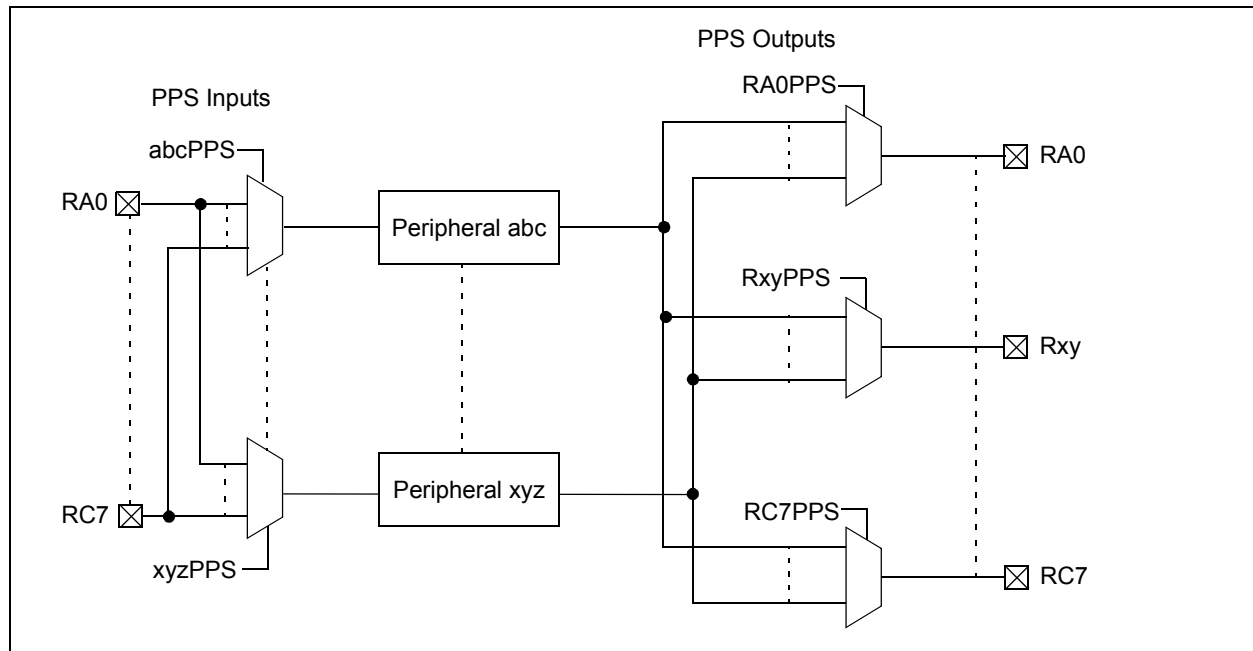
Each I/O pin has a PPS register with which the pin output source is selected. With few exceptions, the port TRIS control associated with that pin retains control over the pin output driver. Peripherals that control the pin output driver as part of the peripheral operation will override the TRIS control as needed. These peripherals include:

- EUSART (synchronous operation)
- MSSP (I²C)
- CWG (auto-shutdown)

Although every pin has its own PPS peripheral selection register, the selections are identical for every pin as shown in [Register 13-2](#).

Note: The notation “Rxy” is a place holder for the pin identifier. For example, RA0PPS.

FIGURE 13-1: SIMPLIFIED PPS BLOCK DIAGRAM



13.3 Bidirectional Pins

PPS selections for peripherals with bidirectional signals on a single pin must be made so that the PPS input and PPS output select the same pin. Peripherals that have bidirectional signals include:

- EUSART (synchronous operation)
- MSSP (I²C)

Note: The I²C default input pins are I²C and SMBus compatible and are the only pins on the device with this compatibility.

13.4 PPS Lock

The PPS includes a mode in which all input and output selections can be locked to prevent inadvertent changes. PPS selections are locked by setting the PPSLOCKED bit of the PPSLOCK register. Setting and clearing this bit requires a special sequence as an extra precaution against inadvertent changes. Examples of setting and clearing the PPSLOCKED bit are shown in [Example 13-1](#).

EXAMPLE 13-1: PPS LOCK/UNLOCK SEQUENCE

```

; suspend interrupts
    bcf    INTCON,GIE
;  BANKSEL PPSLOCK    ; set bank
;  required sequence, next 5 instructions
    movlw 0x55
    movwf PPSLOCK
    movlw 0xAA
    movwf PPSLOCK
;  Set PPSLOCKED bit to disable writes or
;  Clear PPSLOCKED bit to enable writes
    bsf    PPSLOCK,PPSLOCKED
;  restore interrupts
    bsf    INTCON,GIE
    
```

13.5 PPS Permanent Lock

The PPS can be permanently locked by setting the PPS1WAY Configuration bit. When this bit is set, the PPSLOCKED bit can only be cleared and set one time after a device Reset. This allows for clearing the PPSLOCKED bit so that the input and output selections can be made during initialization. When the PPSLOCKED bit is set after all selections have been made, it will remain set and cannot be cleared until after the next device Reset event.

13.6 Operation During Sleep

PPS input and output selections are unaffected by Sleep.

13.7 Effects of a Reset

A device Power-On-Reset (POR) clears all PPS input and output selections to their default values. All other Resets leave the selections unchanged. Default input selections are shown in [Table 13-1](#).

13.8 Register Definitions: PPS Input Selection

REGISTER 13-1: xxxPPS: PERIPHERAL xxx INPUT SELECTION

| | | | | | | | |
|-------|-----|-----|-------------|---------|---------|---------|---------|
| U-0 | U-0 | U-0 | R/W-q/u | R/W-q/u | R/W-q/u | R/W-q/u | R/W-q/u |
| — | — | — | xxxPPS<4:0> | | | | |
| 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 peripheral |

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

bit 4-3 **xxxPPS<4:3>:** Peripheral xxx Input PORT Selection bits
 11 = Reserved. Do not use.
 10 = Peripheral input is PORTC
 01 = Peripheral input is PORTB (PIC16(L)F1619 only)
 00 = Peripheral input is PORTA

bit 2-0 **xxxPPS<2:0>:** Peripheral xxx Input Bit Selection bits ⁽¹⁾
 111 = Peripheral input is from PORTx Bit 7 (Rx7)
 110 = Peripheral input is from PORTx Bit 6 (Rx6)
 101 = Peripheral input is from PORTx Bit 5 (Rx5)
 100 = Peripheral input is from PORTx Bit 4 (Rx4)
 011 = Peripheral input is from PORTx Bit 3 (Rx3)
 010 = Peripheral input is from PORTx Bit 2 (Rx2)
 001 = Peripheral input is from PORTx Bit 1 (Rx1)
 000 = Peripheral input is from PORTx Bit 0 (Rx0)

Note 1: See [Table 13-1](#) for Reset values.

REGISTER 13-2: RxyPPS: PIN Rxy OUTPUT SOURCE SELECTION REGISTER

| | | | | | | | |
|-------|-----|-----|-------------|---------|---------|---------|---------|
| U-0 | U-0 | U-0 | R/W-0/u | R/W-0/u | R/W-0/u | R/W-0/u | R/W-0/u |
| — | — | — | RxyPPS<4:0> | | | | |
| 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 7-5 **Unimplemented:** Read as '0'

bit 4-0 **RxyPPS<4:0>:** Pin Rxy Output Source Selection bits
 Selection code determines the output signal on the port pin.
 See [Table 13-2](#) for the selection codes

REGISTER 13-3: PPSLOCK: PPS LOCK REGISTER

| | | | | | | | |
|-------|-----|-----|-----|-----|-----|-----|-----------|
| U-0 | U-0 | U-0 | U-0 | U-0 | U-0 | U-0 | R/W-0/0 |
| — | — | — | — | — | — | — | PPSLOCKED |
| 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 **Unimplemented:** Read as '0'
- bit 0 **PPSLOCKED:** PPS Locked bit
 - 1 = PPS is locked. PPS selections can not be changed.
 - 0 = PPS is not locked. PPS selections can be changed.

TABLE 13-1: PPS INPUT REGISTER RESET VALUES

| Peripheral | xxxPPS Register | Default Pin Selection | | Reset Value (xxxPPS<4:0>) | |
|--------------------------------|-----------------|-----------------------|---------------|---------------------------|---------------|
| | | PIC16(L)F1619 | PIC16(L)F1615 | PIC16(L)F1619 | PIC16(L)F1615 |
| Interrupt on change | INTPPS | RA2 | RA2 | 00010 | 00010 |
| Timer 0 clock | T0CKIPPS | RA2 | RA2 | 00010 | 00010 |
| Timer 1 clock | T1CKIPPS | RA5 | RA5 | 00101 | 00101 |
| Timer 1 gate | T1GPPS | RA4 | RA4 | 00100 | 00100 |
| Timer 2 clock | T2CKIPPS | RA5 | RA5 | 0101 | 0101 |
| Timer 3 clock | T3CKIPPS | RC5 | RC5 | 10101 | 10101 |
| Timer 3 gate | T3GPPS | RC4 | RC4 | 10100 | 10100 |
| Timer 4 clock | T4CKIPPS | RC1 | RC1 | 10001 | 10001 |
| Timer 5 clock | T5CKIPPS | RC0 | RC0 | 10000 | 10000 |
| Timer 5 gate | T5GPPS | RC3 | RC3 | 10011 | 10011 |
| Timer 6 clock | T6CKIPPS | RA3 | RA3 | 00011 | 00011 |
| CCP1 | CCP1PPS | RC5 | RC5 | 10101 | 10101 |
| CCP2 | CCP2PPS | RC3 | RC3 | 10011 | 10011 |
| CWG1 | CWG1INPPS | RA2 | RA2 | 00010 | 00010 |
| SPI and I ² C clock | SSPCLKPPS | RB6 | RC0 | 01110 | 10000 |
| SPI and I ² C data | SSPDATPPS | RB4 | RC1 | 01100 | 10001 |
| SPI slave select | SSPSSPPS | RC6 | RC3 | 10110 | 10011 |
| EUSART RX | RXPPS | RB5 | RC5 | 01101 | 10101 |
| EUSART CK | CKPPS | RB7 | RC4 | 01111 | 10100 |
| All CLCs | CLCIN0PPS | RC3 | RC3 | 10011 | 10011 |
| All CLCs | CLCIN1PPS | RC4 | RC4 | 10100 | 10100 |
| All CLCs | CLCIN2PPS | RC1 | RC1 | 10001 | 10001 |
| All CLCs | CLCIN3PPS | RA5 | RA5 | 00101 | 00101 |
| SMT1 Window Input | SMTWIN1PPS | RA5 | RA5 | 00101 | 00101 |
| SMT1 Signal Input | SMTSIG1PPS | RA4 | RA4 | 00100 | 00100 |
| SMT2 Window Input | SMTWIN2PPS | RA3 | RA3 | 00101 | 00101 |
| SMT2 Signal Input | SMTSIG2PPS | RC1 | RC1 | 10001 | 10001 |
| Angular Timer 1 Clock Input | AT1INPPS | RC5 | RC5 | 10101 | 10101 |
| Angular Timer 1 CC1 Input | AT1CC1PPS | RC3 | RC3 | 10011 | 10011 |
| Angular Timer 1 CC2 Input | AT1CC2PPS | RC4 | RC4 | 10100 | 10100 |
| Angular Timer 1 CC3 Input | AT1CC3PPS | RC5 | RC5 | 10101 | 10101 |

Example: CCP1PPS = 0x13 selects RC3 as the CCP1 input.

TABLE 13-2: AVAILABLE PORTS FOR OUTPUT BY PERIPHERAL⁽²⁾

| RxyPPS<4:0> | Output Signal | PIC16(L)F1619 | | | PIC16(L)F1615 | |
|-------------|-------------------------|---------------|-------|-------|---------------|-------|
| | | PORTA | PORTB | PORTC | PORTA | PORTC |
| 11xxx | Reserved | • | • | • | • | • |
| 10111 | Reserved | • | • | • | • | • |
| 10110 | Reserved | • | • | • | • | • |
| 10101 | Reserved | • | • | • | • | • |
| 10100 | Reserved | • | • | • | • | • |
| 10011 | DT | • | • | • | • | • |
| 10010 | TX/CK | • | • | • | • | • |
| 10001 | SDO/SDA ⁽¹⁾ | • | • | • | • | • |
| 10000 | SCK/SCL ⁽¹⁾ | • | • | • | • | • |
| 01111 | PWM4_out | • | • | • | • | • |
| 01110 | PWM3_out | • | • | • | • | • |
| 01101 | CCP2_out | • | • | • | • | • |
| 01100 | CCP1_out | • | • | • | • | • |
| 01011 | CWG1OUTD ⁽¹⁾ | • | • | • | • | • |
| 01010 | CWG1OUTC ⁽¹⁾ | • | • | • | • | • |
| 01001 | CWG1OUTB ⁽¹⁾ | • | • | • | • | • |
| 01000 | CWG1OUTA ⁽¹⁾ | • | • | • | • | • |
| 00111 | LC4_out | • | • | • | • | • |
| 00110 | LC3_out | • | • | • | • | • |
| 00101 | LC2_out | • | • | • | • | • |
| 00100 | LC1_out | • | • | • | • | • |
| 00011 | ZCD1_out | • | • | • | • | • |
| 00010 | sync_C2OUT | • | • | • | • | • |
| 00001 | sync_C1OUT | • | • | • | • | • |
| 00000 | LATxy | • | • | • | • | • |

Note 1: TRIS control is overridden by the peripheral as required.

2: Unsupported peripherals will output a '0'.

TABLE 13-3: SUMMARY OF REGISTERS ASSOCIATED WITH THE PPS MODULE

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Register on page |
|-----------------------|-------|-------|-------|-----------------|-------|-------|-------|-----------|------------------|
| PPSLOCK | — | — | — | — | — | — | — | PPSLOCKED | 173 |
| INTPPS | — | — | — | INTPPS<4:0> | | | | | 172 |
| T0CKIPPS | — | — | — | T0CKIPPS<4:0> | | | | | 172 |
| T1CKIPPS | — | — | — | T1CKIPPS<4:0> | | | | | 172 |
| T1GPPS | — | — | — | T1GPPS<4:0> | | | | | 172 |
| T2CKIPPS | — | — | — | T2CKIPPS<4:0> | | | | | 172 |
| T3CKIPPS | — | — | — | T3CKIPPS<4:0> | | | | | 172 |
| T3GPPS | — | — | — | T3GPPS<4:0> | | | | | 172 |
| T4CKIPPS | — | — | — | T4CKIPPS<4:0> | | | | | 172 |
| T5CKIPPS | — | — | — | T5CKIPPS<4:0> | | | | | 172 |
| T5GPPS | — | — | — | T5GPPS<4:0> | | | | | 172 |
| T6CKIPPS | — | — | — | T6CKIPPS<4:0> | | | | | 172 |
| CCP1PPS | — | — | — | CCP1PPS<4:0> | | | | | 172 |
| CCP2PPS | — | — | — | CCP2PPS<4:0> | | | | | 172 |
| CWG1INPPS | — | — | — | CWG1INPPS<4:0> | | | | | 172 |
| SSPCLKPPS | — | — | — | SSPCLKPPS<4:0> | | | | | 172 |
| SSPDATPPS | — | — | — | SSPDATPPS<4:0> | | | | | 172 |
| SSPSSPPS | — | — | — | SSPSSPPS<4:0> | | | | | 172 |
| RXPPS | — | — | — | RXPPS<4:0> | | | | | 172 |
| CKPPS | — | — | — | CKPPS<4:0> | | | | | 172 |
| CLCIN0PPS | — | — | — | CLCIN0PPS<4:0> | | | | | 172 |
| CLCIN1PPS | — | — | — | CLCIN1PPS<4:0> | | | | | 172 |
| CLCIN2PPS | — | — | — | CLCIN2PPS<4:0> | | | | | 172 |
| CLCIN3PPS | — | — | — | CLCIN3PPS<4:0> | | | | | 172 |
| AT1INPPS | — | — | — | AT1INPPS<4:0> | | | | | 172 |
| ATCC1PPS | — | — | — | ATCC1PPS<4:0> | | | | | 172 |
| ATCC2PPS | — | — | — | ATCC2PPS<4:0> | | | | | 172 |
| ATCC3PPS | — | — | — | ATCC3PPS<4:0> | | | | | 172 |
| SMT1SIGPPS | — | — | — | SMT1SIGPPS<4:0> | | | | | 172 |
| SMT1WINPPS | — | — | — | SMT1WINPPS<4:0> | | | | | 172 |
| SMT2SIGPPS | — | — | — | SMT2SIGPPS<4:0> | | | | | 172 |
| SMT2WINPPS | — | — | — | SMT2WINPPS<4:0> | | | | | 172 |
| RA0PPS | — | — | — | RA0PPS<4:0> | | | | | 172 |
| RA1PPS | — | — | — | RA1PPS<4:0> | | | | | 172 |
| RA2PPS | — | — | — | RA2PPS<4:0> | | | | | 172 |
| RA4PPS | — | — | — | RA4PPS<4:0> | | | | | 172 |
| RA5PPS | — | — | — | RA5PPS<4:0> | | | | | 172 |
| RB4PPS ⁽¹⁾ | — | — | — | RB4PPS<4:0> | | | | | 172 |
| RB5PPS ⁽¹⁾ | — | — | — | RB5PPS<4:0> | | | | | 172 |
| RB6PPS ⁽¹⁾ | — | — | — | RB6PPS<4:0> | | | | | 172 |
| RB7PPS ⁽¹⁾ | — | — | — | RB7PPS<4:0> | | | | | 172 |
| RC0PPS | — | — | — | RC0PPS<4:0> | | | | | 172 |

Note 1: PIC16(L)F1619 only.

TABLE 13-3: SUMMARY OF REGISTERS ASSOCIATED WITH THE PPS MODULE (CONTINUED)

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Register on page |
|-----------------------|-------|-------|-------|-------|-------|-------------|-------|-------|---------------------|
| RC1PPS | — | — | — | | | RC1PPS<4:0> | | | 172 |
| RC2PPS | — | — | — | | | RC2PPS<4:0> | | | 172 |
| RC3PPS | — | — | — | | | RC3PPS<4:0> | | | 172 |
| RC4PPS | — | — | — | | | RC4PPS<4:0> | | | 172 |
| RC5PPS | — | — | — | | | RC5PPS<4:0> | | | 172 |
| RC6PPS ⁽¹⁾ | — | — | — | | | RC6PPS<4:0> | | | 172 |
| RC7PPS ⁽¹⁾ | — | — | — | | | RC7PPS<4:0> | | | 172 |

Note 1: PIC16(L)F1619 only.

14.0 INTERRUPT-ON-CHANGE

The PORTA, PORTB⁽¹⁾ and PORTC 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 port pin, or combination of port 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 14-1 is a block diagram of the IOC module.

Note 1: PORTB available on PIC16(L)F1619 only.

14.1 Enabling the Module

To allow individual port 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.

14.2 Individual Pin Configuration

For each port pin, a rising edge detector and a falling edge detector are present. To enable a pin to detect a rising edge, the associated bit of the IOCxP register is set. To enable a pin to detect a falling edge, the associated bit of the IOCxN register is set.

A pin can be configured to detect rising and falling edges simultaneously by setting both associated bits of the IOCxP and IOCxN registers, respectively.

14.3 Interrupt Flags

The IOCAF_x, IOCBF_x and IOCCF_x bits located in the IOCAF, IOCBF and IOCCF registers, respectively, are status flags that correspond to the interrupt-on-change pins of the associated port. 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 IOCAF_x, IOCBF_x and IOCCF_x bits.

14.4 Clearing Interrupt Flags

The individual status flags, (IOCAF_x, IOCBF_x and IOCCF_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 14-1: CLEARING INTERRUPT FLAGS (PORTA EXAMPLE)

```
MOVLW    0xff
XORWF    IOCAF, W
ANDWF    IOCAF, F
```

14.5 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 IOCxF register will be updated prior to the first instruction executed out of Sleep.

FIGURE 14-1: INTERRUPT-ON-CHANGE BLOCK DIAGRAM (PORTA EXAMPLE)



14.6 Register Definitions: Interrupt-on-Change Control

REGISTER 14-1: IOCAP: INTERRUPT-ON-CHANGE PORTA POSITIVE EDGE 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 |
|-------|-----|---------|---------|---------|---------|---------|---------|
| — | — | IOCAP5 | IOCAP4 | IOCAP3 | IOCAP2 | IOCAP1 | IOCAP0 |
| 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 **IOCAP<5:0>:** Interrupt-on-Change PORTA Positive Edge Enable bits

- 1 = Interrupt-on-Change enabled on the pin for a positive going edge. IOCAF_x bit and IOCIF flag will be set upon detecting an edge.
- 0 = Interrupt-on-Change disabled for the associated pin.

REGISTER 14-2: IOCAN: INTERRUPT-ON-CHANGE PORTA NEGATIVE EDGE 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 |
|-------|-----|---------|---------|---------|---------|---------|---------|
| — | — | IOCAN5 | IOCAN4 | IOCAN3 | IOCAN2 | IOCAN1 | IOCAN0 |
| 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 **IOCAN<5:0>:** Interrupt-on-Change PORTA Negative Edge Enable bits

- 1 = Interrupt-on-Change enabled on the pin for a negative going edge. IOCAF_x bit and IOCIF flag will be set upon detecting an edge.
- 0 = Interrupt-on-Change disabled for the associated pin.

REGISTER 14-3: IOCAF: INTERRUPT-ON-CHANGE PORTA FLAG REGISTER

| U-0 | U-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 |
|-------|-----|------------|------------|------------|------------|------------|------------|
| — | — | IOCAF5 | IOCAF4 | IOCAF3 | IOCAF2 | IOCAF1 | IOCAF0 |
| 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 7-6 **Unimplemented:** Read as '0'

bit 5-0 **IOCAF<5:0>:** Interrupt-on-Change PORTA Flag bits

- 1 = An enabled change was detected on the associated pin.
Set when IOCAP_x = 1 and a rising edge was detected on RAX, or when IOCAN_x = 1 and a falling edge was detected on RAX.
- 0 = No change was detected, or the user cleared the detected change.

REGISTER 14-4: IOCBP: INTERRUPT-ON-CHANGE PORTB POSITIVE EDGE REGISTER⁽¹⁾

| R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | U-0 | U-0 | U-0 | U-0 |
|---------|---------|---------|---------|-----|-----|-----|-------|
| IOCBP7 | IOCBP6 | IOCBP5 | IOCBP4 | — | — | — | — |
| 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 **IOCBP<7:4>**: 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.

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

Note 1: PIC16(L)F1619 only.

REGISTER 14-5: IOCBN: INTERRUPT-ON-CHANGE PORTB NEGATIVE EDGE REGISTER⁽¹⁾

| R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | U-0 | U-0 | U-0 | U-0 |
|---------|---------|---------|---------|-----|-----|-----|-------|
| IOCBN7 | IOCBN6 | IOCBN5 | IOCBN4 | — | — | — | — |
| 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 **IOCBN<7:4>**: 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.

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

Note 1: PIC16(L)F1619 only.

REGISTER 14-6: IOCBF: INTERRUPT-ON-CHANGE PORTB FLAG REGISTER⁽¹⁾

| R/W/HS-0/0 | R/W/HS-0/0 | R/W/HS-0/0 | R/W/HS-0/0 | U-0 | U-0 | U-0 | U-0 |
|------------|------------|------------|------------|-----|-----|-----|-------|
| IOCBF7 | IOCBF6 | IOCBF5 | IOCBF4 | — | — | — | — |
| 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 7-4 **IOCBF<7:4>**: 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.

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

Note 1: PIC16(L)F1619 only.

REGISTER 14-7: IOCCP: INTERRUPT-ON-CHANGE PORTC 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 |
|-----------------------|-----------------------|---------|---------|---------|---------|---------|---------|
| IOCCP7 ⁽¹⁾ | IOCCP6 ⁽¹⁾ | IOCCP5 | IOCCP4 | IOCCP3 | IOCCP2 | IOCCP1 | IOCCP0 |
| 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 **IOCCP<7:0>**: Interrupt-on-Change PORTC Positive Edge Enable bits(1)
1 = Interrupt-on-Change enabled on the pin for a positive going edge. IOCCFx bit and IOCIF flag will be set upon detecting an edge.
0 = Interrupt-on-Change disabled for the associated pin.

Note 1: IOCCP<7:6> available on PIC16(L)F1619 only.

REGISTER 14-8: IOCCN: INTERRUPT-ON-CHANGE PORTC NEGATIVE EDGE 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 |
|-----------------------|-----------------------|---------|---------|---------|---------|---------|---------|
| IOCCN7 ⁽¹⁾ | IOCCN6 ⁽¹⁾ | IOCCN5 | IOCCN4 | IOCCN3 | IOCCN2 | IOCCN1 | IOCCN0 |
| 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 **IOCCN<7:0>**: Interrupt-on-Change PORTC Negative Edge Enable bits(1)
1 = Interrupt-on-Change enabled on the pin for a negative going edge. IOCCFx bit and IOCIF flag will be set upon detecting an edge.
0 = Interrupt-on-Change disabled for the associated pin.

Note 1: IOCCN<7:6> available on PIC16(L)F1619 only.

REGISTER 14-9: IOCCF: INTERRUPT-ON-CHANGE PORTC FLAG REGISTER⁽¹⁾

| 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 |
|----------------------|-----------------------|------------|------------|------------|------------|------------|------------|
| IOCCF ⁽¹⁾ | IOCCF6 ⁽¹⁾ | IOCCF5 | IOCCF4 | IOCCF3 | IOCCF2 | IOCCF1 | IOCCF0 |
| 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 **IOCCF<7:0>**: Interrupt-on-Change PORTC Flag bits(1)
1 = An enabled change was detected on the associated pin.
Set when IOCCPx = 1 and a rising edge was detected on RCx, or when IOCCNx = 1 and a falling edge was detected on RCx.
0 = No change was detected, or the user cleared the detected change.

Note 1: IOCCF<7:6> available on PIC16(L)F1619 only.

TABLE 14-1: 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 |
|----------------------|-----------------------|-----------------------|--------|--------|------------------|--------|--------|--------|------------------|
| ANSELA | — | — | — | ANSA4 | — | ANSA2 | ANSA1 | ANSA0 | 152 |
| INTCON | GIE | PEIE | TMR0IE | INTE | IOCIE | TMR0IF | INTF | IOCIF | 97 |
| IOCAF | — | — | IOCAF5 | IOCAF4 | IOCAF3 | IOCAF2 | IOCAF1 | IOCAF0 | 180 |
| IOCAN | — | — | IOCAN5 | IOCAN4 | IOCAN3 | IOCAN2 | IOCAN1 | IOCAN0 | 180 |
| IOCAP | — | — | IOCAP5 | IOCAP4 | IOCAP3 | IOCAP2 | IOCAP1 | IOCAP0 | 180 |
| IOCBF ⁽²⁾ | IOCBF7 | IOCBF6 | IOCBF5 | IOCBF4 | — | — | — | — | 181 |
| IOCBN ⁽²⁾ | IOCBN7 | IOCBN6 | IOCBN5 | IOCBN4 | — | — | — | — | 181 |
| IOCBP ⁽²⁾ | IOCBP7 | IOCBP6 | IOCBP5 | IOCBP4 | — | — | — | — | 181 |
| IOCCF | IOCCF7 ⁽²⁾ | IOCCF6 ⁽²⁾ | IOCCF5 | IOCCF4 | IOCCF3 | IOCCF2 | IOCCF1 | IOCCF0 | 182 |
| IOCCN | IOCCN7 ⁽²⁾ | IOCCN6 ⁽²⁾ | IOCCN5 | IOCCN4 | IOCCN3 | IOCCN2 | IOCCN1 | IOCCN0 | 182 |
| IOCCP | IOCCP7 ⁽²⁾ | IOCCP6 ⁽²⁾ | IOCCP5 | IOCCP4 | IOCCP3 | IOCCP2 | IOCCP1 | IOCCP0 | 182 |
| TRISA | — | — | TRISA5 | TRISA4 | — ⁽¹⁾ | TRISA2 | TRISA1 | TRISA0 | 151 |
| TRISC | TRISC7 ⁽²⁾ | TRISC7 ⁽²⁾ | TRISC5 | TRISC4 | TRISC3 | TRISC2 | TRISC1 | TRISC0 | 165 |

Legend: — = unimplemented location, read as '0'. Shaded cells are not used by interrupt-on-change.

Note 1: Unimplemented, read as '1'.

2: PIC16(L)F1619 only.

15.0 FIXED VOLTAGE REFERENCE (FVR)

The Fixed Voltage Reference (FVR) is a stable voltage reference, independent of V_{DD} , with a nominal output level (V_{FVR}) of 1.024V. The output of the FVR can be configured to supply a reference voltage to the following:

- ADC input channel
- Comparator positive input
- Comparator negative input

The FVR can be enabled by setting the FVREN bit of the FVRCON register.

15.1 Independent Gain Amplifier

The output of the FVR supplied to the peripherals, (listed above), is routed through a programmable gain amplifier. Each amplifier can be programmed for a gain of 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 17.0 “Analog-to-Digital Converter \(ADC\) Module”](#) for additional information.

The CDAFVR<1:0> bits of the FVRCON register are used to enable and configure the gain amplifier settings for the reference supplied to the comparator modules. Reference [Section 19.0 “Comparator Module”](#) for additional information.

To minimize current consumption when the FVR is disabled, the FVR buffers should be turned off by clearing the Buffer Gain Selection bits.

15.2 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 [Figure 36-62: FVR Stabilization Period, PIC16LF1614/8 Only](#).

FIGURE 15-1: VOLTAGE REFERENCE BLOCK DIAGRAM



TABLE 15-1: PERIPHERALS REQUIRING THE FIXED VOLTAGE REFERENCE (FVR)

| Peripheral | Conditions | Description |
|------------|---|--|
| HFINTOSC | FOSC<2:0> = 010 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 PIC16F1615/9 devices, when VREGPM = 1 and not in Sleep | The device runs off of the Low-Power Regulator when in Sleep mode. |

15.3 Register Definitions: FVR Control

REGISTER 15-1: FVRCON: FIXED VOLTAGE REFERENCE CONTROL REGISTER

| | | | | | | | |
|----------------------|-----------------------|---------------------|----------------------|----------------------------|---------|---------------------------|---------|
| R/W-0/0 | R-q/q | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 |
| FVREN ⁽¹⁾ | FVRRDY ⁽²⁾ | TSEN ⁽³⁾ | TSRNG ⁽³⁾ | CDAFVR<1:0> ⁽¹⁾ | | ADFVR<1:0> ⁽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 **FVREN:** Fixed Voltage Reference Enable bit⁽¹⁾
 1 = Fixed Voltage Reference is enabled
 0 = Fixed Voltage Reference is disabled
- bit 6 **FVRRDY:** Fixed Voltage Reference Ready Flag bit⁽²⁾
 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⁽³⁾
 1 = Temperature Indicator is enabled
 0 = Temperature Indicator is disabled
- bit 4 **TSRNG:** Temperature Indicator Range Selection bit⁽³⁾
 1 = V_{OUT} = V_{DD} - 4V_T (High Range)
 0 = V_{OUT} = V_{DD} - 2V_T (Low Range)
- bit 3-2 **CDAFVR<1:0>:** Comparator FVR Buffer Gain Selection bits⁽¹⁾
 11 = Comparator FVR Buffer Gain is 4x, with output V_{CDAFVR} = 4x V_{FVR}⁽⁴⁾
 10 = Comparator FVR Buffer Gain is 2x, with output V_{CDAFVR} = 2x V_{FVR}⁽⁴⁾
 01 = Comparator FVR Buffer Gain is 1x, with output V_{CDAFVR} = 1x V_{FVR}
 00 = Comparator FVR Buffer is off
- bit 1-0 **ADFVR<1:0>:** ADC FVR Buffer Gain Selection bit⁽¹⁾
 11 = ADC FVR Buffer Gain is 4x, with output V_{ADFVR} = 4x V_{FVR}⁽⁴⁾
 10 = ADC FVR Buffer Gain is 2x, with output V_{ADFVR} = 2x V_{FVR}⁽⁴⁾
 01 = ADC FVR Buffer Gain is 1x, with output V_{ADFVR} = 1x V_{FVR}
 00 = ADC FVR Buffer is off

- Note 1:** To minimize current consumption when the FVR is disabled, the FVR buffers should be turned off by clearing the Buffer Gain Selection bits.
- 2:** FVRRDY is always '1' for the PIC16LF1615/9 devices.
- 3:** See [Section 16.0 "Temperature Indicator Module"](#) for additional information.
- 4:** Fixed Voltage Reference output cannot exceed V_{DD}.

TABLE 15-2: SUMMARY OF REGISTERS ASSOCIATED WITH THE FIXED VOLTAGE REFERENCE

| 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 | CDAFVR<1:0> | | ADFVR<1:0> | | 186 |

Legend: Shaded cells are unused by the Fixed Voltage Reference module.

16.0 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.

16.1 Circuit Operation

Figure 16-1 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 16-1 describes the output characteristics of the temperature indicator.

EQUATION 16-1: V_{OUT} RANGES

High Range: $V_{OUT} = V_{DD} - 4V_T$

Low Range: $V_{OUT} = V_{DD} - 2V_T$

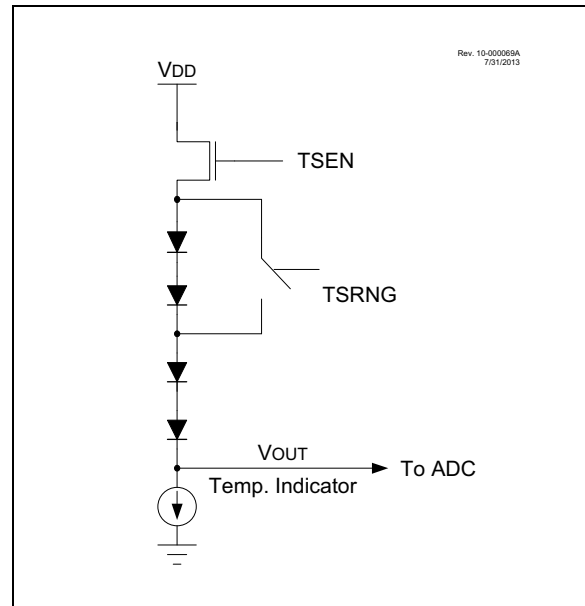
The temperature sense circuit is integrated with the Fixed Voltage Reference (FVR) module. See Section 15.0 "Fixed Voltage Reference (FVR)" for more information.

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 16-1: TEMPERATURE CIRCUIT DIAGRAM



16.2 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 16-1 shows the recommended minimum V_{DD} vs. range setting.

TABLE 16-1: RECOMMENDED V_{DD} VS. RANGE

| Min. V _{DD} , TSRNG = 1 | Min. V _{DD} , TSRNG = 0 |
|----------------------------------|----------------------------------|
| 3.6V | 1.8V |

16.3 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 17.0 "Analog-to-Digital Converter (ADC) Module" for detailed information.

16.4 ADC Acquisition Time

To ensure accurate temperature measurements, the user must wait at least 200 μs after the ADC input multiplexer is connected to the temperature indicator output before the conversion is performed. In addition, the user must wait 200 μs between sequential conversions of the temperature indicator output.

TABLE 16-2: 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 | CDAFVR<1:0> | | ADFVR<1:0> | | 118 |

Legend: Shaded cells are unused by the temperature indicator module.

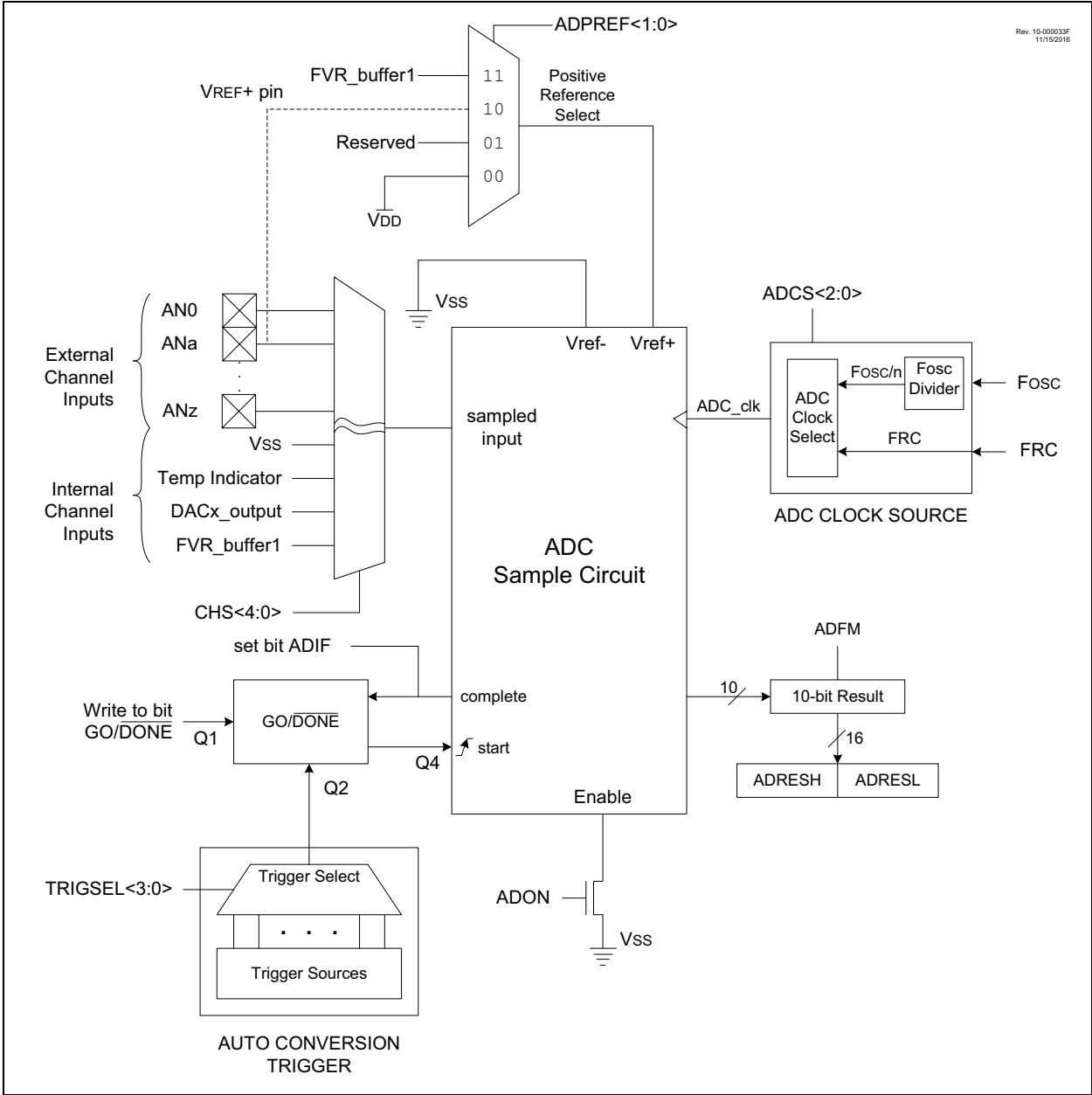
17.0 ANALOG-TO-DIGITAL CONVERTER (ADC) MODULE

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 17-1 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 17-1: ADC BLOCK DIAGRAM



17.1 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

17.1.1 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.

17.1.2 CHANNEL SELECTION

There are up to 15 channel selections available:

- AN<11:0> pins (PIC16(L)F1619 only)
- AN<7:0> pins (PIC16(L)F1615 only)
- Temperature Indicator
- DAC1_output
- FVR_buffer1

The CHS bits of the ADCON0 register determine which channel is connected to the sample and hold circuit.

When changing channels, a delay (TACQ) is required before starting the next conversion. Refer to [Section 17.2.6 “ADC Conversion Procedure”](#) for more information.

Note: It is recommended that when switching from an ADC channel of a higher voltage to a channel of a lower voltage, the software selects the VSS channel before switching to the channel of the lower voltage. If the ADC does not have a dedicated VSS input channel, the VSS selection (DAC1R<4:0> = b'00000') through the DAC output channel can be used. If the DAC is in use, a free input channel can be connected to VSS, and can be used in place of the DAC.

17.1.3 ADC VOLTAGE REFERENCE

The ADC module uses a positive and a negative voltage reference. The positive reference is labeled ref+ and the negative reference is labeled ref-.

The positive voltage reference (ref+) is selected by the ADPREF bits in the ADCON1 register. The positive voltage reference source can be:

- VREF+ pin
- VDD
- FVR_buffer1

The negative voltage reference (ref-) source is:

- VSS

17.1.4 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 (internal RC 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 17-2](#).

For correct conversion, the appropriate TAD specification must be met. Refer to the ADC conversion requirements in [Section 35.0 “Electrical Specifications”](#) for more information. [Table 17-1](#) 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 17-1: ADC CLOCK PERIOD (TAD) Vs. DEVICE OPERATING FREQUENCIES

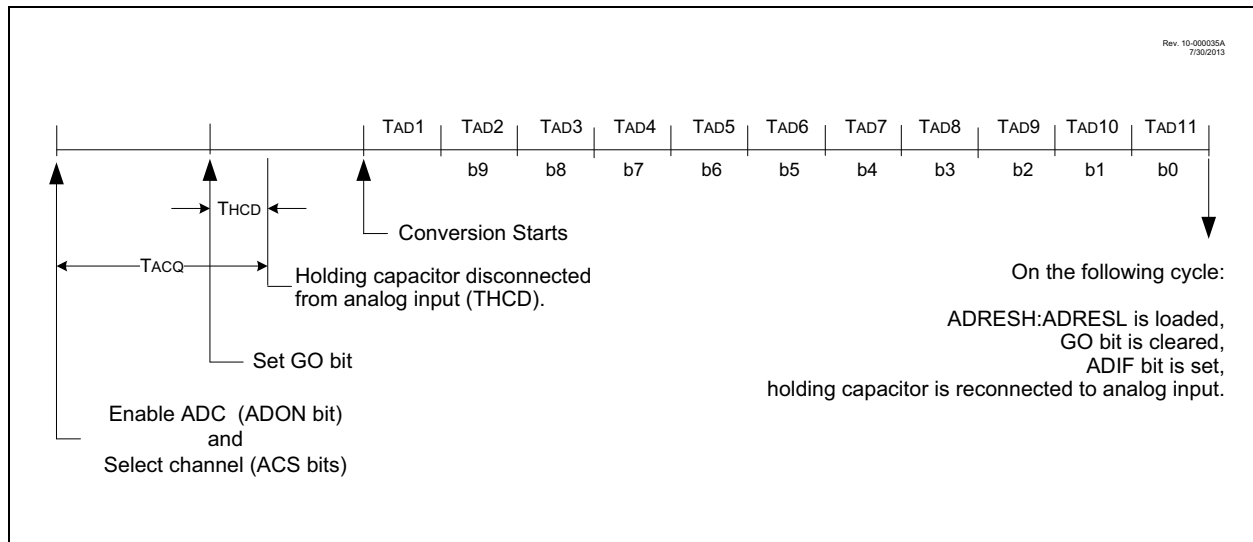
| ADC Clock Period (TAD) | | Device Frequency (Fosc) | | | | |
|------------------------|-----------|-------------------------|------------|------------|------------|------------|
| ADC Clock Source | ADCS<2:0> | 20 MHz | 16 MHz | 8 MHz | 4 MHz | 1 MHz |
| Fosc/2 | 000 | 100 ns | 125 ns | 250 ns | 500 ns | 2.0 μs |
| Fosc/4 | 100 | 200 ns | 250 ns | 500 ns | 1.0 μs | 4.0 μs |
| Fosc/8 | 001 | 400 ns | 500 ns | 1.0 μs | 2.0 μs | 8.0 μs |
| Fosc/16 | 101 | 800 ns | 1.0 μs | 2.0 μs | 4.0 μs | 16.0 μs |
| Fosc/32 | 010 | 1.6 μs | 2.0 μs | 4.0 μs | 8.0 μs | 32.0 μs |
| Fosc/64 | 110 | 3.2 μs | 4.0 μs | 8.0 μs | 16.0 μs | 64.0 μs |
| FRC | x11 | 1.0-6.0 μs | 1.0-6.0 μs | 1.0-6.0 μs | 1.0-6.0 μs | 1.0-6.0 μs |

Legend: Shaded cells are outside of recommended range.

Note 1: The FRC source has a typical TAD time of 1.7 μs.

- When the device frequency is greater than 1 MHz, the FRC clock source is only recommended if the conversion will be performed during Sleep.
- The TAD period when using the FRC clock source can fall within a specified range, (see TAD parameter). The TAD period when using the Fosc-based clock source can be configured for a more precise TAD period. However, the FRC clock source must be used when conversions are to be performed with the device in Sleep mode.

FIGURE 17-2: ANALOG-TO-DIGITAL CONVERSION TAD CYCLES



17.1.5 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.

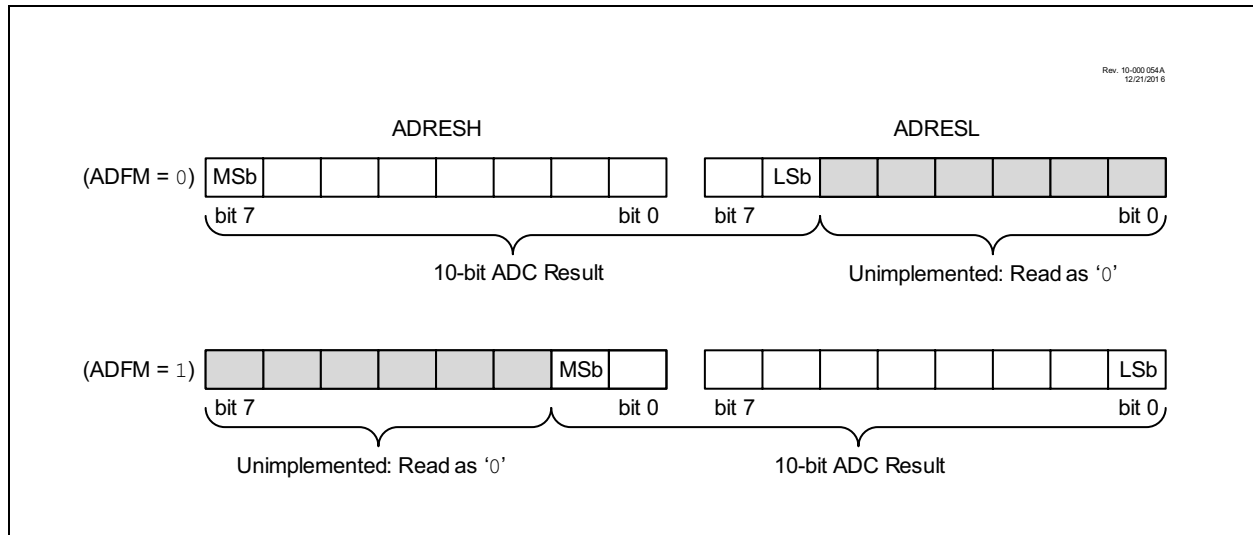
This interrupt can be generated while the device is operating or while in Sleep. If the device is in Sleep, the interrupt will wake-up the device. Upon waking from Sleep, the next instruction following the `SLEEP` instruction is always executed. If the user is attempting to wake-up from Sleep and resume 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.

17.1.6 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 17-3 shows the two output formats.

FIGURE 17-3: 10-BIT ADC CONVERSION RESULT FORMAT



17.2 ADC Operation

17.2.1 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 17.2.6 “ADC Conversion Procedure”](#).

17.2.2 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

17.2.3 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.

17.2.4 ADC OPERATION DURING SLEEP

The ADC module can operate during Sleep. This requires the ADC clock source to be set to the FRC option. Performing the ADC conversion during Sleep can reduce system noise. 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.

17.2.5 AUTO-CONVERSION TRIGGER

The auto-conversion trigger allows periodic ADC measurements without software intervention. When a rising edge of the selected source occurs, the GO/DONE bit is set by hardware.

The auto-conversion trigger source is selected with the TRIGSEL<4:0> bits of the `ADCON2` register.

Using the auto-conversion trigger does not assure proper ADC timing. It is the user's responsibility to ensure that the ADC timing requirements are met.

See [Table 17-2](#) for auto-conversion sources.

TABLE 17-2: AUTO-CONVERSION SOURCES

| Source Peripheral | Signal Name |
|-------------------|-----------------|
| Timer0 | T0_overflow |
| Timer1 | T1_overflow |
| Timer2 | TMR2_postscaled |
| Timer4 | TMR4_postscaled |
| Timer6 | TMR6_postscaled |
| Comparator C1 | C1_OUT_sync |
| Comparator C2 | C2_OUT_sync |
| SMT1 | SMT1_CPW |
| SMT1 | SMT1_CPR |
| SMT1 | SMT1_PR |
| SMT2 | SMT2_CPW |
| SMT2 | SMT2_CPR |
| SMT2 | SMT2_PR |
| CCP1 | CCP1_out |
| CCP2 | CCP2_out |

17.2.6 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).

EXAMPLE 17-1: ADC CONVERSION

```

;This code block configures the ADC
;for polling, Vdd and Vss references, FRC
;oscillator and AN0 input.
;
;Conversion start & polling for completion
;are included.
;
BANKSEL    ADCON1    ;
MOVLW     B'11110000' ;Right justify, FRC
;oscillator
MOVWF     ADCON1    ;Vdd and Vss Vref+
BANKSEL    TRISA     ;
BSF       TRISA,0   ;Set RA0 to input
BANKSEL    WPUA     ;
BCF       WPUA,0    ;Disable weak
;pull-up on RA0
BANKSEL    ANSEL    ;
BSF       ANSEL,0   ;Set RA0 to analog
BANKSEL    ADCON0   ;
MOVLW     B'00000001' ;Select channel AN0
MOVWF     ADCON0    ;Turn ADC On
CALL      SampleTime ;Acquisition 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
BANKSEL    ADRESL   ;
MOVF     ADRESL,W   ;Read lower 8 bits
MOVWF    RESULTLO   ;Store in GPR space
    
```

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 17.4 “ADC Acquisition Requirements”](#).

17.3 Register Definitions: ADC Control

REGISTER 17-1: 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⁽³⁾
 - 11110 = DAC (Digital-to-Analog Converter)⁽²⁾
 - 11101 = Temperature Indicator⁽¹⁾
 - 11100 = Reserved. No channel connected.
 -
 -
 -
 - 01100 = Reserved. No channel connected.
 - 01011 = AN11⁽⁴⁾
 - 01010 = AN10⁽⁴⁾
 - 01001 = AN9⁽⁴⁾
 - 01000 = AN8⁽⁴⁾
 - 00111 = Reserved. No channel connected.
 - 00110 = Reserved. No channel connected.
 - 00101 = Reserved. No channel connected.
 - 00100 = Reserved. No channel connected.
 - 01000 = Reserved. No channel connected.
 - 00111 = AN7
 - 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 16.0 “Temperature Indicator Module”](#).
- Note 2:** See [Section 18.0 “8-bit Digital-to-Analog Converter \(DAC1\) Module”](#) for more information.
- Note 3:** See [Section 15.0 “Fixed Voltage Reference \(FVR\)”](#) for more information.
- Note 4:** AN<11:8> available on PIC16(L)F1619 only.

REGISTER 17-2: 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 an internal RC oscillator)
 110 = Fosc/64
 101 = Fosc/16
 100 = Fosc/4
 011 = FRC (clock supplied from an internal RC 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 = VRPOS is connected to internal Fixed Voltage Reference (FVR)
 10 = VRPOS is connected to external VREF+ pin⁽¹⁾
 01 = Reserved
 00 = VRPOS is connected to VDD

Note 1: When selecting the VREF+ pin as the source of the positive reference, be aware that a minimum voltage specification exists. See [Section TABLE 35-13: "Analog-to-Digital Converter \(ADC\) Characteristics\(1,2,3\)"](#) for details.

REGISTER 17-3: ADCON2: ADC CONTROL REGISTER 2

| | | | | | | | |
|-----------------------------|---------|---------|---------|---------|-------|-----|-----|
| R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | U-0 | U-0 | U-0 |
| TRIGSEL<4:0> ⁽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 | |

bit 7-3 **TRIGSEL<4:0>**: Auto-Conversion Trigger Selection bits⁽¹⁾

- 11111 = Reserved
-
-
-
- 10101 = Reserved
- 10100 = AT1_cmp3
- 10011 = AT1_cmp2
- 10010 = AT1_cmp1
- 10001 = CLC4OUT
- 10000 = CLC3OUT
- 01111 = CLC2OUT
- 01110 = CLC1OUT
- 01101 = TMR5_overflow
- 01100 = TMR3_overflow
- 01011 = SMT2_match
- 01010 = SMT1_match
- 01001 = TMR6_postscaled
- 01000 = TMR4_postscaled
- 00111 = C2_OUT_sync
- 00110 = C1_OUT_sync
- 00101 = TMR2_postscaled
- 00100 = T1_overflow⁽²⁾
- 00011 = T0_overflow⁽²⁾
- 00010 = CCP2_out
- 00001 = CCP1_out
- 00000 = No auto-conversion trigger selected

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

- Note 1:** This is a rising edge sensitive input for all sources.
2: Signal also sets its corresponding interrupt flag.

REGISTER 17-4: 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 eight bits of 10-bit conversion result

REGISTER 17-5: 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 two bits of 10-bit conversion result

bit 5-0 **Reserved**: Do not use.

REGISTER 17-6: 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 two bits of 10-bit conversion result

REGISTER 17-7: 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 eight bits of 10-bit conversion result

17.4 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 17-4. 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 17-4. **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 17-1 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 17-1: 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:

$$\begin{aligned} T_C &= -CHOLD(RIC + RSS + RS) \ln(1/2047) \\ &= -12.5pF(1k\Omega + 7k\Omega + 10k\Omega) \ln(0.0004885) \\ &= 1.12\mu s \end{aligned}$$

Therefore:

$$\begin{aligned} T_{ACQ} &= 2\mu s + 1.12\mu s + [(50^\circ C - 25^\circ C)(0.05\mu s/^\circ C)] \\ &= 4.37\mu s \end{aligned}$$

Note 1: The reference voltage (VRPOS) 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.

FIGURE 17-4: ANALOG INPUT MODEL

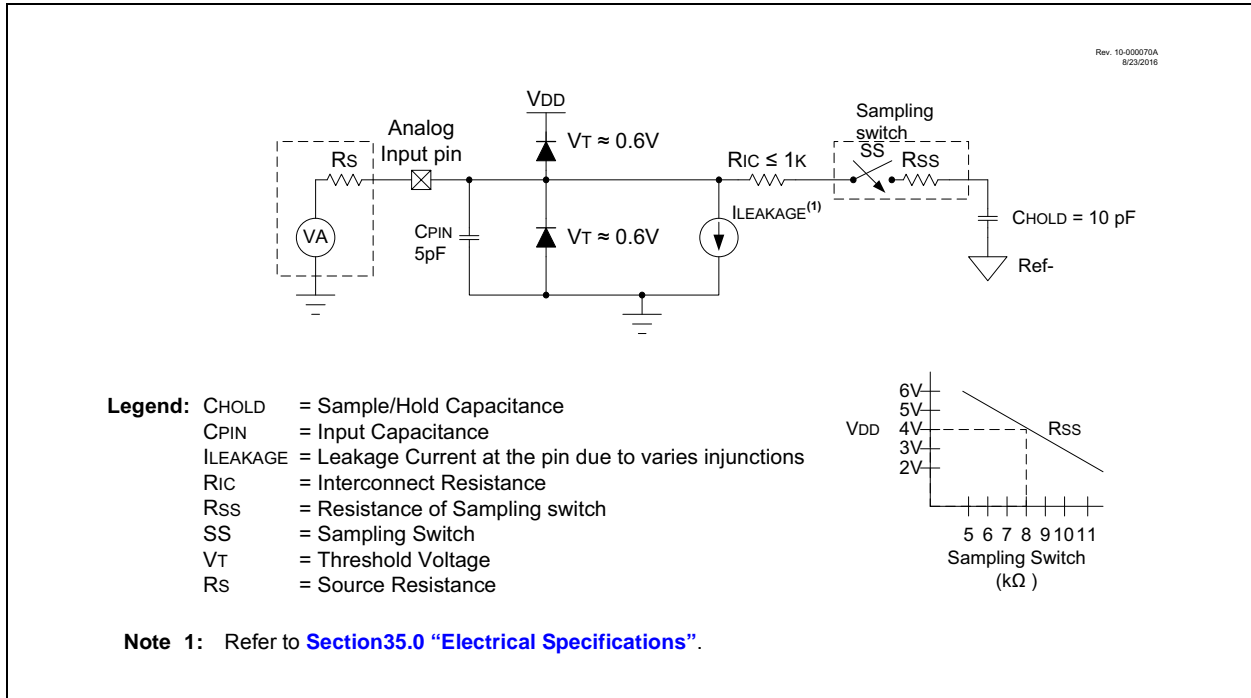


FIGURE 17-5: ADC TRANSFER FUNCTION

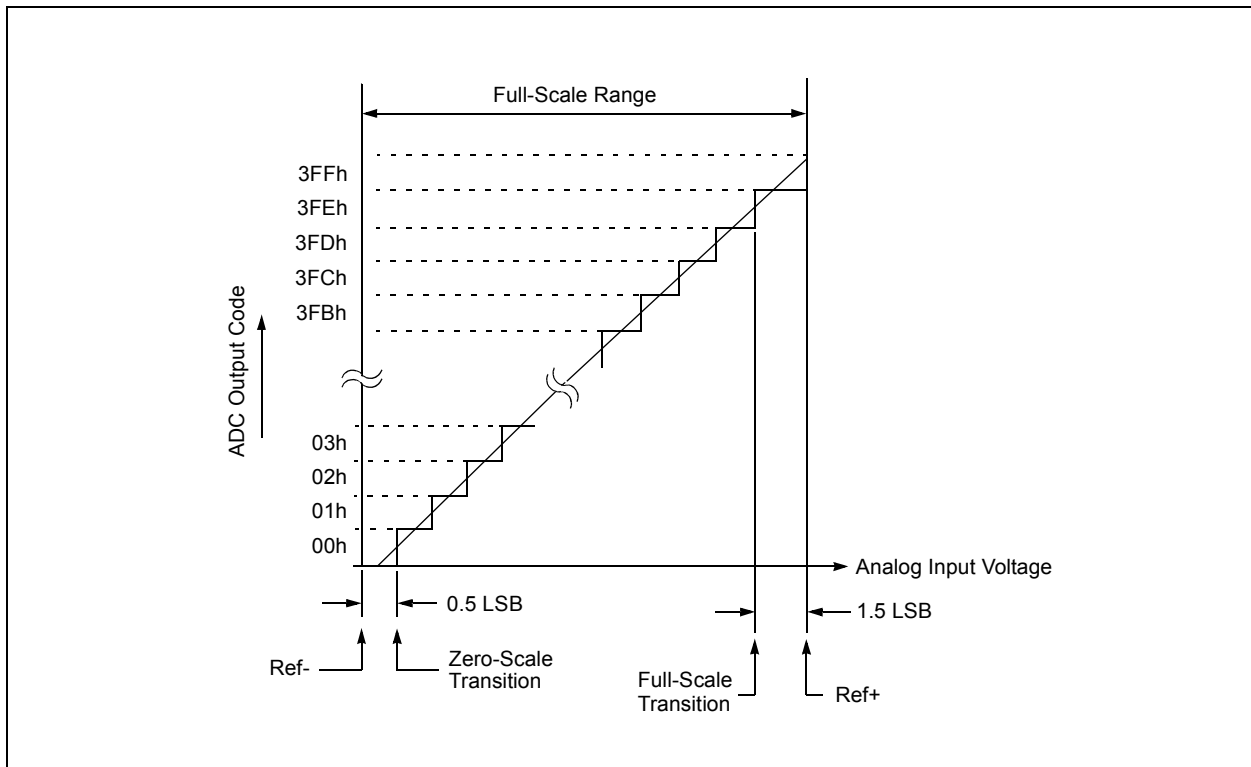


TABLE 17-3: SUMMARY OF REGISTERS ASSOCIATED WITH ADC

| 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 | 195 |
| ADCON1 | ADFM | ADCS<2:0> | | | — | — | ADPREF<1:0> | | 196 |
| ADCON2 | TRIGSEL<4:0> | | | | | — | — | — | 197 |
| ADRESH | ADC Result Register High | | | | | | | | 198, 199 |
| ADRESL | ADC Result Register Low | | | | | | | | 198, 199 |
| ANSELA | — | — | — | ANSA4 | — | ANSA2 | ANSA1 | ANSA0 | 152 |
| ANSELC | ANSC7 ⁽²⁾ | ANSC6 ⁽²⁾ | — | — | ANSC3 | ANSC2 | ANSC1 | ANSC0 | 166 |
| INTCON | GIE | PEIE | TMR0IE | INTE | IOCIE | TMR0IF | INTF | IOCIF | 97 |
| PIE1 | TMR1GIE | ADIE | RCIE | TXIE | SSP1IE | CCP1IE | TMR2IE | TMR1IE | 98 |
| PIR1 | TMR1GIF | ADIF | RCIF | TXIF | SSP1IF | CCP1IF | TMR2IF | TMR1IF | 103 |
| TRISA | — | — | TRISA5 | TRISA4 | — ⁽¹⁾ | TRISA2 | TRISA1 | TRISA0 | 151 |
| TRISC | TRISC7 ⁽²⁾ | TRISC6 ⁽²⁾ | TRISC5 | TRISC4 | TRISC3 | TRISC2 | TRISC1 | TRISC0 | 165 |
| FVRCON | FVREN | FVRRDY | TSEN | TSRNG | CDAFVR<1:0> | | ADFVR<1:0> | | 186 |

Legend: x = unknown, u = unchanged, — = unimplemented read as '0', q = value depends on condition. Shaded cells are not used for ADC module.

- Note 1:** Unimplemented, read as '1'.
Note 2: PIC16(L)F1619 only.

18.0 8-BIT DIGITAL-TO-ANALOG CONVERTER (DAC1) MODULE

The Digital-to-Analog Converter supplies a variable voltage reference, ratiometric with the input source, with 256 selectable output levels.

The input of the DAC can be connected to:

- External VREF pins
- VDD supply voltage
- FVR (Fixed Voltage Reference)

The output of the DAC can be configured to supply a reference voltage to the following:

- Comparator positive input
- ADC input channel
- DACxOUT1 pin

The Digital-to-Analog Converter (DAC) is enabled by setting the DAC1EN bit of the DAC1CON0 register.

18.1 Output Voltage Selection

The DAC has 256 voltage level ranges. The 256 levels are set with the DAC1R<7:0> bits of the DAC1CON1 register.

The DAC output voltage is determined by [Equation 18-1](#):

EQUATION 18-1: DAC OUTPUT VOLTAGE

IF DAC1EN = 1

$$V_{OUT} = \left((V_{SOURCE+} - V_{SOURCE-}) \times \frac{DAC1R[7:0]}{2^8} \right) + V_{SOURCE-}$$

V_{SOURCE+} = VDD, VREF, or FVR BUFFER 2

V_{SOURCE-} = VSS

18.2 Ratiometric Output Level

The DAC output value is derived using a resistor ladder with each end of the ladder tied to a positive and negative voltage reference input source. If the voltage of either input source fluctuates, a similar fluctuation will result in the DAC output value.

The value of the individual resistors within the ladder can be found in [Section 35.0 “Electrical Specifications”](#).

18.3 DAC Voltage Reference Output

The DAC voltage can be output to the DACxOUT1 pin by setting the DAC1OE1 bit of the DAC1CON0 register. Selecting the DAC reference voltage for output on the DACxOUT1 pin automatically overrides the digital output buffer and digital input threshold detector functions of that pin. Reading the DACxOUT1 pin when it has been configured for DAC reference voltage output will always return a '0'.

Due to the limited current drive capability, a buffer must be used on the DAC voltage reference output for external connections to either DACxOUT1 pin. [Figure 18-2](#) shows an example buffering technique.

FIGURE 18-1: DIGITAL-TO-ANALOG CONVERTER BLOCK DIAGRAM

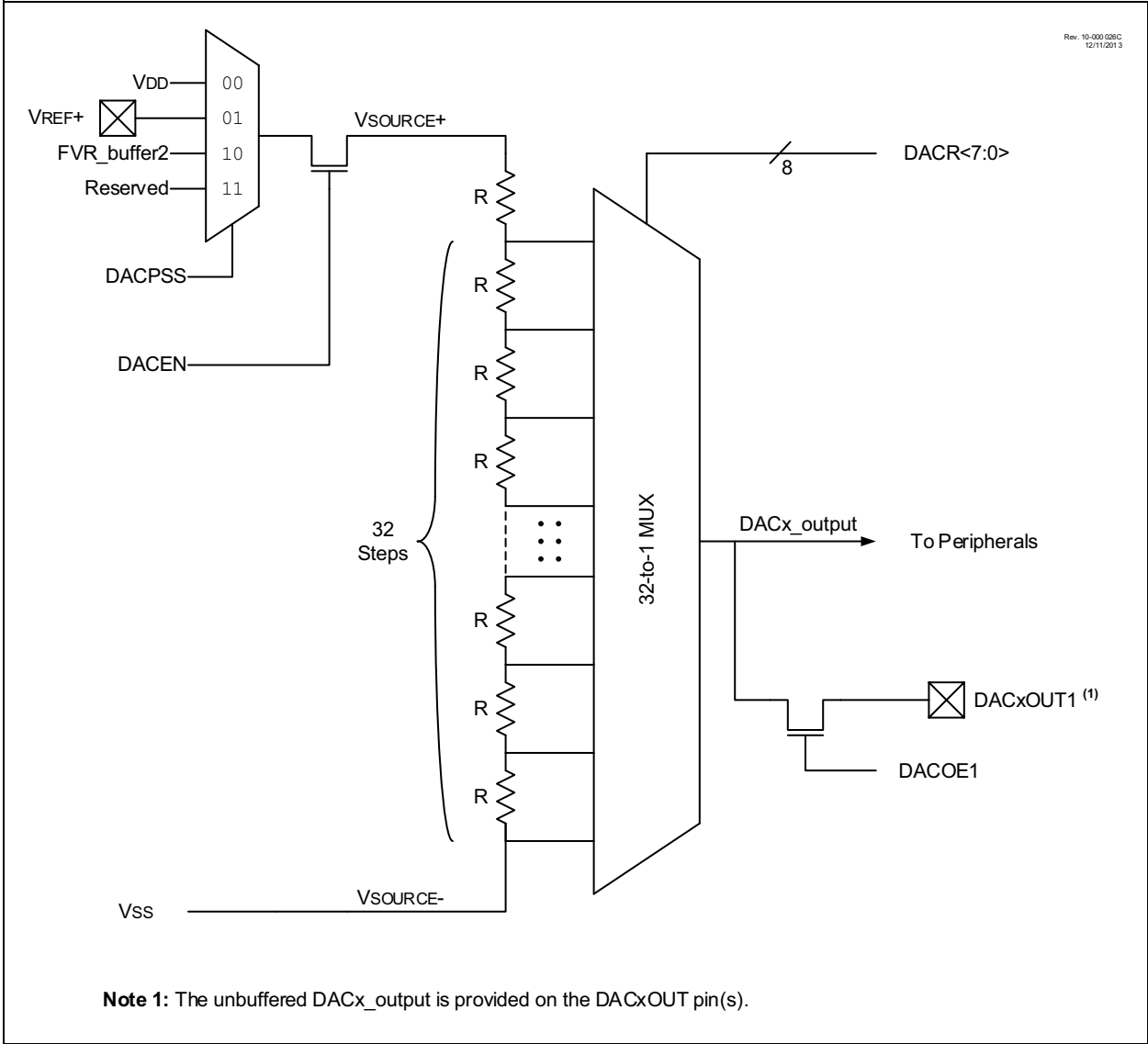
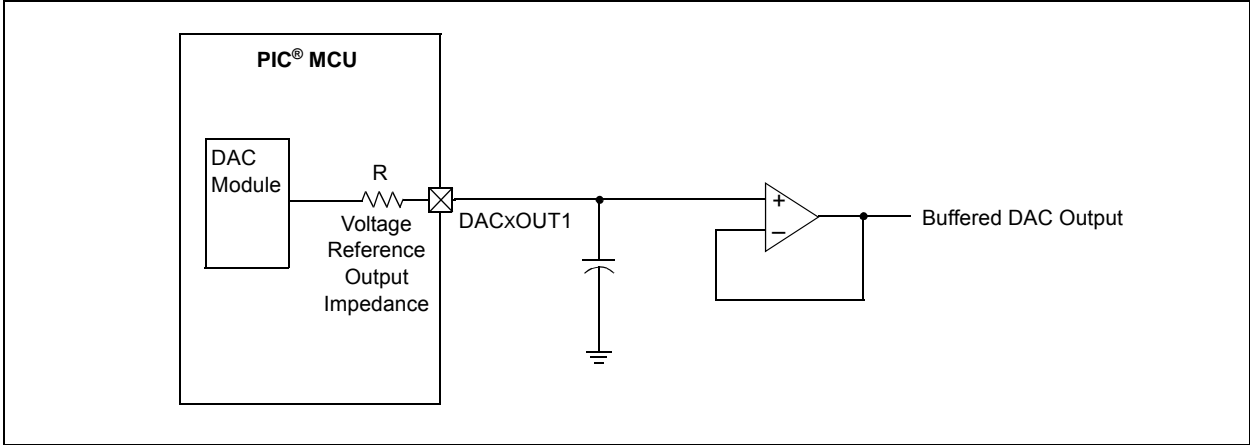


FIGURE 18-2: VOLTAGE REFERENCE OUTPUT BUFFER EXAMPLE



18.4 Operation During Sleep

When the device wakes up from Sleep through an interrupt or a Watchdog Timer time-out, the contents of the DAC1CON0 register are not affected. To minimize current consumption in Sleep mode, the voltage reference should be disabled.

18.5 Effects of a Reset

A device Reset affects the following:

- DAC is disabled.
- DAC output voltage is removed from the DACxOUT1 pin.
- The DAC1R<7:0> range select bits are cleared.

18.6 Register Definitions: DAC Control

REGISTER 18-1: DAC1CON0: DAC1 CONTROL REGISTER 0

| | | | | | | | |
|---------|-----|---------|-----|--------------|---------|-----|-------|
| R/W-0/0 | U-0 | R/W-0/0 | U-0 | R/W-0/0 | R/W-0/0 | U-0 | U-0 |
| DAC1EN | — | DAC1OE1 | — | DAC1PSS<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 **DAC1EN:** DAC1 Enable bit
1 = DAC is enabled
0 = DAC is disabled
- bit 6 **Unimplemented:** Read as '0'
- bit 5 **DAC1OE1:** DAC1 Voltage Output 1 Enable bit
1 = DAC voltage level is also an output on the DACxOUT1 pin
0 = DAC voltage level is disconnected from the DACxOUT1 pin
- bit 4 **Unimplemented:** Read as '0'
- bit 3-2 **DAC1PSS<1:0>:** DAC1 Positive Source Select bits
11 = Reserved, do not use
10 = FVR Buffer2 output
01 = VREF+ pin
00 = VDD
- bit 1-0 **Unimplemented:** Read as '0'

REGISTER 18-2: DAC1CON1: DAC1 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 |
| DAC1R<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 **DAC1R<7:0>:** DAC1 Voltage Output Select bits

TABLE 18-1: SUMMARY OF REGISTERS ASSOCIATED WITH THE DAC1 MODULE

| 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 | CDAFVR<1:0> | | ADFVR<1:0> | | 186 |
| DAC1CON0 | DAC1EN | — | DAC1OE1 | — | DAC1PSS<1:0> | | — | — | 206 |
| DAC1CON1 | DAC1R<7:0> | | | | | | | | 206 |

Legend: — = Unimplemented location, read as '0'. Shaded cells are not used with the DAC module.

19.0 COMPARATOR MODULE

Comparators are used to interface analog circuits to a digital circuit by comparing two analog voltages and providing a digital indication of their relative magnitudes. Comparators are very useful mixed signal building blocks because they provide analog functionality independent of program execution. The analog comparator module includes the following features:

- Independent comparator control
- Programmable input selection
- Comparator output is available internally/externally
- Programmable output polarity
- Interrupt-on-change
- Wake-up from Sleep
- Programmable Speed/Power optimization
- PWM shutdown
- Programmable and Fixed Voltage Reference

19.1 Comparator Overview

A single comparator is shown in [Figure 19-1](#) along with the relationship between the analog input levels and the digital output. When the analog voltage at V_{IN+} is less than the analog voltage at V_{IN-} , the output of the comparator is a digital low level. When the analog voltage at V_{IN+} is greater than the analog voltage at V_{IN-} , the output of the comparator is a digital high level.

The comparators available for this device are located in [Table 19-1](#).

FIGURE 19-1: SINGLE COMPARATOR

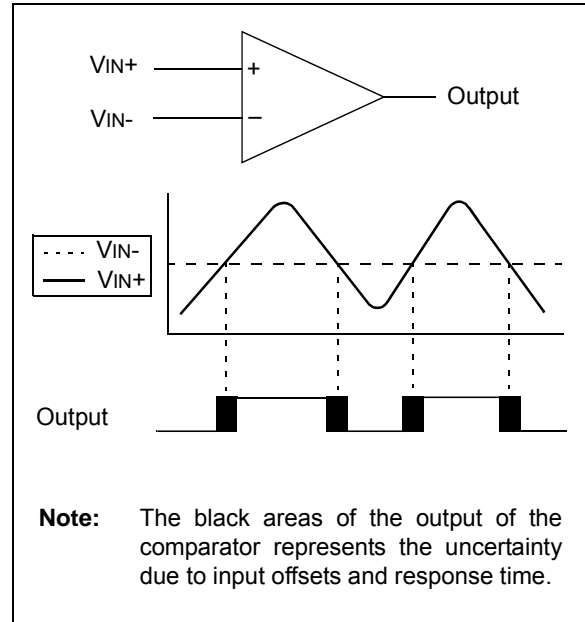


TABLE 19-1: COMPARATOR AVAILABILITY PER DEVICE

| Device | C1 | C2 |
|---------------|----|----|
| PIC16(L)F1619 | • | • |
| PIC16(L)F1615 | • | • |

FIGURE 19-2: COMPARATOR MODULE SIMPLIFIED BLOCK DIAGRAM



19.2 Comparator Control

Each comparator has two control registers: CMxCON0 and CMxCON1.

The CMxCON0 registers (see [Register 19-1](#)) contain Control and Status bits for the following:

- Enable
- Output selection
- Output polarity
- Speed/Power selection
- Hysteresis enable
- Output synchronization

The CMxCON1 registers (see [Register 19-2](#)) contain Control bits for the following:

- Interrupt enable
- Interrupt edge polarity
- Positive input channel selection
- Negative input channel selection

19.2.1 COMPARATOR ENABLE

Setting the CxON bit of the CMxCON0 register enables the comparator for operation. Clearing the CxON bit disables the comparator resulting in minimum current consumption.

19.2.2 COMPARATOR OUTPUT SELECTION

The output of the comparator can be monitored by reading either the CxOUT bit of the CMxCON0 register or the MCxOUT bit of the CMOUT register. In order to make the output available for an external connection, the following conditions must be true:

- CxOE bit of the CMxCON0 register must be set
- Corresponding TRIS bit must be cleared
- CxON bit of the CMxCON0 register must be set

Note 1: The CxOE bit of the CMxCON0 register overrides the PORT data latch. Setting the CxON bit of the CMxCON0 register has no impact on the port override.

2: The internal output of the comparator is latched with each instruction cycle. Unless otherwise specified, external outputs are not latched.

19.2.3 COMPARATOR OUTPUT POLARITY

Inverting the output of the comparator is functionally equivalent to swapping the comparator inputs. The polarity of the comparator output can be inverted by setting the CxPOL bit of the CMxCON0 register. Clearing the CxPOL bit results in a non-inverted output.

[Table 19-2](#) shows the output state versus input conditions, including polarity control.

TABLE 19-2: COMPARATOR OUTPUT STATE VS. INPUT CONDITIONS

| Input Condition | CxPOL | CxOUT |
|-----------------|-------|-------|
| $CxVN > CxVP$ | 0 | 0 |
| $CxVN < CxVP$ | 0 | 1 |
| $CxVN > CxVP$ | 1 | 1 |
| $CxVN < CxVP$ | 1 | 0 |

19.2.4 COMPARATOR SPEED/POWER SELECTION

The trade-off between speed or power can be optimized during program execution with the CxSP control bit. The default state for this bit is '1' which selects the Normal Speed mode. Device power consumption can be optimized at the cost of slower comparator propagation delay by clearing the CxSP bit to '0'.

19.3 Comparator Hysteresis

A selectable amount of separation voltage can be added to the input pins of each comparator to provide a hysteresis function to the overall operation. Hysteresis is enabled by setting the CxHYS bit of the CMxCON0 register.

See [Section 35.0 “Electrical Specifications”](#) for more information.

19.4 Timer1 Gate Operation

The output resulting from a comparator operation can be used as a source for gate control of Timer1. See [Section 22.5 “Timer1 Gate”](#) for more information. This feature is useful for timing the duration or interval of an analog event.

It is recommended that the comparator output be synchronized to Timer1. This ensures that Timer1 does not increment while a change in the comparator is occurring.

19.4.1 COMPARATOR OUTPUT SYNCHRONIZATION

The output from a comparator can be synchronized with Timer1 by setting the CxSYNC bit of the CMxCON0 register.

Once enabled, the comparator output is latched on the falling edge of the Timer1 source clock. If a prescaler is used with Timer1, the comparator output is latched after the prescaling function. To prevent a race condition, the comparator output is latched on the falling edge of the Timer1 clock source and Timer1 increments on the rising edge of its clock source. See the Comparator Block Diagram ([Figure 19-2](#)) and the Timer1 Block Diagram ([Figure 22-1](#)) for more information.

19.5 Comparator Interrupt

An interrupt can be generated upon a change in the output value of the comparator for each comparator, a rising edge detector and a falling edge detector are present.

When either edge detector is triggered and its associated enable bit is set (CxINTP and/or CxINTN bits of the CMxCON1 register), the Corresponding Interrupt Flag bit (CxIF bit of the PIR2 register) will be set.

To enable the interrupt, the following bits must be set:

- CxON, CxPOL and CxSP bits of the CMxCON0 register
- CxIE bit of the PIE2 register
- CxINTP bit of the CMxCON1 register (for a rising edge detection)
- CxINTN bit of the CMxCON1 register (for a falling edge detection)
- PEIE and GIE bits of the INTCON register

The associated interrupt flag bit, CxIF bit of the PIR2 register, must be cleared in software. If another edge is detected while this flag is being cleared, the flag will still be set at the end of the sequence.

Note: Although a comparator is disabled, an interrupt can be generated by changing the output polarity with the CxPOL bit of the CMxCON0 register, or by switching the comparator on or off with the CxON bit of the CMxCON0 register.

19.6 Comparator Positive Input Selection

Configuring the CxPCH<1:0> bits of the CMxCON1 register directs an internal voltage reference or an analog pin to the non-inverting input of the comparator:

- CxIN+ analog pin
- DAC output
- FVR (Fixed Voltage Reference)
- Vss (Ground)

See [Section 15.0 “Fixed Voltage Reference \(FVR\)”](#) for more information on the Fixed Voltage Reference module.

See [Section 18.0 “8-bit Digital-to-Analog Converter \(DAC1\) Module”](#) for more information on the DAC input signal.

Any time the comparator is disabled (CxON = 0), all comparator inputs are disabled.

19.7 Comparator Negative Input Selection

The CxNCH<2:0> bits of the CMxCON1 register direct an analog input pin or analog ground to the inverting input of the comparator:

- CxIN0- pin
- CxIN1- pin
- CxIN2- pin
- CxIN3- pin
- Analog Ground
- FVR_buffer2

Some inverting input selections share a pin with the operational amplifier output function. Enabling both functions at the same time will direct the operational amplifier output to the comparator inverting input.

Note: To use CxINy+ and CxINy- pins as analog input, the appropriate bits must be set in the ANSEL register and the corresponding TRIS bits must also be set to disable the output drivers.

19.8 Comparator Response Time

The comparator output is indeterminate for a period of time after the change of an input source or the selection of a new reference voltage. This period is referred to as the response time. The response time of the comparator differs from the settling time of the voltage reference. Therefore, both of these times must be considered when determining the total response time to a comparator input change. See the Comparator and Voltage Reference Specifications in [Section 35.0 “Electrical Specifications”](#) for more details.

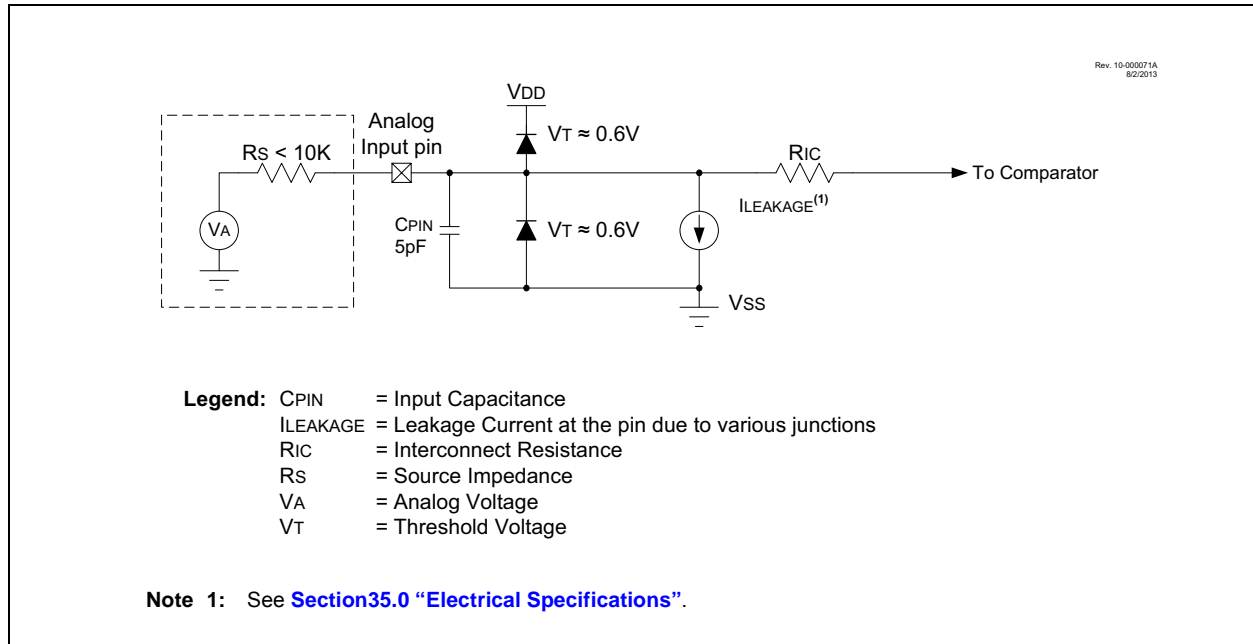
19.9 Analog Input Connection Considerations

A simplified circuit for an analog input is shown in [Figure 19-3](#). Since the analog input pins share their connection with a digital input, they have reverse biased ESD protection diodes to VDD and VSS. The analog input, therefore, must be between VSS and VDD. If the input voltage deviates from this range by more than 0.6V in either direction, one of the diodes is forward biased and a latch-up may occur.

A maximum source impedance of 10 kΩ is recommended for the analog sources. Also, any external component connected to an analog input pin, such as a capacitor or a Zener diode, should have very little leakage current to minimize inaccuracies introduced.

- Note 1:** When reading a PORT register, all pins configured as analog inputs will read as a '0'. Pins configured as digital inputs will convert as an analog input, according to the input specification.
- 2:** Analog levels on any pin defined as a digital input, may cause the input buffer to consume more current than is specified.

FIGURE 19-3: ANALOG INPUT MODEL



19.10 Register Definitions: Comparator Control

REGISTER 19-1: CMxCON0: COMPARATOR Cx CONTROL REGISTER 0

| | | | | | | | |
|---------|-------|-----|---------|-----|---------|---------|---------|
| R/W-0/0 | R-0/0 | U-0 | R/W-0/0 | U-0 | R/W-1/1 | R/W-0/0 | R/W-0/0 |
| CxON | CxOUT | — | CxPOL | — | CxSP | CxHYS | CxSYNC |
| 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 **CxON:** Comparator Enable bit
 1 = Comparator is enabled
 0 = Comparator is disabled and consumes no active power
- bit 6 **CxOUT:** Comparator Output bit
 If CxPOL = 1 (inverted polarity):
 1 = CxVP < CxVN
 0 = CxVP > CxVN
 If CxPOL = 0 (non-inverted polarity):
 1 = CxVP > CxVN
 0 = CxVP < CxVN
- bit 5 **Unimplemented:** Read as '0'
- bit 4 **CxPOL:** Comparator Output Polarity Select bit
 1 = Comparator output is inverted
 0 = Comparator output is not inverted
- bit 3 **Unimplemented:** Read as '0'
- bit 2 **CxSP:** Comparator Speed/Power Select bit
 1 = Comparator operates in normal power, higher speed mode
 0 = Comparator operates in Low-power, Low-speed mode
- bit 1 **CxHYS:** Comparator Hysteresis Enable bit
 1 = Comparator hysteresis enabled
 0 = Comparator hysteresis disabled
- bit 0 **CxSYNC:** Comparator Output Synchronous Mode bit
 1 = Comparator output to Timer1 and I/O pin is synchronous to changes on Timer1 clock source.
 Output updated on the falling edge of Timer1 clock source.
 0 = Comparator output to Timer1 and I/O pin is asynchronous

REGISTER 19-2: CMxCON1: COMPARATOR Cx 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 |
| CxINTP | CxINTN | CxPCH<1:0> | — | CxNCH<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 **CxINTP:** Comparator Interrupt on Positive Going Edge Enable bits
 1 = The CxIF interrupt flag will be set upon a positive going edge of the CxOUT bit
 0 = No interrupt flag will be set on a positive going edge of the CxOUT bit
- bit 6 **CxINTN:** Comparator Interrupt on Negative Going Edge Enable bits
 1 = The CxIF interrupt flag will be set upon a negative going edge of the CxOUT bit
 0 = No interrupt flag will be set on a negative going edge of the CxOUT bit
- bit 5-4 **CxPCH<1:0>:** Comparator Positive Input Channel Select bits
 11 = CxVP connects to AGND
 10 = CxVP connects to FVR Buffer 2
 01 = CxVP connects to VDAC
 00 = CxVP connects to CxIN+ pin
- bit 3 **Unimplemented:** Read as '0'
- bit 2-0 **CxNCH<2:0>:** Comparator Negative Input Channel Select bits
 111 = CxVN connects to AGND
 110 = CxVN connects to FVR Buffer 2
 101 = Reserved
 100 = Reserved
 011 = CxVN connects to CxIN3- pin
 010 = CxVN connects to CxIN2- pin
 001 = CxVN connects to CxIN1- pin
 000 = CxVN connects to CxIN0- pin

REGISTER 19-3: CMOUT: COMPARATOR OUTPUT REGISTER

| | | | | | | | |
|-------|-----|-----|-----|-----|-----|--------|--------|
| U-0 | U-0 | U-0 | U-0 | U-0 | U-0 | R-0/0 | R-0/0 |
| — | — | — | — | — | — | MC2OUT | MC1OUT |
| 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 **MC2OUT:** Mirror Copy of C2OUT bit
- bit 0 **MC1OUT:** Mirror Copy of C1OUT bit

TABLE 19-3: SUMMARY OF REGISTERS ASSOCIATED WITH COMPARATOR MODULE

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Register on Page |
|----------------------|-----------------------|-----------------------|------------|--------|------------------|------------|------------|--------|------------------|
| ANSELA | — | — | — | ANSA4 | — | ANSA2 | ANSA1 | ANSA0 | 152 |
| CM1CON0 | C1ON | C1OUT | — | C1POL | — | C1SP | C1HYS | C1SYNC | 212 |
| CM1CON1 | C1INTP | C1INTN | C1PCH<1:0> | | — | C1NCH<2:0> | | | 213 |
| CM2CON0 | C2ON | C2OUT | — | C2POL | — | C2SP | C2HYS | C2SYNC | 212 |
| CM2CON1 | C2INTP | C2INTN | C2PCH<1:0> | | — | C2NCH<2:0> | | | 213 |
| CMOUT | — | — | — | — | — | — | MC2OUT | MC1OUT | 213 |
| FVRCON | FVREN | FVRRDY | TSEN | TSRNG | CDAFVR<1:0> | | ADFVR<1:0> | | 186 |
| DAC1CON0 | DAC1EN | — | DAC1OE1 | — | DAC1PSS<1:0> | | — | — | 206 |
| DAC1CON1 | DAC1R<7:0> | | | | | | | | 206 |
| INTCON | GIE | PEIE | TMR0IE | INTE | IOCIE | TMR0IF | INTF | IOCIF | 97 |
| PIE2 | OSFIE | C2IE | C1IE | — | BCL1IE | TMR6IE | TMR4IE | CCP2IE | 99 |
| PIR2 | OSFIF | C2IF | C1IF | — | BCL1IF | TMR6IF | TMR4IF | CCP2IF | 104 |
| TRISA | — | — | TRISA5 | TRISA4 | — ⁽¹⁾ | TRISA2 | TRISA1 | TRISA0 | 151 |
| TRISC ⁽²⁾ | TRISC7 ⁽²⁾ | TRISC6 ⁽²⁾ | TRISC5 | TRISC4 | TRISC3 | TRISC2 | TRISC1 | TRISC0 | 165 |

Legend: — = unimplemented location, read as '0'. Shaded cells are unused by the comparator module.

Note 1: Unimplemented, read as '1'.

2: PIC16F1619 only.

20.0 ZERO-CROSS DETECTION (ZCD) MODULE

The ZCD module detects when an A/C signal crosses through the ground potential. The actual zero crossing threshold is the zero crossing reference voltage, V_{CPINV} , which is typically 0.75V above ground.

The connection to the signal to be detected is through a series current limiting resistor. The module applies a current source or sink to the ZCD pin to maintain a constant voltage on the pin, thereby preventing the pin voltage from forward biasing the ESD protection diodes. When the applied voltage is greater than the reference voltage, the module sinks current. When the applied voltage is less than the reference voltage, the module sources current. The current source and sink action keeps the pin voltage constant over the full range of the applied voltage. The ZCD module is shown in the simplified block diagram [Figure 20-2](#).

The ZCD module is useful when monitoring an A/C waveform for, but not limited to, the following purposes:

- A/C period measurement
- Accurate long term time measurement
- Dimmer phase delayed drive
- Low EMI cycle switching

20.1 External Resistor Selection

The ZCD module requires a current limiting resistor in series with the external voltage source. The impedance and rating of this resistor depends on the external source peak voltage. Select a resistor value that will drop all of the peak voltage when the current through the resistor is nominally 300 μ A. Refer to [Equation 20-1](#) and [Figure 20-1](#). Make sure that the ZCD I/O pin internal weak pull-up is disabled so it does not interfere with the current source and sink.

EQUATION 20-1: EXTERNAL RESISTOR

$$R_{SERIES} = \frac{V_{PEAK}}{3 \times 10^{-4}}$$

FIGURE 20-1: EXTERNAL VOLTAGE

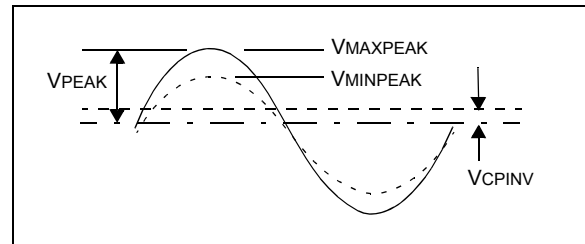
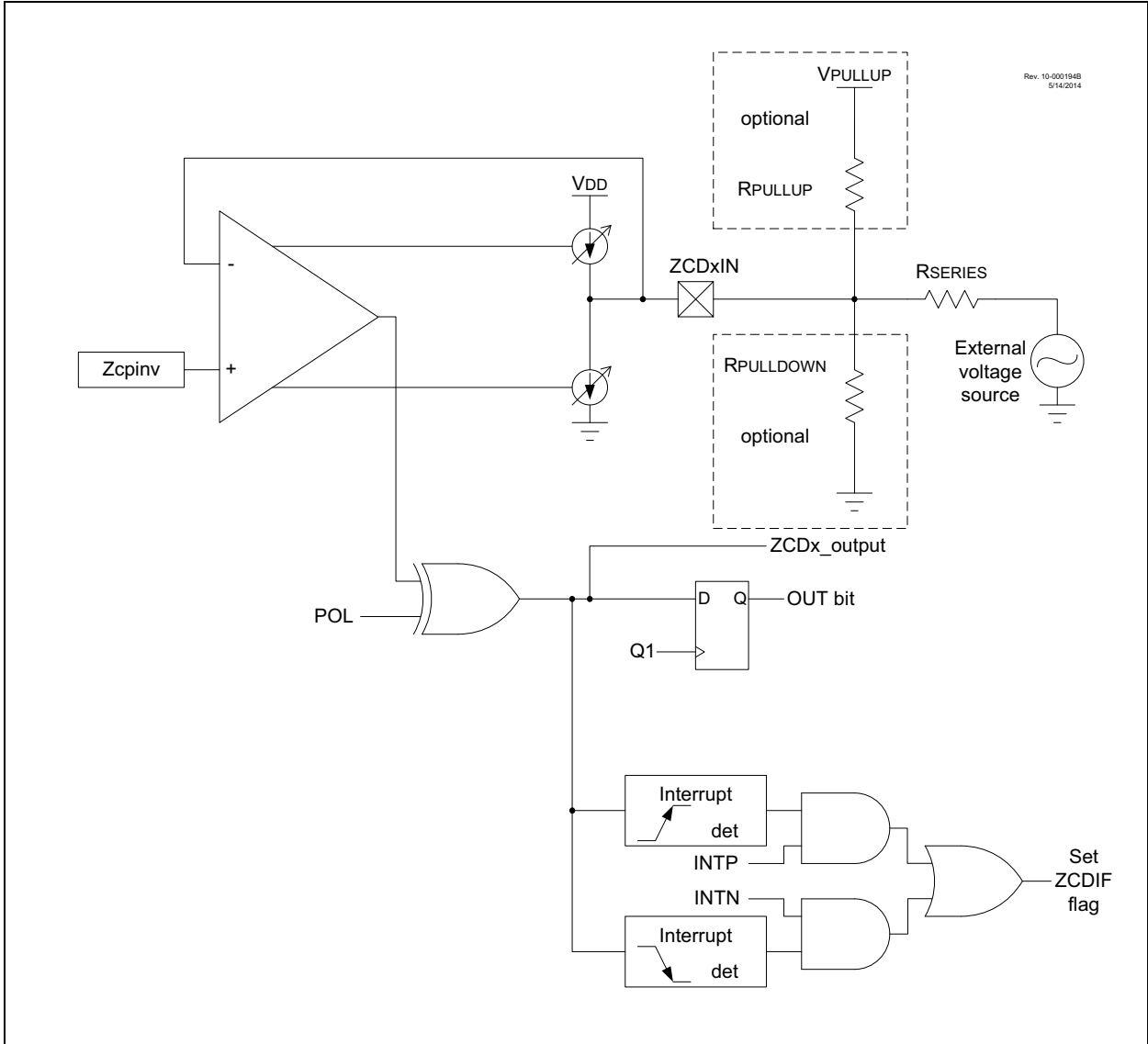


FIGURE 20-2: SIMPLIFIED ZCD BLOCK DIAGRAM



20.2 ZCD Logic Output

The ZCD module includes a Status bit, which can be read to determine whether the current source or sink is active. The ZCDxOUT bit of the ZCDxCON register is set when the current sink is active, and cleared when the current source is active. The ZCDxOUT bit is affected by the polarity bit.

20.3 ZCD Logic Polarity

The ZCDxPOL bit of the ZCDxCON register inverts the ZCDxOUT bit relative to the current source and sink output. When the ZCDxPOL bit is set, a ZCDxOUT high indicates that the current source is active, and a low output indicates that the current sink is active.

The ZCDxPOL bit affects the ZCD interrupts. See [Section 20.4 “ZCD Interrupts”](#).

20.4 ZCD Interrupts

An interrupt will be generated upon a change in the ZCD logic output when the appropriate interrupt enables are set. A rising edge detector and a falling edge detector are present in the ZCD for this purpose.

The ZCDIF bit of the PIR3 register will be set when either edge detector is triggered and its associated enable bit is set. The ZCDxINTP enables rising edge interrupts and the ZCDxINTN bit enables falling edge interrupts. Both are located in the ZCDxCON register.

To fully enable the interrupt, the following bits must be set:

- ZCDIE bit of the PIE3 register
- ZCDxINTP bit of the ZCDxCON register (for a rising edge detection)
- ZCDxINTN bit of the ZCDxCON register (for a falling edge detection)
- PEIE and GIE bits of the INTCON register

Changing the ZCDxPOL bit will cause an interrupt, regardless of the level of the ZCDxEN bit.

The ZCDIF bit of the PIR3 register must be cleared in software as part of the interrupt service. If another edge is detected while this flag is being cleared, the flag will still be set at the end of the sequence.

20.5 Correcting for VCPINV offset

The actual voltage at which the ZCD switches is the reference voltage at the non-inverting input of the ZCD op amp. For external voltage source waveforms other than square waves, this voltage offset from zero causes the zero-cross event to occur either too early or too late.

20.5.1 CORRECTION BY AC COUPLING

When the external voltage source is sinusoidal, the effects of the ZCPINV offset can be eliminated by isolating the external voltage source from the ZCD pin with a capacitor, in addition to the voltage reducing resistor. The capacitor will cause a phase shift resulting in the ZCD output switch in advance of the actual zero-crossing event. The phase shift will be the same for both rising and falling zero crossings, which can be compensated for by either delaying the CPU response to the ZCD switch by a timer or other means, or selecting a capacitor value large enough that the phase shift is negligible.

To determine the series resistor and capacitor values for this configuration, start by computing the impedance, Z , to obtain a peak current of 300 μ A. Next, arbitrarily select a suitably large non-polar capacitor and compute its reactance, X_c , at the external voltage source frequency. Finally, compute the series resistor, capacitor peak voltage, and phase shift by the formulas shown in [Equation 20-2](#).

When this technique is used and the input signal is not present, the ZCD will tend to oscillate. To avoid this oscillation, connect the ZCD pin to VDD or GND with a high-impedance resistor such as 200K.

EQUATION 20-2: R-C CALCULATIONS

V_{PEAK} = External voltage source peak voltage
 f = External voltage source frequency
 C = Series capacitor
 R = Series resistor
 V_C = Peak capacitor voltage
 Φ = Capacitor induced zero crossing phase advance in radians
 T_Φ = Time ZC event occurs before actual zero crossing

$$Z = \frac{V_{PEAK}}{3 \times 10^{-4}}$$

$$X_C = \frac{1}{2\pi f C}$$

$$R = \sqrt{Z^2 - X_C^2}$$

$$V_C = X_C(3 \times 10^{-4})$$

$$\Phi = \tan^{-1}\left(\frac{X_C}{R}\right)$$

$$T_\Phi = \frac{\Phi}{2\pi f}$$

EXAMPLE 20-1: R-C CALCULATIONS

$V_{RMS} = 120$
 $V_{PEAK} = V_{RMS} \times \sqrt{2} = 169.7$
 $f = 60 \text{ Hz}$
 $C = 0.1 \mu\text{F}$

$$Z = \frac{V_{PEAK}}{3 \times 10^{-4}} = \frac{169.7}{3 \times 10^{-4}} = 565.7 \text{ k}\Omega$$

$$X_C = \frac{1}{2\pi f C} = \frac{1}{(2\pi \times 60 \times 1 \times 10^{-7})} = 26.53 \text{ k}\Omega$$

$$R = \sqrt{(Z^2 - X_C^2)} = 565.1 \text{ k}\Omega \text{ (computed)}$$

$$R = 560 \text{ k}\Omega \text{ (used)}$$

$$Z_R = \sqrt{R^2 + X_C^2} = 560.6 \text{ k}\Omega \text{ (using actual resistor)}$$

$$I_{PEAK} = \frac{V_{PEAK}}{Z_R} = 302.7 \times 10^{-6}$$

$$V_C = X_C \times I_{peak} = 8.0 \text{ V}$$

$$\Phi = \tan^{-1}\left(\frac{X_C}{R}\right) = 0.047 \text{ radians}$$

$$T_\Phi = \frac{\Phi}{2\pi f} = 125.6 \mu\text{s}$$

20.5.2 CORRECTION BY OFFSET CURRENT

When the waveform is varying relative to V_{SS} , then the zero cross is detected too early as the waveform falls and too late as the waveform rises. When the waveform is varying relative to V_{DD} , then the zero cross is detected too late as the waveform rises and too early as the waveform falls. The actual offset time can be determined for sinusoidal waveforms with the corresponding equations shown in [Equation 20-3](#).

EQUATION 20-3: ZCD EVENT OFFSET

When External Voltage Source is relative to V_{SS} :

$$T_{OFFSET} = \frac{\text{asin}\left(\frac{V_{Cpinv}}{V_{PEAK}}\right)}{2\pi \cdot \text{Freq}}$$

When External Voltage Source is relative to V_{DD} :

$$T_{OFFSET} = \frac{\text{asin}\left(\frac{V_{DD} - V_{Cpinv}}{V_{PEAK}}\right)}{2\pi \cdot \text{Freq}}$$

This offset time can be compensated for by adding a pull-up or pull-down biasing resistor to the ZCD pin. A pull-up resistor is used when the external voltage source is varying relative to V_{SS} . A pull-down resistor is used when the voltage is varying relative to V_{DD} . The resistor adds a bias to the ZCD pin so that the target external voltage source must go to zero to pull the pin voltage to the V_{Cpinv} switching voltage. The pull-up or pull-down value can be determined with the equations shown in [Equation 20-4](#) or [Equation 20-5](#).

EQUATION 20-4: ZCD PULL-UP/DOWN

When External Signal is relative to V_{SS} :

$$R_{PULLUP} = \frac{R_{SERIES}(V_{PULLUP} - V_{Cpinv})}{V_{Cpinv}}$$

When External Signal is relative to V_{DD} :

$$R_{PULLDOWN} = \frac{R_{SERIES}(V_{Cpinv})}{(V_{DD} - V_{Cpinv})}$$

20.6 Handling VPEAK variations

If the peak amplitude of the external voltage is expected to vary, the series resistor must be selected to keep the ZCD current source and sink below the design maximum range of $\pm 600 \mu\text{A}$ and above a reasonable minimum range. A general rule of thumb is that the maximum peak voltage can be no more than six times the minimum peak voltage. To ensure that the maximum current does not exceed $\pm 600 \mu\text{A}$ and the minimum is at least $\pm 100 \mu\text{A}$, compute the series resistance as shown in [Equation 20-5](#). The compensating pull-up for this series resistance can be determined with [Equation 20-4](#) because the pull-up value is independent from the peak voltage.

EQUATION 20-5: SERIES R FOR V RANGE

$$R_{SERIES} = \frac{V_{MAXPEAK} + V_{MINPEAK}}{7 \times 10^{-4}}$$

20.7 Operation During Sleep

The ZCD current sources and interrupts are unaffected by Sleep.

20.8 Effects of a Reset

The ZCD circuit can be configured to default to the active or inactive state on Power-On-Reset (POR). When the $\overline{\text{ZCD}}$ Configuration bit is cleared, the ZCD circuit will be active at POR. When the $\overline{\text{ZCD}}$ Configuration bit is set, the ZCDxEN bit of the ZCDxCON register must be set to enable the ZCD module.

20.9 Register Definitions: ZCD Control

REGISTER 20-1: ZCDxCON: ZERO-CROSS DETECTION CONTROL REGISTER

| | | | | | | | |
|---------|-----|---------|---------|-----|-----|----------|----------|
| R/W-q/q | U-0 | R-x/x | R/W-0/0 | U-0 | U-0 | R/W-0/0 | R/W-0/0 |
| ZCDxEN | — | ZCDxOUT | ZCDxPOL | — | — | ZCDxINTP | ZCDxINTN |
| 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 configuration bits |

- bit 7 **ZCDxEN:** Zero-Cross Detection Enable bit
 1 = Zero-cross detect is enabled. ZCD pin is forced to output to source and sink current.
 0 = Zero-cross detect is disabled. ZCD pin operates according to PPS and TRIS controls.
- bit 6 **Unimplemented:** Read as '0'
- bit 5 **ZCDxOUT:** Zero-Cross Detection Logic Level bit
 ZCDxPOL bit = 0:
 1 = ZCD pin is sinking current
 0 = ZCD pin is sourcing current
 ZCDxPOL bit = 1:
 1 = ZCD pin is sourcing current
 0 = ZCD pin is sinking current
- bit 4 **ZCDxPOL:** Zero-Cross Detection Logic Output Polarity bit
 1 = ZCD logic output is inverted
 0 = ZCD logic output is not inverted
- bit 3-2 **Unimplemented:** Read as '0'
- bit 1 **ZCDxINTP:** Zero-Cross Positive Edge Interrupt Enable bit
 1 = ZCDIF bit is set on low-to-high ZCDx_output transition
 0 = ZCDIF bit is unaffected by low-to-high ZCDx_output transition
- bit 0 **ZCDxINTN:** Zero-Cross Negative Edge Interrupt Enable bit
 1 = ZCDIF bit is set on high-to-low ZCDx_output transition
 0 = ZCDIF bit is unaffected by high-to-low ZCDx_output transition

TABLE 20-1: SUMMARY OF REGISTERS ASSOCIATED WITH THE ZCD MODULE

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Register on page |
|---------|--------|-------|---------|---------|--------|--------|----------|----------|------------------|
| PIE3 | — | — | CWGIE | ZCDIE | CLC4IE | CLC3IE | CLC2IE | CLC1IE | 108 |
| PIR3 | — | — | CWGIF | ZCDIF | CLC4IF | CLC3IF | CLC2IF | CLC1IF | 113 |
| ZCD1CON | ZCD1EN | — | ZCD1OUT | ZCD1POL | — | — | ZCD1INTP | ZCD1INTN | 220 |

Legend: — = unimplemented, read as '0'. Shaded cells are unused by the ZCD module.

TABLE 20-2: SUMMARY OF CONFIGURATION WORD WITH THE ZCD MODULE

| 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 | PLLEN | 69 |
| | 7:0 | ZCD | — | — | — | — | PPS1WAY | WRT<1:0> | | |

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

21.0 TIMER0 MODULE

The Timer0 module is an 8-bit timer/counter with the following features:

- 8-bit timer/counter register (TMR0)
- 3-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

Figure 21-1 is a block diagram of the Timer0 module.

21.1 Timer0 Operation

The Timer0 module can be used as either an 8-bit timer or an 8-bit counter.

21.1.1 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.

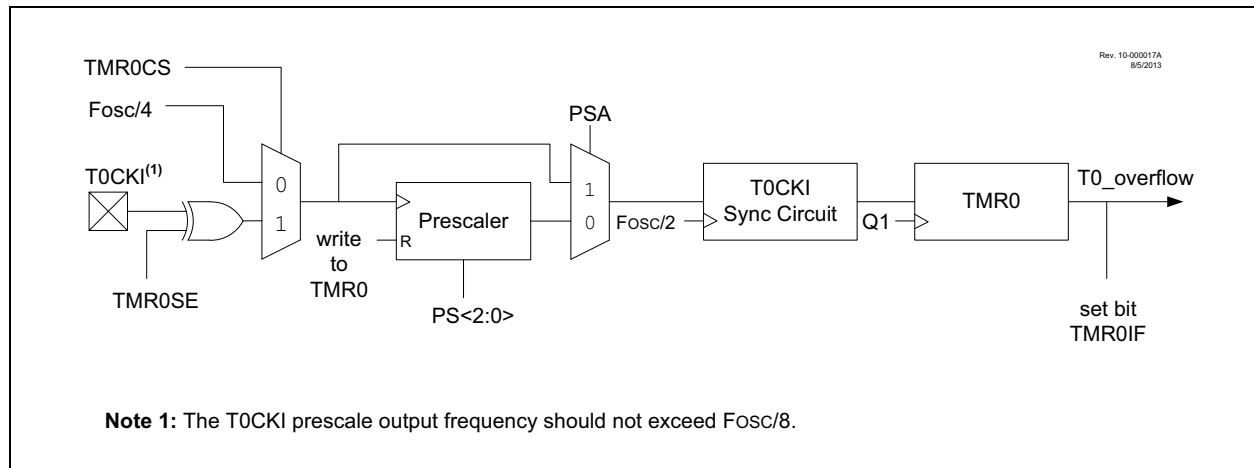
21.1.2 8-BIT COUNTER MODE

In 8-Bit Counter mode, the Timer0 module will increment on every rising or falling edge of the T0CKI pin.

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 21-1: TIMER0 BLOCK DIAGRAM



21.1.3 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 eight 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.

21.1.4 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.

21.1.5 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 35.0 “Electrical Specifications”](#).

21.1.6 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.

21.2 Register Definitions: Option Register

REGISTER 21-1: 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 WPUx 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 21-1: 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 |
|------------|---|--------|--------|--------|------------------|---------|--------|--------|------------------|
| ADCON2 | TRIGSEL<4:0> | | | | | — | — | — | 197 |
| INTCON | GIE | PEIE | TMR0IE | INTE | IOCFIE | TMR0IF | INTF | IOCF | 97 |
| OPTION_REG | $\overline{\text{WPUEN}}$ | INTEDG | TMR0CS | TMR0SE | PSA | PS<2:0> | | | 223 |
| TMR0 | Holding Register for the 8-bit Timer0 Count | | | | | | | | 221* |
| TRISA | — | — | TRISA5 | TRISA4 | — ⁽¹⁾ | TRISA2 | TRISA1 | TRISA0 | 151 |

Legend: — = Unimplemented location, read as '0'. Shaded cells are not used by the Timer0 module.

* Page provides register information.

Note 1: Unimplemented, read as '1'.

22.0 TIMER1/3/5 MODULE WITH GATE CONTROL

The Timer1/3/5 modules are a 16-bit timers/counters with the following features:

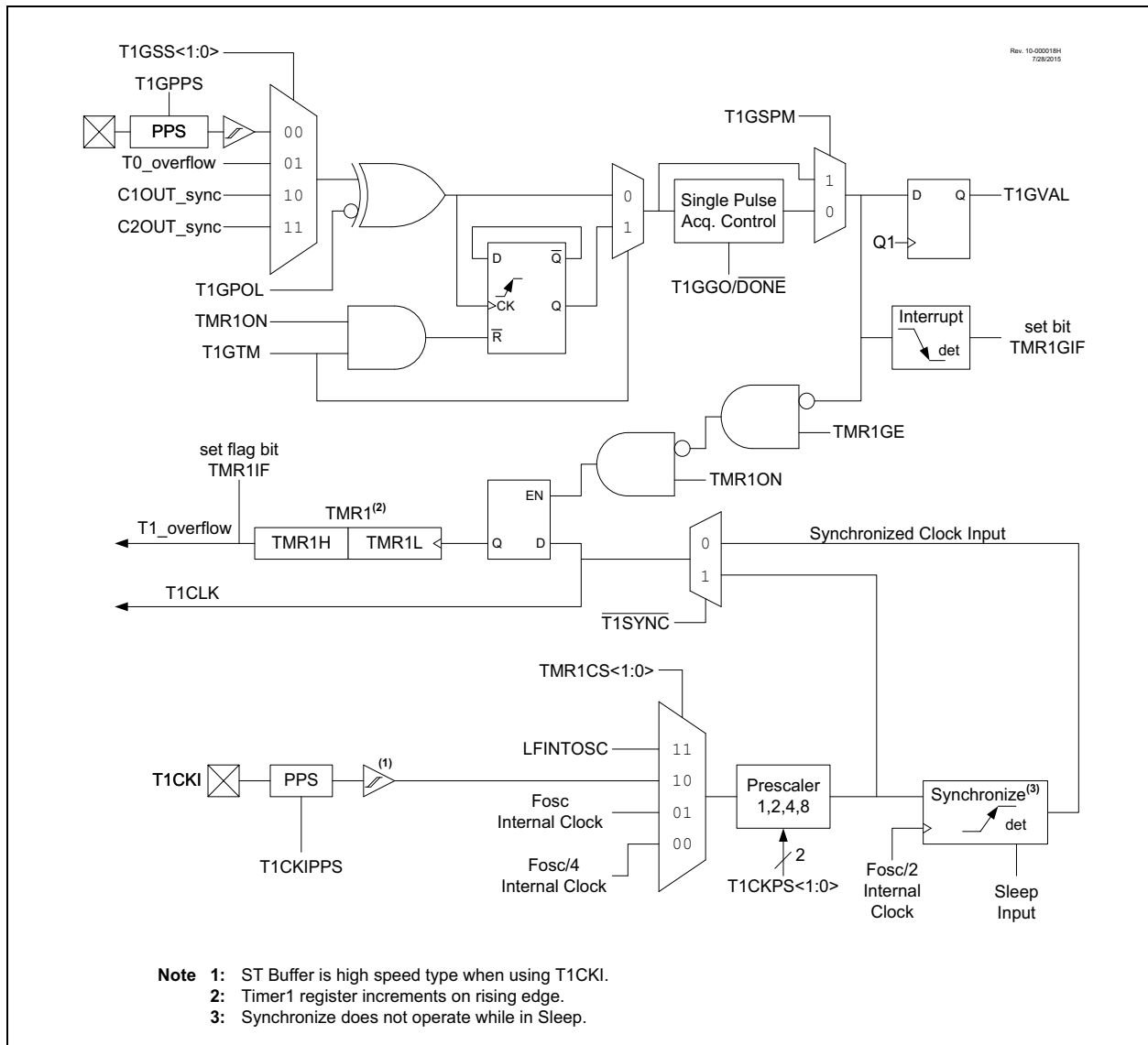
- 16-bit timer/counter register pair (TMR1H:TMR1L)
- Programmable internal or external clock source
- 2-bit prescaler
- Optionally synchronized comparator out
- Multiple Timer1 gate (count enable) sources
- Interrupt on overflow
- Wake-up on overflow (external clock, Asynchronous mode only)
- ADC Auto-Conversion Trigger(s)

- Selectable Gate Source Polarity
- Gate Toggle mode
- Gate Single-Pulse mode
- Gate Value Status
- Gate Event Interrupt

Figure 22-1 is a block diagram of the Timer1 module.

Note: Three identical Timer1 modules are implemented on this device. The timers are named Timer1, Timer3, and Timer5. All references to Timer1 apply as well to Timer3 and Timer5, as well as references to their associated registers.

FIGURE 22-1: TIMER1 BLOCK DIAGRAM



22.1 Timer1 Operation

The Timer1 module is a 16-bit incrementing counter which is accessed through the TMR1H:TMR1L register pair. Writes to TMR1H or TMR1L 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 is enabled by configuring the TMR1ON and TMR1GE bits in the T1CON and T1GCON registers, respectively. Table 22-1 displays the Timer1 enable selections.

TABLE 22-1: TIMER1 ENABLE SELECTIONS

| TMR1ON | TMR1GE | Timer1 Operation |
|--------|--------|------------------|
| 0 | 0 | Off |
| 0 | 1 | Off |
| 1 | 0 | Always On |
| 1 | 1 | Count Enabled |

22.2 Clock Source Selection

The TMR1CS<1:0> bits of the T1CON register are used to select the clock source for Timer1. Table 22-2 displays the clock source selections.

22.2.1 INTERNAL CLOCK SOURCE

When the internal clock source is selected, the TMR1H:TMR1L register pair will increment on multiples of FOSC as determined by the Timer1 prescaler.

When the FOSC internal clock source is selected, the Timer1 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 value. To utilize the full resolution of Timer1, an asynchronous input signal must be used to gate the Timer1 clock input.

The following asynchronous sources may be used:

- Asynchronous event on the T1G pin to Timer1 gate
- C1 or C2 comparator input to Timer1 gate

22.2.2 EXTERNAL CLOCK SOURCE

When the external clock source is selected, the Timer1 module may work as a timer or a counter.

When enabled to count, Timer1 is incremented on the rising edge of the external clock input T1CKI. The external clock source can be synchronized to the microcontroller system clock or it can run asynchronously.

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 enabled after POR
- Write to TMR1H or TMR1L
- Timer1 is disabled
- Timer1 is disabled (TMR1ON = 0) when T1CKI is high then Timer1 is enabled (TMR1ON=1) when T1CKI is low.

TABLE 22-2: CLOCK SOURCE SELECTIONS

| TMR1CS<1:0> | Clock Source |
|-------------|--------------------------------|
| 11 | LFINTOSC |
| 10 | External Clocking on T1CKI Pin |
| 01 | System Clock (FOSC) |
| 00 | Instruction Clock (FOSC/4) |

22.3 Timer1 Prescaler

Timer1 has four prescaler options allowing 1, 2, 4 or 8 divisions of the clock input. The T1CKPS bits of the T1CON register control the prescale counter. The prescale counter is not directly readable or writable; however, the prescaler counter is cleared upon a write to TMR1H or TMR1L.

22.4 Timer1 Operation in Asynchronous Counter Mode

If control bit T1SYN \bar{C} of the T1CON 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 22.4.1 “Reading and Writing Timer1 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.

22.4.1 READING AND WRITING TIMER1 IN ASYNCHRONOUS COUNTER MODE

Reading TMR1H or TMR1L 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.

TABLE 22-4: TIMER1 GATE SOURCES

| T1GSS | Timer1 Gate Source |
|-------|---|
| 00 | Timer1 Gate pin (T1G) |
| 01 | Overflow of Timer0 (T0_overflow) (TMR0 increments from FFh to 00h) |
| 10 | Comparator 1 Output (C1_OUT_sync) ⁽¹⁾ |
| 11 | Comparator 2 Output (C2_OUT_sync) ⁽¹⁾ |

Note 1: Optionally synchronized comparator output.

22.5.2.1 T1G Pin Gate Operation

The T1G pin is one source for Timer1 gate control. It can be used to supply an external source to the Timer1

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 TMR1H:TMR1L register pair.

22.5 Timer1 Gate

Timer1 can be configured to count freely or the count can be enabled and disabled using Timer1 gate circuitry. This is also referred to as Timer1 Gate Enable.

Timer1 gate can also be driven by multiple selectable sources.

22.5.1 TIMER1 GATE ENABLE

The Timer1 Gate Enable mode is enabled by setting the TMR1GE bit of the T1GCON register. The polarity of the Timer1 Gate Enable mode is configured using the T1GPOL bit of the T1GCON register.

When Timer1 Gate Enable mode is enabled, Timer1 will increment on the rising edge of the Timer1 clock source. When Timer1 Gate Enable mode is disabled, no incrementing will occur and Timer1 will hold the current count. See [Figure 22-3](#) for timing details.

TABLE 22-3: TIMER1 GATE ENABLE SELECTIONS

| T1CLK | T1GPOL | T1G | Timer1 Operation |
|-------|--------|-----|------------------|
| ↑ | 0 | 0 | Counts |
| ↑ | 0 | 1 | Holds Count |
| ↑ | 1 | 0 | Holds Count |
| ↑ | 1 | 1 | Counts |

22.5.2 TIMER1 GATE SOURCE SELECTION

Timer1 gate source selections are shown in [Table 22-4](#). Source selection is controlled by the T1GSS<1:0> bits of the T1GCON register. The polarity for each available source is also selectable. Polarity selection is controlled by the T1GPOL bit of the T1GCON register.

gate circuitry.

22.5.2.2 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 gate circuitry.

22.5.3 TIMER1 GATE TOGGLE MODE

When Timer1 Gate Toggle mode is enabled, it is possible to measure the full-cycle length of a Timer1 gate signal, as opposed to the duration of a single level pulse.

The Timer1 gate source is routed through a flip-flop that changes state on every incrementing edge of the signal. See [Figure 22-4](#) for timing details.

Timer1 Gate Toggle mode is enabled by setting the T1GTM bit of the T1GCON register. When the T1GTM 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. |
|---|

22.5.4 TIMER1 GATE SINGLE-PULSE MODE

When Timer1 Gate Single-Pulse mode is enabled, it is possible to capture a single pulse gate event. Timer1 Gate Single-Pulse mode is first enabled by setting the T1GSPM bit in the T1GCON register. Next, the T1GGO/DONE bit in the T1GCON register must be set. The Timer1 will be fully enabled on the next incrementing edge. On the next trailing edge of the pulse, the T1GGO/DONE bit will automatically be cleared. No other gate events will be allowed to increment Timer1 until the T1GGO/DONE bit is once again set in software. See [Figure 22-5](#) for timing details.

If the Single Pulse Gate mode is disabled by clearing the T1GSPM bit in the T1GCON register, the T1GGO/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 gate source to be measured. See [Figure 22-6](#) for timing details.

22.5.5 TIMER1 GATE VALUE STATUS

When Timer1 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 T1GVAL bit in the T1GCON register. The T1GVAL bit is valid even when the Timer1 gate is not enabled (TMR1GE bit is cleared).

22.5.6 TIMER1 GATE EVENT INTERRUPT

When Timer1 Gate Event Interrupt is enabled, it is possible to generate an interrupt upon the completion of a gate event. When the falling edge of T1GVAL occurs, the TMR1GIF flag bit in the PIR1 register will be set. If the TMR1GIE bit in the PIE1 register is set, then an interrupt will be recognized.

The TMR1GIF flag bit operates even when the Timer1 gate is not enabled (TMR1GE bit is cleared).

22.6 Timer1 Interrupt

The Timer1 register pair (TMR1H:TMR1L) increments to FFFFh and rolls over to 0000h. When Timer1 rolls over, the Timer1 interrupt flag bit of the PIR1 register is set. To enable the interrupt on rollover, the following bits must be set:

- TMR1ON bit of the T1CON register
- TMR1IE bit of the PIE1 register
- PEIE bit of the INTCON register
- GIE bit of the INTCON register

The interrupt is cleared by clearing the TMR1IF bit in the Interrupt Service Routine.

Note: The TMR1H:TMR1L register pair and the TMR1IF bit should be cleared before enabling interrupts.

Timer1 oscillator will continue to operate in Sleep regardless of the $\overline{T1SYNC}$ bit setting.

22.7 Timer1 Operation During Sleep

Timer1 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:

- TMR1ON bit of the T1CON register must be set
- TMR1IE bit of the PIE1 register must be set
- PEIE bit of the INTCON register must be set
- $\overline{T1SYNC}$ bit of the T1CON register must be set
- TMR1CS bits of the T1CON 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.

