chanhvinh.com

# PIC16F17156/76

## PIC16F17156/76 Full-Featured 28/40-Pin Microcontrollers

## Introduction

The PIC16F171 microcontroller family has a suite of analog peripherals that enables precision sensor applications. This product family is available from 8 to 44-pin packages in a memory range of 7 KB to 28 KB, with speeds up to 32 MHz. This product includes a low-noise Operational Amplifier (Op-Amp), a 12-bit differential Analog-to-Digital Converter with Computation (ADCC), two 8-bit Digital-to-Analog Converters (DAC), a 16-bit Pulse-Width Modulation (PWM) peripheral, and many more waveform control and communication peripherals. This small form factor, feature-rich device is well suited for low-cost, energy-efficient analog sensor applications with higher resolution requirements.

## PIC16F171 Family Summary

**Table 1. Devices Included in This Data Sheet**

| Device | Program Flash Memory (bytes) | Data Flash Memory (EEPROM) (bytes) | Data SRAM (bytes) | Memory Access Partition/ Device Information Area | I/O Pins(1)/ Peripheral Pin Select | 8-Bit Timers with HLT/ 16-Bit Timers(2) | 16-Bit PWM/ CCP | 12-Bit ADC Channels (External+/External-/Internal) | I2C/SPI | EUSART | NCO | CWG | CLC | FVR | CMP | 8-bit DAC | Operational Amplifier | ZCD | SMBus Compatible I/O Pads | External Interrupt Pins | Interrupt-on-Change Pins | Windowed Watchdog Timer |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| PIC16F17156 | 28k | 256 | 2048 | Y/ Y | 25/ Y | 3/2 | 4/2 | 24/11/7 | 2/2 | 2 | 1 | 1 | 4 | 2 | 2 | 2 | 1 | 1 | Y | 1 | 25 | Y |
| PIC16F17176 | 28k | 256 | 2048 | Y/ Y | 36/ Y | 3/2 | 4/2 | 35/16/7 | 2/2 | 2 | 1 | 1 | 4 | 2 | 2 | 2 | 1 | 1 | Y | 1 | 25 | Y |

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



**Table 2. Devices Not Included in This Data Sheet**

| Device | Program Flash Memory (bytes) | Data Flash Memory (EEPROM) (bytes) | Data SRAM (bytes) | Memory Access Partition/ Device Information Area | I/O Pins(1)/ Peripheral Pin Select | 8-Bit Timers with HLT/ 16-Bit Timers(2) | 16-Bit PWM/ CCP | 12-Bit ADC Channels (External+/External-/Internal) | I2C/SPI | EUSART | NCO | CWG | CLC | FVR | CMP | 8-bit DAC | Operational Amplifier | ZCD | SMBus Compatible I/O Pads | External Interrupt Pins | Interrupt-on-Change Pins | Watchdog Timer |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| PIC16F17114 | 7k | 128 | 512 | Y/Y | 6/Y | 2/2 | 2/2 | 5/5/7 | 2/2 | 2 | 1 | 1 | 4 | 2 | 2 | 2 | 1 | 1 | Y | 1 | 6 | Y |
| PIC16F17115 | 14k | 128 | 1024 | Y/Y | 6/Y | 2/2 | 2/2 | 5/5/7 | 2/2 | 2 | 1 | 1 | 4 | 2 | 2 | 2 | 1 | 1 | Y | 1 | 6 | Y |
| PIC16F17124 | 7k | 128 | 512 | Y/Y | 12/Y | 2/2 | 2/2 | 11/11/7 | 2/2 | 2 | 1 | 1 | 4 | 2 | 2 | 2 | 1 | 1 | Y | 1 | 12 | Y |
| PIC16F17125 | 14k | 128 | 1024 | Y/Y | 12/Y | 2/2 | 2/2 | 11/11/7 | 2/2 | 2 | 1 | 1 | 4 | 2 | 2 | 2 | 1 | 1 | Y | 1 | 12 | Y |
| PIC16F17126 | 28k | 256 | 2048 | Y/Y | 12/Y | 2/2 | 2/2 | 11/11/7 | 2/2 | 2 | 1 | 1 | 4 | 2 | 2 | 2 | 1 | 1 | Y | 1 | 12 | Y |
| PIC16F17144 | 7k | 128 | 512 | Y/Y | 18/Y | 2/2 | 2/2 | 17/17/7 | 2/2 | 2 | 1 | 1 | 4 | 2 | 2 | 2 | 1 | 1 | Y | 1 | 18 | Y |
| PIC16F17145 | 14k | 128 | 1024 | Y/Y | 18/Y | 2/2 | 2/2 | 17/17/7 | 2/2 | 2 | 1 | 1 | 4 | 2 | 2 | 2 | 1 | 1 | Y | 1 | 18 | Y |
| PIC16F17146 | 28k | 256 | 2048 | Y/Y | 18/Y | 2/2 | 2/2 | 17/17/7 | 2/2 | 2 | 1 | 1 | 4 | 2 | 2 | 2 | 1 | 1 | Y | 1 | 18 | Y |
| PIC16F17154 | 7k | 128 | 512 | Y/Y | 25/Y | 3/2 | 4/2 | 24/11/7 | 2/2 | 2 | 1 | 1 | 4 | 2 | 2 | 2 | 1 | 1 | Y | 1 | 25 | Y |
| PIC16F17155 | 14k | 128 | 1024 | Y/Y | 25/Y | 3/2 | 4/2 | 24/11/7 | 2/2 | 2 | 1 | 1 | 4 | 2 | 2 | 2 | 1 | 1 | Y | 1 | 25 | Y |
| PIC16F17174 | 7k | 128 | 512 | Y/Y | 36/Y | 3/2 | 4/2 | 35/16/7 | 2/2 | 2 | 1 | 1 | 4 | 2 | 2 | 2 | 1 | 1 | Y | 1 | 25 | Y |
| PIC16F17175 | 14k | 128 | 1024 | Y/Y | 36/Y | 3/2 | 4/2 | 35/16/7 | 2/2 | 2 | 1 | 1 | 4 | 2 | 2 | 2 | 1 | 1 | Y | 1 | 25 | Y |

**Notes:**

1. Total I/O count includes one input-only pin ($\overline{MCLR}$).
2. Timer0 can be configured as either an 8 or 16-bit timer.

## Core Features

- C Compiler Optimized RISC Architecture
- Operating Speed:
  - DC – 32 MHz clock input
  - 125 ns minimum instruction time
- 16-Level Deep Hardware Stack
- Low-Current Power-on Reset (POR)
- Configurable Power-up Timer (PWRT)
- Brown-out Reset (BOR)
- Low-Power Brown-out Reset (LPBOR)
- Windowed Watchdog Timer (WWDT)

## Memory

- Up to 28 KB of Program Flash Memory
- Up to 2 KB of Data SRAM Memory
- Up to 256 Bytes of Data EEPROM Memory
- Memory Access Partition (MAP) with Program Flash Memory Partitioned into:

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



  - Application block
  - Boot block
  - Storage Area Flash (SAF) block
- Programmable Code Protection and Write Protection
- Device Information Area (DIA) Stores:
  - Temperature Indicator calibration coefficients
  - Fixed Voltage Reference (FVR) measurement data
  - Microchip Unique Identifier (MUI)
- Device Characteristics Area (DCI) Stores:
  - Program/erase row sizes
  - Pin count details
- Direct, Indirect, and Relative Addressing Modes

## Operating Characteristics

- Operating Voltage Range:
  - 1.8V to 5.5V
- Temperature Range:
  - Industrial: -40°C to 85°C
  - Extended: -40°C to 125°C

## Power-Saving Functionality

- Doze: CPU and Peripherals Running at Different Cycle Rates (typically CPU is lower)
- Idle: CPU Halted While Peripherals Operate
- Sleep:
  - Lowest power consumption
  - Reduce system electrical noise while performing ADC conversions
- Peripheral Module Disable (PMD): Selectively Minimize the Active Power Consumption of Unused Peripherals
- Low Power Mode Features:
  - Sleep current:
    - < 900 nA typical @ 3V/25°C (WDT enabled)
    - < 600 nA typical @ 3V/25°C (WDT disabled)
  - Operating current:
    - 48 µA typical @ 32 kHz, 3V/25°C
    - < 1 mA typical @ 4 MHz, 5V/25°C

## Digital Peripherals

- Two Capture/Compare/PWM (CCP) Modules:
  - 16-bit resolution for Capture/Compare modes
  - 10-bit resolution for PWM mode
- Up to Four Pulse-Width Modulators (PWM):
  - 16-bit resolution
  - Independent pulse outputs
  - External Reset Signal (ERS) inputs
- Four Configurable Logic Cells (CLC):
  - Integrated combinational and sequential logic
- One Complimentary Waveform Generator (CWG):

© 2023 Microchip Technology Inc.
and its subsidiaries

chanhvinh.com



  - Rising and falling edge dead-band control
  - Full-bridge, half-bridge, 1-channel drive
  - Multiple signal sources
  - Programmable dead band
  - Fault-shutdown input
- One Configurable 8/16-Bit Timer (TMR0)
- Two 16-Bit Timers (TMR1/3) with Gate Control
- Up to Three 8-Bit Timers (TMR2/4/6) with Hardware Limit Timer (HLT)
- One Numerically Controlled Oscillator (NCO):
  - Generates true linear frequency control and increased frequency resolution
  - Input clock up to 64 MHz
- Programmable CRC with Memory Scan:
  - Reliable data/program memory monitoring for Fail-Safe operation (e.g., Class B)
  - Calculate 32-bit CRC over any portion of Program Flash Memory
- Two Enhanced Universal Synchronous Asynchronous Receiver Transmitters (EUSART):
  - RS-232, RS-485, LIN compatible
  - Auto-wake-up on Start
- Two Host Synchronous Serial Ports (MSSP):
  - Serial Peripheral Interface (SPI) mode
    - Chip Select Synchronization
  - Inter-Integrated Circuit ($I^2C$) mode
    - 7/10-bit Addressing modes
- Peripheral Pin Select (PPS):
  - Enables pin mapping of digital I/O
- Device I/O Port Features:
  - Up to 35 I/O pins
  - One input-only pin
  - Individual I/O direction, open-drain, input threshold, slew rate, and weak pull-up control
  - Interrupt-on-Change (IOC) on up to 25 pins
  - One external interrupt pin

## Analog Peripherals

- Differential Analog-to-Digital Converter with Computation (ADCC):
  - 12-bit resolution
  - Up to 35 external positive input channels
  - Up to 17 external negative input channels
  - Seven internal input channels
  - Internal ADC oscillator (ADCRC)
  - Operates in Sleep
  - Selectable Auto-Conversion Trigger (ACT) sources
- Two 8-Bit Digital-to-Analog Converters (DAC):
  - Output available on up to two I/O pins
  - Internal connections to ADC, Operational Amplifier and Comparators
- Two Comparators (CMP):
  - Up to four external inputs
  - Configurable output polarity
  - External output via Peripheral Pin Select
- One Operational Amplifier:

© 2023 Microchip Technology Inc.
and its subsidiaries

chanhvinh.com



  - 2.3 MHz gain bandwidth
  - Programmable gain
  - Internal gain resistor ladder
- Zero-Cross Detect (ZCD):
  - Detect when AC signal on pin crosses ground
- Two Fixed Voltage References (FVR):
  - Selectable 1.024V, 2.048V and 4.096V output levels
  - FVR1 internally connected to ADC
  - FVR2 internally connected to Comparator

## Clocking Structure

- High-Precision Internal Oscillator Block (HFINTOSC):
  - Selectable frequencies up to 32 MHz
  - ±2% at calibration
  - Active Clock Tuning of HFINTOSC for improved accuracy
- Internal 31 kHz Oscillator (LFINTOSC)
- External 32 kHz Secondary Oscillator (SOSC)
- External High-Frequency Clock Input:
  - Three Crystal/Resonator modes
  - Two External Clock (EC) Power modes
  - 4x PLL available for external sources
- Fail-Safe Clock Monitor:
  - Allows for operational recovery if the external clock source stops
- Oscillator Start-up Timer (OST):
  - Ensures the stability of crystal oscillator sources

## Programming/Debug Features

- In-Circuit Serial Programming™ (ICSP™) via Two Pins
- In-Circuit Debug (ICD) with Three Breakpoints via Two Pins
- Debug Integrated On-Chip

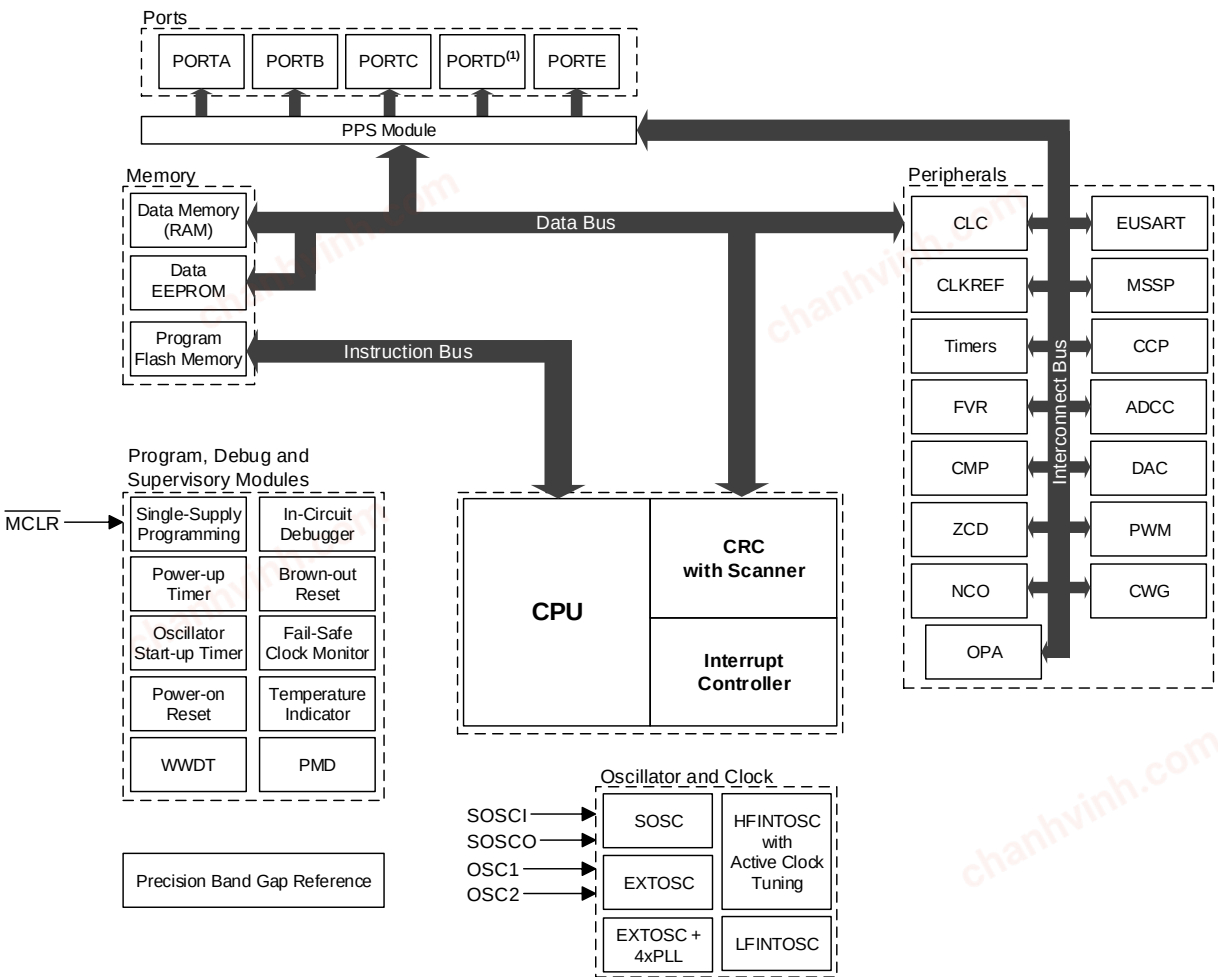
© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



## Block Diagram

**Figure 1. PIC16F17156/76 Block Diagram**




**Note:**

1. Available on 40/44-pin devices only.

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



# Table of Contents

Introduction....1

PIC16F171 Family Summary....1

Core Features....2

1. Packages....9
2. Pin Diagrams....10
3. Pin Allocation Tables....13
4. Guidelines for Getting Started with PIC16F171 Microcontrollers....19
5. Register and Bit Naming Conventions....22
6. Register Legend....24
7. Enhanced Mid-Range CPU....25
8. Device Configuration....27
9. Memory Organization....39
10. Resets....73
11. OSC - Oscillator Module (With Fail-Safe Clock Monitor)....85
12. INT - Interrupts....110
13. Power-Saving Modes....132
14. WWDT - Windowed Watchdog Timer....139
15. NVM - Nonvolatile Memory Control....150
16. I/O Ports....174
17. IOC - Interrupt-on-Change....190
18. PPS - Peripheral Pin Select Module....196
19. CRC - Cyclic Redundancy Check Module with Memory Scanner....206
20. PMD - Peripheral Module Disable....225
21. CLKREF - Reference Clock Output Module....232
22. TMR0 - Timer0 Module....237
23. TMR1 - Timer1 Module with Gate Control....245
24. TMR2 - Timer2 Module....260
25. NCO - Numerically Controlled Oscillator Module....281
26. CWG - Complementary Waveform Generator Module....290

© 2023 Microchip Technology Inc.
and its subsidiaries

chanhvinh.com



27. CCP - Capture/Compare/PWM Module ..... 317
28. Capture, Compare, and PWM Timers Selection ..... 330
29. PWM - Pulse-Width Modulator with Compare ..... 333
30. CLC - Configurable Logic Cell ..... 360
31. MSSP - Host Synchronous Serial Port Module ..... 380
32. EUSART - Enhanced Universal Synchronous Asynchronous Receiver Transmitter ..... 441
33. ADC - Analog-to-Digital Converter with Computation Module ..... 470
34. OPA - Operational Amplifier ..... 523
35. DAC - Digital-to-Analog Converter Module ..... 535
36. CMP - Comparator Module ..... 543
37. FVR - Fixed Voltage Reference ..... 554
38. Temperature Indicator Module ..... 558
39. ZCD - Zero-Cross Detection Module ..... 563
40. Charge Pump ..... 570
41. Instruction Set Summary ..... 574
42. ICSP™ - In-Circuit Serial Programming™ ..... 594
43. Register Summary ..... 597
44. Electrical Specifications ..... 608
45. DC and AC Characteristics Graphs and Tables ..... 638
46. Packaging Information ..... 639
47. Appendix A: Revision History ..... 659

Microchip Information ..... 660

chanhvinh.com

chanhvinh.com



# 1. Packages

**Table 1-1. Packages**

| Device | 28-Pin SPDIP | 28-Pin SOIC | 28-Pin SSOP | 28-Pin VQFN 4x4x1 | 40-Pin PDIP | 40-Pin VQFN 5x5x0.9 | 44-Pin TQFP 10x10x1 |
|---|---|---|---|---|---|---|---|
| PIC16F17156 | • | • | • | • | | | |
| PIC16F17176 | | | | | • | • | • |

chanhvinh.com

chanhvinh.com



# 2. Pin Diagrams

**Figure 2-1. 28-Pin SPDIP, SOIC, SSOP**

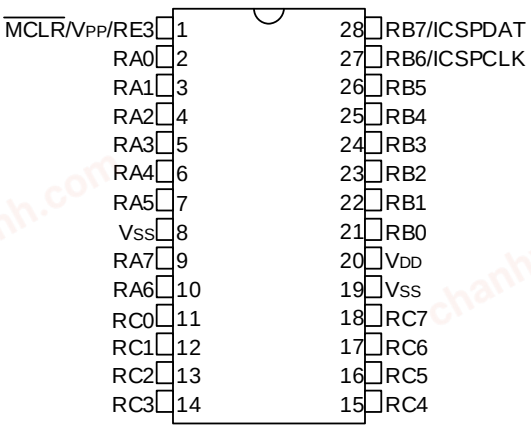
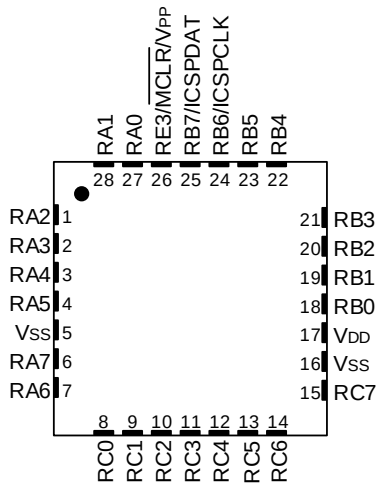



**Figure 2-2. 28-Pin VQFN**




© 2023 Microchip Technology Inc.
and its subsidiaries

chanhvinh.com



**Figure 2-3. 40-Pin PDIP**

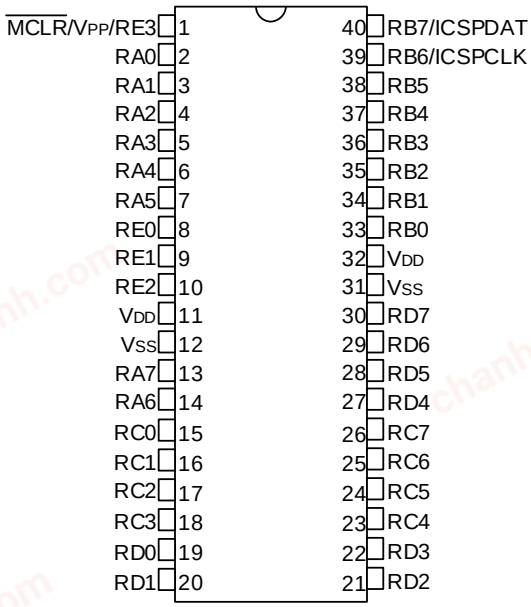
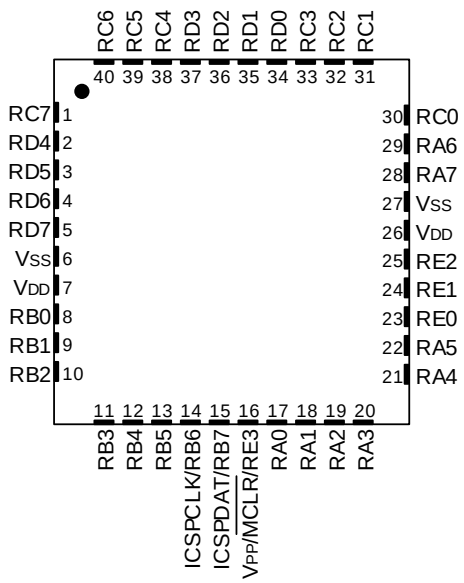


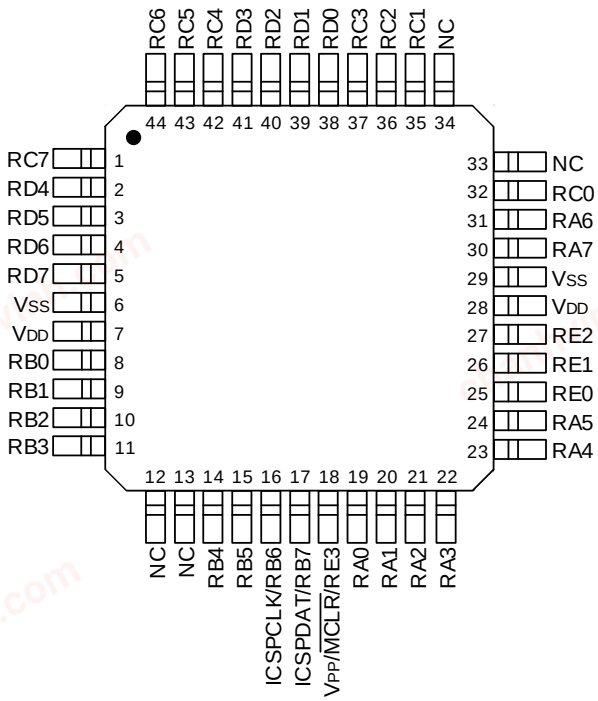

**Figure 2-4. 40-Pin VQFN**



chanhvinh.com



**Figure 2-5. 44-Pin TQFP**



chanhvinh.com



# 3. Pin Allocation Tables

**Table 3-1. 28-Pin Allocation Table**

| I/O | 28-Pin SPDIP SOIC SSOP | 28-Pin VQFN | ADC | DAC | Operational Amplifier | Comparator | ZCD | Timers | CCP | 16-Bit PWM | CWG | CLC | MSSP | EUSART | IOC | Interrupt | Basic |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| RA0 | 2 | 27 | ANA0 | — | OPA1(1) | C1IN0- C2IN0- | — | — | — | — | — | CLCIN0(1) | — | — | IOCA0 | — | — |
| RA1 | 3 | 28 | ANA1(5) | — | — | C1IN1- C2IN1- | — | — | — | — | — | CLCIN1(1) | — | — | IOCA1 | — | — |
| RA2 | 4 | 1 | ANA2 VREF- (ADC) | DAC1REF0- | OPA1IN0+ OPA1IN0- | C1IN0+ C2IN0+ | — | — | — | — | — | — | — | — | IOCA2 | — | — |
| RA3 | 5 | 2 | ANA3(5) VREF+ (ADC) | DAC1REF0+ | — | C1IN1+ | — | — | — | — | — | — | — | — | IOCA3 | — | — |
| RA4 | 6 | 3 | ANA4 | — | OPA1IN1+ OPA1IN1- | — | — | T0CKI(1) | — | — | — | — | — | — | IOCA4 | — | — |
| RA5 | 7 | 4 | ANA5(5) | — | OPA1IN2+ OPA1IN2- | — | — | — | — | — | — | — | $\overline{SS1}$(1) | — | IOCA5 | — | — |
| RA6 | 10 | 7 | ANA6(5) | — | — | — | — | — | — | — | — | — | — | — | IOCA6 | — | CLKOUT OSC2 |
| RA7 | 9 | 6 | ANA7 | — | — | — | — | — | — | — | — | — | — | — | IOCA7 | — | CLKIN OSC1 |
| RB0 | 21 | 18 | ANB0(5) | DAC2REF0+ | — | C2IN1+ | ZCD1 | — | — | — | CWG1(1) | — | $\overline{SS2}$(1) | — | IOCB0 | INT(1) | — |
| RB1 | 22 | 19 | ANB1 | — | OPA1IN3+ OPA1IN3- | C1IN3- C2IN3- | — | — | — | — | — | — | SCL2(1,3,4) SCK2(1,3,4) | — | IOCB1 | — | — |
| RB2 | 23 | 20 | ANB2(5) | — | — | — | — | — | — | — | — | — | SDA2(1,3,4) SDI2(1,3,4) | — | IOCB2 | — | — |
| RB3 | 24 | 21 | ANB3 | — | — | C1IN2- C2IN2- | — | — | — | — | — | — | — | — | IOCB3 | — | — |
| RB4 | 25 | 22 | ANB4(5) ADACT(1) | — | — | — | — | — | — | — | — | — | — | — | IOCB4 | — | — |
| RB5 | 26 | 23 | ANB5 | DAC2REF0- | — | — | — | T1G(1) | — | — | — | — | — | — | IOCB5 | — | — |
| RB6 | 27 | 24 | ANB6(5) | — | — | — | — | — | — | — | — | CLCIN2(1) | — | CK2(1,3) | IOCB6 | — | ICSPCLK ICDCLK |

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



.....continued

| I/O | 28-Pin SPDIP SOIC SSOP | 28-Pin VQFN | ADC | DAC | Operational Amplifier | Comparator | ZCD | Timers | CCP | 16-Bit PWM | CWG | CLC | MSSP | EUSART | IOC | Interrupt | Basic |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| RB7 | 28 | 25 | ANB7 | — | — | — | — | T6IN(1) | — | PWM3ERS(1) | — | CLCIN3(1) | — | RX2(1) DT2(1,3) | IOCB7 | — | ICSPDAT ICDDAT |
| RC0 | 11 | 8 | ANC0 | — | — | — | — | T1CKI(1) T3CKI(1) T3G(1) | — | — | — | — | — | — | IOCC0 | — | SOSCO |
| RC1 | 12 | 9 | ANC1(5) | — | — | — | — | — | CCP2(1) | PWMIN1(1) | — | — | — | — | IOCC1 | — | SOSCI |
| RC2 | 13 | 10 | ANC2 | — | — | — | — | — | CCP1(1) | PWMIN0(1) | — | — | — | — | IOCC2 | — | — |
| RC3 | 14 | 11 | ANC3(5) | — | — | — | — | T2IN(1) | — | PWM1ERS(1) PWM4ERS(1) | — | — | SCL1(1,3,4) SCK1(1,3,4) | — | IOCC3 | — | — |
| RC4 | 15 | 12 | ANC4 | — | — | — | — | — | — | — | — | — | SDA1(1,3,4) SDI1(1,3,4) | — | IOCC4 | — | — |
| RC5 | 16 | 13 | ANC5 | — | — | — | — | T4IN(1) | — | PWM2ERS(1) | — | — | — | — | IOCC5 | — | — |
| RC6 | 17 | 14 | ANC6(5) | — | — | — | — | — | — | — | — | — | — | CK1(1,3) | IOCC6 | — | — |
| RC7 | 18 | 15 | ANC7 | — | — | — | — | — | — | — | — | — | — | RX1(1) DT1(1,3) | IOCC7 | — | — |
| RE3 | 1 | 26 | — | — | — | — | — | — | — | — | — | — | — | — | IOCE3 | — | $\overline{MCLR}$ $V_{PP}$ |
| $V_{DD}$ | 20 | 17 | — | — | — | — | — | — | — | — | — | — | — | — | — | — | $V_{DD}$ |
| $V_{SS}$ | 8 19 | 5 16 | — | — | — | — | — | — | — | — | — | — | — | — | — | — | $V_{SS}$ |
| OUT(2) | — | — | ADGRDA ADGRDB | — | OPA1OUT | CMP1 CMP2 | — | TMR0 | CCP1 CCP2 | PWM11 PWM12 PWM21 PWM22 PWM31 PWM32 PWM41 PWM42 | CWG1A CWG1B CWG1C CWG1D | CLC1OUT CLC2OUT CLC3OUT CLC4OUT | SCL1 SCK1 SDA1 SDO1 SCL2 SCK2 SDA2 SDO2 | TX1 DT1 CK1 TX2 DT2 CK2 | — | — | — |

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



.....continued

| I/O | 28-Pin SPDIP SOIC SSOP | 28-Pin VQFN | ADC | DAC | Operational Amplifier | Comparator | ZCD | Timers | CCP | 16-Bit PWM | CWG | CLC | MSSP | EUSART | IOC | Interrupt | Basic |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|

**Notes:**

1. This is a PPS remappable input signal. The input function may be moved from the default location shown to one of several other PORTx pins. Refer to the PPS input table in the device data sheet for details on which PORT pins may be used for this signal.
2. All output signals shown in this row are PPS remappable.
3. This is a bidirectional signal. For normal operation, user software must map this signal to the same pin via the PPS input and PPS output registers.
4. These pins can be configured for $I^2C$ or SMBus logic levels via the RxyI2C registers. The SCL1/SDA1 signals may be assigned to these pins for expected operation. PPS assignments of these signals to other pins will operate; however, the logic levels will be standard TTL/ST as selected by the INLVL register.
5. This pin can be used as either a positive or negative analog input channel to the ADC.

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



# 3. Pin Allocation Tables

**Table 3-2. 40/44-Pin Allocation Table**

| I/O | 40-Pin PDIP | 40-Pin VQFN | 44-Pin TQFP | ADC | DAC | Operational Amplifier | Comparator | ZCD | Timers | CCP | 16-Bit PWM | CWG | CLC | MSSP | EUSART | IOC | Interrupt | Basic |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| RA0 | 2 | 17 | 19 | ANA0 | — | OPA1(1) | C1IN0-<br>C2IN0- | — | — | — | — | — | CLCIN0(1) | — | — | IOCA0 | — | — |
| RA1 | 3 | 18 | 20 | ANA1(5) | — | — | C1IN1-<br>C2IN1- | — | — | — | — | — | CLCIN1(1) | — | — | IOCA1 | — | — |
| RA2 | 4 | 19 | 21 | ANA2<br>VREF- (ADC) | DAC1REF0- | OPA1IN0+<br>OPA1IN0- | C1IN0+<br>C2IN0+ | — | — | — | — | — | — | — | — | IOCA2 | — | — |
| RA3 | 5 | 20 | 22 | ANA3(5)<br>VREF+ (ADC) | DAC1REF0+ | — | C1IN1+ | — | — | — | — | — | — | — | — | IOCA3 | — | — |
| RA4 | 6 | 21 | 23 | ANA4 | — | OPA1IN1+<br>OPA1IN1- | — | — | T0CKI(1) | — | — | — | — | — | — | IOCA4 | — | — |
| RA5 | 7 | 22 | 24 | ANA5(5) | — | OPA1IN2+<br>OPA1IN2- | — | — | — | — | — | — | — | $\overline{SS1}$(1) | — | IOCA5 | — | — |
| RA6 | 14 | 29 | 31 | ANA6(5) | — | — | — | — | — | — | — | — | — | — | — | IOCA6 | — | CLKOUT<br>OSC2 |
| RA7 | 13 | 28 | 30 | ANA7 | — | — | — | — | — | — | — | — | — | — | — | IOCA7 | — | CLKIN<br>OSC1 |
| RB0 | 33 | 8 | 8 | ANB0(5) | DAC2REF0+ | — | C2IN1+ | ZCD1 | — | — | — | CWG1(1) | — | $\overline{SS2}$(1) | — | IOCB0 | INT(1) | — |
| RB1 | 34 | 9 | 9 | ANB1 | — | OPA1IN3+<br>OPA1IN3- | C1IN3-<br>C2IN3- | — | — | — | — | — | — | SCL2(1,3,4)<br>SCK2(1,3,4) | — | IOCB1 | — | — |
| RB2 | 35 | 10 | 10 | ANB2(5) | — | — | — | — | — | — | — | — | — | SDA2(1,3,4)<br>SDI2(1,3,4) | — | IOCB2 | — | — |
| RB3 | 36 | 11 | 11 | ANB3 | — | — | C1IN2-<br>C2IN2- | — | — | — | — | — | — | — | — | IOCB3 | — | — |
| RB4 | 37 | 12 | 14 | ANB4(5)<br>ADACT(1) | — | — | — | — | — | — | — | — | — | — | — | IOCB4 | — | — |
| RB5 | 38 | 13 | 15 | ANB5 | DAC2REF0- | — | — | — | T1G(1) | — | — | — | — | — | — | IOCB5 | — | — |
| RB6 | 39 | 14 | 16 | ANB6(5) | — | — | — | — | — | — | — | — | CLCIN2(1) | — | CK2(1,3) | IOCB6 | — | ICSPCLK<br>ICDCLK |
| RB7 | 40 | 15 | 17 | ANB7 | — | — | — | — | T6IN(1) | — | PWM3ERS(1) | — | CLCIN3(1) | — | RX2(1)<br>DT2(1,3) | IOCB7 | — | ICSPDAT<br>ICDDAT |

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



.....continued

| I/O | 40-Pin PDIP | 40-Pin VQFN | 44-Pin TQFP | ADC | DAC | Operational Amplifier | Comparator | ZCD | Timers | CCP | 16-Bit PWM | CWG | CLC | MSSP | EUSART | IOC | Interrupt | Basic |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| RC0 | 15 | 30 | 32 | ANC0 | — | — | — | — | T1CKI(1)<br>T3CKI(1)<br>T3G(1) | — | — | — | — | — | — | IOCC0 | — | SOSCO |
| RC1 | 16 | 31 | 35 | ANC1(5) | — | — | — | — | — | CCP2(1) | PWMIN1(1) | — | — | — | — | IOCC1 | — | SOSCI |
| RC2 | 17 | 32 | 36 | ANC2 | — | — | — | — | — | CCP1(1) | PWMIN0(1) | — | — | — | — | IOCC2 | — | — |
| RC3 | 18 | 33 | 37 | ANC3(5) | — | — | — | — | T2IN(1) | — | PWM1ERS(1)<br>PWM4ERS(1) | — | — | SCL1(1,3,4)<br>SCK1(1,3,4) | — | IOCC3 | — | — |
| RC4 | 23 | 38 | 42 | ANC4 | — | — | — | — | — | — | — | — | — | SDA1(1,3,4)<br>SDI1(1,3,4) | — | IOCC4 | — | — |
| RC5 | 24 | 39 | 43 | ANC5 | — | — | — | — | T4IN(1) | — | PWM2ERS(1) | — | — | — | — | IOCC5 | — | — |
| RC6 | 25 | 40 | 44 | ANC6(5) | — | — | — | — | — | — | — | — | — | — | CK1(1,3) | IOCC6 | — | — |
| RC7 | 26 | 1 | 1 | ANC7 | — | — | — | — | — | — | — | — | — | — | RX1(1)<br>DT1(1,3) | IOCC7 | — | — |
| RD0 | 19 | 34 | 38 | AND0 | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| RD1 | 20 | 35 | 39 | AND1 | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| RD2 | 21 | 36 | 40 | AND2(5) | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| RD3 | 22 | 37 | 41 | AND3(5) | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| RD4 | 27 | 2 | 2 | AND4 | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| RD5 | 28 | 3 | 3 | AND5(5) | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| RD6 | 29 | 4 | 4 | AND6 | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| RD7 | 30 | 5 | 5 | AND7(5) | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| RE0 | 8 | 23 | 25 | ANE0 | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| RE1 | 9 | 24 | 26 | ANE1(5) | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| RE2 | 10 | 25 | 27 | ANE2 | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| RE3 | 1 | 16 | 18 | — | — | — | — | — | — | — | — | — | — | — | — | IOCE3 | — | $\overline{MCLR}$<br>VPP |
| VDD | 11<br>32 | 7<br>26 | 7<br>28 | — | — | — | — | — | — | — | — | — | — | — | — | — | — | VDD |
| VSS | 12<br>31 | 6<br>27 | 6<br>29 | — | — | — | — | — | — | — | — | — | — | — | — | — | — | VSS |

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



.....continued

| I/O | 40-Pin PDIP | 40-Pin VQFN | 44-Pin TQFP | ADC | DAC | Operational Amplifier | Comparator | ZCD | Timers | CCP | 16-Bit PWM | CWG | CLC | MSSP | EUSART | IOC | Interrupt | Basic |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| OUT(2) | — | — | — | ADGRDA<br>ADGRDB | — | OPA1OUT | CMP1<br>CMP2 | — | TMR0 | CCP1<br>CCP2 | PWM11<br>PWM12<br>PWM21<br>PWM22<br>PWM31<br>PWM32<br>PWM41<br>PWM42 | CWG1A<br>CWG1B<br>CWG1C<br>CWG1D | CLC1OUT<br>CLC2OUT<br>CLC3OUT<br>CLC4OUT | SCL1<br>SCK1<br>SDA1<br>SDO1<br>SCL2<br>SCK2<br>SDA2<br>SDO2 | TX1<br>DT1<br>CK1<br>TX2<br>DT2<br>CK2 | — | — | — |

**Notes:**

1. This is a PPS remappable input signal. The input function may be moved from the default location shown to one of several other PORTx pins. Refer to the PPS input table in the device data sheet for details on which PORT pins may be used for this signal.
2. All output signals shown in this row are PPS remappable.
3. This is a bidirectional signal. For normal operation, user software must map this signal to the same pin via the PPS input and PPS output registers.
4. These pins can be configured for $I^2C$ or SMBus logic levels via the RxyI2C registers. The SCL1/SDA1 signals may be assigned to these pins for expected operation. PPS assignments of these signals to other pins will operate; however, the logic levels will be standard TTL/ST as selected by the INLVL register.
5. This pin can be used as either a positive or negative analog input channel to the ADC.

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



# 4. Guidelines for Getting Started with PIC16F171 Microcontrollers

## 4.1 Basic Connection Requirements

Getting started with the PIC16F171 family of 8-bit microcontrollers requires attention to a minimal set of device pin connections before proceeding with development.

The following pins must always be connected:

- All $V_{DD}$ and $V_{SS}$ pins (see 4.2. Power Supply Pins)
- $\overline{MCLR}$ pin (see 4.3. Master Clear (MCLR) Pin)

These pins must also be connected if they are being used in the end application:

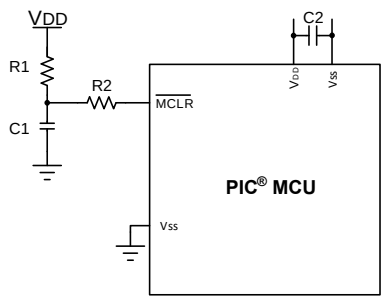
- PGC/PGD pins used for In-Circuit Serial Programming™ (ICSP™) and debugging purposes (see 4.4. In-Circuit Serial Programming (ICSP) Pins)
- CLKIN pin when an external clock source is used.

Additionally, the following may be required:

- $V_{REF}+/V_{REF}-$ pins are used when external voltage reference for analog modules is implemented

The minimum recommended connections are shown in the figure below.

**Figure 4-1. Minimum Recommended Connections**




**Key (all values are recommendations):**
C1: 10 nF, 16V ceramic
C2: 0.1 µF, 16V ceramic
R1: 10 kΩ
R2: 100Ω to 470Ω

## 4.2 Power Supply Pins

### 4.2.1 Decoupling Capacitors

The use of decoupling capacitors on every pair of power supply pins ($V_{DD}$ and $V_{SS}$) is required.

Consider the following criteria when using decoupling capacitors:

- Value and type of capacitor: A 0.1 µF (100 nF), 10-25V capacitor is recommended. The capacitor may be a low-ESR device, with a resonance frequency in the range of 200 MHz and higher. Ceramic capacitors are recommended.
- Placement on the printed circuit board: The decoupling capacitors may be placed as close to the pins as possible. It is recommended to place the capacitors on the same side of the board as the device. If space is constricted, the capacitor can be placed on another layer on the PCB using a via; however, ensure that the trace length from the pin to the capacitor is no greater than 0.25 inch (6 mm).

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



- Handling high-frequency noise: If the board is experiencing high-frequency noise (upward of tens of MHz), add a second ceramic type capacitor in parallel to the above described decoupling capacitor. The value of the second capacitor can be in the range of 0.01 µF to 0.001 µF. Place this second capacitor next to each primary decoupling capacitor. In high-speed circuit designs, consider implementing a decade pair of capacitances as close to the power and ground pins as possible (e.g., 0.1 µF in parallel with 0.001 µF).
- Maximizing performance: On the board layout from the power supply circuit, run the power and return traces to the decoupling capacitors first, and then to the device pins. This ensures that the decoupling capacitors are first in the power chain. Equally important is to keep the trace length between the capacitor and the power pins to a minimum, thereby reducing PCB trace inductance.

### 4.2.2 Tank Capacitors

With on boards with power traces running longer than six inches in length, it is suggested to use a tank capacitor for integrated circuits, including microcontrollers, to supply a local power source. The value of the tank capacitor may be determined based on the trace resistance that connects the power supply source to the device, and the maximum current drawn by the device in the application. In other words, select the tank capacitor that meets the acceptable voltage sag at the device. Typical values range from 4.7 µF to 47 µF.

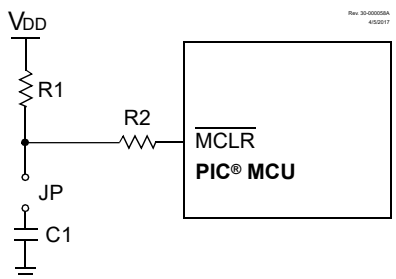
## 4.3 Master Clear ($\overline{MCLR}$) Pin

The $\overline{MCLR}$ pin provides two specific device functions: Device Reset, and device programming and debugging. If programming and debugging are not required in the end application, a direct connection to $V_{DD}$ may be all that is required. The addition of other components, to help increase the application's resistance to spurious Resets from voltage sags, may be beneficial. A typical configuration is shown in Figure 4-1. Other circuit designs may be implemented, depending on the application's requirements.

During programming and debugging, the resistance and capacitance that can be added to the pin must be considered. Device programmers and debuggers drive the $\overline{MCLR}$ pin. Consequently, specific voltage levels ($V_{IH}$ and $V_{IL}$) and fast signal transitions must not be adversely affected. Therefore, specific values of R1 and C1 will need to be adjusted based on the application and PCB requirements. For example, it is recommended that the capacitor, C1, be isolated from the $\overline{MCLR}$ pin during programming and debugging operations by using a jumper (Figure 4-2). The jumper is replaced for normal run-time operations.

Any components associated with the $\overline{MCLR}$ pin may be placed within 0.25 inch (6 mm) of the pin.

**Figure 4-2. Example of $\overline{MCLR}$ Pin Connections**




**Notes:**

1. R1 ≤ 10 kΩ is recommended. A suggested starting value is 10 kΩ. Ensure that the $\overline{MCLR}$ pin $V_{IH}$ and $V_{IL}$ specifications are met.
2. R2 ≤ 470Ω will limit any current flowing into $\overline{MCLR}$ from the extended capacitor, C1, in the event of $\overline{MCLR}$ pin breakdown, due to Electrostatic Discharge (ESD) or Electrical Overstress (EOS). Ensure that the $\overline{MCLR}$ pin $V_{IH}$ and $V_{IL}$ specifications are met.

## 4.4 In-Circuit Serial Programming™ (ICSP™) Pins

The ICSPCLK and ICSPDAT pins are used for ICSP and debugging purposes. It is recommended to keep the trace length between the ICSP connector and the ICSP pins on the device as short as possible. If the ICSP connector is expected to experience an ESD event, a series resistor is recommended, with the value in the range of a few tens of ohms, not to exceed 100Ω.

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



Pull-up resistors, series diodes and capacitors on the ICSPCLK and ICSPDAT pins are not recommended as they can interfere with the programmer/debugger communications to the device. If such discrete components are an application requirement, they may be removed from the circuit during programming and debugging. Alternatively, refer to the AC/DC characteristics and timing requirements information in the respective device Flash programming specification for information on capacitive loading limits, and pin input voltage high ($V_{IH}$) and input low ($V_{IL}$) requirements.

For device emulation, ensure that the Communication Channel Select (i.e., ICSPCLK/ICSPDAT pins), programmed into the device, matches the physical connections for the ICSP to the Microchip debugger/emulator tool.

## 4.5 Unused I/Os

Unused I/O pins may be configured as outputs and driven to a Logic Low state. Alternatively, connect a 1 kΩ to 10 kΩ resistor to $V_{SS}$ on unused pins to drive the output to Logic Low.

© 2023 Microchip Technology Inc.
and its subsidiaries

chanhvinh.com



# 5. Register and Bit Naming Conventions

## 5.1 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.

## 5.2 Bit Names

There are two variants for bit names:

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

### 5.2.1 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 RegisterNamebits.ShortName. For example, the enable bit, ON, in the ADCON0 register can be set in C programs with the instruction `ADCON0bits.ON = 1`.

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

### 5.2.2 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 ADC enable bit is the ADC prefix, AD, appended with the enable bit short name, ON, resulting in the unique bit name ADON.

Long bit names are useful in both C and assembly programs. For example, in C the ADCON0 enable bit can be set with the `ADON = 1` instruction. In assembly, this bit can be set with the `BSF ADCON0,ADON` instruction.

### 5.2.3 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 ADCON2 register contain the ADC Operating Mode Selection bit. The short name for this field is MD and the long name is ADMD. Bit field access is only possible in C programs. The following example demonstrates a C program instruction for setting the ADC to operate in Accumulate mode:

```
ADCON2bits.MD = 0b001;
```

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 ADMD2. The following two examples demonstrate assembly program sequences for setting the ADC to operate in Accumulate mode:

```
MOVLW  ~(1<<MD2 | 1<<MD1)
ANDWF  ADCON2,F
MOVLW  1<<MD0
IORWF  ADCON2,F
```

© 2023 Microchip Technology Inc.
and its subsidiaries

chanhvinh.com



```
BCF     ADCON2,ADMD2
BCF     ADCON2,ADMD1
BSF     ADCON2,ADMD0
```

## 5.3 Register and Bit Naming Exceptions

### 5.3.1 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.

chanhvinh.com

chanhvinh.com



# 6. Register Legend

**Table 6-1. Register Legend**

| Symbol | Definition |
|---|---|
| R | Readable bit |
| W | Writable bit |
| HS | Hardware settable bit |
| HC | Hardware clearable bit |
| S | Set only bit |
| C | Clear only bit |
| U | Unimplemented bit, read as '0' |
| '1' | Bit value is set |
| '0' | Bit value is cleared |
| x | Bit value is unknown |
| u | Bit value is unchanged |
| q | Bit value depends on condition |
| m | Bit value is predefined |

chanhvinh.com

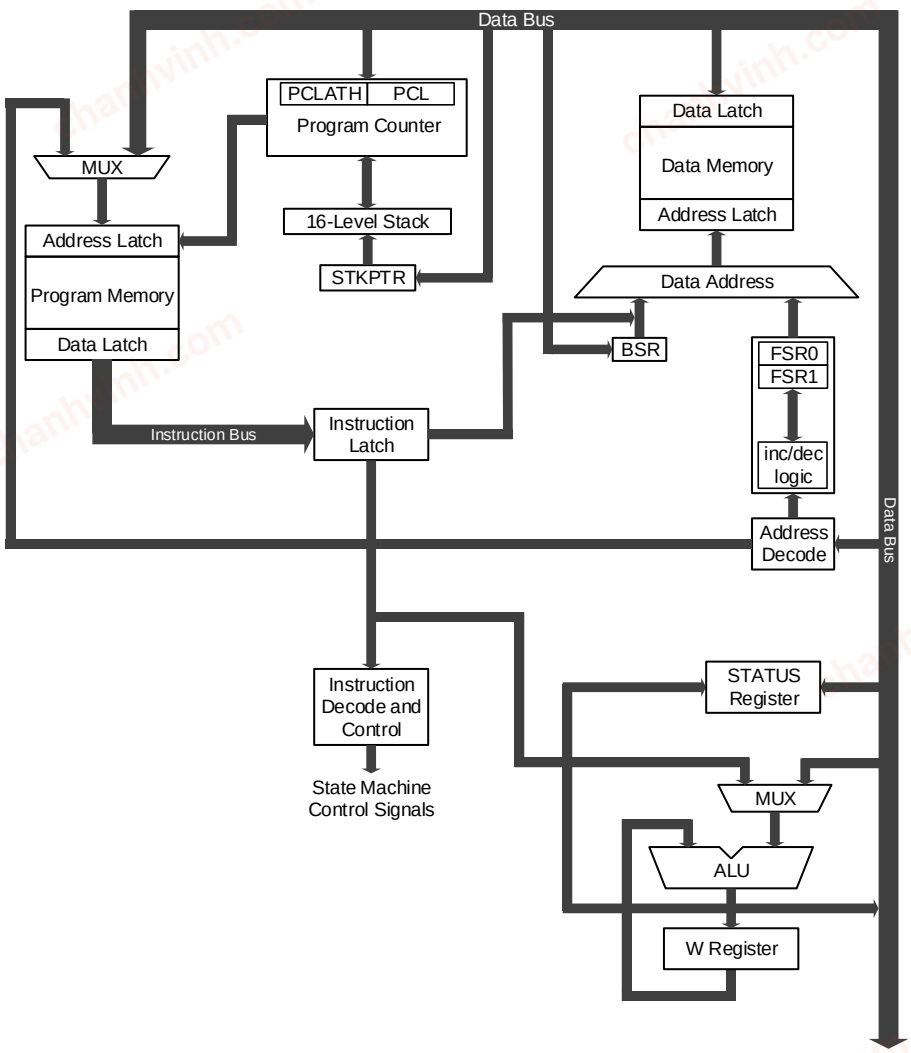
chanhvinh.com



# 7. Enhanced Mid-Range CPU

This family of devices contains an enhanced mid-range 8-bit CPU core. The CPU has 50 instructions. Interrupt capability includes automatic context saving. The hardware stack is 16-level deep and has overflow and underflow Reset capability. Direct, Indirect, and Relative Addressing modes are available. Two File Select Registers (FSR) provide the ability to read program and data memory.

**Figure 7-1. Core Data Path Diagram**




## 7.1 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.

chanhvinh.com



## 7.2 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), and if enabled, will cause a software Reset.

## 7.3 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 also be addressed linearly, providing the ability to access contiguous data larger than 80 bytes.

## 7.4 Instruction Set

There are 50 instructions for the enhanced mid-range CPU to support the features of the CPU. See the **"Instruction Set Summary"** section for more details.

chanhvinh.com

chanhvinh.com



# 8. Device Configuration

Device configuration consists of the Configuration Words, User ID, Device ID, Device Information Area (DIA) and the Device Configuration Information (DCI) regions.

## 8.1 Configuration Words

There are five Configuration Words that allow the user to select the device oscillator, Reset and memory protection options. These are implemented at addresses 0x8007 - 0x800B.

**Note:** The DEBUG bit in the Configuration Words is managed automatically by device development tools, including debuggers and programmers. For normal device operation, this bit needs to be maintained as a '1'.

## 8.2 Code Protection

Program memory code protection is controlled using the $\overline{CP}$ bit, while data EEPROM memory code protection is controlled using the $\overline{CPD}$ bit. When code protection is enabled, all program memory and Data EEPROM locations read as '0'. Further programming is disabled for the program memory and Data EEPROM, until a Bulk Erase operation is performed on the configuration memory region. Program memory and Data EEPROM can still be programmed and read during program execution.

The User ID locations and Configuration Bytes can be programmed and read out regardless of the code protection settings.

The only way to disable code protection is to use the Bulk Erase Program Memory command with bit 4 of the payload set to '1'. This will disable code protection and erase all memory locations.

## 8.3 Write Protection

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

The $\overline{WRTn}$ Configuration bits determine which of the program memory blocks are protected.

## 8.4 User ID

Four words in the memory space (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 the **"NVMREG Access to DIA, DCI, User ID, DEV/REV ID, and Configuration Words"** section for more information on accessing these memory locations. For more information on checksum calculation, see the **"Memory Programming Specification"** section in the **"Electrical Specifications"** chapter for more information on accessing these memory locations. For more information on checksum calculation, see the **"Family Programming Specification"** section.

## 8.5 Device ID and Revision ID

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

Development tools, such as device programmers and debuggers, may be used to read the Device ID, Revision ID and Configuration Words. Refer to the "**NVM - Nonvolatile Memory Control**" section for more information on accessing these locations.

## 8.6 Register Definitions: Configuration Settings

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



### 8.6.1 CONFIG1

**Name:** CONFIG1
**Offset:** 0x8007

Configuration Word 1

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | FCMEN | VDDAR | CSWEN | | | $\overline{\text{CLKOUTEN}}$ |
| Access | | | R/W | R/W | R/W | | | R/W |
| Reset | | | 1 | 1 | 1 | | | 1 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | RSTOSC[2:0] | | | | FEXTOSC[2:0] | | |
| Access | | R/W | R/W | R/W | | R/W | R/W | R/W |
| Reset | | 1 | 1 | 1 | | 1 | 1 | 1 |

**Bit 13 – FCMEN** Fail-Safe Clock Monitor Enable

| Value | Description |
|---|---|
| 1 | Fail-Safe Clock Monitor is enabled |
| 0 | Fail-Safe Clock Monitor is disabled |

**Bit 12 – VDDAR** $V_{DD}$ Analog Range Calibration Selection

| Value | Description |
|---|---|
| 1 | Internal analog systems are calibrated for operation between $V_{DD}$ = 2.3V - 5.5V |
| 0 | Internal analog systems are calibrated for operation between $V_{DD}$ = 1.8V - 3.6V |

**Bit 11 – CSWEN** Clock Switch Enable

| Value | Description |
|---|---|
| 1 | Writing to NOSC and NDIV is allowed |
| 0 | The NOSC and NDIV bit fields cannot be changed by user software |

**Bit 8 – $\overline{\text{CLKOUTEN}}$** Clock Out Enable

| Value | Description |
|---|---|
| 1 | CLKOUT function is disabled; I/O function on CLKOUT pin |
| 0 | CLKOUT function is enabled; $F_{OSC}$/4 clock appears on CLKOUT pin |

**Bits 6:4 – RSTOSC[2:0]** Power-up Default Value for the NOSC/COSC bits
Selects the oscillator source used by user software.

| Value | Description |
|---|---|
| 111 | EXTOSC operating per the FEXTOSC bits |
| 110 | HFINTOSC = 1 MHz (FRQ = 4 MHz, CDIV = 4:1) |
| 101 | LFINTOSC |
| 100 | SOSC |
| 011 | Reserved |
| 010 | EXTOSC with 4x PLL, EXTOSC operating per FEXTOSC bits |
| 001 | HFINTOSC = 16 MHz (FRQ = 16 MHz, CDIV = 1:1) |
| 000 | HFINTOSC = 32 MHz (FRQ = 32 MHz, CDIV = 1:1) |

**Bits 2:0 – FEXTOSC[2:0]** External Oscillator Mode Selection

| Value | Description |
|---|---|
| 111 | ECH (16 MHz and higher) |
| 110 | Reserved |
| 101 | ECL (below 16 MHz) |
| 100 | Oscillator not enabled |
| 011 | Reserved |

© 2023 Microchip Technology Inc.
and its subsidiaries

chanhvinh.com



| Value | Description |
|---|---|
| 010 | HS (crystal oscillator) above 4 MHz |
| 001 | XT (crystal oscillator) between 100 kHz and 4 MHz |
| 000 | LP (crystal oscillator) 32 kHz |

chanhvinh.com

chanhvinh.com



### 8.6.2 CONFIG2

**Name:** CONFIG2
**Offset:** 0x8008

Configuration Word 2

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | $\overline{\text{DEBUG}}$ | STVREN | PPS1WAY | $\overline{\text{ZCD}}$ | BORV | $\overline{\text{DACAUTOEN}}$ |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | BOREN[1:0] | | $\overline{\text{LPBOREN}}$ | | | PWRTS[1:0] | | MCLRE |
| Access | R/W | R/W | R/W | | | R/W | R/W | R/W |
| Reset | 1 | 1 | 1 | | | 1 | 1 | 1 |

**Bit 13 – $\overline{\text{DEBUG}}$** Debugger Enable(1)

| Value | Description |
|---|---|
| 1 | Background debugger disabled |
| 0 | Background debugger enabled |

**Bit 12 – STVREN** Stack Overflow/Underflow Reset Enable

| Value | Description |
|---|---|
| 1 | Stack Overflow or Underflow will cause a Reset |
| 0 | Stack Overflow or Underflow will not cause a Reset |

**Bit 11 – PPS1WAY** PPSLOCKED One-Way Set Enable

| Value | Description |
|---|---|
| 1 | The PPSLOCKED bit can only be set once after an unlocking sequence is executed; once PPSLOCKED is set, all future changes to PPS registers are prevented |
| 0 | The PPSLOCKED bit can be set and cleared as needed (unlocking sequence is required) |

**Bit 10 – $\overline{\text{ZCD}}$** Zero-Cross Detect Disable

| Value | Description |
|---|---|
| 1 | ZCD disabled, ZCD can be enabled by setting the ZCDSEN bit of ZCDCON |
| 0 | ZCD always enabled, the PMDx [ZCDMD] bit is ignored |

**Bit 9 – BORV** Brown-out Reset (BOR) Voltage Selection(2)

| Value | Description |
|---|---|
| 1 | Brown-out Reset voltage ($V_{BOR}$) set to 1.9V |
| 0 | Brown-out Reset voltage ($V_{BOR}$) set to 2.65V |

**Bit 8 – $\overline{\text{DACAUTOEN}}$** DAC Buffer Automatic Range Select Enable(3)

| Value | Description |
|---|---|
| 1 | DAC Buffer reference range is determined by the REFRNG bit of DACxCON |
| 0 | DAC Buffer reference range is automatically determined by module hardware |

**Bits 7:6 – BOREN[1:0]** Brown-out Reset (BOR) Enable(4)

| Value | Description |
|---|---|
| 11 | Brown-out Reset enabled, the SBOREN bit is ignored |
| 10 | Brown-out Reset enabled while running, disabled in Sleep; the SBOREN bit is ignored |
| 01 | Brown-out Reset enabled according to SBOREN |
| 00 | Brown-out Reset disabled |

**Bit 5 – $\overline{\text{LPBOREN}}$** Low-Power BOR Enable

chanhvinh.com



| Value | Description |
|---|---|
| 1 | Low-Power BOR disabled |
| 0 | Low-Power BOR enabled |

**Bits 2:1 – PWRTS[1:0]** Power-Up Timer (PWRT) Selection

| Value | Description |
|---|---|
| 11 | PWRT disabled |
| 10 | PWRT is set at 64 ms |
| 01 | PWRT is set at 16 ms |
| 00 | PWRT is set at 1 ms |

**Bit 0 – MCLRE** Master Clear ($\overline{\text{MCLR}}$) Enable

| Value | Condition | Description |
|---|---|---|
| x | If LVP = 1 | $\overline{\text{MCLR}}$ pin is $\overline{\text{MCLR}}$ |
| 1 | If LVP = 0 | $\overline{\text{MCLR}}$ pin is $\overline{\text{MCLR}}$ |
| 0 | If LVP = 0 | $\overline{\text{MCLR}}$ pin function is port-defined function |

**Notes:**

1. The $\overline{\text{DEBUG}}$ bit is managed automatically by device development tools including debuggers and programmers. For normal device operation, this bit needs to be maintained as a '1'.
2. The higher voltage selection is recommended for operation at or above 16 MHz.
3. The REFRNG bit is not available on the internal-only DAC (DAC2).
4. When enabled, Brown-out Reset voltage ($V_{BOR}$) is set by the BORV bit.

chanhvinh.com

chanhvinh.com



### 8.6.3 CONFIG3

**Name:** CONFIG3
**Offset:** 0x8009

Configuration Word 3

**Note:** This register is reserved.

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | WDTCCS[2:0] | | | WDTCWS[2:0] | | |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | WDTE[1:0] | | WDTCPS[4:0] | | | | |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

**Bits 13:11 – WDTCCS[2:0]** WDT Input Clock Selector

| Value | Condition | Description |
|---|---|---|
| x | WDTE = 00 | These bits have no effect |
| 111 | WDTE ≠ 00 | Software control |
| 110 to 011 | WDTE ≠ 00 | Reserved |
| 010 | WDTE ≠ 00 | WDT reference clock is the SOSC |
| 001 | WDTE ≠ 00 | WDT reference clock is the 31.25 kHz MFINTOSC |
| 000 | WDTE ≠ 00 | WDT reference clock is the 31.0 kHz LFINTOSC |

**Bits 10:8 – WDTCWS[2:0]** WDT Window Select

| WDTCWS | WDTCON1[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 | 110 | 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 | | |

**Bits 6:5 – WDTE[1:0]** WDT Operating Mode

| Value | Description |
|---|---|
| 11 | WDT enabled regardless of Sleep; the SEN bit in WDTCON0 is ignored |
| 10 | WDT enabled while Sleep = 0, suspended when Sleep = 1; the SEN bit in WDTCON0 is ignored |
| 01 | WDT enabled/disabled by the SEN bit in WDTCON0 |
| 00 | WDT disabled, the SEN bit in WDTCON0 is ignored |

**Bits 4:0 – WDTCPS[4:0]** WDT Period Select

| WDTCPS | WDTCON0[WDTPS] at POR | | | | Software Control of WDTPS? |
|---|---|---|---|---|---|
| | Value | Divider Ratio | | Typical Time-Out ($F_{IN}$ = 31 kHz) | |
| 11111 | 01011 | 1:65536 | $2^{16}$ | 2s | Yes |

chanhvinh.com



.....continued

| WDTCPS | WDTCON0[WDTPS] at POR | | | | Software Control of WDTPS? |
|---|---|---|---|---|---|
| | Value | Divider Ratio | | Typical Time-Out ($F_{IN}$ = 31 kHz) | |
| 11110 to 10011 | 11110 to 10011 | 1:32 | $2^5$ | 1 ms | No |
| 10010 | 10010 | 1:8388608 | $2^{23}$ | 256s | No |
| 10001 | 10001 | 1:4194304 | $2^{22}$ | 128s | No |
| 10000 | 10000 | 1:2097152 | $2^{21}$ | 64s | No |
| 01111 | 01111 | 1:1048576 | $2^{20}$ | 32s | No |
| 01110 | 01110 | 1:524288 | $2^{19}$ | 16s | No |
| 01101 | 01101 | 1:262144 | $2^{18}$ | 8s | No |
| 01100 | 01100 | 1:131072 | $2^{17}$ | 4s | No |
| 01011 | 01011 | 1:65536 | $2^{16}$ | 2s | No |
| 01010 | 01010 | 1:32768 | $2^{15}$ | 1s | No |
| 01001 | 01001 | 1:16384 | $2^{14}$ | 512 ms | No |
| 01000 | 01000 | 1:8192 | $2^{13}$ | 256 ms | No |
| 00111 | 00111 | 1:4096 | $2^{12}$ | 128 ms | No |
| 00110 | 00110 | 1:2048 | $2^{11}$ | 64 ms | No |
| 00101 | 00101 | 1:1024 | $2^{10}$ | 32 ms | No |
| 00100 | 00100 | 1:512 | $2^9$ | 16 ms | No |
| 00011 | 00011 | 1:256 | $2^8$ | 8 ms | No |
| 00010 | 00010 | 1:128 | $2^7$ | 4 ms | No |
| 00001 | 00001 | 1:64 | $2^6$ | 2 ms | No |
| 00000 | 00000 | 1:32 | $2^5$ | 1 ms | No |

© 2023 Microchip Technology Inc.
and its subsidiaries

chanhvinh.com



### 8.6.4 CONFIG4

**Name:** CONFIG4
**Offset:** 0x800A

Configuration Word 4

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | LVP | | WRTSAF | WRTD | WRTC | WRTB |
| Access | | | R/W | | R/W | R/W | R/W | R/W |
| Reset | | | 1 | | 1 | 1 | 1 | 1 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | WRTAPP | | | SAFEN | BBEN | BBSIZE[2:0] | | |
| Access | R/W | | | R/W | R/W | R/W | R/W | R/W |
| Reset | 1 | | | 1 | 1 | 1 | 1 | 1 |

**Bit 13 – LVP** Low-Voltage Programming Enable(1)

| Value | Description |
|---|---|
| 1 | Low-Voltage Programming is enabled. $\overline{MCLR}/V_{PP}$ pin function is $\overline{MCLR}$. The MCLRE bit is ignored. |
| 0 | High voltage (HV) on $\overline{MCLR}/V_{PP}$ must be used for programming |

**Bit 11 – WRTSAF** Storage Area Flash (SAF) Write Protection(2,3)

| Value | Description |
|---|---|
| 1 | SAF is not write-protected |
| 0 | SAF is write-protected |

**Bit 10 – WRTD** Data EEPROM Write Protection

| Value | Description |
|---|---|
| 1 | Data EEPROM is not write-protected |
| 0 | Data EEPROM is write-protected |

**Bit 9 – WRTC** Configuration Registers Write Protection(2)

| Value | Description |
|---|---|
| 1 | Configuration registers are not write-protected |
| 0 | Configuration registers are write-protected |

**Bit 8 – WRTB** Boot Block Write Protection(2,4)

| Value | Description |
|---|---|
| 1 | Boot Block is not write-protected |
| 0 | Boot Block is write-protected |

**Bit 7 – WRTAPP** Application Block Write Protection(2)

| Value | Description |
|---|---|
| 1 | Application Block is not write-protected |
| 0 | Application Block is write-protected |

**Bit 4 – SAFEN** Storage Area Flash (SAF) Enable(2)

| Value | Description |
|---|---|
| 1 | SAF is disabled |
| 0 | SAF is enabled |

**Bit 3 – BBEN** Boot Block Enable(2)

| Value | Description |
|---|---|
| 1 | Boot Block is disabled |
| 0 | Boot Block is enabled |

© 2023 Microchip Technology Inc.
and its subsidiaries

chanhvinh.com



**Bits 2:0 – BBSIZE[2:0]** Boot Block Size Selection(5,6)

**Table 8-1. Boot Block Size**

| $\overline{BBEN}$ | BBSIZE | End Address of Boot Block | Boot Block Size (words) |
|---|---|---|---|
| | | | PIC16F171x6 |
| 1 | xxx | – | – |
| 0 | 111 | 01FFh | 512 |
| 0 | 110 | 03FFh | 1024 |
| 0 | 101 | 07FFh | 2048 |
| 0 | 100 | 0FFFh | 4096 |
| 0 | 011 | 1FFFh | 8192 |
| 0 | 010 | 3FFFh | –(6) |
| 0 | 001 | 3FFFh | –(6) |
| 0 | 000 | 3FFFh | –(6) |

**Notes:**

1. The LVP bit cannot be written (to zero) while operating from the LVP programming interface. The purpose of this rule is to prevent the user from dropping out of LVP mode while programming from LVP mode, or accidentally eliminating LVP mode from the Configuration state.
2. Once protection is enabled through ICSP or a self-write, it can only be reset through a Bulk Erase.
3. Applicable only if $\overline{SAFEN}$ = 0.
4. Applicable only if $\overline{BBEN}$ = 0.
5. BBSIZE[2:0] bits can only be changed when $\overline{BBEN}$ = 1. Once $\overline{BBEN}$ = 0, BBSIZE[2:0] can only be changed through a Bulk Erase.
6. The maximum Boot Block size is half of the user program memory size. Any selection that will exceed the half of a device's program memory will default to a maximum Boot Block size of half PFM.

chanhvinh.com



### 8.6.5 CONFIG5

**Name:** CONFIG5
**Offset:** 0x800B

Configuration Word 5[1]

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | $\overline{CPD}$ | $\overline{CP}$ |
| Access | | | | | | | R/W | R/W |
| Reset | | | | | | | 1 | 1 |

**Bit 1 – $\overline{CPD}$** Data EEPROM Code Protection[2]

| Value | Description |
|---|---|
| 1 | Data EEPROM code protection is disabled |
| 0 | Data EEPROM code protection is enabled |

**Bit 0 – $\overline{CP}$** User Program Flash Memory (PFM) Code Protection[2]

| Value | Description |
|---|---|
| 1 | User PFM code protection is disabled |
| 0 | User PFM code protection is enabled |

**Notes:**

1. Since device code protection takes effect immediately, this Configuration Word may be written last.
2. Once code protection is enabled, it can only be removed through a Bulk Erase.