FIGURE 22-2: TIMER1 INCREMENTING EDGE

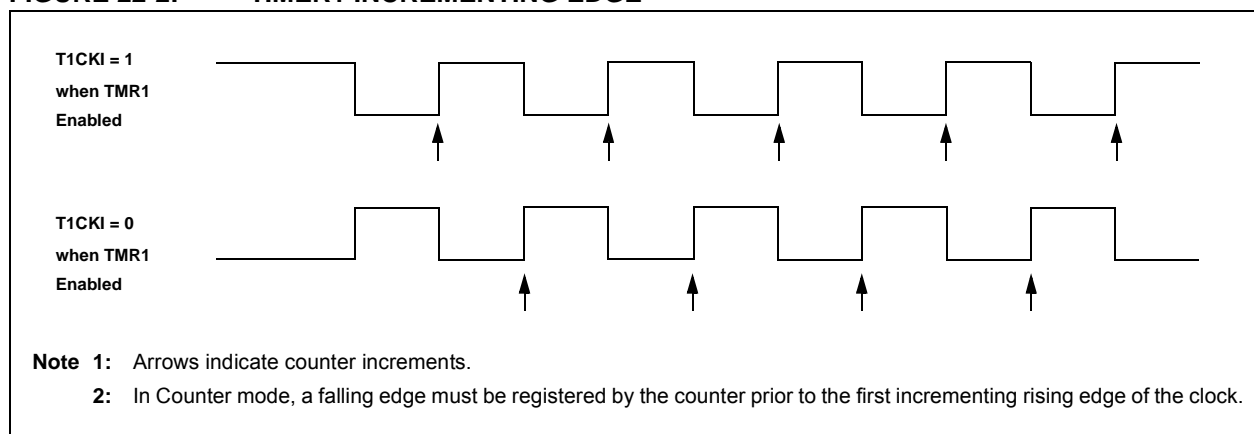


FIGURE 22-3: TIMER1 GATE ENABLE MODE

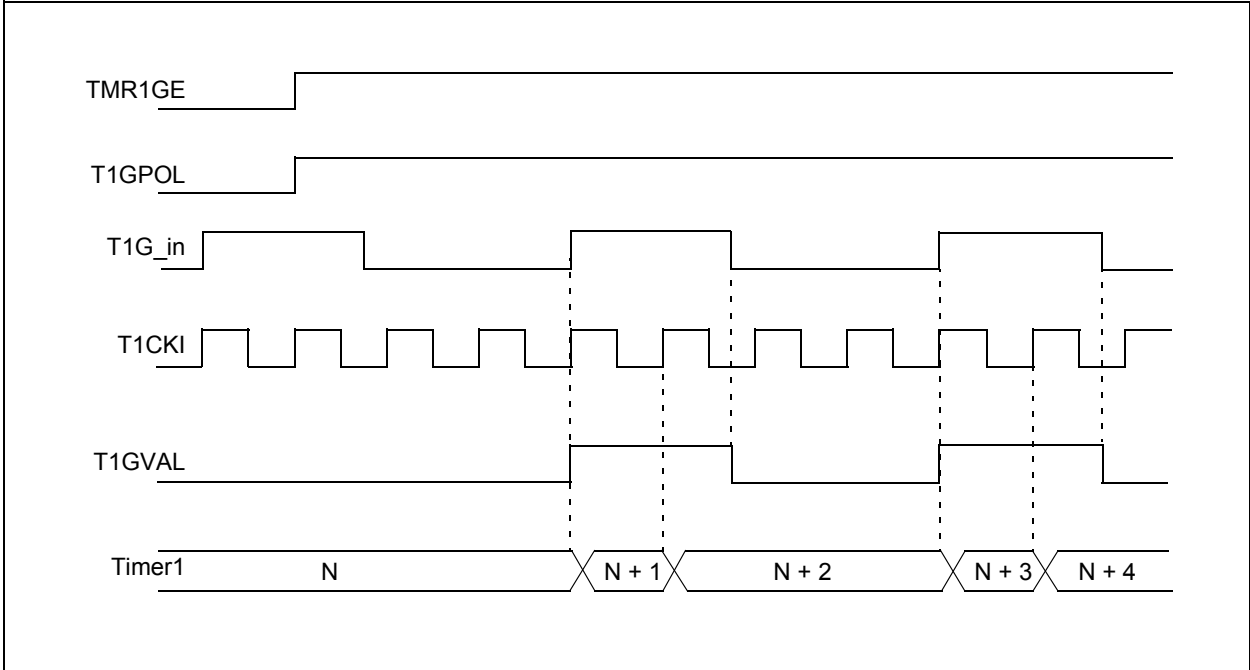


FIGURE 22-4: TIMER1 GATE TOGGLE MODE

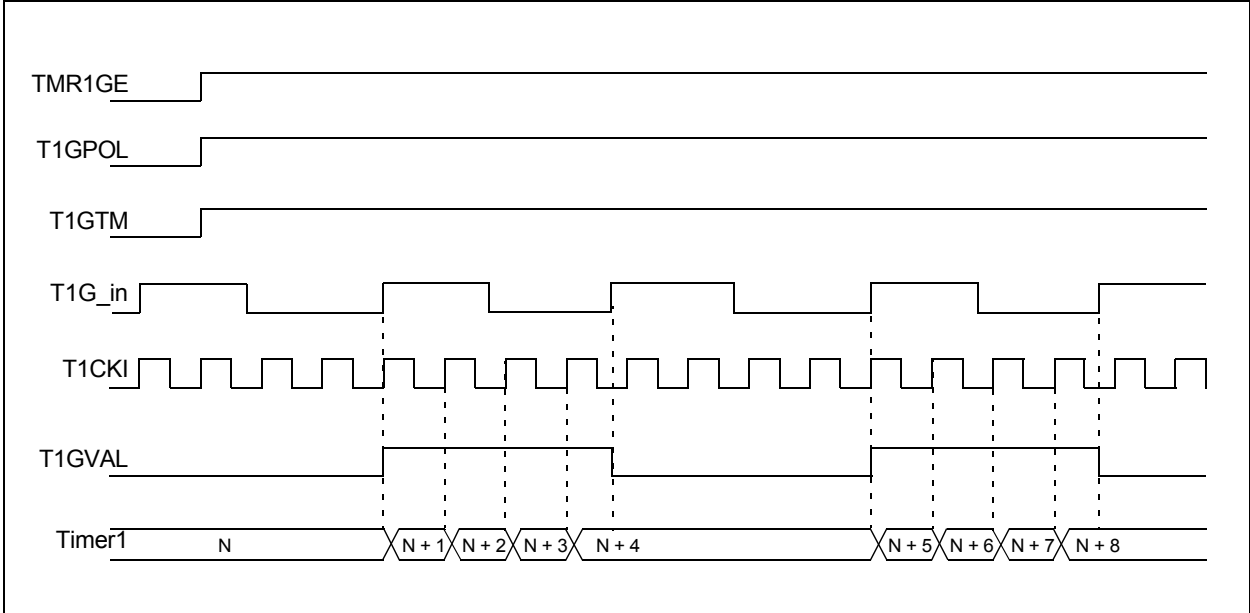


FIGURE 22-5: TIMER1 GATE SINGLE-PULSE MODE

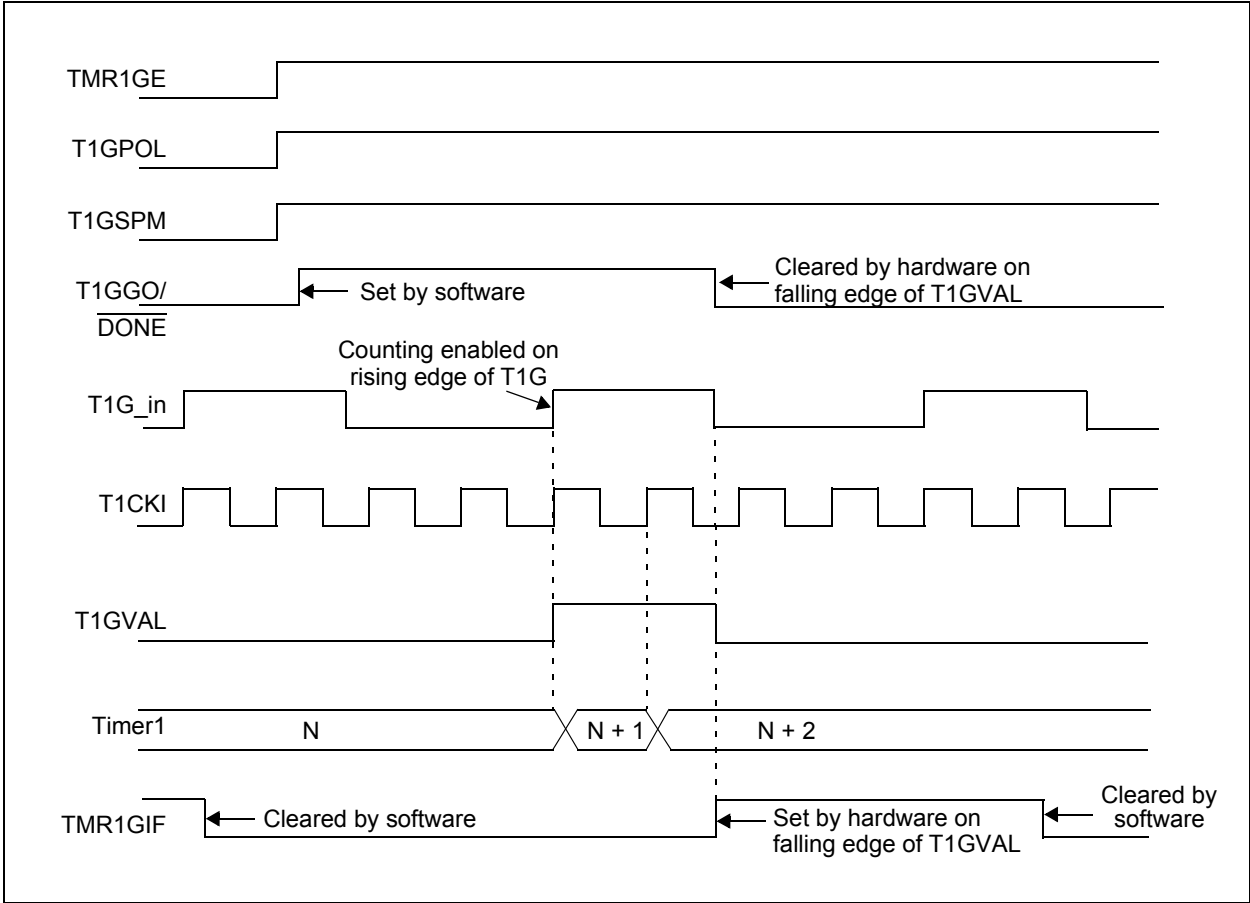
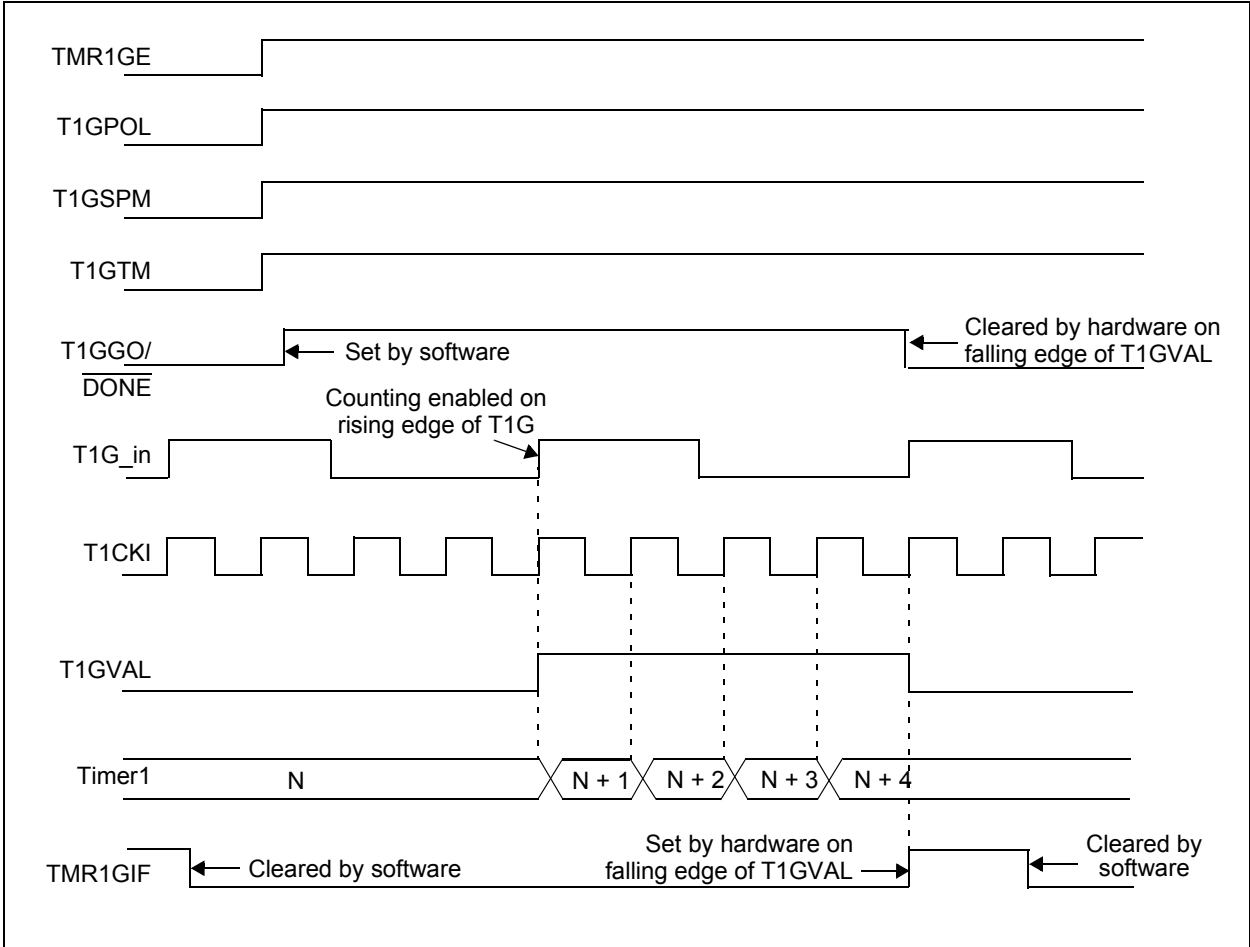


FIGURE 22-6: TIMER1 GATE SINGLE-PULSE AND TOGGLE COMBINED MODE



22.8 Register Definitions: Timer1 Control

REGISTER 22-1: T1CON: TIMER1 CONTROL REGISTER

| | | | | | | | |
|-------------|---------|-------------|---------|-----|---------------------|-----|---------|
| R/W-0/u | R/W-0/u | R/W-0/u | R/W-0/u | U-0 | R/W-0/u | U-0 | R/W-0/u |
| TMR1CS<1:0> | | T1CKPS<1:0> | | — | $\overline{T1SYNC}$ | — | TMR1ON |
| 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 **TMR1CS<1:0>**: Timer1 Clock Source Select bits

- 11 = LFINTOSC
- 10 = T1CKI
- 01 = Fosc
- 00 = Fosc/4

bit 5-4 **T1CKPS<1:0>**: Timer1 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 **Unimplemented**: Read as '0'

bit 2 **T1SYNC**: Timer1 Synchronization Control bit

- 1 = Do not synchronize asynchronous clock input
- 0 = Synchronize asynchronous clock input with system clock (Fosc)

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

bit 0 **TMR1ON**: Timer1 On bit

- 1 = Enables Timer1
- 0 = Stops Timer1 and clears Timer1 gate flip-flop

REGISTER 22-2: T1GCON: TIMER1 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 |
| TMR1GE | T1GPOL | T1GTM | T1GSPM | T1GGO/ DONE | T1GVAL | T1GSS<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 **TMR1GE:** Timer1 Gate Enable bit
If TMR1ON = 0:
This bit is ignored
If TMR1ON = 1:
1 = Timer1 counting is controlled by the Timer1 gate function
0 = Timer1 counts regardless of Timer1 gate function
- bit 6 **T1GPOL:** Timer1 Gate Polarity bit
1 = Timer1 gate is active-high (Timer1 counts when gate is high)
0 = Timer1 gate is active-low (Timer1 counts when gate is low)
- bit 5 **T1GTM:** Timer1 Gate Toggle Mode bit
1 = Timer1 Gate Toggle mode is enabled
0 = Timer1 Gate Toggle mode is disabled and toggle flip-flop is cleared
Timer1 gate flip-flop toggles on every rising edge.
- bit 4 **T1GSPM:** Timer1 Gate Single-Pulse Mode bit
1 = Timer1 gate Single-Pulse mode is enabled and is controlling Timer1 gate
0 = Timer1 gate Single-Pulse mode is disabled
- bit 3 **T1GGO/DONE:** Timer1 Gate Single-Pulse Acquisition Status bit
1 = Timer1 gate single-pulse acquisition is ready, waiting for an edge
0 = Timer1 gate single-pulse acquisition has completed or has not been started
- bit 2 **T1GVAL:** Timer1 Gate Value Status bit
Indicates the current state of the Timer1 gate that could be provided to TMR1H:TMR1L.
Unaffected by Timer1 Gate Enable (TMR1GE).
- bit 0 **T1GSS<1:0>:** Timer1 Gate Source Select bits
11 =Comparator 2 optionally synchronized output (C2_OUT_sync)
10 =Comparator 1 optionally synchronized output (C1_OUT_sync)
01 =Timer0 overflow output (T0_overflow)
00 =Timer1 gate pin (T1G)

TABLE 22-5: SUMMARY OF REGISTERS ASSOCIATED WITH TIMER1

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Register on Page |
|--------|--|--------|-------------|--------|------------------|--------|------------|--------|------------------|
| ANSELA | — | — | — | ANSA4 | — | ANSA2 | ANSA1 | ANSA0 | 152 |
| INTCON | GIE | PEIE | TMR0IE | INTE | IOCIE | TMR0IF | INTF | IOCIF | 97 |
| PIE1 | TMR1GIE | ADIE | RCIE | TXIE | SSP1IE | CCP1IE | TMR2IE | TMR1IE | 98 |
| PIR1 | TMR1GIF | ADIF | RCIF | TXIF | SSP1IF | CCP1IF | TMR2IF | TMR1IF | 103 |
| TMR1H | Holding Register for the Most Significant Byte of the 16-bit TMR1 Count | | | | | | | | 228* |
| TMR1L | Holding Register for the Least Significant Byte of the 16-bit TMR1 Count | | | | | | | | 228* |
| TMR3H | Holding Register for the Most Significant Byte of the 16-bit TMR3 Count | | | | | | | | 228* |
| TMR3L | Holding Register for the Least Significant Byte of the 16-bit TMR3 Count | | | | | | | | 228* |
| TMR5H | Holding Register for the Most Significant Byte of the 16-bit TMR5 Count | | | | | | | | 228* |
| TMR5L | Holding Register for the Least Significant Byte of the 16-bit TMR5 Count | | | | | | | | 228* |
| TRISA | — | — | TRISA5 | TRISA4 | — ⁽¹⁾ | TRISA2 | TRISA1 | TRISA0 | 151 |
| T1CON | TMR1CS<1:0> | | T1CKPS<1:0> | | — | T1SYNC | — | TMR1ON | 232 |
| T1GCON | TMR1GE | T1GPOL | T1GTM | T1GSPM | T1GGO/ DONE | T1GVAL | T1GSS<1:0> | | 233 |
| T3CON | TMR3CS<1:0> | | T3CKPS<1:0> | | — | T3SYNC | — | TMR3ON | 232 |
| T3GCON | TMR3GE | T3GPOL | T3GTM | T3GSPM | T3GGO/ DONE | T3GVAL | T3GSS<1:0> | | 233 |
| T5CON | TMR5CS<1:0> | | T5CKPS<1:0> | | — | T5SYNC | — | TMR5ON | 232 |
| T5GCON | TMR5GE | T5GPOL | T5GTM | T5GSPM | T5GGO/ DONE | T5GVAL | T5GSS<1:0> | | 233 |

Legend: — = unimplemented location, read as '0'. Shaded cells are not used by the Timer1 module.

* Page provides register information.

Note 1: Unimplemented, read as '1'.

23.0 TIMER2/4/6 MODULE

The Timer2/4/6 modules are 8-bit timers that can operate as free-running period counters or in conjunction with external signals that control start, run, freeze, and reset operation in One-Shot and Monostable modes of operation. Sophisticated waveform control such as pulse density modulation are possible by combining the operation of these timers with other internal peripherals such as the comparators and CCP modules. Features of the timer include:

- 8-bit timer register
- 8-bit period register
- Selectable external hardware timer Resets
- Programmable prescaler (1:1 to 1:128)
- Programmable postscaler (1:1 to 1:16)
- Selectable synchronous/asynchronous operation
- Alternate clock sources
- Interrupt-on-period

- Three modes of operation:
 - Free Running Period
 - One-shot
 - Monostable

See [Figure 23-1](#) for a block diagram of Timer2. See [Figure 23-2](#) for the clock source block diagram.

Note: Three identical Timer2 modules are implemented on this device. The timers are named Timer2, Timer4, and Timer6. All references to Timer2 apply as well to Timer4 and Timer6. All references to T2PR apply as well to T4PR and T6PR.

FIGURE 23-1: TIMER2 BLOCK DIAGRAM

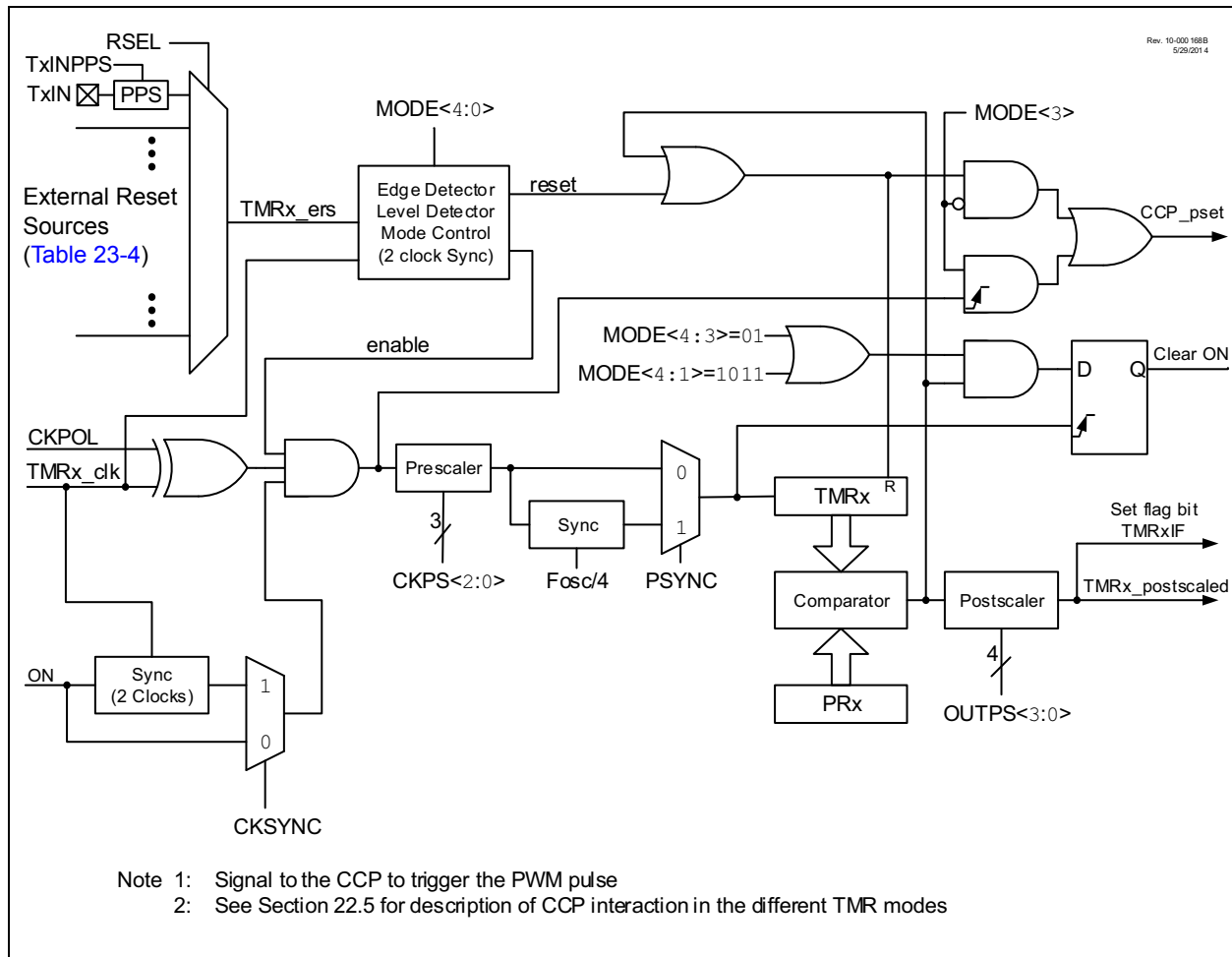
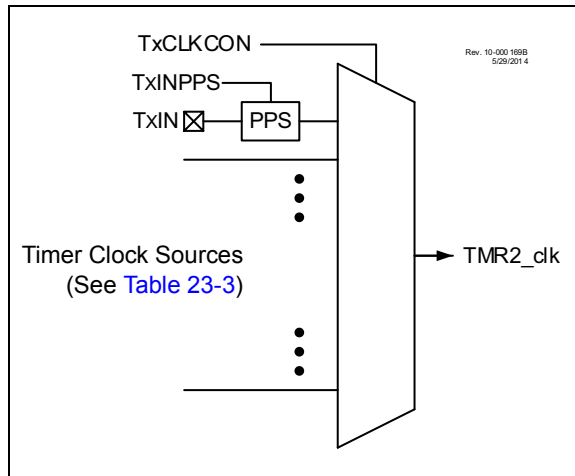


FIGURE 23-2: TIMER2 CLOCK SOURCE BLOCK DIAGRAM



23.1 Timer2 Operation

Timer2 operates in three major modes:

- Free Running Period
- One-shot
- Monostable

Within each mode there are several options for starting, stopping, and reset. [Table 23-1](#) lists the options.

In all modes, the TMR2 count register is incremented on the rising edge of the clock signal from the programmable prescaler. When TMR2 equals T2PR, a high level is output to the postscaler counter. TMR2 is cleared on the next clock input.

An external signal from hardware can also be configured to gate the timer operation or force a TMR2 count Reset. In Gate modes the counter stops when the gate is disabled and resumes when the gate is enabled. In Reset modes the TMR2 count is reset on either the level or edge from the external source.

The TMR2 register is directly readable and writable. The TMR2 register is cleared on any device Reset. The T2PR register is double-buffered and initializes to 0xFF on any device Reset. The SFR is directly readable and writable, but the actual period buffer is only updated with the SFR value when the following events occur:

- a write to the TMR2 register
- a write to the T2CON register
- a write to the T2HLT register
- TMR2 = T2PR and the prescaler is full
- External Reset Source event that resets the timer

Both the prescaler and postscaler counters are cleared on the following events:

- a write to the TMR2 register
- a write to the T2CON register
- any device Reset
- External Reset Source event that resets the timer.

Note: TMR2 is not cleared when T2CON is written.

23.1.1 FREE RUNNING PERIOD MODE

The value of TMR2 is compared to that of the Period register, T2PR, on each clock cycle. When the two values match, the comparator resets the value of TMR2 to 00h on the next cycle and increments the output postscaler counter. When the postscaler count equals the value in the OUTPS<4:0> bits of the TMRxCON1 register then a one clock period wide pulse occurs on the TMR2_postscaled output, and the postscaler count is cleared.

23.1.2 ONE-SHOT MODE

The One-Shot mode is identical to the Free Running Period mode except that the ON bit is cleared and the timer is stopped when TMR2 matches T2PR and will not restart until the T2ON bit is cycled off and on. Postscaler OUTPS<4:0> values other than 0 are meaningless in this mode because the timer is stopped at the first period event and the postscaler is reset when the timer is restarted.

23.1.3 MONOSTABLE MODE

Monostable modes are similar to One-Shot modes except that the ON bit is not cleared and the timer can be restarted by an external Reset event.

23.2 PRx Period Register

The PRx period register is double buffered. Software reads and writes the PRx register. However, the timer uses a buffered PRx register for operation. Software does not have direct access to the buffered PRx register. The contents of the PRx register is transferred to the buffer by any of the following events:

- A write to the TMRx register
- A write to the TMRxCON register
- When TMRx = PRx buffer and the prescaler rolls over
- An external Reset event

23.3 Timer2 Output

The Timer2 module's primary output is TMR2_postscaled, which pulses for a single TMR2_clk period when the postscaler counter matches the value in the OUTPS bits of the TMR2xCON register. The T2PR postscaler is incremented each time the TMR2 value matches the T2PR value. This signal can be selected as an input to several other input modules:

- The ADC module, as an Auto-conversion Trigger
- CWG, as an auto-shutdown source

In addition, the Timer2 is also used by the CCP module for pulse generation in PWM mode. Both the actual TMR2 value as well as other internal signals are sent to the CCP module to properly clock both the period and pulse width of the PWM signal. See [Section 26.4 “CCP/PWM Clock Selection”](#) for more details on setting up Timer2 for use with the CCP, as well as the timing diagrams in [Section 23.6 “Operation Examples”](#) for examples of how the varying Timer2 modes affect CCP PWM output.

23.4 External Reset Sources

In addition to the clock source, the Timer2 also takes in an external Reset source. This external Reset source is selected for Timer2, Timer4, and Timer6 with the T2RST, T4RST, and T6RST registers, respectively. This source can control starting and stopping of the timer, as well as resetting the timer, depending on which mode the timer is in. The mode of the timer is controlled by the MODE<4:0> bits of the TMRxHLT register. Edge-Triggered modes require six Timer clock periods between external triggers. Level-Triggered modes require the triggering level to be at least three Timer clock periods long. External triggers are ignored while in Debug Freeze mode.

TABLE 23-1: TIMER2 OPERATING MODES

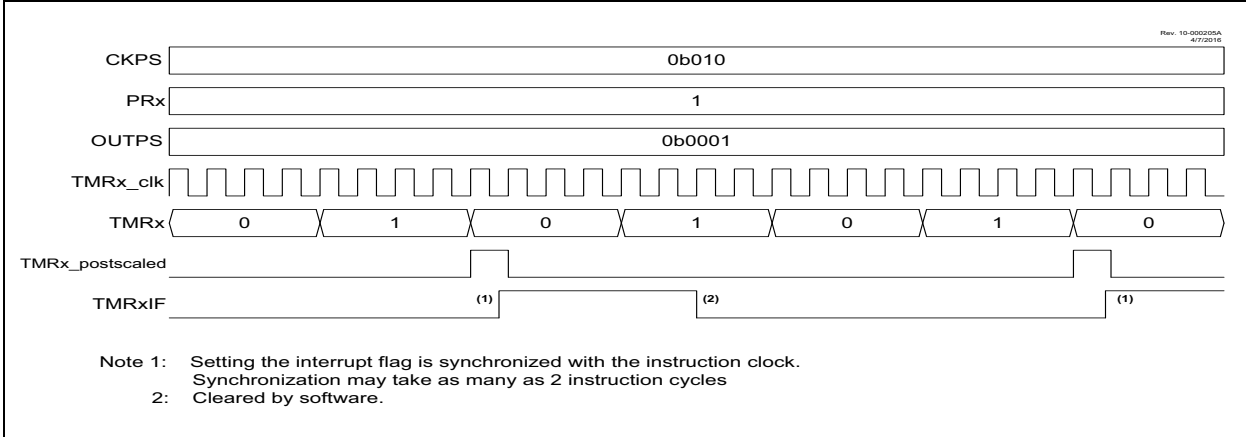
| Mode | MODE<4:0> | | Output Operation | Operation | Timer Control | | | |
|------------------------------------|-------------------------|----------|--|--|---|-------------------------|--|----------------------------------|
| | <4:3> | <2:0> | | | Start | Reset | Stop | |
| Free Running Period | 00 | 000 | Period Pulse | Software gate (Figure 23-4) | ON = 1 | — | ON = 0 | |
| | | 001 | | Hardware gate, active-high (Figure 23-5) | ON = 1 and TMRx_ers = 1 | — | ON = 0 or TMRx_ers = 0 | |
| | | 010 | | Hardware gate, active-low | ON = 1 and TMRx_ers = 0 | — | ON = 0 or TMRx_ers = 1 | |
| | | 011 | Period Pulse with Hardware Reset | Rising or falling edge Reset | ON = 1 | TMRx_ers ↓ | ON = 0 | |
| | | 100 | | Rising edge Reset (Figure 23-6) | | TMRx_ers ↑ | | |
| | | 101 | | Falling edge Reset | | TMRx_ers ↓ | | |
| | | 110 | | Low level Reset | | TMRx_ers = 0 | ON = 0 or TMRx_ers = 0 | |
| | | 111 | | High level Reset (Figure 23-7) | | TMRx_ers = 1 | ON = 0 or TMRx_ers = 1 | |
| One-shot | 01 | 000 | One-shot | Software start (Figure 23-8) | ON = 1 | — | ON = 0 or Next clock after TMRx = PRx (Note 2) | |
| | | 001 | Edge triggered start (Note 1) | Rising edge start (Figure 23-9) | ON = 1 and TMRx_ers ↑ | — | | |
| | | 010 | | Falling edge start | ON = 1 and TMRx_ers ↓ | — | | |
| | | 011 | | Any edge start | ON = 1 and TMRx_ers ↓ | — | | |
| | | 100 | Edge triggered start and hardware Reset (Note 1) | Rising edge start and Rising edge Reset (Figure 23-10) | ON = 1 and TMRx_ers ↑ | TMRx_ers ↑ | | |
| | | 101 | | Falling edge start and Falling edge Reset | ON = 1 and TMRx_ers ↓ | TMRx_ers ↓ | | |
| | | 110 | | Rising edge start and Low level Reset (Figure 23-11) | ON = 1 and TMRx_ers ↑ | TMRx_ers = 0 | | |
| | | 111 | | Falling edge start and High level Reset | ON = 1 and TMRx_ers ↓ | TMRx_ers = 1 | | |
| Mono-stable | 10 | 000 | Reserved | | | | | |
| | | 001 | Edge triggered start (Note 1) | Rising edge start (Figure 23-12) | ON = 1 and TMRx_ers ↑ | — | ON = 0 or Next clock after TMRx = PRx (Note 3) | |
| | | 010 | | Falling edge start | ON = 1 and TMRx_ers ↓ | — | | |
| | | 011 | | Any edge start | ON = 1 and TMRx_ers ↓ | — | | |
| | | Reserved | 100 | Reserved | | | | |
| | | Reserved | 101 | Reserved | | | | |
| | | One-shot | 110 | Level triggered start and hardware Reset | High level start and Low level Reset (Figure 23-13) | ON = 1 and TMRx_ers = 1 | TMRx_ers = 0 | ON = 0 or Held in Reset (Note 2) |
| Low level start & High level Reset | ON = 1 and TMRx_ers = 0 | | | | TMRx_ers = 1 | | | |
| Reserved | 11 | xxx | Reserved | | | | | |

- Note 1:** If ON = 0 then an edge is required to restart the timer after ON = 1.
Note 2: When TMRx = PRx then the next clock clears ON and stops TMRx at 00h.
Note 3: When TMRx = PRx then the next clock stops TMRx at 00h but does not clear ON.

23.5 Timer2 Interrupt

Timer2 can also generate a device interrupt. The interrupt is generated when the postscaler counter matches one of 16 postscale options (from 1:1 through 1:16), which are selected with the postscaler control bits, OUTPS<3:0> of the T2CON register. The interrupt is enabled by setting the TMR2IE interrupt enable bit of the PIE1 register. Interrupt timing is illustrated in Figure 23-3.

FIGURE 23-3: TIMER2 PRESCALER, POSTSCALER, AND INTERRUPT TIMING DIAGRAM



23.6 Operation Examples

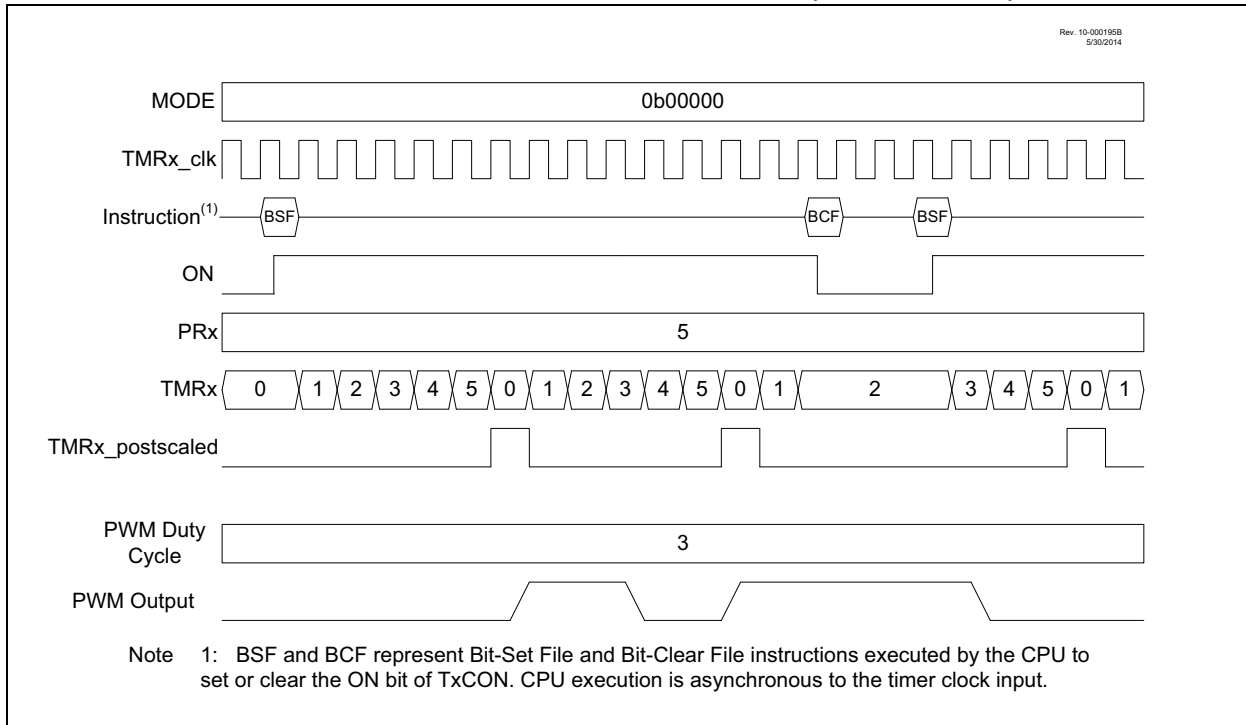
Unless otherwise specified, the following notes apply to the following timing diagrams:

- Both the prescaler and postscaler are set to 1:1 (both the CKPS and OUTPS bits in the TxCON register are cleared).
- The diagrams illustrate any clock except $F_{osc}/4$ and show clock-sync delays of at least two full cycles for both ON and Timer2_ers. When using $F_{osc}/4$, the clock-sync delay is at least one instruction period for Timer2_ers; ON applies in the next instruction period.
- The PWM Duty Cycle and PWM output are illustrated assuming that the timer is used for the PWM function of the CCP module as described in [Section 26.4 "CCP/PWM Clock Selection"](#). The signals are not a part of the Timer2 module.

23.6.1 SOFTWARE GATE MODE

This mode corresponds to legacy Timer2 operation. The timer increments with each clock input when $ON = 1$ and does not increment when $ON = 0$. When the TMRx count equals the PRx period count the timer resets on the next clock and continues counting from 0. Operation with the ON bit software controlled is illustrated in [Figure 23-4](#). With $PRx = 5$, the counter advances until $TMRx = 5$, and goes to zero with the next clock.

FIGURE 23-4: SOFTWARE GATE MODE TIMING DIAGRAM (MODE = 00000)



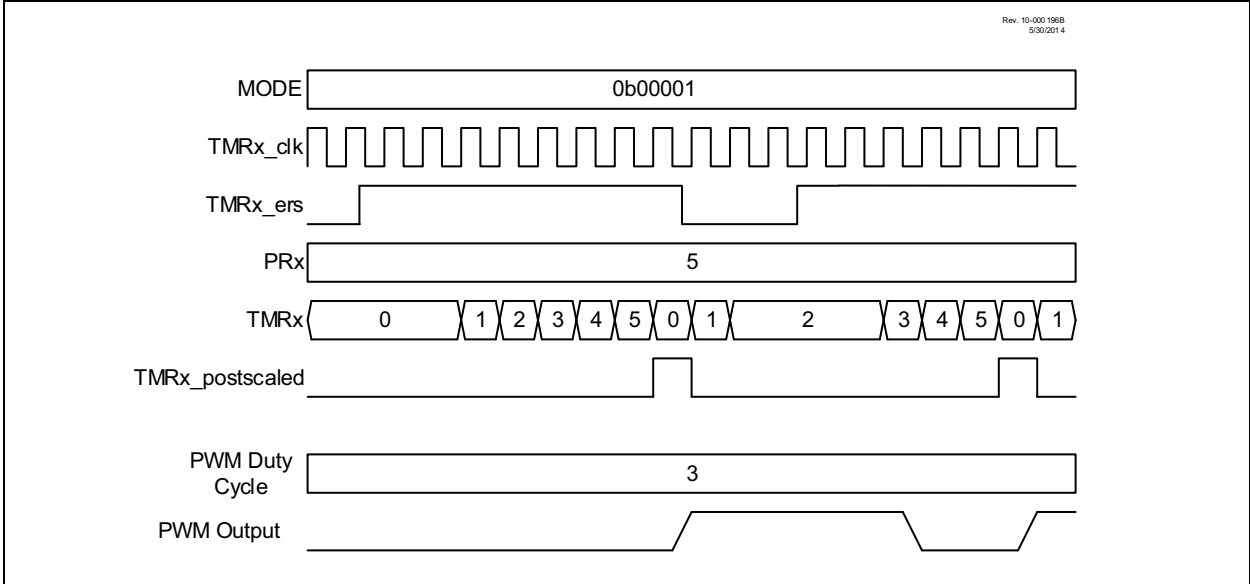
23.6.2 HARDWARE GATE MODE

The Hardware Gate modes operate the same as the Software Gate mode except the TMRx_ers external signal can also gate the timer. When used with the CCP the gating extends the PWM period. If the timer is stopped when the PWM output is high then the duty cycle is also extended.

When MODE<4:0> = 00001 then the timer is stopped when the external signal is high. When MODE<4:0> = 00010 then the timer is stopped when the external signal is low.

Figure 23-5 illustrates the Hardware Gating mode for MODE<4:0> = 00001 in which a high input level starts the counter.

FIGURE 23-5: HARDWARE GATE MODE TIMING DIAGRAM (MODE = 00001)



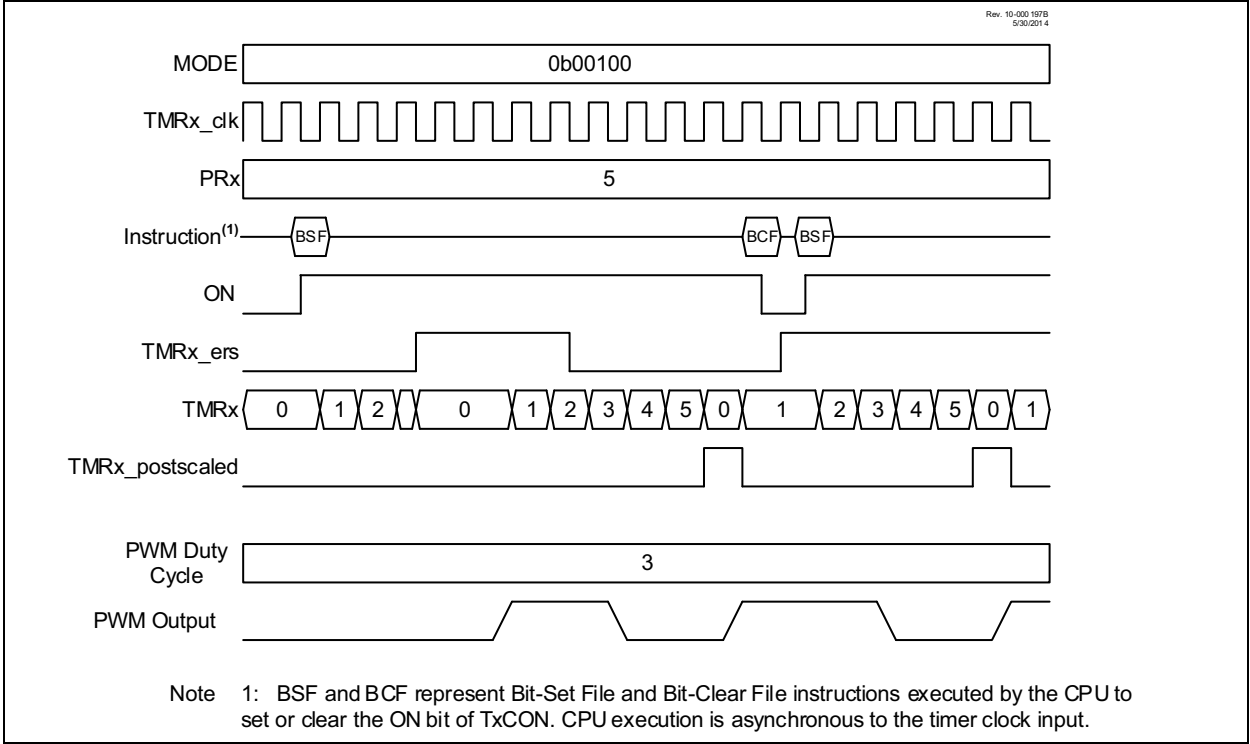
23.6.3 EDGE-TRIGGERED HARDWARE LIMIT MODE

In Hardware Limit mode the timer can be reset by the TMRx_ers external signal before the timer reaches the period count. Three types of Resets are possible:

- Reset on rising or falling edge (MODE<4:0>= 00011)
- Reset on rising edge (MODE<4:0> = 00100)
- Reset on falling edge (MODE<4:0> = 00101)

When the timer is used in conjunction with the CCP in PWM mode then an early Reset shortens the period and restarts the PWM pulse after a two-clock delay. Refer to Figure 23-6.

FIGURE 23-6: EDGE-TRIGGERED HARDWARE LIMIT MODE TIMING DIAGRAM (MODE = 00100)



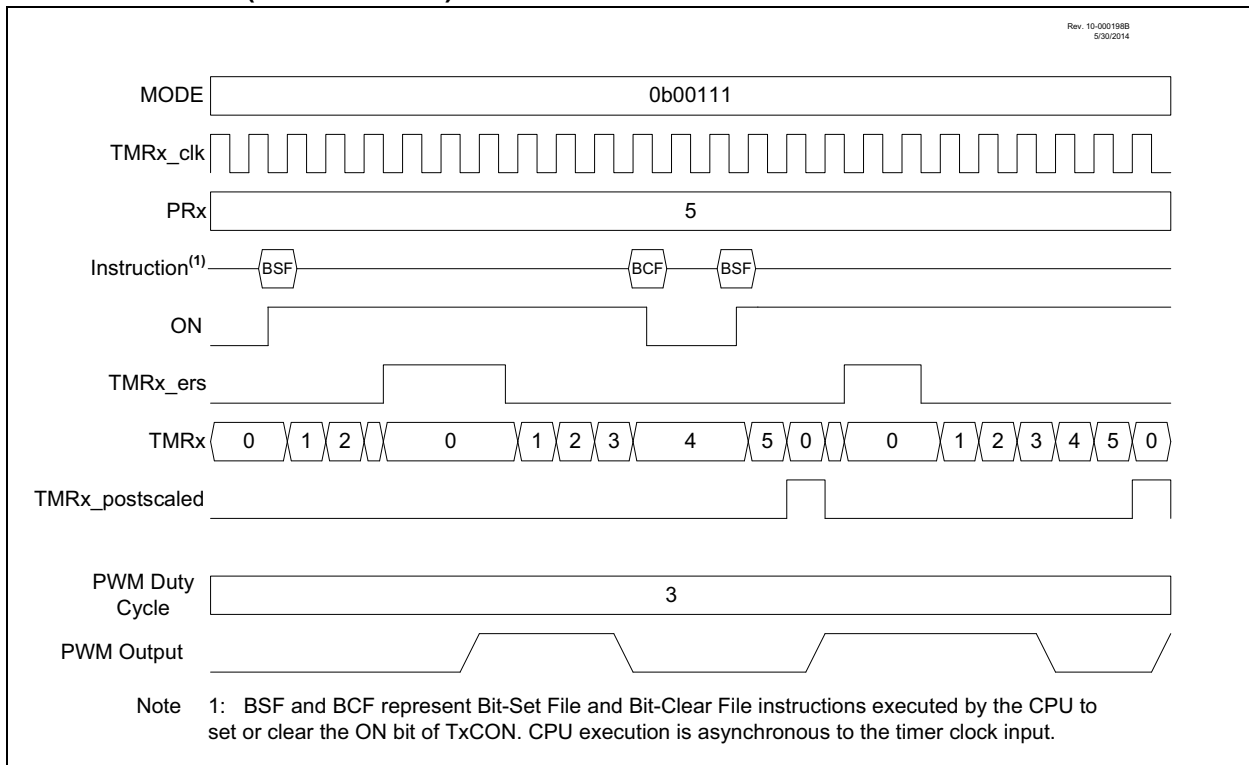
23.6.4 LEVEL-TRIGGERED HARDWARE LIMIT MODE

In the Level-Triggered Hardware Limit Timer modes the counter is reset by high or low levels of the external signal TMRx_ers, as shown in Figure 23-7. Selecting MODE<4:0> = 00110 will cause the timer to reset on a low level external signal. Selecting MODE<4:0> = 00111 will cause the timer to reset on a high level external signal. In the example, the counter is reset while TMRx_ers = 1. ON is controlled by BSF and BCF instructions. When ON = 0 the external signal is ignored.

When the CCP uses the timer as the PWM time base then the PWM output will be set high when the timer starts counting and then set low only when the timer count matches the CCPRx value. The timer is reset when either the timer count matches the PRx value or two clock periods after the external Reset signal goes true and stays true.

The timer starts counting, and the PWM output is set high, on either the clock following the PRx match or two clocks after the external Reset signal relinquishes the Reset. The PWM output will remain high until the timer counts up to match the CCPRx pulse width value. If the external Reset signal goes true while the PWM output is high then the PWM output will remain high until the Reset signal is released allowing the timer to count up to match the CCPRx value.

FIGURE 23-7: LEVEL-TRIGGERED HARDWARE LIMIT MODE TIMING DIAGRAM (MODE = 00111)

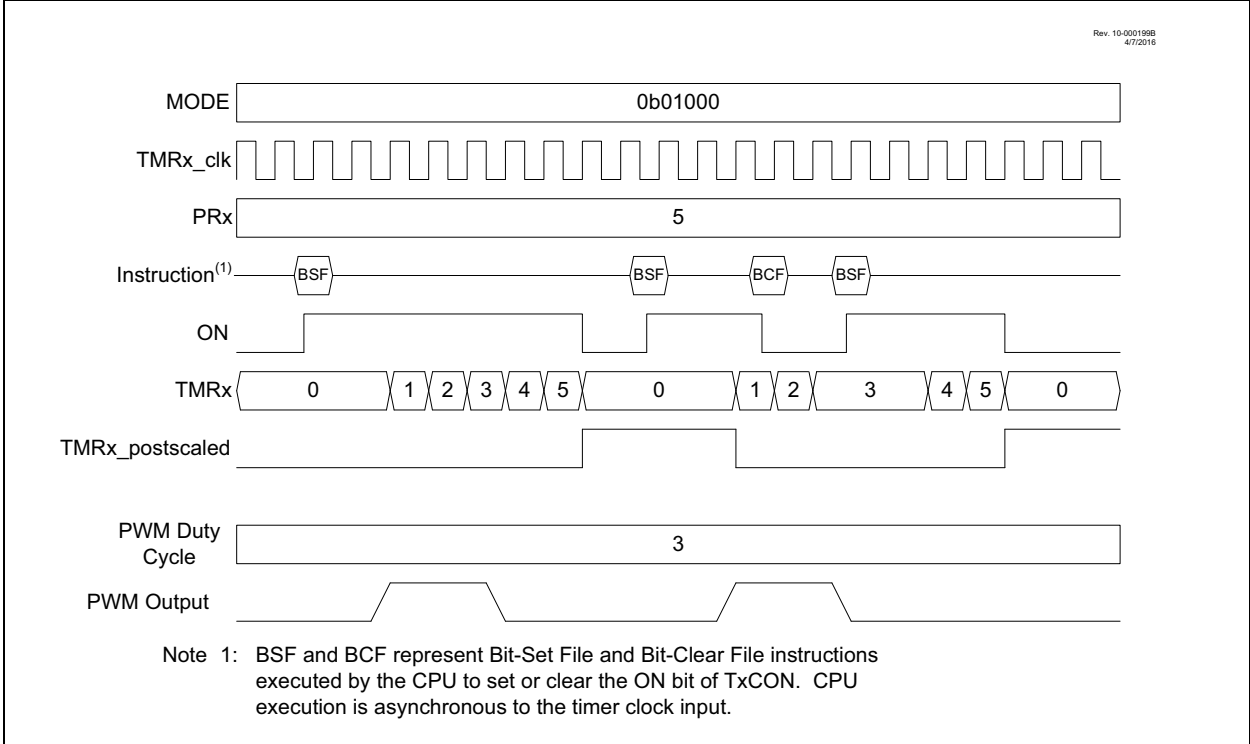


23.6.5 SOFTWARE START ONE-SHOT MODE

In One-Shot mode the timer resets and the ON bit is cleared when the timer value matches the PRx period value. The ON bit must be set by software to start another timer cycle. Setting MODE<4:0> = 01000 selects One-Shot mode which is illustrated in Figure 23-8. In the example, ON is controlled by BSF and BCF instructions. In the first case, a BSF instruction sets ON and the counter runs to completion and clears ON. In the second case, a BSF instruction starts the cycle, BCF/BSF instructions turn the counter off and on during the cycle, and then it runs to completion.

When One-Shot mode is used in conjunction with the CCP PWM operation the PWM pulse drive starts concurrent with setting the ON bit. Clearing the ON bit while the PWM drive is active will extend the PWM drive. The PWM drive will terminate when the timer value matches the CCPRx pulse width value. The PWM drive will remain off until software sets the ON bit to start another cycle. If software clears the ON bit after the CCPRx match but before the PRx match then the PWM drive will be extended by the length of time the ON bit remains cleared. Another timing cycle can only be initiated by setting the ON bit after it has been cleared by a PRx period count match.

FIGURE 23-8: SOFTWARE START ONE-SHOT MODE TIMING DIAGRAM (MODE = 01000)



23.6.6 EDGE-TRIGGERED ONE-SHOT MODE

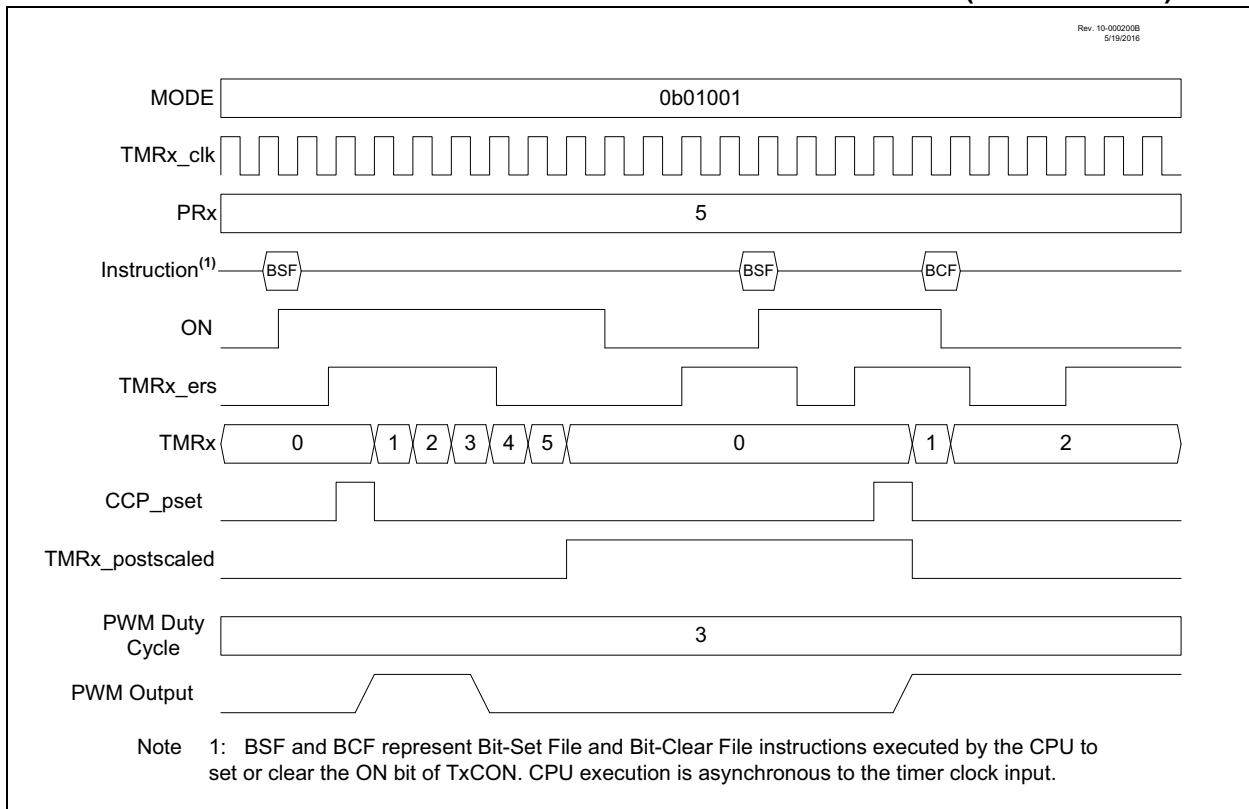
The Edge-Triggered One-Shot modes start the timer on an edge from the external signal input, after the ON bit is set, and clear the ON bit when the timer matches the PRx period value. The following edges will start the timer:

- Rising edge (MODE<4:0> = 01001)
- Falling edge (MODE<4:0> = 01010)
- Rising or Falling edge (MODE<4:0> = 01011)

If the timer is halted by clearing the ON bit then another TMRx_ers edge is required after the ON bit is set to resume counting. Figure 23-9 illustrates operation in the rising edge One-Shot mode.

When Edge-Triggered One-Shot mode is used in conjunction with the CCP then the edge-trigger will activate the PWM drive and the PWM drive will deactivate when the timer matches the CCPRx pulse width value and stay deactivated when the timer halts at the PRx period count match.

FIGURE 23-9: EDGE-TRIGGERED ONE-SHOT MODE TIMING DIAGRAM (MODE = 01001)



23.6.7 EDGE-TRIGGERED HARDWARE LIMIT ONE-SHOT MODE

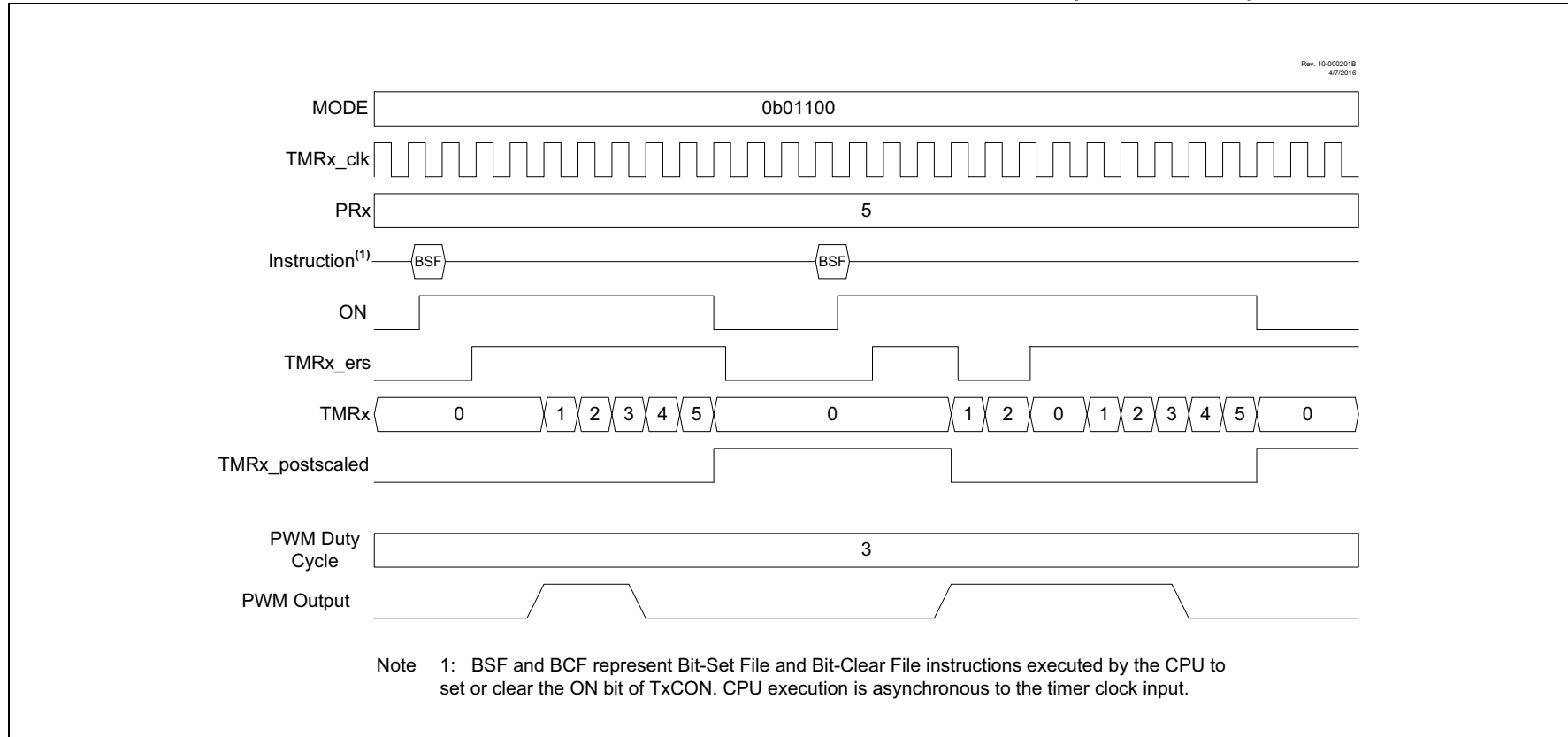
In Edge-Triggered Hardware Limit One-Shot modes the timer starts on the first external signal edge after the ON bit is set and resets on all subsequent edges. Only the first edge after the ON bit is set is needed to start the timer. The counter will resume counting automatically two clocks after all subsequent external Reset edges. Edge triggers are as follows:

- Rising edge start and Reset
(MODE<4:0> = 01100)
- Falling edge start and Reset
(MODE<4:0> = 01101)

The timer resets and clears the ON bit when the timer value matches the PRx period value. External signal edges will have no effect until after software sets the ON bit. [Figure 23-10](#) illustrates the rising edge hardware limit one-shot operation.

When this mode is used in conjunction with the CCP then the first starting edge trigger, and all subsequent Reset edges, will activate the PWM drive. The PWM drive will deactivate when the timer matches the CCPRx pulse-width value and stay deactivated until the timer halts at the PRx period match unless an external signal edge resets the timer before the match occurs.

FIGURE 23-10: EDGE-TRIGGERED HARDWARE LIMIT ONE-SHOT MODE TIMING DIAGRAM (MODE = 01100)



23.6.8 LEVEL RESET, EDGE-TRIGGERED HARDWARE LIMIT ONE-SHOT MODES

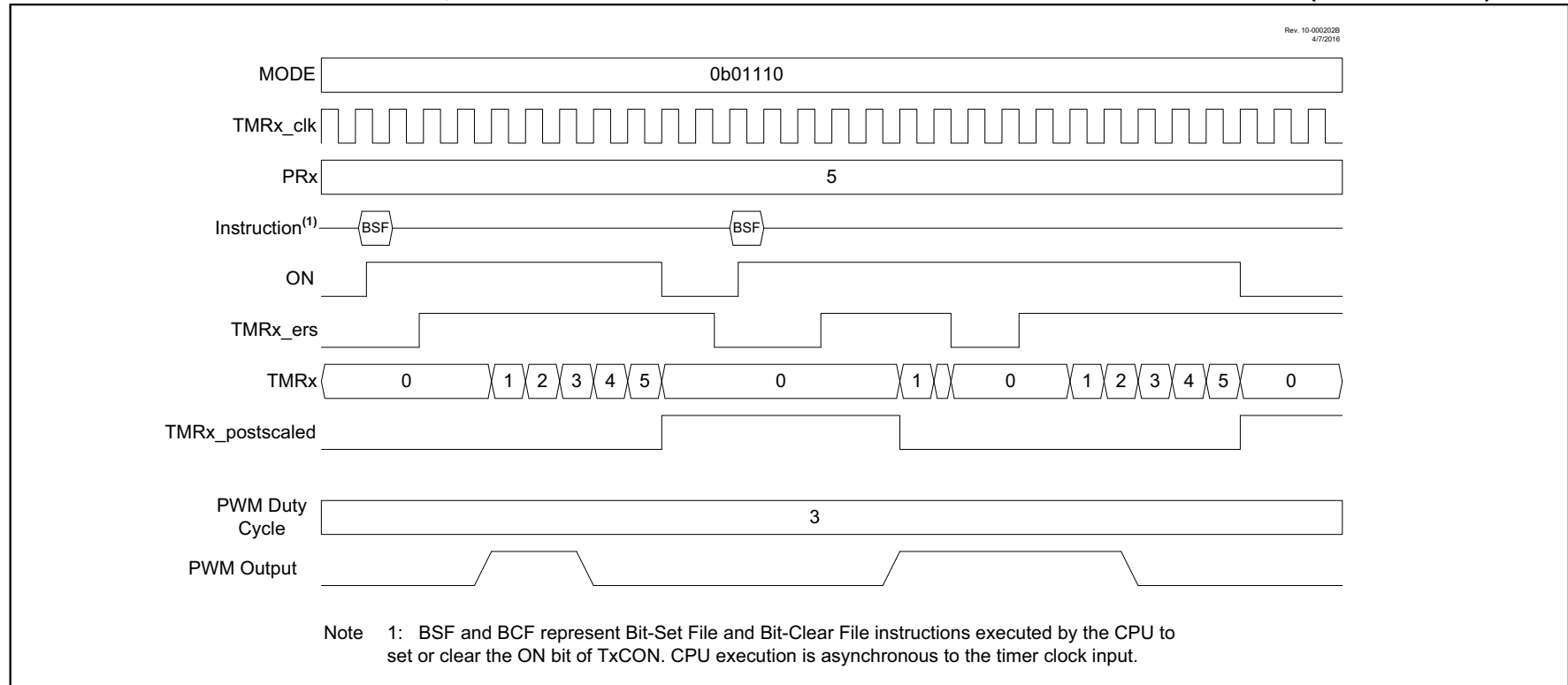
In Level -Triggered One-Shot mode the timer count is reset on the external signal level and starts counting on the rising/falling edge of the transition from Reset level to the active level while the ON bit is set. Reset levels are selected as follows:

- Low Reset level (MODE<4:0> = 01110)
- High Reset level (MODE<4:0> = 01111)

When the timer count matches the PRx period count, the timer is reset and the ON bit is cleared. When the ON bit is cleared by either a PRx match or by software control a new external signal edge is required after the ON bit is set to start the counter.

When Level-Triggered Reset One-Shot mode is used in conjunction with the CCP PWM operation the PWM drive goes active with the external signal edge that starts the timer. The PWM drive goes inactive when the timer count equals the CCPRx pulse width count. The PWM drive does not go active when the timer count clears at the PRx period count match.

FIGURE 23-11: LOW LEVEL RESET, EDGE-TRIGGERED HARDWARE LIMIT ONE-SHOT MODE TIMING DIAGRAM (MODE = 01110)



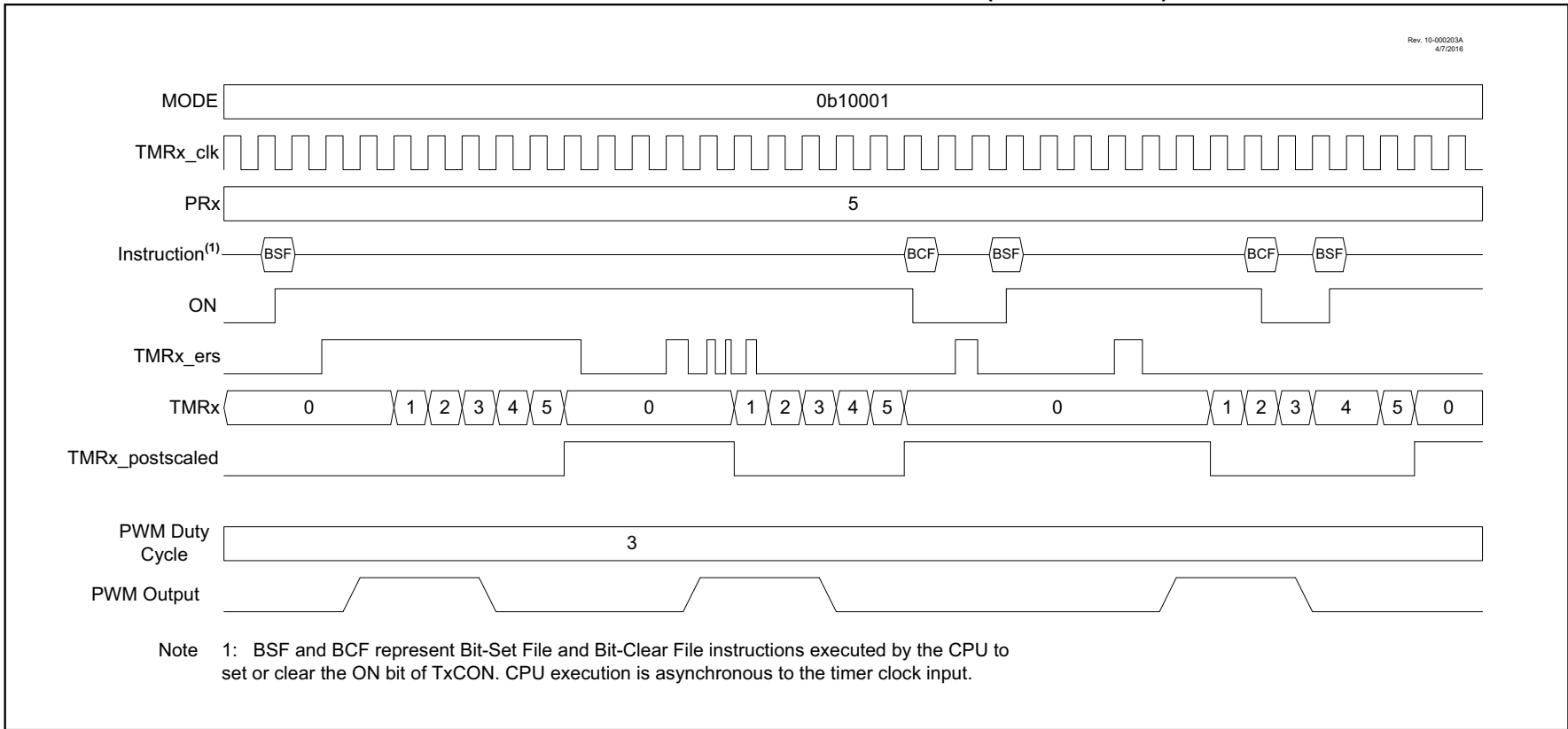
23.6.9 EDGE-TRIGGERED MONOSTABLE MODES

The Edge-Triggered Monostable modes start the timer on an edge from the external Reset signal input, after the ON bit is set, and stop incrementing the timer when the timer matches the PRx period value. The following edges will start the timer:

- Rising edge (MODE<4:0> = 10001)
- Falling edge (MODE<4:0> = 10010)
- Rising or Falling edge (MODE<4:0> = 10011)

When an Edge-Triggered Monostable mode is used in conjunction with the CCP PWM operation the PWM drive goes active with the external Reset signal edge that starts the timer, but will not go active when the timer matches the PRx value. While the timer is incrementing, additional edges on the external Reset signal will not affect the CCP PWM.

FIGURE 23-12: RISING EDGE-TRIGGERED MONOSTABLE MODE TIMING DIAGRAM (MODE = 10001)



23.6.10 LEVEL-TRIGGERED HARDWARE LIMIT ONE-SHOT MODES

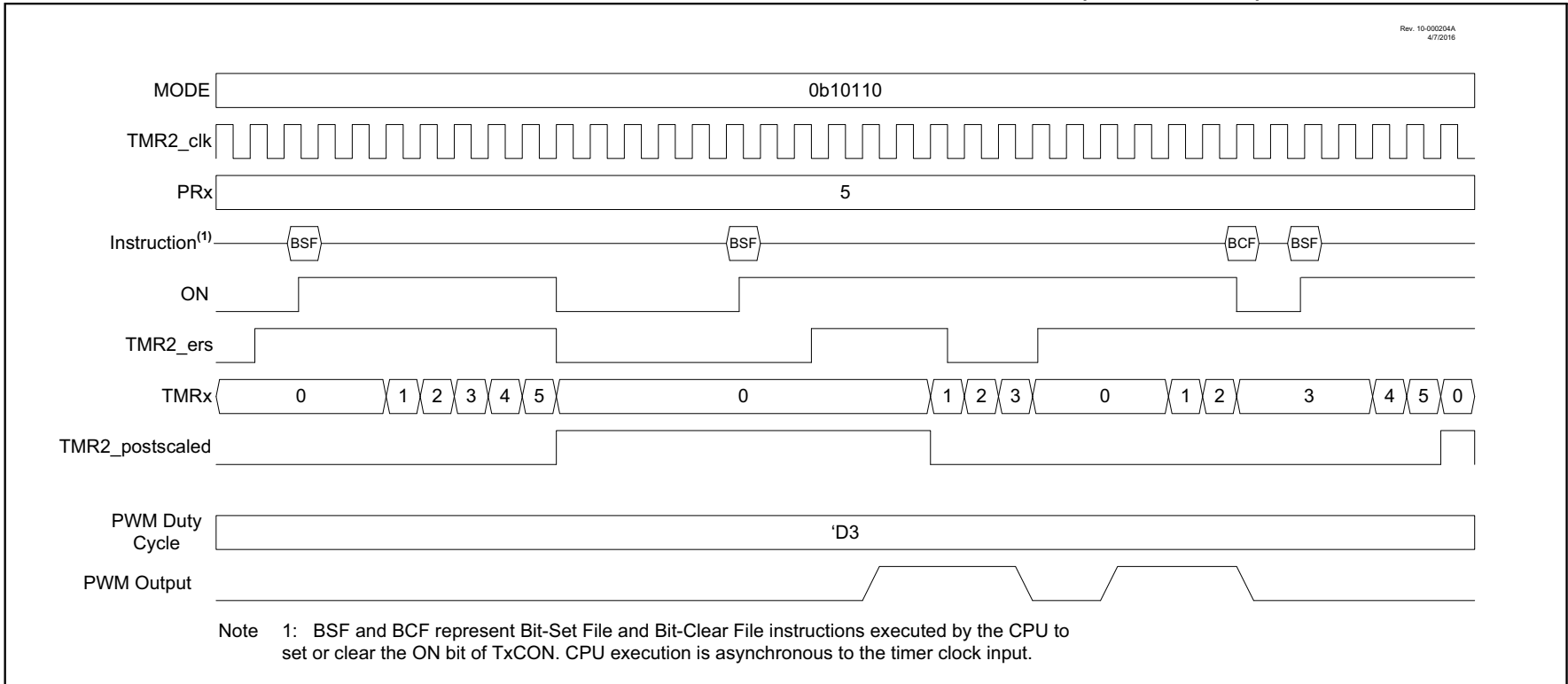
The Level-Triggered Hardware Limit One-Shot modes hold the timer in Reset on an external Reset level and start counting when both the ON bit is set and the external signal is not at the Reset level. If one of either the external signal is not in Reset or the ON bit is set then the other signal being set/made active will start the timer. Reset levels are selected as follows:

- Low Reset level (MODE<4:0> = 10110)
- High Reset level (MODE<4:0> = 10111)

When the timer count matches the PRx period count, the timer is reset and the ON bit is cleared. When the ON bit is cleared by either a PRx match or by software control the timer will stay in Reset until both the ON bit is set and the external signal is not at the Reset level.

When Level-Triggered Hardware Limit One-Shot modes are used in conjunction with the CCP PWM operation the PWM drive goes active with either the external signal edge or the setting of the ON bit, whichever of the two starts the timer.

FIGURE 23-13: LEVEL-TRIGGERED HARDWARE LIMIT ONE-SHOT MODE TIMING DIAGRAM (MODE = 10110)



23.7 Timer2 Operation During Sleep

When PSYNC = 1, Timer2 cannot be operated while the processor is in Sleep mode. The contents of the TMR2 and T2PR registers will remain unchanged while processor is in Sleep mode.

When PSYNC = 0, Timer2 will operate in Sleep as long as the clock source selected is also still running. Selecting the LFINTOSC, MFINTOSC, or HFINTOSC oscillator as the timer clock source will keep the selected oscillator running during Sleep.

23.8 Register Definitions: Timer2/4/6 Control

Long bit name prefixes for the Timer2/4/6 peripherals are shown in [Table 23-2](#). Refer to [Section 1.1.2.2 “Long Bit Names”](#) for more information

TABLE 23-2:

| Peripheral | Bit Name Prefix |
|------------|-----------------|
| Timer2 | T2 |
| Timer4 | T4 |
| Timer6 | T6 |

REGISTER 23-1: TxCLKCON: TIMERx CLOCK SELECTION REGISTER

| U-0 | U-0 | U-0 | U-0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 |
|-------|-----|-----|-----|---------|---------|---------|---------|
| — | — | — | — | CS<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 | |

bit 7-4 **Unimplemented:** Read as '0'
bit 3-0 **CS<3:0>:** Timerx Clock Selection bits
See [Table 23-3](#).

TABLE 23-3: TIMERx CLOCK SOURCES

| CS<3:0> | Timer2 | Timer4 | Timer6 |
|-----------|-------------------------|-------------------------|-------------------------|
| 1101-1111 | Reserved | Reserved | Reserved |
| 1011 | AT1_perclk | AT1_perclk | AT1_perclk |
| 1010 | LC4_out | LC4_out | LC4_out |
| 1001 | LC3_out | LC3_out | LC3_out |
| 1000 | LC2_out | LC2_out | LC2_out |
| 0111 | LC1_out | LC1_out | LC1_out |
| 0110 | Pin selected by T2INPPS | Pin selected by T2INPPS | Pin selected by T2INPPS |
| 0101 | MFINTOSC 31.25 kHz | MFINTOSC 31.25 kHz | MFINTOSC 31.25 kHz |
| 0100 | ZCD1_output | ZCD1_output | ZCD1_output |
| 0011 | LFINTOSC | LFINTOSC | LFINTOSC |
| 0010 | HFINTOSC 16 MHz | HFINTOSC 16 MHz | HFINTOSC 16 MHz |
| 0001 | Fosc | Fosc | Fosc |
| 0000 | Fosc/4 | Fosc/4 | Fosc/4 |

REGISTER 23-2: TxCON: TIMERx CONTROL REGISTER

| | | | | | | | |
|-------------------|-----------|---------|---------|------------|---------|---------|---------|
| R/W/HC-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 |
| ON ⁽¹⁾ | CKPS<2:0> | | | OUTPS<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 | HC = Bit is cleared by hardware |

bit 7 **ON:** Timerx On bit
 1 = Timerx is on
 0 = Timerx is off: all counters and state machines are reset

bit 6-4 **CKPS<2:0>:** Timer2-type Clock Prescale Select bits
 111 = 1:128 Prescaler
 110 = 1:64 Prescaler
 101 = 1:32 Prescaler
 100 = 1:16 Prescaler
 011 = 1:8 Prescaler
 010 = 1:4 Prescaler
 001 = 1:2 Prescaler
 000 = 1:1 Prescaler

bit 3-0 **OUTPS<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

Note 1: In certain modes, the ON bit will be auto-cleared by hardware. See [Section 23.6 "Operation Examples"](#).

REGISTER 23-3: TxHLT: TIMERx HARDWARE LIMIT 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/W-0/0 |
| PSYNC ^(1, 2) | CKPOL ⁽³⁾ | CKSYNC ^(4, 5) | MODE<4:0> ^(6, 7) | | | | |
| 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 | PSYNC: Timerx Prescaler Synchronization Enable bit ^(1, 2) 1 = TMRx Prescaler Output is synchronized to Fosc/4 0 = TMRx Prescaler Output is not synchronized to Fosc/4 |
| bit 6 | CKPOL: Timerx Clock Polarity Selection bit ⁽³⁾ 1 = Falling edge of input clock clocks timer/prescaler 0 = Rising edge of input clock clocks timer/prescaler |
| bit 5 | CKSYNC: Timerx Clock Synchronization Enable bit ^(4, 5) 1 = ON register bit is synchronized to TMR2_clk input 0 = ON register bit is not synchronized to TMR2_clk input |
| bit 4-0 | MODE<4:0>: Timerx Control Mode Selection bits ^(6, 7) See Table 23-1 . |

- Note**
- Setting this bit ensures that reading TMRx will return a valid value.
 - When this bit is '1', Timer2 cannot operate in Sleep mode.
 - CKPOL should not be changed while ON = 1.
 - Setting this bit ensures glitch-free operation when the ON is enabled or disabled.
 - When this bit is set then the timer operation will be delayed by two TMRx input clocks after the ON bit is set.
 - Unless otherwise indicated, all modes start upon ON = 1 and stop upon ON = 0 (stops occur without affecting the value of TMRx).
 - When TMRx = PRx, the next clock clears TMRx, regardless of the operating mode.

REGISTER 23-4: TxRST: TIMERx EXTERNAL RESET SIGNAL SELECTION REGISTER

| | | | | | | | |
|-------|-----|-----|-----|-----------|---------|---------|---------|
| U-0 | U-0 | U-0 | U-0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 |
| — | — | — | — | RSEL<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 | |

bit 7-4 **Unimplemented:** Read as '0'
bit 3-0 **RSEL<4:0>:** TimerX External Reset Signal Source Selection bits
See [Table 23-4](#).

TABLE 23-4: EXTERNAL RESET SOURCES

| RSEL<4:0> | Timer2 | Timer4 | Timer6 |
|-----------|-------------------------|-------------------------|-------------------------|
| 1111 | Reserved | Reserved | Reserved |
| 1110 | PWM4_out | PWM4_out | PWM4_out |
| 1101 | PWM3_out | PWM3_out | PWM3_out |
| 1100 | LC4_out | LC4_out | LC4_out |
| 1011 | LC3_out | LC3_out | LC3_out |
| 1010 | LC2_out | LC2_out | LC2_out |
| 1001 | LC1_out | LC1_out | LC1_out |
| 1000 | ZCD1_out | ZCD1_out | ZCD1_out |
| 0111 | TMR6_postscaled | TMR6_postscaled | Reserved |
| 0110 | TMR4_postscaled | Reserved | TMR4_postscaled |
| 0101 | Reserved | TMR2_postscaled | TMR2_postscaled |
| 0100 | CCP2_out | CCP2_out | CCP2_out |
| 0011 | CCP1_out | CCP1_out | CCP1_out |
| 0010 | C2OUT_sync | C2OUT_sync | C2OUT_sync |
| 0001 | C1OUT_sync | C1OUT_sync | C1OUT_sync |
| 0000 | Pin selected by T2INPPS | Pin selected by T2INPPS | Pin selected by T2INPPS |

TABLE 23-5: SUMMARY OF REGISTERS ASSOCIATED WITH TIMER2

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Register on Page |
|----------|--|-----------|--------|-----------|------------|-----------|--------|--------|------------------|
| CCP1CON | EN | — | OUT | FMT | MODE<3:0> | | | | 354 |
| CCP2CON | EN | — | OUT | FMT | MODE<3:0> | | | | 354 |
| INTCON | GIE | PEIE | TMR0IE | INTE | IOCFIE | TMR0IF | INTF | IOCFIF | 97 |
| PIE1 | TMR1GIE | ADIE | — | — | — | CCP1IE | TMR2IE | TMR1IE | 98 |
| PIR1 | TMR1GIF | ADIF | RCIF | TXIF | SSP1IF | CCP1IF | TMR2IF | TMR1IF | 103 |
| PR2 | Timer2 Module Period Register | | | | | | | | 244* |
| TMR2 | Holding Register for the 8-bit TMR2 Register | | | | | | | | 236* |
| T2CON | ON | CKPS<2:0> | | | OUTPS<3:0> | | | | 256 |
| T2CLKCON | — | — | — | — | CS<3:0> | | | | 255 |
| T2RST | — | — | — | — | RSEL<3:0> | | | | 258 |
| T2HLT | PSYNC | CKPOL | CKSYNC | MODE<4:0> | | | | | 257 |
| PR4 | Timer4 Module Period Register | | | | | | | | 244* |
| TMR4 | Holding Register for the 8-bit TMR4 Register | | | | | | | | 236* |
| T4CON | ON | CKPS<2:0> | | | OUTPS<3:0> | | | | 256 |
| T4CLKCON | — | — | — | — | CS<3:0> | | | | 255 |
| T4RST | — | — | — | — | RSEL<3:0> | | | | 258 |
| T4HLT | PSYNC | CKPOL | CKSYNC | MODE<4:0> | | | | | 257 |
| PR6 | Timer6 Module Period Register | | | | | | | | 244* |
| TMR6 | Holding Register for the 8-bit TMR6 Register | | | | | | | | 236* |
| T6CON | ON | CKPS<2:0> | | | OUTPS<3:0> | | | | 256 |
| T6CLKCON | — | — | — | — | — | T6CS<2:0> | | | 255 |
| T6RST | — | — | — | — | RSEL<3:0> | | | | 258 |
| T6HLT | PSYNC | CKPOL | CKSYNC | MODE<4:0> | | | | | 257 |

Legend: — = unimplemented location, read as '0'. Shaded cells are not used for Timer2 module.

* Page provides register information.

24.0 MASTER SYNCHRONOUS SERIAL PORT (MSSP) MODULE

24.1 MSSP Module Overview

The Master Synchronous Serial Port (MSSP) module is a serial interface useful for communicating with other peripheral or microcontroller devices. These peripheral devices may be serial EEPROMs, shift registers, display drivers, A/D converters, etc. The MSSP module can operate in one of two modes:

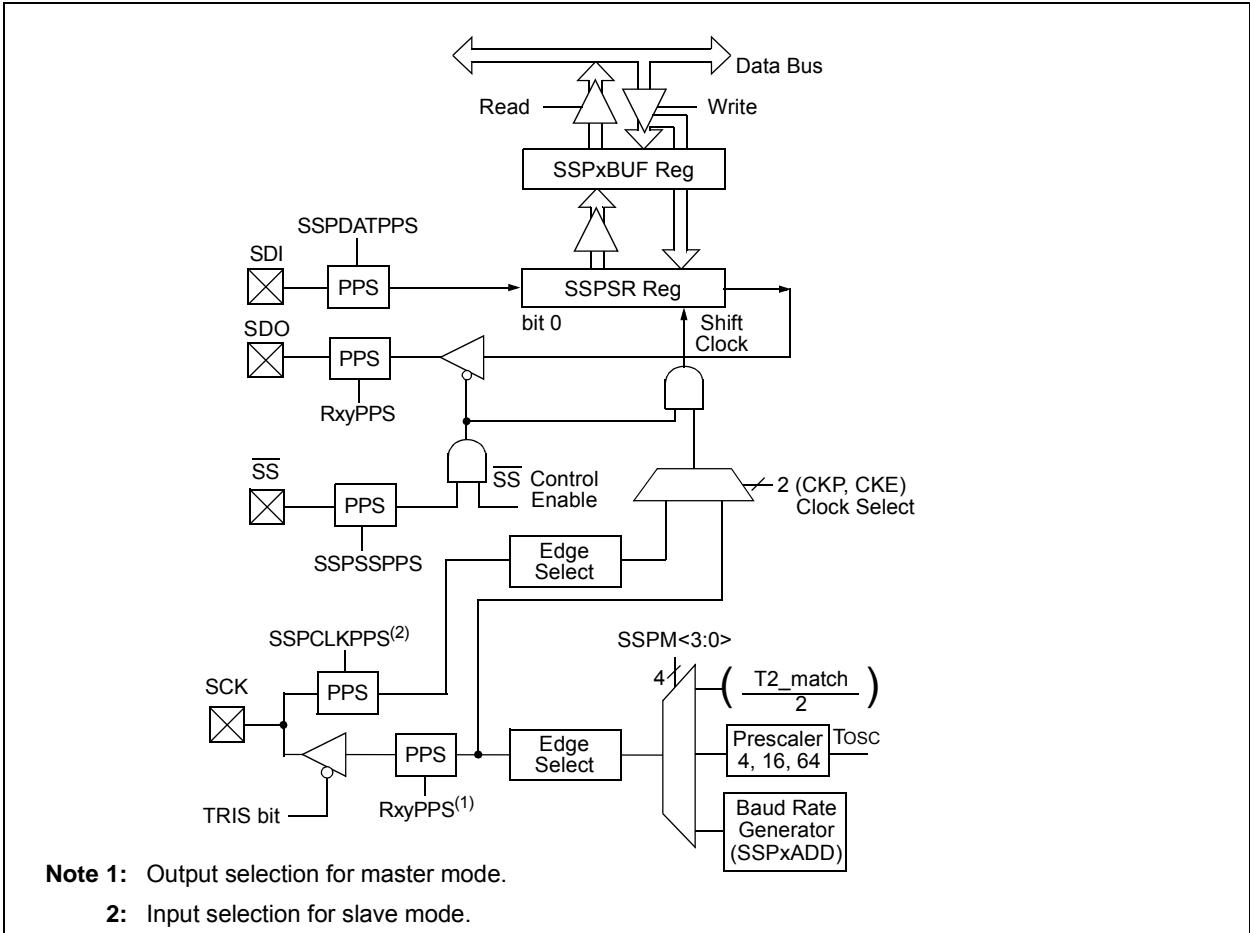
- Serial Peripheral Interface (SPI)
- Inter-Integrated Circuit (I²C)

The SPI interface supports the following modes and features:

- Master mode
- Slave mode
- Clock Parity
- Slave Select Synchronization (Slave mode only)
- Daisy-chain connection of slave devices

Figure 24-1 is a block diagram of the SPI interface module.

FIGURE 24-1: MSSP BLOCK DIAGRAM (SPI MODE)



Note 1: Output selection for master mode.

2: Input selection for slave mode.

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 SDA hold times

Figure 24-2 is a block diagram of the I²C interface module in Master mode. Figure 24-3 is a diagram of the I²C interface module in Slave mode.

FIGURE 24-2: MSSP BLOCK DIAGRAM (I²C MASTER MODE)

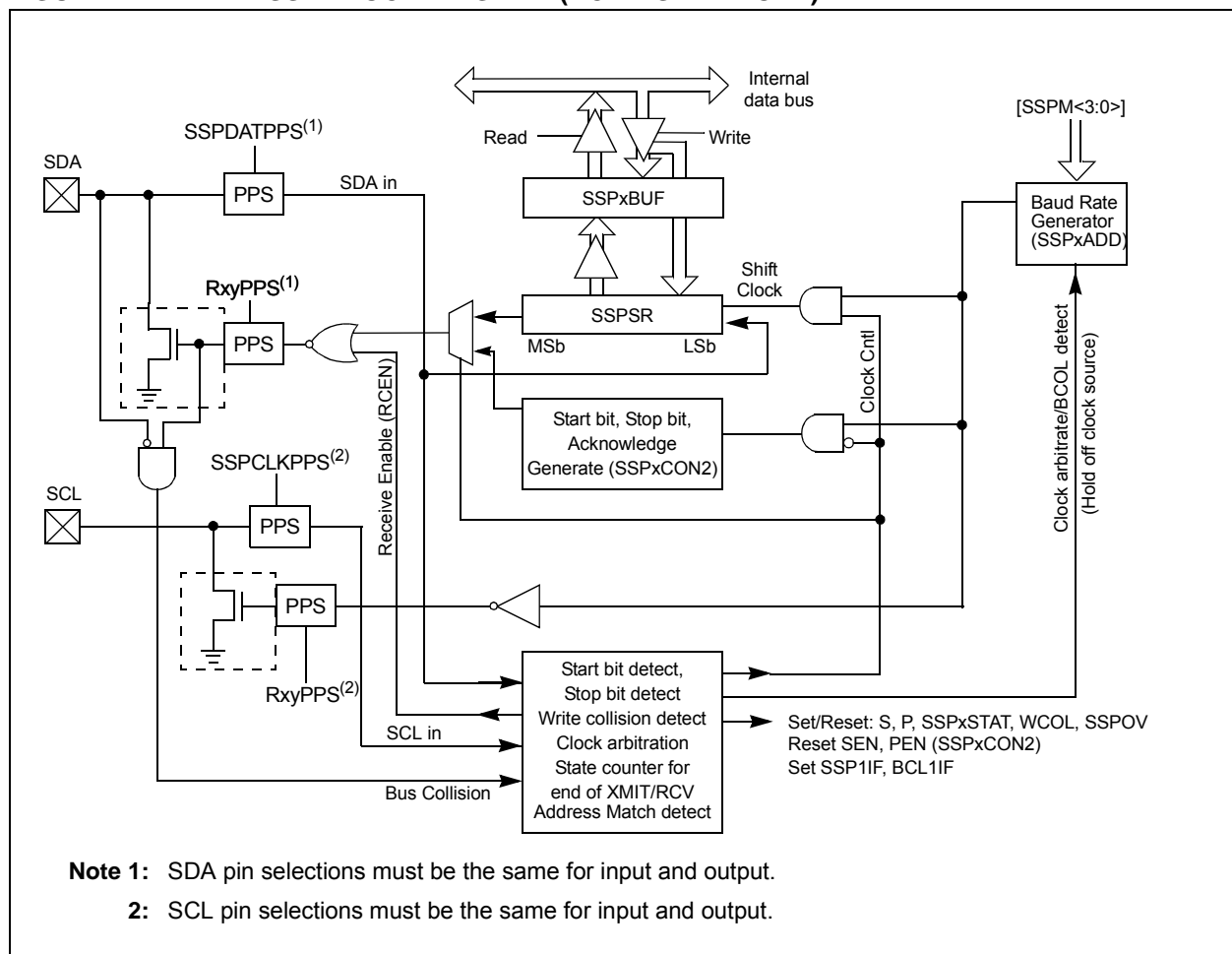
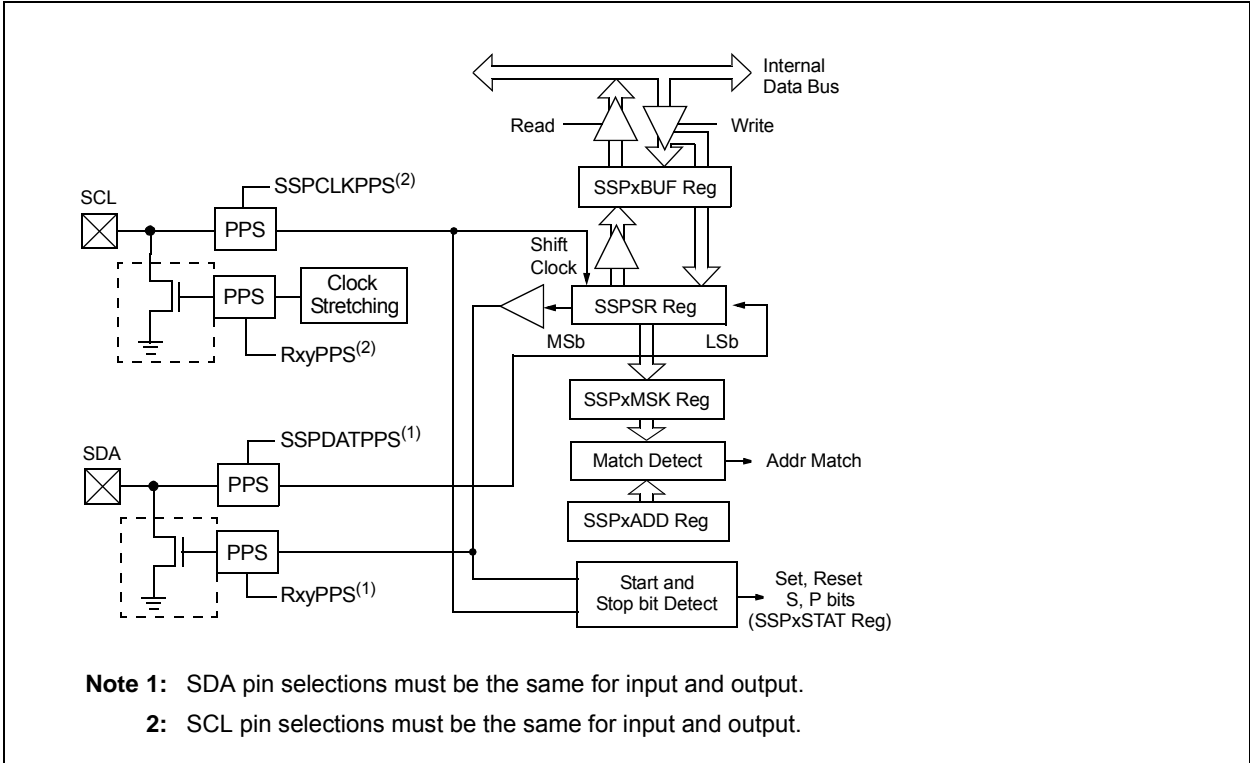


FIGURE 24-3: MSSP BLOCK DIAGRAM (I²C SLAVE MODE)



24.2 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 (SCK)
- Serial Data Out (SDO)
- Serial Data In (SDI)
- Slave Select (\overline{SS})

Figure 24-1 shows the block diagram of the MSSP 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 24-4 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 24-5 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 SDO output pin which is connected to, and received by, the slave's SDI input pin. The slave device transmits information out on its SDO output pin, which is connected to, and received by, the master's SDI 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 SDO 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 SDO pin) and the master device is reading this bit and saving it as the LSb of its shift register.

After eight 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.

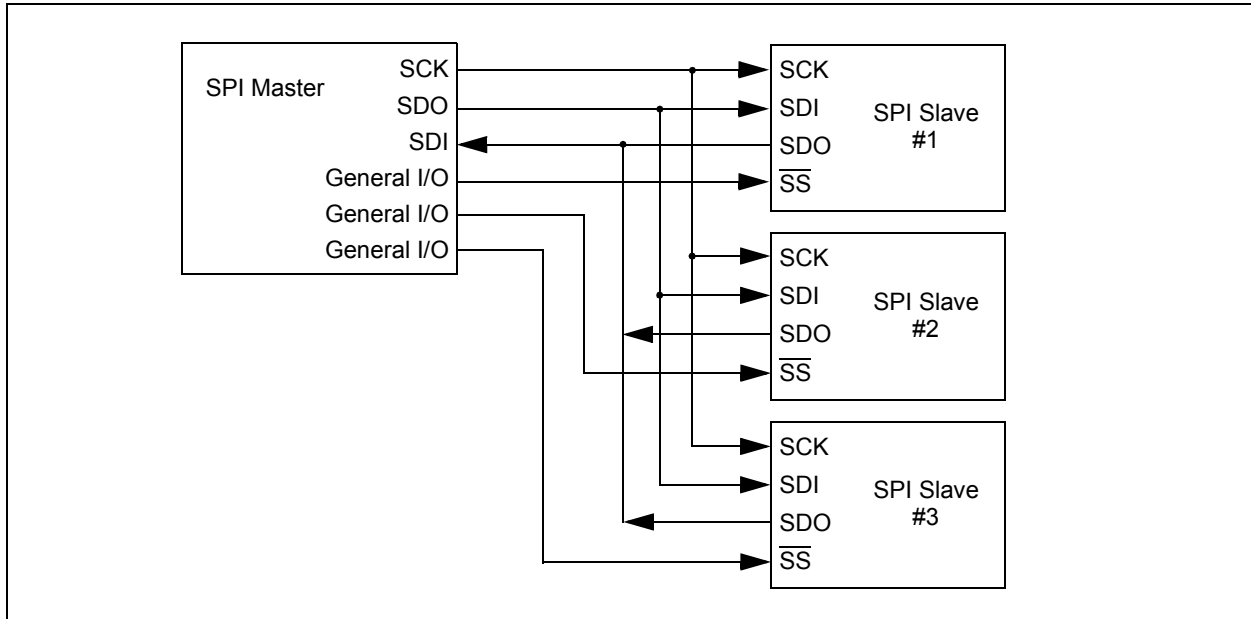
Whether the data is meaningful or not (dummy data), depends on the application software. This leads to three scenarios for data transmission:

- 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 24-4: SPI MASTER AND MULTIPLE SLAVE CONNECTION



24.2.1 SPI MODE REGISTERS

The MSSP module has five registers for SPI mode operation. These are:

- MSSP STATUS register (SSPxSTAT)
- MSSP Control register 1 (SSPxCON1)
- MSSP Control register 3 (SSPxCON3)
- MSSP Data Buffer register (SSPxBUF)
- MSSP Address register (SSPxADD)
- MSSP Shift register (SSPSR)
(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 six 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 24.7 “Baud Rate Generator”](#).

SSPSR is the shift register used for shifting data in and out. SSPxBUF provides indirect access to the SSPSR register. SSPxBUF is the buffer register to which data bytes are written, and from which data bytes are read.

In receive operations, SSPSR and SSPxBUF together create a buffered receiver. When SSPSR 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 SSPSR.

24.2.2 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 (SCK is the clock output)
- Slave mode (SCK is the clock input)
- Clock Polarity (Idle state of SCK)
- Data Input Sample Phase (middle or end of data output time)
- Clock Edge (output data on rising/falling edge of SCK)
- Clock Rate (Master mode only)
- Slave Select mode (Slave mode only)

To enable the serial port, SSP 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 SDI, SDO, SCK and SS 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:

- SDI must have corresponding TRIS bit set
- SDO must have corresponding TRIS bit cleared
- SCK (Master mode) must have corresponding TRIS bit cleared
- SCK (Slave mode) must have corresponding TRIS bit set
- SS must have corresponding TRIS bit set

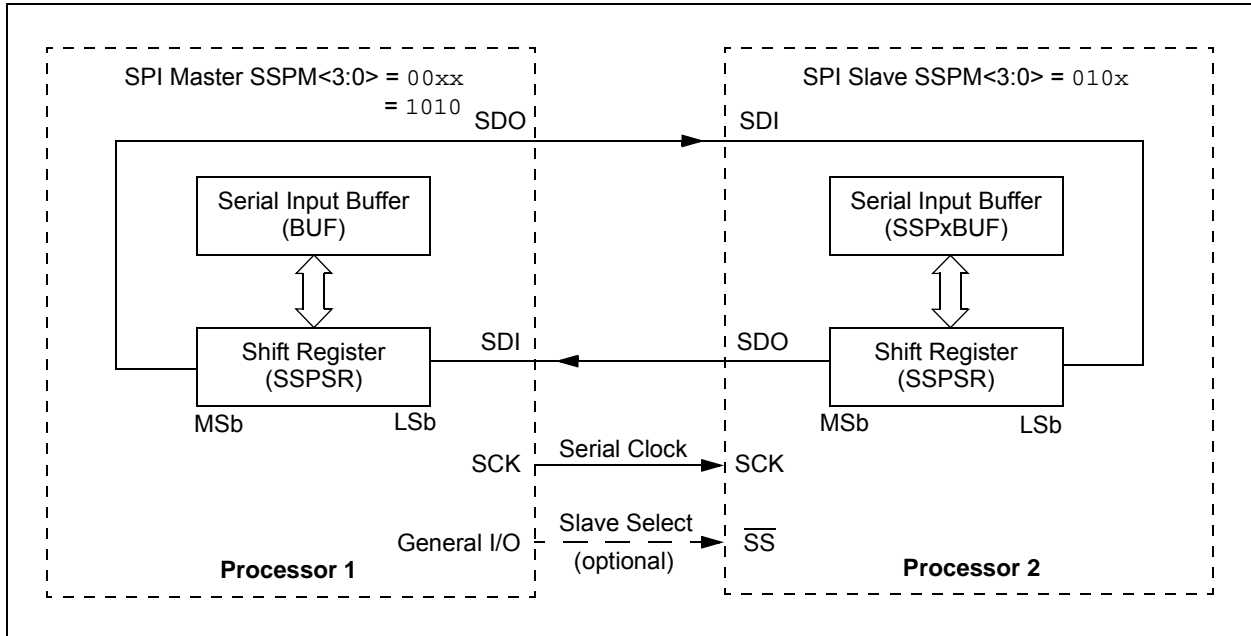
Any serial port function that is not desired may be overridden by programming the corresponding data direction (TRIS) register to the opposite value.

The MSSP consists of a transmit/receive shift register (SSPSR) and a buffer register (SSPxBUF). The SSPSR shifts the data in and out of the device, MSb first. The SSPxBUF holds the data that was written to the SSPSR until the received data is ready. Once the eight 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 MSSP 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.

The SSPSR is not directly readable or writable and can only be accessed by addressing the SSPxBUF register. Additionally, the SSPxSTAT register indicates the various Status conditions.

FIGURE 24-5: SPI MASTER/SLAVE CONNECTION



24.2.3 SPI MASTER MODE

The master can initiate the data transfer at any time because it controls the SCK line. The master determines when the slave (Processor 2, [Figure 24-5](#)) 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 SDO output could be disabled (programmed as an input). The SSPSR register will continue to shift in the signal present on the SDI 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 24-6](#), [Figure 24-8](#), [Figure 24-9](#) and [Figure 24-10](#), where the MSB is transmitted first. In Master mode, the SPI clock rate (bit rate) is user programmable to be one of the following:

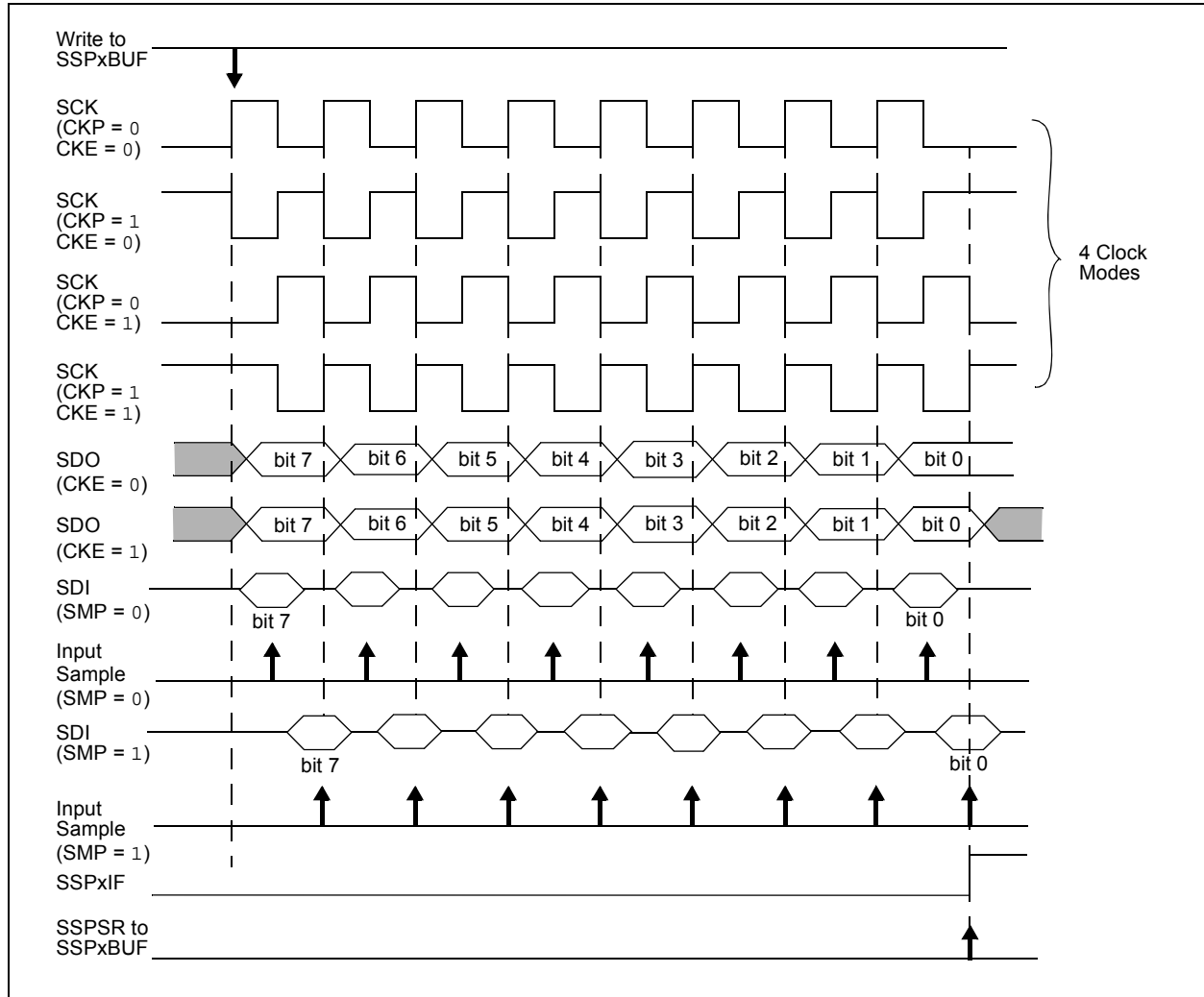
- $F_{osc}/4$ (or T_{CY})
- $F_{osc}/16$ (or $4 * T_{CY}$)
- $F_{osc}/64$ (or $16 * T_{CY}$)
- Timer2 output/2
- $F_{osc}/(4 * (SSPxADD + 1))$

[Figure 24-6](#) shows the waveforms for Master mode.

When the CKE bit is set, the SDO data is valid before there is a clock edge on SCK. 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.

| |
|---|
| <p>Note: In Master mode the clock signal output to the SCK pin is also the clock signal input to the peripheral. The pin selected for output with the RxyPPS register must also be selected as the peripheral input with the SSPCLKPPS register.</p> |
|---|

FIGURE 24-6: SPI MODE WAVEFORM (MASTER MODE)



24.2.4 SPI SLAVE MODE

In Slave mode, the data is transmitted and received as external clock pulses appear on SCK. 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 SCK 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 SCK 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 SCK pin input and when a byte is received, the device will generate an interrupt. If enabled, the device will wake-up from Sleep.

24.2.4.1 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 24-7 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.

24.2.5 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{SS} pin allows a Synchronous Slave mode. The SPI must be in Slave mode with \overline{SS} pin control enabled ($SSPxCON1<3:0> = 0100$).

When the \overline{SS} pin is low, transmission and reception are enabled and the SDO pin is driven.

When the \overline{SS} pin goes high, the SDO 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.

- Note 1:** When the SPI is in Slave mode with \overline{SS} pin control enabled ($SSPxCON1<3:0> = 0100$), the SPI module will reset if the \overline{SS} pin is set to VDD.
- 2:** When the SPI is used in Slave mode with CKE set; the user must enable \overline{SS} pin control.
- 3:** While operated in SPI Slave mode the SMP bit of the SSPxSTAT register must remain clear.

When the SPI module resets, the bit counter is forced to '0'. This can be done by either forcing the \overline{SS} pin to a high level or clearing the SSPEN bit.

FIGURE 24-7: SPI DAISY-CHAIN CONNECTION

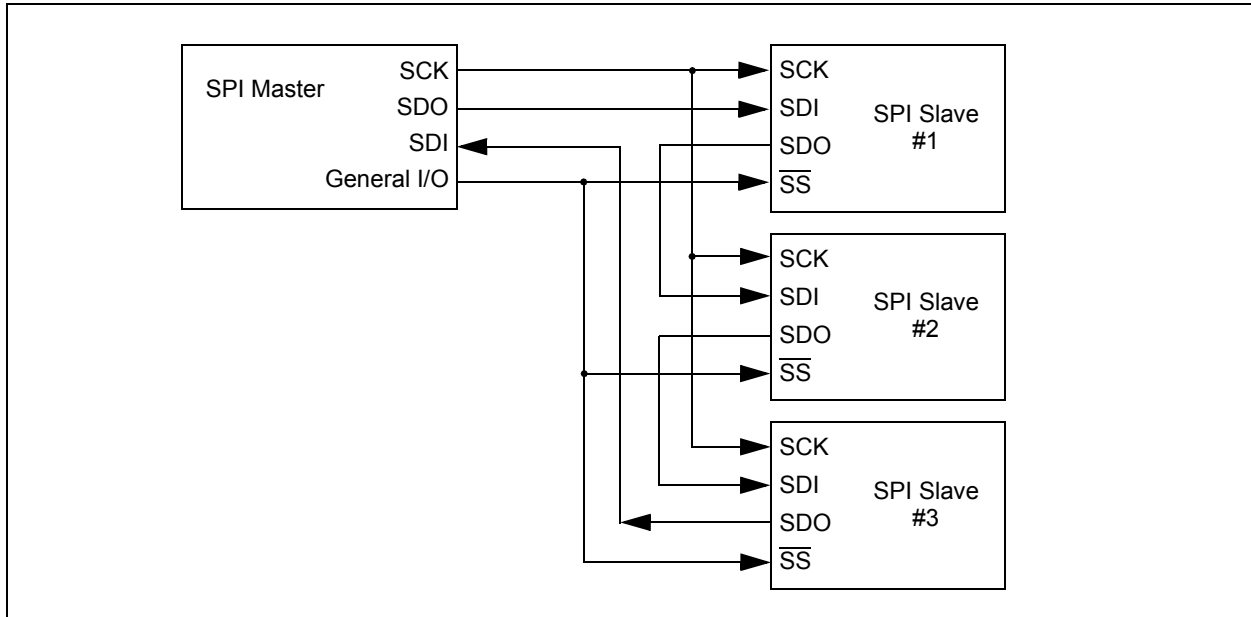


FIGURE 24-8: SLAVE SELECT SYNCHRONOUS WAVEFORM

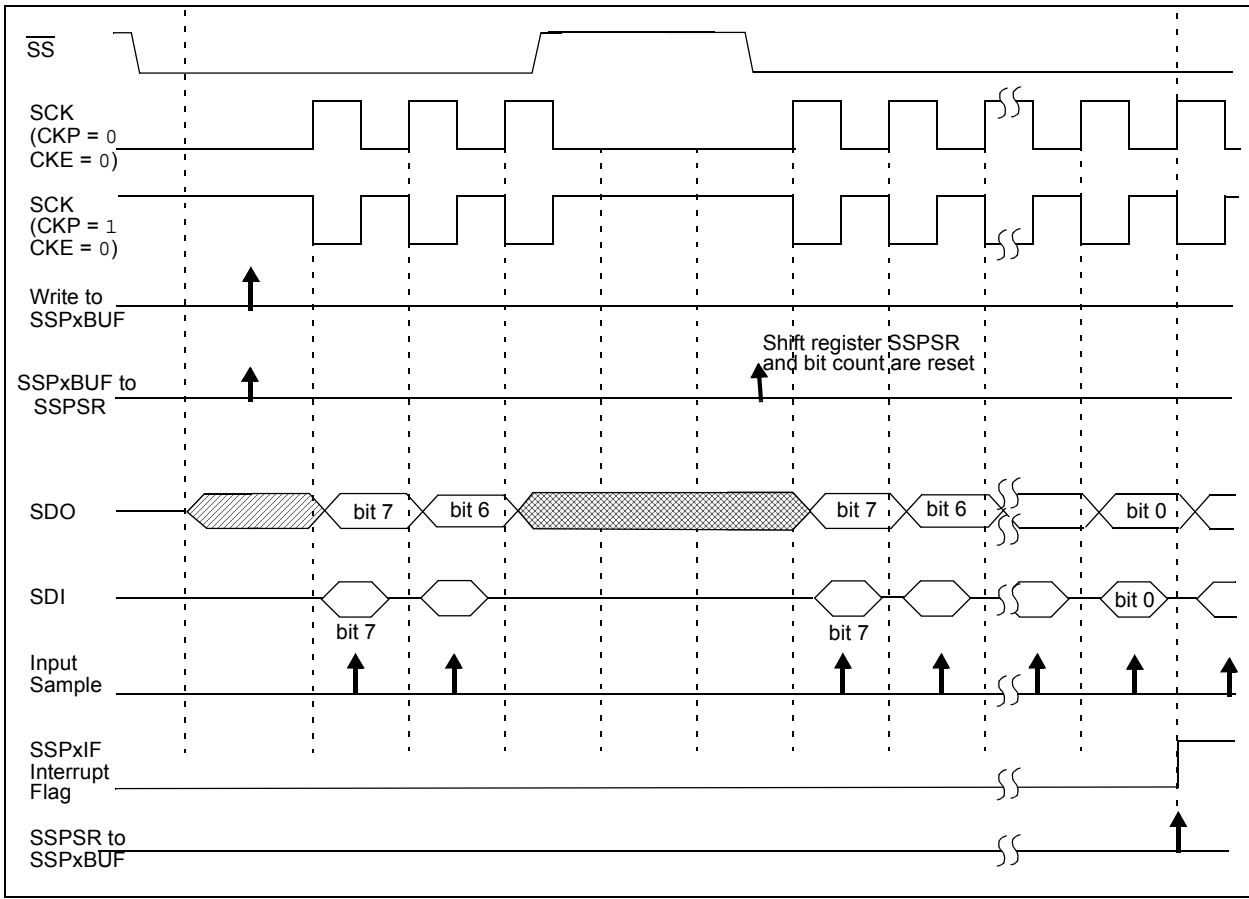


FIGURE 24-9: SPI MODE WAVEFORM (SLAVE MODE WITH CKE = 0)

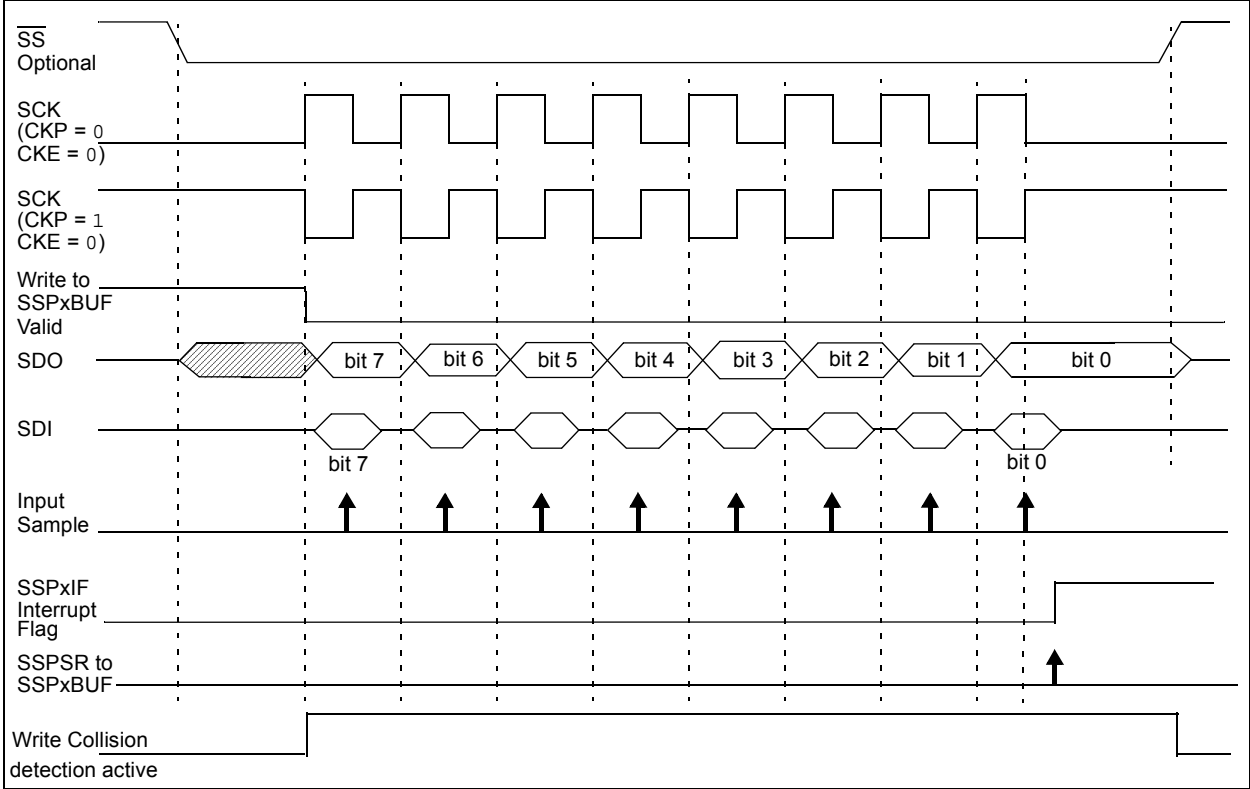
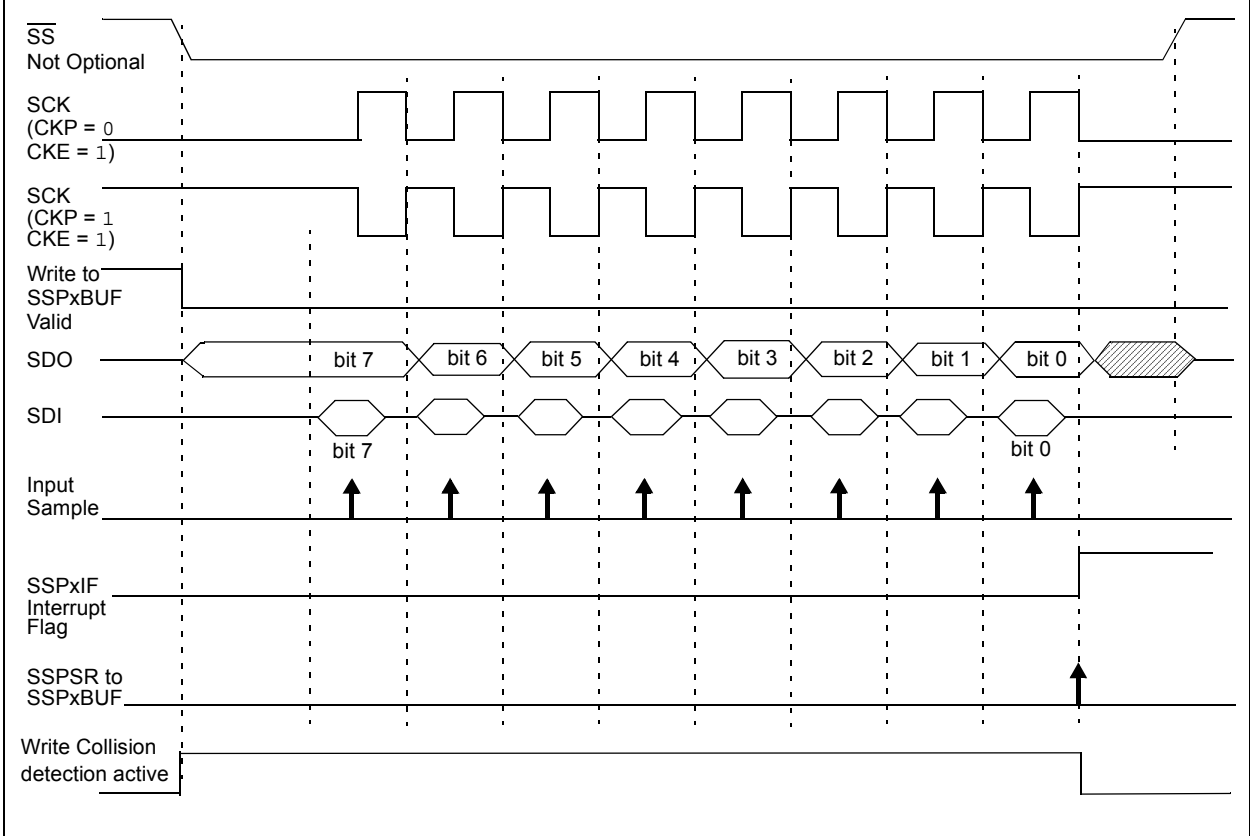


FIGURE 24-10: SPI MODE WAVEFORM (SLAVE MODE WITH CKE = 1)



24.2.6 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 MSSP clock is much faster than the system clock.

In Slave mode, when MSSP interrupts are enabled, after the master completes sending data, an MSSP interrupt will wake the controller from Sleep.

If an exit from Sleep mode is not desired, MSSP 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 eight bits have been received, the MSSP interrupt flag bit will be set and if enabled, will wake the device.

TABLE 24-1: 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 |
|----------------------|--|-----------------------|--------|----------------|------------------|--------|--------|--------|------------------|
| ANSELA | — | — | — | ANSA4 | — | ANSA2 | ANSA1 | ANSA0 | 152 |
| ANSELC | ANSC7 ⁽²⁾ | ANSC6 ⁽²⁾ | — | — | ANSC3 | ANSC2 | ANSC1 | ANSC0 | 166 |
| INTCON | GIE | PEIE | TMR0IE | INTE | IOCIE | TMR0IF | INTF | IOCIF | 97 |
| PIE1 | TMR1GIE | ADIE | RCIE | TXIE | SSP1IE | CCP1IE | TMR2IE | TMR1IE | 98 |
| PIR1 | TMR1GIF | ADIF | RCIF | TXIF | SSP1IF | CCP1IF | TMR2IF | TMR1IF | 103 |
| RxyPPS | — | — | — | RxyPPS<4:0> | | | | | 172 |
| SSPCLKPPS | — | — | — | SSPCLKPPS<4:0> | | | | | 174, 172 |
| SSPDATPPS | — | — | — | SSPDATPPS<4:0> | | | | | 174, 172 |
| SSPSSPPS | — | — | — | SSPSSPPS<4:0> | | | | | 174, 172 |
| SSP1BUF | Synchronous Serial Port Receive Buffer/Transmit Register | | | | | | | | 264* |
| SSP1CON1 | WCOL | SSPOV | SSPEN | CKP | SSPM<3:0> | | | | 309 |
| SSP1CON3 | ACKTIM | PCIE | SCIE | BOEN | SDAHT | SBCDE | AHEN | DHEN | 308 |
| SSP1STAT | SMP | CKE | D/Ā | P | S | R/W | UA | BF | 308 |
| TRISA | — | — | TRISA5 | TRISA4 | — ⁽¹⁾ | TRISA2 | TRISA1 | TRISA0 | 151 |
| TRISB ⁽²⁾ | TRISB7 | TRISB6 | TRISB5 | TRISB4 | — | — | — | — | 158 |
| TRISC | TRISC7 ⁽²⁾ | TRISC6 ⁽²⁾ | TRISC5 | TRISC4 | TRISC3 | TRISC2 | TRISC1 | TRISC0 | 165 |

Legend: — = Unimplemented location, read as '0'. Shaded cells are not used by the MSSP in SPI mode.

* Page provides register information.

Note 1: Unimplemented, read as '1'.

2: PIC16(L)F1619 only.

24.3 I²C MODE OVERVIEW

The Inter-Integrated Circuit (I²C) bus 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 (SCL)
- Serial Data (SDA)

Figure 24-11 shows the block diagram of the MSSP module when operating in I²C mode.

Both the SCL and SDA 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 24-11 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:

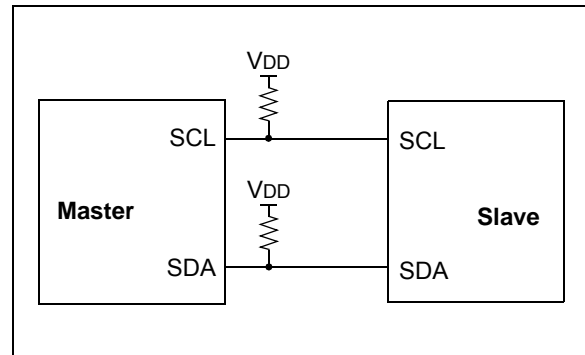
- Master Transmit mode
(master is transmitting data to a slave)
- Master Receive mode
(master is receiving data from a slave)
- Slave Transmit mode
(slave is transmitting data to a master)
- Slave Receive mode
(slave is receiving data from the master)

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 SDA line while the SCL 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 24-11: I²C MASTER/SLAVE CONNECTION



The Acknowledge bit ($\overline{\text{ACK}}$) is an active-low signal, which holds the SDA 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 SCL line is held low. Transitions that occur while the SCL 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 $\overline{\text{ACK}}$ bit. A Stop bit is indicated by a low-to-high transition of the SDA line while the SCL 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 SCL 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 SDA line, it is called arbitration. Arbitration ensures that there is only one master device communicating at any single time.

24.3.1 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 SCL 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 SCL line in order to transfer the next bit, but will detect that the clock line has not yet been released. Because the SCL 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.

24.3.2 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 SDA 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 SDA line.

For example, if one transmitter holds the SDA 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 SDA 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 SDA line. If this transmitter is also a master device, it also must stop driving the SCL 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 SDA 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.

24.4 I²C MODE OPERATION

All MSSP I²C communication is byte oriented and shifted out MSb first. Six SFR registers and two interrupt flags interface the module with the PIC[®] microcontroller and user software. Two pins, SDA and SCL, are exercised by the module to communicate with other external I²C devices.

24.4.1 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 eighth falling edge of the SCL line, the device outputting data on the SDA changes that pin to an input and reads in an acknowledge value on the next clock pulse.

The clock signal, SCL, is provided by the master. Data is valid to change while the SCL signal is low, and sampled on the rising edge of the clock. Changes on the SDA line while the SCL line is high define special conditions on the bus, explained below.

24.4.2 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.

24.4.3 SDA AND SCL PINS

Selection of any I²C mode with the SSPEN bit set, forces the SCL and SDA pins to be open-drain. These pins should be set by the user to inputs by setting the appropriate TRIS bits.

Note 1: Data is tied to output zero when an I²C mode is enabled.

- 2:** Any device pin can be selected for SDA and SCL functions with the PPS peripheral. These functions are bidirectional. The SDA input is selected with the SSPDATPPS registers. The SCL input is selected with the SSPCLKPPS registers. Outputs are selected with the RxyPPS registers. It is the user's responsibility to make the selections so that both the input and the output for each function is on the same pin.

24.4.4 SDA HOLD TIME

The hold time of the SDA pin is selected by the SDAHT bit of the SSPxCON3 register. Hold time is the time SDA is held valid after the falling edge of SCL. Setting the SDAHT bit selects a longer 300 ns minimum hold time and may help on buses with large capacitance.

TABLE 24-2: 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 SDA and SCL 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 holds SCL low to stall communication. |
| Bus Collision | Any time the SDA line is sampled low by the module while it is outputting and expected high state. |

24.4.5 START CONDITION

The I²C specification defines a Start condition as a transition of SDA from a high to a low state while SCL 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 24-12 shows wave forms for Start and Stop conditions.

A bus collision can occur on a Start condition if the module samples the SDA line low before asserting it low. This does not conform to the I²C Specification that states no bus collision can occur on a Start.

24.4.6 STOP CONDITION

A Stop condition is a transition of the SDA line from low-to-high state while the SCL line is high.

Note: At least one SCL low time must appear before a Stop is valid, therefore, if the SDA line goes low then high again while the SCL line stays high, only the Start condition is detected.

24.4.7 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 24-13 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.

24.4.8 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 24-12: I²C START AND STOP CONDITIONS

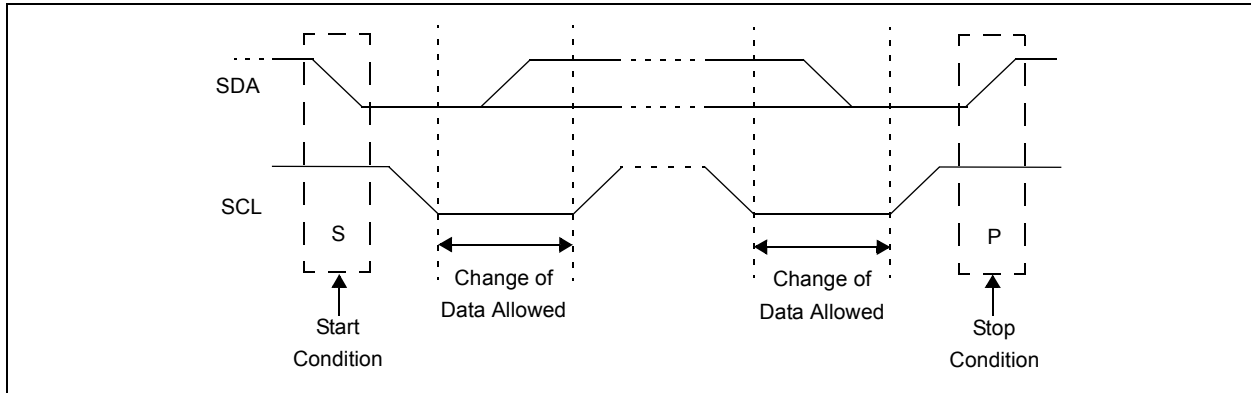
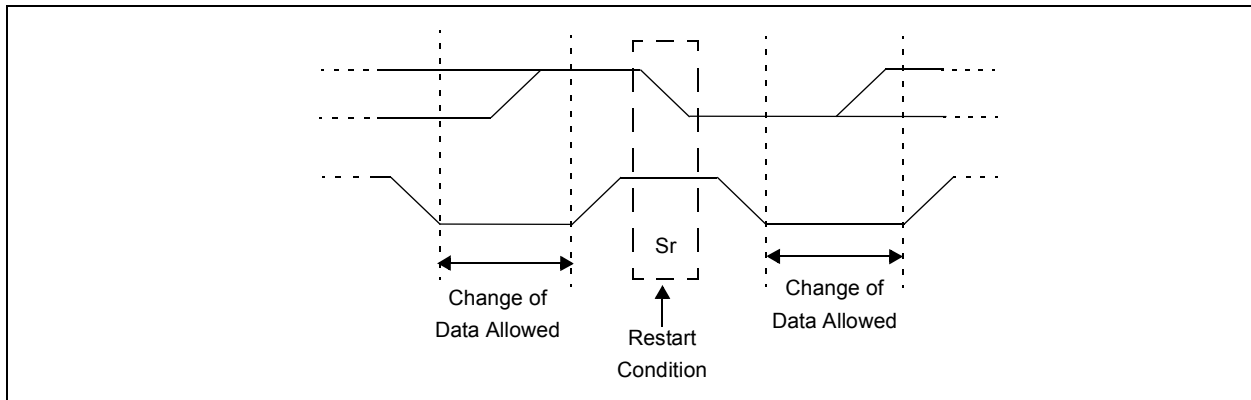


FIGURE 24-13: I²C RESTART CONDITION



24.4.9 ACKNOWLEDGE SEQUENCE

The 9th SCL 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 SDA line low. The transmitter must release control of the line during this time to shift in the response. The Acknowledge ($\overline{\text{ACK}}$) is an active-low signal, pulling the SDA line low indicates 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 eighth falling edge of SCL 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.

24.5 I²C SLAVE MODE OPERATION

The MSSP Slave mode operates in one of four modes selected by 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 operate the same as the other modes with SSPxIF additionally getting set upon detection of a Start, Restart, or Stop condition.

24.5.1 SLAVE MODE ADDRESSES

The SSPxADD register ([Register 24-6](#)) 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 SSP Mask register ([Register 24-5](#)) affects the address matching process. See [Section 24.5.8 “SSP Mask Register”](#) for more information.

24.5.1.1 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.

24.5.1.2 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 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 SCL is held low until the user updates SSPxADD with the low address. The low address byte is clocked in and all eight bits are compared to the low address value in SSPxADD. Even if there is not an address match; SSPxIF and UA are set, and SCL 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 $\overline{\text{R/W}}$ 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.

24.5.2 SLAVE RECEPTION

When the R/\overline{W} bit of a matching received address byte is clear, the R/\overline{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 24-4](#).

An MSSP 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, SCL 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 24.5.6.2 “10-bit Addressing Mode”](#) for more detail.

24.5.2.1 7-bit Addressing Reception

This section describes a standard sequence of events for the MSSP module configured as an I²C slave in 7-bit Addressing mode. [Figure 24-14](#) and [Figure 24-15](#) 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 R/\overline{W} bit clear is received.
4. The slave pulls SDA 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 SCL line.
8. The master clocks out a data byte.
9. Slave drives SDA 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.

24.5.2.2 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 eighth falling edge of SCL. 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 24-16](#) displays a module using both address and data holding. [Figure 24-17](#) 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 R/\overline{W} bit clear is clocked in. SSPxIF is set and CKP cleared after the eighth falling edge of SCL.
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 SCL 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 eighth falling edge of SCL 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 SSTSTAT register.

FIGURE 24-14: I²C SLAVE, 7-BIT ADDRESS, RECEPTION (SEN = 0, AHEN = 0, DHEN = 0)

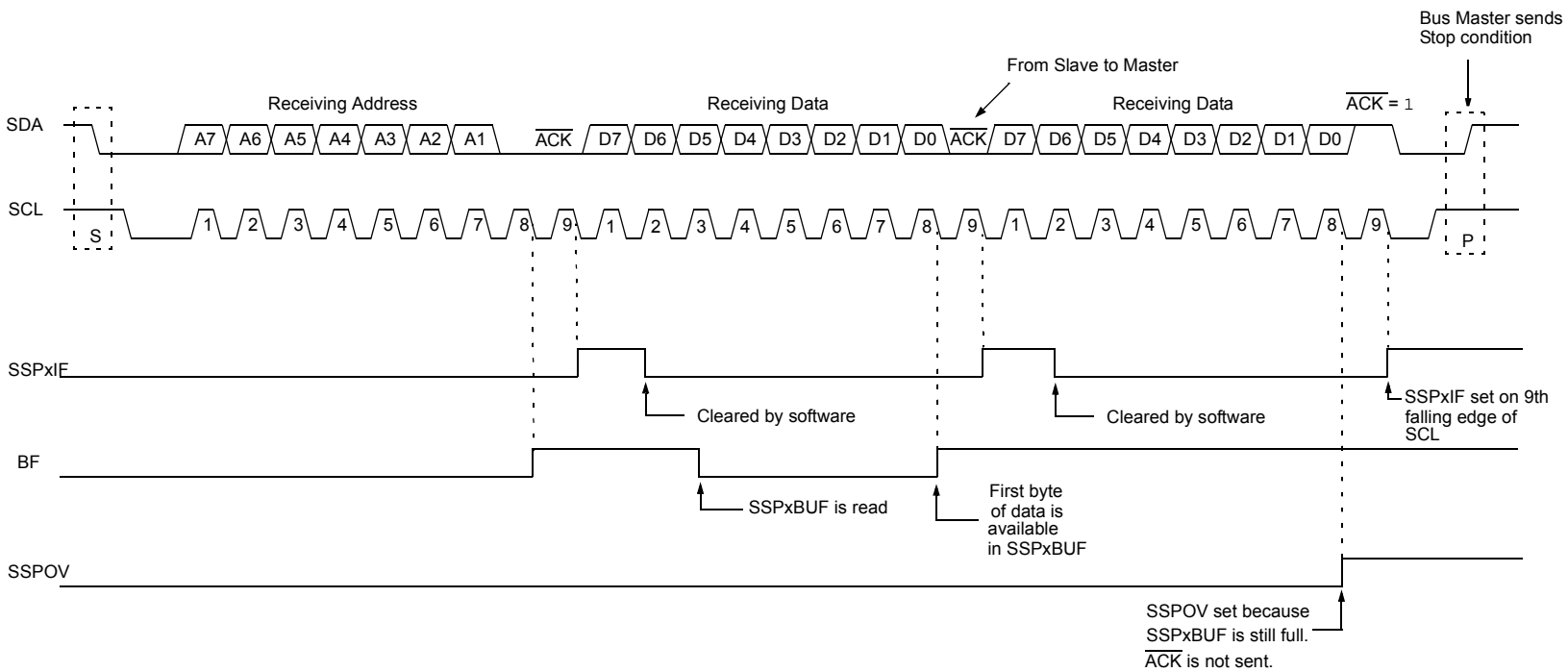


FIGURE 24-15: I²C SLAVE, 7-BIT ADDRESS, RECEPTION (SEN = 1, AHEN = 0, DHEN = 0)

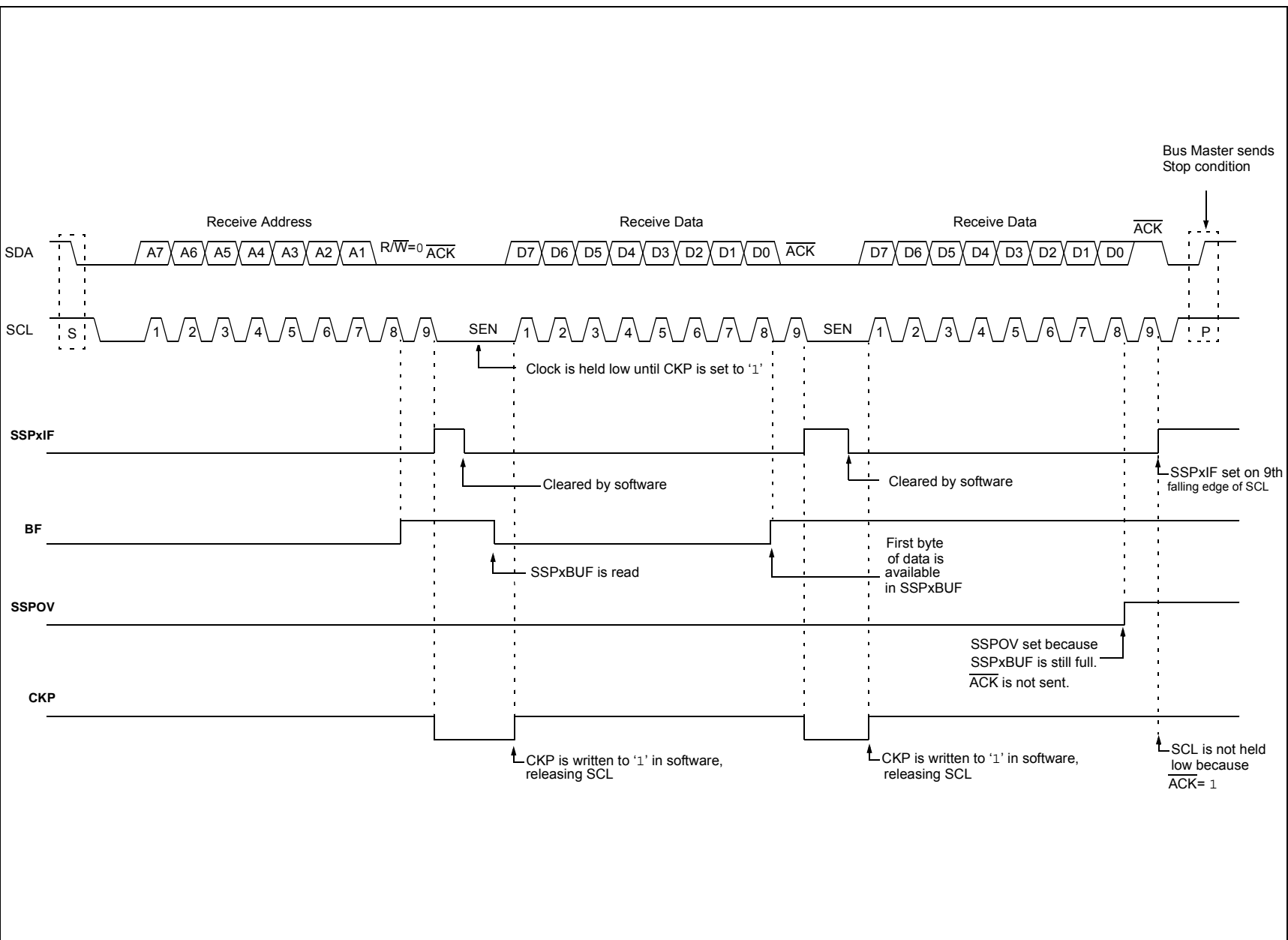


FIGURE 24-16: I²C SLAVE, 7-BIT ADDRESS, RECEPTION (SEN = 0, AHEN = 1, DHEN = 1)

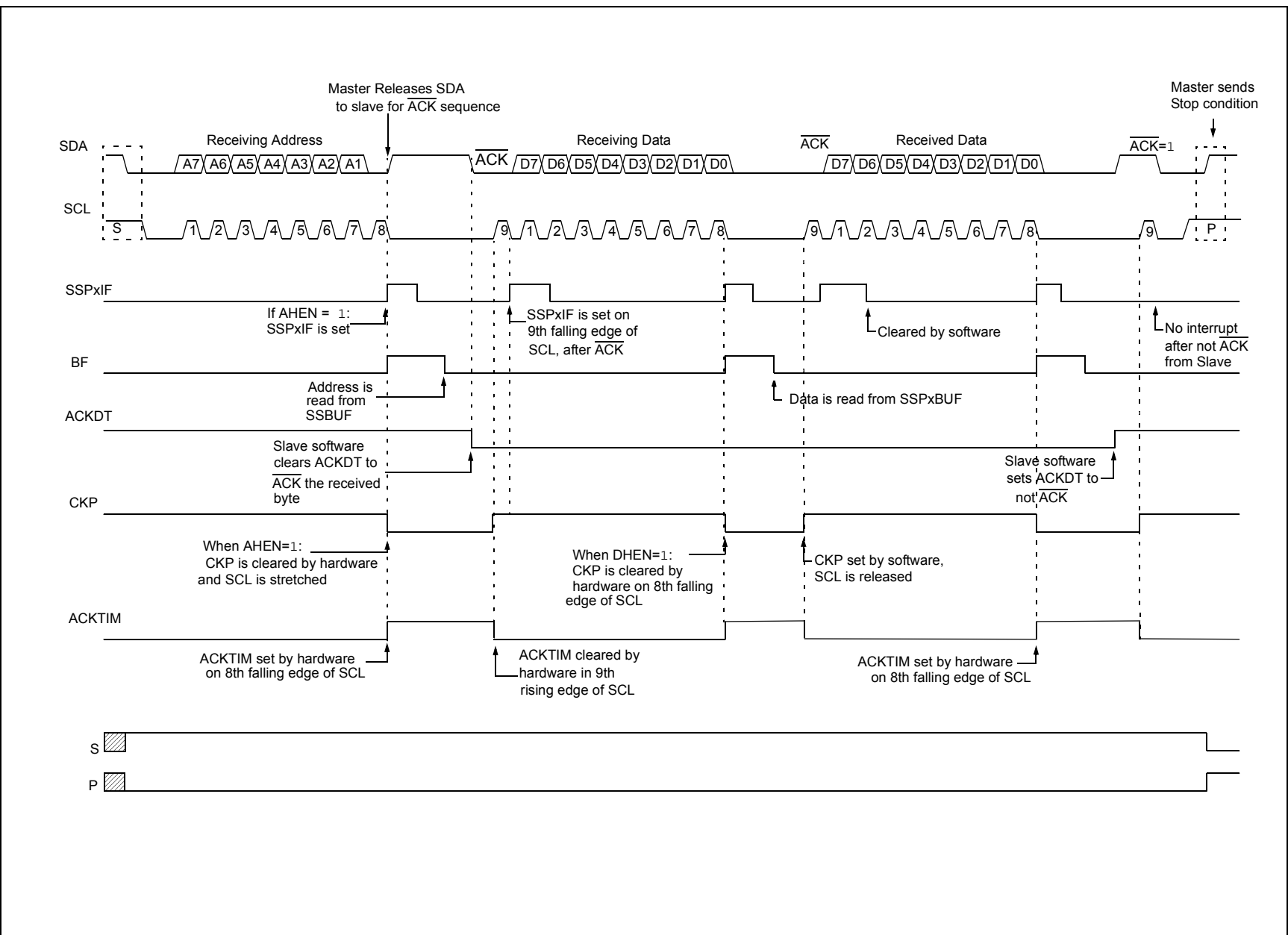
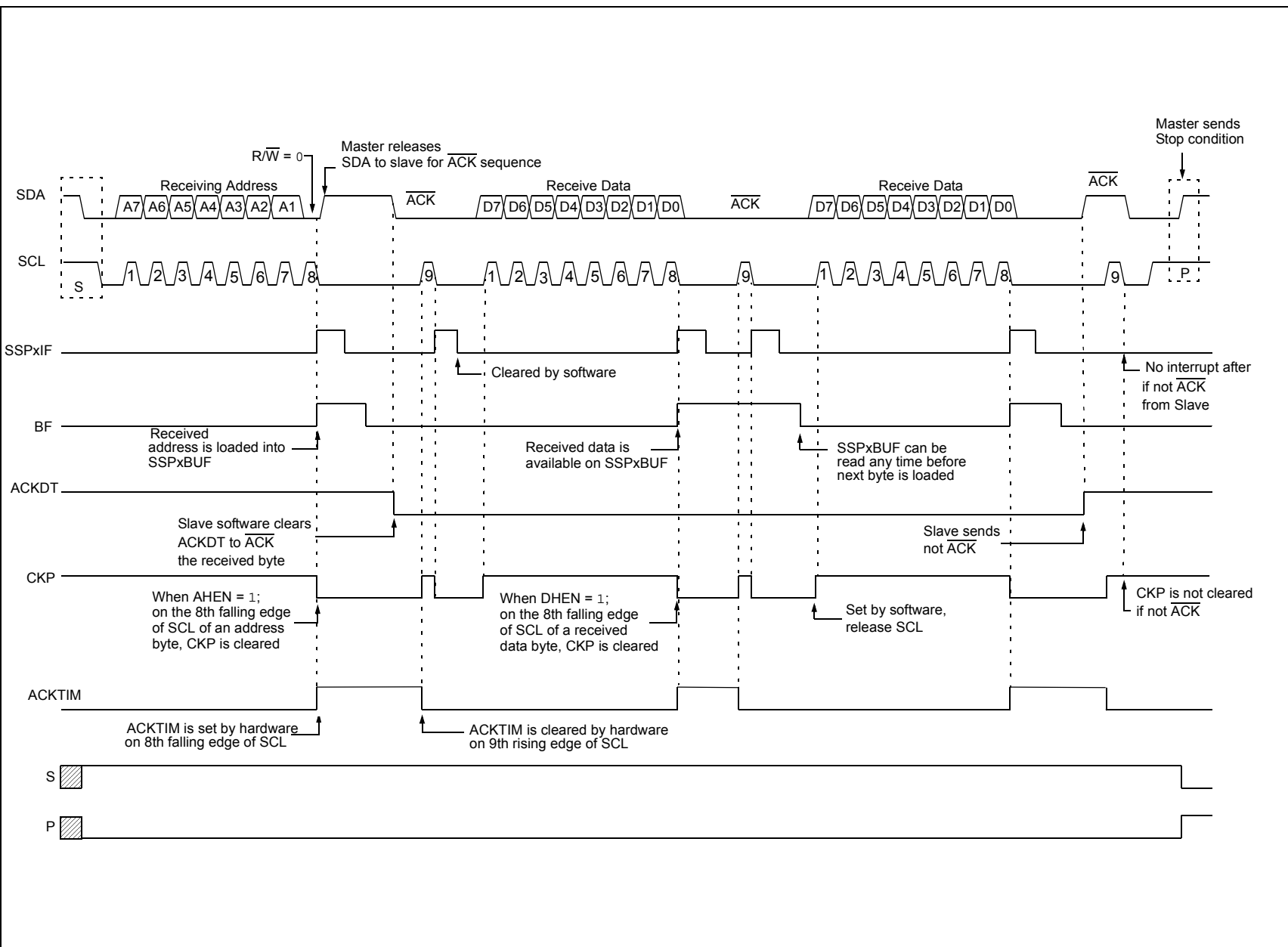


FIGURE 24-17: I²C SLAVE, 7-BIT ADDRESS, RECEPTION (SEN = 1, AHEN = 1, DHEN = 1)



24.5.3 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 SCL pin is held low (see [Section 24.5.6 “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 SSPSR register. Then the SCL 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 SCL input. This ensures that the SDA signal is valid during the SCL high time.

The \overline{ACK} pulse from the master-receiver is latched on the rising edge of the ninth SCL 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 SDA line was low (\overline{ACK}), the next transmit data must be loaded into the SSPxBUF register. Again, the SCL pin must be released by setting bit CKP.

An MSSP 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.

24.5.3.1 Slave Mode Bus Collision

A slave receives a Read request and begins shifting data out on the SDA line. If a bus collision is detected and the SBCDE bit of the SSPxCON3 register is set, the BCL1IF bit of the PIR2 register is set. Once a bus collision is detected, the slave goes idle and waits to be addressed again. User software can use the BCL1IF bit to handle a slave bus collision.

24.5.3.2 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 24-18](#) can be used as a reference to this list.

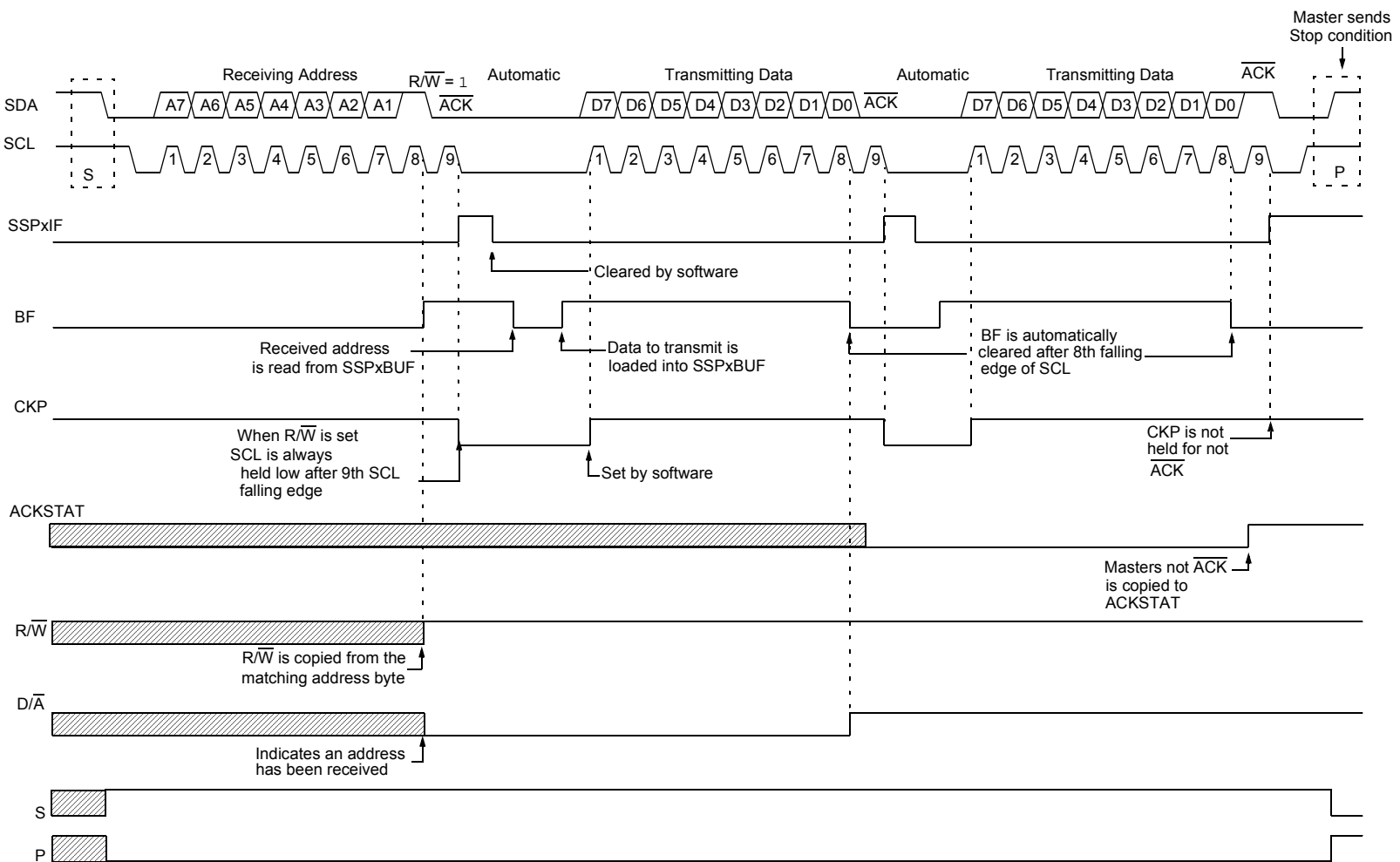
1. Master sends a Start condition on SDA and SCL.
2. S bit of SSPxSTAT is set; SSPxIF is set if interrupt on Start detect is enabled.
3. Matching address with $\overline{R/W}$ bit set is received by the Slave setting SSPxIF bit.
4. Slave hardware generates an \overline{ACK} and sets SSPxIF.
5. SSPxIF 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 SCL, 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 SCL (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.

FIGURE 24-18: I²C SLAVE, 7-BIT ADDRESS, TRANSMISSION (AHEN = 0)



24.5.3.3 7-bit Transmission with Address Hold Enabled

Setting the AHEN bit of the SSPxCON3 register enables additional clock stretching and interrupt generation after the eighth falling edge of a received address. Once a matching address has been clocked in, CKP is cleared and the SSPxIF interrupt is set.

Figure 24-19 displays a standard waveform of a 7-bit address slave transmission with AHEN enabled.

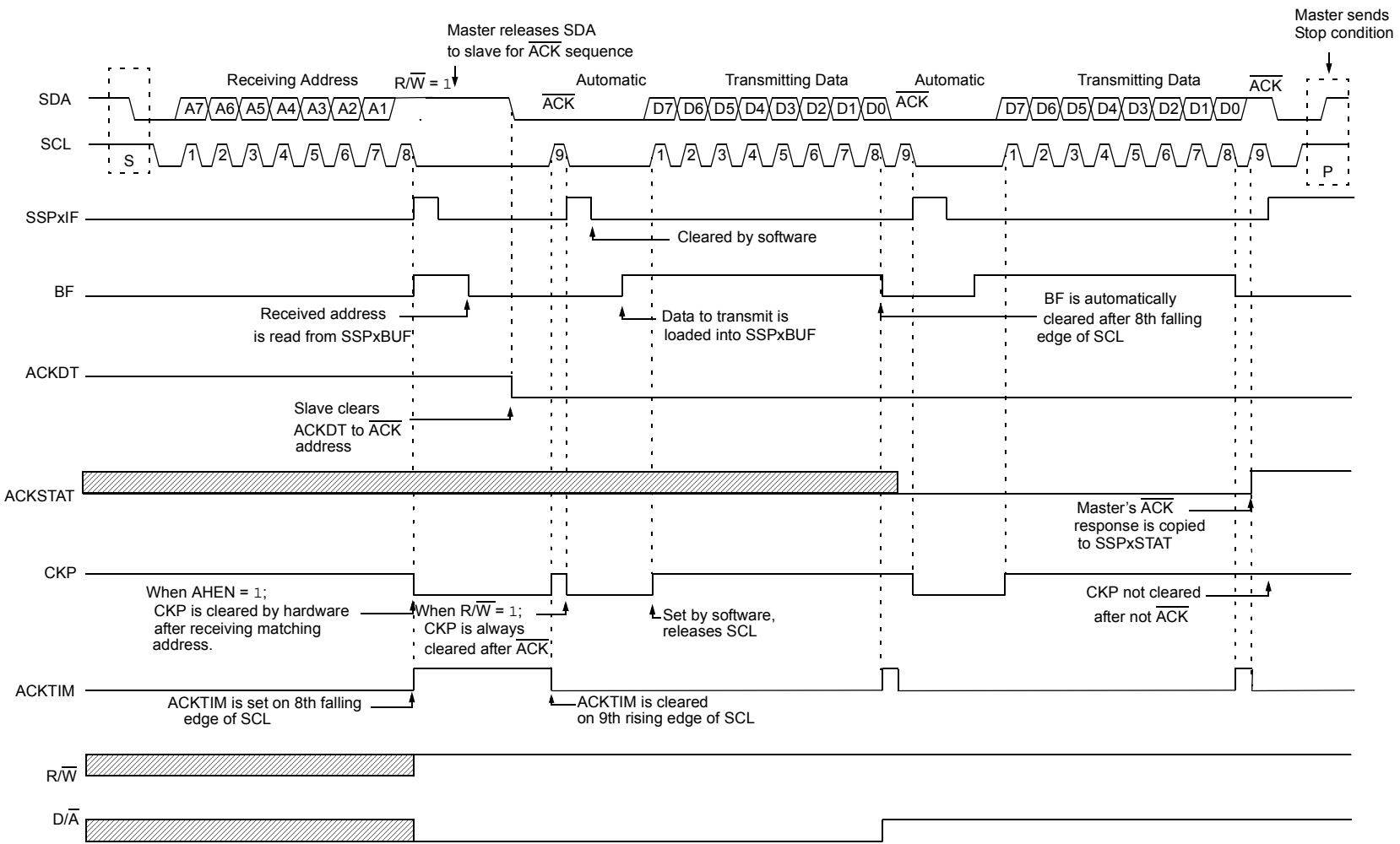
1. Bus starts Idle.
2. Master sends Start condition; the S bit of SSPxSTAT is set; SSPxIF is set if interrupt on Start detect is enabled.
3. Master sends matching address with $\overline{R/\overline{W}}$ bit set. After the eighth falling edge of the SCL line the CKP bit is cleared and SSPxIF interrupt is generated.
4. Slave software clears SSPxIF.
5. Slave software reads ACKTIM bit of SSPxCON3 register, and $\overline{R/\overline{W}}$ and $\overline{D/\overline{A}}$ 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 SCL.
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 $\overline{R/\overline{W}}$ 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 the CKP bit releasing the clock.
14. Master clocks out the data from the slave and sends an \overline{ACK} value on the 9th SCL 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 SCL line to receive a Stop.

FIGURE 24-19: I²C SLAVE, 7-BIT ADDRESS, TRANSMISSION (AHEN = 1)



24.5.4 SLAVE MODE 10-BIT ADDRESS RECEPTION

This section describes a standard sequence of events for the MSSP module configured as an I²C slave in 10-bit Addressing mode.

Figure 24-20 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 $\overline{R/\overline{W}}$ 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 SCL.
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 \overline{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 \overline{ACK} on the 9th SCL 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 SCL.
18. Steps 13-17 repeat for each received byte.
19. Master sends Stop to end the transmission.

24.5.5 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 SCL line is held low are the same. Figure 24-21 can be used as a reference of a slave in 10-bit addressing with AHEN set.

Figure 24-22 shows a standard waveform for a slave transmitter in 10-bit Addressing mode.

FIGURE 24-20: I²C SLAVE, 10-BIT ADDRESS, RECEPTION (SEN = 1, AHEN = 0, DHEN = 0)

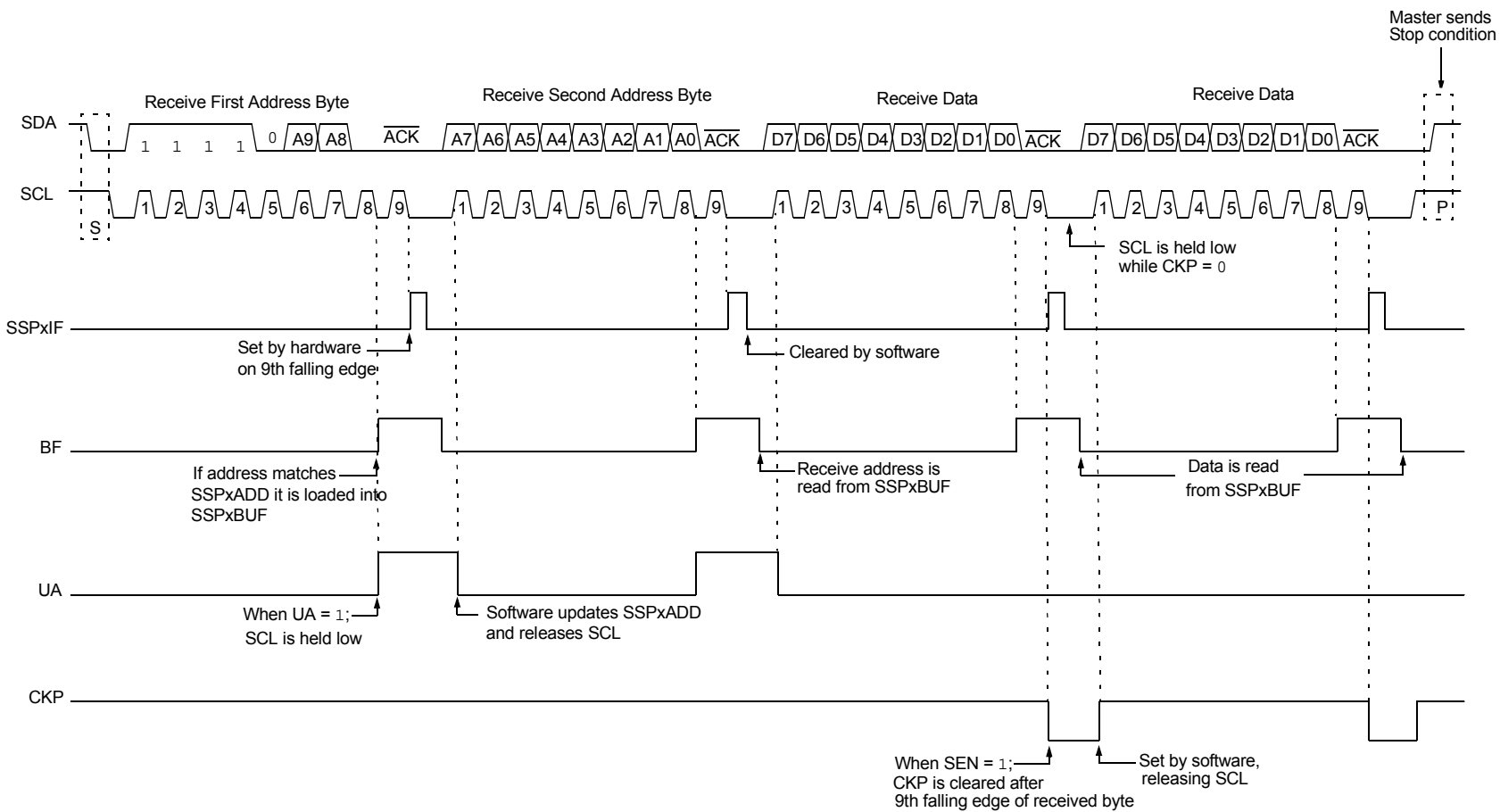


FIGURE 24-21: I²C SLAVE, 10-BIT ADDRESS, RECEPTION (SEN = 0, AHEN = 1, DHEN = 0)

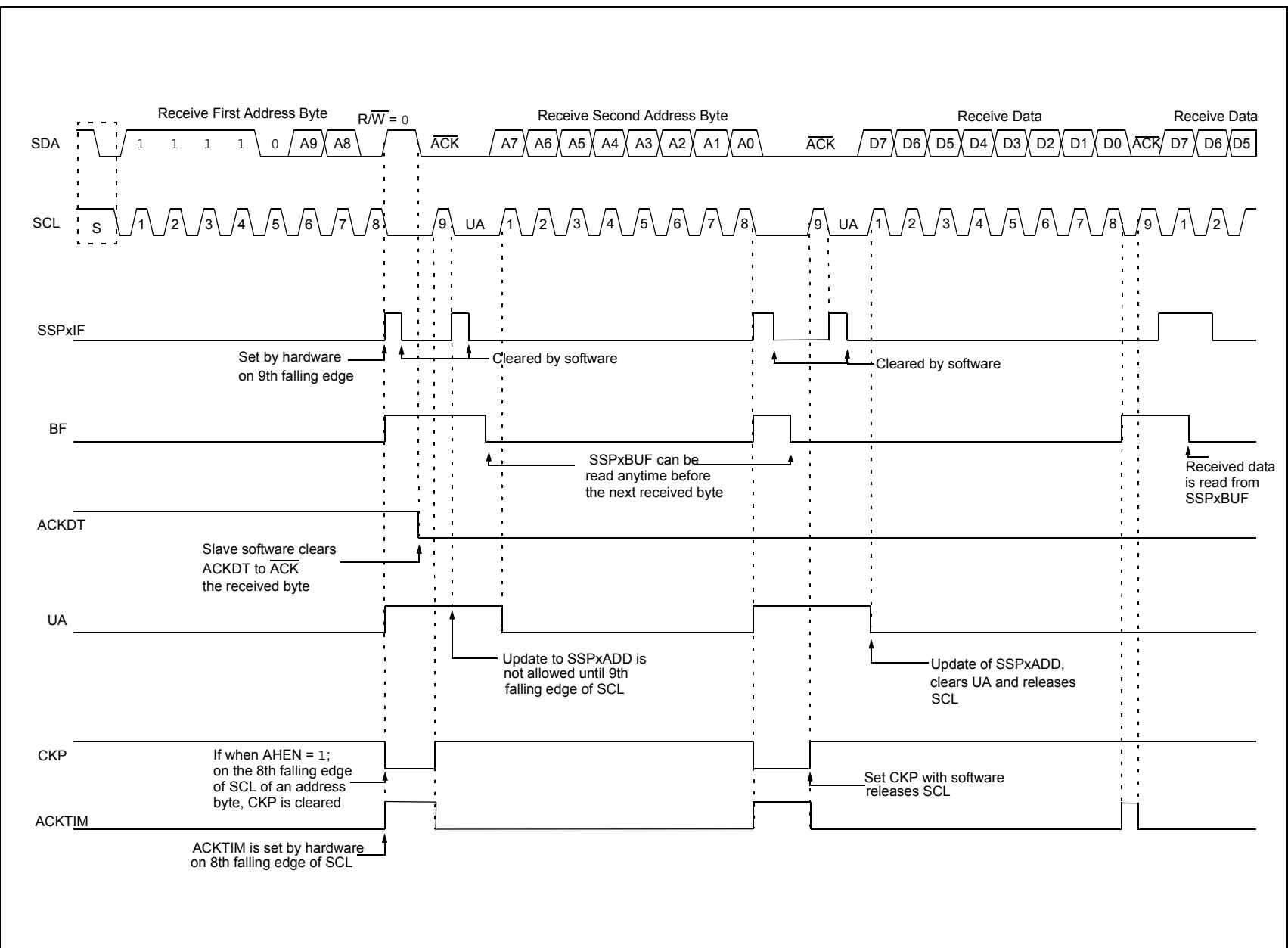
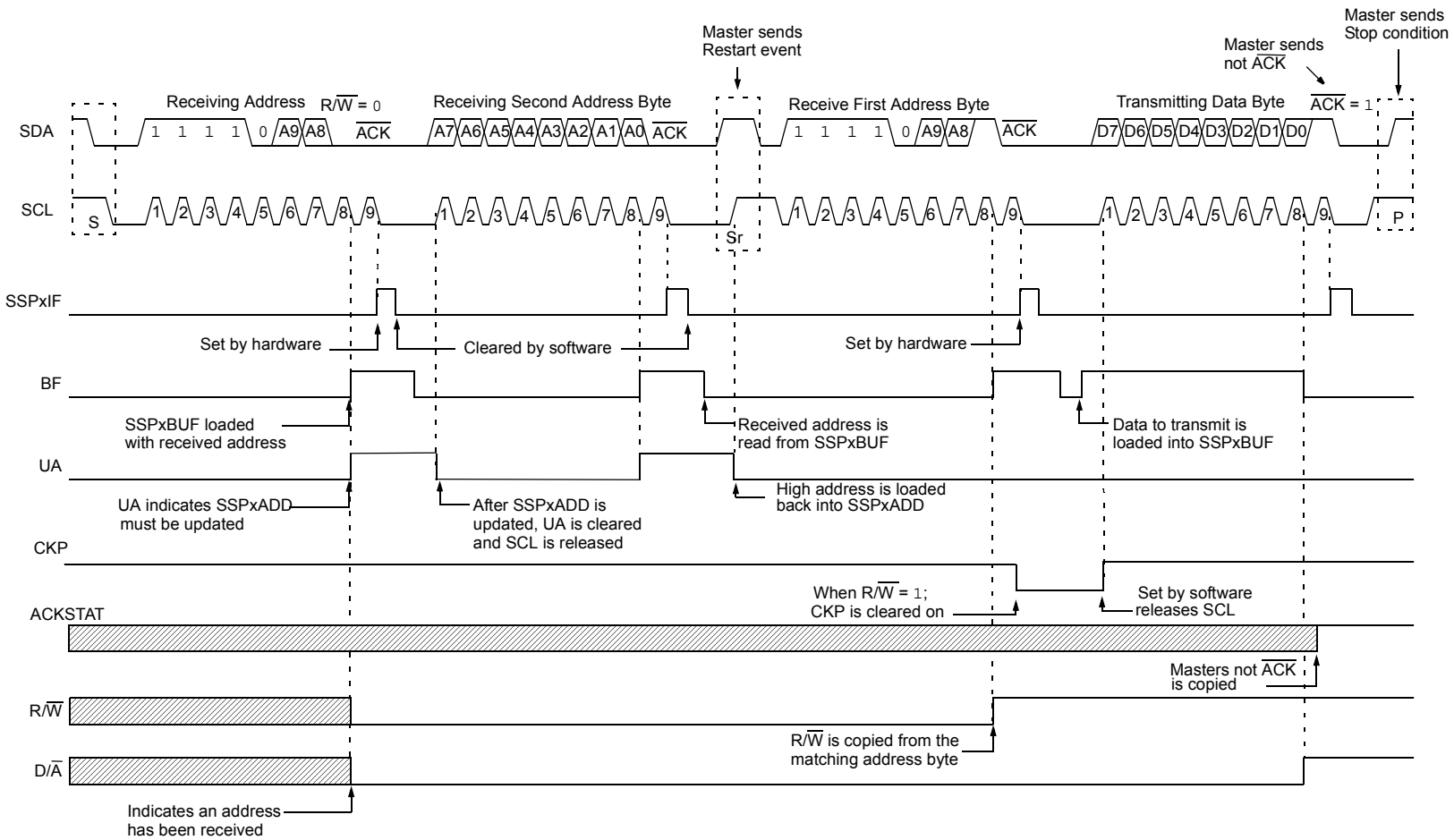


FIGURE 24-22: I²C SLAVE, 10-BIT ADDRESS, TRANSMISSION (SEN = 0, AHEN = 0, DHEN = 0)



24.5.6 CLOCK STRETCHING

Clock stretching occurs when a device on the bus holds the SCL 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 SCL.

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 SCL line to go low and then hold it. Setting CKP will release SCL and allow more communication.

24.5.6.1 Normal Clock Stretching

Following an $\overline{\text{ACK}}$ if the 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 SCL.
- 2:** Previous versions of the module did not stretch the clock for a transmission if SSPxBUF was loaded before the 9th falling edge of SCL. It is now always cleared for read requests.

24.5.6.2 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 SCL is stretched without CKP being cleared. SCL 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.

24.5.6.3 Byte NACKing

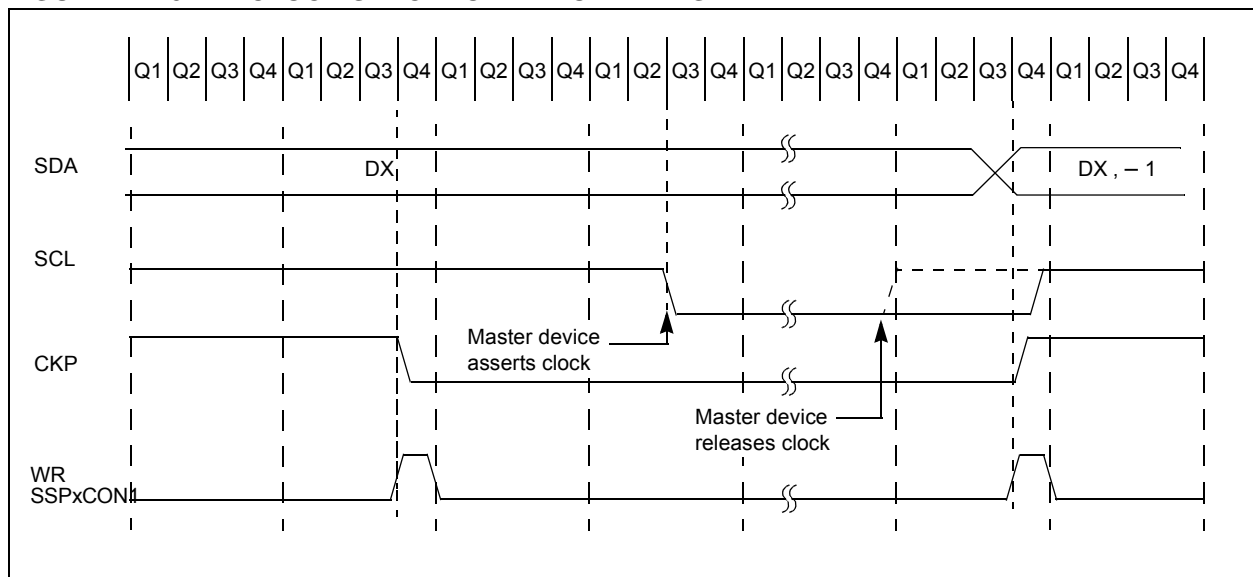
When the AHEN bit of SSPxCON3 is set, CKP is cleared by hardware after the eighth falling edge of SCL for a received matching address byte. When the DHEN bit of SSPxCON3 is set, CKP is cleared after the eighth falling edge of SCL for received data.

Stretching after the eighth falling edge of SCL allows the slave to look at the received address or data and decide if it wants to ACK the received data.

24.5.6.4 Clock Synchronization and the CKP Bit

Any time the CKP bit is cleared, the module will wait for the SCL line to go low and then hold it. However, clearing the CKP bit will not assert the SCL output low until the SCL output is already sampled low. Therefore, the CKP bit will not assert the SCL line until an external I²C master device has already asserted the SCL line. The SCL output will remain low until the CKP bit is set and all other devices on the I²C bus have released SCL. This ensures that a write to the CKP bit will not violate the minimum high time requirement for SCL (see [Figure 24-23](#)).

FIGURE 24-23: CLOCK SYNCHRONIZATION TIMING



24.5.7 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.

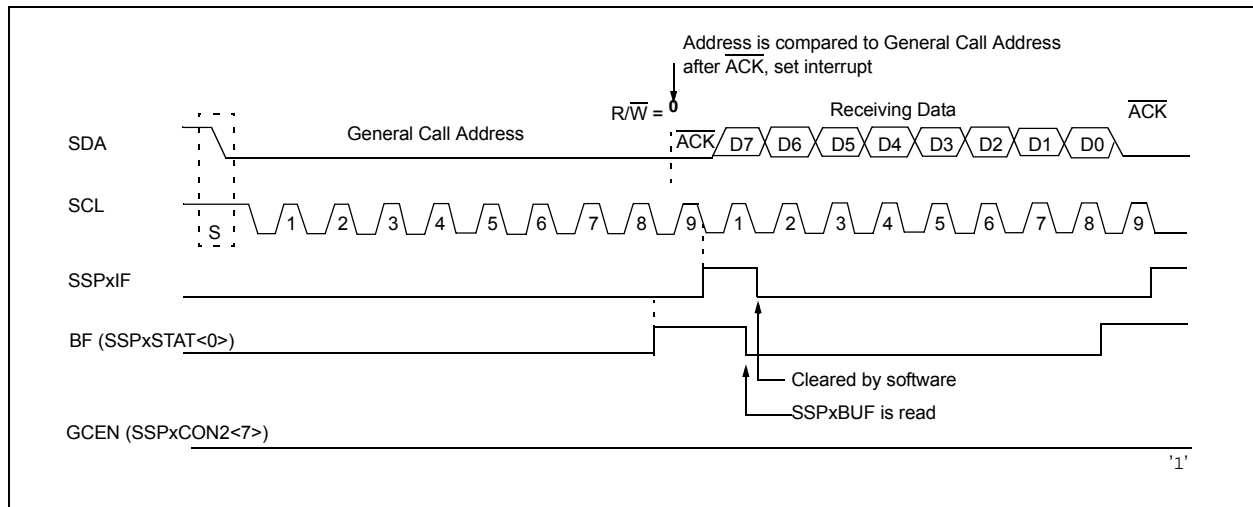
The general call address is a reserved address in the I²C protocol, defined as address 0x00. When the GCEN bit of the SSPxCON2 register is set, the slave module will automatically $\overline{\text{ACK}}$ the reception of this address regardless of the value stored in SSPxADD. After the slave clocks in an address of all zeros with the

R/\overline{W} bit clear, an interrupt is generated and slave software can read SSPxBUF and respond. Figure 24-24 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 eighth falling edge of SCL. The slave must then set its ACKDT value and release the clock with communication progressing as it would normally.

FIGURE 24-24: SLAVE MODE GENERAL CALL ADDRESS SEQUENCE



24.5.8 SSP MASK REGISTER

An SSP Mask (SSPxMSK) register (Register 24-5) is available in I²C Slave mode as a mask for the value held in the SSPSR 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 SSP operation until written with a mask value.

The SSP 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 SSP mask has no effect during the reception of the first (high) byte of the address.

24.6 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 MSSP 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 SDA and SCL lines.

The following events will cause the SSP Interrupt Flag bit, SSPxIF, to be set (SSP interrupt, if enabled):

- Start condition detected
- Stop condition detected
- Data transfer byte transmitted/received
- Acknowledge transmitted/received
- Repeated Start generated

Note 1: The MSSP 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:** Master mode suspends Start/Stop detection when sending the Start/Stop condition by means of the SEN/PEN control bits. The SSPxIF bit is set at the end of the Start/Stop generation when hardware clears the control bit.

24.6.1 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 SDA, while SCL 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 eight 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.

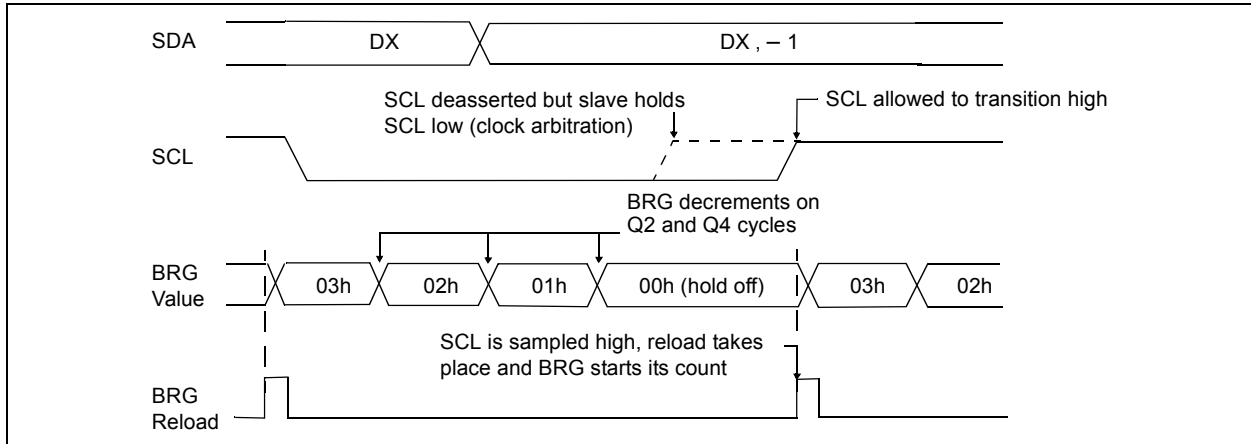
In Master Receive mode, the first byte transmitted contains the slave address of the transmitting device (7 bits) and the R/W bit. In this case, the R/W bit will be logic '1'. Thus, the first byte transmitted is a 7-bit slave address followed by a '1' to indicate the receive bit. Serial data is received via SDA, while SCL outputs the serial clock. Serial data is received eight 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 SCL. See [Section 24.7 "Baud Rate Generator"](#) for more detail.

24.6.2 CLOCK ARBITRATION

Clock arbitration occurs when the master, during any receive, transmit or Repeated Start/Stop condition, releases the SCL pin (SCL allowed to float high). When the SCL pin is allowed to float high, the Baud Rate Generator (BRG) is suspended from counting until the SCL pin is actually sampled high. When the SCL pin is sampled high, the Baud Rate Generator is reloaded with the contents of SSPxADD<7:0> and begins counting. This ensures that the SCL high time will always be at least one BRG rollover count in the event that the clock is held low by an external device (Figure 24-25).

FIGURE 24-25: BAUD RATE GENERATOR TIMING WITH CLOCK ARBITRATION



24.6.3 WCOL STATUS FLAG

If the user writes the SSPxBUF when a Start, Restart, Stop, Receive or Transmit sequence is in progress, the WCOL bit 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 queuing of events is not allowed, writing to the lower five bits of SSPxCON2 is disabled until the Start condition is complete.

24.6.4 I²C MASTER MODE START CONDITION TIMING

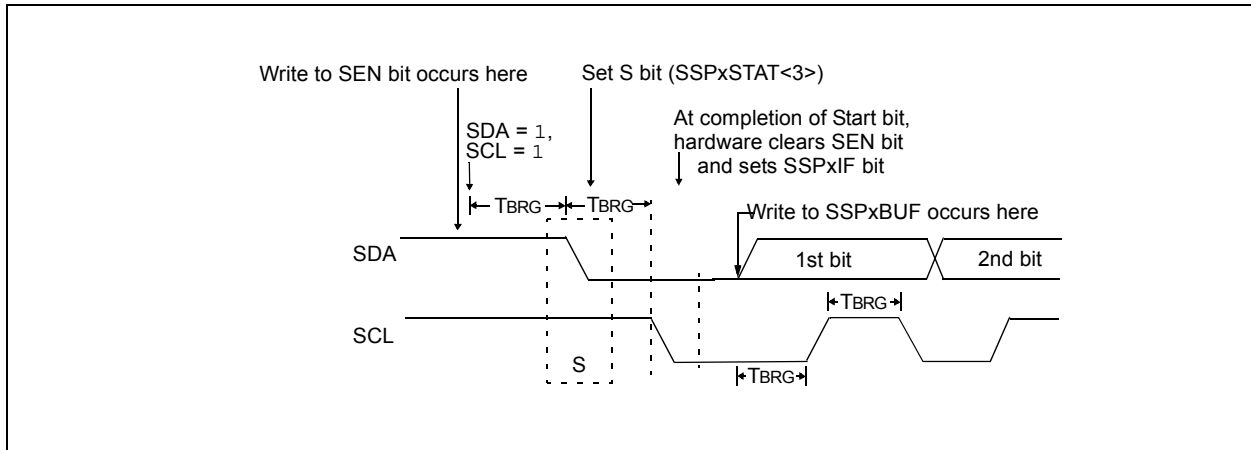
To initiate a Start condition (Figure 24-26), the user sets the Start Enable bit, SEN bit of the SSPxCON2 register. If the SDA and SCL pins are sampled high, the Baud Rate Generator is reloaded with the contents of SSPxADD<7:0> and starts its count. If SCL and SDA are both sampled high when the Baud Rate Generator times out (TBRG), the SDA pin is driven low. The action of the SDA being driven low while SCL 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 SDA line held low and the Start condition is complete.

Note 1: If at the beginning of the Start condition, the SDA and SCL pins are already sampled low, or if during the Start condition, the SCL line is sampled low before the SDA line is driven low, a bus collision occurs, the Bus Collision Interrupt Flag, BCL1IF, 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 24-26: FIRST START BIT TIMING



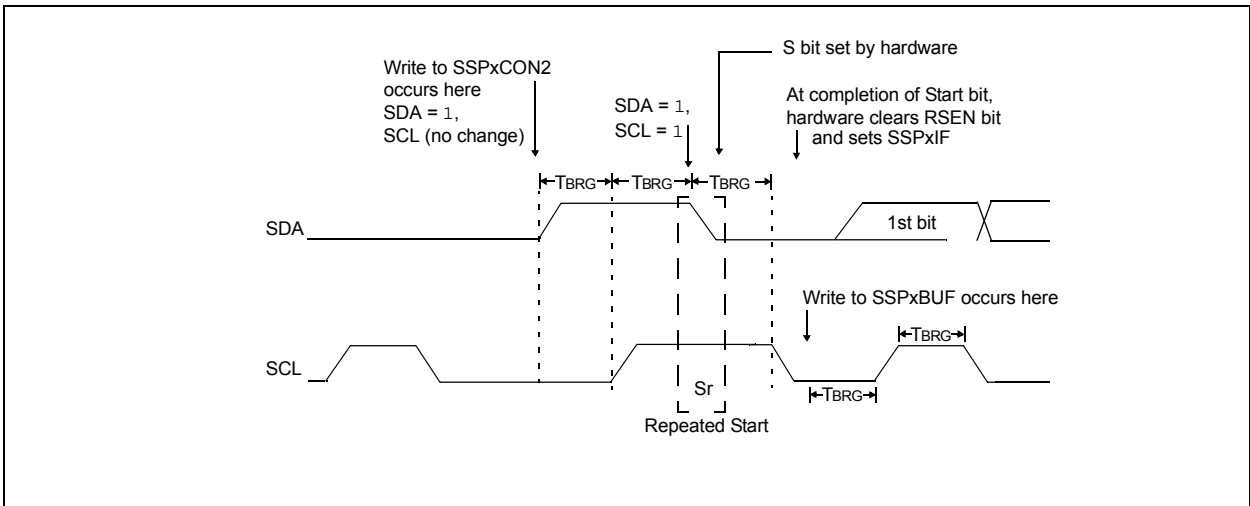
24.6.5 I²C MASTER MODE REPEATED START CONDITION TIMING

A Repeated Start condition (Figure 24-27) occurs when the RSEN bit of the SSPxCON2 register is programmed high and the master state machine is no longer active. When the RSEN bit is set, the SCL pin is asserted low. When the SCL pin is sampled low, the Baud Rate Generator is loaded and begins counting. The SDA pin is released (brought high) for one Baud Rate Generator count (TBRG). When the Baud Rate Generator times out, if SDA is sampled high, the SCL pin will be deasserted (brought high). When SCL is sampled high, the Baud Rate Generator is reloaded and begins counting. SDA and SCL must be sampled high for one TBRG. This action is then followed by assertion of the SDA pin (SDA = 0) for one TBRG while SCL is high. SCL is asserted low. Following this, the RSEN bit of the SSPxCON2 register will be automati-

cally cleared and the Baud Rate Generator will not be reloaded, leaving the SDA pin held low. As soon as a Start condition is detected on the SDA and SCL pins, the S bit of the SSPxSTAT register will be set. The SSPxIF bit will not be set until the Baud Rate Generator has timed out.

- Note 1:** If RSEN is programmed while any other event is in progress, it will not take effect.
- 2:** A bus collision during the Repeated Start condition occurs if:
- SDA is sampled low when SCL goes from low-to-high.
 - SCL goes low before SDA is asserted low. This may indicate that another master is attempting to transmit a data '1'.

FIGURE 24-27: REPEATED START CONDITION WAVEFORM



24.6.6 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 SDA pin after the falling edge of SCL is asserted. SCL is held low for one Baud Rate Generator rollover count (TBRG). Data should be valid before SCL is released high. When the SCL pin is released high, it is held that way for TBRG. The data on the SDA pin must remain stable for that duration and some hold time after the next falling edge of SCL. After the eighth bit is shifted out (the falling edge of the eighth clock), the BF flag is cleared and the master releases SDA. 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 SCL low and SDA unchanged (Figure 24-28).

After the write to the SSPxBUF, each bit of the address will be shifted out on the falling edge of SCL 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 SDA pin, allowing the slave to respond with an Acknowledge. On the falling edge of the ninth clock, the master will sample the SDA 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 SCL low and allowing SDA to float.

24.6.6.1 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 eight bits are shifted out.

24.6.6.2 WCOL Status Flag

If the user writes the SSPxBUF when a transmit is already in progress (i.e., SSPSR 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.

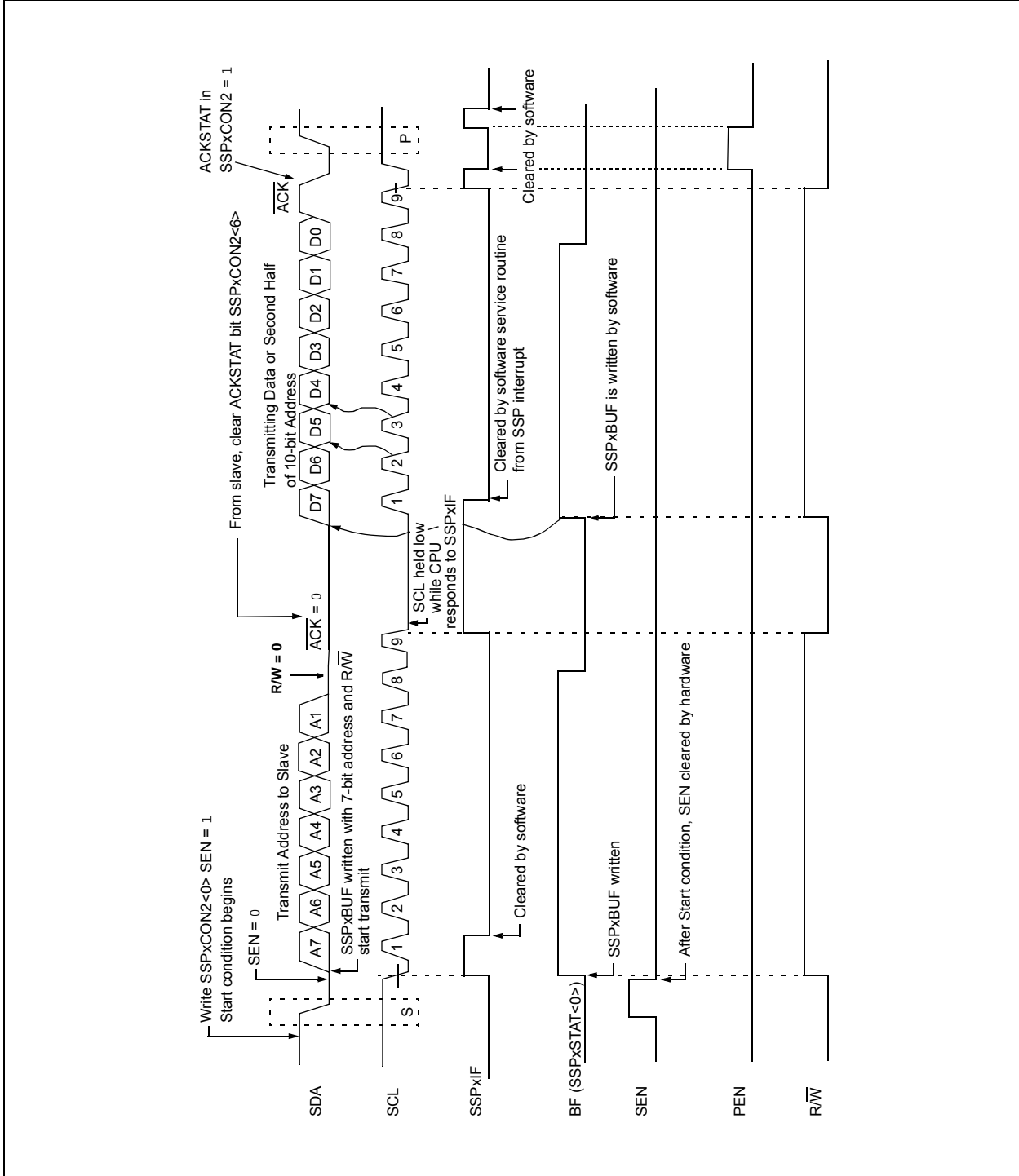
24.6.6.3 ACKSTAT Status Flag

In Transmit mode, the ACKSTAT bit of the SSPxCON2 register is cleared when the slave has sent an Acknowledge ($\overline{\text{ACK}} = 0$) and is set when the slave does not Acknowledge ($\overline{\text{ACK}} = 1$). A slave sends an Acknowledge when it has recognized its address (including a general call), or when the slave has properly received its data.

24.6.6.4 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 MSSP 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 SDA pin until all eight bits are transmitted. Transmission begins as soon as SSPxBUF is written to.
7. The MSSP 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 MSSP 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 SDA pin until all eight bits are transmitted.
11. The MSSP 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.

FIGURE 24-28: I²C MASTER MODE WAVEFORM (TRANSMISSION, 7 OR 10-BIT ADDRESS)



24.6.7 I²C MASTER MODE RECEPTION

Master mode reception (Figure 24-29) is enabled by programming the Receive Enable bit, RCEN bit of the SSPxCON2 register.

Note: The MSSP 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 SCL pin changes (high-to-low/low-to-high) and data is shifted into the SSPSR. After the falling edge of the eighth clock, the receive enable flag is automatically cleared, the contents of the SSPSR 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 SCL low. The MSSP 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.

24.6.7.1 BF Status Flag

In receive operation, the BF bit is set when an address or data byte is loaded into SSPxBUF from SSPSR. It is cleared when the SSPxBUF register is read.

24.6.7.2 SSPOV Status Flag

In receive operation, the SSPOV bit is set when eight bits are received into the SSPSR and the BF flag bit is already set from a previous reception.

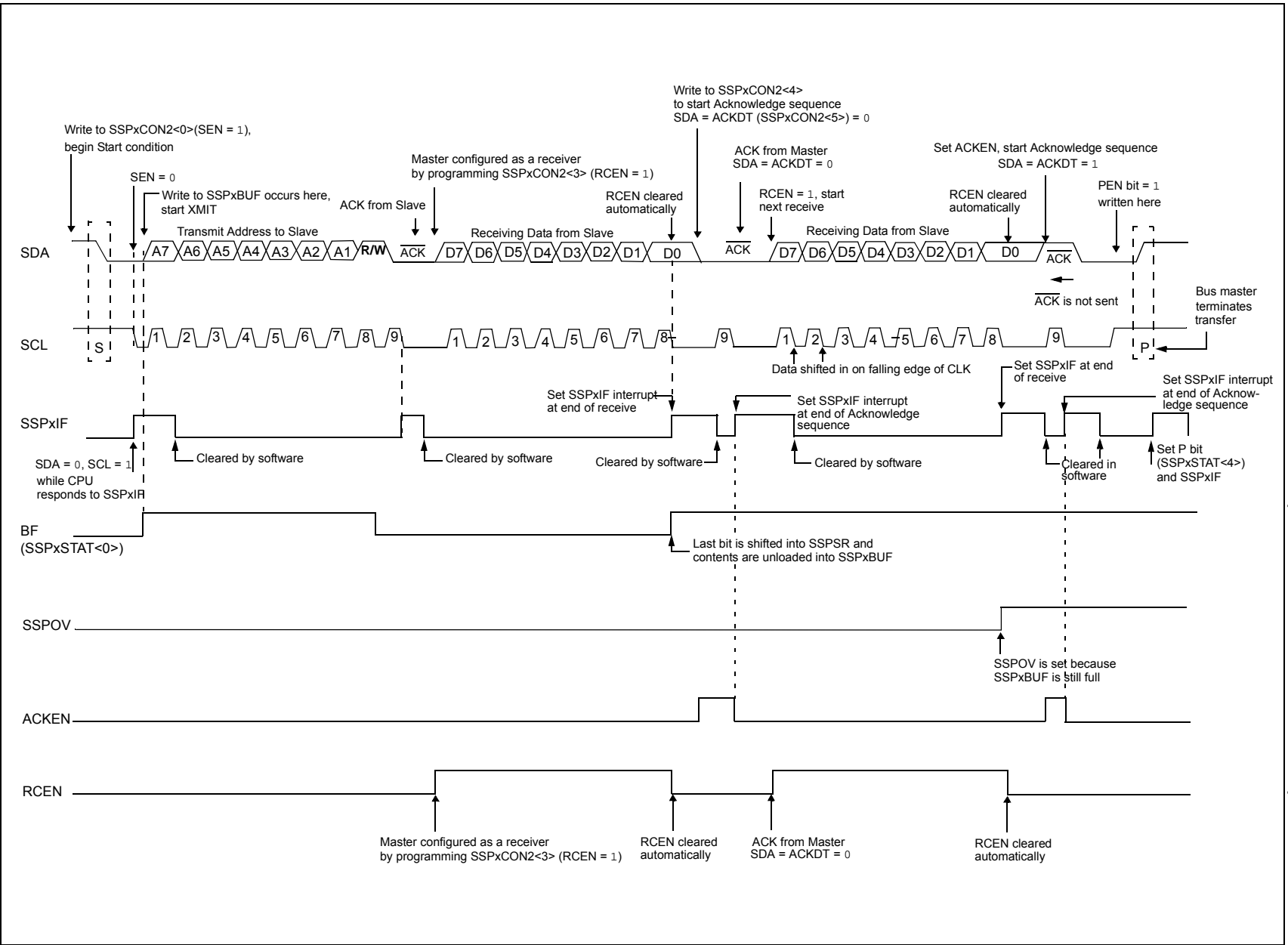
24.6.7.3 WCOL Status Flag

If the user writes the SSPxBUF when a receive is already in progress (i.e., SSPSR 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).

24.6.7.4 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 SDA pin until all eight bits are transmitted. Transmission begins as soon as SSPxBUF is written to.
6. The MSSP module shifts in the $\overline{\text{ACK}}$ bit from the slave device and writes its value into the ACKSTAT bit of the SSPxCON2 register.
7. The MSSP 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 eighth falling edge of SCL, SSPxIF and BF are set.
10. Master clears SSPxIF and reads the received byte from SSPxBUF, clears BF.
11. Master sets $\overline{\text{ACK}}$ value sent to slave in ACKDT bit of the SSPxCON2 register and initiates the ACK by setting the ACKEN bit.
12. Master's $\overline{\text{ACK}}$ 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 $\overline{\text{ACK}}$ or Stop to end communication.

FIGURE 24-29: I²C MASTER MODE WAVEFORM (RECEPTION, 7-BIT ADDRESS)



24.6.8 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 SCL pin is pulled low and the contents of the Acknowledge data bit are presented on the SDA 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 SCL pin is deasserted (pulled high). When the SCL pin is sampled high (clock arbitration), the Baud Rate Generator counts for TBRG. The SCL pin is then pulled low. Following this, the ACKEN bit is automatically cleared, the Baud Rate Generator is turned off and the MSSP module then goes into Idle mode (Figure 24-30).

24.6.8.1 WCOL Status Flag

If the user writes the SSPxBUF when an Acknowledge sequence is in progress, then WCOL bit is set and the contents of the buffer are unchanged (the write does not occur).

24.6.9 STOP CONDITION TIMING

A Stop bit is asserted on the SDA 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 SCL line is held low after the falling edge of the ninth clock. When the PEN bit is set, the master will assert the SDA line low. When the SDA line is sampled low, the Baud Rate Generator is reloaded and counts down to '0'. When the Baud Rate Generator times out, the SCL pin will be brought high and one TBRG (Baud Rate Generator rollover count) later, the SDA pin will be deasserted. When the SDA pin is sampled high while SCL 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 24-31).

24.6.9.1 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 24-30: ACKNOWLEDGE SEQUENCE WAVEFORM

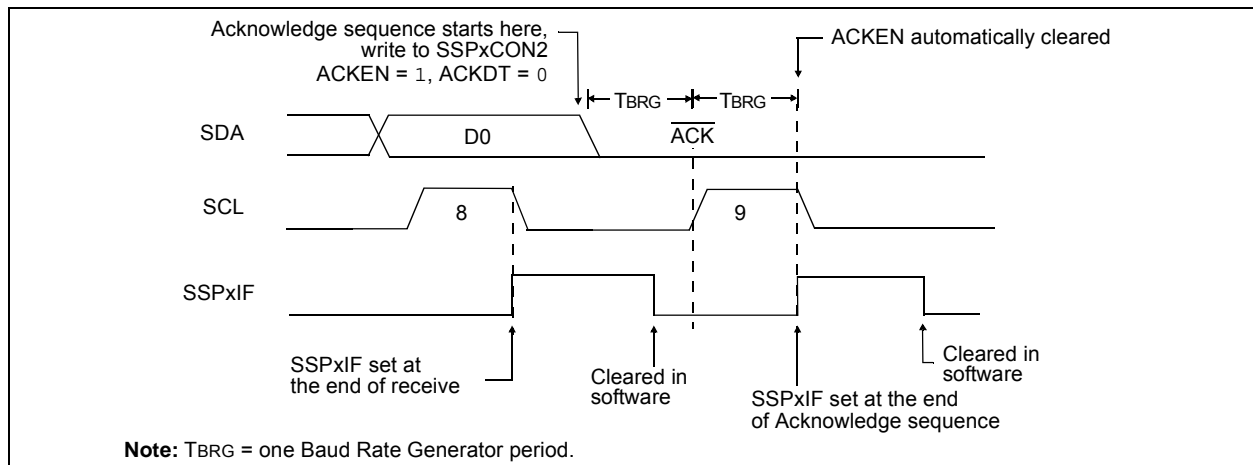
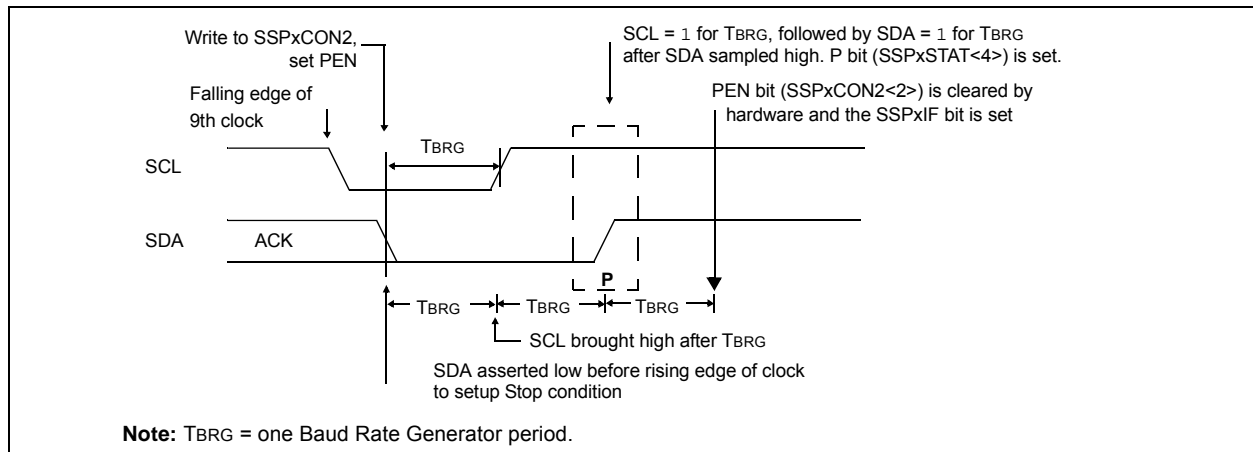


FIGURE 24-31: STOP CONDITION RECEIVE OR TRANSMIT MODE



24.6.10 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 MSSP interrupt is enabled).

24.6.11 EFFECTS OF A RESET

A Reset disables the MSSP module and terminates the current transfer.

24.6.12 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 MSSP 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 SSP interrupt will generate the interrupt when the Stop condition occurs.

In multi-master operation, the SDA 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 BCL1IF bit.

The states where arbitration can be lost are:

- Address Transfer
- Data Transfer
- A Start Condition
- A Repeated Start Condition
- An Acknowledge Condition

24.6.13 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 SDA pin, arbitration takes place when the master outputs a '1' on SDA, by letting SDA float high and another master asserts a '0'. When the SCL pin floats high, data should be stable. If the expected data on SDA is a '1' and the data sampled on the SDA pin is '0', then a bus collision has taken place. The master will set the Bus Collision Interrupt Flag, BCL1IF and reset the I²C port to its Idle state (Figure 24-32).

If a transmit was in progress when the bus collision occurred, the transmission is halted, the BF flag is cleared, the SDA and SCL 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.

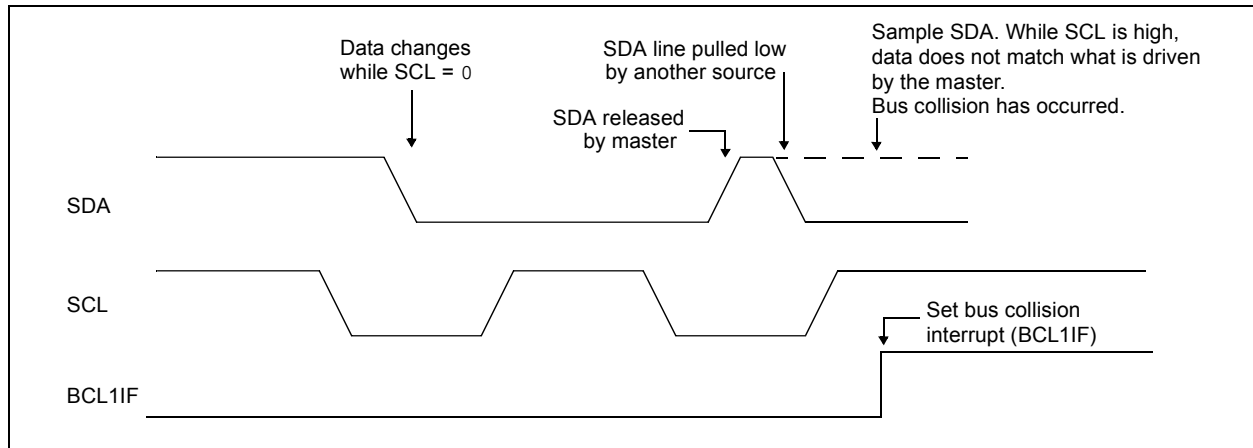
If a Start, Repeated Start, Stop or Acknowledge condition was in progress when the bus collision occurred, the condition is aborted, the SDA and SCL lines are deasserted and the respective control bits in the SSPxCON2 register are cleared. When the user services the bus collision Interrupt Service Routine and if the I²C bus is free, the user can resume communication by asserting a Start condition.

The master will continue to monitor the SDA and SCL 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 24-32: BUS COLLISION TIMING FOR TRANSMIT AND ACKNOWLEDGE



24.6.13.1 Bus Collision During a Start Condition

During a Start condition, a bus collision occurs if:

- SDA or SCL are sampled low at the beginning of the Start condition (Figure 24-33).
- SCL is sampled low before SDA is asserted low (Figure 24-34).

During a Start condition, both the SDA and the SCL pins are monitored.

If the SDA pin is already low, or the SCL pin is already low, then all of the following occur:

- the Start condition is aborted,
- the BCL1IF flag is set and
- the MSSP module is reset to its Idle state (Figure 24-33).

The Start condition begins with the SDA and SCL pins deasserted. When the SDA pin is sampled high, the Baud Rate Generator is loaded and counts down. If the SCL pin is sampled low while SDA 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 SDA pin is sampled low during this count, the BRG is reset and the SDA line is asserted early (Figure 24-35). If, however, a '1' is sampled on the SDA pin, the SDA 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 SCL pin is sampled as '0' during this time, a bus collision does not occur. At the end of the BRG count, the SCL 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 SDA 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 24-33: BUS COLLISION DURING START CONDITION (SDA ONLY)

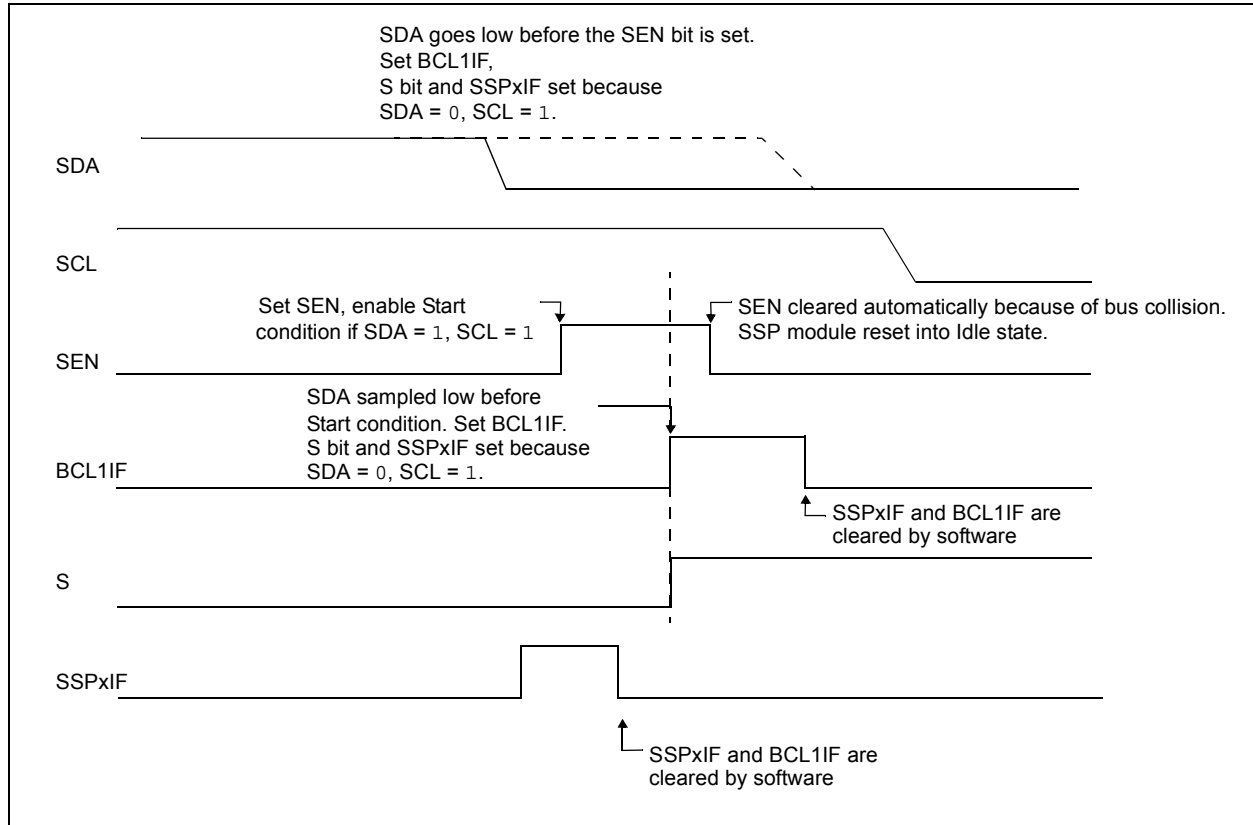


FIGURE 24-34: BUS COLLISION DURING START CONDITION (SCL = 0)

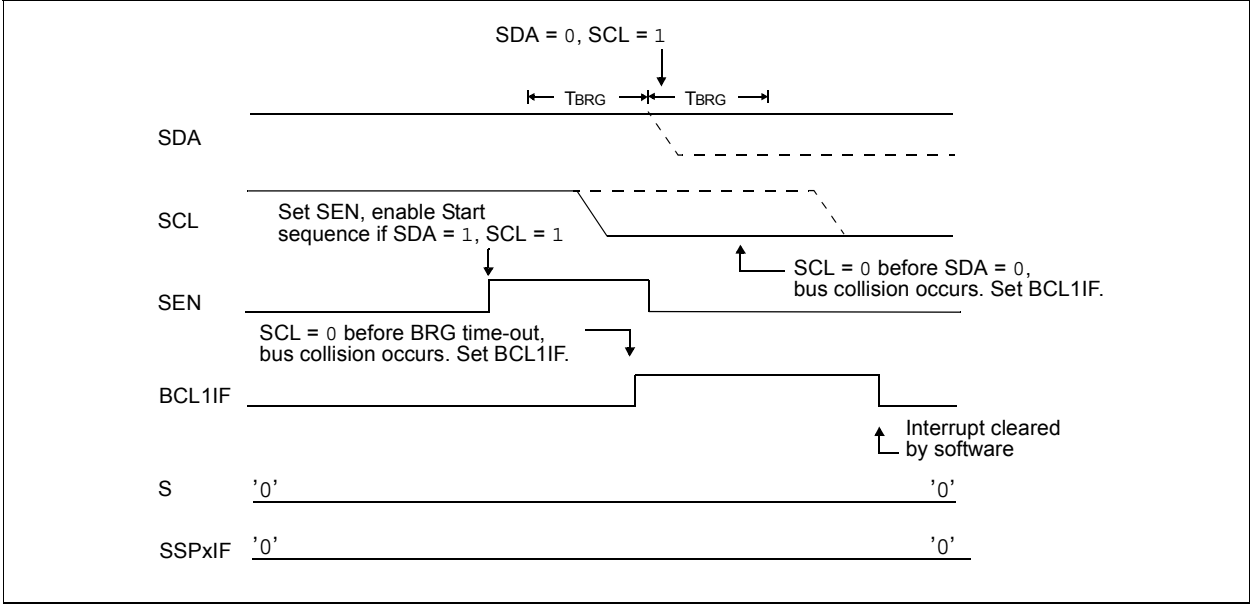
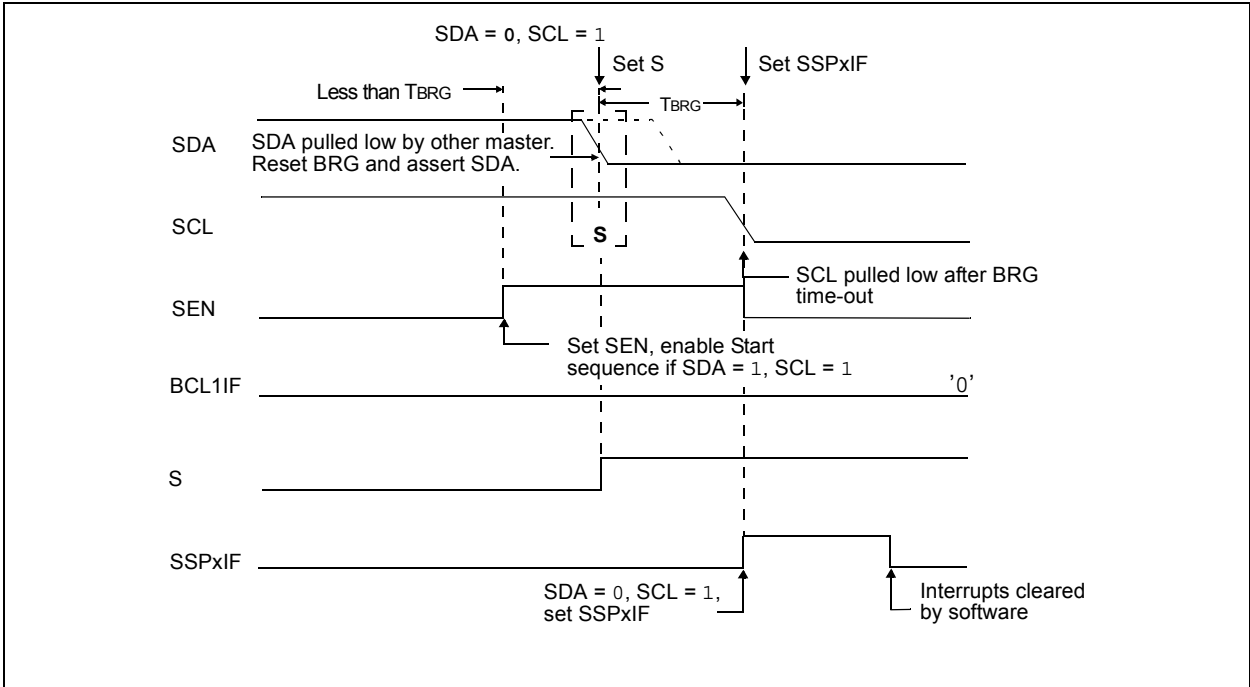


FIGURE 24-35: BRG RESET DUE TO SDA ARBITRATION DURING START CONDITION



24.6.13.2 Bus Collision During a Repeated Start Condition

During a Repeated Start condition, a bus collision occurs if:

- A low level is sampled on SDA when SCL goes from low level to high level (Case 1).
- SCL goes low before SDA is asserted low, indicating that another master is attempting to transmit a data '1' (Case 2).

When the user releases SDA and the pin is allowed to float high, the BRG is loaded with SSPxADD and counts down to zero. The SCL pin is then deasserted and when sampled high, the SDA pin is sampled.

If SDA is low, a bus collision has occurred (i.e., another master is attempting to transmit a data '0', [Figure 24-36](#)). If SDA is sampled high, the BRG is reloaded and begins counting. If SDA goes from high-to-low before the BRG times out, no bus collision occurs because no two masters can assert SDA at exactly the same time.

If SCL goes from high-to-low before the BRG times out and SDA 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 24-37](#).

If, at the end of the BRG time-out, both SCL and SDA are still high, the SDA pin is driven low and the BRG is reloaded and begins counting. At the end of the count, regardless of the status of the SCL pin, the SCL pin is driven low and the Repeated Start condition is complete.

FIGURE 24-36: BUS COLLISION DURING A REPEATED START CONDITION (CASE 1)

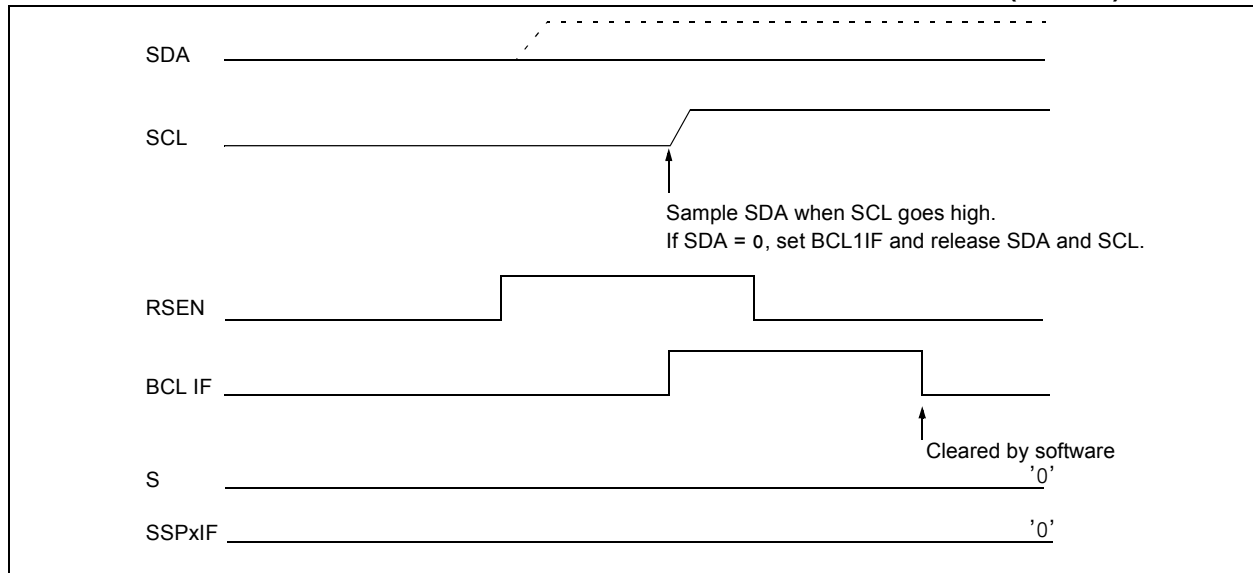
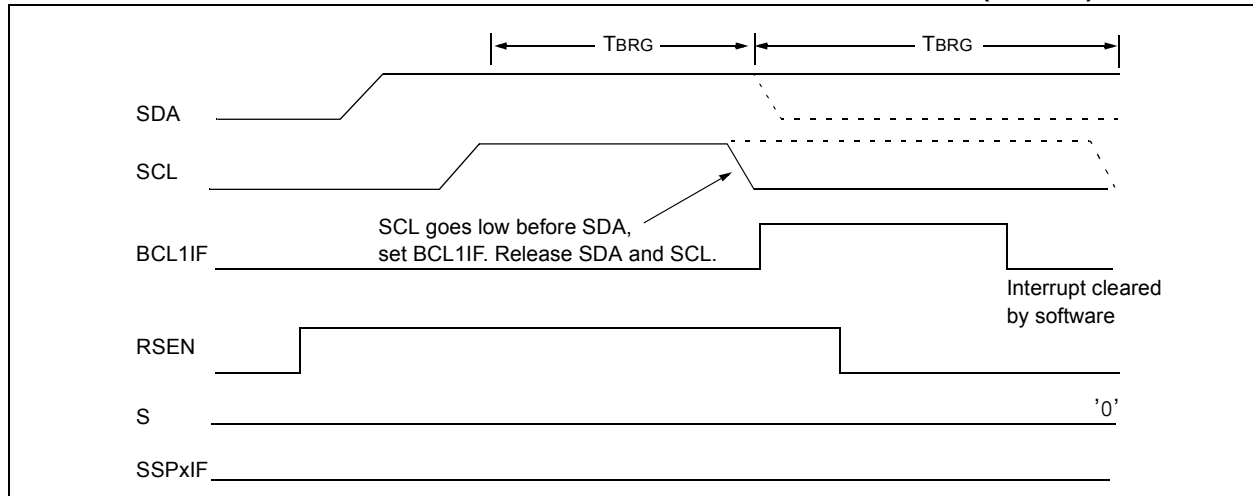


FIGURE 24-37: BUS COLLISION DURING REPEATED START CONDITION (CASE 2)



24.6.13.3 Bus Collision During a Stop Condition

Bus collision occurs during a Stop condition if:

- After the SDA pin has been deasserted and allowed to float high, SDA is sampled low after the BRG has timed out (Case 1).
- After the SCL pin is deasserted, SCL is sampled low before SDA goes high (Case 2).

The Stop condition begins with SDA asserted low. When SDA is sampled low, the SCL 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 zero. After the BRG times out, SDA is sampled. If SDA is sampled low, a bus collision has occurred. This is due to another master attempting to drive a data '0' (Figure 24-38). If the SCL pin is sampled low before SDA is allowed to float high, a bus collision occurs. This is another case of another master attempting to drive a data '0' (Figure 24-39).

FIGURE 24-38: BUS COLLISION DURING A STOP CONDITION (CASE 1)

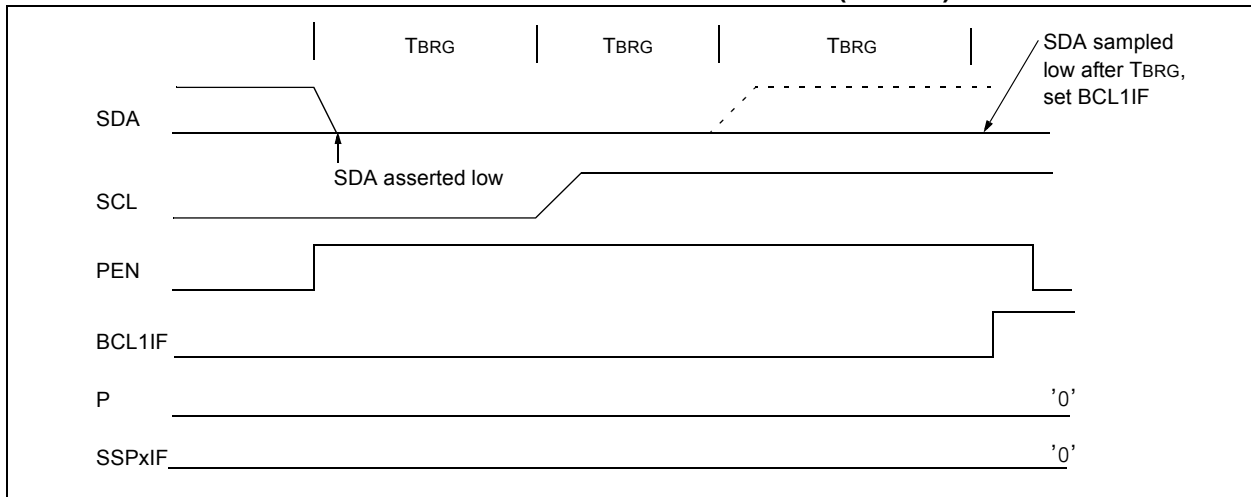


FIGURE 24-39: BUS COLLISION DURING A STOP CONDITION (CASE 2)

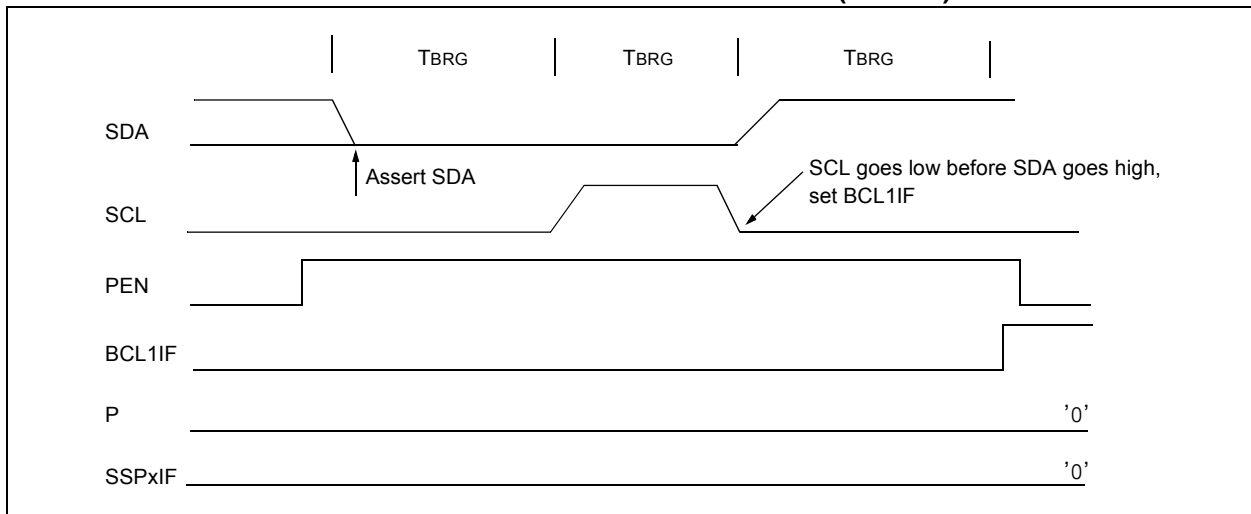


TABLE 24-3: SUMMARY OF REGISTERS ASSOCIATED WITH I²C OPERATION

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Reset Values on Page: |
|-----------------------|--|-----------------------|--------|----------------|------------------|--------|--------|--------|-----------------------|
| ANSELA | — | — | — | ANSA4 | — | ANSA2 | ANSA1 | ANSA0 | 152 |
| ANSELB ⁽¹⁾ | — | — | ANSB5 | ANSB4 | — | — | — | — | 159 |
| ANSELC | ANSC7 ⁽¹⁾ | ANSC6 ⁽¹⁾ | — | — | ANSC3 | ANSC2 | ANSC1 | ANSC0 | 166 |
| INTCON | GIE | PEIE | TMR0IE | INTE | IOIE | TMR0IF | INTF | IOCF | 97 |
| PIE1 | TMR1GIE | ADIE | RCIE | TXIE | SSP1IE | CCP1IE | TMR2IE | TMR1IE | 98 |
| PIE2 | OSFIE | C2IE | C1IE | — | BCL1IE | TMR6IE | TMR4IE | CCP2IE | 99 |
| PIR1 | TMR1GIF | ADIF | RCIF | TXIF | SSP1IF | CCP1IF | TMR2IF | TMR1IF | 103 |
| PIR2 | OSFIF | C2IF | C1IF | — | BCL1IF | TMR6IF | TMR4IF | CCP2IF | 104 |
| RxyPPS | — | — | — | RxyPPS<4:0> | | | | | 172 |
| SSPCLKPPS | — | — | — | SSPCLKPPS<4:0> | | | | | 174, 172 |
| SSPDATPPS | — | — | — | SSPDATPPS<4:0> | | | | | 174, 172 |
| SSPSSPPS | — | — | — | SSPSSPPS<4:0> | | | | | 174, 172 |
| SSP1ADD | ADD<7:0> | | | | | | | | 312 |
| SSP1BUF | Synchronous Serial Port Receive Buffer/Transmit Register | | | | | | | | 264* |
| SSP1CON1 | WCOL | SSPOV | SSPEN | CKP | SSPM<3:0> | | | | 309 |
| SSP1CON2 | GCEN | ACKSTAT | ACKDT | ACKEN | RCEN | PEN | RSEN | SEN | 310 |
| SSP1CON3 | ACKTIM | PCIE | SCIE | BOEN | SDAHT | SBCDE | AHEN | DHEN | 311 |
| SSP1MSK | MSK<7:0> | | | | | | | | 312 |
| SSP1STAT | SMP | CKE | D/Ā | P | S | R/W | UA | BF | 308 |
| TRISA | — | — | TRISA5 | TRISA4 | — ⁽²⁾ | TRISA2 | TRISA1 | TRISA0 | 151 |
| TRISB ⁽¹⁾ | TRISB7 | TRISB6 | TRISB5 | TRISB4 | — | — | — | — | 158 |
| TRISC | TRISC7 ⁽¹⁾ | TRISC6 ⁽¹⁾ | TRISC5 | TRISC4 | TRISC3 | TRISC2 | TRISC1 | TRISC0 | 165 |

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)F1619 only.

Note 2: Unimplemented, read as '1'.

24.7 BAUD RATE GENERATOR

The MSSP 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 24-6). 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 24-40 triggers the value from SSPxADD to be loaded into the BRG counter. This occurs twice for each oscillation of the

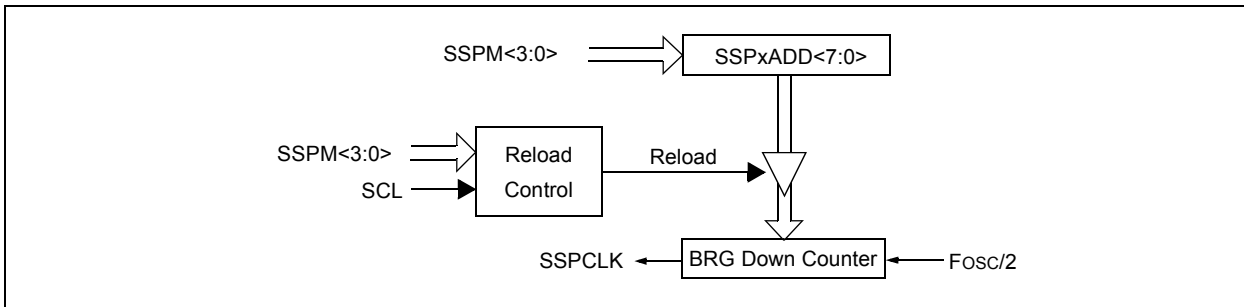
module clock line. The logic dictating when the reload signal is asserted depends on the mode the MSSP is being operated in.

Table 24-4 demonstrates clock rates based on instruction cycles and the BRG value loaded into SSPxADD.

EQUATION 24-1:

$$F_{CLOCK} = \frac{F_{OSC}}{(SSPxADD + 1)(4)}$$

FIGURE 24-40: 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 24-4: MSSP CLOCK RATE W/BRG

| Fosc | Fcy | BRG Value | F _{CLOCK} (2 Rollovers of BRG) |
|--------|-------|-----------|--|
| 32 MHz | 8 MHz | 13h | 400 kHz |
| 32 MHz | 8 MHz | 19h | 308 kHz |
| 32 MHz | 8 MHz | 4Fh | 100 kHz |
| 16 MHz | 4 MHz | 09h | 400 kHz |
| 16 MHz | 4 MHz | 0Ch | 308 kHz |
| 16 MHz | 4 MHz | 27h | 100 kHz |
| 4 MHz | 1 MHz | 09h | 100 kHz |

Note: Refer to the I/O port electrical specifications in Figure 35-7 and Table 35-10 to ensure the system is designed to support the I/O timing requirements.

24.8 Register Definitions: MSSP Control

REGISTER 24-1: SSP1STAT: SSP 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/A | P | S | R/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 | <p>SMP: SPI Data Input Sample bit</p> <p><u>SPI Master mode:</u> 1 = Input data sampled at end of data output time 0 = Input data sampled at middle of data output time</p> <p><u>SPI Slave mode:</u> SMP must be cleared when SPI is used in Slave mode</p> <p><u>In I²C Master or Slave mode:</u> 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)</p> |
| bit 6 | <p>CKE: SPI Clock Edge Select bit (SPI mode only)</p> <p><u>In SPI Master or Slave mode:</u> 1 = Transmit occurs on transition from active to Idle clock state 0 = Transmit occurs on transition from Idle to active clock state</p> <p><u>In I²C™ mode only:</u> 1 = Enable input logic so that thresholds are compliant with SMBus specification 0 = Disable SMBus specific inputs</p> |
| bit 5 | <p>D/A: Data/Address bit (I²C mode only)</p> <p>1 = Indicates that the last byte received or transmitted was data 0 = Indicates that the last byte received or transmitted was address</p> |
| bit 4 | <p>P: Stop bit</p> <p>(I²C mode only. This bit is cleared when the MSSP 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</p> |
| bit 3 | <p>S: Start bit</p> <p>(I²C mode only. This bit is cleared when the MSSP 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</p> |
| bit 2 | <p>R/W: Read/Write bit information (I²C mode only)</p> <p>This bit holds the R/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.</p> <p><u>In I²C Slave mode:</u> 1 = Read 0 = Write</p> <p><u>In I²C Master mode:</u> 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 MSSP is in Idle mode.</p> |
| bit 1 | <p>UA: Update Address bit (10-bit I²C mode only)</p> <p>1 = Indicates that the user needs to update the address in the SSP1ADD register 0 = Address does not need to be updated</p> |
| bit 0 | <p>BF: Buffer Full Status bit</p> <p><u>Receive (SPI and I²C modes):</u> 1 = Receive complete, SSP1BUF is full 0 = Receive not complete, SSP1BUF is empty</p> <p><u>Transmit (I²C mode only):</u> 1 = Data transmit in progress (does not include the ACK and Stop bits), SSP1BUF is full 0 = Data transmit complete (does not include the ACK and Stop bits), SSP1BUF is empty</p> |

REGISTER 24-2: SSP1CON1: SSP 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 | SSPOV ⁽¹⁾ | 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> 1 = A write to the SSP1BUF register was attempted while the I²C conditions were not valid for a transmission to be started 0 = No collision</p> <p><u>Slave mode:</u> 1 = The SSP1BUF register is written while it is still transmitting the previous word (must be cleared in software) 0 = No collision</p> |
| bit 6 | <p>SSPOV: Receive Overflow Indicator bit⁽¹⁾</p> <p><u>In SPI mode:</u> 1 = A new byte is received while the SSP1BUF register is still holding the previous data. In case of overflow, the data in SSPSR is lost. Overflow can only occur in Slave mode. In Slave mode, the user must read the SSP1BUF, 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 SSP1BUF register (must be cleared in software). 0 = No overflow</p> <p><u>In I²C mode:</u> 1 = A byte is received while the SSP1BUF register is still holding the previous byte. SSPOV is a "don't care" in Transmit mode (must be cleared in software). 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> 1 = Enables serial port and configures SCK, SDO, SDI and \overline{SS} as the source of the serial port pins⁽²⁾ 0 = Disables serial port and configures these pins as I/O port pins</p> <p><u>In I²C mode:</u> 1 = Enables the serial port and configures the SDA and SCL pins as the source of the serial port pins⁽³⁾ 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> 1 = Idle state for clock is a high level 0 = Idle state for clock is a low level</p> <p><u>In I²C Slave mode:</u> SCL release control 1 = Enable clock 0 = Holds clock low (clock stretch). (Used to ensure data setup time.)</p> <p><u>In I²C Master mode:</u> 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 1110 = I²C Slave mode, 7-bit address with Start and Stop bit interrupts enabled 1101 = Reserved 1100 = Reserved 1011 = I²C firmware controlled Master mode (slave idle) 1010 = SPI Master mode, clock = $F_{osc}/(4 * (SSP1ADD+1))$⁽⁵⁾ 1001 = Reserved 1000 = I²C Master mode, clock = $F_{osc} / (4 * (SSP1ADD+1))$⁽⁴⁾ 0111 = I²C Slave mode, 10-bit address 0110 = I²C Slave mode, 7-bit address 0101 = SPI Slave mode, clock = SCK pin, \overline{SS} pin control disabled, \overline{SS} can be used as I/O pin 0100 = SPI Slave mode, clock = SCK pin, \overline{SS} pin control enabled 0011 = SPI Master mode, clock = $T2_match/2$ 0010 = SPI Master mode, clock = $F_{osc}/64$ 0001 = SPI Master mode, clock = $F_{osc}/16$ 0000 = SPI Master mode, clock = $F_{osc}/4$</p> |

- Note**
- 1: In Master mode, the overflow bit is not set since each new reception (and transmission) is initiated by writing to the SSP1BUF register.
 - 2: When enabled, these pins must be properly configured as input or output. Use SSPSSPPS, SSPCLKPPS, SSPDATPPS, and RxyPPS to select the pins.
 - 3: When enabled, the SDA and SCL pins must be configured as inputs. Use SSPCLKPPS, SSPDATPPS, and RxyPPS to select the pins.
 - 4: SSP1ADD values of 0, 1 or 2 are not supported for I²C mode.
 - 5: SSP1ADD value of '0' is not supported. Use SSPM = 0000 instead.

REGISTER 24-3: SSP1CON2: SSP CONTROL REGISTER 2⁽¹⁾

| 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 SSPSR
 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 SDA and SCL 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)
SCKMSSP Release Control:
 1 = Initiate Stop condition on SDA and SCL pins. Automatically cleared by hardware.
 0 = Stop condition Idle
- bit 1 **RSEN:** Repeated Start Condition Enable bit (in I²C Master mode only)
 1 = Initiate Repeated Start condition on SDA and SCL 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 SDA and SCL 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 SSP1BUF may not be written (or writes to the SSP1BUF are disabled).

REGISTER 24-4: SSP1CON3: SSP 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 ⁽³⁾ | 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 eighth falling edge of SCL clock
 0 = Not an Acknowledge sequence, cleared on 9th rising edge of SCL clock
- bit 6 **PCIE:** Stop Condition Interrupt Enable bit (I²C Slave 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 Slave 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 = SSP1BUF 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 SSP1STAT register already set, SSPOV bit of the SSP1CON1 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 = SSP1BUF 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 = SSP1BUF is only updated when SSPOV is clear
- bit 3 **SDAHT:** SDA Hold Time Selection bit (I²C mode only)
 1 = Minimum of 300 ns hold time on SDA after the falling edge of SCL
 0 = Minimum of 100 ns hold time on SDA after the falling edge of SCL
- bit 2 **SBCDE:** Slave Mode Bus Collision Detect Enable bit (I²C Slave mode only)
 If, on the rising edge of SCL, SDA is sampled low when the module is outputting a high state, the BCL1IF 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 eighth falling edge of SCL for a matching received address byte; CKP bit of the SSP1CON1 register will be cleared and the SCL will be held low.
 0 = Address holding is disabled
- bit 0 **DHEN:** Data Hold Enable bit (I²C Slave mode only)
 1 = Following the eighth falling edge of SCL for a received data byte; slave hardware clears the CKP bit of the SSP1CON1 register and SCL 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 SSP1BUF.
- 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 24-5: SSP1MSK: SSP 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 SSP1ADD<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 SSP1ADD<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 24-6: SSP1ADD: MSSP 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
 SCL 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”.

25.0 ENHANCED UNIVERSAL SYNCHRONOUS ASYNCHRONOUS RECEIVER TRANSMITTER (EUSART)

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 polarity in synchronous modes
- Sleep operation

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 25-1](#) and [Figure 25-2](#).

The EUSART transmit output (TX_out) is available to the TX/CK pin and internally to the following peripherals:

- Configurable Logic Cell (CLC)

FIGURE 25-1: EUSART TRANSMIT BLOCK DIAGRAM

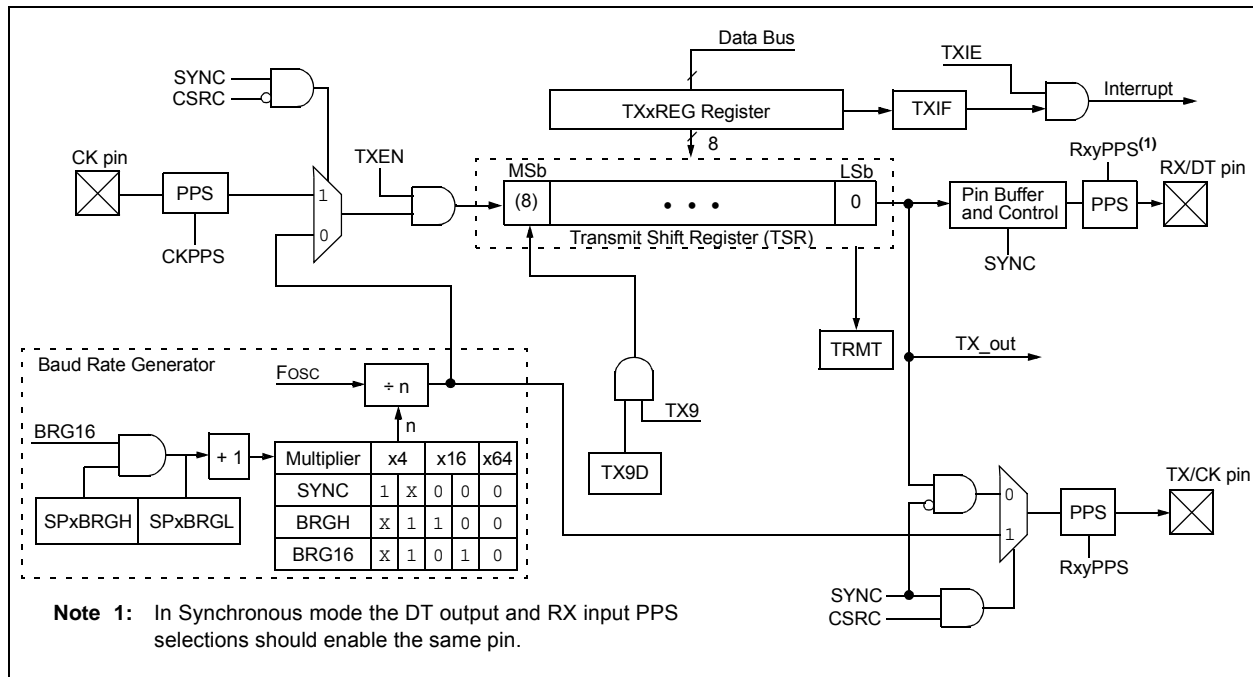
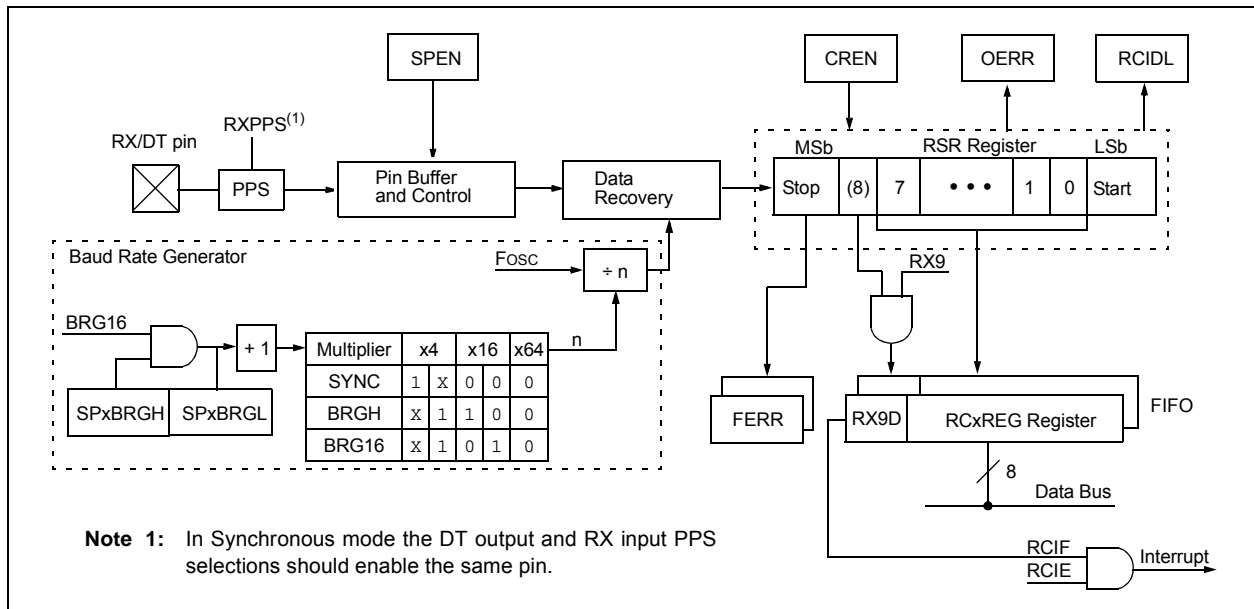


FIGURE 25-2: EUSART RECEIVE BLOCK DIAGRAM



The operation of the EUSART module is controlled through three registers:

- Transmit Status and Control (TXxSTA)
- Receive Status and Control (RCxSTA)
- Baud Rate Control (BAUDxCON)

These registers are detailed in [Register 25-1](#), [Register 25-2](#) and [Register 25-3](#), respectively.