## 8.7 Register Definitions: Device ID and Revision ID

chanhvinh.com

chanhvinh.com



### 8.7.1 Device ID

**Name:** DEVICEID
**Offset:** 0x8006

Device ID Register

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | Reserved | Reserved | DEV[11:8] | | | |
| Access | | | R | R | R | R | R | R |
| Reset | | | 1 | 1 | q | q | q | q |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | DEV[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | q | q | q | q | q | q | q | q |

**Bit 13 – Reserved** Reserved - Read as '1'

**Bit 12 – Reserved** Reserved - Read as '1'

**Bits 11:0 – DEV[11:0]** Device ID

| Device | Device ID |
|---|---|
| PIC16F17156 | 3105h |
| PIC16F17176 | 3106h |

chanhvinh.com

chanhvinh.com



### 8.7.2 Revision ID

**Name:** REVISIONID
**Offset:** 0x8005

Revision ID Register

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | Reserved | Reserved | MJRREV[5:2] | | | |
| Access | | | R | R | R | R | R | R |
| Reset | | | 1 | 0 | q | q | q | q |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | MJRREV[1:0] | | MNRREV[5:0] | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | q | q | q | q | q | q | q | q |

**Bit 13 – Reserved** Reserved - Read as '1'

**Bit 12 – Reserved** Reserved - Read as '0'

**Bits 11:6 – MJRREV[5:0]** Major Revision ID
These bits are used to identify a major revision (A0, B0, C0, etc.).

**Bits 5:0 – MNRREV[5:0]** Minor Revision ID
These bits are used to identify a minor revision.

chanhvinh.com

chanhvinh.com



# 9. Memory Organization

There are three types of memory in PIC16F171 microcontroller devices:

- Program Memory
  - Program Flash Memory
  - Configuration Words
  - Device ID
  - Revision ID
  - User ID
  - Device Information Area (DIA)
  - Device Configuration Information (DCI)
- Data Memory
  - Core Registers
  - Special Function Registers (FSR)
  - General Purpose RAM (GPR)
  - Common RAM
- Data EEPROM

In Harvard architecture devices, the data and program memories use separate buses that allow for concurrent access of the two memory spaces. The data EEPROM, for practical purposes, can be regarded as a peripheral device, since it is addressed and accessed through a set of control registers.

Additional detailed information on the operation of the Program Flash Memory and data EEPROM memory is provided in the **"NVM - Nonvolatile Memory Module"** section.

## 9.1 Program Memory Organization

The enhanced mid-range core has a 15-bit Program Counter capable of addressing 32K x 14 program memory space. The table below 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. Refer to the **"Interrupts"** chapter for more details.

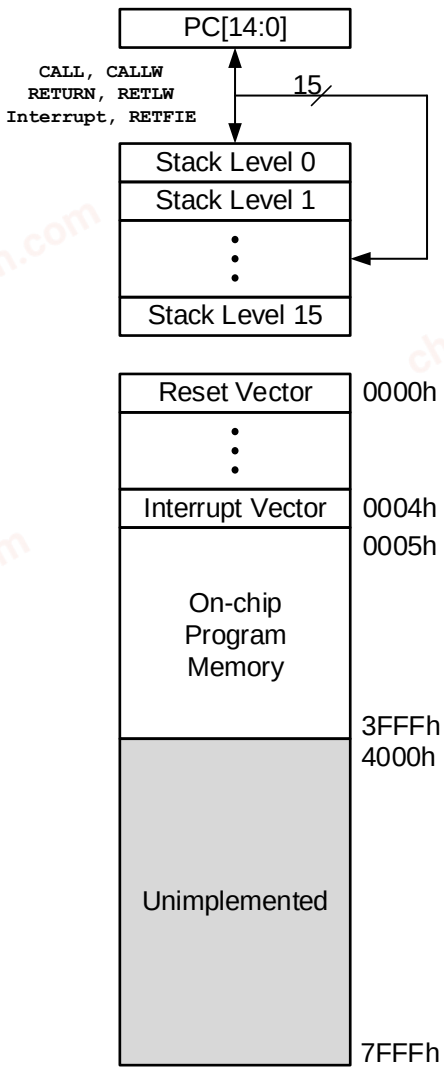
**Table 9-1. Device Sizes and Addresses**

| Device | Program Memory Size (Words) | Last Program Memory Address |
|---|---|---|
| PIC16F17156 | 16,384 | 3FFFh |
| PIC16F17176 | 16,384 | 3FFFh |

chanhvinh.com



**Figure 9-1. Program Memory and Stack (PIC16F171x6)**




### 9.1.1 Reading Program Memory as Data

There are three methods of accessing constants in program memory. The first method is to use tables of `RETLW` instructions, the second, to set an FSR to point to the program memory, and the third is to use the NVMREG interface to access the program memory.

#### 9.1.1.1 `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 the following example.

**Example 9-1. Accessing Table of Constants Using the `RETLW` Instruction**

```
constants
    BRW                     ;Add Index in W to
                            ;program counter to
                            ;select data
```

chanhvinh.com



```
    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 eases the implementation of this type of table.

#### 9.1.1.2 Indirect Read with FSR

The program memory can be accessed as data by setting bit 7 of an 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 INDFx registers. Instructions that read the program memory via the FSR require one extra instruction cycle to complete. The following example demonstrates reading the program memory via an FSR.

The high directive will set bit 7 if a label points to a location in the program memory. This applies to the assembly code shown below.

**Example 9-2. Read of Program Memory Using FSR Register**

```
constants
    RETLW DATA0             ;Index0 data
    RETLW DATA1             ;Index1 data
    RETLW DATA2
    RETLW DATA3
my_function
    ;LOTS OF CODE....
    MOVLW   LOW constants
    MOVWF   FSR1L
    MOVLW   HIGH constants
    MOVWF   FSR1H
    MOVIW   2[FSR1]         ;DATA2 IS IN W
```

### 9.1.2 Memory Access Partition (MAP)

User Flash is partitioned into:

- Application Block
- Boot Block
- Storage Area Flash (SAF) Block

The user can allocate the memory usage by setting the $\overline{\text{BBEN}}$ bit, selecting the size of the partition defined by BBSIZE bits and enabling the Storage Area Flash by the $\overline{\text{SAFEN}}$ bit.

#### 9.1.2.1 Application Block

Default settings of the Configuration bits ($\overline{\text{BBEN}}$ = 1 and $\overline{\text{SAFEN}}$ = 1) assign all memory in the user Flash area to the application block.

#### 9.1.2.2 Boot Block

If $\overline{\text{BBEN}}$ = 1, the Boot Block is enabled and a specific address range is allotted as the Boot Block, based on the value of the BBSIZE bits.

#### 9.1.2.3 Storage Area Flash

Storage Area Flash (SAF) is enabled by clearing the $\overline{\text{SAFEN}}$ bit. If enabled, the SAF block is placed at the end of memory and spans 128 words. If the Storage Area Flash (SAF) is enabled, the SAF area is not available for program execution.

© 2023 Microchip Technology Inc.
and its subsidiaries

chanhvinh.com



**Important:** Storage Area Flash, when enabled, may be used to store variables or other information, often in devices without EEPROM; however, the SAF is accessed in the same manner as other Flash memory areas.

#### 9.1.2.4 Memory Write Protection

All the memory blocks have corresponding write protection bits ($\overline{\text{WRTAPP}}$, $\overline{\text{WRTB}}$, $\overline{\text{WRTC}}$, $\overline{\text{WRTD}}$, and $\overline{\text{WRTSAF}}$). If write-protected locations are written from NVMCON registers, the memory is not changed and the WRERR bit of the NVMCON1 register is set as explained in the **"WRERR Bit"** section from the **"NVM - Nonvolatile Memory Control"** chapter.

#### 9.1.2.5 Memory Violation

A Memory Execution Violation Reset occurs while executing an instruction that has been fetched from outside a valid execution area, clearing the $\overline{\text{MEMV}}$ bit. Refer to the **"Memory Execution Violation"** section in the **"Resets"** chapter for the available valid program execution areas and the PCON1 register definition for $\overline{\text{MEMV}}$ bit conditions.

**Table 9-2. Memory Access Partition**

| REG | Address | Partition | | | |
|---|---|---|---|---|---|
| | | $\overline{\text{BBEN}}$ = 1<br>$\overline{\text{SAFEN}}$ = 1 | $\overline{\text{BBEN}}$ = 1<br>$\overline{\text{SAFEN}}$ = 0 | $\overline{\text{BBEN}}$ = 0<br>$\overline{\text{SAFEN}}$ = 1 | $\overline{\text{BBEN}}$ = 0<br>$\overline{\text{SAFEN}}$ = 0 |
| PFM | 00 0000h ... Last Block Memory Address | Application Block(4) | Application Block(4) | Boot Block(4) | Boot Block(4) |
| | Last Boot Block Memory Address + 1(1) ... Last Program Memory Address - 80h | | | Application Block(4) | Application Block(4) |
| | Last Program Memory Address - 7Fh(2) ... Last Program Memory Address | | SAF(4) | | SAF(4) |
| CONFIG | Config Memory Address(3) | CONFIG | | | |
| EEPROM | F000h-F0FFh | EEPROM | | | |

**Notes:**

1. Last Boot Block Memory Address is based on the BBSIZE Configuration bits.
2. Last Program Memory Address is the Flash size given in the **"Program Memory Organization"** section in the **"NVM - Nonvolatile Memory Control"** chapter.
3. Config Memory Address are the address locations of the Configuration Words given in the **"NVMREG Access to DIA, DCI, User ID, DEV/REV ID, and Configuration Words"** section in the **"NVM - Nonvolatile Memory Control"** chapter.
4. Each memory block has a corresponding write protection fuse defined by the $\overline{\text{WRTAPP}}$, $\overline{\text{WRTB}}$, $\overline{\text{WRTC}}$, $\overline{\text{WRTD}}$, and $\overline{\text{WRTSAF}}$ Configuration bits.

### 9.1.3 Device Information Area (DIA)

The Device Information Area (DIA) is a dedicated region in the Program Flash Memory. The data is mapped from address 8100h to 813Fh. These locations are read-only and cannot be erased or modified. The DIA contains the Microchip Unique Identifier words, Temperature Indicator range data, and the Fixed Voltage Reference (FVR) voltage readings in millivolts (mV). The DIA Table holds the DIA information for the PIC16F171 family of microcontrollers.

chanhvinh.com



**Table 9-3. Device Information Area**

| Address Range | Name of Region | Standard Device Information |
|---|---|---|
| 8100h - 8108h | MUI0 | Microchip Unique Identifier (9 Words) |
| | MUI1 | |
| | MUI2 | |
| | MUI3 | |
| | MUI4 | |
| | MUI5 | |
| | MUI6 | |
| | MUI7 | |
| | MUI8 | |
| 8109h | MUI9 | Reserved (1 Word) |
| 810Ah - 8111h | EUI0 | Optional External Unique Identifier (8 Words) |
| | EUI1 | |
| | EUI2 | |
| | EUI3 | |
| | EUI4 | |
| | EUI5 | |
| | EUI6 | |
| | EUI7 | |
| 8112h | TSLR1(1) | Gain = $\frac{0.1C \times 256}{count}$ (low range setting) |
| 8113h | TSLR2(1) | Temperature sensor ADC reading at 90°C (low range setting) |
| 8114h | TSLR3(1) | Offset (low range setting) |
| 8115h | TSHR1(2) | Gain = $\frac{0.1C \times 256}{count}$ (high range setting) |
| 8116h | TSHR2(2) | Temperature sensor ADC reading at 90°C (high range setting) |
| 8117h | TSHR3(2) | Offset (high range setting) |
| 8118h | FVRA1X | ADC FVR1 output voltage for 1x setting (in mV) |
| 8119h | FVRA2X | ADC FVR1 output voltage for 2x setting (in mV) |
| 811Ah | FVRA4X | ADC FVR1 output voltage for 4x setting (in mV) |
| 811Bh | FVRC1X | Comparator FVR2 output voltage for 1x setting (in mV) |
| 811Ch | FVRC2X | Comparator FVR2 output voltage for 2x setting (in mV) |
| 811Dh | FVRC4X | Comparator FVR2 output voltage for 4x setting (in mV) |
| 811Eh - 811Fh | Reserved | Reserved (2 Words) |

© 2023 Microchip Technology Inc.
and its subsidiaries

chanhvinh.com



.....continued

| Address Range | Name of Region | Standard Device Information |
|---|---|---|

**Notes:**

1. TSLR: Addresses 8112h - 8114h store the measurements for the low range setting of the temperature sensor at $V_{DD}$ = 3.0V, $V_{REF}$+ = 2.048V from FVR1.
2. TSHR: Addresses 8115h - 8117h store the measurements for the high range setting of the temperature sensor at $V_{DD}$ = 3.0V, $V_{REF}$+ = 2.048V from FVR1.

#### 9.1.3.1 Microchip Unique Identifier (MUI)

The PIC16F171 devices are individually encoded during final manufacturing with a Microchip Unique Identifier (MUI). The MUI cannot be erased by a Bulk Erase command or any other user-accessible means. This feature allows for manufacturing traceability of Microchip Technology devices in applications where this is required. It may also be used by the application manufacturer for a number of functions that require unverified unique identification, such as:

- Tracking the device
- Unique serial number

The MUI consists of nine program words. When taken together, these fields form a unique identifier. The MUI is stored in read-only locations, located between 8100h to 8108h in the DIA space. The DIA Table lists the addresses of the identifier words.

**Important:** For applications requiring verified unique identification, contact the Microchip Technology sales office to create a serialized quick turn programming option.

#### 9.1.3.2 External Unique Identifier (EUI)

The EUI data is stored at locations 810Ah to 8111h in the program memory region. This region is an optional space for placing application specific information. The data is coded per customer requirements during manufacturing. The EUI cannot be erased by a Bulk Erase command.

**Important:** Data is stored in this address range on receiving a request from the customer. The customer may contact the local sales representative or Field Applications Engineer, and provide them the unique identifier information that is required to be stored in this region.

#### 9.1.3.3 Standard Parameters for the Temperature Sensor

The purpose of the temperature indicator module is to provide a temperature-dependent voltage that can be measured by an analog module. The DIA Table contains standard parameters for the temperature sensor for low and high range. The values are measured during test and are unique to each device. The calibration data can be used to plot the approximate sensor output voltage, $V_{TSENSE}$ vs. Temperature curve. The **"Temperature Indicator Module"** chapter explains the operation of the Temperature Indicator module and defines terms such as the low range and high range settings of the sensor.

#### 9.1.3.4 Fixed Voltage Reference Data

The Fixed Voltage Reference (FVR) is a stable voltage reference, independent of $V_{DD}$, with 1.024V, 2.048V or 4.096V selectable output levels. The output of the FVR can be configured to supply a reference voltage to the following:

- ADC input channel
- ADC positive reference
- Comparator positive input
- Digital-to-Analog Converter (DAC)

For more information on the FVR, refer to the **"FVR - Fixed Voltage Reference"** chapter.

The DIA stores measured FVR voltages for this device in mV for the different buffer settings of 1x, 2x or 4x.

- FVRA1X stores the value of ADC FVR1 Output Voltage for 1x setting (in mV)

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



- FVRA2X stores the value of ADC FVR1 Output Voltage for 2x setting (in mV)
- FVRA4X stores the value of ADC FVR1 Output Voltage for 4x setting (in mV)
- FVRC1X stores the value of Comparator FVR2 Output Voltage for 1x setting (in mV)
- FVRC2X stores the value of Comparator FVR2 Output Voltage for 2x setting (in mV)
- FVRC4X stores the value of Comparator FVR2 Output Voltage for 4x setting (in mV)

### 9.1.4 Device Configuration Information (DCI)

The Device Configuration Information (DCI) is a dedicated region in the memory that holds information about the device, which is useful for programming and bootloader applications. The data stored in this region is read-only and cannot be modified/erased. Refer to the table below for complete DCI table addresses and description.

**Table 9-4. Device Configuration Information**

| Address | Name | Description | Value | Units |
|---|---|---|---|---|
| | | | PIC16F171x6 | |
| 8200h | ERSIZ | Erase Row Size | 32 | Words |
| 8201h | WLSIZ | Number of write latches per row | 32 | Words |
| 8202h | URSIZ | Number of user erasable rows | 512 | Rows |
| 8203h | EESIZ | Data EEPROM memory size | 256 | Bytes |
| 8204h | PCNT | Pin Count | 28/40 | Pins |

#### 9.1.4.1 DIA and DCI Access

The DIA and DCI data are read-only and cannot be erased or modified. See the **"NVMREG Access to DIA, DCI, User ID, DEV/REV ID, and Configuration Words"** section in the **"NVM - Nonvolatile Memory Control"** chapter for more information on accessing these memory locations.

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

## 9.2 Data Memory Organization

The data memory is partitioned into up to 64 memory banks with 128 bytes in each bank. Each bank consists of:

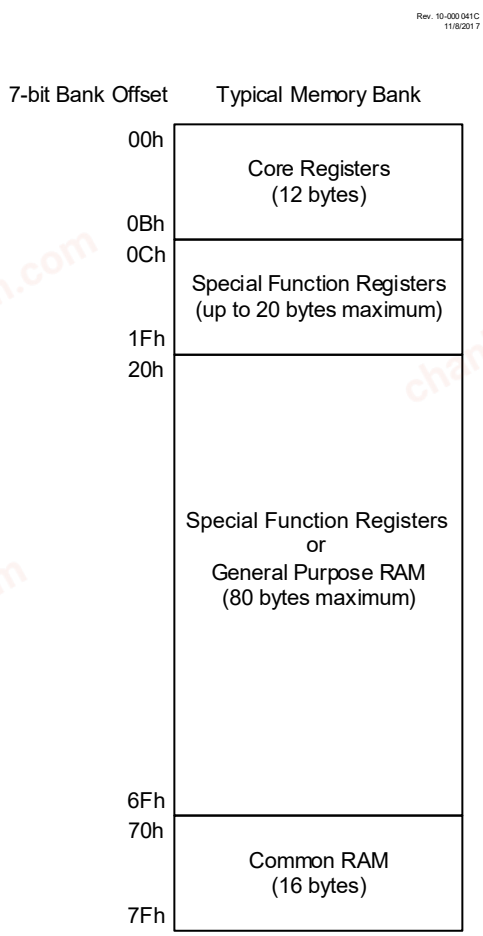
- 12 core registers
- Up to 20 Special Function Registers (SFR)
- Up to 80 bytes of General Purpose RAM (GPR)
- 16 bytes of common RAM

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



**Figure 9-2. Banked Memory Partition**




### 9.2.1 Bank Selection

The active bank is selected by writing the bank number into the Bank Select Register (BSR). All data memory can be accessed either directly via instructions that use the file registers, or indirectly via the two File Select Registers (FSRs). Data memory uses a 13-bit address. The upper six bits of the address define the Bank Address and the lower seven bits select the registers/RAM in that bank.

### 9.2.2 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. These registers are listed in the Core Registers table below.

**Table 9-5. Core Registers**

| Addresses in BANKx | Core Registers |
|---|---|
| x00h or x80h | INDF0 |
| x01h or x81h | INDF1 |
| x02h or x82h | PCL |
| x03h or x83h | STATUS |
| x04h or x84h | FSR0L |

chanhvinh.com



.....continued

| Addresses in BANKx | Core Registers |
|---|---|
| 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 |

### 9.2.3 Special Function Register

The Special Function Registers (SFR) are registers used by the application to control the desired operation of peripheral functions in the device. The SFRs occupy the first 20 bytes of the data banks 0-59 and the first 100 bytes of the data banks 60-63, after the core registers.

The SFRs associated with the operation of the peripherals are described in the appropriate peripheral chapter of this data sheet.

### 9.2.4 General Purpose RAM

There are up to 80 bytes of GPR in each data memory bank. The general purpose RAM can be accessed in a non-banked method via the FSRs. This can simplify access to large memory structures.

Refer to the **"Linear Data Memory"** section in the **"Memory Organization"** chapter for details about linear memory accessing.

### 9.2.5 Common RAM

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

### 9.2.6 Device Memory Maps

The memory maps for the devices in this data sheet are listed in the following figures.

© 2023 Microchip Technology Inc.
and its subsidiaries

chanhvinh.com



## Figure 9-3. Memory Map Banks 0 - 7

| BANK 0 | | BANK 1 | | BANK 2 | | BANK 3 | | BANK 4 | | BANK 5 | | BANK 6 | | BANK 7 | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 000h | Core Registers | 080h | Core Registers | 100h | Core Registers | 180h | Core Registers | 200h | Core Registers | 280h | Core Registers | 300h | Core Registers | 380h | Core Registers |
| 00Bh | | 08Bh | | 10Bh | | 18Bh | | 20Bh | | 28Bh | | 30Bh | | 38Bh | |
| 00Ch | PORTA | 08Ch | PIR0 | 10Ch | PMD0 | 18Ch | WDTCON0 | 20Ch | FVRCON | 28Ch | CPUDOZE | 30Ch | TMR1L | 38Ch | T2TMR |
| 00Dh | PORTB | 08Dh | PIR1 | 10Dh | PMD1 | 18Dh | WDTCON1 | 20Dh | CPCON | 28Dh | OSCCON1 | 30Dh | TMR1H | 38Dh | T2PR |
| 00Eh | PORTC | 08Eh | PIR2 | 10Eh | PMD2 | 18Eh | WDTPSL | 20Eh | — | 28Eh | OSCCON2 | 30Eh | T1CON | 38Eh | T2CON |
| 00Fh | PORTD(1) | 08Fh | PIR3 | 10Fh | PMD3 | 18Fh | WDTPSH | 20Fh | — | 28Fh | OSCCON3 | 30Fh | T1GCON | 38Fh | T2HLT |
| 010h | PORTE | 090h | PIR4 | 110h | PMD4 | 190h | WDTTMR | 210h | — | 290h | OSCSTAT | 310h | T1GATE | 390h | T2CLKCON |
| 011h | — | 091h | PIR5 | 111h | — | 191h | BORCON | 211h | — | 291h | OSCEN | 311h | T1CLK | 391h | T2RST |
| 012h | TRISA | 092h | PIR6 | 112h | — | 192h | PCON0 | 212h | — | 292h | OSCTUNE | 312h | TMR3L | 392h | T4TMR |
| 013h | TRISB | 093h | — | 113h | — | 193h | PCON1 | 213h | — | 293h | OSCFRQ | 313h | TMR3H | 393h | T4PR |
| 014h | TRISC | 094h | — | 114h | — | 194h | — | 214h | — | 294h | ACTCON | 314h | T3CON | 394h | T4CON |
| 015h | TRISD(1) | 095h | — | 115h | — | 195h | — | 215h | — | 295h | — | 315h | T3GCON | 395h | T4HLT |
| 016h | TRISE | 096h | PIE0 | 116h | — | 196h | — | 216h | — | 296h | CLKRCON | 316h | T3GATE | 396h | T4CLKCON |
| 017h | — | 097h | PIE1 | 117h | — | 197h | — | 217h | — | 297h | CLKRCLK | 317h | T3CLK | 397h | T4RST |
| 018h | LATA | 098h | PIE2 | 118h | — | 198h | — | 218h | — | 298h | — | 318h | — | 398h | T6TMR |
| 019h | LATB | 099h | PIE3 | 119h | — | 199h | — | 219h | — | 299h | — | 319h | — | 399h | T6PR |
| 01Ah | LATC | 09Ah | PIE4 | 11Ah | — | 19Ah | — | 21Ah | — | 29Ah | — | 31Ah | — | 39Ah | T6CON |
| 01Bh | LATD(1) | 09Bh | PIE5 | 11Bh | — | 19Bh | — | 21Bh | — | 29Bh | — | 31Bh | — | 39Bh | T6HLT |
| 01Ch | LATE | 09Ch | PIE6 | 11Ch | — | 19Ch | TMR0L | 21Ch | — | 29Ch | — | 31Ch | — | 39Ch | T6CLKCON |
| 01Dh | — | 09Dh | — | 11Dh | — | 19Dh | TRM0H | 21Dh | — | 29Dh | — | 31Dh | — | 39Dh | T6RST |
| 01Eh | — | 09Eh | — | 11Eh | — | 19Eh | T0CON0 | 21Eh | — | 29Eh | — | 31Eh | — | 39Eh | — |
| 01Fh | — | 09Fh | — | 11Fh | — | 19Fh | T0CON1 | 21Fh | ZCDCON | 29Fh | — | 31Fh | — | 39Fh | — |
| 020h | General Purpose Registers 80 Bytes | 0A0h | General Purpose Registers 80 Bytes | 120h | General Purpose Registers 80 Bytes | 1A0h | General Purpose Registers 80 Bytes | 220h | General Purpose Registers 80 Bytes | 2A0h | General Purpose Registers 80 Bytes | 320h | General Purpose Registers 80 Bytes | 3A0h | General Purpose Registers 80 Bytes |
| 06Fh | | 0EFh | | 16Fh | | 1EFh | | 26Fh | | 2EFh | | 36Fh | | 3EFh | |
| 070h | Common RAM (Accesses 70h-7Fh) | 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 | |

**Note:** **1. Available on 40/44-pin devices only**

Legend:

Unimplemented data memory locations, read as '0'

## Figure 9-4. Memory Map Banks 8 - 15

| BANK 8 | | BANK 9 | | BANK 10 | | BANK 11 | | BANK 12 | | BANK 13 | | BANK 14 | | BANK 15 | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 400h | Core Registers | 480h | Core Registers | 500h | Core Registers | 580h | Core Registers | 600h | Core Registers | 680h | Core Registers | 700h | Core Registers | 780h | Core Registers |
| 40Bh | | 48Bh | | 50Bh | | 58Bh | | 60Bh | | 68Bh | | 70Bh | | 78Bh | |
| 40Ch | CCPR1L | 48Ch | PWM1ERS | 50Ch | PWM2ERS | 58Ch | NCO1ACCL | 60Ch | CWG1CLK | 68Ch | CLCnCON | 70Ch | RC1REG | 78Ch | SSP1BUF |
| 40Dh | CCPR1H | 48Dh | PWM1CLK | 50Dh | PWM2CLK | 58Dh | NCO1ACCH | 60Dh | CWG1ISM | 68Dh | CLCnPOL | 70Dh | TX1REG | 78Dh | SSP1ADD |
| 40Eh | CCP1CON | 48Eh | PWM1DLS | 50Eh | PWM2DLS | 58Eh | NCO1ACCU | 60Eh | CWG1DBR | 68Eh | CLCnSEL0 | 70Eh | SP1GRBL | 78Eh | SSP1MSK |
| 40Fh | CCP1CAP | 48Fh | PWM1PRL | 50Fh | PWM2PRL | 58Fh | NCO1INCL | 60Fh | CWG1DBF | 68Fh | CLCnSEL1 | 70Fh | SP1BRGH | 78Fh | SSP1STAT |
| 410h | CCPR2L | 490h | PWM1PRH | 510h | PWM2PRH | 590h | NCO1INCH | 610h | CWG1CON0 | 690h | CLCnSEL2 | 710h | RC1STA | 790h | SSP1CON1 |
| 411h | CCPR2H | 491h | PWM1CPRE | 511h | PWM2CPRE | 591h | NCO1INCU | 611h | CWG1CON1 | 691h | CLCnSEL3 | 711h | TX1STA | 791h | SSP1CON2 |
| 412h | CCP2CON | 492h | PWM1PIPOS | 512h | PWM2PIPOS | 592h | NCO1CON | 612h | CWG1AS0 | 692h | CLCnGLS0 | 712h | BAUD1CON | 792h | SSP1CON3 |
| 413h | CCP2CAP | 493h | PWM1GIR | 513h | PWM2GIR | 593h | NCO1CLK | 613h | CWG1AS1 | 693h | CLCnGLS1 | 713h | — | 793h | — |
| 414h | — | 494h | PWM1GIE | 514h | PWM2GIE | 594h | — | 614h | CWG1STR | 694h | CLCnGLS2 | 714h | — | 794h | — |
| 415h | — | 495h | PWM1CON | 515h | PWM2CON | 595h | — | 615h | — | 695h | CLCnGLS3 | 715h | — | 795h | — |
| 416h | — | 496h | PWM1S1CFG | 516h | PWM2S1CFG | 596h | — | 616h | — | 696h | CLCSELECT | 716h | RC2REG | 796h | SSP2BUF |
| 417h | — | 497h | PWM1S1P1L | 517h | PWM2S1P1L | 597h | — | 617h | — | 697h | CLCDATA | 717h | TX2REG | 797h | SSP2ADD |
| 418h | — | 498h | PWM1S1P1H | 518h | PWM2S1P1H | 598h | — | 618h | — | 698h | — | 718h | SP2GRBL | 798h | SSP2MSK |
| 419h | — | 499h | PWM1S1P2L | 519h | PWM2S1P2L | 599h | — | 619h | — | 699h | — | 719h | SP2BRGH | 799h | SSP2STAT |
| 41Ah | — | 49Ah | PWM1S1P2H | 51Ah | PWM2S1P2H | 59Ah | — | 61Ah | — | 69Ah | — | 71Ah | RC2STA | 79Ah | SSP2CON1 |
| 41Bh | — | 49Bh | — | 51Bh | — | 59Bh | — | 61Bh | — | 69Bh | — | 71Bh | TX2STA | 79Bh | SSP2CON2 |
| 41Ch | — | 49Ch | — | 51Ch | — | 59Ch | — | 61Ch | — | 69Ch | — | 71Ch | BAUD2CON | 79Ch | SSP2CON3 |
| 41Dh | — | 49Dh | — | 51Dh | — | 59Dh | — | 61Dh | — | 69Dh | — | 71Dh | — | 79Dh | — |
| 41Eh | — | 49Eh | PWMLOAD | 51Eh | — | 59Eh | — | 61Eh | — | 69Eh | — | 71Eh | — | 79Eh | — |
| 41Fh | CCPTMRS0 | 49Fh | PWMEN | 51Fh | — | 59Fh | — | 61Fh | — | 69Fh | — | 71Fh | — | 79Fh | — |
| 420h | General Purpose Registers 80 Bytes | 4A0h | General Purpose Registers 80 Bytes | 520h | General Purpose Registers 80 Bytes | 5A0h | General Purpose Registers 80 Bytes | 620h | General Purpose Registers 80 Bytes | 6A0h | General Purpose Registers 80 Bytes | 720h | General Purpose Registers 80 Bytes | 7A0h | General Purpose Registers 80 Bytes |
| 46Fh | | 4EFh | | 56Fh | | 5EFh | | 66Fh | | 6EFh | | 76Fh | | 7EFh | |
| 470h | Common RAM (Accesses 70h-7Fh) | 4F0h | Common RAM (Accesses 70h-7Fh) | 570h | Common RAM (Accesses 70h-7Fh) | 5F0h | Common RAM (Accesses 70h-7Fh) | 670h | Common RAM (Accesses 70h-7Fh) | 6F0h | Common RAM (Accesses 70h-7Fh) | 770h | Common RAM (Accesses 70h-7Fh) | 7F0h | Common RAM (Accesses 70h-7Fh) |
| 47Fh | | 4FFh | | 57Fh | | 5FFh | | 67Fh | | 6FFh | | 77Fh | | 7FFh | |

Legend:

Unimplemented data memory locations, read as '0'

chanhvinh.com



## Figure 9-5. Memory Map Banks 16 - 23

| BANK 16 | | BANK 17 | | BANK 18 | | BANK 19 | | BANK 20 | | BANK 21 | | BANK 22 | | BANK 23 | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 800h | Core Registers | 880h | Core Registers | 900h | Core Registers | 980h | Core Registers | A00h | Core Registers | A80h | Core Registers | B00h | Core Registers | B80h | Core Registers |
| 80Bh | | 88Bh | | 90Bh | | 98Bh | | A0Bh | | A8Bh | | B0Bh | | B8Bh | |
| 80Ch | CM1CON0 | 88Ch | DAC1CON0 | 90Ch | OPA1CON0 | 98Ch | PWM3ERS | A0Ch | PWM4ERS | A8Ch | — | B0Ch | — | B8Ch | — |
| 80Dh | CM1CON1 | 88Dh | DAC1DATL | 90Dh | OPA1CON1 | 98Dh | PWM3CLK | A0Dh | PWM4CLK | A8Dh | — | B0Dh | — | B8Dh | — |
| 80Eh | CM1NCH | 88Eh | — | 90Eh | OPA1CON2 | 98Eh | PWM3LDS | A0Eh | PWM4LDS | A8Eh | — | B0Eh | — | B8Eh | — |
| 80Fh | CM1PCH | 88Fh | — | 90Fh | OPA1CON3 | 98Fh | PWM3PRL | A0Fh | PWM4PRL | A8Fh | — | B0Fh | — | B8Fh | — |
| 810h | CM2CON0 | 890h | DAC2CON0 | 910h | OPA1HWC | 990h | PWM3PRH | A10h | PWM4PRH | A90h | — | B10h | — | B90h | — |
| 811h | CM2CON1 | 891h | DAC2DATL | 911h | OPA1OFFSET | 991h | PWM3CPRE | A11h | PWM4CPRE | A91h | — | B11h | — | B91h | — |
| 812h | CM2NCH | 892h | — | 912h | OPA1ORS | 992h | PWM3PIPOS | A12h | PWM4PIPOS | A92h | — | B12h | — | B92h | — |
| 813h | CM2PCH | 893h | — | 913h | — | 993h | PWM3GIR | A13h | PWM4GIR | A93h | — | B13h | — | B93h | — |
| 814h | — | 894h | — | 914h | — | 994h | PWM3GIE | A14h | PWM4GIE | A94h | — | B14h | — | B94h | — |
| 815h | — | 895h | — | 915h | — | 995h | PWM3CON | A15h | PWM4CON | A95h | — | B15h | — | B95h | — |
| 816h | — | 896h | — | 916h | — | 996h | PWM3S1CFG | A16h | PWM4S1CFG | A96h | — | B16h | — | B96h | — |
| 817h | — | 897h | — | 917h | — | 997h | PWM3S1P1L | A17h | PWM4S1P1L | A97h | — | B17h | — | B97h | — |
| 818h | — | 898h | — | 918h | — | 998h | PWM3S1P1H | A18h | PWM4S1P1H | A98h | — | B18h | — | B98h | — |
| 819h | — | 899h | — | 919h | — | 999h | PWM3S1P2L | A19h | PWM4S1P2L | A99h | — | B19h | — | B99h | — |
| 81Ah | — | 89Ah | — | 91Ah | — | 99Ah | PWM3S1P2H | A1Ah | PWM4S1P2H | A9Ah | — | B1Ah | — | B9Ah | — |
| 81Bh | — | 89Bh | — | 91Bh | — | 99Bh | — | A1Bh | — | A9Bh | — | B1Bh | — | B9Bh | — |
| 81Ch | — | 89Ch | — | 91Ch | — | 99Ch | — | A1Ch | — | A9Ch | — | B1Ch | — | B9Ch | — |
| 81Dh | — | 89Dh | — | 91Dh | — | 99Dh | — | A1Dh | — | A9Dh | — | B1Dh | — | B9Dh | — |
| 81Eh | — | 89Eh | — | 91Eh | — | 99Eh | — | A1Eh | — | A9Eh | — | B1Eh | — | B9Eh | — |
| 81Fh | CMOUT | 89Fh | — | 91Fh | — | 99Fh | — | A1Fh | — | A9Fh | — | B1Fh | — | B9Fh | — |
| 820h | General Purpose Registers 80 Bytes | 8A0h | General Purpose Registers 80 Bytes | 920h | General Purpose Registers 80 Bytes | 9A0h | General Purpose Registers 80 Bytes | A20h | General Purpose Registers 80 Bytes | AA0h | General Purpose Registers 80 Bytes | B20h | General Purpose Registers 80 Bytes | BA0h | General Purpose Registers 80 Bytes |
| 86Fh | | 8EFh | | 96Fh | | 9EFh | | A6Fh | | AEFh | | B6Fh | | BEFh | |
| 870h | Common RAM (Accesses 70h-7Fh) | 8F0h | Common RAM (Accesses 70h-7Fh) | 970h | Common RAM (Accesses 70h-7Fh) | 9F0h | Common RAM (Accesses 70h-7Fh) | A70h | Common RAM (Accesses 70h-7Fh) | AF0h | Common RAM (Accesses 70h-7Fh) | B70h | Common RAM (Accesses 70h-7Fh) | BF0h | Common RAM (Accesses 70h-7Fh) |
| 87Fh | | 8FFh | | 97Fh | | 9FFh | | A7Fh | | AFFh | | B7Fh | | BFFh | |

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

## Figure 9-6. Memory Map Banks 24 - 31

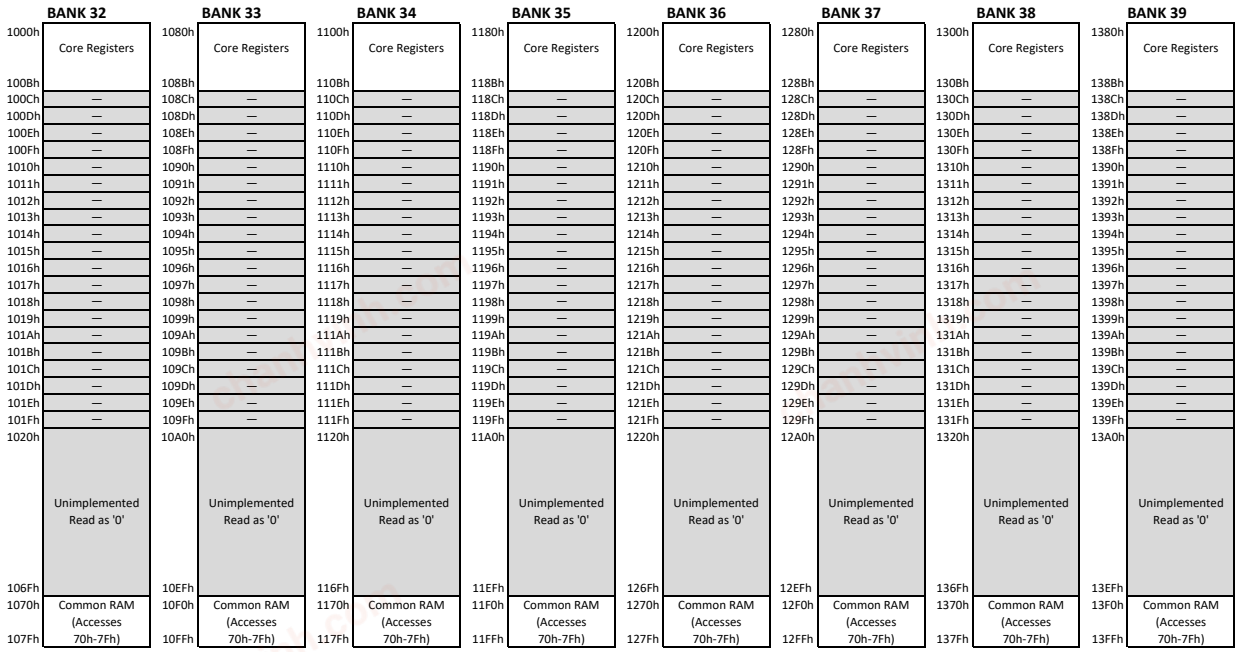
| BANK 24 | | BANK 25 | | BANK 26 | | BANK 27 | | BANK 28 | | BANK 29 | | BANK 30 | | BANK 31 | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| C00h | Core Registers | C80h | Core Registers | D00h | Core Registers | D80h | Core Registers | E00h | Core Registers | E80h | Core Registers | F00h | Core Registers | F80h | Core Registers |
| 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 | General Purpose Registers 80 Bytes | CA0h | General Purpose Registers 32 Bytes | D20h | Unimplemented Read as '0' | DA0h | Unimplemented Read as '0' | E20h | Unimplemented Read as '0' | EA0h | Unimplemented Read as '0' | F20h | Unimplemented Read as '0' | FA0h | Unimplemented Read as '0' |
| C6Fh | | CBFh | | D6Fh | | DEFh | | E6Fh | | EEFh | | F6Fh | | FEFh | |
| C70h | Common RAM (Accesses 70h-7Fh) | CC0h | Unimplemented Read as '0' | D70h | Common RAM (Accesses 70h-7Fh) | DF0h | Common RAM (Accesses 70h-7Fh) | E70h | Common RAM (Accesses 70h-7Fh) | EF0h | Common RAM (Accesses 70h-7Fh) | F70h | Common RAM (Accesses 70h-7Fh) | FF0h | Common RAM (Accesses 70h-7Fh) |
| C7Fh | | CFFh | | D7Fh | | DFFh | | E7Fh | | EFFh | | F7Fh | | FFFh | |

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

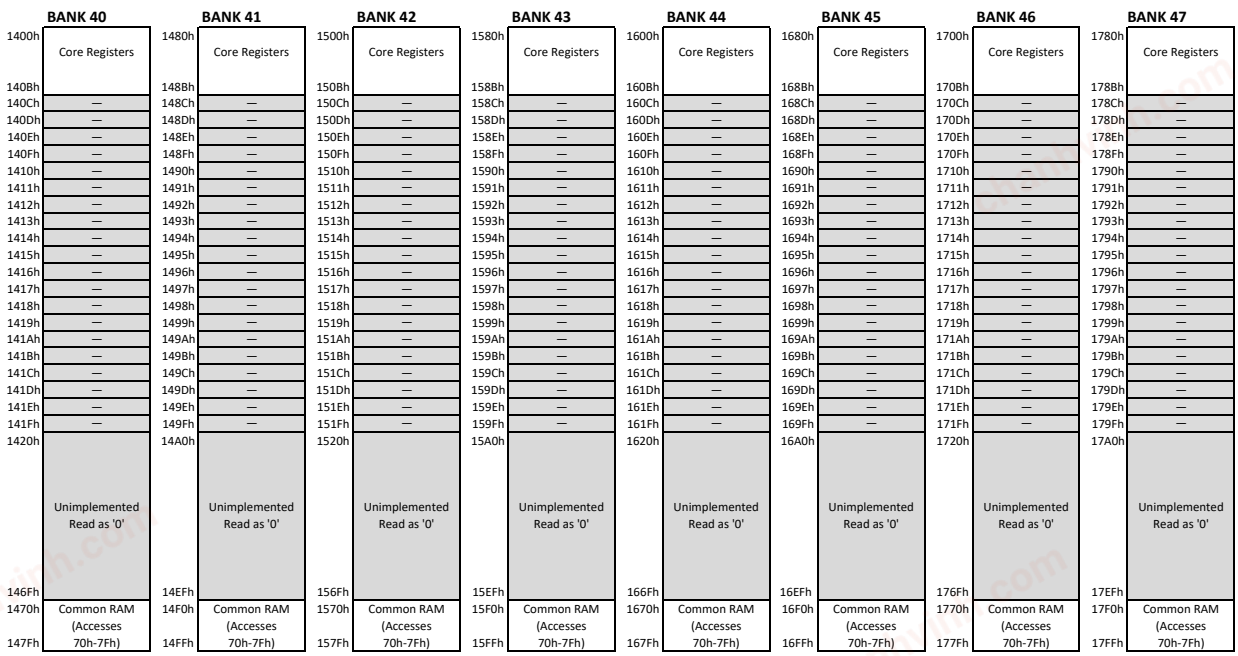
chanhvinh.com



## Figure 9-7. Memory Map Banks 32 - 39



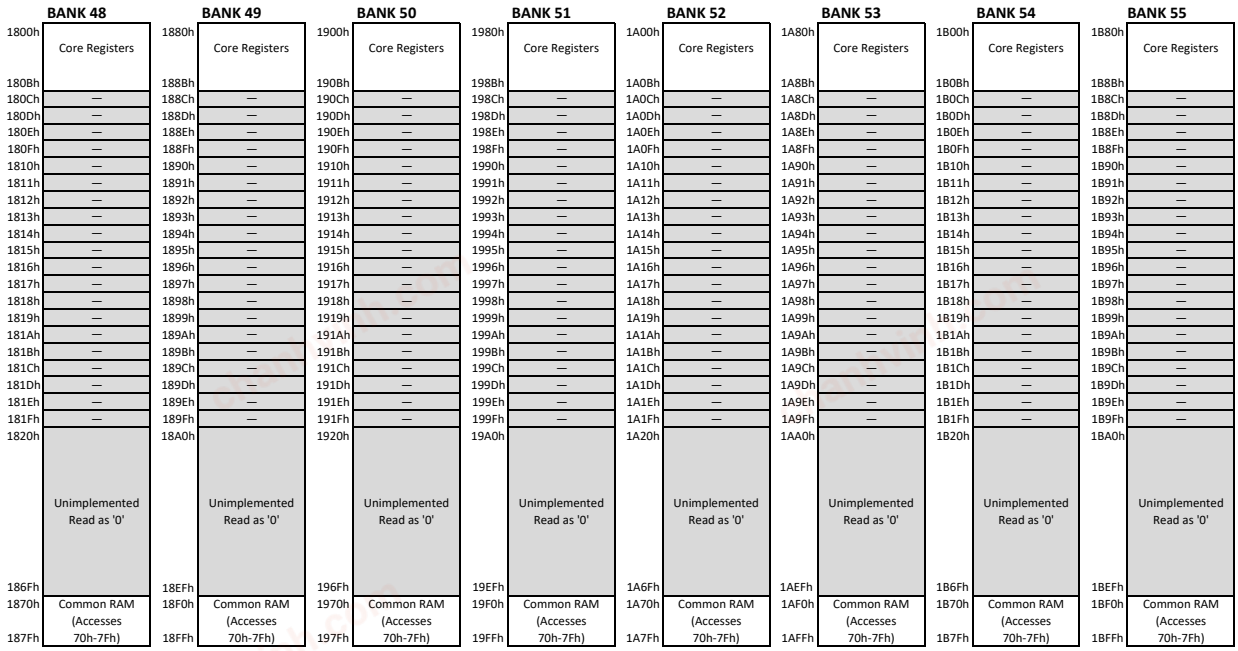

## Figure 9-8. Memory Map Banks 40 - 47



chanhvinh.com



**Figure 9-9. Memory Map Banks 48 - 55**



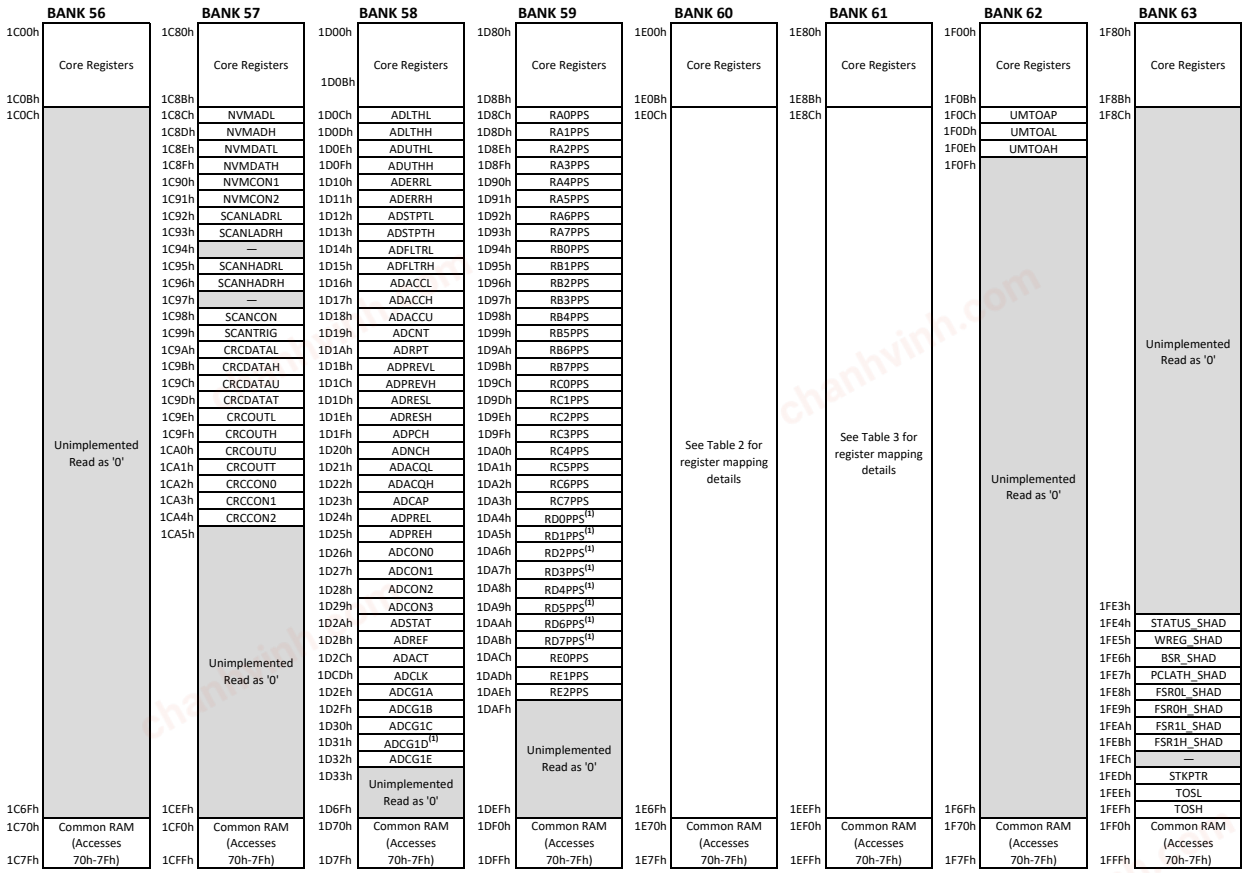

Legend:

Unimplemented data memory locations, read as '0'

chanhvinh.com



## Figure 9-10. Memory Map Banks 56 - 63




**Note:** **1. Available on 40/44-pin devices only**

Legend:

Unimplemented data memory locations, read as '0'

chanhvinh.com



**Figure 9-11. Memory Map Bank 61**

**BANK 60**

| | | | |
|---|---|---|---|
| 1E00h | Core Registers | 1E3Ah | Unimplemented Read as '0' |
| 1E0Bh | | 1E3Ch | |
| 1E0Ch | PPSLOCK | 1E3Dh | CLCIN0PPS |
| 1E0Dh | INTPPS | 1E3Eh | CLCIN1PPS |
| 1E0Eh | T0CKIPPS | 1E3Fh | CLCIN2PPS |
| 1E0Fh | T1CKIPPS | 1E40h | CLCIN3PPS |
| 1E10h | T1GPPS | 1E41h | DT1PPS |
| 1E11h | T3CKIPPS | 1E42h | RX1PPS |
| 1E12h | T3GPPS | 1E43h | DT2PPS |
| 1E13h | Unimplemented Read as '0' | 1E44h | RX2PPS |
| | | 1E45h | — |
| 1E18h | | 1E46h | — |
| 1E19h | T2INPPS | 1E47h | SSP1CLKPPS |
| 1E1Ah | T4INPPS | 1E48h | SSP1DATPPS |
| 1E1Bh | T6INPPS | 1E49h | SSP1SSPPS |
| 1E1Ch | — | 1E4Ah | SSP2CLKPPS |
| 1E1Dh | — | 1E4Bh | SSP2DATPPS |
| 1E1Eh | CCP1PPS | 1E4Ch | SSP2SSPPS |
| 1E1Fh | CCP2PPS | 1E4Dh | Unimplemented Read as '0' |
| 1E20h | Unimplemented Read as '0' | 1E4Fh | |
| 1E23h | | 1E50h | ADACTPPS |
| 1E24h | PWMIN0PPS | 1E51h | — |
| 1E25h | PWMIN1PPS | 1E52h | — |
| 1E26h | PWM1ERSPPS | 1E53h | OPA1PPS |
| 1E27h | PWM2ERSPPS | 1E54h | Unimplemented Read as '0' |
| 1E28h | PWM3ERSPPS | | |
| 1E29h | PWM4ERSPPS | 1E6Fh | |
| 1E2Ah | Unimplemented Read as '0' | 1E70h | Common RAM (Accesses 70h-7Fh) |
| 1E38h | | 1E7Fh | |
| 1E39h | CWG1PPS | | |

Legend:

Unimplemented data memory locations, read as '0'

chanhvinh.com



**Figure 9-12. Memory Map Bank 62**

**BANK 61**

| | | | |
|---|---|---|---|
| 1E80h | Core Registers | 1EAAh | ANSELD(1) |
| | | 1EABh | WPUD(1) |
| 1E8Bh | | 1EACh | ODCOND(1) |
| 1E8Ch | ANSELA | 1EADh | SLRCOND(1) |
| 1E8Dh | WPUA | 1EAEh | INLVLD(1) |
| 1E8Eh | ODCONA | 1EAFh | Unimplemented Read as '0' |
| 1E8Fh | SLRCONA | | |
| 1E90h | INLVLA | 1EB3h | |
| 1E91h | IOCAP | 1EB4h | ANSELE |
| 1E92h | IOCAN | 1EB5h | WPUE |
| 1E93h | IOCAF | 1EB6h | ODCONE |
| 1E94h | — | 1EB7h | SLRCONE |
| 1E95h | — | 1EB8h | INLVLE |
| 1E96h | ANSELB | 1EB9h | IOCEP |
| 1E97h | WPUB | 1EBAh | IOCEN |
| 1E98h | ODCONB | 1EBBh | IOCEF |
| 1E99h | SLRCONB | 1EBCh | Unimplemented Read as '0' |
| 1E9Ah | INLVLB | | |
| 1E9Bh | IOCBP | 1EE2h | |
| 1E9Ch | IOCBN | 1EE3h | RB1I2C |
| 1E9Dh | IOCBF | 1EE4h | RB2I2C |
| 1E9Eh | — | 1EE5h | Unimplemented Read as '0' |
| 1E9Fh | — | | |
| 1EA0h | ANSELC | 1EEAh | |
| 1EA1h | WPUC | 1EEBh | RC3I2C |
| 1EA2h | ODCONC | 1EECh | RC4I2C |
| 1EA3h | SLRCONC | 1EEDh | — |
| 1EA4h | INLVLC | 1EEEh | RD0I2C(1) |
| 1EA5h | IOCCP | 1EEFh | RD1I2C(1) |
| 1EA6h | IOCCN | 1EF0h | Common RAM (Accesses 70h-7Fh) |
| 1EA7h | IOCCF | | |
| 1EA8h | — | | |
| 1EA9h | — | 1EFFh | |

**Note:** **1. Available on 40/44-pin devices only**

Legend:

Unimplemented data memory locations, read as '0'

## 9.3 STATUS Register

The STATUS register contains:

- the arithmetic status of the ALU

chanhvinh.com



- 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 writes to these three bits are disabled. These bits are set or cleared according to the device logic. Furthermore, the $\overline{TO}$ and $\overline{PD}$ 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 bits [4:3] and [1:0], and set the Z bit. This leaves the STATUS register as '`000u u1uu`' (where u = unchanged).

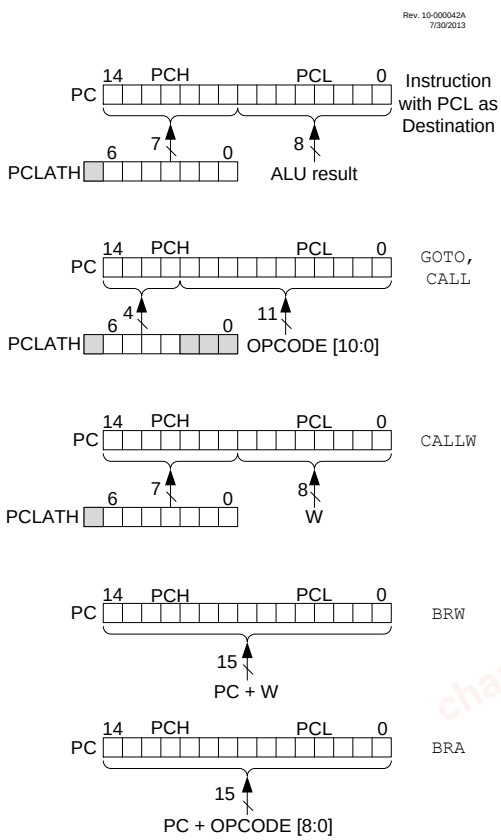
It is recommended, therefore, that only `BCF`, `BSF`, `SWAPF` and `MOVWF` instructions are used to alter the STATUS register, because these instructions do not affect any Status bits. For other instructions not affecting any Status bits, refer to the **"Instruction Set Summary"** chapter.

**Important:** The C and DC bits operate as $\overline{Borrow}$ and $\overline{Digit\ Borrow}$ out bits, respectively, in subtraction.

## 9.4 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. Loading of PC in Different Situations shows the five situations for the loading of the PC.

**Figure 9-13. Loading of PC in Different Situations**



chanhvinh.com



### 9.4.1 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.

### 9.4.2 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 has to 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).

### 9.4.3 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 has to 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.

### 9.4.4 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.

## 9.5 Stack

All devices have a 16-level by 15-bit wide hardware stack. The stack space is not part of either program or data space. The PC is PUSHed onto the stack when the 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 Configuration bit is programmed to '0'. 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.

If the STVREN bit is programmed to '1', the device will be reset if the stack is PUSHed beyond the sixteenth level or POPed beyond the fist level, setting the appropriate bits (STKOVF or STKUNF, respectively).

**Important:** 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.

### 9.5.1 Accessing the Stack

The stack is accessible through the TOSH, TOSL and STKPTR registers. STKPTR is the current value of the Stack Pointer. The TOSH:TOSL register pair points to the TOP of the stack. Both registers are read/writable. TOS is split

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



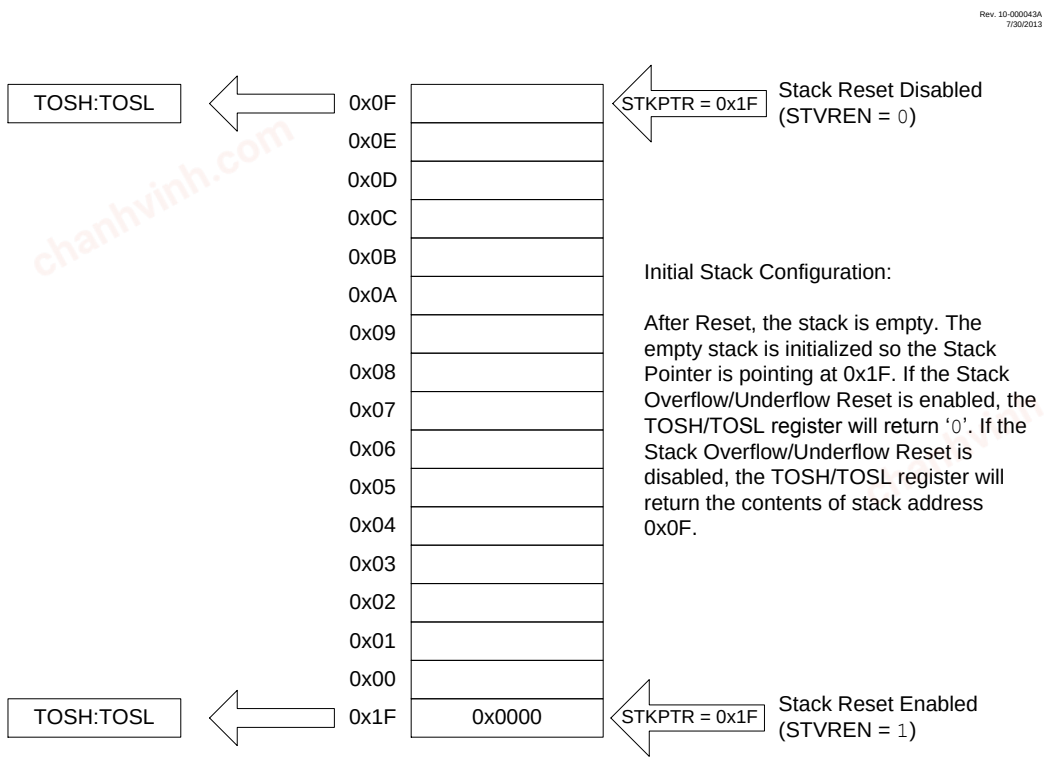
into TOSH and TOSL due to the 15-bit size of the PC. To access the stack, adjust the value of STKPTR, which will position TOSH:TOSL, then read/write to TOSH:TOSL. STKPTR also allows the detection of Overflow and Underflow condition.

**Important:** Care must be taken when modifying STKPTR while interrupts are enabled.

During normal program operation, CALL, CALLW and interrupts will increment STKPTR, while RETLW, RETURN and RETFIE will decrement STKPTR. STKPTR can be monitored to obtain the value of stack memory left at any given time. STKPTR always points at the currently used place on the stack. Therefore, a CALL or CALLW will increment STKPTR and then write the PC, and a return will unload the PC value from the stack and then decrement STKPTR.

Reference the following figures for examples of accessing the stack.

**Figure 9-14. Accessing the Stack Example 1**



chanhvinh.com



**Figure 9-15. Accessing the Stack Example 2**

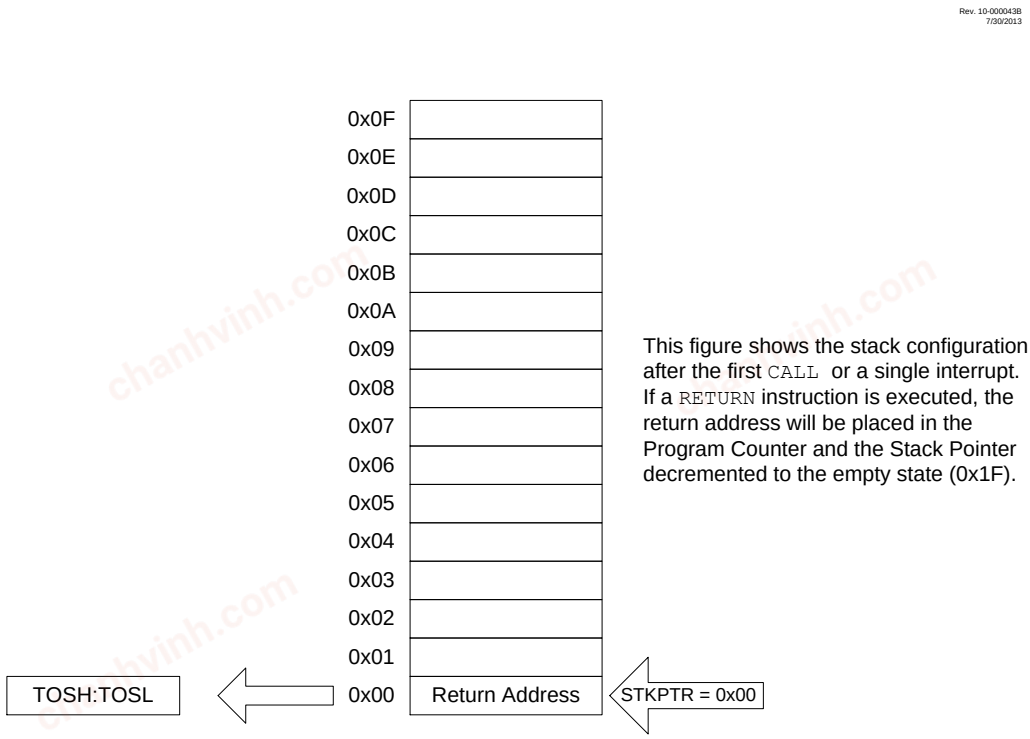
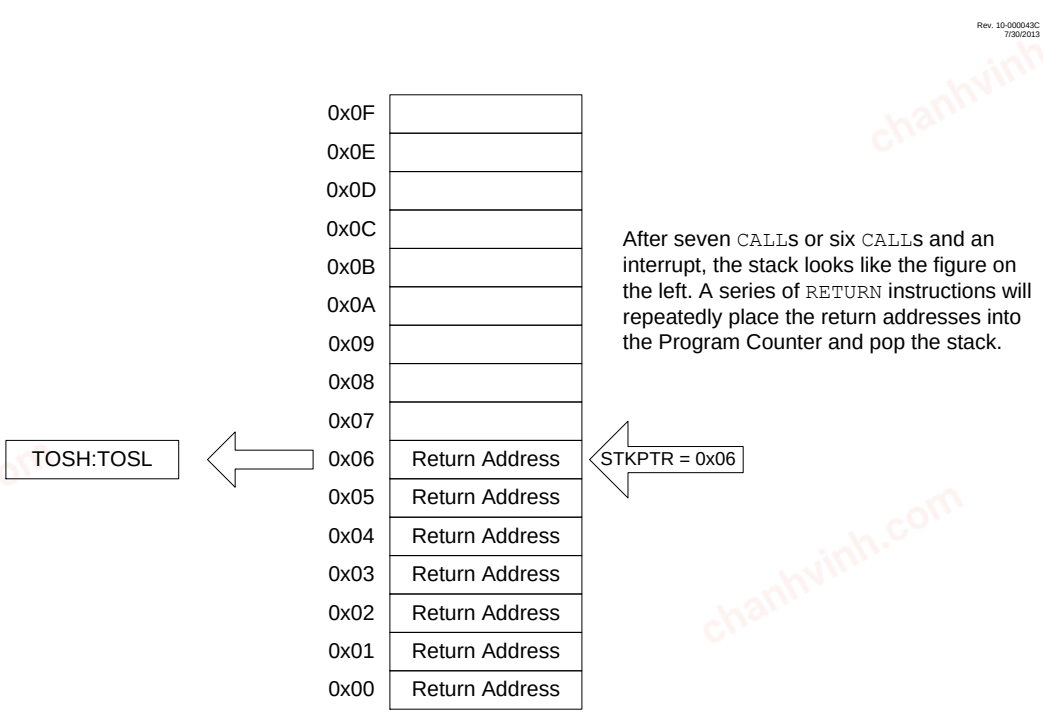


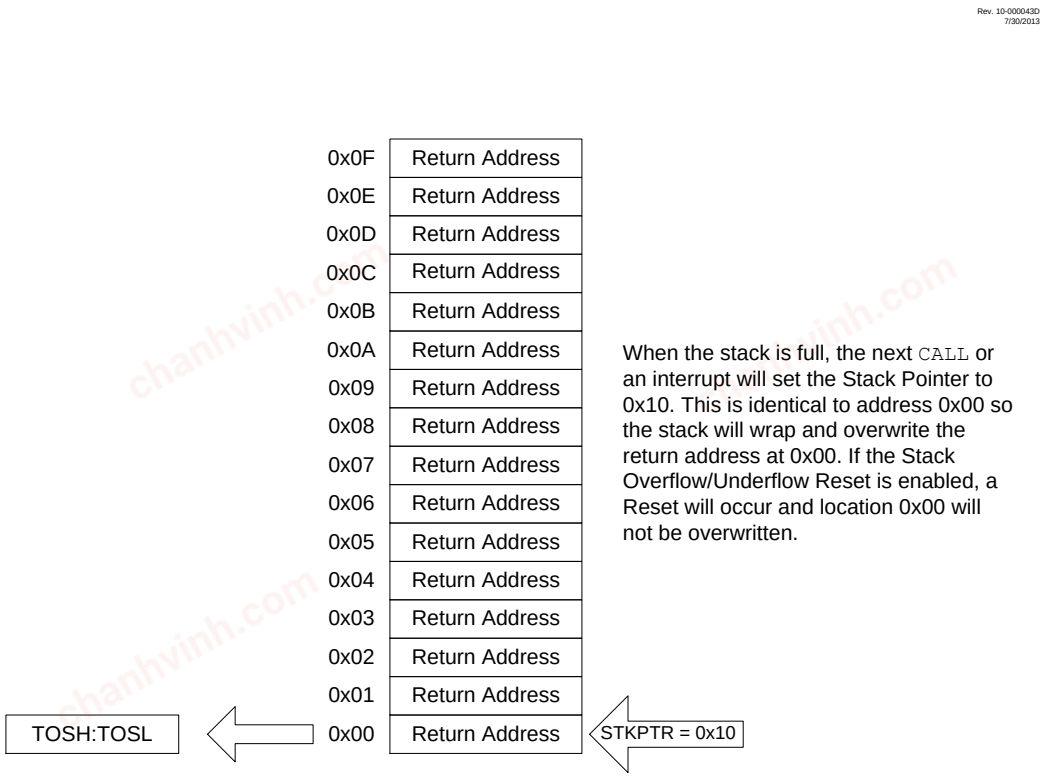

**Figure 9-16. Accessing the Stack Example 3**



chanhvinh.com



**Figure 9-17. Accessing the Stack Example 4**




### 9.5.2 Overflow/Underflow Reset

If the STVREN bit 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).

## 9.6 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.

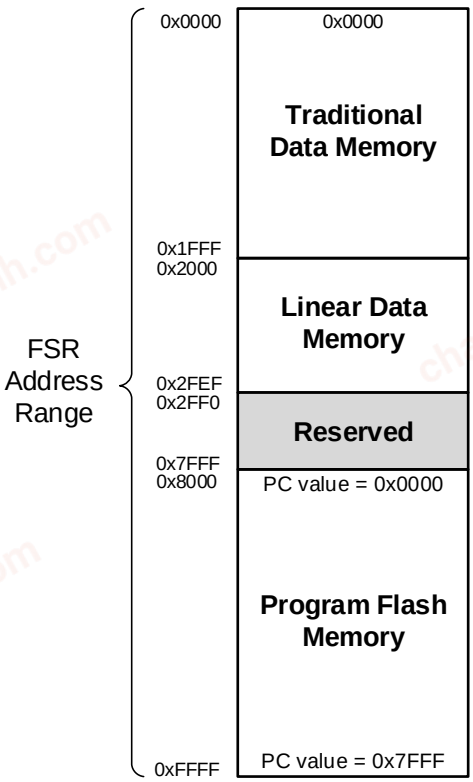
The FSR registers form a 16-bit address that allows an addressing space with 65536 locations. These locations are divided into three memory regions:

- Traditional/Banked Data Memory
- Linear Data Memory
- Program Flash Memory

chanhvinh.com



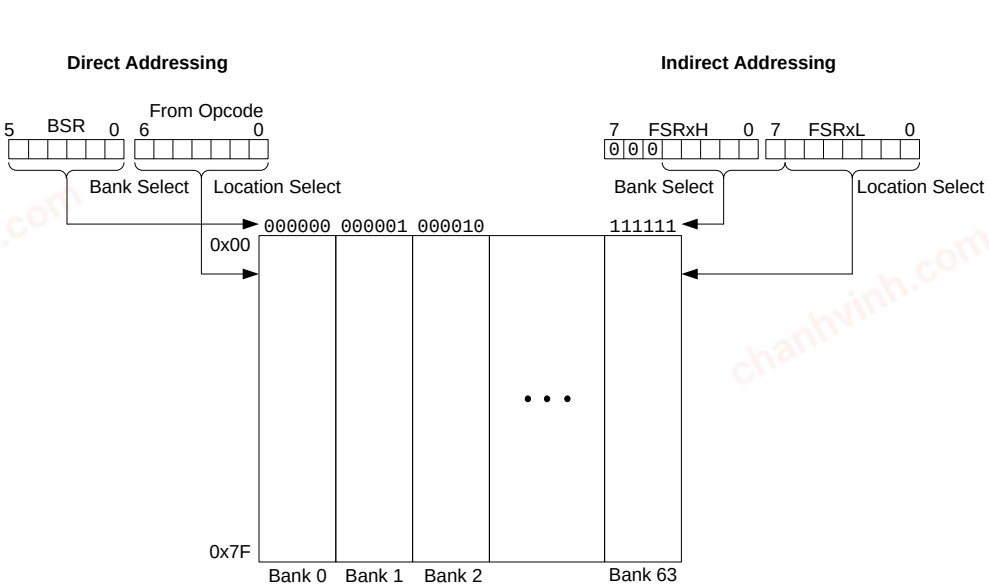
**Figure 9-18. Indirect Addressing**




## 9.6.1 Traditional/Banked Data Memory

The traditional or banked data memory is a region from FSR address 0x0000 to FSR address 0x1FFF. The addresses correspond to the absolute addresses of all SFR, GPR and common registers.

**Figure 9-19. Traditional/Banked Data Memory Map**



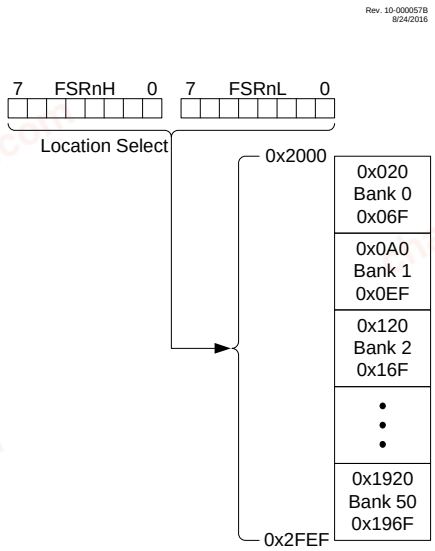
chanhvinh.com



### 9.6.2 Linear Data Memory

The linear data memory is the region from FSR address 0x2000 to FSR address 0x2FEF. This region is a virtual region that points back to the 80-byte blocks of GPR memory in all the banks. Refer to Figure 9-20 for the Linear Data Memory Map.

**Figure 9-20. Linear Data Memory Map**




**Important:** The address range 0x2000 to 0x2FEF represents the complete addressable Linear Data Memory for PIC® devices (up to Bank 50). The actual implemented Linear Data Memory will differ from one device to the other in a family.

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.

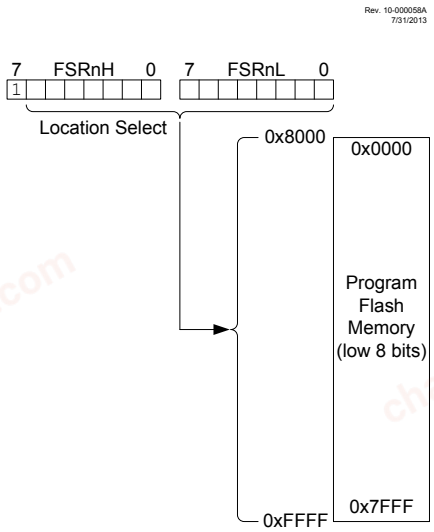
### 9.6.3 Program Flash Memory

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 are 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.

chanhvinh.com



**Figure 9-21. Program Flash Memory Map**




### 9.6.4 Data EEPROM Memory

The EEPROM memory can be read or written through the NVMCON register interface. However, to make access to the EEPROM easier, read-only access to the EEPROM contents are also available through indirect addressing via an FSR. When the MSP of the FSR (ex: FSRxH) is set to 0x70, the lower 8-bit address value (in FSRxL) determines the EEPROM location that may be read via the INDF register). In other words, the EEPROM address range 0x00-0xFF is mapped into the FSR address space between 0x7000 and 0x70FF. Writing to the EEPROM cannot be accomplished via the FSR/INDF interface. Reads from the EEPROM through the FSR/INDF interface will require one additional instruction cycle to complete.

## 9.7 Register Definitions: Memory Organization

© 2023 Microchip Technology Inc.
and its subsidiaries

chanhvinh.com



### 9.7.1 INDF0

**Name:** INDF0
**Offset:** 0x0000

Indirect Data Register. This is a virtual register. The GPR/SFR register addressed by the FSR0 register is the target for all operations involving the INDF0 register.

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | INDF0[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 7:0 – INDF0[7:0]**
Indirect data pointed to by the FSR0 register

chanhvinh.com



### 9.7.2 INDF1

**Name:** INDF1
**Offset:** 0x0001

Indirect Data Register. This is a virtual register. The GPR/SFR register addressed by the FSR1 register is the target for all operations involving the INDF1 register.

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | INDF1[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 7:0 – INDF1[7:0]**
Indirect data pointed to by the FSR1 register

chanhvinh.com



### 9.7.3 PCL

**Name:** PCL
**Offset:** 0x0002

Low byte of the Program Counter

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PCL[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 7:0 – PCL[7:0]**
Provides direct read and write access to the Program Counter

chanhvinh.com

chanhvinh.com



### 9.7.4 STATUS

**Name:** STATUS
**Offset:** 0x0003

Status Register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | $\overline{TO}$ | $\overline{PD}$ | Z | DC | C |
| Access | | | | R | R | R/W | R/W | R/W |
| Reset | | | | 1 | 1 | 0 | 0 | 0 |

**Bit 4 – $\overline{TO}$** Time-Out

Reset States: POR/BOR = 1
All Other Resets = q

| Value | Description |
|---|---|
| 1 | Set at power-up or by execution of CLRWDT or SLEEP instruction |
| 0 | A WDT time-out occurred |

**Bit 3 – $\overline{PD}$** Power-Down

Reset States: POR/BOR = 1
All Other Resets = q

| Value | Description |
|---|---|
| 1 | Set at power-up or by execution of CLRWDT instruction |
| 0 | Cleared by execution of the SLEEP instruction |

**Bit 2 – Z** Zero

Reset States: POR/BOR = 0
All Other Resets = u

| Value | Description |
|---|---|
| 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/$\overline{Borrow}$[1]

ADDWF, ADDLW, SUBLW, SUBWF instructions

Reset States: POR/BOR = 0
All Other Resets = u

| Value | Description |
|---|---|
| 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/$\overline{Borrow}$[1]

ADDWF, ADDLW, SUBLW, SUBWF instructions

Reset States: POR/BOR = 0
All Other Resets = u

| Value | Description |
|---|---|
| 1 | A carry-out from the Most Significant bit of the result occurred |
| 0 | No carry-out from the Most Significant bit of the result occurred |

**Note:**

1. For $\overline{Borrow}$, the polarity is reversed. A subtraction is executed by adding the two's complement of the second operand. For Rotate (RRCF, RLCF) instructions, this bit is loaded with either the high or low-order bit of the Source register.

chanhvinh.com



### 9.7.5 FSR0

**Name:** FSR0
**Offset:** 0x0004

Indirect Address Register

The FSR0 value is the address of the data to which the INDF0 register points.

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | FSR0[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | FSR0[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 15:0 – FSR0[15:0]** Address of INDF0 data

**Notes:** The individual bytes in this multibyte register can be accessed with the following register names:

1. FSR0H: Accesses the high byte FSR0[15:8].
2. FSR0L: Accesses the low byte FSR0[7:0].

© 2023 Microchip Technology Inc.
and its subsidiaries

chanhvinh.com



### 9.7.6 FSR1

**Name:** FSR1
**Offset:** 0x0006

Indirect Address Register

The FSR1 value is the address of the data to which the INDF1 register points.

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | FSR1[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | FSR1[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 15:0 – FSR1[15:0]**
Address of INDF1 data

**Notes:** The individual bytes in this multibyte register can be accessed with the following register names:

1. FSR1H: Accesses the high byte FSR1[15:8].
2. FSR1L: Accesses the low byte FSR1[7:0].

© 2023 Microchip Technology Inc.
and its subsidiaries

chanhvinh.com



### 9.7.7 BSR

**Name:** BSR
**Offset:** 0x0008

Bank Select Register
The BSR indicates the data memory bank by writing the bank number into the register. All data memory can be accessed directly via instructions, or indirectly via FSRs.

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | BSR[5:0] | | | | | |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 5:0 – BSR[5:0]**
Six Most Significant bits of the data memory address

chanhvinh.com

chanhvinh.com



### 9.7.8 WREG

**Name:** WREG
**Offset:** 0x0009

Working Data Register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | WREG[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 7:0 – WREG[7:0]**

chanhvinh.com

chanhvinh.com



### 9.7.9 PCLATH

**Name:** PCLATH
**Offset:** 0x000A

Program Counter Latches

Write Buffer for the upper seven bits of the Program Counter

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | PCLATH[6:0] | | | | | | |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 6:0 – PCLATH[6:0]** High PC Latch Register
Holding register for Program Counter bits [6:0]

chanhvinh.com

chanhvinh.com



## 9.8 Register Summary - Memory Organization

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | INDF0 | 7:0 | INDF0[7:0] | | | | | | | |
| 0x01 | INDF1 | 7:0 | INDF1[7:0] | | | | | | | |
| 0x02 | PCL | 7:0 | PCL[7:0] | | | | | | | |
| 0x03 | STATUS | 7:0 | | | | $\overline{TO}$ | $\overline{PD}$ | Z | DC | C |
| 0x04 | FSR0 | 7:0 | FSR0[7:0] | | | | | | | |
| | | 15:8 | FSR0[15:8] | | | | | | | |
| 0x06 | FSR1 | 7:0 | FSR1[7:0] | | | | | | | |
| | | 15:8 | FSR1[15:8] | | | | | | | |
| 0x08 | BSR | 7:0 | | | BSR[5:0] | | | | | |
| 0x09 | WREG | 7:0 | WREG[7:0] | | | | | | | |
| 0x0A | PCLATH | 7:0 | | PCLATH[6:0] | | | | | | |

© 2023 Microchip Technology Inc.
and its subsidiaries

chanhvinh.com



# 10. Resets

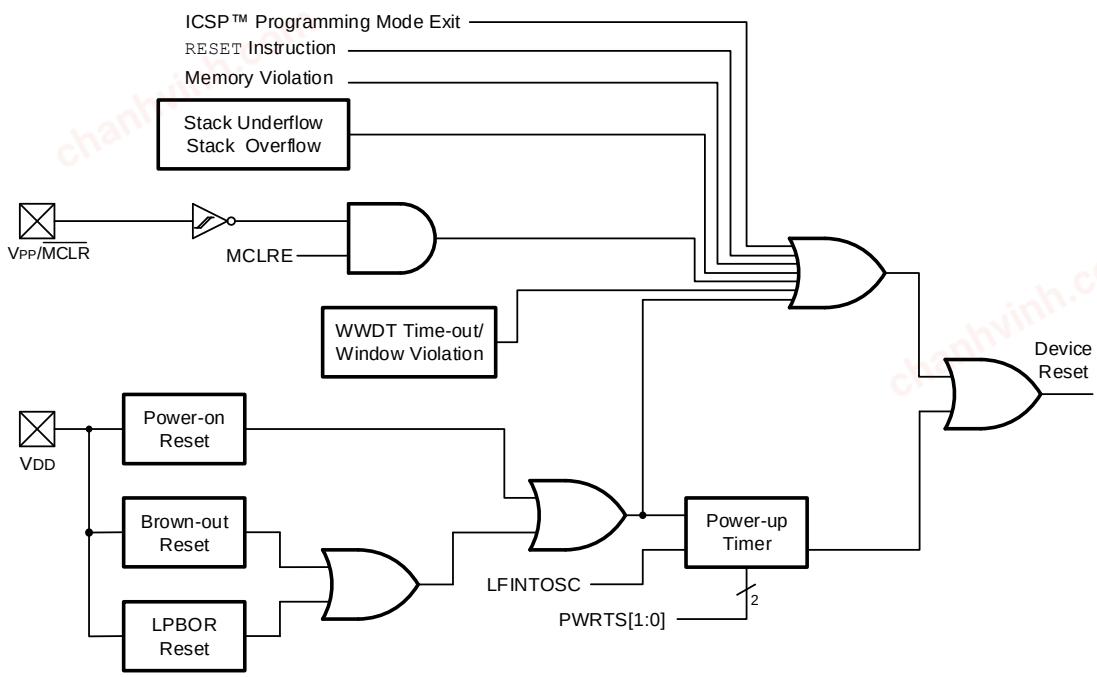
There are multiple ways to reset this device:

- Power-on Reset (POR)
- Brown-out Reset (BOR)
- $\overline{\text{MCLR}}$ Reset
- WDT Reset
- `RESET` instruction
- Stack Overflow
- Stack Underflow
- Programming mode exit

To allow $V_{DD}$ 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 10-1.

**Figure 10-1. Simplified Block Diagram of On-Chip Reset Circuit**




## 10.1 Power-on Reset (POR)

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

### 10.1.1 Programming Mode Exit

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

chanhvinh.com



## 10.2 Brown-out Reset (BOR)

The BOR circuit holds the device in Reset when $V_{DD}$ 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 bits. 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 10-1 for more information.

The Brown-out Reset voltage level is selectable by configuring the BORV bits.

A $V_{DD}$ noise rejection filter prevents the BOR from triggering on small events. If $V_{DD}$ falls below $V_{BOR}$ for a duration greater than parameter $T_{BORDC}$, the device will reset and the $\overline{BOR}$ bit will be cleared, indicating the Brown-out Reset condition occurred. See Figure 10-2.

### 10.2.1 BOR Is Always On

When the BOREN bits are programmed to '11', the BOR is always on. The device start-up will be delayed until the BOR is ready and $V_{DD}$ is higher than the BOR threshold.

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

### 10.2.2 BOR Is Off in Sleep

When the BOREN bits are programmed to '10', the BOR is on, except in Sleep. BOR protection is not active during Sleep, but device wake-up will be delayed until the BOR can determine that $V_{DD}$ is higher than the BOR threshold. The device wake-up will be delayed until the BOR is ready.

### 10.2.3 BOR Controlled by Software

When the BOREN bits of Configuration Words are programmed to '01', the BOR is controlled by the SBOREN bit. The device start-up is not delayed by the BOR Ready condition or the $V_{DD}$ level.

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

BOR protection is unchanged by Sleep.

**Table 10-1. BOR Operating Modes**

| BOREN | SBOREN | Device Mode | BOR Mode | Instruction Execution upon: | |
|---|---|---|---|---|---|
| | | | | Release of POR | Wake-up from Sleep |
| 11(1) | x | X | Active | Wait for release of BOR (BORRDY = 1) | Begins immediately |
| 10 | x | Awake | Active | Wait for release of BOR (BORRDY = 1) | N/A |
| | | Sleep | Hibernate | N/A | Wait for release of BOR (BORRDY = 1) |
| 01 | 1 | X | Active | Wait for release of BOR (BORRDY = 1) | Begins immediately |
| | 0 | X | Hibernate | | |
| 00 | x | X | Disabled | Begins immediately | |

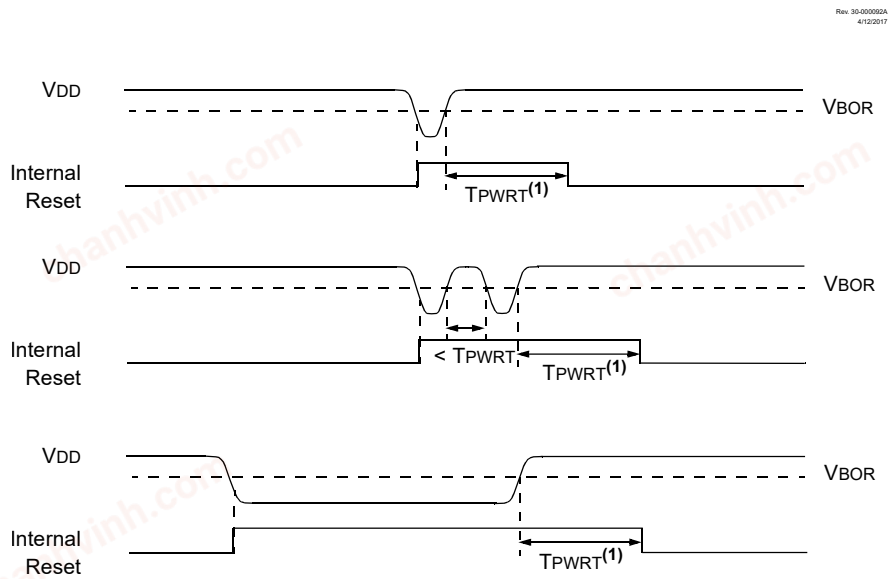
chanhvinh.com



**Note:**

1. In this specific case, '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 bits.

**Figure 10-2. Brown-Out Situations**




**Note:** $T_{PWRT}$ delay when the PWRTS bits are enabled ($\overline{PWRTS}$ != 00).

### 10.2.4 BOR Is Always Off

When the BOREN bits are programmed to '00', the BOR is always disabled. In the configuration, setting the SBOREN bit will have no affect on BOR operations.

## 10.3 $\overline{MCLR}$ Reset

The $\overline{MCLR}$ is an optional external input that can reset the device. The $\overline{MCLR}$ function is controlled by the MCLRE bit and the LVP bit (see Table 10-2). The $\overline{RMCLR}$ bit will be set to '0' if a $\overline{MCLR}$ has occurred.

**Table 10-2. $\overline{MCLR}$ Configuration**

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

### 10.3.1 $\overline{MCLR}$ Enabled

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

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

**Important:** An internal Reset event (RESET instruction, BOR, WDT, POR, STKOVF, STKUNF) does not drive the $\overline{MCLR}$ pin low.

chanhvinh.com



### 10.3.2 $\overline{\text{MCLR}}$ Disabled

When $\overline{\text{MCLR}}$ is disabled, the $\overline{\text{MCLR}}$ becomes input-only and pin functions such as internal weak pull-ups are under software control.

## 10.4 Watchdog Timer (WDT) Reset

The Watchdog Timer generates a Reset if the firmware does not issue a `CLRWDT` instruction within the time-out period. The TO, PD and $\overline{\text{RWDT}}$ bits are changed to indicate a WDT Reset caused by the timer overflowing.

## 10.5 `RESET` Instruction

A `RESET` instruction will cause a device Reset. The $\overline{\text{RI}}$ bit will be set to '0'. See Table 10-4 for default conditions after a `RESET` instruction has occurred.

## 10.6 Stack Overflow/Underflow Reset

The device can reset when the Stack Overflows or Underflows. The STKOVF or STKUNF bits indicate the Reset condition. These Resets are enabled by setting the STVREN bit.

## 10.7 Power-Up Timer (PWRT)

The Power-up Timer provides up to a 64 ms time-out period on POR or BOR. The device is held in Reset as long as PWRT is active. The PWRT delay allows additional time for the $V_{DD}$ to rise to an acceptable level.

The Power-up Timer is controlled by the PWRTS bits. The Power-up Timer starts after the release of the POR and BOR. For additional information, refer to the "Power-up Trouble Shooting" Application Note AN607.

## 10.8 Start-Up Sequence

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

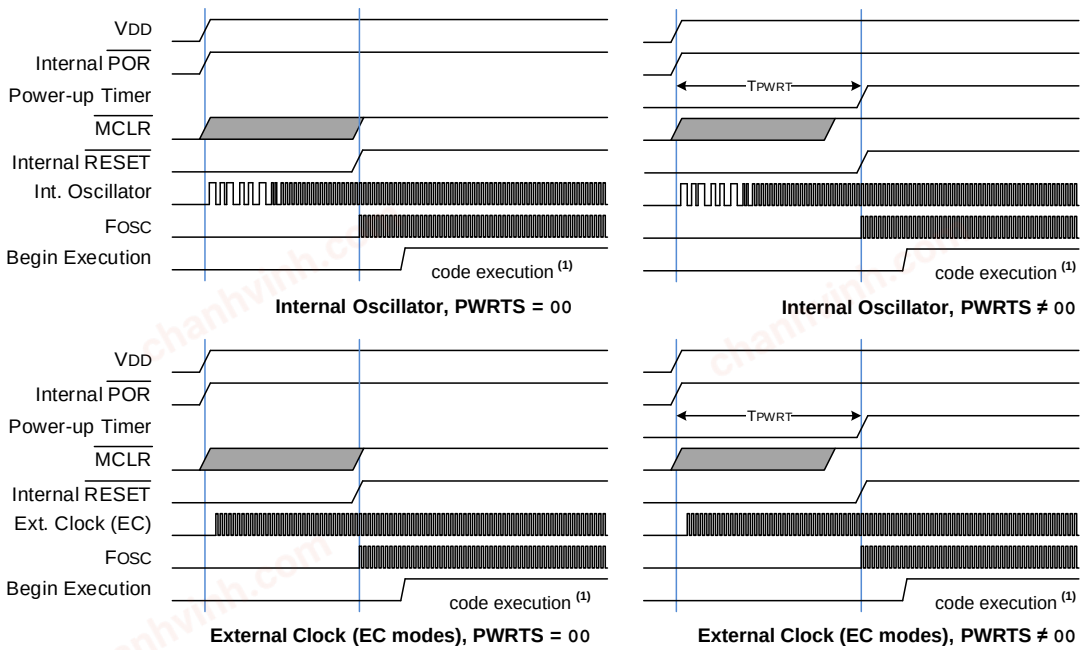
1. Power-up Timer runs to completion (if enabled).
2. $\overline{\text{MCLR}}$ must be released (if enabled).

The 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 10-3). This is useful for testing purposes or for synchronizing more than one device operating in parallel.

chanhvinh.com



**Figure 10-3. Reset Start-Up Sequence**




**Note:**

1. Code execution begins 10 $F_{OSC}$ cycles after the $F_{OSC}$ clock is released.

## 10.9 Memory Execution Violation

A memory execution violation Reset occurs if executing an instruction being fetched from outside the valid execution area. The invalid execution areas are:

1. Addresses outside implemented program memory. Refer to the **"Memory Organization"** chapter for details about available Flash size.
2. Storage Area Flash (SAF) inside program memory, if enabled.

When a memory execution violation is generated, the device is reset and the $\overline{MEMV}$ bit is cleared to signal the cause of the Reset. The $\overline{MEMV}$ bit must be set in the user code after a memory execution violation Reset has occurred to detect further violation Resets.

## 10.10 Determining the Cause of a Reset

Upon any Reset, multiple bits in the STATUS, PCON0 and PCON1 registers are updated to indicate the cause of the Reset. The following tables show the Reset conditions of these registers.

**Table 10-3. Reset Status Bits and Their Significance**

| STKOVF | STKUNF | $\overline{RWDT}$ | $\overline{RMCLR}$ | $\overline{RI}$ | $\overline{POR}$ | $\overline{BOR}$ | $\overline{TO}$ | $\overline{PD}$ | $\overline{MEMV}$ | Condition |
|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 1 | 1 | 1 | 0 | x | 1 | 1 | 1 | Power-on Reset |
| 0 | 0 | 1 | 1 | 1 | 0 | x | 0 | x | u | Illegal, $\overline{TO}$ is set on POR |
| 0 | 0 | 1 | 1 | 1 | 0 | x | x | 0 | u | Illegal, $\overline{PD}$ is set on POR |

chanhvinh.com



| .....continued | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| STKOVF | STKUNF | RWDT | RMCLR | RI | POR | BOR | TO | PD | MEMV | Condition |
| 0 | 0 | u | 1 | 1 | u | 0 | 1 | 1 | u | Brown-out Reset |
| u | u | 0 | u | u | u | u | 0 | u | u | WDT Reset |
| u | u | u | u | u | u | u | 0 | 0 | u | WDT Wake-up from Sleep |
| u | u | u | u | u | u | u | 1 | 0 | u | Interrupt Wake-up from Sleep |
| u | u | u | 0 | u | u | u | u | u | 1 | MCLR Reset during normal operation |
| u | u | u | 0 | u | u | u | 1 | 0 | u | MCLR Reset during Sleep |
| u | u | u | u | 0 | u | u | u | u | u | RESET Instruction Executed |
| 1 | u | u | u | u | u | u | u | u | u | Stack Overflow Reset (STVREN = 1) |
| u | 1 | u | u | u | u | u | u | u | u | Stack Underflow Reset (STVREN = 1) |
| u | u | u | u | u | u | u | u | u | 0 | Memory Violation Reset |

**Table 10-4. Reset Conditions for Special Registers**

| Condition | Program Counter | STATUS Register | PCON0 Register | PCON1 Register |
|---|---|---|---|---|
| Power-on Reset | 0 | ---1 1000 | 0011 110x | ---- --1- |
| Brown-out Reset | 0 | ---1 1000 | 0011 11u0 | ---- --u- |
| MCLR Reset during normal operation | 0 | -uuu uuuu | uuuu 0uuu | ---- --1- |
| MCLR Reset during Sleep | 0 | ---1 0uuu | uuuu 0uuu | ---- --u- |
| WDT Time-out Reset | 0 | ---0 uuuu | uuu0 uuuu | ---- --u- |
| WDT Wake-up from Sleep | PC + 1 | ---0 0uuu | uuuu uuuu | ---- --u- |
| Interrupt Wake-up from Sleep | PC + 1(1) | ---1 0uuu | uuuu uuuu | ---- --u- |
| RESET Instruction Executed | 0 | ---u uuuu | uuuu u0uu | ---- --u- |
| Stack Overflow Reset (STVREN = 1) | 0 | ---u uuuu | 1uuu uuuu | ---- --u- |
| Stack Underflow Reset (STVREN = 1) | 0 | ---u uuuu | u1uu uuuu | ---- --u- |
| Memory Violation Reset | 0 | -uuu uuuu | uuuu uuuu | ---- --0- |

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

**Note:**

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

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



## 10.11 Power Control (PCONx) Register

The Power Control (PCONx) registers contain flag bits to differentiate between a:

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

Hardware will change the corresponding register bit during the Reset process; if the Reset was not caused by the condition, the bit remains unchanged.

Software may reset the bit to the Inactive state after restart (hardware will not reset the bit).

Software may also set any PCONx bit to the Active state, so that user code may be tested, but no Reset action will be generated.

## 10.12 Register Definitions: Power Control

chanhvinh.com



### 10.12.1 BORCON

**Name:** BORCON
**Offset:** 0x0191

Brown-out Reset Control Register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SBOREN | | | | | | | BORRDY |
| Access | R/W | | | | | | | R |
| Reset | 1 | | | | | | | q |

**Bit 7 – SBOREN** Software Brown-Out Reset Enable
Reset States: POR/BOR = 1
All Other Resets = u

| Value | Condition | Description |
|---|---|---|
| – | If BOREN ≠ 01 | SBOREN is read/write, but has no effect on the BOR |
| 1 | If BOREN = 01 | BOR Enabled |
| 0 | If BOREN = 01 | BOR Disabled |

**Bit 0 – BORRDY** Brown-Out Reset Circuit Ready Status
Reset States: POR/BOR = q
All Other Resets = u

| Value | Description |
|---|---|
| 1 | The Brown-out Reset circuit is active and armed |
| 0 | The Brown-out Reset circuit is disabled or is warming up |

© 2023 Microchip Technology Inc.
and its subsidiaries

chanhvinh.com



### 10.12.2 PCON0

**Name:** PCON0
**Offset:** 0x0192

Power Control Register 0

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | STKOVF | STKUNF | | $\overline{RWDT}$ | $\overline{RMCLR}$ | $\overline{RI}$ | $\overline{POR}$ | $\overline{BOR}$ |
| Access | R/W/HS | R/W/HS | | R/W/HC | R/W/HC | R/W/HC | R/W/HC | R/W/HC |
| Reset | 0 | 0 | | 1 | 1 | 1 | 0 | q |

**Bit 7 – STKOVF** Stack Overflow Flag
Reset States: POR/BOR = 0
All Other Resets = q

| Value | Description |
|---|---|
| 1 | A Stack Overflow occurred (more CALLs than fit on the stack) |
| 0 | A Stack Overflow has not occurred or set to '0' by firmware |

**Bit 6 – STKUNF** Stack Underflow Flag
Reset States: POR/BOR = 0
All Other Resets = q

| Value | Description |
|---|---|
| 1 | A Stack Underflow occurred (more RETURNs than CALLs) |
| 0 | A Stack Underflow has not occurred or set to '0' by firmware |

**Bit 4 – $\overline{RWDT}$** WDT Reset Flag
Reset States: POR/BOR = 1
All Other Resets = q

| Value | Description |
|---|---|
| 1 | A WDT Overflow/Time-out Reset has not occurred or set to '1' by firmware |
| 0 | A WDT Overflow/Time-out Reset has occurred (set to '0' in hardware when a WDT Reset occurs) |

**Bit 3 – $\overline{RMCLR}$** $\overline{MCLR}$ Reset Flag
Reset States: POR/BOR = 1
All Other Resets = q

| Value | Description |
|---|---|
| 1 | A $\overline{MCLR}$ Reset has not occurred or set to '1' by firmware |
| 0 | A $\overline{MCLR}$ Reset has occurred (set to '0' in hardware when a $\overline{MCLR}$ Reset occurs) |

**Bit 2 – $\overline{RI}$** RESET Instruction Flag
Reset States: POR/BOR = 1
All Other Resets = q

| Value | Description |
|---|---|
| 1 | A RESET instruction has not been executed or set to '1' by firmware |
| 0 | A RESET instruction has been executed (set to '0' in hardware upon executing a RESET instruction) |

**Bit 1 – $\overline{POR}$** Power-on Reset Status
Reset States: POR/BOR = 0
All Other Resets = u

| Value | Description |
|---|---|
| 1 | No Power-on Reset occurred or set to '1' by firmware |
| 0 | A Power-on Reset occurred (set to '0' in hardware when a Power-on Reset occurs) |

**Bit 0 – $\overline{BOR}$** Brown-out Reset Status
Reset States: POR/BOR = q

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



All Other Resets = u

| Value | Description |
|---|---|
| 1 | No Brown-out Reset occurred or set to '1' by firmware |
| 0 | A Brown-out Reset occurred (set to '0' in hardware when a Brown-out Reset occurs) |

chanhvinh.com

chanhvinh.com



### 10.12.3 PCON1

**Name:** PCON1
**Offset:** 0x0193

Power Control Register 1

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | $\overline{\text{MEMV}}$ | |
| Access | | | | | | | R/W/HC | |
| Reset | | | | | | | 1 | |

**Bit 1 – $\overline{\text{MEMV}}$** Memory Violation Flag
Reset States: POR/BOR = 1
All Other Resets = u

| Value | Description |
|---|---|
| 1 | No Memory Violation Reset occurred or set to '1' by firmware |
| 0 | A Memory Violation Reset occurred (set to '0' in hardware when a Memory Violation occurs) |

chanhvinh.com

chanhvinh.com



## 10.13 Register Summary - Power Control

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00<br>...<br>0x0190 | Reserved | | | | | | | | | |
| 0x0191 | BORCON | 7:0 | SBOREN | | | | | | | BORRDY |
| 0x0192 | PCON0 | 7:0 | STKOVF | STKUNF | | $\overline{\text{RWDT}}$ | $\overline{\text{RMCLR}}$ | $\overline{\text{RI}}$ | $\overline{\text{POR}}$ | $\overline{\text{BOR}}$ |
| 0x0193 | PCON1 | 7:0 | | | | | | | $\overline{\text{MEMV}}$ | |

chanhvinh.com

chanhvinh.com



# 11. OSC - Oscillator Module (With Fail-Safe Clock Monitor)

The oscillator module contains multiple clock sources and selection features that allow it to be used in a wide range of applications while maximizing performance and minimizing power consumption.

Clock sources can be supplied either internally or externally. External sources include:

- External clock oscillators
- Quartz crystal resonators
- Ceramic resonators
- Secondary Oscillator (SOSC)

Internal sources include:

- High-Frequency Internal Oscillator (HFINTOSC)
- Low-Frequency Internal Oscillator (LFINTOSC)
- Analog-to-Digital Converter RC Oscillator (ADCRC)

Special features of the oscillator module include:

- Oscillator Start-up Timer (OST): Ensures stability of quartz crystal or ceramic resonators.
- 4x Phase-Locked Loop (PLL): Frequency multiplier for external clock sources.
- HFINTOSC Frequency Adjustment: Provides the ability to adjust the HFINTOSC frequency.
- Clock switching: Allows the system clock to switch between internal or external sources via software during run time.
- Fail-Safe Clock Monitor (FSCM): Designed to detect a failure of the external clock source and automatically switch to an internal clock source.

The Reset Oscillator (RSTOSC) Configuration bits determine the type of oscillator that will be used when the device runs after a Reset, including when the device is first powered up (see the table below).

**Table 11-1. RSTOSC Selection Table**

| RSTOSC | SFR Reset Values | | | Clock Source |
|---|---|---|---|---|
| | NOSC / COSC | NDIV / CDIV | OSCFRQ | |
| 111 | 111 | 0000 (1:1) | 010 (4 MHz) | EXTOSC per FEXTOSC |
| 110 | 110 | 0010 (4:1) | | HFINTOSC @ 1 MHz |
| 101 | 101 | 0000 (1:1) | | LFINTOSC |
| 100 | 100 | 0000 (1:1) | | SOSC |
| 011 | Reserved | | | |
| 010 | 010 | 0000 (1:1) | 010 (4 MHz) | EXTOSC + 4x PLL(1) |
| 001 | Reserved | | | |
| 000 | 110 | 0000 (1:1) | 101 (32 MHz) | HFINTOSC @ 32 MHz |

**Note:**

1. EXTOSC must meet the PLL specifications (see the data sheet Electrical Specifications).

If an external clock source is selected by the RSTOSC bits, the External Oscillator Mode Select (FEXTOSC) Configuration bits must be used to select the External Clock mode. These modes include:

- ECL: External Clock Low Power mode
- ECH: External Clock High Power mode
- LP: 32 kHz Low-Gain Crystal mode
- XT: Medium-Gain Crystal or Ceramic Resonator mode
- HS: High-Gain Crystal or Ceramic Resonator mode

© 2023 Microchip Technology Inc. and its subsidiaries

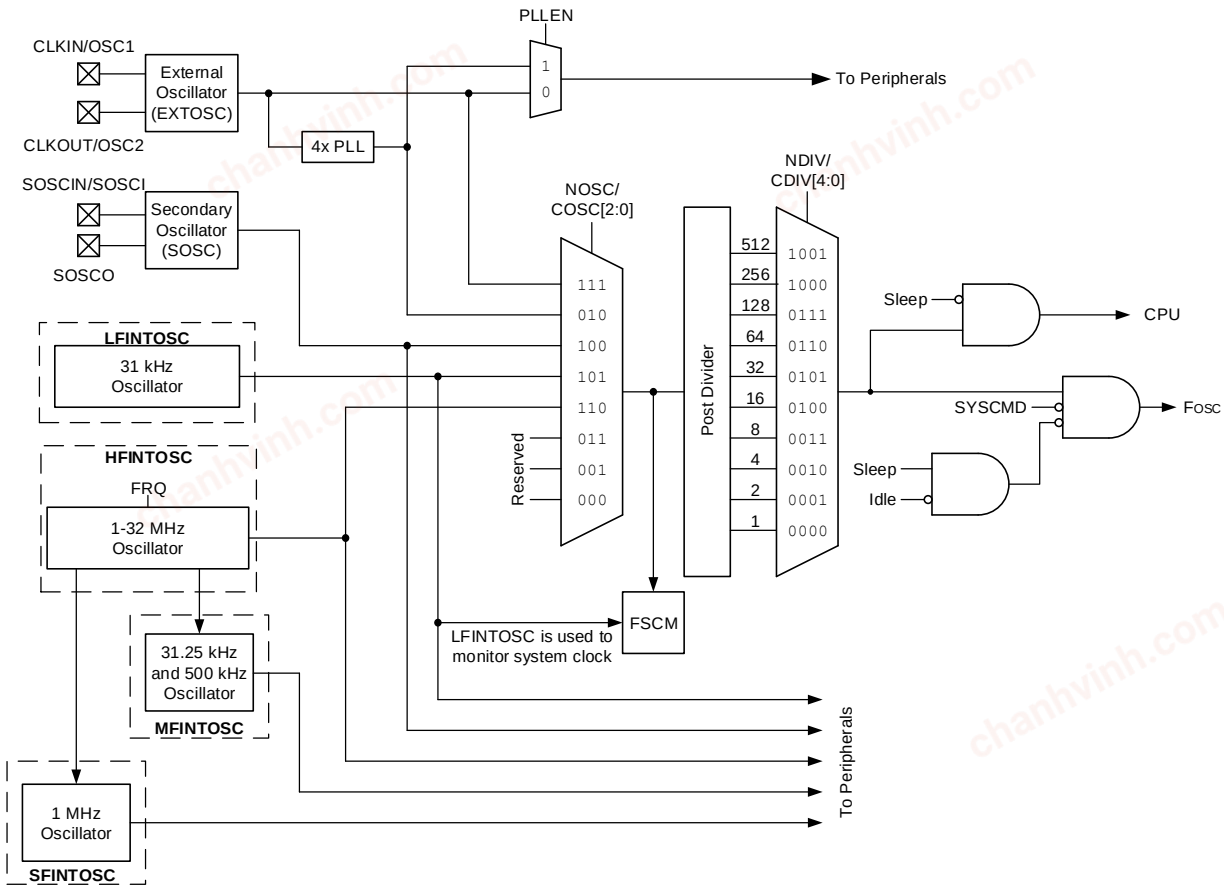
chanhvinh.com



The ECH and ECL modes rely on an external logic-level signal as the device clock source. The LP, XT and HS modes rely on an external quartz crystal or ceramic resonator as the device clock source. Each mode is optimized for a specific frequency range. The internal oscillator block produces both low-frequency and high-frequency clock signals, designated LFINTOSC and HFINTOSC, respectively. Multiple system operating frequencies may be derived from these clock sources.

The figure below illustrates a block diagram of the oscillator module.

**Figure 11-1. Clock Source Block Diagram**




## 11.1 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 of external clock sources include:

- Digital oscillator modules
- Quartz crystal resonators
- Ceramic resonators

A 4x PLL is provided for use with external clock sources.

Internal clock sources are contained within the oscillator module. The internal oscillator block features two internal oscillators that are used to generate internal system clock sources. The High-Frequency Internal Oscillator (HFINTOSC) can produce a wide range of frequencies which are determined via the HFINTOSC Frequency Selection (OSCFRQ) register. The Low-Frequency Internal Oscillator (LFINTOSC) generates a fixed nominal 31 kHz clock signal. The internal oscillator block also features an RC oscillator which is dedicated to the Analog-to-Digital Converter (ADC).

chanhvinh.com



The oscillator module allows the system clock source or system clock frequency to be changed through clock switching. Clock source selections are made via the New Oscillator Source Request (NOSC) bits. Once the clock source has been selected, the clock source base frequency can be divided (post-scaled) via the New Divider Selection Request (NDIV) bits.

The instruction clock ($F_{OSC}/4$) can be routed to the OSC2/CLKOUT pin when the pin is not in use. The Clock Out Enable ($\overline{CLKOUTEN}$) Configuration bit controls the functionality of the CLKOUT signal. When $\overline{CLKOUTEN}$ is clear ($\overline{CLKOUTEN}$ = 0), the CLKOUT signal is routed to the OSC2/CLKOUT pin. When $\overline{CLKOUTEN}$ is set ($\overline{CLKOUTEN}$ = 1), the OSC2/CLKOUT pin functions as an I/O pin.

### 11.1.1 External Clock Sources

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

- Program the RSTOSC and FEXTOSC Configuration bits to select an external clock source that will be used as the default system clock upon a device Reset.
- Write the NOSC and NDIV bits to switch the system clock source during run time.

#### 11.1.1.1 EC Mode

The External Clock (EC) mode allows an externally generated logic level signal to be the system clock source. When operating in EC mode, an external clock source is connected to the OSC1/CLKIN input pin. The OSC2/CLKOUT pin is available as a general purpose I/O pin or as the CLKOUT signal pin.

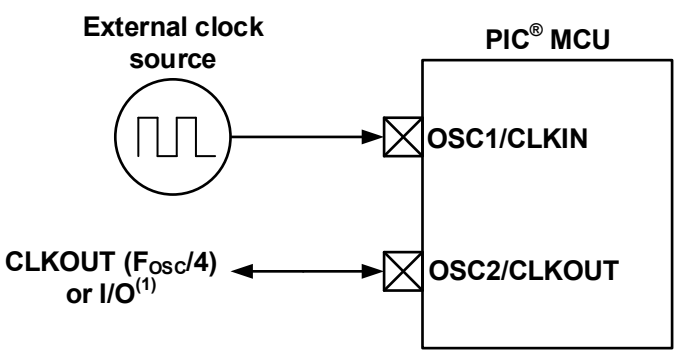
EC mode provides two Power mode selections:

- ECH: High Power mode
- ECL: Low Power mode

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

The figure below shows the pin connections for EC mode.

**Figure 11-2. External Clock (EC) Mode Operation**




**Note:**

1. Output depends on the setting of the $\overline{CLKOUTEN}$ Configuration bit.

#### 11.1.1.2 LP, XT, HS Modes

The LP, XT and HS modes support the use of quartz crystals or ceramic resonators connected to the OSC1 and OSC2 pins, as shown in the figures below. These three modes select a low, medium, or high-gain setting of the internal inverter-amplifier to support various resonator types and speeds.

The LP Oscillator mode selects the lowest gain setting of the internal inverter-amplifier, and consumes the least amount of current. LP mode is designed to drive 32.768 kHz tuning-fork type crystals (watch crystals), but can operate up to 100 kHz.

chanhvinh.com

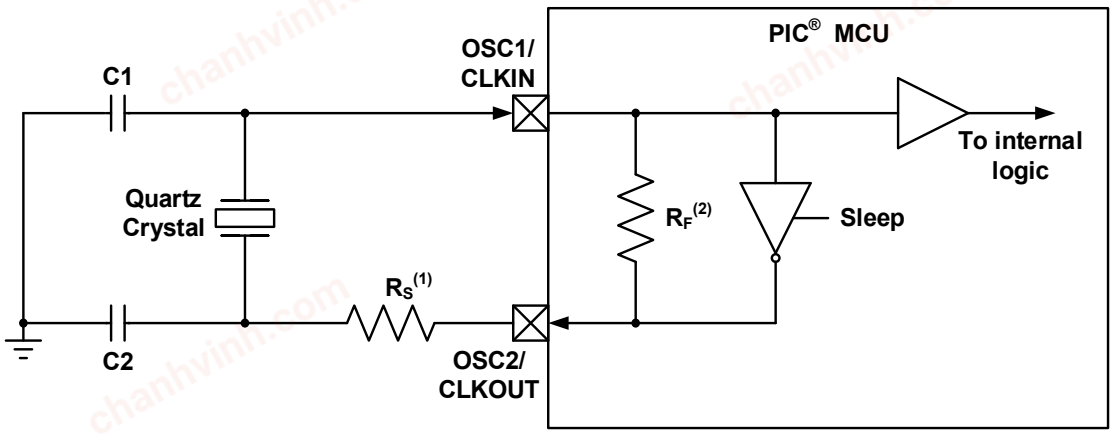


The XT Oscillator mode selects the intermediate gain setting of the internal inverter-amplifier. Current consumption is at a medium level when compared to the other two modes. XT mode is best suited to drive crystal and ceramic resonators with a frequency range up to 4 MHz.

The HS Oscillator mode selects the highest gain setting of the internal inverter-amplifier, and consumes the most current. This mode is best suited for crystal and ceramic resonators that require operating frequencies up to 20 MHz.

The figures below show typical circuits for quartz crystal and ceramic resonators.

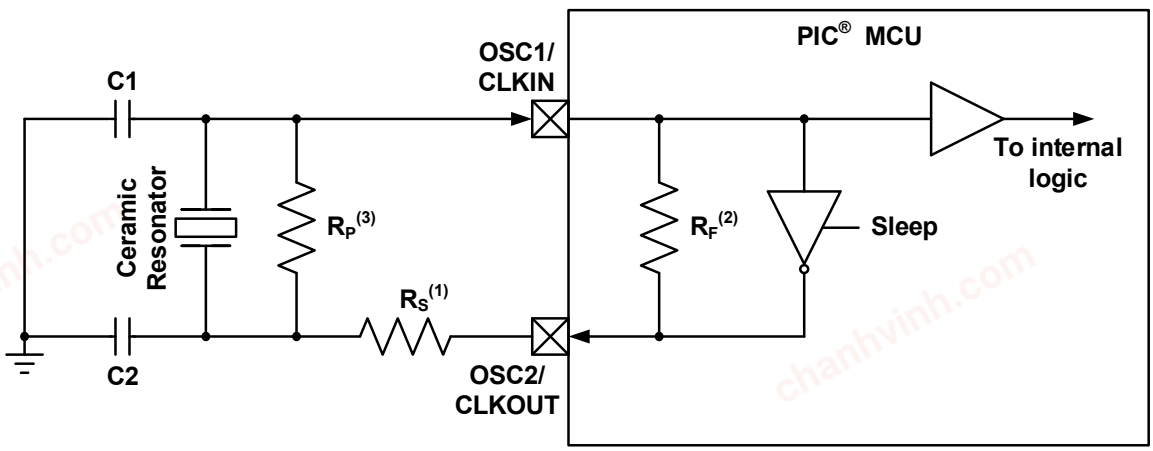
**Figure 11-3. Quartz Crystal Operation**




**Notes:**

1. A series resistor ($R_S$) may be required for quartz crystals with low drive level.
2. The value of $R_F$ varies with the Oscillator mode selected (typically between 2 MΩ and 10 MΩ).

**Figure 11-4. Ceramic Resonator Operation**



chanhvinh.com



**Notes:**

1. A series resistor ($R_S$) may be required for ceramic resonators with low drive level.
2. The value of $R_F$ varies with the Oscillator mode selected (typically between 2 MΩ and 10 MΩ).
3. An additional parallel feedback resistor ($R_P$) may be required for proper ceramic resonator operation.

#### 11.1.1.3 Oscillator Start-Up Timer (OST)

The Oscillator Start-up Timer (OST) ensures that the oscillator circuit has started and is providing a stable system clock to the oscillator module. Quartz crystals or ceramic resonators do not start immediately, and may take a few hundred cycles before the oscillator becomes stable. The oscillations must build up until sufficient amplitude is generated to properly toggle between logic states. The OST counts 1024 oscillation periods from the OSC1 input following a Power-on Reset (POR), Brown-out Reset (BOR), or wake-up from Sleep event to ensure that the oscillator has enough time to reach stable and accurate operation. Once the OST has completed its count, module hardware sets the External Oscillator Ready (EXTOR) bit, indicating that the oscillator is stable and ready to use.

#### 11.1.1.4 4x PLL

The oscillator module contains a 4x Phase-Locked Loop (PLL) circuit that can be used with the external clock sources to provide a system clock source. The input frequency for the PLL must fall within a specified range. See the PLL Clock Timing Specifications found in the **"Electrical Specifications"** chapter for more information.

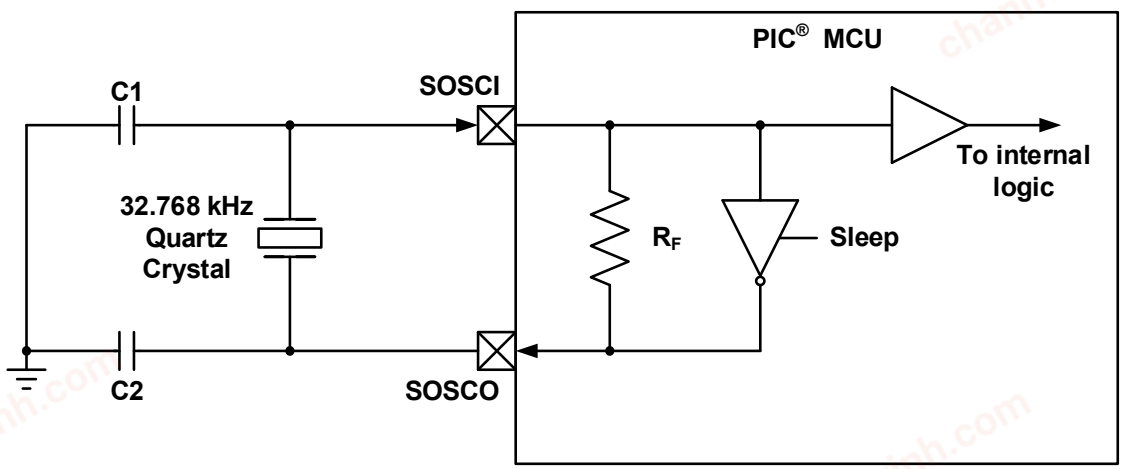
The PLL can be enabled for use through one of two methods:

1. Program the RSTOSC Configuration bits to select the 'EXTOSC with 4x PLL' option.
2. Write the NOSC bits to select the 'EXTOSC with 4x PLL' option.

#### 11.1.1.5 Secondary Oscillator

The Secondary Oscillator (SOSC) is a separate external oscillator block that can be used as an alternate system clock source or as a Timer clock source. The SOSC is optimized for 32.768 kHz, and can be used with either an external quartz crystal connected to the SOSCI and SOSCO pins, or with an external clock source connected to the SOSCI pin, as shown in the figures below.

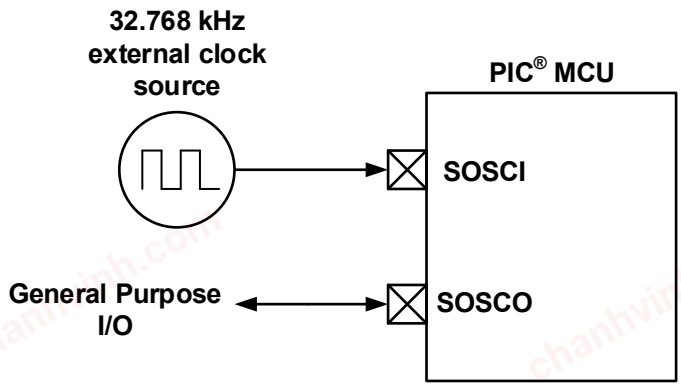
**Figure 11-5. SOSC 32.768 kHz Quartz Crystal Oscillator Operation**



chanhvinh.com



**Figure 11-6. SOSC 32.768 kHz External Clock Operation**




The SOSC can be enabled through one of two methods:

- Programming the RSTOSC Configuration bits to select the SOSC as the system clock
- Programming the NOSC bits to select the SOSC during run time