The RX and CK input pins are selected with the RXPPS and CKPPS registers, respectively. TX, CK, and DT output pins are selected with each pin's RxyPPS register. Since the RX input is coupled with the DT output in Synchronous mode, it is the user's responsibility to select the same pin for both of these functions when operating in Synchronous mode. The EUSART control logic will control the data direction drivers automatically.

25.1 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 eight 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 25-5](#) 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.

25.1.1 EUSART ASYNCHRONOUS TRANSMITTER

The EUSART transmitter block diagram is shown in [Figure 25-1](#). 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.

25.1.1.1 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 and automatically configures the TX/CK I/O pin as an output. If the TX/CK pin is shared with an analog peripheral, the analog I/O function must be disabled by clearing the corresponding ANSEL bit.

Note: The TXIF Transmitter Interrupt flag is set when the TXEN enable bit is set.

25.1.1.2 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.

25.1.1.3 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 in Asynchronous mode only. In Synchronous mode, the SCKP bit has a different function. See [Section 25.5.1.2 "Clock Polarity"](#).

25.1.1.4 Transmit Interrupt Flag

The TXIF interrupt flag bit of the PIR1 register is set whenever the EUSART transmitter is enabled and no character is being held for transmission in the TXxREG. In other words, the TXIF 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 TXIF flag bit is not cleared immediately upon writing TXxREG. TXIF becomes valid in the second instruction cycle following the write execution. Polling TXIF immediately following the TXxREG write will return invalid results. The TXIF bit is read-only, it cannot be set or cleared by software.

The TXIF interrupt can be enabled by setting the TXIE interrupt enable bit of the PIE1 register. However, the TXIF flag bit will be set whenever the TXxREG is empty, regardless of the state of TXIE enable bit.

To use interrupts when transmitting data, set the TXIE bit only when there is more data to send. Clear the TXIE interrupt enable bit upon writing the last character of the transmission to the TXxREG.

25.1.1.5 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 has 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.

25.1.1.6 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 nine 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 eight 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 25.1.2.7 “Address Detection”](#) for more information on the Address mode.

25.1.1.7 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 25.4 “EUSART Baud Rate Generator \(BRG\)”](#)).
2. Enable the asynchronous serial port by clearing the SYNC bit and setting the SPEN bit.
3. If 9-bit transmission is desired, set the TX9 control bit. A set ninth data bit will indicate that the eight Least Significant data bits are an address when the receiver is set for address detection.
4. Set SCKP bit if inverted transmit is desired.
5. Enable the transmission by setting the TXEN control bit. This will cause the TXIF interrupt bit to be set.
6. If interrupts are desired, set the TXIE interrupt enable bit of the PIE1 register. An interrupt will occur immediately provided that the GIE and PEIE bits of the INTCON register are also set.
7. If 9-bit transmission is selected, the ninth bit should be loaded into the TX9D data bit.
8. Load 8-bit data into the TXxREG register. This will start the transmission.

FIGURE 25-3: ASYNCHRONOUS TRANSMISSION

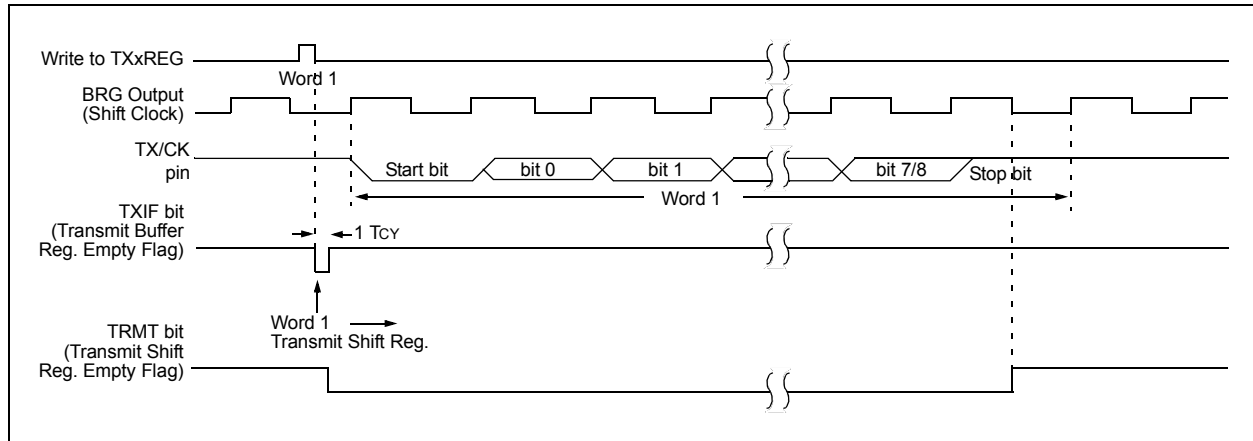


FIGURE 25-4: ASYNCHRONOUS TRANSMISSION (BACK-TO-BACK)

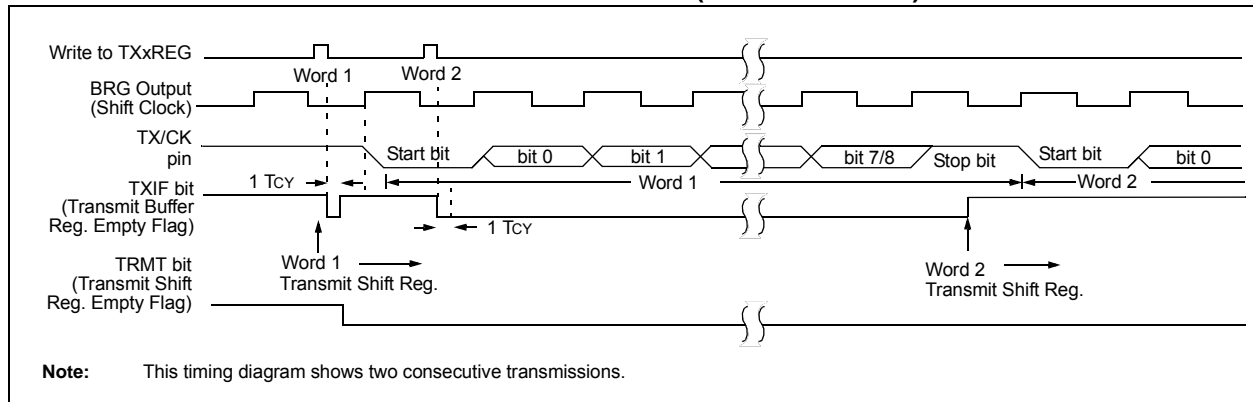


TABLE 25-1: SUMMARY OF REGISTERS ASSOCIATED WITH ASYNCHRONOUS TRANSMISSION

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Register on Page |
|-----------------------|-------------------------------|-----------------------|--------|-------------|------------------|--------|--------|--------|------------------|
| ANSELA | — | — | — | ANSA4 | — | ANSA2 | ANSA1 | ANSA0 | 152 |
| ANSELB ⁽¹⁾ | — | — | ANSB5 | ANSB4 | — | — | — | — | 159 |
| ANSELC | ANSC7 ⁽¹⁾ | ANSC6 ⁽¹⁾ | — | — | ANSC3 | ANSC2 | ANSC1 | ANSC0 | 166 |
| BAUD1CON | ABDOVF | RCIDL | — | SCKP | BRG16 | — | WUE | ABDEN | 325 |
| INTCON | GIE | PEIE | TMR0IE | INTE | IOCIE | TMR0IF | INTF | IOCIF | 97 |
| PIE1 | TMR1GIE | ADIE | RCIE | TXIE | SSP1IE | CCP1IE | TMR2IE | TMR1IE | 98 |
| PIR1 | TMR1GIF | ADIF | RCIF | TXIF | SSP1IF | CCP1IF | TMR2IF | TMR1IF | 103 |
| RC1STA | SPEN | RX9 | SREN | CREN | ADDEN | FERR | OERR | RX9D | 324 |
| RxyPPS | — | — | — | RxyPPS<4:0> | | | | | 172 |
| SP1BRGL | BRG<7:0> | | | | | | | | 326* |
| SP1BRGH | BRG<15:8> | | | | | | | | 326* |
| TRISA | — | — | TRISA5 | TRISA4 | — ⁽²⁾ | TRISA2 | TRISA1 | TRISA0 | 149 |
| TRISB ⁽¹⁾ | TRISB7 | TRISB6 | TRISB5 | TRISB4 | — | — | — | — | 158 |
| TRISC | TRISC7 ⁽¹⁾ | TRISC6 ⁽¹⁾ | TRISC5 | TRISC4 | TRISC3 | TRISC2 | TRISC1 | TRISC0 | 165 |
| TX1REG | EUSART Transmit Data Register | | | | | | | | 315* |
| TX1STA | CSRC | TX9 | TXEN | SYNC | SENDB | BRGH | TRMT | TX9D | 323 |

Legend: — = unimplemented location, read as '0'. Shaded cells are not used for asynchronous transmission.

* Page provides register information.

Note 1: PIC16(L)F1619 only.

Note 2: Unimplemented, read as '1'.

25.1.2 EUSART ASYNCHRONOUS RECEIVER

The Asynchronous mode is typically used in RS-232 systems. The receiver block diagram is shown in [Figure 25-2](#). The data is received on the RX/DT 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 eight or nine 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.

25.1.2.1 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 RX/DT I/O pin as an input.

Note: If the RX/DT function is on an analog pin, the corresponding ANSEL bit must be cleared for the receiver to function.

25.1.2.2 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 25.1.2.4 "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 RCIF interrupt flag bit of the PIR1 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 25.1.2.5 "Receive Overrun Error"](#) for more information on overrun errors.

25.1.2.3 Receive Interrupts

The RCIF interrupt flag bit of the PIR1 register is set whenever the EUSART receiver is enabled and there is an unread character in the receive FIFO. The RCIF interrupt flag bit is read-only, it cannot be set or cleared by software.

RCIF interrupts are enabled by setting all of the following bits:

- RCIE, Interrupt Enable bit of the PIE1 register
- PEIE, Peripheral Interrupt Enable bit of the INTCON register
- GIE, Global Interrupt Enable bit of the INTCON register

The RCIF interrupt flag bit will be set when there is an unread character in the FIFO, regardless of the state of interrupt enable bits.

25.1.2.4 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.

Note: If all receive characters in the receive FIFO have framing errors, repeated reads of the RCxREG will not clear the FERR bit.

25.1.2.5 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.

25.1.2.6 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 nine 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 eight Least Significant bits from the RCxREG.

25.1.2.7 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 RCIF 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.

25.1.2.8 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 25.4 “EUSART Baud Rate Generator \(BRG\)”](#)).
2. Clear the ANSEL bit for the RX pin (if applicable).
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 RCIE bit of the PIE1 register and 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 RCIF 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 RCIE 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 eight 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.

25.1.2.9 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 25.4 “EUSART Baud Rate Generator \(BRG\)”](#)).
2. Clear the ANSEL bit for the RX pin (if applicable).
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 RCIE bit of the PIE1 register and 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 RCIF 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 RCIE 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 eight 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 25-5: ASYNCHRONOUS RECEPTION

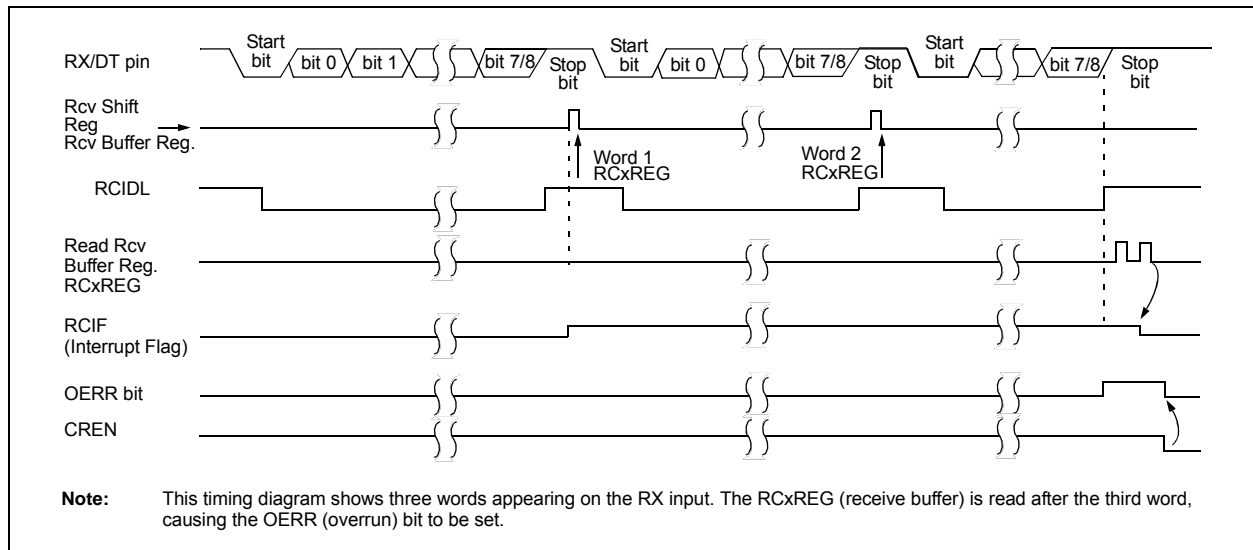


TABLE 25-2: SUMMARY OF REGISTERS ASSOCIATED WITH ASYNCHRONOUS RECEPTION

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Register on Page |
|-----------------------|------------------------------|-----------------------|--------|-------------|------------------|--------|--------|--------|------------------|
| ANSELA | — | — | — | ANSA4 | — | ANSA2 | ANSA1 | ANSA0 | 152 |
| ANSELB ⁽¹⁾ | — | — | ANSB5 | ANSB4 | — | — | — | — | 159 |
| ANSELC | ANSC7 ⁽¹⁾ | ANSC6 ⁽¹⁾ | — | — | ANSC3 | ANSC2 | ANSC1 | ANSC0 | 166 |
| BAUD1CON | ABDOVF | RCIDL | — | SCKP | BRG16 | — | WUE | ABDEN | 325 |
| INTCON | GIE | PEIE | TMR0IE | INTE | IOCIE | TMR0IF | INTF | IOCIF | 97 |
| PIE1 | TMR1GIE | ADIE | RCIE | TXIE | SSP1IE | CCP1IE | TMR2IE | TMR1IE | 97 |
| PIR1 | TMR1GIF | ADIF | RCIF | TXIF | SSP1IF | CCP1IF | TMR2IF | TMR1IF | 103 |
| RC1REG | EUSART Receive Data Register | | | | | | | | 318* |
| RC1STA | SPEN | RX9 | SREN | CREN | ADDEN | FERR | OERR | RX9D | 324 |
| RxyPPS | — | — | — | RxyPPS<4:0> | | | | | 172 |
| SP1BRGL | BRG<7:0> | | | | | | | | 326 |
| SP1BRGH | BRG<15:8> | | | | | | | | 326 |
| TRISA | — | — | TRISA5 | TRISA4 | — ⁽²⁾ | TRISA2 | TRISA1 | TRISA0 | 151 |
| TRISB ⁽¹⁾ | TRISB7 | TRISB6 | TRISB5 | TRISB4 | — | — | — | — | 158 |
| TRISC | TRISC7 ⁽¹⁾ | TRISC6 ⁽¹⁾ | TRISC5 | TRISC4 | TRISC3 | TRISC2 | TRISC1 | TRISC0 | 165 |
| TX1STA | CSRC | TX9 | TXEN | SYNC | SENCB | BRGH | TRMT | TX9D | 323 |

Legend: — = unimplemented location, read as '0'. Shaded cells are not used for asynchronous reception.

* Page provides register information.

Note 1: PIC16(L)F1619 only.

Note 2: Unimplemented, read as '1'.

25.2 Clock Accuracy with Asynchronous Operation

The factory calibrates the internal oscillator block output (INTOSC). However, the INTOSC frequency may drift as VDD or temperature changes, and this directly affects the asynchronous baud rate. Two methods may be used to adjust the baud rate clock, but both require a reference clock source of some kind.

The first (preferred) method uses the OSCTUNE register to adjust the INTOSC output. Adjusting the value in the OSCTUNE register allows for fine resolution changes to the system clock source. See [Section 5.2.2.3 “Internal Oscillator Frequency Adjustment”](#) 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 25.4.1 “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.

25.3 Register Definitions: EUSART Control

REGISTER 25-1: TX1STA: TRANSMIT STATUS AND CONTROL REGISTER

| R/W-/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R-1/1 | R/W-0/0 |
|--------|---------|---------------------|---------|---------|---------|-------|---------|
| 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'

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 **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 25-2: RC1STA: RECEIVE STATUS AND CONTROL REGISTER

| | | | | | | | |
|---------|---------|---------|---------|---------|-------|-------|-------|
| R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R-0/0 | R-0/0 | R-0/0 |
| 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' |
| 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 **SPEN:** Serial Port Enable bit
1 = Serial port enabled
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.

REGISTER 25-3: BAUD1CON: 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 TX/CK pin
- 0 = Transmit non-inverted data to the TX/CK 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

25.4 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.

Table 25-3 contains the formulas for determining the baud rate. Example 25-1 provides a sample calculation for determining the 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 25-5. 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 25-1: CALCULATING BAUD RATE ERROR

For a device with Fosc of 16 MHz, desired baud rate of 9600, Asynchronous mode, 8-bit BRG:

$$\text{Desired Baud Rate} = \frac{F_{OSC}}{64([SPBRGH:SPBRGL] + 1)}$$

Solving for SPxBRGH:SPxBRGL:

$$X = \frac{F_{OSC}}{\text{Desired Baud Rate} \cdot 64} - 1$$

$$= \frac{16000000}{9600 \cdot 64} - 1$$

$$= [25.042] = 25$$

$$\text{Calculated Baud Rate} = \frac{16000000}{64(25 + 1)}$$

$$= 9615$$

$$\text{Error} = \frac{\text{Calc. Baud Rate} - \text{Desired Baud Rate}}{\text{Desired Baud Rate}}$$

$$= \frac{(9615 - 9600)}{9600} = 0.16\%$$

TABLE 25-3: BAUD RATE FORMULAS

| Configuration Bits | | | BRG/EUSART Mode | Baud Rate Formula |
|--------------------|-------|------|---------------------|----------------------|
| SYNC | BRG16 | BRGH | | |
| 0 | 0 | 0 | 8-bit/Asynchronous | $F_{osc}/[64 (n+1)]$ |
| 0 | 0 | 1 | 8-bit/Asynchronous | $F_{osc}/[16 (n+1)]$ |
| 0 | 1 | 0 | 16-bit/Asynchronous | |
| 0 | 1 | 1 | 16-bit/Asynchronous | $F_{osc}/[4 (n+1)]$ |
| 1 | 0 | x | 8-bit/Synchronous | |
| 1 | 1 | x | 16-bit/Synchronous | |

Legend: x = Don't care, n = value of SPxBRGH, SPxBRGL register pair.

TABLE 25-4: 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 | Register on Page |
|----------|-----------|-------|-------|-------|--------|-------|-------|-------|------------------|
| BAUD1CON | ABDOVF | RCIDL | — | SCKP | BRG16 | — | WUE | ABDEN | 325 |
| RC1STA | SPEN | RX9 | SREN | CREN | ADDEN | FERR | OERR | RX9D | 324 |
| SP1BRGL | BRG<7:0> | | | | | | | | 326 |
| SP1BRGH | BRG<15:8> | | | | | | | | 326 |
| TX1STA | CSRC | TX9 | TXEN | SYNC | SENDER | BRGH | TRMT | TX9D | 323 |

Legend: — = unimplemented location, read as '0'. Shaded cells are not used for the Baud Rate Generator.

* Page provides register information.

TABLE 25-5: 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 |

TABLE 25-5: 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 25-5: 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 | — | — | — |

25.4.1 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 RX signal, the RX 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. 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 SPxBRG begins counting up using the BRG counter clock as shown in Figure 25-6. The fifth rising edge will occur on the RX 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 RCIF interrupt flag is set. The value in the RCxREG needs to be read to clear the RCIF 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 25-6. 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 25.4.3 "Auto-Wake-up on Break").

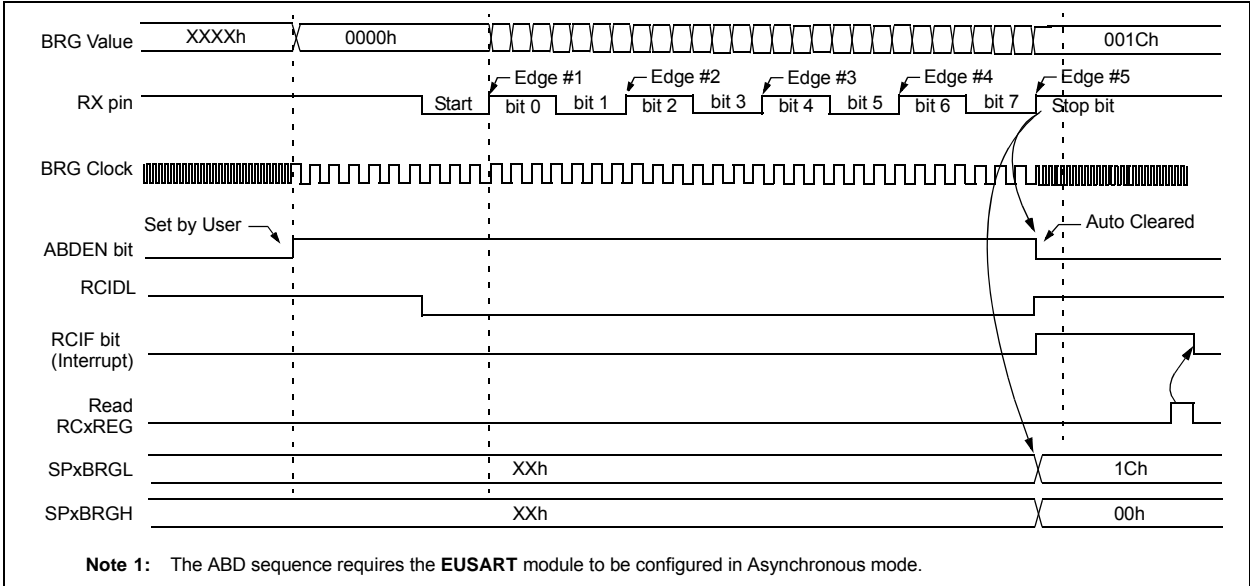
- 2: It is up to the user to determine that the incoming character baud rate is within the range of the selected BRG clock source. Some combinations of oscillator frequency and EUSART baud rates are not possible.
- 3: During the auto-baud process, the auto-baud counter starts counting at one. Upon completion of the auto-baud sequence, to achieve maximum accuracy, subtract 1 from the SPxBRGH:SPxBRGL register pair.

TABLE 25-6: 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 the BRG16 setting.

FIGURE 25-6: AUTOMATIC BAUD RATE CALIBRATION



25.4.2 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.

25.4.3 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 RX/DT 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 RX/DT 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 RX/DT 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 RCIF interrupt coincident with the wake-up event. The interrupt is generated synchronously to the Q clocks in normal CPU operating modes (Figure 25-7), and asynchronously if the device is in Sleep mode (Figure 25-8). The interrupt condition is cleared by reading the RCxREG register.

The WUE bit is automatically cleared by the low-to-high transition on the RX 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.

25.4.3.1 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 ten or more bit times, 13-bit times recommended for LIN bus, or any number of bit times for standard RS-232 devices.

Oscillator Start-up Time

Oscillator start-up time must be considered, especially in applications using oscillators with longer start-up intervals (i.e., HS/PLL 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 RCIF bit. The WUE bit is cleared in hardware by a rising edge on RX/DT. The interrupt condition is then cleared in 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.

FIGURE 25-7: AUTO-WAKE-UP BIT (WUE) TIMING DURING NORMAL OPERATION

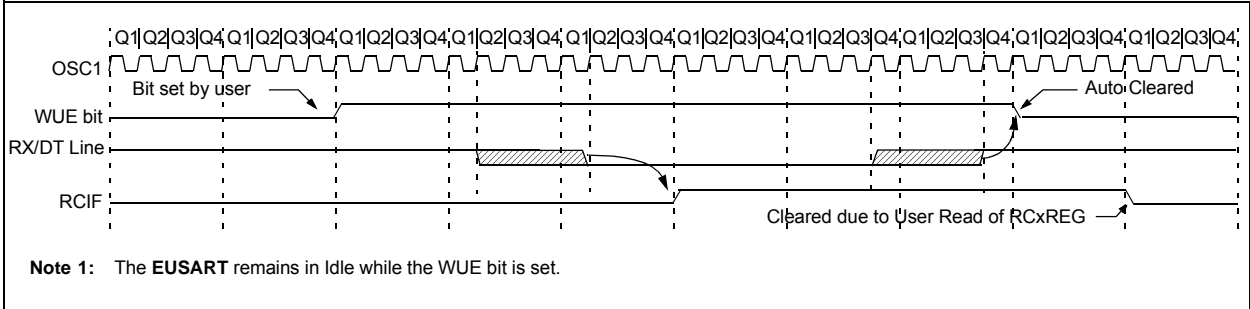
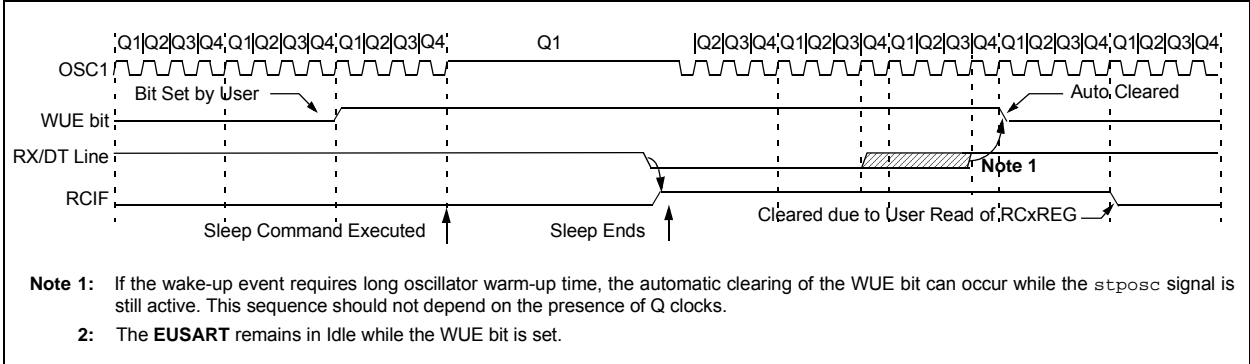


FIGURE 25-8: AUTO-WAKE-UP BIT (WUE) TIMINGS DURING SLEEP



25.4.4 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 25-9](#) for the timing of the Break character sequence.

25.4.4.1 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 TXIF, the next data byte can be written to TXxREG.

25.4.5 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.

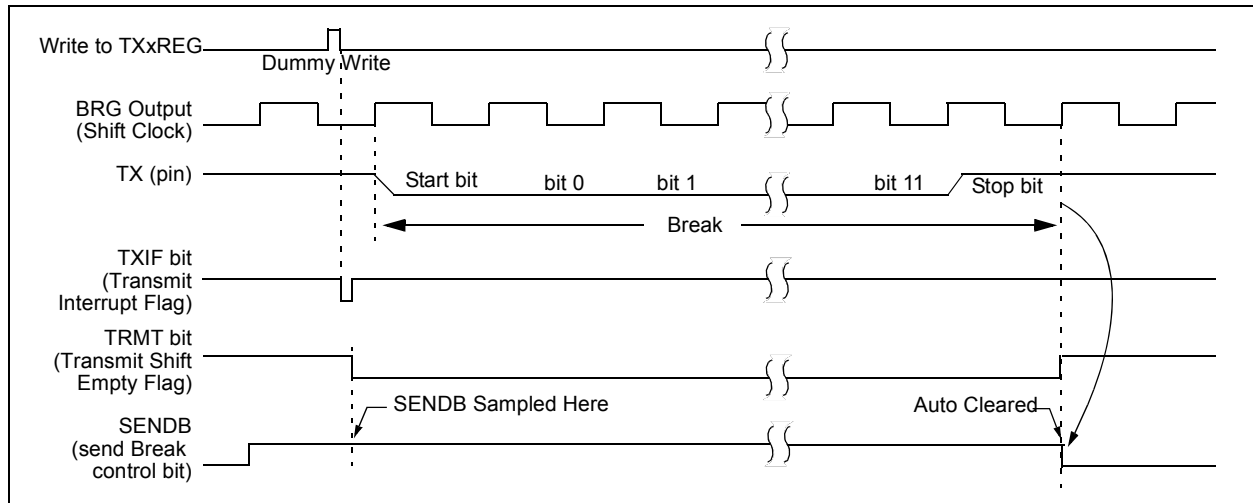
A Break character has been received when;

- RCIF bit is set
- FERR bit is set
- RCxREG = 00h

The second method uses the Auto-Wake-up feature described in [Section 25.4.3 "Auto-Wake-up on Break"](#). By enabling this feature, the EUSART will sample the next two transitions on RX/DT, cause an RCIF 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 25-9: SEND BREAK CHARACTER SEQUENCE



25.5 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.

25.5.1 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.

25.5.1.1 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 TX/CK line. The TX/CK 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.

25.5.1.2 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.

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.

25.5.1.3 Synchronous Master Transmission

Data is transferred out of the device on the RX/DT pin. The RX/DT and TX/CK 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.

25.5.1.4 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 25.4 “EUSART Baud Rate Generator \(BRG\)”](#)).
2. Enable the synchronous master serial port by setting bits SYNC, SPEN and CSRC.
3. Disable Receive mode by clearing bits SREN and CREN.
4. Enable Transmit mode by setting the TXEN bit.
5. If 9-bit transmission is desired, set the TX9 bit.
6. If interrupts are desired, set the TXIE bit of the PIE1 register and the GIE and PEIE bits of the INTCON register.
7. If 9-bit transmission is selected, the ninth bit should be loaded in the TX9D bit.
8. Start transmission by loading data to the TXxREG register.

FIGURE 25-10: SYNCHRONOUS TRANSMISSION

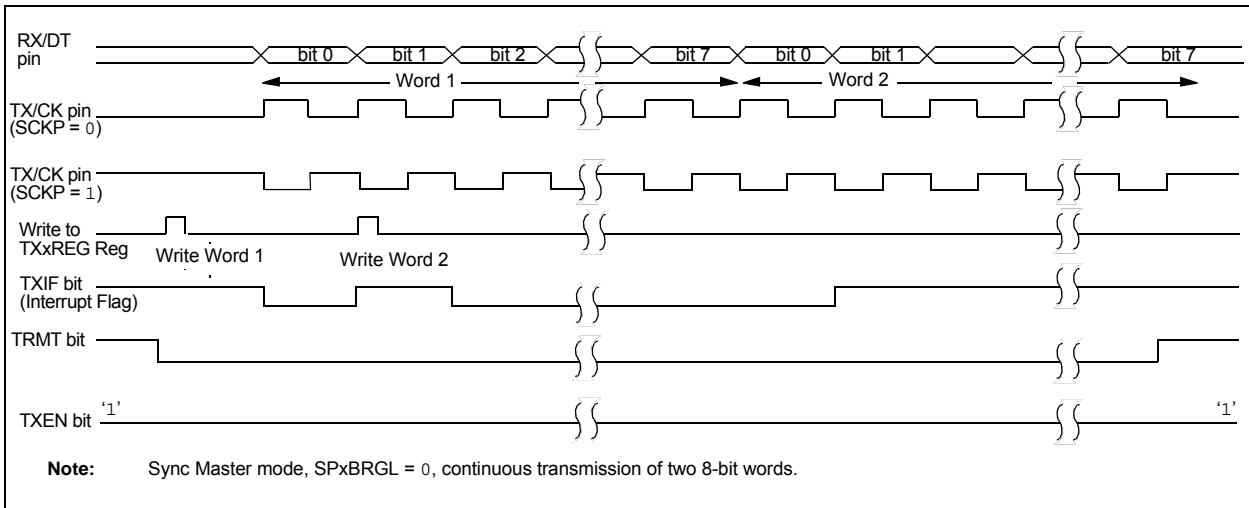


FIGURE 25-11: SYNCHRONOUS TRANSMISSION (THROUGH TXEN)

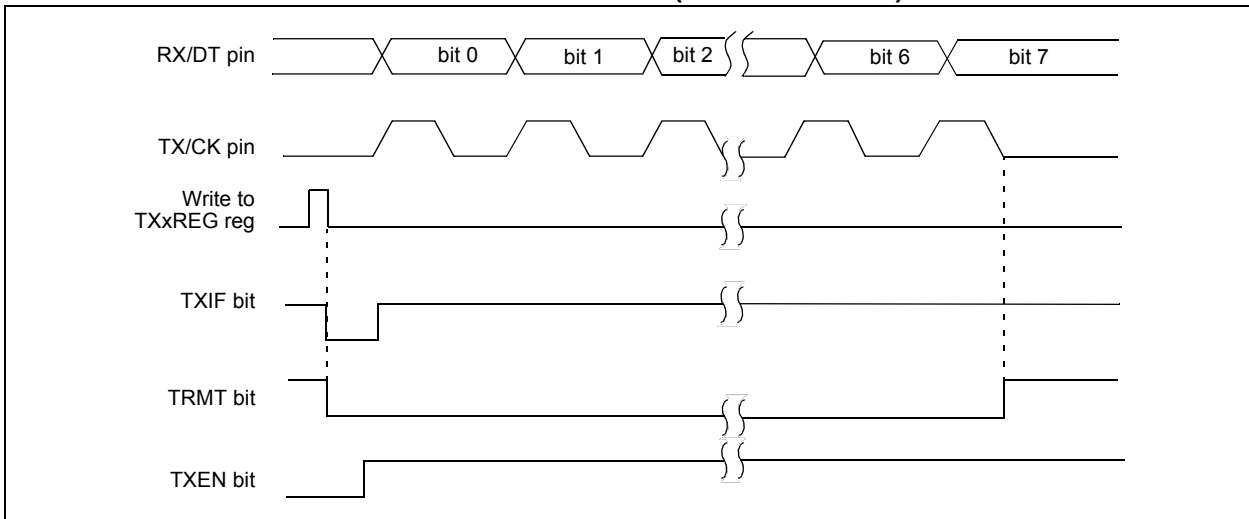


TABLE 25-7: SUMMARY OF REGISTERS ASSOCIATED WITH SYNCHRONOUS MASTER TRANSMISSION

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Register on Page |
|-----------------------|-------------------------------|-----------------------|--------|-------------|------------------|--------|--------|--------|------------------|
| ANSELA | — | — | — | ANSA4 | — | ANSA2 | ANSA1 | ANSA0 | 152 |
| ANSELB ⁽¹⁾ | — | — | ANSB5 | ANSB4 | — | — | — | — | 159 |
| ANSELC | ANSC7 ⁽¹⁾ | ANSC6 ⁽¹⁾ | — | — | ANSC3 | ANSC2 | ANSC1 | ANSC0 | 166 |
| BAUD1CON | ABDOVF | RCIDL | — | SCKP | BRG16 | — | WUE | ABDEN | 325 |
| INTCON | GIE | PEIE | TMR0IE | INTE | IOCIE | TMR0IF | INTF | IOCIF | 97 |
| PIE1 | TMR1GIE | ADIE | RCIE | TXIE | SSP1IE | CCP1IE | TMR2IE | TMR1IE | 98 |
| PIR1 | TMR1GIF | ADIF | RCIF | TXIF | SSP1IF | CCP1IF | TMR2IF | TMR1IF | 103 |
| RC1STA | SPEN | RX9 | SREN | CREN | ADDEN | FERR | OERR | RX9D | 324 |
| RxyPPS | — | — | — | RxyPPS<4:0> | | | | | 172 |
| SP1BRGL | BRG<7:0> | | | | | | | | 326 |
| SP1BRGH | BRG<15:8> | | | | | | | | 326 |
| TRISA | — | — | TRISA5 | TRISA4 | — ⁽²⁾ | TRISA2 | TRISA1 | TRISA0 | 151 |
| TRISB ⁽¹⁾ | TRISB7 | TRISB6 | TRISB5 | TRISB4 | — | — | — | — | 158 |
| TRISC | TRISC7 ⁽¹⁾ | TRISC6 ⁽¹⁾ | TRISC5 | TRISC4 | TRISC3 | TRISC2 | TRISC1 | TRISC0 | 165 |
| TX1REG | EUSART Transmit Data Register | | | | | | | | 315* |
| TX1STA | CSRC | TX9 | TXEN | SYNC | SENCB | BRGH | TRMT | TX9D | 323 |

Legend: — = unimplemented location, read as '0'. Shaded cells are not used for synchronous master transmission.

* Page provides register information.

Note 1: PIC16(L)F1619 only.

2: Unimplemented, read as '1'.

25.5.1.5 Synchronous Master Reception

Data is received at the RX/DT pin. The RX/DT pin output driver is automatically disabled 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 RX/DT pin on the trailing edge of the TX/CK clock pin and is shifted into the Receive Shift Register (RSR). When a complete character is received into the RSR, the RCIF 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 RCIF bit remains set as long as there are unread characters in the receive FIFO.

Note: If the RX/DT function is on an analog pin, the corresponding ANSEL bit must be cleared for the receiver to function.

25.5.1.6 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 TX/CK line. The TX/CK pin output driver is automatically disabled 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.

Note: If the device is configured as a slave and the TX/CK function is on an analog pin, the corresponding ANSEL bit must be cleared.

25.5.1.7 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.

25.5.1.8 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 nine 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 eight Least Significant bits from the RCxREG.

25.5.1.9 Synchronous Master Reception Setup:

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. Clear the ANSEL bit for the RX pin (if applicable).
3. Enable the synchronous master serial port by setting bits SYNC, SPEN and CSRC.
4. Ensure bits CREN and SREN are clear.
5. If interrupts are desired, set the RCIE bit of the PIE1 register and the GIE and PEIE bits of the INTCON register.
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 RCIF will be set when reception of a character is complete. An interrupt will be generated if the enable bit RCIE 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.

FIGURE 25-12: SYNCHRONOUS RECEPTION (MASTER MODE, SREN)

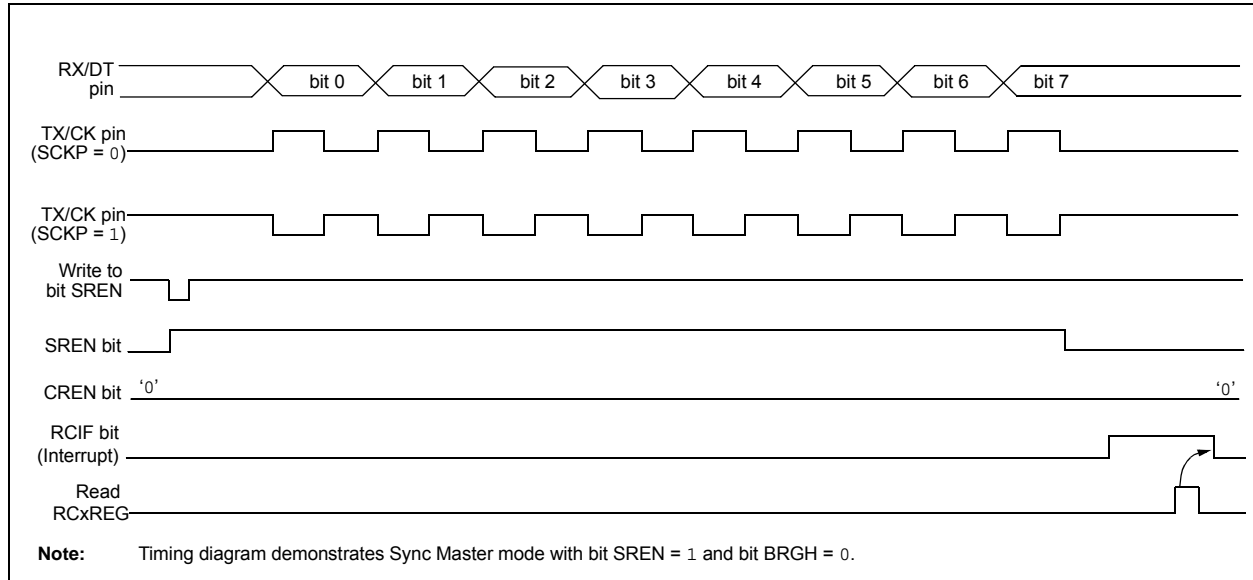


TABLE 25-8: SUMMARY OF REGISTERS ASSOCIATED WITH SYNCHRONOUS MASTER RECEPTION

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Register on Page |
|-----------------------|------------------------------|-----------------------|--------|-------------|------------------|--------|--------|--------|------------------|
| ANSELA | — | — | — | ANSA4 | — | ANSA2 | ANSA1 | ANSA0 | 152 |
| ANSELB ⁽¹⁾ | — | — | ANSB5 | ANSB4 | — | — | — | — | 159 |
| ANSELC | ANSC7 ⁽¹⁾ | ANSC6 ⁽¹⁾ | — | — | ANSC3 | ANSC2 | ANSC1 | ANSC0 | 166 |
| BAUD1CON | ABDOVF | RCIDL | — | SCKP | BRG16 | — | WUE | ABDEN | 325 |
| CKPPS | — | — | — | CKPPS<4:0> | | | | | 174, 172 |
| INTCON | GIE | PEIE | TMR0IE | INTE | IOCIE | TMR0IF | INTF | IOCIF | 97 |
| PIE1 | TMR1GIE | ADIE | RCIE | TXIE | SSP1IE | CCP1IE | TMR2IE | TMR1IE | 98 |
| PIR1 | TMR1GIF | ADIF | RCIF | TXIF | SSP1IF | CCP1IF | TMR2IF | TMR1IF | 103 |
| RC1REG | EUSART Receive Data Register | | | | | | | | 318* |
| RC1STA | SPEN | RX9 | SREN | CREN | ADDEN | FERR | OERR | RX9D | 324 |
| RXPPS | — | — | — | RXPPS<4:0> | | | | | 174, 172 |
| RxyPPS | — | — | — | RxyPPS<4:0> | | | | | 172 |
| SP1BRGL | BRG<7:0> | | | | | | | | 326* |
| SP1BRGH | BRG<15:8> | | | | | | | | 326* |
| TRISA | — | — | TRISA5 | TRISA4 | — ⁽²⁾ | TRISA2 | TRISA1 | TRISA0 | 151 |
| TRISB ⁽¹⁾ | TRISB7 | TRISB6 | TRISB5 | TRISB4 | — | — | — | — | 158 |
| TRISC | TRISC7 ⁽¹⁾ | TRISC6 ⁽¹⁾ | TRISC5 | TRISC4 | TRISC3 | TRISC2 | TRISC1 | TRISC0 | 165 |
| TX1STA | CSRC | TX9 | TXEN | SYNC | SENCB | BRGH | TRMT | TX9D | 323 |

Legend: — = unimplemented location, read as '0'. Shaded cells are not used for synchronous master reception.

* Page provides register information.

Note 1: PIC16(L)F1619 only.

Note 2: Unimplemented, read as '1'.

25.5.2 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.

25.5.2.1 EUSART Synchronous Slave Transmit

The operation of the Synchronous Master and Slave modes are identical (see [Section 25.5.1.3 “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 the TXxREG register.
3. The TXIF 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 TXIF bit will now be set.
5. If the PEIE and TXIE 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.

25.5.2.2 Synchronous Slave Transmission Set-up:

1. Set the SYNC and SPEN bits and clear the CSRC bit.
2. Clear the ANSEL bit for the CK pin (if applicable).
3. Clear the CREN and SREN bits.
4. If interrupts are desired, set the TXIE bit of the PIE1 register and the GIE and PEIE bits of the INTCON register.
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 eight bits to the TXxREG register.

TABLE 25-9: 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 |
|-----------------------|-------------------------------|-----------------------|--------|-------------|------------------|--------|--------|--------|------------------|
| ANSELA | — | — | — | ANSA4 | — | ANSA2 | ANSA1 | ANSA0 | 152 |
| ANSELB ⁽¹⁾ | — | — | ANSB5 | ANSB4 | — | — | — | — | 159 |
| ANSELC | ANSC7 ⁽¹⁾ | ANSC6 ⁽¹⁾ | — | — | ANSC3 | ANSC2 | ANSC1 | ANSC0 | 166 |
| BAUD1CON | ABDOVF | RCIDL | — | SCKP | BRG16 | — | WUE | ABDEN | 325 |
| CKPPS | — | — | — | CKPPS<4:0> | | | | | 174, 172 |
| INTCON | GIE | PEIE | TMR0IE | INTE | IOCF | TMR0IF | INTF | IOCF | 97 |
| PIE1 | TMR1GIE | ADIE | RCIE | TXIE | SSP1IE | CCP1IE | TMR2IE | TMR1IE | 98 |
| PIR1 | TMR1GIF | ADIF | RCIF | TXIF | SSP1IF | CCP1IF | TMR2IF | TMR1IF | 103 |
| RC1STA | SPEN | RX9 | SREN | CREN | ADDEN | FERR | OERR | RX9D | 324 |
| RXPPS | — | — | — | RXPPS<4:0> | | | | | 174, 172 |
| RxyPPS | — | — | — | RxyPPS<4:0> | | | | | 172 |
| TRISA | — | — | TRISA5 | TRISA4 | — ⁽²⁾ | TRISA2 | TRISA1 | TRISA0 | 151 |
| TRISB ⁽¹⁾ | TRISB7 | TRISB6 | TRISB5 | TRISB4 | — | — | — | — | 158 |
| TRISC | TRISC7 ⁽¹⁾ | TRISC6 ⁽¹⁾ | TRISC5 | TRISC4 | TRISC3 | TRISC2 | TRISC1 | TRISC0 | 165 |
| TX1REG | EUSART Transmit Data Register | | | | | | | | 315* |
| TX1STA | CSRC | TX9 | TXEN | SYNC | SEnDB | BRGH | TRMT | TX9D | 323 |

Legend: — = unimplemented location, read as '0'. Shaded cells are not used for synchronous slave transmission.

* Page provides register information.

Note 1: PIC16(L)F1619 only.

Note 2: Unimplemented, read as '1'.

25.5.2.3 EUSART Synchronous Slave Reception

The operation of the Synchronous Master and Slave modes is identical ([Section 25.5.1.5 “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 RCIE 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.

25.5.2.4 Synchronous Slave Reception Set-up:

1. Set the SYNC and SPEN bits and clear the CSRC bit.
2. Clear the ANSEL bit for both the CK and DT pins (if applicable).
3. If interrupts are desired, set the RCIE bit of the PIE1 register and the GIE and PEIE bits of the INTCON register.
4. If 9-bit reception is desired, set the RX9 bit.
5. Set the CREN bit to enable reception.
6. The RCIF bit will be set when reception is complete. An interrupt will be generated if the RCIE 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 eight 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 25-10: SUMMARY OF 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 |
|-----------------------|------------------------------|-----------------------|--------|------------|------------------|--------|--------|--------|------------------|
| ANSELA | — | — | — | ANSA4 | — | ANSA2 | ANSA1 | ANSA0 | 152 |
| ANSELB ⁽¹⁾ | — | — | ANSB5 | ANSB4 | — | — | — | — | 159 |
| ANSELC | ANSC7 ⁽¹⁾ | ANSC6 ⁽¹⁾ | — | — | ANSC3 | ANSC2 | ANSC1 | ANSC0 | 166 |
| BAUD1CON | ABDOVF | RCIDL | — | SCKP | BRG16 | — | WUE | ABDEN | 325 |
| CKPPS | — | — | — | CKPPS<4:0> | | | | | 174, 172 |
| INTCON | GIE | PEIE | TMR0IE | INTE | IOCIE | TMR0IF | INTF | IOCIF | 97 |
| PIE1 | TMR1GIE | ADIE | RCIE | TXIE | SSP1IE | CCP1IE | TMR2IE | TMR1IE | 98 |
| PIR1 | TMR1GIF | ADIF | RCIF | TXIF | SSP1IF | CCP1IF | TMR2IF | TMR1IF | 103 |
| RC1REG | EUSART Receive Data Register | | | | | | | | 318* |
| RC1STA | SPEN | RX9 | SREN | CREN | ADDEN | FERR | OERR | RX9D | 324 |
| RXPPS | — | — | — | RXPPS<4:0> | | | | | 174, 172 |
| TRISA | — | — | TRISA5 | TRISA4 | — ⁽²⁾ | TRISA2 | TRISA1 | TRISA0 | 151 |
| TRISB ⁽¹⁾ | TRISB7 | TRISB6 | TRISB5 | TRISB4 | — | — | — | — | 158 |
| TRISC | TRISC7 ⁽¹⁾ | TRISC6 ⁽¹⁾ | TRISC5 | TRISC4 | TRISC3 | TRISC2 | TRISC1 | TRISC0 | 165 |
| TX1STA | CSRC | TX9 | TXEN | SYNC | SENDB | BRGH | TRMT | TX9D | 323 |

Legend: — = unimplemented location, read as ‘0’. Shaded cells are not used for synchronous slave reception.

* Page provides register information.

Note 1: PIC16(L)F1619 only.

2: Unimplemented, read as ‘1’.

25.6 EUSART Operation During Sleep

The EUSART will remain active during Sleep only in the Synchronous Slave mode. All other modes require the system clock and therefore cannot generate the necessary signals to run the Transmit or Receive Shift registers during Sleep.

Synchronous Slave mode uses an externally generated clock to run the Transmit and Receive Shift registers.

25.6.1 SYNCHRONOUS RECEIVE DURING SLEEP

To receive during Sleep, all the following conditions must be met before entering Sleep mode:

- RCxSTA and TXxSTA Control registers must be configured for Synchronous Slave Reception (see [Section 25.5.2.4 “Synchronous Slave Reception Set-up:”](#)).
- If interrupts are desired, set the RCIE bit of the PIE1 register and the GIE and PEIE bits of the INTCON register.
- The RCIF interrupt flag must be cleared by reading RCxREG to unload any pending characters in the receive buffer.

Upon entering Sleep mode, the device will be ready to accept data and clocks on the RX/DT and TX/CK pins, respectively. When the data word has been completely clocked in by the external device, the RCIF interrupt flag bit of the PIR1 register will be set. Thereby, waking the processor from Sleep.

Upon waking from Sleep, the instruction following the SLEEP instruction will be executed. If the Global Interrupt Enable (GIE) bit of the INTCON register is also set, then the Interrupt Service Routine at address 004h will be called.

25.6.2 SYNCHRONOUS TRANSMIT DURING SLEEP

To transmit during Sleep, all the following conditions must be met before entering Sleep mode:

- The RCxSTA and TXxSTA Control registers must be configured for synchronous slave transmission (see [Section 25.5.2.2 “Synchronous Slave Transmission Set-up:”](#)).
- The TXIF interrupt flag must be cleared by writing the output data to the TXxREG, thereby filling the TSR and transmit buffer.
- If interrupts are desired, set the TXIE bit of the PIE1 register and the PEIE bit of the INTCON register.
- Interrupt enable bits TXIE of the PIE1 register and PEIE of the INTCON register must be set.

Upon entering Sleep mode, the device will be ready to accept clocks on TX/CK pin and transmit data on the RX/DT pin. When the data word in the TSR has been completely clocked out by the external device, the pending byte in the TXxREG will transfer to the TSR and the TXIF flag will be set. Thereby, waking the processor from Sleep. At this point, the TXxREG is available to accept another character for transmission, which will clear the TXIF flag.

Upon waking from Sleep, the instruction following the SLEEP instruction will be executed. If the Global Interrupt Enable (GIE) bit is also set then the Interrupt Service Routine at address 0004h will be called.

26.0 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 family of devices contains two standard Capture/Compare/PWM modules (CCP1 and CCP2).

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 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.

26.1 Capture Mode

The Capture mode function described in this section is available and identical for all CCP modules.

Capture mode makes use of the 16-bit Timer1 resource. When an event occurs on the CCPx input, the 16-bit CCPRxH:CCPRxL register pair captures and stores the 16-bit value of the TMR1H:TMR1L register pair, respectively. An event is defined as one of the following and is configured by the MODE<3:0> bits of the CCPxCON register:

- Every edge (rising or falling)
- Every falling edge
- Every rising edge
- Every 4th rising edge
- Every 16th rising edge

The CCPx capture input signal is configured by the CTS bits of the CCPxCAP register with the following options:

- CCPx pin
- Comparator 1 output (C1_OUT_sync)
- Comparator 2 output (C2_OUT_sync)
- Interrupt-on-change interrupt trigger (IOC_interrupt)

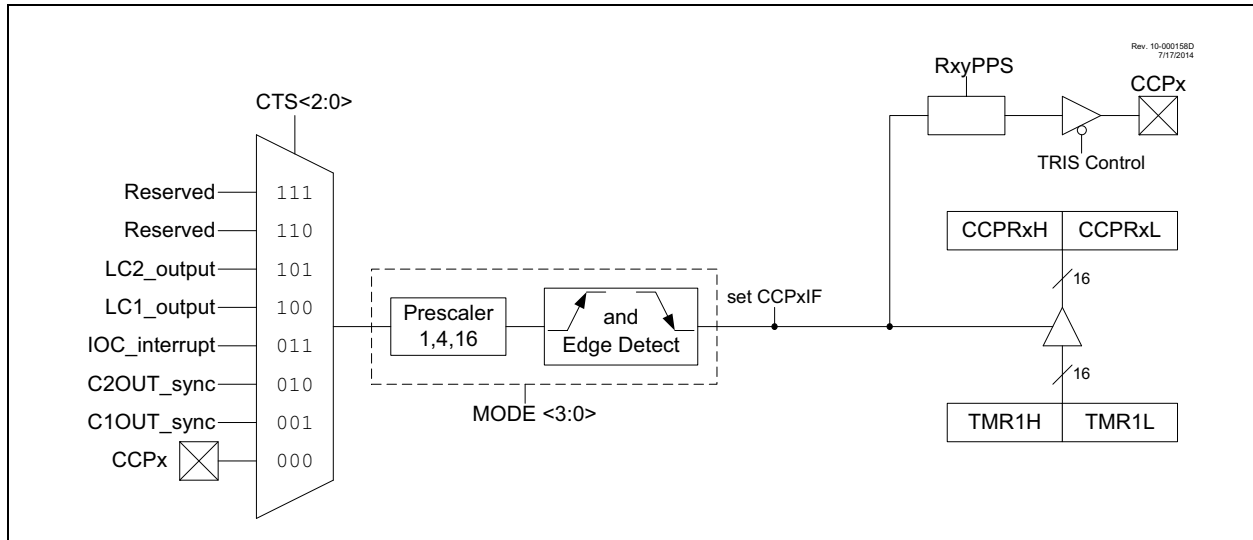
When a capture is made, the Interrupt Request Flag bit CCPxIF of the PIRx register is set. The interrupt flag must be cleared in software. If another capture occurs before the value in the CCPRxH, CCPRxL register pair is read, the old captured value is overwritten by the new captured value.

Figure 26-1 shows a simplified diagram of the capture operation.

26.1.1 CCP PIN CONFIGURATION

In Capture mode, select the interrupt source using the CTS bits of the CCPxCAP register. If the CCPx pin is chosen, it should be configured as an input by setting the associated TRIS control bit.

FIGURE 26-1: CAPTURE MODE OPERATION BLOCK DIAGRAM



26.1.2 TIMER1 MODE RESOURCE

Timer1 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 22.0 “Timer1/3/5 Module with Gate Control”](#) for more information on configuring Timer1.

26.1.3 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.

Note: Clocking Timer1 from the system clock (F_{osc}) should not be used in Capture mode. In order for Capture mode to recognize the trigger event on the CCPx pin, Timer1 must be clocked from the instruction clock ($F_{osc}/4$) or from an external clock source.

26.1.4 CCP PRESCALER

There are four prescaler settings specified by the MODE<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 EN bit of the CCPxCON register before changing the prescaler.

26.1.5 CAPTURE DURING SLEEP

Capture mode depends upon the Timer1 module for proper operation. There are two options for driving the Timer1 module in Capture mode. It can be driven by the instruction clock ($F_{osc}/4$), or by an external clock source.

When Timer1 is clocked by $F_{osc}/4$, Timer1 will not increment during Sleep. When the device wakes from Sleep, Timer1 will continue from its previous state.

Capture mode will operate during Sleep when Timer1 is clocked by an external clock source.

26.1.6 CAPTURE OUTPUT

Whenever a capture occurs, the output of the CCP will go high for a period equal to one system clock period ($1/F_{osc}$). This output is available as an input signal to the CWG, as an auto-conversion trigger for the ADC, as an External Reset Signal for the TMR2 modules, as a window input to the SMT, and as an input to the CLC module. In addition, the CCPx pin output can be mapped to output pins through the use of PPS (see [13.2 “PPS Outputs”](#)).

26.2 Compare Mode

The Compare mode function described in this section is available and identical for all CCP modules.

Compare mode makes use of the 16-bit Timer1 resource. The 16-bit value of the CCPRxH:CCPRxL register pair is constantly compared against the 16-bit value of the TMR1H:TMR1L 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
- Pulse the CCPx output
- Generate a Software Interrupt
- Optionally Reset TMR1

The action on the pin is based on the value of the MODE<3:0> control bits of the CCPxCON register. At the same time, the interrupt flag CCPxIF bit is set.

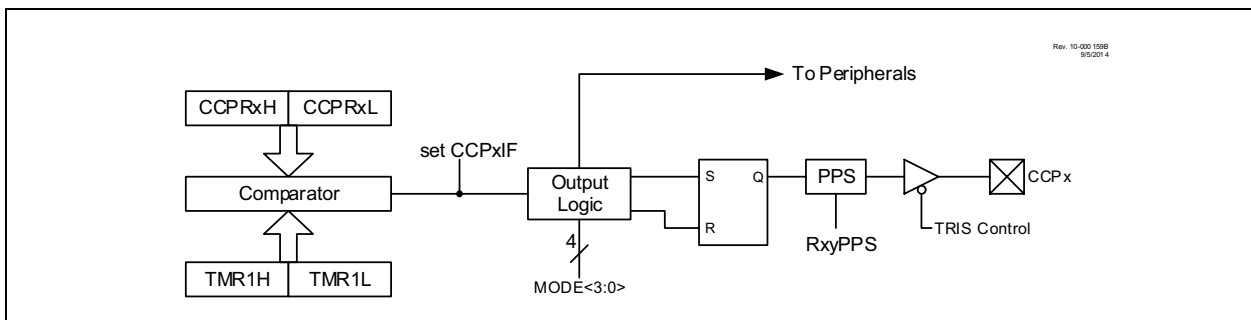
All Compare modes can generate an interrupt.

Figure 26-2 shows a simplified diagram of the compare operation.

26.2.1 CCPx PIN CONFIGURATION

The user must configure the CCPx pin as an output by clearing the associated TRIS bit.

FIGURE 26-2: COMPARE MODE OPERATION BLOCK DIAGRAM



26.2.2 TIMER1 MODE RESOURCE

In Compare mode, Timer1 must be running in either Timer mode or Synchronized Counter mode. The compare operation may not work in Asynchronous Counter mode.

See [Section 22.0 “Timer1/3/5 Module with Gate Control”](#) for more information on configuring Timer1.

Note: Clocking Timer1 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 must be clocked from the instruction clock (Fosc/4) or from an external clock source.

26.2.3 SOFTWARE INTERRUPT MODE

When Generate Software Interrupt mode is chosen (MODE<3:0> = 1010), the CCPx module does not assert control of the CCPx pin (see the CCPxCON register).

26.2.4 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.

26.2.5 CAPTURE OUTPUT

When in Compare mode, the CCP will provide an output upon the 16-bit value of the CCPRxH:CCPRxL register pair matching the TMR1H:TMR1L register pair. The compare output depends on which Compare mode the CCP is configured as. If the MODE bits of CCPxCON register are equal to '1011' or '1010', the CCP module will output high, while TMR1 is equal to CCPRxH:CCPRxL register pair. This means that the pulse width is determined by the TMR1 prescaler. If the MODE bits of CCPxCON are equal to '0001' or '0010', the output will toggle upon a match, going from '0' to '1' or vice-versa. If the MODE bits of CCPxCON are equal to '1001', the output is cleared on a match, and if the MODE bits are equal to '1000', the output is set on a match. This output is available as an input signal to the CWG, as an auto-conversion trigger for the ADC, as an external Reset signal for the TMR2 modules, as a window input to the SMT, and as an input to the CLC module. In addition, the CCPx pin output can be mapped to output pins through the use of PPS (see [Section 13.2 “PPS Outputs”](#)).

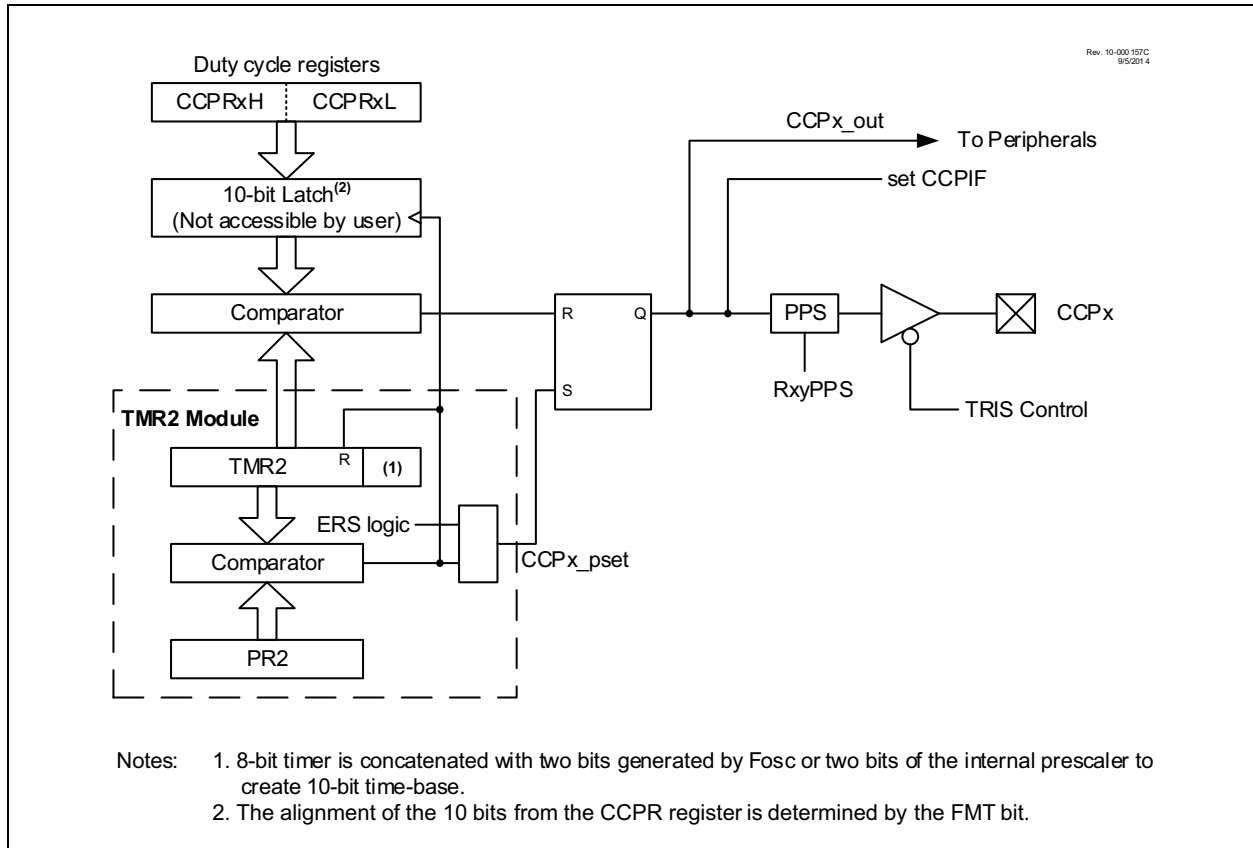
26.3 PWM Overview

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 26-3: SIMPLIFIED PWM BLOCK DIAGRAM



26.3.1 STANDARD PWM OPERATION

The standard PWM function described in this section is available and identical for all 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:

- PR2/4/6 registers
- T2CON/T4CON/T6CON registers
- CCPRxH:CCPRxL register pair

Figure 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.

26.3.2 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. Determine which timer will be used to clock the CCP; Timer2/4/6.
3. Load the associated PR2/4/6 register with the PWM period value.
4. Configure the CCP module for the PWM mode by loading the CCPxCON register with the appropriate values.
5. Load the CCPRxH:CCPRxL register pair with the PWM duty cycle value.
6. Configure and start Timer2/4/6:
 - Clear the TMR2IF/TMR4IF/TMR6IF interrupt flag bit of the PIRx register. See Note below.
 - Configure the CKPS bits of the TxCON register with the Timer prescale value.
 - Enable the Timer by setting the ON bit of the TxCON register.
7. Enable PWM output pin:
 - Wait until the Timer overflows and the TMR2IF/TMR4IF/TMR6IF 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.

26.4 CCP/PWM Clock Selection

The PIC16(L)F1615/9 allows each individual CCP and PWM module to select the timer source that controls the module. Each module has an independent selection.

As there are up to three 8-bit timers with auto-reload (Timer2/4/6), PWM mode on the CCP and PWM modules can use any of these timers.

The CCPTMRS register is used to select which timer is used.

26.4.1 USING THE TMR2/4/6 WITH THE CCP MODULE

This device has a new version of the TMR2 module that has many new modes, which allow for greater customization and control of the PWM signals than older parts. Refer to [Section 23.6 “Operation Examples”](#) for examples of PWM signal generation using the different modes of Timer2. The CCP operation requires that the timer used as the PWM time base has the FOSC/4 clock source selected.

26.4.2 PWM PERIOD

The PWM period is specified by the PR2/4/6 register of Timer2/4/6. The PWM period can be calculated using the formula of [Equation 26-1](#).

EQUATION 26-1: PWM PERIOD

$$PWM\ Period = [(PR2) + 1] \cdot 4 \cdot TOSC \cdot (TMR2\ Prescale\ Value)$$

Note 1: TOSC = 1/FOSC

When TMR2/4/6 is equal to its respective PR2/4/6 register, the following three events occur on the next increment cycle:

- TMR2/4/6 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 the CCPRxH:CCPRxL pair into the internal 10-bit latch.

Note: The Timer postscaler (see [Figure](#)) is not used in the determination of the PWM frequency.

26.4.3 PWM DUTY CYCLE

The PWM duty cycle is specified by writing a 10-bit value to two registers: the CCPRxH:CCPRxL register pair. Where the particular bits go is determined by the FMT bit of the CCPxCON register. If FMT = 0, the two Most Significant bits of the duty cycle value should be written to bits <1:0> of CCPRxH register and the remaining eight bits to the CCPRxL register. If FMT = 1, the Least

Significant two bits of the duty cycle should be written to bits <7:6> of the CCPRxL register and the Most Significant eight bits to the CCPRxH register. This is illustrated in [Figure 26-4](#). These bits can be written at any time. The duty cycle value is not latched into the internal latch until after the period completes (i.e., a match between PR2/4/6 and TMR2/4/6 registers occurs).

[Equation 26-2](#) is used to calculate the PWM pulse width. [Equation 26-3](#) is used to calculate the PWM duty cycle ratio.

EQUATION 26-2: PULSE WIDTH

$$Pulse\ Width = CCPRxH:CCPRxL \cdot TOSC \cdot (TMR2\ Prescale\ Value)$$

EQUATION 26-3: DUTY CYCLE RATIO

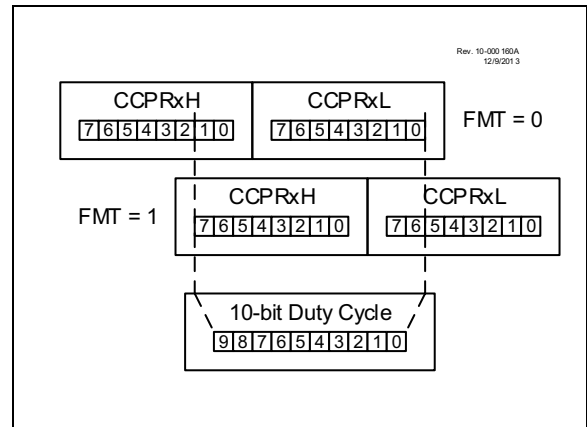
$$Duty\ Cycle\ Ratio = \frac{CCPRxH:CCPRxL}{4(PR_x + 1)}$$

The PWM duty cycle registers are double buffered for glitchless PWM operation.

The 8-bit timer TMR2/4/6 register is concatenated with either the 2-bit internal system clock (FOSC), or two bits of the prescaler, to create the 10-bit time base. The system clock is used if the Timer2/4/6 prescaler is set to 1:1.

When the 10-bit time base matches the internal buffer register, then the CCPx pin is cleared (see [Figure](#)).

FIGURE 26-4: CCPx DUTY-CYCLE ALIGNMENT



26.4.4 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 ten bits when PR2/4/6 is 255. The resolution is a function of the PR2/4/6 register value as shown by [Equation 26-4](#).

EQUATION 26-4: PWM RESOLUTION

$$Resolution = \frac{\log[4(PR2 + 1)]}{\log(2)} \text{ bits}$$

Note: If the pulse width value is greater than the period, the assigned PWM pin(s) will remain unchanged.

TABLE 26-1: 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 | 16 | 4 | 1 | 1 | 1 | 1 |
| PR2 Value | 0xFF | 0xFF | 0xFF | 0x3F | 0x1F | 0x17 |
| Maximum Resolution (bits) | 10 | 10 | 10 | 8 | 7 | 6 |

TABLE 26-2: 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 | 16 | 4 | 1 | 1 | 1 | 1 |
| PR2 Value | 0x65 | 0x65 | 0x65 | 0x19 | 0x0C | 0x09 |
| Maximum Resolution (bits) | 8 | 8 | 8 | 6 | 5 | 5 |

26.4.5 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”](#) for additional details.

26.4.6 EFFECTS OF RESET

Any Reset will force all ports to Input mode and the CCP registers to their Reset states.

26.4.7 PWM OUTPUT

The output of the CCP in PWM mode is the PWM signal generated by the module and described above. This output is available as an input signal to the CWG, as an auto-conversion trigger for the ADC, as an external Reset signal for the TMR2 modules, as a window input to the SMT, and as an input to the CLC module. In addition, the CCPx pin output can be mapped to output pins through the use of PPS (see [Section 13.2 “PPS Outputs”](#)).

26.5 Register Definitions: CCP Control

REGISTER 26-1: CCPxCON: CCPx CONTROL REGISTER

| | | | | | | | |
|---------|-----|-----|---------|-----------|---------|---------|---------|
| R/W-0/0 | U-0 | R-x | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 |
| EN | — | OUT | FMT | MODE<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 **EN:** CCPx Module Enable bit
 1 = CCPx is enabled
 0 = CCPx is disabled
- bit 6 **Unimplemented:** Read as '0'
- bit 5 **OUT:** CCPx Output Data bit (read-only)
- bit 4 **FMT:** CCPW (Pulse-Width) Alignment bit
 If MODE = PWM Mode:
 1 = Left-aligned format, CCPRxH <7> is the MSb of the PWM duty cycle
 0 = Right-aligned format, CCPRxL <0> is the LSb of the PWM duty cycle
- bit 3-0 **MODE<3:0>:** CCPx Mode Selection bit
 11xx = PWM mode
- 1011 = Compare mode: Pulse output, clear TMR1
 1010 = Compare mode: Pulse output (0 - 1 - 0)
 1001 = Compare mode: clear output on compare match
 1000 = Compare mode: set output on compare match
- 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 = Capture mode: every rising or falling edge
 0010 = Compare mode: toggle output on match
 0001 = Compare mode: Toggle output and clear TMR1 on match
 0000 = Capture/Compare/PWM off (resets CCPx module) (reserved for backwards compatibility)

REGISTER 26-2: CCPTMRS: PWM 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 |
| P4TSEL<1:0> | | P3TSEL<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 **P4TSEL<1:0>**: PWM4 Timer Selection bits
 11 = Reserved
 10 = PWM4 is based off Timer6 in PWM mode
 01 = PWM4 is based off Timer4 in PWM mode
 00 = PWM4 is based off Timer2 in PWM mode
- bit 5-4 **P3TSEL<1:0>**: PWM3 Timer Selection bits
 11 = Reserved
 10 = PWM3 is based off Timer6 in PWM mode
 01 = PWM3 is based off Timer4 in PWM mode
 00 = PWM3 is based off Timer2 in PWM mode
- bit 3-2 **C2TSEL<1:0>**: CCP2 (PWM2) Timer Selection bits
 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 (PWM1) Timer Selection bits
 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 26-3: CCPRxL: CCPx 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 |
| CCPR<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 Reset |
| '1' = Bit is set | '0' = Bit is cleared | |

bit 7-0

MODE = Capture Mode

CCPRxL<7:0>: LSB of captured TMR1 value

MODE = Compare Mode

CCPRxL<7:0>: LSB compared to TMR1 value

MODE = PWM Mode && FMT = 0

CCPRxL<7:0>: CCPW<7:0> — Pulse width Least Significant eight bits

MODE = PWM Mode && FMT = 1

CCPRxL<7:6>: CCPW<1:0> — Pulse width Least Significant two bits

CCPRxL<5:0>: Not used

REGISTER 26-4: CCPRxH: CCPx HIGH 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 |
| CCPR<15:8> | | | | | | | |
| 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 Reset |
| '1' = Bit is set | '0' = Bit is cleared | |

bit 7-0 MODE = Capture Mode:
CCPRxH<7:0>: MSB of captured TMR1 value
MODE = Compare Mode:
CCPRxH<7:0>: MSB compared to TMR1 value
MODE = PWM Mode && FMT = 0:
CCPRxH<7:2>: Not used
CCPRxH<1:0>: CCPW<9:8> — Pulse width Most Significant two bits
MODE = PWM Mode && FMT = 1:
CCPRxH<7:0>: CCPW<9:2> — Pulse width Most Significant eight bits

REGISTER 26-5: CCPxCAP: CCPx CAPTURE INPUT SELECTION REGISTER

| | | | | | | | |
|-------|-----|-----|-----|-----|----------|---------|---------|
| U-0 | U-0 | U-0 | U-0 | U-0 | R/W-0/0 | R/W-0/0 | R/W-0/0 |
| — | — | — | — | — | CTS<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 Reset |
| '1' = Bit is set | '0' = Bit is cleared | |

bit 7-3 **Unimplemented:** Read as '0'
bit 2-0 **CTS<2:0>**: Capture Trigger Input Selection bits
111 = LC4_out
110 = LC3_out
101 = LC2_out
100 = LC1_out
011 = IOC_interrupt
010 = C2_OUT_sync
001 = C1_OUT_sync
000 = CCPx pin

TABLE 26-3: 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 |
|---------|--------------------------------------|-----------|-------------|--------|------------------|--------|-------------|--------|------------------|
| CCPxCAP | — | — | — | — | — | — | CTS<1:0> | | 357 |
| CCP1CON | EN | — | OUT | FMT | MODE<3:0> | | | | 354 |
| CCP2CON | EN | — | OUT | FMT | MODE<3:0> | | | | 354 |
| CCPRxL | Capture/Compare/PWM Register x (LSB) | | | | | | | | 356 |
| CCPRxH | Capture/Compare/PWM Register x (MSB) | | | | | | | | 357 |
| CCPTMRS | P4TSEL<1:0> | | P3TSEL<1:0> | | C2TSEL<1:0> | | C1TSEL<1:0> | | 355 |
| INTCON | GIE | PEIE | TMR0IE | INTE | IOCIE | TMR0IF | INTF | IOCIF | 97 |
| PIE1 | TMR1GIE | ADIE | RCIE | TXIE | SSP1IE | CCP1IE | TMR2IE | TMR1IE | 98 |
| PIE2 | OSFIE | C2IE | C1IE | — | BCL1IE | TMR6IE | TMR4IE | CCP2IE | 107 |
| PR2 | Timer2 Period Register | | | | | | | | 244* |
| T2CON | ON | CKPS<2:0> | | | OUTPS<3:0> | | | | 256 |
| TMR2 | Timer2 Module Register | | | | | | | | 236* |
| PR4 | Timer4 Period Register | | | | | | | | 244* |
| T4CON | ON | CKPS<2:0> | | | OUTPS<3:0> | | | | 256 |
| TMR4 | Timer4 Module Register | | | | | | | | 236* |
| PR6 | Timer6 Period Register | | | | | | | | 244* |
| T6CON | ON | CKPS<2:0> | | | OUTPS<3:0> | | | | 256 |
| TMR6 | Timer6 Module Register | | | | | | | | 236* |
| TRISA | — | — | TRISA5 | TRISA4 | — ⁽¹⁾ | TRISA2 | TRISA1 | TRISA0 | 151 |

Legend: — = Unimplemented location, read as '0'. Shaded cells are not used by the PWM.

* Page provides register information.

Note 1: Unimplemented, read as '1'.

27.0 PULSE-WIDTH MODULATION (PWM) MODULE

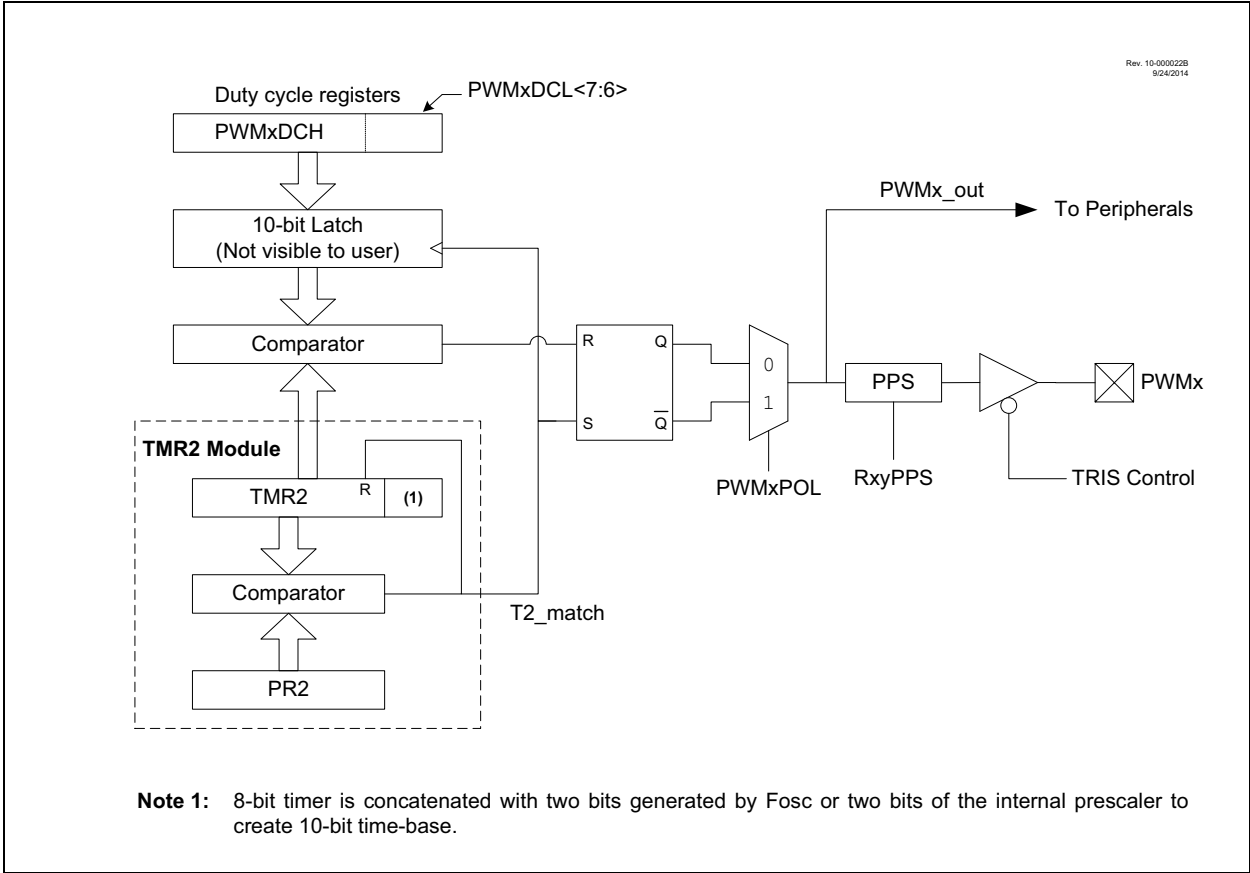
The PWM module generates a Pulse-Width Modulated signal determined by the duty cycle, period, and resolution that are configured by the following registers:

- PR2
- T2CON
- PWMxDCH
- PWMxDCL
- PWMxCON

Figure 27-1 shows a simplified block diagram of PWM operation.

For a step-by-step procedure on how to set up this module for PWM operation, refer to Section 27.1.9 "Setup for PWM Operation using PWMx Pins".

FIGURE 27-1: SIMPLIFIED PWM BLOCK DIAGRAM



27.1 PWMx Pin Configuration

All PWM outputs are multiplexed with the PORT data latch. The user must configure the pins as outputs by clearing the associated TRIS bits.

27.1.1 FUNDAMENTAL OPERATION

The PWM module produces a 10-bit resolution output. Timer2 and PR2 set the period of the PWM. The PWMxDCL and PWMxDCH registers configure the duty cycle. The period is common to all PWM modules, whereas the duty cycle is independently controlled.

Note: The Timer2 postscaler is not used in the determination of the PWM frequency. The postscaler could be used to have a servo update rate at a different frequency than the PWM output.

All PWM outputs associated with Timer2 are set when TMR2 is cleared. Each PWMx is cleared when TMR2 is equal to the value specified in the corresponding PWMxDCH (8 MSb) and PWMxDCL<7:6> (2 LSb) registers. When the value is greater than or equal to PR2, the PWM output is never cleared (100% duty cycle).

Note: The PWMxDCH and PWMxDCL registers are double buffered. The buffers are updated when Timer2 matches PR2. Care should be taken to update both registers before the timer match occurs.

27.1.2 PWM OUTPUT POLARITY

The output polarity is inverted by setting the PWMxPOL bit of the PWMxCON register.

27.1.3 PWM PERIOD

The PWM period is specified by the PR2 register of Timer2. The PWM period can be calculated using the formula of [Equation 27-1](#).

EQUATION 27-1: PWM PERIOD

$$PWM\ Period = [(PR2) + 1] \cdot 4 \cdot TOSC \cdot (TMR2\ Prescale\ Value)$$

Note: $TOSC = 1/FOSC$

When TMR2 is equal to PR2, the following three events occur on the next increment cycle:

- TMR2 is cleared
- The PWM output is active. (Exception: When the PWM duty cycle = 0%, the PWM output will remain inactive.)
- The PWMxDCH and PWMxDCL register values are latched into the buffers.

Note: The Timer2 postscaler has no effect on the PWM operation.

27.1.4 PWM DUTY CYCLE

The PWM duty cycle is specified by writing a 10-bit value to the PWMxDCH and PWMxDCL register pair. The PWMxDCH register contains the eight MSBs and the PWMxDCL<7:6>, the two LSbs. The PWMxDCH and PWMxDCL registers can be written to at any time.

[Equation 27-2](#) is used to calculate the PWM pulse width.

[Equation 27-3](#) is used to calculate the PWM duty cycle ratio.

EQUATION 27-2: PULSE WIDTH

$$Pulse\ Width = (PWMxDCH:PWMxDCL<7:6>) \cdot TOSC \cdot (TMR2\ Prescale\ Value)$$

Note: $TOSC = 1/FOSC$

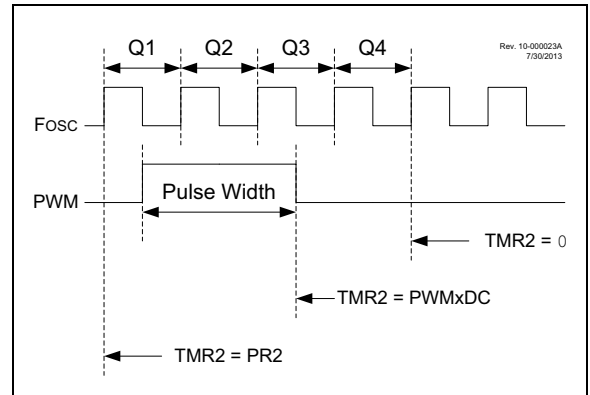
EQUATION 27-3: DUTY CYCLE RATIO

$$Duty\ Cycle\ Ratio = \frac{(PWMxDCH:PWMxDCL<7:6>)}{4(PR2 + 1)}$$

The 8-bit timer TMR2 register is concatenated with the two Least Significant bits of 1/FOSC, adjusted by the Timer2 prescaler to create the 10-bit time base. The system clock is used if the Timer2 prescaler is set to 1:1.

[Figure 27-2](#) shows a waveform of the PWM signal when the duty cycle is set for the smallest possible pulse.

FIGURE 27-2: PWM OUTPUT



27.1.5 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 ten bits when PR2 is 255. The resolution is a function of the PR2 register value as shown by [Equation 27-4](#).

EQUATION 27-4: PWM RESOLUTION

$$Resolution = \frac{\log[4(PR2 + 1)]}{\log(2)} \text{ bits}$$

Note: If the pulse width value is greater than the period the assigned PWM pin(s) will remain unchanged.

TABLE 27-1: EXAMPLE PWM FREQUENCIES AND RESOLUTIONS (Fosc = 20 MHz)

| PWM Frequency | 0.31 kHz | 4.88 kHz | 19.53 kHz | 78.12 kHz | 156.3 kHz | 208.3 kHz |
|---------------------------|----------|----------|-----------|-----------|-----------|-----------|
| Timer Prescale | 64 | 4 | 1 | 1 | 1 | 1 |
| PR2 Value | 0xFF | 0xFF | 0xFF | 0x3F | 0x1F | 0x17 |
| Maximum Resolution (bits) | 10 | 10 | 10 | 8 | 7 | 6.6 |

TABLE 27-2: EXAMPLE PWM FREQUENCIES AND RESOLUTIONS (Fosc = 8 MHz)

| PWM Frequency | 0.31 kHz | 4.90 kHz | 19.61 kHz | 76.92 kHz | 153.85 kHz | 200.0 kHz |
|---------------------------|----------|----------|-----------|-----------|------------|-----------|
| Timer Prescale | 64 | 4 | 1 | 1 | 1 | 1 |
| PR2 Value | 0x65 | 0x65 | 0x65 | 0x19 | 0x0C | 0x09 |
| Maximum Resolution (bits) | 8 | 8 | 8 | 6 | 5 | 5 |

27.1.6 OPERATION IN SLEEP MODE

In Sleep mode, the TMR2 register will not increment and the state of the module will not change. If the PWMx pin is driving a value, it will continue to drive that value. When the device wakes up, TMR2 will continue from its previous state.

27.1.7 CHANGES IN SYSTEM CLOCK FREQUENCY

The PWM frequency is derived from the system clock frequency (Fosc). Any changes in the system clock frequency will result in changes to the PWM frequency. Refer to [Section 5.0 “Oscillator Module”](#) for additional details.

27.1.8 EFFECTS OF RESET

Any Reset will force all ports to Input mode and the PWM registers to their Reset states.

27.1.9 SETUP FOR PWM OPERATION USING PWMx PINS

The following steps should be taken when configuring the module for PWM operation using the PWMx pins:

1. Disable the PWMx pin output driver(s) by setting the associated TRIS bit(s).
2. Clear the PWMxCON register.
3. Load the PR2 register with the PWM period value.
4. Clear the PWMxDCH register and bits <7:6> of the PWMxDCL register.
5. Configure and start Timer2:
 - Clear the TMR2IF interrupt flag bit of the PIR1 register. See note below.
 - Configure the CKPS bits of the T2CON register with the Timer2 prescale value.
 - Enable Timer2 by setting the ON bit of the T2CON register.
6. Enable PWM output pin and wait until Timer2 overflows, TMR2IF bit of the PIR1 register is set. See note below.
7. Enable the PWMx pin output driver(s) by clearing the associated TRIS bit(s) and setting the PWMxOE bit of the PWMxCON register.
8. Configure the PWM module by loading the PWMxCON register with the appropriate values.

Note 1: In order to send a complete duty cycle and period on the first PWM output, the above steps must be followed in the order given. If it is not critical to start with a complete PWM signal, then move Step 8 to replace Step 4.

2: For operation with other peripherals only, disable PWMx pin outputs.

27.2 Register Definitions: PWM Control

REGISTER 27-1: PWMxCON: PWM CONTROL REGISTER

| | | | | | | | |
|---------|-----|---------|---------|-----|-----|-----|-------|
| R/W-0/0 | U-0 | R-0/0 | R/W-0/0 | U-0 | U-0 | U-0 | U-0 |
| PWMxEN | — | PWMxOUT | PWMxPOL | — | — | — | — |
| 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 **PWMxEN:** PWM Module Enable bit
 1 = PWM module is enabled
 0 = PWM module is disabled
- bit 6 **Unimplemented:** Read as '0'
- bit 5 **PWMxOUT:** PWM Module Output Value bit
- bit 4 **PWMxPOL:** PWMx Output Polarity Select bit
 1 = PWM output is active-low
 0 = PWM output is active-high
- bit 3-0 **Unimplemented:** Read as '0'

REGISTER 27-2: PWMxDCH: PWM DUTY CYCLE HIGH BITS

| | | | | | | | |
|--------------|---------|---------|---------|---------|---------|---------|---------|
| 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 |
| PWMxDCH<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 **PWMxDCH<7:0>:** PWM Duty Cycle Most Significant bits
 These bits are the MSBs of the PWM duty cycle. The two LSBs are found in the PWMxDCL register.

REGISTER 27-3: PWMxDCL: PWM DUTY CYCLE LOW BITS

| | | | | | | | |
|--------------|---------|-----|-----|-----|-----|-----|-------|
| R/W-x/u | R/W-x/u | U-0 | U-0 | U-0 | U-0 | U-0 | U-0 |
| PWMxDCL<7:6> | | — | — | — | — | — | — |
| 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 **PWMxDCL<7:6>:** PWM Duty Cycle Least Significant bits
 These bits are the LSBs of the PWM duty cycle. The MSBs are found in the PWMxDCH register.
- bit 5-0 **Unimplemented:** Read as '0'

TABLE 27-3: SUMMARY OF REGISTERS ASSOCIATED WITH PWM

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Register on Page |
|---------|-------------------------------|-----------------------|--------|--------|------------------|--------|--------|--------|------------------|
| PR2 | Timer2 module Period Register | | | | | | | | 361* |
| PWM3CON | EN | — | OUT | POL | — | — | — | — | 363 |
| PWM3DCH | DC<9:2> | | | | | | | | 363 |
| PWM3DCL | DC<1:0> | | — | — | — | — | — | — | 363 |
| PWM4CON | EN | — | OUT | POL | — | — | — | — | 363 |
| PWM4DCH | DC<9:2> | | | | | | | | 363 |
| PWM4DCL | DC<1:0> | | — | — | — | — | — | — | 363 |
| T2CON | ON | CKPS<2:0> | | | OUTPS<3:0> | | | | 256 |
| TMR2 | Timer2 module Register | | | | | | | | 236* |
| TRISA | — | — | TRISA5 | TRISA4 | — ⁽¹⁾ | TRISA2 | TRISA1 | TRISA0 | 151 |
| TRISC | TRISC7 ⁽²⁾ | TRISC6 ⁽²⁾ | TRISC5 | TRISC4 | TRISC3 | TRISC2 | TRISC1 | TRISC0 | 165 |

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

* Page provides register information.

Note 1: Unimplemented, read as '1'.

Note 2: PIC16(L)F1619 only.

28.0 COMPLEMENTARY WAVEFORM GENERATOR (CWG) MODULE

The Complementary Waveform Generator (CWG) produces half-bridge, full-bridge, and steering of PWM waveforms. It is backwards compatible with previous ECCP functions.

The CWG has the following features:

- Six operating modes:
 - Synchronous Steering mode
 - Asynchronous Steering mode
 - Full-Bridge mode, Forward
 - Full-Bridge mode, Reverse
 - Half-Bridge mode
 - Push-Pull mode
- Output polarity control
- Output steering
 - Synchronized to rising event
 - Immediate effect
- Independent 6-bit rising and falling event dead-band timers
 - Clocked dead band
 - Independent rising and falling dead-band enables
- Auto-shutdown control with:
 - Selectable shutdown sources
 - Auto-restart enable
 - Auto-shutdown pin override control

28.1 Fundamental Operation

The CWG module can operate in six different modes, as specified by MODE of the CWGxCON0 register:

- Half-Bridge mode (Figure 28-9)
- Push-Pull mode (Figure 28-2)
 - Full-Bridge mode, Forward (Figure 28-3)
 - Full-Bridge mode, Reverse (Figure 28-3)
- Steering mode (Figure 28-10)
- Synchronous Steering mode (Figure 28-11)

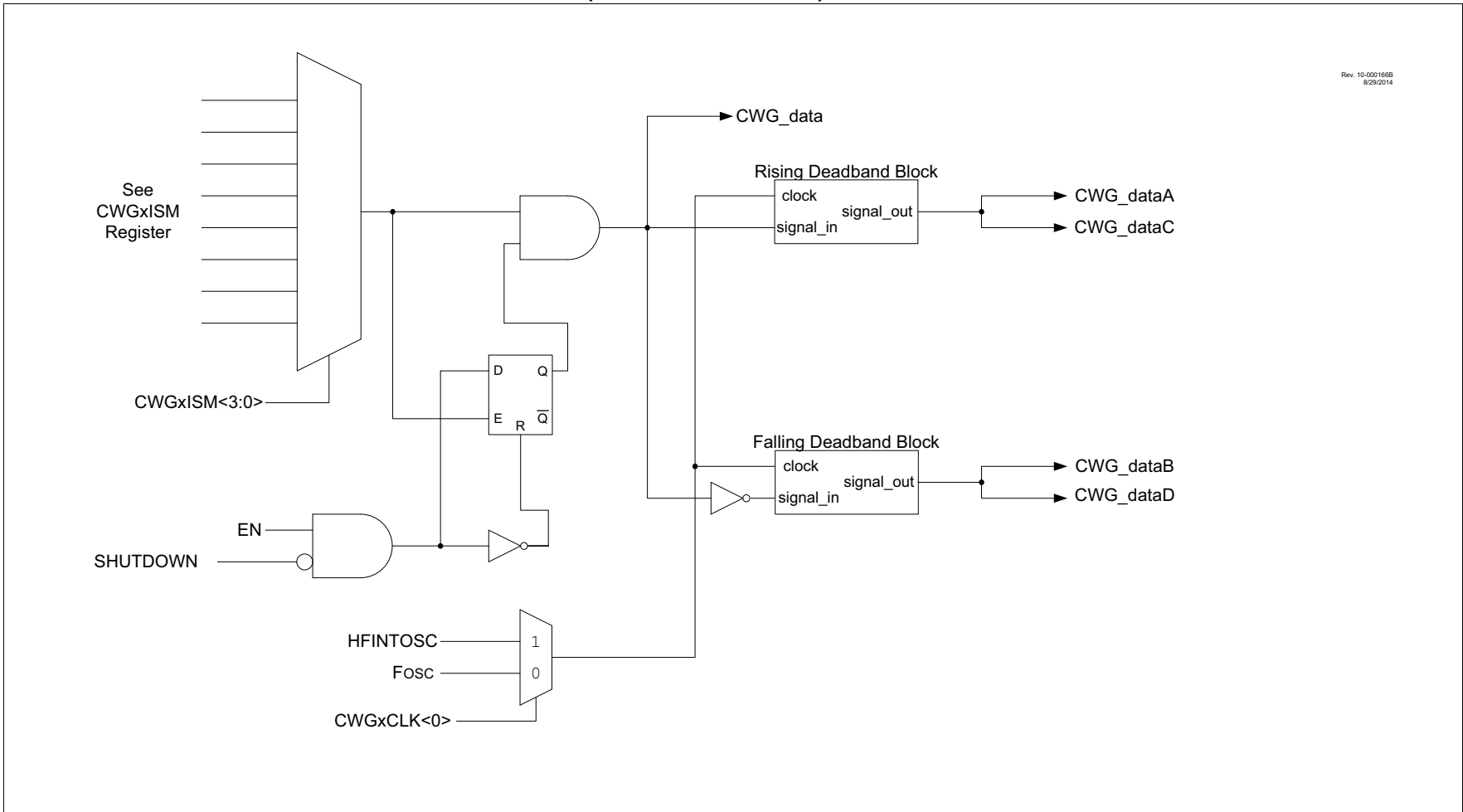
It may be necessary to guard against the possibility of circuit faults or a feedback event arriving too late or not at all. In this case, the active drive must be terminated before the Fault condition causes damage. Thus, all output modes support auto-shutdown, which is covered in [28.10 “Auto-Shutdown”](#).

28.1.1 HALF-BRIDGE MODE

In Half-Bridge mode, two output signals are generated as true and inverted versions of the input as illustrated in [Figure 28-9](#). A non-overlap (dead-band) time is inserted between the two outputs to prevent shoot through current in various power supply applications. Dead-band control is described in [Section 28.5 “Dead-Band Control”](#).

The unused outputs CWGxC and CWGxD drive similar signals, with polarity independently controlled by the POLC and POLD bits of the CWGxCON1 register, respectively.

FIGURE 28-1: SIMPLIFIED CWG BLOCK DIAGRAM (HALF-BRIDGE MODE)



28.1.2 PUSH-PULL MODE

In Push-Pull mode, two output signals are generated, alternating copies of the input as illustrated in [Figure 28-2](#). This alternation creates the push-pull effect required for driving some transformer-based power supply designs.

The push-pull sequencer is reset whenever $EN = 0$ or if an auto-shutdown event occurs. The sequencer is clocked by the first input pulse, and the first output appears on CWGxA.

The unused outputs CWGxC and CWGxD drive copies of CWGxA and CWGxB, respectively, but with polarity controlled by the POLC and POLD bits of the CWGxCON1 register, respectively.

28.1.3 FULL-BRIDGE MODES

In Forward and Reverse Full-Bridge modes, three outputs drive static values while the fourth is modulated by the input data signal. In Forward Full-Bridge mode, CWGxA is driven to its active state, CWGxB and CWGxC are driven to their inactive state, and CWGxD is modulated by the input signal. In Reverse Full-Bridge mode, CWGxC is driven to its active state, CWGxA and CWGxD are driven to their inactive states, and CWGxB is modulated by the input signal. In Full-Bridge mode, the dead-band period is used when there is a switch from forward to reverse or vice-versa. This dead-band control is described in [Section 28.5 “Dead-Band Control”](#), with additional details in [Section 28.6 “Rising Edge and Reverse Dead Band”](#) and [Section 28.7 “Falling Edge and Forward Dead Band”](#).

The mode selection may be toggled between forward and reverse by toggling the MODE<0> bit of the CWGxCON0 while keeping MODE<2:1> static, without disabling the CWG module.

FIGURE 28-2: SIMPLIFIED CWG BLOCK DIAGRAM (PUSH-PULL MODE)

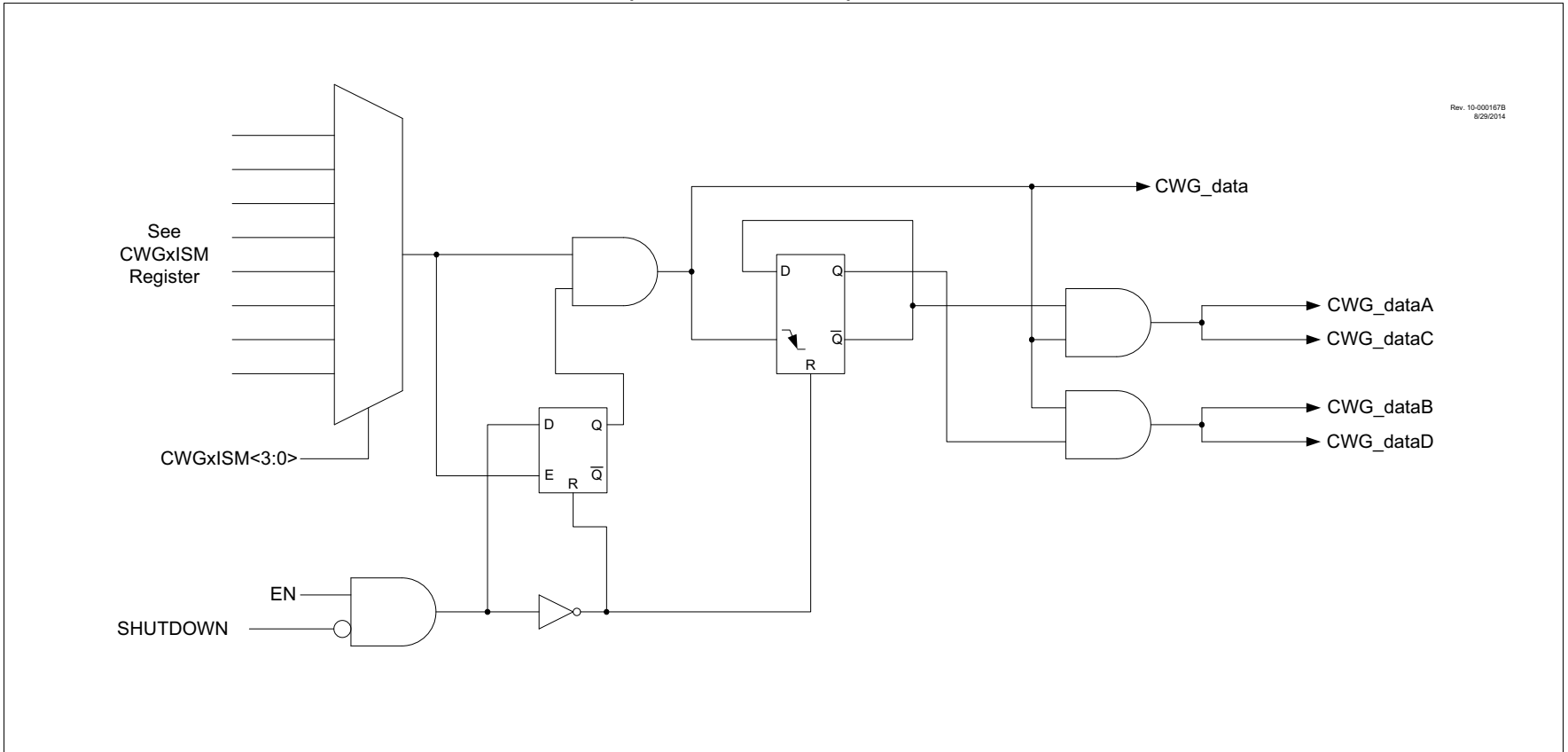
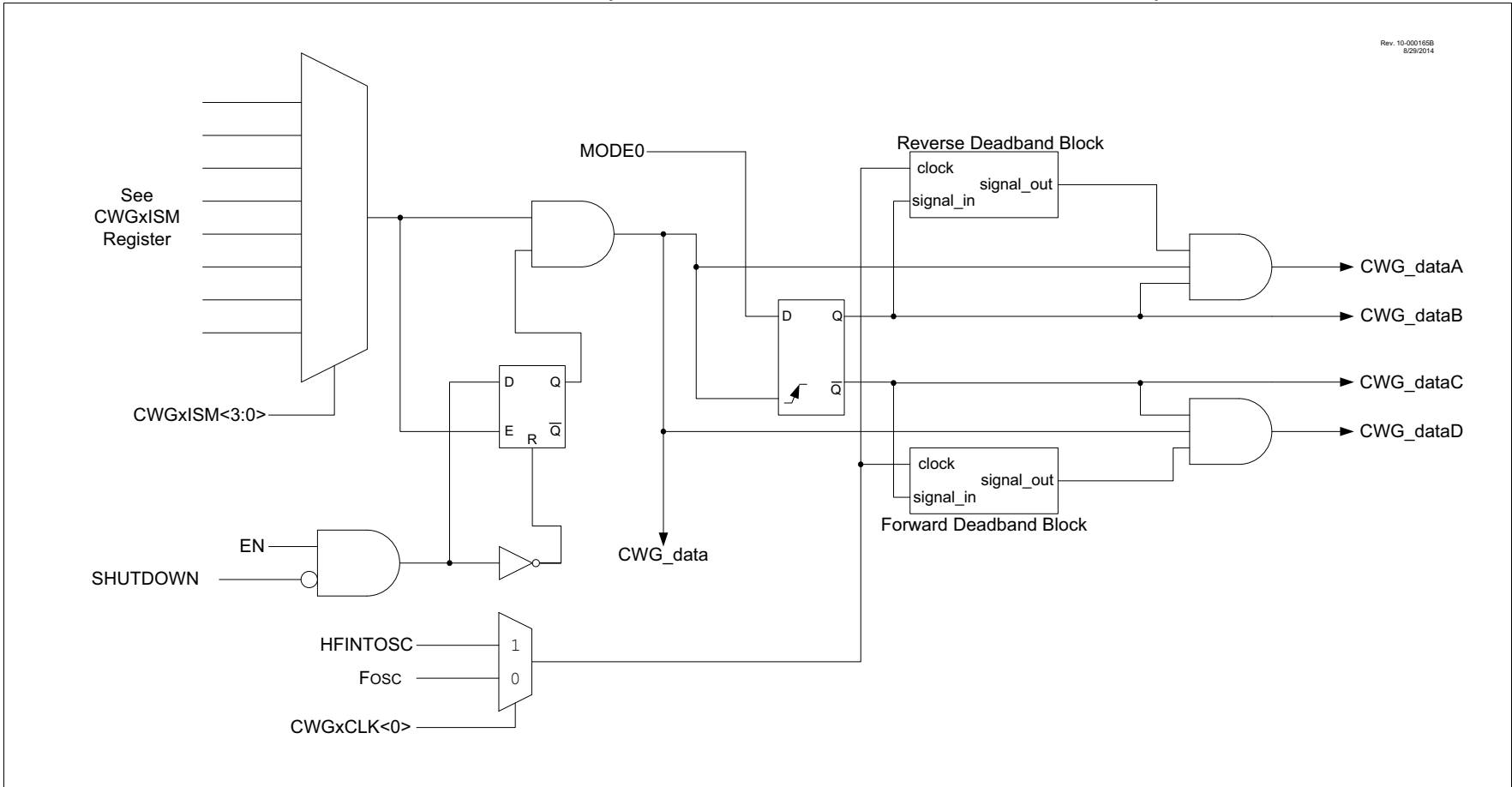


FIGURE 28-3: SIMPLIFIED CWG BLOCK DIAGRAM (FORWARD AND REVERSE FULL-BRIDGE MODES)

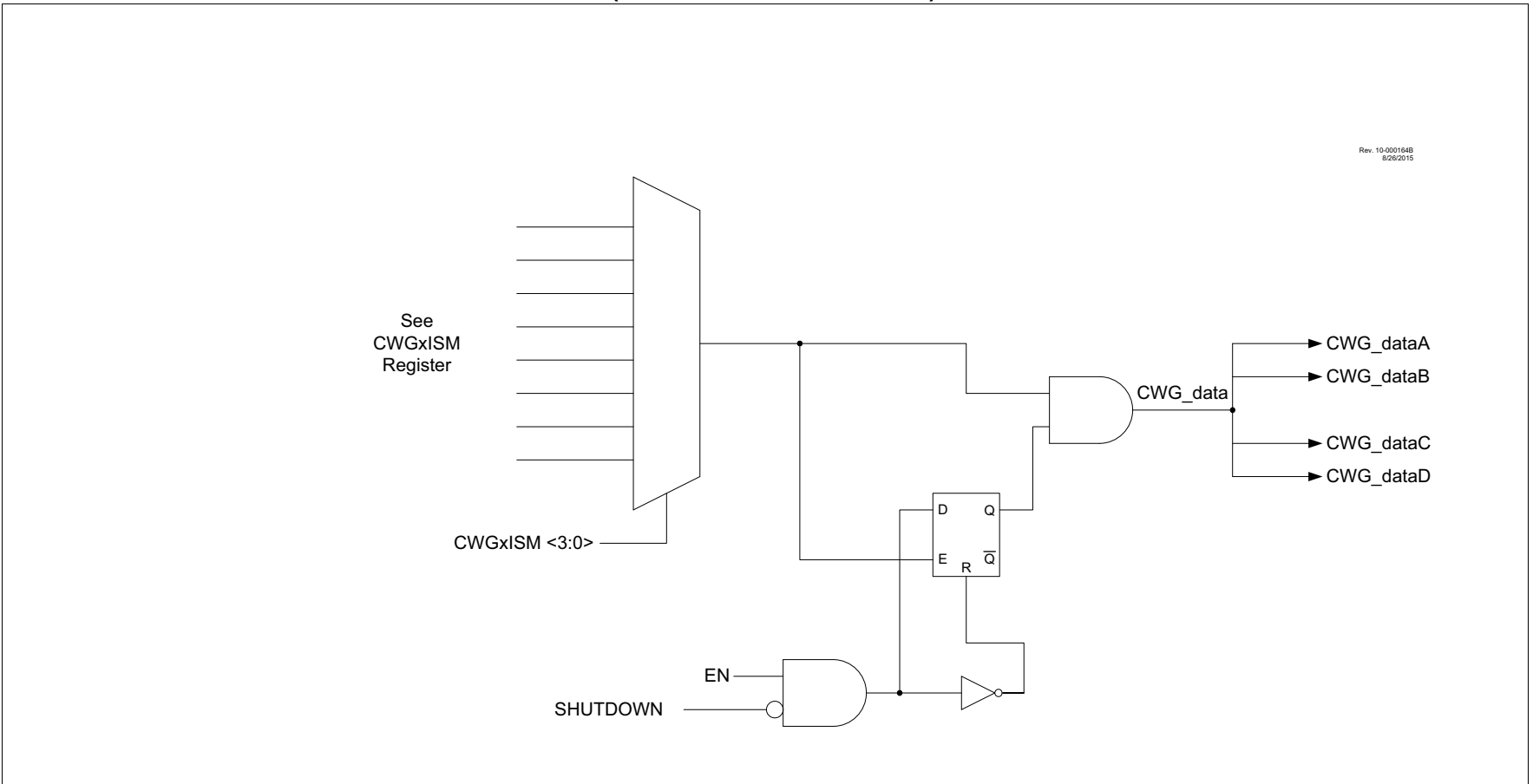


28.1.4 STEERING MODES

In Steering modes, the data input can be steered to any or all of the four CWG output pins. In Synchronous Steering mode, changes to steering selection registers take effect on the next rising input.

In Non-Synchronous mode, steering takes effect on the next instruction cycle. Additional details are provided in [Section 28.9 “CWG Steering Mode”](#).

FIGURE 28-4: SIMPLIFIED CWG BLOCK DIAGRAM (OUTPUT STEERING MODES)



28.2 Clock Source

The CWG module allows the following clock sources to be selected:

- Fosc (system clock)
- HFINTOSC (16 MHz only)

The clock sources are selected using the CS bit of the CWGxCLKCON register.

28.3 Selectable Input Sources

The CWG generates the output waveforms from the input sources in [Table 28-1](#).

TABLE 28-1: SELECTABLE INPUT SOURCES

| Source Peripheral | Signal Name |
|-------------------|---------------|
| CWG pin | PPS selection |
| Comparator C1 | C1_OUT_sync |
| Comparator C2 | C2_OUT_sync |
| CCP1 | CCP1_out |
| CCP2 | CCP2_out |
| CLC1 | LC1_out |
| CLC2 | LC2_out |
| CLC3 | LC3_out |
| CLC4 | LC4_out |
| PWM3 | PWM3_out |
| PWM4 | PWM4_out |

The input sources are selected using the CWGxISM register.

28.4 Output Control

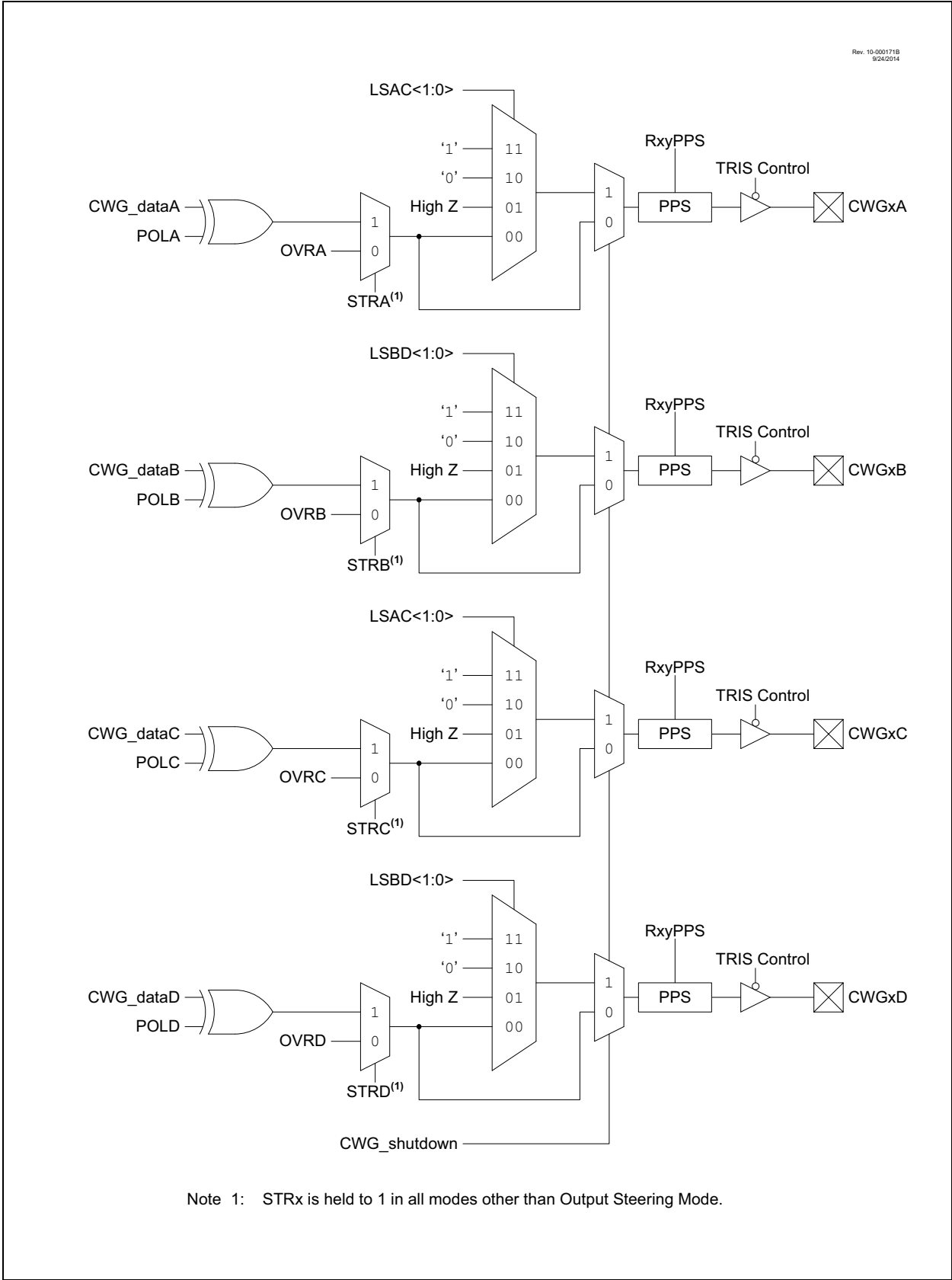
28.4.1 OUTPUT ENABLES

Each CWG output pin has individual output enable control. Output enables are selected with the Gx1OEx <3:0> bits. When an output enable control is cleared, the module asserts no control over the pin. When an output enable is set, the override value or active PWM waveform is applied to the pin per the port priority selection. The output pin enables are dependent on the module enable bit, EN of the CWGxCON0 register. When EN is cleared, CWG output enables and CWG drive levels have no effect.

28.4.2 POLARITY CONTROL

The polarity of each CWG output can be selected independently. When the output polarity bit is set, the corresponding output is active-high. Clearing the output polarity bit configures the corresponding output as active-low. However, polarity does not affect the override levels. Output polarity is selected with the POLx bits of the CWGxCON1. Auto-shutdown and steering options are unaffected by polarity.

FIGURE 28-5: CWG OUTPUT BLOCK DIAGRAM



28.5 Dead-Band Control

The dead-band control provides non-overlapping PWM signals to prevent shoot-through current in PWM switches. Dead-band operation is employed for Half-Bridge and Full-Bridge modes. The CWG contains two 6-bit dead-band counters. One is used for the rising edge of the input source control in Half-Bridge mode or for reverse dead-band Full-Bridge mode. The other is used for the falling edge of the input source control in Half-Bridge mode or for forward dead band in Full-Bridge mode.

Dead band is timed by counting CWG clock periods from zero up to the value in the rising or falling dead-band counter registers. See CWGxDBR and CWGxDBF registers, respectively.

28.5.1 DEAD-BAND FUNCTIONALITY IN HALF-BRIDGE MODE

In Half-Bridge mode, the dead-band counters dictate the delay between the falling edge of the normal output and the rising edge of the inverted output. This can be seen in [Figure 28-9](#).

28.5.2 DEAD-BAND FUNCTIONALITY IN FULL-BRIDGE MODE

In Full-Bridge mode, the dead-band counters are used when undergoing a direction change. The MODE<0> bit of the CWGxCON0 register can be set or cleared while the CWG is running, allowing for changes from Forward to Reverse mode. The CWGxA and CWGxC signals will change immediately upon the first rising input edge following a direction change, but the modulated signals (CWGxB or CWGxD, depending on the direction of the change) will experience a delay dictated by the dead-band counters. This is demonstrated in [Figure 28-3](#).

28.6 Rising Edge and Reverse Dead Band

CWGxDBR controls the rising edge dead-band time at the leading edge of CWGxA (Half-Bridge mode) or the leading edge of CWGxB (Full-Bridge mode). The CWGxDBR value is double-buffered. When EN = 0, the CWGxDBR register is loaded immediately when CWGxDBR is written. When EN = 1, then software must set the LD bit of the CWGxCON0 register, and the buffer will be loaded at the next falling edge of the CWG input signal. If the input source signal is not present for enough time for the count to be completed, no output will be seen on the respective output.

28.7 Falling Edge and Forward Dead Band

CWGxDBF controls the dead-band time at the leading edge of CWGxB (Half-Bridge mode) or the leading edge of CWGxD (Full-Bridge mode). The CWGxDBF value is double-buffered. When EN = 0, the CWGxDBF register is loaded immediately when CWGxDBF is written. When EN = 1 then software must set the LD bit of the CWGxCON0 register, and the buffer will be loaded at the next falling edge of the CWG input signal. If the input source signal is not present for enough time for the count to be completed, no output will be seen on the respective output.

Refer to [Figure 28.6](#) and [Figure 28-7](#) for examples.

FIGURE 28-6: DEAD-BAND OPERATION CWGXDBR = 0X01, CWGXDBF = 0X02

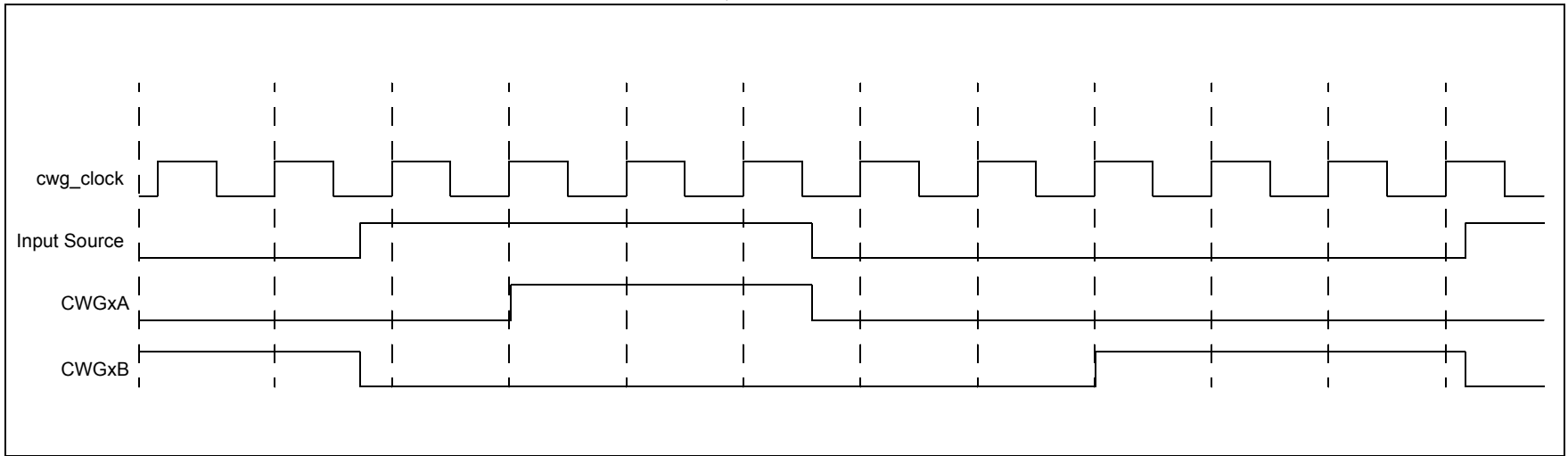
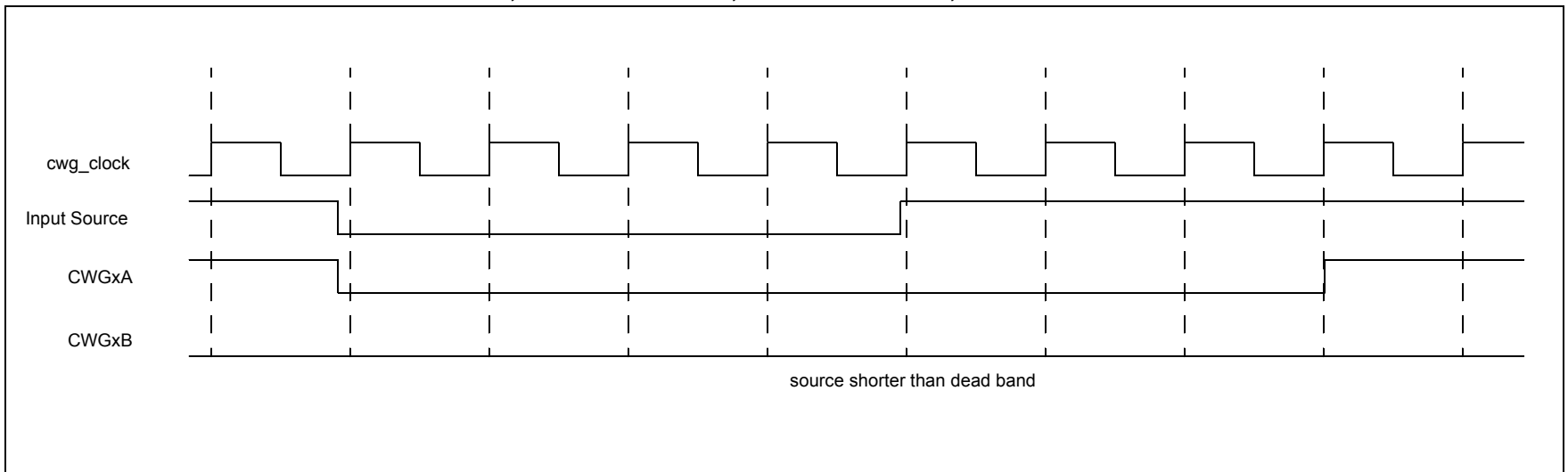


FIGURE 28-7: DEAD-BAND OPERATION, CWGXDBR = 0X03, CWGXDBF = 0X04, SOURCE SHORTER THAN DEAD BAND



28.8 Dead-Band Uncertainty

When the rising and falling edges of the input source are asynchronous to the CWG clock, it creates uncertainty in the dead-band time delay. The maximum uncertainty is equal to one CWG clock period. Refer to [Equation 28-1](#) for more details.