Two Power modes are available for the secondary oscillator, and are selected using the Secondary Oscillator Power Mode Select (SOSCPWR) bit. When SOSCPWR is clear (SOSCPWR = 0), the oscillator operates in Low Power mode, which is ideal for crystal oscillators with low drive strength. When SOSCPWR is set (SOSCPWR = 1), the oscillator operates in High Power mode, which is ideal for crystal oscillators with high drive strength or high Equivalent Series Resistance (ESR).

**Important:** The SOSC module must be disabled before changing Power modes. Changes to the Power mode during operation may result in undefined oscillator behavior.

#### 11.1.1.5.1 SOSC Start-Up Timing

The SOSC utilizes the Oscillator Start-up Timer (OST) to ensure that the 32.768 kHz crystal oscillator has started and is available for use. Since crystal oscillators do not start immediately and may take a few hundred cycles before achieving stable operation, the OST counts 1024 oscillation periods from the SOSCI input. Once the OST completes its count, module hardware sets the Secondary Oscillator Ready (SOR) bit, indicating that the SOSC is stable and ready to use.

### 11.1.2 Internal Clock Sources

The internal oscillator block contains two independent oscillators that can produce two internal system clock sources:

- High-Frequency Internal Oscillator (HFINTOSC)
- Low-Frequency Internal Oscillator (LFINTOSC)

Internal oscillator selection is performed one of two ways:

1. Program the RSTOSC Configuration bits to select one of the INTOSC sources which will be used upon a device Reset.
2. Write the New Oscillator Source Request (NOSC) bits to select an internal oscillator during run time.

In INTOSC mode, the OSC1/CLKIN and OSC2/CLKOUT pins are available for use as a general purpose I/Os, provided that no external oscillator is connected. The function of the OSC2/CLKOUT pin is determined by the $\overline{\text{CLKOUTEN}}$ Configuration bit. When $\overline{\text{CLKOUTEN}}$ is set ($\overline{\text{CLKOUTEN}}$ = 1), the pin functions as a general-purpose I/O. When $\overline{\text{CLKOUTEN}}$ is clear ($\overline{\text{CLKOUTEN}}$ = 0), the system instruction clock ($F_{OSC}/4$) is available as an output signal on the pin.

chanhvinh.com



#### 11.1.2.1 HFINTOSC

The High-Frequency Internal Oscillator (HFINTOSC) is a factory-calibrated, precision digitally-controlled internal clock source that produces a wide range of stable clock frequencies. The HFINTOSC can be enabled through one of the following methods:

- Program the RSTOSC Configuration bits to select the HFINTOSC upon device Reset or power-up
- Write to the New Oscillator Source Request (NOSC) bits to select the HFINTOSC during run time.

The HFINTOSC frequency is selected via the HFINTOSC Frequency Selection (FRQ) bits. Fine-tuning of the HFINTOSC is done via the HFINTOSC Frequency Tuning (TUN) bits. The HFINTOSC output frequency can be divided (post-scaled) via the New Divider Selection Request (NDIV) bits.

##### 11.1.2.1.1 HFINTOSC Frequency Tuning

The HFINTOSC frequency can be fine-tuned via the HFINTOSC Tuning (OSCTUNE) register. The OSCTUNE register is used by Active Clock Tuning hardware or user software to provide small adjustments to the HFINTOSC nominal frequency.

The OSCTUNE register contains the HFINTOSC Frequency Tuning (TUN) bits. The TUN bits default to a 6-bit, two's complement value of `0x00`, which indicates that the oscillator is operating at the selected frequency. When a value between `0x01` and `0x1F` is written to the TUN bits, the HFINTOSC frequency is increased. When a value between `0x3F` and `0x20` is written to the TUN bits, the HFINTOSC frequency is decreased.

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

**Important:** OSCTUNE tuning does not affect the LFINTOSC frequency.

#### 11.1.2.2 MFINTOSC

The Medium-Frequency Internal Oscillator (MFINTOSC) generates two constant clock outputs (500 kHz and 31.25 kHz). The MFINTOSC clock signals are created from the HFINTOSC using dynamic divider logic, which provides constant MFINTOSC clock rates regardless of selected HFINTOSC frequency.

The MFINTOSC cannot be used as the system clock, but can be used as a clock source for certain peripherals, such as a Timer.

#### 11.1.2.3 SFINTOSC

The Specified-Frequency Internal Oscillator (SFINTOSC) generates a constant 1MHz clock output. The SFINTOSC clock signal is created from the HFINTOSC using dynamic divider logic, which provides constant SFINTOSC clock rates regardless of the selected HFINTOSC frequency.

The SFINTOSC cannot be used as the system clock, but provides a constant 1 MHz clock signal that is used for NVM Write and Erase operations. Additionally, SFINTOSC can be used as a clock source for certain peripherals.

#### 11.1.2.4 LFINTOSC

The Low-Frequency Internal Oscillator (LFINTOSC) is a factory-calibrated 31 kHz internal clock source.

The LFINTOSC can be used as a system clock source, and may be used by certain peripheral modules as a clock source. Additionally, the LFINTOSC provides a time base for the following:

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

The LFINTOSC is enabled through one of the following methods:

- Program the RSTOSC Configuration bits to select LFINTOSC
- Write the NOSC bits to select LFINTOSC during run time

#### 11.1.2.5 ADCRC

The Analog-to-Digital RC (ADCRC) oscillator is dedicated to the ADC module. This oscillator is also referred to as the FRC clock. The ADCRC operates at a fixed frequency of approximately 600 kHz, and is used as a conversion clock

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



source. The ADCRC allows the ADC module to operate in Sleep mode, which can reduce system noise during the ADC conversion. The ADCRC is automatically enabled when it is selected as the clock source for the ADC module, or when selected as the clock source of any peripheral that may use it. The ADCRC may also be manually enabled via the ADC Oscillator Enable (ADOEN) bit, thereby avoiding start-up delays when this source is used intermittently.

### 11.1.3 Oscillator Status and Manual Enable

The Oscillator Status (OSCSTAT) register displays the Ready status for each of the following oscillators:

- External oscillator
- HFINTOSC
- MFINTOSC
- LFINTOSC
- SOSC
- ADCRC
- SFINTOSC

The OSCSTAT register also displays the Ready status for the 4xPLL.

The HFINTOSC Oscillator Ready (HFOR), MFINTOSC Oscillator Ready (MFOR), LFINTOSC Oscillator Ready (LFOR), ADCRC Oscillator Ready (ADOR) and SFINTOSC Oscillator Ready (SFOR) Status bits indicate whether the respective oscillators are ready for use. These clock sources are available for use at any time, but may require a finite amount of time before they have reached the specified accuracy levels. When the oscillators are ready and have achieved the specified accuracy, module hardware sets the respective bits.

When a new value is loaded into the OSCFRQ register, the HFOR bit is cleared by hardware, and will be set again once the HFINTOSC is ready. During pending OSCFRQ changes, the HFINTOSC will stall at either a high or a low state until the oscillator locks in the new frequency and resumes operation.

The External Oscillator Ready (EXTOR) and SOSC Oscillator Ready (SOR) Status bits indicate whether the respective external clock sources are ready for use. The external clock sources use the Oscillator Start-up Timer (OST) to determine when the oscillator is ready. Once the OST has expired, the external oscillator is ready for use, and module hardware sets the respective bits.

The Oscillator Enable (OSCEN) register can be used to manually enable the following oscillators:

- External oscillator
- HFINTOSC
- MFINTOSC
- LFINTOSC
- SOSC
- ADCRC



**Important:** OSCEN cannot be used to manually enable the 4xPLL.

## 11.2 Clock Switching

The system clock source can be switched between external and internal clock sources via software using the New Oscillator Source Request (NOSC) and New Divider Selection Request (NDIV) bits. The following sources can be selected:

- External Oscillator (EXTOSC)
- EXTOSC with 4x PLL
- High-Frequency Internal Oscillator (HFINTOSC)
- Low-Frequency Internal Oscillator (LFINTOSC)
- Secondary Oscillator (SOSC)

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



The Clock Switch Enable (CSWEN) Configuration bit can be used to enable or disable the clock switching capability. When CSWEN is set (CSWEN = 1), writes to NOSC and NDIV by user software will allow the system clock to switch between sources or frequencies. When CSWEN is clear (CSWEN = 0), writes to NOSC and NDIV are ignored, preventing the system clock from switching from one source to another.

### 11.2.1 NOSC and NDIV Bits

The New Oscillator Source Request (NOSC) and the New Divider Selection Request (NDIV) bits are used to select the system clock source and clock frequency divider that will be used by the CPU and peripherals (see the tables below).

When new values are written into NOSC and/or NDIV, the current oscillator selection will continue to operate as the system clock while waiting for the new source to indicate that it is ready. Writes to NDIV without changing the clock source (e.g., changing the HFINTOSC frequency from 1 MHz to 2 MHz) are handled in the same manner as a clock switch.

When the new oscillator selection is ready, the New Oscillator is Ready (NOSCR) bit and the Clock Switch Interrupt Flag (CSWIF) are set by module hardware. If the Clock Switch Interrupt Enable (CSWIE) bit is set (CSWIE = 1), an interrupt will be generated when CSWIF is set. Additionally, the Oscillator Ready (ORDY) bit can be polled to determine that the clock switch has completed and the new oscillator source has replaced the old source as the system clock.

**Important:** The CSWIF interrupt does not wake the device from Sleep.

**Table 11-2. NOSC/COSC Clock Source Selection Table**

| NOSC / COSC | Clock Source |
|---|---|
| 111 | EXTOSC(1) |
| 110 | HFINTOSC(2) |
| 101 | LFINTOSC |
| 100 | SOSC |
| 011 | Reserved |
| 010 | EXTOSC + 4xPLL(3) |
| 001 | Reserved |
| 000 | Reserved |

**Notes:**

1. EXTOSC is configured via the FEXTOSC Configuration bits.
2. HFINTOSC frequency is determined by the FRQ bits.
3. EXTOSC must meet the PLL specifications (see the data sheet Electrical Specifications).

**Table 11-3. NDIV/CDIV Clock Divider Selection Table**

| NDIV / CDIV | Clock Divider |
|---|---|
| 1111-1010 | Reserved |
| 1001 | 512 |
| 1000 | 256 |
| 0111 | 128 |
| 0110 | 64 |

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



.....continued

| NDIV / CDIV | Clock Divider |
|---|---|
| 0101 | 32 |
| 0100 | 16 |
| 0011 | 8 |
| 0010 | 4 |
| 0001 | 2 |
| 0000 | 1 |

### 11.2.2 COSC and CDIV Bits

The Current Oscillator Source Select (COSC) bits and the Current Divider Select (CDIV) bits indicate the current oscillator source and clock divider, respectively. When a new oscillator or divider is requested via the NOSC/NDIV bits, the COSC and CDIV bits remain unchanged until the clock switch actually occurs. When the switch actually occurs, hardware copies the NOSC and NDIV values into COSC and CDIV, the Oscillator Ready (ORDY) bit is set, and the NOSCR bit is cleared by hardware, indicating that the clock switch is complete.

### 11.2.3 CSWHOLD

When the system oscillator changes frequencies, peripherals using the system clock may be affected. For example, if the $I^2C$ module is actively using the system clock as its Serial Clock (SCL) time base, changing the system clock frequency will change the SCL frequency. The Clock Switch Hold (CSWHOLD) bit can be used to suspend a requested clock switch. In this example, software can request a new clock source, use the CSWHOLD bit to suspend the switch, wait for the $I^2C$ bus to become Idle, then reconfigure the SCL frequency based on the new clock source. Once the $I^2C$ has been reconfigured, software can use CSWHOLD to complete the clock switch without causing any issues with the $I^2C$ bus.

When CSWHOLD is set (CSWHOLD = 1), a write to NOSC and/or NDIV is accepted, but the clock switch is suspended and does not automatically complete. While the switch is suspended, code execution continues using the old (current) clock source. Module hardware will still enable the new oscillator selection and set the NOSCR bit. Once the NOSCR bit is set, software may either:

- clear CSWHOLD so that the clock switch can complete, or
- copy the Current Oscillator Source Select (COSC) value into NOSC to abandon the clock switch.

When CSWHOLD is clear (CSWHOLD = 0), the clock switch will occur when the NOSCR bit is set. When NOSCR is set, the CSWIF is also set, and if CSWIE is set, the generated interrupt will be serviced using the new oscillator.

### 11.2.4 PLL Input Switch

Switching between the PLL and any non-PLL source is handled in the same manner as any other clock source change.

When the NOSC selects a source with a PLL, the system continues to operate using the current oscillator until the new oscillator is ready. When the new source is ready, the associated Status bit in the Oscillator Status (OSCSTAT) register is set, and once the PLL is locked and ready for use, the PLL is Ready (PLLR) bit is set. Once both the source and PLL are ready, the switch will complete.

### 11.2.5 Clock Switch and Sleep

If the NOSC/NDIV bits are written with new values and the device is put to Sleep before the clock switch completes, the switch will not take place and the device will enter Sleep mode.

When the device wakes up from Sleep and CSWHOLD is clear (CSWHOLD = 0), the clock switch will complete and the device will wake with the new clock active, setting CSWIF.

When the device wakes from Sleep and CSWHOLD is set (CSWHOLD = 1), the device will wake up with the old clock active, and the new clock source will be requested again.

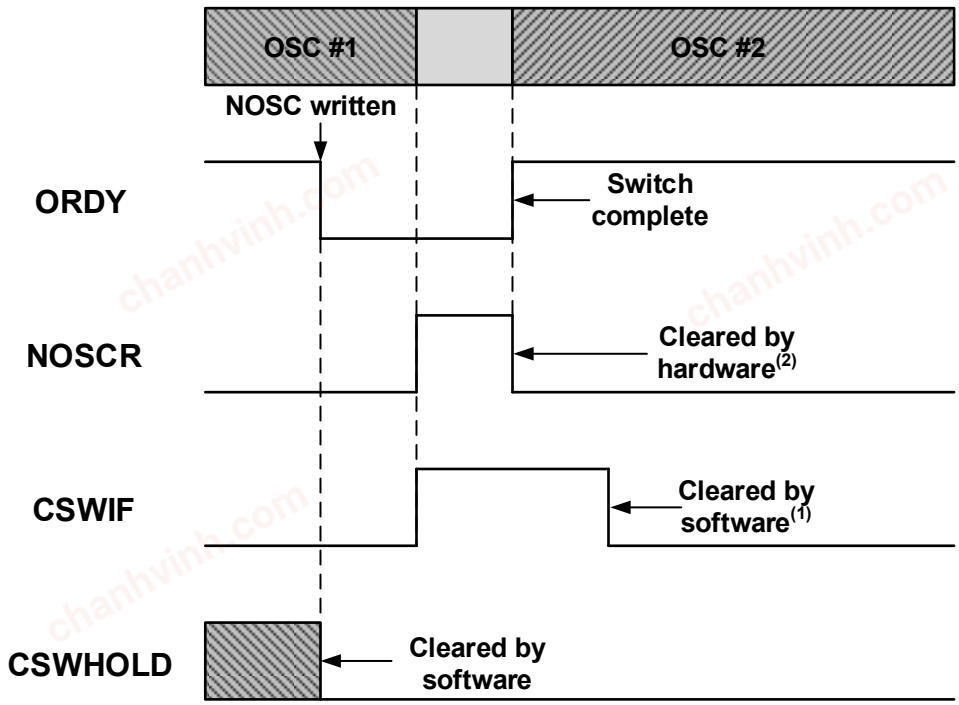
If Doze mode is in effect, the clock switch occurs on the next clock cycle regardless of whether or not the CPU is active during that clock cycle.

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



**Figure 11-7. Clock Switch (CSWHOLD = 0)**



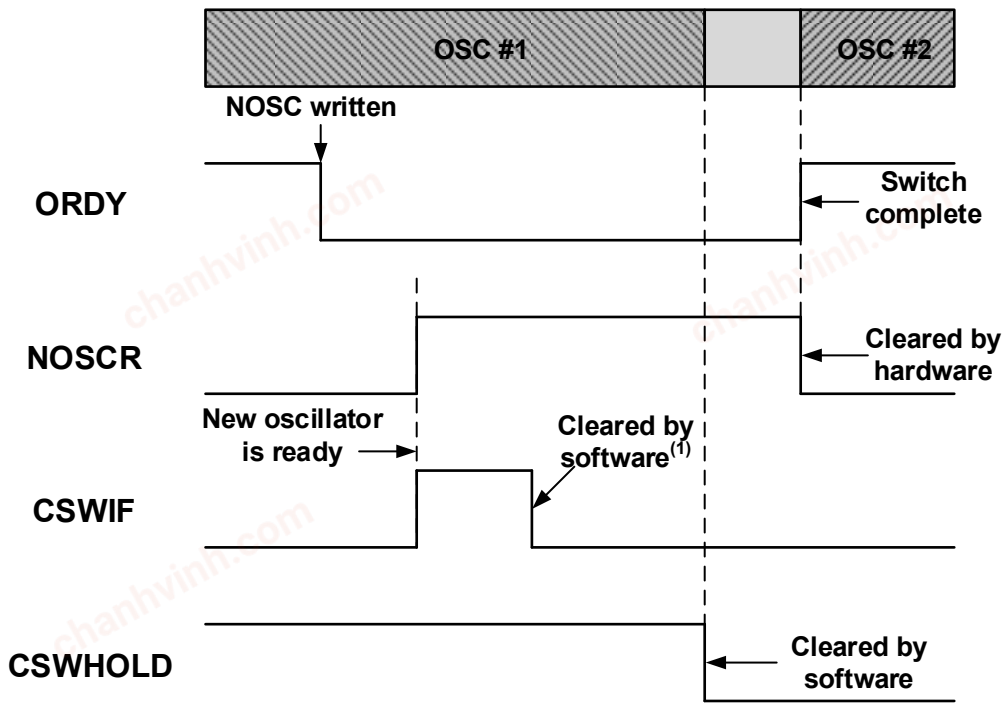

**Notes:**

1. CSWIF is asserted coincident with NOSCR; interrupt is serviced at OSC#2 speed.
2. The assertion of NOSCR may not be seen by the user as it is only set for the duration of the switch.

chanhvinh.com



**Figure 11-8. Clock Switch (CSWHOLD = 1)**



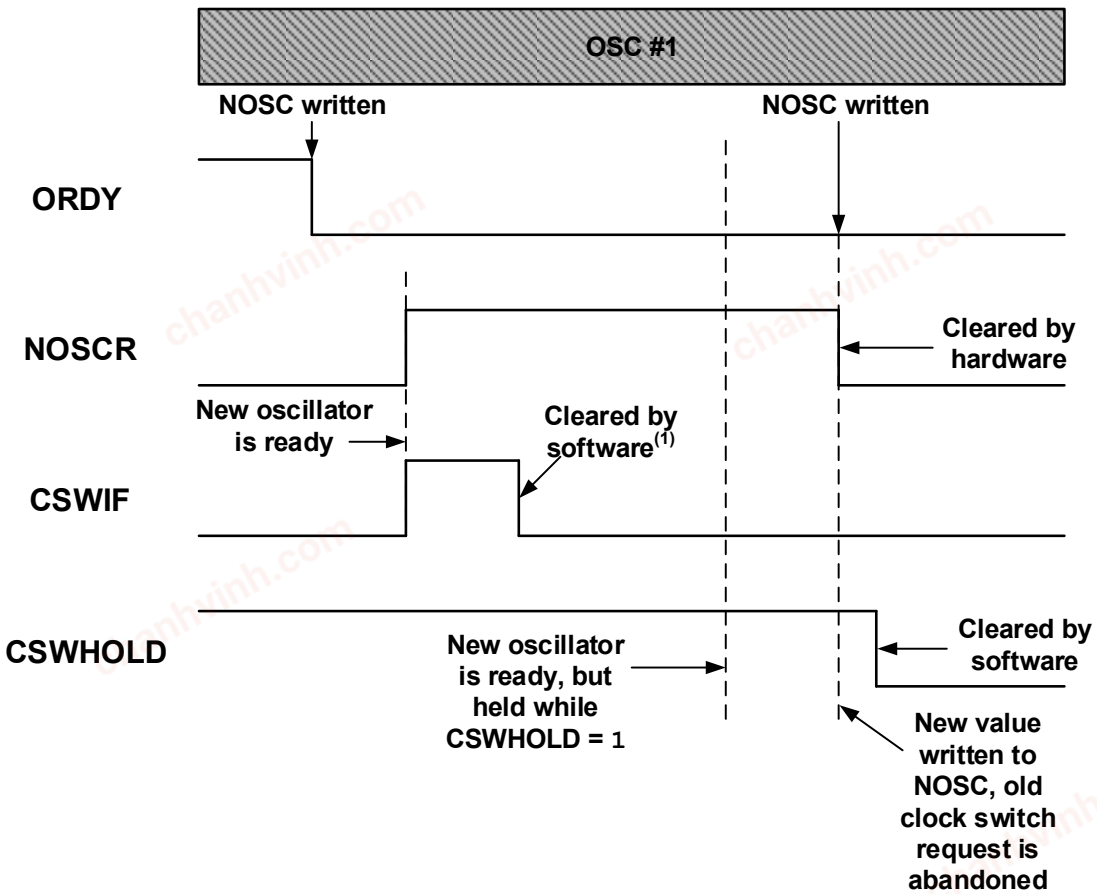

**Note:**

1. CSWIF may be cleared before or after clearing CSWHOLD.

chanhvinh.com



**Figure 11-9. Clock Switch Abandoned**




**Note:**

1. CSWIF may be cleared before or after rewriting NOSC; CSWIF is not automatically cleared.

## 11.3 Fail-Safe Clock Monitor (FSCM)

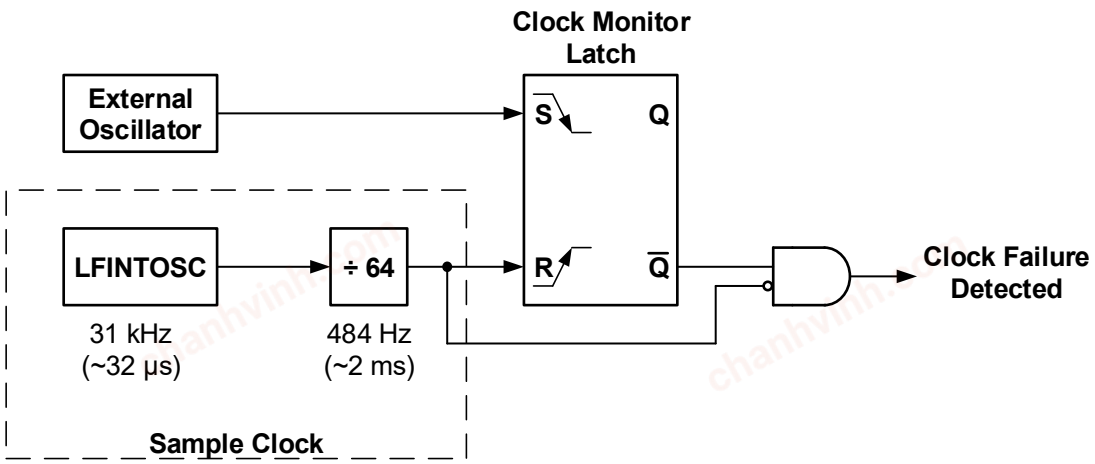
The Fail-Safe Clock Monitor (FSCM) allows the device to continue operating if the external oscillator fails. The FSCM is applicable to all external Oscillator modes. The FSCM is enabled by setting the Fail-Safe Clock Monitor Enable (FCMEN) Configuration bit.

The figure below shows the FSCM block diagram.

chanhvinh.com



**Figure 11-10. FSCM Block Diagram**




### 11.3.1 Fail-Safe Detection

The FSCM detects a failed oscillator by comparing the external oscillator to the FSCM sample clock. The sample clock is generated by dividing the LFINTOSC by 64. The fail detector block contains a latch that is set upon each falling edge of the external clock. The latch is cleared on the rising edge of the sample clock. A failure is detected when a half-period of the sample clock elapses before the external clock goes low.

### 11.3.2 Fail-Safe Operation

When the external clock fails, the FSCM overwrites the NOSC/COSC bits to select the HFINTOSC, and sets the Oscillator Fail Interrupt Flag (OSFIF) bit of the PIR registers. If the Oscillator Fail Interrupt Enable (OSFIE) bit is set, an interrupt will be generated when OSFIF is set. The frequency of HFINTOSC depends on the previous state of the FRQ bits and the NDIV/CDIV bits.

Once a failure is detected, software can take steps to mitigate the effects of a failed oscillator. The system clock will continue to be sourced from the HFINTOSC until the external oscillator is restarted. Once the external source is operational, software can switch back to the external oscillator via the NOSC/NDIV bits.

### 11.3.3 Fail-Safe Condition Clearing

The Fail-Safe condition is cleared after a device Reset, execution of a `SLEEP` instruction, or a change to the NOSC/NDIV bits. When switching to the external oscillator or PLL, the Oscillator Start-up Timer (OST) is restarted. While the OST is running, the device continues to from HFINTOSC. When the OST expires, the Fail-Safe condition is cleared after successfully switching to the external clock source.

**Important:** Software must clear the OSFIF bit before switching to the external oscillator. If the Fail-Safe condition still exists, the OSFIF bit will be set again by module hardware.

### 11.3.4 Reset or Wake-Up from Sleep

The FSCM is designed to detect an oscillator failure after the OST has expired. The OST is used after waking up from Sleep or after any type of Reset, when in either LP, XT or HS mode. If the device is using the EC mode, the FSCM will be active as soon as the Reset or wake-up event has completed.

© 2023 Microchip Technology Inc.
and its subsidiaries

chanhvinh.com



## 11.4 Active Clock Tuning (ACT)

Many applications, such as those using UART communication, require an oscillator with an accuracy of ± 1% over the full temperature and voltage range. To meet this level of accuracy, the Active Clock Tuning (ACT) feature utilizes the SOSC frequency of 32.768 kHz to adjust the frequency of the HFINTOSC over voltage and temperature.

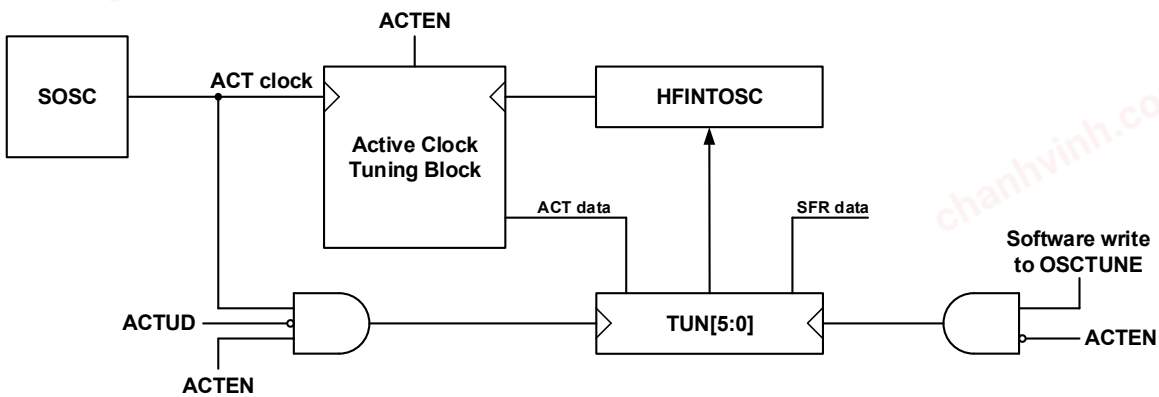
**Important:** Active Clock Tuning requires the use of a 32.768 kHz external oscillator connected to the SOSCI/SOSCO pins.

Active Clock Tuning is enabled via the Active Clock Tuning Enable (ACTEN) bit. When ACTEN is set (ACTEN = 1), the ACT module uses the SOSC time base to measure the HFINTOSC frequency, and uses the HFINTOSC Frequency Tuning (TUN) bits to adjust the HFINTOSC frequency. When ACTEN is clear (ACTEN = 0), the ACT feature is disabled, and user software can utilize the TUN bits to adjust the HFINTOSC frequency.

**Important:** When the ACT feature is enabled, the TUN bits are controlled directly through module hardware and become read-only bits to user software. Writes to the TUN bits when the ACT feature is enabled are ignored.

The figure below shows the Active Clock Tuning block diagram.

**Figure 11-11. Active Clock Tuning (ACT) Block Diagram**




### 11.4.1 ACT Lock Status

The Active Clock Tuning Lock Status (ACTLOCK) bit can be used to determine when the HFINTOSC has been tuned. When ACTLOCK is set (ACTLOCK = 1), the HFINTOSC frequency has been locked to within ± 1% of the nominal frequency. When ACTLOCK is clear (ACTLOCK = 0), the following conditions may be true:

- The HFINTOSC frequency has not been locked to within ± 1%
- A device Reset occurred
- The ACT feature is disabled

**Important:** The ACTLOCK bit is read-only. Writes to ACTLOCK are ignored.

chanhvinh.com



### 11.4.2 ACT Out-of-Range Status

When Active Clock Tuning is enabled, module hardware uses the TUN bits to achieve high accuracy levels. If the module requires a TUN value outside of its range, the ACT Out-of-Range Status (ACTORS) bit is set by hardware (ACTORS = 1).

The ACTORS bit will be set when:

- The HFINTOSC is tuned to its lowest frequency as determined by the TUN bits, and may require a value lower than the TUN bits can provide to achieve accuracy within ± 1%.
- The HFINTOSC is tuned to its highest frequency as determined by the TUN bits, and may require a value higher than the TUN bits can provide to achieve accuracy within ± 1%.

When an ACT out-of-range event occurs, the HFINTOSC will continue to use the last TUN value until the HFINTOSC frequency returns to the tunable range. Once the HFINTOSC returns to the tunable range, module hardware clears the ACTORS bit.

**Important:** The ACTORS bit is read-only. Writes to ACTORS are ignored.

### 11.4.3 ACT Update Disable

When Active Clock Tuning is enabled, the OSCTUNE register is continuously updated every ACT clock cycle. The ACT Update Disable (ACTUD) bit can be used to suspend updates to the OSCTUNE register. When ACTUD is set (ACTUD = 1), updates to OSCTUNE are suspended, although the module continues to operate. The last value written to OSCTUNE is used for tuning, and the ACTLOCK bit is continually updated for each ACT cycle. When ACTUD is clear (ACTUD = 0), the module updates OSCTUNE register every ACT cycle.

### 11.4.4 ACT Interrupts

When Active Clock Tuning is enabled (ACTEN = 1) and either the ACTLOCK or ACTORS bits change state (e.g., from a Locked to an Unlocked state), the ACT Interrupt Flag (ACTIF) of the PIR registers is set (ACTIF = 1). If the ACT Interrupt Enable (ACTIE) bit is set (ACTIE = 1), an interrupt will be generated when ACTIF becomes set. No interrupts are generated for each OSCTUNE update unless the update results in a change of Lock status or Out-of-Range status.

## 11.5 Register Definitions: Oscillator Module

chanhvinh.com



### 11.5.1 ACTCON

**Name:** ACTCON
**Offset:** 0x0294

Active Clock Tuning Control Register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ACTEN | ACTUD | | | ACTLOCK | | ACTORS | |
| Access | R/W | R/W | | | R | | R | |
| Reset | 0 | 0 | | | 0 | | 0 | |

**Bit 7 – ACTEN** Active Clock Tuning Enable

| Value | Description |
|---|---|
| 1 | ACT enabled: HFINTOSC tuning is controlled by the ACT |
| 0 | ACT disabled: HFINTOSC tuning is controlled by the OSCTUNE register via user software |

**Bit 6 – ACTUD** Active Clock Tuning Update Disable

| Value | Condition | Description |
|---|---|---|
| 1 | ACTEN = 1 | Updates to the OSCTUNE register from ACT hardware are disabled |
| 0 | ACTEN = 1 | Updates to the OSCTUNE register from ACT hardware are allowed |
| x | ACTEN = 0 | Updates to the OSCTUNE register through user software are allowed |

**Bit 3 – ACTLOCK** Active Clock Tuning Lock Status

| Value | Description |
|---|---|
| 1 | Locked: HFINTOSC is within ± 1% of its nominal value |
| 0 | Not locked: HFINTOSC may or may not be within ± 1% of its nominal value |

**Bit 1 – ACTORS** Active Clock Tuning Out-of-Range Status

| Value | Description |
|---|---|
| 1 | Value required for tuning is outside of the OSCTUNE range |
| 0 | Value required for tuning is within the OSCTUNE range |

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



### 11.5.2 OSCCON1

**Name:** OSCCON1
**Offset:** 0x028D

Oscillator Control Register 1

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | NOSC[2:0] | | | NDIV[3:0] | | | |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | f | f | f | q | q | q | q |

**Bits 6:4 – NOSC[2:0]** New Oscillator Source Request(1,2,3)
Requests a new oscillator source per the NOSC/COSC Clock Source Selection Table

**Bits 3:0 – NDIV[3:0]** New Divider Selection Request
Requests the new post-scaler division ratio per the NDIV/CDIV Clock Divider Selection Table

**Notes:**

1. The default value is determined by the RSTOSC Configuration bits. See the Reset Oscillator (RSTOSC) selection table for RSTOSC selections.
2. If NOSC is written with a reserved value, the operation is ignored and neither NOSC nor NDIV is written.
3. When CSWEN = 0, these bits are read-only and cannot be changed from the RSTOSC value.

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



### 11.5.3 OSCCON2

**Name:** OSCCON2
**Offset:** 0x028E

Oscillator Control Register 2

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | COSC[2:0] | | | CDIV[3:0] | | | |
| Access | | R | R | R | R | R | R | R |
| Reset | | f | f | f | f | f | f | f |

**Bits 6:4 – COSC[2:0]** Current Oscillator Source Select (read-only)(1)
Indicates the current oscillator source per the NOSC/COSC Clock Source Selection Table

**Bits 3:0 – CDIV[3:0]** Current Divider Select (read-only)
Indicates the current post-scaler divider ratio per the NDIV/CDIV Clock Divider Table

**Note:**

1. The RSTOSC value is the value present when user code execution begins. Refer to the RSTOSC configuration bits or the RSTOSC selection table for the Reset Oscillator selections.

© 2023 Microchip Technology Inc.
and its subsidiaries

chanhvinh.com



### 11.5.4 OSCCON3

**Name:** OSCCON3
**Offset:** 0x028F

Oscillator Control Register 3

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CSWHOLD | SOSCPWR | | ORDY | NOSCR | | | |
| Access | R/W/HC | R/W | | R | R | | | |
| Reset | 0 | 1 | | 0 | 0 | | | |

**Bit 7 – CSWHOLD** Clock Switch Hold Control

| Value | Description |
|---|---|
| 1 | Clock switch (and interrupt) will hold when the oscillator selected by NOSC is ready |
| 0 | Clock switch will proceed when the oscillator selected by NOSC is ready |

**Bit 6 – SOSCPWR** Secondary Oscillator Power Mode Select

| Value | Description |
|---|---|
| 1 | Secondary Oscillator operates in High Power mode |
| 0 | Secondary Oscillator operates in Low Power mode |

**Bit 4 – ORDY** Oscillator Ready (read-only)

| Value | Description |
|---|---|
| 1 | OSCCON1 = OSCCON2; the current system clock is the clock specified by NOSC |
| 0 | A clock switch is in progress |

**Bit 3 – NOSCR** New Oscillator is Ready (read-only)(1)

| Value | Description |
|---|---|
| 1 | A clock switch is in progress and the oscillator selected by NOSC indicates a 'ready' condition |
| 0 | A clock switch is not in progress, or the NOSC-selected oscillator is not ready |

**Note:**

1. If CSWHOLD = 0, the user may not see this bit set (NOSCR = 1). When the oscillator becomes ready, there may be a delay of one instruction cycle before NOSCR is set. The clock switch occurs in the next instruction cycle and NOSCR is cleared.

chanhvinh.com

chanhvinh.com



### 11.5.5 OSCTUNE

**Name:** OSCTUNE
**Offset:** 0x0292

HFINTOSC Frequency Tuning Register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | TUN[5:0] | | | | | |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 5:0 – TUN[5:0]** HFINTOSC Frequency Tuning

| TUN | Condition |
|---|---|
| 01 1111 | Maximum frequency |
| • | • |
| • | • |
| • | • |
| 00 0000 | Center frequency. Oscillator is operating at the selected nominal frequency. (Default value) |
| • | • |
| • | • |
| • | • |
| 10 0000 | Minimum frequency |

chanhvinh.com



### 11.5.6 OSCFRQ

**Name:** OSCFRQ
**Offset:** 0x0293

HFINTOSC Frequency Selection Register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | FRQ[2:0] | | |
| Access | | | | | | R/W | R/W | R/W |
| Reset | | | | | | 0 | 0 | 0 |

**Bits 2:0 – FRQ[2:0]** HFINTOSC Frequency Selection

| FRQ | Nominal Freq (MHz) |
|---|---|
| 111-110 | Reserved |
| 101 | 32 |
| 100 | 16 |
| 011 | 8 |
| 010 | 4 |
| 001 | 2 |
| 000 | 1 |

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



### 11.5.7 OSCSTAT

**Name:** OSCSTAT
**Offset:** 0x0290

Oscillator Status Register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | EXTOR | HFOR | MFOR | LFOR | SOR | ADOR | SFOR | PLLR |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 7 – EXTOR** External Oscillator Ready

| Value | Description |
|---|---|
| 1 | The External oscillator is ready for use |
| 0 | The External oscillator is not enabled, or is not ready for use |

**Bit 6 – HFOR** HFINTOSC Ready

| Value | Description |
|---|---|
| 1 | The HFINTOSC is ready for use |
| 0 | The HFINTOSC is not enabled, or it is not ready for use |

**Bit 5 – MFOR** MFINTOSC Ready

| Value | Description |
|---|---|
| 1 | The MFINTOSC is ready for use |
| 0 | The MFINTOSC is not enabled, or it is not ready for use |

**Bit 4 – LFOR** LFINTOSC Ready

| Value | Description |
|---|---|
| 1 | The LFINTOSC is ready for use |
| 0 | The LFINTOSC is not enabled, or is not ready for use |

**Bit 3 – SOR** Secondary Oscillator (SOSC) Ready

| Value | Description |
|---|---|
| 1 | The Secondary oscillator is ready for use |
| 0 | The Secondary oscillator is not enabled, or is not ready for use |

**Bit 2 – ADOR** ADCRC Oscillator Ready

| Value | Description |
|---|---|
| 1 | The ADCRC oscillator is ready for use |
| 0 | The ADCRC oscillator is not enabled, or is not ready for use |

**Bit 1 – SFOR** SFINTOSC Oscillator Ready

| Value | Description |
|---|---|
| 1 | The SFINTOSC oscillator is ready for use |
| 0 | The SFINTOSC oscillator is not enabled, or is not ready for use |

**Bit 0 – PLLR** PLL is Ready

| Value | Description |
|---|---|
| 1 | The PLL is ready for use |
| 0 | The PLL is not enabled, the required input source is not ready, or the PLL is not locked |

© 2023 Microchip Technology Inc.
and its subsidiaries

chanhvinh.com



### 11.5.8 OSCEN

**Name:** OSCEN
**Offset:** 0x0291

Oscillator Enable Register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | EXTOEN | HFOEN | MFOEN | LFOEN | SOSCEN | ADOEN | | PLLEN |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | | 0 |

**Bit 7 – EXTOEN** External Oscillator Enable

| Value | Description |
|---|---|
| 1 | EXTOSC is explicitly enabled, operating as specified by FEXTOSC |
| 0 | EXTOSC can be enabled by a peripheral request |

**Bit 6 – HFOEN** HFINTOSC Enable

| Value | Description |
|---|---|
| 1 | HFINTOSC is explicitly enabled, operating as specified by OSCFRQ |
| 0 | HFINTOSC can be enabled by a peripheral request |

**Bit 5 – MFOEN** MFINTOSC Enable

| Value | Description |
|---|---|
| 1 | MFINTOSC is explicitly enabled |
| 0 | MFINTOSC can be enabled by a peripheral request |

**Bit 4 – LFOEN** LFINTOSC Enable

| Value | Description |
|---|---|
| 1 | LFINTOSC is explicitly enabled |
| 0 | LFINTOSC can be enabled by a peripheral request |

**Bit 3 – SOSCEN** Secondary Oscillator Enable

| Value | Description |
|---|---|
| 1 | SOSC is explicitly enabled, operating as specified by SOSCPWR |
| 0 | SOSC can be enabled by a peripheral request |

**Bit 2 – ADOEN** ADCRC Oscillator Enable

| Value | Description |
|---|---|
| 1 | ADCRC is explicitly enabled |
| 0 | ADCRC may be enabled by a peripheral request |

**Bit 0 – PLLEN** PLL Enable(1)

| Value | Description |
|---|---|
| 1 | EXTOSC multiplied by the 4x system PLL is used by a peripheral request |
| 0 | EXTOSC is used by a peripheral request |

**Note:**

1. This bit only controls external clock source supplied to the peripherals and has no effect on the system clock.

chanhvinh.com



## 11.6 Register Summary - Oscillator Module

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 ... 0x028C | Reserved | | | | | | | | | |
| 0x028D | OSCCON1 | 7:0 | | NOSC[2:0] | | | NDIV[3:0] | | | |
| 0x028E | OSCCON2 | 7:0 | | COSC[2:0] | | | CDIV[3:0] | | | |
| 0x028F | OSCCON3 | 7:0 | CSWHOLD | SOSCPWR | | ORDY | NOSCR | | | |
| 0x0290 | OSCSTAT | 7:0 | EXTOR | HFOR | MFOR | LFOR | SOR | ADOR | SFOR | PLLR |
| 0x0291 | OSCEN | 7:0 | EXTOEN | HFOEN | MFOEN | LFOEN | SOSCEN | ADOEN | | PLLEN |
| 0x0292 | OSCTUNE | 7:0 | | | TUN[5:0] | | | | | |
| 0x0293 | OSCFRQ | 7:0 | | | | | | | FRQ[2:0] | |
| 0x0294 | ACTCON | 7:0 | ACTEN | ACTUD | | | ACTLOCK | | ACTORS | |

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



# 12. INT - Interrupts

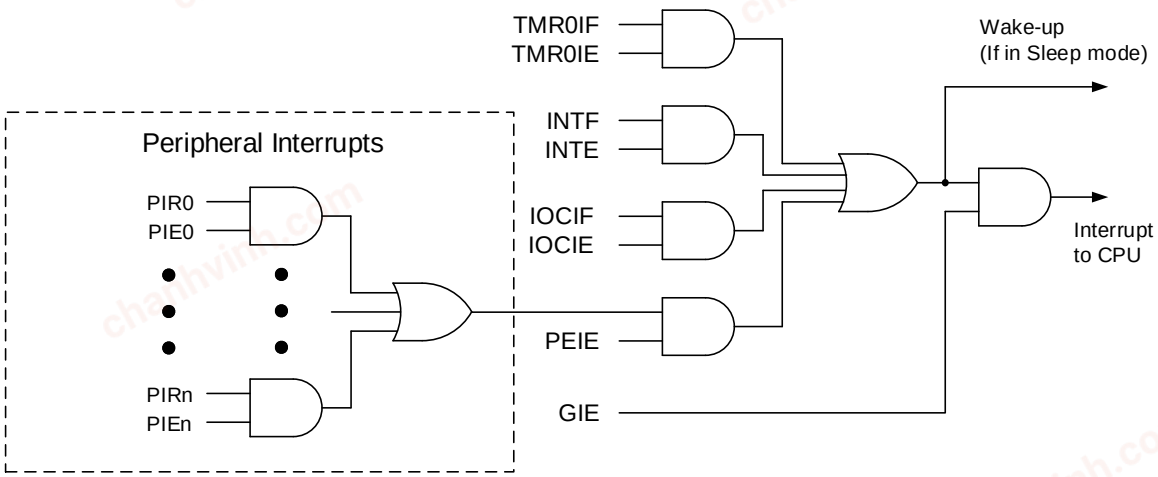
## 12.1 Overview

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.

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

A block diagram of the interrupt logic is shown in Figure 12-1.

**Figure 12-1. Interrupt Logic**




## 12.2 INTCON Register

The Interrupt Control (INTCON) register is readable and writable, and contains the Global Interrupt Enable (GIE), Peripheral Interrupt Enable (PEIE) and External Interrupt Edge Select (INTEDG) bits.

## 12.3 PIE Registers

The Peripheral Interrupt Enable (PIE) registers contain the individual enable bits for the peripheral interrupts. Due to the number of peripheral interrupt sources, there are seven PIE registers in the PIC16F171 family.

## 12.4 PIR Registers

The Peripheral Interrupt Request (PIR) registers contain the individual flag bits for the peripheral interrupts. Due to the number of peripheral interrupt sources, there are seven PIR registers.

## 12.5 Operation

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

- GIE bit
- PEIE bit (if the Interrupt Enable bit of the interrupt event is contained in the PIE registers)

chanhvinh.com



- Interrupt Enable bit(s) for the specific interrupt event(s)

The PIR 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 the "**Automatic Context Saving**" section)
- PC is loaded with the interrupt vector 0004h

The firmware within the Interrupt Service Routine (ISR) may 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 interrupts operation, refer to its peripheral chapter.

**Important:**

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.

## 12.6 Interrupt Latency

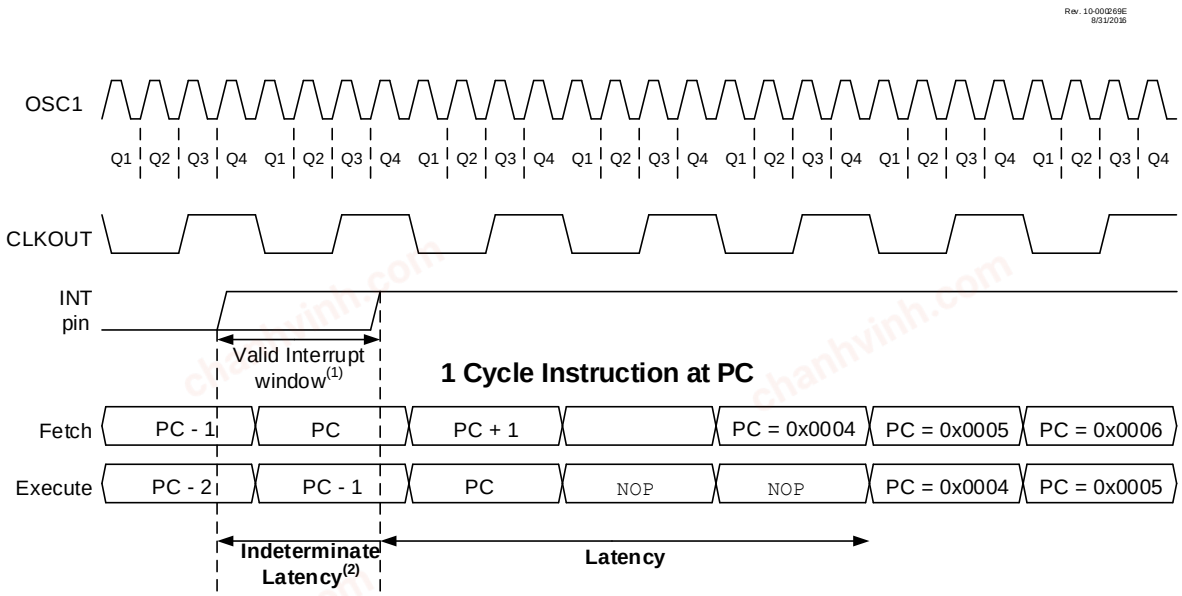
Interrupt latency is defined as the time from when the interrupt event occurs to the time code execution at the interrupt vector begins. The interrupt is sampled during Q1 of the instruction cycle. The actual interrupt latency then depends on the instruction that is executing at the time the interrupt is detected. See the following figures for more details.

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



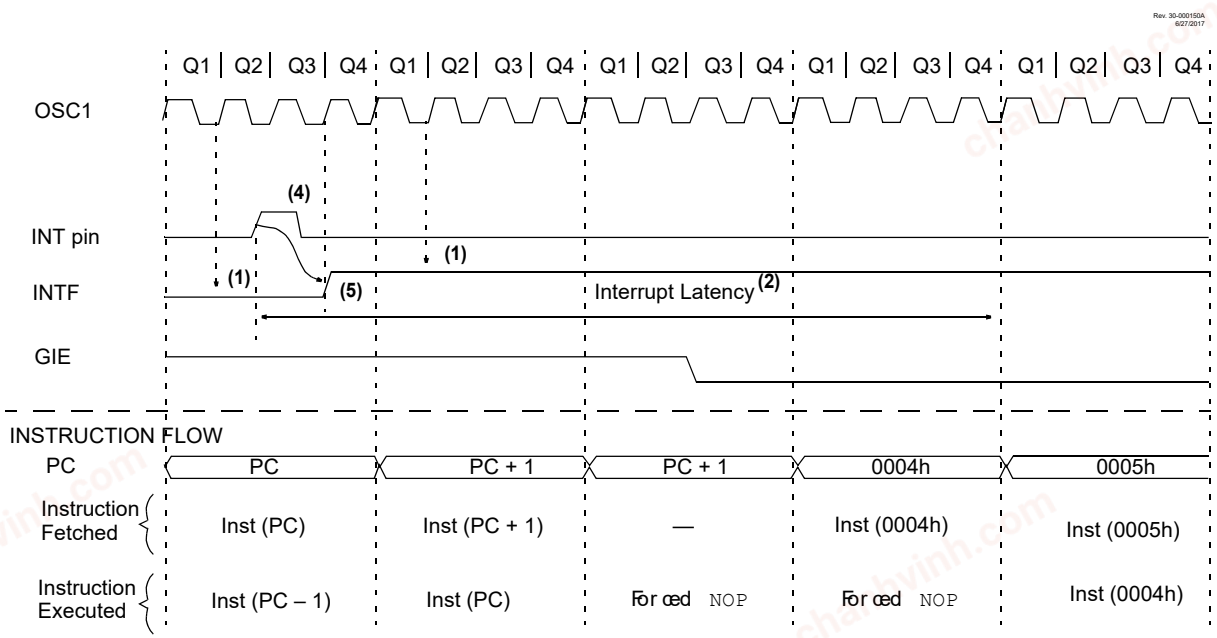
**Figure 12-2. Interrupt Latency**




**Notes:**

1. An interrupt may occur at any time during the interrupt window.
2. Since an interrupt may occur any time during the interrupt window, the actual latency can vary.

**Figure 12-3. INT Pin Interrupt Timing**




**Notes:**

1. INTF flag is sampled here (every Q1).
2. Asynchronous interrupt latency = 3-5 $T_{CY}$. Synchronous latency = 3-4 $T_{CY}$, where $T_{CY}$ = instruction cycle time. Latency is the same whether Inst (PC) is a single cycle or a two-cycle instruction.
3. For minimum width of INT pulse, refer to AC specifications in the **"Electrical Specifications"** chapter.
4. INTF may be set any time during the Q4-Q1 cycles.

chanhvinh.com



## 12.7 Interrupts During Sleep

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.

## 12.8 INT Pin

The INT pin can be used to generate an asynchronous edge-triggered interrupt. This interrupt is enabled by setting the External Interrupt Enable (INTE) bit. The External Interrupt Edge Select (INTEDG) bit 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 External Interrupt Flag (INTF) bit 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.

## 12.9 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:

- WREG 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 may be modified and the value will be restored when exiting the ISR. The shadow registers are available in Bank 63 and are readable and writable. Depending on the user's application, other registers may also need to be saved.

## 12.10 Register Definitions: Interrupt Control

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



### 12.10.1 INTCON

**Name:** INTCON
**Offset:** 0x000B

Interrupt Control Register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | GIE | PEIE | | | | | | INTEDG |
| Access | R/W | R/W | | | | | | R/W |
| Reset | 0 | 0 | | | | | | 1 |

**Bit 7 – GIE** Global Interrupt Enable

| Value | Description |
|---|---|
| 1 | Enables all active interrupts |
| 0 | Disables all interrupts |

**Bit 6 – PEIE** Peripheral Interrupt Enable

| Value | Description |
|---|---|
| 1 | Enables all active peripheral interrupts |
| 0 | Disables all peripheral interrupts |

**Bit 0 – INTEDG** External Interrupt Edge Select

| Value | Description |
|---|---|
| 1 | Interrupt on rising edge of INT pin |
| 0 | Interrupt on falling edge of INT pin |

**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. User software may ensure the appropriate interrupt flag bits are clear prior to enabling an interrupt. This feature allows for software polling.

chanhvinh.com

chanhvinh.com



### 12.10.2 PIE0

**Name:** PIE0
**Offset:** 0x0096

Peripheral Interrupt Enable Register 0

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | TMR0IE | IOCIE | | | | INTE |
| Access | | | R/W | R/W | | | | R/W |
| Reset | | | 0 | 0 | | | | 0 |

**Bit 5 – TMR0IE** Timer0 Interrupt Enable

| Value | Description |
|---|---|
| 1 | TMR0 interrupts are enabled |
| 0 | TMR0 interrupts are disabled |

**Bit 4 – IOCIE** Interrupt-on-Change Enable

| Value | Description |
|---|---|
| 1 | IOC interrupts are enabled |
| 0 | IOC interrupts are disabled |

**Bit 0 – INTE** External Interrupt Enable(1)

| Value | Description |
|---|---|
| 1 | External interrupts are enabled |
| 0 | External interrupts are disabled |

**Notes:**

1. The External Interrupt INT pin is selected by INTPPS.
2. Bit PEIE in the INTCON register must be set to enable any peripheral interrupt controlled by registers PIE1 through PIE6. Interrupt sources controlled by the PIE0 register do not require the PEIE bit to be set to allow interrupt vectoring (when the GIE bit in the INTCON register is set).

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



### 12.10.3 PIE1

**Name:** PIE1
**Offset:** 0x0097

Peripheral Interrupt Enable Register 1

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TMR1GIE | TMR1IE | OSFIE | CSWIE | ACTIE | SCANIE | CRCIE | NVMIE |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 7 – TMR1GIE** TMR1 Gate Interrupt Enable

| Value | Description |
|---|---|
| 1 | TMR1 Gate interrupts are enabled |
| 0 | TMR1 Gate interrupts are disabled |

**Bit 6 – TMR1IE** TMR1 Interrupt Enable

| Value | Description |
|---|---|
| 1 | TMR1 interrupts are enabled |
| 0 | TMR1 interrupts are disabled |

**Bit 5 – OSFIE** Oscillator Failure Interrupt Enable

| Value | Description |
|---|---|
| 1 | Oscillator Failure interrupts are enabled |
| 0 | Oscillator Failure interrupts are disabled |

**Bit 4 – CSWIE** Clock Switch Interrupt Enable

| Value | Description |
|---|---|
| 1 | Clock Switch interrupts are enabled |
| 0 | Clock Switch interrupts are disabled |

**Bit 3 – ACTIE** Active Clock Tuning Interrupt Enable

| Value | Description |
|---|---|
| 1 | Active Clock Tuning interrupts are enabled |
| 0 | Active Clock Tuning interrupts are disabled |

**Bit 2 – SCANIE** Memory Scanner Interrupt Enable

| Value | Description |
|---|---|
| 1 | Memory Scanner interrupts are enabled |
| 0 | Memory Scanner interrupts are disabled |

**Bit 1 – CRCIE** CRC Interrupt Enable

| Value | Description |
|---|---|
| 1 | CRC interrupts are enabled |
| 0 | CRC interrupts are disabled |

**Bit 0 – NVMIE** NVM Interrupt Enable

| Value | Description |
|---|---|
| 1 | NVM interrupts are enabled |
| 0 | NVM interrupts are disabled |

**Note:** Bit PEIE of the INTCON register must be set to enable any peripheral interrupt controlled by registers PIE1 through PIE6.

chanhvinh.com



### 12.10.4 PIE2

**Name:** PIE2
**Offset:** 0x0098

Peripheral Interrupt Enable Register 2

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | CCP2IE | CCP1IE | TMR6IE | TMR4IE | TMR2IE | TMR3GIE | TMR3IE |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 6 – CCP2IE** CCP2 Interrupt Enable

| Value | Description |
|---|---|
| 1 | CCP2 interrupts are enabled |
| 0 | CCP2 interrupts are disabled |

**Bit 5 – CCP1IE** CCP1 Interrupt Enable

| Value | Description |
|---|---|
| 1 | CCP1 interrupts are enabled |
| 0 | CCP1 interrupts are disabled |

**Bit 4 – TMR6IE** TMR6 Interrupt Enable

| Value | Description |
|---|---|
| 1 | TMR6 interrupts are enabled |
| 0 | TMR6 interrupts are disabled |

**Bit 3 – TMR4IE** TMR4 Interrupt Enable

| Value | Description |
|---|---|
| 1 | TMR4 interrupts are enabled |
| 0 | TMR4 interrupts are disabled |

**Bit 2 – TMR2IE** TMR2 Interrupt Enable

| Value | Description |
|---|---|
| 1 | TMR2 interrupts are enabled |
| 0 | TMR2 interrupts are disabled |

**Bit 1 – TMR3GIE** TMR3 Gate Interrupt Enable

| Value | Description |
|---|---|
| 1 | TMR3 Gate interrupts are enabled |
| 0 | TMR3 Gate interrupts are disabled |

**Bit 0 – TMR3IE** TMR3 Interrupt Enable

| Value | Description |
|---|---|
| 1 | TMR3 interrupts are enabled |
| 0 | TMR3 interrupts are disabled |

**Note:** Bit PEIE of the INTCON register must be set to enable any peripheral interrupt controlled by registers PIE1 through PIE6.

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



### 12.10.5 PIE3

**Name:** PIE3
**Offset:** 0x0099

Peripheral Interrupt Enable Register 3

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PWM4IE | PWM4PIE | PWM3IE | PWM3PIE | PWM2IE | PWM2PIE | PWM1IE | PWM1PIE |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 7 – PWM4IE** PWM4 Parameter Interrupt Enable

| Value | Description |
|---|---|
| 1 | PWM4 Parameter interrupts are enabled |
| 0 | PWM4 Parameter interrupts are disabled |

**Bit 6 – PWM4PIE** PWM4 Period Interrupt Enable

| Value | Description |
|---|---|
| 1 | PWM4 Period interrupts are enabled |
| 0 | PWM4 Period interrupts are disabled |

**Bit 5 – PWM3IE** PWM3 Parameter Interrupt Enable

| Value | Description |
|---|---|
| 1 | PWM3 Parameter interrupts are enabled |
| 0 | PWM3 Parameter interrupts are disabled |

**Bit 4 – PWM3PIE** PWM3 Period Interrupt Enable

| Value | Description |
|---|---|
| 1 | PWM3 Period interrupts are enabled |
| 0 | PWM3 Period interrupts are disabled |

**Bit 3 – PWM2IE** PWM2 Parameter Interrupt Enable

| Value | Description |
|---|---|
| 1 | PWM2 Parameter interrupts are enabled |
| 0 | PWM2 Parameter interrupts are disabled |

**Bit 2 – PWM2PIE** PWM2 Period Interrupt Enable

| Value | Description |
|---|---|
| 1 | PWM2 Period interrupts are enabled |
| 0 | PWM2 Period interrupts are disabled |

**Bit 1 – PWM1IE** PWM1 Parameter Interrupt Enable

| Value | Description |
|---|---|
| 1 | PWM1 Parameter interrupts are enabled |
| 0 | PWM1 Parameter interrupts are disabled |

**Bit 0 – PWM1PIE** PWM1 Period Interrupt Enable

| Value | Description |
|---|---|
| 1 | PWM1 Period interrupts are enabled |
| 0 | PWM1 Period interrupts are disabled |

**Note:** Bit PEIE in the INTCON register must be set to enable any peripheral interrupt controlled by registers PIE1 through PIE6.

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



### 12.10.6 PIE4

**Name:** PIE4
**Offset:** 0x009A

Peripheral Interrupt Enable Register 4

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RC1IE | TX1IE | CLC4IE | CLC3IE | CLC2IE | CLC1IE | CWG1IE | NCO1IE |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 7 – RC1IE** EUSART1 Receive Interrupt Enable

| Value | Description |
|---|---|
| 1 | EUSART1 receive interrupts are enabled |
| 0 | EUSART1 receive interrupts are disabled |

**Bit 6 – TX1IE** EUSART1 Transmit Interrupt Enable

| Value | Description |
|---|---|
| 1 | EUSART1 transmit interrupts are enabled |
| 0 | EUSART1 transmit interrupts are disabled |

**Bit 5 – CLC4IE** CLC4 Interrupt Enable

| Value | Description |
|---|---|
| 1 | CLC4 interrupts are enabled |
| 0 | CLC4 interrupts are disabled |

**Bit 4 – CLC3IE** CLC3 Interrupt Enable

| Value | Description |
|---|---|
| 1 | CLC3 interrupts are enabled |
| 0 | CLC3 interrupts are disabled |

**Bit 3 – CLC2IE** CLC2 Interrupt Enable

| Value | Description |
|---|---|
| 1 | CLC2 interrupts are enabled |
| 0 | CLC2 interrupts are disabled |

**Bit 2 – CLC1IE** CLC1 Interrupt Enable

| Value | Description |
|---|---|
| 1 | CLC1 interrupts are enabled |
| 0 | CLC1 interrupts are disabled |

**Bit 1 – CWG1IE** CWG1 Interrupt Enable

| Value | Description |
|---|---|
| 1 | CWG1 interrupts are enabled |
| 0 | CWG1 interrupts are disabled |

**Bit 0 – NCO1IE** NCO1 Interrupt Enable

| Value | Description |
|---|---|
| 1 | NCO1 interrupts are enabled |
| 0 | NCO1 interrupts are disabled |

**Note:** Bit PEIE of the INTCON register must be set to enable any peripheral interrupt controlled by registers PIE1 through PIE6.

chanhvinh.com



### 12.10.7 PIE5

**Name:** PIE5
**Offset:** 0x009B

Peripheral Interrupt Enable Register 5

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CM2IE | CM1IE | BCL2IE | SSP2IE | BCL1IE | SSP1IE | RC2IE | TX2IE |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 7 – CM2IE** Comparator 2 Interrupt Enable

| Value | Description |
|---|---|
| 1 | Comparator 2 interrupts are enabled |
| 0 | Comparator 2 interrupts are disabled |

**Bit 6 – CM1IE** Comparator 1 Interrupt Enable

| Value | Description |
|---|---|
| 1 | Comparator 1 interrupts are enabled |
| 0 | Comparator 1 interrupts are disabled |

**Bit 5 – BCL2IE** MSSP2 Bus Collision Interrupt Enable

| Value | Description |
|---|---|
| 1 | MSSP2 Bus Collision interrupts are enabled |
| 0 | MSSP2 Bus Collision interrupts are disabled |

**Bit 4 – SSP2IE** MSSP2 Interrupt Enable

| Value | Description |
|---|---|
| 1 | MSSP2 interrupts are enabled |
| 0 | MSSP2 interrupts are disabled |

**Bit 3 – BCL1IE** MSSP1 Bus Collision Interrupt Enable

| Value | Description |
|---|---|
| 1 | MSSP1 Bus Collision interrupts are enabled |
| 0 | MSSP1 Bus Collision interrupts are disabled |

**Bit 2 – SSP1IE** MSSP1 Interrupt Enable

| Value | Description |
|---|---|
| 1 | MSSP1 interrupts are enabled |
| 0 | MSSP1 interrupts are disabled |

**Bit 1 – RC2IE** EUSART2 Receive Interrupt Enable

| Value | Description |
|---|---|
| 1 | EUSART2 receive interrupts are enabled |
| 0 | EUSART2 receive interrupts are disabled |

**Bit 0 – TX2IE** EUSART2 Transmit Interrupt Enable

| Value | Description |
|---|---|
| 1 | EUSART2 transmit interrupts are enabled |
| 0 | EUSART2 transmit interrupts are disabled |

**Note:** Bit PEIE of the INTCON register must be set to enable any peripheral interrupt controlled by registers PIE1 through PIE6.

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



### 12.10.8 PIE6

**Name:** PIE6
**Offset:** 0x009C

Peripheral Interrupt Enable Register 6

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | ZCDIE | ADTIE | ADIE |
| Access | | | | | | R/W | R/W | R/W |
| Reset | | | | | | 0 | 0 | 0 |

**Bit 2 – ZCDIE** ZCD Interrupt Enable

| Value | Description |
|---|---|
| 1 | ZCD interrupts are enabled |
| 0 | ZCD interrupts are disabled |

**Bit 1 – ADTIE** ADC Threshold Interrupt Enable

| Value | Description |
|---|---|
| 1 | ADC Threshold interrupts are enabled |
| 0 | ADC Threshold interrupts are disabled |

**Bit 0 – ADIE** ADC Interrupt Enable

| Value | Description |
|---|---|
| 1 | ADC interrupts are enabled |
| 0 | ADC interrupts are disabled |

**Note:** Bit PEIE of the INTCON register must be set to enable any peripheral interrupt controlled by registers PIE1 through PIE6.

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



### 12.10.9 PIR0

**Name:** PIR0
**Offset:** 0x008C

Peripheral Interrupt Request Register 0

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | TMR0IF | IOCIF | | | | INTF |
| Access | | | R/W/HS | R | | | | R/W/HS |
| Reset | | | 0 | 0 | | | | 0 |

**Bit 5 – TMR0IF** Timer0 Interrupt Flag

| Value | Description |
|---|---|
| 1 | TMR0 register has overflowed (must be cleared by software) |
| 0 | TMR0 register has not overflowed |

**Bit 4 – IOCIF** Interrupt-on-Change Flag(2)

| Value | Description |
|---|---|
| 1 | One or more of the IOCAF-IOCCF register bits are currently set, indicating an enabled edge was detected by the IOC module. |
| 0 | None of the IOCAF-IOCCF register bits are currently set |

**Bit 0 – INTF** External Interrupt Flag(1)

| Value | Description |
|---|---|
| 1 | External Interrupt has occurred |
| 0 | External Interrupt has not occurred |

**Notes:**

1. The External Interrupt INT pin is selected by INTPPS.
2. The IOCIF bit is the logical OR of all the IOCAF-IOCCF flags. Therefore, to clear the IOCIF flag, application firmware must clear all of the lower level IOCAF-IOCCF register bits.
3. Interrupt flag bits are set when an Interrupt condition occurs, regardless of the state of its corresponding enable bit or the Global Enable (GIE) bit. User software may ensure the appropriate interrupt flag bits are cleared before enabling an interrupt.

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



### 12.10.10 PIR1

**Name:** PIR1
**Offset:** 0x008D

Peripheral Interrupt Request Register 1

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TMR1GIF | TMR1IF | OSFIF | CSWIF | ACTIF | SCANIF | CRCIF | NVMIF |
| Access | R/W/HS | R/W/HS | R | R | R/W/HS | R/W/HS | R/W/HS | R/W/HS |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 7 – TMR1GIF** TMR1 Gate Interrupt Flag

| Value | Description |
|---|---|
| 1 | The TMR1 Gate has gone inactive (must be cleared in software) |
| 0 | TMR1 Gate is active |

**Bit 6 – TMR1IF** TMR1 Interrupt Flag

| Value | Description |
|---|---|
| 1 | TMR1 interrupt has occurred (must be cleared in software) |
| 0 | TMR1 interrupt event has not occurred |

**Bit 5 – OSFIF** Oscillator Failure Interrupt Flag

| Value | Description |
|---|---|
| 1 | An oscillator failure event has been detected (must be cleared in software) |
| 0 | An oscillator failure event has not been detected |

**Bit 4 – CSWIF** Clock Switch Interrupt Flag

| Value | Description |
|---|---|
| 1 | A Clock Switch interrupt has occurred (must be cleared in software) |
| 0 | A Clock Switch interrupt event has not occurred |

**Bit 3 – ACTIF** Active Clock Tuning Interrupt Flag

| Value | Description |
|---|---|
| 1 | Active Clock Tuning interrupt occurred (must be cleared in software) |
| 0 | Active Clock Tuning interrupt event has not occurred |

**Bit 2 – SCANIF** Memory Scanner Interrupt Flag

| Value | Description |
|---|---|
| 1 | Memory Scanner interrupt has occurred (must be cleared in software) |
| 0 | Memory Scanner interrupt event has not occurred |

**Bit 1 – CRCIF** CRC Interrupt Flag

| Value | Description |
|---|---|
| 1 | CRC interrupt has occurred (must be cleared in software) |
| 0 | CRC interrupt event has not occurred |

**Bit 0 – NVMIF** Nonvolatile Memory (NVM) Interrupt Flag

| Value | Description |
|---|---|
| 1 | The requested NVM operation has completed (must be cleared in software) |
| 0 | NVM interrupt event has not occurred |

**Note:** Interrupt flag bits are set when an Interrupt condition occurs, regardless of the state of its corresponding enable bit or the Global Enable (GIE) bit. User software may ensure the appropriate interrupt flag bits are cleared before enabling an interrupt.

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



### 12.10.11 PIR2

**Name:** PIR2
**Offset:** 0x008E

Peripheral Interrupt Request Register 2

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | CCP2IF | CCP1IF | TMR6IF | TMR4IF | TMR2IF | TMR3GIF | TMR3IF |
| Access | | R/W/HS | R/W/HS | R/W/HS | R/W/HS | R/W/HS | R/W/HS | R/W/HS |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 6 – CCP2IF** CCP2 Interrupt Flag

| Value | CCP Mode | | |
|---|---|---|---|
| | Capture | Compare | PWM |
| 1 | Capture occurred (must be cleared in software) | Compare match occurred (must be cleared in software) | Output trailing edge occurred (must be cleared in software) |
| 0 | Capture did not occur | Compare match did not occur | Output trailing edge did not occur |

**Bit 5 – CCP1IF** CCP1 Interrupt Flag

| Value | CCP Mode | | |
|---|---|---|---|
| | Capture | Compare | PWM |
| 1 | Capture occurred (must be cleared in software) | Compare match occurred (must be cleared in software) | Output trailing edge occurred (must be cleared in software) |
| 0 | Capture did not occur | Compare match did not occur | Output trailing edge did not occur |

**Bit 4 – TMR6IF** TMR6 Interrupt Flag

| Value | Description |
|---|---|
| 1 | TMR6 interrupt has occurred (must be cleared in software) |
| 0 | TMR6 interrupt event has not occurred |

**Bit 3 – TMR4IF** TMR4 Interrupt Flag

| Value | Description |
|---|---|
| 1 | TMR4 interrupt has occurred (must be cleared in software) |
| 0 | TMR4 interrupt event has not occurred |

**Bit 2 – TMR2IF** TMR2 Interrupt Flag

| Value | Description |
|---|---|
| 1 | TMR2 interrupt has occurred (must be cleared in software) |
| 0 | TMR2 interrupt event has not occurred |

**Bit 1 – TMR3GIF** TMR3 Gate Interrupt Flag

| Value | Description |
|---|---|
| 1 | The TMR3 Gate has gone inactive (must be cleared in software) |
| 0 | TMR3 Gate is active |

**Bit 0 – TMR3IF** TMR3 Interrupt Flag

| Value | Description |
|---|---|
| 1 | TMR3 interrupt has occurred (must be cleared in software) |
| 0 | TMR3 interrupt event has not occurred |

**Note:** Interrupt flag bits are set when an Interrupt condition occurs, regardless of the state of its corresponding enable bit or the Global Enable (GIE) bit. User software may ensure the appropriate interrupt flag bits are cleared before enabling an interrupt.

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



### 12.10.12 PIR3

**Name:** PIR3
**Offset:** 0x008F

Peripheral Interrupt Request Register 3

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PWM4IF | PWM4PIF | PWM3IF | PWM3PIF | PWM2IF | PWM2PIF | PWM1IF | PWM1PIF |
| Access | R/W/HS | R/W/HS | R/W/HS | R/W/HS | R/W/HS | R/W/HS | R/W/HS | R/W/HS |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 7 – PWM4IF** PWM4 Parameter Interrupt Flag

| Value | Description |
|---|---|
| 1 | PWM4 Parameter interrupt has occurred (must be cleared in software) |
| 0 | PWM4 Parameter interrupt event has not occurred |

**Bit 6 – PWM4PIF** PWM4 Period Interrupt Flag

| Value | Description |
|---|---|
| 1 | PWM4 Period interrupt has occurred (must be cleared in software) |
| 0 | PWM4 Period interrupt event has not occurred |

**Bit 5 – PWM3IF** PWM3 Parameter Interrupt Flag

| Value | Description |
|---|---|
| 1 | PWM3 Parameter interrupt has occurred (must be cleared in software) |
| 0 | PWM3 Parameter interrupt event has not occurred |

**Bit 4 – PWM3PIF** PWM3 Period Interrupt Flag

| Value | Description |
|---|---|
| 1 | PWM3 Period interrupt has occurred (must be cleared in software) |
| 0 | PWM3 Period interrupt event has not occurred |

**Bit 3 – PWM2IF** PWM2 Parameter Interrupt Flag

| Value | Description |
|---|---|
| 1 | PWM2 Parameter interrupt has occurred (must be cleared in software) |
| 0 | PWM2 Parameter interrupt event has not occurred |

**Bit 2 – PWM2PIF** PWM2 Period Interrupt Flag

| Value | Description |
|---|---|
| 1 | PWM2 Period interrupt has occurred (must be cleared in software) |
| 0 | PWM2 Period interrupt event has not occurred |

**Bit 1 – PWM1IF** PWM1 Parameter Interrupt Flag

| Value | Description |
|---|---|
| 1 | PWM1 Parameter interrupt has occurred (must be cleared in software) |
| 0 | PWM1 Parameter interrupt event has not occurred |

**Bit 0 – PWM1PIF** PWM1 Period Interrupt Flag

| Value | Description |
|---|---|
| 1 | PWM1 Period interrupt has occurred (must be cleared in software) |
| 0 | PWM1 Period interrupt event has not occurred |

**Note:** Interrupt flag bits are set when an Interrupt condition occurs, regardless of the state of its corresponding enable bit or the Global Enable (GIE) bit. User software may ensure the appropriate interrupt flag bits are cleared before enabling an interrupt.

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



### 12.10.13 PIR4

**Name:** PIR4
**Offset:** 0x0090

Peripheral Interrupt Request Register 4

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RC1IF | TX1IF | CLC4IF | CLC3IF | CLC2IF | CLC1IF | CWG1IF | NCO1IF |
| Access | R | R | R/W/HS | R/W/HS | R/W/HS | R/W/HS | R/W/HS | R/W/HS |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 7 – RC1IF** EUSART1 Receive Interrupt Flag(1)

| Value | Description |
|---|---|
| 1 | The EUSART1 receive buffer (RC1REG) is not empty (contains at least one byte) |
| 0 | The EUSART1 receive buffer is empty |

**Bit 6 – TX1IF** EUSART1 Transmit Interrupt Flag(2)

| Value | Description |
|---|---|
| 1 | The EUSART1 transmit buffer (TX1REG) is empty |
| 0 | The EUSART1 transmit buffer is not empty |

**Bit 5 – CLC4IF** CLC4 Interrupt Flag

| Value | Description |
|---|---|
| 1 | CLC4 interrupt has occurred (must be cleared in software) |
| 0 | CLC4 interrupt event has not occurred |

**Bit 4 – CLC3IF** CLC3 Interrupt Flag

| Value | Description |
|---|---|
| 1 | CLC3 interrupt has occurred (must be cleared in software) |
| 0 | CLC3 interrupt event has not occurred |

**Bit 3 – CLC2IF** CLC2 Interrupt Flag

| Value | Description |
|---|---|
| 1 | CLC2 interrupt has occurred (must be cleared in software) |
| 0 | CLC2 interrupt event has not occurred |

**Bit 2 – CLC1IF** CLC1 Interrupt Flag

| Value | Description |
|---|---|
| 1 | CLC1 interrupt has occurred (must be cleared in software) |
| 0 | CLC1 interrupt event has not occurred |

**Bit 1 – CWG1IF** CWG1 Interrupt Flag

| Value | Description |
|---|---|
| 1 | CWG1 interrupt has occurred (must be cleared in software) |
| 0 | CWG1 interrupt event has not occurred |

**Bit 0 – NCO1IF** NCO1 Interrupt Flag

| Value | Description |
|---|---|
| 1 | NCO1 interrupt has occurred (must be cleared in software) |
| 0 | NCO1 interrupt event has not occurred |

© 2023 Microchip Technology Inc.
and its subsidiaries

chanhvinh.com



**Notes:**

1. RC1IF is read-only. User software must read RC1REG to clear RC1IF.
2. TX1IF is read-only. User software must load TX1REG to clear TX1IF. TX1IF does not indicate a completed transmission (use TMRT for this purpose instead).
3. Interrupt flag bits are set when an Interrupt condition occurs, regardless of the state of its corresponding enable bit or the Global Enable (GIE) bit. User software may ensure the appropriate interrupt flag bits are cleared before enabling an interrupt.

chanhvinh.com

chanhvinh.com



### 12.10.14 PIR5

**Name:** PIR5
**Offset:** 0x0091

Peripheral Interrupt Request Register 5

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CM2IF | CM1IF | BCL2IF | SSP2IF | BCL1IF | SSP1IF | RC2IF | TX2IF |
| Access | R/W/HS | R/W/HS | R/W/HS | R/W/HS | R/W/HS | R/W/HS | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 7 – CM2IF** Comparator 2 Interrupt Flag

| Value | Description |
|---|---|
| 1 | Comparator 2 interrupt has occurred (must be cleared in software) |
| 0 | Comparator 2 interrupt event has not occurred |

**Bit 6 – CM1IF** Comparator 1 Interrupt Flag

| Value | Description |
|---|---|
| 1 | Comparator 1 interrupt has occurred (must be cleared in software) |
| 0 | Comparator 1 interrupt event has not occurred |

**Bit 5 – BCL2IF** MSSP2 Bus Collision Interrupt Flag

| Value | Description |
|---|---|
| 1 | An MSSP2 Bus Collision interrupt has occurred (must be cleared in software) |
| 0 | No MSSP2 Bus Collision event was detected |

**Bit 4 – SSP2IF** MSSP2 Interrupt Flag

| Value | Description |
|---|---|
| 1 | MSSP2 interrupt has occurred (must be cleared in software) |
| 0 | MSSP2 interrupt event has not occurred |

**Bit 3 – BCL1IF** MSSP1 Bus Collision Interrupt Flag

| Value | Description |
|---|---|
| 1 | An MSSP1 Bus Collision was detected (must be cleared in software) |
| 0 | No MSSP1 Bus Collision event was detected |

**Bit 2 – SSP1IF** MSSP1 Interrupt Flag

| Value | Description |
|---|---|
| 1 | MSSP1 interrupt has occurred (must be cleared in software) |
| 0 | MSSP1 interrupt event has not occurred |

**Bit 1 – RC2IF** EUSART2 Receive Interrupt Flag(1)

| Value | Description |
|---|---|
| 1 | The EUSART2 receive buffer (RC2REG) is not empty (contains at least one byte) |
| 0 | The EUSART2 receive buffer is empty |

**Bit 0 – TX2IF** EUSART2 Transmit Interrupt Flag(2)

| Value | Description |
|---|---|
| 1 | The EUSART2 transmit buffer (TX2REG) is empty |
| 0 | The EUSART2 transmit buffer is not empty |

© 2023 Microchip Technology Inc.
and its subsidiaries

chanhvinh.com



**Notes:**

1. RC2IF is read-only. User software must read RC2REG to clear RC2IF.
2. TX2IF is read-only. User software must load TX2REG to clear TX2IF. TX2IF does not indicate a completed transmission (use TMRT for this purpose instead).
3. Interrupt flag bits are set when an Interrupt condition occurs, regardless of the state of its corresponding enable bit or the Global Enable (GIE) bit. User software may ensure the appropriate interrupt flag bits are cleared before enabling an interrupt.

chanhvinh.com

chanhvinh.com



### 12.10.15 PIR6

**Name:** PIR6
**Offset:** 0x0092

Peripheral Interrupt Request Register 6

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | ZCDIF | ADTIF | ADIF |
| Access | | | | | | R/W/HS | R/W/HS | R/W/HS |
| Reset | | | | | | 0 | 0 | 0 |

**Bit 2 – ZCDIF** Zero-Cross Detect (ZCD) Interrupt Flag

| Value | Description |
|---|---|
| 1 | A ZCD interrupt occurred (must be cleared in software) |
| 0 | A ZCD interrupt has not occurred |

**Bit 1 – ADTIF** ADC Threshold Interrupt Flag

| Value | Description |
|---|---|
| 1 | ADC Threshold interrupt has occurred (must be cleared in software) |
| 0 | ADC Threshold interrupt event has not occurred |

**Bit 0 – ADIF** ADC Interrupt Flag

| Value | Description |
|---|---|
| 1 | ADC interrupt has occurred (must be cleared in software) |
| 0 | ADC interrupt event has not occurred |

**Note:** Interrupt flag bits are set when an Interrupt condition occurs, regardless of the state of its corresponding enable bit or the Global Enable (GIE) bit. User software may ensure the appropriate interrupt flag bits are cleared before enabling an interrupt.

chanhvinh.com

chanhvinh.com



## 12.11 Register Summary - Interrupt Control

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 ... 0x0A | Reserved | | | | | | | | | |
| 0x0B | INTCON | 7:0 | GIE | PEIE | | | | | | INTEDG |
| 0x0C ... 0x8B | Reserved | | | | | | | | | |
| 0x8C | PIR0 | 7:0 | | | TMR0IF | IOCIF | | | | INTF |
| 0x8D | PIR1 | 7:0 | TMR1GIF | TMR1IF | OSFIF | CSWIF | ACTIF | SCANIF | CRCIF | NVMIF |
| 0x8E | PIR2 | 7:0 | | CCP2IF | CCP1IF | TMR6IF | TMR4IF | TMR2IF | TMR3GIF | TMR3IF |
| 0x8F | PIR3 | 7:0 | PWM4IF | PWM4PIF | PWM3IF | PWM3PIF | PWM2IF | PWM2PIF | PWM1IF | PWM1PIF |
| 0x90 | PIR4 | 7:0 | RC1IF | TX1IF | CLC4IF | CLC3IF | CLC2IF | CLC1IF | CWG1IF | NCO1IF |
| 0x91 | PIR5 | 7:0 | CM2IF | CM1IF | BCL2IF | SSP2IF | BCL1IF | SSP1IF | RC2IF | TX2IF |
| 0x92 | PIR6 | 7:0 | | | | | | ZCDIF | ADTIF | ADIF |
| 0x93 ... 0x95 | Reserved | | | | | | | | | |
| 0x96 | PIE0 | 7:0 | | | TMR0IE | IOCIE | | | | INTE |
| 0x97 | PIE1 | 7:0 | TMR1GIE | TMR1IE | OSFIE | CSWIE | ACTIE | SCANIE | CRCIE | NVMIE |
| 0x98 | PIE2 | 7:0 | | CCP2IE | CCP1IE | TMR6IE | TMR4IE | TMR2IE | TMR3GIE | TMR3IE |
| 0x99 | PIE3 | 7:0 | PWM4IE | PWM4PIE | PWM3IE | PWM3PIE | PWM2IE | PWM2PIE | PWM1IE | PWM1PIE |
| 0x9A | PIE4 | 7:0 | RC1IE | TX1IE | CLC4IE | CLC3IE | CLC2IE | CLC1IE | CWG1IE | NCO1IE |
| 0x9B | PIE5 | 7:0 | CM2IE | CM1IE | BCL2IE | SSP2IE | BCL1IE | SSP1IE | RC2IE | TX2IE |
| 0x9C | PIE6 | 7:0 | | | | | | ZCDIE | ADTIE | ADIE |

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



# 13. Power-Saving Modes

The purpose of the Power-Saving modes is to reduce power consumption. There are three Power-Saving modes:

- Doze mode
- Sleep mode
- Idle mode

## 13.1 Doze Mode

Doze mode allows for power saving by reducing CPU operation and Program Flash Memory (PFM) access, without affecting peripheral operation. Doze mode differs from Sleep mode because the band gap and system oscillators continue to operate, while only the CPU and PFM are affected. The reduced execution saves power by eliminating unnecessary operations within the CPU and memory.

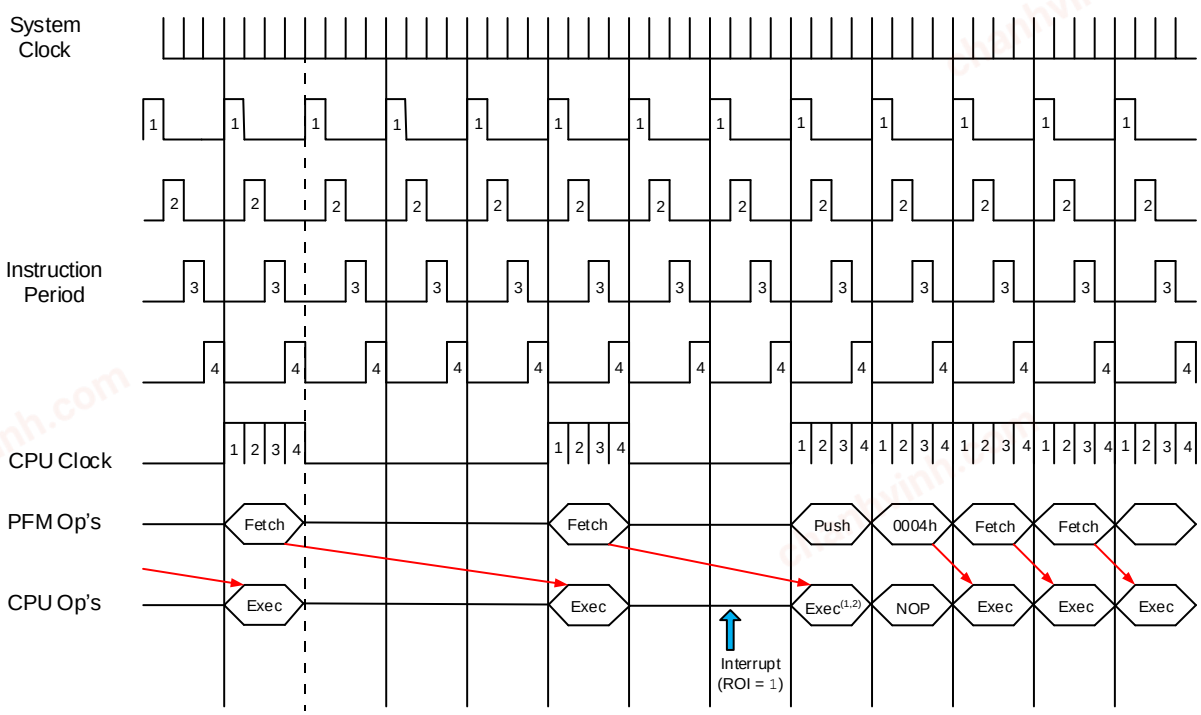
When the Doze Enable bit is set (DOZEN = 'b1) the CPU executes only one instruction cycle out of every N cycles as defined by the DOZE bits. For example, if DOZE = 001, the instruction cycle ratio is 1:4. The CPU and memory execute for one instruction cycle and then lay Idle for three instruction cycles. During the unused cycles, the peripherals continue to operate at the system clock speed.

### 13.1.1 Doze Operation

The Doze operation is illustrated in Figure 13-1. As with normal operation, the instruction is fetched for the next instruction cycle while the previous instruction is executed. The Q-clocks to the peripherals continue throughout the periods in which no instructions are fetched or executed. The following configuration settings apply for this example:

- Doze enabled (DOZEN = 1)
- CPU instruction cycle to peripheral instruction cycle ratio of 1:4
- Recover-on-Interrupt enabled (ROI = 1)

**Figure 13-1. Doze Mode Operation Example**



chanhvinh.com



**Notes:**

1. Multicycle instructions are executed to completion before fetching `0x0004`.
2. If the prefetched instruction clears GIE, the ISR will not occur, but DOZEN is still cleared and the CPU will resume execution at full speed.

### 13.1.2 Interrupts During Doze

System behavior for interrupts that may occur during Doze mode are configured using the ROI and DOE bits. Refer to the example below for details about system behavior in all cases for a transition from Main to ISR back to Main.

**Example 13-1. Doze Software Example**

```
// Mainline operation
bool somethingToDo = FALSE;

void main() {
    initializeSystem();
    // DOZE = 64:1 (for example)
    // ROI = 1;
    GIE = 1; // enable interrupts
    while (1) {
        // If ADC completed, process data
        if (somethingToDo) {
            doSomething();
            DOZEN = 1; // resume low-power
        }
    }
}
// Data interrupt handler

void interrupt() {
    // DOZEN = 0 because ROI = 1
    if (ADIF) {
        somethingToDo = TRUE;
        DOE = 0; // make main() go fast
        ADIF = 0;
    }
    // else check other interrupts...
    if (TMR0IF) {
        timerTick++;
        DOE = 1; // make main() go slow
        TMR0IF = 0;
    }
}
```

**Note:** User software can change the DOE bit in the ISR.

## 13.2 Sleep Mode

Sleep mode provides the greatest power savings because both the CPU and selected peripherals cease to operate. However, some peripheral clocks continue to operate during Sleep. The peripherals that use those clocks also continue to operate. Sleep mode is entered by executing the `SLEEP` instruction, while the IDLEN bit is clear. Upon entering Sleep mode, the following conditions exist:

1. The WDT will be cleared, but keeps running if enabled for operation during Sleep.
2. The $\overline{PD}$ bit of the STATUS register is cleared.
3. The $\overline{TO}$ bit of the STATUS register is set.
4. The CPU clock is disabled.
5. LFINTOSC, SOSC, HFINTOSC and ADCRC are unaffected. Peripherals using them may continue operation during Sleep.
6. I/O ports maintain the status they had before Sleep was executed (driving high, low, or high-impedance).
7. Resets other than WDT are not affected by Sleep mode.

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



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

To minimize current consumption, consider the following conditions:

- I/O pins must not be floating
- External circuitry sinking current from I/O pins
- Internal circuitry sourcing current to I/O pins
- Current draw from pins with internal weak pull-ups
- Peripherals using clock source unaffected by Sleep

I/O pins that are high-impedance inputs need to be pulled to $V_{DD}$ or $V_{SS}$ externally to avoid switching currents caused by floating inputs. Examples of internal circuitry that might be consuming current include modules such as the DAC and FVR peripherals.

### 13.2.1 Wake-Up from Sleep

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

1. External Reset input on $\overline{\text{MCLR}}$ pin, if enabled.
2. BOR Reset, if enabled.
3. Low-Power Brown-out Reset (LPBOR), if enabled.
4. POR Reset.
5. Windowed Watchdog Timer, if enabled.
6. All interrupt sources except clock switch interrupt can wake up the part.

**Important:** The first five events will cause a device Reset. The last event in the list is considered a continuation of program execution. Fore more information about determining whether a device Reset or wake-up event occurred, refer to the **"Resets"** chapter.

When the `SLEEP` instruction is being executed, the next instruction (PC + 2) is prefetched. For the device to wake up through an interrupt event, the corresponding Interrupt Enable bit must be enabled in the PIEx register. Wake-up will occur regardless of the state of the Global Interrupt Enable (GIE) bit. If the GIE bit is disabled, the device will continue execution at the instruction after the `SLEEP` instruction. If the GIE bit is enabled, the device executes the instruction after the `SLEEP` instruction and then call the Interrupt Service Routine (ISR).

**Important:** It is recommended to add a `NOP` as the immediate instruction after the `SLEEP` instruction.

The WDT is cleared when the device wakes up from Sleep, regardless of the source of wake-up. Upon a wake-from-Sleep event, the core will wait for a combination of three conditions before beginning execution. The conditions are:

- PFM Ready
- System Clock Ready
- BOR Ready (unless BOR is disabled)

### 13.2.2 Wake-Up Using Interrupts

When global interrupts are disabled (GIE cleared) and any interrupt source, with the exception of the clock switch interrupt, 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

© 2023 Microchip Technology Inc. and its subsidiaries

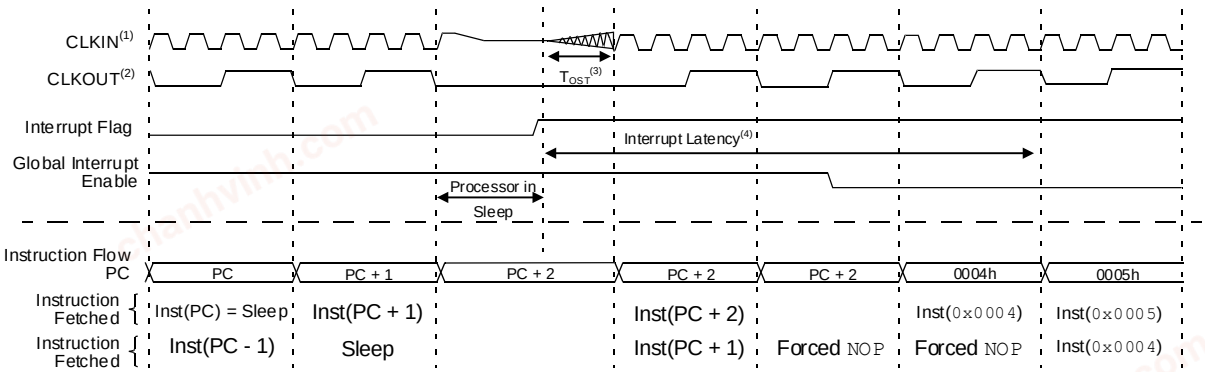
chanhvinh.com



  - The SLEEP instruction will execute as a NOP
  - The WDT and WDT prescaler will not be cleared
  - The $\overline{TO}$ bit of the STATUS register will not be set
  - The $\overline{PD}$ bit of the STATUS register will not be cleared
- If the interrupt occurs during or after the execution of a SLEEP instruction
  - The SLEEP instruction will be completely executed
  - The device will immediately wake up from Sleep
  - The WDT and WDT prescaler will be cleared
  - The $\overline{TO}$ bit of the STATUS register will be set
  - The $\overline{PD}$ bit of the STATUS register will be cleared

In the event where flag bits were checked before executing a SLEEP instruction, it may be possible for flag bits to have 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 13-2. Wake-Up from Sleep through Interrupt**




**Notes:**

1. External clock - High, Low mode assumed.
2. CLKOUT is shown here for timing reference.
3. $T_{OST}$ = 1024 $T_{OSC}$. This delay does not apply to EC and INTOSC Oscillator modes.
4. GIE = 1 assumed. In this case after wake-up, the processor calls the ISR at 0x0004. If GIE = 0, execution will continue in-line.

## 13.3 Idle Mode

When the IDLEN bit is clear, the SLEEP instruction will put the device into full Sleep mode. When IDLEN is set, the SLEEP instruction will put the device into Idle mode. In Idle mode, the CPU and memory operations are halted, but the peripheral clocks continue to run. This mode is similar to Doze mode, except that in Idle both the CPU and program memory are shut off.

**Important:**

1. Peripherals using $F_{OSC}$ will continue to operate while in Idle (but not in Sleep). Peripherals using HFINTOSC:LFINTOSC will continue running in both Idle and Sleep.
2. When the Clock Out Enable ($\overline{CLKOUTEN}$) Configuration bit is cleared, the CLKOUT pin will continue operating while in Idle.

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



### 13.3.1 Idle and Interrupts

Idle mode ends when an interrupt occurs (even if global interrupts are disabled), but IDLEN is not changed. The device can re-enter Idle by executing the `SLEEP` instruction. If Recover-on-Interrupt is enabled (ROI = `1`), the interrupt that brings the device out of Idle also restores full-speed CPU execution when Doze is also enabled.

### 13.3.2 Idle and WWDT

When in Idle, the WWDT Reset is blocked and will instead wake the device. The WWDT wake-up is not an interrupt, therefore ROI does not apply.

**Important:** The WWDT can bring the device out of Idle, in the same way it brings the device out of Sleep. The DOZEN bit is not affected.

## 13.4 Peripheral Operation in Power-Saving Modes

All selected clock sources and the peripherals running from them are active in both Idle and Doze modes. Only in Sleep mode, both the $F_{OSC}$ and $F_{OSC}/4$ clocks are unavailable. However, all other clock sources enabled specifically or through peripheral clock selection before the part enters Sleep, remain operating in Sleep.

## 13.5 Register Definitions: Power-Savings Control

© 2023 Microchip Technology Inc.
and its subsidiaries

chanhvinh.com



### 13.5.1 CPUDOZE

**Name:** CPUDOZE
**Offset:** 0x028C

Doze and Idle Register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | IDLEN | DOZEN | ROI | DOE | | DOZE[2:0] | | |
| Access | R/W | R/W/HC/HS | R/W | R/W/HC/HS | | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | | 0 | 0 | 0 |

**Bit 7 – IDLEN** Idle Enable

| Value | Description |
|---|---|
| 1 | A SLEEP instruction places device into Idle mode |
| 0 | A SLEEP instruction places the device into Sleep mode |

**Bit 6 – DOZEN** Doze Enable(1)

| Value | Description |
|---|---|
| 1 | Places devices into Doze setting |
| 0 | Places devices into Normal mode |

**Bit 5 – ROI** Recover-on-Interrupt(1)

| Value | Description |
|---|---|
| 1 | Entering the Interrupt Service Routine (ISR) makes DOZEN = 0 |
| 0 | Entering the Interrupt Service Routine (ISR) does not change DOZEN |

**Bit 4 – DOE** Doze-on-Exit(1)

| Value | Description |
|---|---|
| 1 | Exiting the ISR makes DOZEN = 1 |
| 0 | Exiting the ISR does not change DOZEN |

**Bits 2:0 – DOZE[2:0]** Ratio of CPU Instruction Cycles to Peripheral Instruction Cycles

| Value | Description |
|---|---|
| 111 | 1:256 |
| 110 | 1:128 |
| 101 | 1:64 |
| 100 | 1:32 |
| 011 | 1:16 |
| 010 | 1:8 |
| 001 | 1:4 |
| 000 | 1:2 |

**Note:**
1. When ROI = 1 or DOE = 1.

chanhvinh.com



## 13.6 Register Summary - Power-Savings Control

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 ... 0x028B | Reserved | | | | | | | | | |
| 0x028C | CPUDOZE | 7:0 | IDLEN | DOZEN | ROI | DOE | | DOZE[2:0] | | |

chanhvinh.com

chanhvinh.com



# 14. WWDT - Windowed Watchdog Timer

A 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. A Watchdog Timer is typically used to recover the system from unexpected events. The Windowed Watchdog Timer (WWDT) differs from nonwindowed operation in that `CLRWDT` instructions are only accepted when they are performed within a specific window during the time-out period.

The WWDT has the following features:

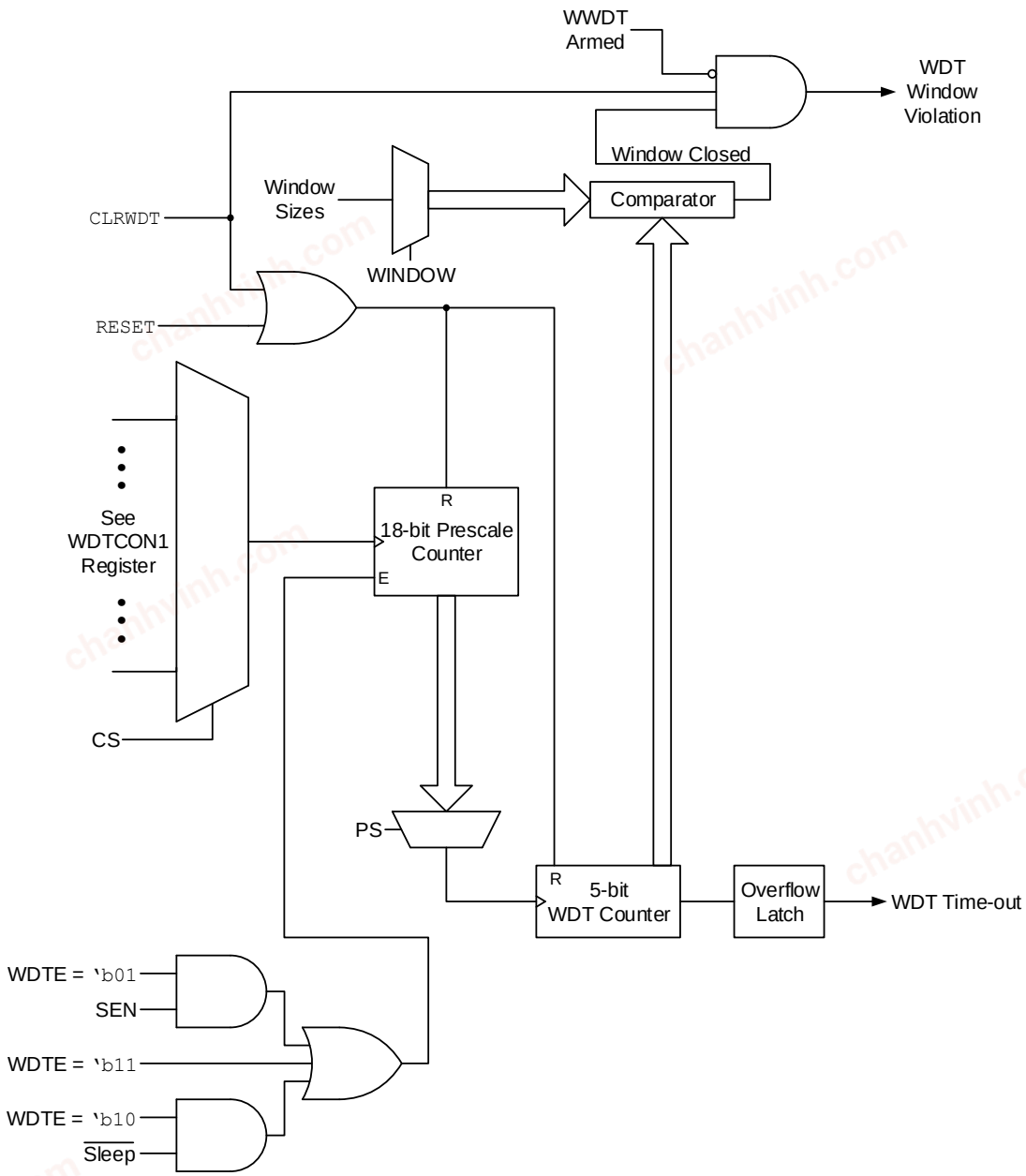
- Selectable clock source
- Multiple operating modes
  - WWDT is always on
  - WWDT is off when in Sleep
  - WWDT is controlled by software
  - WWDT is always off
- Configurable time-out period from 1 ms to 256s (nominal)
- Configurable window size from 12.5% to 100% of the time-out period
- Multiple Reset conditions

© 2023 Microchip Technology Inc.
and its subsidiaries

chanhvinh.com



**Figure 14-1. Windowed Watchdog Timer Block Diagram**




## 14.1 Independent Clock Source

The WWDT can derive its time base from either the 31 KHz LFINTOSC or 31.25 kHz MFINTOSC internal oscillators, depending on the value of WDT Operating Mode (WDTE) Configuration bits. If WDTE = 'b1x, then the clock source will be enabled depending on the WDTCCS Configuration bits. If WDTE = 'b01, the SEN bit will be set by software to enable WWDT, and the clock source is enabled by the CS bits. Time intervals in this chapter are based on a minimum nominal interval of 1 ms. See the device Electrical Specifications for LFINTOSC and MFINTOSC tolerances.

chanhvinh.com



## 14.2 WWDT Operating Modes

The Windowed Watchdog Timer module has four operating modes that are controlled by the WDTE Configuration bit. The table below summarizes the different WWDT operating modes.

**Table 14-1. WWDT Operating Modes**

| WDTE | SEN | Device Mode | WWDT Mode |
|---|---|---|---|
| 11 | X | X | Active |
| 10 | X | Awake | Active |
| | | Sleep | Disabled |
| 01 | 1 | X | Active |
| | 0 | X | Disabled |
| 00 | X | X | Disabled |

### 14.2.1 WWDT Is Always On

When the WDTE Configuration bits are set to 'b11, the WWDT is always on. WWDT protection is active during Sleep.

### 14.2.2 WWDT Is Off in Sleep

When the WDTE Configuration bits are set to 'b10, the WWDT is on, except in Sleep mode. WWDT protection is not active during Sleep.

### 14.2.3 WWDT Controlled by Software

When the WDTE Configuration bits are set to 'b01, the WWDT is controlled by the SEN bit. WWDT protection is unchanged by Sleep. See Table 14-1 for more details.

## 14.3 Time-Out Period

When the WDTCPS Configuration bits are set to the default value of 'b11111, the PS bits set the time-out period from 1 ms to 256 seconds (nominal). If any value other than the default value is assigned to the WDTCPS Configuration bits, then the timer period will be based on the WDTCPS Configuration bits. After a Reset, the default time-out period is 2s.

## 14.4 Watchdog Window

The Windowed Watchdog Timer has an optional Windowed mode that is controlled by either the WDTCWS Configuration bits or the WINDOW bits. In the Windowed mode (WINDOW < 'b1111), 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 WWDT Reset, similar to a WWDT time-out. See Figure 14-2 for an example.

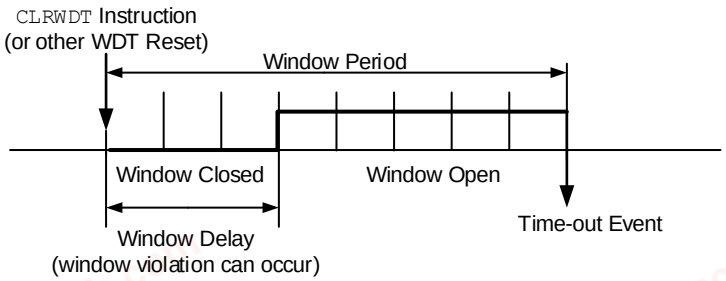
When the WDTCWS Configuration bits are 'b111, then the window size is controlled by the WINDOW bits, otherwise the window size is controlled by the WDTCWS bits. The five Most Significant bits of the WDTTMR register are used to determine whether the window is open, as defined by the window size. In the event of a window violation, a Reset will be generated and the WDTWV bit of the PCON0 register will be cleared. This bit is set by a POR and can be set by software.

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



**Figure 14-2. Window Period and Delay**




## 14.5 Clearing the Watchdog Timer

The Watchdog Timer is cleared when any of the following conditions occur:

- Any Reset
- A valid CLRWDT instruction is executed
- The device enters Sleep
- The devices exits Sleep by Interrupt
- The WWDT is disabled
- The Oscillator Start-up Timer (OST) is running
- Any write to the WDTCON0 or WDTCON1 registers

### 14.5.1 CLRWDT Considerations (Windowed Mode)

When in Windowed mode, the WWDT 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 regardless of whether the window is open or not. See Table 14-2 for more information.

## 14.6 Operation During Sleep

When the device enters Sleep, the Watchdog Timer is cleared. If the WWDT is enabled during Sleep, the Watchdog Timer resumes counting. When the device exits Sleep, the Watchdog Timer is cleared again. The Watchdog Timer remains clear until the Oscillator Start-up Timer (OST) completes, if enabled. When a WWDT time-out occurs while the device is in Sleep, no Reset is generated. Instead, the device wakes up and resumes operation. The $\overline{TO}$ and $\overline{PD}$ bits in the STATUS register are changed to indicate the event. The $\overline{RWDT}$ bit in the PCON0 register indicates that a Watchdog Reset has occurred.

**Table 14-2. WWDT Clearing Conditions**

| Conditions | WWDT |
|---|---|
| WDTE = 'b00 | Cleared |
| WDTE = 'b01 and SEN = 0 | |
| WDTE = 'b10 and enter Sleep | |
| CLRWDT Command | |
| Oscillator Fail Detected | |
| Exit Sleep + System Clock = SOSC, EXTRC, INTOSC, EXTCLK | |
| Exit Sleep + System Clock = XT, HS, LP | Cleared until the end of OST |
| Change INTOSC divider (NOSC bits) | Unaffected |

chanhvinh.com



## 14.7 Register Definitions: Windowed Watchdog Timer Control

Long bit name prefixes for the Windowed Watchdog Timer peripherals are shown in the following table. Refer to the "**Long Bit Names**" section in the **"Register and Bit Naming Conventions"** chapter for more information.

**Table 14-3. WWDT Long Bit Name Prefixes**

| Peripheral | Bit Name Prefix |
|---|---|
| WDT | WDT |

chanhvinh.com

chanhvinh.com



### 14.7.1 WDTCON0

**Name:** WDTCON0
**Offset:** 0x018C

Watchdog Timer Control Register 0

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | PS[4:0] | | | | | SEN |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | q | q | q | q | q | 0 |

**Bits 5:1 – PS[4:0]** Watchdog Timer Prescaler Select[2]

| Value | Description |
|---|---|
| 11111 to 10011 | Reserved. Results in minimum interval (1 ms) |
| 10010 | 1:8388608 ($2^{23}$) (Interval 256s nominal) |
| 10001 | 1:4194304 ($2^{22}$) (Interval 128s nominal) |
| 10000 | 1:2097152 ($2^{21}$) (Interval 64s nominal) |
| 01111 | 1:1048576 ($2^{20}$) (Interval 32s nominal) |
| 01110 | 1:524288 ($2^{19}$) (Interval 16s nominal) |
| 01101 | 1:262144 ($2^{18}$) (Interval 8s nominal) |
| 01100 | 1:131072 ($2^{17}$) (Interval 4s nominal) |
| 01011 | 1:65536 (Interval 2s nominal) (Reset value) |
| 01010 | 1:32768 (Interval 1s nominal) |
| 01001 | 1:16384 (Interval 512 ms nominal) |
| 01000 | 1:8192 (Interval 256 ms nominal) |
| 00111 | 1:4096 (Interval 128 ms nominal) |
| 00110 | 1:2048 (Interval 64 ms nominal) |
| 00101 | 1:1024 (Interval 32 ms nominal) |
| 00100 | 1:512 (Interval 16 ms nominal) |
| 00011 | 1:256 (Interval 8 ms nominal) |
| 00010 | 1:128 (Interval 4 ms nominal) |
| 00001 | 1:64 (Interval 2 ms nominal) |
| 00000 | 1:32 (Interval 1 ms nominal) |

**Bit 0 – SEN** Software Enable/Disable for Watchdog Timer

| Value | Condition | Description |
|---|---|---|
| x | If WDTE = 1x | This bit is ignored |
| 1 | If WDTE = 01 | WDT is turned on |
| 0 | If WDTE = 01 | WDT is turned off |
| x | If WDTE = 00 | This bit is ignored |

**Notes:**

1. When the WDTCPS Configuration bits = 'b11111, the Reset value (q) of WDTPS is 'b01011. Otherwise, the Reset value of WDTPS is equal to the WDTCPS in Configuration bits.
2. When the WDTCPS in Configuration bits ≠ 'b11111, these bits are read-only.

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



### 14.7.2 WDTCON1

**Name:** WDTCON1
**Offset:** 0x018D

Watchdog Timer Control Register 1

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | CS[2:0] | | | | WINDOW[2:0] | | |
| Access | | R/W | R/W | R/W | | R/W | R/W | R/W |
| Reset | | q | q | q | | q | q | q |

**Bits 6:4 – CS[2:0]** Watchdog Timer Clock Select[1,3]

| CS | Clock Source |
|---|---|
| 111-100 | Reserved |
| 011 | EXTOSC |
| 010 | SOSC |
| 001 | MFINTOSC (32 kHz) |
| 000 | LFINTOSC (31 kHz) |

**Bits 2:0 – WINDOW[2:0]** Watchdog Timer Window Select[2,4]

| WINDOW | 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 |

**Notes:**

1. When the WDTCCS in Configuration bits = 'b111, the Reset value of WDTCS is 'b000.
2. The Reset value (q) of WINDOW is determined by the value of WDTCWS in the Configuration bits.
3. When the WDTCCS in Configuration bits ≠ 'b111, these bits are read-only.
4. When the WDTCWS in Configuration bits ≠ 'b111, these bits are read-only.

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



### 14.7.3 WDTPSH

**Name:** WDTPSH
**Offset:** 0x018F

WWDT Prescaler Select Register (Read-Only)

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PSCNTH[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 7:0 – PSCNTH[7:0]** Prescaler Select High Byte[1]

**Note:**

1. The 18-bit WDT prescaler 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 will be read during normal operation.

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



### 14.7.4 WDTPSL

**Name:** WDTPSL
**Offset:** 0x018E

WWDT Prescaler Select Register (Read-Only)

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PSCNTL[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 7:0 – PSCNTL[7:0]** Prescaler Select Low Byte[1]

**Note:**

1. The 18-bit WDT prescaler 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 will be read during normal operation.

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



### 14.7.5 WDTTMR

**Name:** WDTTMR
**Offset:** 0x0190

WDT Timer Register (Read-Only)

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TMR[4:0] | | | | | STATE | PSCNT[17:16] | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 7:3 – TMR[4:0]** Watchdog Window Value

| WINDOW | WDT Window State | | Open Percent |
|---|---|---|---|
| | Closed | Open | |
| 111 | N/A | 00000-11111 | 100 |
| 110 | 00000-00011 | 00100-11111 | 87.5 |
| 101 | 00000-00111 | 01000-11111 | 75 |
| 100 | 00000-01011 | 01100-11111 | 62.5 |
| 011 | 00000-01111 | 10000-11111 | 50 |
| 010 | 00000-10011 | 10100-11111 | 37.5 |
| 001 | 00000-10111 | 11000-11111 | 25 |
| 000 | 00000-11011 | 11100-11111 | 12.5 |

**Bit 2 – STATE** WDT Armed Status

| Value | Description |
|---|---|
| 1 | WDT is armed |
| 0 | WDT is not armed |

**Bits 1:0 – PSCNT[17:16]** Prescaler Select Upper Byte(1)

**Note:**

1. The 18-bit WDT prescaler 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 will not be read during normal operation.

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



## 14.8 Register Summary - WDT Control

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 ... 0x018B | Reserved | | | | | | | | | |
| 0x018C | WDTCON0 | 7:0 | | | PS[4:0] | | | | | SEN |
| 0x018D | WDTCON1 | 7:0 | | CS[2:0] | | | | WINDOW[2:0] | | |
| 0x018E | WDTPSL | 7:0 | PSCNTL[7:0] | | | | | | | |
| 0x018F | WDTPSH | 7:0 | PSCNTH[7:0] | | | | | | | |
| 0x0190 | WDTTMR | 7:0 | TMR[4:0] | | | | | STATE | PSCNT[17:16] | |

chanhvinh.com

chanhvinh.com



# 15. NVM - Nonvolatile Memory Control

The Nonvolatile Memory (NVM) module provides run-time read and write access to the Program Flash Memory (PFM), Data Flash Memory (DFM), and Configuration bits. PFM includes the program memory and user ID space. DFM is also referred to as EEPROM which is accessed one byte at a time and the erase before write is automatic.

NVM is accessible using both FSR and INDF registers, or through the NVMREG register interface (see Table 15-1).

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.

PFM and DFM can be protected in two ways: code protection and write protection. Code protection (Configuration bits $\overline{CP}$ (PFM)/$\overline{CPD}$ (DFM)) disables PFM/DFM read and write access through an external device programmer. Write protection prevents user software writes to NVM areas tagged for protection by the $\overline{WRTn}$ Configuration bits. Code protection does not affect the self-write and erase functionality, whereas write protection does. Attempts to write a protected location will set the WRERR bit. Code protection and write protection can only be reset on a Bulk Erase performed by an external programmer.

The Bulk Erase command is used to completely erase program memory. The Bulk Erase command can only be issued through an external programmer. There is no run time access for this command.

If the device is code-protected and a Bulk Erase command for the configuration memory is issued; all other memory regions are also erased. Refer to the **"Programming Specifications"** document for more details.

**Table 15-1. NVM Organization and Access Information**

| Main Values | | | NVMREG Access | | | FSR Access | |
|---|---|---|---|---|---|---|---|
| Memory Function | Memory Type | Program Counter (PC), ICSP™ Address | NVMREGS bit (NVMCON1) | NVMADR[14:0] | Allowed Operations | FSR Address | FSR Programming Access |
| Reset Vector | Program Flash Memory | 0x0000 | 0 | 0x0000 | Read/Write | 0x8000 | Read-Only |
| User Memory | | 0x0001 | 0 | 0x0001 | | 0x8001 | |
| | | 0x0003 | | 0x0003 | | 0x8003 | |
| INT Vector | | 0x0004 | 0 | 0x0004 | | 0x8004 | |
| User Memory | | 0x0005 | 0 | 0x0005 | | 0x8005 | |
| | | 0x3FFF(1) | | 0x3FFF(1) | | 0xFFFF | |
| User ID | Program Flash Memory | 0x8000 | 1 | 0x0000 | Read/Write | No Access | |
| | | 0x8003 | | 0x0003 | | | |
| Reserved | — | — | — | 0x0004 | — | | |
| Revision ID | Hard Coded in Program Flash Memory | 0x8005 | 1 | 0x0005 | Read | | |
| Device ID | | 0x8006 | 1 | 0x0006 | | | |
| CONFIG1 | Program Flash Memory | 0x8007 | 1 | 0x0007 | Read/Write | | |
| CONFIG2 | | 0x8008 | 1 | 0x0008 | | | |
| CONFIG3 | | 0x8009 | 1 | 0x0009 | | | |
| CONFIG4 | | 0x800A | 1 | 0x000A | | | |
| CONFIG5 | | 0x800B | 1 | 0x000B | | | |
| DIA and DCI | Hard Coded in Program Flash Memory | 0x8100 | 1 | 0x0100 | Read | | |
| | | 0x82FF | 1 | 0x02FF | | | |
| EEPROM | Data Flash Memory | 0xF000 | 1 | 0x7000 | Read/Write | 0x7000 | Read |
| | | 0xF0FF | | 0x70FF | | 0x70FF | |

**Note:**

1. The maximum Program Flash Memory address for the PIC16F171 family is 0x3FFF.

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



## 15.1 Program Flash Memory (PFM)

The Program Flash Memory (PFM) is readable, writable and erasable over the entire $V_{DD}$ range.

PFM consists of the following regions:

- User program memory (read/write)
- Configuration Words (read/write)
- Device ID (read-only)
- Revision ID (read-only)
- User ID (read-write)
- Device Information Area (read-only)
- Device Configuration Information (read-only)

PFM can be read and/or written to through:

- CPU instruction fetch (read-only)
- FSR/INDF indirect access (read-only)
- NVMREG access (read-write)
- In-Circuit Serial Programming™ (ICSP™) (external read-write)

It is important to understand the program memory structure for erase and programming operations. Program memory is arranged in rows. A row consists of 32 14-bit program memory words. A row is the minimum size that can be erased by user software. A Bulk Erase command cannot be issued from user code.

Read operations return a single word of memory. Write and erase operations are done on a row basis. Program memory will erase to a logic '1' and program to a logic '0'.

All or a portion of a row can be programmed. Data to be written into the program memory row is written to 14-bit wide data write latches. These latches are not directly accessible, but may be loaded via sequential writes to the NVMDATH:NVMDATL register pair.

**Important:** To modify only a portion of a previously programmed row, the contents of the entire row must be read. Then, the new data and retained data can be written into the write latches to reprogram the row of 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.

Writing or erasing program memory will cease instruction fetches until the operation is complete. The program memory cannot be accessed during the write or erase, so code cannot execute. An internal programming timer controls the write time of program memory writes and erases.

A value written to program memory does not need to be a valid instruction. Executing a program memory location that forms an invalid instruction results in a NOP.

### 15.1.1 FSR and INDF Access

The File Select (FSR) and INDF registers allow indirect access to the Program Flash Memory. Indirect addressing is a mode in which the memory address in the instruction is determined by another register. The value of the FSR registers is used to determine the memory address location to be accessed.

#### 15.1.1.1 FSR Read

The FSRs are used to provide read access to program memory.

Program memory is accessed by loading the FSRxH:FSRxL register pair with the address to be read, and setting bit 7 of the FSRxH register to '1'. When a MOVIW instruction, or any instruction that accesses INDFx, is executed, the value loaded into the FSRx register pair points to the location in program memory to be accessed. If the FSRx register pair points to an INDFx register, the read will return '0'.

Reading from NVM requires one instruction cycle. The CPU operation is suspended during the read and resumes immediately after. Read operations return a single byte of memory.

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



#### 15.1.1.2 FSR Write

Writing/erasing the NVM through the FSR registers (e.g., the `MOVWI` instruction) is not supported in the PIC16F171 microcontroller family.

### 15.1.2 NVMREG Access

The NVMREG interface allows read/write access to all the locations accessible by FSRs, read/write access to the User ID locations, and read-only access to the device identification, revision, and configuration data.

Writing or erasing of NVM via the NVMREG interface is prevented when the device is write-protected.

#### 15.1.2.1 NVMREG Read Operation

To read a NVM location using the NVMREG interface, the user must:

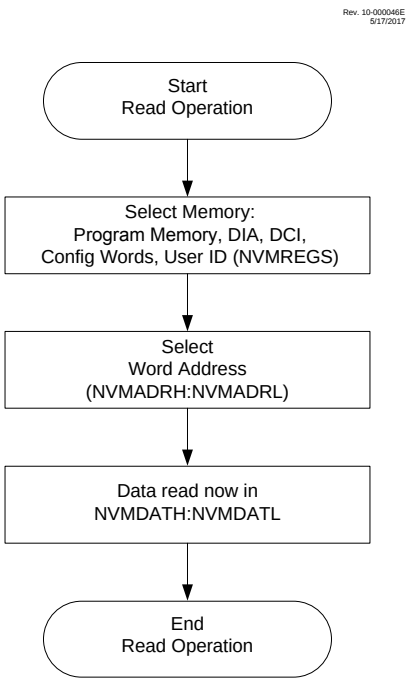
1. Clear the NVMREGS bit if the user intends to access program memory locations, or set NMVREGS if the user intends to access User ID or configuration locations.
2. Write the desired address into the NVMADRH:NVMADRL register pair.
3. Set the RD bit to initiate the read.

Once the read control bit is set, the CPU operation is suspended during the read and resumes immediately after. The data is available in the very next cycle, in the NVMDATH:NVMDATL register pair; therefore, it can be read as two bytes in the following instructions.

The NVMDATH:NVMDATL register pair will hold this value until another read or until it is written to by the user.

Upon completion, the RD bit is cleared by hardware.

**Figure 15-1. Program Flash Memory Read Sequence**




**Example 15-1. Program Memory Read**

```
// This code block will read 1 word of program memory

NVMCON1bits.NVMREGS = 0;          // Point to PFM
NVMADR = PFM_ADDRESS;             // Load NVMADRH:NVMADRL with PFM address
NVMCON1bits.RD = 1;               // Initiate read cycle
PFM_DATA_LOW = NVMDATL;           // PFM data low byte
PFM_DATA_HIGH = NVMDATH;          // PFM data high byte
```

chanhvinh.com



### 15.1.2.2 NVM Unlock Sequence

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

- PFM Row Erase
- Write of PFM write latches to PFM memory
- Write of PFM write latches to User IDs
- Write to Configuration Words

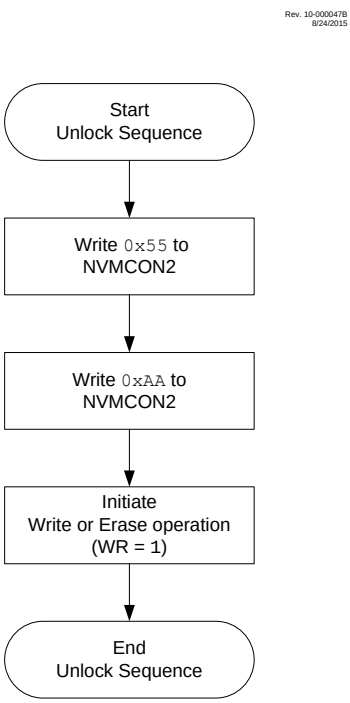
The unlock sequence consists of the following steps and must be completed in order:

- Write 55h to NVMCON2
- Write AAh to NMVCON2
- Set the WR bit

Once the WR bit is set, the processor will stall internal operations until the operation is complete and then resume with the next instruction.

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

**Figure 15-2. NVM Unlock Sequence**




**Example 15-2. NVM Unlock Sequence**

```
NVMCON1bits.WREN = 1;           // Enable write/erase
INTCONbits.GIE = 0;             // Disable global interrupts

// The next three steps are the required unlock sequence
NVMCON2 = 0x55;                 // First unlock code
NVMCON2 = 0xAA;                 // Second unlock code
NVMCON1bits.WR = 1;             // Initiate write/erase cycle

INTCONbits.GIE = 1;             // Enable global interrupts
NVMCON1bits.WREN = 0;           // Disable further write/erase cycles
```

chanhvinh.com



**Note:** Sequence begins when NVMCON2 is written; the three unlock steps must occur in the cycle-accurate order shown. If the timing of the sequence is corrupted by an interrupt or a debugger Halt, the action will not take place.

#### 15.1.2.3 NVMREG Erase of Program Memory

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 to program memory. To erase a program memory row:

1. Clear the NVMREGS bit to erase program memory locations, or set the NMVREGS bit to erase User ID locations.
2. Write the desired address into the NVMADRH:NVMADRL register pair.
3. Set the FREE and WREN bits.
4. Perform the unlock sequence as described in the NVM Unlock Sequence section.

If the program memory address is write-protected, the WR bit will be cleared and the erase operation will not take place.

While erasing program memory, the CPU operation is suspended and resumes when the operation is complete. Upon completion, the NVMIF bit is set, and an interrupt will occur if the NVMIE bit is also set.

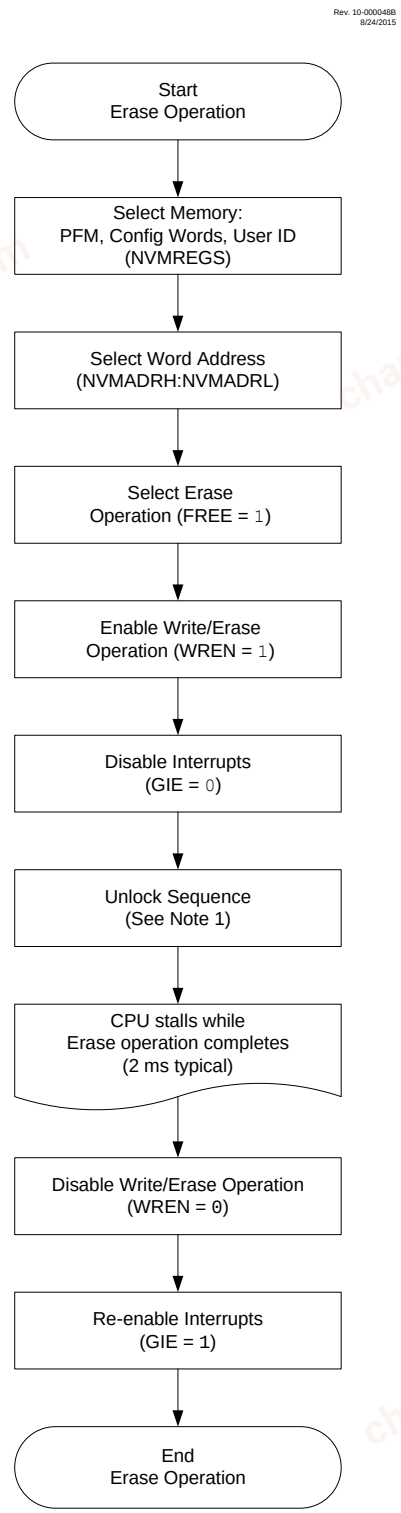
Write latch data is not affected by erase operations, and WREN will remain unchanged.

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



**Figure 15-3. NVM Erase Sequence**




**Note:**

1. See the NVM Unlock Sequence section.

chanhvinh.com



**Example 15-3. Erasing One Row of Program Flash Memory**

```
NVMCON1bits.NVMREGS = 0;          // Point to PFM
NVMADR = PFM_ADD;                 // 14-bit PFM address
NVMCON1bits.FREE = 1;             // Specify an erase operation
NVMCON1bits.WREN = 1;             // Enable write/erase cycle
INTCONbits.GIE = 0;               // Disable interrupts during unlock sequence

//The next three steps are the required unlock sequence
NVMCON2 = 0x55;                   // First unlock code
NVMCON2 = 0xAA;                   // Second unlock code
NVMCON1bits.WR = 1;               // Initiate write/erase cycle

INTCONbits.GIE = 1;               // Enable interrupts
NVMCON1bits.WREN = 1;             // Disable writes
```

#### 15.1.2.4 NVMREG Write to Program Memory

Program memory is programmed using the following steps:

1. Load the address of the row to be programmed into NVMADRH:NVMADRL.
2. Load each write latch with data via the NMVDATH:NVMDATL registers.
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 15-4 for more details.

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

The following steps must 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 NVMDATH:NVMDATL 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.

**Important:** 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.
2. Clear the NVMREGS bit.
3. Set the LWLO bit. When the LWLO bit is set (LWLO = 1), the write sequence will only load the write latches and will not initiate the write to Program Flash Memory.
4. Load the NVMADRH:NVMADRL register pair with the address of the location to be written.
5. Load the NVMDATH:NVMDATL register pair with the program memory data to be written.
6. Execute the unlock sequence. The write latch is now loaded.
7. Increment the NVMADRH:NVMADRL register pair to point to the next location.
8. Repeat steps 5 through 7 until all except the last write latch has been loaded.
9. Clear the LWLO bit. When the LWLO bit is clear (LWLO = 0), the write sequence will initiate the write to Program Flash Memory.
10. Load the NVMDATH:NVMDATL register pair with the program memory data to be written.

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com

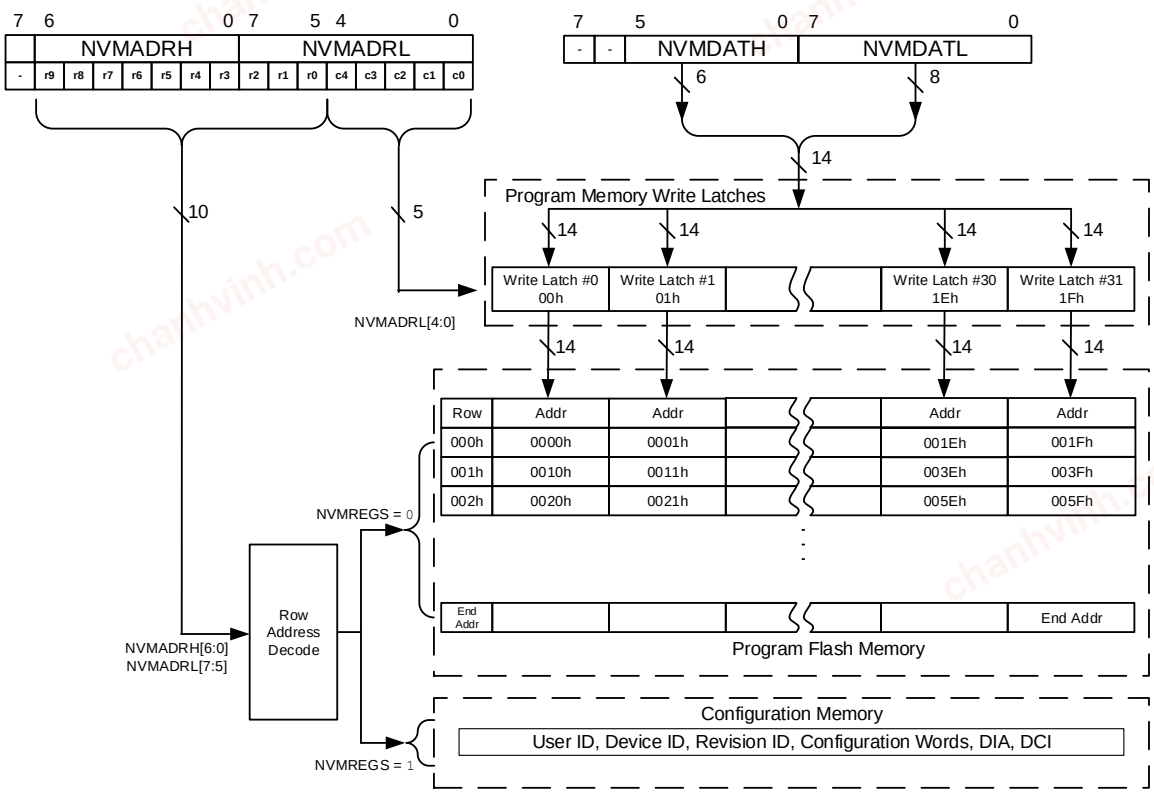


11. Execute the unlock sequence. The entire program memory latch content is now written to Flash program memory.

**Important:** 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 15-4. The initial address is loaded into the NVMADRH:NVMADRL register pair; the data is loaded using indirect addressing.

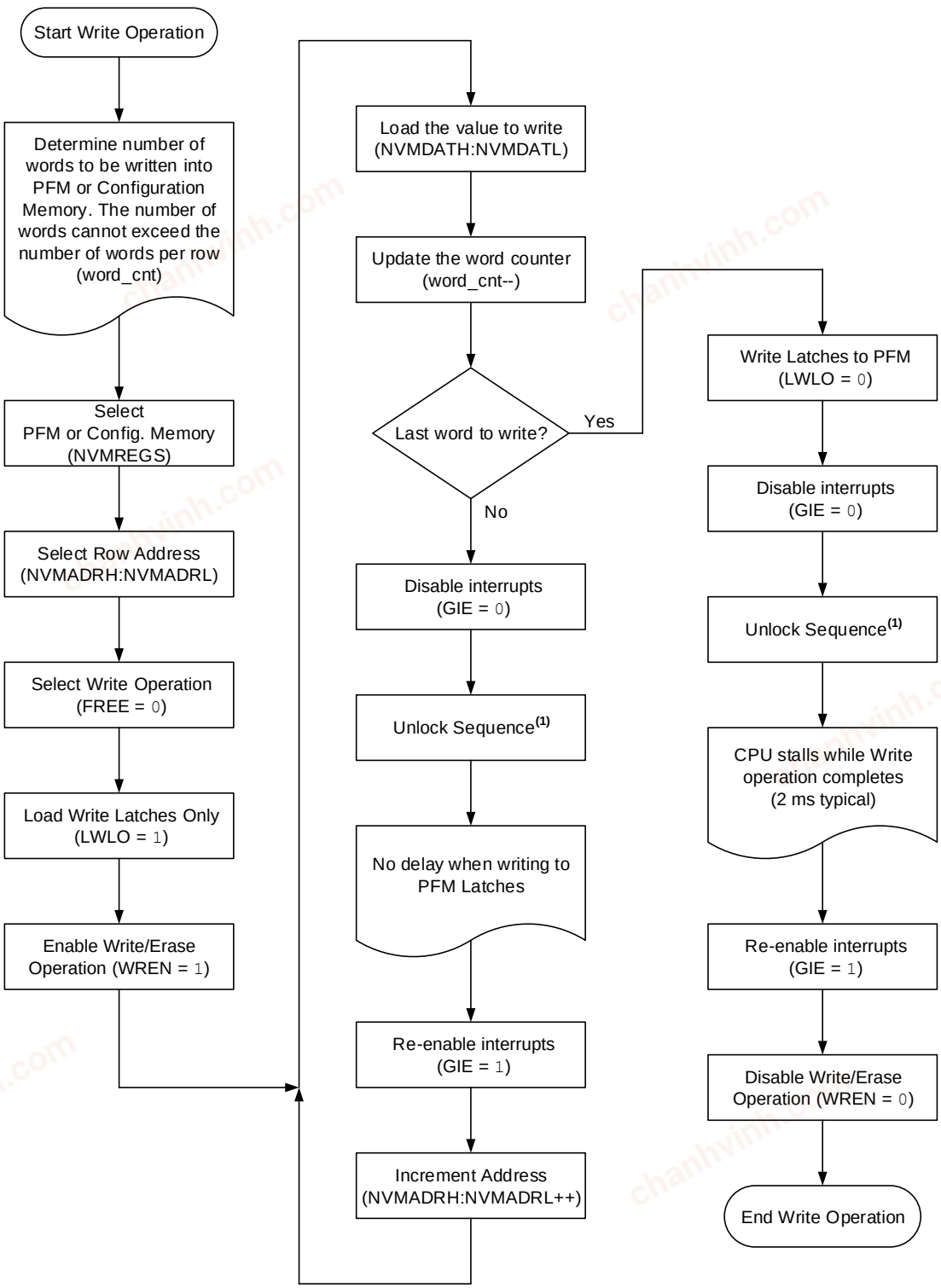
**Figure 15-4. NVMREG Writes to Program Flash Memory with 32 Write Latches**



chanhvinh.com



**Figure 15-5. Program Flash Memory Write Sequence**



chanhvinh.com



**Note:**

1. See the NVM Unlock Sequence section.

**Example 15-4. Writing to Program Flash Memory**

```
INTCONbits.GIE = 0;                 // Disable interrupts

// PFM row must be erased before writes can occur
NVMCON1bits.NVMREGS = 0;            // Point to PFM
NVMADR = PFMStartAddress;           // Must start at beginning of PFM row
NVMCON1bits.FREE = 1;               // Specify an erase operation
NVMCON1bits.WREN = 1;               // Allow erase cycle

// Required unlock sequence
NVMCON2 = 0x55;
NVMCON2 = 0xAA;
NVMCON1bits.WR = 1;

NVMCON1bits.LWLO = 1;               // Load write latches

// Write to the data latches
for (i = 0; i < PFM_ROW_SIZE; i++)
{
    NVMADR = PFMStartAddress;       // Load starting address
    NVMDAT = PFM_WRITE_DATA;        // Load data

    // Required unlock sequence
    NVMCON2 = 0x55;
    NVMCON2 = 0xAA;
    NVMCON1bits.WR = 1;

    PFMStartAddress++;              // Increment address
    if(i = (PFM_ROW_SIZE - 1))      // All latches loaded?
    {
        NVMCON1bits.LWLO = 0;       // Start PFM write
    }
}

NVMCON1bits.WREN = 0;               // Disable writes
INTCONbits.GIE = 1;                 // Enable interrupts
```

### 15.1.2.5 Modifying Flash Program Memory

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

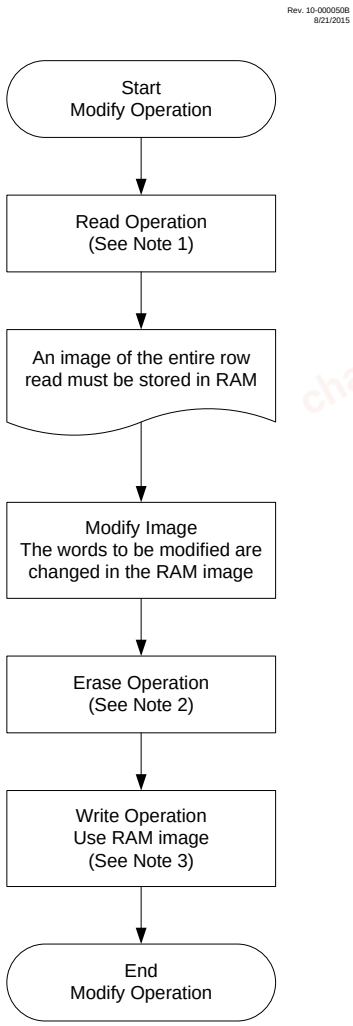
1. Load the starting address of the row to be modified.
2. Read the existing data from the row into a RAM image.
3. Modify the RAM image to contain the new data to be written into program memory.
4. Load the starting address of the row to be rewritten.
5. Erase the program memory row.
6. Load the write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

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



**Figure 15-6. Program Flash Memory Modify Sequence**




**Notes:**

1. See Figure 15-1.
2. See Figure 15-3.
3. See Figure 15-5.

### 15.1.2.6 NVMREG Access to DIA, DCI, User ID, Device ID, Revision ID, and Configuration Words

NVMREGS can be used to access the following memory regions:

- Device Information Area (DIA)
- Device Configuration Information (DCI)
- User ID region
- Device ID and Revision ID
- Configuration Words

The value of NVMREGS is set to '1' to access these regions. The memory regions listed above will be pointed to by PC[15] = 1, but not all addresses reference valid data. Different access may exist for reads and writes. Refer to the table below. When read access is initiated on an address outside the parameters listed in the following table, the NVMDATH: NVMDATL register pair is cleared, reading back '0's.

chanhvinh.com



**Table 15-2. NVMREG Access to DIA, DCI, User ID, Device ID, Revision ID and Configuration Words (NVMREGS = 1)**

| Address | Function | Read Access | Write Access |
|---|---|---|---|
| 0x8000 - 0x8003 | User IDs | Yes | Yes |
| 0x8005 - 0x8006 | Device ID/Revision ID | Yes | No |
| 0x8007 - 0x800B | Configuration Words 1-5 | Yes | Yes |
| 0x8100 - 0x82FF | DIA and DCI | Yes | No |
| 0xF000 - 0xF0FF | EEPROM | Yes | Yes |

#### 15.1.2.7 Write Verify

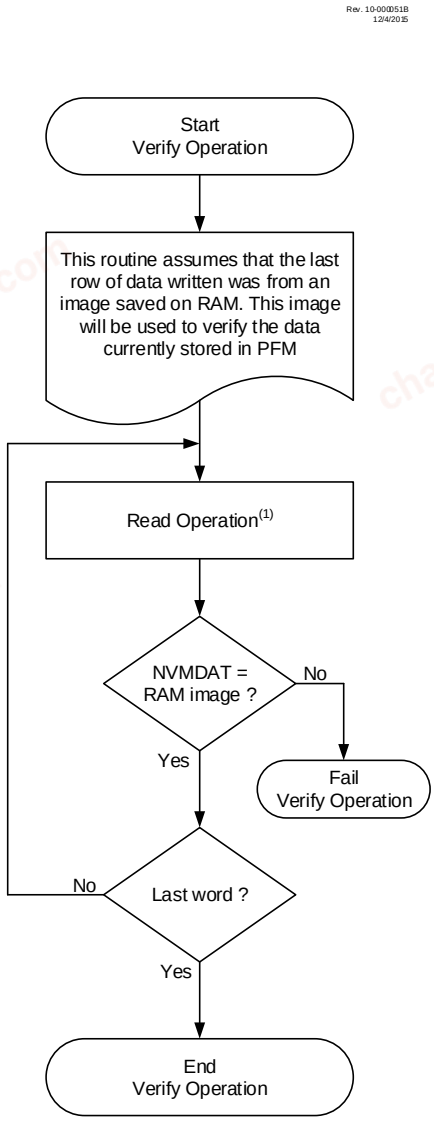
It is considered good programming practice to verify that program memory writes agree with the intended value. Since program memory is stored as a full row then the stored program memory contents are compared with the intended data stored in RAM after the last write is complete.

chanhvinh.com

chanhvinh.com



**Figure 15-7. Program Flash Memory Write Verify Sequence**




**Note:**

1. See Figure 15-1.

### 15.1.2.8 WRERR Bit

The WRERR bit can be used to determine if a write error occurred. WRERR will be set if one of the following conditions occurs:

- If WR is set while the NVMADRH:NMVADRL points to a write-protected address
- A Reset occurs while a self-write operation was in progress
- An unlock sequence was interrupted

The WRERR bit is normally set by hardware, but can be set by the user for test purposes. Once set, WRERR must be cleared in software.

chanhvinh.com



**Table 15-3. Actions for PFM When WR = 1**

| Free | LWLO | Actions for PFM when WR = 1 | Comments |
|---|---|---|---|
| 1 | x | Erase the 32-word row of NVMADRH:NVMADRL location. | • If WP is enabled, WR is cleared and WRERR is set<br>• All 32 words are erased<br>• NVMDATH:NVMDATL is ignored |
| 0 | 1 | Copy NVMDATH:NVMDATL to the write latch corresponding to NVMADR LSBs. | • Write protection is ignored<br>• No memory access occurs |
| 0 | 0 | Write the write-latch data to PFM row. | • If WP is enabled, WR is cleared and WRERR is set<br>• Write latches are reset to 0x3FFF<br>• NVMDATH:NVMDATL is ignored |

## 15.2 Data Flash Memory (DFM)

The Data Flash Memory is a nonvolatile memory array, also referred to as EEPROM. The DFM is mapped above program memory space. The DFM can be accessed using the FSRs or NVM Special Function Registers (SFRs). The DFM is readable and writable during normal operation over the entire $V_{DD}$ range.

The DFM can only be read and written one byte at a time. When interfacing to the data memory block, the NVMDATL register holds the 8-bit data for read/write and the NVMADR register pair holds the address of the DFM location being accessed.

The DFM is rated for high erase/write cycle endurance. A byte write automatically erases the location and writes the new data (erase-before-write). The write time is controlled by an internal programming timer; it will vary with voltage and temperature as well as from device-to-device. Refer to the data EEPROM memory parameters in the **"Electrical Specifications"** chapter for more information.

### 15.2.1 FSR and INDF Access

The File Select (FSR) and INDF registers allow read-only access to the DFM. The value of the FSR registers is used to determine the memory address location to be accessed.

#### 15.2.1.1 FSR DFM Read

DFM is accessed by loading the FSRxH:FSRxL register pair with the address to be read, and loading bits [7-4] of the FSRxH register with a value of '0111'. When a MOVIW instruction, or any instruction that accesses INDFx, is executed, the value loaded into the FSRx register pair points to the location in program memory to be accessed. If the FSRx register pair points to an unimplemented address location, the read will return '0'.

#### 15.2.1.2 FSR DFM Write

Writing/erasing the DFM through the FSR registers (e.g., the MOVWI instruction) is not supported in the PIC16F171 microcontroller family.

### 15.2.2 NVMREG Access

The NVMREG interface allows read/write access to the DFM.

Writing or erasing of DFM via the NVMREG interface is prevented when the DFM is write-protected (Configuration bit $\overline{\text{WRTD}}$ = '0').

#### 15.2.2.1 NVMREG Read Operation

To read a DFM location using the NVMREG interface, the user must:

1. Set the NVMREGS bit.
2. Write the desired address into the NVMADRH:NVMADRL register pair.
3. Set the RD bit to initiate the read.

Once the read control bit is set, the CPU operation is suspended during the read and resumes immediately after. The data is available in the very next cycle, in the NVMDATL register; therefore, it can be read in the following instructions.

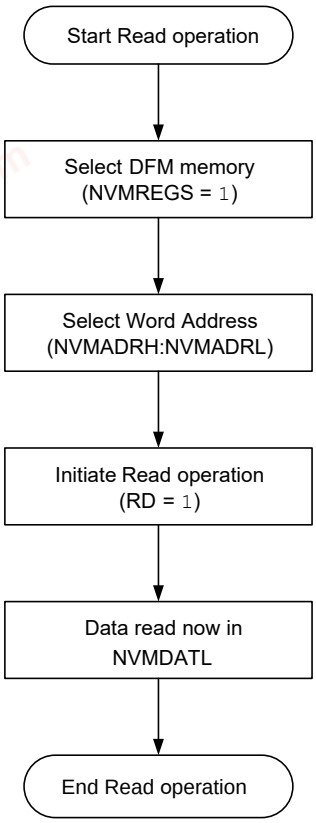
chanhvinh.com



The NVMDATL register will hold this value until another read or until it is written to by the user.

Upon completion, the RD bit is cleared by hardware.

**Figure 15-8. Program Flash Memory Read Sequence**




**Example 15-5. DFM Read**

```
// This code block will read 1 word (byte) of DFM

NVMCON1bits.NVMREGS = 1;         // Point to DFM
NVMADR = DFM_ADDRESS;            // NVMADRH = 0x70, NVMADRL = DFM address
NVMCON1bits.RD = 1;              // Initiate read cycle
MY_8BIT_VARIABLE = NVMDATL;      // DFM data byte is in NVMDATL
```

### 15.2.2.2 NVM Unlock Sequence

The unlock sequence is a mechanism that protects the DFM from unintended self-write programming or erasing. The sequence must be executed and completed without interruption to successfully complete an erase or write operation.

The unlock sequence consists of the following steps and must be completed in order:

- Write 55h to NVMCON2
- Write AAh to NMVCON2
- Set the WR bit

Once the WR bit is set, the processor will stall internal operations until the operation is complete and then resume with the next instruction.

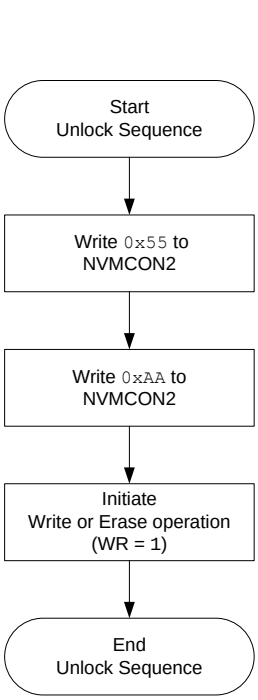
Since the unlock sequence must not be interrupted, global interrupts must be disabled prior to the unlock sequence and re-enabled after the unlock sequence is completed.

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



**Figure 15-9. NVM Unlock Sequence**




**Example 15-6. NVM Unlock Sequence**

```
NVMCON1bits.WREN = 1;           // Enable write/erase
INTCONbits.GIE = 0;             // Disable global interrupts

// The next three steps are the required unlock sequence
NVMCON2 = 0x55;                 // First unlock code
NVMCON2 = 0xAA;                 // Second unlock code
NVMCON1bits.WR = 1;             // Initiate write/erase cycle

INTCONbits.GIE = 1;             // Enable global interrupts
NVMCON1bits.WREN = 0;           // Disable further write/erase cycles
```

**Note:** Sequence begins when NVMCON2 is written; the three unlock steps must occur in the cycle-accurate order shown. If the timing of the sequence is corrupted by an interrupt or a debugger Halt, the action will not take place.

#### 15.2.2.3 NVMREG Erase of DFM

DFM can be erased by writing '0xFF' to all locations that need to be erased. To erase a DFM word:

1. Set the NVMREGS bit.
2. Write the desired address into the NVMADRH:NVMADRL register pair.
3. Set the WREN bit.
4. Perform the unlock sequence as described in the NVM Unlock Sequence section.
5. Set the WR bit.

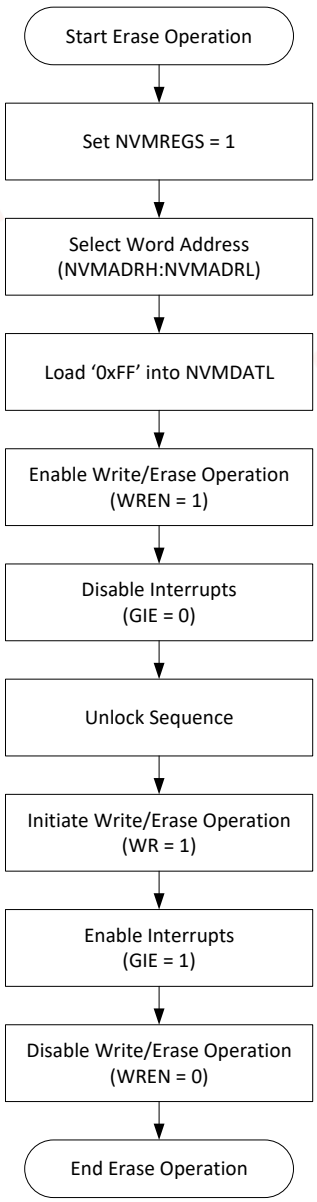
A single byte (0xFF) is written into the DFM location. The DFM circuitry automatically erases the memory location before performing the write operation. If the DFM address is write-protected, the WR bit will be cleared and the write operation will not take place. Upon completion, hardware clears the WR bit, the NVMIF bit is set, and an interrupt will occur if the NVMIE bit is also set.

© 2023 Microchip Technology Inc.
and its subsidiaries

chanhvinh.com



**Figure 15-10. DFM Erase Sequence**




**Example 15-7. Erasing One Byte of Data Flash Memory**

```
// Code sequence to erase one byte of DFM

NVMCON1bits.NVMREGS = 1;        // Point to DFM
NVMADR = DFM_ADDR;              // NVMADRH = 0x70, NVMADRL = DFM address
NVMDATL = 0xFF;                 // Load NVMDATL with 0xFF
NVMCON1bits.WREN = 1;           // Allows program/erase cycles
INTCONbits.GIE = 0;             // Disable interrupts

// Perform required unlock sequence
NVMCON2 = 0x55;
NVMCON2 = 0xAA;
```

© 2023 Microchip Technology Inc.
and its subsidiaries

chanhvinh.com



```
NVMCON1bits.WR = 1;             // Begin program/erase cycle

INTCONbits.GIE = 1;             // Restore interrupt enable bit value
NVMCON1bits.WREN = 0;           // Disable program/erase

// Verify byte erase operation success and call the recovery function if needed
if (NVMCON1bits.WRERR)
{
    ERASE_FAULT_RECOVERY();
}
```

#### 15.2.2.4 NVMREG Write to DFM

DFM is programmed using the following steps:

1. Set the NVMREGS bit.
2. Load the NVMADRH:NVMADRL registers with the DFM address.
3. Load the NVMDATL register with the data byte to be written.
4. Set the WREN bit to allow write/erase cycles.
5. Disable the GIE bit.
6. Perform the NVM Unlock Sequence.
7. Set the WR bit.

A single byte (0xFF) is written into the DFM location. The DFM circuitry automatically erases the memory location before performing the write operation. If the DFM address is write-protected, the WR bit will be cleared and the write operation will not take place.

While erasing/writing data memory, the CPU operation is suspended and resumes when the operation is complete. Upon completion, hardware clears the WR bit, the NVMIF bit is set, and an interrupt will occur if the NVMIE bit is also set.

#### 15.2.2.5 WRERR Bit

The WRERR bit can be used to determine if a write error occurred. WRERR will be set if one of the following conditions occurs:

- If WR is set while the NVMADRH:NMVADRL points to a write-protected DFM address
- A Reset occurs while a self-write operation was in progress
- An unlock sequence was interrupted

The WRERR bit is normally set by hardware, but can be set by the user for test purposes. Once set, WRERR must be cleared in software.

## 15.3 Register Definitions: Nonvolatile Memory Control

chanhvinh.com



### 15.3.1 NVMADR

**Name:** NVMADR
**Offset:** 0x1C8C

Nonvolatile Memory Address Register

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | NVMADR[14:8] | | | | | | |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | NVMADR[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 14:0 – NVMADR[14:0]** NVM Address Bits

**Notes:**

1. The individual bytes in this multibyte register can be accessed with the following register names:
   – NVMADRH: Accesses the high byte NVMADR[15:8]
   – NVMADRL: Accesses the low byte NVMADR[7:0].
2. Bit [15] is undefined while WR = `1`.

chanhvinh.com



### 15.3.2 NVMDAT

**Name:** NVMDAT
**Offset:** 0x1C8E

Nonvolatile Memory Data Register

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | NVMDAT[13:8] | | | | | |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | x | x | x | x | x | x |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | NVMDAT[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | x | x | x | x | x | x | x | x |

**Bits 13:0 – NVMDAT[13:0]** NVM Data bits
Reset States: POR/BOR = xxxxxxxxxxxxxx
All Other Resets = uuuuuuuuuuuuuu

**Notes:** The individual bytes in this multibyte register can be accessed with the following register names:

- NVMDATH: Accesses the high byte NVMDAT[13:8]
- NVMDATL: Accesses the low byte NVMDAT[7:0]

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



### 15.3.3 NVMCON1

**Name:** NVMCON1
**Offset:** 0x1C90

Nonvolatile Memory Control 1 Register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | NVMREGS | LWLO | FREE | WRERR | WREN | WR | RD |
| Access | | R/W | R/W | R/S/HC | R/W/HS | R/W | R/S/HC | R/S/HC |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 6 – NVMREGS** NVM Region Selection

| Value | Description |
|---|---|
| 1 | Access DFM (EEPROM), DIA, DCI, Configuration, User ID, Revision ID, and Device ID Registers |
| 0 | Access Program Flash Memory |

**Bit 5 – LWLO** Load Write Latches Only

| Value | Condition | Description |
|---|---|---|
| 1 | When FREE = 0 | The next WR command updates the write latch for this word within the row; no memory operation is initiated |
| 0 | When FREE = 0 | The next WR command writes data or erases |
| – | Otherwise: | This bit is ignored |

**Bit 4 – FREE** Program Flash Memory Erase Enable

| Value | Description |
|---|---|
| 1 | Performs an erase operation with the next WR command; the 32-word pseudo-row containing the indicated address is erased (to all 1s) to prepare for writing |
| 0 | The next WR command writes without erasing |

**Bit 3 – WRERR**
Write-Reset Error Flag(1,2,3)

| Value | Description |
|---|---|
| 1 | A write operation error has occurred |
| 0 | All write operations have completed normally |

**Bit 2 – WREN** Program/Erase Enable

| Value | Description |
|---|---|
| 1 | Allows program/erase cycles |
| 0 | Inhibits programming/erasing of program Flash |

**Bit 1 – WR** Write Control(4,5,6)

| Value | Description |
|---|---|
| 1 | Initiates the program/erase operation at the corresponding NVM location |
| 0 | NVM program/erase operation is complete and inactive |

**Bit 0 – RD** Read Control

| Value | Description |
|---|---|
| 1 | Initiates a read at address = NVMADR |
| 0 | NVM read operation is complete and inactive |

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



**Notes:**

1. Bit is undefined while WR = 1.
2. Bit must be cleared by software; hardware will not clear this bit.
3. Bit may be written to '1' by the user to implement test sequences.
4. This bit can only be set by following the sequence described in the **"NVM Unlock Sequence"** section.
5. Operations are self-timed and the WR bit is cleared by hardware when complete.
6. Once a write operation is initiated, setting this bit to zero will have no effect.

chanhvinh.com

chanhvinh.com



### 15.3.4 NVMCON2

**Name:** NVMCON2
**Offset:** 0x1C91

Nonvolatile Memory Control 2 Register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | NVMCON2[7:0] | | | | | | | |
| Access | WO | WO | WO | WO | WO | WO | WO | WO |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 7:0 – NVMCON2[7:0]** Flash Memory Unlock Pattern bits

**Note:** To unlock writes, a `0x55` must be written first followed by an `0xAA` before setting the WR bit of the NVMCON1 register. The value written to this register is used to unlock the writes.

chanhvinh.com

chanhvinh.com



## 15.4 Register Summary - NVM Control

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 ... 0x1C8B | Reserved | | | | | | | | | |
| 0x1C8C | NVMADR | 7:0 | NVMADR[7:0] | | | | | | | |
| | | 15:8 | | NVMADR[14:8] | | | | | | |
| 0x1C8E | NVMDAT | 7:0 | NVMDAT[7:0] | | | | | | | |
| | | 15:8 | | | NVMDAT[13:8] | | | | | |
| 0x1C90 | NVMCON1 | 7:0 | | NVMREGS | LWLO | FREE | WRERR | WREN | WR | RD |
| 0x1C91 | NVMCON2 | 7:0 | NVMCON2[7:0] | | | | | | | |

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



# 16. I/O Ports

## 16.1 Overview

**Table 16-1. Port Availability per Device**

| Device | PORTA | PORTB | PORTC | PORTD | PORTE |
|---|---|---|---|---|---|
| 28-pin devices | • | • | • | | • |
| 40/44-pin devices | • | • | • | • | • |

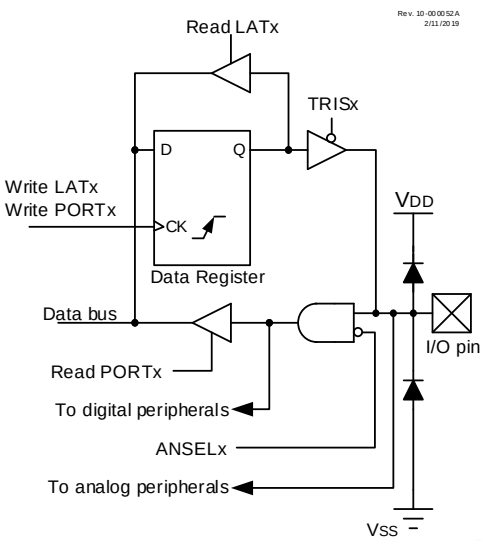
Each port has eight registers to control the operation. These registers are:

- PORTx registers (reads the levels on the pins of the device)
- LATx registers (output latch)
- TRISx registers (data direction)
- ANSELx registers (analog select)
- WPUx registers (weak pull-up)
- INLVLx (input level control)
- SLRCONx registers (slew rate control)
- ODCONx registers (open-drain control)

In this section, the generic names such as PORTx, LATx, TRISx, etc. can be associated with PORTA, PORTB, PORTC, etc., depending on availability per device.

A simplified model of a generic I/O port, without the interfaces to other peripherals, is shown in the following figure:

**Figure 16-1. Generic I/O Port Operation**




## 16.2 PORTx - Data Register

PORTx is a bidirectional port, and its corresponding data direction register is TRISx.

Reading the PORTx register 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, and

chanhvinh.com



this value is modified, then written to the PORT data latch (LATx). The PORT data latch LATx holds the output port data and contains the latest value of a LATx or PORTx write. The example below shows how to initialize PORTA.

**Example 16-1. Initializing PORTA in Assembly**

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

    BANKSEL    PORTA        ;
    CLRF       PORTA        ;Clear PORTA
    BANKSEL    LATA         ;
    CLRF       LATA         ;Clear Data Latch
    BANKSEL    ANSELA       ;
    CLRF       ANSELA       ;Enable digital drivers
    BANKSEL    TRISA        ;
    MOVLW      B'00111000'  ;Set RA[5:3] as inputs
    MOVWF      TRISA        ;and set others as outputs
```