EQUATION 28-1: DEAD-BAND UNCERTAINTY

$$T_{DEADBAND_UNCERTAINTY} = \frac{1}{F_{cwg_clock}}$$

Example:
 $F_{CWG_CLOCK} = 16\text{ MHz}$

Therefore:

$$T_{DEADBAND_UNCERTAINTY} = \frac{1}{F_{cwg_clock}}$$

$$= \frac{1}{16\text{ MHz}}$$

$$= 62.5\text{ ns}$$

FIGURE 28-8: EXAMPLE OF PWM DIRECTION CHANGE

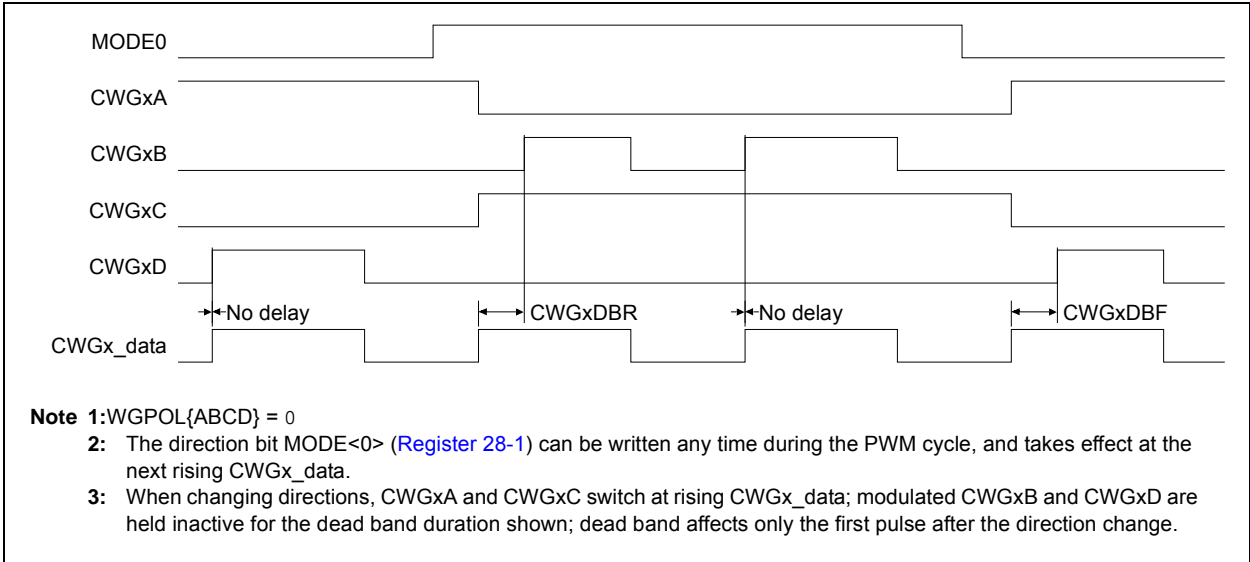
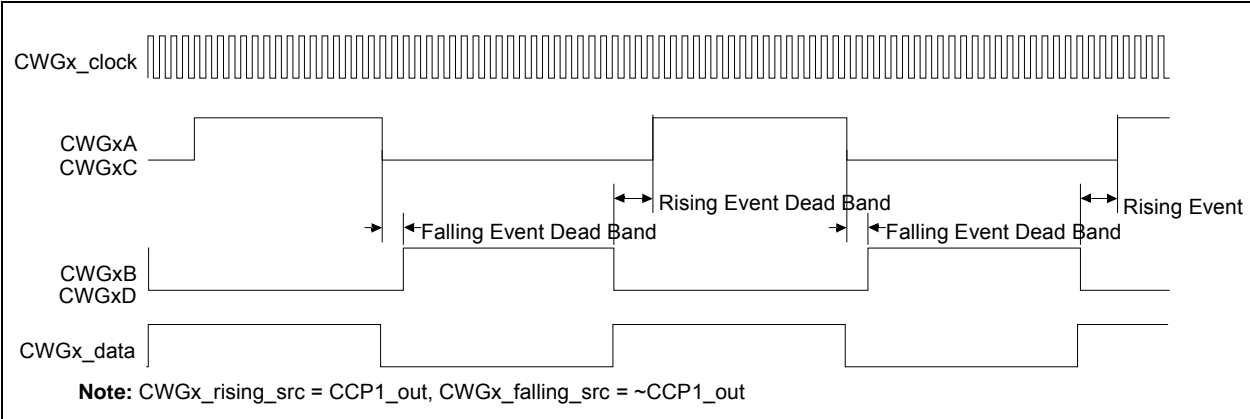


FIGURE 28-9: CWG HALF-BRIDGE MODE OPERATION



28.9 CWG Steering Mode

In Steering mode (MODE = 00x), the CWG allows any combination of the CWGxx pins to be the modulated signal. The same signal can be simultaneously available on multiple pins, or a fixed-value output can be presented.

When the respective STRx bit of CWGxOCON0 is '0', the corresponding pin is held at the level defined. When the respective STRx bit of CWGxOCON0 is '1', the pin is driven by the input data signal. The user can assign the input data signal to one, two, three, or all four output pins.

The POLx bits of the CWGxCON1 register control the signal polarity only when STRx = 1.

The CWG auto-shutdown operation also applies in Steering modes as described in [Section 28.10 "Auto-Shutdown"](#). An auto-shutdown event will only affect pins that have STRx = 1.

28.9.1 STEERING SYNCHRONIZATION

Changing the MODE bits allows for two modes of steering, synchronous and asynchronous.

When MODE = 000, the steering event is asynchronous and will happen at the end of the instruction that writes to STRx (that is, immediately). In this case, the output signal at the output pin may be an incomplete waveform. This can be useful for immediately removing a signal from the pin.

When MODE = 001, the steering update is synchronous and occurs at the beginning of the next rising edge of the input data signal. In this case, steering the output on/off will always produce a complete waveform.

[Figure 28-10](#) and [Figure 28-11](#) illustrate the timing of asynchronous and synchronous steering, respectively.

FIGURE 28-10: EXAMPLE OF STEERING EVENT AT END OF INSTRUCTION (MODE<2:0> = 000)

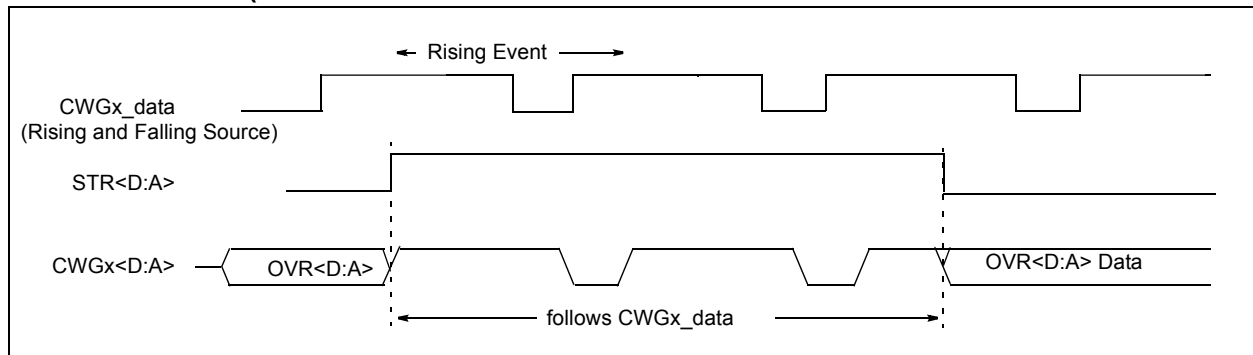
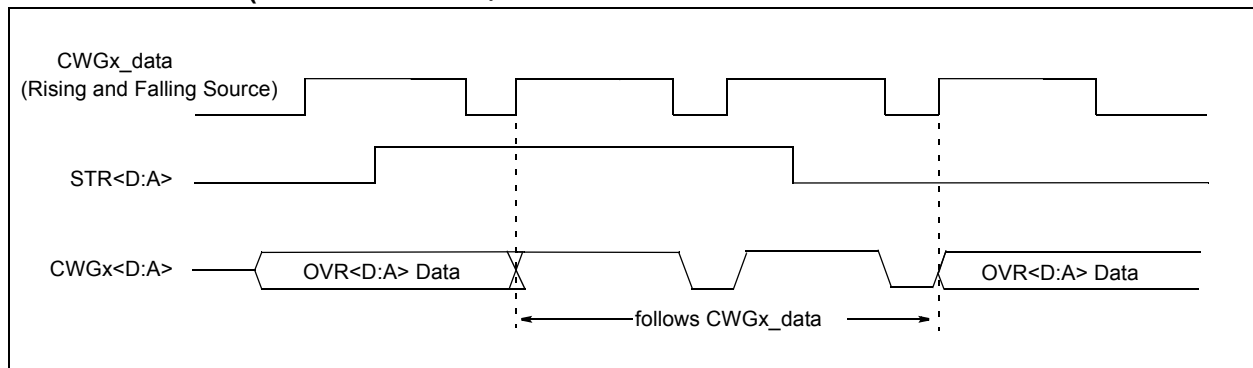


FIGURE 28-11: EXAMPLE OF STEERING EVENT AT BEGINNING OF INSTRUCTION (MODE<2:0> = 001)



28.10 Auto-Shutdown

Auto-shutdown is a method to immediately override the CWG output levels with specific overrides that allow for safe shutdown of the circuit. The shutdown state can be either cleared automatically or held until cleared by software. The auto-shutdown circuit is illustrated in [Figure 28-12](#).

28.10.1 SHUTDOWN

The shutdown state can be entered by either of the following two methods:

- Software generated
- External Input

28.10.1.1 Software Generated Shutdown

Setting the SHUTDOWN bit of the CWGxAS0 register will force the CWG into the shutdown state.

When the auto-restart is disabled, the shutdown state will persist as long as the SHUTDOWN bit is set.

When auto-restart is enabled, the SHUTDOWN bit will clear automatically and resume operation on the next rising edge event.

28.10.2 EXTERNAL INPUT SOURCE

External shutdown inputs provide the fastest way to safely suspend CWG operation in the event of a Fault condition. When any of the selected shutdown inputs goes active, the CWG outputs will immediately go to the selected override levels without software delay. Several input sources can be selected to cause a shutdown condition. All input sources are active-low. The sources are:

- Comparator C1_OUT_sync
- Comparator C2_OUT_sync
- Timer2 – TMR2_postscaled
- Timer4 – TMR4_postscaled
- Timer6 – TMR6_postscaled
- CWGxIN input pin

Shutdown inputs are selected using the CWGxAS1 register ([Register 28-6](#)).

Note: Shutdown inputs are level sensitive, not edge sensitive. The shutdown state cannot be cleared, except by disabling auto-shutdown, as long as the shutdown input level persists.

28.11 Operation During Sleep

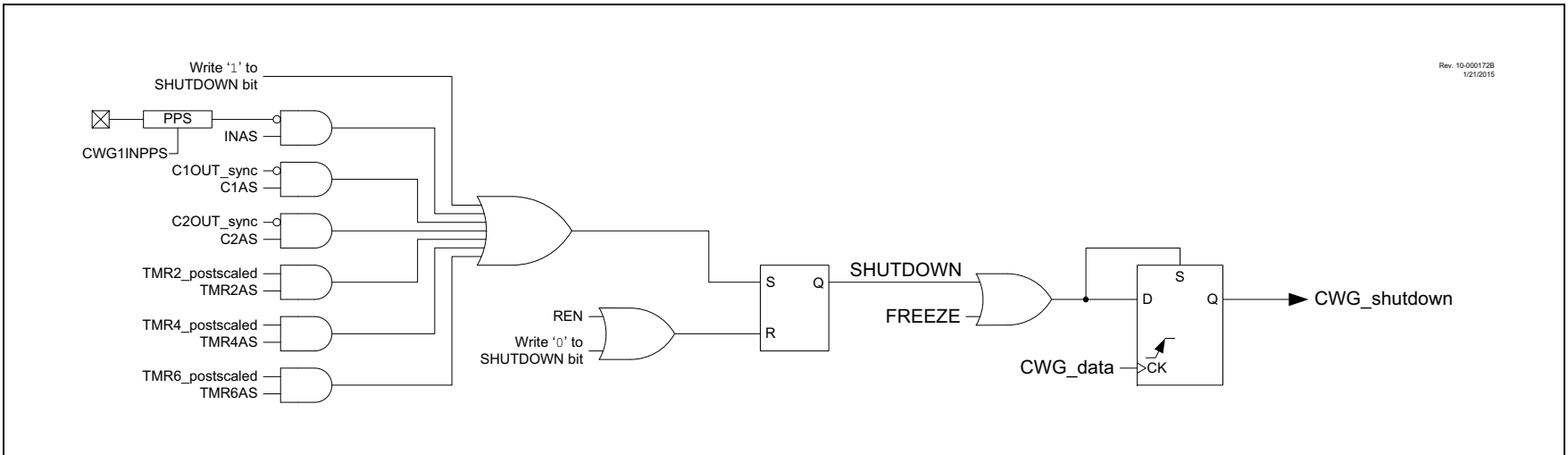
The CWG module operates independently from the system clock and will continue to run during Sleep, provided that the clock and input sources selected remain active.

The HFINTOSC remains active during Sleep when all the following conditions are met:

- CWG module is enabled
- Input source is active
- HFINTOSC is selected as the clock source, regardless of the system clock source selected.

In other words, if the HFINTOSC is simultaneously selected as the system clock and the CWG clock source, when the CWG is enabled and the input source is active, then the CPU will go idle during Sleep, but the HFINTOSC will remain active and the CWG will continue to operate. This will have a direct effect on the Sleep mode current.

FIGURE 28-12: CWG SHUTDOWN BLOCK DIAGRAM



28.12 Configuring the CWG

The following steps illustrate how to properly configure the CWG.

1. Ensure that the TRIS control bits corresponding to the desired CWG pins for your application are set so that the pins are configured as inputs.
2. Clear the EN bit, if not already cleared.
3. Set desired mode of operation with the MODE bits.
4. Set desired dead-band times, if applicable to mode, with the CWGxDBR and CWGxDBF registers.
5. Setup the following controls in the CWGxAS0 and CWGxAS1 registers.
 - a. Select the desired shutdown source.
 - b. Select both output overrides to the desired levels (this is necessary even if not using auto-shutdown because start-up will be from a shutdown state).
 - c. Set which pins will be affected by auto-shutdown with the CWGxAS1 register.
 - d. Set the SHUTDOWN bit and clear the REN bit.
6. Select the desired input source using the CWGxISM register.
7. Configure the following controls.
 - a. Select desired clock source using the CWGxCLKCON register.
 - b. Select the desired output polarities using the CWGxCON1 register.
 - c. Set the output enables for the desired outputs.
8. Set the EN bit.
9. Clear TRIS control bits corresponding to the desired output pins to configure these pins as outputs.
10. If auto-restart is to be used, set the REN bit and the SHUTDOWN bit will be cleared automatically. Otherwise, clear the SHUTDOWN bit to start the CWG.

28.12.1 PIN OVERRIDE LEVELS

The levels driven to the output pins, while the shutdown input is true, are controlled by the LSB and LSAC bits of the CWGxAS0 register. LSB<1:0> controls the CWGxB and D override levels and LSAC<1:0> controls the CWGxA and C override levels. The control bit logic level corresponds to the output logic drive level while in the shutdown state. The polarity control does not affect the override level.

28.12.2 AUTO-SHUTDOWN RESTART

After an auto-shutdown event has occurred, there are two ways to resume operation:

- Software controlled
- Auto-restart

The restart method is selected with the REN bit of the CWGxAS0 register. Waveforms of software controlled and automatic restarts are shown in [Figure 28-13](#) and [Figure 28-14](#).

28.12.2.1 Software Controlled Restart

When the REN bit of the CWGxAS0 register is cleared, the CWG must be restarted after an auto-shutdown event by software. Clearing the shutdown state requires all selected shutdown inputs to be low, otherwise the SHUTDOWN bit will remain set. The overrides will remain in effect until the first rising edge event after the SHUTDOWN bit is cleared. The CWG will then resume operation.

28.12.2.2 Auto-Restart

When the REN bit of the CWGxAS0 register is set, the CWG will restart from the auto-shutdown state automatically. The SHUTDOWN bit will clear automatically when all shutdown sources go low. The overrides will remain in effect until the first rising edge event after the SHUTDOWN bit is cleared. The CWG will then resume operation.

FIGURE 28-13: SHUTDOWN FUNCTIONALITY, AUTO-RESTART DISABLED (REN = 0, LSAC = 01, LSBD = 01)

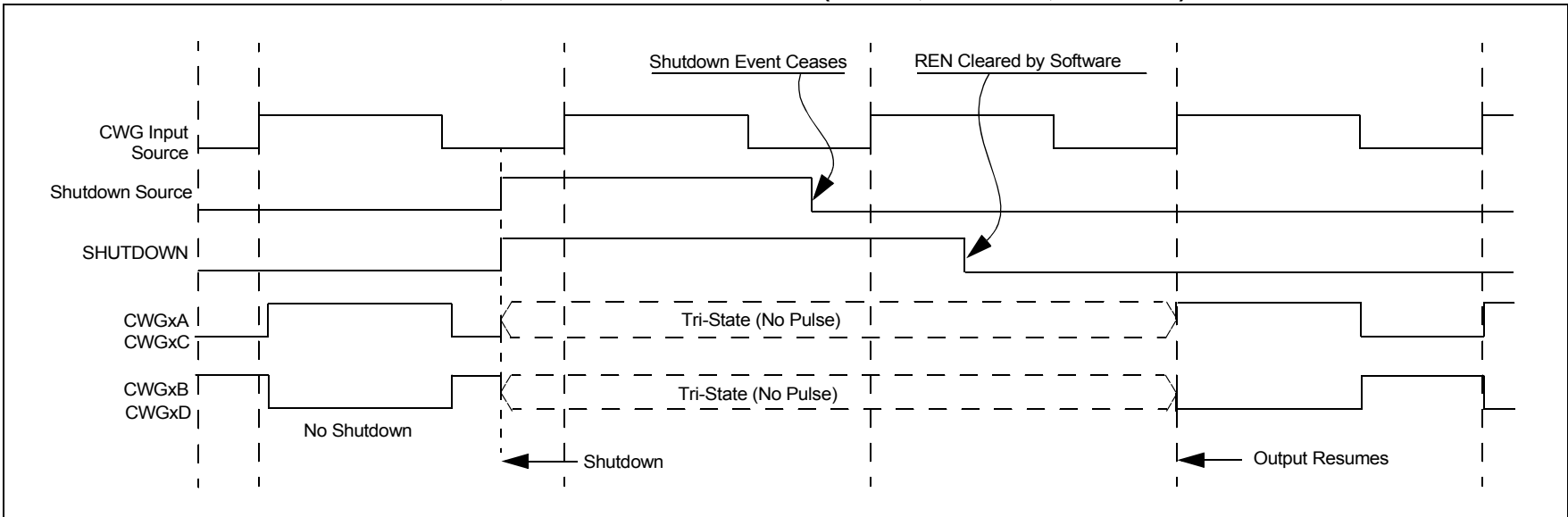
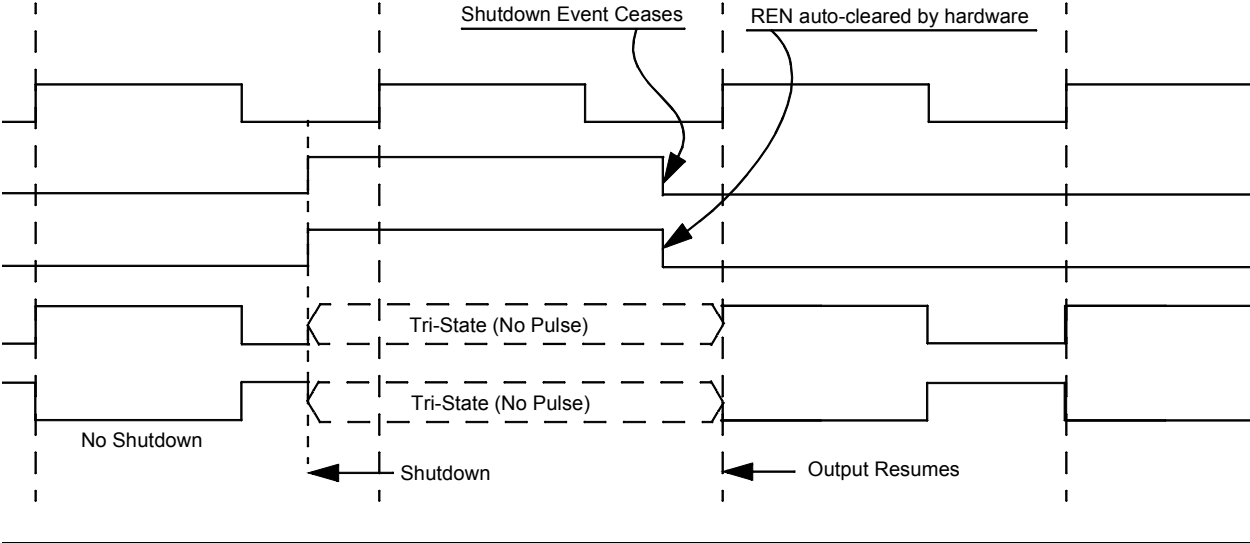


FIGURE 28-14: SHUTDOWN FUNCTIONALITY, AUTO-RESTART ENABLED (REN = 1, LSAC = 01, LSBD = 01)



28.13 Register Definitions: CWG Control

REGISTER 28-1: CWGxCON0: CWGx CONTROL REGISTER 0

| | | | | | | | |
|---------|-------------------|-----|-----|-----|-----------|---------|---------|
| R/W-0/0 | R/W/HC-0/0 | U-0 | U-0 | U-0 | R/W-0/0 | R/W-0/0 | R/W-0/0 |
| EN | LD ⁽¹⁾ | — | — | — | MODE<2:0> | | |
| 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

-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 **EN:** CWGx Enable bit
 1 = Module is enabled
 0 = Module is disabled

- bit 6 **LD:** CWGx Load Buffer bits⁽¹⁾
 1 = Buffers to be loaded on the next rising/falling event
 0 = Buffers not loaded

- bit 5-3 **Unimplemented:** Read as '0'

- bit 2-0 **MODE<2:0>:** CWGx Mode bits
 111 = Reserved
 110 = Reserved
 101 = CWG outputs operate in Push-Pull mode
 100 = CWG outputs operate in Half-Bridge mode
 011 = CWG outputs operate in Reverse Full-Bridge mode
 010 = CWG outputs operate in Forward Full-Bridge mode
 001 = CWG outputs operate in Synchronous Steering mode
 000 = CWG outputs operate in Steering mode

Note 1: This bit can only be set after EN = 1 and cannot be set in the same instruction that EN is set.

REGISTER 28-2: CWGxCON1: CWGx CONTROL REGISTER 1

| | | | | | | | |
|-------|-----|-----|-----|---------|---------|---------|---------|
| U-0 | U-0 | R-x | U-0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 |
| — | — | IN | — | POLD | POLC | POLB | POLA |
| 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 7-6 | Unimplemented: Read as '0' |
| bit 5 | IN: CWG Input Value |
| bit 4 | Unimplemented: Read as '0' |
| bit 3 | POLD: CWGxD Output Polarity bit 1 = Signal output is inverted polarity 0 = Signal output is normal polarity |
| bit 2 | POLC: CWGxC Output Polarity bit 1 = Signal output is inverted polarity 0 = Signal output is normal polarity |
| bit 1 | POLB: CWGxB Output Polarity bit 1 = Signal output is inverted polarity 0 = Signal output is normal polarity |
| bit 0 | POLA: CWGxA Output Polarity bit 1 = Signal output is inverted polarity 0 = Signal output is normal polarity |

REGISTER 28-3: CWGxDBR: CWGx RISING DEAD-BAND COUNTER 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 |
|-------|-----|----------|---------|---------|---------|---------|---------|
| — | — | DBR<5:0> | | | | | |
| 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 7-6 **Unimplemented:** Read as '0'
bit 5-0 **DBR<5:0>:** Rising Event Dead-Band Value for Counter bits

REGISTER 28-4: CWGxDBF: CWGx FALLING DEAD-BAND COUNTER 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 |
|-------|-----|----------|---------|---------|---------|---------|---------|
| — | — | DBF<5:0> | | | | | |
| 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 7-6 **Unimplemented:** Read as '0'
bit 5-0 **DBF<5:0>:** Falling Event Dead-Band Value for Counter bits

REGISTER 28-5: CWGxAS0: CWGx AUTO-SHUTDOWN CONTROL REGISTER 0

| | | | | | | | |
|---------------------------|---------|-----------|---------|-----------|---------|-------|-----|
| R/W/HS-0/0 | R/W-0/0 | R/W-0/0 | R/W-1/1 | R/W-0/0 | R/W-1/1 | U-0 | U-0 |
| SHUTDOWN ^(1,2) | REN | LSBD<1:0> | | LSAC<1:0> | | — | — |
| 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

-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 **SHUTDOWN:** Auto-Shutdown Event Status bit^(1, 2)
 1 = An Auto-Shutdown state is in effect
 0 = No Auto-shutdown event has occurred
- bit 6 **REN:** Auto-Restart Enable bit
 1 = Auto-restart enabled
 0 = Auto-restart disabled
- bit 5-4 **LSBD<1:0>:** CWGxB and CWGxD Auto-Shutdown State Control bits
 11 = A logic '1' is placed on CWGxB/D when an auto-shutdown event is present
 10 = A logic '0' is placed on CWGxB/D when an auto-shutdown event is present
 01 = Pin is tri-stated on CWGxB/D when an auto-shutdown event is present
 00 = The inactive state of the pin, including polarity, is placed on CWGxB/D after the required dead-band interval
- bit 3-2 **LSAC<1:0>:** CWGxA and CWGxC Auto-Shutdown State Control bits
 11 = A logic '1' is placed on CWGxA/C when an auto-shutdown event is present
 10 = A logic '0' is placed on CWGxA/C when an auto-shutdown event is present
 01 = Pin is tri-stated on CWGxA/C when an auto-shutdown event is present
 00 = The inactive state of the pin, including polarity, is placed on CWGxA/C after the required dead-band interval
- bit 1-0 **Unimplemented:** Read as '0'

- Note 1:** This bit may be written while EN = 0 (CWGxCON0 register) to place the outputs into the shutdown configuration.
- 2:** The outputs will remain in auto-shutdown state until the next rising edge of the input signal after this bit is cleared.

REGISTER 28-6: CWGxAS1: CWGx AUTO-SHUTDOWN CONTROL REGISTER 1

| | | | | | | | |
|-------|---------|---------|---------|-----|---------------------|---------|---------|
| U-1 | R/W-0/0 | R/W-0/0 | R/W-0/0 | U-1 | R/W-0/0 | R/W-0/0 | R/W-0/0 |
| — | TMR6AS | TMR4AS | TMR2AS | — | C2AS ⁽¹⁾ | C1AS | INAS |
| 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 | Unimplemented: Read as '1' |
| bit 6 | TMR6AS: TMR6 Postscale Output bit 1 = TMR6 postscale shut-down is enabled 0 = TMR6 postscale shut-down is disabled |
| bit 5 | TMR4AS: TMR4 Postscale Output bit 1 = TMR4 postscale shut-down is enabled 0 = TMR4 postscale shut-down is disabled |
| bit 4 | TMR2AS: TMR2 Postscale Output bit 1 = TMR2 postscale shut-down is enabled 0 = TMR2 postscale shut-down is disabled |
| bit 3 | Unimplemented: Read as '1' |
| bit 2 | C2AS: Comparator C2 Output bit 1 = C2 output shut-down is enabled 0 = C2 output shut-down is disabled |
| bit 1 | C1AS: Comparator C1 Output bit 1 = C1 output shut-down is enabled 0 = C1 output shut-down is disabled |
| bit 0 | INAS: CWGx Input Pin bit 1 = CWGxIN input pin shut-down is enabled 0 = CWGxIN input pin shut-down is disabled |

REGISTER 28-7: CWGxOCON0: CWGx STEERING CONTROL REGISTER 0⁽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 |
| OVRD | OVRC | OVRB | OVRA | STRD ⁽²⁾ | STRC ⁽²⁾ | STRB ⁽²⁾ | STRA ⁽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 | q = Value depends on condition |

| | |
|-------|--|
| bit 7 | OVRD: Steering Data D bit |
| bit 6 | OVRC: Steering Data C bit |
| bit 5 | OVRB: Steering Data B bit |
| bit 4 | OVRA: Steering Data A bit |
| bit 3 | STRD: Steering Enable D bit ⁽²⁾ 1 = CWGxD output has the CWGx_data waveform with polarity control from POLD bit 0 = CWGxD output is assigned the value of OVRD bit |
| bit 2 | STRC: Steering Enable C bit ⁽²⁾ 1 = CWGxC output has the CWGx_data waveform with polarity control from POLC bit 0 = CWGxC output is assigned the value of OVRC bit |
| bit 1 | STRB: Steering Enable B bit ⁽²⁾ 1 = CWGxB output has the CWGx_data waveform with polarity control from POLB bit 0 = CWGxB output is assigned the value of OVRB bit |
| bit 0 | STRA: Steering Enable A bit ⁽²⁾ 1 = CWGxA output has the CWGx_data waveform with polarity control from POLA bit 0 = CWGxA output is assigned the value of OVRA bit |

Note 1: The bits in this register apply only when MODE<2:0> = 00x.

2: This bit is effectively double-buffered when MODE<2:0> = 001.

REGISTER 28-8: CWGxCLKCON: CWGx CLOCK SELECTION CONTROL REGISTER

| | | | | | | | |
|-------|-----|-----|-----|-----|-----|-----|---------|
| U-0 | U-0 | U-0 | U-0 | U-0 | U-0 | U-0 | R/W-0/0 |
| — | — | — | — | — | — | — | CS |
| 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 7-1 **Unimplemented:** Read as '0'
bit 0 **CS:** CWGx Clock Selection bit
 1 = HFINTOSC 16 MHz is selected
 0 = FOSC is selected

REGISTER 28-9: CWGxISM: CWGx INPUT SELECTION REGISTER

| | | | | | | | |
|-------|-----|-----|-----|---------|---------|---------|---------|
| U-0 | U-0 | U-0 | U-0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 |
| — | — | — | — | IS<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 | q = Value depends on condition |

bit 7-4 **Unimplemented:** Read as '0'
bit 3-0 **IS<3:0>:** CWGx Input Selection bits
 1111 = Reserved. No channel connected.
 .
 .
 .
 1011 = Reserved. No channel connected.
 1010 = PWM4_out
 1001 = PWM3_out
 1000 = LC4_out
 0111 = LC3_out
 0110 = LC2_out
 0101 = LC1_out
 0100 = CCP2_out
 0011 = CCP1_out
 0010 = C2_OUT_sync
 0001 = C1_OUT_sync
 0000 = CWGxIN pin

TABLE 28-2: SUMMARY OF REGISTERS ASSOCIATED WITH CWG

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Register on Page |
|------------|----------|--------|-----------|--------|-----------|-----------|-------|-------|---------------------|
| CWG1AS0 | SHUTDOWN | REN | LSBD<1:0> | | LSAC<1:0> | | — | — | 386 |
| CWG1AS1 | — | TMR6AS | TMR4AS | TMR2AS | — | C2AS | C1AS | INAS | 387 |
| CWG1CLKCON | — | — | — | — | — | — | — | CS | 389 |
| CWG1CON0 | EN | LD | — | — | — | MODE<2:0> | | | 388 |
| CWG1CON1 | — | — | IN | — | POLD | POLC | POLB | POLA | 384 |
| CWG1DBF | — | — | DBF<5:0> | | | | | | 385 |
| CWG1DBR | — | — | DBR<5:0> | | | | | | 385 |
| CWG1ISM | — | — | — | — | IS<3:0> | | | | 389 |
| CWG1OCON0 | OVRD | OVRC | OVRB | OVRA | STRD | STRC | STRB | STRA | 388 |

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

29.0 SIGNAL MEASUREMENT TIMER (SMT)

The SMT is a 24-bit counter with advanced clock and gating logic, which can be configured for measuring a variety of digital signal parameters such as pulse width, frequency and duty cycle, and the time difference between edges on two signals.

Features of the SMT include:

- 24-bit timer/counter
 - Four 8-bit registers (SMTxTMRL/H/U)
 - Readable and writable
 - Optional 16-bit operating mode
- Two 24-bit measurement capture registers
- One 24-bit period match register
- Multi-mode operation, including relative timing measurement
- Interrupt on period match
- Multiple clock, gate and signal sources
- Interrupt on acquisition complete
- Ability to read current input values

| |
|---|
| <p>Note: These devices implement two SMT modules. All references to SMTx apply to SMT1 and SMT2.</p> |
|---|

FIGURE 29-1: SMT BLOCK DIAGRAM

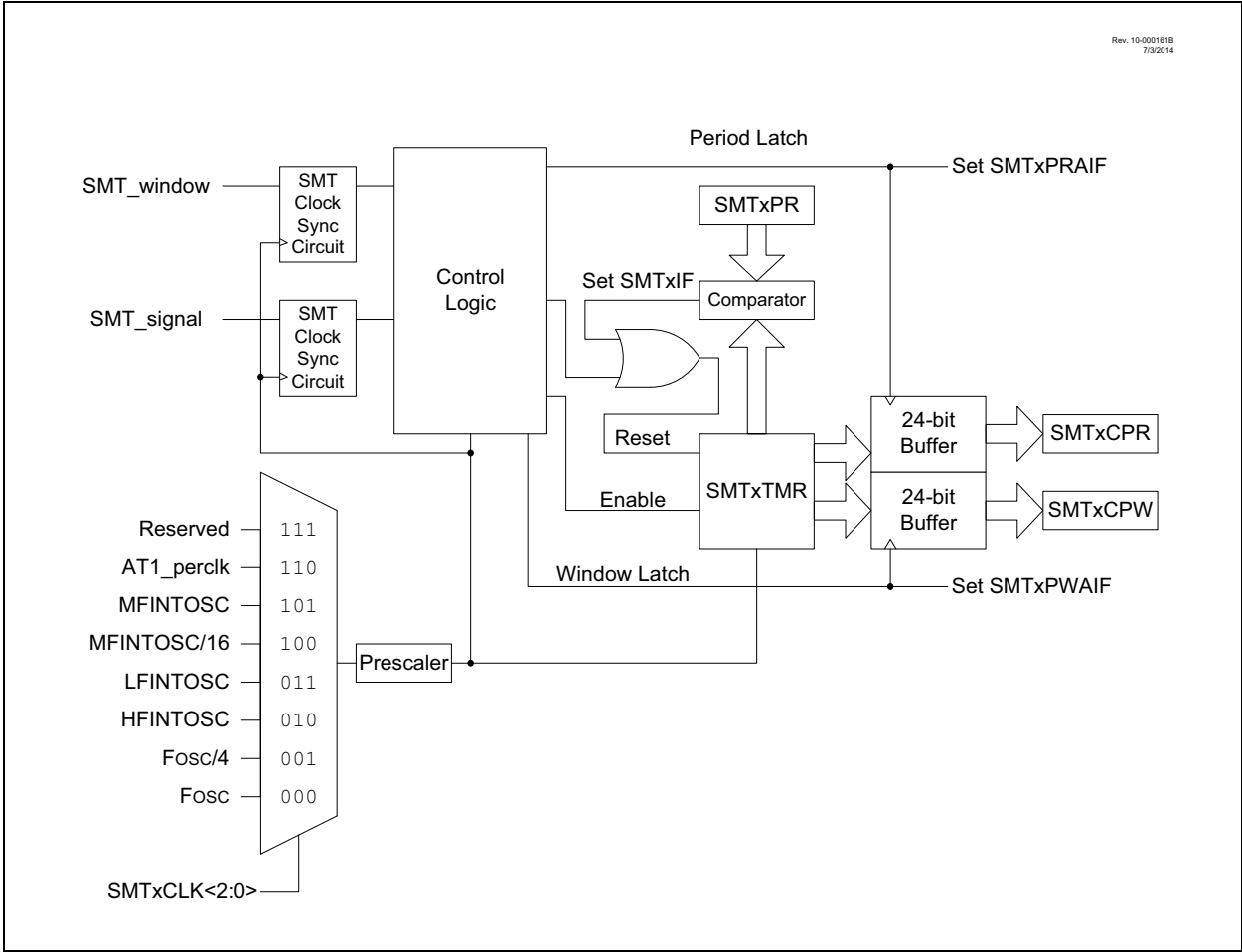
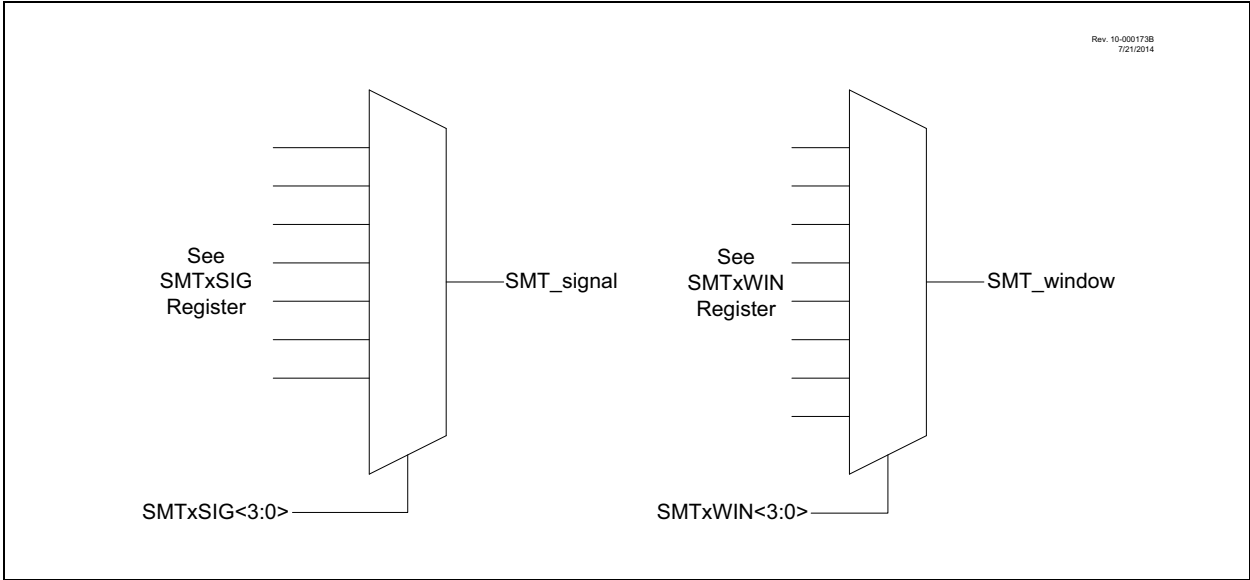


FIGURE 29-2: SMT SIGNAL AND WINDOW BLOCK DIAGRAM



29.1 SMT Operation

The core of the module is the 24-bit counter, SMTxTMR combined with a complex data acquisition front-end. Depending on the mode of operation selected, the SMT can perform a variety of measurements summarized in [Table 29-1](#).

29.1.1 CLOCK SOURCES

Clock sources available to the SMT include:

- FOSC
- FOSC/4
- HFINTOSC 16 MHz
- LFINTOSC
- MFINTOSC 31.25 kHz

The SMT clock source is selected by configuring the CSEL<2:0> bits in the SMTxCLK register. The clock source can also be prescaled using the PS<1:0> bits of the SMTxCON0 register. The prescaled clock source is used to clock both the counter and any synchronization logic used by the module.

29.1.2 PERIOD MATCH INTERRUPT

Similar to other timers, the SMT triggers an interrupt when SMTxTMR rolls over to '0'. This happens when SMTxTMR = SMTxPR, regardless of mode. Hence, in any mode that relies on an external signal or a window to reset the timer, proper operation requires that SMTxPR be set to a period larger than that of the expected signal or window.

29.2 Basic Timer Function Registers

The SMTxTMR time base and the SMTxCPW/SMTxPR/SMTxCPR buffer registers serve several functions and can be manually updated using software.

29.2.1 TIME BASE

The SMTxTMR is the 24-bit counter that is the center of the SMT. It is used as the basic counter/timer for measurement in each of the modes of the SMT. It can be reset to a value of 24'h00_0000 by setting the RST bit of the SMTxSTAT register. It can be written to and read from software, but it is not guarded for atomic access, therefore reads and writes to the SMTxTMR should only be made when the GO = 0, or the software should have other measures to ensure integrity of SMTxTMR reads/writes.

29.2.2 PULSE WIDTH LATCH REGISTERS

The SMTxCPW registers are the 24-bit SMT pulse width latch. They are used to latch in the value of the SMTxTMR when triggered by various signals, which are determined by the mode the SMT is currently in. The SMTxCPW registers can also be updated with the current value of the SMTxTMR value by setting the CPWUP bit of the SMTxSTAT register.

29.2.3 PERIOD LATCH REGISTERS

The SMTxCPR registers are the 24-bit SMT period latch. They are used to latch in other values of the SMTxTMR when triggered by various other signals, which are determined by the mode the SMT is currently in.

The SMTxCPR registers can also be updated with the current value of the SMTxTMR value by setting the CPRUP bit in the SMTxSTAT register.

29.3 Halt Operation

The counter can be prevented from rolling-over using the STP bit in the SMTxCON0 register. When halting is enabled, the period match interrupt persists until the SMTxTMR is reset (either by a manual reset, [Section 29.2.1 "Time Base"](#)) or by clearing the SMTxGO bit of the SMTxCON1 register and writing the SMTxTMR values in software.

29.4 Polarity Control

The three input signals for the SMT have polarity control to determine whether or not they are active high/positive edge or active low/negative edge signals.

The following bits apply to Polarity Control:

- WSEL bit (Window Polarity)
- SSEL bit (Signal Polarity)
- CSEL bit (Clock Polarity)

These bits are located in the SMTxCON0 register.

29.5 Status Information

The SMT provides input status information for the user without requiring the need to deal with the polarity of the incoming signals.

29.5.1 WINDOW STATUS

Window status is determined by the WS bit of the SMTxSTAT register. This bit is only used in Windowed Measure, Gated Counter and Gated Window Measure modes, and is only valid when TS = 1, and will be delayed in time by synchronizer delays in non-Counter modes.

29.5.2 SIGNAL STATUS

Signal status is determined by the AS bit of the SMTxSTAT register. This bit is used in all modes except Window Measure, Time of Flight and Capture modes, and is only valid when TS = 1, and will be delayed in time by synchronizer delays in non-Counter modes.

29.5.3 GO STATUS

Timer run status is determined by the TS bit of the SMTxSTAT register, and will be delayed in time by synchronizer delays in non-Counter modes.

29.6 Modes of Operation

The modes of operation are summarized in [Table 29-1](#). The following sections provide detailed descriptions, examples of how the modes can be used. Note that all waveforms assume WPOL/SPOL/CPOL = 0. When WPOL/SPOL/CPOL = 1, all SMTSIGx, SMTWINx and SMT clock signals will have a polarity opposite to that indicated. For all modes, the REPEAT bit controls whether the acquisition is repeated or single. When REPEAT = 0 (Single Acquisition mode), the timer will stop incrementing and the SMTxGO bit will be reset upon the completion of an acquisition. Otherwise, the timer will continue and allow for continued acquisitions to overwrite the previous ones until the timer is stopped in software.

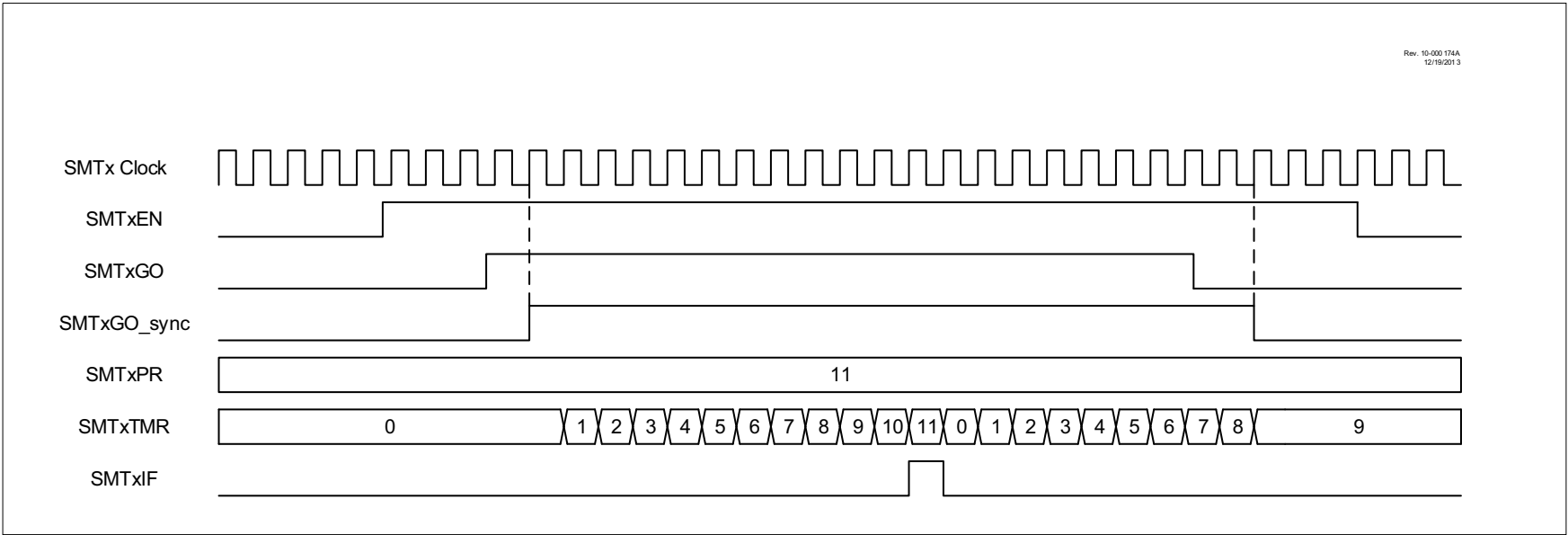
29.6.1 TIMER MODE

Timer mode is the simplest mode of operation where the SMTxTMR is used as a 16/24-bit timer. No data acquisition takes place in this mode. The timer increments as long as the SMTxGO bit has been set by software. No SMT window or SMT signal events affect the SMTxGO bit. Everything is synchronized to the SMT clock source. When the timer experiences a period match (SMTxTMR = SMTxPR), SMTxTMR is reset and the period match interrupt trips. See [Figure 29-3](#).

TABLE 29-1: MODES OF OPERATION

| MODE | Mode of Operation | Synchronous Operation | Reference |
|-------------|-----------------------------------|-----------------------|--|
| 0000 | Timer | Yes | Section 29.6.1 “Timer Mode” |
| 0001 | Gated Timer | Yes | Section 29.6.2 “Gated Timer Mode” |
| 0010 | Period and Duty Cycle Acquisition | Yes | Section 29.6.3 “Period and Duty-Cycle Mode” |
| 0011 | High and Low Time Measurement | Yes | Section 29.6.4 “High and Low Measure Mode” |
| 0100 | Windowed Measurement | Yes | Section 29.6.5 “Windowed Measure Mode” |
| 0101 | Gated Windowed Measurement | Yes | Section 29.6.6 “Gated Window Measure Mode” |
| 0110 | Time of Flight | Yes | Section 29.6.7 “Time of Flight Measure Mode” |
| 0111 | Capture | Yes | Section 29.6.8 “Capture Mode” |
| 1000 | Counter | No | Section 29.6.9 “Counter Mode” |
| 1001 | Gated Counter | No | Section 29.6.10 “Gated Counter Mode” |
| 1010 | Windowed Counter | No | Section 29.6.11 “Windowed Counter Mode” |
| 1011 - 1111 | Reserved | — | — |

FIGURE 29-3: TIMER MODE TIMING DIAGRAM



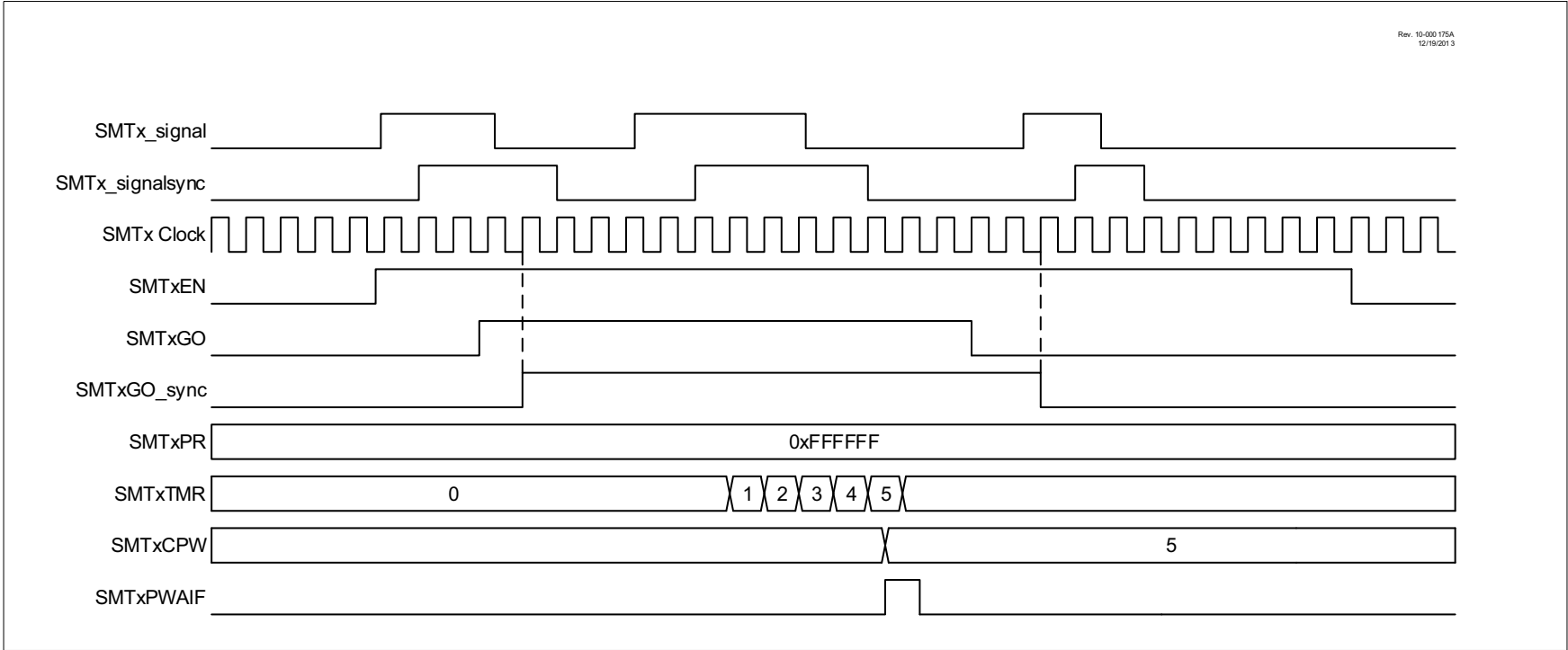
29.6.2 GATED TIMER MODE

Gated Timer mode uses the SMTSIGx input to control whether or not the SMTxTMR will increment. Upon a falling edge of the external signal, the SMTxCPW register will update to the current value of the SMTxTMR. Example waveforms for both repeated and single acquisitions are provided in [Figure 29-4](#) and [Figure 29-5](#).

FIGURE 29-4: GATED TIMER MODE REPEAT ACQUISITION TIMING DIAGRAM



FIGURE 29-5: GATED TIMER MODE SINGLE ACQUISITION TIMING DIAGRAM



29.6.3 PERIOD AND DUTY-CYCLE MODE

In Duty-Cycle mode, either the duty cycle or period (depending on polarity) of the SMTx_signal can be acquired relative to the SMT clock. The CPW register is updated on a falling edge of the signal, and the CPR register is updated on a rising edge of the signal, along with the SMTxTMR resetting to 0x0001. In addition, the SMTxGO bit is reset on a rising edge when the SMT is in Single Acquisition mode. See [Figure 29-6](#) and [Figure 29-7](#).

FIGURE 29-6: PERIOD AND DUTY-CYCLE REPEAT ACQUISITION MODE TIMING DIAGRAM

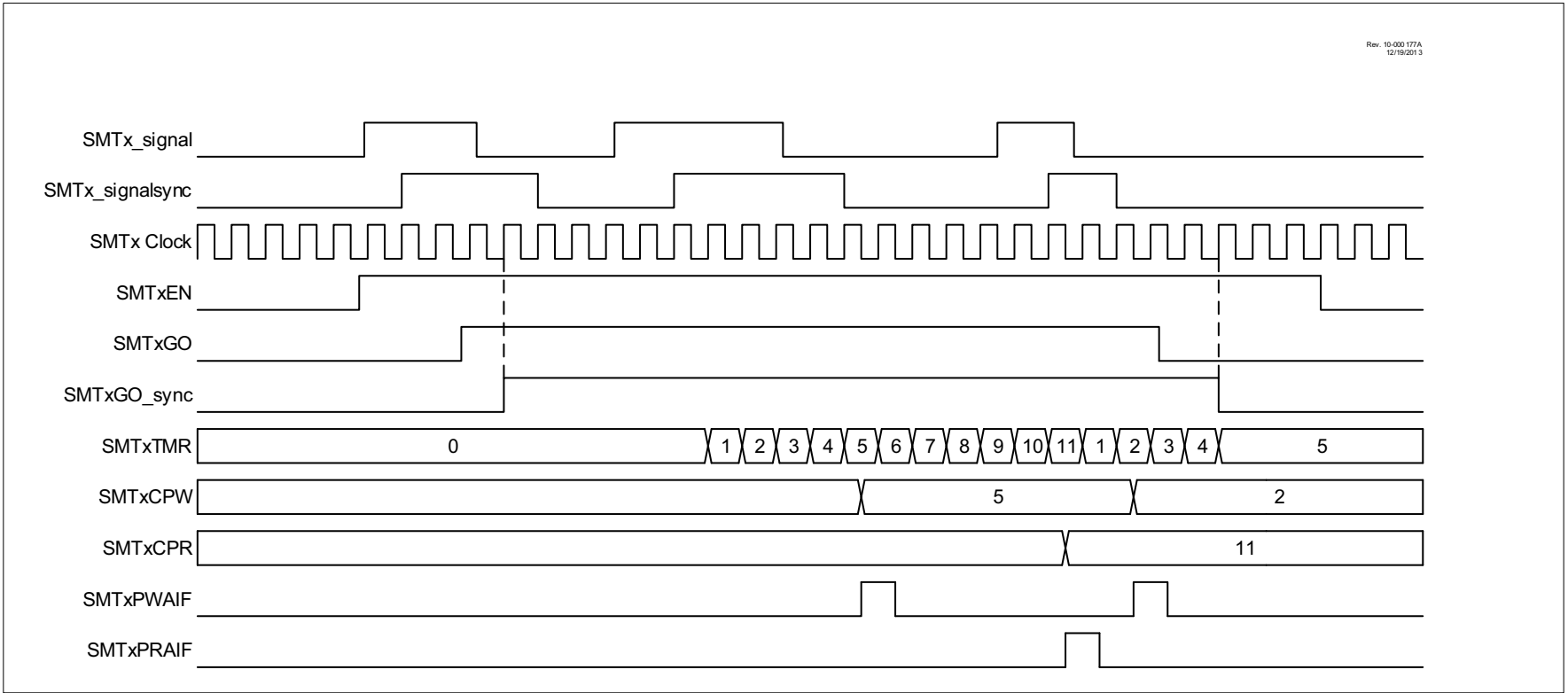
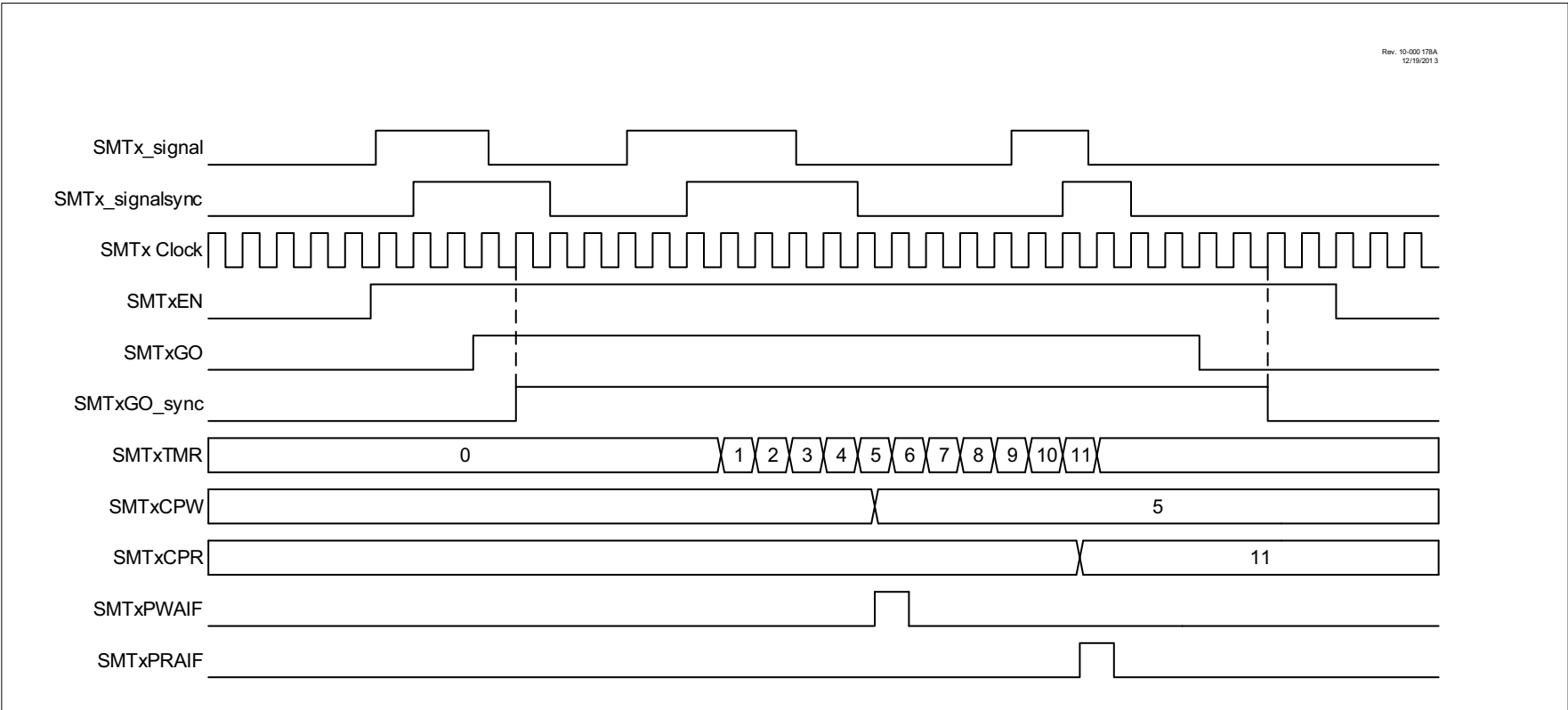


FIGURE 29-7: PERIOD AND DUTY-CYCLE SINGLE ACQUISITION TIMING DIAGRAM



29.6.4 HIGH AND LOW MEASURE MODE

This mode measures the high and low pulse time of the SMTSIGx relative to the SMT clock. It begins incrementing the SMTxTMR on a rising edge on the SMTSIGx input, then updates the SMTxCPW register with the value and resets the SMTxTMR on a falling edge, starting to increment again. Upon observing another rising edge, it updates the SMTxCPR register with its current value and once again resets the SMTxTMR value and begins incrementing again. See [Figure 29-8](#) and [Figure 29-9](#).

FIGURE 29-8: HIGH AND LOW MEASURE MODE REPEAT ACQUISITION TIMING DIAGRAM

Rev. 10-000 180A
12/19/2013

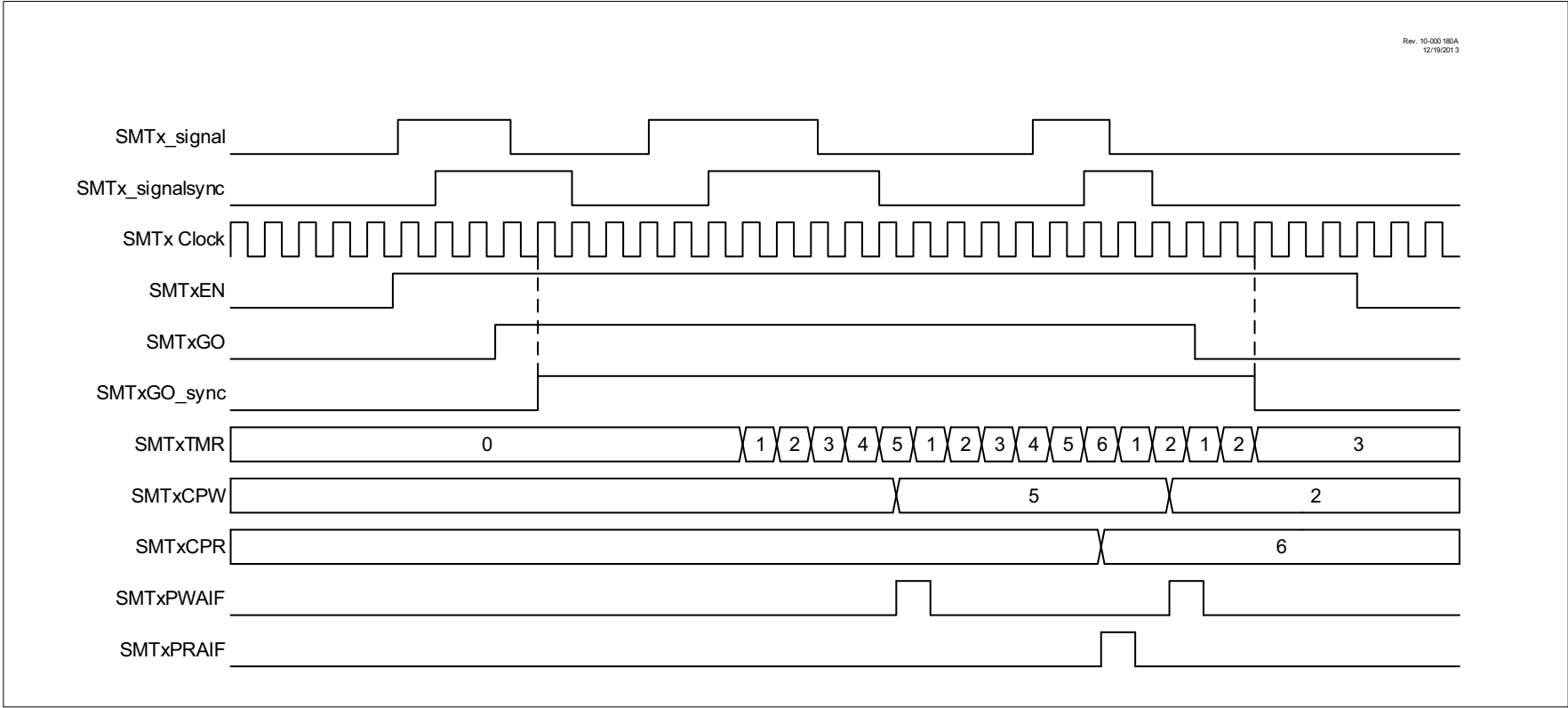
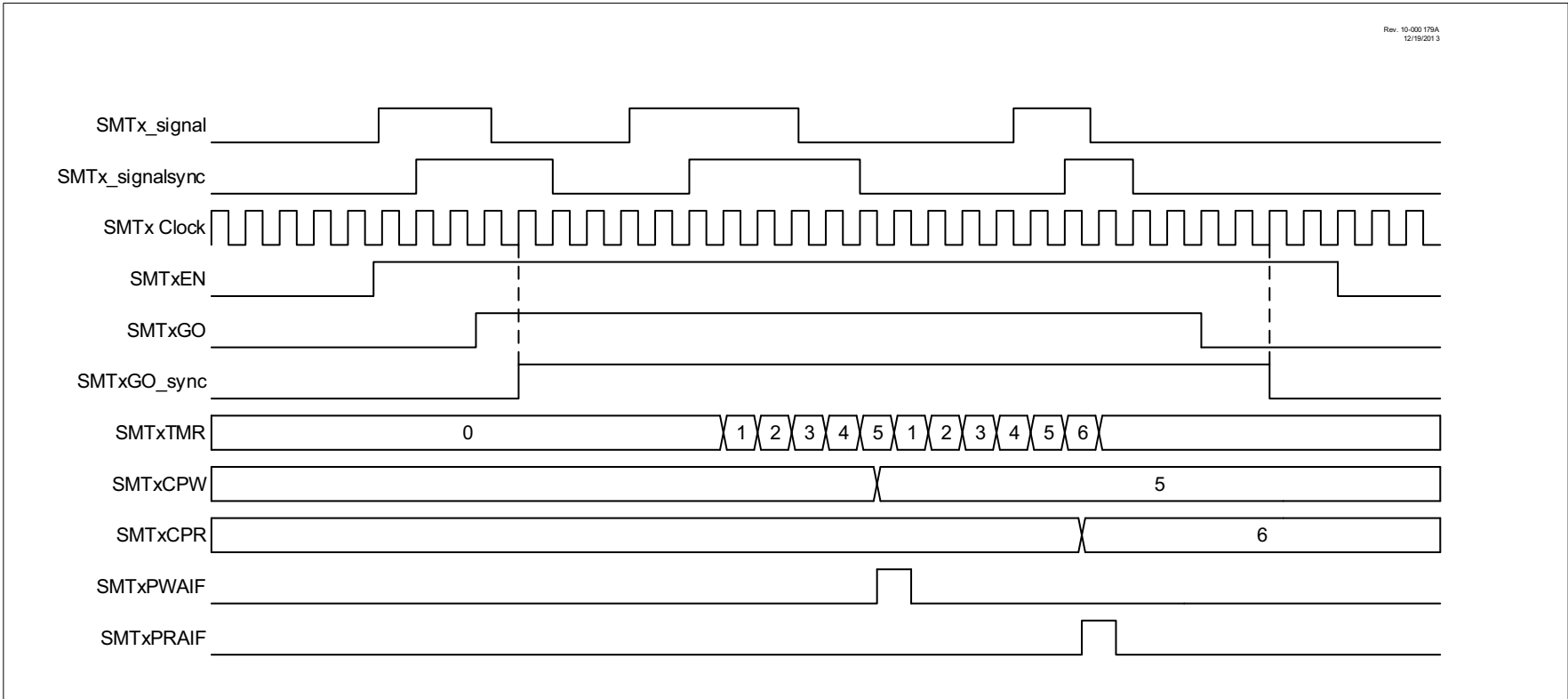


FIGURE 29-9: HIGH AND LOW MEASURE MODE SINGLE ACQUISITION TIMING DIAGRAM

Rev. 10-000 179A
12/19/2013



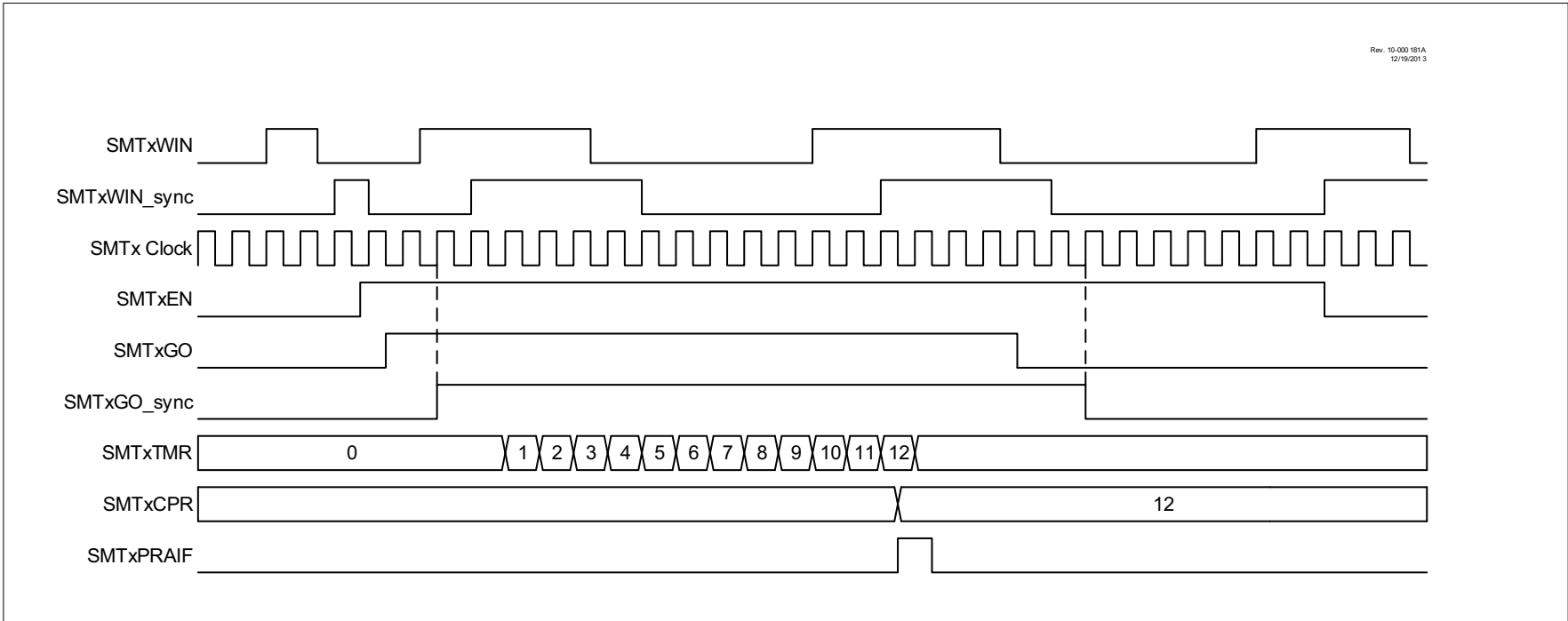
29.6.5 WINDOWED MEASURE MODE

This mode measures the window duration of the SMTWINx input of the SMT. It begins incrementing the timer on a rising edge of the SMTWINx input and updates the SMTxCPR register with the value of the timer and resets the timer on a second rising edge. See [Figure 29-10](#) and [Figure 29-11](#).

FIGURE 29-10: WINDOWED MEASURE MODE REPEAT ACQUISITION TIMING DIAGRAM



FIGURE 29-11: WINDOWED MEASURE MODE SINGLE ACQUISITION TIMING DIAGRAM



29.6.6 GATED WINDOW MEASURE MODE

This mode measures the duty cycle of the SMTx_signal input over a known input window. It does so by incrementing the timer on each pulse of the clock signal while the SMTx_signal input is high, updating the SMTxCPR register and resetting the timer on every rising edge of the SMTWINx input after the first. See [Figure 29-12](#) and [Figure 29-13](#).

FIGURE 29-12: GATED WINDOWED MEASURE MODE REPEAT ACQUISITION TIMING DIAGRAM

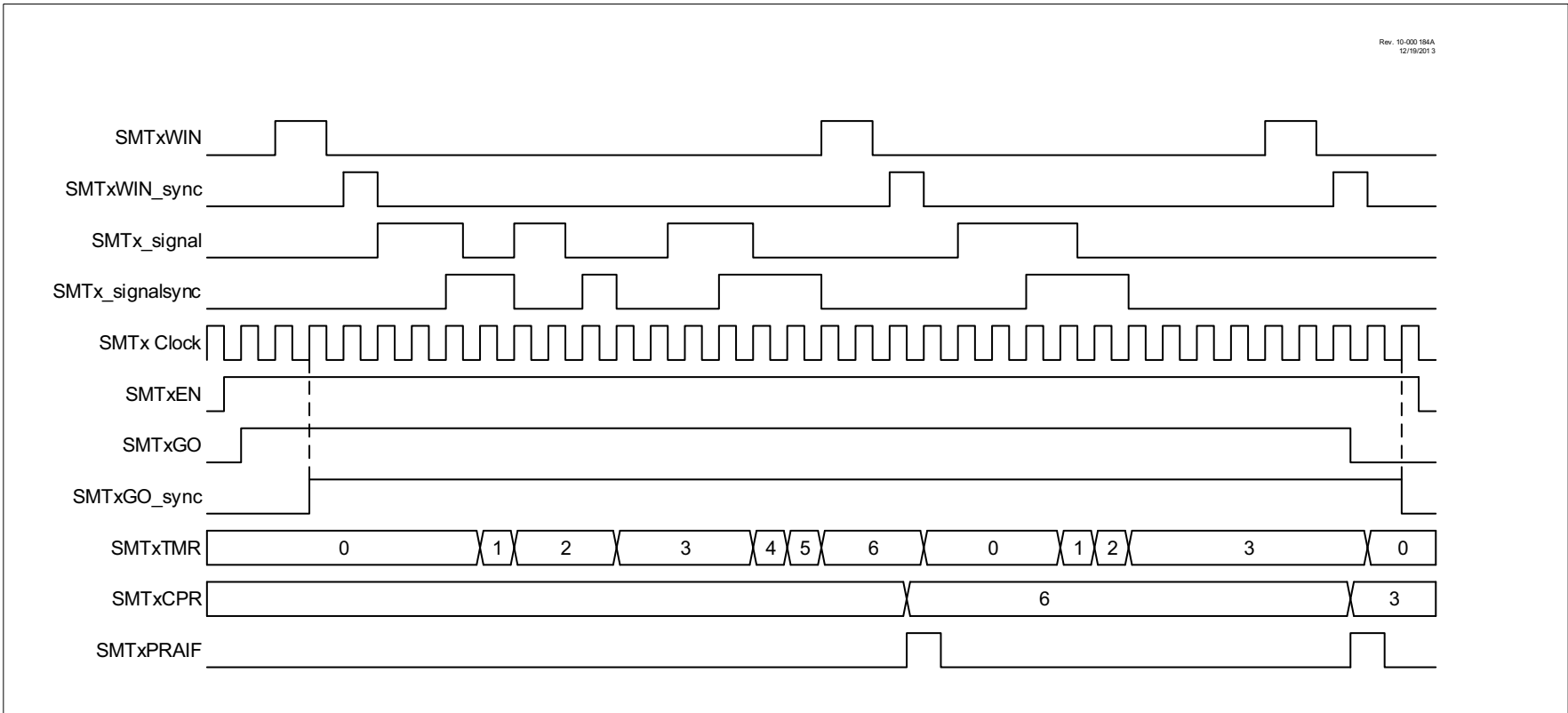


FIGURE 29-13: GATED WINDOWED MEASURE MODE SINGLE ACQUISITION TIMING DIAGRAMS



29.6.7 TIME OF FLIGHT MEASURE MODE

This mode measures the time interval between a rising edge on the SMTWINx input and a rising edge on the SMTx_signal input, beginning to increment the timer upon observing a rising edge on the SMTWINx input, while updating the SMTxCPR register and resetting the timer upon observing a rising edge on the SMTx_signal input. In the event of two SMTWINx rising edges without an SMTx_signal rising edge, it will update the SMTxCPW register with the current value of the timer and reset the timer value. See [Figure 29-14](#) and [Figure 29-15](#).

FIGURE 29-14: TIME OF FLIGHT MODE REPEAT ACQUISITION TIMING DIAGRAM

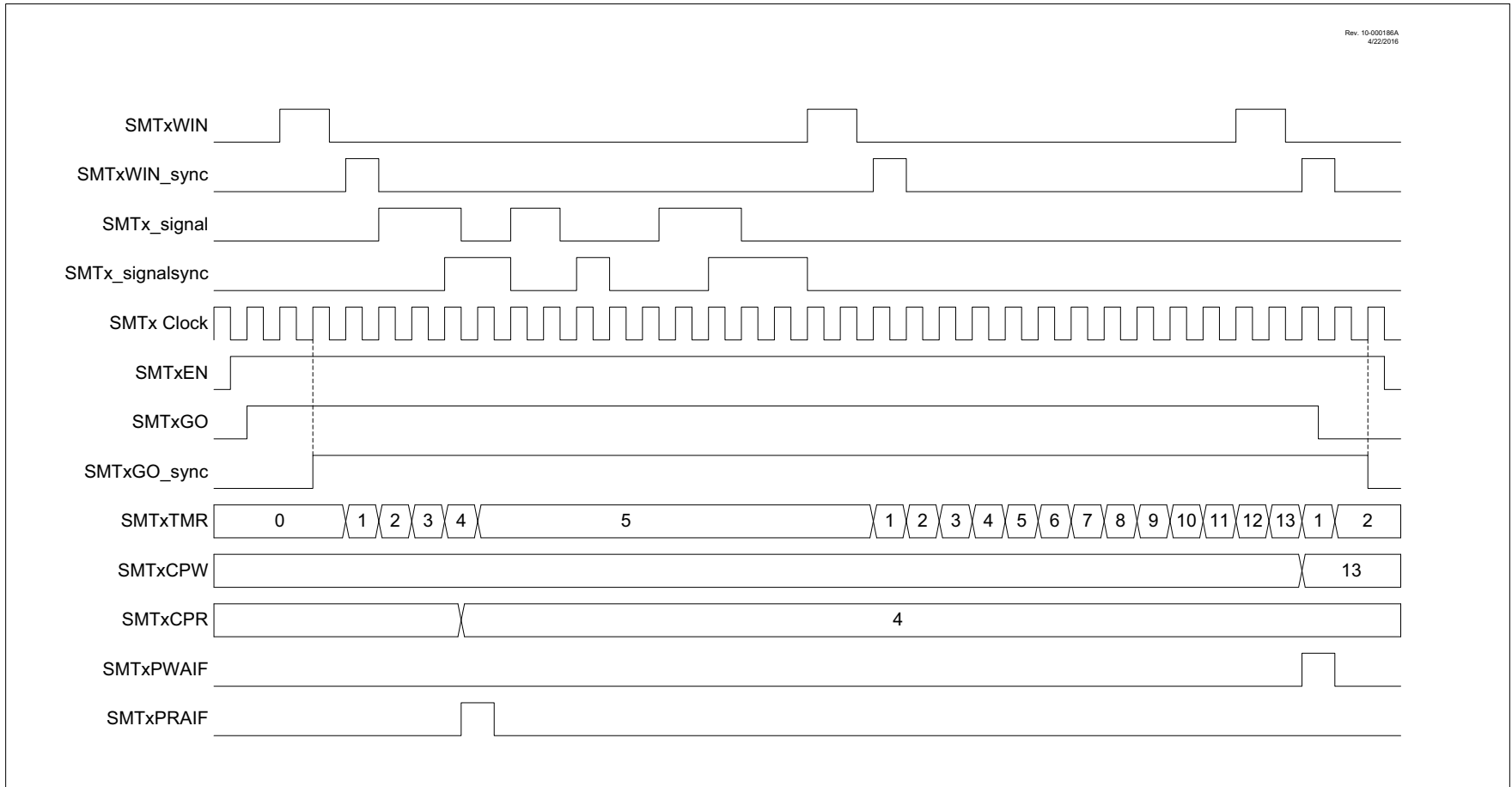


FIGURE 29-15: TIME OF FLIGHT MODE SINGLE ACQUISITION TIMING DIAGRAM

Rev. 10-000185A
4/26/2016



29.6.8 CAPTURE MODE

This mode captures the Timer value based on a rising or falling edge on the SMTWINx input and triggers an interrupt. This mimics the capture feature of a CCP module. The timer begins incrementing upon the SMTxGO bit being set, and updates the value of the SMTxCPR register on each rising edge of SMTWINx, and updates the value of the CPW register on each falling edge of the SMTWINx. The timer is not reset by any hardware conditions in this mode and must be reset by software, if desired. See [Figure 29-16](#) and [Figure 29-17](#).

FIGURE 29-16: CAPTURE MODE REPEAT ACQUISITION TIMING DIAGRAM



FIGURE 29-17: CAPTURE MODE SINGLE ACQUISITION TIMING DIAGRAM

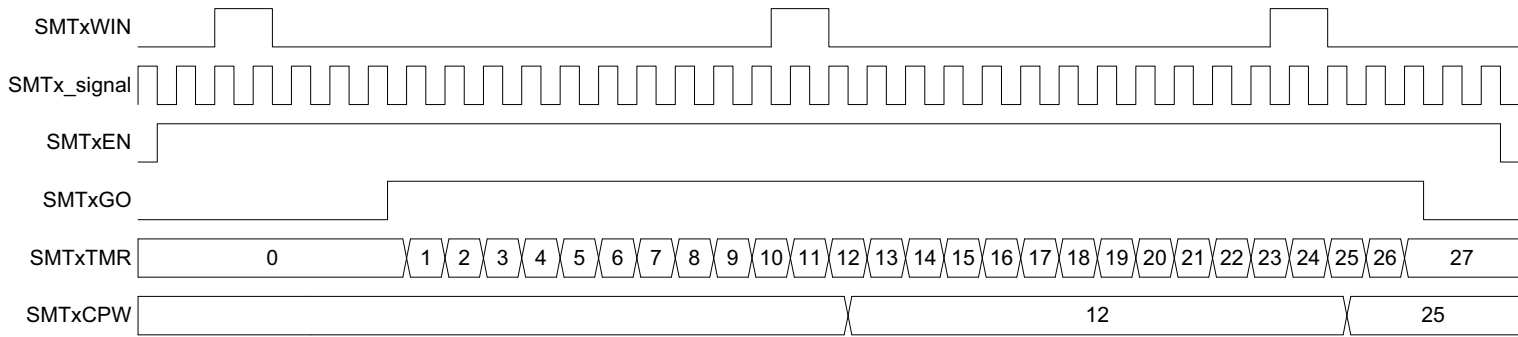


29.6.9 COUNTER MODE

This mode increments the timer on each pulse of the SMTx_signal input. This mode is asynchronous to the SMT clock and uses the SMTx_signal as a time source. The SMTxCPW register will be updated with the current SMTxTMR value on the falling edge of the SMTxWIN input. See [Figure 29-18](#).

FIGURE 29-18: COUNTER MODE TIMING DIAGRAM

Rev. 10-020168A
4/12/2016



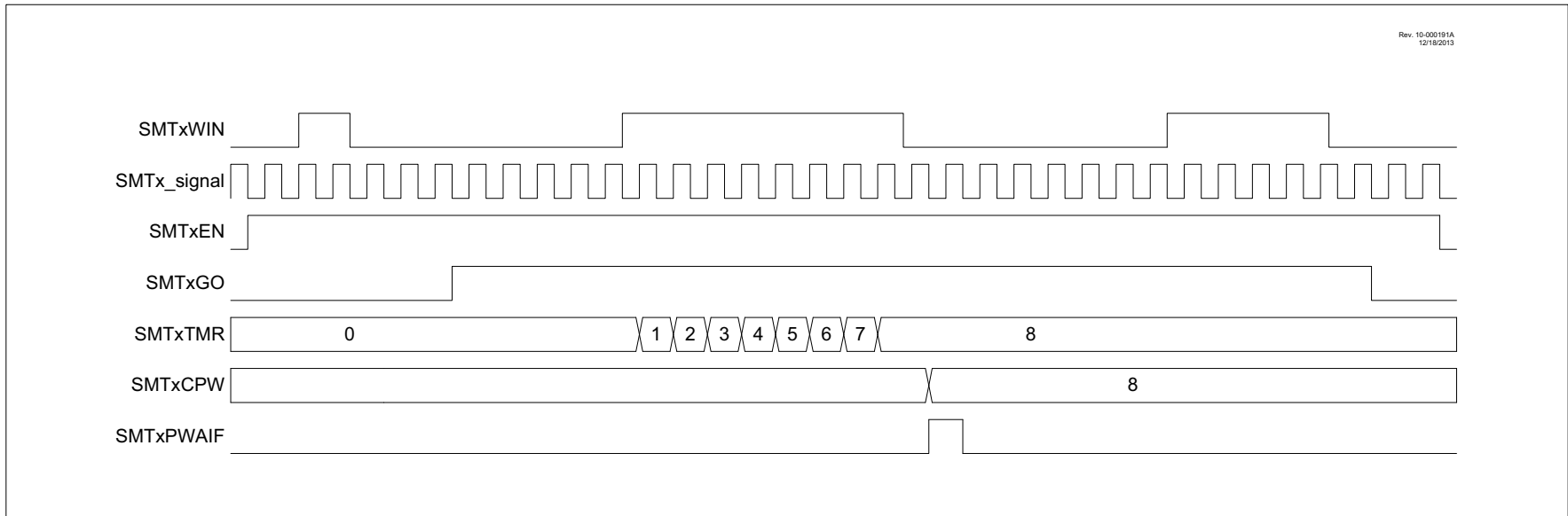
29.6.10 GATED COUNTER MODE

This mode counts pulses on the SMTx_signal input, gated by the SMTxWIN input. It begins incrementing the timer upon seeing a rising edge of the SMTxWIN input and updates the SMTxCPW register upon a falling edge on the SMTxWIN input. See [Figure 29-19](#) and [Figure 29-20](#).

FIGURE 29-19: GATED COUNTER MODE REPEAT ACQUISITION TIMING DIAGRAM



FIGURE 29-20: GATED COUNTER MODE SINGLE ACQUISITION TIMING DIAGRAM



29.6.11 WINDOWED COUNTER MODE

This mode counts pulses on the SMTx_signal input, within a window dictated by the SMTxWIN input. It begins counting upon seeing a rising edge of the SMTxWIN input, updates the SMTxCPW register on a falling edge of the SMTxWIN input, and updates the SMTxCPR register on each rising edge of the SMTxWIN input beyond the first. See [Figure 29-21](#) and [Figure 29-22](#).

FIGURE 29-21: WINDOWED COUNTER MODE REPEAT ACQUISITION TIMING DIAGRAM



FIGURE 29-22: WINDOWED COUNTER MODE SINGLE ACQUISITION TIMING DIAGRAM



29.7 Interrupts

The SMT can trigger an interrupt under three different conditions:

- PW Acquisition Complete
- PR Acquisition Complete
- Counter Period Match

The interrupts are controlled by the PIR and PIE registers of the device.

29.7.1 PW AND PR ACQUISITION INTERRUPTS

The SMT can trigger interrupts whenever it updates the SMTxCPW and SMTxCPR registers, the circumstances for which are dependent on the SMT mode, and are discussed in each mode's specific section. The SMTxCPW interrupt is controlled by SMTxPWAIF and SMTxPWAIE bits in registers PIR4 and PIE4, respectively. The SMTxCPR interrupt is controlled by the SMTxPRAIF and SMTxPRAIE bits, also located in registers PIR4 and PIE4, respectively.

In synchronous SMT modes, the interrupt trigger is synchronized to the SMTxCLK. In Asynchronous modes, the interrupt trigger is asynchronous. In either mode, once triggered, the interrupt will be synchronized to the CPU clock.

29.7.2 COUNTER PERIOD MATCH INTERRUPT

As described in [Section 29.1.2 “Period Match interrupt”](#), the SMT will also interrupt upon SMTxTMR, matching SMTxPR with its period match limit functionality described in [Section 29.3 “Halt Operation”](#). The period match interrupt is controlled by SMTxIF and SMTxIE, located in registers PIR4 and PIE4, respectively.

29.8 Register Definitions: SMT Control

Long bit name prefixes for the Signal Measurement Timer peripherals are shown in Table 29-2. Refer to Section 1.1 “Register and Bit Naming Conventions” for more information.

TABLE 29-2:

| Peripheral | Bit Name Prefix |
|------------|-----------------|
| SMT1 | SMT1 |
| SMT2 | SMT2 |

REGISTER 29-1: SMTxCON0: SMT CONTROL REGISTER 0

| | | | | | | | |
|-------------------|-----|---------|---------|---------|---------|-------------|---------|
| R/W-0/0 | 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 |
| EN ⁽¹⁾ | — | STP | WPOL | SPOL | CPOL | SMTxPS<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 **EN:** SMT Enable bit⁽¹⁾
1 = SMT is enabled
0 = SMT is disabled; internal states are reset, clock requests are disabled
- bit 6 **Unimplemented:** Read as '0'
- bit 5 **STP:** SMT Counter Halt Enable bit
When SMTxTMR = SMTxPR:
1 = Counter remains SMTxPR; period match interrupt occurs when clocked
0 = Counter resets to 24'h000000; period match interrupt occurs when clocked
- bit 4 **WPOL:** SMTxWIN Input Polarity Control bit
1 = SMTxWIN signal is active-low/falling edge enabled
0 = SMTxWIN signal is active-high/rising edge enabled
- bit 3 **SPOL:** SMTxSIG Input Polarity Control bit
1 = SMTx_signal is active-low/falling edge enabled
0 = SMTx_signal is active-high/rising edge enabled
- bit 2 **CPOL:** SMT Clock Input Polarity Control bit
1 = SMTxTMR increments on the falling edge of the selected clock signal
0 = SMTxTMR increments on the rising edge of the selected clock signal
- bit 1-0 **SMTxPS<1:0>:** SMT Prescale Select bits
11 = Prescaler = 1:8
10 = Prescaler = 1:4
01 = Prescaler = 1:2
00 = Prescaler = 1:1

Note 1: Setting EN to '0' does not affect the register contents.

REGISTER 29-2: SMTxCON1: SMT CONTROL REGISTER 1

| | | | | | | | |
|------------|---------|-----|-----|-----------|---------|---------|---------|
| R/W/HC-0/0 | R/W-0/0 | U-0 | U-0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 |
| SMTxGO | REPEAT | — | — | MODE<3:0> | | | |
| 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

-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 **SMTxGO:** SMT GO Data Acquisition bit
 1 = Incrementing, acquiring data is enabled
 0 = Incrementing, acquiring data is disabled
- bit 6 **REPEAT:** SMT Repeat Acquisition Enable bit
 1 = Repeat Data Acquisition mode is enabled
 0 = Single Acquisition mode is enabled
- bit 5-4 **Unimplemented:** Read as '0'
- bit 3-0 **MODE<3:0>** SMT Operation Mode Select bits
 1111 = Reserved
 •
 •
 •
 1011 = Reserved
 1010 = Windowed counter
 1001 = Gated counter
 1000 = Counter
 0111 = Capture
 0110 = Time of flight
 0101 = Gated windowed measure
 0100 = Windowed measure
 0011 = High and low time measurement
 0010 = Period and Duty-Cycle Acquisition
 0001 = Gated Timer
 0000 = Timer

REGISTER 29-3: SMTxSTAT: SMT STATUS REGISTER

| | | | | | | | |
|------------|------------|------------|-----|-----|-------|-------|-------|
| R/W/HC-0/0 | R/W/HC-0/0 | R/W/HC-0/0 | U-0 | U-0 | R-0/0 | R-0/0 | R-0/0 |
| CPRUP | CPWUP | RST | — | — | TS | WS | AS |
| bit 7 | | | | | | | bit 0 |

Legend:

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

- bit 7 **CPRUP:** SMT Manual Period Buffer Update bit
1 = Request update to SMTxCPRx registers
0 = SMTxCPRx registers update is complete
- bit 6 **CPWUP:** SMT Manual Pulse Width Buffer Update bit
1 = Request update to SMTxCPW registers
0 = SMTxCPW registers update is complete
- bit 5 **RST:** SMT Manual Timer Reset bit
1 = Request Reset to SMTxTMR registers
0 = SMTxTMR registers update is complete
- bit 4-3 **Unimplemented:** Read as '0'
- bit 2 **TS:** SMT GO Value Status bit
1 = SMT timer is incrementing
0 = SMT timer is not incrementing
- bit 1 **WS:** SMTxWIN Value Status bit
1 = SMT window is open
0 = SMT window is closed
- bit 0 **AS:** SMT_signal Value Status bit
1 = SMT acquisition is in progress
0 = SMT acquisition is not in progress

REGISTER 29-4: SMTxCLK: SMT CLOCK SELECTION REGISTER

| | | | | | | | |
|-------|-----|-----|-----|-----|-----------|---------|---------|
| U-0 | U-0 | U-0 | U-0 | U-0 | R/W-0/0 | R/W-0/0 | R/W-0/0 |
| — | — | — | — | — | CSEL<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 | q = Value depends on condition |

| | |
|---------|--|
| bit 7-3 | Unimplemented: Read as '0' |
| bit 2-0 | CSEL<2:0>: SMT Clock Selection bits |
| | 111 = Reserved |
| | 110 = AT1_perclk |
| | 101 = MFINTOSC |
| | 100 = MFINTOSC/16 |
| | 011 = LFINTOSC |
| | 010 = HFINTOSC 16 MHz |
| | 001 = Fosc/4 |
| | 000 = Fosc |

REGISTER 29-5: SMT1WIN: SMT1 WINDOW INPUT SELECT REGISTER

| | | | | | | | | |
|-------|-----|-----|-----------|---------|---------|---------|---------|-------|
| U-0 | U-0 | U-0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | |
| — | — | — | WSEL<4:0> | | | | | |
| 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 7-5 | Unimplemented: Read as '0' |
| bit 4-0 | WSEL<4:0>: SMT1 Window Selection bits |
| | 11111 = Reserved |
| | • |
| | • |
| | • |
| | 11000 = Reserved |
| | 10111 = MFINTOSC/16 |
| | 10110 = AT1_perclk |
| | 10101 = LFINTOSC |
| | 10100 = PWM4_out |
| | 10011 = PWM3_out |
| | 10010 = SMT2_match |
| | 10001 = Reserved |
| | 10000 = TMR0_overflow |
| | 01111 = TMR5_overflow |
| | 01110 = TMR3_overflow |
| | 01101 = TMR1_overflow |
| | 01100 = LC4_out |
| | 01011 = LC3_out |
| | 01010 = LC2_out |
| | 01001 = LC1_out |
| | 01000 = TMR6_postscaled |
| | 00111 = TMR4_postscaled |
| | 00110 = TMR2_postscaled |
| | 00101 = ZCD1_out |
| | 00100 = CCP2_out |
| | 00011 = CCP1_out |
| | 00010 = C2OUT_sync |
| | 00001 = C1OUT_sync |
| | 00000 = SMTWINx pin |

REGISTER 29-6: SMT2WIN: SMT2 WINDOW INPUT SELECT REGISTER

| | | | | | | | | |
|-------|-----|-----|-----------|---------|---------|---------|---------|-------|
| U-0 | U-0 | U-0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | |
| — | — | — | WSEL<4:0> | | | | | |
| 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 7-5 **Unimplemented:** Read as '0'

bit 4-0 **WSEL<4:0>:** SMT2 Window Selection bits

- 11111 = Reserved
-
-
-
- 11000 = Reserved
- 10111 = MFINTOSC/16
- 10110 = AT1_perclk
- 10101 = LFINTOSC
- 10100 = PWM4_out
- 10011 = PWM3_out
- 10010 = Reserved
- 10001 = SMT1_match
- 10000 = TMR0_overflow
- 01111 = TMR5_overflow
- 01110 = TMR3_overflow
- 01101 = TMR1_overflow
- 01100 = LC4_out
- 01011 = LC3_out
- 01010 = LC2_out
- 01001 = LC1_out
- 01000 = TMR6_postscaled
- 00111 = TMR4_postscaled
- 00110 = TMR2_postscaled
- 00101 = ZCD1_out
- 00100 = CCP2_out
- 00011 = CCP1_out
- 00010 = C2OUT_sync
- 00001 = C1OUT_sync
- 00000 = SMTWINx pin

REGISTER 29-7: SMT1SIG: SMT1 SIGNAL INPUT SELECT REGISTER

| | | | | | | | | |
|-------|-----|-----|-----------|---------|---------|---------|---------|-------|
| U-0 | U-0 | U-0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | |
| — | — | — | SSEL<4:0> | | | | | |
| 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 7-5 | Unimplemented: Read as '0' |
| bit 4-0 | SSEL<4:0>: SMT1 Signal Selection bits |
| | 11111 = Reserved |
| | • |
| | • |
| | • |
| | 10101 = Reserved |
| | 10100 = PWM4_out |
| | 10011 = PWM3_out |
| | 10010 = CCP2_out |
| | 10001 = CCP1_out |
| | 10000 = TMR0_overflow |
| | 01111 = SMT2_match |
| | 01110 = Reserved |
| | 01101 = TMR5_overflow |
| | 01100 = TMR3_overflow |
| | 01011 = TMR1_overflow |
| | 01010 = LC4_out |
| | 01001 = LC3_out |
| | 01000 = LC2_out |
| | 00111 = LC1_out |
| | 00110 = TMR6_postscaled |
| | 00101 = TMR4_postscaled |
| | 00100 = TMR2_postscaled |
| | 00011 = ZCD1_out |
| | 00010 = C2OUT_sync |
| | 00001 = C1OUT_sync |
| | 00000 = SMTxSIG pin |

REGISTER 29-8: SMT2SIG: SMT2 SIGNAL INPUT SELECT REGISTER

| | | | | | | | | |
|-------|-----|-----|-----------|---------|---------|---------|---------|-------|
| U-0 | U-0 | U-0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | |
| — | — | — | SSEL<4:0> | | | | | |
| 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 7-5 | Unimplemented: Read as '0' |
| bit 4-0 | SSEL<4:0>: SMT2 Signal Selection bits 11111 = Reserved • • • 10101 = Reserved 10100 = PWM4_out 10011 = PWM3_out 10010 = CCP2_out 10001 = CCP1_out 10000 = TMR0_overflow 01111 = Reserved 01110 = SMT1_match 01101 = TMR5_overflow 01100 = TMR3_overflow 01011 = TMR1_overflow 01010 = LC4_out 01001 = LC3_out 01000 = LC2_out 00111 = LC1_out 00110 = TMR6_postscaled 00101 = TMR4_postscaled 00100 = TMR2_postscaled 00011 = ZCD1_out 00010 = C2OUT_sync 00001 = C1OUT_sync 00000 = SMTxSIG pin |

REGISTER 29-9: SMTxTMRL: SMT TIMER REGISTER – LOW BYTE

| | | | | | | | |
|--------------|---------|---------|---------|---------|---------|---------|---------|
| 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 |
| SMTxTMR<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 **SMTxTMR<7:0>**: Significant bits of the SMT Counter – Low Byte

REGISTER 29-10: SMTxTMRH: SMT TIMER REGISTER – HIGH BYTE

| | | | | | | | |
|---------------|---------|---------|---------|---------|---------|---------|---------|
| 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 |
| SMTxTMR<15:8> | | | | | | | |
| 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 7-0 **SMTxTMR<15:8>**: Significant bits of the SMT Counter – High Byte

REGISTER 29-11: SMTxTMRU: SMT TIMER REGISTER – UPPER BYTE

| | | | | | | | |
|----------------|---------|---------|---------|---------|---------|---------|---------|
| 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 |
| SMTxTMR<23:16> | | | | | | | |
| 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 **SMTxTMR<23:16>**: Significant bits of the SMT Counter – Upper Byte

REGISTER 29-12: SMTxCPRL: SMT CAPTURED PERIOD REGISTER – LOW BYTE

| | | | | | | | |
|--------------|-------|-------|-------|-------|-------|-------|-------|
| R-x/x | R-x/x | R-x/x | R-x/x | R-x/x | R-x/x | R-x/x | R-x/x |
| SMTxCPR<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 **SMTxCPR<7:0>**: Significant bits of the SMT Period Latch – Low Byte

REGISTER 29-13: SMTxCPRH: SMT CAPTURED PERIOD REGISTER – HIGH BYTE

| | | | | | | | |
|---------------|-------|-------|-------|-------|-------|-------|-------|
| R-x/x | R-x/x | R-x/x | R-x/x | R-x/x | R-x/x | R-x/x | R-x/x |
| SMTxCPR<15:8> | | | | | | | |
| 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 7-0 **SMTxCPR<15:8>**: Significant bits of the SMT Period Latch – High Byte

REGISTER 29-14: SMTxCPRU: SMT CAPTURED PERIOD REGISTER – UPPER BYTE

| | | | | | | | |
|----------------|-------|-------|-------|-------|-------|-------|-------|
| R-x/x | R-x/x | R-x/x | R-x/x | R-x/x | R-x/x | R-x/x | R-x/x |
| SMTxCPR<23:16> | | | | | | | |
| 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 **SMTxCPR<23:16>**: Significant bits of the SMT Period Latch – Upper Byte

REGISTER 29-15: SMTxCPWL: SMT CAPTURED PULSE WIDTH REGISTER – LOW BYTE

| | | | | | | | |
|--------------|-------|-------|-------|-------|-------|-------|-------|
| R-x/x | R-x/x | R-x/x | R-x/x | R-x/x | R-x/x | R-x/x | R-x/x |
| SMTxCPW<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 **SMTxCPW<7:0>**: Significant bits of the SMT PW Latch – Low Byte

REGISTER 29-16: SMTxCPWH: SMT CAPTURED PULSE WIDTH REGISTER – HIGH BYTE

| | | | | | | | |
|---------------|-------|-------|-------|-------|-------|-------|-------|
| R-x/x | R-x/x | R-x/x | R-x/x | R-x/x | R-x/x | R-x/x | R-x/x |
| SMTxCPW<15:8> | | | | | | | |
| 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 7-0 **SMTxCPW<15:8>**: Significant bits of the SMT PW Latch – High Byte

REGISTER 29-17: SMTxCPWU: SMT CAPTURED PULSE WIDTH REGISTER – UPPER BYTE

| | | | | | | | |
|----------------|-------|-------|-------|-------|-------|-------|-------|
| R-x/x | R-x/x | R-x/x | R-x/x | R-x/x | R-x/x | R-x/x | R-x/x |
| SMTxCPW<23:16> | | | | | | | |
| 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 **SMTxCPW<23:16>**: Significant bits of the SMT PW Latch – Upper Byte

REGISTER 29-18: SMTxPRL: SMT PERIOD REGISTER – LOW BYTE

| | | | | | | | |
|-------------|---------|---------|---------|---------|---------|---------|---------|
| R/W-x/1 | R/W-x/1 | R/W-x/1 | R/W-x/1 | R/W-x/1 | R/W-x/1 | R/W-x/1 | R/W-x/1 |
| SMTxPR<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 **SMTxPR<7:0>**: Significant bits of the SMT Timer Value for Period Match – Low Byte

REGISTER 29-19: SMTxPRH: SMT PERIOD REGISTER – HIGH BYTE

| | | | | | | | |
|--------------|---------|---------|---------|---------|---------|---------|---------|
| R/W-x/1 | R/W-x/1 | R/W-x/1 | R/W-x/1 | R/W-x/1 | R/W-x/1 | R/W-x/1 | R/W-x/1 |
| SMTxPR<15:8> | | | | | | | |
| 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 7-0 **SMTxPR<15:8>**: Significant bits of the SMT Timer Value for Period Match – High Byte

REGISTER 29-20: SMTxPRU: SMT PERIOD REGISTER – UPPER BYTE

| | | | | | | | |
|---------------|---------|---------|---------|---------|---------|---------|---------|
| R/W-x/1 | R/W-x/1 | R/W-x/1 | R/W-x/1 | R/W-x/1 | R/W-x/1 | R/W-x/1 | R/W-x/1 |
| SMTxPR<23:16> | | | | | | | |
| 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 **SMTxPR<23:16>**: Significant bits of the SMT Timer Value for Period Match – Upper Byte

TABLE 29-3: SUMMARY OF REGISTERS ASSOCIATED WITH SMTx

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Register on Page | |
|----------|----------------|--------|-----------|-----------|-----------|-----------|-------------|--------|------------------|-----|
| PIE4 | SCANIE | CRCIE | SMT2PWAIE | SMT2PRAIE | SMT2IE | SMT1PWAIE | SMT1PRAIE | SMT1IE | 101 | |
| PIR4 | SCANIF | CRCIF | SMT2PWAIF | SMT2PRAIF | SMT2IF | SMT1PWAIF | SMT1PRAIF | SMT1IF | 106 | |
| SMT1CON0 | EN | — | STP | WPOL | SPOL | CPOL | SMT1PS<1:0> | | 424 | |
| SMT1CON1 | SMT1GO | REPEAT | — | — | MODE<3:0> | | | | 425 | |
| SMT1CPRH | SMT1CPR<15:8> | | | | | | | | 433 | |
| SMT1CPRL | SMT1CPR<7:0> | | | | | | | | 433 | |
| SMT1CPRU | SMT1CPR<23:16> | | | | | | | | 433 | |
| SMT1CPWH | SMT1CPW<15:8> | | | | | | | | 434 | |
| SMT1CPWL | SMT1CPW<7:0> | | | | | | | | 434 | |
| SMT1CPWU | SMT1CPW<23:16> | | | | | | | | 434 | |
| SMT1PRH | SMT1PR<15:8> | | | | | | | | 435 | |
| SMT1PRL | SMT1PR<7:0> | | | | | | | | 435 | |
| SMT1PRU | SMT1PR<23:16> | | | | | | | | 435 | |
| SMT1SIG | — | — | — | SSEL<4:0> | | | | | | 430 |
| SMT1STAT | CPRUP | CPWUP | RST | — | — | TS | WS | AS | 426 | |
| SMT1TMRH | SMT1TMR<15:8> | | | | | | | | 432 | |
| SMT1TMRL | SMT1TMR<7:0> | | | | | | | | 432 | |
| SMT1TMRU | SMT1TMR<23:16> | | | | | | | | 432 | |
| SMT1WIN | — | — | — | WSEL<4:0> | | | | | | 428 |
| SMT2CLK | — | — | — | — | — | CSEL<2:0> | | | | 427 |
| SMT2CON0 | EN | — | STP | WPOL | SPOL | CPOL | SMT2PS<1:0> | | 424 | |
| SMT2CON1 | SMT2GO | REPEAT | — | — | MODE<3:0> | | | | 425 | |
| SMT2CPRH | SMT2CPR<15:8> | | | | | | | | 433 | |
| SMT2CPRL | SMT2CPR<7:0> | | | | | | | | 433 | |
| SMT2CPRU | SMT2CPR<23:16> | | | | | | | | 433 | |
| SMT2CPWH | SMT2CPW<15:8> | | | | | | | | 434 | |
| SMT2CPWL | SMT2CPW<7:0> | | | | | | | | 434 | |
| SMT2CPWU | SMT2CPW<23:16> | | | | | | | | 434 | |
| SMT2PRH | SMT2PR<15:8> | | | | | | | | 435 | |
| SMT2PRL | SMT2PR<7:0> | | | | | | | | 435 | |
| SMT2PRU | SMT2PR<23:16> | | | | | | | | 435 | |
| SMT2SIG | — | — | — | — | — | SSEL<2:0> | | | | 430 |
| SMT2STAT | CPRUP | CPWUP | RST | — | — | TS | WS | AS | 426 | |
| SMT2TMRH | SMT2TMR<15:8> | | | | | | | | 432 | |
| SMT2TMRL | SMT2TMR<7:0> | | | | | | | | 432 | |
| SMT2TMRU | SMT2TMR<23:16> | | | | | | | | 432 | |
| SMT2WIN | — | — | — | WSEL<4:0> | | | | | | 427 |

Legend: x = unknown, u = unchanged, — = unimplemented read as '0', q = value depends on condition. Shaded cells are not used for SMTx module.

30.0 CONFIGURABLE LOGIC CELL (CLC)

The Configurable Logic Cell (CLCx) provides programmable logic that operates outside the speed limitations of software execution. The logic cell takes up to 16 input signals, and through the use of configurable gates, reduces the 16 inputs to four logic lines that drive one of eight selectable single-output logic functions.

Input sources are a combination of the following:

- I/O pins
- Internal clocks
- Peripherals
- Register bits

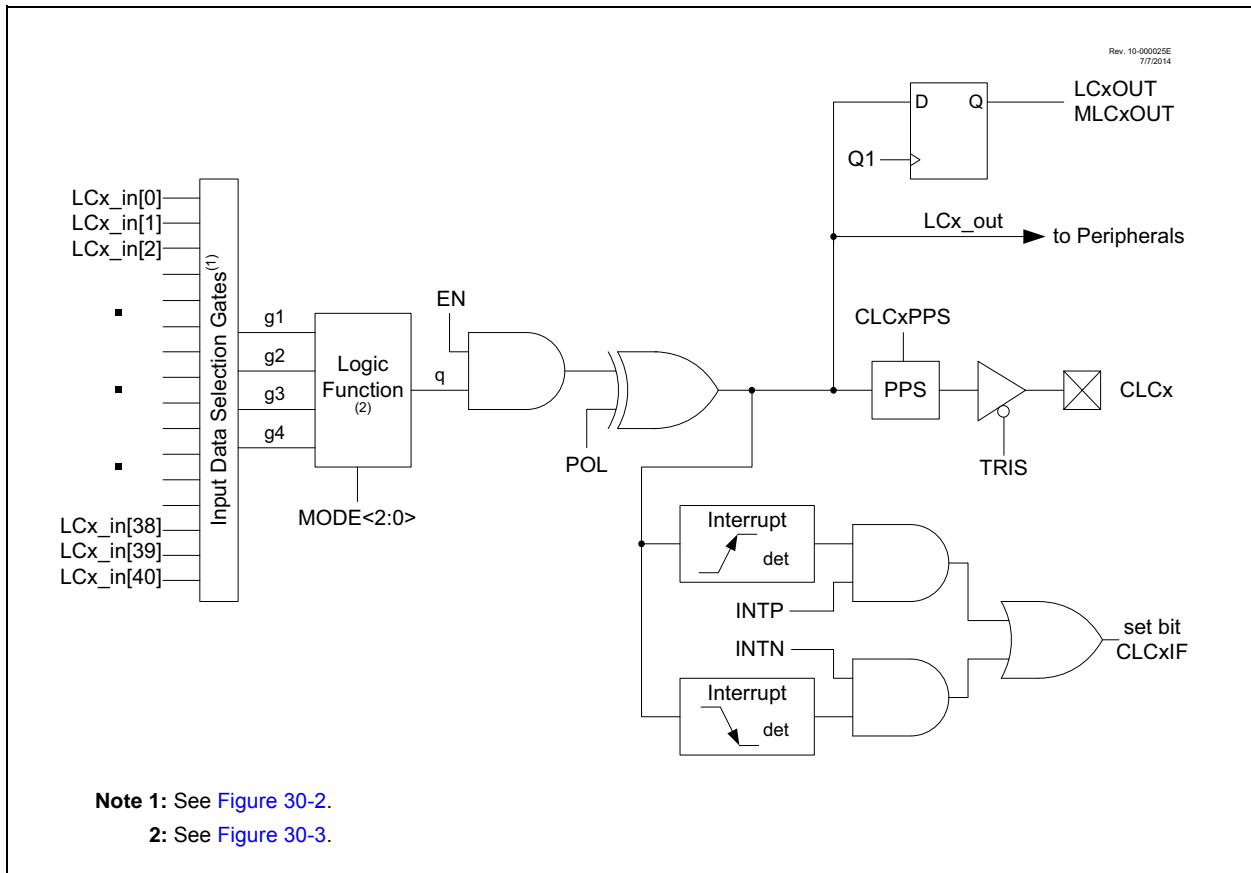
The output can be directed internally to peripherals and to an output pin.

Refer to [Figure 30-1](#) for a simplified diagram showing signal flow through the CLCx.

Possible configurations include:

- Combinatorial Logic
 - AND
 - NAND
 - AND-OR
 - AND-OR-INVERT
 - OR-XOR
 - OR-XNOR
- Latches
 - S-R
 - Clocked D with Set and Reset
 - Transparent D with Set and Reset
 - Clocked J-K with Reset

FIGURE 30-1: CONFIGURABLE LOGIC CELL BLOCK DIAGRAM



30.1 CLCx Setup

Programming the CLCx module is performed by configuring the four stages in the logic signal flow. The four stages are:

- Data selection
- Data gating
- Logic function selection
- Output polarity

Each stage is setup at run time by writing to the corresponding CLCx Special Function Registers. This has the added advantage of permitting logic reconfiguration on-the-fly during program execution.

30.1.1 DATA SELECTION

There are 41 signals available as inputs to the configurable logic. Four 41 input multiplexers are used to select the inputs to pass on to the next stage. This allows for any of the possible input signals to be used as any of the four inputs to the CLC module.

Data selection is through four multiplexers as indicated on the left side of [Figure 30-2](#). Data inputs in the figure are identified by a generic numbered input name.

[Table 30-1](#) correlates the generic input name to the actual signal for each CLC module. The column labeled CLCxSELY refers to the value of any of the four registers associated with the four multiplexers, CLCxSELO through CLCxSEL3.

Data inputs for each multiplexer are selected with their respective CLCxSELY registers.

Note: Data selections are undefined at power-up.

TABLE 30-1: CLCx DATA INPUT SELECTION

| Data Input | CLCxSELY | CLC Input Signal |
|------------|----------|------------------|
| LCx_in[0] | 000000 | CLCIN0 |
| LCx_in[1] | 000001 | CLCIN1 |
| LCx_in[2] | 000010 | CLCIN2 |
| LCx_in[3] | 000011 | CLCIN3 |
| LCx_in[4] | 000100 | LC1_out |
| LCx_in[5] | 000101 | LC2_out |
| LCx_in[6] | 000110 | LC3_out |
| LCx_in[7] | 000111 | LC4_out |
| LCx_in[8] | 001000 | C1OUT_sync |
| LCx_in[9] | 001001 | C2OUT_sync |
| LCx_in[10] | 001010 | CWGOUTA |
| LCx_in[11] | 001011 | CWGOUTB |
| LCx_in[12] | 001100 | CCP1_out |
| LCx_in[13] | 001101 | CCP2_out |
| LCx_in[14] | 001110 | PWM3_out |
| LCx_in[15] | 001111 | PWM4_out |
| LCx_in[16] | 010000 | AT1_cmp1 |
| LCx_in[17] | 010001 | AT1_cmp2 |
| LCx_in[18] | 010010 | AT1_cmp3 |
| LCx_in[19] | 010011 | SMT1_match |
| LCx_in[20] | 010100 | SMT2_match |
| LCx_in[21] | 010101 | ZCD1_output |
| LCx_in[22] | 010110 | TMR0_overflow |
| LCx_in[23] | 010111 | TMR1_overflow |
| LCx_in[24] | 011000 | TMR2_postscaled |
| LCx_in[25] | 011001 | TMR3_overflow |
| LCx_in[26] | 011010 | TMR4_postscaled |
| LCx_in[27] | 011011 | TMR5_overflow |
| LCx_in[28] | 011100 | TMR6_postscaled |
| LCx_in[29] | 011101 | IOC_interrupt |
| LCx_in[30] | 011110 | ADC_rc |
| LCx_in[31] | 011111 | LFINTOSC |
| LCx_in[32] | 100000 | HFINTOSC |
| LCx_in[33] | 100001 | FOSC |
| LCx_in[34] | 100010 | AT1_missedpulse |
| LCx_in[35] | 100011 | AT1_perclk |
| LCx_in[36] | 100100 | AT1_phsclk |
| LCx_in[37] | 100101 | TX |
| LCx_in[38] | 100110 | RX |
| LCx_in[39] | 100111 | SCK |
| LCx_in[40] | 101000 | SDO |

30.2 Data Gating

Outputs from the input multiplexers are directed to the desired logic function input through the data gating stage. Each data gate can direct any combination of the four selected inputs.

Note: Data gating is undefined at power-up.

The gate stage is more than just signal direction. The gate can be configured to direct each input signal as inverted or non-inverted data. Directed signals are ANDed together in each gate. The output of each gate can be inverted before going on to the logic function stage.

The gating is in essence a 1-to-4 input AND/NAND/OR/NOR gate. When every input is inverted and the output is inverted, the gate is an OR of all enabled data inputs. When the inputs and output are not inverted, the gate is an AND of all enabled inputs.

Table 30-2 summarizes the basic logic that can be obtained in gate 1 by using the gate logic select bits. The table shows the logic of four input variables, but each gate can be configured to use less than four. If no inputs are selected, the output will be zero or one, depending on the gate output polarity bit.

TABLE 30-2: DATA GATING LOGIC

| CLCxGLS0 | LCxG1POL | Gate Logic |
|----------|----------|------------|
| 0x55 | 1 | AND |
| 0x55 | 0 | NAND |
| 0xAA | 1 | NOR |
| 0xAA | 0 | OR |
| 0x00 | 0 | Logic 0 |
| 0x00 | 1 | Logic 1 |

It is possible (but not recommended) to select both the true and negated values of an input. When this is done, the gate output is zero, regardless of the other inputs, but may emit logic glitches (transient-induced pulses). If the output of the channel must be zero or one, the recommended method is to set all gate bits to zero and use the gate polarity bit to set the desired level.

Data gating is configured with the logic gate select registers as follows:

- Gate 1: CLCxGLS0 (Register 30-6)
- Gate 2: CLCxGLS1 (Register 30-7)
- Gate 3: CLCxGLS2 (Register 30-8)
- Gate 4: CLCxGLS3 (Register 30-9)

Register number suffixes are different than the gate numbers because other variations of this module have multiple gate selections in the same register.

Data gating is indicated in the right side of Figure 30-2. Only one gate is shown in detail. The remaining three gates are configured identically with the exception that the data enables correspond to the enables for that gate.

30.2.1 LOGIC FUNCTION

There are eight available logic functions including:

- AND-OR
- OR-XOR
- AND
- S-R Latch
- D Flip-Flop with Set and Reset
- D Flip-Flop with Reset
- J-K Flip-Flop with Reset
- Transparent Latch with Set and Reset

Logic functions are shown in Figure 30-3. Each logic function has four inputs and one output. The four inputs are the four data gate outputs of the previous stage. The output is fed to the inversion stage and from there to other peripherals, an output pin, and back to the CLCx itself.

30.2.2 OUTPUT POLARITY

The last stage in the configurable logic cell is the output polarity. Setting the LCxPOL bit of the CLCxCON register inverts the output signal from the logic stage. Changing the polarity while the interrupts are enabled will cause an interrupt for the resulting output transition.

30.2.3 CLCx SETUP STEPS

The following steps should be followed when setting up the CLCx:

- Disable CLCx by clearing the LCxEN bit.
- Select desired inputs using CLCxSEL0, CLCxSEL1, CLCxSEL2 and CLCxSEL3 registers (See [Table 30-1](#)).
- Clear any associated ANSEL bits.
- Set all TRIS bits associated with inputs.
- Clear all TRIS bits associated with outputs.
- Enable the chosen inputs through the four gates using CLCxGLS0, CLCxGLS1, CLCxGLS2, and CLCxGLS3 registers.
- Select the gate output polarities with the LCxPOLy bits of the CLCxPOL register.
- Select the desired logic function with the LCxMODE<2:0> bits of the CLCxCON register.
- Select the desired polarity of the logic output with the LCxPOL bit of the CLCxPOL register. (This step may be combined with the previous gate output polarity step).
- If driving a device, set the desired pin PPS control register and also clear the TRIS bit corresponding to that output.
- If interrupts are desired, configure the following bits:
 - Set the LCxINTP bit in the CLCxCON register for rising event.
 - Set the LCxINTN bit in the CLCxCON register or falling event.
 - Set the CLCxIE bit of the associated PIE registers.
 - Set the GIE and PEIE bits of the INTCON register.
- Enable the CLCx by setting the LCxEN bit of the CLCxCON register.

30.3 CLCx Interrupts

An interrupt will be generated upon a change in the output value of the CLCx when the appropriate interrupt enables are set. A rising edge detector and a falling edge detector are present in each CLC for this purpose.

The CLCxIF bit of the associated PIR registers will be set when either edge detector is triggered and its associated enable bit is set. The LCxINTP enables rising edge interrupts and the LCxINTN bit enables falling edge interrupts. Both are located in the CLCxCON register.

To fully enable the interrupt, set the following bits:

- LCxON bit of the CLCxCON register
- CLCxIE bit of the associated PIE registers
- LCxINTP bit of the CLCxCON register (for a rising edge detection)
- LCxINTN bit of the CLCxCON register (for a falling edge detection)
- PEIE and GIE bits of the INTCON register

The CLCxIF bit of the associated PIR registers, must be cleared in software as part of the interrupt service. If another edge is detected while this flag is being cleared, the flag will still be set at the end of the sequence.

30.4 Output Mirror Copies

Mirror copies of all LCxCON output bits are contained in the CLCxDATA register. Reading this register reads the outputs of all CLCs simultaneously. This prevents any reading skew introduced by testing or reading the CLCxOUT bits in the individual CLCxCON registers.

30.5 Effects of a Reset

The CLCxCON register is cleared to zero as the result of a Reset. All other selection and gating values remain unchanged.

30.6 Operation During Sleep

The CLC module operates independently from the system clock and will continue to run during Sleep, provided that the input sources selected remain active.

The HFINTOSC remains active during Sleep when the CLC module is enabled and the HFINTOSC is selected as an input source, regardless of the system clock source selected.

In other words, if the HFINTOSC is simultaneously selected as the system clock and as a CLC input source, when the CLC is enabled, the CPU will go idle during Sleep, but the CLC will continue to operate and the HFINTOSC will remain active.

This will have a direct effect on the Sleep mode current.