**Example 16-2. Initializing PORTA in C**

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

    PORTA  = 0x00;       // Clear PORTA
    LATA   = 0x00;       // Clear Data Latch
    ANSELA = 0x00;       // Enable digital drivers
    TRISA  = 0x38;       // Set RA[5:3] as inputs and set others as outputs
```

**Important:** Most PORT pins share functions with device peripherals, both analog and digital. 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.

## 16.3 LATx - Output Latch

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 the values held in the I/O PORT latches, while a read of the PORTx register reads the actual I/O pin value.

**Important:** As a general rule, output operations to a port must use the LAT register to avoid Read-Modify-Write issues. For example, a bit set or clear operation reads the port, modifies the bit, and writes the result back to the port. When two bit operations are executed in succession, output loading on the changed bit may delay the change at the output in which case the bit will be misread in the second bit operation and written to an unexpected level. The LAT registers are isolated from the port loading and therefore changes are not delayed.

## 16.4 TRISx - Direction Control

The TRISx register controls the PORTx pin output drivers, even when the pins are being used as analog inputs. The user must ensure the bits in the TRISx register are set when using the pins as analog inputs. I/O pins configured as analog inputs always read '0'.

chanhvinh.com



Setting a TRISx bit (TRISx = 1) will make the corresponding PORTx pin an input (i.e., disable the output driver). Clearing a TRISx bit (TRISx = 0) will make the corresponding PORTx pin an output (i.e., it enables output driver and puts the contents of the output latch on the selected pin).

## 16.5 ANSELx - Analog Control

Ports that support analog inputs have an associated ANSELx register. The ANSELx register is used to configure the Input mode of an I/O pin to analog. Setting an ANSELx bit high will disable the digital input buffer associated with that bit and cause the corresponding input value to always read '0', whether the value is read in PORTx register or selected by PPS as a peripheral input.

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.

The state of the ANSELx bits has no effect on digital or analog 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 PORTx register.

**Important:** The ANSELx 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 changed to '0' by the user.

## 16.6 WPUx - Weak Pull-Up Control

The WPUx register controls the individual weak pull-ups for each PORT pin. When a WPUx bit is set (WPUx = 1), the weak pull-up will be enabled for the corresponding pin. When a WPUx bit is cleared (WPUx = 0), the weak pull-up will be disabled for the corresponding pin.

## 16.7 INLVLx - Input Threshold Control

The INLVLx register controls the input voltage threshold for each of the available PORTx 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 PORTx register and also the level at which an interrupt-on-change occurs, if that feature is enabled. Refer to the I/O Ports table in the **"Electrical Specifications"** chapter for more details on threshold levels..

**Important:** Changing the input threshold selection must 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.

## 16.8 SLRCONx - Slew Rate Control

The SLRCONx register controls the slew rate option for each PORT pin. Slew rate for each PORT pin can be controlled independently. When a SLRCONx bit is set (SLRCONx = 1), the corresponding PORT pin drive is slew rate limited. When a SLRCONx bit is cleared (SLRCONx = 0), the corresponding PORT pin drive slews at the maximum rate possible.

## 16.9 ODCONx - Open-Drain Control

The ODCONx register controls the open-drain feature of the port. Open-drain operation is independently selected for each pin. When a ODCONx bit is set (ODCONx = 1), the corresponding port output becomes an open-drain driver

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



capable of sinking current only. When a ODCONx bit is cleared (ODCONx = 0), the corresponding port output pin is the standard push-pull drive capable of sourcing and sinking current.

**Important:** It is necessary to set open-drain control when using the pin for $I^2C$.

## 16.10 Edge Selectable Interrupt-on-Change

An interrupt can be generated by detecting a signal at the PORT pin that has either a rising edge or a falling edge. Individual pins can be independently configured to generate an interrupt. Refer to the "**IOC - Interrupt-on-Change**" chapter for more details.

## 16.11 $I^2C$ Pad Control

For this family of devices, the $I^2C$ specific pads are available on RB1, RB2, RC3, and RC4 (28/40/44-pin deivces), and RD0 and RD1 (40/44-pin devices only) pins. The $I^2C$ characteristics of each of these pins is controlled by the RxyI2C registers. These characteristics include enabling $I^2C$ specific slew rate (over standard GPIO slew rate), selecting internal pull-ups for $I^2C$ pins, and selecting appropriate input threshold as per SMBus specifications.

**Important:** Any peripheral using the $I^2C$ pins reads the $I^2C$ input levels when enabled via RxyI2C.

## 16.12 I/O Priorities

Each pin defaults to the data latch after Reset. Other functions are selected with the Peripheral Pin Select logic. Refer to the **"PPS - Peripheral Pin Select Module"** chapter for more details.

Analog input functions, such as ADC and comparator 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 ANSELx register. Digital output functions may continue to control the pin when it is in Analog mode.

Analog outputs, when enabled, take priority over digital outputs and force the digital output driver into a High-Impedance state.

The pin function priorities are as follows:

1. Port functions determined by the Configuration bits.
2. Analog outputs (input buffers must be disabled).
3. Analog inputs.
4. Port inputs and outputs from PPS.

## 16.13 $\overline{MCLR}/V_{PP}$/RE3 Pin

The $\overline{MCLR}/V_{PP}$ pin is an input-only pin. Its operation is controlled by the MCLRE Configuration bit. When selected as a PORT pin (MCLRE = 0), it functions as a digital input-only pin; as such, it does not have TRISx and LATx bits associated with its operation. Otherwise, it functions as the device's Master Clear input. In either configuration, the $\overline{MCLR}/V_{PP}$ pin also functions as the programming voltage input pin during high-voltage programming.

The $\overline{MCLR}/V_{PP}$ pin is a read-only bit and will read '1' when MCLRE = 1 (i.e., Master Clear enabled).

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



**Important:** On a Power-on Reset (POR), the $\overline{MCLR}/V_{PP}$ pin is enabled as a digital input-only if Master Clear functionality is disabled.

The $\overline{MCLR}/V_{PP}$ pin has an individually controlled internal weak pull-up. When set, the corresponding WPU bit enables the pull-up. When the $\overline{MCLR}/V_{PP}$ pin is configured as $\overline{MCLR}$ (MCLRE = 1 and LVP = 0), or configured for Low-Voltage Programming (MCLRE = x and LVP = 1), the pull-up is always enabled and the WPU bit has no effect.

## 16.14 Register Definitions: Port Control

chanhvinh.com



### 16.14.1 PORTx

**Name:** PORTx

PORTx Register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | Rx7 | Rx6 | Rx5 | Rx4 | Rx3 | Rx2 | Rx1 | Rx0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | x | x | x | x | x | x | x | x |

**Bits 0, 1, 2, 3, 4, 5, 6, 7 – Rxn** Port I/O Value
Reset States: POR/BOR = xxxxxxxx
All Other Resets = uuuuuuuu

| Value | Description |
|---|---|
| 1 | PORT pin is ≥ $V_{IH}$ |
| 0 | PORT pin is ≤ $V_{IL}$ |

**Important:**

- Writes to PORTx are actually written to the corresponding LATx register. Reads from PORTx register return actual I/O pin values.
- The PORT bit associated with the $\overline{MCLR}$ pin is read-only and will read '1' when the $\overline{MCLR}$ function is enabled (LVP = 1 or (LVP = 0 and MCLRE = 1))
- Refer to the **"Pin Allocation Table"** for details about $\overline{MCLR}$ pin and pin availability per port
- Unimplemented bits will read back as '0'
- Bits RB6 and RB7 read '1' while in Debug mode.

chanhvinh.com



### 16.14.2 LATx

**Name:** LATx

Output Latch Register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | LATx7 | LATx6 | LATx5 | LATx4 | LATx3 | LATx2 | LATx1 | LATx0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | x | x | x | x | x | x | x | x |

**Bits 0, 1, 2, 3, 4, 5, 6, 7 – LATxn** Output Latch Value
Reset States: POR/BOR = xxxxxxxx
All Other Resets = uuuuuuuu

**Important:**
- Writes to LATx are equivalent to writes to the corresponding PORTx register. Reads from LATx register return register values, not I/O pin values.
- Refer to the **"Pin Allocation Table"** for details about pin availability per port
- Unimplemented bits will read back as '0'

chanhvinh.com



### 16.14.3 TRISx

**Name:** TRISx

Tri-State Control Register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TRISx7 | TRISx6 | TRISx5 | TRISx4 | TRISx3 | TRISx2 | TRISx1 | TRISx0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

**Bits 0, 1, 2, 3, 4, 5, 6, 7 – TRISxn** Port I/O Tri-state Control

| Value | Description |
|---|---|
| 1 | PORTx output driver is disabled. PORTx pin configured as an input (tri-stated) |
| 0 | PORTx output driver is enabled. PORTx pin configured as an output |

**Important:**

- The TRIS bit associated with the $\overline{MCLR}$ pin is read-only and the value is '1'
- Refer to the **"Pin Allocation Table"** for details about $\overline{MCLR}$ pin and pin availability per port
- Unimplemented bits will read back as '0'

chanhvinh.com



### 16.14.4 ANSELx

**Name:** ANSELx

Analog Select Register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ANSELx7 | ANSELx6 | ANSELx5 | ANSELx4 | ANSELx3 | ANSELx2 | ANSELx1 | ANSELx0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

**Bits 0, 1, 2, 3, 4, 5, 6, 7 – ANSELxn** Analog Select on RX Pin

| Value | Description |
|---|---|
| 1 | Analog input. Pin is assigned as analog input. Digital input buffer disabled. |
| 0 | Digital I/O. Pin is assigned to port or digital special function. |

**Important:**

- When setting a pin as an analog input, the corresponding TRIS bit must be set to Input mode to allow external control of the voltage on the pin
- Refer to the "**Pin Allocation Table**" for details about pin availability per port
- Unimplemented bits will read back as '0'

chanhvinh.com



### 16.14.5 WPUx

**Name:** WPUx

Weak Pull-Up Register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | WPUx7 | WPUx6 | WPUx5 | WPUx4 | WPUx3 | WPUx2 | WPUx1 | WPUx0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 0, 1, 2, 3, 4, 5, 6, 7 – WPUxn** Weak Pull-up PORTx Control

| Value | Description |
|---|---|
| 1 | Weak pull-up enabled |
| 0 | Weak pull-up disabled |

**Important:**

- The weak pull-up device is automatically disabled if the pin is configured as an output, but this register remains unchanged
- If MCLRE = 1, the weak pull-up on $\overline{\text{MCLR}}$ pin is always enabled and the corresponding WPU bit is not affected
- Refer to the **"Pin Allocation Table"** for details about pin availability per port
- Unimplemented bits will read back as '0'

chanhvinh.com



### 16.14.6 INLVLx

**Name:** INLVLx

Input Level Control Register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | INLVLx7 | INLVLx6 | INLVLx5 | INLVLx4 | INLVLx3 | INLVLx2 | INLVLx1 | INLVLx0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

**Bits 0, 1, 2, 3, 4, 5, 6, 7 – INLVLxn** Input Level Select on RX Pin

| Value | Description |
|---|---|
| 1 | ST input used for port reads and interrupt-on-change |
| 0 | TTL input used for port reads and interrupt-on-change |

**Important:**

- Refer to the "**Pin Allocation Table**" for details about pin availability per port
- Unimplemented bits will read back as '0'
- Any peripheral using the $I^2C$ pins read the $I^2C$ ST inputs when enabled via RxyI2C.

chanhvinh.com



### 16.14.7 SLRCONx

**Name:** SLRCONx

Slew Rate Control Register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SLRx7 | SLRx6 | SLRx5 | SLRx4 | SLRx3 | SLRx2 | SLRx1 | SLRx0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

**Bits 0, 1, 2, 3, 4, 5, 6, 7 – SLRxn** Slew Rate Control on RX Pin

| Value | Description |
|---|---|
| 1 | PORT pin slew rate is limited |
| 0 | PORT pin slews at maximum rate |

**Important:**
- Refer to the "**Pin Allocation Table**" for details about pin availability per port
- Unimplemented bits will read back as '0'

chanhvinh.com



### 16.14.8 ODCONx

**Name:** ODCONx

Open-Drain Control Register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ODCx7 | ODCx6 | ODCx5 | ODCx4 | ODCx3 | ODCx2 | ODCx1 | ODCx0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 0, 1, 2, 3, 4, 5, 6, 7 – ODCxn** Open-Drain Configuration on Rx Pin

| Value | Description |
|---|---|
| 1 | PORT pin operates as open-drain drive (sink current only) |
| 0 | PORT pin operates as standard push-pull drive (source and sink current) |

**Important:**

- Refer to the "**Pin Allocation Table**" for details about pin availability per port
- Unimplemented bits will read back as '0'

chanhvinh.com



### 16.14.9 RxyI2C

**Name:** RxyI2C

I²C Pad Rxy Control Register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | SLEW | PU[1:0] | | | | TH[1:0] | |
| Access | | R/W | R/W | R/W | | | R/W | R/W |
| Reset | | 0 | 0 | 0 | | | 0 | 0 |

**Bit 6 – SLEW** I²C Specific Slew Rate Limiting Control

| Value | Description |
|---|---|
| 1 | I²C specific slew rate limiting is enabled. Standard pad slew limiting is disabled. The SLRxy bit is ignored |
| 0 | Standard GPIO Slew Rate; enabled/disabled via SLRxy bit |

**Bits 5:4 – PU[1:0]** I²C Pull-Up Selection

| Value | Description |
|---|---|
| 11 | Reserved |
| 10 | 10x current of standard weak pull-up |
| 01 | 2x current of standard weak pull-up |
| 00 | Standard GPIO weak pull-up, enabled via the WPUxy bit |

**Bits 1:0 – TH[1:0]** I²C Input Threshold Selection

| Value | Description |
|---|---|
| 11 | SMBus 3.0 (1.35V) input threshold |
| 10 | SMBus 2.0 (2.1V) input threshold |
| 01 | I²C-specific input thresholds |
| 00 | Standard GPIO Input pull-up, enabled via the INLVLxy registers |

**Important:**
- Refer to the "**Pin Allocation Table**" for details about pin availability per port
- Unimplemented bits will read back as '0'

chanhvinh.com



## 16.15 Register Summary - I/O Ports

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 ... 0x0B | Reserved | | | | | | | | | |
| 0x0C | PORTA | 7:0 | RA7 | RA6 | RA5 | RA4 | RA3 | RA2 | RA1 | RA0 |
| 0x0D | PORTB | 7:0 | RB7 | RB6 | RB5 | RB4 | RB3 | RB2 | RB1 | RB0 |
| 0x0E | PORTC | 7:0 | RC7 | RC6 | RC5 | RC4 | RC3 | RC2 | RC1 | RC0 |
| 0x0F | PORTD | 7:0 | RD7 | RD6 | RD5 | RD4 | RD3 | RD2 | RD1 | RD0 |
| 0x10 | PORTE | 7:0 | | | | | RE3 | RE2 | RE1 | RE0 |
| 0x11 | Reserved | | | | | | | | | |
| 0x12 | TRISA | 7:0 | TRISA7 | TRISA6 | TRISA5 | TRISA4 | TRISA3 | TRISA2 | TRISA1 | TRISA0 |
| 0x13 | TRISB | 7:0 | TRISB7 | TRISB6 | TRISB5 | TRISB4 | TRISB3 | TRISB2 | TRISB1 | TRISB0 |
| 0x14 | TRISC | 7:0 | TRISC7 | TRISC6 | TRISC5 | TRISC4 | TRISC3 | TRISC2 | TRISC1 | TRISC0 |
| 0x15 | TRISD | 7:0 | TRISD7 | TRISD6 | TRISD5 | TRISD4 | TRISD3 | TRISD2 | TRISD1 | TRISD0 |
| 0x16 | TRISE | 7:0 | | | | | Reserved | TRISE2 | TRISE1 | TRISE0 |
| 0x17 | Reserved | | | | | | | | | |
| 0x18 | LATA | 7:0 | LATA7 | LATA6 | LATA5 | LATA4 | LATA3 | LATA2 | LATA1 | LATA0 |
| 0x19 | LATB | 7:0 | LATB7 | LATB6 | LATB5 | LATB4 | LATB3 | LATB2 | LATB1 | LATB0 |
| 0x1A | LATC | 7:0 | LATC7 | LATC6 | LATC5 | LATC4 | LATC3 | LATC2 | LATC1 | LATC0 |
| 0x1B | LATD | 7:0 | LATD7 | LATD6 | LATD5 | LATD4 | LATD3 | LATD2 | LATD1 | LATD0 |
| 0x1C | LATE | 7:0 | | | | | | LATE2 | LATE1 | LATE0 |
| 0x1D ... 0x1E8B | Reserved | | | | | | | | | |
| 0x1E8C | ANSELA | 7:0 | ANSELA7 | ANSELA6 | ANSELA5 | ANSELA4 | ANSELA3 | ANSELA2 | ANSELA1 | ANSELA0 |
| 0x1E8D | WPUA | 7:0 | WPUA7 | WPUA6 | WPUA5 | WPUA4 | WPUA3 | WPUA2 | WPUA1 | WPUA0 |
| 0x1E8E | ODCONA | 7:0 | ODCA7 | ODCA6 | ODCA5 | ODCA4 | ODCA3 | ODCA2 | ODCA1 | ODCA0 |
| 0x1E8F | SLRCONA | 7:0 | SLRA7 | SLRA6 | SLRA5 | SLRA4 | SLRA3 | SLRA2 | SLRA1 | SLRA0 |
| 0x1E90 | INLVLA | 7:0 | INLVLA7 | INLVLA6 | INLVLA5 | INLVLA4 | INLVLA3 | INLVLA2 | INLVLA1 | INLVLA0 |
| 0x1E91 ... 0x1E95 | Reserved | | | | | | | | | |
| 0x1E96 | ANSELB | 7:0 | ANSELB7 | ANSELB6 | ANSELB5 | ANSELB4 | ANSELB3 | ANSELB2 | ANSELB1 | ANSELB0 |
| 0x1E97 | WPUB | 7:0 | WPUB7 | WPUB6 | WPUB5 | WPUB4 | WPUB3 | WPUB2 | WPUB1 | WPUB0 |
| 0x1E98 | ODCONB | 7:0 | ODCB7 | ODCB6 | ODCB5 | ODCB4 | ODCB3 | ODCB2 | ODCB1 | ODCB0 |
| 0x1E99 | SLRCONB | 7:0 | SLRB7 | SLRB6 | SLRB5 | SLRB4 | SLRB3 | SLRB2 | SLRB1 | SLRB0 |
| 0x1E9A | INLVLB | 7:0 | INLVLB7 | INLVLB6 | INLVLB5 | INLVLB4 | INLVLB3 | INLVLB2 | INLVLB1 | INLVLB0 |
| 0x1E9B ... 0x1E9F | Reserved | | | | | | | | | |
| 0x1EA0 | ANSELC | 7:0 | ANSELC7 | ANSELC6 | ANSELC5 | ANSELC4 | ANSELC3 | ANSELC2 | ANSELC1 | ANSELC0 |
| 0x1EA1 | WPUC | 7:0 | WPUC7 | WPUC6 | WPUC5 | WPUC4 | WPUC3 | WPUC2 | WPUC1 | WPUC0 |
| 0x1EA2 | ODCONC | 7:0 | ODCC7 | ODCC6 | ODCC5 | ODCC4 | ODCC3 | ODCC2 | ODCC1 | ODCC0 |
| 0x1EA3 | SLRCONC | 7:0 | SLRC7 | SLRC6 | SLRC5 | SLRC4 | SLRC3 | SLRC2 | SLRC1 | SLRC0 |
| 0x1EA4 | INLVLC | 7:0 | INLVLC7 | INLVLC6 | INLVLC5 | INLVLC4 | INLVLC3 | INLVLC2 | INLVLC1 | INLVLC0 |
| 0x1EA5 ... 0x1EA9 | Reserved | | | | | | | | | |
| 0x1EAA | ANSELD | 7:0 | ANSELD7 | ANSELD6 | ANSELD5 | ANSELD4 | ANSELD3 | ANSELD2 | ANSELD1 | ANSELD0 |
| 0x1EAB | WPUD | 7:0 | WPUD7 | WPUD6 | WPUD5 | WPUD4 | WPUD3 | WPUD2 | WPUD1 | WPUD0 |
| 0x1EAC | ODCOND | 7:0 | ODCD7 | ODCD6 | ODCD5 | ODCD4 | ODCD3 | ODCD2 | ODCD1 | ODCD0 |
| 0x1EAD | SLRCOND | 7:0 | SLRD7 | SLRD6 | SLRD5 | SLRD4 | SLRD3 | SLRD2 | SLRD1 | SLRD0 |
| 0x1EAE | INLVLD | 7:0 | INLVLD7 | INLVLD6 | INLVLD5 | INLVLD4 | INLVLD3 | INLVLD2 | INLVLD1 | INLVLD0 |
| 0x1EAF ... 0x1EB3 | Reserved | | | | | | | | | |
| 0x1EB4 | ANSELE | 7:0 | | | | | | ANSELE2 | ANSELE1 | ANSELE0 |

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x1EB5 | WPUE | 7:0 | | | | | WPUE3 | WPUE2 | WPUE1 | WPUE0 |
| 0x1EB6 | ODCONE | 7:0 | | | | | | ODCE2 | ODCE1 | ODCE0 |
| 0x1EB7 | SLRCONE | 7:0 | | | | | | SLRE2 | SLRE1 | SLRE0 |
| 0x1EB8 | INLVLE | 7:0 | | | | | INLVLE3 | INLVLE2 | INLVLE1 | INLVLE0 |
| 0x1EB9 ... 0x1EE2 | Reserved | | | | | | | | | |
| 0x1EE3 | RB1I2C | 7:0 | | SLEW | PU[1:0] | | | | TH[1:0] | |
| 0x1EE4 | RB2I2C | 7:0 | | SLEW | PU[1:0] | | | | TH[1:0] | |
| 0x1EE5 ... 0x1EEA | Reserved | | | | | | | | | |
| 0x1EEB | RC3I2C | 7:0 | | SLEW | PU[1:0] | | | | TH[1:0] | |
| 0x1EEC | RC4I2C | 7:0 | | SLEW | PU[1:0] | | | | TH[1:0] | |
| 0x1EED | Reserved | | | | | | | | | |
| 0x1EEE | RD0I2C | 7:0 | | SLEW | PU[1:0] | | | | TH[1:0] | |
| 0x1EEF | RD1I2C | 7:0 | | SLEW | PU[1:0] | | | | TH[1:0] | |

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



# 17. IOC - Interrupt-on-Change

## 17.1 Overview

The pins denoted in the table below can be configured to operate as interrupt-on-change (IOC) pins for this device. 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.

**Table 17-1. IOC Pin Availability per Device**

| Device | PORTA | PORTB | PORTC | PORTD | PORTE |
|---|---|---|---|---|---|
| 28-pin devices | • | • | • | | • |
| 40/44-pin devices | • | • | • | | • |

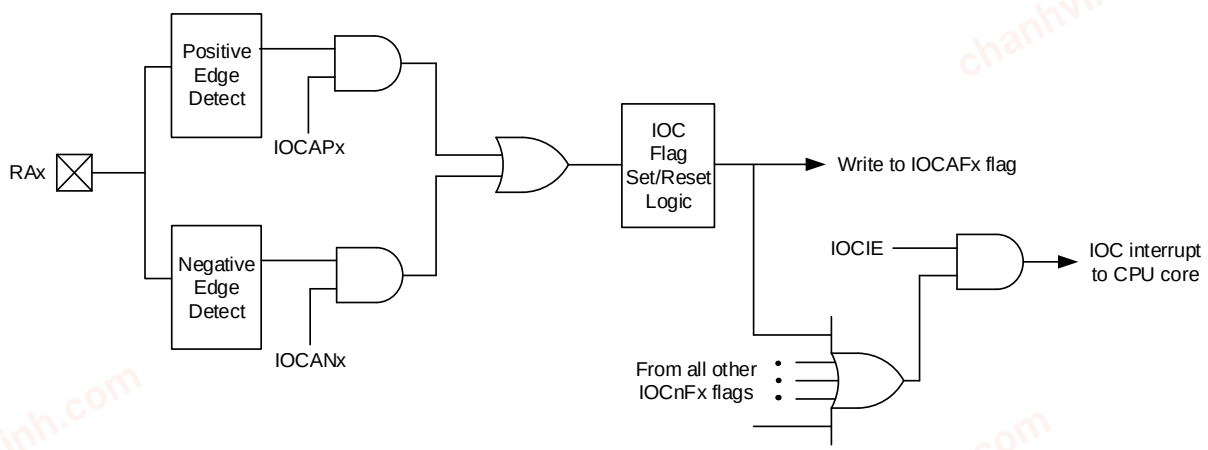
**Important:** If MCLRE = 1 or LVP = 1, the $\overline{\text{MCLR}}$ pin port functionality is disabled and IOC on that pin is not available.

The interrupt-on-change module has the following features:

- Interrupt-on-change enable (Host Switch)
- Individual pin configuration
- Rising and falling edge detection
- Individual pin interrupt flags

The following figure is a block diagram of the IOC module.

**Figure 17-1. Interrupt-on-Change Block Diagram (PORTA Example)**




## 17.2 Enabling the Module

For individual PORT pins to generate an interrupt, the IOC Interrupt Enable (IOCIE) bit of the Peripheral Interrupt Enable (PIEx) register must be set. If the IOC Interrupt Enable bit is disabled, the edge detection on the pin will still occur, but an interrupt will not be generated.

chanhvinh.com



## 17.3 Individual Pin Configuration

A rising edge detector and a falling edge detector are present for each PORT pin. To enable a pin to detect a rising edge, the associated bit of the IOCxP register must be set. To enable a pin to detect a falling edge, the associated bit of the IOCxN register must be set. A PORT pin can be configured to detect rising and falling edges simultaneously by setting both associated bits of the IOCxP and IOCxN registers, respectively.

## 17.4 Interrupt Flags

The bits located in the IOCxF registers are status flags that correspond to the interrupt-on-change pins of each 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 located in the corresponding Peripheral Interrupt Request (PIRx) register, is all the IOCxF bits ORd together. The IOCIF bit is read-only. All of the IOCxF Status bits must be cleared to clear the IOCIF bit.

## 17.5 Clearing Interrupt Flags

The individual status flags (IOCxF register bits) will 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.

To ensure that no detected edge is lost while clearing flags, only AND operations masking out known changed bits must be performed. The following sequence is an example of clearing an IOC interrupt flag using this method.

**Example 17-1. Clearing Interrupt Flags (PORTA Example)**

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

## 17.6 Operation in Sleep

An interrupt-on-change event 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.

## 17.7 Register Definitions: Interrupt-on-Change Control

chanhvinh.com



### 17.7.1 IOCxF

**Name:** IOCxF

Interrupt-on-Change Flag Register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | IOCxF7 | IOCxF6 | IOCxF5 | IOCxF4 | IOCxF3 | IOCxF2 | IOCxF1 | IOCxF0 |
| Access | R/W/HS | R/W/HS | R/W/HS | R/W/HS | R/W/HS | R/W/HS | R/W/HS | R/W/HS |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 0, 1, 2, 3, 4, 5, 6, 7 – IOCxFn** Interrupt-on-Change Flag

| Value | Condition | Description |
|---|---|---|
| 1 | IOCxP[n] = 1 | A positive edge was detected on the Rx[n] pin |
| 1 | IOCxN[n] = 1 | A negative edge was detected on the Rx[n] pin |
| 0 | IOCxP[n] = x and IOCxN[n] = x | No change was detected, or the user cleared the detected change |

**Important:**
- If MCLRE = 1 or LVP = 1, the $\overline{\text{MCLR}}$ pin port functionality is disabled and IOC on that pin is not available
- Refer to the **"Pin Allocation Table"** for details about pins with configurable IOC per port

chanhvinh.com



### 17.7.2 IOCxN

**Name:** IOCxN

Interrupt-on-Change Negative Edge Register Example

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | IOCxN7 | IOCxN6 | IOCxN5 | IOCxN4 | IOCxN3 | IOCxN2 | IOCxN1 | IOCxN0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 0, 1, 2, 3, 4, 5, 6, 7 – IOCxNn** Interrupt-on-Change Negative Edge Enable

| Value | Description |
|---|---|
| 1 | Interrupt-on-change enabled on the IOCx pin for a negative-going edge. Associated Status bit and interrupt flag will be set upon detecting an edge. |
| 0 | Falling edge interrupt-on-change disabled for the associated pin |

**Important:**

- If MCLRE = 1 or LVP = 1, the $\overline{\text{MCLR}}$ pin port functionality is disabled and IOC on that pin is not available
- Refer to the **"Pin Allocation Table"** for details about pins with configurable IOC per port

chanhvinh.com



### 17.7.3 IOCxP

**Name:** IOCxP

Interrupt-on-Change Positive Edge Register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | IOCxP7 | IOCxP6 | IOCxP5 | IOCxP4 | IOCxP3 | IOCxP2 | IOCxP1 | IOCxP0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 0, 1, 2, 3, 4, 5, 6, 7 – IOCxPn** Interrupt-on-Change Positive Edge Enable

| Value | Description |
|---|---|
| 1 | Interrupt-on-change enabled on the IOCx pin for a positive-going edge. Associated Status bit and interrupt flag will be set upon detecting an edge. |
| 0 | Rising edge interrupt-on-change disabled for the associated pin. |

**Important:**

- If MCLRE = 1 or LVP = 1, the $\overline{\text{MCLR}}$ pin port functionality is disabled and IOC on that pin is not available
- Refer to the **"Pin Allocation Table"** for details about pins with configurable IOC per port

chanhvinh.com



## 17.8 Register Summary - Interrupt-on-Change

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 ... 0x1E90 | Reserved | | | | | | | | | |
| 0x1E91 | IOCAP | 7:0 | IOCAP7 | IOCAP6 | IOCAP5 | IOCAP4 | IOCAP3 | IOCAP2 | IOCAP1 | IOCAP0 |
| 0x1E92 | IOCAN | 7:0 | IOCAN7 | IOCAN6 | IOCAN5 | IOCAN4 | IOCAN3 | IOCAN2 | IOCAN1 | IOCAN0 |
| 0x1E93 | IOCAF | 7:0 | IOCAF7 | IOCAF6 | IOCAF5 | IOCAF4 | IOCAF3 | IOCAF2 | IOCAF1 | IOCAF0 |
| 0x1E94 ... 0x1E9A | Reserved | | | | | | | | | |
| 0x1E9B | IOCBP | 7:0 | IOCBP7 | IOCBP6 | IOCBP5 | IOCBP4 | IOCBP3 | IOCBP2 | IOCBP1 | IOCBP0 |
| 0x1E9C | IOCBN | 7:0 | IOCBN7 | IOCBN6 | IOCBN5 | IOCBN4 | IOCBN3 | IOCBN2 | IOCBN1 | IOCBN0 |
| 0x1E9D | IOCBF | 7:0 | IOCBF7 | IOCBF6 | IOCBF5 | IOCBF4 | IOCBF3 | IOCBF2 | IOCBF1 | IOCBF0 |
| 0x1E9E ... 0x1EA4 | Reserved | | | | | | | | | |
| 0x1EA5 | IOCCP | 7:0 | IOCCP7 | IOCCP6 | IOCCP5 | IOCCP4 | IOCCP3 | IOCCP2 | IOCCP1 | IOCCP0 |
| 0x1EA6 | IOCCN | 7:0 | IOCCN7 | IOCCN6 | IOCCN5 | IOCCN4 | IOCCN3 | IOCCN2 | IOCCN1 | IOCCN0 |
| 0x1EA7 | IOCCF | 7:0 | IOCCF7 | IOCCF6 | IOCCF5 | IOCCF4 | IOCCF3 | IOCCF2 | IOCCF1 | IOCCF0 |
| 0x1EA8 ... 0x1EB8 | Reserved | | | | | | | | | |
| 0x1EB9 | IOCEP | 7:0 | | | | | IOCEP3 | | | |
| 0x1EBA | IOCEN | 7:0 | | | | | IOCEN3 | | | |
| 0x1EBB | IOCEF | 7:0 | | | | | IOCEF3 | | | |

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



# 18. PPS - Peripheral Pin Select Module

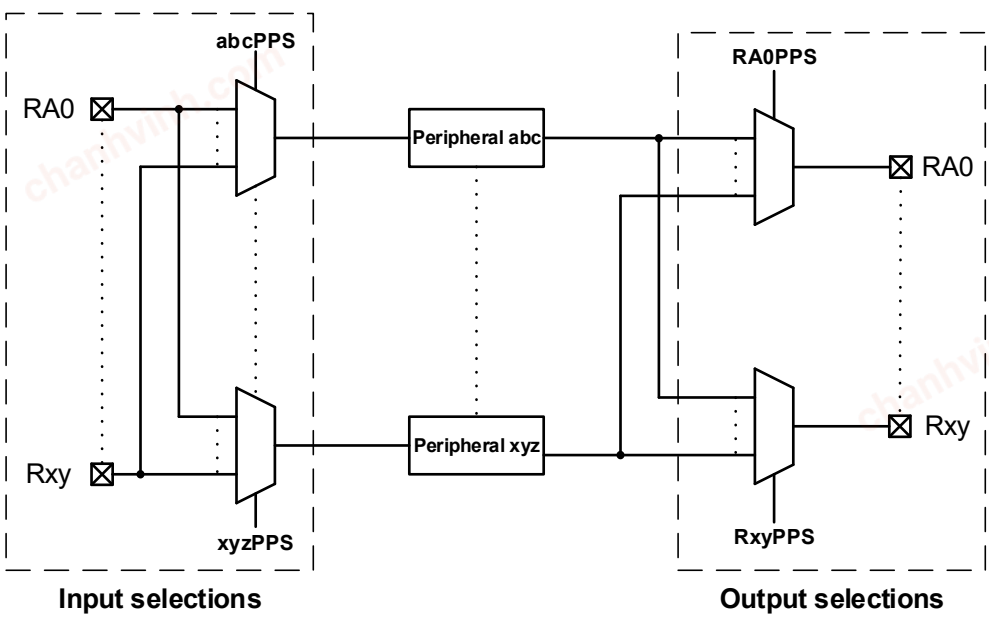
## 18.1 Overview

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.

**Important:** All analog inputs and outputs remain fixed to their assigned pins and cannot be changed through PPS.

Input and output selections are independent as shown in the figure below.

**Figure 18-1. PPS Block Diagram**




## 18.2 PPS Inputs

Each digital peripheral has a dedicated PPS Peripheral Input Selection (xxxPPS) register with which the input pin to the peripheral is selected. Devices that have 20 leads or less (8/14/16/20) allow PPS routing to any I/O pin, while devices with 28 leads or more allow PPS routing to I/Os contained within two ports (see the table below).

**Important:** The notation “xxx” in the generic register name is a placeholder for the peripheral identifier. For example, xxx = T0CKI for the T0CKIPPS register.

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 analog functions associated, the ANSEL bit for that pin must be cleared to enable the digital input buffer.

chanhvinh.com



**Table 18-1. PPS Input Selection Table**

| Peripheral | PPS Input Register | Default Pin Selection at POR | Register Reset Value at POR | Available Input Port | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | 28-Pin Devices | | | 40-Pin Devices | | | | |
| External Interrupt | INTPPS | RB0 | 'b001 000 | A | B | — | A | B | — | — | — |
| Timer0 Clock | T0CKIPPS | RA4 | 'b000 100 | A | B | — | A | B | — | — | — |
| Timer1 Clock | T1CKIPPS | RC0 | 'b010 000 | A | — | C | A | — | C | — | — |
| Timer1 Gate | T1GPPS | RB5 | 'b001 101 | — | B | C | — | B | C | — | — |
| Timer3 Clock | T3CKIPPS | RC0 | 'b010 000 | — | B | C | — | B | C | — | — |
| Timer3 Gate | T3GPPS | RC0 | 'b010 000 | A | — | C | A | — | C | — | — |
| Timer2 Input | T2INPPS | RC3 | 'b010 011 | A | — | C | A | — | C | — | — |
| Timer4 Input | T4INPPS | RC5 | 'b010 101 | — | B | C | — | B | C | — | — |
| Timer6 Input | T6INPPS | RB7 | 'b001 111 | — | B | C | — | B | — | D | — |
| CCP1 | CCP1PPS | RC2 | 'b010 010 | — | B | C | — | B | C | — | — |
| CCP2 | CCP2PPS | RC1 | 'b010 001 | — | B | C | — | B | C | — | — |
| PWM Input 0 | PWMIN0PPS | RC2 | 'b010 010 | — | B | C | — | B | C | — | — |
| PWM Input 1 | PWMIN1PPS | RC1 | 'b010 001 | — | B | C | — | B | C | — | — |
| PWM1 External Reset | PWMIN1ERSPPS | RC3 | 'b010 011 | A | — | C | A | — | C | — | — |
| PWM2 External Reset | PWMIN2ERSPPS | RC5 | 'b010 101 | — | B | C | — | B | C | — | — |
| PWM3 External Reset | PWMIN3ERSPPS | RB7 | 'b001 111 | — | B | C | — | B | C | — | — |
| PWM4 External Reset | PWMIN4ERSPPS | RC3 | 'b010 011 | A | — | C | A | — | C | — | — |
| CWG1 | CWG1PPS | RB0 | 'b001 000 | — | B | C | — | B | — | D | — |
| CLCIN0 | CLCIN0PPS | RA0 | 'b000 000 | A | — | C | A | — | C | — | — |
| CLCIN1 | CLCIN1PPS | RA1 | 'b000 001 | A | — | C | A | — | C | — | — |
| CLCIN2 | CLCIN2PPS | RB6 | 'b001 110 | — | B | C | — | B | — | D | — |
| CLCIN3 | CLCIN3PPS | RB7 | 'b001 111 | — | B | C | — | B | — | D | — |
| SCL1/SCK1 | SSP1CLKPPS(1) | RC3 | 'b010 011 | — | B | C | — | B | C | — | — |
| SDA1/SDI1 | SSP1DATPPS(1) | RC4 | 'b010 100 | — | B | C | — | B | C | — | — |
| $\overline{SS1}$ | SSP1SSPPS | RA5 | 'b000 101 | A | — | C | A | — | — | D | — |
| SCL2/SCK2 | SSP2CLKPPS(1) | RB1 | 'b001 001 | — | B | C | — | B | — | D | — |
| SDA2/SDI2 | SSP2DATPPS(1) | RB2 | 'b001 010 | — | B | C | — | B | — | D | — |
| $\overline{SS2}$ | SSP2SSPPS | RB0 | 'b001 000 | — | B | C | — | B | — | D | — |
| RX1/DT1 | RX1PPS | RC7 | 'b010 111 | — | B | C | — | B | C | — | — |
| CK1 | CK1PPS | RC6 | 'b010 110 | — | B | C | — | B | C | — | — |
| RX2/DT2 | RX2PPS | RB7 | 'b001 111 | — | B | C | — | B | — | D | — |
| CK2 | CK2PPS | RB6 | 'b001 110 | — | B | C | — | B | — | D | — |
| ADC Conversion Trigger | ADACTPPS | RB4 | 'b001 100 | — | B | C | — | B | — | D | — |
| OPA1 Input | OPA1PPS | RA0 | 'b000 000 | A | — | C | A | — | C | — | — |

**Note:**

1. Bidirectional pin. The corresponding output must select the same pin.

## 18.3 PPS Outputs

Each digital peripheral has a dedicated Pin Rxy Output Source Selection (RxyPPS) 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. The $I^2C$ module is an example of such a peripheral.

chanhvinh.com



**Important:** The notation 'Rxy' is a placeholder for the pin identifier. The 'x' holds the place of the PORT letter and the 'y' holds the place of the bit number. For example, Rxy = RA0 for the RA0PPS register.

The table below shows the output codes for each peripheral, as well as the available Port selections.

**Table 18-2. PPS Output Selection Table**

| RxyPPS | Output Source | Available Output Ports | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | 28-Pin Devices | | | 40-Pin Devices | | | | |
| 0x23 | ADGRDB | A | — | C | A | — | C | — | — |
| 0x22 | ADGRDA | A | — | C | A | — | C | — | — |
| 0x21 | CLKR | — | B | C | — | B | C | — | — |
| 0x20 | NCO1 | A | — | C | A | — | — | D | — |
| 0x1F | TMR0 | — | B | C | — | B | C | — | — |
| 0x1E | SDA2/SDO2(1) | — | B | C | — | B | — | D | — |
| 0x1D | SCL2/SCK2(1) | — | B | C | — | B | — | D | — |
| 0x1C | SDA1/SDO1(1) | — | B | C | — | B | C | — | — |
| 0x1B | SCL1/SCK1(1) | — | B | C | — | B | C | — | — |
| 0x1A | C2OUT | A | — | C | A | — | — | — | E |
| 0x19 | C1OUT | A | — | C | A | — | — | D | — |
| 0x18 | DT2 | — | B | C | — | B | — | D | — |
| 0x17 | Reserved | | | | | | | | |
| 0x16 | TX2/CK2 | — | B | C | — | B | — | D | — |
| 0x15 | DT1 | — | B | C | — | B | C | — | — |
| 0x14 | Reserved | | | | | | | | |
| 0x13 | TX1/CK1 | — | B | C | — | B | C | — | — |
| 0x12 | PWM4S1P2_OUT | — | B | C | — | B | C | — | — |
| 0x11 | PWM4S1P1_OUT | — | B | C | — | B | C | — | — |
| 0x10 | PWM3S1P2_OUT | — | B | C | — | B | — | D | — |
| 0x0F | PWM3S1P1_OUT | — | B | C | — | B | — | D | — |
| 0x0E | PWM2S1P2_OUT | — | B | — | — | B | — | D | — |
| 0x0D | PWM2S1P1_OUT | — | B | — | — | B | — | D | — |
| 0x0C | PWM1S1P2_OUT | — | B | C | — | B | C | — | — |
| 0x0B | PWM1S1P1_OUT | — | B | C | — | B | C | — | — |
| 0x0A | CCP2 | — | B | C | — | B | C | — | — |
| 0x09 | CCP1 | — | B | C | — | B | C | — | — |
| 0x08 | CWG1D | — | B | C | — | B | — | D | — |
| 0x07 | CWG1C | — | B | C | — | B | — | D | — |
| 0x06 | CWG1B | — | B | C | — | B | — | D | — |
| 0x05 | CWG1A | — | B | C | — | B | C | — | — |
| 0x04 | CLC4OUT | — | B | C | — | B | — | D | — |
| 0x03 | CLC3OUT | — | B | C | — | B | — | D | — |
| 0x02 | CLC2OUT | A | — | C | A | — | C | — | — |
| 0x01 | CLC1OUT | A | — | C | A | — | C | — | — |
| 0x00 | LATxy | A | B | C | A | B | C | D | E |

**Note:**

1. Bidirectional pin. The corresponding input must select the same pin.

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



## 18.4 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. The I²C Serial Clock (SCL) and Serial Data (SDA) are examples of such pins.

**Important:** The I²C default pins and a limited number of other alternate pins are I²C and SMBus compatible. SDA and SCL signals can be routed to any pin; however, pins without I²C compatibility will operate at standard TTL/ST logic levels as selected by the port's INLVL register.

## 18.5 PPS Lock

The PPS module provides an extra layer of protection to prevent inadvertent changes to the PPS selection registers. The PPSLOCKED bit is used in combination with specific code execution blocks to lock/unlock the PPS selection registers.

**Important:** The PPSLOCKED bit is clear by default (PPSLOCKED = 0), which allows the PPS selection registers to be modified without an unlock sequence.

PPS selection registers are locked when the PPSLOCKED bit is set (PPSLOCKED = 1). Setting the PPSLOCKED bit requires a specific lock sequence as shown in the examples below in both C and assembly languages.

PPS selection registers are unlocked when the PPSLOCKED bit is clear (PPSLOCKED = 0). Clearing the PPSLOCKED bit requires a specific unlock sequence as shown in the examples below in both C and assembly languages.

**Important:** All interrupts must be disabled before starting the lock/unlock sequence to ensure proper execution.

**Example 18-1. PPS Lock Sequence (assembly language)**

```
; suspend interrupts
   BCF      INTCON0,GIE
   BANKSEL  PPSLOCK
; required sequence, next 5 instructions
   MOVLW    0x55
   MOVWF    PPSLOCK
   MOVLW    0xAA
   MOVWF    PPSLOCK
; Set PPSLOCKED bit
   BSF      PPSLOCK,PPSLOCKED
; restore interrupts
   BSF      INTCON0,GIE
```

**Example 18-2. PPS Lock Sequence (C language)**

```
INTCON0bits.GIE = 0;         //Suspend interrupts
PPSLOCK = 0x55;              //Required sequence
PPSLOCK = 0xAA;              //Required sequence
PPSLOCKbits.PPSLOCKED = 1;   //Set PPSLOCKED bit
INTCON0bits.GIE = 1;         //Restore interrupts
```

chanhvinh.com



**Example 18-3. PPS Unlock Sequence (assembly language)**

```
; suspend interrupts
   BCF      INTCON0,GIE
   BANKSEL  PPSLOCK
; required sequence, next 5 instructions
   MOVLW    0x55
   MOVWF    PPSLOCK
   MOVLW    0xAA
   MOVWF    PPSLOCK
; Clear PPSLOCKED bit
   BCF      PPSLOCK,PPSLOCKED
; restore interrupts
   BSF      INTCON0,GIE
```

**Example 18-4. PPS Unlock Sequence (C language)**

```
INTCON0bits.GIE = 0;         //Suspend interrupts
PPSLOCK = 0x55;              //Required sequence
PPSLOCK = 0xAA;              //Required sequence
PPSLOCKbits.PPSLOCKED = 0;   //Clear PPSLOCKED bit
INTCON0bits.GIE = 1;         //Restore interrupts
```

### 18.5.1 PPS One-Way Lock

The PPS1WAY Configuration bit can also be used to prevent inadvertent modification to the PPS selection registers.

When the PPS1WAY bit is set (PPS1WAY = 1), the PPSLOCKED bit can only be set one time after a device Reset. Once the PPSLOCKED bit has been set, it cannot be cleared again unless a device Reset is executed.

When the PPS1WAY bit is clear (PPS1WAY = 0), the PPSLOCKED bit can be set or cleared as needed; however, the PPS lock/unlock sequences must be executed.

## 18.6 Operation During Sleep

PPS input and output selections are unaffected by Sleep.

## 18.7 Effects of a Reset

A device Power-on Reset (POR) or Brown-out Reset (BOR) returns all PPS input selection registers to their default values and clears all PPS output selection registers. All other Resets leave the selections unchanged. Default input selections are shown in the PPS input register details table. The PPSLOCKED bit is cleared in all Reset conditions.

## 18.8 Register Definitions: Peripheral Pin Select (PPS)

chanhvinh.com



### 18.8.1 xxxPPS

**Name:** xxxPPS

Peripheral Input Selection Register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | PORT[2:0] | | | PIN[2:0] | | |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | m | m | m | m | m | m |

**Bits 5:3 – PORT[2:0]** Peripheral Input PORT Selection(1)
See the PPS Input Selection Table for the list of available Ports and default pin locations.
Reset States: POR = mmm
All other Resets = uuu

| Value | Description |
|---|---|
| 100 | PORTE |
| 011 | PORTD |
| 010 | PORTC |
| 001 | PORTB |
| 000 | PORTA |

**Bits 2:0 – PIN[2:0]** Peripheral Input PORT Pin Selection(2)
Reset States: POR = mmm
All other Resets = uuu

| Value | Description |
|---|---|
| 111 | Peripheral input is from PORTx Pin 7 (Rx7) |
| 110 | Peripheral input is from PORTx Pin 6 (Rx6) |
| 101 | Peripheral input is from PORTx Pin 5 (Rx5) |
| 100 | Peripheral input is from PORTx Pin 4 (Rx4) |
| 011 | Peripheral input is from PORTx Pin 3 (Rx3) |
| 010 | Peripheral input is from PORTx Pin 2 (Rx2) |
| 001 | Peripheral input is from PORTx Pin 1 (Rx1) |
| 000 | Peripheral input is from PORTx Pin 0 (Rx0) |

**Notes:**

1. The Reset value ‘m’ is determined by device default locations for that input.
2. Refer to the **“Pin Allocation Table”** for details about available pins per port.

chanhvinh.com



### 18.8.2 RxyPPS

**Name:** RxyPPS

Pin Rxy Output Source Selection Register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | RxyPPS[5:0] | | | | | |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 5:0 – RxyPPS[5:0]** Pin Rxy Output Source Selection
See the PPS Output Selection Table for the list of RxyPPS Output Source codes
Reset States: POR = 000000
All other Resets = uuuuuu

chanhvinh.com



### 18.8.3 PPSLOCK

**Name:** PPSLOCK

PPS Lock Register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | PPSLOCKED |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

**Bit 0 – PPSLOCKED** PPS Locked
Reset States: POR = 0
All other Resets = 0

| Value | Description |
|---|---|
| 1 | PPS is locked. PPS selections cannot be changed. Writes to any PPS register are ignored. |
| 0 | PPS is not locked. PPS selections can be changed, but may require the PPS lock/unlock sequence. |

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



## 18.9 Register Summary - Peripheral Pin Select Module

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 ... 0x1D8B | Reserved | | | | | | | | | |
| 0x1D8C | RA0PPS | 7:0 | | | RA0PPS[5:0] | | | | | |
| 0x1D8D | RA1PPS | 7:0 | | | RA1PPS[5:0] | | | | | |
| 0x1D8E | RA2PPS | 7:0 | | | RA2PPS[5:0] | | | | | |
| 0x1D8F | RA3PPS | 7:0 | | | RA3PPS[5:0] | | | | | |
| 0x1D90 | RA4PPS | 7:0 | | | RA4PPS[5:0] | | | | | |
| 0x1D91 | RA5PPS | 7:0 | | | RA5PPS[5:0] | | | | | |
| 0x1D92 | RA6PPS | 7:0 | | | RA6PPS[5:0] | | | | | |
| 0x1D93 | RA7PPS | 7:0 | | | RA7PPS[5:0] | | | | | |
| 0x1D94 | RB0PPS | 7:0 | | | RB0PPS[5:0] | | | | | |
| 0x1D95 | RB1PPS | 7:0 | | | RB1PPS[5:0] | | | | | |
| 0x1D96 | RB2PPS | 7:0 | | | RB2PPS[5:0] | | | | | |
| 0x1D97 | RB3PPS | 7:0 | | | RB3PPS[5:0] | | | | | |
| 0x1D98 | RB4PPS | 7:0 | | | RB4PPS[5:0] | | | | | |
| 0x1D99 | RB5PPS | 7:0 | | | RB5PPS[5:0] | | | | | |
| 0x1D9A | RB6PPS | 7:0 | | | RB6PPS[5:0] | | | | | |
| 0x1D9B | RB7PPS | 7:0 | | | RB7PPS[5:0] | | | | | |
| 0x1D9C | RC0PPS | 7:0 | | | RC0PPS[5:0] | | | | | |
| 0x1D9D | RC1PPS | 7:0 | | | RC1PPS[5:0] | | | | | |
| 0x1D9E | RC2PPS | 7:0 | | | RC2PPS[5:0] | | | | | |
| 0x1D9F | RC3PPS | 7:0 | | | RC3PPS[5:0] | | | | | |
| 0x1DA0 | RC4PPS | 7:0 | | | RC4PPS[5:0] | | | | | |
| 0x1DA1 | RC5PPS | 7:0 | | | RC5PPS[5:0] | | | | | |
| 0x1DA2 | RC6PPS | 7:0 | | | RC6PPS[5:0] | | | | | |
| 0x1DA3 | RC7PPS | 7:0 | | | RC7PPS[5:0] | | | | | |
| 0x1DA4 | RD0PPS | 7:0 | | | RD0PPS[5:0] | | | | | |
| 0x1DA5 | RD1PPS | 7:0 | | | RD1PPS[5:0] | | | | | |
| 0x1DA6 | RD2PPS | 7:0 | | | RD2PPS[5:0] | | | | | |
| 0x1DA7 | RD3PPS | 7:0 | | | RD3PPS[5:0] | | | | | |
| 0x1DA8 | RD4PPS | 7:0 | | | RD4PPS[5:0] | | | | | |
| 0x1DA9 | RD5PPS | 7:0 | | | RD5PPS[5:0] | | | | | |
| 0x1DAA | RD6PPS | 7:0 | | | RD6PPS[5:0] | | | | | |
| 0x1DAB | RD7PPS | 7:0 | | | RD7PPS[5:0] | | | | | |
| 0x1DAC | RE0PPS | 7:0 | | | RE0PPS[5:0] | | | | | |
| 0x1DAD | RE1PPS | 7:0 | | | RE1PPS[5:0] | | | | | |
| 0x1DAE | RE2PPS | 7:0 | | | RE2PPS[5:0] | | | | | |
| 0x1DAF ... 0x1E0B | Reserved | | | | | | | | | |
| 0x1E0C | PPSLOCK | 7:0 | | | | | | | | PPSLOCKED |
| 0x1E0D | INTPPS | 7:0 | | | PORT[2:0] | | | PIN[2:0] | | |
| 0x1E0E | T0CKIPPS | 7:0 | | | PORT[2:0] | | | PIN[2:0] | | |
| 0x1E0F | T1CKIPPS | 7:0 | | | PORT[2:0] | | | PIN[2:0] | | |
| 0x1E10 | T1GPPS | 7:0 | | | PORT[2:0] | | | PIN[2:0] | | |
| 0x1E11 | T3CKIPPS | 7:0 | | | PORT[2:0] | | | PIN[2:0] | | |
| 0x1E12 | T3GPPS | 7:0 | | | PORT[2:0] | | | PIN[2:0] | | |
| 0x1E13 ... 0x1E18 | Reserved | | | | | | | | | |
| 0x1E19 | T2INPPS | 7:0 | | | PORT[2:0] | | | PIN[2:0] | | |
| 0x1E1A | T4INPPS | 7:0 | | | PORT[2:0] | | | PIN[2:0] | | |
| 0x1E1B | T6INPPS | 7:0 | | | PORT[2:0] | | | PIN[2:0] | | |

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x1E1C ... 0x1E1D | Reserved | | | | | | | | | |
| 0x1E1E | CCP1PPS | 7:0 | | | PORT[2:0] | | | PIN[2:0] | | |
| 0x1E1F | CCP2PPS | 7:0 | | | PORT[2:0] | | | PIN[2:0] | | |
| 0x1E20 ... 0x1E23 | Reserved | | | | | | | | | |
| 0x1E24 | PWMIN0PPS | 7:0 | | | PORT[2:0] | | | PIN[2:0] | | |
| 0x1E25 | PWMIN1PPS | 7:0 | | | PORT[2:0] | | | PIN[2:0] | | |
| 0x1E26 | PWM1ERSPPS | 7:0 | | | PORT[2:0] | | | PIN[2:0] | | |
| 0x1E27 | PWM2ERSPPS | 7:0 | | | PORT[2:0] | | | PIN[2:0] | | |
| 0x1E28 | PWM3ERSPPS | 7:0 | | | PORT[2:0] | | | PIN[2:0] | | |
| 0x1E29 | PWM4ERSPPS | 7:0 | | | PORT[2:0] | | | PIN[2:0] | | |
| 0x1E2A ... 0x1E38 | Reserved | | | | | | | | | |
| 0x1E39 | CWG1PPS | 7:0 | | | PORT[2:0] | | | PIN[2:0] | | |
| 0x1E3A ... 0x1E3C | Reserved | | | | | | | | | |
| 0x1E3D | CLCIN0PPS | 7:0 | | | PORT[2:0] | | | PIN[2:0] | | |
| 0x1E3E | CLCIN1PPS | 7:0 | | | PORT[2:0] | | | PIN[2:0] | | |
| 0x1E3F | CLCIN2PPS | 7:0 | | | PORT[2:0] | | | PIN[2:0] | | |
| 0x1E40 | CLCIN3PPS | 7:0 | | | PORT[2:0] | | | PIN[2:0] | | |
| 0x1E41 | RX1PPS | 7:0 | | | PORT[2:0] | | | PIN[2:0] | | |
| 0x1E42 | CK1PPS | 7:0 | | | PORT[2:0] | | | PIN[2:0] | | |
| 0x1E43 | RX2PPS | 7:0 | | | PORT[2:0] | | | PIN[2:0] | | |
| 0x1E44 | CK2PPS | 7:0 | | | PORT[2:0] | | | PIN[2:0] | | |
| 0x1E45 ... 0x1E46 | Reserved | | | | | | | | | |
| 0x1E47 | SSP1CLKPPS | 7:0 | | | PORT[2:0] | | | PIN[2:0] | | |
| 0x1E48 | SSP1DATPPS | 7:0 | | | PORT[2:0] | | | PIN[2:0] | | |
| 0x1E49 | SSP1SSPPS | 7:0 | | | PORT[2:0] | | | PIN[2:0] | | |
| 0x1E4A | SSP2CLKPPS | 7:0 | | | PORT[2:0] | | | PIN[2:0] | | |
| 0x1E4B | SSP2DATPPS | 7:0 | | | PORT[2:0] | | | PIN[2:0] | | |
| 0x1E4C | SSP2SSPPS | 7:0 | | | PORT[2:0] | | | PIN[2:0] | | |
| 0x1E4D ... 0x1E4F | Reserved | | | | | | | | | |
| 0x1E50 | ADACTPPS | 7:0 | | | PORT[2:0] | | | PIN[2:0] | | |
| 0x1E51 ... 0x1E52 | Reserved | | | | | | | | | |
| 0x1E53 | OPA1PPS | 7:0 | | | PORT[2:0] | | | PIN[2:0] | | |

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



# 19. CRC - Cyclic Redundancy Check Module with Memory Scanner

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 32 bits can be used
- Configurable polynomial
- Any seed value up to 32 bits can be used
- Standard and reversed bit order available
- Augmented zeros can be added automatically or by the user
- Memory scanner for core-independent CRC calculations on any program memory locations
- Software configurable data registers for communication CRCs

## 19.1 Module Overview

The CRC module is coupled with a memory scanner that provides a means of performing CRC calculations in hardware, without CPU intervention. The memory scanner can automatically provide data from program Flash memory to the CRC module. The CRC module can also be operated by directly writing data to SFRs, without using a scanner.

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 32-bit polynomial with up to a 32-bit seed value. A CRC calculated check value (or checksum) will then be generated into the CRCOUT registers for user storage. The CRC module uses an XOR shift register implementation to perform the polynomial division required for the CRC calculation. This feature is useful for calculating CRC values of data being transmitted or received using communications peripherals such as the SPI, UART or $I^2C$.

## 19.2 Polynomial Implementation

The CRC polynomial equation is user configurable, allowing any polynomial equation to be used for the CRC checksum calculation. The polynomial and accumulator sizes are determined by the PLEN bits. For an n-bit accumulator, PLEN = n-1 and the corresponding polynomial is n+1 bits. This allows the accumulator to be any size up to 32 bits with a corresponding polynomial up to 33 bits. The MSb and LSb of the polynomial are always '1' which is forced by hardware. Therefore, the LSb of the CRCXOR Low Byte register is hardwired high and always reads as '1'.

All polynomial bits between the MSb and LSb are specified by the CRCXOR registers.
For example, when using the standard CRC32, the polynomial is defined as `0x4C11DB7` $\left(x^{32}+x^{26}+x^{23}+x^{22}+x^{16}+x^{12}+x^{11}+x^{10}+x^{8}+x^{7}+x^{5}+x^{4}+x^{2}+x+1\right)$. In this polynomial, the $X^{32}$ and $X^{0}$ terms are the MSb and LSb controlled by hardware. The $X^{31}$ and $X^{1}$ terms are specified by setting the CRCXOR[31:0] bits with the corresponding polynomial value, which in this example is `0x04C11DB6`. Reading the CRCXOR registers will return `0x04C11DB7` because the LSb is always '1'. Refer to the following example for more details.

**Example 19-1. CRC32 Example**

**Standard CRC32 Polynomial (33 bits):**

$\left(x^{32}+x^{26}+x^{23}+x^{22}+x^{16}+x^{12}+x^{11}+x^{10}+x^{8}+x^{7}+x^{5}+x^{4}+x^{2}+x+1\right)$

**Standard 32-bit Polynomial Representation:** `0x04C11DB7`

```
CRCXORT = 0x04 = 0b00000100
CRCXORU = 0xC1 = 0b11000001
CRCXORH = 0x1D = 0b00011101
CRCXORL = 0xB7 = 0b1011011-  (1)
```

chanhvinh.com



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

```
DLEN = 0b00111 // Number of bits written to CRCDATA registers (Data Length)
PLEN = 0b11111 // MSb position of the polynomial (Polynomial Length)
```

**Data Passed into the CRC:**

```
// SHIFTM = 0(Shift Mode: MSb first)
0x55 0x66 0x77 0x88 = 01010101 01100110 01110111 10001000

// SHIFTM = 1(Shift Mode: LSb first)
0x55 0x66 0x77 0x88 = 10101010 01100110 11101110 00010001
```

**CRC Check Value (ACCM = 1, data is augmented with zeros)**

```
// When SHIFTM = 0, CRC Result = 0xC60D8323
CRCOUTT = 0xC6 = 0b11000110
CRCOUTU = 0x0D = 0b00001101
CRCOUTH = 0x83 = 0b10000011
CRCOUTL = 0x23 = 0b00100011
// When SHIFTM = 1, CRC Result = 0x843529CC
CRCOUTT = 0x84 = 0b10000100
CRCOUTU = 0x35 = 0b00110101
CRCOUTH = 0x29 = 0b00101001
CRCOUTL = 0xCC = 0b11001100
```

**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'.

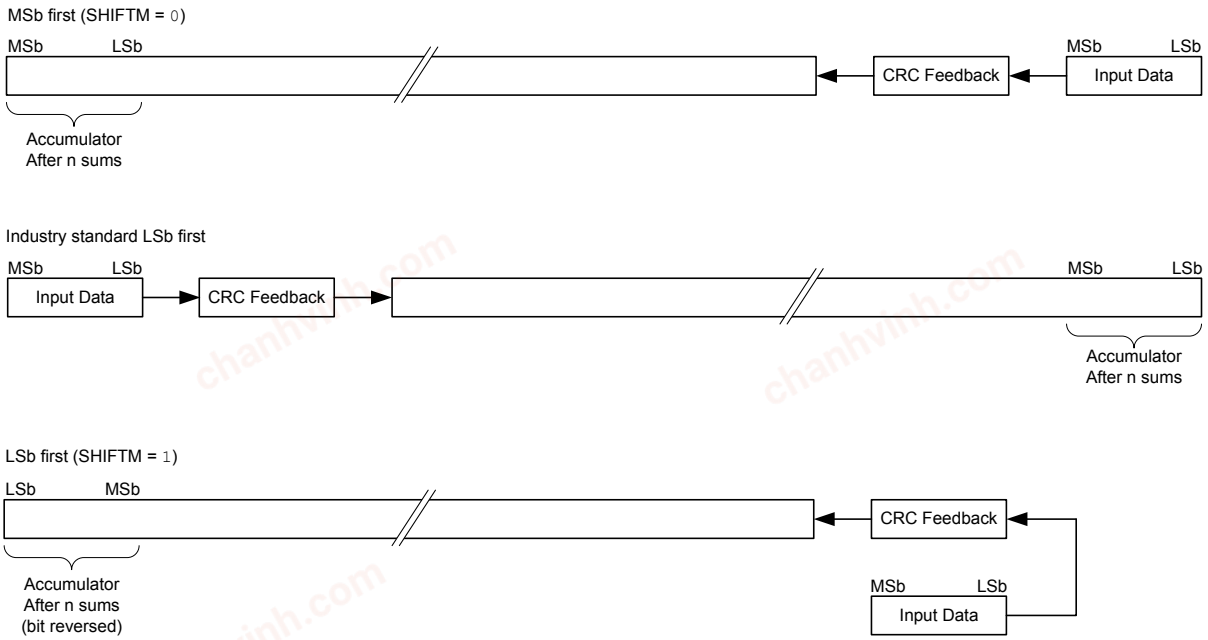
## 19.3 Data Sources

Data is supplied to the CRC module using the CRCDATA registers and can either be loaded manually or automatically by using the scanner module. The length of the data word being supplied to the CRC module is specified by the DLEN bits and can be configured for data words up to 32 bits in length. The DLEN field indicates how many bits in the CRCDATA registers are valid and any bits outside of the specified data word size will be ignored. Data is moved into the CRCSHIFT registers as an intermediate to calculate the check value located in the CRCOUT registers. The SHIFTM bit is used to determine the bit order of the data being shifted into the accumulator and the bit order of the result.

chanhvinh.com



**Figure 19-1. CRC Process**




When the SHIFTM bit is not set, data will be shifted into the CRC, MSb first and the result will be big-endian. When the SHIFTM bit is set, data will be shifted into the accumulator in the reversed order (LSb first) and the result will be little-endian. The CRC module can be seeded with an initial value by setting the CRCOUT registers to the appropriate value before beginning the CRC process.

### 19.3.1 CRC from User Data

Data can be supplied to the CRC module by writing to the CRCDATA registers. Once data has been loaded into the CRCDATA registers, it will then be latched onto the CRC Shift (CRCSHIFT) registers. If data is still being shifted from an earlier write to the CRCDATA registers and the user attempts to write more data, the most recently written data will be held in the CRCDATA registers until the previous shift has completed.

### 19.3.2 CRC from Flash

Data can also be supplied to the CRC module using the memory scanner, as opposed to writing the data manually using the CRCDATA registers, allowing users to automate CRC calculations. An automated scan of Program Flash Memory or Data EEPROM can be performed by configuring the scanner accordingly to copy data into the CRCDATA registers. The user can initialize the program memory scanner as defined in the **"Scanner Module Overview"** and **"Configuring the Scanner"** sections.

## 19.4 CRC Check Value

The CRC check value can be accessed using the CRCOUT registers after a CRC calculation has completed. The check value is dependent on the configuration of the ACCM and SHIFTM mode settings. When the ACCM bit is set, the CRC module will augment 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 and no additional zeroes will be augmented to the final value. The user can manually augment a number of additional zeroes equal to the length of the polynomial by entering them into the CRCDATA register, which will yield the same check value as Augmented mode. Alternatively, the expected check value can be entered at this point to make the final result equal zero.

When the CRC check value is computed with the SHIFTM (LSb first) and ACCM bits set, the final value in the CRCOUT registers will be reversed such that the LSb will be in the MSb position and vice versa (Figure 19-1).

When creating a check value to be appended to a data stream, then a reversal must be performed on the final value to achieve the correct checksum. The CRC can be used to do this reversal by following the steps below.

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



1. Save CRCOUT value in user RAM space.
2. Clear the CRCOUT registers.
3. Clear the CRCXOR registers.
4. Write the saved CRCOUT value to the CRCDATA input.

If the steps listed above were followed completely, the properly orientated check value will be in the CRCOUT registers as the result.

## 19.5 CRC Interrupt

The CRC module will generate an interrupt when the BUSY bit transitions from '1' to '0'. The CRC Interrupt Flag (CRCIF) bit of the corresponding PIR register will be set every time the BUSY bit transitions, regardless of whether or not the CRC Interrupt Enable (CRCIE) has been set. The CRCIF bit must be cleared by software by the user. If the user has the CRCIE bit set, then the CPU will jump to the Interrupt Service Routine (ISR) every time that the CRCIF bit is set.

## 19.6 Configuring the CRC Module

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 the CRC Data Sources section.
   a. To configure the scanner module to be used with CRC, refer to the **"Configuring the Scanner"** section for more information.
2. When applicable, seed a starting CRC value into the CRCOUT registers.
3. Program the CRCXOR registers with the desired generator polynomial.
4. Program the DLEN bits with the length of the data word (refer to Figure 19-1). This value determines how many times the shifter will shift into the accumulator for each data word.
5. Program the PLEN bits with the length of the polynomial (refer to Figure 19-1).
6. Determine whether shifting in trailing zeroes is desired and set the ACCM bit accordingly.
7. Determine whether the MSb or LSb first shifting is desired, and write the SHIFTM bit accordingly.
8. Set the GO bit to begin the shifting process.
9. If manual SFR entry is used, monitor the FULL bit.
   a. When FULL = 0, another word of data can be written to the CRCDATA registers. It is important to note that the Most Significant Byte (CRCDATAH) must be written first if the data has more than eight bits, as the shifter will begin upon the CRCDATAL register being written.
   b. If the scanner is used, the scanner will automatically load words into the CRCDATA registers as needed, as long as the GO bit is set.
10. If using the Flash memory scanner, monitor the SCANIF bit of the corresponding PIR register to determine when the scanner has finished pushing data into the CRCDATA registers.
    a. After the scan is completed, monitor the SGO bit to determine that the CRC has been completed and the check value can be read from the CRCOUT registers.
    b. When both the interrupt flags are set (or both BUSY and SGO bits are cleared), the completed CRC calculation can be read from the CRCOUT registers.
11. If manual entry is used, monitor the BUSY bit to determine when the CRCOUT registers hold the valid check value.

### 19.6.1 Register Overlay

The CRCOUT, CRCSHIFT and CRCXOR registers are grouped together and share SFR space. Since these register groups are located within the same addresses, the SETUP bits must be configured accordingly, to access any of these registers. Refer to the CRCCON2 register for more information about how the SETUP bits can be configured to access each of the available CRC registers.

chanhvinh.com



## 19.7 Scanner Module Overview

The scanner allows segments of the Program Flash Memory or Data EEPROM to be read out (scanned) to the CRC peripheral. The scanner module interacts with the CRC module and supplies it data, one word at a time. Data is fetched from the address range defined by SCANLADR registers up to the SCANHADR registers. The scanner begins operation when the SGO bit is set and ends when either SGO is cleared by the user or when SCANLADR increments past SCANHADR. The SGO bit is also cleared when the EN bit in the CRCCON0 register is cleared.

**Important:** When using the scanner module in conjunction with the CRC module, the CRCEN and CRCGO bits must be set before setting the SGO bit.

## 19.8 Scanning Modes

The interaction of the scanner with system operation is determined by the Scanner Memory Access Mode Select (MD) bits. The MD bits also determine the operation of the scanner.

The following modes are determined by the MD bits:

- Concurrent mode
- Burst mode
- Peek mode
- Triggered mode

### 19.8.1 Concurrent Mode

In Concurrent mode, the scanner will access memory on the instruction cycle following the instruction that sets SGO. During memory access, the CPU is stalled. Once the memory location has been accessed, the CPU resumes normal operation until the scanner is ready to access the next location.

### 19.8.2 Burst Mode

Burst mode provides the highest scan throughput of all modes. CPU operation is stalled on the instruction following the instruction that sets SGO, and remains stalled until the last NVM location has been transferred. Once hardware clears SGO, the CPU resumes normal operation.

**Important:** In Burst mode, software cannot clear SGO while the scan is in progress since the CPU cannot execute instructions while stalled.

### 19.8.3 Peek Mode

Peek mode provides the least impact on CPU operation. In Peek mode, the scanner waits until the CPU does not need to access the NVM and 'steals' the instruction cycle to perform a scan. Program branching (`GOTO`, `BRA`, `BRW`), subroutine entry (`CALL`, `CALLW`), or returns from interrupts or subroutines (`RETURN`, `RETFIE`, `RETLW`) operations take two instruction cycles but do not access NVM, so the scanner may use one of those instruction cycles for scanning operations.

While Peek mode may have the least impact on CPU operation, the time required to complete the scan will vary depending on which instructions the CPU is executing.

### 19.8.4 Triggered Mode

Triggered mode uses a clock or timer source to set a minimum time interval between successive data transfers. The trigger source is selected through the Scanner Data Trigger Input Selection (TSEL) bits. Triggered mode operation is the same as in Concurrent mode, except that the scanner will wait until it detects a rising-edge trigger event before performing a scan operation. The CPU operations are stalled while the scan is in progress and resume immediately

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



after the scan completes. If the Scanner Busy Indicator (BUSY) bit is set and a trigger event occurs, the event will be ignored.

**Note:** The trigger source must be active for scanner operation to complete.

## 19.9 Configuring the Scanner

The scanner module may be used in conjunction with the CRC module to perform a CRC calculation over a range of program memory or Data EEPROM addresses. To set up the scanner to work with the CRC, perform the following steps:

1. Set up the CRC module (see the **"Configuring the CRC Module"** section) and enable the scanner module by setting the EN bit in the SCANCON0 register.
2. Choose the scanning mode by configuring the MD bits.
3. If Trigger mode is selected for scanner operation, select the trigger source using the TSEL bits.
4. Set the SCANLADR and SCANHADR registers with the beginning and ending locations in memory that are to be scanned.
5. Both EN and GO bits in the CRCCON0 register must be enabled to use the scanner. Setting the SGO bit will start the scanner operation.

## 19.10 Scanner Interrupts

Scanner hardware will generate an interrupt when any of the following events occur:

- The last memory location (as determined by the SCANHADR registers) has been read and the data loaded into the CRCDATA registers
- The SCANLADR register pair points to an invalid address location
- The CRCGO and/or CRCEN bits are cleared

When the last memory location has been scanned and the data has been entered into the CRCDATA registers, scanner hardware clears SGO and sets the Scanner Interrupt Flag (SCANIF) bit in the PIR registers.

If the SCANLADR registers point to an invalid address location, or the CRCGO and/or CRCEN bits are cleared, scanner hardware clears SGO, sets SCANIF, and sets the Scan Abort Signal (DABORT) bit.

The SCANIF bit can only be cleared by software. If the Scanner Interrupt Enable (SCANIE) bit is set and hardware sets SCANIF, an interrupt event will occur.

### 19.10.1 Operation During Interrupts

The Scanner Interrupt Management Mode Select (INTM) bit determines scanner priority during an interrupt event.

When INTM = 0 and the MD bits select Burst mode, scanner operation is given higher priority than the interrupt response. This allows the scanner to delay the interrupt response until the scan operation is complete, but at the expense of increased interrupt response time.

When INTM = 0 and the MD bits select Concurrent or Triggered modes, the scanner accesses memory during the interrupt response. Scanner hardware will stall the CPU while the scan operation is in progress. Once the scan is complete, CPU operation resumes, allowing the interrupt response to resume. Interrupt response time is increased by one instruction cycle each time the scanner accesses memory during the interrupt response.

When INTM = 1, the interrupt response is given higher priority than the scanner operation. Scanner operation is delayed until the interrupt response is complete. This allows interrupts to be serviced immediately, but decreases scanner throughput time.

The INTM bit has no effect on scanner or interrupt priority in Peek mode.

## 19.11 WWDT Interaction

The Windowed Watch Dog Timer (WWDT) operates in the background during scanner activity. It is possible that long scans, particularly in Burst mode, may exceed the WWDT time-out period and result in an undesired device Reset. This must be considered when performing memory scans with an application that also utilizes WWDT.

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



## 19.12 Operation During Sleep

If a SLEEP instruction is executed during operation, the module will be suspended in its current state until the device wakes from Sleep.

## 19.13 Peripheral Module Disable

Both the CRC and scanner module can be disabled individually by setting the CRCMD and SCANMD bits of one of the PMD registers (see the **"Peripheral Module Disable"** chapter for more details). The SCANMD bit can be used to enable or disable the scanner module only if the SCANE Configuration bit is set. If the SCANE bit is cleared, then the scanner module is not available for use and the SCANMD bit is ignored.

## 19.14 Register Definitions: CRC and Scanner Control

Long bit name prefixes for the CRC are shown in the table below. Refer to the "**Long Bit Names**" section in the "**Register and Bit Naming Conventions**" chapter for more information.

**Table 19-1. CRC Long Bit Name Prefixes**

| Peripheral | Bit Name Prefix |
|---|---|
| CRC | CRC |

chanhvinh.com



### 19.14.1 CRCCON0

**Name:** CRCCON0
**Offset:** 0x1CA2

CRC Control Register 0

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | EN | GO | BUSY | ACCM | SETUP[1:0] | | SHIFTM | FULL |
| Access | R/W | R/W | R | R/W | R/W | | R/W | R |
| Reset | 0 | 0 | 0 | 0 | 0 | | 0 | 0 |

**Bit 7 – EN** CRC Enable

| Value | Description |
|---|---|
| 1 | CRC module is released from Reset |
| 0 | CRC is disabled and consumes no operating current |

**Bit 6 – GO** CRC Start

| Value | Description |
|---|---|
| 1 | Start CRC serial shifter |
| 0 | CRC serial shifter turned off |

**Bit 5 – BUSY** CRC Busy

| Value | Description |
|---|---|
| 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

| Value | Description |
|---|---|
| 1 | Data is augmented with zeros |
| 0 | Data is not augmented with zeros |

**Bits 4:3 – SETUP[1:0]**
Register Overlay Setup

| Value | Description |
|---|---|
| 11 | CRC Register Overlay Selection; Read / Write access to CRCOUT |
| 10 | CRC Register Overlay Selection; Read / Write access to CRCXOR |
| 01 | CRC Register Overlay Selection; Read / Write access to CRCSHIFT |
| 00 | CRC Register Overlay Selection; Read / Write access to CRCOUT |

**Bit 1 – SHIFTM** Shift Mode

| Value | Description |
|---|---|
| 1 | Shift right (LSb first) |
| 0 | Shift left (MSb first) |

**Bit 0 – FULL** Data Path Full Indicator

| Value | Description |
|---|---|
| 1 | CRCDATAT/U/H/L registers are full |
| 0 | CRCDATAT/U/H/L registers have shifted their data into the shifter |

© 2023 Microchip Technology Inc.
and its subsidiaries

chanhvinh.com



### 19.14.2 CRCCON1

**Name:** CRCCON1
**Offset:** 0x1CA3

CRC Control Register 1

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | PLEN[4:0] | | | | |
| Access | | | | R/W | R/W | R/W | R/W | R/W |
| Reset | | | | 0 | 0 | 0 | 0 | 0 |

**Bits 4:0 – PLEN[4:0]** Polynomial Length
Denotes the length of the polynomial (n-1)

chanhvinh.com

chanhvinh.com



### 19.14.3 CRCCON2

**Name:** CRCCON2
**Offset:** 0x1CA4

CRC Control Register 2

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | DLEN[4:0] | | | | |
| Access | | | | R/W | R/W | R/W | R/W | R/W |
| Reset | | | | 0 | 0 | 0 | 0 | 0 |

**Bits 4:0 – DLEN[4:0]** Data Length
Denotes the length of the data word (n-1)

chanhvinh.com

chanhvinh.com



### 19.14.4 CRCDATA

**Name:** CRCDATA
**Offset:** 0x1C9A

CRC Data Registers

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | CRCDATAT[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | CRCDATAU[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | CRCDATAH[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CRCDATAL[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:24 – CRCDATAT[7:0]** CRC Data Top Byte

**Bits 23:16 – CRCDATAU[7:0]** CRC Data Upper Byte

**Bits 15:8 – CRCDATAH[7:0]** CRC Data High Byte

**Bits 7:0 – CRCDATAL[7:0]** CRC Data Low Byte

© 2023 Microchip Technology Inc.
and its subsidiaries

chanhvinh.com



### 19.14.5 CRCOUT

**Name:** CRCOUT
**Offset:** 0x1C9E

CRC Output Registers

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | CRCOUTT[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | CRCOUTU[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | CRCOUTH[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CRCOUTL[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:24 – CRCOUTT[7:0]** CRC Output Register Top Byte
Writing to this register writes the Most Significant Byte of the CRC output register. Reading from this register reads the Most Significant Byte of the CRC output.

**Bits 23:16 – CRCOUTU[7:0]** CRC Output Register Upper Byte

**Bits 15:8 – CRCOUTH[7:0]** CRC Output Register High Byte

**Bits 7:0 – CRCOUTL[7:0]** CRC Output Register Low Byte
Writing to this register writes the Least Significant Byte of the CRC output register. Reading from this register reads the Least Significant Byte of the CRC output.

chanhvinh.com

chanhvinh.com



### 19.14.6 CRCSHIFT

**Name:** CRCSHIFT
**Offset:** 0x1C9E

CRC Shift Registers

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | CRCSHIFTT[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | CRCSHIFTU[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | CRCSHIFTH[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CRCSHIFTL[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:24 – CRCSHIFTT[7:0]** CRC Shift Register Top Byte
Reading from this register reads the Most Significant Byte of the CRC Shifter.

**Bits 23:16 – CRCSHIFTU[7:0]** CRC Shift Register Upper Byte

**Bits 15:8 – CRCSHIFTH[7:0]** CRC Shift Register High Byte

**Bits 7:0 – CRCSHIFTL[7:0]** CRC Shift Register Low Byte
Reading from this register reads the Least Significant Byte of the CRC Shifter.

© 2023 Microchip Technology Inc.
and its subsidiaries

chanhvinh.com



### 19.14.7 CRCXOR

**Name:** CRCXOR
**Offset:** 0x1C9E

CRC XOR Registers

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | CRCXORT[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | CRCXORU[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | CRCXORH[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CRCXORL[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:24 – CRCXORT[7:0]** XOR of Polynomial Term XN Enable Top Byte

**Bits 23:16 – CRCXORU[7:0]** XOR of Polynomial Term XN Enable Upper Byte

**Bits 15:8 – CRCXORH[7:0]** XOR of Polynomial Term XN Enable High Byte

**Bits 7:0 – CRCXORL[7:0]** XOR of Polynomial Term XN Enable Low Byte

chanhvinh.com

chanhvinh.com



### 19.14.8 SCANCON0

**Name:** SCANCON0
**Offset:** 0x1C98

Scanner Access Control Register 0

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | EN | SGO | BUSY | DABORT | INTM | | MD[1:0] | |
| Access | R/W | R/W/HC | R | R | R/W | | R/W | R/W |
| Reset | 0 | 0 | 0 | 1 | 0 | | 0 | 0 |

**Bit 7 – EN** Scanner Enable(1)

| Value | Description |
|---|---|
| 1 | Scanner is enabled |
| 0 | Scanner is disabled |

**Bit 6 – SGO** Scanner GO(2,3)

| Value | Description |
|---|---|
| 1 | Begin scanner operations |
| 0 | Scanner operations will not occur |

**Bit 5 – BUSY** Scanner Busy Indicator

| Value | Description |
|---|---|
| 1 | Scanner cycle is in process |
| 0 | Scanner cycle is compete (or never started) |

**Bit 4 – DABORT** Scanner Abort Signal

| Value | Description |
|---|---|
| 1 | SCANLADR points to an invalid NVM address, or the CRC is disabled |
| 0 | SCANLADR points to a valid NVM address and the CRC is enabled |

**Bit 3 – INTM** Scanner Interrupt Management Mode Select

| Value | Name | Description |
|---|---|---|
| 1 | MD = '00', '01', or '11' | Scan operations are paused during the interrupt response and resume after the interrupt has been serviced |
| 0 | MD = '00' or '11' | Interrupt operations are stalled each time the scanner accesses memory and resume between scan operations |
| 0 | MD = '01' | Interrupt operations are stalled until SGO is cleared by hardware |
| x | MD = '10' | This bit is ignored |

**Bits 1:0 – MD[1:0]** Scanner Memory Access Mode Select(4)

| Value | Description |
|---|---|
| 11 | Trigger Mode |
| 10 | Peek Mode |
| 01 | Burst Mode |
| 00 | Concurrent Mode |

**Notes:**

1. Setting EN = 0 does not affect any other register content.
2. This bit can be cleared in software. It is cleared in hardware when LADR > HADR (and a data cycle is not occurring) or when CRCGO = 0.
3. CRCEN and CRCGO bits must be set before setting the SGO bit.
4. Trigger Mode source selection can be set using the SCANTRIG register.

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



### 19.14.9 SCANLADR

**Name:** SCANLADR
**Offset:** 0x1C95

Scan Low Address Registers

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | SCANLADRH[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SCANLADRL[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 15:8 – SCANLADRH[7:0]** Scan Start/Current Address high byte
High byte of the current address to be fetched from, value increments on each fetch of memory.

**Bits 7:0 – SCANLADRL[7:0]** Scan Start/Current Address low byte
Low byte of the current address to be fetched from, value increments on each fetch of memory.

**Notes:**

1. Registers SCANLADRH/L form a 16-bit value, but are not guarded for atomic or asynchronous access; registers may only be read or written while SGO = 0.
2. While SGO = 1, writing to this register is ignored.

chanhvinh.com



### 19.14.10 SCANHADR

**Name:** SCANHADR
**Offset:** 0x1C92

Scan High Address Registers

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | SCANHADRH[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SCANHADRL[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

**Bits 15:8 – SCANHADRH[7:0]** Scan End Address
High byte of the address at the end of the designated scan.

**Bits 7:0 – SCANHADRL[7:0]** Scan End Address
Low byte of the address at the end of the designated scan.

**Notes:**

1. Registers SCANHADRH/L form a 16-bit value, but are not guarded for atomic or asynchronous access; registers may only be read or written while SGO = 0.
2. While SGO = 1, writing to this register is ignored.

chanhvinh.com



### 19.14.11 SCANTRIG

**Name:** SCANTRIG
**Offset:** 0x1C99

SCAN Trigger Selection Register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | TSEL[3:0] | | | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

**Bits 3:0 – TSEL[3:0]** Scanner Data Trigger Input Selection

**Table 19-2. Scanner Data Trigger Input Sources(1)**

| TSEL Value | Trigger Input Sources |
|---|---|
| 1111-1100 | Reserved |
| 1011 | CLC4_OUT |
| 1010 | CLC3_OUT |
| 1001 | CLC2_OUT |
| 1000 | CLC1_OUT |
| 0111 | Reserved |
| 0110 | TMR4_Postscaled_OUT |
| 0101 | TMR3_overflow |
| 0100 | TMR2_Postscaled_OUT |
| 0011 | TMR1_overflow |
| 0010 | TMR0_overflow |
| 0001 | CLKR_OUT |
| 0000 | LFINTOSC |

**Note:**

1. The number of implemented bits varies by device.

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



## 19.15 Register Summary - CRC

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 ... 0x1C91 | Reserved | | | | | | | | | |
| 0x1C92 | SCANHADR | 7:0 | SCANHADRL[7:0] | | | | | | | |
| | | 15:8 | SCANHADRH[7:0] | | | | | | | |
| 0x1C94 | Reserved | | | | | | | | | |
| 0x1C95 | SCANLADR | 7:0 | SCANLADRL[7:0] | | | | | | | |
| | | 15:8 | SCANLADRH[7:0] | | | | | | | |
| 0x1C97 | Reserved | | | | | | | | | |
| 0x1C98 | SCANCON0 | 7:0 | EN | SGO | BUSY | DABORT | INTM | | MD[1:0] | |
| 0x1C99 | SCANTRIG | 7:0 | | | | | TSEL[3:0] | | | |
| 0x1C9A | CRCDATA | 7:0 | CRCDATAL[7:0] | | | | | | | |
| | | 15:8 | CRCDATAH[7:0] | | | | | | | |
| | | 23:16 | CRCDATAU[7:0] | | | | | | | |
| | | 31:24 | CRCDATAT[7:0] | | | | | | | |
| 0x1C9E | CRCOUT | 7:0 | CRCOUTL[7:0] | | | | | | | |
| | | 15:8 | CRCOUTH[7:0] | | | | | | | |
| | | 23:16 | CRCOUTU[7:0] | | | | | | | |
| | | 31:24 | CRCOUTT[7:0] | | | | | | | |
| 0x1C9E | CRCSHIFT | 7:0 | CRCSHIFTL[7:0] | | | | | | | |
| | | 15:8 | CRCSHIFTH[7:0] | | | | | | | |
| | | 23:16 | CRCSHIFTU[7:0] | | | | | | | |
| | | 31:24 | CRCSHIFTT[7:0] | | | | | | | |
| 0x1C9E | CRCXOR | 7:0 | CRCXORL[7:0] | | | | | | | |
| | | 15:8 | CRCXORH[7:0] | | | | | | | |
| | | 23:16 | CRCXORU[7:0] | | | | | | | |
| | | 31:24 | CRCXORT[7:0] | | | | | | | |
| 0x1CA2 | CRCCON0 | 7:0 | EN | GO | BUSY | ACCM | SETUP[1:0] | | SHIFTM | FULL |
| 0x1CA3 | CRCCON1 | 7:0 | | | | PLEN[4:0] | | | | |
| 0x1CA4 | CRCCON2 | 7:0 | | | | DLEN[4:0] | | | | |

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



# 20. PMD - Peripheral Module Disable

## 20.1 Overview

This module provides the ability to selectively enable or disable a peripheral. Disabling a peripheral places it in its lowest possible Power state. The user can selectively disable unused modules to reduce the overall power consumption.

**Important:** All modules are ON by default following any system Reset.

## 20.2 Disabling a Module

A peripheral can be disabled by setting the corresponding peripheral disable bit in the PMDx register. Disabling a module has the following effects:

- The module is held in Reset and does not function.
- All the SFRs pertaining to that peripheral become “unimplemented”
  - Writing is disabled
  - Reading returns `0x00`
- Module outputs are disabled

## 20.3 Enabling a Module

Clearing the corresponding module disable bit in the PMDx register, re-enables the module and the SFRs will reflect the Power-on Reset values.

**Important:** There will be no reads/writes to the module SFRs for at least two instruction cycles after it has been re-enabled.

## 20.4 Register Definitions: Peripheral Module Disable

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



### 20.4.1 PMD0

**Name:** PMD0
**Offset:** 0x010C

PMD Control Register 0

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TMR0MD | CLKRMD | IOCMD | ACTMD | SYSCMD | SCANMD | CRCMD | NVMMD |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 7 – TMR0MD** Disable TMR0

| Value | Description |
|---|---|
| 1 | TMR0 module disabled |
| 0 | TMR0 module enabled |

**Bit 6 – CLKRMD** Disable Clock Reference

| Value | Description |
|---|---|
| 1 | Clock reference module disabled |
| 0 | Clock reference module enabled |

**Bit 5 – IOCMD** Disable Interrupt-on-Change

| Value | Description |
|---|---|
| 1 | Interrupt-on-change module is disabled |
| 0 | Interrupt-on-change module is enabled |

**Bit 4 – ACTMD** Disable Active Clock Tuning

| Value | Description |
|---|---|
| 1 | ACT module disabled |
| 0 | ACT module enabled |

**Bit 3 – SYSCMD** Disable Peripheral System Clock Network(1)

| Value | Description |
|---|---|
| 1 | System clock network disabled ($F_{OSC}$) |
| 0 | System clock network enabled |

**Bit 2 – SCANMD** Disable NVM Memory Scanner

| Value | Description |
|---|---|
| 1 | NVM memory scanner module disabled |
| 0 | NVM memory scanner module enabled |

**Bit 1 – CRCMD** Disable CRC Module

| Value | Description |
|---|---|
| 1 | CRC module disabled |
| 0 | CRC module enabled |

**Bit 0 – NVMMD** Disable NVM access

| Value | Description |
|---|---|
| 1 | All memory reading and writing is disabled; NVMCON registers cannot be written |
| 0 | All memory reading and writing is enabled |

**Note:**

1. Clearing the SYSCMD bit disables the system clock ($F_{OSC}$) to peripherals, however peripherals clocked by $F_{OSC}/4$ are not affected.

© 2023 Microchip Technology Inc.
and its subsidiaries

chanhvinh.com



### 20.4.2 PMD1

**Name:** PMD1
**Offset:** 0x010D

PMD Control Register 1

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PWM1MD | CCP2MD | CCP1MD | TMR6MD | TMR4MD | TMR2MD | TMR3MD | TMR1MD |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 7 – PWM1MD** Disable PWM1 Module

| Value | Description |
|---|---|
| 1 | PWM1 module disabled |
| 0 | PWM1 module enabled |

**Bit 6 – CCP2MD** Disable CCP2 Module

| Value | Description |
|---|---|
| 1 | CCP2 module disabled |
| 0 | CCP2 module enabled |

**Bit 5 – CCP1MD** Disable CCP1 Module

| Value | Description |
|---|---|
| 1 | CCP1 module disabled |
| 0 | CCP1 module enabled |

**Bit 4 – TMR6MD** Disable Timer TMR6

| Value | Description |
|---|---|
| 1 | TMR6 module disabled |
| 0 | TMR6 module enabled |

**Bit 3 – TMR4MD** Disable Timer TMR4

| Value | Description |
|---|---|
| 1 | TMR4 module disabled |
| 0 | TMR4 module enabled |

**Bit 2 – TMR2MD** Disable Timer TMR2

| Value | Description |
|---|---|
| 1 | TMR2 module disabled |
| 0 | TMR2 module enabled |

**Bit 1 – TMR3MD** Disable Timer TMR3

| Value | Description |
|---|---|
| 1 | TMR3 module disabled |
| 0 | TMR3 module enabled |

**Bit 0 – TMR1MD** Disable Timer TMR1

| Value | Description |
|---|---|
| 1 | TMR1 module disabled |
| 0 | TMR1 module enabled |

© 2023 Microchip Technology Inc.
and its subsidiaries

chanhvinh.com



### 20.4.3 PMD2

**Name:** PMD2
**Offset:** 0x010E

PMD Control Register 2

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CLC3MD | CLC2MD | CLC1MD | CWG1MD | NCO1MD | PWM4MD | PWM3MD | PWM2MD |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 7 – CLC3MD** Disable CLC3

| Value | Description |
|---|---|
| 1 | CLC3 disabled |
| 0 | CLC3 enabled |

**Bit 6 – CLC2MD** Disable CLC2

| Value | Description |
|---|---|
| 1 | CLC2 disabled |
| 0 | CLC2 enabled |

**Bit 5 – CLC1MD** Disable CLC1

| Value | Description |
|---|---|
| 1 | CLC1 disabled |
| 0 | CLC1 enabled |

**Bit 4 – CWG1MD** Disable CWG1

| Value | Description |
|---|---|
| 1 | CWG1 disabled |
| 0 | CWG1 enabled |

**Bit 3 – NCO1MD** Disable NCO1

| Value | Description |
|---|---|
| 1 | NCO1 disabled |
| 0 | NCO1 enabled |

**Bit 2 – PWM4MD** Disable PWM4

| Value | Description |
|---|---|
| 1 | PWM4 disabled |
| 0 | PWM4 enabled |

**Bit 1 – PWM3MD** Disable PWM3

| Value | Description |
|---|---|
| 1 | PWM3 disabled |
| 0 | PWM3 enabled |

**Bit 0 – PWM2MD** Disable PWM2

| Value | Description |
|---|---|
| 1 | PWM2 disabled |
| 0 | PWM2 enabled |

© 2023 Microchip Technology Inc.
and its subsidiaries

chanhvinh.com



### 20.4.4 PMD3

**Name:** PMD3
**Offset:** 0x010F

PMD Control Register 3

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CM2MD | CM1MD | FVRMD | MSSP2MD | MSSP1MD | UART2MD | UART1MD | CLC4MD |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 7 – CM2MD** Disable Comparator 2

| Value | Description |
|---|---|
| 1 | CM2 module disabled |
| 0 | CM2 module enabled |

**Bit 6 – CM1MD** Disable Comparator 1

| Value | Description |
|---|---|
| 1 | CM1 module disabled |
| 0 | CM1 module enabled |

**Bit 5 – FVRMD** Disable Fixed Voltage Reference Module

| Value | Description |
|---|---|
| 1 | FVR module disabled |
| 0 | FVR module enabled |

**Bit 4 – MSSP2MD** Disable MSSP2 Module

| Value | Description |
|---|---|
| 1 | MSSP2 module disabled |
| 0 | MSSP2 module enabled |

**Bit 3 – MSSP1MD** Disable MSSP1 Module

| Value | Description |
|---|---|
| 1 | MSSP1 module disabled |
| 0 | MSSP1 module enabled |

**Bit 2 – UART2MD** Disable UART2 Module

| Value | Description |
|---|---|
| 1 | UART2 module disabled |
| 0 | UART2 module enabled |

**Bit 1 – UART1MD** Disable UART1 Module

| Value | Description |
|---|---|
| 1 | UART1 module disabled |
| 0 | UART1 module enabled |

**Bit 0 – CLC4MD** Disable CLC4

| Value | Description |
|---|---|
| 1 | CLC4 disabled |
| 0 | CLC4 enabled |

© 2023 Microchip Technology Inc.
and its subsidiaries

chanhvinh.com



### 20.4.5 PMD4

**Name:** PMD4
**Offset:** 0x0110

PMD Control Register 4

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | ZCDMD | OPA1MD | DAC2MD | DAC1MD | ADCMD |
| Access | | | | R/W | R/W | R/W | R/W | R/W |
| Reset | | | | 0 | 0 | 0 | 0 | 0 |

**Bit 4 – ZCDMD** Disable Zero-Cross Detect(1)

| Value | Description |
|---|---|
| 1 | ZCD module disabled |
| 0 | ZCD module enabled |

**Bit 3 – OPA1MD** Disable Operational Amplifier 1

| Value | Description |
|---|---|
| 1 | OPA1 module disabled |
| 0 | OPA1 module enabled |

**Bit 2 – DAC2MD** Disable Digital-to-Analog Converter 2

| Value | Description |
|---|---|
| 1 | DAC2 module disabled |
| 0 | DAC2 module enabled |

**Bit 1 – DAC1MD** Disable Digital-to-Analog Converter 1

| Value | Description |
|---|---|
| 1 | DAC1 module disabled |
| 0 | DAC1 module enabled |

**Bit 0 – ADCMD** Disable Analog-to-Digital Converter

| Value | Description |
|---|---|
| 1 | ADC module disabled |
| 0 | ADC module enabled |

**Note:**

1. Subject to the value of $\overline{ZCD}$ Configuration bit.

chanhvinh.com



## 20.5 Register Summary - PMD

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 ... 0x010B | Reserved | | | | | | | | | |
| 0x010C | PMD0 | 7:0 | TMR0MD | CLKRMD | IOCMD | ACTMD | SYSCMD | SCANMD | CRCMD | NVMMD |
| 0x010D | PMD1 | 7:0 | PWM1MD | CCP2MD | CCP1MD | TMR6MD | TMR4MD | TMR2MD | TMR3MD | TMR1MD |
| 0x010E | PMD2 | 7:0 | CLC3MD | CLC2MD | CLC1MD | CWG1MD | NCO1MD | PWM4MD | PWM3MD | PWM2MD |
| 0x010F | PMD3 | 7:0 | CM2MD | CM1MD | FVRMD | MSSP2MD | MSSP1MD | UART2MD | UART1MD | CLC4MD |
| 0x0110 | PMD4 | 7:0 | | | | ZCDMD | OPA1MD | DAC2MD | DAC1MD | ADCMD |

chanhvinh.com

chanhvinh.com



# 21. CLKREF - Reference Clock Output Module

The reference clock output module provides the ability to send a clock signal to the clock reference output pin (CLKR). The reference clock output can be routed internally as an input signal for other peripherals, such as the timers and CLCs.

The reference clock output module has the following features:

- Selectable clock source using the CLKRCLK register
- Programmable clock divider
- Selectable duty cycle

The figure below shows the simplified block diagram of the clock reference module.

**Figure 21-1. Clock Reference Block Diagram**

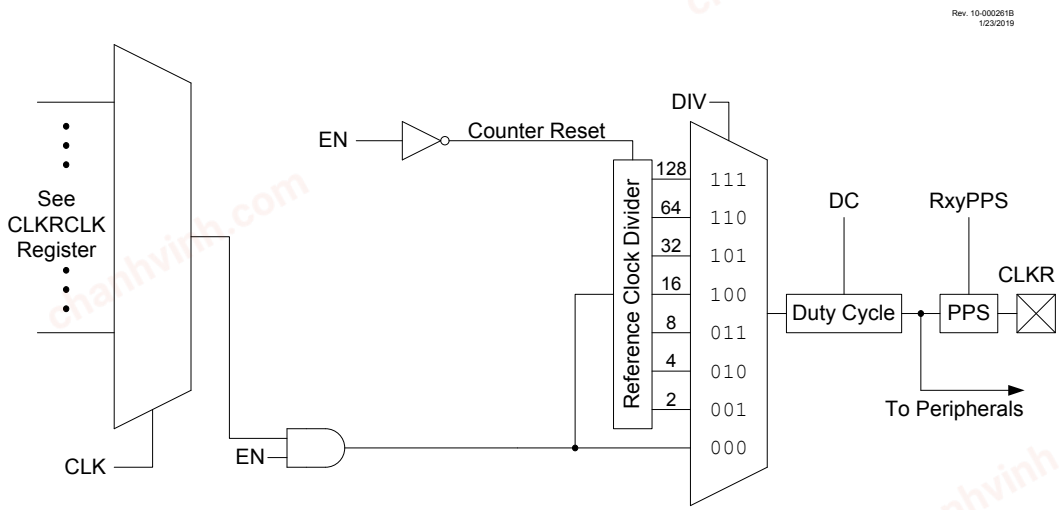
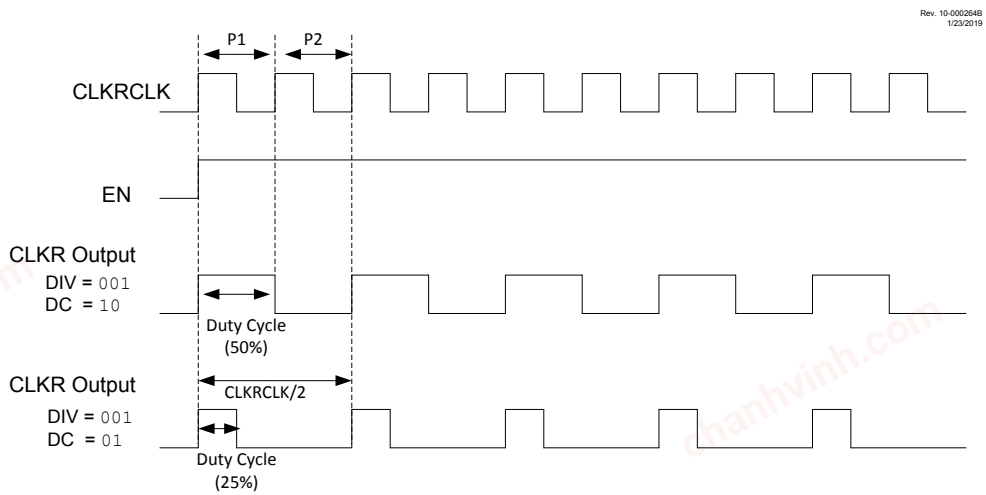



**Figure 21-2. Clock Reference Timing**




## 21.1 Clock Source

The clock source of the reference clock peripheral is selected with the CLK bits.

chanhvinh.com



### 21.1.1 Clock Synchronization

The CLKR output signal is ensured to be glitch-free when the EN bit is set to start the module and enable the CLKR output. When the reference clock output is disabled, the output signal will be disabled immediately.

## 21.2 Programmable Clock Divider

The module takes the clock input and divides it based on the value of the DIV bits.

The following configurations are available:

- Base clock frequency value
- Base clock frequency divided by 2
- Base clock frequency divided by 4
- Base clock frequency divided by 8
- Base clock frequency divided by 16
- Base clock frequency divided by 32
- Base clock frequency divided by 64
- Base clock frequency divided by 128

## 21.3 Selectable Duty Cycle

The DC bits are used to modify the duty cycle of the output clock. A duty cycle of 0%, 25%, 50%, or 75% can be selected for all clock rates when the DIV value is not `0b000`. When DIV = `0b000`, the duty cycle defaults to 50% for all values of DC except `0b00`, in which case the duty cycle is 0% (constant low output).

**Important:** The DC value at Reset is `10`. This makes the default duty cycle 50% and not 0%.

**Important:** Clock dividers and clock duty cycles can be changed while the module is enabled but doing so may cause glitches to occur on the output. To avoid possible glitches, clock dividers and clock duty cycles will be changed only when the EN bit is clear.

## 21.4 Operation in Sleep Mode

The reference clock module continues to operate and provide a signal output in Sleep for all clock source selections except $F_{OSC}$ (CLK = `0`).

## 21.5 Register Definitions: Reference Clock

Long bit name prefixes for the Reference Clock peripherals are shown in the following table. Refer to the **"Long Bit Names"** section in the **"Register and Bit Naming Conventions"** chapter for more information.

**Table 21-1. CLKREF Long Bit Name Prefixes**

| Peripheral | Bit Name Prefix |
|---|---|
| CLKR | CLKR |

chanhvinh.com



### 21.5.1 CLKRCON

**Name:** CLKRCON
**Offset:** 0x0296

Reference Clock Control Register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | EN | | | DC[1:0] | | DIV[2:0] | | |
| Access | R/W | | | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | | | 1 | 0 | 0 | 0 | 0 |

**Bit 7 – EN** Reference Clock Module Enable

| Value | Description |
|---|---|
| 1 | Reference clock module enabled |
| 0 | Reference clock module is disabled |

**Bits 4:3 – DC[1:0]** Reference Clock Duty Cycle(1)

| Value | Description |
|---|---|
| 11 | Clock outputs duty cycle of 75% |
| 10 | Clock outputs duty cycle of 50% |
| 01 | Clock outputs duty cycle of 25% |
| 00 | Clock outputs duty cycle of 0% |

**Bits 2:0 – DIV[2:0]** Reference Clock Divider

| Value | Description |
|---|---|
| 111 | Base clock value divided by 128 |
| 110 | Base clock value divided by 64 |
| 101 | Base clock value divided by 32 |
| 100 | Base clock value divided by 16 |
| 011 | Base clock value divided by 8 |
| 010 | Base clock value divided by 4 |
| 001 | Base clock value divided by 2 |
| 000 | Base clock value |

**Note:**

1. Bits are valid for DIV ≥ 001. For DIV = 000, duty cycle is fixed at 50%.

chanhvinh.com



### 21.5.2 CLKRCLK

**Name:** CLKRCLK
**Offset:** 0x0297

Clock Reference Clock Selection Register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | CLK[3:0] | | | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

**Bits 3:0 – CLK[3:0]** CLKR Clock Selection

**Table 21-2. Clock Reference Module Clock Sources**

| CLK | Clock Source |
|---|---|
| 1111-1100 | Reserved |
| 1011 | CLC4_OUT |
| 1010 | CLC3_OUT |
| 1001 | CLC2_OUT |
| 1000 | CLC1_OUT |
| 0111 | NCO1_OUT |
| 0110 | EXTOSC |
| 0101 | SOSC |
| 0100 | MFINTOSC (32 kHz) |
| 0011 | MFINTOSC (500 kHz) |
| 0010 | LFINTOSC |
| 0001 | HFINTOSC |
| 0000 | $F_{OSC}$ |

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



## 21.6 Register Summary - Reference CLK

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00<br>...<br>0x0295 | Reserved | | | | | | | | | |
| 0x0296 | CLKRCON | 7:0 | EN | | | DC[1:0] | | DIV[2:0] | | |
| 0x0297 | CLKRCLK | 7:0 | | | | | CLK[3:0] | | | |

chanhvinh.com

chanhvinh.com

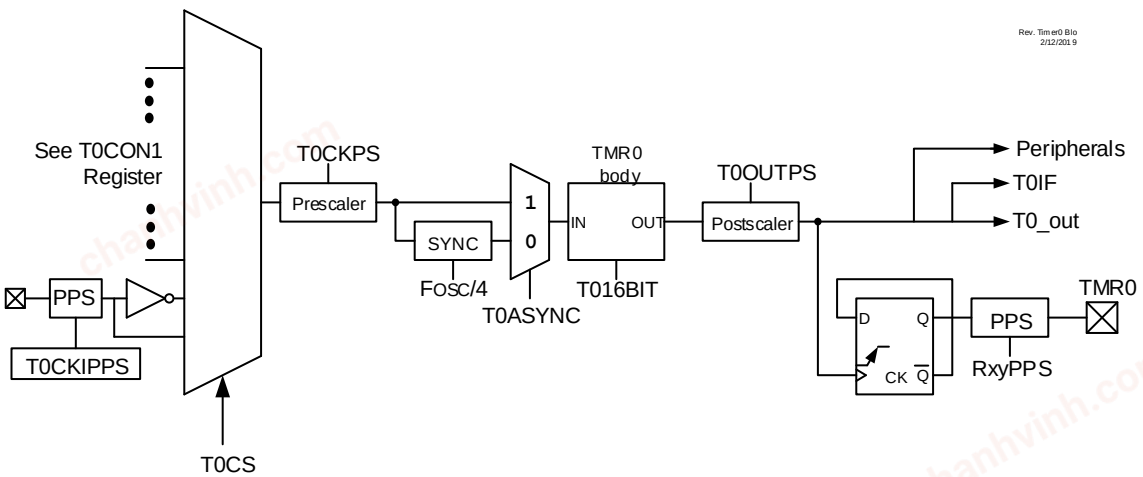


# 22. TMR0 - Timer0 Module

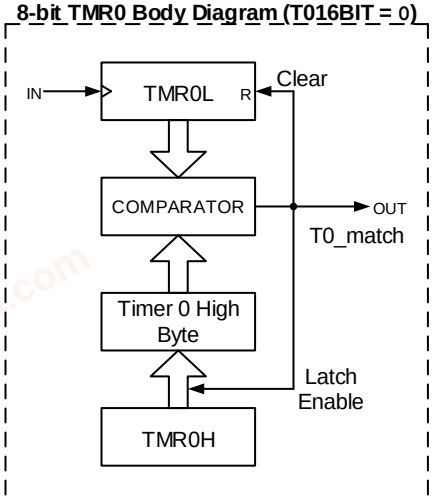
The Timer0 module has the following features:

- 8-bit timer with programmable period
- 16-bit timer
- Selectable clock sources
- Synchronous and asynchronous operation
- Programmable prescaler (Independent of Watchdog Timer)
- Programmable postscaler
- Interrupt on match or overflow
- Output on I/O pin (via PPS) or to other peripherals
- Operation during Sleep

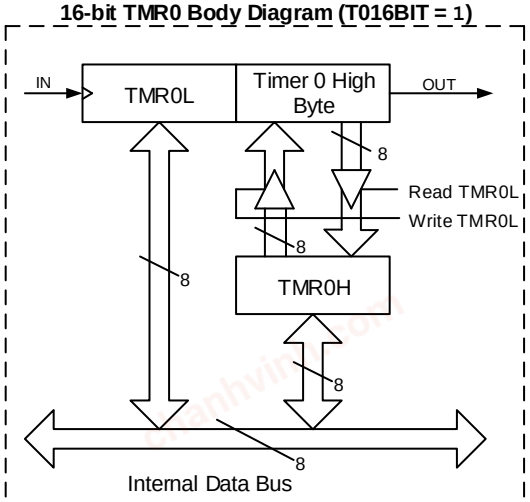
**Figure 22-1. Timer0 Block Diagram**




**8-bit TMR0 Body Diagram (T016BIT = 0)**




**16-bit TMR0 Body Diagram (T016BIT = 1)**




## 22.1 Timer0 Operation

Timer0 can operate as either an 8-bit or 16-bit timer. The mode is selected with the MD16 bit.

chanhvinh.com



### 22.1.1 8-Bit Mode

In this mode, Timer0 increments on the rising edge of the selected clock source. A prescaler on the clock input gives several prescale options (see the prescaler control bits, CKPS). In this mode, as shown in Figure 22-1, a buffered version of TMR0H is maintained.

This is compared with the value of TMR0L on each cycle of the selected clock source. When the two values match, the following events occur:

- TMR0L is reset
- The contents of TMR0H are copied to the TMR0H buffer for next comparison

### 22.1.2 16-Bit Mode

In this mode, Timer0 increments on the rising edge of the selected clock source. A prescaler on the clock input gives several prescale options (see the prescaler control bits, CKPS). In this mode, TMR0H:TMR0L form the 16-bit timer value. As shown in Figure 22-1, reads and writes of the TMR0H register are buffered. The TMR0H register is updated with the contents of the high byte of Timer0 when the TMR0L register is read. Similarly, writing the TMR0L register causes a transfer of the TMR0H register value to the Timer0 high byte.

This buffering allows all 16 bits of Timer0 to be read and written at the same time. Timer0 rolls over to `0x0000` on incrementing past `0xFFFF`. This makes the timer free-running. While actively operating in 16-bit mode, the Timer0 value can be read but not written.

## 22.2 Clock Selection

Timer0 has several options for clock source selections, the option to operate synchronously/asynchronously and an available programmable prescaler. The CS bits are used to select the clock source for Timer0.

### 22.2.1 Synchronous Mode

When the ASYNC bit is clear, Timer0 clock is synchronized to the system clock ($F_{OSC}/4$). When operating in Synchronous mode, Timer0 clock frequency cannot exceed $F_{OSC}/4$. During Sleep mode, the system clock is not available and Timer0 cannot operate.

### 22.2.2 Asynchronous Mode

When the ASYNC bit is set, Timer0 increments with each rising edge of the input source (or output of the prescaler, if used). Asynchronous mode allows Timer0 to continue operation during Sleep mode provided the selected clock source operates during Sleep.

### 22.2.3 Programmable Prescaler

Timer0 has 16 programmable input prescaler options ranging from 1:1 to 1:32768. The prescaler values are selected using the CKPS bits. The prescaler counter is not directly readable or writable. The prescaler counter is cleared on the following events:

- A write to the TMR0L register
- A write to either the T0CON0 or T0CON1 registers
- Any device Reset

### 22.2.4 Programmable Postscaler

Timer0 has 16 programmable output postscaler options ranging from 1:1 to 1:16. The postscaler values are selected using the OUTPS bits. The postscaler divides the output of Timer0 by the selected ratio. The postscaler counter is not directly readable or writable. The postscaler counter is cleared on the following events:

- A write to the TMR0L register
- A write to either the T0CON0 or T0CON1 registers
- Any device Reset

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



## 22.3 Timer0 Output and Interrupt

### 22.3.1 Timer0 Output

TMR0_out toggles on every match between TMR0L and TMR0H in 8-bit mode, or when TMR0H:TMR0L rolls over in 16-bit mode. If the output postscaler is used, the output is scaled by the ratio selected. The Timer0 output can be routed to an I/O pin via the RxyPPS output selection register, or internally to a number of Core Independent Peripherals. The Timer0 output can be monitored through software via the OUT output bit.

### 22.3.2 Timer0 Interrupt

The Timer0 Interrupt Flag (TMR0IF) bit is set when the TMR0_out toggles. If the Timer0 interrupt is enabled (TMR0IE), the CPU will be interrupted when the TMR0IF bit is set. When the postscaler bits (T0OUTPS) are set to 1:1 operation (no division), the T0IF flag bit will be set with every TMR0 match or rollover. In general, the TMR0IF flag bit will be set every T0OUTPS +1 matches or rollovers.

### 22.3.3 Timer0 Example

Timer0 Configuration:

- Timer0 mode = 16-bit
- Clock Source = $F_{OSC}/4$ (250 kHz)
- Synchronous operation
- Prescaler = 1:1
- Postscaler = 1:2 (T0OUTPS = 1)

In this case, the TMR0_out toggles every two rollovers of TMR0H:TMR0L.
i.e., (0xFFFF)*2*(1/250 kHz) = 524.28 ms

## 22.4 Operation During Sleep

When operating synchronously, Timer0 will halt when the device enters Sleep mode. When operating asynchronously and the selected clock source is active, Timer0 will continue to increment and wake the device from Sleep mode if the Timer0 interrupt is enabled.

## 22.5 Register Definitions: Timer0 Control

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



### 22.5.1 T0CON0

**Name:** T0CON0
**Offset:** 0x019E

Timer0 Control Register 0

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | EN | | OUT | MD16 | OUTPS[3:0] | | | |
| Access | R/W | | R | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 7 – EN** TMR0 Enable

| Value | Description |
|---|---|
| 1 | The module is enabled and operating |
| 0 | The module is disabled |

**Bit 5 – OUT** TMR0 Output

**Bit 4 – MD16** 16-Bit Timer Operation Select

| Value | Description |
|---|---|
| 1 | TMR0 is a 16-bit timer |
| 0 | TMR0 is an 8-bit timer |

**Bits 3:0 – OUTPS[3:0]** TMR0 Output Postscaler (Divider) Select

| Value | Description |
|---|---|
| 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 |

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



### 22.5.2 T0CON1

**Name:** T0CON1
**Offset:** 0x019F

Timer0 Control Register 1

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CS[2:0] | | | ASYNC | CKPS[3:0] | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 7:5 – CS[2:0]** Timer0 Clock Source Select

**Table 22-1. Timer0 Clock Source Selections**

| T0CS | Clock Source |
|---|---|
| 111 | CLC1_OUT |
| 110 | SOSC |
| 101 | MFINTOSC (500 kHz) |
| 100 | LFINTOSC |
| 011 | HFINTOSC |
| 010 | $F_{OSC}/4$ |
| 001 | Pin selected by T0CKIPPS (Inverted) |
| 000 | Pin selected by T0CKIPPS (Noninverted) |

**Bit 4 – ASYNC** TMR0 Input Asynchronization Enable

| Value | Description |
|---|---|
| 1 | The input to the TMR0 counter is not synchronized to system clocks |
| 0 | The input to the TMR0 counter is synchronized to Fosc/4 |

**Bits 3:0 – CKPS[3:0]** Prescaler Rate Select

| Value | Description |
|---|---|
| 1111 | 1:32768 |
| 1110 | 1:16384 |
| 1101 | 1:8192 |
| 1100 | 1:4096 |
| 1011 | 1:2048 |
| 1010 | 1:1024 |
| 1001 | 1:512 |
| 1000 | 1:256 |
| 0111 | 1:128 |
| 0110 | 1:64 |
| 0101 | 1:32 |
| 0100 | 1:16 |
| 0011 | 1:8 |
| 0010 | 1:4 |
| 0001 | 1:2 |
| 0000 | 1:1 |

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



### 22.5.3 TMR0H

**Name:** TMR0H
**Offset:** 0x019D

Timer0 Period/Count High Register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TMR0H[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

**Bits 7:0 – TMR0H[7:0]** TMR0 Most Significant Counter

| Value | Condition | Description |