FIGURE 30-2: INPUT DATA SELECTION AND GATING

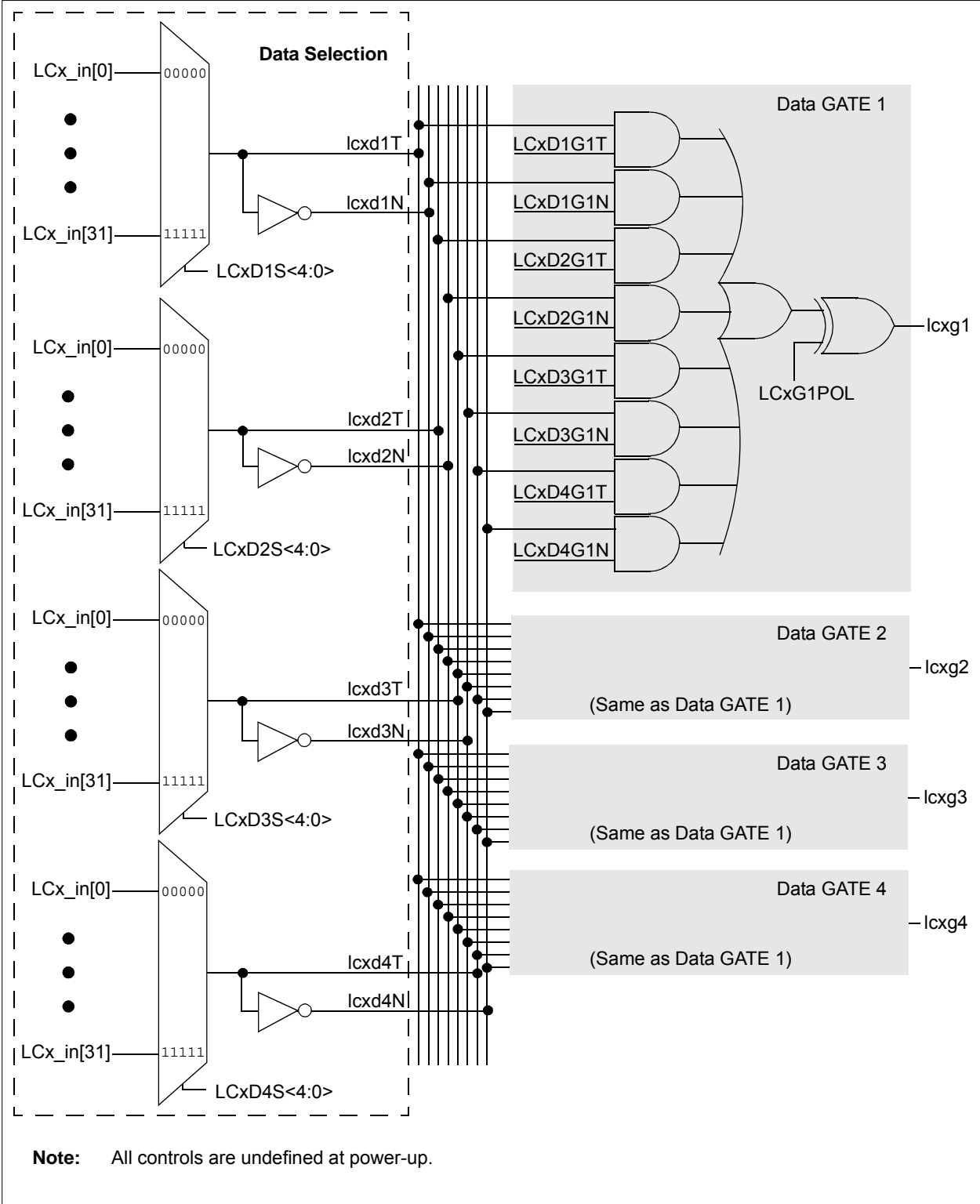
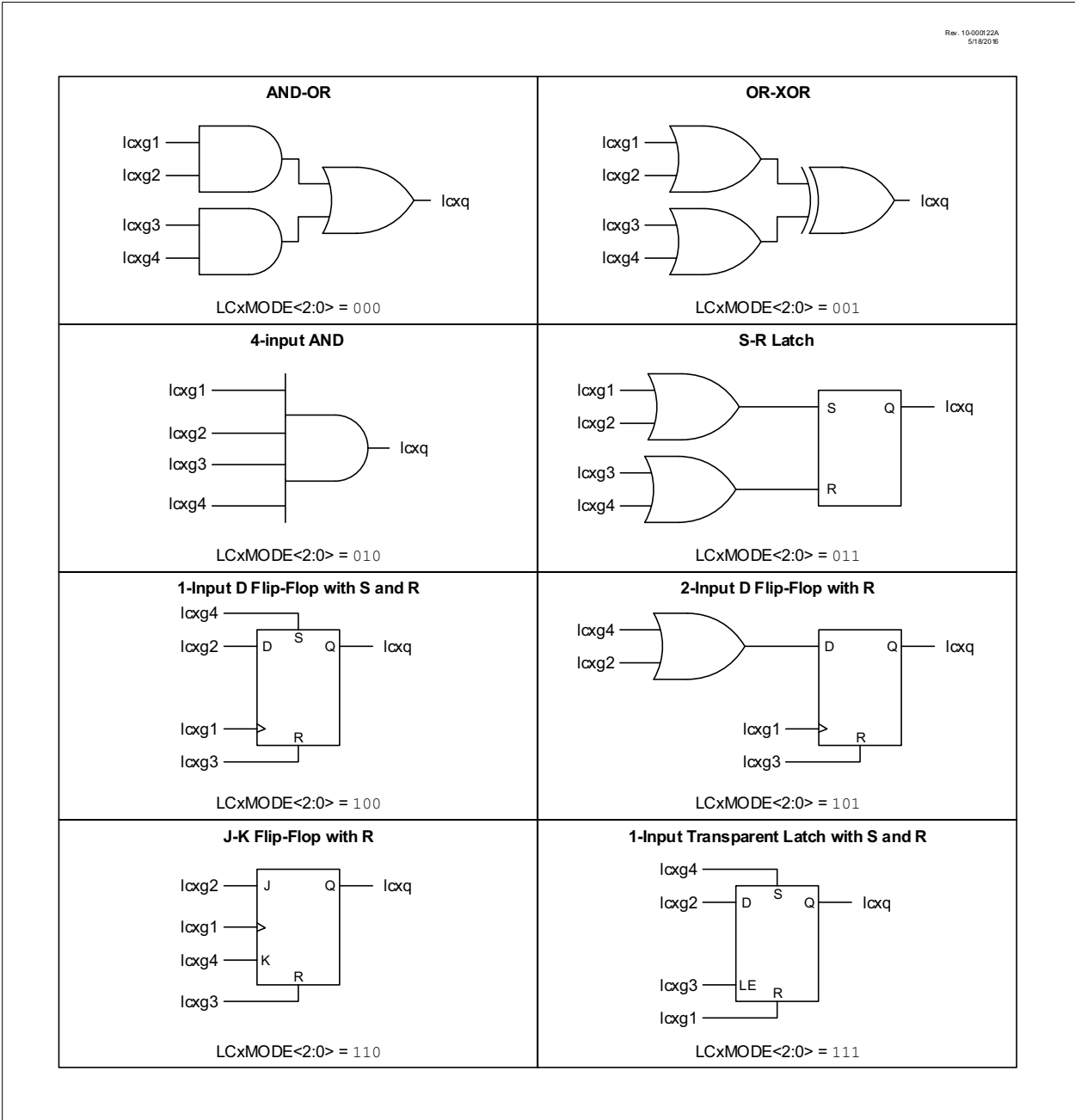


FIGURE 30-3: PROGRAMMABLE LOGIC FUNCTIONS



30.7 Register Definitions: CLC Control

REGISTER 30-1: CLCxCON: CONFIGURABLE LOGIC CELL CONTROL REGISTER

| | | | | | | | |
|---------|-----|--------|---------|---------|--------------|---------|---------|
| R/W-0/0 | U-0 | R-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 |
| LCxEN | — | LCxOUT | LCxINTP | LCxINTN | LCxMODE<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 **LCxEN:** Configurable Logic Cell Enable bit
 1 = Configurable logic cell is enabled and mixing input signals
 0 = Configurable logic cell is disabled and has logic zero output
- bit 6 **Unimplemented:** Read as '0'
- bit 5 **LCxOUT:** Configurable Logic Cell Data Output bit
 Read-only: logic cell output data, after LCxPOL; sampled from lcx_out wire.
- bit 4 **LCxINTP:** Configurable Logic Cell Positive Edge Going Interrupt Enable bit
 1 = CLCxIF will be set when a rising edge occurs on lcx_out
 0 = CLCxIF will not be set
- bit 3 **LCxINTN:** Configurable Logic Cell Negative Edge Going Interrupt Enable bit
 1 = CLCxIF will be set when a falling edge occurs on lcx_out
 0 = CLCxIF will not be set
- bit 2-0 **LCxMODE<2:0>:** Configurable Logic Cell Functional Mode bits
 111 = Cell is 1-input transparent latch with S and R
 110 = Cell is J-K flip-flop with R
 101 = Cell is 2-input D flip-flop with R
 100 = Cell is 1-input D flip-flop with S and R
 011 = Cell is S-R latch
 010 = Cell is 4-input AND
 001 = Cell is OR-XOR
 000 = Cell is AND-OR

REGISTER 30-2: CLCxPOL: SIGNAL POLARITY CONTROL REGISTER

| | | | | | | | |
|---------|-----|-----|-----|----------|----------|----------|----------|
| R/W-0/0 | U-0 | U-0 | U-0 | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u |
| LCxPOL | — | — | — | LCxG4POL | LCxG3POL | LCxG2POL | LCxG1POL |
| 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 **LCxPOL:** LCOOUT Polarity Control bit
1 = The output of the logic cell is inverted
0 = The output of the logic cell is not inverted
- bit 6-4 **Unimplemented:** Read as '0'
- bit 3 **LCxG4POL:** Gate 4 Output Polarity Control bit
1 = The output of gate 4 is inverted when applied to the logic cell
0 = The output of gate 4 is not inverted
- bit 2 **LCxG3POL:** Gate 3 Output Polarity Control bit
1 = The output of gate 3 is inverted when applied to the logic cell
0 = The output of gate 3 is not inverted
- bit 1 **LCxG2POL:** Gate 2 Output Polarity Control bit
1 = The output of gate 2 is inverted when applied to the logic cell
0 = The output of gate 2 is not inverted
- bit 0 **LCxG1POL:** Gate 1 Output Polarity Control bit
1 = The output of gate 1 is inverted when applied to the logic cell
0 = The output of gate 1 is not inverted

REGISTER 30-3: CLCxSEL0: MULTIPLEXER DATA 0 SELECT REGISTERS

| 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 |
|-------|-----|-------------|---------|---------|---------|---------|---------|
| — | — | LCxD1S<5:0> | | | | | |
| 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 **Unimplemented:** Read as '0'
bit 5-0 **LCxD1S<5:0>:** Input Data 1 Selection Control bits
See [Table 30-1](#) for signal names associated with inputs.

REGISTER 30-4: CLCxSEL1: MULTIPLEXER DATA 1 SELECT REGISTERS

| 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 |
|-------|-----|-------------|---------|---------|---------|---------|---------|
| — | — | LCxD2S<5:0> | | | | | |
| 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 **Unimplemented:** Read as '0'
bit 5-0 **LCxD2S<5:0>:** Input Data 2 Selection Control bits
See [Table 30-1](#) for signal names associated with inputs.

REGISTER 30-5: CLCxSEL2: MULTIPLEXER DATA 2 SELECT REGISTERS

| 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 |
|-------|-----|-------------|---------|---------|---------|---------|---------|
| — | — | LCxD3S<5:0> | | | | | |
| 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 **Unimplemented:** Read as '0'
bit 5-0 **LCxD3S<5:0>:** Input Data 3 Selection Control bits
See [Table 30-1](#) for signal names associated with inputs.

REGISTER 30-6: CLCxSEL3: MULTIPLEXER DATA 3 SELECT REGISTERS

| | | | | | | | |
|-------|-----|-------------|---------|---------|---------|---------|---------|
| 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 |
| — | — | LCxD4S<5:0> | | | | | |
| 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 **Unimplemented:** Read as '0'

bit 5-0 **LCxD4S<5:0>:** Input Data 4 Selection Control bits
 See [Table 30-1](#) for signal names associated with inputs.

REGISTER 30-7: CLCxGLS0: GATE 1 LOGIC SELECT 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 |
|----------|----------|----------|----------|----------|----------|----------|----------|
| LCxG1D4T | LCxG1D4N | LCxG1D3T | LCxG1D3N | LCxG1D2T | LCxG1D2N | LCxG1D1T | LCxG1D1N |
| 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 **LCxG1D4T:** Gate 1 Data 4 True (non-inverted) bit
1 = lcx4T is gated into lcxg1
0 = lcx4T is not gated into lcxg1
- bit 6 **LCxG1D4N:** Gate 1 Data 4 Negated (inverted) bit
1 = lcx4N is gated into lcxg1
0 = lcx4N is not gated into lcxg1
- bit 5 **LCxG1D3T:** Gate 1 Data 3 True (non-inverted) bit
1 = lcx3T is gated into lcxg1
0 = lcx3T is not gated into lcxg1
- bit 4 **LCxG1D3N:** Gate 1 Data 3 Negated (inverted) bit
1 = lcx3N is gated into lcxg1
0 = lcx3N is not gated into lcxg1
- bit 3 **LCxG1D2T:** Gate 1 Data 2 True (non-inverted) bit
1 = lcx2T is gated into lcxg1
0 = lcx2T is not gated into lcxg1
- bit 2 **LCxG1D2N:** Gate 1 Data 2 Negated (inverted) bit
1 = lcx2N is gated into lcxg1
0 = lcx2N is not gated into lcxg1
- bit 1 **LCxG1D1T:** Gate 1 Data 1 True (non-inverted) bit
1 = lcx1T is gated into lcxg1
0 = lcx1T is not gated into lcxg1
- bit 0 **LCxG1D1N:** Gate 1 Data 1 Negated (inverted) bit
1 = lcx1N is gated into lcxg1
0 = lcx1N is not gated into lcxg1

REGISTER 30-8: CLCxGLS1: GATE 2 LOGIC SELECT 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 |
|----------|----------|----------|----------|----------|----------|----------|----------|
| LCxG2D4T | LCxG2D4N | LCxG2D3T | LCxG2D3N | LCxG2D2T | LCxG2D2N | LCxG2D1T | LCxG2D1N |
| 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 **LCxG2D4T:** Gate 2 Data 4 True (non-inverted) bit
 1 = lcx4T is gated into lcxg2
 0 = lcx4T is not gated into lcxg2
- bit 6 **LCxG2D4N:** Gate 2 Data 4 Negated (inverted) bit
 1 = lcx4N is gated into lcxg2
 0 = lcx4N is not gated into lcxg2
- bit 5 **LCxG2D3T:** Gate 2 Data 3 True (non-inverted) bit
 1 = lcx3T is gated into lcxg2
 0 = lcx3T is not gated into lcxg2
- bit 4 **LCxG2D3N:** Gate 2 Data 3 Negated (inverted) bit
 1 = lcx3N is gated into lcxg2
 0 = lcx3N is not gated into lcxg2
- bit 3 **LCxG2D2T:** Gate 2 Data 2 True (non-inverted) bit
 1 = lcx2T is gated into lcxg2
 0 = lcx2T is not gated into lcxg2
- bit 2 **LCxG2D2N:** Gate 2 Data 2 Negated (inverted) bit
 1 = lcx2N is gated into lcxg2
 0 = lcx2N is not gated into lcxg2
- bit 1 **LCxG2D1T:** Gate 2 Data 1 True (non-inverted) bit
 1 = lcx1T is gated into lcxg2
 0 = lcx1T is not gated into lcxg2
- bit 0 **LCxG2D1N:** Gate 2 Data 1 Negated (inverted) bit
 1 = lcx1N is gated into lcxg2
 0 = lcx1N is not gated into lcxg2

REGISTER 30-9: CLCxGLS2: GATE 3 LOGIC SELECT 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 |
|----------|----------|----------|----------|----------|----------|----------|----------|
| LCxG3D4T | LCxG3D4N | LCxG3D3T | LCxG3D3N | LCxG3D2T | LCxG3D2N | LCxG3D1T | LCxG3D1N |
| 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 | LCxG3D4T: Gate 3 Data 4 True (non-inverted) bit 1 = lcx4T is gated into lcxg3 0 = lcx4T is not gated into lcxg3 |
| bit 6 | LCxG3D4N: Gate 3 Data 4 Negated (inverted) bit 1 = lcx4N is gated into lcxg3 0 = lcx4N is not gated into lcxg3 |
| bit 5 | LCxG3D3T: Gate 3 Data 3 True (non-inverted) bit 1 = lcx3T is gated into lcxg3 0 = lcx3T is not gated into lcxg3 |
| bit 4 | LCxG3D3N: Gate 3 Data 3 Negated (inverted) bit 1 = lcx3N is gated into lcxg3 0 = lcx3N is not gated into lcxg3 |
| bit 3 | LCxG3D2T: Gate 3 Data 2 True (non-inverted) bit 1 = lcx2T is gated into lcxg3 0 = lcx2T is not gated into lcxg3 |
| bit 2 | LCxG3D2N: Gate 3 Data 2 Negated (inverted) bit 1 = lcx2N is gated into lcxg3 0 = lcx2N is not gated into lcxg3 |
| bit 1 | LCxG3D1T: Gate 3 Data 1 True (non-inverted) bit 1 = lcx1T is gated into lcxg3 0 = lcx1T is not gated into lcxg3 |
| bit 0 | LCxG3D1N: Gate 3 Data 1 Negated (inverted) bit 1 = lcx1N is gated into lcxg3 0 = lcx1N is not gated into lcxg3 |

REGISTER 30-10: CLCxGLS3: GATE 4 LOGIC SELECT 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 |
|----------|----------|----------|----------|----------|----------|----------|----------|
| LCxG4D4T | LCxG4D4N | LCxG4D3T | LCxG4D3N | LCxG4D2T | LCxG4D2N | LCxG4D1T | LCxG4D1N |
| 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 **LCxG4D4T:** Gate 4 Data 4 True (non-inverted) bit
1 = lcx4T is gated into lcxg4
0 = lcx4T is not gated into lcxg4
- bit 6 **LCxG4D4N:** Gate 4 Data 4 Negated (inverted) bit
1 = lcx4N is gated into lcxg4
0 = lcx4N is not gated into lcxg4
- bit 5 **LCxG4D3T:** Gate 4 Data 3 True (non-inverted) bit
1 = lcx3T is gated into lcxg4
0 = lcx3T is not gated into lcxg4
- bit 4 **LCxG4D3N:** Gate 4 Data 3 Negated (inverted) bit
1 = lcx3N is gated into lcxg4
0 = lcx3N is not gated into lcxg4
- bit 3 **LCxG4D2T:** Gate 4 Data 2 True (non-inverted) bit
1 = lcx2T is gated into lcxg4
0 = lcx2T is not gated into lcxg4
- bit 2 **LCxG4D2N:** Gate 4 Data 2 Negated (inverted) bit
1 = lcx2N is gated into lcxg4
0 = lcx2N is not gated into lcxg4
- bit 1 **LCxG4D1T:** Gate 4 Data 1 True (non-inverted) bit
1 = lcx1T is gated into lcxg4
0 = lcx1T is not gated into lcxg4
- bit 0 **LCxG4D1N:** Gate 4 Data 1 Negated (inverted) bit
1 = lcx1N is gated into lcxg4
0 = lcx1N is not gated into lcxg4

REGISTER 30-11: CLCDATA: CLC DATA OUTPUT

| | | | | | | | |
|-------|-----|-----|-----|---------|---------|---------|---------|
| U-0 | U-0 | U-0 | U-0 | R-0 | R-0 | R-0 | R-0 |
| — | — | — | — | MLC4OUT | MLC3OUT | MLC2OUT | MLC1OUT |
| 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 | MLC4OUT: Mirror copy of LC4OUT bit |
| bit 2 | MLC3OUT: Mirror copy of LC3OUT bit |
| bit 1 | MLC2OUT: Mirror copy of LC2OUT bit |
| bit 0 | MLC1OUT: Mirror copy of LC1OUT bit |

TABLE 30-3: SUMMARY OF REGISTERS ASSOCIATED WITH CLCx

| Name | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | Register on Page |
|----------|----------------------|----------------------|-------------|----------|----------|--------------|----------|----------|------------------|
| ANSELA | — | — | — | ANSA4 | — | ANSA2 | ANSA1 | ANSA0 | 152 |
| ANSELB | — | — | ANSB5 | ANSB4 | — | — | — | — | 159 |
| ANSELC | ANSC7 ⁽²⁾ | ANSC6 ⁽²⁾ | — | — | ANSC3 | ANSC2 | ANSC1 | ANSC0 | 166 |
| CLC1CON | LC1EN | — | LC1OUT | LC1INTP | LC1INTN | LC1MODE<2:0> | | | 443 |
| CLCDATA | — | — | — | — | MLC4OUT | MLC3OUT | MLC2OUT | MLC1OUT | 450 |
| CLC1GLS0 | LC1G1D4T | LC1G1D4N | LC1G1D3T | LC1G1D3N | LC1G1D2T | LC1G1D2N | LC1G1D1T | LC1G1D1N | 446 |
| CLC1GLS1 | LC1G2D4T | LC1G2D4N | LC1G2D3T | LC1G2D3N | LC1G2D2T | LC1G2D2N | LC1G2D1T | LC1G2D1N | 447 |
| CLC1GLS2 | LC1G3D4T | LC1G3D4N | LC1G3D3T | LC1G3D3N | LC1G3D2T | LC1G3D2N | LC1G3D1T | LC1G3D1N | 448 |
| CLC1GLS3 | LC1G4D4T | LC1G4D4N | LC1G4D3T | LC1G4D3N | LC1G4D2T | LC1G4D2N | LC1G4D1T | LC1G4D1N | 449 |
| CLC1POL | LC1POL | — | — | — | LC1G4POL | LC1G3POL | LC1G2POL | LC1G1POL | 444 |
| CLC1SEL0 | — | — | LC1D1S<5:0> | | | | | | 445 |
| CLC1SEL1 | — | — | LC1D2S<5:0> | | | | | | 445 |
| CLC1SEL2 | — | — | LC1D3S<5:0> | | | | | | 453 |
| CLC1SEL3 | — | — | LC1D4S<5:0> | | | | | | 454 |
| CLC2CON | LC2EN | — | LC2OUT | LC2INTP | LC2INTN | LC2MODE<2:0> | | | 443 |
| CLC2GLS0 | LC2G1D4T | LC2G1D4N | LC2G1D3T | LC2G1D3N | LC2G1D2T | LC2G1D2N | LC2G1D1T | LC2G1D1N | 446 |
| CLC2GLS1 | LC2G2D4T | LC2G2D4N | LC2G2D3T | LC2G2D3N | LC2G2D2T | LC2G2D2N | LC2G2D1T | LC2G2D1N | 447 |
| CLC2GLS2 | LC2G3D4T | LC2G3D4N | LC2G3D3T | LC2G3D3N | LC2G3D2T | LC2G3D2N | LC2G3D1T | LC2G3D1N | 448 |
| CLC2GLS3 | LC2G4D4T | LC2G4D4N | LC2G4D3T | LC2G4D3N | LC2G4D2T | LC2G4D2N | LC2G4D1T | LC2G4D1N | 449 |
| CLC2POL | LC2POL | — | — | — | LC2G4POL | LC2G3POL | LC2G2POL | LC2G1POL | 444 |
| CLC2SEL0 | — | — | LC2D1S<5:0> | | | | | | 445 |
| CLC2SEL1 | — | — | LC2D2S<5:0> | | | | | | 445 |
| CLC2SEL2 | — | — | LC2D3S<5:0> | | | | | | 453 |
| CLC2SEL3 | — | — | LC2D4S<5:0> | | | | | | 454 |
| CLC3CON | LC3EN | — | LC3OUT | LC3INTP | LC3INTN | LC3MODE<2:0> | | | 451 |
| CLC3GLS0 | LC3G1D4T | LC3G1D4N | LC3G1D3T | LC3G1D3N | LC3G1D2T | LC3G1D2N | LC3G1D1T | LC3G1D1N | 455 |

Legend: — = unimplemented read as '0'. Shaded cells are not used for CLC module.

Note 1: Unimplemented, read as '1'.

Note 2: PIC16(L)F1619 only.

TABLE 30-3: SUMMARY OF REGISTERS ASSOCIATED WITH CLCx (continued)

| Name | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | Register on Page |
|----------|-----------------------|-----------------------|-------------|----------|------------------|--------------|----------|----------|------------------|
| CLC3GLS1 | LC3G2D4T | LC3G2D4N | LC3G2D3T | LC3G2D3N | LC3G2D2T | LC3G2D2N | LC3G2D1T | LC3G2D1N | 456 |
| CLC3GLS2 | LC3G3D4T | LC3G3D4N | LC3G3D3T | LC3G3D3N | LC3G3D2T | LC3G3D2N | LC3G3D1T | LC3G3D1N | 457 |
| CLC3GLS3 | LC3G4D4T | LC3G4D4N | LC3G4D3T | LC3G4D3N | LC3G4D2T | LC3G4D2N | LC3G4D1T | LC3G4D1N | 458 |
| CLC3POL | LC3POL | — | — | — | LC3G4POL | LC3G3POL | LC3G2POL | LC3G1POL | 452 |
| CLC3SEL0 | — | — | LC3D1S<5:0> | | | | | | 453 |
| CLC3SEL1 | — | — | LC3D2S<5:0> | | | | | | 453 |
| CLC3SEL2 | — | — | LC3D3S<5:0> | | | | | | 453 |
| CLC3SEL3 | — | — | LC3D4S<5:0> | | | | | | 454 |
| CLC4CON | LC4EN | — | LC4OUT | LC4INTP | LC4INTN | LC4MODE<2:0> | | | 451 |
| CLC4GLS0 | LC4G1D4T | LC4G1D4N | LC4G1D3T | LC4G1D3N | LC4G1D2T | LC4G1D2N | LC4G1D1T | LC4G1D1N | 455 |
| CLC4GLS1 | LC4G2D4T | LC4G2D4N | LC4G2D3T | LC4G2D3N | LC4G2D2T | LC4G2D2N | LC4G2D1T | LC4G2D1N | 456 |
| CLC4GLS2 | LC4G3D4T | LC4G3D4N | LC4G3D3T | LC4G3D3N | LC4G3D2T | LC4G3D2N | LC4G3D1T | LC4G3D1N | 457 |
| CLC4GLS3 | LC4G4D4T | LC4G4D4N | LC4G4D3T | LC4G4D3N | LC4G4D2T | LC4G4D2N | LC4G4D1T | LC4G4D1N | 458 |
| CLC4POL | LC4POL | — | — | — | LC4G4POL | LC4G3POL | LC4G2POL | LC4G1POL | 452 |
| CLC4SEL0 | — | — | LC4D1S<5:0> | | | | | | 453 |
| CLC4SEL1 | — | — | LC4D2S<5:0> | | | | | | 453 |
| CLC4SEL2 | — | — | LC4D3S<5:0> | | | | | | 453 |
| CLC4SEL3 | — | — | LC4D4S<5:0> | | | | | | 454 |
| INTCON | GIE | PEIE | TMR0IE | INTE | IOCFIE | TMR0IF | INTF | IOCFIF | 97 |
| PIE3 | — | — | CWGIE | ZCDIE | CLC4IE | CLC3IE | CLC2IE | CLC1IE | 108 |
| PIR3 | — | — | CWGIF | ZCDIF | CLC4IF | CLC3IF | CLC2IF | CLC1IF | 113 |
| TRISA | — | — | TRISA5 | TRISA4 | — ⁽¹⁾ | TRISA2 | TRISA1 | TRISA0 | 151 |
| TRISB | TRISB7 | TRISB6 | TRISB5 | TRISB4 | — | — | — | — | 158 |
| TRISC | TRISC7 ⁽²⁾ | TRISC6 ⁽²⁾ | TRISC5 | TRISC4 | TRISC3 | TRISC2 | TRISC1 | TRISC0 | 165 |

Legend: — = unimplemented read as '0'. Shaded cells are not used for CLC module.

Note 1: Unimplemented, read as '1'.

2: PIC16(L)F1619 only.

31.0 ANGULAR TIMER (AT) MODULE

The Angular Timer (AT) module subdivides periodic signals into smaller equally spaced intervals, the number of which remain constant as the periodic signal frequency changes. A counter tracks the intervals starting at zero at each period event. The counter can be compared to user defined values to cause events, or the counter value can be captured by events external to the module. This allows for a variety of applications, such as measuring of A/C mains, stall detection for motors, commutation for brushless motors, and TDC detection for internal combustion engines. A second counter tracks the period time. This can be used to measure the error of the period based on a pre-programmed set point, as well as detect missing pulses in the signal. The angular timer includes the following features:

- Two operating modes
 - Single-pulse per period
 - Multiple-pulses per period
- Two missing pulse modes
 - Adaptive
 - Fixed
- Multiple selectable clock sources
- Phase clock output with polarity control
- Period clock output with polarity control
- Missing pulse output with polarity control
- Interrupts for phase and period clock generation, as well as for missing pulse detect
- Period set point and error register
- Compare-pulse outputs
 - Independent interrupts
- Capture inputs
 - Input polarity control
 - Independent interrupts

31.1 Principle of Operation

Consider the statements in [Equation 31-1](#):

EQUATION 31-1:

| | | |
|-------------------|-------------------|---------|
| If: | And: | Then: |
| $P = \frac{F}{R}$ | $A = \frac{F}{P}$ | $A = R$ |

In these three equations:

- P represents the period count ATxPER
- A represents the angle or phase count ATxPHS
- R represents the desired resolution ATxRES
- F represents some arbitrary scaler value

Notice that the phase count equals the desired resolution regardless of what F is. If we let F equal the ratio of a system clock to the input signal frequency then that means the phase count is a constant equaling the desired resolution regardless of the input frequency. This has many extraordinary uses including:

- Use phase compare feature to create an event at a fixed phase angle in the period
- Use capture feature to capture the phase angle at which an event occurs
- Use error feature to monitor deviations from a user specified period time

The details of these features, and more, are described in the following sections of this chapter.

31.2 Angular Timer Operating Modes

The AT module operates in two basic modes:

- Single-Pulse mode
- Multi-Pulse mode

Both modes function on the same principle: Dividing a periodic input signal into intervals, and allowing events to trigger off of these smaller intervals. The primary difference between these two modes is how the period is determined. The Single-Pulse mode determines the period as the time between every pulse in the input pulse stream. The Multi-Pulse mode determines the period as the time between missing pulses in the input pulse stream.

The primary parameter for both modes is the ATxRES register pair. This value is used to determine the granularity of the phase counter and the frequency of the phase clock output of the module.

31.2.1 SINGLE-PULSE MODE

The operation of Single-Pulse mode is illustrated in [Figure 31-1](#). The calculations on the input signal are done in a few distinct steps. First, there is a divider that divides the module clock by the ATxRES register pair and uses the resulting signal to increment a period counter. This operation is expressed by [Equation 31-2](#). This equation differs slightly from that of [Equation 31-1](#) because the counters include the count of zero. To compensate for this, the number written to the resolution register, ATxRES, must be one less than the desired resolution.

EQUATION 31-2:

$$ATxPER = \frac{F(ATxclk)}{F(ATxsig)} \cdot (ATxRES + 1)$$

Variables in [Equation 31-2](#) are as follows:

- ATxPER is the value of the period counter latched by the input signal.
- ATxRES is the user-specified resolution. The phase counter will count up to this value.
- F(ATxclk) is the ATx clock frequency.
- F(ATxsig) is the input signal frequency.

The second step in the angular timer's operation is the creation of the phase clock, which is also illustrated in [Figure 31-1](#). The input clock is divided by the ATxPER value, latched-in during the previous step, and the resulting signal is used to increment the phase counter. This signal also is used as the phase clock output, and for setting the PHSIF interrupt flag bit of the ATxIR0 register. The result is that the phase counter counts from zero to a final value expressed in [Equation 31-3](#), outputting a pulse each time the counter increments. The value of the phase counter can be accessed by software by reading the ATxPHS register pair. However, because of the synchronization required, in order for reads of this register pair to be accurate, the instruction clock (FOSC/4) needs to be at least 3x the ATx_phsclk output frequency.

EQUATION 31-3:

$$ATxPHS(final) = \frac{\left(\frac{F(ATxclk)}{F(ATxsig)}\right)}{(ATxPER + 1)}$$

The variables in [Equation 31-3](#) are as follows:

- ATxPHS(final) is the maximum value that the phase counter will reach before being reset by the input signal. As noted in [Equation 31-1](#), this will equal ATxRES.
- ATxPER is the maximum value of the period counter.
- F(ATxclk) is the ATx clock frequency.
- F(ATxsig) is the input signal frequency.

Notice that the division is ATxPER + 1. Ideally, this would be just ATxPER but the divider includes zero in the count. In most applications, ATxPER is a large number so the error introduced by adding one is negligible.

ATxPHS counting from 0 to ATxRES is useful when the input signal represents a rotation (for example, a motor or A/C mains). In this case, the input signal is understood to provide a period pulse every 360 degrees. Since the phase clock equally divides the signal period into a number of intervals determined by the ATxRES register pair, each pulse on the phase clock output marks a fixed phase angle in that rotation, as expressed by [Equation 31-4](#).

EQUATION 31-4:

$$AngleResolution = \frac{360degrees}{ATxRES + 1}$$

ATxRES can then be used with the instantaneous value of the ATxPHS register pair to get the instantaneous angle of the rotation using [Equation 31-5](#).

EQUATION 31-5:

$$Angle = 360degrees \cdot \frac{ATxPHS}{ATxRES + 1}$$

31.2.2 MULTI-PULSE MODE

The operation of Multi-Pulse mode is illustrated in [Figure 31-3](#). The calculations on the input signal are similar to those in Single-Pulse mode, with the primary difference relating to when the ATxPHS register pair is reset.

The period counter is latched into the ATxPER register pair and reset on every input pulse except the pulse immediately following a missing pulse. The first active pulse following a missing pulse triggers all of the following:

- Period clock output
- PERIF interrupt
- Phase counter reset

The result is a period clock output that has a period length equal to the time between missing pulses (e.g., a missing tooth in a gear). This leads to a significantly different relation between ATxRES and the maximum phase count, ATxPHS, as shown in [Equation 31-6](#).

EQUATION 31-6:

$$ATxPHS(final) = ATxRES \left(\frac{MissP}{PulseP} \right)$$

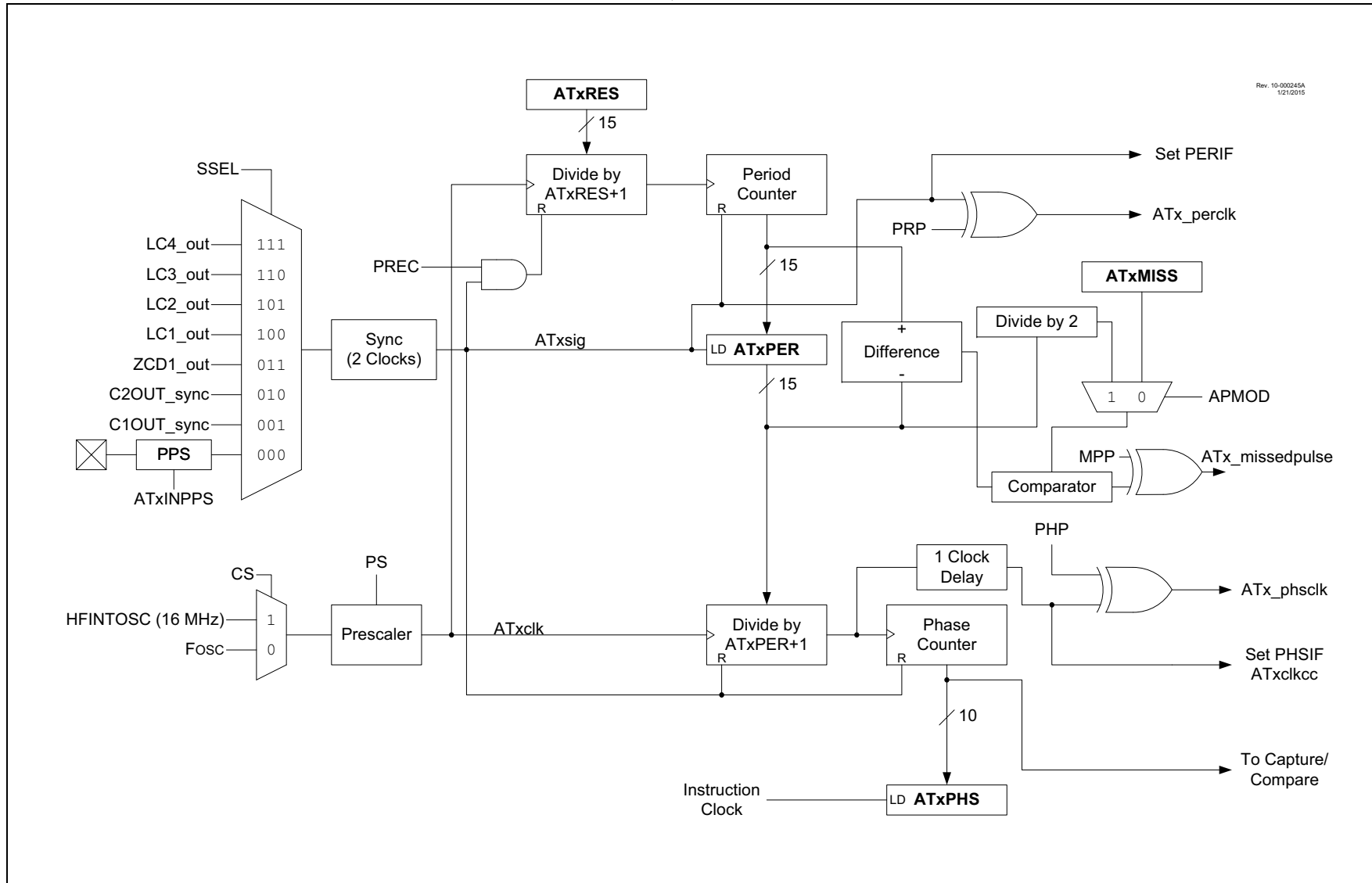
The variables in [Equation 31-6](#) are as follows:

- MissP is the period between missing pulses
- PulseP is the period between input pulses
- ATxPHS(final) is the maximum value of the phase counter

This results in a phase clock output that pulses ATxRES+1 times every input pulse, and a phase counter that increments from 0 to ATxPHS(final) over the entire time between the missing pulses.

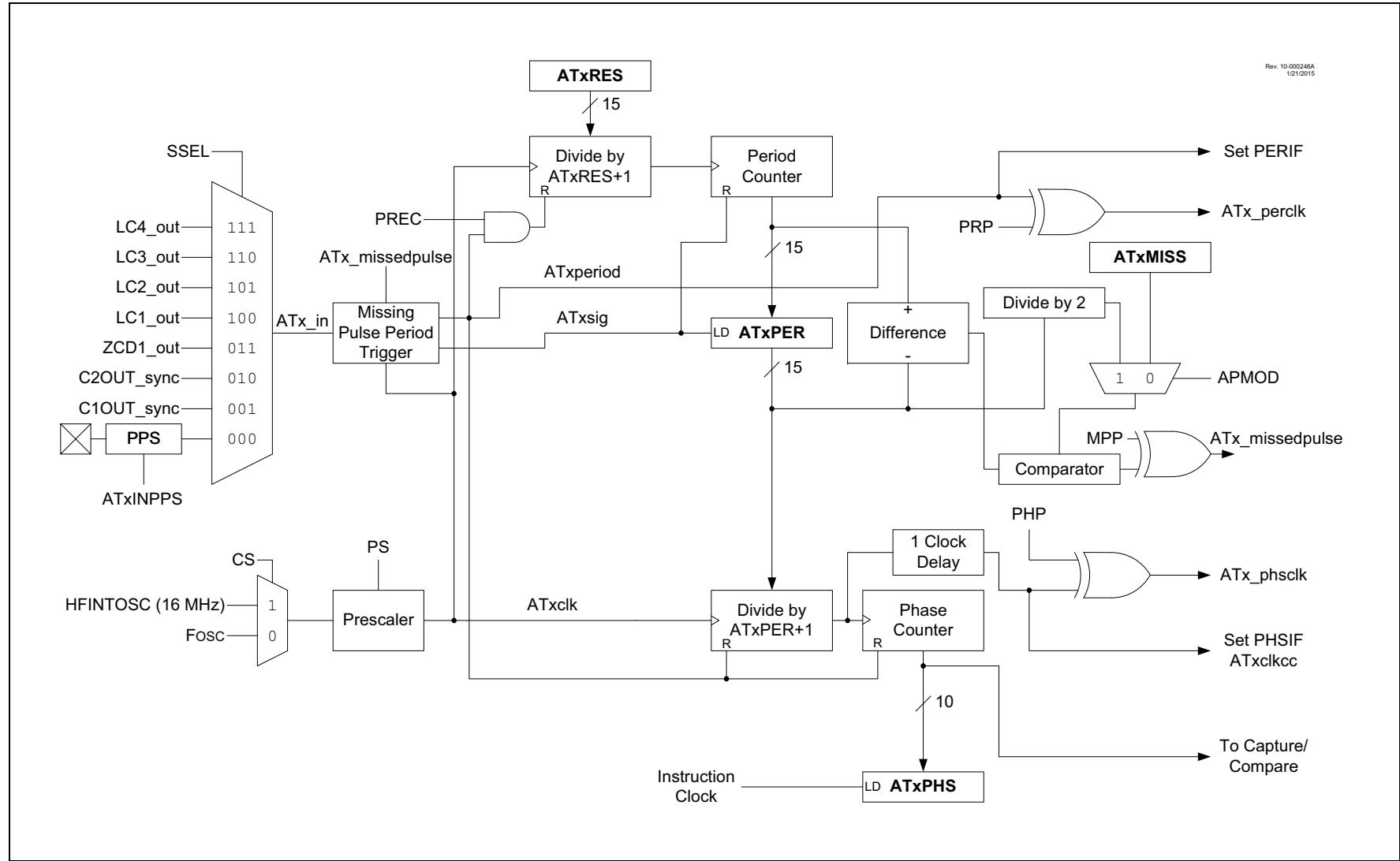
Similar to Single-Pulse mode, this allows for triggered events to occur at fixed phase angles in the signal's period where the period is defined as the time between missing pulses. An example of multi-pulse operation is illustrated in the timing diagram of [Figure 31-5](#), which also demonstrates what happens as a result of variations in the input signal period.

FIGURE 31-1: ANGULAR TIMER SIMPLIFIED BLOCK DIAGRAM, SINGLE-PULSE MODE



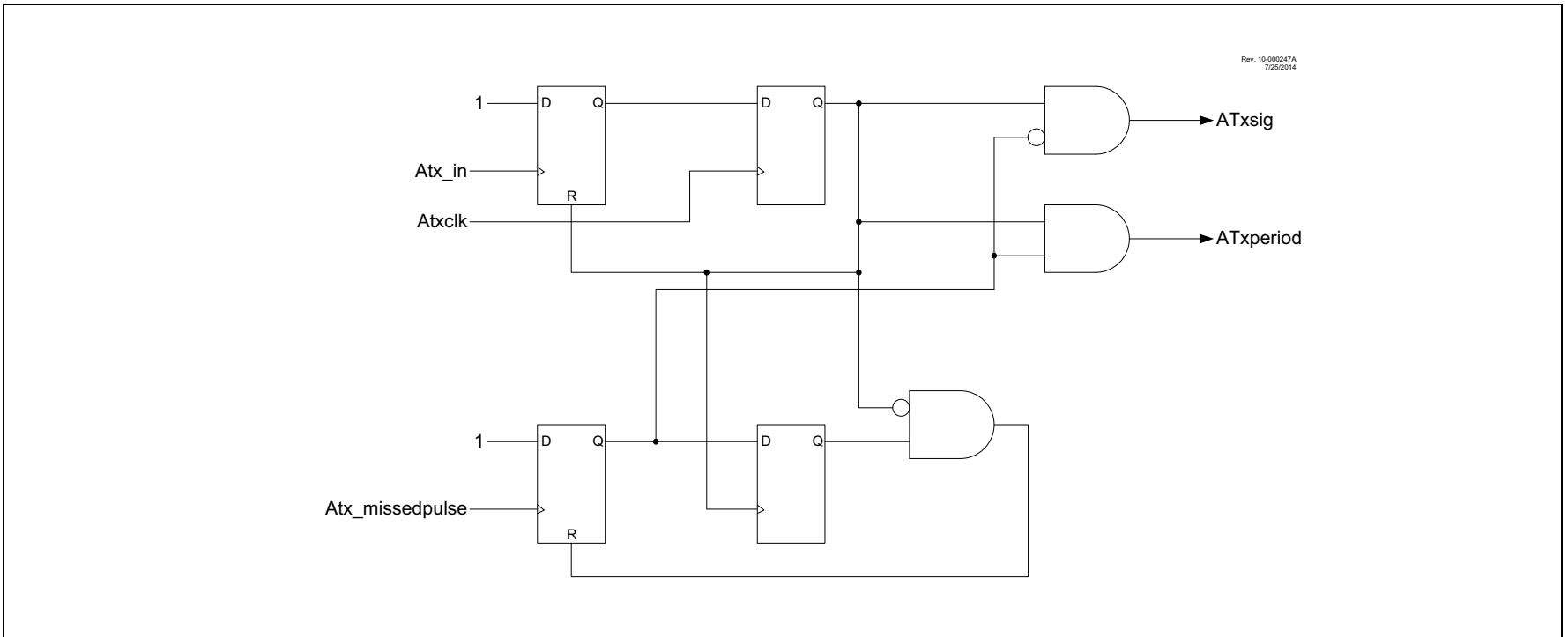
Rev. 10-000245A
1/21/2015

FIGURE 31-2: ANGULAR TIMER SIMPLIFIED BLOCK DIAGRAM, MULTI-PULSE MODE



Rev. 10-000248A
1/21/2015

FIGURE 31-3: ANGULAR TIMER SIMPLIFIED MULTI-PULSE PERIOD TRIGGER BLOCK DIAGRAM



31.2.3 MISSING PULSE DETECTION

In both Single-Pulse and Multi-Pulse modes, the AT module monitors for missing pulses in the following manner. The latched value of the ATxPER register pair is continuously subtracted from the value of the period counter as it counts up. The result of this subtraction is compared to a third value and a missing pulse event is generated when the comparison is equal.

The third value is either the ATxMISS register pair or the ATxPER register pair divided by two. The APMOD bit of ATxCON0 register ([Register 31-1](#)) selects which of these two values is used.

In Single-Pulse mode, a missing pulse event generates the missing pulse output of the module as well as triggering the MISSIF interrupt.

In Multi-Pulse mode, a missing pulse event generates the output and interrupt, and is also used to determine the period signal timing.

31.2.4 MISSING PULSE MODES

Missing pulse detection has two modes of operation selected with the APMOD bit of the ATxCON0 register:

- Adaptive
- Fixed

31.2.4.1 Adaptive Missing Pulse Mode

When $APMOD = 1$, the missing pulse detection is in the Adaptive mode. In Adaptive mode, the difference between the period counter and the latched ATxPER value is compared to the latched ATxPER value divided by two. A missing pulse event will occur when an input signal pulse is not detected within 1.5 times the previous time between pulses. If the signal input period changes, the missing pulse comparison adapts to the change to maintain the relative time to the missing pulse event at 1.5 times the previous pulse interval.

31.2.4.2 Fixed Missing Pulse Mode

When $APMOD = 0$, the missing pulse detection is in the Fixed mode. In Fixed mode, the difference between the period counter and the latched ATxPER value is compared to the value in the ATxMISS register pair. This gives the user absolute control over when the missing pulse will be detected, with the trade-off of not being adaptive to changes in the period.

FIGURE 31-4: TIMING DIAGRAM FOR SINGLE PULSE MODE

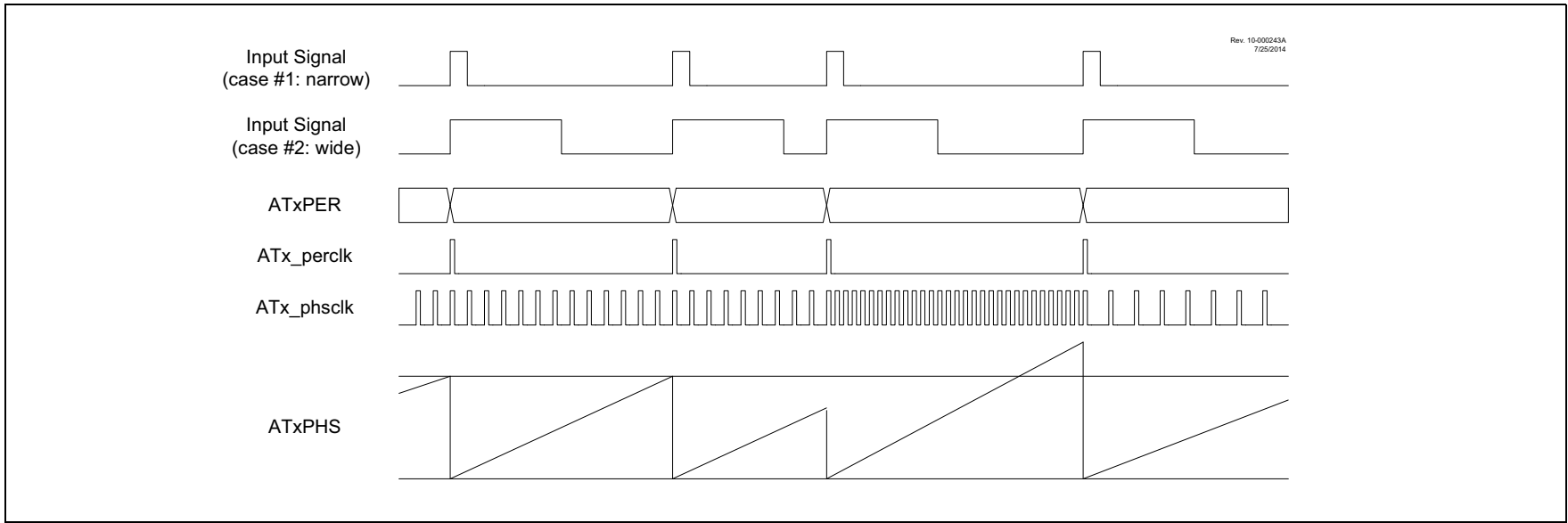
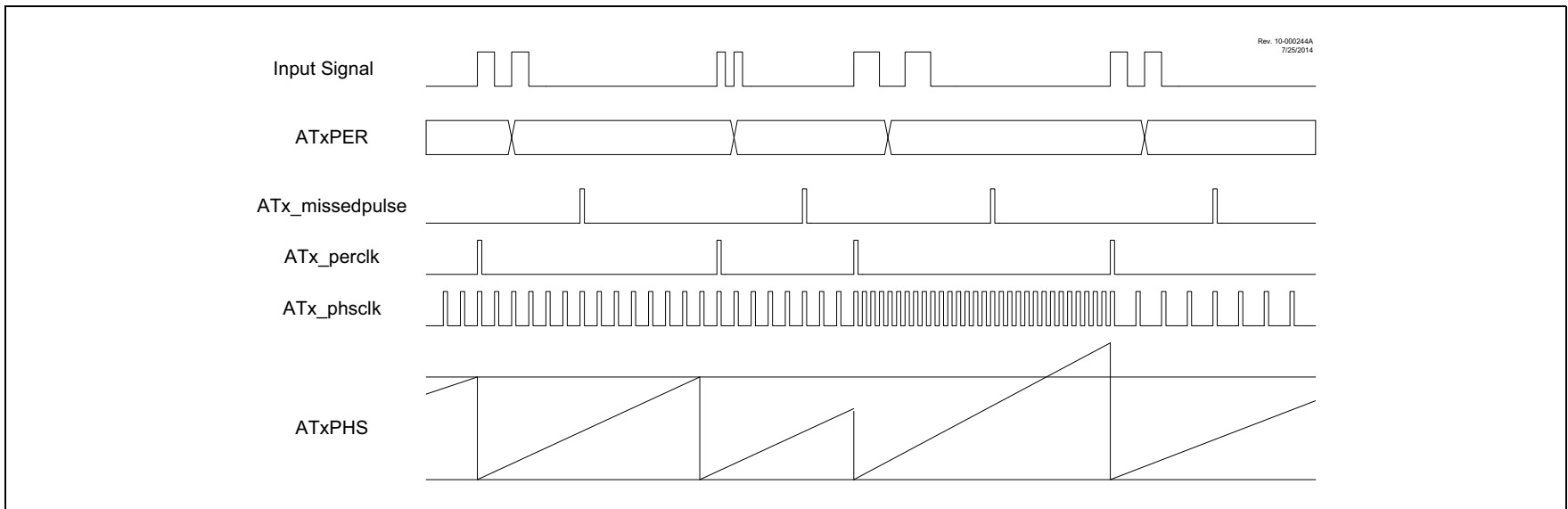


FIGURE 31-5: TIMING DIAGRAM FOR MULTI-PULSE MODE



31.2.5 VALID BIT

Several values used by the AT module must be calculated from external signals. As such, these values may be inaccurate for a period of time after the angular timer starts up. Because of this, the module will not output signals or trigger interrupts for a period of time after the module is enabled, or under certain other conditions that might jeopardize accurate output values. This output inhibition is indicated by the read-only VALID bit of the ATxCON1 being clear.

The following cases will clear the VALID bit in hardware:

- Any write to ATxRES register pair
- Phase counter overflow (ATxPHS register pair) clocked beyond 0x3FF)
- In-Circuit Debugger halt
- EN = 0
- ATxPER register pair = 0
- Device Reset

As long as the VALID bit is cleared, the following occurs:

- Period clock is not output and associated interrupts do not trigger.
- Missed pulse is not output and associated interrupts do not trigger.
- Phase clock is not output and associated interrupts do not trigger.
- Phase counter does not increment.
- Capture logic does not function and associated interrupts do not trigger.
- Compare logic does not function and associated interrupts do not trigger.
- Every ATxsig edge latches the period counter into the ATxPER register pair, regardless of mode.

In single-pulse modes, the VALID bit becomes set upon the 3rd active input edge of the signal that latches the ATxPER register pair. In multi-pulse modes, a missing pulse trigger is also required, ensuring that at least one full revolution of the input has occurred.

An example of the VALID bit in Single-Pulse mode is shown in [Figure 31-6](#).

31.2.6 DETERMINING ACCURACY

The ATxRES register pair determines the resolution of the period measurement and, by extension, the maximum value that the phase counter reaches at the end of each input signal period. The interim value, ATxPER, used to derive the phase counter is, by nature of the circuitry, an integer. The ratio of the integer value obtained by the circuit and the calculated floating point value is the inherent error of the measurement. When ATxRES is small then integer rounding results in large errors. Factors that contribute to large errors include:

- Large values for ATxRES
- Relatively low ATxclk frequency
- Relatively high ATxsig input frequency

The actual error can be determined with [Equation 31-7](#).

EQUATION 31-7:

$$period = \frac{F(ATxclk)}{F(ATxsig) \cdot (ATxRES + 1)}$$

$$error\% = 100 \cdot \left(\frac{period - int(period + 1)}{period} \right)$$

31.3 Input and Clock Selection

The input clock for the AT module can come from either the FOSC system clock or the 16 MHz HFINTOSC, and is chosen by the CS0 bit of the ATxCLK register. In addition, the clock is run through a prescaler that can be /1, /2, /4, or /8, which is configured by the PS<1:0> bits of the ATxCON0 register. This prescaled clock is then used for all clock operations of the Angular Timer, and as such, should be used for all of the equations demonstrated above determining the Angular Timer's behavior.

The input signal for the AT module can come from a variety of sources. The source is selected by the SSEL bits of the ATxSIG register ([Register 31-4](#)).

31.4 Module Outputs

31.4.1 ANGLE/PHASE CLOCK OUTPUT

The angle/phase clock signal (ATx_phsclk) can be used by the CLC as an input signal to combinational logic. The polarity of this signal is configured by the PHP bit of the ATxCON1 register.

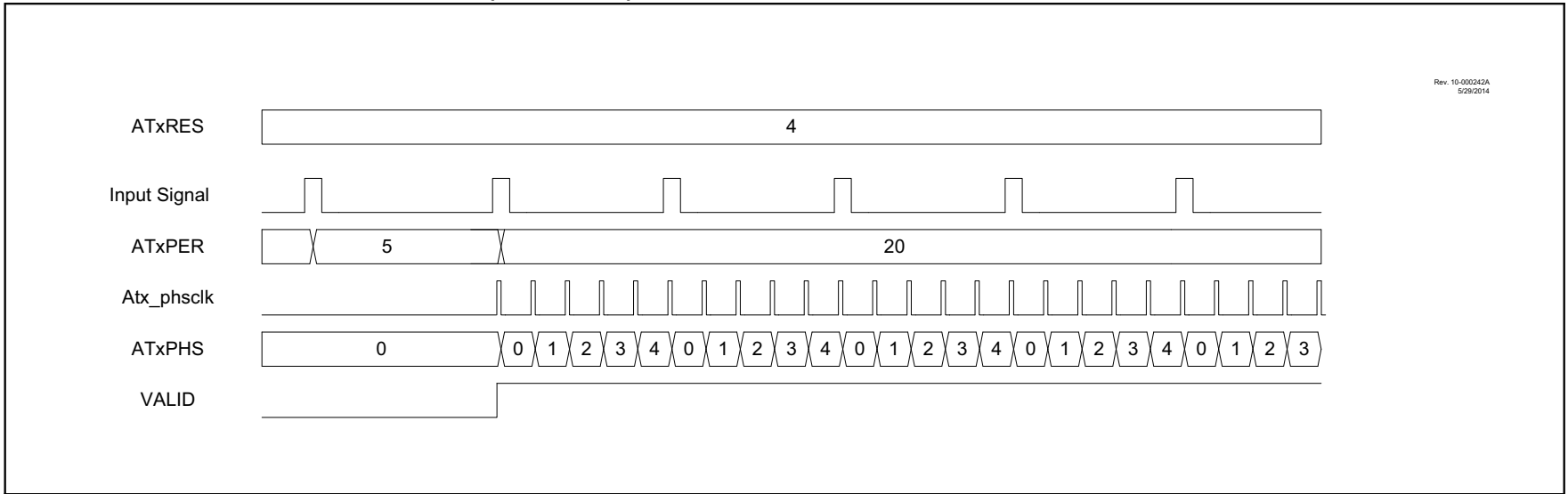
31.4.2 PERIOD CLOCK OUTPUT

The period clock signal (ATx_perclk) can be used as an input clock for the Timer2/4/6 and Signal Measurement module, as well as an input signal to the CLC for combinational logic. The polarity of this signal is configured by the PRP bit of the ATxCON1 register ([Register 31-2](#)).

31.4.3 MISSED PULSE OUTPUT

The missed pulse signal (ATx_missedpulse) can be used by the CLC as an input signal to combinational logic. The polarity of this signal is configured by the MPP bit of the ATxCON1 register.

FIGURE 31-6: EXAMPLE OPERATION (ATxRES = 4)

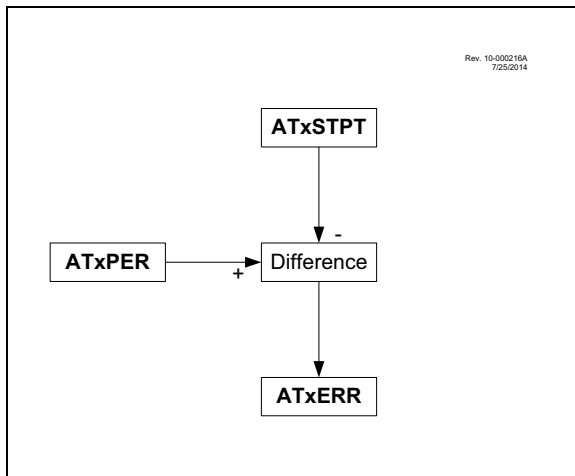


31.5 Period Set Point and Error Measurement

The ATxSTPT register pair controls the period set point of the AT module. The signal period captured in the ATxPER register pair at every signal input pulse. The unsigned 15-bit ATxSTPT value is subtracted from the unsigned 15-bit ATxPER value and the signed 16-bit result is placed in the ATxERR register pair.

The ATxSTPT value is double buffered requiring an ATxSTPTL value write for the ATxSTPTH value to take effect. This is done so that all 16 bits update at the same time, thereby avoiding a miscalculation of the error.

FIGURE 31-7: ANGULAR TIMER SET POINT CALCULATION BLOCK DIAGRAM



31.6 Capture and Compare Functions

The angular timer contains multiple built-in capture/compare modules. These are controlled by their respective ATxCCONy registers where “x” refers to the AT instance and “y” refers to the Capture/Compare instance within that AT module.

This particular device contains three capture/compare modules within the AT module. The CCyMODE bit of the ATxCCONy register controls whether each particular module is in Capture or Compare mode. The polarity of each module’s respective output signal is controlled by the CCyPOL bit of the ATxCCONy register (Register 31-21). Both the Capture and Compare modes use an edge detect that runs off of the ATxclk signal.

31.6.1 CAPTURE MODE

Capture mode is selected when the CCyMODE bit (of the ATxCCONy register) = 1. Refer to Figure 31-8.

In Capture mode, the value of the phase counter is written to the respective ATxCCy registers on the rising edge of the capture input signal.

The capture event also generates a pulse that can be used for the following:

- Trigger an ADC reading
- CLC logic input
- Set the CCyIF bit

See Section 31.7 “Interrupts” for more details on the interrupts triggered by the AT module.

The capture input signal source is selected by the capture/compare’s respective ATxCSELY register (Register 31-22), and its polarity is selected by the ATxCAPyP bit of the ATxCCONy register (Register 31-21). Note that when in Capture mode, the ATxCCy register pair is read-only.

31.6.2 COMPARE MODE

Compare mode is selected when the CCyMODE bit (of the ATxCCONy register) = 0. Refer to Figure 31-9.

In Compare mode, the module compares the current value in the ATxCCy register pair to the phase counter value. When the two values are equal then a compare event is generated and output to the following:

- Trigger an ADC reading
- CLC logic input
- Set the CCyIF bit

See Section 31.7 “Interrupts” for more details on the interrupts triggered by the AT module.

FIGURE 31-8: ANGULAR TIMER CAPTURE/COMPARE UNIT BLOCK DIAGRAM: CAPTURE MODE

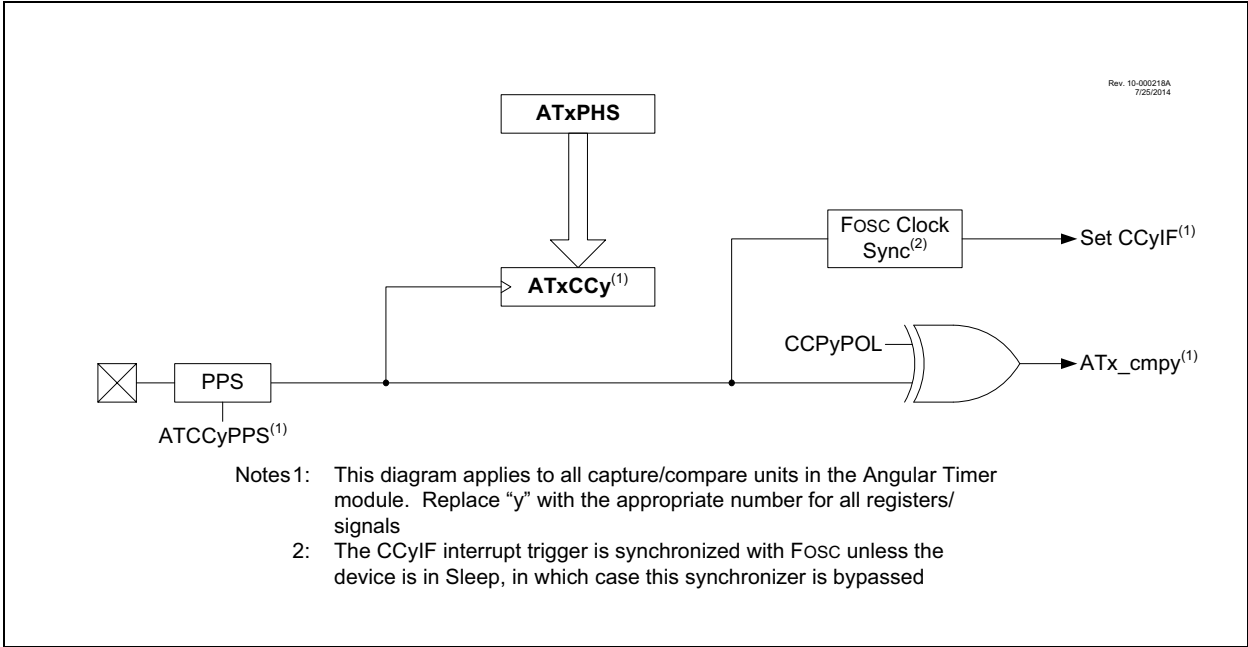
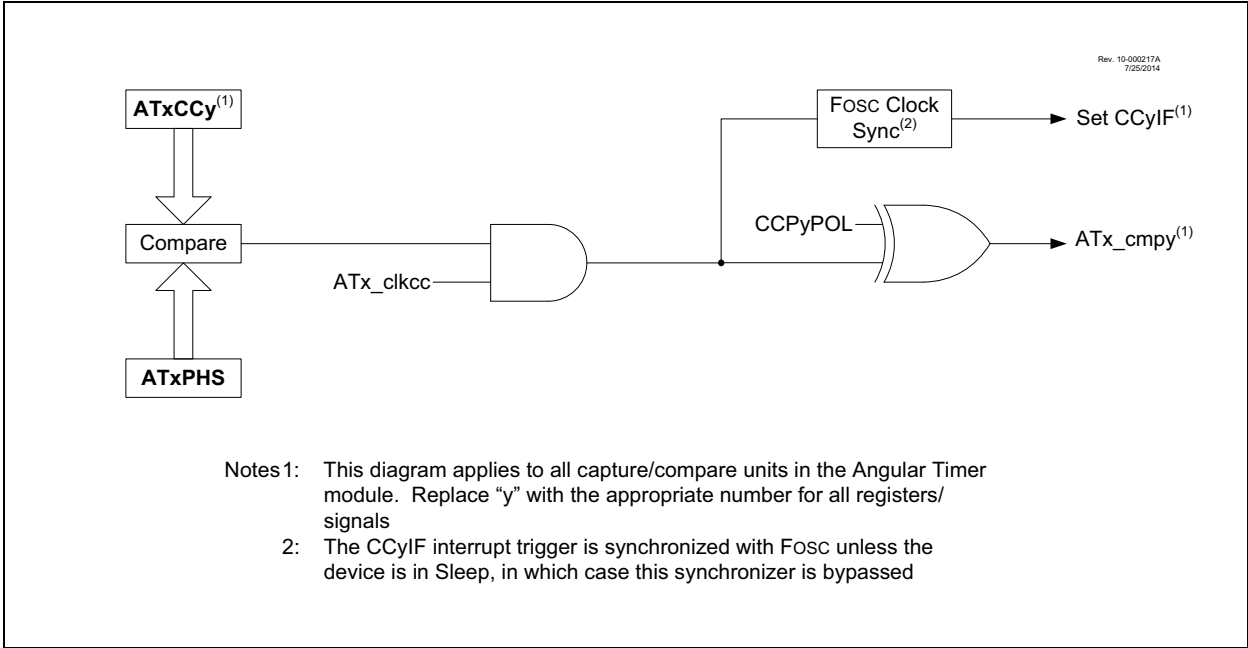


FIGURE 31-9: ANGULAR TIMER CAPTURE/COMPARE UNIT BLOCK DIAGRAM: COMPARE MODE



31.7 Interrupts

The angular timer and its capture/compare features can generate multiple interrupt conditions. To accommodate all of these interrupt sources, the module is provided with its own interrupt logic structure, similar to that of the micro controller. Angular timer interrupts are enabled by the ATxIE0 register ([Register 31-13](#)) and their respective flags are located in the ATxIR0 register ([Register 31-14](#)). The capture/compare interrupts are enabled by the ATxIE1 register ([Register 31-15](#)) with flags in the ATxIR1 register ([Register 31-16](#)). All sources are funneled into a single Angular Timer Interrupt Flag bit, ATxIF of the PIR5 register ([Register 7-11](#)). This means that upon a triggered interrupt, the ATxIR0 and ATxIR1 register bits will indicate the source of the triggered interrupt. It also means that in order for specific interrupts to generate a microcontroller interrupt, both the ATxIE bit of the PIE register and the desired enable bit in either ATxIE0 or ATxIE1 must be set.

Note: Due to the nature of the angular timer interrupts, the ATxIF flag bit of the PIR5 register is read-only.

31.7.4 ANGULAR TIMER CAPTURE/COMPARE INTERRUPTS

Capture and compare interrupts are triggered by the capture/compare functions of the module. If configured for Capture mode, the interrupt will trigger after the capture signal has successfully latched the value of the phase counter into the capture registers. If configured for Compare mode, the interrupt will trigger when a match is detected between the value placed in the compare register and the value of the phase counter. These interrupts are controlled by the CC1IE, CC2IE, and CC3IE bits of the ATxIE1 register, respectively, and are similarly indicated by the CC1IF, CC2IF, and CC3IF bits of the ATxIR1 register.

31.7.1 ANGULAR TIMER PERIOD INTERRUPT

This interrupt is triggered when the AT module detects a period event. In Single-Pulse mode, a period event occurs on every input signal edge. In Multi-Pulse mode, a period event occurs on the input signal edge following a missed pulse. The period interrupt generation matches with the pulses on the period clock output of the timer. It is enabled by the ATPERIE bit of the ATxIE0 register and the status is indicated by the PERIF bit of the ATxIR0 register.

31.7.2 ANGULAR TIMER PHASE CLOCK INTERRUPT

This interrupt is triggered on each pulse of the phase clock output of the timer. It is enabled by the ATPHIE bit of the ATxIE0 register and the status is indicated by the PHSIF bit of the ATxIR0 register.

31.7.3 ANGULAR TIMER MISSING PULSE INTERRUPT

This interrupt is triggered upon the output of a missing pulse detection signal. Refer to [Section 31.2.3 “Missing Pulse Detection”](#) for more information. This interrupt is enabled by the ATMISSIE bit of the ATxIE0 register and its status is indicated by the ATMISSIF bit of the ATxIR0 register.

31.8 Angular Timer Control Registers

Long bit name prefixes for the angular timer peripherals are shown in Table 31-1. Refer to Section 1.1 “Register and Bit Naming Conventions” for more information.

TABLE 31-1:

| Peripheral | Bit Name Prefix |
|------------|-----------------|
| AT1 | AT1 |

REGISTER 31-1: ATxCON0: ANGULAR TIMER CONTROL 0 REGISTER

| R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | U-0 | R/W-0/0 | R/W-0/0 |
|---------|---------|---------|---------|---------|-------|---------|---------|
| EN | PREC | PS<1:0> | POL | — | APMOD | MODE | |
| 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 **EN:** Angular Timer Enable bit
 1 = Angular timer is enabled; internal clocks are active
 0 = Angular timer is disabled
- bit 6 **PREC:** Period Precision Control bit
 1 = Period prescaler is reset at the start of every period
 0 = Period prescaler is not reset at the start of every period; fraction period affects next period measurement
- bit 5-4 **PS<1:0>:** Clock Prescaler Control bits
 11 = Resolution and phase counter prescale logic is clocked by ATxCLK/8
 10 = Resolution and phase counter prescale logic is clocked by ATxCLK/4
 01 = Resolution and phase counter prescale logic is clocked by ATxCLK/2
 00 = Resolution and phase counter prescale logic is clocked by ATxCLK
- bit 3 **POL:** ATxsig Active Edge (Polarity) Select bit
 1 = Falling edge of ATxsig is the active edge
 0 = Rising edge of ATxsig is the active edge
- bit 2 **Unimplemented:** Read as '0'
- bit 1 **APMOD:** Adaptive Missing Pulse Mode Select bit
 1 = Adaptive Missing Pulse mode. Missing pulse is detected when no pulse is detected within 1.5 times ATxPER
 0 = Fixed Missing Pulse mode. ATxMISS register pair determines missing pulse event.
- bit 0 **MODE:** Angular Timer Mode Select bit
 1 = Angular timer is in Multi-Pulse mode (period of input signal defined by missing pulses)
 0 = Angular timer is in Single-Pulse mode (period of input signal defined by input pulses)

REGISTER 31-2: ATxCON1: ANGULAR TIMER CONTROL 1 REGISTER

| | | | | | | | |
|-------|---------|-----|---------|-----|---------|-------|-------|
| U-0 | R/W-0/0 | U-0 | R/W-0/0 | U-0 | R/W-0/0 | R-0/0 | R-0/0 |
| — | PHP | — | PRP | — | MPP | ACCS | VALID |
| 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 | Unimplemented: Read as '0' |
| bit 6 | PHP: Phase Clock Output Polarity bit 1 = Phase clock output is active-low 0 = Phase clock output is active-high |
| bit 5 | Unimplemented: Read as '0' |
| bit 4 | PRP: Period Clock Output Polarity bit 1 = Period clock output is active-low 0 = Period clock output is active-high |
| bit 3 | Unimplemented: Read as '0' |
| bit 2 | MPP: Missing Pulse Output Polarity bit 1 = Missing pulse output is active-low 0 = Missing pulse output is active-high |
| bit 1 | ACCS: Acceleration Sign bit 1 = The value currently in ATxPER is less than the previous value 0 = The value currently in ATxPER is greater than or equal to the previous value |
| bit 0 | VALID: Valid Measurement bit 1 = Sufficient input cycles have occurred to make ATxPER and ATxPHS valid. 0 = The values in ATxPER and ATxPHS are not valid; not enough input cycles have occurred |

REGISTER 31-3: ATxCLK: ANGULAR TIMER CLOCK SELECTION REGISTER

| | | | | | | | |
|-------|-----|-----|-----|-----|-----|-----|---------|
| U-0 | U-0 | U-0 | U-0 | U-0 | U-0 | U-0 | R/W-x/x |
| — | — | — | — | — | — | — | CS0 |
| 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 7-1 **Unimplemented:** Read as '0'

bit 0 **CS0:** Angular Timer Clock Selection bit
 1 = HFINTOSC 16 MHz
 0 = Fosc

REGISTER 31-4: ATxSIG: ANGULAR TIMER INPUT SIGNAL SELECTION REGISTER

| | | | | | | | |
|-------|-----|-----|-----|-----|-----------|---------|---------|
| U-0 | U-0 | U-0 | U-0 | U-0 | R/W-x/x | R/W-x/x | R/W-x/x |
| — | — | — | — | — | SSEL<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 | q = Value depends on condition |

bit 7-3 **Unimplemented:** Read as '0'

bit 2-0 **SSEL<2:0>:** Angular Input Signal Selection bit
 111 = LC4_out
 110 = LC3_out
 101 = LC2_out
 100 = LC1_out
 011 = ZCD1_out
 010 = cmp2_sync
 001 = cmp1_sync
 000 = ATxINPPS

REGISTER 31-5: ATxRESH: ANGULAR TIMER RESOLUTION HIGH REGISTER

| | | | | | | | |
|-------|-----|-----|-----|-----|-----|----------|---------|
| U-0 | U-0 | U-0 | U-0 | U-0 | U-0 | R/W-x/u | R/W-x/u |
| — | — | — | — | — | — | RES<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 | q = Value depends on condition |

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

bit 1-0 **RES<9:8>:** ATxRES Most Significant bits, the Phase Counter Resolution

Note 1: Writing to this register resets VALID bit of the ATxCON1 ([Register 31-2](#)); output signals are inhibited for at least two input cycles.

2: This register is not guarded for atomic access, and should only be accessed while the timer is not running.

REGISTER 31-6: ATxRESL: ANGULAR TIMER RESOLUTION LOW 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 |
| RES<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 | q = Value depends on condition |

bit 7-0 **RES<7:0>:** ATxRES Least Significant bits, the Phase Counter Resolution

Note 1: Writing to this register resets VALID bit of the ATxCON1 ([Register 31-2](#)); output signals are inhibited for at least two input cycles.

2: This register is not guarded for atomic access, and should only be accessed while the timer is not running.

REGISTER 31-7: ATxMISSH: ANGULAR TIMER MISSING PULSE DELAY HIGH 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 |
| MISS<15:8> ⁽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 7-0 **MISS<15:8>⁽¹⁾**: Most Significant bits (2's complement) of ATxMISS. ATxMISS defines the period counter value at which the missing pulse output becomes valid, based on the difference between the current counter value and the latched-in value of ATxPER.

Note 1: ATxMISSH is held until ATxMISSL is written. Proper writes of ATxMISS should write to ATxMISSH first, then ATxMISSL to ensure the value is properly written.

REGISTER 31-8: ATxMISSL: ANGULAR TIMER MISSING PULSE DELAY LOW 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 |
| MISS<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 | q = Value depends on condition |

bit 7-0 **MISS<7:0>**: Least Significant bits (2's complement) of ATxMISS. ATxMISS defines the period counter value at which the missing pulse output becomes valid, based on the difference between the current counter value and the latched-in value of ATxPER.

REGISTER 31-9: ATxPERH: ANGULAR TIMER MEASURED PERIOD HIGH REGISTER

| | | | | | | | |
|-------|-----------|-------|-------|-------|-------|-------|-------|
| R-x/x | R-x/x | R-x/x | R-x/x | R-x/x | R-x/x | R-x/x | R-x/x |
| POV | PER<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 | q = Value depends on condition |

- bit 7 **POV:** Period Counter Overflow bit
 1 = Counter rolled over one or more times during measurement
 0 = Value shown by ATxPER is valid
- bit 6-0 **PER<14:8>:** Most Significant bits of ATxPER. ATxPER is the measured period value from the period counter.

REGISTER 31-10: ATxPERL: ANGULAR TIMER MEASURED PERIOD LOW REGISTER

| | | | | | | | |
|----------|-------|-------|-------|-------|-------|-------|-------|
| R-x/x | R-x/x | R-x/x | R-x/x | R-x/x | R-x/x | R-x/x | R-x/x |
| PER<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 | q = Value depends on condition |

- bit 7-0 **PER<7:0>:** Least Significant bits of ATxPER. ATxPER is the measured period value from the period counter.

REGISTER 31-11: ATxPHSH: ANGULAR TIMER PHASE COUNTER HIGH REGISTER

| | | | | | | | |
|-------|-----|-----|-----|-----|-----|----------|-------|
| U-0 | U-0 | U-0 | U-0 | U-0 | U-0 | R-x/x | R-x/x |
| — | — | — | — | — | — | PHS<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 | q = Value depends on condition |

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

bit 1-0 **PHS<9:8>:** Most Significant bits of ATxPHS. ATxPHS is the instantaneous value of the phase counter.

REGISTER 31-12: ATxPHSL: ANGULAR TIMER PHASE COUNTER LOW REGISTER

| | | | | | | | |
|----------|-------|-------|-------|-------|-------|-------|-------|
| R-x/x | R-x/x | R-x/x | R-x/x | R-x/x | R-x/x | R-x/x | R-x/x |
| PHS<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 | q = Value depends on condition |

bit 7-0 **PHS<7:0>:** Least Significant bits of ATxPHS. ATxPHS is the instantaneous value of the phase counter.

REGISTER 31-13: ATxIE0: ANGULAR TIMER ENABLE 0 REGISTER

| | | | | | | | |
|-------|-----|-----|-----|-----|---------|---------|---------|
| U-0 | U-0 | U-0 | U-0 | U-0 | R/W-0/0 | R/W-0/0 | R/W-0/0 |
| — | — | — | — | — | PHSIE | MISSIE | PERIE |
| 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 7-3 **Unimplemented:** Read as '0'
- bit 2 **PHSIE:** Phase Interrupt Enable bit
1 = The phase interrupt is enabled
0 = The phase interrupt is disabled
- bit 1 **MISSIE:** Missed Pulse Interrupt Enable bit
1 = The missed pulse interrupt is enabled
0 = The missed pulse interrupt is disabled
- bit 0 **PERIE:** Period Interrupt Enable bit
1 = The period interrupt is enabled
0 = The period interrupt is disabled

REGISTER 31-14: ATxIR0: ANGULAR TIMER INTERRUPT FLAG 0 REGISTER

| | | | | | | | |
|-------|-----|-----|-----|-----|---------|---------|---------|
| U-0 | U-0 | U-0 | U-0 | U-0 | R/W-0/0 | R/W-0/0 | R/W-0/0 |
| — | — | — | — | — | PHSIF | MISSIF | PERIF |
| 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 7-3 **Unimplemented:** Read as '0'
- bit 2 **PHSIF:** Phase Interrupt Flag bit
1 = The phase interrupt has occurred
0 = The phase interrupt has not occurred, or has been cleared
- bit 1 **MISSIF:** Missed Pulse Interrupt Flag bit
1 = The missed pulse interrupt has occurred
0 = The missed pulse interrupt has not occurred, or has been cleared
- bit 0 **PERIF:** Period Interrupt Flag bit
1 = The period interrupt has occurred
0 = The period interrupt has not occurred, or has been cleared

REGISTER 31-15: ATxIE1: ANGULAR TIMER ENABLE 1 REGISTER

| | | | | | | | |
|-------|-----|-----|-----|-----|---------|---------|---------|
| U-0 | U-0 | U-0 | U-0 | U-0 | R/W-0/0 | R/W-0/0 | R/W-0/0 |
| — | — | — | — | — | CC3IE | CC2IE | CC1IE |
| 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 7-3 | Unimplemented: Read as '0' |
| bit 2 | CC3IE: Capture/Compare Interrupt 3 Enable bit <u>If CC3MODE = 1 (Capture)</u> 1 = Capture interrupt 3 is enabled 0 = Capture interrupt 3 is disabled <u>If CC3MODE = 0 (Compare)</u> 1 = Compare interrupt 3 is enabled 0 = Compare interrupt 3 is disabled |
| bit 1 | CC2IE: Capture/Compare Interrupt 2 Enable bit <u>If CC2MODE = 1 (Capture)</u> 1 = Capture interrupt 2 is enabled 0 = Capture interrupt 2 is disabled <u>If CC2MODE = 0 (Compare)</u> 1 = Compare interrupt 2 is enabled 0 = Compare interrupt 2 is disabled |
| bit 0 | CC1IE: Capture/Compare Interrupt 1 Enable bit <u>If CC1MODE = 1 (Capture)</u> 1 = Capture interrupt 1 is enabled 0 = Capture interrupt 1 is disabled <u>If CC1MODE = 0 (Compare)</u> 1 = Compare interrupt 1 is enabled 0 = Compare interrupt 1 is disabled |

REGISTER 31-16: ATxIR1: ANGULAR TIMER INTERRUPT FLAG 1 REGISTER

| | | | | | | | |
|-------|-----|-----|-----|-----|---------|---------|---------|
| U-0 | U-0 | U-0 | U-0 | U-0 | R/W-0/0 | R/W-0/0 | R/W-0/0 |
| — | — | — | — | — | CC3IF | CC2IF | CC1IF |
| 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 7-3 **Unimplemented:** Read as '0'
- bit 2 **CC3IF:** Capture/Compare Interrupt 3 Flag bit
 If CC3MODE = 1 (Capture)
 1 = Capture interrupt 3 has occurred; captured phase value is in ATxCC3
 0 = Capture interrupt 3 has not occurred, or has been cleared
 If CC3MODE = 0 (Compare)
 1 = Compare interrupt 3 has occurred
 0 = Compare interrupt 3 has not occurred, or has been cleared
- bit 1 **CC2IF:** Capture/Compare Interrupt 2 Flag bit
 If CC2MODE = 1 (Capture)
 1 = Capture interrupt 2 has occurred; captured phase value is in ATxCC2
 0 = Capture interrupt 2 has not occurred, or has been cleared
 If CC2MODE = 0 (Compare)
 1 = Compare interrupt 2 has occurred
 0 = Compare interrupt 2 has not occurred, or has been cleared
- bit 0 **CC1IF:** Capture/Compare Interrupt 1 Flag bit
 If CC1MODE = 1 (Capture)
 1 = Capture interrupt 1 has occurred; captured phase value is in ATxCC1
 0 = Capture interrupt 1 has not occurred, or has been cleared
 If CC1MODE = 0 (Compare)
 1 = Compare interrupt 1 has occurred
 0 = Compare interrupt 1 has not occurred, or has been cleared

REGISTER 31-17: ATxSTPTH: ANGULAR TIMER SET POINT HIGH REGISTER ⁽¹⁾

| | | | | | | | |
|-------|------------|---------|---------|---------|---------|---------|---------|
| 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 | R/W-x/u |
| — | STPT<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 | q = Value depends on condition |

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

bit 6-0 **STPT<14:8>:** Set Point Most Significant bits. ATxSTPT determines the threshold setting that the period is compared against for error calculation.

Note 1: Writes to ATxSTPTH are double buffered. The value written to this register is held until a write to ATxSTPTL occurs, at which point the value will be latched into the register

REGISTER 31-18: ATxSTPTL: ANGULAR TIMER SET POINT LOW 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 |
| STPT<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 | q = Value depends on condition |

bit 7-0 **STPT<7:0>:** Set Point Least Significant bits. ATxSTPT determines the threshold setting that the period is compared against for error calculation.

REGISTER 31-19: ATxERRH: ANGULAR TIMER SET POINT ERROR VALUE HIGH REGISTER

| | | | | | | | |
|-----------|-------|-------|-------|-------|-------|-------|-------|
| R-x/x | R-x/x | R-x/x | R-x/x | R-x/x | R-x/x | R-x/x | R-x/x |
| ERR<15:8> | | | | | | | |
| 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 | q = Value depends on condition |

bit 7-0 **ERR<15:8>**: Most Significant bits of ATxERR. ATxERR is the error of the measured period value compared to the threshold setting, defined as ATxPER-ATxSTPTP.

REGISTER 31-20: ATxERRL: ANGULAR TIMER SET POINT ERROR VALUE LOW REGISTER

| | | | | | | | |
|----------|-------|-------|-------|-------|-------|-------|-------|
| R-x/x | R-x/x | R-x/x | R-x/x | R-x/x | R-x/x | R-x/x | R-x/x |
| ERR<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 | q = Value depends on condition |

bit 7-0 **ERR<7:0>**: Least Significant bits of ATxERR. ATxERR is the error of the measured period value compared to the threshold setting, defined as ATxPER-ATxSTPTP.

REGISTER 31-21: ATxCONy: ANGULAR TIMER CAPTURE/COMPARE CONTROL 1 REGISTER

| | | | | | | | |
|---------|-----|-----|---------|---------|-----|-----|---------|
| R/W-0/0 | U-0 | U-0 | R/W-0/0 | R/W-0/0 | U-0 | U-0 | R/W-0/0 |
| CCyEN | — | — | CCPyPOL | CAPyP | — | — | CCyMODE |
| 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 **CCyEN:** Capture/Compare Enable bit
1 = Capture/Compare logic is enabled
0 = Capture/Compare logic is disabled
- bit 6-5 **Unimplemented:** Read as '0'
- bit 4 **CCyPOL:** Capture/Compare Output Polarity bit
In Capture mode (CCyMODE = 1):
1 = ATxCCOUT1 is active low when ATxCCy is updated
0 = ATxCCOUT1 is active high when ATxCCy is updated
In Compare mode (CCyMODE = 0):
1 = ATxCCOUT1 is active low when ATxPHS = ATxCCy
0 = ATxCCOUT1 is active high when ATxPHS = ATxCCy
- bit 3 **CAPyP:** Capture Input Polarity bit
In Capture mode (CCyMODE = 1):
1 = At falling edge of the capture input (Selected by ATxCSELY) the value of the phase counter is captured in ATxCC1
0 = At rising edge of the capture input (Selected by ATxCSELY) the value of the phase counter is captured in ATxCC1
In Compare mode (CCyMODE = 0):
This bit is ignored.
- bit 2-1 **Unimplemented:** Read as '0'
- bit 0 **CCyMODE:** Capture/Compare Mode Select bit
1 = Capture/compare logic is in Capture mode
0 = Capture/compare logic is in Compare mode

REGISTER 31-22: ATxCSELY: ANGULAR TIMER CAPTURE INPUT SELECT y REGISTER

| | | | | | | | |
|-------|-----|-----|-----|-----|-----------|---------|---------|
| U-0 | U-0 | U-0 | U-0 | U-0 | R/W-0/0 | R/W-0/0 | R/W-0/0 |
| — | — | — | — | — | CPyS<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

q = Value depends on condition

bit 7-3

Unimplemented: Read as '0'

bit 2-0

CPyS<2:0>: Capture Input Source Select bits

111 = CWG_interrupt

110 = LC4_out

101 = LC3_out

100 = LC2_out

111 = LC1_out

010 = cmp2_sync

001 = cmp1_sync

000 = ATxCCy pin

REGISTER 31-23: ATxCCyH: ANGULAR TIMER CAPTURE/COMPARE y HIGH REGISTER

| | | | | | | | |
|-------|-----|-----|-----|-----|-----|----------|---------|
| U-0 | U-0 | U-0 | U-0 | U-0 | U-0 | R/q-0/0 | R/q-0/0 |
| — | — | — | — | — | — | CCy<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 | q = Value depends on condition |

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

bit 1-0 **CCy<9:8>:** ATxCCy Most Significant bits

In Capture mode (CCyMODE = 1) (Read-only):

ATxCCy is the captured value of ATxPHS when the capture input is signaled.

In Compare mode (CCyMODE = 0):

ATxCCy is the value that is compared to the current value of ATxPHS to trigger an interrupt/output pulse.

Note 1: Writes to ATxCCyH are double buffered. The value written to this register is held until a write to ATxCCyL occurs, at which point the value will be latched into the register

REGISTER 31-24: ATxCCyL: ANGULAR TIMER CAPTURE/COMPARE y LOW REGISTER

| | | | | | | | |
|----------|---------|---------|---------|---------|---------|---------|---------|
| R/q-0/0 | R/q-0/0 | R/q-0/0 | R/q-0/0 | R/q-0/0 | R/q-0/0 | R/q-0/0 | R/q-0/0 |
| CCy<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 | q = Value depends on condition |

bit 7-0 **CCy<7:0>:** ATxCCy Least Significant bits

In Capture mode (CCyMODE = 1) (Read-only):

ATxCCy is the captured value of ATxPHS when the capture input is signaled.

In Compare mode (CCyMODE = 0):

ATxCCy is the value that is compared to the current value of ATxPHS to trigger an interrupt/output pulse.

TABLE 31-2: SUMMARY OF REGISTERS ASSOCIATED WITH THE ANGULAR TIMER MODULE

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Register on page |
|----------|------------|------------|---------|--------|-------|-----------|----------|---------|------------------|
| AT1CC1H | — | — | — | — | — | — | CC1<9:8> | | 478 |
| AT1CC1L | CC1<7:0> | | | | | | | | 478 |
| AT1CCON1 | CC1EN | — | — | CC1POL | CAP1P | — | — | CC1MODE | 476 |
| AT1CCON2 | CC2EN | — | — | CC2POL | CAP2P | — | — | CC2MODE | 476 |
| AT1CCON3 | CC3EN | — | — | CC3POL | CAP3P | — | — | CC3MODE | 476 |
| AT1CLK | — | — | — | — | — | — | — | CS0 | 466 |
| AT1CON0 | EN | PREC | PS<1:0> | | POL | — | APMOD | MODE | 464 |
| AT1CON1 | — | PHP | — | PRP | — | MPP | ACCS | VALID | 465 |
| AT1CSEL1 | — | — | — | — | — | CP1S<2:0> | | | 477 |
| AT1CSEL2 | — | — | — | — | — | CP2S<2:0> | | | 477 |
| AT1CSEL3 | — | — | — | — | — | CP3S<2:0> | | | 477 |
| AT1ERRH | ERR<15:8> | | | | | | | | 475 |
| AT1ERRL | ERR<7:0> | | | | | | | | 475 |
| AT1IE0 | — | — | — | — | — | PHSIE | MISSIE | PERIE | 471 |
| AT1IR0 | — | — | — | — | — | PHSIF | MISSIF | PERIF | 471 |
| AT1IE1 | — | — | — | — | — | CC3IE | CC2IE | CC1IE | 472 |
| AT1IR1 | — | — | — | — | — | CC3IF | CC2IF | CC1IF | 473 |
| AT1MISSH | MISS<15:8> | | | | | | | | 468 |
| AT1MISSL | MISS<7:0> | | | | | | | | 468 |
| AT1PERH | POV | PER<14:8> | | | | | | | 469 |
| AT1PERL | PER<7:0> | | | | | | | | 469 |
| AT1PHSH | — | — | — | — | — | — | PHS<9:8> | | 470 |
| AT1PHSL | PHS<7:0> | | | | | | | | 470 |
| AT1RESH | — | — | — | — | — | — | RES<9:8> | | 467 |
| AT1RESL | RES<7:0> | | | | | | | | 467 |
| AT1SIG | — | — | — | — | — | SSEL<2:0> | | | 466 |
| AT1STPTH | — | STPT<14:8> | | | | | | | 474 |
| AT1STPTL | STPT<7:0> | | | | | | | | 474 |
| PIE5 | TMR3GIE | TMR3IE | TMR5GIE | TMR5IE | — | AT1IE | PID1EIE | PID1DIE | 102 |
| PIR5 | TMR3GIF | TMR3IF | TMR5GIF | TMR5IF | — | AT1IF | PID1EIF | PID1DIF | 107 |

Legend: — = unimplemented, read as '0'. Shaded cells are unused by the AT module.

32.0 MATH ACCELERATOR WITH PROPORTIONAL-INTEGRAL-DERIVATIVE (PID) MODULE

The math accelerator module is a mathematics module that can perform a variety of operations, most prominently acting as a PID (Proportional-Integral-Derivative) controller. A PID controller is an algorithm that uses the present error (proportional), the sum of the present and all previous errors (integral), and the difference between the present and previous change (derivative) to correct errors and provide stability in a system. It provides feedback to a system through a series of iterations, using the present error as well as previous errors to calculate a new input to the controller. The data flow for both PID modes is illustrated in [Figure 32-1](#).

The module accomplishes the task of calculating the PID algorithm by utilizing user-provided coefficients along with a multiplier and accumulator. As such, this multiplier and accumulator can also be configured to quickly and efficiently perform signed and unsigned multiply-and-add calculations both with and without accumulation. The data flow for these modes is illustrated in [Figure 32-2](#).

Features of this module include:

- Signed multiplier
- 35-bit signed accumulator
- PID controller support with user inputs for K1, K2, K3, system error and desired set point
- Completion and Error interrupts
- Multiple user modes allowing for PID with or without accumulation as well as several multiplication operations

32.1 PID Module Setup Summary

The PID module can be configured either as a PID controller or as a multiply and accumulate module. Multiply and accumulate can be performed in four modes:

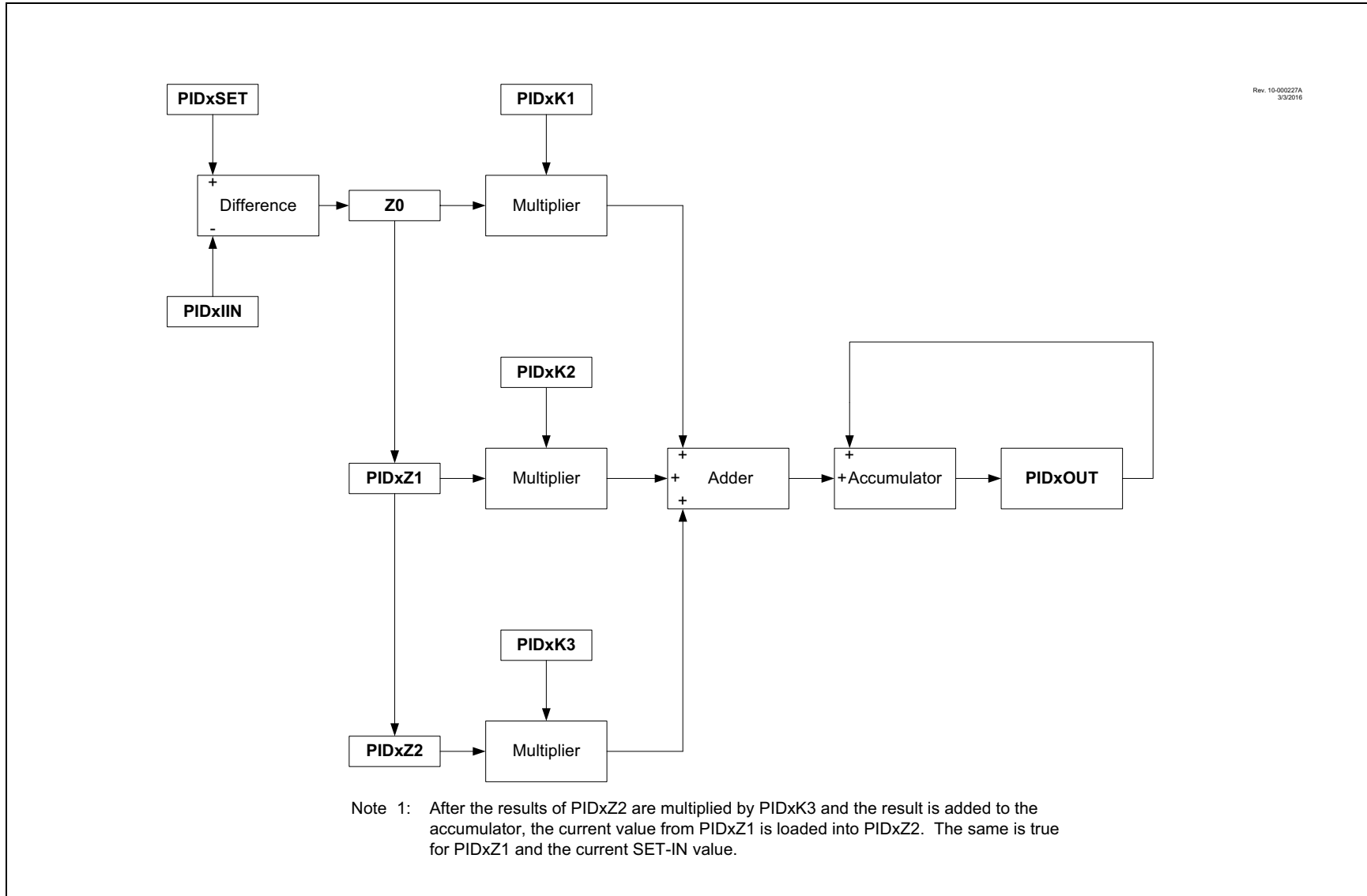
- Unsigned multiply and add, without accumulation
- Unsigned multiply and accumulate
- Signed multiply and add, without accumulation
- Signed multiply and accumulate

All of the modes are selected by the MODE<2:0> bits of the PIDxCON register.

32.1.1 PID MODE SETUP AND OPERATION

When the MODE<2:0> bits of the PIDxCON register are equal to '101', the module is in PID controller mode. The operation of the module in PID controller mode is generally performed as a loop. The input from an external system is fed into the controller, and the controller's output is fed back into the external system. This will produce a new response from the system that is then looped back into the PID controller. The data flow for the PID operation is illustrated in [Figure 32-1](#).

FIGURE 32-1: PID MODULE BASIC DATA FLOW BLOCK DIAGRAM, PID MODES



Within the controller, the input is subtracted from a pre-programmed set point to get an error value. This error value, along with the previous two error values (if any), are multiplied by user-input coefficients and the results of these multiplications are added together to make up the output. If the MODE<2:0> bits of the PIDxCON register = 101, the PID output is equal to the current output added to any previous outputs.

The three user-input coefficients (K1, K2, and K3) are derived from the three classic PID coefficients Kp, Ki, and Kd, and must be calculated prior to using the PID module.

1. K1 is the coefficient that is multiplied with the current error (SET-IN). It is defined by the following equation:

EQUATION 32-1:

$$K1 = Kp + Ki \cdot T + \frac{Kd}{T}$$

Note: T is the sampling period.

2. K2 is the coefficient that is multiplied with the previous iteration's error (Z1). Where T is the sampling period, it is defined by the following equation:

EQUATION 32-2:

$$K2 = -\left(Kp + \frac{2Kd}{T}\right)$$

Note: T is the sampling period.

3. K3 is the coefficient that is multiplied with the error that occurred two iterations previous to the current one (Z2). It is defined by the following equation:

EQUATION 32-3:

$$K3 = \frac{Kd}{T}$$

Note: T is the sampling period.

To operate the module in PID controller mode, perform the following steps:

1. Set the MODE<2:0> bits of the PIDxCON register to '101', then set the EN bit of the PIDxCON register.
2. Write the previously calculated K1, K2, and K3 values to the PIDxK1, PIDxK2, and PIDxK3 registers, respectively.
3. Write the desired set point that the input will be compared against to the PIDxSET registers.
4. Write the high byte of the value from the external system to PIDxINH. Then write the low byte of the value from the external system to PIDxINL. This will begin the calculation and set the BUSY bit of the PIDxCON register.
5. Either poll the BUSY bit of the PIDxCON register to check for it clearing or wait for the PIDxDIF interrupt to trigger, indicating that the operation has completed.
6. Read the PIDxOUT registers for the output value. If the PID was in Accumulation mode, PIDxOUT will contain the accumulation of the output added to the previous outputs, otherwise, it will contain only the latest output.
7. For proper PID operation, this output needs to be applied to the external system before the next input to the PID is applied. This is to ensure that the system can adjust based on the PID controller's feedback before the next calculation is made.

Note: The BUSY bit of the PIDxCON register goes high as soon as PIDxINL is written and remains high until all computation is complete. Until the BUSY bit goes low, the PIDxOUT values are not valid, and none of the registers associated with the PID module should be written to, as any such writes will corrupt the calculation.

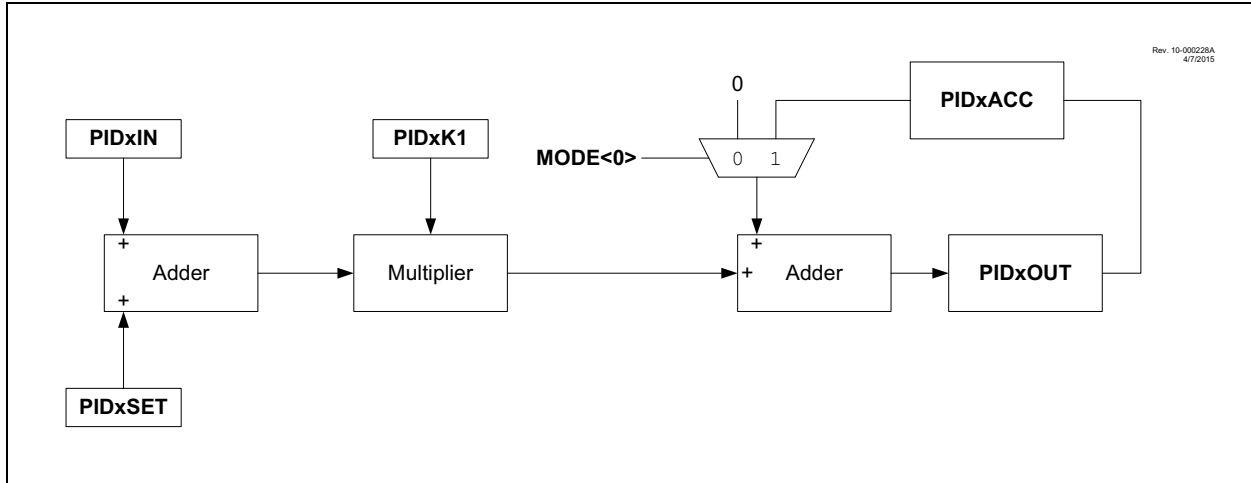
32.1.2 CONTEXT SAVING

It is possible to save the current state of the PID controller in software and restore it at a later time. In order to perform this, a calculation must not currently be active (BUSY = 0). Saving the PIDxOUT, PIDxZ1, and PIDxZ2 values elsewhere in memory will save the current state of the PID controller, although it may be desirable to also save PIDxK1, PIDxK2, PIDxK3, and/or PIDxSET, depending on the application. At the desired later time, these values can be written back into their respective registers, writing PIDxINL last, and the PID will continue from its previous state.

32.2 Add and Multiply Mode Setup and Operation

The PID module can also be used to perform 16-bit Add and Multiply computations. When the MODE<2:0> bits of the PIDxCON register are equal to '000', '001', '010', or '011', the module is in Add and Multiply mode. The data flow for the multiply and add operation is illustrated in Figure 32-2.

FIGURE 32-2: PID MODULE BASIC DATA FLOW BLOCK DIAGRAM, ADD AND MULTIPLY MODES



All Add and Multiply modes perform operations of the following form.

EQUATION 32-4:

$$OUTPUT = (A + B) \cdot C$$

Note: A = PIDxIN, B = PICxSET, and C = PIDxK1.

The four different Add and Multiply modes are:

- MODE<2:0> = 000: Inputs are unsigned, and the output does not accumulate
- MODE<2:0> = 001: Inputs are unsigned, and the output accumulates with previous outputs
- MODE<2:0> = 010: Inputs are signed, and the output does not accumulate
- MODE<2:0> = 011: Inputs are signed, and the output accumulates with previous outputs

In order to perform an Add and Multiply operation, perform the following steps:

1. Set the MODE<2:0> bits of the PIDxCON register to one of the four Add/Multiply modes, depending on which form of the calculation is desired, then set the EN bit of the PIDxCON register.
2. Write the value of C to the PIDxK1H/L register pair and the value of B to the PIDxSETH/L register pair, as well as the high byte of A to the PIDxINH register.
3. Finally, write the low byte of A to the PIDxINL register. This will begin the mathematical operation and set the BUSY bit of the PIDxCON register.
4. Either poll the BUSY bit of the PIDxCON register to check for it clearing or wait for the PIDxDIF interrupt to trigger, indicating that the operation has completed.
5. Read the PIDxOUT registers for the result of the calculation. In accumulation modes, the PIDxOUT register will hold any previous values added to the current calculation's value. In non-accumulation modes, the PIDxOUT register will just hold the current calculation's value.

These modes can also be used to perform 16-bit addition (by setting the C term in the above equation to 1) or 16-bit multiplication (by setting A or B to 0).

32.3 Interrupts

The PID module has two interrupts, indicated by the interrupt flags PIDxDIF and PIDxEIF in the PIR5 register, and controlled by the interrupt control bits PIDxDIE and PIDxEIE, respectively, in the PIE5 register.

The PIDxDIF interrupt triggers at the successful completion of a calculation, when the BUSY bit of the PIDxCON register goes low.

The PIDxEIF interrupt triggers when there is an error in the PID or multiply and add calculation, specifically an overflow error on the output value.

32.4 Handling Error Overflow

If a calculation causes an overflow of the value in the OUT registers, the value in said registers will roll over and the PIDxEIF interrupt will trigger. In the case of a PID calculation, this indicates that the error has outpaced the PID's capability to correct for the error of the system. In this case, it is recommended to 'saturate' the OUT registers in software whenever the PIDxEIF interrupt is set as part of the Interrupt Service Routine (ISR), as shown in [Example 32-1](#).

EXAMPLE 32-1: HANDLING PID OVERFLOWS

```
//Interrupt service routine
void interrupt ISR(void)

IF (PIR5BITS.PID1EIF==1&&PIE5BITS.PID1EIE==1)
{
    //saturate the PID1OUT registers
    PID1OUTH=0xFF;
    PID1OUTL=0xFF;
    PID1OUTLH=0xFF;
    PID1OUTLL=0xFF;
    PID1OUTHH=0xFF;
    //clear the interrupt flag
    PIR5bits.PID1EIF=0;
}
```

32.5 PID Control Registers

Long bit name prefixes for the 16-bit PID peripherals are shown in Table 32-1. Refer to Section 1.1 “Register and Bit Naming Conventions” for more information

TABLE 32-1:

| Peripheral | Bit Name Prefix |
|------------|-----------------|
| PID1 | PID1 |

REGISTER 32-1: PIDxCON: PID CONFIGURATION REGISTER

| R/W-0/0 | R/HS/HC-0/0 | U-0 | U-0 | U-0 | R/W-0/0 | R/W-0/0 | R/W-0/0 |
|---------|-------------|-----|-----|-----|-----------|---------|---------|
| EN | BUSY | — | — | — | MODE<2:0> | | |
| 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 **EN:** PID Module Enable bit

1 = PID module is enabled

0 = PID module is disabled

bit 6 **BUSY:** PID module is currently calculating

bit 5-3 **Unimplemented:** Read as '0'

bit 2-0 **MODE<2:0>:** PID Mode Control bits

11x = Reserved. Do not use.

101 = PID output is the calculated output (current error plus accumulated previous errors) in 2's complement notation

100 = Reserved. Do not use.

011 = (IN<15:0>+SET<15:0>)*K1<15:0> 2's complement signed inputs, with accumulation

010 = (IN<15:0>+SET<15:0>)*K1<15:0> 2's complement signed inputs, without accumulation

001 = (IN<15:0>+SET<15:0>)*K1<15:0> unsigned inputs, with accumulation

000 = (IN<15:0>+SET<15:0>)*K1<15:0> unsigned inputs, without accumulation

REGISTER 32-2: PIDxINH: PID INPUT HIGH 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 |
| IN<15:8> | | | | | | | |
| 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 | q = Value depends on condition |

bit 7-0 **IN<15:8>**: IN upper eight bits. IN is the 16-bit input from the control system to the PID module

REGISTER 32-3: PIDxINL: PID INPUT LOW 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 |
| IN<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 | q = Value depends on condition |

bit 7-0 **IN<7:0>**: IN lower eight bits. IN is the 16-bit input from the control system to the PID module

REGISTER 32-4: PIDxSETH: PID SET POINT HIGH 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 |
| SET<15:8> | | | | | | | |
| 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 | q = Value depends on condition |

bit 7-0 **SET<15:8>**: SET upper eight bits. SET is the 16-bit user-controlled variable that the input from the control system is compared against to determine the error in the system

REGISTER 32-5: PIDxSETL: PID SET POINT LOW 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 |
| SET<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 | q = Value depends on condition |

bit 7-0 **SET<7:0>**: SET lower eight bits. SET is the 16-bit user-controlled variable that the input from the control system is compared against to determine the error in the system

REGISTER 32-6: PIDxK1H: PID K1 HIGH 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 |
| K1<15:8> | | | | | | | |
| 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 | q = Value depends on condition |

bit 7-0 **K1<15:8>**: K1 upper eight bits. K1 is the 16-bit user-controlled coefficient calculated from $K_p + K_i + K_d$

REGISTER 32-7: PIDxK1L: PID K1 LOW 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 |
| K1<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 | q = Value depends on condition |

bit 7-0 **K1<7:0>**: K1 lower eight bits. K1 is the 16-bit user-controlled coefficient calculated from $K_p + K_i + K_d$

REGISTER 32-8: PIDxK2H: PID K2 HIGH 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 |
| K2<15:8> | | | | | | | |
| 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 | q = Value depends on condition |

bit 7-0 **K2<15:8>**: K2 upper eight bits. K2 is the 16-bit user-controlled coefficient calculated from $-(K_p + 2K_d)$

REGISTER 32-9: PIDxK2L: PID K2 LOW 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 |
| K2<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 | q = Value depends on condition |

bit 7-0 **K2<7:0>**: K2 lower eight bits. K2 is the 16-bit user-controlled coefficient calculated from $-(K_p + 2K_d)$

REGISTER 32-10: PIDxK3H: PID K3 HIGH 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 |
| K3<15:8> | | | | | | | |
| 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 | q = Value depends on condition |

bit 7-0 **K3<15:8>**: K3 upper eight bits. K3 is the 16-bit user-controlled coefficient calculated from Kd

REGISTER 32-11: PIDxK3L: PID K3 LOW 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 |
| K3<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 | q = Value depends on condition |

bit 7-0 **K3<7:0>**: K3 lower eight bits. K3 is the 16-bit user-controlled coefficient calculated from Kd

REGISTER 32-12: PIDxOUTU: PID OUTPUT UPPER REGISTER

| | | | | | | | |
|-------|-----|-----|-----|------------|---------|---------|---------|
| U-0 | U-0 | U-0 | U-0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 |
| — | — | — | — | OUT<35:32> | | | |
| 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 7-4 **Unimplemented:** Read as '0'

bit 3-0 **OUT<35:32>:** Bits <35:32> of OUT. OUT is the output value of the PID after completing the designated calculation on the specified inputs.

REGISTER 32-13: PIDxOUTH: PID OUTPUT HIGH HIGH 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 |
| OUT<31:24> | | | | | | | |
| 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 7-0 **OUT<31:24>:** Bits <31:24> of OUT. OUT is the output value of the PID after completing the designated calculation on the specified inputs.

REGISTER 32-14: PIDxOUTHL: PID OUTPUT HIGH LOW 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 |
| OUT<23:16> | | | | | | | |
| 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 7-0 **OUT<23:16>**: Bits <23:16> of OUT. OUT is the output value of the PID after completing the designated calculation on the specified inputs.

REGISTER 32-15: PIDxOUTLH: PID OUTPUT LOW HIGH 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 |
| OUT<15:8> | | | | | | | |
| 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 | q = Value depends on condition |

bit 7-0 **OUT<15:8>**: Bits <15:8> of OUT. OUT is the output value of the PID after completing the designated calculation on the specified inputs.

REGISTER 32-16: PIDxOUTLL: PID OUTPUT LOW LOW 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 |
| OUT<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 | q = Value depends on condition |

bit 7-0 **OUT<7:0>**: Bits <7:0> of OUT. OUT is the output value of the PID after completing the designated calculation on the specified inputs.

REGISTER 32-17: PIDxZ1U: PID Z1 UPPER REGISTER

| | | | | | | | |
|-------|-----|-----|-----|-----|-----|-----|---------|
| U-0 | U-0 | U-0 | U-0 | U-0 | U-0 | U-0 | R/W-0/0 |
| — | — | — | — | — | — | — | Z116 |
| 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 7-1 **Unimplemented:** Read as '0'
bit 0 **Z116:** Bit 16 of Z1. In PID mode, Z1 is the value of the error (IN minus SET) from the previous iteration of the PID control loop.

REGISTER 32-18: PIDxZ1H: PID Z1 HIGH 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 |
| Z1<15:8> | | | | | | | |
| 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 | q = Value depends on condition |

bit 7-0 **Z1<15:8>:** Bits <15:8> of Z1. In PID mode, Z1 is the value of the error (IN minus SET) from the previous iteration of the PID control loop.

REGISTER 32-19: PIDxZ1L: PID Z1 LOW 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 |
| Z1<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 | q = Value depends on condition |

bit 7-0 **Z1<7:0>:** Bits <7:0> of Z1. In PID mode, Z1 is the value of the error (IN minus SET) from the previous iteration of the PID control loop.

REGISTER 32-20: PIDxZ2U: PID Z2 UPPER REGISTER

| | | | | | | | |
|-------|-----|-----|-----|-----|-----|-----|---------|
| U-0 | U-0 | U-0 | U-0 | U-0 | U-0 | U-0 | R/W-0/0 |
| — | — | — | — | — | — | — | Z216 |
| 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 7-1 **Unimplemented:** Read as '0'
bit 0 **Z216:** Bit 16 of Z2. In PID mode, Z2 is the value of the error (IN minus SET) from the previous iteration of the PID control loop.

REGISTER 32-21: PIDxZ2H: PID Z2 HIGH 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 |
| Z2<15:8> | | | | | | | |
| 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 | q = Value depends on condition |

bit 7-0 **Z2<15:8>:** Bits <15:8> of Z2. In PID mode, Z2 is the value of the error (IN minus SET) from the previous iteration of the PID control loop.

REGISTER 32-22: PIDxZ2L: PID Z2 LOW 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 |
| Z2<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 | q = Value depends on condition |

bit 7-0 **Z2<7:0>:** Bits <7:0> of Z2. In PID mode, Z2 is the value of the error (IN minus SET) from the previous iteration of the PID control loop.

REGISTER 32-23: PIDxACCU: PID ACCUMULATOR UPPER REGISTER

| | | | | | | | |
|-------|-----|-----|-----|-----|------------|---------|---------|
| U-0 | U-0 | U-0 | U-0 | U-0 | R/W-0/0 | R/W-0/0 | R/W-0/0 |
| — | — | — | — | — | ACC<34:32> | | |
| 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 7-3 **Unimplemented:** Read as '0'

bit 2-0 **ACC<34:32>:** Bits <34:32> of ACC. ACC is the accumulator register in which all of the multiplier results for the PID are accumulated before being written to the output.

REGISTER 32-24: PIDxACCHH: PID ACCUMULATOR HIGH HIGH 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 |
| ACC<31:24> | | | | | | | |
| 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 7-0 **ACC<31:24>:** Bits <31:24> of ACC. ACC is the accumulator register in which all of the multiplier results for the PID are accumulated before being written to the output.

REGISTER 32-25: PIDxACCHL: PID ACCUMULATOR HIGH LOW 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 |
| ACC<23:16> | | | | | | | |
| 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 7-0 **ACC<23:16>:** Bits <23:16> of ACC. ACC is the accumulator register in which all of the multiplier results for the PID are accumulated before being written to the output.

REGISTER 32-26: PIDxACCLH: PID ACCUMULATOR LOW HIGH 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 |
| ACC<15:8> | | | | | | | |
| 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 | q = Value depends on condition |

bit 7-0 **ACC<15:8>**: Bits <15:8> of ACC. ACC is the accumulator register in which all of the multiplier results for the PID are accumulated before being written to the output.

REGISTER 32-27: PIDxACCLL: PID ACCUMULATOR LOW LOW 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 |
| ACC<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 | q = Value depends on condition |

bit 7-0 **ACC<7:0>**: Bits <7:0> of ACC. ACC is the accumulator register in which all of the multiplier results for the PID are accumulated before being written to the output.

TABLE 32-2: SUMMARY OF REGISTERS ASSOCIATED WITH THE PID MODULE

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Register on page |
|-----------|------------|--------|---------|--------|------------|------------|---------|---------|------------------|
| PID1ACCU | — | — | — | — | — | ACC<34:32> | | | 493 |
| PID1ACCHH | ACC<31:24> | | | | | | | | 493 |
| PID1ACCHL | ACC<23:16> | | | | | | | | 493 |
| PID1ACCLH | ACC<15:8> | | | | | | | | 494 |
| PID1ACCLL | ACC<7:0> | | | | | | | | 494 |
| PID1CON | EN | BUSY | — | — | — | MODE<2:0> | | | 485 |
| PID1INH | IN<15:8> | | | | | | | | 486 |
| PID1INL | IN<7:0> | | | | | | | | 486 |
| PID1K1H | K1<15:8> | | | | | | | | 487 |
| PID1K1L | K1<7:0> | | | | | | | | 487 |
| PID1K2H | K2<15:8> | | | | | | | | 487 |
| PID1K2L | K2<7:0> | | | | | | | | 487 |
| PID1K3H | K3<15:8> | | | | | | | | 488 |
| PID1K3L | K3<7:0> | | | | | | | | 488 |
| PID1OUTU | — | — | — | — | OUT<35:32> | | | | 489 |
| PID1OUTHH | OUT<31:24> | | | | | | | | 489 |
| PID1OUTHL | OUT<23:16> | | | | | | | | 490 |
| PID1OUTLH | OUT<15:8> | | | | | | | | 490 |
| PID1OUTLL | OUT<7:0> | | | | | | | | 490 |
| PID1SETH | SET<15:8> | | | | | | | | 486 |
| PID1SETL | SET<7:0> | | | | | | | | 486 |
| PID1Z1U | — | — | — | — | — | — | — | Z116 | 491 |
| PID1Z1H | Z1<15:8> | | | | | | | | 491 |
| PID1Z1L | Z1<7:0> | | | | | | | | 491 |
| PID1Z2U | — | — | — | — | — | — | — | Z216 | 492 |
| PID1Z2H | Z2<15:8> | | | | | | | | 492 |
| PID1Z2L | Z2<7:0> | | | | | | | | 492 |
| PIE5 | TMR3GIE | TMR3IE | TMR5GIE | TMR5IE | — | AT1IE | PID1EIE | PID1DIE | 102 |
| PIR5 | TMR3GIF | TMR3IF | TMR5GIF | TMR5IF | — | AT1IF | PID1EIF | PID1DIF | 107 |

Legend: — = unimplemented, read as '0'. Shaded cells are unused by the PID module.

33.0 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
- $\overline{\text{MCLR}}/\text{VPP}$
- 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 “PIC12(L)F1612/PIC16(L)F161X Memory Programming Specification” (DS40001720).

33.1 High-Voltage Programming Entry Mode

The device is placed into High-Voltage Programming Entry mode by holding the $\overline{\text{MCLR}}$ and ICSPDAT pins low then raising the voltage on $\overline{\text{MCLR}}/\text{VPP}$ to V_{IH} .

33.2 Low-Voltage Programming Entry Mode

The Low-Voltage Programming Entry mode allows the PIC® Flash MCUs to be programmed using VDD only, without high voltage. When the LVP bit of Configuration Words is set to ‘1’, the ICSP Low-Voltage Programming Entry mode 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. $\overline{\text{MCLR}}$ is brought to V_{IL} .
2. A 32-bit key sequence is presented on ICSPDAT, while clocking ICSPCLK.

Once the key sequence is complete, $\overline{\text{MCLR}}$ must be held at V_{IL} for as long as Program/Verify mode is to be maintained.

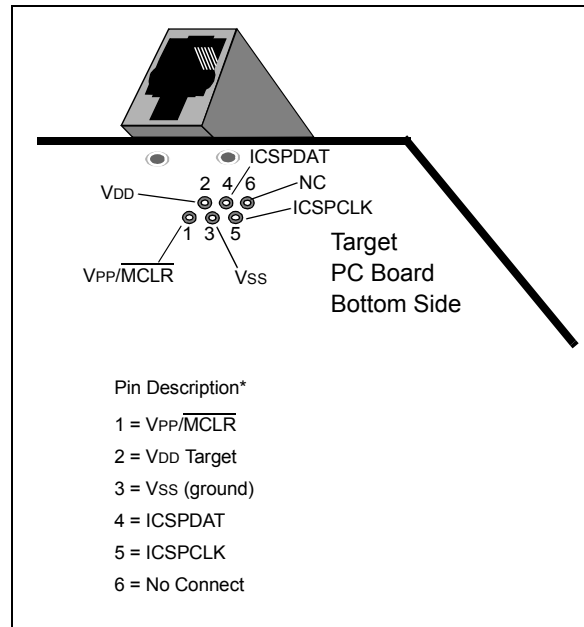
If low-voltage programming is enabled ($\text{LVP} = 1$), the $\overline{\text{MCLR}}$ Reset function is automatically enabled and cannot be disabled. See [Section 6.5 “MCLR”](#) for more information.

The LVP bit can only be reprogrammed to ‘0’ by using the High-Voltage Programming mode.

33.3 Common Programming Interfaces

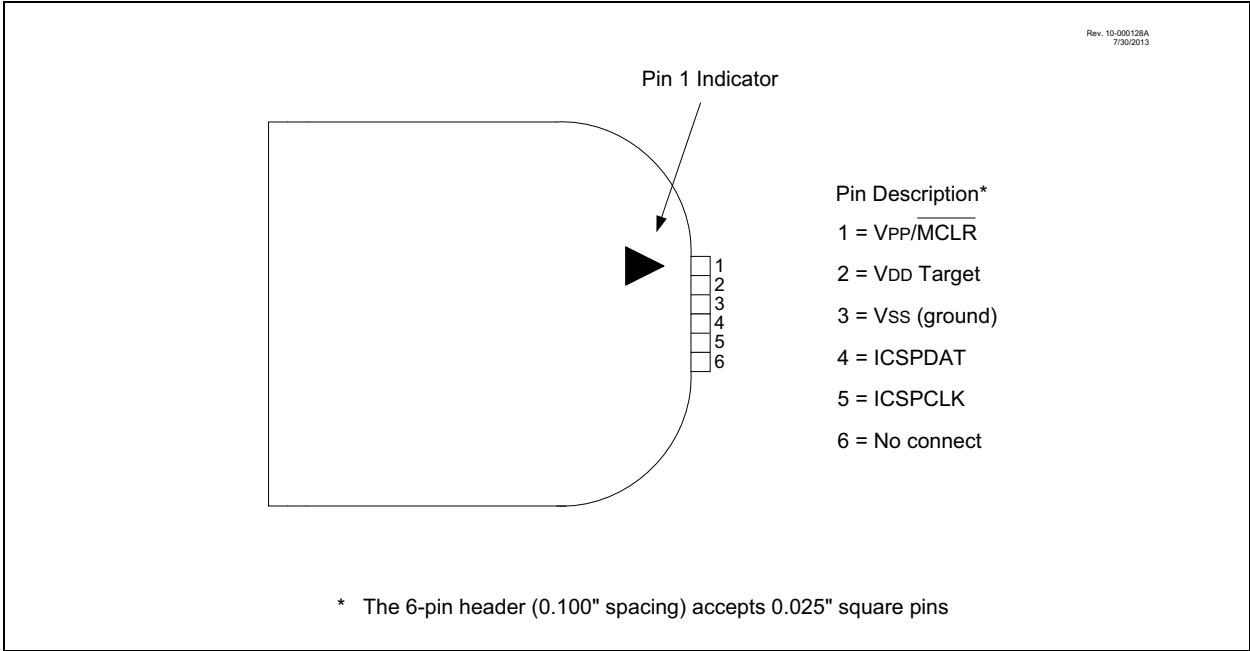
Connection to a target device is typically done through an ICSP™ header. A commonly found connector on development tools is the RJ-11 in the 6P6C (6-pin, 6-conductor) configuration. See [Figure 33-1](#).

FIGURE 33-1: 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 33-2](#).

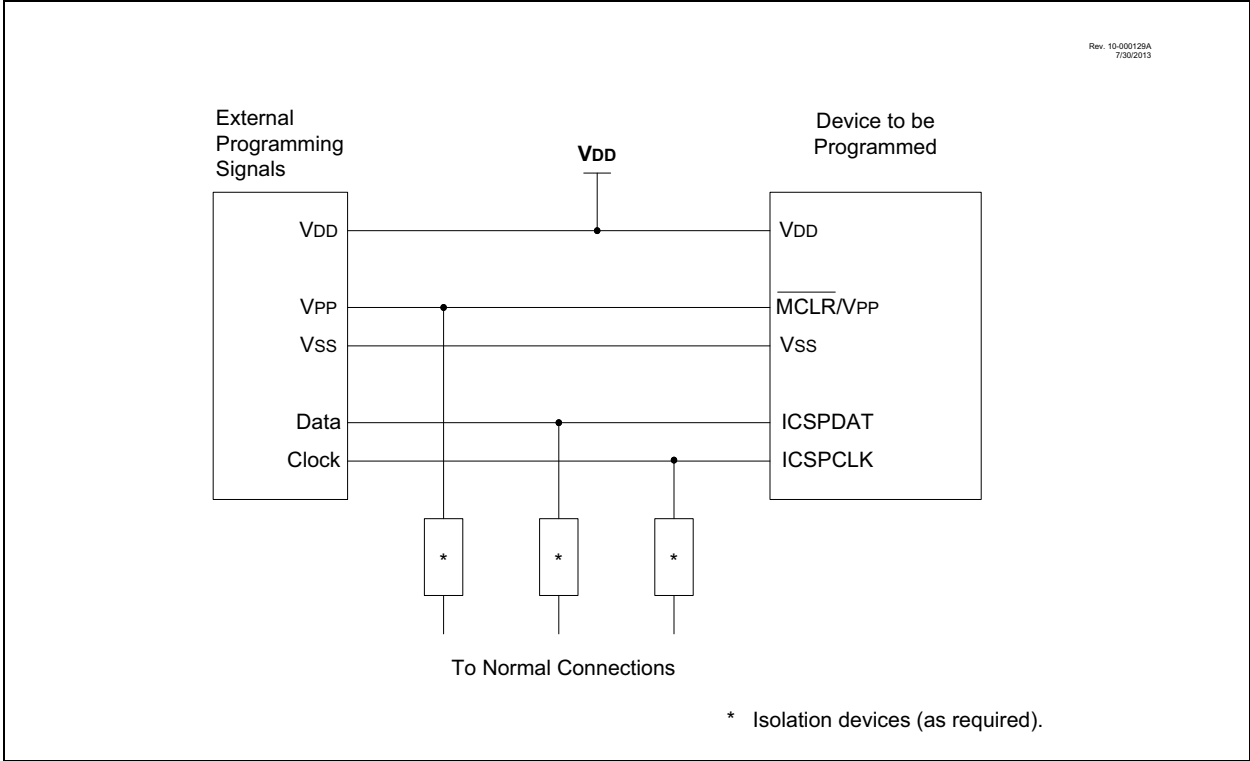
FIGURE 33-2: PICKIT™ PROGRAMMER STYLE CONNECTOR INTERFACE



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 33-3](#) for more information.

FIGURE 33-3: TYPICAL CONNECTION FOR ICSP™ PROGRAMMING



34.0 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 34-3 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.

34.1 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 34-1: 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 34-2: 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 34-1: GENERAL FORMAT FOR INSTRUCTIONS

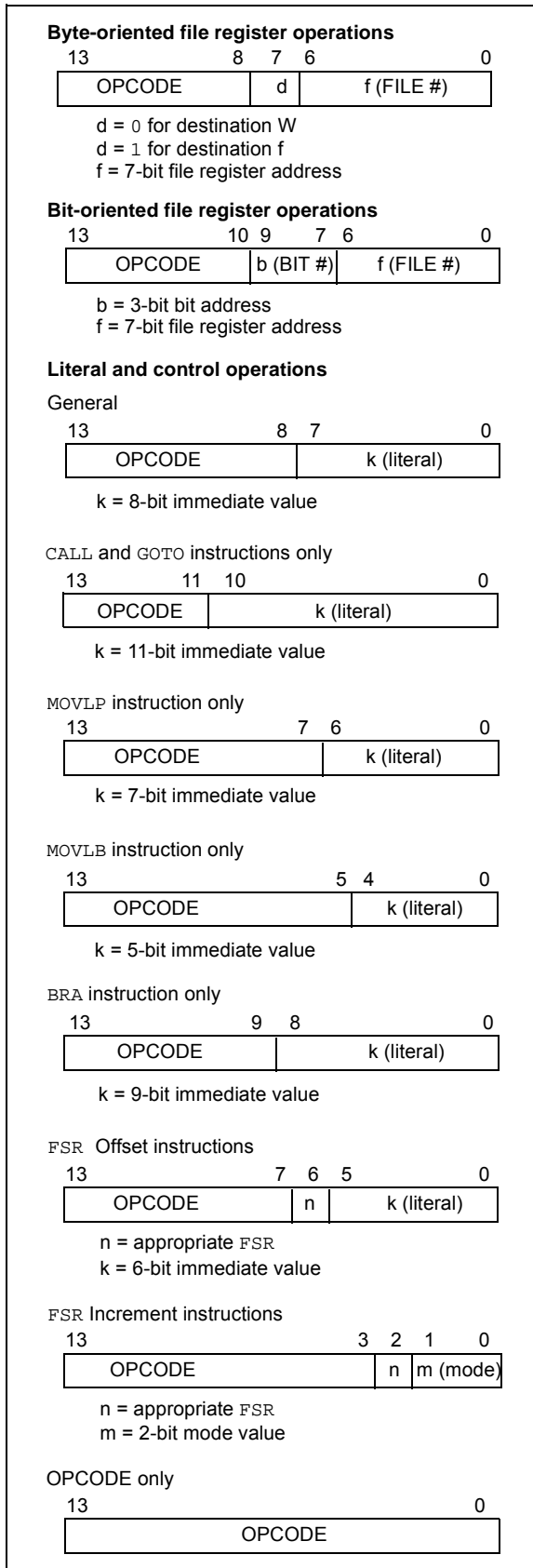


TABLE 34-3: ENHANCED MID-RANGE INSTRUCTION SET

| 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 | |
| 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 | | |
| MOVLP | 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.

Note 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 34-3: ENHANCED MID-RANGE 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 kkkk | Z | 2, 3 |
| | k[n] | Move INDFn to W, Indexed Indirect. | 1 | 11 | 1111 | 0nkk | 1nmm | Z | 2 |
| MOVWI | n mm | Move W to Indirect FSRn with pre/post inc/dec modifier, mm | 1 | 00 | 0000 | 0001 | kkkk | | 2, 3 |
| | k[n] | Move W to INDFn, Indexed Indirect. | 1 | 11 | 1111 | 1nkk | | | 2 |

- 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.

34.2 Instruction Descriptions

| ADDFSR | Add Literal to FSRn |
|------------------|--|
| Syntax: | [<i>label</i>] ADDFSR FSRn, k |
| Operands: | -32 ≤ k ≤ 31 n ∈ [0, 1] |
| Operation: | FSR(n) + k → 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. |

| ANDLW | AND literal with W |
|------------------|---|
| Syntax: | [<i>label</i>] ANDLW k |
| Operands: | 0 ≤ k ≤ 255 |
| Operation: | (W) .AND. (k) → (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. |

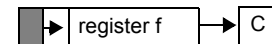
| ADDLW | Add literal and W |
|------------------|---|
| Syntax: | [<i>label</i>] ADDLW k |
| Operands: | 0 ≤ k ≤ 255 |
| Operation: | (W) + k → (W) |
| Status Affected: | C, DC, Z |
| Description: | The contents of the W register are added to the 8-bit literal 'k' and the result is placed in the W register. |

| ANDWF | AND W with f |
|------------------|--|
| Syntax: | [<i>label</i>] ANDWF f,d |
| Operands: | 0 ≤ f ≤ 127 d ∈ [0,1] |
| Operation: | (W) .AND. (f) → (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'. |

| ADDWF | Add W and f |
|------------------|--|
| Syntax: | [<i>label</i>] ADDWF f,d |
| Operands: | 0 ≤ f ≤ 127 d ∈ [0,1] |
| Operation: | (W) + (f) → (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'. |

| ASRF | Arithmetic Right Shift |
|------------------|--|
| Syntax: | [<i>label</i>] ASRF f {,d} |
| Operands: | 0 ≤ f ≤ 127 d ∈ [0,1] |
| Operation: | (f<7>) → dest<7> (f<7:1>) → dest<6:0>, (f<0>) → 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'. |

| ADDWFC | ADD W and CARRY bit to f |
|------------------|--|
| Syntax: | [<i>label</i>] ADDWFC f {,d} |
| Operands: | 0 ≤ f ≤ 127 d ∈ [0,1] |
| Operation: | (W) + (f) + (C) → 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'. |



| BCF | Bit Clear f |
|------------------|-------------------------------------|
| Syntax: | [<i>label</i>] BCF f,b |
| Operands: | 0 ≤ f ≤ 127 0 ≤ b ≤ 7 |
| Operation: | 0 → (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 ≤ f ≤ 127 0 ≤ b ≤ 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 ≤ label - PC + 1 ≤ 255 -256 ≤ k ≤ 255 |
| Operation: | (PC) + 1 + k → 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 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 ≤ f ≤ 127 0 ≤ 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: | (PC) + (W) → 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 PC + 1 + (W). This instruction is a 2-cycle instruction. |

| CALL | Call Subroutine |
|------------------|---|
| Syntax: | [<i>label</i>] CALL k |
| Operands: | 0 ≤ k ≤ 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 11-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. |

| BSF | Bit Set f |
|------------------|---------------------------------|
| Syntax: | [<i>label</i>] BSF f,b |
| Operands: | 0 ≤ f ≤ 127 0 ≤ b ≤ 7 |
| Operation: | 1 → (f) |
| Status Affected: | None |
| Description: | Bit 'b' in register 'f' is set. |

| CALLW | Subroutine Call With W |
|------------------|---|
| Syntax: | [<i>label</i>] 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: | [<i>label</i>] CLRF f |
| Operands: | 0 ≤ f ≤ 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: | [<i>label</i>] CLRW |
| Operands: | None |
| Operation: | 00h → (W) 1 → Z |
| Status Affected: | Z |
| Description: | W register is cleared. Zero bit (Z) is set. |

| CLRWDT | Clear Watchdog Timer |
|------------------|---|
| Syntax: | [<i>label</i>] CLRWDT |
| Operands: | None |
| Operation: | 00h → WDT 0 → WDT prescaler, 1 → \overline{TO} 1 → \overline{PD} |
| Status Affected: | \overline{TO} , \overline{PD} |
| Description: | CLRWDT 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: | [<i>label</i>] COMF f,d |
| Operands: | 0 ≤ f ≤ 127 d ∈ [0,1] |
| Operation: | (\overline{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: | [<i>label</i>] DECF f,d |
| Operands: | 0 ≤ f ≤ 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: | [<i>label</i>] DECFSZ f,d |
| Operands: | 0 ≤ f ≤ 127 d ∈ [0,1] |
| Operation: | (f) - 1 → (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. |

GOTO Unconditional Branch

Syntax: [*label*] GOTO *k*

Operands: $0 \leq k \leq 2047$

Operation: $k \rightarrow PC<10:0>$
 $PCLATH<6:3> \rightarrow PC<14:11>$

Status Affected: None

Description: GOTO is an unconditional branch. The 11-bit immediate value is loaded into PC bits <10:0>. The upper bits of PC are loaded from PCLATH<4:3>. GOTO is a 2-cycle instruction.

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'.

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.

IORLW Inclusive OR literal with W

Syntax: [*label*] IORLW *k*

Operands: $0 \leq k \leq 255$

Operation: $(W) .OR. k \rightarrow (W)$

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.

IORWF Inclusive OR W with f

Syntax: [*label*] IORWF *f,d*

Operands: $0 \leq f \leq 127$
 $d \in [0,1]$

Operation: $(W) .OR. (f) \rightarrow (\text{destination})$

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'.

LSLF Logical Left Shift

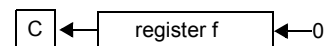
Syntax: [*label*] LSLF *f {,d}*

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

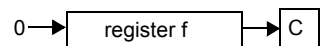
Syntax: [*label*] LSRF *f {,d}*

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) → (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: $n \in [0,1]$
 $mm \in [00,01, 10, 11]$
 $-32 \leq k \leq 31$

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: $0 \leq k \leq 31$

Operation: k → BSR

Status Affected: None

Description: The 5-bit literal 'k' is loaded into the Bank Select Register (BSR).

MOVLW **Move literal to W**

Syntax: [*label*] MOVLW *k*

Operands: $0 \leq k \leq 255$

Operation: $k \rightarrow (W)$

Status Affected: None

Description: The 8-bit literal 'k' is loaded into W register. The "don't cares" will assemble as '0's.

MOVLW **Move literal to W**

Syntax: [*label*] MOVLW *k*

Operands: $0 \leq k \leq 255$

Operation: $k \rightarrow (W)$

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: $0 \leq f \leq 127$

Operation: $(W) \rightarrow (f)$

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

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) → OPTION_REG

Status Affected: None

Description: Move data from W register to OPTION_REG register.

RESET **Software Reset**

Syntax: [*label*] RESET

Operands: None

Operation: Execute a device Reset. Resets the 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

RETLW **Return with literal in W**

Syntax: [*label*] RETLW k

Operands: 0 ≤ k ≤ 255

Operation: k → (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

TABLE

```

• ;W now has table value
•
•
ADDWF PC ;W = offset
RETLW k1 ;Begin table
RETLW k2 ;
•
•
•
RETLW kn ; End of table

```

Before Instruction
 W = 0x07

After Instruction
 W = value of k8

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.

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'.



RLF **Rotate Left f through Carry**

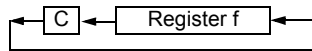
Syntax: [*label*] RLF f,d

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

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 | $W<3:0> > k<3:0>$ |
| DC = 1 | $W<3:0> \leq k<3:0>$ |

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 | $W<3:0> > f<3:0>$ |
| DC = 1 | $W<3:0> \leq f<3:0>$ |

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: $(f<3:0>) \rightarrow (\text{destination}<7:4>)$,
 $(f<7:4>) \rightarrow (\text{destination}<3:0>)$
 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'.

TRIS Load TRIS Register with W

Syntax: [*label*] TRIS *f*
 Operands: $5 \leq f \leq 7$
 Operation: $(W) \rightarrow \text{TRIS register 'f'}$
 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.

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.

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'.

35.0 ELECTRICAL SPECIFICATIONS

35.1 Absolute Maximum Ratings^(†)

| | |
|---|-----------------------------------|
| Ambient temperature under bias | -40°C to +125°C |
| Storage temperature | -65°C to +150°C |
| Voltage on pins with respect to V _{SS} | |
| on V _{DD} pin | |
| PIC16F1615/9 | -0.3V to +6.5V |
| PIC16LF1615/9 | -0.3V to +4.0V |
| on $\overline{\text{MCLR}}$ pin | -0.3V to +9.0V |
| on all other pins | -0.3V to (V _{DD} + 0.3V) |
| Maximum current | |
| on V _{SS} pin ⁽¹⁾ | |
| -40°C ≤ T _A ≤ +85°C | 250 mA |
| +85°C ≤ T _A ≤ +125°C | 85 mA |
| on V _{DD} pin ⁽¹⁾ | |
| -40°C ≤ T _A ≤ +85°C | 250 mA |
| +85°C ≤ T _A ≤ +125°C | 85 mA |
| Sunk by any standard I/O pin | 50 mA |
| Sourced by any standard I/O pin | 50 mA |
| Sunk by any High Current I/O pin | 100 mA |
| Sourced by any High Current I/O pin | 100 mA |
| Clamp current, I _K (V _{PIN} < 0 or V _{PIN} > V _{DD}) | ±20 mA |
| Total power dissipation ⁽²⁾ | 800 mW |

Note 1: Maximum current rating requires even load distribution across I/O pins. Maximum current rating may be limited by the device package power dissipation characterizations, see [Table 35-6](#): “Thermal Characteristics” to calculate device specifications.

2: 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 above maximum rating conditions for extended periods may affect device reliability.

35.2 Standard Operating Conditions

The standard operating conditions for any device are defined as:

Operating Voltage: $V_{DDMIN} \leq V_{DD} \leq V_{DDMAX}$

Operating Temperature: $T_{A_MIN} \leq T_A \leq T_{A_MAX}$

V_{DD} — Operating Supply Voltage⁽¹⁾

PIC16LF1615/9

V_{DDMIN} (F_{osc} ≤ 16 MHz) +1.8V

V_{DDMIN} (F_{osc} ≤ 32 MHz) +2.5V

V_{DDMAX} +3.6V

PIC16F1615/9

V_{DDMIN} (F_{osc} ≤ 16 MHz) +2.3V

V_{DDMIN} (F_{osc} ≤ 32 MHz) +2.5V

V_{DDMAX} +5.5V

T_A — Operating Ambient Temperature Range

Industrial Temperature

T_{A_MIN} -40°C

T_{A_MAX} +85°C

Extended Temperature

T_{A_MIN} -40°C

T_{A_MAX} +125°C

Note 1: See Parameter [D001](#), DS Characteristics: Supply Voltage.

FIGURE 35-1: VOLTAGE FREQUENCY GRAPH, $-40^{\circ}\text{C} \leq T_A \leq +125^{\circ}\text{C}$, PIC16F1615/9 ONLY

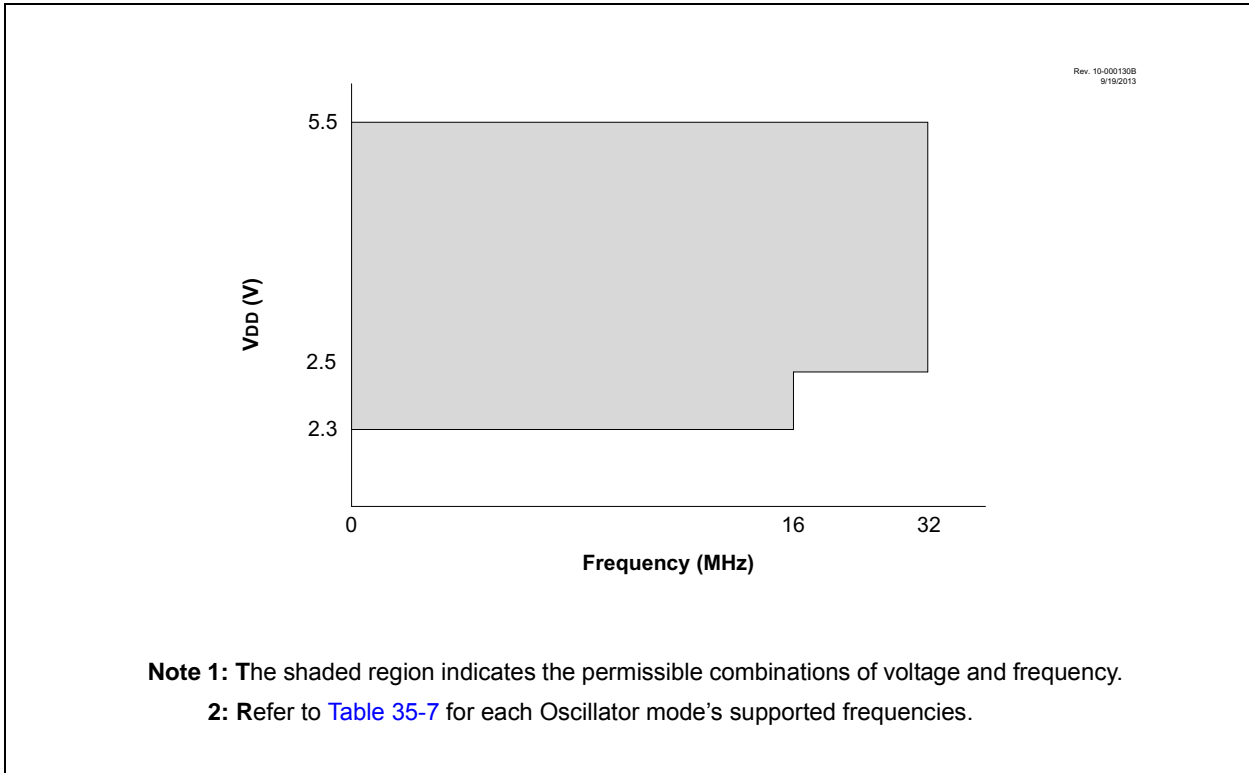
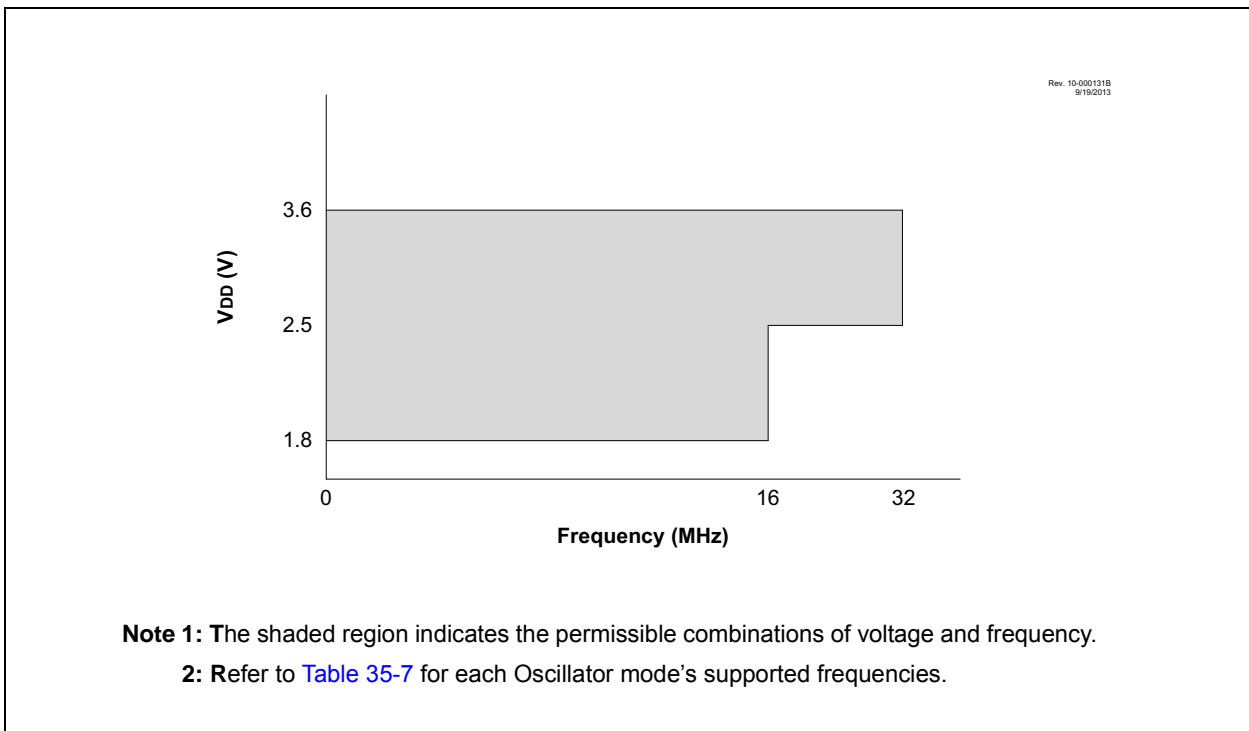


FIGURE 35-2: VOLTAGE FREQUENCY GRAPH, $-40^{\circ}\text{C} \leq T_A \leq +125^{\circ}\text{C}$, PIC16LF1615/9 ONLY



35.3 DC Characteristics

TABLE 35-1: SUPPLY VOLTAGE

| PIC16LF1615/9 | | Standard Operating Conditions (unless otherwise stated) | | | | | |
|---------------|---------|---|--------|-------|--------|-------|--|
| PIC16F1615/9 | | | | | | | |
| Param. No. | Sym. | Characteristic | Min. | Typ† | Max. | Units | Conditions |
| D001 | VDD | Supply Voltage | VDDMIN | — | VDDMAX | V | FOSC ≤ 16 MHz |
| | | | 1.8 | — | 3.6 | V | FOSC ≤ 32 MHz |
| D001 | | | 2.3 | — | 5.5 | V | FOSC ≤ 16 MHz |
| | | | 2.5 | — | 5.5 | V | FOSC ≤ 32 MHz |
| D002* | VDR | RAM Data Retention Voltage⁽¹⁾ | 1.5 | — | — | V | Device in Sleep mode |
| | | | 1.7 | — | — | V | Device in Sleep mode |
| D002A* | VPOR | Power-on Reset Release Voltage⁽²⁾ | — | 1.6 | — | V | |
| | | | — | 1.6 | — | V | |
| D002B* | VPORR* | Power-on Reset Rearm Voltage⁽²⁾ | — | 0.8 | — | V | |
| | | | — | 1.5 | — | V | |
| D003 | VFVR | Fixed Voltage Reference Voltage | — | 1.024 | — | V | -40°C ≤ TA ≤ +85°C |
| | | | — | 1.024 | — | V | -40°C ≤ TA ≤ +85°C |
| D003A | VADFVR | FVR Gain Voltage Accuracy for ADC | -4 | — | +4 | % | 1x VFVR, VDD ≥ 2.5V 2x VFVR, VDD ≥ 2.5V |
| | | | -5 | — | +5 | % | 1x VFVR, VDD ≥ 2.5V 2x VFVR, VDD ≥ 2.5V 4x VFVR, VDD ≥ 4.75V |
| D003B | VCDAFVR | FVR Gain Voltage Accuracy for Comparator/DAC | -4 | — | +4 | % | 1x VFVR, VDD ≥ 2.5V 2x VFVR, VDD ≥ 2.5V |
| | | | -7 | — | +7 | % | 1x VFVR, VDD ≥ 2.5V 2x VFVR, VDD ≥ 2.5V 4x VFVR, VDD ≥ 4.75V |
| D004* | SVDD | VDD Rise Rate⁽²⁾ | 0.05 | — | — | V/ms | Ensures that the Power-on Reset signal is released properly. |
| | | | 0.05 | — | — | V/ms | Ensures that the Power-on Reset signal is released properly. |

* 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.

Note 2: See [Figure 35-3](#), POR and POR REARM with Slow Rising VDD.

FIGURE 35-3: POR AND POR REARM WITH SLOW RISING V_{DD}



TABLE 35-2: SUPPLY CURRENT (I_{DD})^(1,2)

| PIC16LF1615/9 | | Standard Operating Conditions (unless otherwise stated) | | | | | |
|---------------|------------------------|---|------|------|---------|------------|---|
| PIC16F1615/9 | | | | | | | |
| Param. No. | Device Characteristics | Min. | Typ† | Max. | Units | Conditions | |
| | | | | | | VDD | Note |
| D013 | | — | 30 | 90 | μ A | 1.8 | Fosc = 1 MHz, External Clock (ECM), Medium-Power mode |
| | | — | 55 | 110 | μ A | 3.0 | |
| D013 | | — | 65 | 120 | μ A | 2.3 | Fosc = 1 MHz, External Clock (ECM), Medium-Power mode |
| | | — | 85 | 150 | μ A | 3.0 | |
| | | — | 115 | 200 | μ A | 5.0 | |
| D014 | | — | 115 | 260 | μ A | 1.8 | Fosc = 4 MHz, External Clock (ECM), Medium-Power mode |
| | | — | 210 | 380 | μ A | 3.0 | |
| D014 | | — | 180 | 310 | μ A | 2.3 | Fosc = 4 MHz, External Clock (ECM), Medium-Power mode |
| | | — | 240 | 410 | μ A | 3.0 | |
| | | — | 295 | 520 | μ A | 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 I_{DD} measurements in active operation mode are: OSC1 = external square wave, from rail-to-rail; all I/O pins tri-stated, pulled to V_{SS}; MCLR = V_{DD}; 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.

TABLE 35-2: SUPPLY CURRENT (IDD)^(1,2) (CONTINUED)

| PIC16LF1615/9 | | Standard Operating Conditions (unless otherwise stated) | | | | | |
|---------------|------------------------|---|------|------|-------|------------|---|
| PIC16F1615/9 | | | | | | | |
| Param. No. | Device Characteristics | Min. | Typ† | Max. | Units | Conditions | |
| | | | | | | VDD | Note |
| D015 | | — | 9.6 | 36 | μA | 1.8 | Fosc = 31 kHz, LFINTOSC, -40°C ≤ Ta ≤ +85°C |
| | | — | 16.2 | 60 | μA | 3.0 | |
| D015 | | — | 39 | 84 | μA | 2.3 | Fosc = 31 kHz, LFINTOSC, -40°C ≤ Ta ≤ +85°C |
| | | — | 45 | 90 | μA | 3.0 | |
| | | — | 51 | 108 | μA | 5.0 | |
| D016 | | — | 215 | 360 | μA | 1.8 | Fosc = 500 kHz, HFINTOSC |
| | | — | 275 | 480 | μA | 3.0 | |
| D016 | | — | 270 | 450 | μA | 2.3 | Fosc = 500 kHz, HFINTOSC |
| | | — | 300 | 500 | μA | 3.0 | |
| | | — | 350 | 620 | μA | 5.0 | |
| D017* | | — | 410 | 800 | μA | 1.8 | Fosc = 8 MHz, HFINTOSC |
| | | — | 630 | 1200 | μA | 3.0 | |
| D017* | | — | 530 | 950 | μA | 2.3 | Fosc = 8 MHz, HFINTOSC |
| | | — | 660 | 1300 | μA | 3.0 | |
| | | — | 730 | 1400 | μA | 5.0 | |
| D018 | | — | 600 | 1200 | μA | 1.8 | Fosc = 16 MHz, HFINTOSC |
| | | — | 970 | 1850 | μA | 3.0 | |
| D018 | | — | 780 | 1500 | μA | 2.3 | Fosc = 16 MHz, HFINTOSC |
| | | — | 1000 | 1900 | μA | 3.0 | |
| | | — | 1090 | 2100 | 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 VSS; 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.

TABLE 35-2: SUPPLY CURRENT (IDD)^(1,2) (CONTINUED)

| PIC16LF1615/9 | | Standard Operating Conditions (unless otherwise stated) | | | | | |
|---------------|------------------------|---|------|------|-------|------------|--|
| PIC16F1615/9 | | | | | | | |
| Param. No. | Device Characteristics | Min. | Typ† | Max. | Units | Conditions | |
| | | | | | | VDD | Note |
| D019 | | — | 1.6 | 5.0 | mA | 3.0 | Fosc = 32 MHz, HFINTOSC |
| | | — | 1.9 | 6.0 | mA | 3.6 | |
| D019 | | — | 1.6 | 5.0 | mA | 3.0 | Fosc = 32 MHz, HFINTOSC |
| | | — | 1.9 | 6.0 | mA | 5.0 | |
| D020A | | — | 1.6 | 5.0 | mA | 3.0 | Fosc = 32 MHz, External Clock (ECH), High-Power mode |
| | | — | 1.9 | 6.0 | mA | 3.6 | |
| D020A | | — | 1.6 | 5.0 | mA | 3.0 | Fosc = 32 MHz, External Clock (ECH), High-Power mode |
| | | — | 1.9 | 6.0 | mA | 5.0 | |
| D020B | | — | 6 | 16 | μA | 1.8 | Fosc = 32 kHz, External Clock (ECL), Low-Power mode |
| | | — | 8 | 22 | μA | 3.0 | |
| D020B | | — | 13 | 43 | μA | 2.3 | Fosc = 32 kHz, External Clock (ECL), Low-Power mode |
| | | — | 15 | 55 | μA | 3.0 | |
| | | — | 16 | 57 | μA | 5.0 | |
| D020C | | — | 19 | 40 | μA | 1.8 | Fosc = 500 kHz, External Clock (ECL), Low-Power mode |
| | | — | 32 | 60 | μA | 3.0 | |
| D020C | | — | 31 | 60 | μA | 2.3 | Fosc = 500 kHz, External Clock (ECL), Low-Power mode |
| | | — | 38 | 90 | μA | 3.0 | |
| | | — | 44 | 100 | 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 VSS; 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.

TABLE 35-3: POWER-DOWN CURRENTS (IPD)^(1,2)

| PIC16LF1615/9 | | Operating Conditions: (unless otherwise stated) Low-Power Sleep Mode | | | | | | |
|---------------|------------------------|---|-------|------------|-------------|-------|------------|---|
| PIC16F1615/9 | | Low-Power Sleep Mode, VREGPM = 1 | | | | | | |
| Param. No. | Device Characteristics | Min. | Typ† | Max. +85°C | Max. +125°C | Units | Conditions | |
| | | | | | | | VDD | Note |
| D022 | Base IPD | — | 0.020 | 1.0 | 8.0 | μA | 1.8 | WDT, BOR, FVR disabled, all Peripherals inactive |
| | | — | 0.025 | 2.0 | 9.0 | μA | 3.0 | |
| D022 | Base IPD | — | 0.25 | 3.0 | 10 | μA | 2.3 | WDT, BOR, FVR disabled, all Peripherals inactive, Low-Power Sleep mode |
| | | — | 0.30 | 4.0 | 12 | μA | 3.0 | |
| | | — | 0.40 | 6.0 | 15 | μA | 5.0 | |
| D022A | Base IPD | — | 9.8 | 16 | 18 | μA | 2.3 | WDT, BOR, FVR disabled, all Peripherals inactive, Normal-Power Sleep mode, VREGPM = 0 |
| | | — | 10.3 | 18 | 20 | μA | 3.0 | |
| | | — | 11.5 | 21 | 26 | μA | 5.0 | |
| D023 | | — | 0.26 | 2.0 | 9.0 | μA | 1.8 | WDT Current |
| | | — | 0.44 | 3.0 | 10 | μA | 3.0 | |
| D023 | | — | 0.43 | 6.0 | 15 | μA | 2.3 | WDT Current |
| | | — | 0.53 | 7.0 | 20 | μA | 3.0 | |
| | | — | 0.64 | 8.0 | 22 | μA | 5.0 | |
| D023A | | — | 15 | 28 | 30 | μA | 1.8 | FVR Current |
| | | — | 18 | 30 | 33 | μA | 3.0 | |
| D023A | | — | 18 | 33 | 35 | μA | 2.3 | FVR Current |
| | | — | 19 | 35 | 37 | μA | 3.0 | |
| | | — | 20 | 37 | 39 | μA | 5.0 | |
| D024 | | — | 6.0 | 17 | 20 | μA | 3.0 | BOR Current |
| D024 | | — | 7.0 | 17 | 30 | μA | 3.0 | BOR Current |
| | | — | 8.0 | 20 | 40 | μA | 5.0 | |
| D24A | | — | 0.1 | 4.0 | 10 | μA | 3.0 | LPBOR Current |
| D24A | | — | 0.35 | 5.0 | 14 | μA | 3.0 | LPBOR Current |
| | | — | 0.45 | 8.0 | 17 | μA | 5.0 | |
| D026 | | — | 0.11 | 1.5 | 9.0 | μA | 1.8 | ADC Current (Note 3), No conversion in progress |
| | | — | 0.12 | 2.7 | 10 | μA | 3.0 | |
| D026 | | — | 0.30 | 4.0 | 11 | μA | 2.3 | ADC Current (Note 3), No conversion in progress |
| | | — | 0.35 | 5.0 | 13 | μA | 3.0 | |
| | | — | 0.45 | 8.0 | 16 | μA | 5.0 | |
| D026A* | | — | 250 | — | — | μA | 1.8 | ADC Current (Note 3), Conversion in progress |
| | | — | 250 | — | — | μA | 3.0 | |
| D026A* | | — | 280 | — | — | μA | 2.3 | ADC Current (Note 3), Conversion in progress |
| | | — | 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.

Legend: TBD = To Be Determined

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.

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 Vss.

3: ADC clock source is FRC.

TABLE 35-3: POWER-DOWN CURRENTS (IPD)^(1,2) (CONTINUED)

| PIC16LF1615/9 | | Operating Conditions: (unless otherwise stated) Low-Power Sleep Mode | | | | | | |
|---------------|------------------------|---|------|------------|-------------|-------|------------|-------------------------|
| PIC16F1615/9 | | Low-Power Sleep Mode, VREGPM = 1 | | | | | | |
| Param. No. | Device Characteristics | Min. | Typ† | Max. +85°C | Max. +125°C | Units | Conditions | |
| | | | | | | | VDD | Note |
| D027 | | — | 7 | 22 | 25 | μA | 1.8 | Comparator, CxSP = 0 |
| | | — | 8 | 23 | 27 | μA | 3.0 | |
| D027 | | — | 17 | 35 | 37 | μA | 2.3 | Comparator, CxSP = 0 |
| | | — | 18 | 37 | 38 | μA | 3.0 | |
| | | — | 19 | 38 | 40 | 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.

Legend: TBD = To Be Determined

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.

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 Vss.

3: ADC clock source is FRC.

TABLE 35-4: I/O PORTS

| Standard Operating Conditions (unless otherwise stated) | | | | | | | |
|---|------------------|--|----------------------------|------|----------------------|-------|--|
| Param. No. | Sym. | Characteristic | Min. | Typ† | Max. | Units | Conditions |
| D030 D030A D031 D032 | V _{IL} | Input Low Voltage | | | | | |
| | | I/O PORT: | | | | | |
| | | with TTL buffer | — | — | 0.8 | V | 4.5V ≤ V _{DD} ≤ 5.5V |
| | | with Schmitt Trigger buffer | — | — | 0.15 V _{DD} | V | 1.8V ≤ V _{DD} ≤ 4.5V |
| | | MCLR | — | — | 0.2 V _{DD} | V | 2.0V ≤ V _{DD} ≤ 5.5V |
| D040 D040A D041 D042 | V _{IH} | Input High Voltage | | | | | |
| | | I/O PORT: | | | | | |
| | | with TTL buffer | 2.0 | — | — | V | 4.5V ≤ V _{DD} ≤ 5.5V |
| | | with Schmitt Trigger buffer | 0.25 V _{DD} + 0.8 | — | — | V | 1.8V ≤ V _{DD} ≤ 4.5V |
| | | MCLR | 0.8 V _{DD} | — | — | V | 2.0V ≤ V _{DD} ≤ 5.5V |
| | | | 0.8 V _{DD} | — | — | V | |
| D060 D061 | I _{IL} | Input Leakage Current⁽¹⁾ | | | | | |
| | | I/O Ports | — | ± 5 | ± 125 | nA | V _{SS} ≤ V _{PIN} ≤ V _{DD} , Pin at high-impedance, 85°C |
| | | | — | ± 5 | ± 1000 | nA | V _{SS} ≤ V _{PIN} ≤ V _{DD} , Pin at high-impedance, 125°C |
| | | MCLR ⁽³⁾ | — | ± 50 | ± 200 | nA | V _{SS} ≤ V _{PIN} ≤ V _{DD} , Pin at high-impedance, 85°C |
| D070* | I _{PUR} | Weak Pull-up Current | | | | | |
| | | | 25 | 100 | 200 | μA | V _{DD} = 3.3V, V _{PIN} = V _{SS} |
| | | | 25 | 140 | 300 | μA | V _{DD} = 5.0V, V _{PIN} = V _{SS} |
| D080 D080A | V _{OL} | Output Low Voltage⁽³⁾ | | | | | |
| | | I/O Ports | — | — | 0.6 | V | I _{OL} = 8.0 mA, V _{DD} = 5.0V I _{OL} = 6.0 mA, V _{DD} = 3.3V I _{OL} = 1.8 mA, V _{DD} = 1.8V |
| | | High Drive I/O ⁽¹⁾ | — | 2.5V | — | V | I _{OL} = 100 mA, V _{DD} = 5.0V |
| D090 D090A | V _{OH} | Output High Voltage⁽³⁾ | | | | | |
| | | I/O Ports | V _{DD} - 0.7 | — | — | V | I _{OH} = 3.5 mA, V _{DD} = 5.0V I _{OH} = 3.0 mA, V _{DD} = 3.3V I _{OH} = 1.0 mA, V _{DD} = 1.8V |
| | | High Drive I/O ⁽¹⁾ | — | 2.5V | — | V | I _{OL} = 100 mA, V _{DD} = 5.0V |
| D101A* | C _{IO} | All I/O pins | — | — | 50 | pF | |

* 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: Negative current is defined as current sourced by the pin.
 - 2: 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.
 - 3: Excluding OSC2 in CLKOUT mode.

TABLE 35-5: MEMORY PROGRAMMING SPECIFICATIONS

| Standard Operating Conditions (unless otherwise stated) | | | | | | | |
|---|----------|---|--------|------|--------|-------|---|
| 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 | I PPPGM | Current on $\overline{\text{MCLR}}/\text{VPP}$ during Erase/Write | — | 1.0 | — | mA | |
| D115 | I DD PGM | Current on VDD during Erase/Write | — | 5.0 | — | mA | |
| Program Flash Memory | | | | | | | |
| D121 | EP | Cell Endurance | 10K | — | — | E/W | -40°C ≤ TA ≤ +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 ≤ TA ≤ +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.

TABLE 35-6: THERMAL CHARACTERISTICS

Standard Operating Conditions (unless otherwise stated)

| Param. No. | Sym. | Characteristic | Typ. | Units | Conditions |
|------------|---------------|--|------|-------|---|
| TH01 | θ_{JA} | Thermal Resistance Junction to Ambient | 62.2 | °C/W | 20-pin DIP package |
| | | | 77.7 | °C/W | 20-pin SOIC package |
| | | | 87.3 | °C/W | 20-pin SSOP package |
| | | | 43 | °C/W | 20-pin QFN 4X4mm package |
| TH02 | θ_{JC} | Thermal Resistance Junction to Case | 27.5 | °C/W | 20-pin DIP package |
| | | | 23.1 | °C/W | 20-pin SOIC package |
| | | | 31.1 | °C/W | 20-pin SSOP package |
| | | | 5.3 | °C/W | 20-pin QFN 4X4mm package |
| TH03 | TJMAX | Maximum Junction Temperature | 150 | °C | |
| TH04 | PD | Power Dissipation | — | W | PD = PINTERNAL + PI/O |
| TH05 | PINTERNAL | Internal Power Dissipation | — | W | PINTERNAL = IDD x VDD ⁽¹⁾ |
| TH06 | PI/O | I/O Power Dissipation | — | W | PI/O = $\Sigma (I_{OL} * V_{OL}) + \Sigma (I_{OH} * (V_{DD} - V_{OH}))$ |
| TH07 | PDER | Derated Power | — | W | PDER = PDMAX (TJ - TA)/ θ_{JA} ⁽²⁾ |

Note 1: IDD is current to run the chip alone without driving any load on the output pins.

Note 2: TA = Ambient Temperature; TJ = Junction Temperature

35.4 AC Characteristics

Timing Parameter Symbology has been created with one of the following formats:

1. TppS2ppS
2. TppS

| | |
|--|---|
| <p>T</p> <p>F Frequency</p> | <p>T Time</p> |
| <p>Lowercase letters (pp) and their meanings:</p> | |
| <p>pp</p> <p>cc CCP1</p> <p>ck CLKOUT</p> <p>cs \overline{CS}</p> <p>di SDIx</p> <p>do SDO</p> <p>dt Data in</p> <p>io I/O PORT</p> <p>mc \overline{MCLR}</p> | <p>osc CLKIN</p> <p>rd \overline{RD}</p> <p>rw \overline{RD} or \overline{WR}</p> <p>sc SCKx</p> <p>ss \overline{SS}</p> <p>t0 T0CKI</p> <p>t1 T1CKI</p> <p>wr \overline{WR}</p> |
| <p>Uppercase letters and their meanings:</p> | |
| <p>S</p> <p>F Fall</p> <p>H High</p> <p>I Invalid (High-impedance)</p> <p>L Low</p> | <p>P Period</p> <p>R Rise</p> <p>V Valid</p> <p>Z High-impedance</p> |

FIGURE 35-4: LOAD CONDITIONS

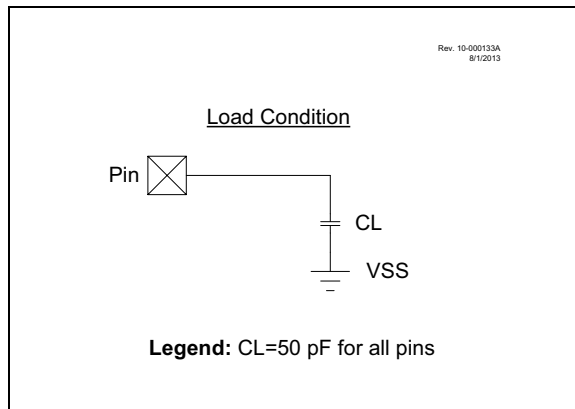


FIGURE 35-5: CLOCK TIMING

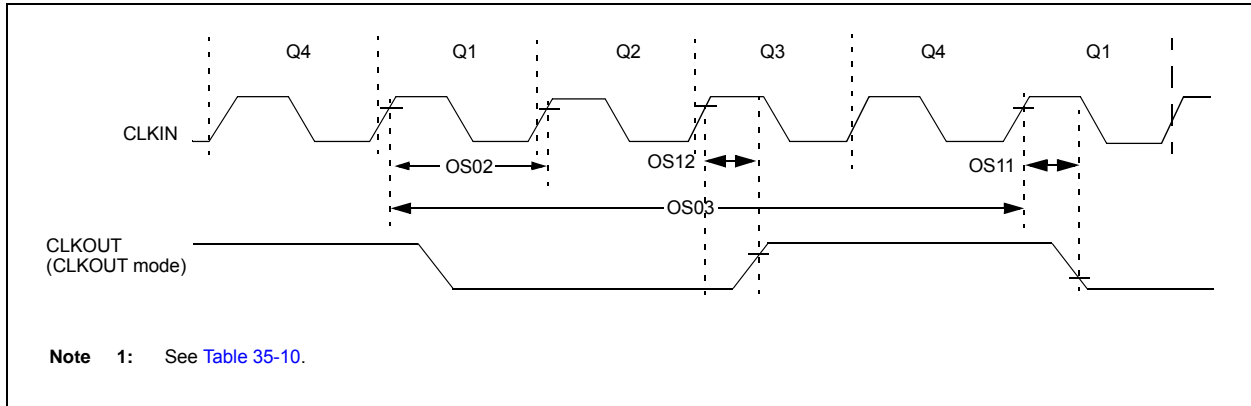


TABLE 35-7: CLOCK OSCILLATOR TIMING REQUIREMENTS

Standard Operating Conditions (unless otherwise stated)

| 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 | — | 32 | MHz | External Clock (ECH) |
| OS02 | Tosc | External CLKIN Period ⁽¹⁾ | 31.25 | — | ∞ | ns | External Clock (EC) |
| OS03 | Tcy | Instruction Cycle Time ⁽¹⁾ | 200 | Tcy | DC | ns | Tcy = 4/Fosc |

* 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 CLKIN pin. When an external clock input is used, the "max" cycle time limit is "DC" (no clock) for all devices.

TABLE 35-8: OSCILLATOR PARAMETERS

| Standard Operating Conditions (unless otherwise stated) | | | | | | | | |
|---|-------|---|-----------------|------|------|------|-------|------------|
| Param. No. | Sym. | Characteristic | Freq. Tolerance | Min. | Typ† | Max. | Units | Conditions |
| OS08 | HFosc | Internal Calibrated HFINTOSC Frequency ⁽¹⁾ | — | — | 16.0 | — | MHz | (Note 2) |
| OS09 | LFosc | Internal LFINTOSC Frequency | — | — | 31 | — | kHz | (Note 3) |
| OS10 | TWARM | HFINTOSC Wake-up from Sleep Start-up Time | — | — | 5 | 15 | μs | |
| | | LFINTOSC Wake-up from Sleep Start-up Time | — | — | 0.5 | — | ms | |

* 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, VDD and VSS must be capacitively decoupled as close to the device as possible. 0.1 μF and 0.01 μF values in parallel are recommended.

2: See Figure 35-6: “HFINTOSC Frequency Accuracy over Device VDD and Temperature”,

3: See Figure 36-43: “LFINTOSC Frequency over VDD and Temperature, PIC16LF1615/9 Only”, and Figure 36-44: “LFINTOSC Frequency over VDD and Temperature, PIC16F1615/9 Only”.

FIGURE 35-6: HFINTOSC FREQUENCY ACCURACY OVER VDD AND TEMPERATURE

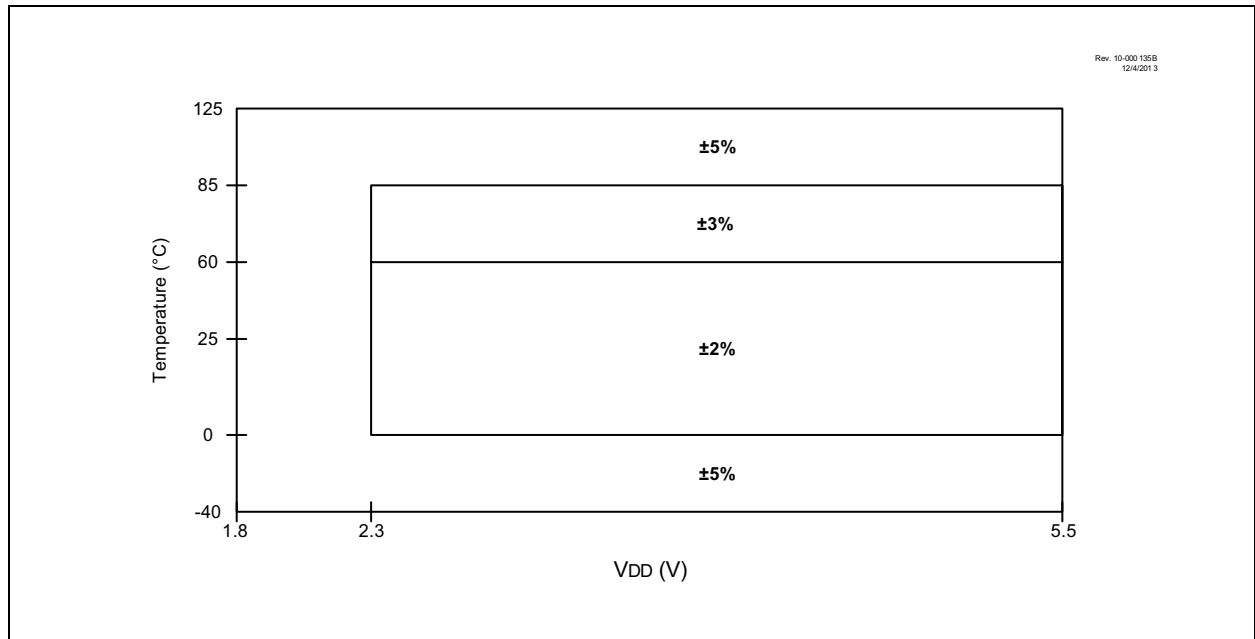


TABLE 35-9: PLL CLOCK TIMING SPECIFICATIONS

| Standard Operating Conditions (unless otherwise stated) | | | | | | | |
|--|-------------|-------------------------------|-------------|-------------|-------------|--------------|-------------------|
| Param No. | Sym. | Characteristic | Min. | Typ† | Max. | Units | Conditions |
| F10 | FOSC | Oscillator Frequency Range | 4 | — | 8 | MHz | |
| F11 | FSYS | On-Chip VCO System Frequency | 16 | — | 32 | MHz | |
| F12 | TRC | PLL Start-up Time (Lock Time) | — | — | 2 | ms | |
| F13* | ΔCLK | CLKOUT Stability (Jitter) | -0.25% | — | +0.25% | % | |

* These parameters are characterized but not tested.

† Data in “Typ” column is at 5V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

FIGURE 35-7: CLKOUT AND I/O TIMING

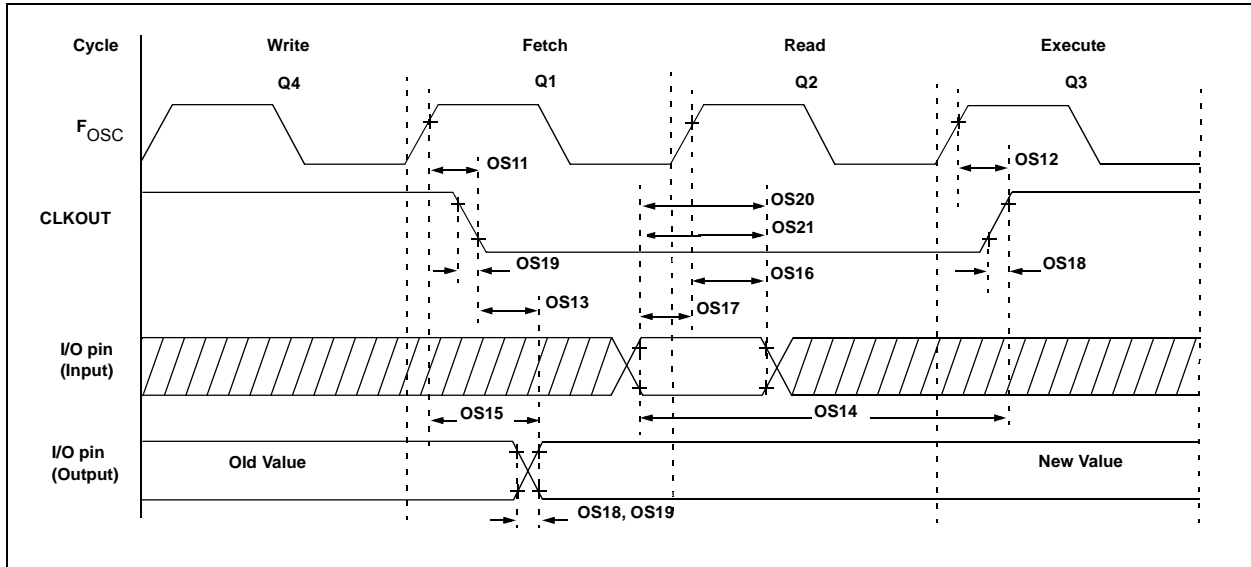


TABLE 35-10: CLKOUT AND I/O TIMING PARAMETERS

| Standard Operating Conditions (unless otherwise stated) | | | | | | | |
|---|----------|--|---------------|----------|----------|-------|---------------------------------|
| Param. No. | Sym. | Characteristic | Min. | Typ† | Max. | Units | Conditions |
| OS11 | TosH2ckL | Fosc↑ to CLKOUT↓ ⁽¹⁾ | — | — | 70 | ns | 3.3V ≤ VDD ≤ 5.0V |
| OS12 | TosH2ckH | Fosc↑ to CLKOUT↑ ⁽¹⁾ | — | — | 72 | ns | 3.3V ≤ VDD ≤ 5.0V |
| OS13 | TckL2ioV | CLKOUT↓ to Port out valid ⁽¹⁾ | — | — | 20 | ns | |
| OS14 | TioV2ckH | Port input valid before CLKOUT↑ ⁽¹⁾ | Tosc + 200 ns | — | — | ns | |
| OS15 | TosH2ioV | Fosc↑ (Q1 cycle) to Port out valid | — | 50 | 70* | ns | 3.3V ≤ VDD ≤ 5.0V |
| OS16 | TosH2ioI | Fosc↑ (Q2 cycle) to Port input invalid (I/O in setup time) | 50 | — | — | ns | 3.3V ≤ VDD ≤ 5.0V |
| OS17 | TioV2osH | Port input valid to Fosc↑ (Q2 cycle) (I/O in setup time) | 20 | — | — | ns | |
| OS18* | TioR | Port output rise time | — | 40 15 | 72 32 | ns | VDD = 1.8V 3.3V ≤ VDD ≤ 5.0V |
| OS19* | TioF | Port output fall time | — | 28 15 | 55 30 | ns | VDD = 1.8V 3.3V ≤ VDD ≤ 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 EXTRC mode where CLKOUT output is 4 x Tosc.

FIGURE 35-8: RESET, WATCHDOG TIMER, OSCILLATOR START-UP TIMER AND POWER-UP TIMER TIMING

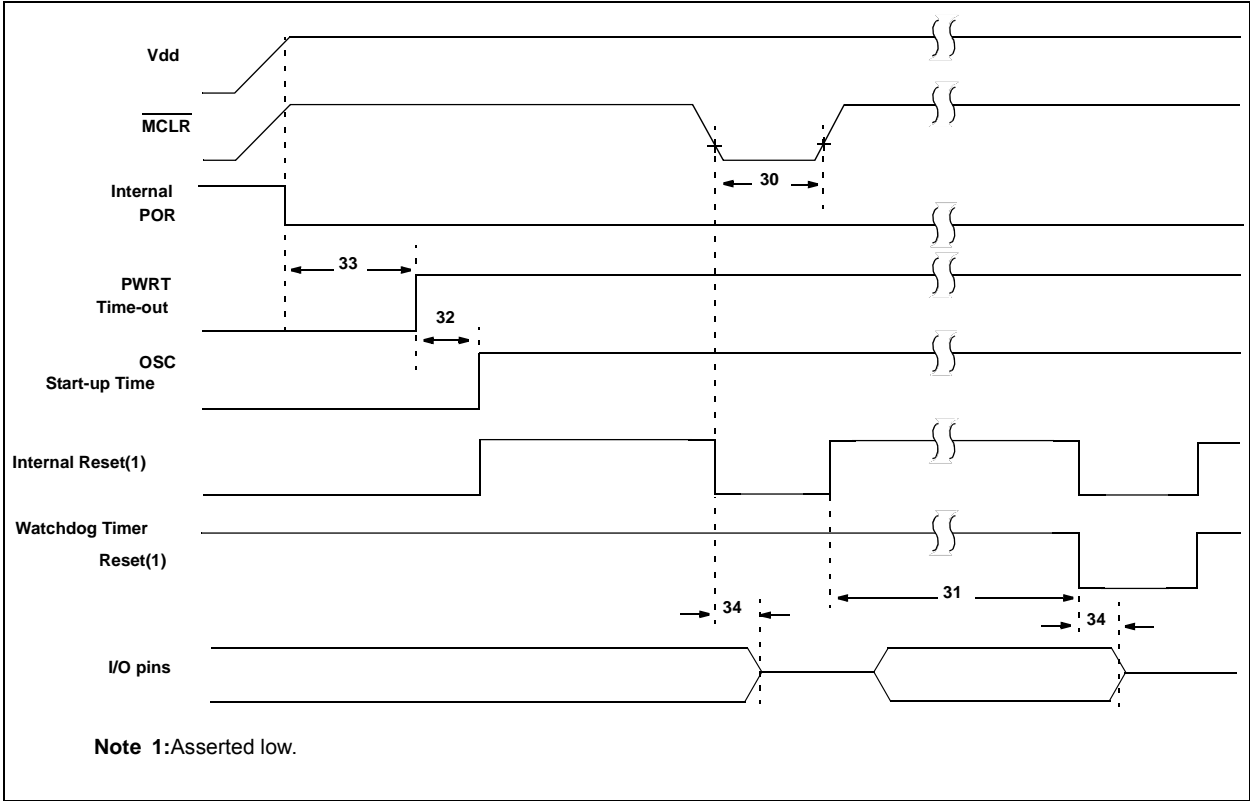


TABLE 35-11: RESET, WATCHDOG TIMER, OSCILLATOR START-UP TIMER, POWER-UP TIMER AND BROWN-OUT RESET PARAMETERS

| Standard Operating Conditions (unless otherwise stated) | | | | | | | |
|---|--------|--|------|------|------|-------|-------------------------------------|
| Param. No. | Sym. | Characteristic | Min. | Typ† | Max. | Units | Conditions |
| 30 | TMCL | MCLR Pulse Width (low) | 2 | — | — | μs | |
| 31 | TWDTLP | Low-Power Watchdog Timer Time-out Period | 10 | 16 | 27 | ms | VDD = 3.3V-5V, 1:512 Prescaler used |
| 32 | TOST | Oscillator Start-up Timer Period ⁽¹⁾ | — | 1024 | — | TOSC | |
| 33* | TPWRT | Power-up Timer Period | 40 | 65 | 140 | ms | PWRT \overline{E} = 0 |
| 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 |
| | | | 2.35 | 2.45 | 2.58 | V | BORV = 1 (PIC16F1615/9) |
| | | | 1.80 | 1.90 | 2.05 | V | BORV = 1 (PIC16LF1615/9) |
| 36* | VHYST | Brown-out Reset Hysteresis | 0 | 25 | 60 | mV | -40°C ≤ TA ≤ +85°C |
| 37* | TBORDC | Brown-out Reset DC Response Time | 1 | 16 | 35 | μs | VDD ≤ VBOR |
| 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.

Note 2: To ensure these voltage tolerances, VDD and VSS must be capacitively decoupled as close to the device as possible. 0.1 μF and 0.01 μF values in parallel are recommended.

FIGURE 35-9: BROWN-OUT RESET TIMING AND CHARACTERISTICS

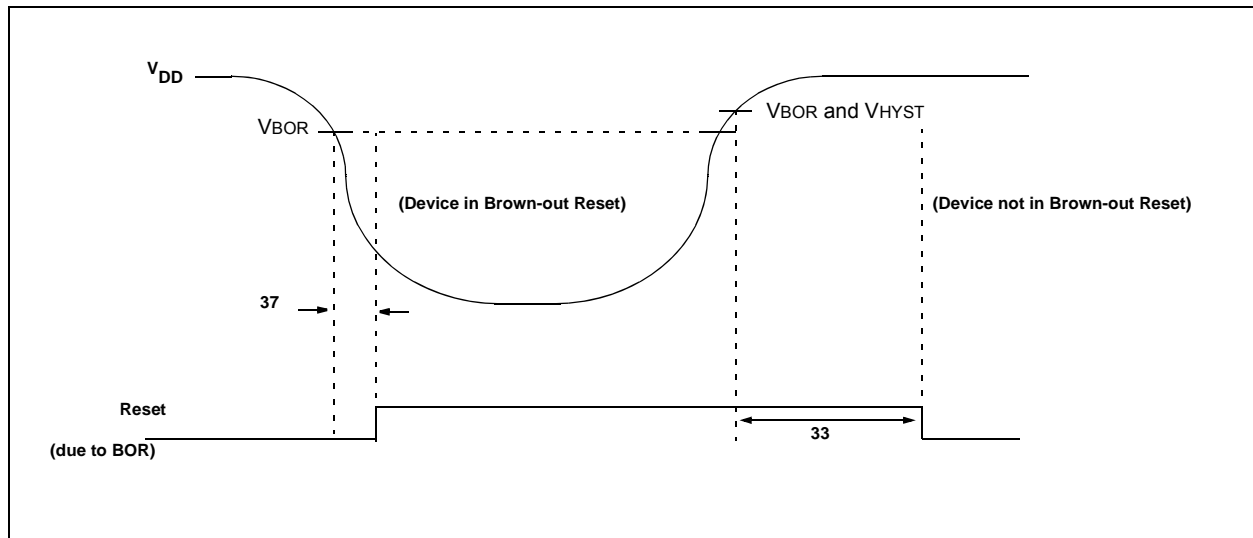


FIGURE 35-10: TIMER0 AND TIMER1 EXTERNAL CLOCK TIMINGS



TABLE 35-12: TIMER0 AND TIMER1 EXTERNAL CLOCK REQUIREMENTS

| Standard Operating Conditions (unless otherwise stated) | | | | | | | | |
|---|-----------|---|-----------------------------|--|------|-------------|-------|---------------------|
| 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 | |
| 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.

TABLE 35-13: ANALOG-TO-DIGITAL CONVERTER (ADC) CHARACTERISTICS^(1,2,3)

| Operating Conditions (unless otherwise stated) V _{DD} = 3.0V, T _A = 25°C | | | | | | | |
|---|------------------|--|-----------------|------|------------------|-------|---|
| Param. No. | Sym. | Characteristic | Min. | Typ† | Max. | Units | Conditions |
| AD01 | NR | Resolution | — | — | 10 | bit | |
| AD02 | EIL | Integral Error | — | ±1 | ±1.7 | LSb | V _{REF} = 3.0V |
| AD03 | EDL | Differential Error | — | ±1 | ±1 | LSb | No missing codes V _{REF} = 3.0V |
| AD04 | EOFF | Offset Error | — | ±1 | ±2.5 | LSb | V _{REF} = 3.0V |
| AD05 | EGN | Gain Error | — | ±1 | ±2.0 | LSb | V _{REF} = 3.0V |
| AD06 | V _{REF} | Reference Voltage | 1.8 | — | V _{DD} | V | V _{REF} = (V _{RPOS} - V _{RNEG}) (Note 4) |
| AD07 | V _{AIN} | Full-Scale Range | V _{SS} | — | V _{REF} | V | |
| AD08 | Z _{AIN} | Recommended Impedance of Analog Voltage Source | — | — | 10 | kΩ | Can go higher if external 0.01μF capacitor is present on input pin. |

* 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: Total Absolute Error includes integral, differential, offset and gain errors.

2: The ADC conversion result never decreases with an increase in the input voltage and has no missing codes.

3: See **Section 36.0 “DC and AC Characteristics Graphs and Charts”** for operating characterization.

4: ADC V_{REF} is selected by ADPREF<0> bit.

FIGURE 35-11: ADC CONVERSION TIMING (ADC CLOCK Fosc-BASED)

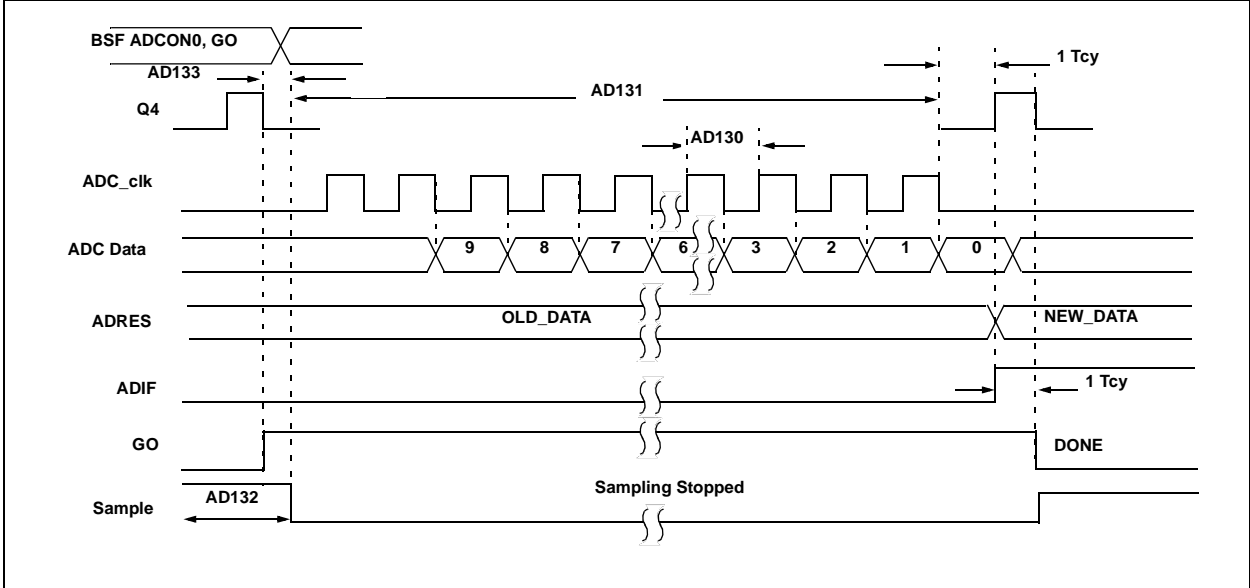


FIGURE 35-12: ADC CONVERSION TIMING (ADC CLOCK FROM FRC)

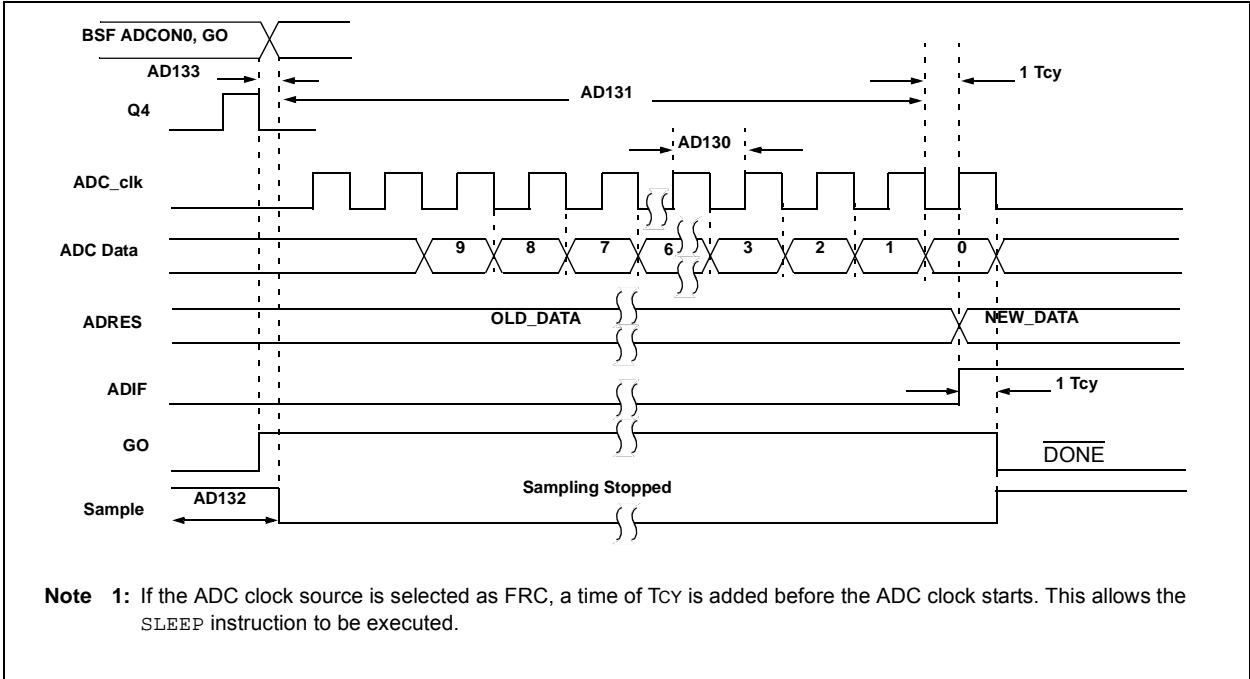


TABLE 35-14: ADC CONVERSION REQUIREMENTS

| Standard Operating Conditions (unless otherwise stated) | | | | | | | |
|---|------|---|------|----------------|------|-------|--|
| Param. No. | Sym. | Characteristic | Min. | Typ† | Max. | Units | Conditions |
| AD130* | TAD | ADC Clock Period (TADC) | 1.0 | — | 6.0 | μs | FOSC-based |
| | | ADC Internal FRC Oscillator Period (TFRC) | 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 Time | — | 1/2 TAD | — | | FOSC-based |
| | | | — | 1/2 TAD + 1TCY | — | | 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 T_{cy} cycle.

TABLE 35-15: COMPARATOR SPECIFICATIONS⁽¹⁾

| Operating Conditions (unless otherwise stated) | | | | | | | |
|--|----------------------------------|--|------|------|-----------------|-------|---|
| V _{DD} = 3.0V, T _A = 25°C | | | | | | | |
| Param. No. | Sym. | Characteristics | Min. | Typ. | Max. | Units | Comments |
| CM01 | V _{ioff} | Input Offset Voltage | — | ±7.5 | ±60 | mV | CxSP = 1, V _{icm} = V _{DD} /2 |
| CM02 | V _{icm} | Input Common Mode Voltage | 0 | — | V _{DD} | V | |
| CM03 | CMRR | Common Mode Rejection Ratio | — | 50 | — | dB | |
| CM04A | T _{resp} ⁽²⁾ | Response Time Rising Edge | — | 400 | 800 | ns | CxSP = 1 |
| CM04B | | Response Time Falling Edge | — | 200 | 400 | ns | CxSP = 1 |
| CM04C | | Response Time Rising Edge | — | 1200 | — | ns | CxSP = 0 |
| CM04D | | Response Time Falling Edge | — | 550 | — | ns | CxSP = 0 |
| CM05* | T _{mc2ov} | Comparator Mode Change to Output Valid | — | — | 10 | μs | |
| CM06 | CHYSTER | Comparator Hysteresis | — | 25 | — | mV | CxHYS = 1, CxSP = 1 |

* These parameters are characterized but not tested.

Note 1: See Section 36.0 "DC and AC Characteristics Graphs and Charts" for operating characterization.

2: Response time measured with one comparator input at V_{DD}/2, while the other input transitions from V_{SS} to V_{DD}.

TABLE 35-16: DIGITAL-TO-ANALOG CONVERTER (DAC) SPECIFICATIONS⁽¹⁾

| Operating Conditions (unless otherwise stated) V _{DD} = 3.0V, T _A = 25°C | | | | | | | |
|---|------|------------------------------|------|----------------------|-------|-------|----------|
| Param. No. | Sym. | Characteristics | Min. | Typ. | Max. | Units | Comments |
| DAC01* | CLSB | Step Size | — | V _{DD} /256 | — | V | |
| DAC02* | CACC | Absolute Accuracy | — | — | ± 1.5 | LSb | |
| DAC03* | CR | Unit Resistor Value (R) | — | — | — | Ω | |
| DAC04* | CST | Settling Time ⁽²⁾ | — | — | 10 | μs | |

* These parameters are characterized but not tested.

Note 1: See Section 36.0 “DC and AC Characteristics Graphs and Charts” for operating characterization.

2: Settling time measured while DACR<4:0> transitions from ‘0000’ to ‘1111’.

TABLE 35-17: ZERO CROSS PIN SPECIFICATIONS

| Operating Conditions (unless otherwise stated) V _{DD} = 3.0V, T _A = 25°C | | | | | | | |
|---|--------|----------------------------|------|------|------|-------|----------|
| Param. No. | Sym. | Characteristics | Min. | Typ. | Max. | Units | Comments |
| ZC01 | ZCPINV | Voltage on Zero Cross Pin | — | 0.75 | — | V | |
| ZC02 | ZCSRC | Source current | — | -300 | -600 | μA | |
| ZC03 | ZCSNK | Sink current | — | 300 | 600 | μA | |
| ZC04 | ZCISW | Response Time Rising Edge | — | 1 | — | μs | |
| | | Response Time Falling Edge | — | 1 | — | μs | |
| ZC05 | ZCOUT | Response Time Rising Edge | — | 1 | — | μs | |
| | | Response Time Falling Edge | — | 1 | — | μs | |

* These parameters are characterized but not tested.

36.0 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 V_{DD} range). This is for **information only** and devices are ensured to operate properly only within the specified range.

Unless otherwise noted, all graphs apply to both the L and LF devices.

| |
|--|
| <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 (mean + 3σ) or (mean - 3σ) respectively, where σ is a standard deviation, over each temperature range.

Note: Unless otherwise noted, $V_{IN} = 5V$, $F_{OSC} = 500\text{ kHz}$, $C_{IN} = 0.1\ \mu\text{F}$, $T_A = 25^\circ\text{C}$.

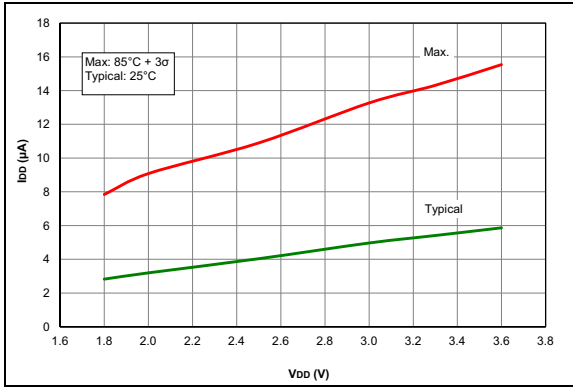


FIGURE 36-1: I_{DD} , EC Oscillator LP Mode, $F_{osc} = 32\text{ kHz}$, PIC16LF1615/9 Only.

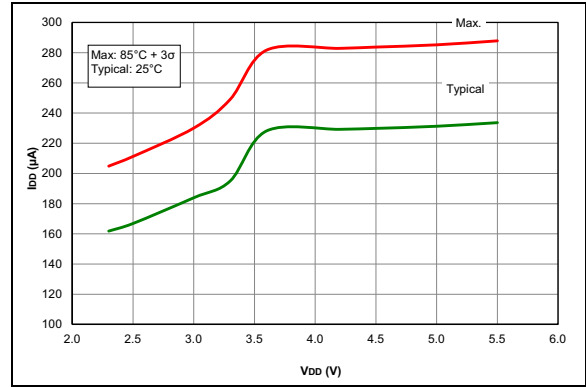


FIGURE 36-4: I_{DD} , EC Oscillator LP Mode, $F_{osc} = 500\text{ kHz}$, PIC16F1615/9 Only.

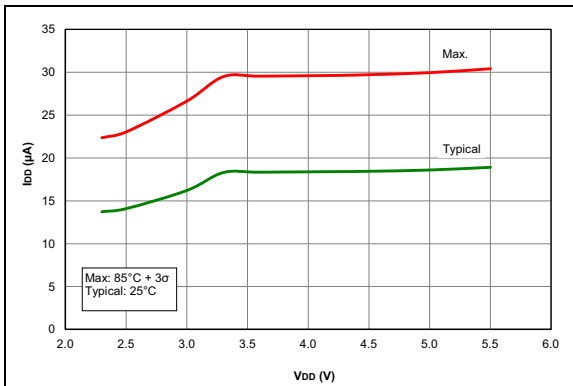


FIGURE 36-2: I_{DD} , EC Oscillator LP Mode, $F_{osc} = 32\text{ kHz}$, PIC16F1615/9 Only.

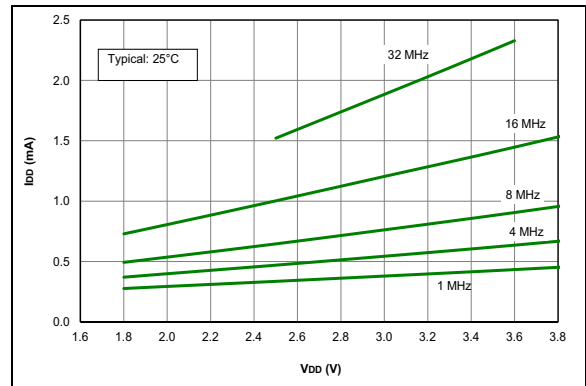


FIGURE 36-5: I_{DD} Typical, EC Oscillator MP Mode, PIC16LF1615/9 Only.

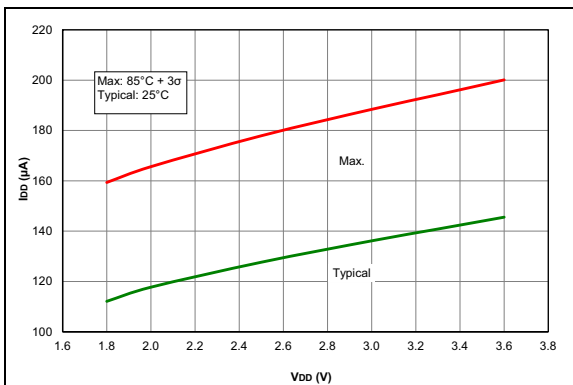


FIGURE 36-3: I_{DD} , EC Oscillator LP Mode, $F_{osc} = 500\text{ kHz}$, PIC16LF1615/9 Only.

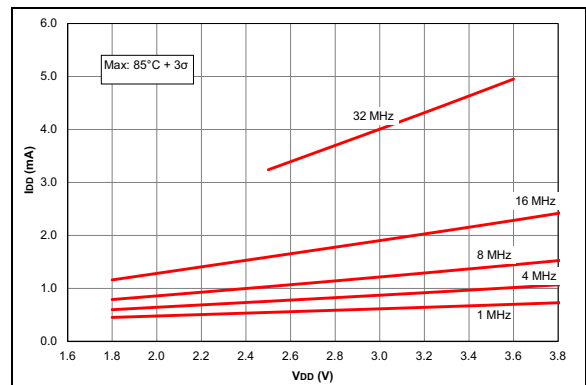


FIGURE 36-6: I_{DD} Maximum, EC Oscillator MP Mode, PIC16LF1615/9 Only.

Note: Unless otherwise noted, $V_{IN} = 5V$, $F_{OSC} = 500\text{ kHz}$, $C_{IN} = 0.1\ \mu F$, $T_A = 25^\circ C$.

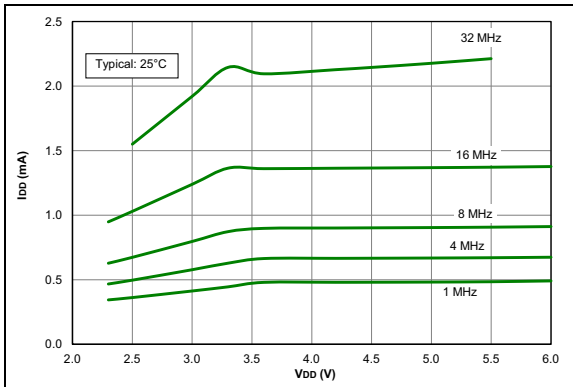


FIGURE 36-7: I_{DD} Typical, EC Oscillator MP Mode, PIC16F1615/9 Only.

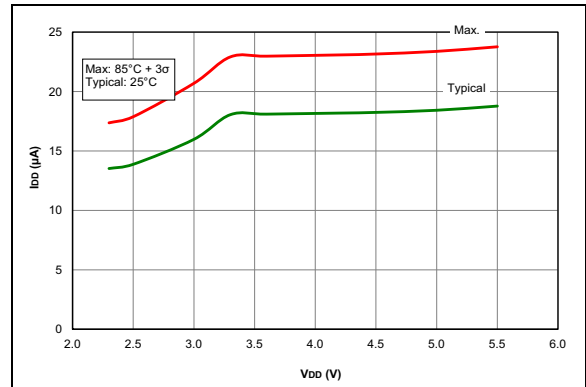


FIGURE 36-10: I_{DD} Maximum, EC Oscillator HP Mode, PIC16LF1615/9 Only.

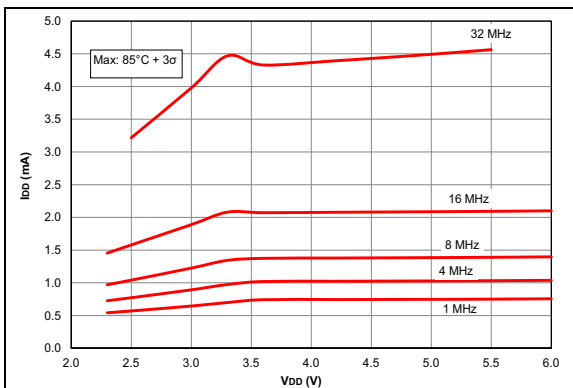


FIGURE 36-8: I_{DD} Maximum, EC Oscillator MP Mode, PIC16F1615/9 Only.

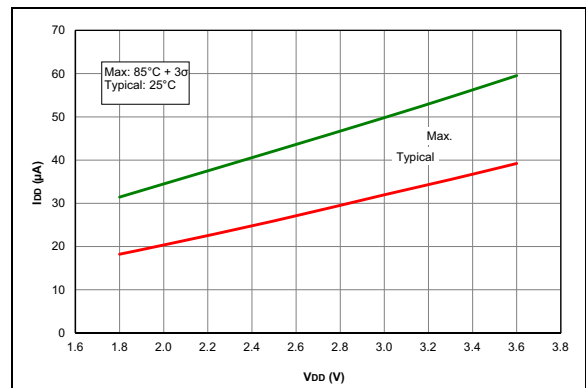


FIGURE 36-11: I_{DD} Typical, EC Oscillator HP Mode, PIC16F1615/9 Only.

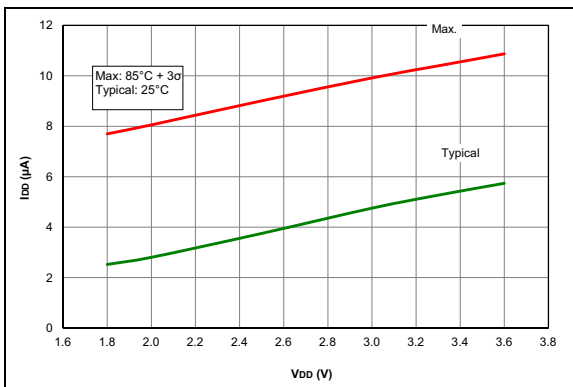


FIGURE 36-9: I_{DD} Typical, EC Oscillator HP Mode, PIC16LF1615/9 Only.

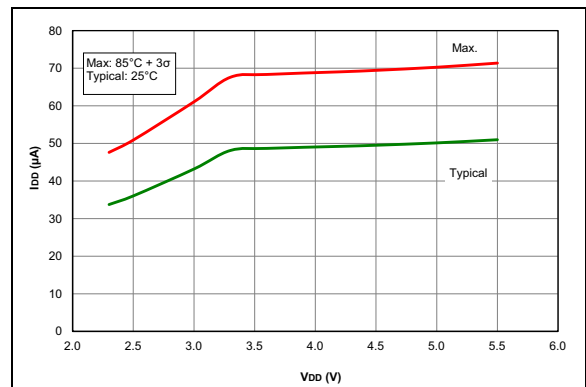


FIGURE 36-12: I_{DD} Maximum, EC Oscillator HP Mode, PIC16F1615/9 Only.

Note: Unless otherwise noted, $V_{IN} = 5V$, $F_{OSC} = 500\text{ kHz}$, $C_{IN} = 0.1\ \mu F$, $T_A = 25^\circ C$.

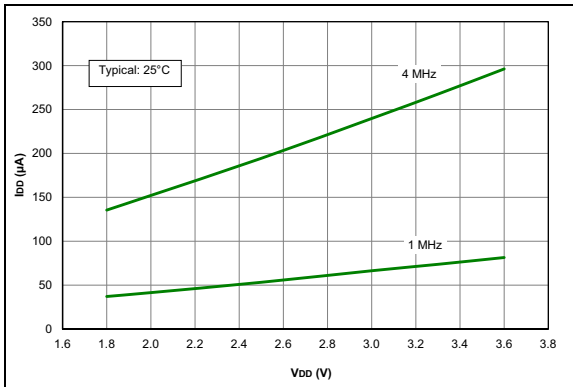


FIGURE 36-13: I_{DD} , LFINTOSC Mode, $F_{osc} = 31\text{ kHz}$, PIC16LF1615/9 Only.

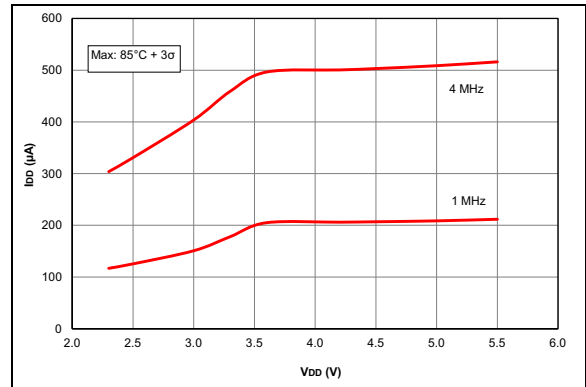


FIGURE 36-16: I_{DD} , MFINTOSC Mode, $F_{osc} = 500\text{ kHz}$, PIC16F1615/9 Only.

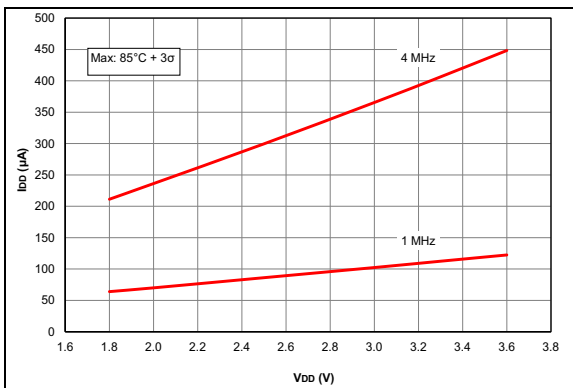


FIGURE 36-14: I_{DD} , LFINTOSC Mode, $F_{osc} = 31\text{ kHz}$, PIC16F1615/9 Only.

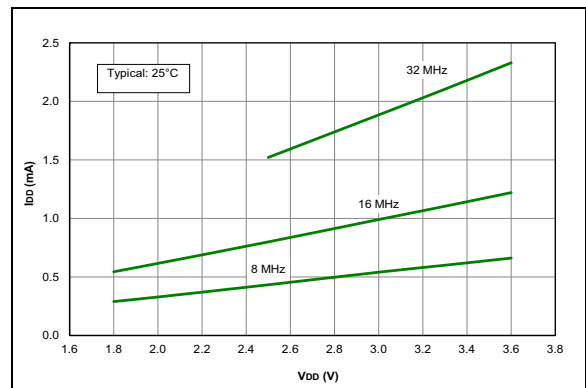


FIGURE 36-17: I_{DD} Typical, HFINTOSC Mode, PIC16LF1615/9 Only.

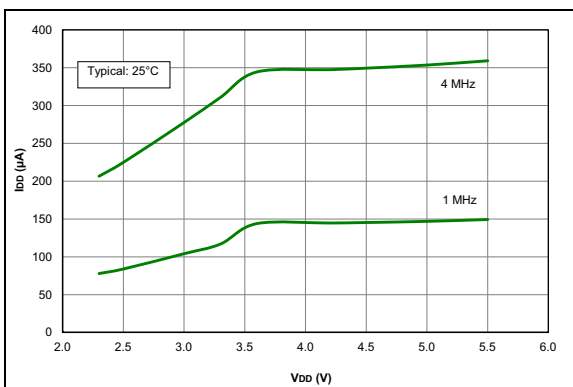


FIGURE 36-15: I_{DD} , MFINTOSC Mode, $F_{osc} = 500\text{ kHz}$, PIC16LF1615/9 Only.

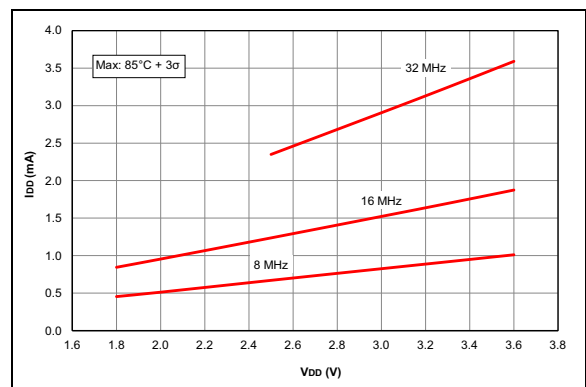


FIGURE 36-18: I_{DD} Maximum, HFINTOSC Mode, PIC16LF1615/9 Only.

Note: Unless otherwise noted, $V_{IN} = 5V$, $F_{OSC} = 500\text{ kHz}$, $C_{IN} = 0.1\ \mu F$, $T_A = 25^\circ C$.

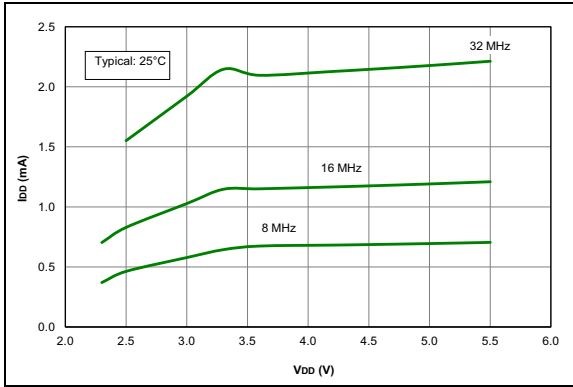


FIGURE 36-19: I_{DD} Typical, HFINTOSC Mode, PIC16F1615/9 Only.

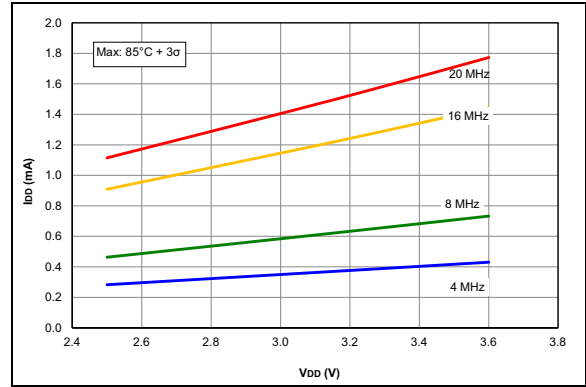


FIGURE 36-22: I_{DD} Maximum, HS Oscillator, PIC16LF1615/9 Only.

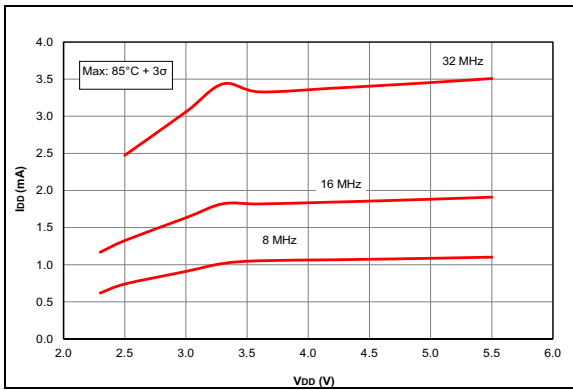


FIGURE 36-20: I_{DD} Maximum, HFINTOSC Mode, PIC16F1615/9 Only.

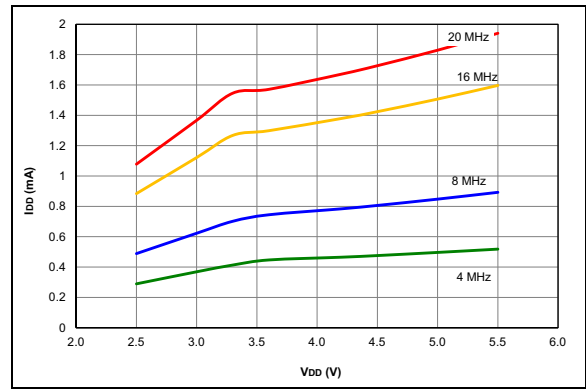


FIGURE 36-23: I_{DD} Typical, HS Oscillator, 25°C, PIC16F1615/9 Only.

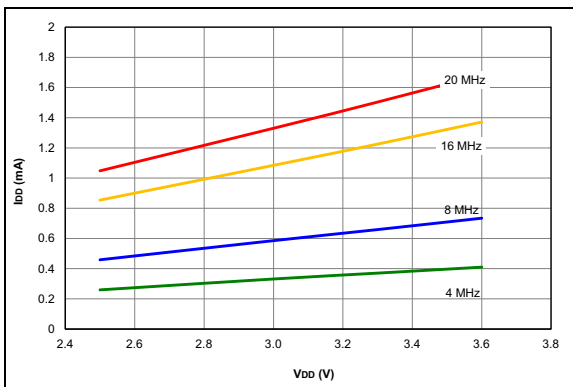


FIGURE 36-21: I_{DD} Typical, HS Oscillator, 25°C, PIC16LF1615/9 Only.

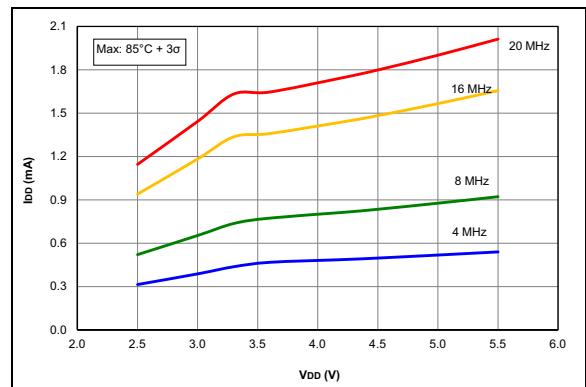


FIGURE 36-24: I_{DD} Maximum, HS Oscillator, PIC16F1615/9 Only.

Note: Unless otherwise noted, $V_{IN} = 5V$, $F_{OSC} = 500\text{ kHz}$, $C_{IN} = 0.1\ \mu F$, $T_A = 25^\circ C$.

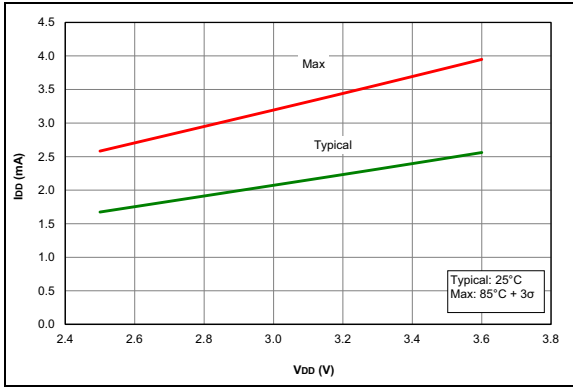


FIGURE 36-25: I_{DD} , HS Oscillator, 32 MHz (8 MHz + 4x PLL), PIC16LF1615/9 Only.

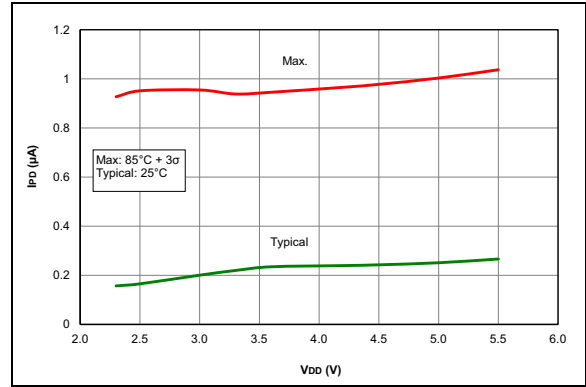


FIGURE 36-28: I_{PD} Base, LP Sleep Mode ($V_{REGPM} = 1$), PIC16F1615/9 Only.

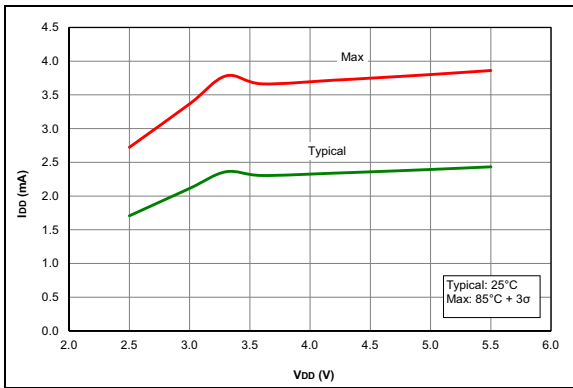


FIGURE 36-26: I_{DD} , HS Oscillator, 32 MHz (8 MHz + 4x PLL), PIC16F1615/9 Only.

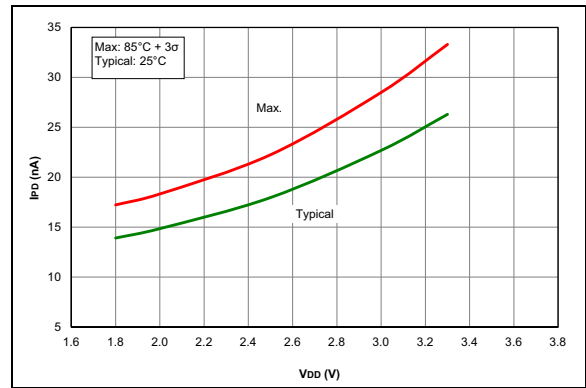


FIGURE 36-29: I_{PD} , Fixed Voltage Reference (FVR), PIC16LF1615/9 Only.

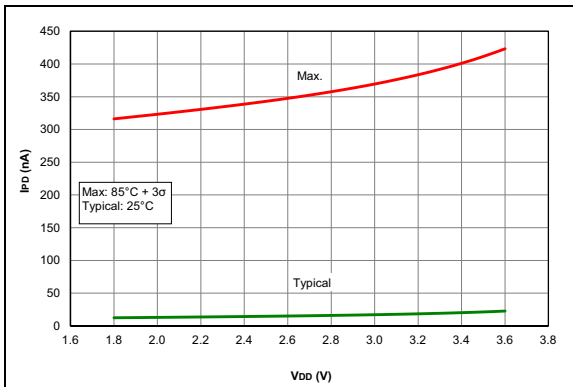


FIGURE 36-27: I_{PD} Base, LP Sleep Mode, PIC16LF1615/9 Only.

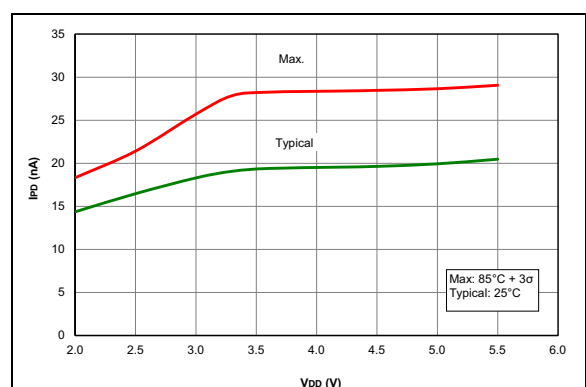


FIGURE 36-30: I_{PD} , Fixed Voltage Reference (FVR), PIC16F1615/9 Only.

Note: Unless otherwise noted, $V_{IN} = 5V$, $F_{OSC} = 500\text{ kHz}$, $C_{IN} = 0.1\ \mu F$, $T_A = 25^\circ C$.

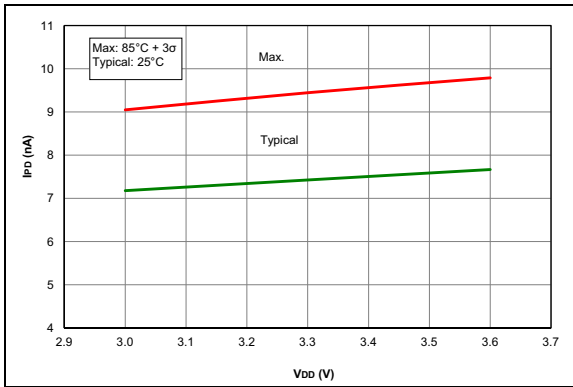


FIGURE 36-31: I_{PD} , Brown-Out Reset (BOR), BORV = 1, PIC16LF1615/9 Only.

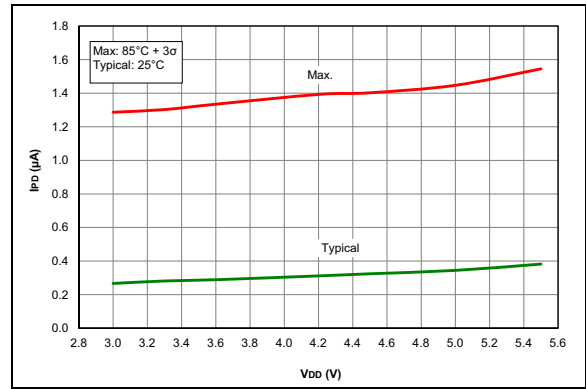


FIGURE 36-34: I_{PD} , LP Brown-Out Reset (LPBOR = 0), PIC16F1615/9 Only.

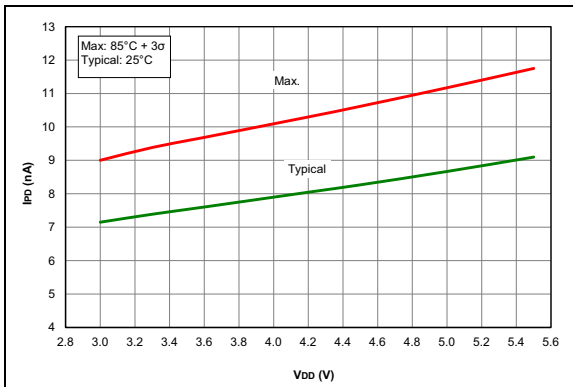


FIGURE 36-32: I_{PD} , Brown-Out Reset (BOR), BORV = 1, PIC16F1615/9 Only.

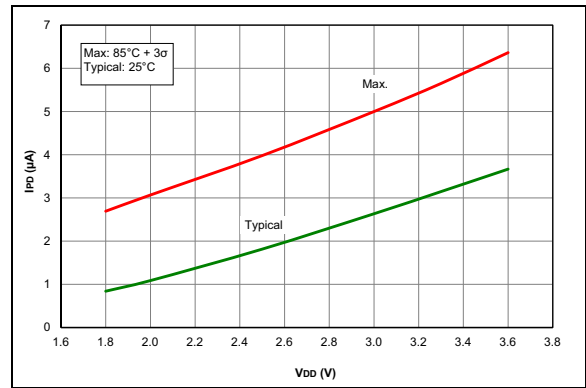


FIGURE 36-35: I_{PD} , Timer1 Oscillator, $F_{OSC} = 32\text{ kHz}$, PIC16LF1615/9 Only.

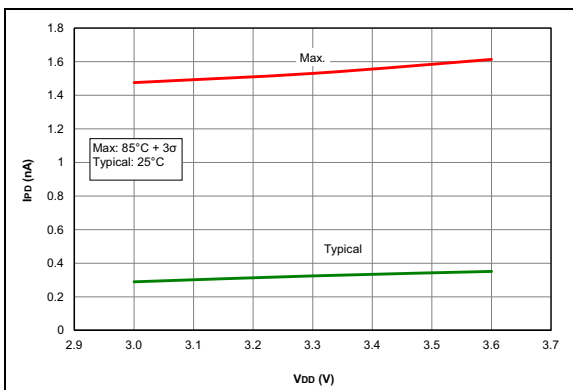


FIGURE 36-33: I_{PD} , LP Brown-Out Reset (LPBOR = 0), PIC16LF1615/9 Only.

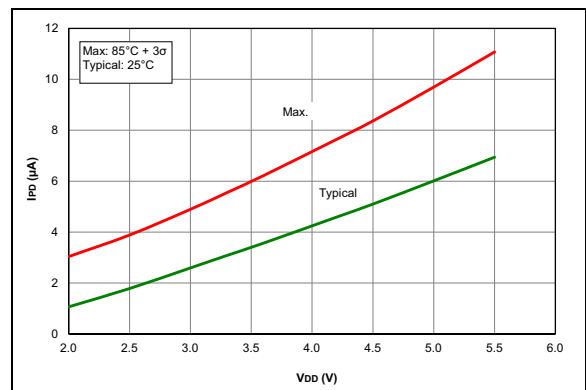


FIGURE 36-36: I_{PD} , Timer1 Oscillator, $F_{OSC} = 32\text{ kHz}$, PIC16F1615/9 Only.

Note: Unless otherwise noted, $V_{IN} = 5V$, $F_{OSC} = 500\text{ kHz}$, $C_{IN} = 0.1\ \mu\text{F}$, $T_A = 25^\circ\text{C}$.

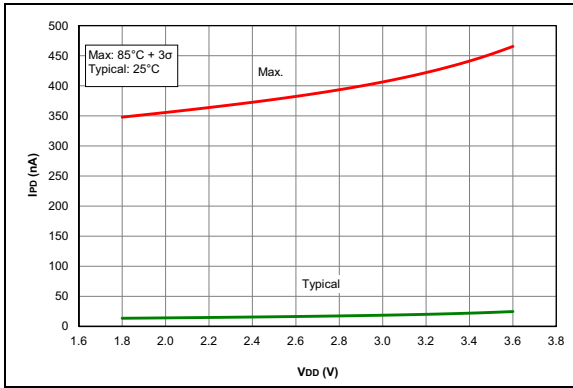


FIGURE 36-37: I_{PD} , ADC Non-Converting, PIC16LF1615/9 Only.

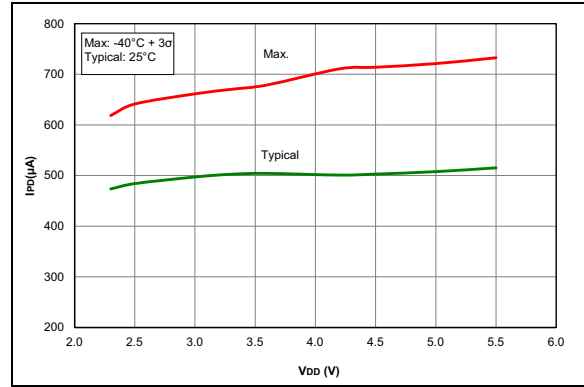


FIGURE 36-40: I_{PD} , Comparator, NP Mode ($CxSP = 1$), PIC16F1615/9 Only.

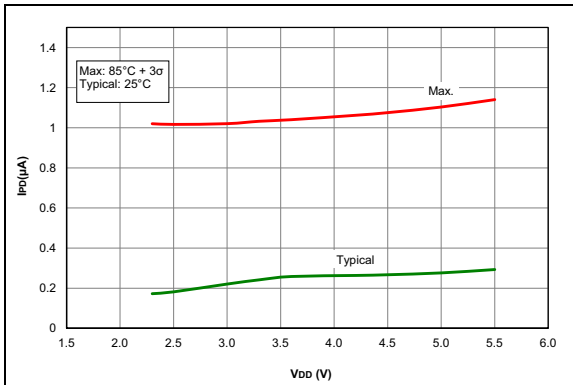


FIGURE 36-38: I_{PD} , ADC Non-Converting, PIC16F1615/9 Only.

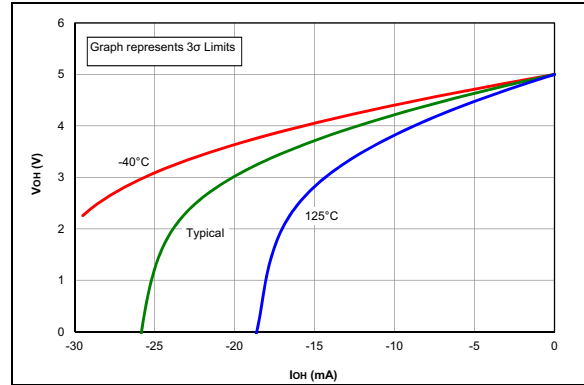


FIGURE 36-41: V_{OH} vs. I_{OH} Over Temperature, $V_{DD} = 5.0V$, PIC16F1615/9 Only.

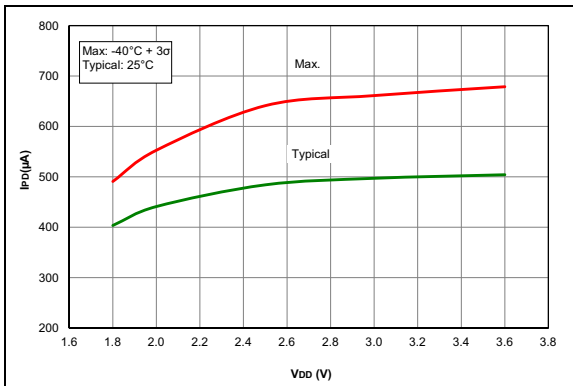


FIGURE 36-39: I_{PD} , Comparator, NP Mode ($CxSP = 1$), PIC16LF1615/9 Only.

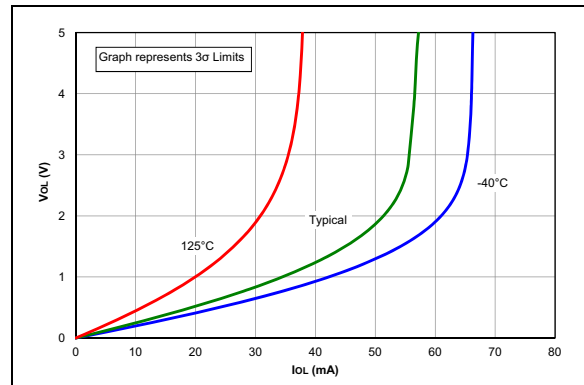


FIGURE 36-42: V_{OL} vs. I_{OL} Over Temperature, $V_{DD} = 5.0V$, PIC16F1615/9 Only.

Note: Unless otherwise noted, $V_{IN} = 5V$, $F_{OSC} = 500\text{ kHz}$, $C_{IN} = 0.1\ \mu\text{F}$, $T_A = 25^\circ\text{C}$.

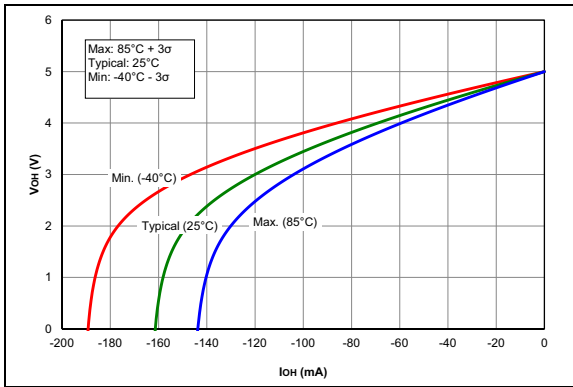


FIGURE 36-43: V_{OH} vs. I_{OH} Over Temperature for High Drive Pins, $V_{DD} = 5.0V$, PIC16F1615/9 Only.

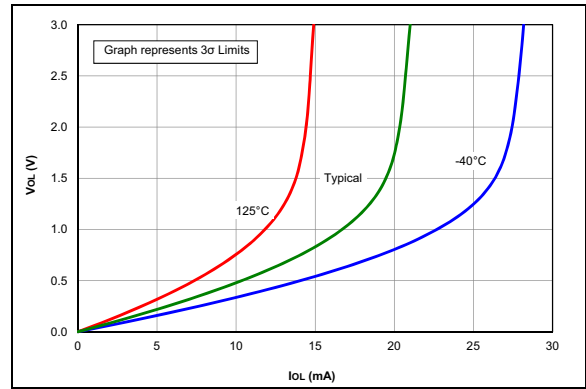


FIGURE 36-46: V_{OL} vs. I_{OL} Over Temperature, $V_{DD} = 3.0V$.

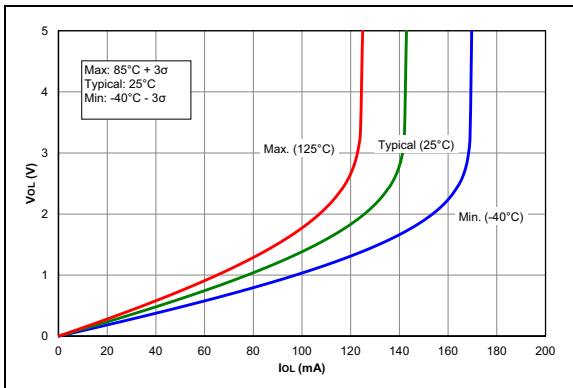


FIGURE 36-44: V_{OL} vs. I_{OL} Over Temperature for High Drive Pins, $V_{DD} = 5.0V$, PIC16F1615/9 Only.

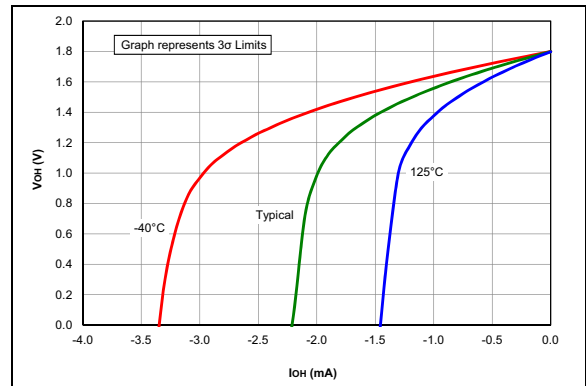


FIGURE 36-47: V_{OH} vs. I_{OH} Over Temperature, $V_{DD} = 1.8V$, PIC16LF1615/9 Only.

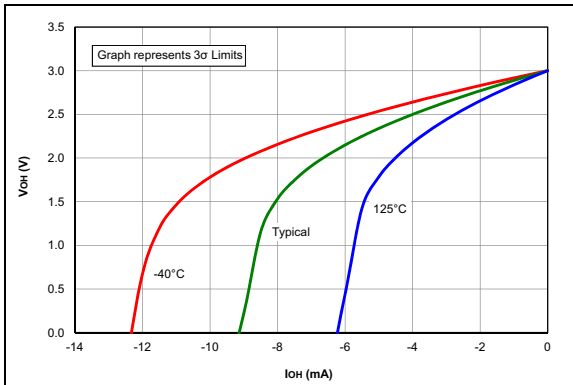


FIGURE 36-45: V_{OH} vs. I_{OH} Over Temperature, $V_{DD} = 3.0V$.

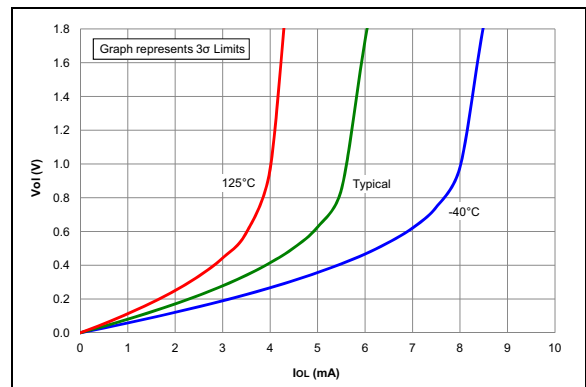


FIGURE 36-48: V_{OL} vs. I_{OL} Over Temperature, $V_{DD} = 1.8V$, PIC16LF1615/9 Only.

Note: Unless otherwise noted, $V_{IN} = 5V$, $F_{OSC} = 500\text{ kHz}$, $C_{IN} = 0.1\ \mu\text{F}$, $T_A = 25^\circ\text{C}$.

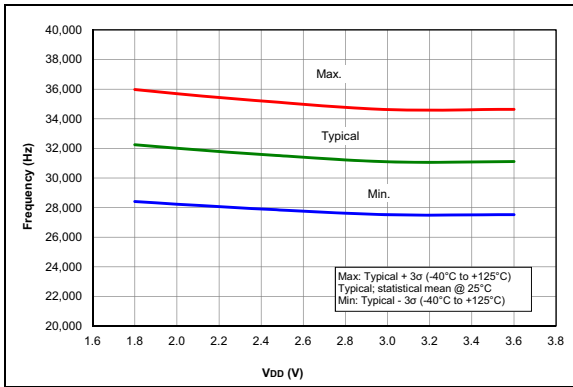


FIGURE 36-49: LFINTOSC Frequency, PIC16LF1615/9 Only.

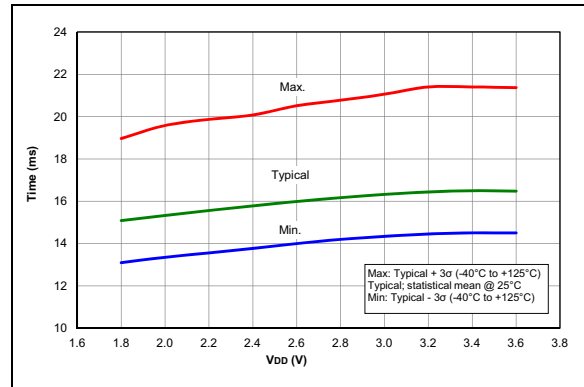


FIGURE 36-52: WDT Time-Out Period, PIC16LF1615/9 Only.

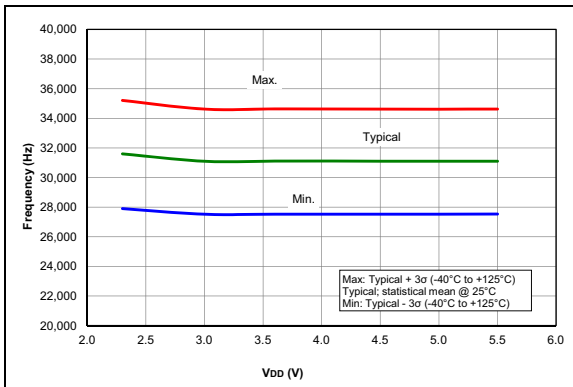


FIGURE 36-50: LFINTOSC Frequency, PIC16F1615/9 Only.

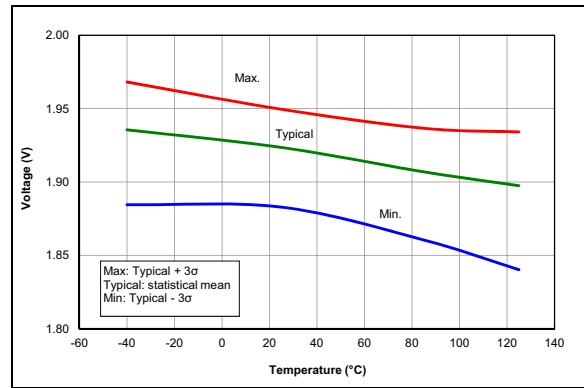


FIGURE 36-53: Brown-Out Reset Voltage, Low Trip Point ($BORV = 1$), PIC16LF1615/9 Only.

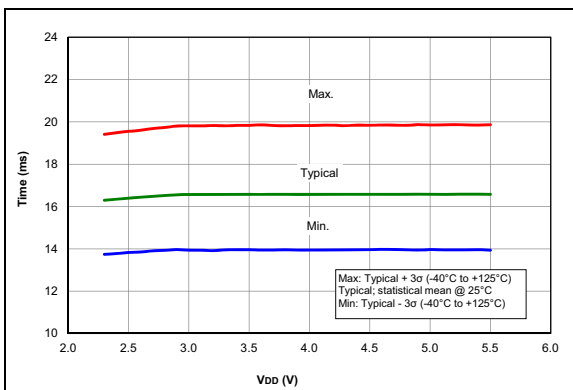


FIGURE 36-51: WDT Time-Out Period, PIC16F1615/9 Only.

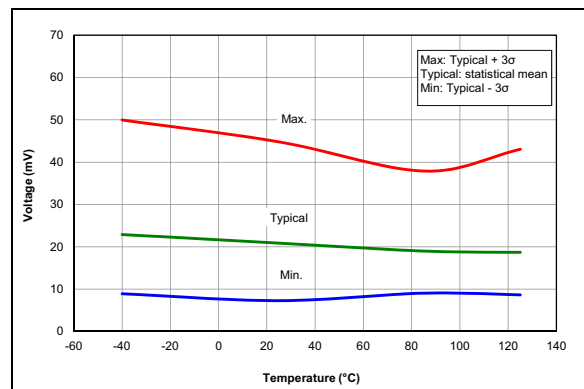


FIGURE 36-54: Brown-Out Reset Hysteresis, Low Trip Point ($BORV = 1$), PIC16LF1615/9 Only.

Note: Unless otherwise noted, $V_{IN} = 5V$, $F_{OSC} = 500\text{ kHz}$, $C_{IN} = 0.1\ \mu\text{F}$, $T_A = 25^\circ\text{C}$.

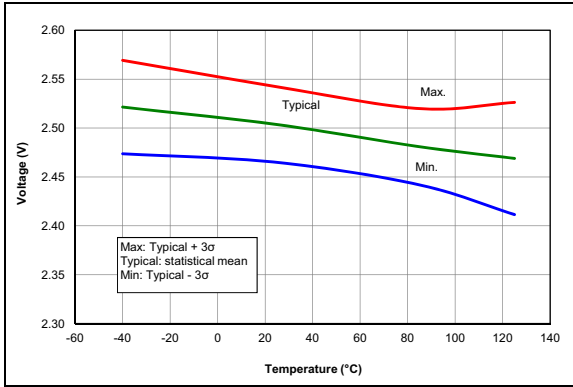


FIGURE 36-55: Brown-Out Reset Voltage, Low Trip Point ($BORV = 1$), PIC16F1615/9 Only.

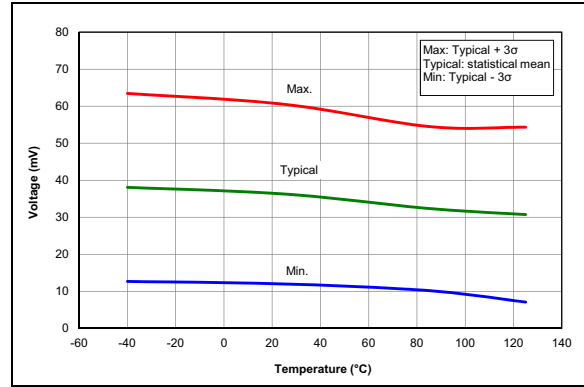


FIGURE 36-58: Brown-Out Reset Hysteresis, High Trip Point ($BORV = 0$).

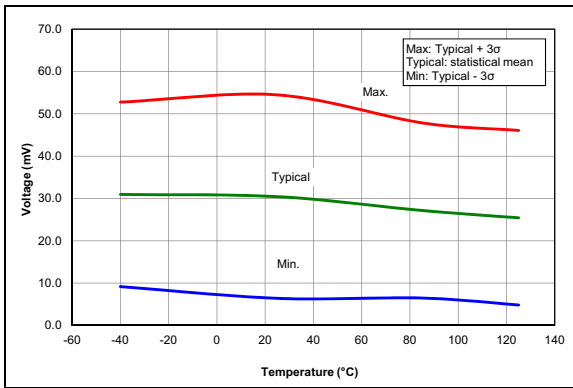


FIGURE 36-56: Brown-Out Reset Hysteresis, Low Trip Point ($BORV = 1$), PIC16F1615/9 Only.

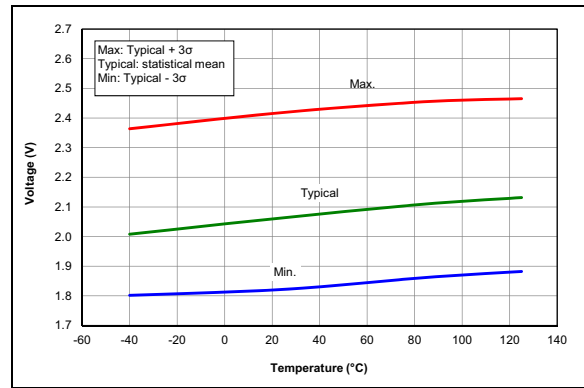


FIGURE 36-59: LPBOR Reset Voltage.

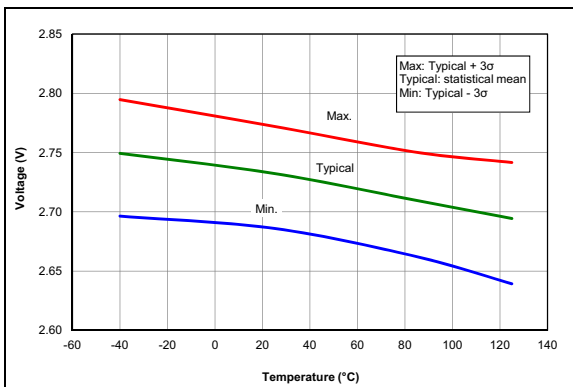


FIGURE 36-57: Brown-Out Reset Voltage, High Trip Point ($BORV = 0$).

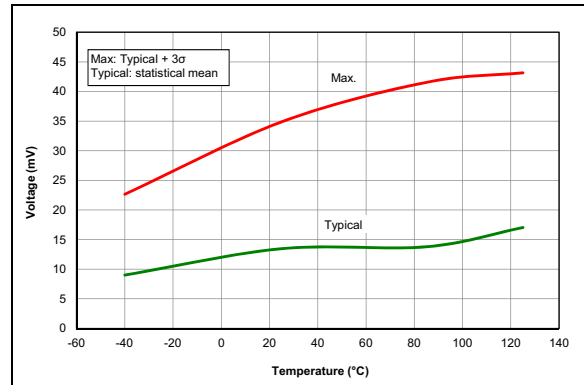


FIGURE 36-60: LPBOR Reset Hysteresis.

Note: Unless otherwise noted, $V_{IN} = 5V$, $F_{OSC} = 500\text{ kHz}$, $C_{IN} = 0.1\ \mu\text{F}$, $T_A = 25^\circ\text{C}$.

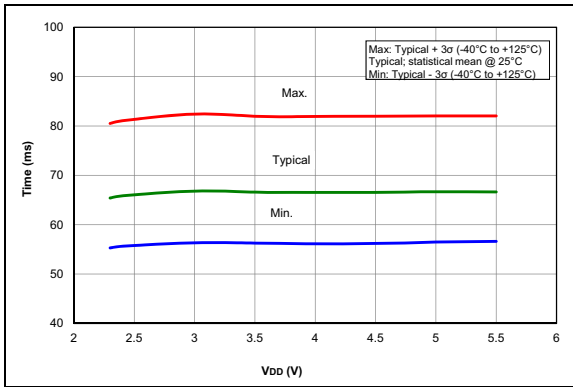


FIGURE 36-61: PWRT Period, PIC16F1615/9 Only.

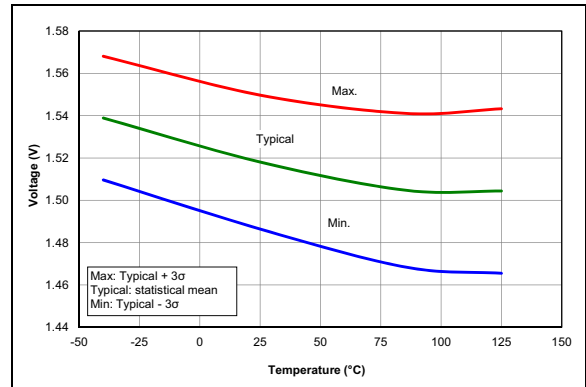


FIGURE 36-64: POR Rearm Voltage, NP Mode ($V_{REGPM1} = 0$), PIC16F1615/9 Only.

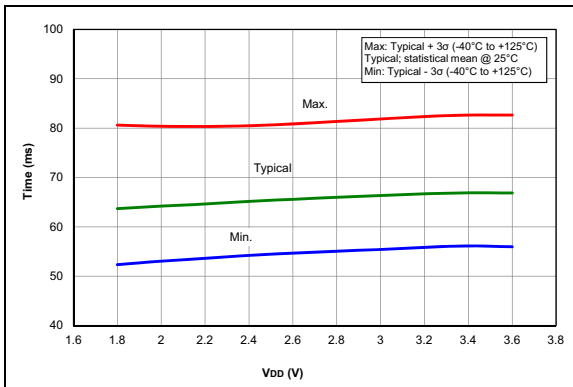


FIGURE 36-62: PWRT Period, PIC16LF1615/9 Only.

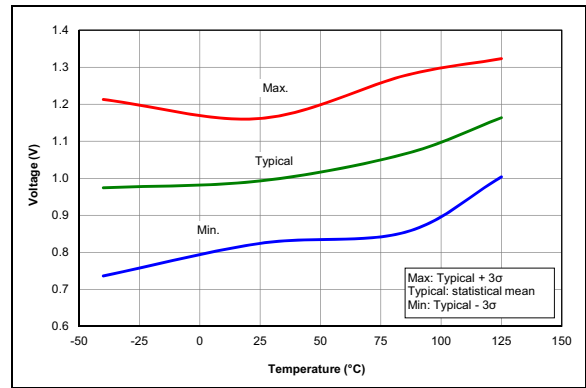


FIGURE 36-65: POR Rearm Voltage, NP Mode, PIC16LF1615/9 Only.

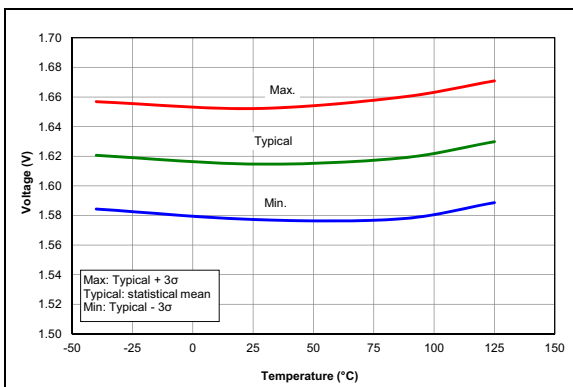


FIGURE 36-63: POR Release Voltage.

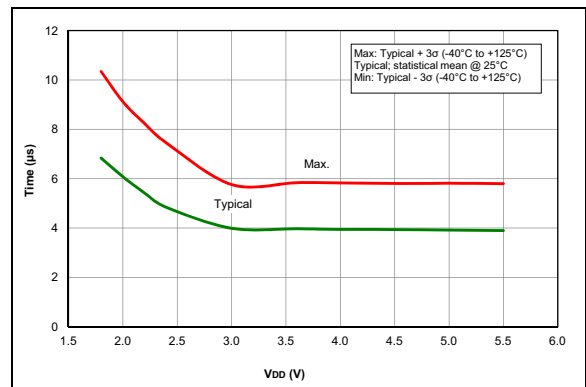


FIGURE 36-66: Wake From Sleep, $V_{REGPM} = 0$.

Note: Unless otherwise noted, $V_{IN} = 5V$, $F_{OSC} = 500\text{ kHz}$, $C_{IN} = 0.1\ \mu\text{F}$, $T_A = 25^\circ\text{C}$.

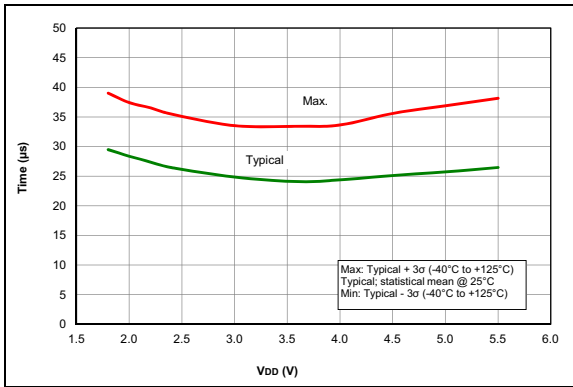


FIGURE 36-67: Wake From Sleep, $V_{REGPM} = 1$.

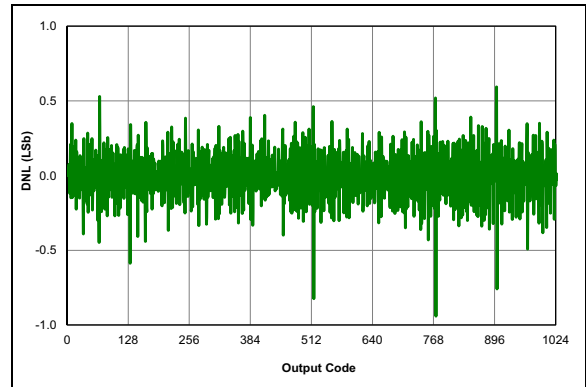


FIGURE 36-70: ADC 10-bit Mode, Single-Ended DNL, $V_{DD} = 3.0V$, $T_{AD} = 4\ \mu\text{S}$, 25°C .

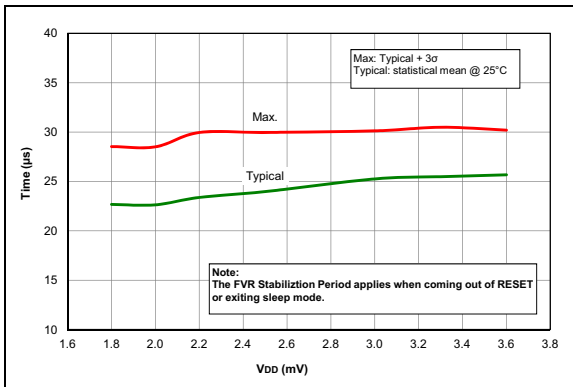


FIGURE 36-68: FVR Stabilization Period, PIC16LF1615/9 Only.

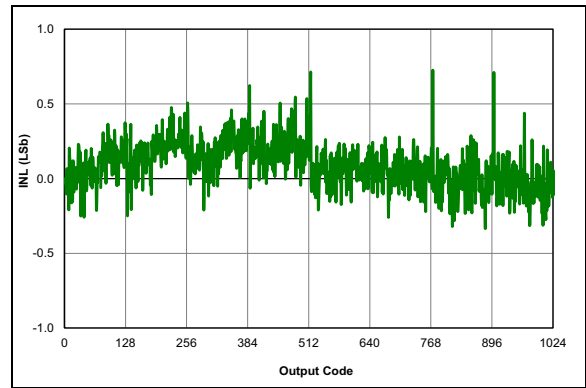


FIGURE 36-71: ADC 10-bit Mode, Single-Ended INL, $V_{DD} = 3.0V$, $T_{AD} = 1\ \mu\text{S}$, 25°C .

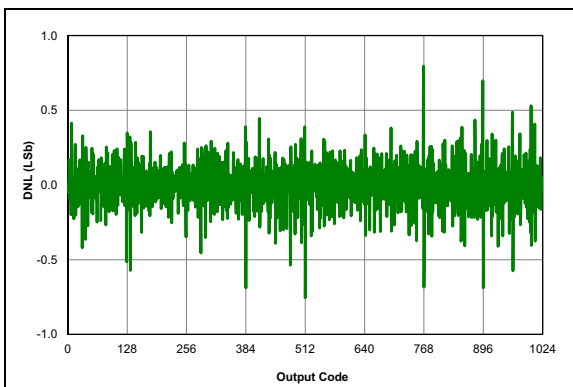


FIGURE 36-69: ADC 10-bit Mode, Single-Ended DNL, $V_{DD} = 3.0V$, $T_{AD} = 1\ \mu\text{S}$, 25°C .

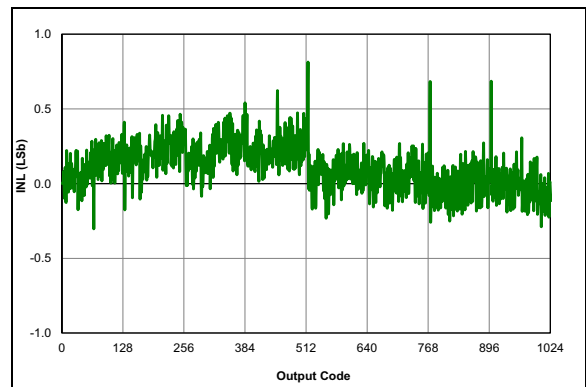


FIGURE 36-72: ADC 10-bit Mode, Single-Ended INL, $V_{DD} = 3.0V$, $T_{AD} = 4\ \mu\text{S}$, 25°C .

Note: Unless otherwise noted, $V_{IN} = 5V$, $F_{OSC} = 500\text{ kHz}$, $C_{IN} = 0.1\ \mu\text{F}$, $T_A = 25^\circ\text{C}$.

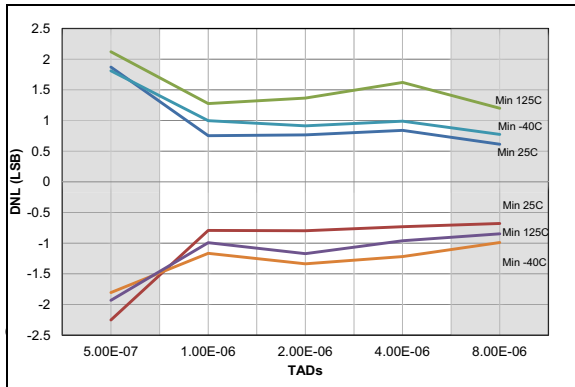


FIGURE 36-73: ADC 10-bit Mode, Single-Ended DNL, $V_{DD} = 3.0V$, $V_{REF} = 3.0V$.

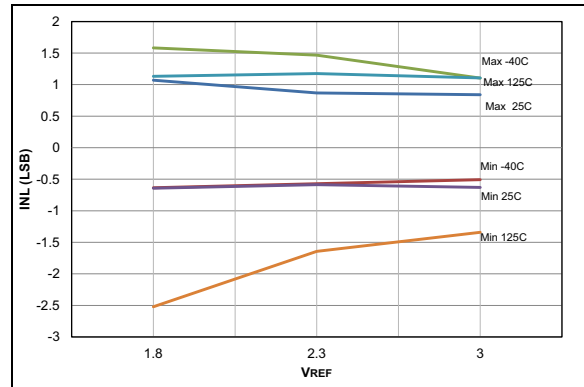


FIGURE 36-76: ADC 10-bit Mode, Single-Ended INL, $V_{DD} = 3.0V$, $T_{AD} = 1\ \mu\text{s}$.

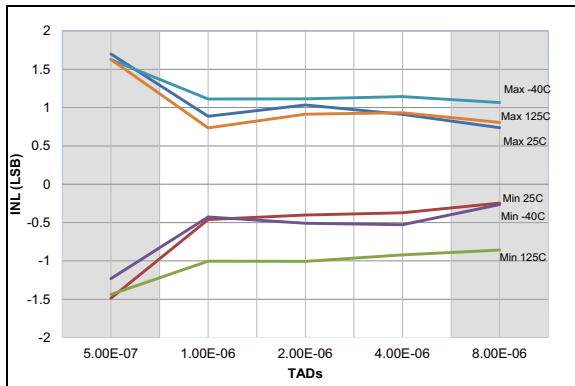


FIGURE 36-74: ADC 10-bit Mode, Single-Ended INL, $V_{DD} = 3.0V$, $V_{REF} = 3.0V$.

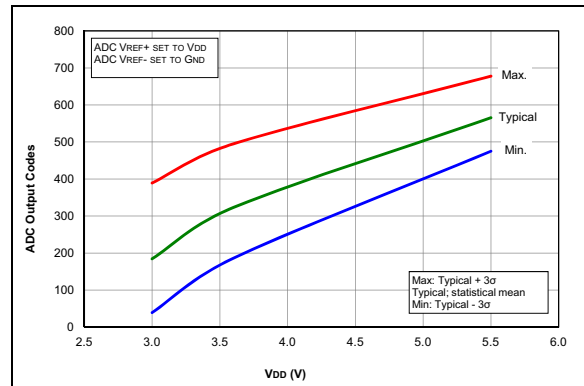


FIGURE 36-77: Temp. Indicator Initial Offset, High Range, Temp. = 20°C , PIC16F1615/9 Only.

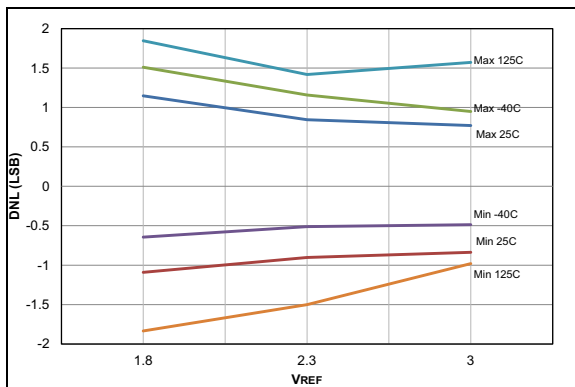


FIGURE 36-75: ADC 10-bit Mode, Single-Ended DNL, $V_{DD} = 3.0V$, $T_{AD} = 1\ \mu\text{s}$.

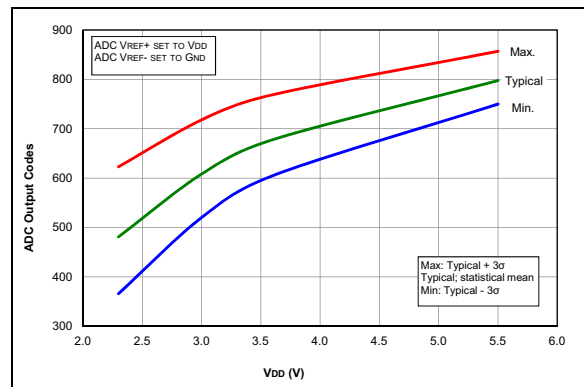


FIGURE 36-78: Temp. Indicator Initial Offset, Low Range, Temp. = 20°C , PIC16F1615/9 Only.

Note: Unless otherwise noted, $V_{IN} = 5V$, $F_{OSC} = 500\text{ kHz}$, $C_{IN} = 0.1\ \mu\text{F}$, $T_A = 25^\circ\text{C}$.

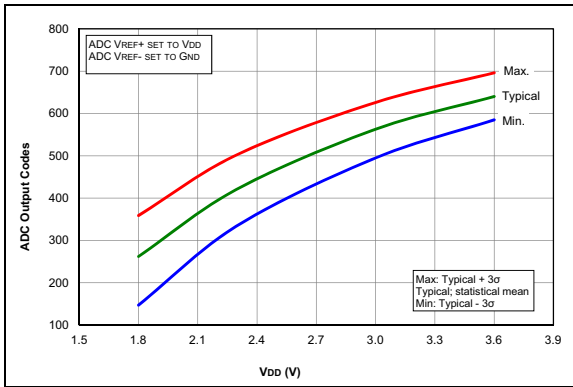


FIGURE 36-79: Temp. Indicator Initial Offset, Low Range, Temp. = 20°C, PIC16LF1615/9 Only.

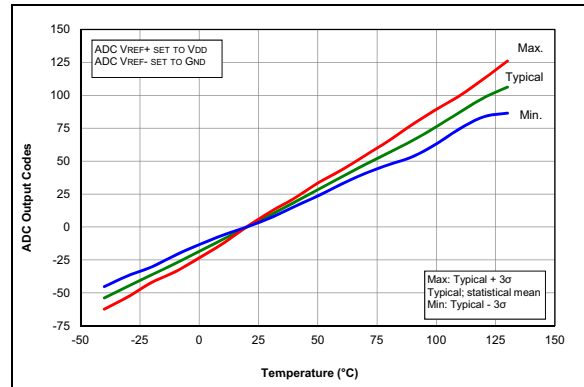


FIGURE 36-82: Temp. Indicator Slope Normalized to 20°C, Low Range, $V_{DD} = 3.0V$, PIC16F1615/9 Only.

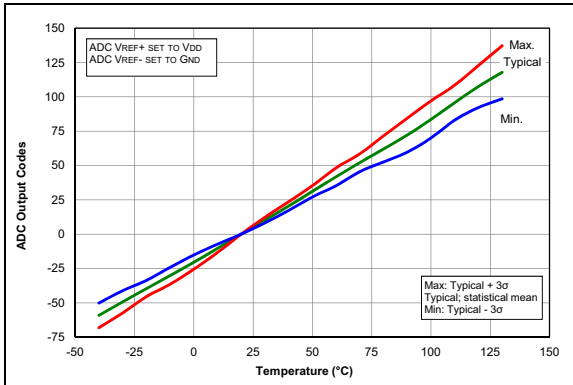


FIGURE 36-80: Temp. Indicator Slope Normalized to 20°C, High Range, $V_{DD} = 5.5V$, PIC16F1615/9 Only.

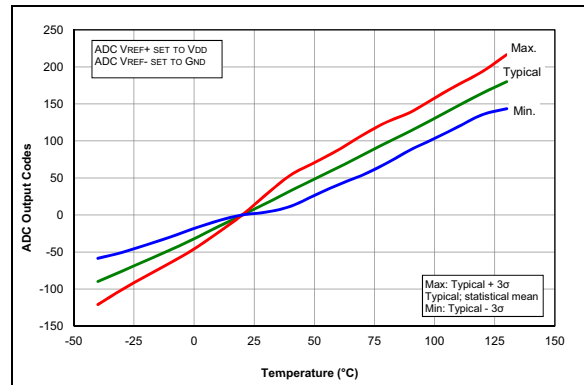


FIGURE 36-83: Temp. Indicator Slope Normalized to 20°C, Low Range, $V_{DD} = 1.8V$, PIC16LF1615/9 Only.

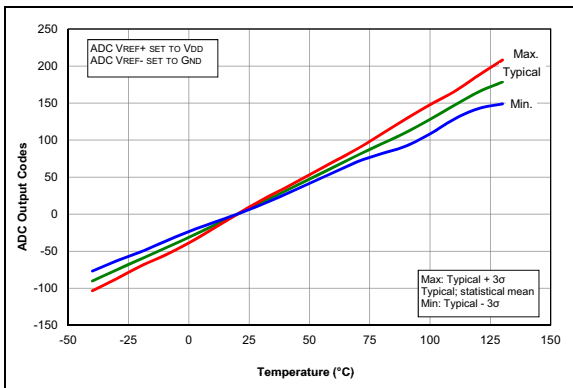


FIGURE 36-81: Temp. Indicator Slope Normalized to 20°C, High Range, $V_{DD} = 3.0V$, PIC16F1615/9 Only.

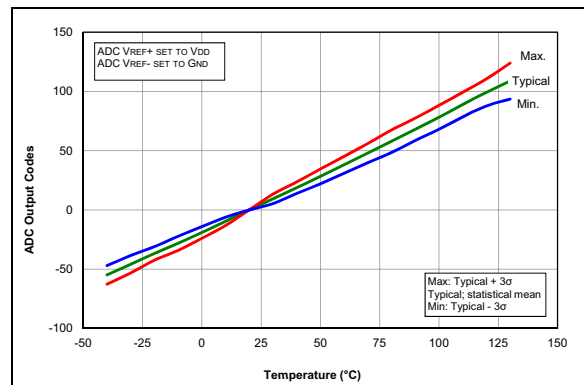


FIGURE 36-84: Temp. Indicator Slope Normalized to 20°C, Low Range, $V_{DD} = 3.0V$, PIC16LF1615/9 Only.

Note: Unless otherwise noted, $V_{IN} = 5V$, $F_{OSC} = 500\text{ kHz}$, $C_{IN} = 0.1\ \mu\text{F}$, $T_A = 25^\circ\text{C}$.

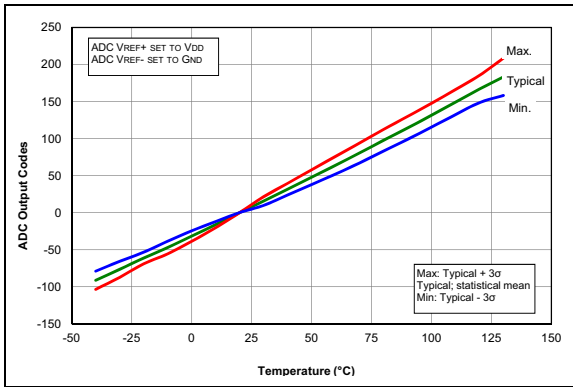


FIGURE 36-85: Temp. Indicator Slope Normalized to 20°C , High Range, $V_{DD} = 3.6V$, PIC16LF1615/9 Only.

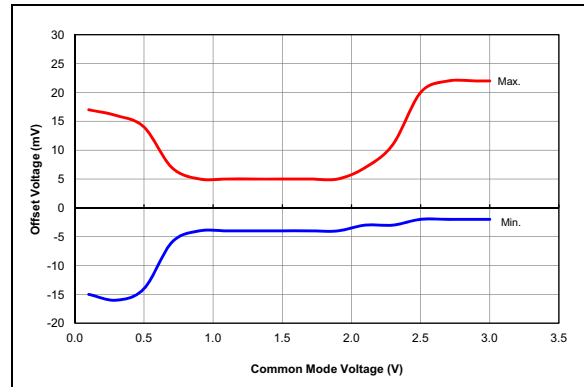


FIGURE 36-88: Comparator Offset, NP Mode ($CxSP = 1$), $V_{DD} = 3.0V$, Typical Measured Values From -40°C to 125°C .

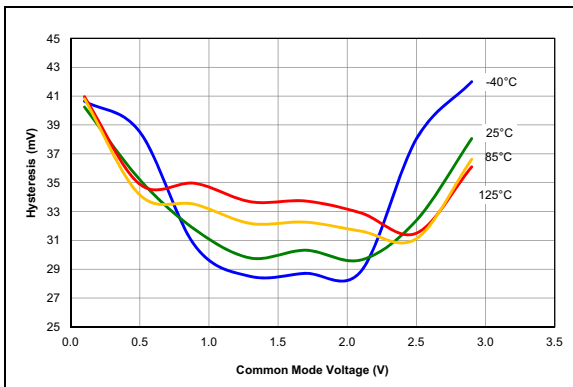


FIGURE 36-86: Comparator Hysteresis, NP Mode ($CxSP = 1$), $V_{DD} = 3.0V$, Typical Measured Values.

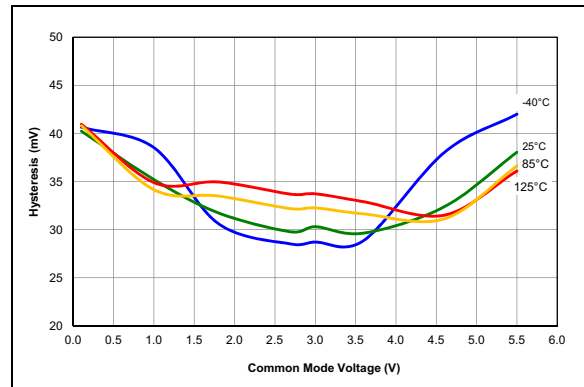


FIGURE 36-89: Comparator Hysteresis, NP Mode ($CxSP = 1$), $V_{DD} = 5.5V$, Typical Measured Values, PIC16F1615/9 Only.

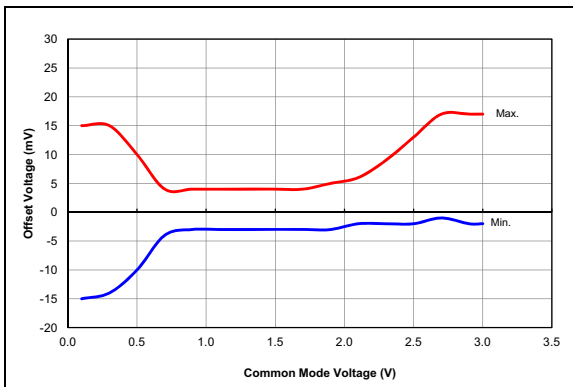


FIGURE 36-87: Comparator Offset, NP Mode ($CxSP = 1$), $V_{DD} = 3.0V$, Typical Measured Values at 25°C .

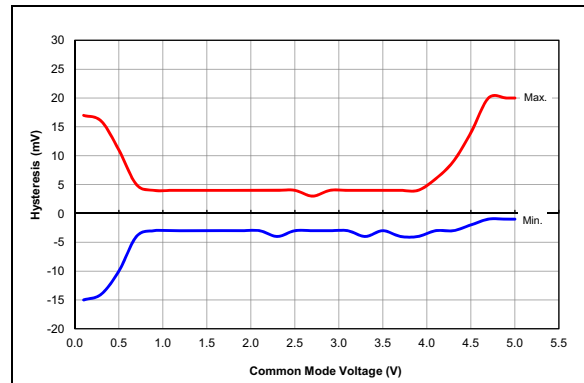


FIGURE 36-90: Comparator Offset, NP Mode ($CxSP = 1$), $V_{DD} = 5.0V$, Typical Measured Values at 25°C , PIC16F1615/9 Only.

Note: Unless otherwise noted, $V_{IN} = 5V$, $F_{OSC} = 500\text{ kHz}$, $C_{IN} = 0.1\ \mu\text{F}$, $T_A = 25^\circ\text{C}$.

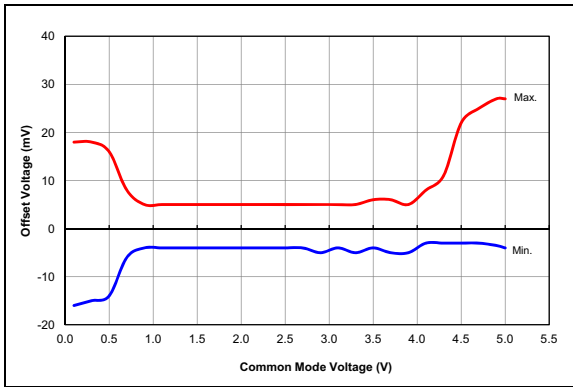


FIGURE 36-91: Comparator Offset, NP Mode ($CxSP = 1$), $V_{DD} = 5.5V$, Typical Measured Values From -40°C to 125°C , PIC16F1615/9 Only.

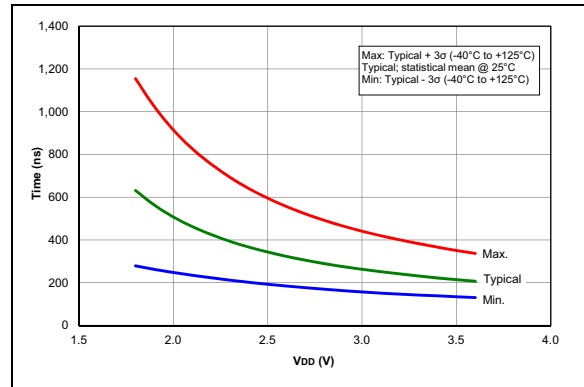


FIGURE 36-94: Comparator Output Filter Delay Time Over Temp., NP Mode ($CxSP = 1$), Typical Measured Values, PIC16LF1615/9 Only.

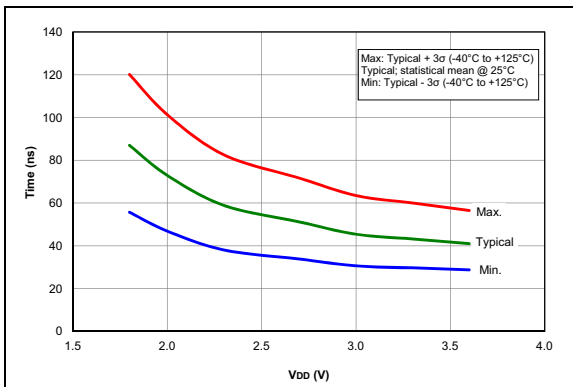


FIGURE 36-92: Comparator Response Time Over Voltage, NP Mode ($CxSP = 1$), Typical Measured Values, PIC16LF1615/9 Only.

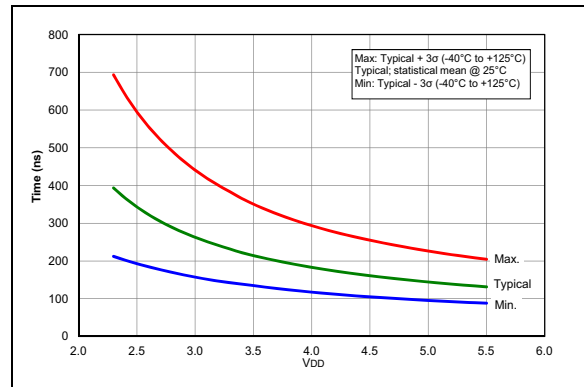


FIGURE 36-95: Comparator Output Filter Delay Time Over Temp., NP Mode ($CxSP = 1$), Typical Measured Values, PIC16F1615/9 Only.

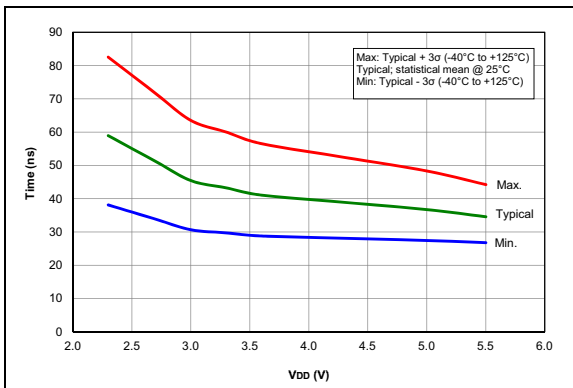


FIGURE 36-93: Comparator Response Time Over Voltage, NP Mode ($CxSP = 1$), Typical Measured Values, PIC16F1615/9 Only.

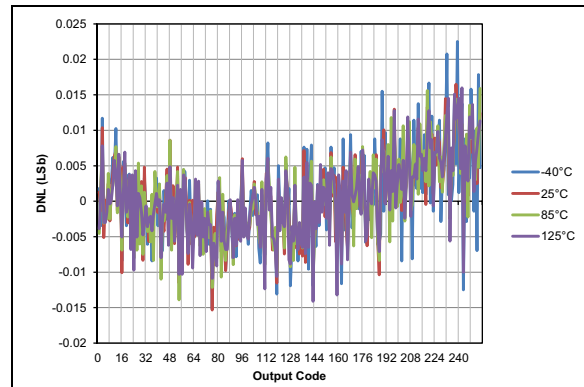


FIGURE 36-96: Typical DAC DNL Error, $V_{DD} = 3.0V$, $V_{REF} = \text{External } 3V$.

Note: Unless otherwise noted, $V_{IN} = 5V$, $F_{OSC} = 500\text{ kHz}$, $C_{IN} = 0.1\ \mu\text{F}$, $T_A = 25^\circ\text{C}$.

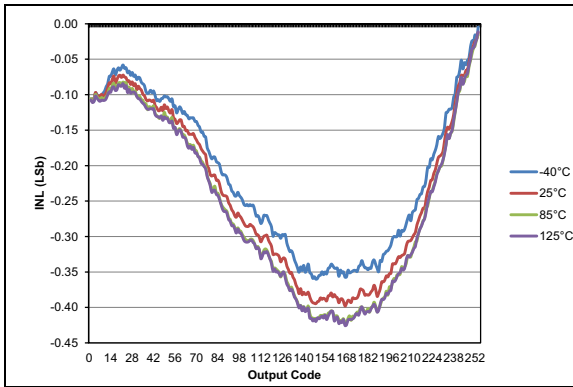


FIGURE 36-97: Typical DAC INL Error, $V_{DD} = 3.0V$, $V_{REF} = \text{External } 3V$.

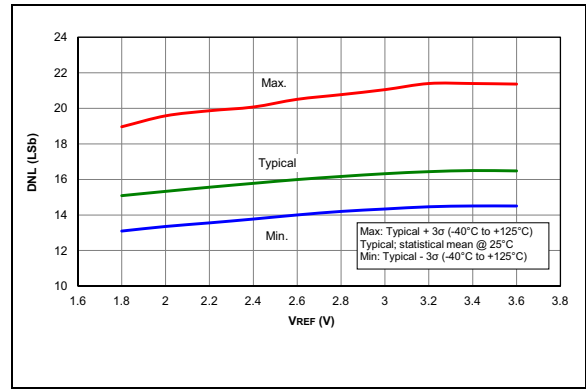


FIGURE 36-100: DAC INL Error, $V_{DD} = 3.0V$.

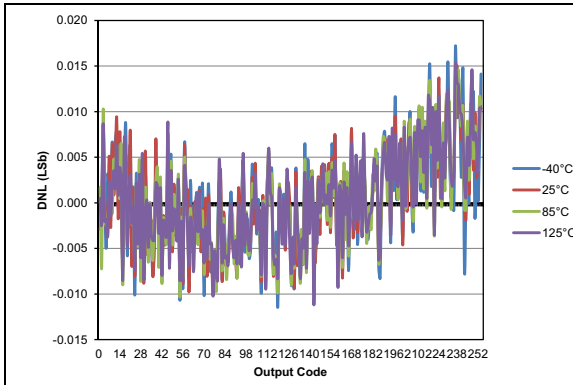


FIGURE 36-98: Typical DAC INL Error, $V_{DD} = 5.0V$, $V_{REF} = \text{External } 5V$, PIC16F1615/9 Only.

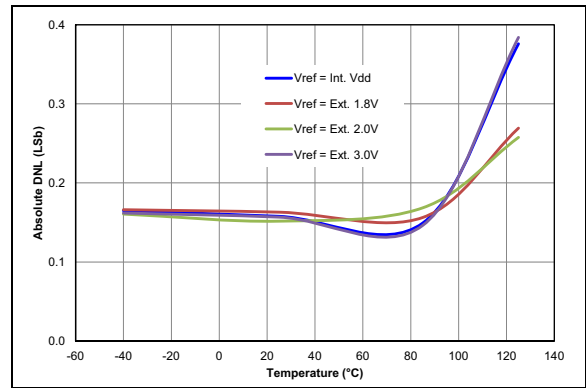


FIGURE 36-101: Absolute Value of DAC DNL Error, $V_{DD} = 3.0V$, $V_{REF} = V_{DD}$.

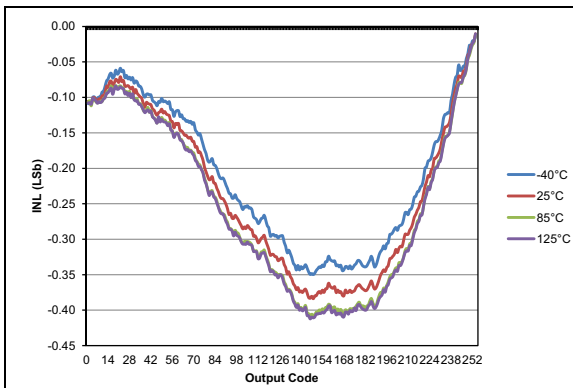


FIGURE 36-99: Typical DAC INL Error, $V_{DD} = 5.0V$, $V_{REF} = \text{External } 5V$, PIC16F1615/9 Only.

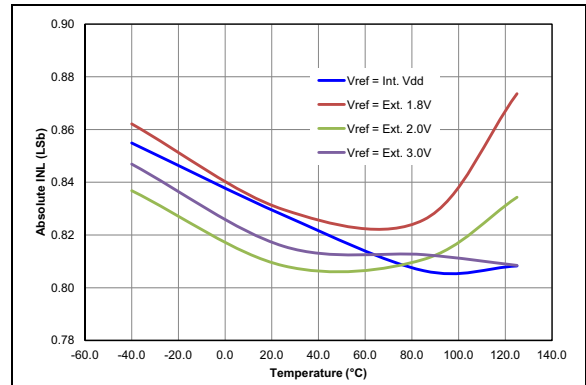


FIGURE 36-102: Absolute Value of DAC INL Error, $V_{DD} = 3.0V$, $V_{REF} = V_{DD}$.

Note: Unless otherwise noted, $V_{IN} = 5V$, $F_{OSC} = 500\text{ kHz}$, $C_{IN} = 0.1\ \mu\text{F}$, $T_A = 25^\circ\text{C}$.

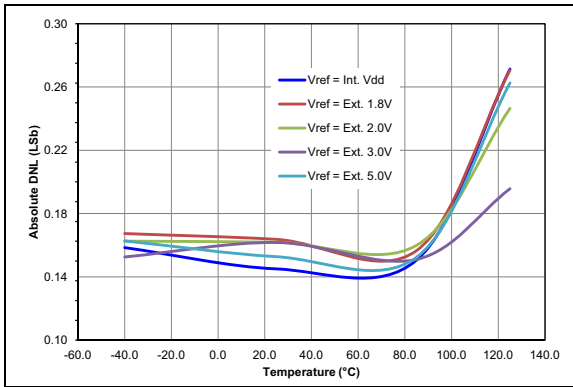


FIGURE 36-103: Absolute Value of DAC DNL Error, $V_{DD} = 5.0V$, $V_{REF} = V_{DD}$, PIC16F1615/9 Only.

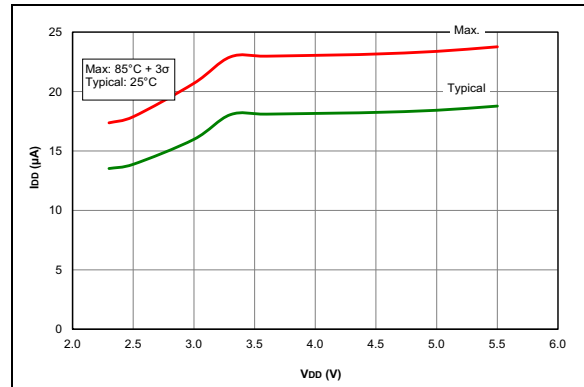


FIGURE 36-106: ZCD Response Time over Voltage Typical Measured Values.

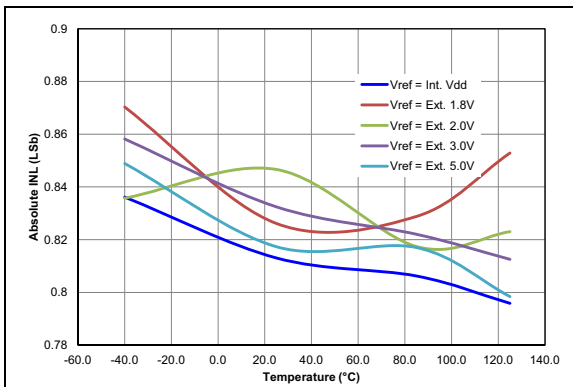


FIGURE 36-104: Absolute Value of DAC INL Error, $V_{DD} = 5.0V$, $V_{REF} = V_{DD}$, PIC16F1615/9 Only.

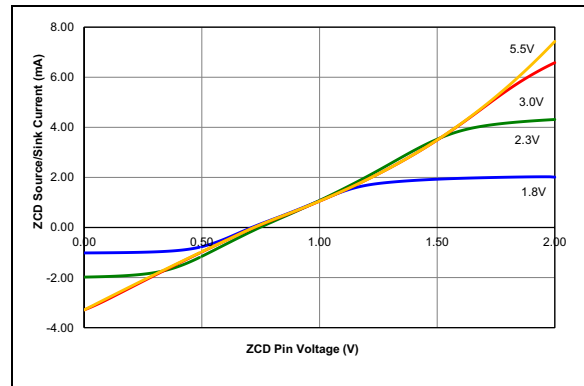


FIGURE 36-107: ZCD Pin Current over ZCD Pin Voltage, Typical Measured Values from -40°C to 125°C .

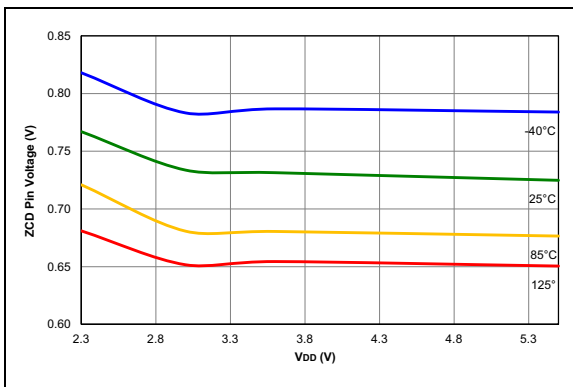


FIGURE 36-105: ZCD Pin Voltage, Typical Measured Values.

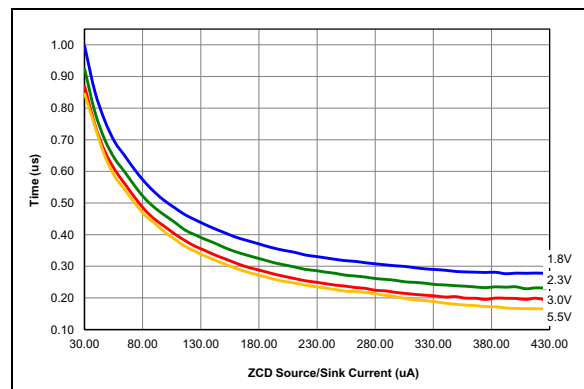


FIGURE 36-108: ZCD Pin Response Time over Current, Typical Measured Values from -40°C to 125°C .

37.0 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

37.1 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

37.2 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

37.3 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

37.4 MPLINK Object Linker/ MPLIB Object Librarian

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

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

The object linker/library features include:

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

37.5 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

37.6 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.

37.7 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 upgradeable 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.

37.8 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.

37.9 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™).

37.10 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.

37.11 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.

37.12 Third-Party Development Tools

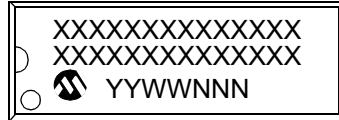
Microchip also offers a great collection of tools from third-party vendors. These tools are carefully selected to offer good value and unique functionality.

- 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®

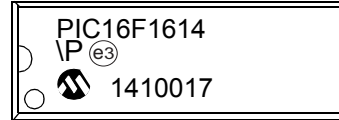
38.0 PACKAGING INFORMATION

38.1 Package Marking Information

14-Lead PDIP



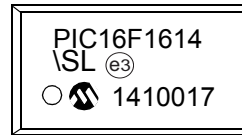
Example



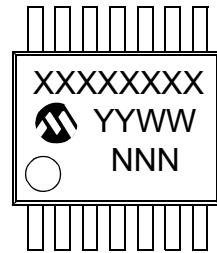
14-Lead SOIC (.150")



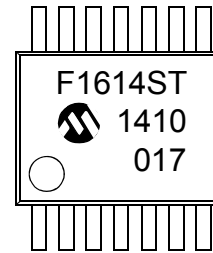
Example



14-Lead TSSOP



Example



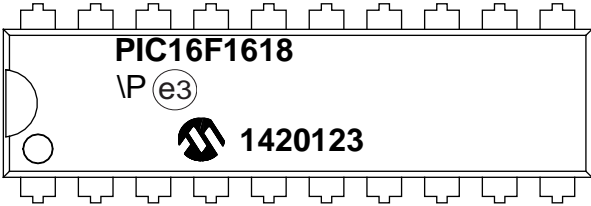
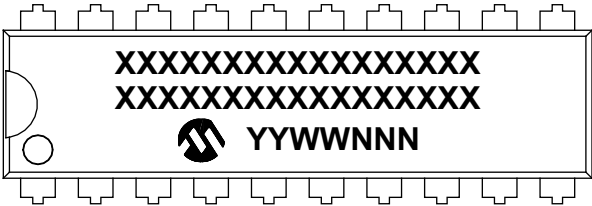
| | | |
|----------------|--------|--|
| Legend: | XX...X | Customer-specific information |
| | Y | Year code (last digit of calendar year) |
| | YY | Year code (last 2 digits of calendar year) |
| | WW | Week code (week of January 1 is week '01') |
| | NNN | Alphanumeric traceability code |
| | (e3) | 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.

38.1 Package Marking Information (Continued)

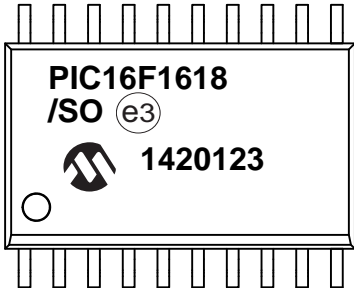
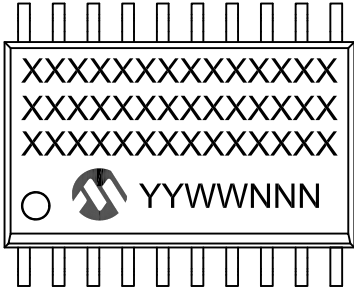
20-Lead PDIP (300 mil)

Example



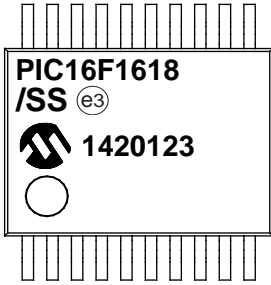
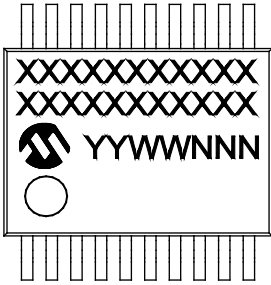
20-Lead SOIC (7.50 mm)

Example



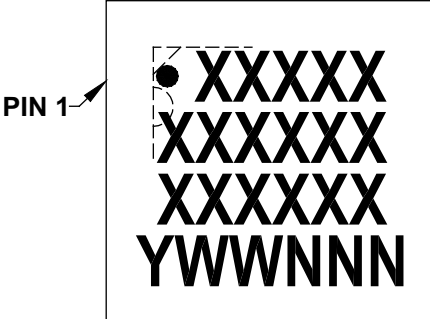
20-Lead SSOP (5.30 mm)

Example

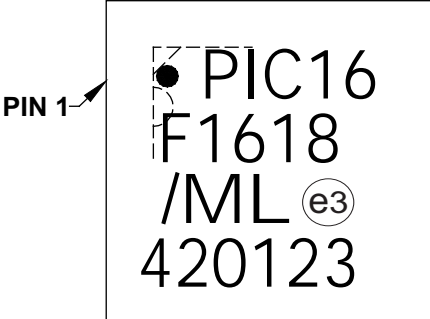


38.1 Package Marking Information (Continued)

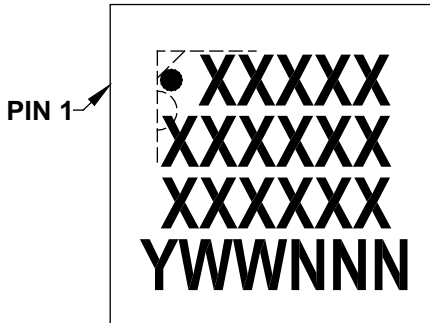
16-Lead QFN (4x4x0.5 mm)



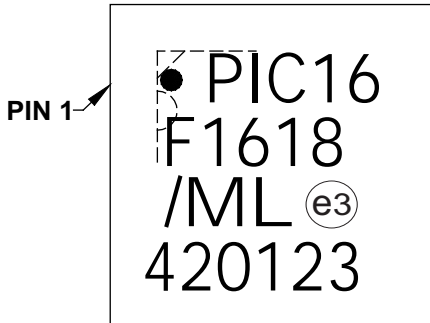
Example



20-Lead QFN/UQFN (4x4x0.5 mm)



Example

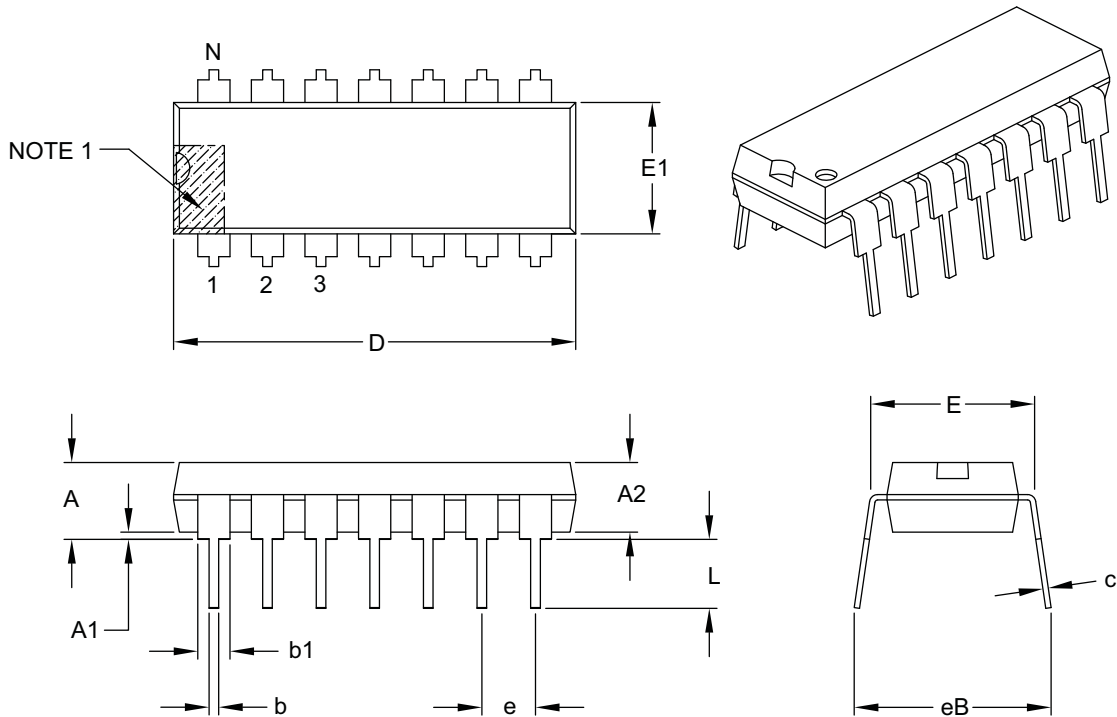


38.2 Package Details

The following sections give the technical details of the packages.

14-Lead Plastic Dual In-Line (P) – 300 mil Body [PDIP]

Note: For the most current package drawings, please see the Microchip Packaging Specification located at <http://www.microchip.com/packaging>



| Dimension Limits | Units | INCHES | | |
|----------------------------|-------|----------|------|------|
| | | MIN | NOM | MAX |
| Number of Pins | N | 14 | | |
| Pitch | e | .100 BSC | | |
| Top to Seating Plane | A | – | – | .210 |
| Molded Package Thickness | A2 | .115 | .130 | .195 |
| Base to Seating Plane | A1 | .015 | – | – |
| Shoulder to Shoulder Width | E | .290 | .310 | .325 |
| Molded Package Width | E1 | .240 | .250 | .280 |
| Overall Length | D | .735 | .750 | .775 |
| Tip to Seating Plane | L | .115 | .130 | .150 |
| Lead Thickness | c | .008 | .010 | .015 |
| Upper Lead Width | b1 | .045 | .060 | .070 |
| Lower Lead Width | b | .014 | .018 | .022 |
| Overall Row Spacing § | eB | – | – | .430 |

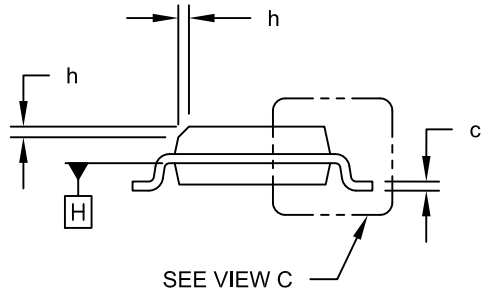
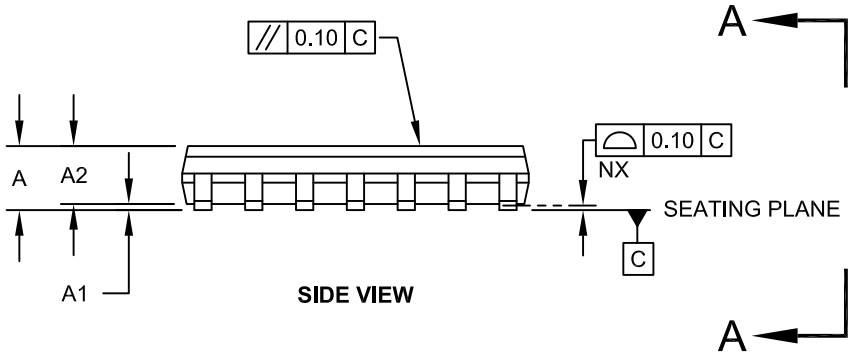
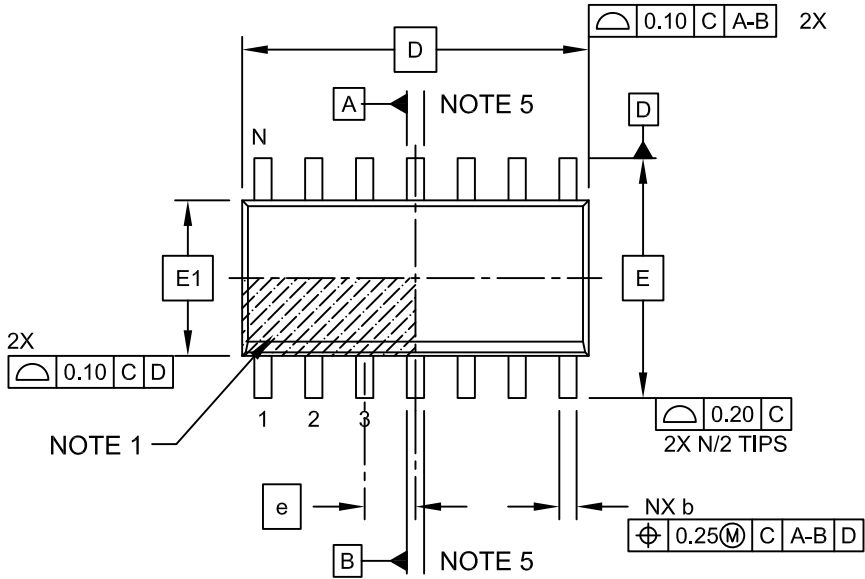
Notes:

1. Pin 1 visual index feature may vary, but must be located with the hatched area.
2. § Significant Characteristic.
3. Dimensions D and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed .010" per side.
4. Dimensioning and tolerancing per ASME Y14.5M.
BSC: Basic Dimension. Theoretically exact value shown without tolerances.

Microchip Technology Drawing C04-005B

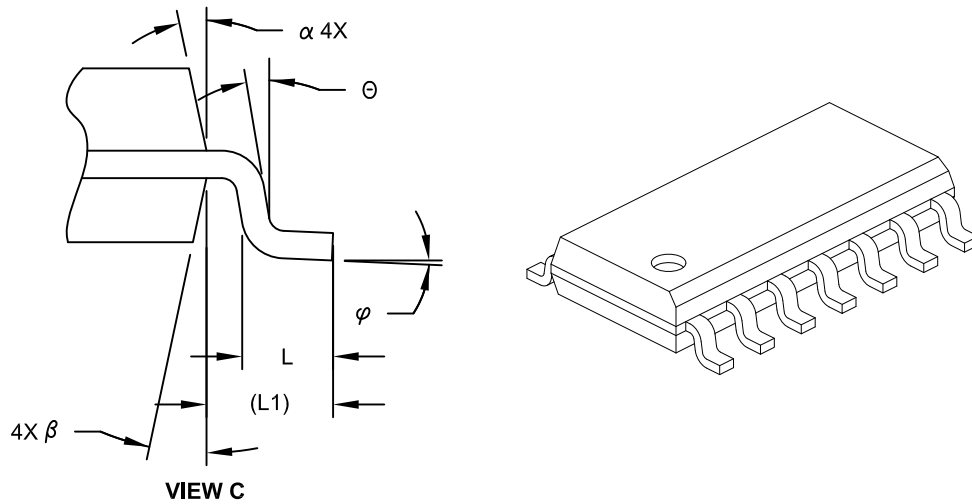
14-Lead Plastic Small Outline (SL) - Narrow, 3.90 mm Body [SOIC]

Note: For the most current package drawings, please see the Microchip Packaging Specification located at <http://www.microchip.com/packaging>



14-Lead Plastic Small Outline (SL) - Narrow, 3.90 mm Body [SOIC]

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 Pins | N | 14 | | |
| Pitch | e | 1.27 BSC | | |
| Overall Height | A | - | - | 1.75 |
| Molded Package Thickness | A2 | 1.25 | - | - |
| Standoff § | A1 | 0.10 | - | 0.25 |
| Overall Width | E | 6.00 BSC | | |
| Molded Package Width | E1 | 3.90 BSC | | |
| Overall Length | D | 8.65 BSC | | |
| Chamfer (Optional) | h | 0.25 | - | 0.50 |
| Foot Length | L | 0.40 | - | 1.27 |
| Footprint | L1 | 1.04 REF | | |
| Lead Angle | Θ | 0° | - | - |
| Foot Angle | φ | 0° | - | 8° |
| Lead Thickness | c | 0.10 | - | 0.25 |
| Lead Width | b | 0.31 | - | 0.51 |
| Mold Draft Angle Top | α | 5° | - | 15° |
| Mold Draft Angle Bottom | β | 5° | - | 15° |

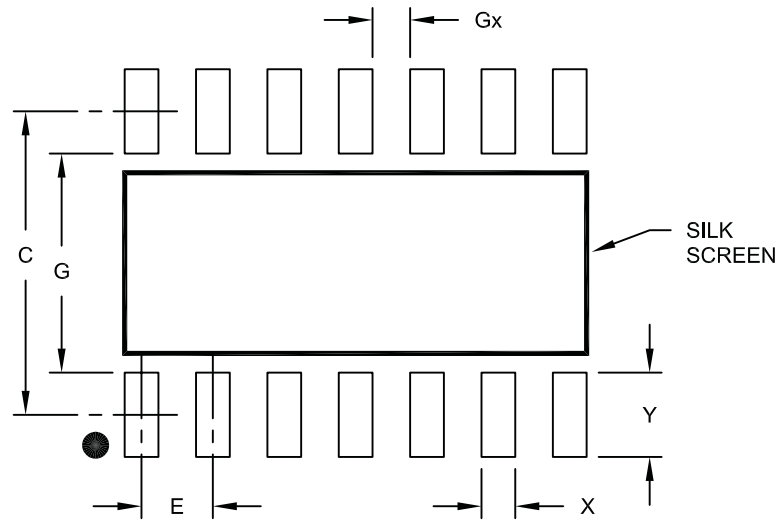
Notes:

- Pin 1 visual index feature may vary, but must be located within the hatched area.
- § Significant Characteristic
- Dimension D does not include mold flash, protrusions or gate burrs, which shall not exceed 0.15 mm per end. Dimension E1 does not include interlead flash or protrusion, which shall not exceed 0.25 mm 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.
- Datums A & B to be determined at Datum H.

Microchip Technology Drawing No. C04-065C Sheet 2 of 2

14-Lead Plastic Small Outline (SL) - Narrow, 3.90 mm Body [SOIC]

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 | 1.27 BSC | | |
| Contact Pad Spacing | C | | 5.40 | |
| Contact Pad Width | X | | | 0.60 |
| Contact Pad Length | Y | | | 1.50 |
| Distance Between Pads | Gx | 0.67 | | |
| Distance Between Pads | G | 3.90 | | |

Notes:

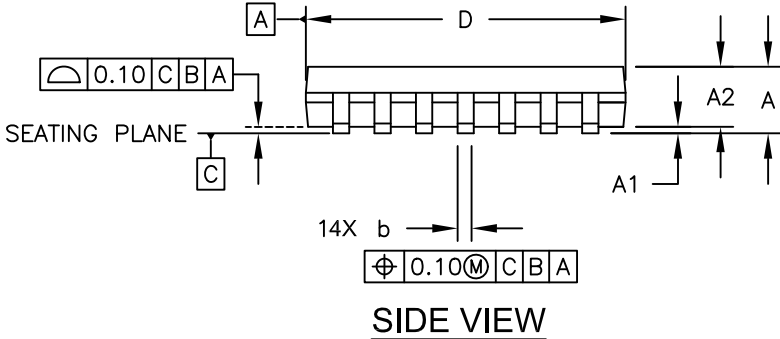
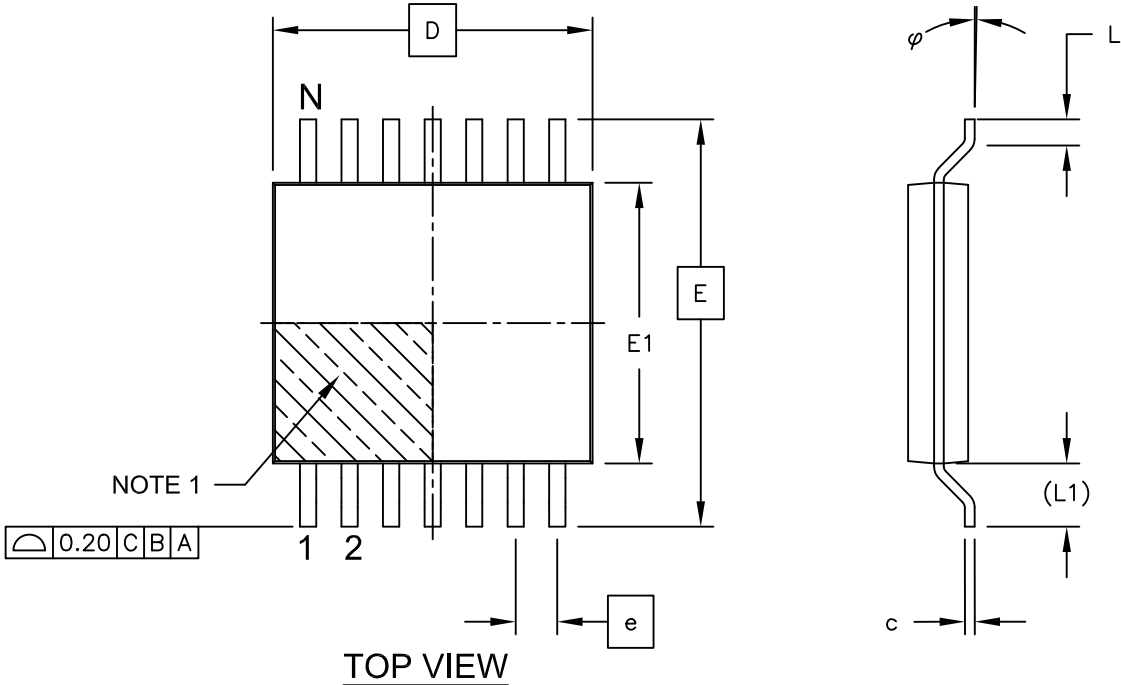
1. Dimensioning and tolerancing per ASME Y14.5M

BSC: Basic Dimension. Theoretically exact value shown without tolerances.

Microchip Technology Drawing No. C04-2065A

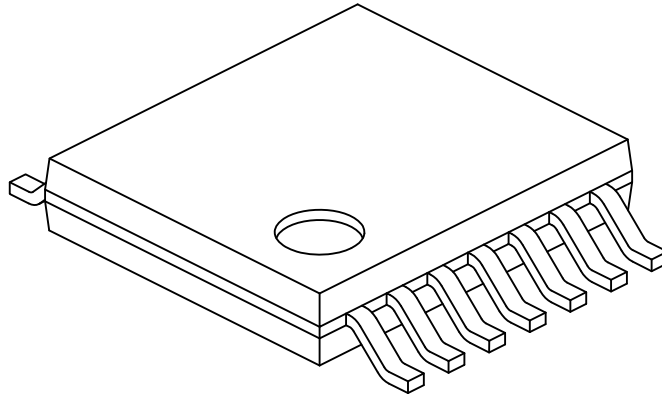
14-Lead Plastic Thin Shrink Small Outline (ST) - 4.4 mm Body [TSSOP]

Note: For the most current package drawings, please see the Microchip Packaging Specification located at <http://www.microchip.com/packaging>



14-Lead Plastic Thin Shrink Small Outline (ST) - 4.4 mm Body [TSSOP]

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 Pins | N | 14 | | |
| Pitch | e | 0.65 BSC | | |
| Overall Height | A | - | - | 1.20 |
| Molded Package Thickness | A2 | 0.80 | 1.00 | 1.05 |
| Standoff | A1 | 0.05 | - | 0.15 |
| Overall Width | E | 6.40 BSC | | |
| Molded Package Width | E1 | 4.30 | 4.40 | 4.50 |
| Molded Package Length | D | 4.90 | 5.00 | 5.10 |
| Foot Length | L | 0.45 | 0.60 | 0.75 |
| Footprint | (L1) | 1.00 REF | | |
| Foot Angle | φ | 0° | - | 8° |
| Lead Thickness | c | 0.09 | - | 0.20 |
| Lead Width | b | 0.19 | - | 0.30 |

Notes:

1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. Dimensions D and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed 0.15mm per side.
3. Dimensioning and tolerancing per ASME Y14.5M

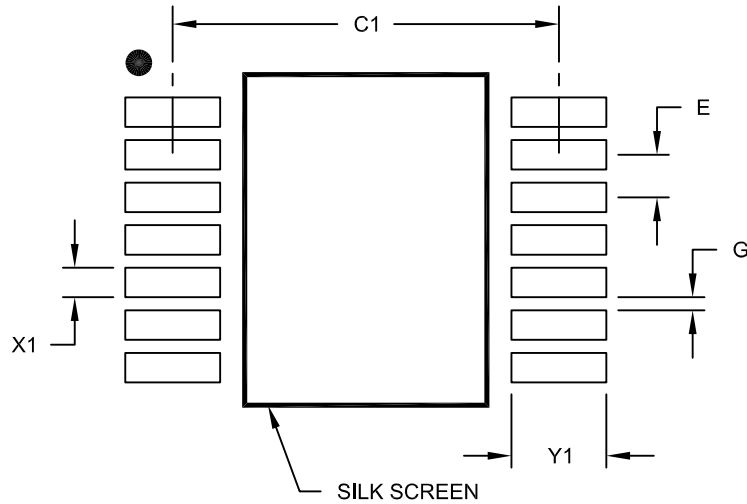
BSC: Basic Dimension. Theoretically exact value shown without tolerances.

REF: Reference Dimension, usually without tolerance, for information purposes only.

Microchip Technology Drawing No. C04-087C Sheet 2 of 2

14-Lead Plastic Thin Shrink Small Outline (ST) - 4.4 mm Body [TSSOP]

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 | 0.65 BSC | | |
| Contact Pad Spacing | C1 | | 5.90 | |
| Contact Pad Width (X14) | X1 | | | 0.45 |
| Contact Pad Length (X14) | Y1 | | | 1.45 |
| Distance Between Pads | G | 0.20 | | |

Notes:

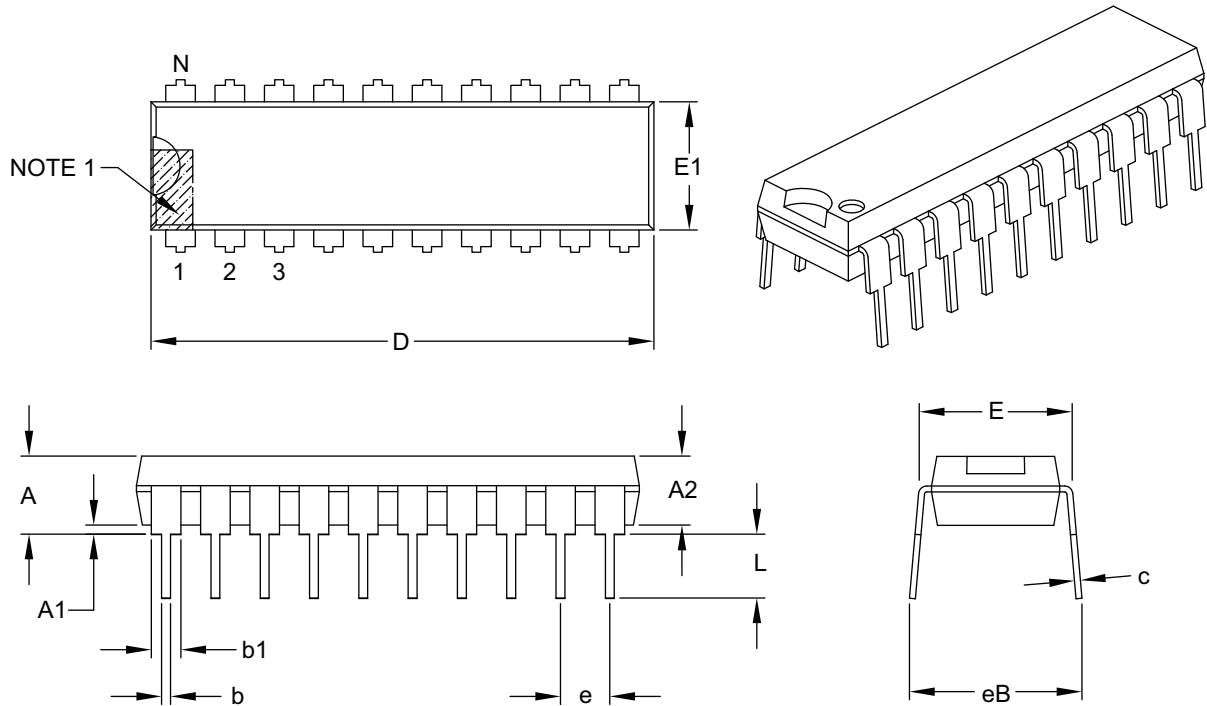
1. Dimensioning and tolerancing per ASME Y14.5M

BSC: Basic Dimension. Theoretically exact value shown without tolerances.

Microchip Technology Drawing No. C04-2087A

20-Lead Plastic Dual In-Line (P) – 300 mil Body [PDIP]

Note: For the most current package drawings, please see the Microchip Packaging Specification located at <http://www.microchip.com/packaging>



| Dimension Limits | Units | INCHES | | |
|----------------------------|-------|----------|-------|-------|
| | | MIN | NOM | MAX |
| Number of Pins | N | 20 | | |
| Pitch | e | .100 BSC | | |
| Top to Seating Plane | A | – | – | .210 |
| Molded Package Thickness | A2 | .115 | .130 | .195 |
| Base to Seating Plane | A1 | .015 | – | – |
| Shoulder to Shoulder Width | E | .300 | .310 | .325 |
| Molded Package Width | E1 | .240 | .250 | .280 |
| Overall Length | D | .980 | 1.030 | 1.060 |
| Tip to Seating Plane | L | .115 | .130 | .150 |
| Lead Thickness | c | .008 | .010 | .015 |
| Upper Lead Width | b1 | .045 | .060 | .070 |
| Lower Lead Width | b | .014 | .018 | .022 |
| Overall Row Spacing § | eB | – | – | .430 |

Notes:

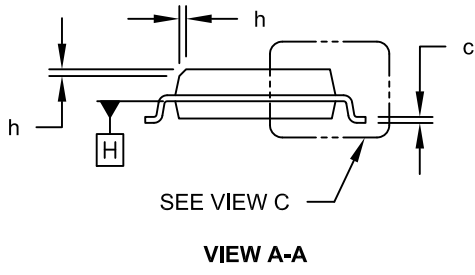
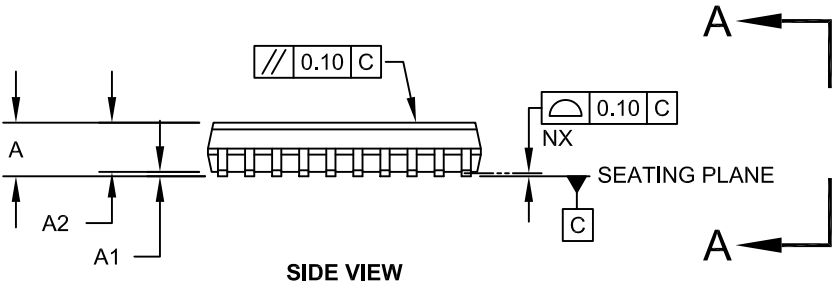
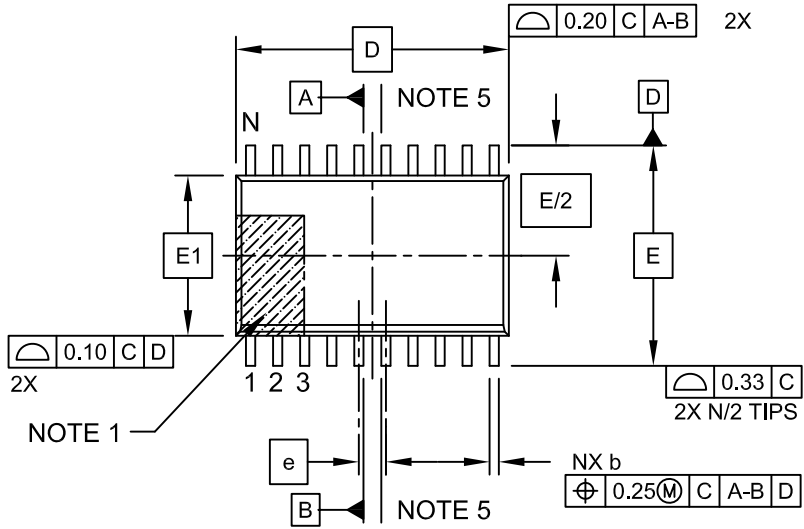
- Pin 1 visual index feature may vary, but must be located within the hatched area.
- § Significant Characteristic.
- Dimensions D and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed .010" per side.
- Dimensioning and tolerancing per ASME Y14.5M.

BSC: Basic Dimension. Theoretically exact value shown without tolerances.

Microchip Technology Drawing C04-019B

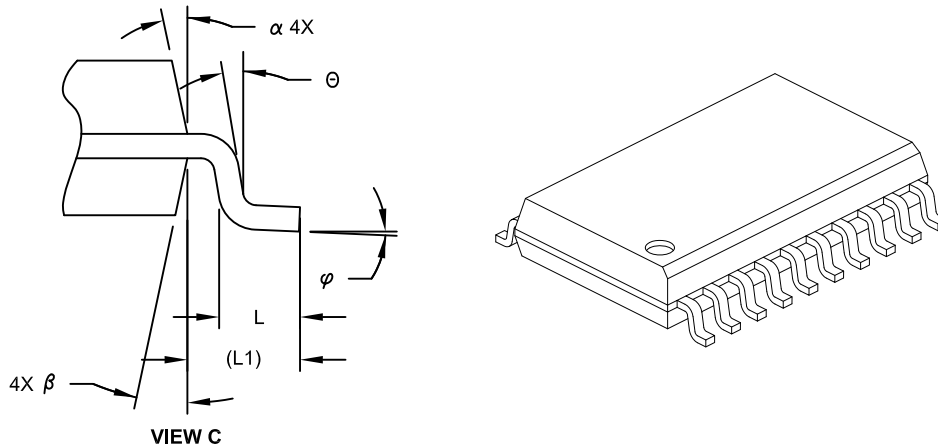
20-Lead Plastic Small Outline (SO) - Wide, 7.50 mm Body [SOIC]

Note: For the most current package drawings, please see the Microchip Packaging Specification located at <http://www.microchip.com/packaging>



20-Lead Plastic Small Outline (SO) - Wide, 7.50 mm Body [SOIC]

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 Pins | N | | 20 | | |
| Pitch | e | | 1.27 BSC | | |
| Overall Height | A | - | - | 2.65 | |
| Molded Package Thickness | A2 | 2.05 | - | - | |
| Standoff § | A1 | 0.10 | - | 0.30 | |
| Overall Width | E | | 10.30 BSC | | |
| Molded Package Width | E1 | | 7.50 BSC | | |
| Overall Length | D | | 12.80 BSC | | |
| Chamfer (Optional) | h | 0.25 | - | 0.75 | |
| Foot Length | L | 0.40 | - | 1.27 | |
| Footprint | L1 | | 1.40 REF | | |
| Lead Angle | θ | 0° | - | - | |
| Foot Angle | φ | 0° | - | 8° | |
| Lead Thickness | c | 0.20 | - | 0.33 | |
| Lead Width | b | 0.31 | - | 0.51 | |
| Mold Draft Angle Top | α | 5° | - | 15° | |
| Mold Draft Angle Bottom | β | 5° | - | 15° | |

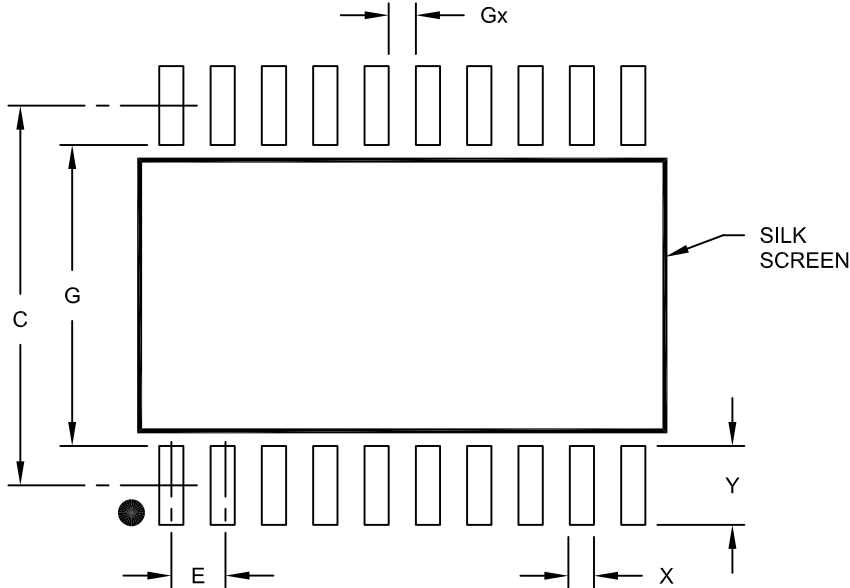
Notes:

- Pin 1 visual index feature may vary, but must be located within the hatched area.
- § Significant Characteristic
- Dimension D does not include mold flash, protrusions or gate burrs, which shall not exceed 0.15 mm per end. Dimension E1 does not include interlead flash or protrusion, which shall not exceed 0.25 mm 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.
- Datums A & B to be determined at Datum H.

Microchip Technology Drawing No. C04-094C Sheet 2 of 2

20-Lead Plastic Small Outline (SO) - Wide, 7.50 mm Body [SOIC]

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 | 1.27 BSC | | |
| Contact Pad Spacing | C | | 9.40 | |
| Contact Pad Width (X20) | X | | | 0.60 |
| Contact Pad Length (X20) | Y | | | 1.95 |
| Distance Between Pads | Gx | 0.67 | | |
| Distance Between Pads | G | 7.45 | | |

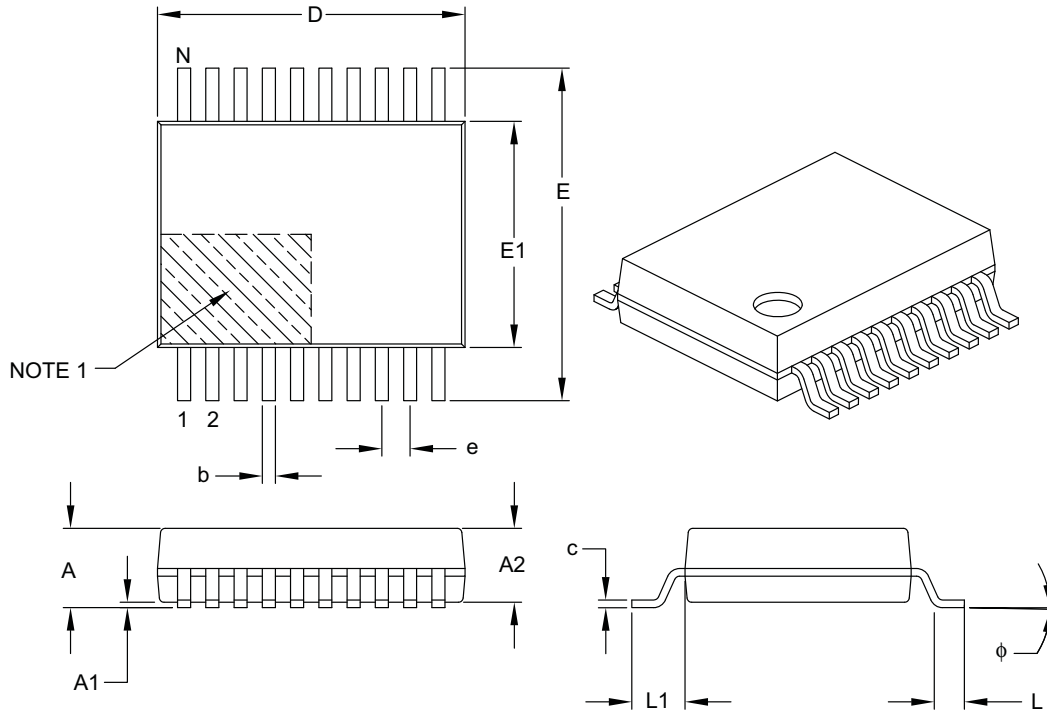
Notes:

- 1. Dimensioning and tolerancing per ASME Y14.5M
 BSC: Basic Dimension. Theoretically exact value shown without tolerances.

Microchip Technology Drawing No. C04-2094A

20-Lead Plastic Shrink Small Outline (SS) – 5.30 mm Body [SSOP]

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 Pins | N | 20 | | |
| Pitch | e | 0.65 BSC | | |
| Overall Height | A | – | – | 2.00 |
| Molded Package Thickness | A2 | 1.65 | 1.75 | 1.85 |
| Standoff | A1 | 0.05 | – | – |
| Overall Width | E | 7.40 | 7.80 | 8.20 |
| Molded Package Width | E1 | 5.00 | 5.30 | 5.60 |
| Overall Length | D | 6.90 | 7.20 | 7.50 |
| Foot Length | L | 0.55 | 0.75 | 0.95 |
| Footprint | L1 | 1.25 REF | | |
| Lead Thickness | c | 0.09 | – | 0.25 |
| Foot Angle | ϕ | 0° | 4° | 8° |
| Lead Width | b | 0.22 | – | 0.38 |

Notes:

1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. Dimensions D and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed 0.20 mm per side.
3. Dimensioning and tolerancing per ASME Y14.5M.

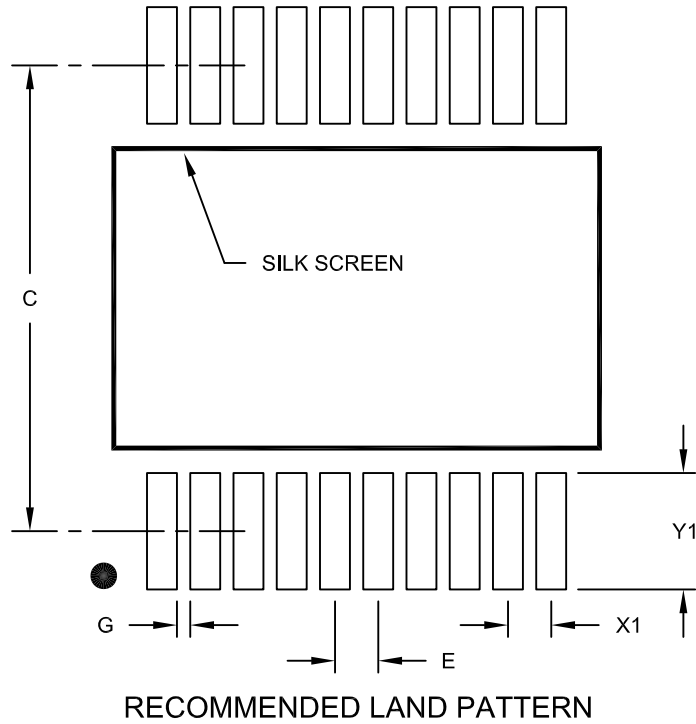
BSC: Basic Dimension. Theoretically exact value shown without tolerances.

REF: Reference Dimension, usually without tolerance, for information purposes only.

Microchip Technology Drawing C04-072B

20-Lead Plastic Shrink Small Outline (SS) - 5.30 mm Body [SSOP]

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 |
| Contact Pitch | E | 0.65 BSC | | |
| Contact Pad Spacing | C | | 7.20 | |
| Contact Pad Width (X20) | X1 | | | 0.45 |
| Contact Pad Length (X20) | Y1 | | | 1.75 |
| Distance Between Pads | G | 0.20 | | |

Notes:

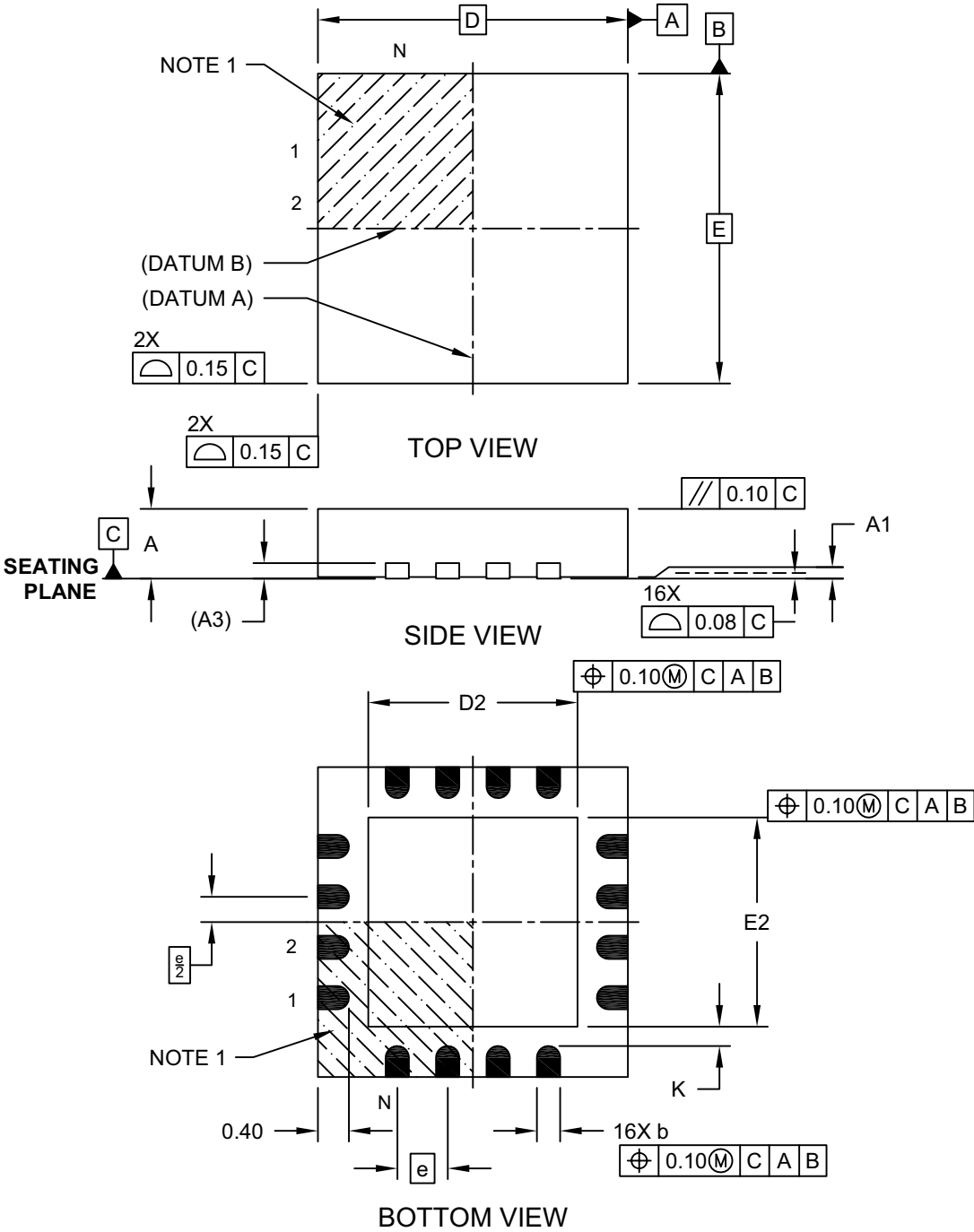
1. Dimensioning and tolerancing per ASME Y14.5M

BSC: Basic Dimension. Theoretically exact value shown without tolerances.

Microchip Technology Drawing No. C04-2072A

16-Lead Plastic Quad Flat, No Lead Package (ML) - 4x4x0.9mm Body [QFN]

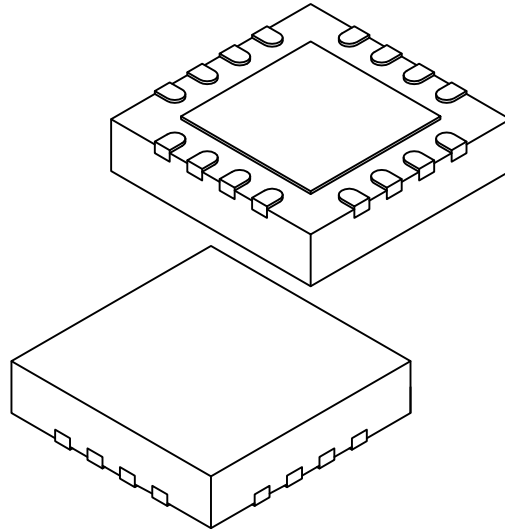
Note: For the most current package drawings, please see the Microchip Packaging Specification located at <http://www.microchip.com/packaging>



Microchip Technology Drawing C04-127D Sheet 1 of 2

16-Lead Plastic Quad Flat, No Lead Package (ML) - 4x4x0.9mm Body [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 | | 16 | | |
| Pitch | e | | 0.65 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 | | 4.00 BSC | | |
| Exposed Pad Width | E2 | | 2.50 | 2.65 | 2.80 |
| Overall Length | D | | 4.00 BSC | | |
| Exposed Pad Length | D2 | | 2.50 | 2.65 | 2.80 |
| Contact Width | b | | 0.25 | 0.30 | 0.35 |
| Contact Length | L | | 0.30 | 0.40 | 0.50 |
| Contact-to-Exposed Pad | K | | 0.20 | - | - |

Notes:

1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. Package is saw singulated
3. Dimensioning and tolerancing per ASME Y14.5M

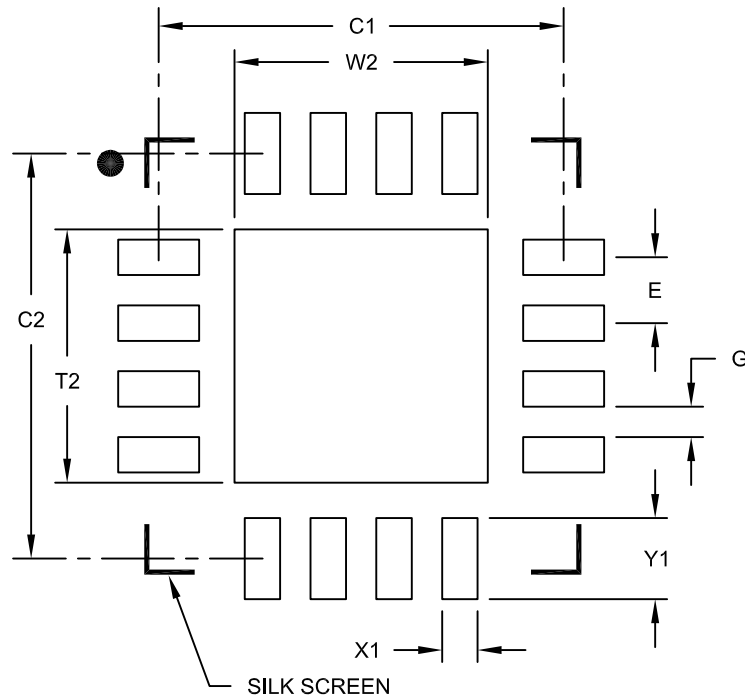
BSC: Basic Dimension. Theoretically exact value shown without tolerances.

REF: Reference Dimension, usually without tolerance, for information purposes only.

Microchip Technology Drawing C04-127D Sheet 2 of 2

16-Lead Plastic Quad Flat, No Lead Package (ML) - 4x4x0.9mm Body [QFN]

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 | 0.65 BSC | | |
| Optional Center Pad Width | W2 | | | 2.50 |
| Optional Center Pad Length | T2 | | | 2.50 |
| Contact Pad Spacing | C1 | | 4.00 | |
| Contact Pad Spacing | C2 | | 4.00 | |
| Contact Pad Width (X28) | X1 | | | 0.35 |
| Contact Pad Length (X28) | Y1 | | | 0.80 |
| Distance Between Pads | G | 0.30 | | |

Notes:

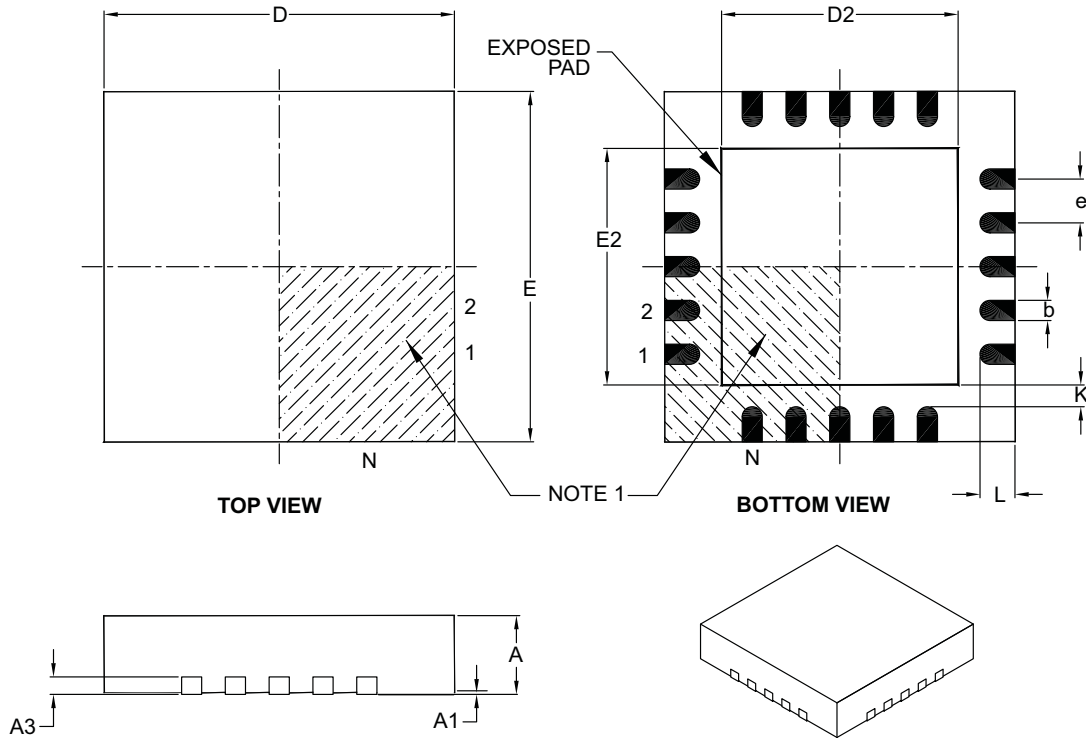
1. Dimensioning and tolerancing per ASME Y14.5M

BSC: Basic Dimension. Theoretically exact value shown without tolerances.

Microchip Technology Drawing No. C04-2127A

20-Lead Plastic Quad Flat, No Lead Package (ML) – 4x4x0.9 mm Body [QFN]

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 Pins | N | 20 | | |
| 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 | 4.00 BSC | | |
| Exposed Pad Width | E2 | 2.60 | 2.70 | 2.80 |
| Overall Length | D | 4.00 BSC | | |
| Exposed Pad Length | D2 | 2.60 | 2.70 | 2.80 |
| Contact Width | b | 0.18 | 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.

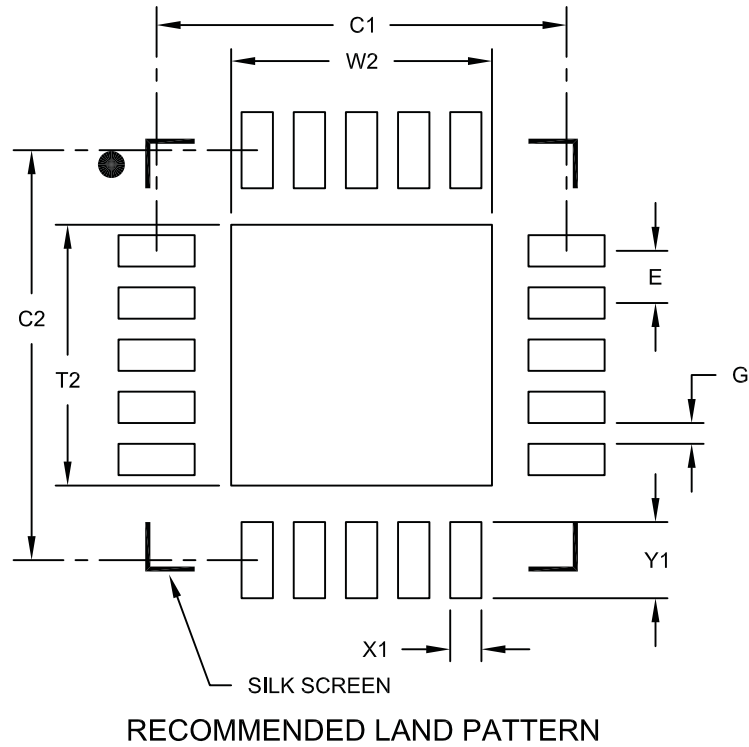
BSC: Basic Dimension. Theoretically exact value shown without tolerances.

REF: Reference Dimension, usually without tolerance, for information purposes only.

Microchip Technology Drawing C04-126B

20-Lead Plastic Quad Flat, No Lead Package (ML) - 4x4 mm Body [QFN]
With 0.40 mm Contact Length

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 |
| Contact Pitch | E | 0.50 BSC | | |
| Optional Center Pad Width | W2 | | | 2.50 |
| Optional Center Pad Length | T2 | | | 2.50 |
| Contact Pad Spacing | C1 | | 3.93 | |
| Contact Pad Spacing | C2 | | 3.93 | |
| Contact Pad Width | X1 | | | 0.30 |
| Contact Pad Length | Y1 | | | 0.73 |
| Distance Between Pads | G | 0.20 | | |

Notes:

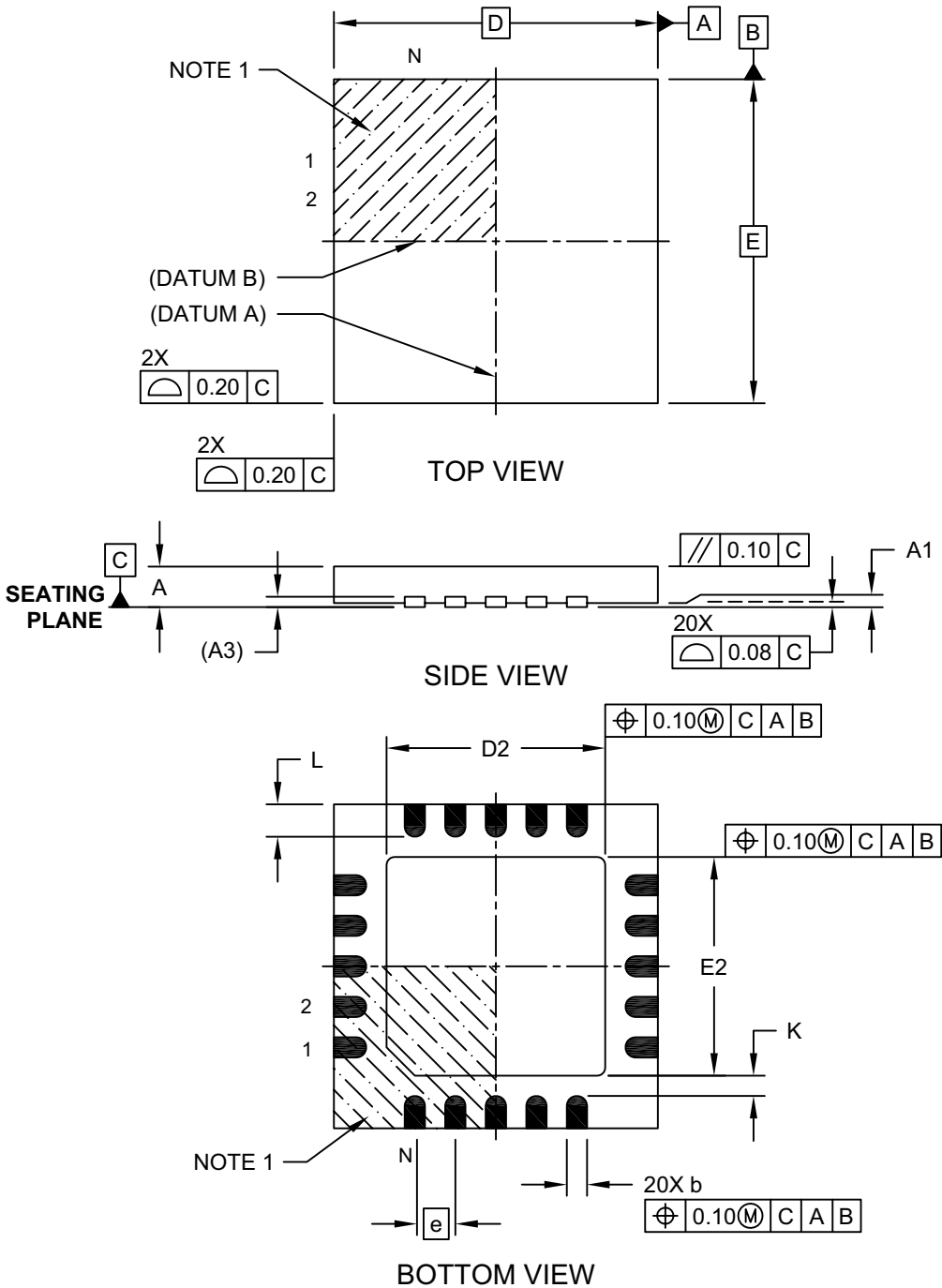
1. Dimensioning and tolerancing per ASME Y14.5M

BSC: Basic Dimension. Theoretically exact value shown without tolerances.

Microchip Technology Drawing No. C04-2126A

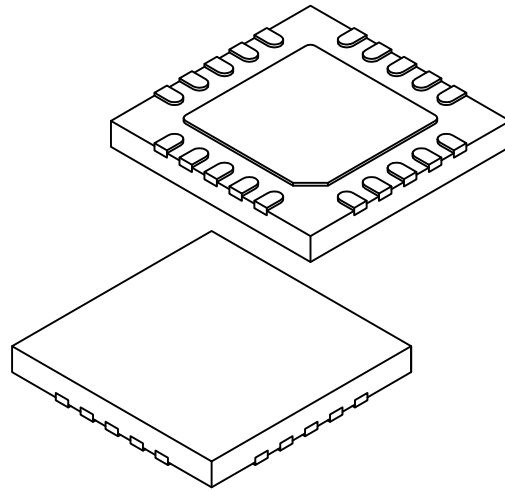
20-Lead Ultra Thin Plastic Quad Flat, No Lead Package (GZ) - 4x4x0.5 mm Body [UQFN]

Note: For the most current package drawings, please see the Microchip Packaging Specification located at <http://www.microchip.com/packaging>



20-Lead Ultra Thin Plastic Quad Flat, No Lead Package (GZ) - 4x4x0.5 mm Body [UQFN]

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 Terminals | N | | 20 | | |
| Pitch | e | | 0.50 BSC | | |
| Overall Height | A | 0.45 | 0.50 | 0.55 | |
| Standoff | A1 | 0.00 | 0.02 | 0.05 | |
| Terminal Thickness | A3 | 0.127 REF | | | |
| Overall Width | E | 4.00 BSC | | | |
| Exposed Pad Width | E2 | 2.60 | 2.70 | 2.80 | |
| Overall Length | D | 4.00 BSC | | | |
| Exposed Pad Length | D2 | 2.60 | 2.70 | 2.80 | |
| Terminal Width | b | 0.20 | 0.25 | 0.30 | |
| Terminal Length | L | 0.30 | 0.40 | 0.50 | |
| Terminal-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

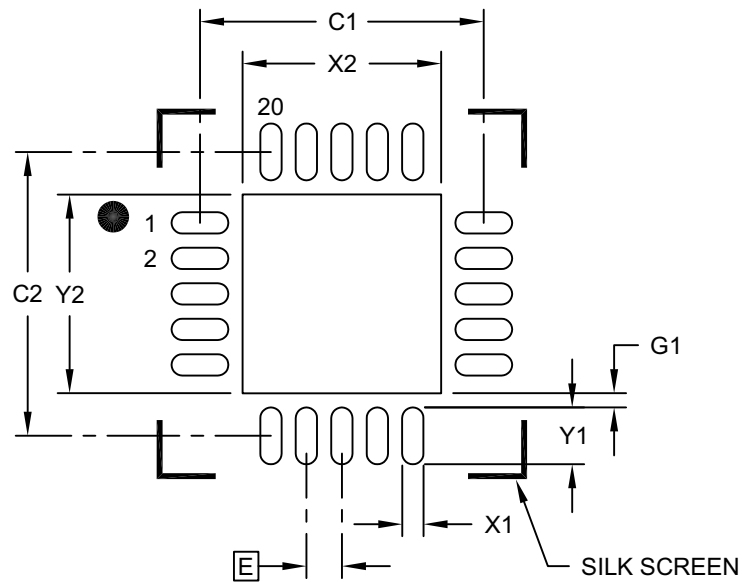
BSC: Basic Dimension. Theoretically exact value shown without tolerances.

REF: Reference Dimension, usually without tolerance, for information purposes only.

Microchip Technology Drawing C04-255A Sheet 2 of 2

20-Lead Ultra Thin Plastic Quad Flat, No Lead Package (GZ) - 4x4x0.5 mm Body [UQFN]

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 | X2 | | | 2.80 |
| Optional Center Pad Length | Y2 | | | 2.80 |
| Contact Pad Spacing | C1 | | 4.00 | |
| Contact Pad Spacing | C2 | | 4.00 | |
| Contact Pad Width (X20) | X1 | | | 0.30 |
| Contact Pad Length (X20) | Y1 | | | 0.80 |
| Contact Pad to Center Pad (X20) | G1 | 0.20 | | |

Notes:

1. Dimensioning and tolerancing per ASME Y14.5M

BSC: Basic Dimension. Theoretically exact value shown without tolerances.

Microchip Technology Drawing C04-2255A

APPENDIX A: DATA SHEET REVISION HISTORY

Revision A (10/2014)

Original release.

Revision B (4/2015)

Added High-Current pins.

Updated PIC12/16(L)F161X Family Types table and Packages Table

Added Figures 36-7 and 36-8 for VOH vs. IOH for high drive pins.

Deleted Figures 36-27 and 36-28.

Updated Example 3-2.

Updated Figures 5-1, 23-1, 23-2, 28-12, 31-1, 31-2, 31-4, 1, and 35-6.

Updated Registers 23-4, 27-2, 31-4 and 31-7.

Updated Sections 26.3, 25.4.2, 27.1, 30.0, 35.0, and 35.1.

Updated Table 1-2, 1-3, 3-5, 3-16, 13-2, 23-3, 23-4, 23-7, 28-1, 35-4, 35-8, and 35-17.

Updated Section 23 - added missing modes/mode summary table, reworded text to be more clear/descriptive.

Minor typos corrected.

Revision C (5/2016)

Added High endurance column to Table 1: PIC12/16(L)F161x Family Types.

Minor typos corrected.

Updated the High-Endurance Flash data memory information on the cover page. Updated Register 21-1.

Updated Section 19.6, 19.7. Updated Registers 19-2, 29-1, and 31-22. Updated Table 3: Pin Allocations Table and Table 5-1. Updated Figure 19-2.

Updated Package Drawings C04-127.

Revision D (11/2017)

Added Equation 21-2: R-C Calculations and Example 21-1. Added Sections 5.3.2 Clock Switching Before Sleep; 20.5.1: Correction by AC Coupling, and 23.2: PRx Period Register.

Updated Example 17-1; Figure 16-1; Register 12-8, 12-16, 24-4; Sections 11.3, 11.5, 17.1.2, 17.2.6, 23.1 and 24.6; Tables 1, 1-3, 5.1, 24-4, 35-8 and 35-11.

Removed Figures 36-29, 36-30, IPD Watchdog Timer.

THE MICROCHIP WEBSITE

Microchip provides online support via our website at www.microchip.com. This website is used as a means to make files and information easily available to customers. Accessible by using your favorite Internet browser, the website 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 website 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 website at: <http://www.microchip.com/support>

PRODUCT IDENTIFICATION SYSTEM

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

| <u>PART NO.</u> | <u>[X]⁽¹⁾</u> | - | <u>X</u> | <u>/XX</u> | <u>XXX</u> |
|-------------------------------|---|---|-------------------|------------|------------|
| Device | Tape and Reel Option | | Temperature Range | Package | Pattern |
| Device: | PIC16LF1615, PIC16F1615, PIC16LF1619, PIC16F1619 | | | | |
| 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⁾ | ML = QFN (16-Lead and 20-Lead) P = Plastic DIP SL = SOIC (14-Lead) ST = TSSOP GZ = UQFN (20-Lead) | | | | |
| Pattern: | QTP, SQTP, Code or Special Requirements (blank otherwise) | | | | |

Examples:

- a) PIC16LF1615T - I/SL
Tape and Reel, Industrial temperature, SOIC package
- b) PIC16F1619 - I/P
Industrial temperature PDIP package
- c) PIC16F1619 - E/ML 298
Extended temperature, QFN package
QTP pattern #298

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

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.

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.

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

Trademarks

The Microchip name and logo, the Microchip logo, AnyRate, AVR, AVR logo, AVR Freaks, BeaconThings, BitCloud, chipKIT, chipKIT logo, CryptoMemory, CryptoRF, dsPIC, FlashFlex, flexPWR, Helder, JukeBlox, KEELoQ, KEELoQ logo, Kleer, LANCheck, LINK MD, maXStylus, maXTouch, MedialB, megaAVR, MOST, MOST logo, MPLAB, OptoLyzer, PIC, picoPower, PICSTART, PIC32 logo, Prochip Designer, QTouch, RightTouch, SAM-BA, SpyNIC, SST, SST Logo, SuperFlash, tinyAVR, UNI/O, and XMEGA are registered trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

ClockWorks, The Embedded Control Solutions Company, EtherSynch, Hyper Speed Control, HyperLight Load, IntelliMOS, mTouch, Precision Edge, and Quiet-Wire are registered trademarks of Microchip Technology Incorporated in the U.S.A.

Adjacent Key Suppression, AKS, Analog-for-the-Digital Age, Any Capacitor, AnyIn, AnyOut, BodyCom, CodeGuard, CryptoAuthentication, CryptoCompanion, CryptoController, dsPICDEM, dsPICDEM.net, Dynamic Average Matching, DAM, ECAN, EtherGREEN, In-Circuit Serial Programming, ICSP, Inter-Chip Connectivity, JitterBlocker, KleerNet, KleerNet logo, Mindi, MiWi, motorBench, MPASM, MPF, MPLAB Certified logo, MPLIB, MPLINK, MultiTRAK, NetDetach, Omniscient Code Generation, PICDEM, PICDEM.net, PICKit, PICtail, PureSilicon, QMatrix, RightTouch logo, REAL ICE, Ripple Blocker, SAM-ICE, Serial Quad I/O, SMART-I.S., SQI, SuperSwitcher, SuperSwitcher II, 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.

© 2014-2017, Microchip Technology Incorporated, All Rights Reserved.

ISBN: 978-1-5224-2316-4



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

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
Tel: 317-536-2380

Los Angeles

Mission Viejo, CA
Tel: 949-462-9523
Fax: 949-462-9608
Tel: 951-273-7800

Raleigh, NC

Tel: 919-844-7510

New York, NY

Tel: 631-435-6000

San Jose, CA

Tel: 408-735-9110
Tel: 408-436-4270

Canada - Toronto

Tel: 905-695-1980
Fax: 905-695-2078

ASIA/PACIFIC

Australia - Sydney
Tel: 61-2-9868-6733

China - Beijing
Tel: 86-10-8569-7000

China - Chengdu
Tel: 86-28-8665-5511

China - Chongqing
Tel: 86-23-8980-9588

China - Dongguan
Tel: 86-769-8702-9880

China - Guangzhou
Tel: 86-20-8755-8029

China - Hangzhou
Tel: 86-571-8792-8115

China - Hong Kong SAR
Tel: 852-2943-5100

China - Nanjing
Tel: 86-25-8473-2460

China - Qingdao
Tel: 86-532-8502-7355

China - Shanghai
Tel: 86-21-3326-8000

China - Shenyang
Tel: 86-24-2334-2829

China - Shenzhen
Tel: 86-755-8864-2200

China - Suzhou
Tel: 86-186-6233-1526

China - Wuhan
Tel: 86-27-5980-5300

China - Xian
Tel: 86-29-8833-7252

China - Xiamen
Tel: 86-592-2388138

China - Zhuhai
Tel: 86-756-3210040

ASIA/PACIFIC

India - Bangalore
Tel: 91-80-3090-4444

India - New Delhi
Tel: 91-11-4160-8631

India - Pune
Tel: 91-20-4121-0141

Japan - Osaka
Tel: 81-6-6152-7160

Japan - Tokyo
Tel: 81-3-6880-3770

Korea - Daegu
Tel: 82-53-744-4301

Korea - Seoul
Tel: 82-2-554-7200

Malaysia - Kuala Lumpur
Tel: 60-3-7651-7906

Malaysia - Penang
Tel: 60-4-227-8870

Philippines - Manila
Tel: 63-2-634-9065

Singapore
Tel: 65-6334-8870

Taiwan - Hsin Chu
Tel: 886-3-577-8366

Taiwan - Kaohsiung
Tel: 886-7-213-7830

Taiwan - Taipei
Tel: 886-2-2508-8600

Thailand - Bangkok
Tel: 66-2-694-1351

Vietnam - Ho Chi Minh
Tel: 84-28-5448-2100

EUROPE

Austria - Wels
Tel: 43-7242-2244-39
Fax: 43-7242-2244-393

Denmark - Copenhagen
Tel: 45-4450-2828
Fax: 45-4485-2829

Finland - Espoo
Tel: 358-9-4520-820

France - Paris
Tel: 33-1-69-53-63-20
Fax: 33-1-69-30-90-79

Germany - Garching
Tel: 49-8931-9700

Germany - Haan
Tel: 49-2129-3766400

Germany - Heilbronn
Tel: 49-7131-67-3636

Germany - Karlsruhe
Tel: 49-721-625370

Germany - Munich
Tel: 49-89-627-144-0
Fax: 49-89-627-144-44

Germany - Rosenheim
Tel: 49-8031-354-560

Israel - Ra'anana
Tel: 972-9-744-7705

Italy - Milan
Tel: 39-0331-742611
Fax: 39-0331-466781

Italy - Padova
Tel: 39-049-7625286

Netherlands - Drunen
Tel: 31-416-690399
Fax: 31-416-690340

Norway - Trondheim
Tel: 47-7289-7561

Poland - Warsaw
Tel: 48-22-3325737

Romania - Bucharest
Tel: 40-21-407-87-50

Spain - Madrid
Tel: 34-91-708-08-90
Fax: 34-91-708-08-91

Sweden - Gothenberg
Tel: 46-31-704-60-40

Sweden - Stockholm
Tel: 46-8-5090-4654

UK - Wokingham
Tel: 44-118-921-5800
Fax: 44-118-921-5820

Mouser Electronics

Authorized Distributor

Click to View Pricing, Inventory, Delivery & Lifecycle Information:

Microchip:

[PIC16F1619-E/ML](#) [PIC16F1619-I/ML](#) [PIC16F1615-I/ST](#) [PIC16LF1615-I/ML](#) [PIC16F1615-I/ML](#) [PIC16LF1615-E/P](#)
[PIC16LF1615-E/ST](#) [PIC16F1615T-I/ML](#) [PIC16LF1615T-I/ML](#) [PIC16F1615-E/ML](#) [PIC16LF1615-E/ML](#) [PIC16F1615-](#)
[E/SL](#) [PIC16F1619-I/P](#) [PIC16LF1615-I/SL](#) [PIC16F1615-E/P](#) [PIC16LF1619-I/P](#) [PIC16F1619-E/SO](#) [PIC16F1615-E/ST](#)
[PIC16F1619-I/SO](#) [PIC16LF1619-I/SS](#) [PIC16LF1619-I/ML](#) [PIC16LF1619-I/SO](#) [PIC16F1619-I/SS](#) [PIC16F1615-I/SL](#)
[PIC16LF1615-I/ST](#) [PIC16F1619-E/P](#) [PIC16LF1615-I/P](#) [PIC16F1615-I/P](#) [PIC16F1615T-I/JQ](#) [PIC16F1615-I/JQ](#)
[PIC16LF1615-I/JQ](#) [PIC16LF1615T-I/JQ](#) [PIC16F1615-E/JQ](#) [PIC16LF1615-E/JQ](#) [PIC16F1615T-I/SL](#) [PIC16F1615T-](#)
[I/ST](#) [PIC16F1619T-I/ML](#) [PIC16F1619T-I/SO](#) [PIC16LF1615T-I/ST](#) [PIC16LF1615T-I/SL](#) [PIC16LF1619T-I/SO](#)
[PIC16LF1619T-I/ML](#) [PIC16F1619-E/SS](#) [PIC16F1619T-E/SS](#) [PIC16LF1619T-I/SS](#) [PIC16F1619T-I/SS](#) [PIC16LF1619-](#)
[E/P](#) [PIC16LF1619-E/SS](#) [PIC16LF1619-E/SO](#) [PIC16LF1619-E/ML](#)

Компания «Океан Электроники» предлагает заключение долгосрочных отношений при поставках импортных электронных компонентов на взаимовыгодных условиях!

Наши преимущества:

- Поставка оригинальных импортных электронных компонентов напрямую с производств Америки, Европы и Азии, а так же с крупнейших складов мира;
- Широкая линейка поставок активных и пассивных импортных электронных компонентов (более 30 млн. наименований);
- Поставка сложных, дефицитных, либо снятых с производства позиций;
- Оперативные сроки поставки под заказ (от 5 рабочих дней);
- Экспресс доставка в любую точку России;
- Помощь Конструкторского Отдела и консультации квалифицированных инженеров;
- Техническая поддержка проекта, помощь в подборе аналогов, поставка прототипов;
- Поставка электронных компонентов под контролем ВП;
- Система менеджмента качества сертифицирована по Международному стандарту ISO 9001;
- При необходимости вся продукция военного и аэрокосмического назначения проходит испытания и сертификацию в лаборатории (по согласованию с заказчиком);
- Поставка специализированных компонентов военного и аэрокосмического уровня качества (Xilinx, Altera, Analog Devices, Intersil, Interpoint, Microsemi, Actel, Aeroflex, Peregrine, VPT, Syfer, Eurofarad, Texas Instruments, MS Kennedy, Miteq, Cobham, E2V, MA-COM, Hittite, Mini-Circuits, General Dynamics и др.);

Компания «Океан Электроники» является официальным дистрибьютором и эксклюзивным представителем в России одного из крупнейших производителей разъемов военного и аэрокосмического назначения «JONHON», а так же официальным дистрибьютором и эксклюзивным представителем в России производителя высокотехнологичных и надежных решений для передачи СВЧ сигналов «FORSTAR».



JONHON

«JONHON» (основан в 1970 г.)

Разъемы специального, военного и аэрокосмического назначения:

(Применяются в военной, авиационной, аэрокосмической, морской, железнодорожной, горно- и нефтедобывающей отраслях промышленности)

«FORSTAR» (основан в 1998 г.)

ВЧ соединители, коаксиальные кабели,
кабельные сборки и микроволновые компоненты:

(Применяются в телекоммуникациях гражданского и специального назначения, в средствах связи, РЛС, а так же военной, авиационной и аэрокосмической отраслях промышленности).



Телефон: 8 (812) 309-75-97 (многоканальный)

Факс: 8 (812) 320-03-32

Электронная почта: ocean@oceanchips.ru

Web: <http://oceanchips.ru/>

Адрес: 198099, г. Санкт-Петербург, ул. Калинина, д. 2, корп. 4, лит. А