|---|---|---|
| 0 to 255 | MD16 = 0 | 8-bit Timer0 Period Value. TMR0L continues counting from 0 when this value is reached. |
| 0 to 255 | MD16 = 1 | 16-bit Timer0 Most Significant Byte |

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



### 22.5.4 TMR0L

**Name:** TMR0L
**Offset:** 0x019C

Timer0 Period/Count Low Register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TMR0L[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 7:0 – TMR0L[7:0]** TMR0 Least Significant Counter

| Value | Condition | Description |
|---|---|---|
| 0 to 255 | MD16 = 0 | 8-bit Timer0 Counter bits |
| 0 to 255 | MD16 = 1 | 16-bit Timer0 Least Significant Byte |

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



## 22.6 Register Summary - Timer0

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 ... 0x019B | Reserved | | | | | | | | | |
| 0x019C | TMR0L | 7:0 | TMR0L[7:0] | | | | | | | |
| 0x019D | TMR0H | 7:0 | TMR0H[7:0] | | | | | | | |
| 0x019E | T0CON0 | 7:0 | EN | | OUT | MD16 | OUTPS[3:0] | | | |
| 0x019F | T0CON1 | 7:0 | CS[2:0] | | | ASYNC | CKPS[3:0] | | | |

chanhvinh.com

chanhvinh.com



# 23. TMR1 - Timer1 Module with Gate Control

The Timer1 module is a 16-bit timer/counter with the following features:

- 16-bit timer/counter register pair (TMRxH:TMRxL)
- Programmable internal or external clock source
- 2-bit prescaler
- Clock source for optional comparator synchronization
- Multiple Timer1 gate (count enable) sources
- Interrupt-on-overflow
- Wake-up on overflow (external clock, Asynchronous mode only)
- 16-bit read/write operation
- Time base for the capture/compare function with the CCP modules
- Special event trigger (with CCP)
- Selectable gate source polarity
- Gate Toggle mode
- Gate Single Pulse mode
- Gate value status
- Gate event interrupt

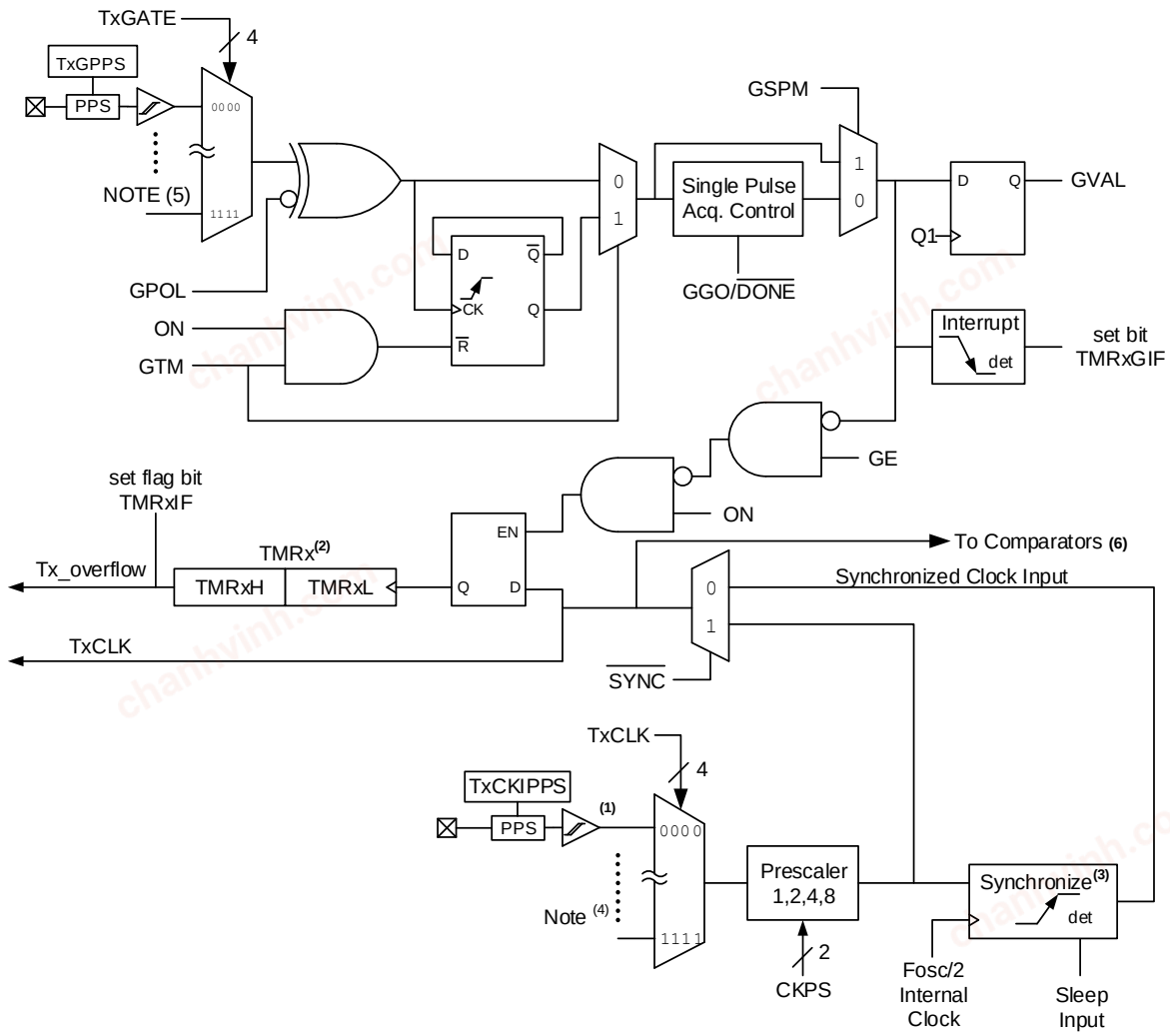
**Important:** References to the module Timer1 apply to all the odd numbered timers on this device.

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



**Figure 23-1. Timer1 Block Diagram**




**Notes:**

1. This signal comes from the pin selected by Timer1 PPS register.
2. TMRx register increments on rising edge.
3. Synchronize does not operate while in Sleep.
4. See TxCLK for clock source selections.
5. See TxGATE for gate source selections.
6. Synchronized comparator output must not be used in conjunction with synchronized input clock.

## 23.1 Timer1 Operation

The Timer1 module is a 16-bit incrementing counter accessed through the TMRx register. Writes to TMRx directly update the counter. When used with an internal clock source, the module is a timer that 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 ON and GE bits. Table 23-1 shows the possible Timer1 enable selections.

chanhvinh.com



**Table 23-1. Timer1 Enable Selections**

| ON | GE | Timer1 Operation |
|---|---|---|
| 1 | 1 | Count enabled |
| 1 | 0 | Always on |
| 0 | 1 | Off |
| 0 | 0 | Off |

## 23.2 Clock Source Selection

The CS bits select the clock source for Timer1. These bits allow the selection of several possible synchronous and asynchronous clock sources.

### 23.2.1 Internal Clock Source

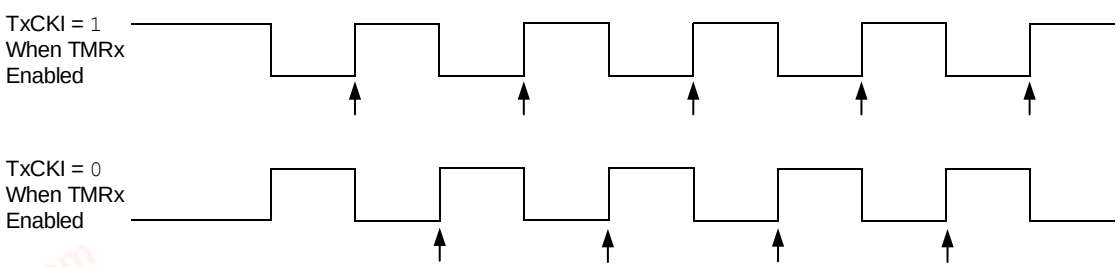
When the internal clock source is selected, the TMRx register will increment on multiples of $F_{OSC}$ as determined by the Timer1 prescaler.

When the $F_{OSC}$ internal clock source is selected, the TMRx register value will increment by four counts every instruction clock cycle. Due to this condition, a two LSB error in resolution will occur when reading the TMRx value. To utilize the full resolution of Timer1, an asynchronous input signal must be used to gate the Timer1 clock input.

**Important:** 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 TMRxH or TMRxL
- Timer1 is disabled
- Timer1 is disabled (ON = 0) when TxCKI is high, then Timer1 is enabled (ON = 1) when TxCKI is low. Refer to the figure below.

**Figure 23-2. Timer1 Incrementing Edge**




**Notes:**

1. Arrows indicate counter increments.
2. In Counter mode, a falling edge must be registered by the counter prior to the first incrementing rising edge of the clock.

### 23.2.2 External Clock Source

When the external clock source is selected, the TMRx 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 of the TxCKIPPS pin. This external clock source can be synchronized to the system clock or it can run asynchronously.

chanhvinh.com



## 23.3 Timer1 Prescaler

Timer1 has four prescaler options allowing 1, 2, 4 or 8 divisions of the clock input. The CKPS bits control the prescale counter. The prescale counter is not directly readable or writable; however, the prescaler counter is cleared upon a write to TMRx.

## 23.4 Secondary Oscillator

A secondary low-power 32.768 kHz oscillator circuit is built-in between pins SOSCI (input) and SOSCO (amplifier output). This internal circuit is to be used in conjunction with an external 32.768 kHz crystal. The secondary oscillator is not dedicated only to Timer1; it can also be used by other modules.

The oscillator circuit is enabled by setting the SOSCEN bit of the OSCEN register. This can be used as one of the Timer1 clock sources selected with the CS bits. The oscillator will continue to run during Sleep.

**Important:** The oscillator requires a start-up and stabilization time before use. Thus, the SOSCEN bit of the OSCEN register must be set and a suitable delay observed prior to enabling Timer1. A software check can be performed to confirm if the secondary oscillator is enabled and ready to use. This is done by polling the secondary oscillator ready Status bit. Refer to the **"OSC - Oscillator Module (with Fail-Safe Clock Monitor)"** chapter for more details.

## 23.5 Timer1 Operation in Asynchronous Counter Mode

When the SYNC Control bit 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.

**Important:** 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.

### 23.5.1 Reading and Writing TMRx in Asynchronous Counter Mode

Reading TMRxH or TMRxL while the timer is running from an external asynchronous clock will ensure a valid read (taken care of in hardware). However, the user must keep in mind that reading the 16-bit timer in two 8-bit values itself poses certain problems, since there may be a carry-out of TMRxL to TMRxH between the reads.

For writes, it is recommended that the user simply stop the timer and write the desired values. A write contention may occur by writing to the timer registers, while the register is incrementing. This may produce an unpredictable value in the TMRxH:TMRxL register pair.

## 23.6 Timer1 16-Bit Read/Write Mode

Timer1 can be configured to read and write all 16 bits of data to and from the 8-bit TMRxL and TMRxH registers, simultaneously. The 16-bit read and write operations are enabled by setting the RD16 bit. To accomplish this function, the TMRxH register value is mapped to a buffer register called the TMRxH buffer register. While in 16-bit mode, the TMRxH register is not directly readable or writable and all read and write operations take place through the use of this TMRxH buffer register.

When a read from the TMRxL register is requested, the value of the TMRxH register is simultaneously loaded into the TMRxH buffer register. When a read from the TMRxH register is requested, the value is provided from the TMRxH buffer register instead. This provides the user with the ability to accurately read all 16 bits of the Timer1 value from a

© 2023 Microchip Technology Inc. and its subsidiaries

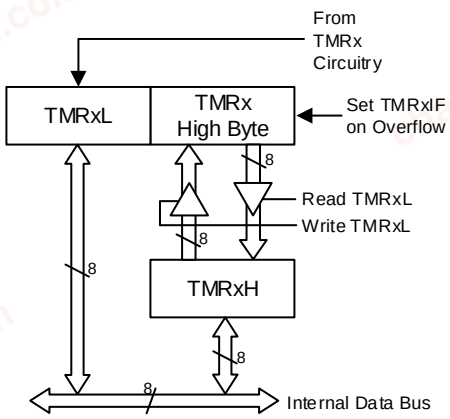
chanhvinh.com



single instance in time (refer to Figure 23-3 for more details). In contrast, when not in 16-bit mode, the user must read each register separately and determine if the values have become invalid due to a rollover that may have occurred between the read operations.

When a write request of the TMRxL register is requested, the TMRxH buffer register is simultaneously updated with the contents of the TMRxH register. The value of TMRxH must be preloaded into the TMRxH buffer register prior to the write request for the TMRxL register. This provides the user with the ability to write all 16 bits to the TMRx register at the same time. Any requests to write to TMRxH directly does not clear the Timer1 prescaler value. The prescaler value is only cleared through write requests to the TMRxL register.

**Figure 23-3. Timer1 16-Bit Read/Write Mode Block Diagram**




## 23.7 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.

### 23.7.1 Timer1 Gate Enable

The Timer1 Gate Enable mode is enabled by setting the GE bit. The polarity of the Timer1 Gate Enable mode is configured using the GPOL bit.

When Timer1 Gate Enable mode is enabled, Timer1 will increment on the rising edge of the Timer1 clock source. When Timer1 Gate signal is inactive, the timer will not increment and hold the current count. Enable mode is disabled, no incrementing will occur and Timer1 will hold the current count. See Figure 23-4 for timing details.

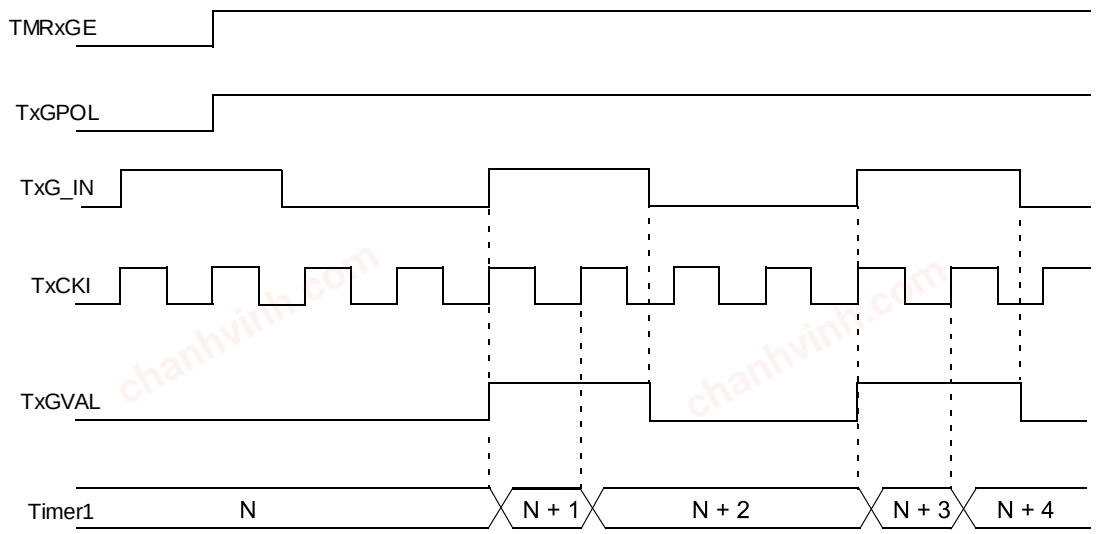
**Table 23-2. Timer1 Gate Enable Selections**

| TMRxCLK | GPOL | TxG | Timer1 Operation |
|---|---|---|---|
| ↑ | 1 | 1 | Counts |
| ↑ | 1 | 0 | Holds Count |
| ↑ | 0 | 1 | Holds Count |
| ↑ | 0 | 0 | Counts |

chanhvinh.com



**Figure 23-4. Timer1 Gate Enable Mode**




### 23.7.2 Timer1 Gate Source Selection

The gate source for Timer1 is selected using the GSS bits. The polarity selection for the gate source is controlled by the GPOL bit.

Any of the above mentioned signals can be used to trigger the gate. The output of the CMPx can be synchronized to the Timer1 clock or left asynchronous. For more information refer to the **"Comparator Output Synchronization"** section.

### 23.7.3 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 the figure below for timing details.

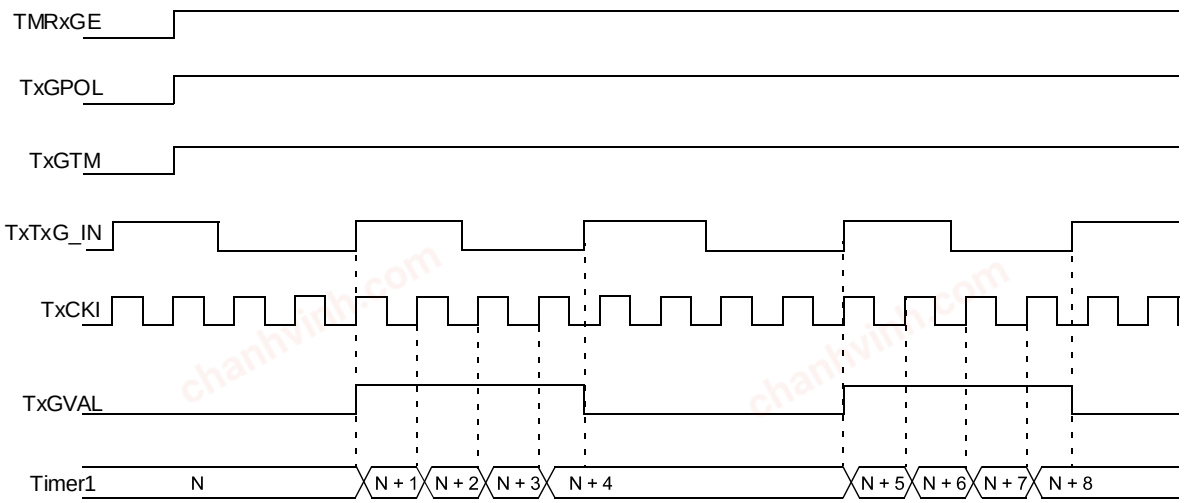
Timer1 Gate Toggle mode is enabled by setting the GTM bit. When the GTM bit is cleared, the flip-flop is cleared and held clear. This is necessary to control which edge is measured.

**Important:** Enabling Toggle mode at the same time as changing the gate polarity may result in indeterminate operation.

chanhvinh.com



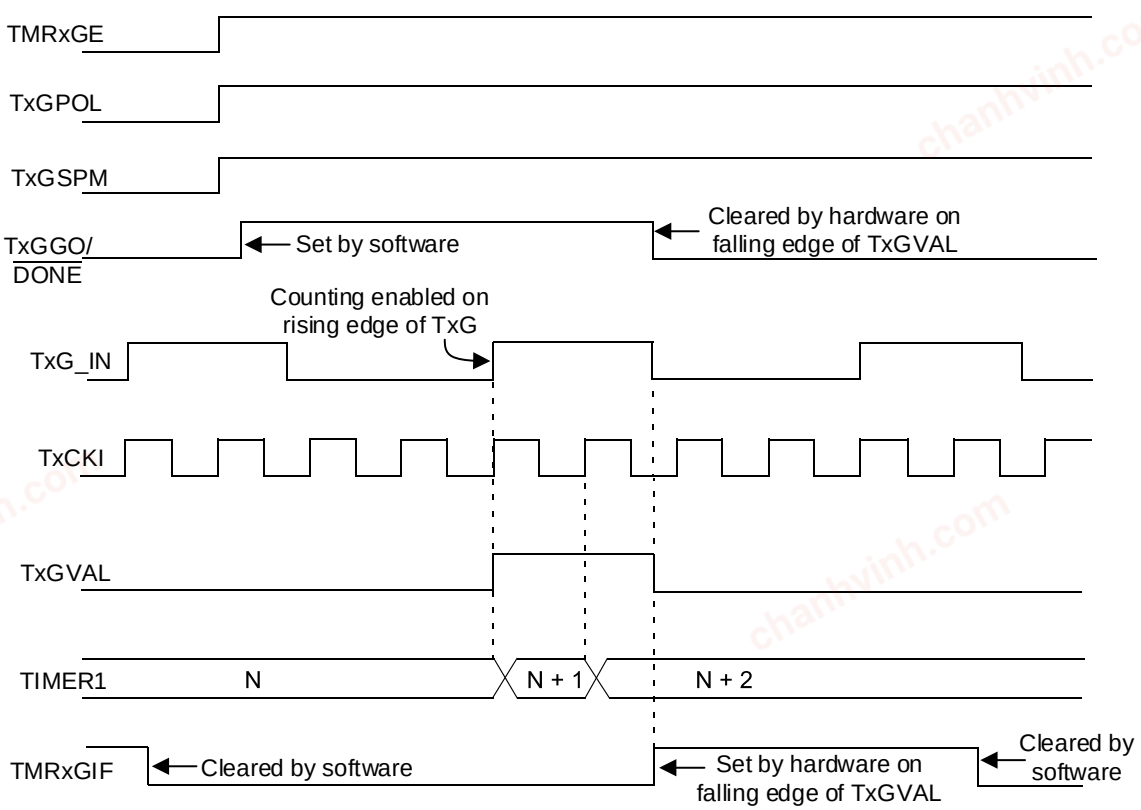
**Figure 23-5. Timer1 Gate Toggle Mode**




### 23.7.4 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 GSPM bit. Next, the GGO/DONE must be set. The Timer1 will be fully enabled on the next incrementing edge. On the next trailing edge of the pulse, the GGO/$\overline{\text{DONE}}$ bit will automatically be cleared. No other gate events will be allowed to increment Timer1 until the GGO/$\overline{\text{DONE}}$ bit is once again set in software.

**Figure 23-6. Timer1 Gate Single Pulse Mode**

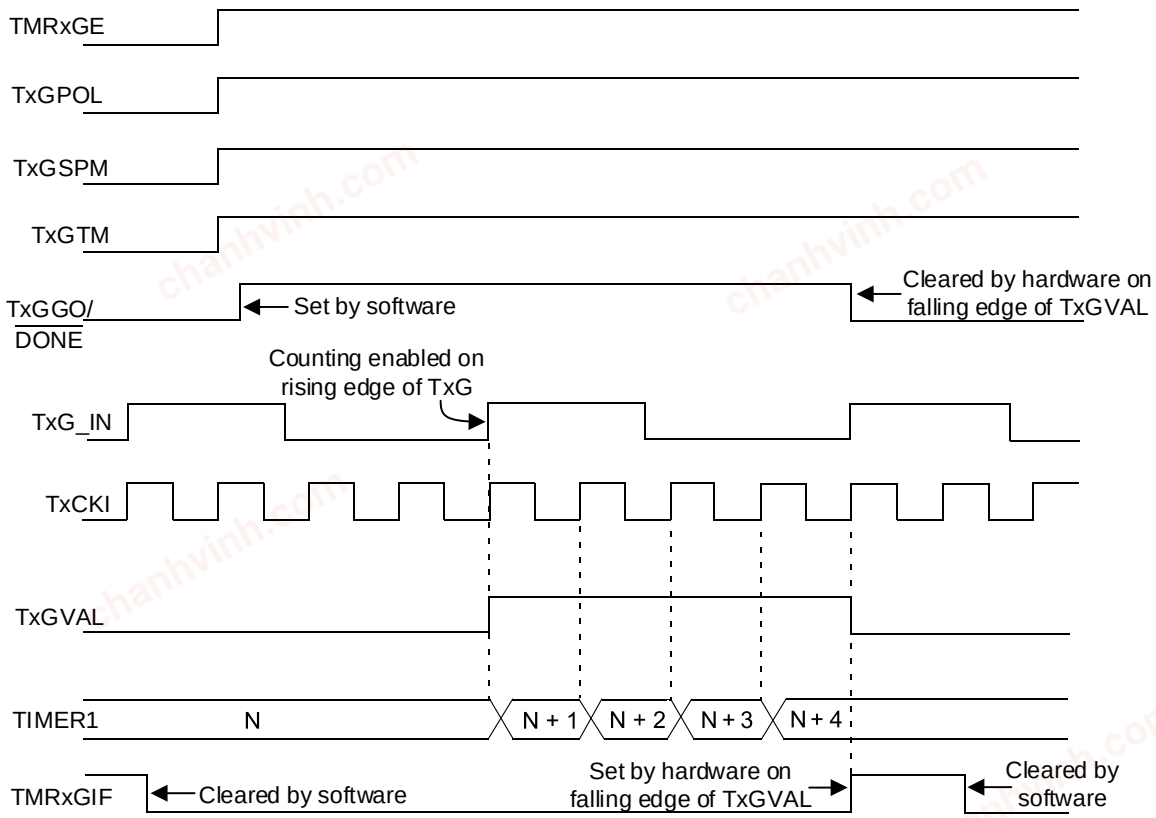


chanhvinh.com



Clearing the GSPM bit will also clear the GGO/$\overline{DONE}$ bit. See the figure below for timing details. 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 the figure below for timing details.

**Figure 23-7. Timer1 Gate Single Pulse and Toggle Combined Mode**




### 23.7.5 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 GVAL bit in the TxGCON register. The GVAL bit is valid even when the Timer1 gate is not enabled (GE bit is cleared).

### 23.7.6 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 GVAL occurs, the TMRxGIF flag bit in one of the PIR registers will be set. If the TMRxGIE bit in the corresponding PIE register is set, then an interrupt will be recognized.

The TMRxGIF flag bit operates even when the Timer1 gate is not enabled (the GE bit is cleared).

## 23.8 Timer1 Interrupt

The TMRx register increments to FFFFh and rolls over to 0000h. When TMRx rolls over, the Timer1 interrupt flag bit of the PIRx register is set. To enable the interrupt-on-rollover, the following bits must be set:

- The ON bit of the TxCON register
- The TMRxIE bits of the PIEx register
- Global interrupts must be enabled

The interrupt is cleared by clearing the TMRxIF bit as a task in the Interrupt Service Routine.

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



**Important:** The TMRx register and the TMRxIF bit must be cleared before enabling interrupts.

## 23.9 Timer1 Operation During Sleep

Timer1 can only operate during Sleep when configured as an asynchronous counter. In this mode, many clock sources can be used to increment the counter. To set up the timer to wake the device:

- The ON bit must be set
- The TMRxIE bit of the PIEx register must be set
- Global interrupts must be enabled
- The SYNC bit must be set
- Configure the TxCLK register for using any clock source other than $F_{OSC}$ and $F_{OSC}/4$

The device will wake up on an overflow and execute the next instruction. If global interrupts are enabled, the device will call the IRS. The secondary oscillator will continue to operate in Sleep regardless of the SYNC bit setting.

## 23.10 CCP Capture/Compare Time Base

The CCP modules use TMRx as the time base when operating in Capture or Compare mode. In Capture mode, the value in TMRx is copied into the CCPRx register on a capture event. In Compare mode, an event is triggered when the value in the CCPRx register matches the value in TMRx. This event can be a Special Event Trigger.

## 23.11 CCP Special Event Trigger

When any of the CCPs are configured to trigger a special event, the trigger will clear the TMRx register. This special event does not cause a Timer1 interrupt. The CCP module may still be configured to generate a CCP interrupt. In this mode of operation, the CCPRx register becomes the period register for Timer1. Timer1 must be synchronized and $F_{OSC}/4$ must be selected as the clock source to utilize the Special Event Trigger. Asynchronous operation of Timer1 can cause a Special Event Trigger to be missed. In the event that a write to TMRxH or TMRxL coincides with a Special Event Trigger from the CCP, the write will take precedence.

## 23.12 Peripheral Module Disable

When a peripheral is not used or inactive, the module can be disabled by setting the Module Disable bit in the PMD registers. This will reduce power consumption to an absolute minimum. Setting the PMD bits holds the module in Reset and disconnects the module's clock source. The Module Disable bits for Timer1 (TMR1MD) are in the PMDx register. See the **"PMD - Peripheral Module Disable"** chapter for more information.

## 23.13 Register Definitions: Timer1 Control

Long bit name prefixes for the Timer registers are shown in the table below, where 'x' refers to the Timer instance number. Refer to the "**Long Bit Names**" section in the "**Register and Bit Naming Conventions**" chapter for more information.

**Table 23-3. Timer1 Register Long Bit Name Prefixes**

| Peripheral | Bit Name Prefix |
|---|---|
| Timer1 | T1 |
| Timer3 | T3 |

chanhvinh.com



### 23.13.1 TxCON

**Name:** TxCON
**Offset:** 0x030E,0x0314

Timer Control Register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | CKPS[1:0] | | | SYNC | RD16 | ON |
| Access | | | R/W | R/W | | R/W | R/W | R/W |
| Reset | | | 0 | 0 | | 0 | 0 | 0 |

**Bits 5:4 – CKPS[1:0]** Timer Input Clock Prescaler Select
Reset States: POR/BOR = 00
All Other Resets = uu

| Value | Description |
|---|---|
| 11 | 1:8 Prescaler value |
| 10 | 1:4 Prescaler value |
| 01 | 1:2 Prescaler value |
| 00 | 1:1 Prescaler value |

**Bit 2 – $\overline{\text{SYNC}}$** Timer External Clock Input Synchronization Control
Reset States: POR/BOR = 0
All Other Resets = u

| Value | Condition | Description |
|---|---|---|
| x | CS = $F_{OSC}/4$ or $F_{OSC}$ | This bit is ignored. Timer uses the incoming clock as is. |
| 1 | All other clock sources | Do not synchronize external clock input |
| 0 | All other clock sources | Synchronize external clock input with system clock |

**Bit 1 – RD16** 16-Bit Read/Write Mode Enable
Reset States: POR/BOR = 0
All Other Resets = u

| Value | Description |
|---|---|
| 1 | Enables register read/write of Timer in one 16-bit operation |
| 0 | Enables register read/write of Timer in two 8-bit operations |

**Bit 0 – ON** Timer On
Reset States: POR/BOR = 0
All Other Resets = u

| Value | Description |
|---|---|
| 1 | Enables Timer |
| 0 | Disables Timer |

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



### 23.13.2 TxGCON

**Name:** TxGCON
**Offset:** 0x030F,0x0315

Timer Gate Control Register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | GE | GPOL | GTM | GSPM | GGO/$\overline{\text{DONE}}$ | GVAL | | |
| Access | R/W | R/W | R/W | R/W | R/W | R | | |
| Reset | 0 | 0 | 0 | 0 | 0 | x | | |

**Bit 7 – GE** Timer Gate Enable
Reset States: POR/BOR = 0
All Other Resets = u

| Value | Condition | Description |
|---|---|---|
| 1 | ON = 1 | Timer counting is controlled by the Timer gate function |
| 0 | ON = 1 | Timer is always counting |
| X | ON = 0 | This bit is ignored |

**Bit 6 – GPOL** Timer Gate Polarity
Reset States: POR/BOR = 0
All Other Resets = u

| Value | Description |
|---|---|
| 1 | Timer gate is active-high (Timer counts when gate is high) |
| 0 | Timer gate is active-low (Timer counts when gate is low) |

**Bit 5 – GTM** Timer Gate Toggle Mode
Timer Gate flip-flop toggles on every rising edge when Toggle mode is enabled.
Reset States: POR/BOR = 0
All Other Resets = u

| Value | Description |
|---|---|
| 1 | Timer Gate Toggle mode is enabled |
| 0 | Timer Gate Toggle mode is disabled and Toggle flip-flop is cleared |

**Bit 4 – GSPM** Timer Gate Single Pulse Mode
Reset States: POR/BOR = 0
All Other Resets = u

| Value | Description |
|---|---|
| 1 | Timer Gate Single Pulse mode is enabled and is controlling Timer gate |
| 0 | Timer Gate Single Pulse mode is disabled |

**Bit 3 – GGO/$\overline{\text{DONE}}$** Timer Gate Single Pulse Acquisition Status
This bit is automatically cleared when TxGSPM is cleared.
Reset States: POR/BOR = 0
All Other Resets = u

| Value | Description |
|---|---|
| 1 | Timer Gate Single Pulse Acquisition is ready, waiting for an edge |
| 0 | Timer Gate Single Pulse Acquisition has completed or has not been started |

**Bit 2 – GVAL** Timer Gate Current State
Indicates the current state of the timer gate that can be provided to TMRxH:TMRxL
Unaffected by the Timer Gate Enable (GE) bit

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



### 23.13.3 TxCLK

**Name:** TxCLK
**Offset:** 0x0311,0x0317

Timer Clock Source Selection Register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | CS[4:0] | | | | |
| Access | | | | R/W | R/W | R/W | R/W | R/W |
| Reset | | | | 0 | 0 | 0 | 0 | 0 |

**Bits 4:0 – CS[4:0]** Timer Clock Source Selection

**Table 23-4. Timer Clock Sources**

| CS | Clock Source | |
|---|---|---|
| | Timer1 | Timer3 |
| 11111-10010 | Reserved | |
| 10001 | CLC4_OUT | |
| 10000 | CLC3_OUT | |
| 01111 | CLC2_OUT | |
| 01110 | CLC1_OUT | |
| 01101 | TMR3_overflow_OUT | Reserved |
| 01100 | Reserved | TMR1_overflow_OUT |
| 01011 | TMR0_overflow_OUT | |
| 01010 | CLKR | |
| 01001 | EXTOSC | |
| 01000 | SOSC | |
| 00111 | MFINTOSC (32 kHz) | |
| 00110 | MFINTOSC (500 kHz) | |
| 00101 | SFINTOSC (1 MHz) | |
| 00100 | LFINTOSC | |
| 00011 | HFINTOSC | |
| 00010 | $F_{OSC}$ | |
| 00001 | $F_{OSC}/4$ | |
| 00000 | Pin selected by T1CKIPPS | Pin selected by T3CKIPPS |

Reset States: POR/BOR = 00000
All Other Resets = uuuuu

chanhvinh.com



### 23.13.4 TxGATE

**Name:** TxGATE
**Offset:** 0x0310,0x0316

Timer Gate Source Selection Register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | GSS[4:0] | | | | |
| Access | | | | R/W | R/W | R/W | R/W | R/W |
| Reset | | | | 0 | 0 | 0 | 0 | 0 |

**Bits 4:0 – GSS[4:0]** Timer Gate Source Selection

**Table 23-5. Timer Gate Sources**

| GSS | Gate Source | |
|---|---|---|
| | Timer1 | Timer3 |
| 11111-11001 | Reserved | |
| 11000 | CLC4_OUT | |
| 10111 | CLC3_OUT | |
| 10110 | CLC2_OUT | |
| 10101 | CLC1_OUT | |
| 10100 | ZCD_OUT | |
| 10011 | C2_OUT | |
| 10010 | C1_OUT | |
| 10001 | NCO1_OUT | |
| 10000 | PWM4S1P2_OUT | |
| 01111 | PWM4S1P1_OUT | |
| 01110 | PWM3S1P2_OUT | |
| 01101 | PWM3S1P1_OUT | |
| 01100 | PWM2S1P2_OUT | |
| 01011 | PWM2S1P1_OUT | |
| 01010 | PWM1S1P2_OUT | |
| 01001 | PWM1S1P1_OUT | |
| 01000 | CCP2_OUT | |
| 00111 | CCP1_OUT | |
| 00110 | TMR6_Postscaled_OUT | |
| 00101 | TMR4_Postscaled_OUT | |
| 00100 | TMR3_overflow_OUT | Reserved |
| 00011 | TMR2_Postscaled_OUT | |
| 00010 | Reserved | TMR1_overflow_OUT |
| 00001 | TMR0_overflow_OUT | |
| 00000 | Pin selected by T1GPPS | Pin selected by T3GPPS |

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



### 23.13.5 TMRx

**Name:** TMRx
**Offset:** 0x030C,0x0312

Timer Register

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | TMRx[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TMRx[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 15:0 – TMRx[15:0]** Timer Register Value
Reset States: POR/BOR = 0000000000000000
All Other Resets = uuuuuuuuuuuuuuuu

**Notes:** The individual bytes in this multibyte register can be accessed with the following register names:

- TMRxH: Accesses the high byte TMRx[15:8]
- TMRxL: Accesses the low byte TMRx[7:0]

© 2023 Microchip Technology Inc.
and its subsidiaries

chanhvinh.com



## 23.14 Register Summary - Timer1

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 ... 0x030B | Reserved | | | | | | | | | |
| 0x030C | TMR1 | 7:0 | TMR1[7:0] | | | | | | | |
| | | 15:8 | TMR1[15:8] | | | | | | | |
| 0x030E | T1CON | 7:0 | | | CKPS[1:0] | | | SYNC | RD16 | ON |
| 0x030F | T1GCON | 7:0 | GE | GPOL | GTM | GSPM | GGO/DONE | GVAL | | |
| 0x0310 | T1GATE | 7:0 | | | | GSS[4:0] | | | | |
| 0x0311 | T1CLK | 7:0 | | | | CS[4:0] | | | | |
| 0x0312 | TMR3 | 7:0 | TMR3[7:0] | | | | | | | |
| | | 15:8 | TMR3[15:8] | | | | | | | |
| 0x0314 | T3CON | 7:0 | | | CKPS[1:0] | | | SYNC | RD16 | ON |
| 0x0315 | T3GCON | 7:0 | GE | GPOL | GTM | GSPM | GGO/DONE | GVAL | | |
| 0x0316 | T3GATE | 7:0 | | | | GSS[4:0] | | | | |
| 0x0317 | T3CLK | 7:0 | | | | CS[4:0] | | | | |

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



# 24. TMR2 - Timer2 Module

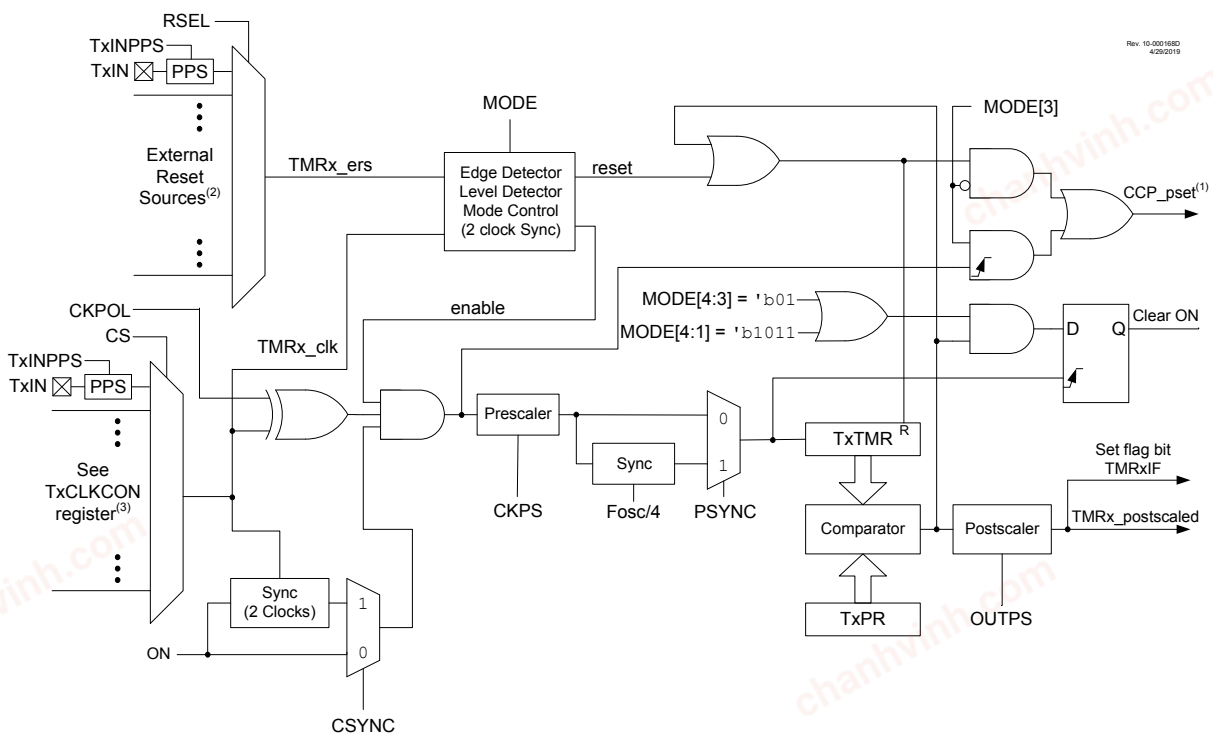
The Timer2 module is an 8-bit timer that incorporates the following features:

- 8-bit timer and period registers
- Readable and writable
- Software programmable prescaler (1:1 to 1:128)
- Software programmable postscaler (1:1 to 1:16)
- Interrupt on T2TMR match with T2PR
- One-shot operation
- Full asynchronous operation
- Includes Hardware Limit Timer (HLT)
- Alternate clock sources
- External timer Reset signal sources
- Configurable timer Reset operation

See figure below for a block diagram of Timer2.

**Important:** References to module Timer2 apply to all the even numbered timers on this device (Timer2, Timer4, etc.).

**Figure 24-1. Timer2 with Hardware Limit Timer (HLT) Block Diagram**




**Notes:**

1. Signal to the CCP peripheral for PWM pulse trigger in PWM mode.
2. See RSEL for external Reset sources.
3. See CS for clock source selections.

chanhvinh.com



## 24.1 Timer2 Operation

Timer2 operates in three major modes:

- Free-Running Period
- One Shot
- Monostable

Within each operating mode, there are several options for starting, stopping and Reset. Table 24-1 lists the options.

In all modes, the T2TMR count register increments on the rising edge of the clock signal from the programmable prescaler. When T2TMR equals T2PR, a high level output to the postscaler counter is generated. T2TMR is cleared on the next clock input.

An external signal from hardware can also be configured to gate the timer operation or force a T2TMR count Reset. In Gate modes, the counter stops when the gate is disabled and resumes when the gate is enabled. In Reset modes, the T2TMR count is reset on either the level or edge from the external source.

The T2TMR and T2PR registers are both directly readable and writable. The T2TMR register is cleared and the T2PR register initializes to 0xFF on any device Reset. Both the prescaler and postscaler counters are cleared on the following events:

- A write to the T2TMR register
- A write to the T2CON register
- Any device Reset
- External Reset source event that resets the timer

**Important:** T2TMR is not cleared when T2CON is written.

### 24.1.1 Free-Running Period Mode

The value of T2TMR is compared to that of the period register, T2PR, on each clock cycle. When the two values match, the comparator resets the value of T2TMR to 0x00 on the next cycle and increments the output postscaler counter. When the postscaler count equals the value in the OUTPS bits of the T2CON register then a one clock period wide pulse occurs on the TMR2_postscaled output, and the postscaler count is cleared.

### 24.1.2 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 T2TMR matches T2PR and will not restart until the ON bit is cycled off and on. Postscaler (OUTPS) values other than zero are ignored in this mode because the timer is stopped at the first period event and the postscaler is reset when the timer is restarted.

### 24.1.3 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.

## 24.2 Timer2 Output

The Timer2 module's primary output is TMR2_postscaled, which pulses for a single TMR2_clk period upon each match of the postscaler counter and the OUTPS bits of the T2CON register. The postscaler is incremented each time the T2TMR value matches the T2PR value. This signal can also be selected as an input to other Core Independent Peripherals.

In addition, the Timer2 is also used by the CCP module for pulse generation in PWM mode. See the "**PWM Overview**" and "**PWM Period**" sections in the **"CCP - Capture/Compare/PWM Module"** chapter for more details on setting up Timer2 for use with the CCP and PWM modules.

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



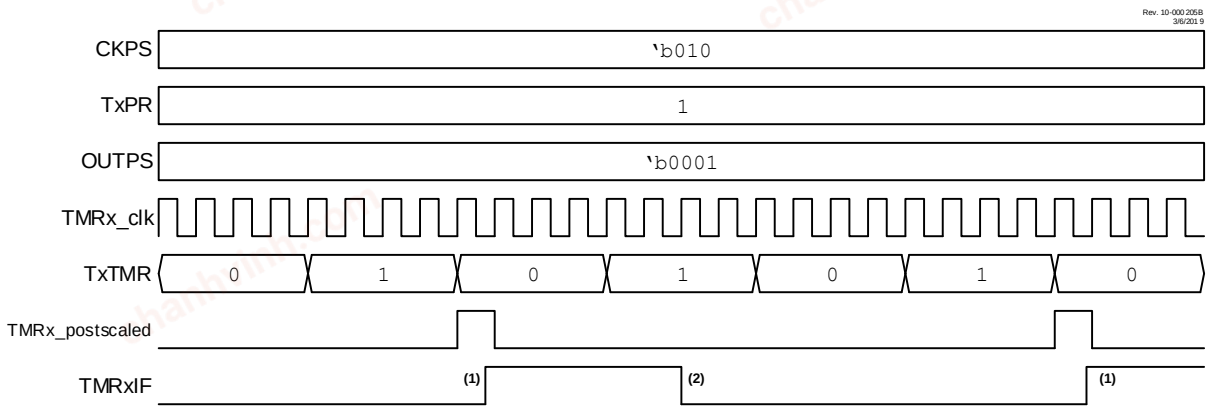
## 24.3 External Reset Sources

In addition to the clock source, the Timer2 can also be driven by an external Reset source input. This external Reset input is selected for each timer with the corresponding TxRST register. The external Reset input can control starting and stopping of the timer, as well as resetting the timer, depending on the mode used.

## 24.4 Timer2 Interrupt

Timer2 can also generate a device interrupt. The interrupt is generated when the postscaler counter matches the selected postscaler value (OUTPS bits of T2CON register). The interrupt is enabled by setting the TMR2IE interrupt enable bit. Interrupt timing is illustrated in the figure below.

**Figure 24-2. Timer2 Prescaler, Postscaler, and Interrupt Timing Diagram**




**Notes:** 1. Setting the interrupt flag is synchronized with the instruction clock. Synchronization may take as many as two instruction cycles.
2. Cleared by software.

## 24.5 PSYNC Bit

Setting the PSYNC bit synchronizes the prescaler output to $F_{OSC}/4$. Setting this bit is required for reading the Timer2 counter register while the selected Timer clock is asynchronous to $F_{OSC}/4$.

**Note:** Setting PSYNC requires that the output of the prescaler is slower than $F_{OSC}/4$. Setting PSYNC when the output of the prescaler is greater than or equal to $F_{OSC}/4$ may cause unexpected results.

## 24.6 CSYNC Bit

All bits in the Timer2 SFRs are synchronized to $F_{OSC}/4$ by default, not the Timer2 input clock. As such, if the Timer2 input clock is not synchronized to $F_{OSC}/4$, it is possible for the Timer2 input clock to transition at the same time as the ON bit is set in software, which may cause undesirable behavior and glitches in the counter. Setting the CSYNC bit remedies this problem by synchronizing the ON bit to the Timer2 input clock instead of $F_{OSC}/4$. However, as this synchronization uses an edge of the TMR2 input clock, up to one input clock cycle will be consumed and not counted by the Timer2 when CSYNC is set. Conversely, clearing the CSYNC bit synchronizes the ON bit to $F_{OSC}/4$, which does not consume any clock edges, but has the previously stated risk of glitches.

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



## 24.7 Operating Modes

The mode of the timer is controlled by the MODE bits. 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 mode.

**Table 24-1. Operating Modes Table**

| Mode | MODE | | Output Operation | Operation | Timer Control | | |
|---|---|---|---|---|---|---|---|
| | [4:3] | [2:0] | | | Start | Reset | Stop |
| Free-Running Period | 00 | 000 | Period Pulse | Software gate (Figure 24-3) | ON = 1 | — | ON = 0 |
| | | 001 | | Hardware gate, active-high (Figure 24-4) | 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 24-5) | | 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 24-6) | | TMRx_ers = 1 | ON = 0 or TMRx_ers = 1 |
| One Shot | 01 | 000 | One-shot | Software start (Figure 24-7) | ON = 1 | — | ON = 0 or Next clock after TxTMR = TxPR (Note 2) |
| | | 001 | Edge-Triggered Start (Note 1) | Rising edge start (Figure 24-8) | 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 24-9) | 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 24-10) | ON = 1 and TMRx_ers ↑ | TMRx_ers = 0 | |
| | | 111 | | Falling edge start and High-level Reset | ON = 1 and TMRx_ers ↓ | TMRx_ers = 1 | |

chanhvinh.com



.....continued

| Mode | MODE | | Output Operation | Operation | Timer Control | | |
|---|---|---|---|---|---|---|---|
| | [4:3] | [2:0] | | | Start | Reset | Stop |
| Monostable | 10 | 000 | Reserved | | | | |
| | | 001 | Edge-Triggered Start (Note 1) | Rising edge start (Figure 24-11) | ON = 1 and TMRx_ers ↑ | — | ON = 0 or Next clock after TxTMR = TxPR (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 24-12) | ON = 1 and TMRx_ers = 1 | TMRx_ers = 0 | ON = 0 or Held in Reset (Note 2) |
| | | 111 | | Low-level start and High-level Reset | ON = 1 and TMRx_ers = 0 | TMRx_ers = 1 | |
| Reserved | 11 | xxx | Reserved | | | | |

**Notes:**

1. If ON = 0, then an edge is required to restart the timer after ON = 1.
2. When T2TMR = T2PR, the next clock clears ON and stops T2TMR at 00h.
3. When T2TMR = T2PR, the next clock stops T2TMR at 00h but does not clear ON.

## 24.8 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).
- The diagrams illustrate any clock except $F_{OSC}/4$ and show clock-sync delays of at least two full cycles for both ON and TMRx_ers. When using $F_{OSC}/4$, the clock-sync delay is at least one instruction period for TMRx_ers; ON applies in the next instruction period.
- ON and TMRx_ers are somewhat generalized, and clock-sync delays may produce results that are slightly different than illustrated.
- 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 the "**PWM Overview**" section. The signals are not a part of the Timer2 module.

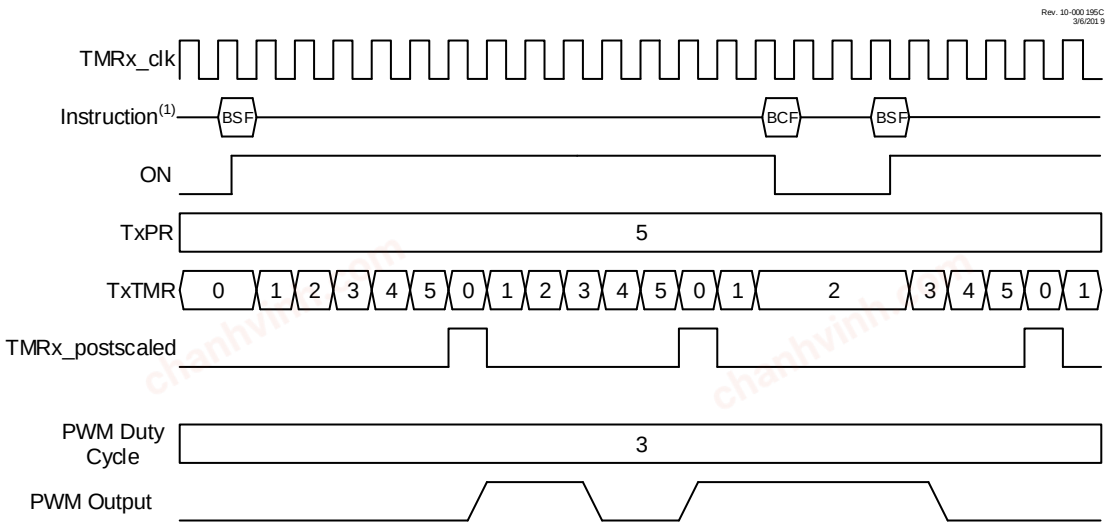
### 24.8.1 Software Gate Mode

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 TxTMR count equals the TxPR period count, the timer resets on the next clock and continues counting from zero. Operation with the ON bit software controlled is illustrated in Figure 24-3. With TxPR = 5, the counter advances until TxTMR = 5, and goes to zero with the next clock.

chanhvinh.com



**Figure 24-3. Software Gate Mode Timing Diagram (MODE = `'b00000`)**




**Note:** 1. `BSF` and `BCF` represent Bit-Set File and Bit-Clear File instructions executed by the CPU to set or clear the ON bit of TxCON. CPU execution is asynchronous to the timer clock input.

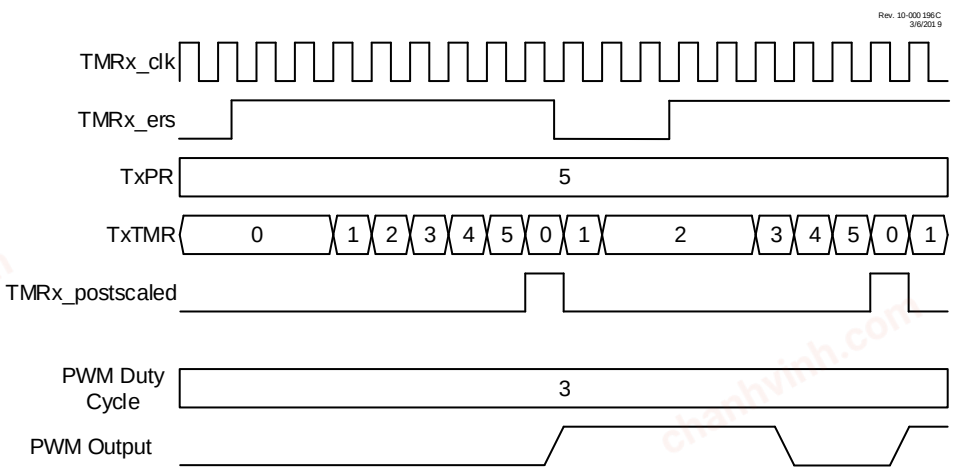
### 24.8.2 Hardware Gate Mode

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 = `'b00001`, then the timer is stopped when the external signal is high. When MODE = `'b00010`, then the timer is stopped when the external signal is low.

Figure 24-4 illustrates the Hardware Gating mode for MODE = `'b00001` in which a high input level starts the counter.

**Figure 24-4. Hardware Gate Mode Timing Diagram (MODE = `'b00001`)**




### 24.8.3 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:

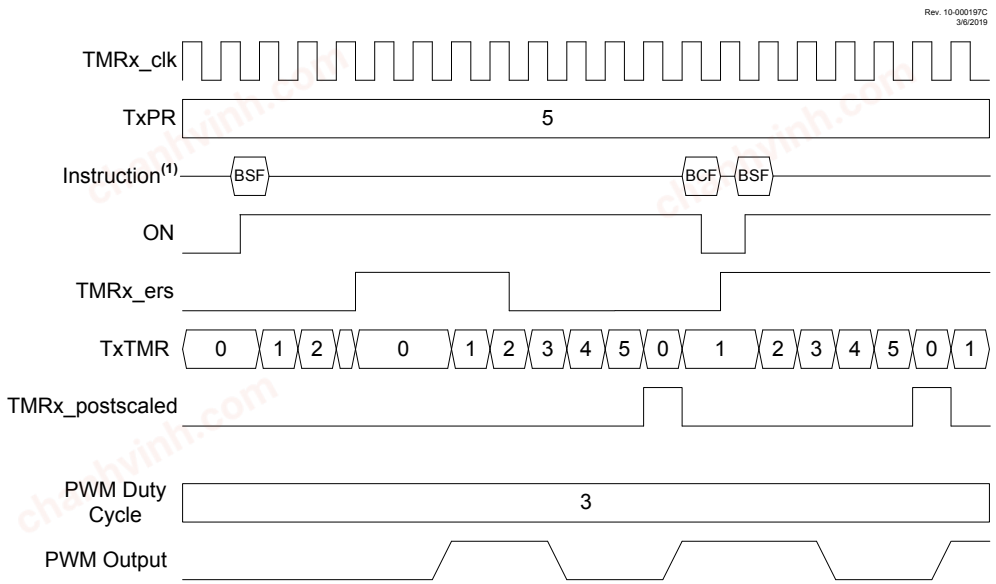
chanhvinh.com



- Reset on rising or falling edge (MODE = `b00011)
- Reset on rising edge (MODE = `b00100)
- Reset on falling edge (MODE = `b00101)

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 24-5.

**Figure 24-5. Edge Triggered Hardware Limit Mode Timing Diagram (MODE = `b00100)**




**Note:** 1. `BSF and `BCF represent Bit-Set File and Bit-Clear File instructions executed by the CPU to set or clear the ON bit of TxCON. CPU execution is asynchronous to the timer clock input.

chanhvinh.com



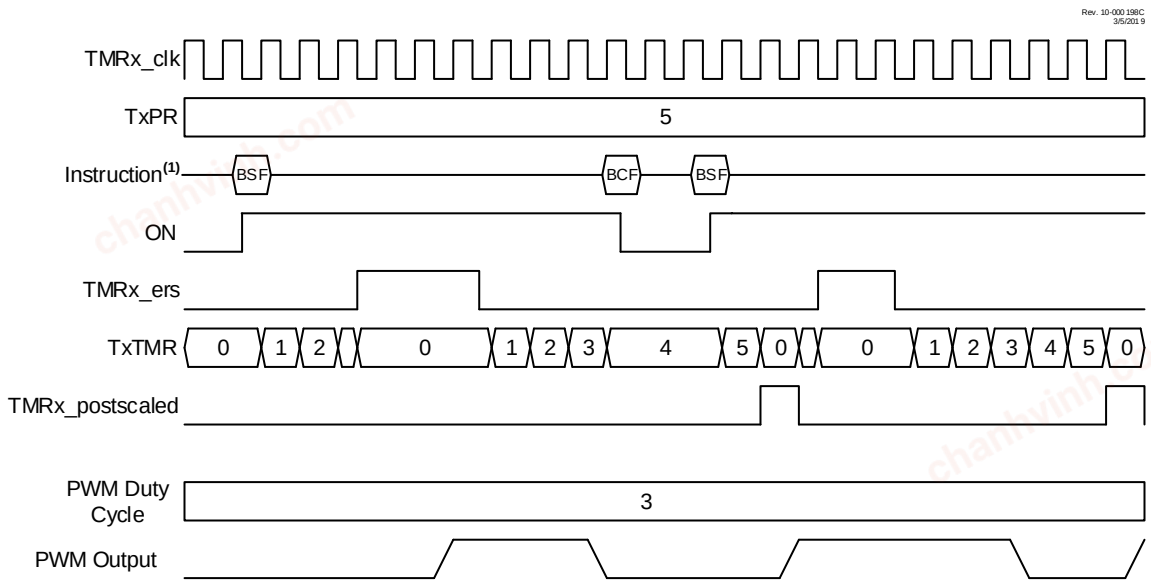
### 24.8.4 Level Triggered Hardware Limit Mode

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 24-6. Selecting MODE = `'b00110` will cause the timer to reset on a low-level external signal. Selecting MODE = `'b00111` 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 TxPR 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 TxPR 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 24-6. Level Triggered Hardware Limit Mode Timing Diagram (MODE = `'b00111`)**




**Note:** 1. `BSF` and `BCF` represent Bit-Set File and Bit-Clear File instructions executed by the CPU to set or clear the ON bit of TxCON. CPU execution is asynchronous to the timer clock input.

chanhvinh.com

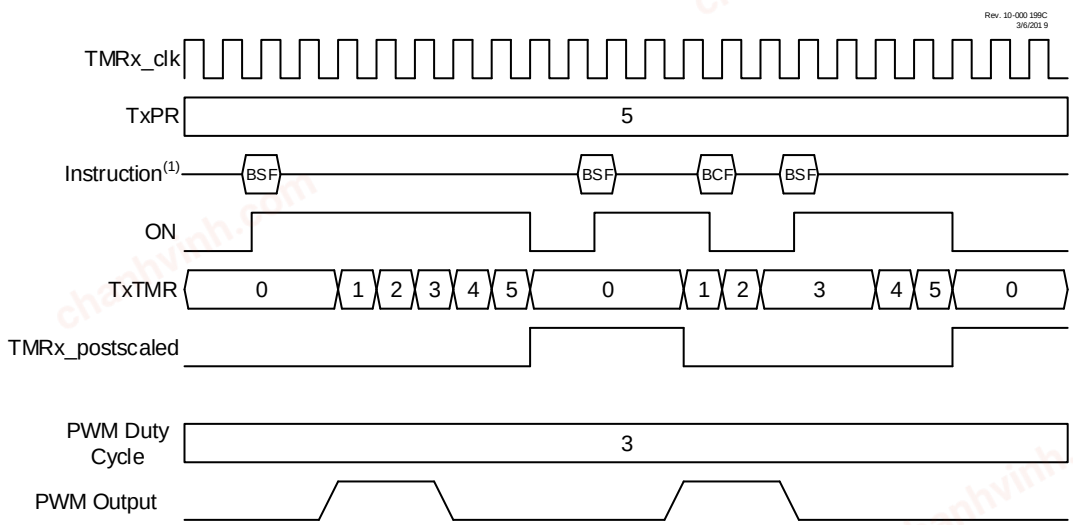


### 24.8.5 Software Start One Shot Mode

In One Shot mode, the timer resets and the ON bit is cleared when the timer value matches the TxPR period value. The ON bit must be set by software to start another timer cycle. Setting MODE = `'b01000` selects One Shot mode which is illustrated in Figure 24-7. 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, the `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 the software sets the ON bit to start another cycle. If the software clears the ON bit after the CCPRx match but before the TxPR 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 TxPR period count match.

**Figure 24-7. Software Start One Shot Mode Timing Diagram (MODE = `'b01000`)**




**Note:** 1. `BSF` and `BCF` represent Bit-Set File and Bit-Clear File instructions executed by the CPU to set or clear the ON bit of TxCON. CPU execution is asynchronous to the timer clock input.

chanhvinh.com



### 24.8.6 Edge Triggered One Shot Mode

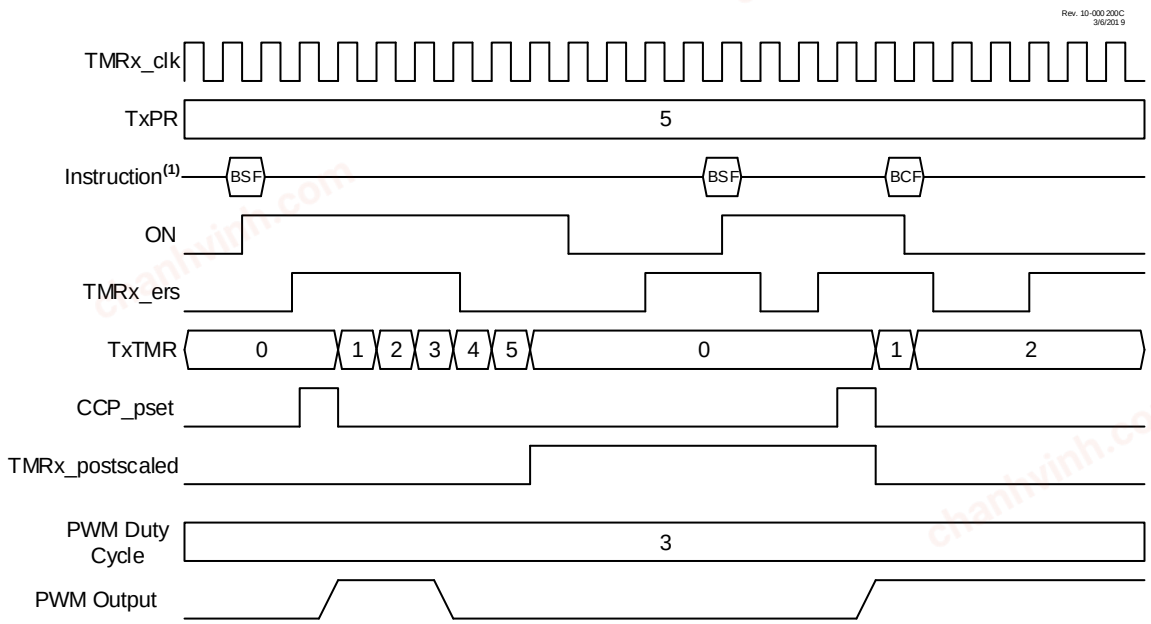
The Edge Triggered One Shot modes start the timer on an edge from the external signal input, after the ON bit is set, and clear the ON bit when the timer matches the TxPR period value. The following edges will start the timer:

- Rising edge (MODE = `'b01001`)
- Falling edge (MODE = `'b01010`)
- Rising or Falling edge (MODE = `'b01011`)

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 24-8 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 TxPR period count match.

**Figure 24-8. Edge Triggered One Shot Mode Timing Diagram (MODE = `'b01001`)**




**Note:** 1. `BSF` and `BCF` represent Bit-Set File and Bit-Clear File instructions executed by the CPU to set or clear the ON bit of TxCON. CPU execution is asynchronous to the timer clock input.

chanhvinh.com



### 24.8.7 Edge Triggered Hardware Limit One Shot Mode

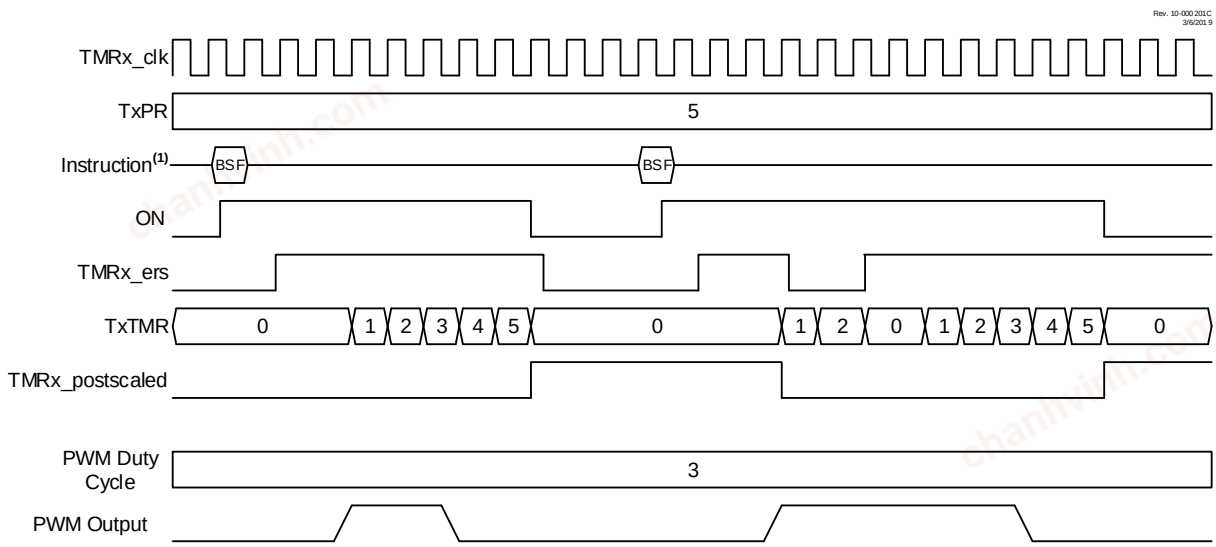
In Edge Triggered Hardware Limit One Shot modes, the timer starts on the first external signal edge after the ON bit is set and resets on all subsequent edges. Only the first edge after the ON bit is set is needed to start the timer. The counter will resume counting automatically two clocks after all subsequent external Reset edges. Edge triggers are as follows:

- Rising edge start and Reset (MODE = `'b01100`)
- Falling edge start and Reset (MODE = `'b01101`)

The timer resets and clears the ON bit when the timer value matches the TxPR period value. External signal edges will have no effect until after software sets the ON bit. Figure 24-9 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 TxPR period match unless an external signal edge resets the timer before the match occurs.

**Figure 24-9. Edge Triggered Hardware Limit One Shot Mode Timing Diagram (MODE = `'b01100`)**




**Note:** 1. `BSF` and `BCF` represent Bit-Set File and Bit-Clear File instructions executed by the CPU to set or clear the ON bit of TxCON. CPU execution is asynchronous to the timer clock input.

chanhvinh.com



### 24.8.8 Level Reset, Edge Triggered Hardware Limit One Shot Modes

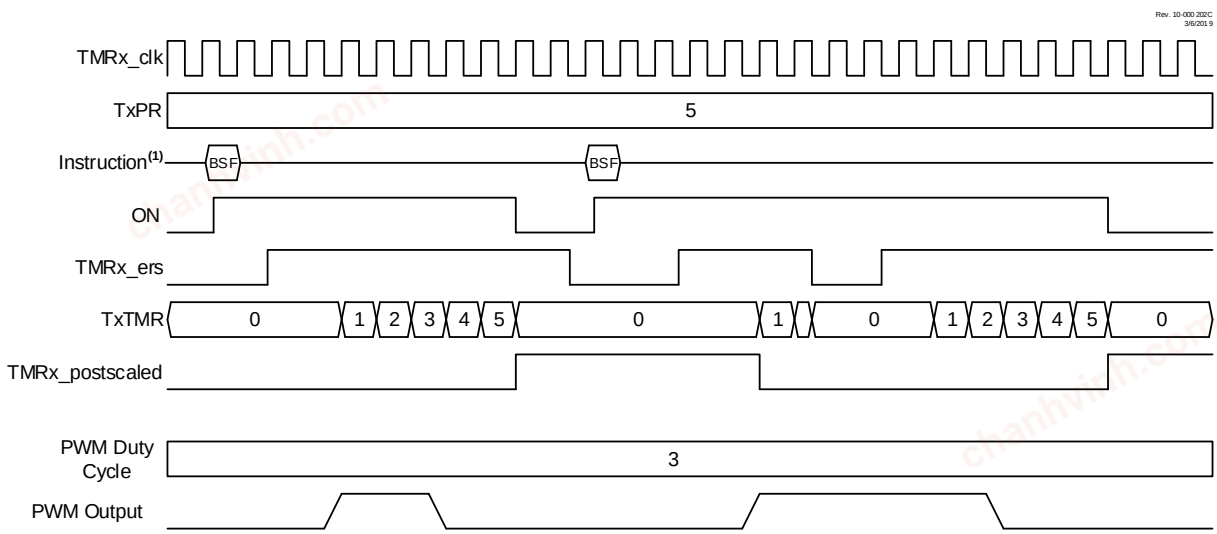
In Level Triggered One Shot mode, the timer count is reset on the external signal level and starts counting on the rising/falling edge of the transition from Reset level to the active level while the ON bit is set. Reset levels are selected as follows:

- Low Reset level (MODE = `'b01110`)
- High Reset level (MODE = `'b01111`)

When the timer count matches the TxPR period count, the timer is reset and the ON bit is cleared. When the ON bit is cleared by either a TxPR 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 TxPR period count match.

**Figure 24-10. Low Level Reset, Edge Triggered Hardware Limit One Shot Mode Timing Diagram (MODE = `'b01110`)**




**Note:** 1. `BSF` and `BCF` represent Bit-Set File and Bit-Clear File instructions executed by the CPU to set or clear the ON bit of TxCON. CPU execution is asynchronous to the timer clock input.

### 24.8.9 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 TxPR period value. The following edges will start the timer:

- Rising edge (MODE = `'b10001`)
- Falling edge (MODE = `'b10010`)
- Rising or Falling edge (MODE = `'b10011`)

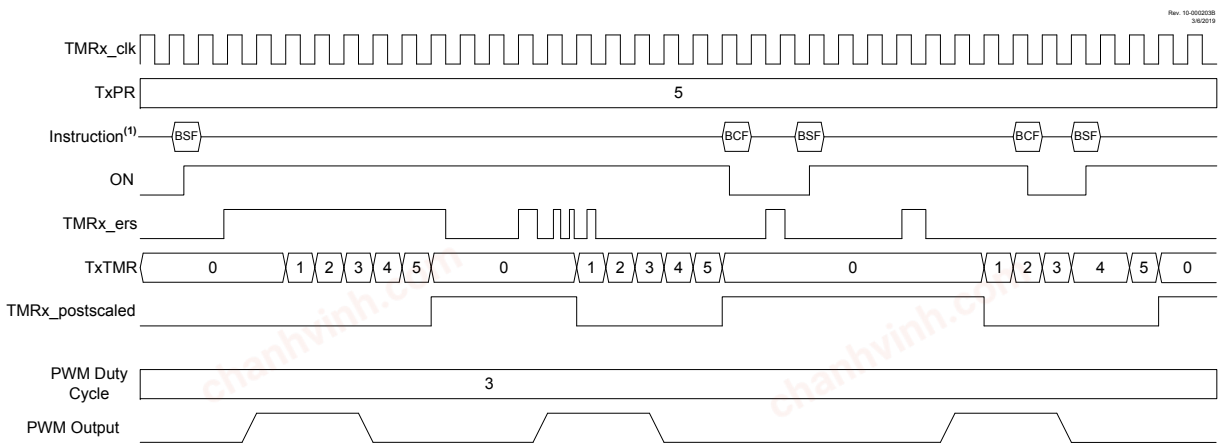
When an Edge Triggered Monostable mode is used in conjunction with the CCP PWM operation, the PWM drive goes active with the external Reset signal edge that starts the timer, but will not go active when the timer matches the TxPR value. While the timer is incrementing, additional edges on the external Reset signal will not affect the CCP PWM.

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



**Figure 24-11. Rising Edge Triggered Monostable Mode Timing Diagram (MODE = 'b10001)**




**Note:** 1. BSF and BCF represent Bit-Set File and Bit-Clear File instructions executed by the CPU to set or clear the ON bit of TxCON. CPU execution is asynchronous to the timer clock input.

### 24.8.10 Level Triggered Hardware Limit One Shot Modes

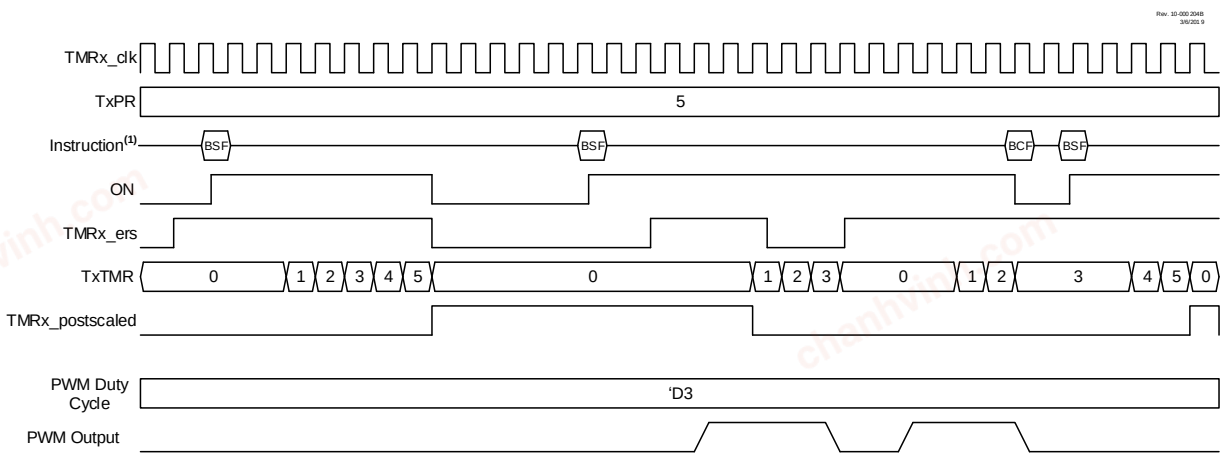
The Level Triggered Hardware Limit One Shot modes hold the timer in Reset on an external Reset level and start counting when both the ON bit is set and the external signal is not at the Reset level. If one of either the external signal is not in Reset or the ON bit is set, then the other signal being set/made active will start the timer. Reset levels are selected as follows:

- Low Reset level (MODE = 'b10110)
- High Reset level (MODE = 'b10111)

When the timer count matches the TxPR period count, the timer is reset and the ON bit is cleared. When the ON bit is cleared by either a TxPR 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 24-12. Level Triggered Hardware Limit One Shot Mode Timing Diagram (MODE = 'b10110)**




**Note:** 1. BSF and BCF represent Bit-Set File and Bit-Clear File instructions executed by the CPU to set or clear the ON bit of TxCON. CPU execution is asynchronous to the timer clock input.

chanhvinh.com



## 24.9 Timer2 Operation During Sleep

When PSYNC = 1, Timer2 cannot be operated while the processor is in Sleep mode. The contents of the T2TMR and T2PR registers will remain unchanged while the processor is in Sleep mode.

When PSYNC = 0, Timer2 will operate in Sleep as long as the clock source selected is also still running. If any internal oscillator is selected as the clock source, it will stay active during Sleep mode.

## 24.10 Register Definitions: Timer2 Control

Long bit name prefixes for the Timer2 peripherals are shown in the table below. Refer to the **"Long Bit Names"** section of the **"Register and Bit Naming Conventions"** chapter for more information.

**Table 24-2. Timer2 Long Bit Name Prefixes**

| Peripheral | Bit Name Prefix |
|---|---|
| Timer2 | T2 |
| Timer4 | T4 |
| Timer6 | T6 |

**Important:** References to module Timer2 apply to all the even numbered timers on this device (Timer2, Timer4, etc.).

chanhvinh.com



### 24.10.1 TxTMR

**Name:** TxTMR
**Offset:** 0x038C,0x0392,0x0398

Timer Counter Register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TxTMR[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 7:0 – TxTMR[7:0]** Timerx Counter

chanhvinh.com

chanhvinh.com



### 24.10.2 TxPR

**Name:** TxPR
**Offset:** 0x038D,0x0393,0x0399

Timer Period Register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TxPR[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

**Bits 7:0 – TxPR[7:0]** Timer Period Register

| Value | Description |
|---|---|
| 0 to 255 | The timer restarts at '0' when TxTMR reaches the TxPR value |

chanhvinh.com

chanhvinh.com



### 24.10.3 TxCON

**Name:** TxCON
**Offset:** 0x038E,0x0394,0x039A

Timerx Control Register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ON | CKPS[2:0] | | | OUTPS[3:0] | | | |
| Access | R/W/HC | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 7 – ON** Timer On(1)

| Value | Description |
|---|---|
| 1 | Timer is on |
| 0 | Timer is off: All counters and state machines are reset |

**Bits 6:4 – CKPS[2:0]** Timer Clock Prescale Select

| Value | Description |
|---|---|
| 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 |

**Bits 3:0 – OUTPS[3:0]** Timer Output Postscaler Select

| Value | Description |
|---|---|
| 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 Table 24-1.

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



### 24.10.4 TxHLT

**Name:** TxHLT
**Offset:** 0x038F,0x0395,0x039B

Timer Hardware Limit Control Register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PSYNC | CPOL | CSYNC | MODE[4:0] | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 7 – PSYNC** Timer Prescaler Synchronization Enable(1, 2)

| Value | Description |
|---|---|
| 1 | Timer Prescaler Output is synchronized to $F_{OSC}/4$ |
| 0 | Timer Prescaler Output is not synchronized to $F_{OSC}/4$ |

**Bit 6 – CPOL** Timer Clock Polarity Selection(3)

| Value | Description |
|---|---|
| 1 | Falling edge of input clock clocks timer/prescaler |
| 0 | Rising edge of input clock clocks timer/prescaler |

**Bit 5 – CSYNC** Timer Clock Synchronization Enable(4, 5)

| Value | Description |
|---|---|
| 1 | ON bit is synchronized to timer clock input |
| 0 | ON bit is not synchronized to timer clock input |

**Bits 4:0 – MODE[4:0]** Timer Control Mode Selection(6, 7)

| Value | Description |
|---|---|
| 00000 to 11111 | See Table 24-1 |

**Notes:**

1. Setting this bit ensures that reading TxTMR will return a valid data value.
2. When this bit is '1', the Timer cannot operate in Sleep mode.
3. CKPOL must not be changed while ON = 1.
4. Setting this bit ensures glitch-free operation when the ON is enabled or disabled.
5. When this bit is set, then the timer operation will be delayed by two input clocks after the ON bit is set.
6. Unless otherwise indicated, all modes start upon ON = 1 and stop upon ON = 0 (stops occur without affecting the value of TxTMR).
7. When TxTMR = TxPR, the next clock clears TxTMR, regardless of the operating mode.

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



### 24.10.5 TxCLKCON

**Name:** TxCLKCON
**Offset:** 0x0390,0x0396,0x039C

Timer Clock Source Selection Register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | CS[3:0] | | | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

**Bits 3:0 – CS[3:0]** Timer Clock Source Selection

**Table 24-3. Clock Source Selection**

| CS | Clock Source | | |
|---|---|---|---|
| | Timer2 | Timer4 | Timer6 |
| 1111 | CLC4_OUT | | |
| 1110 | CLC3_OUT | | |
| 1101 | CLC2_OUT | | |
| 1100 | CLC1_OUT | | |
| 1011 | ZCD_OUT | | |
| 1010 | NCO1_OUT | | |
| 1001 | CLKR_OUT | | |
| 1000 | EXTOSC | | |
| 0111 | SOSC | | |
| 0110 | MFINTOSC (32 kHz) | | |
| 0101 | MFINTOSC (500 kHz) | | |
| 0100 | LFINTOSC | | |
| 0011 | HFINTOSC | | |
| 0010 | $F_{OSC}$ | | |
| 0001 | $F_{OSC}/4$ | | |
| 0000 | Pin selected by T2INPPS | Pin selected by T4INPPS | Pin selected by T6INPPS |

© 2023 Microchip Technology Inc.
and its subsidiaries

chanhvinh.com



### 24.10.6 TxRST

**Name:** TxRST
**Offset:** 0x0391,0x0397,0x039D

Timer External Reset Signal Selection Register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | RSEL[4:0] | | | | |
| Access | | | | R/W | R/W | R/W | R/W | R/W |
| Reset | | | | 0 | 0 | 0 | 0 | 0 |

**Bits 4:0 – RSEL[4:0]** External Reset Source Selection

**Table 24-4. External Reset Sources**

| RSEL | Reset Source | | |
|---|---|---|---|
| | TMR2 | TMR4 | TMR6 |
| 11111-10101 | Reserved | | |
| 10100 | CLC4_OUT | | |
| 10011 | CLC3_OUT | | |
| 10010 | CLC2_OUT | | |
| 10001 | CLC1_OUT | | |
| 10000 | ZCD_OUT | | |
| 01111 | C2_OUT | | |
| 01110 | C1_OUT | | |
| 01101 | PWM4S1P2_OUT | | |
| 01100 | PWM4S1P1_OUT | | |
| 01011 | PWM3S1P2_OUT | | |
| 01010 | PWM3S1P1_OUT | | |
| 01001 | PWM2S1P2_OUT | | |
| 01000 | PWM2S1P1_OUT | | |
| 00111 | PWM1S1P2_OUT | | |
| 00110 | PWM1S1P1_OUT | | |
| 00101 | CCP2_OUT | | |
| 00100 | CCP1_OUT | | |
| 00011 | TMR6_Postscaled_OUT | | Reserved |
| 00010 | TMR4_Postscaled_OUT | Reserved | TMR4_Postscaled_OUT |
| 00001 | Reserved | TMR2_Postscaled_OUT | |
| 00000 | Pin selected by T2INPPS | Pin selected by T4INPPS | Pin selected by T6INPPS |

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



## 24.11 Register Summary - Timer2

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 ... 0x038B | Reserved | | | | | | | | | |
| 0x038C | T2TMR | 7:0 | T2TMR[7:0] | | | | | | | |
| 0x038D | T2PR | 7:0 | T2PR[7:0] | | | | | | | |
| 0x038E | T2CON | 7:0 | ON | | CKPS[2:0] | | | OUTPS[3:0] | | |
| 0x038F | T2HLT | 7:0 | PSYNC | CPOL | CSYNC | | MODE[4:0] | | | |
| 0x0390 | T2CLKCON | 7:0 | | | | | | CS[3:0] | | |
| 0x0391 | T2RST | 7:0 | | | | | RSEL[4:0] | | | |
| 0x0392 | T4TMR | 7:0 | T4TMR[7:0] | | | | | | | |
| 0x0393 | T4PR | 7:0 | T4PR[7:0] | | | | | | | |
| 0x0394 | T4CON | 7:0 | ON | | CKPS[2:0] | | | OUTPS[3:0] | | |
| 0x0395 | T4HLT | 7:0 | PSYNC | CPOL | CSYNC | | MODE[4:0] | | | |
| 0x0396 | T4CLKCON | 7:0 | | | | | | CS[3:0] | | |
| 0x0397 | T4RST | 7:0 | | | | | RSEL[4:0] | | | |
| 0x0398 | T6TMR | 7:0 | T6TMR[7:0] | | | | | | | |
| 0x0399 | T6PR | 7:0 | T6PR[7:0] | | | | | | | |
| 0x039A | T6CON | 7:0 | ON | | CKPS[2:0] | | | OUTPS[3:0] | | |
| 0x039B | T6HLT | 7:0 | PSYNC | CPOL | CSYNC | | MODE[4:0] | | | |
| 0x039C | T6CLKCON | 7:0 | | | | | | CS[3:0] | | |
| 0x039D | T6RST | 7:0 | | | | | RSEL[4:0] | | | |

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



# 25. NCO - Numerically Controlled Oscillator Module

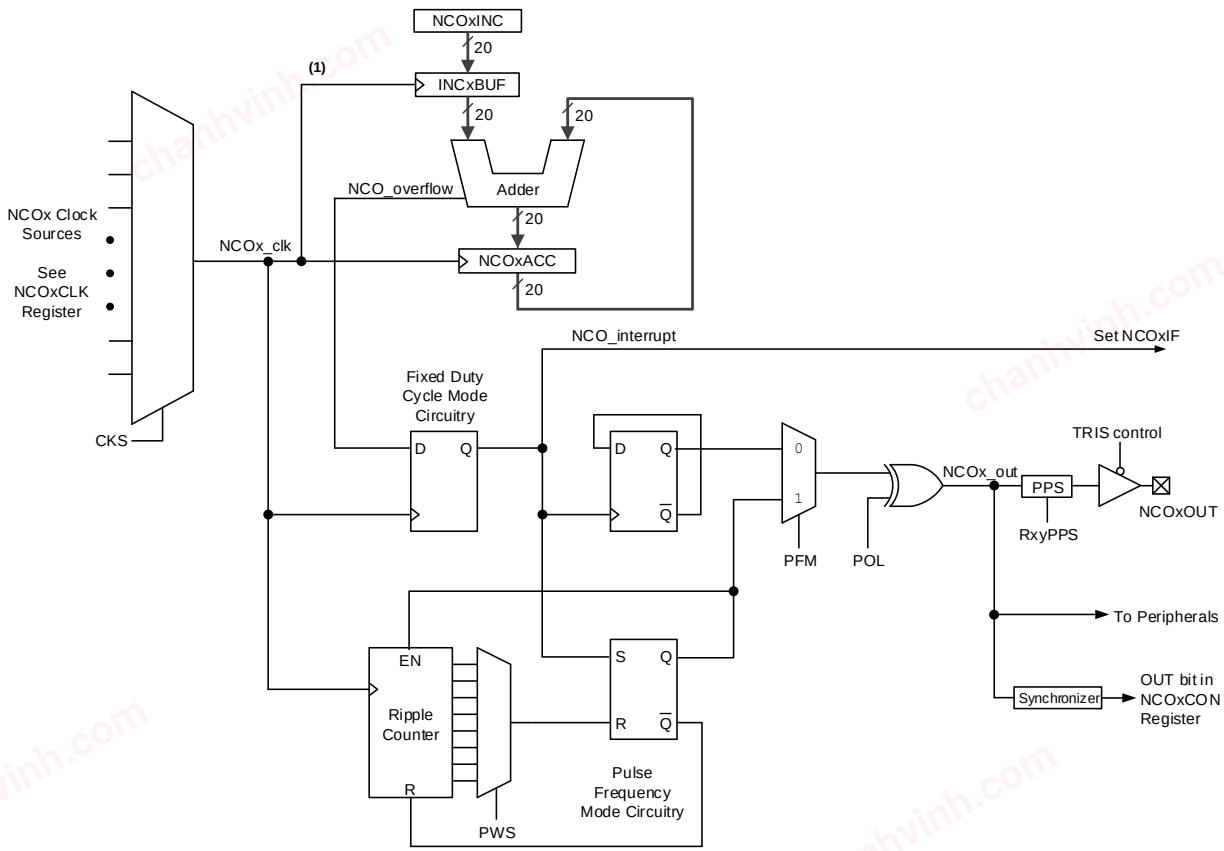
The Numerically Controlled Oscillator (NCO) module is a timer that uses overflow from the addition of an increment value to divide the input frequency. The advantage of the addition method over a simple counter driven timer is that the output frequency resolution does not vary with the divider value. The NCO is most useful for applications that require frequency accuracy and fine resolution at a fixed duty cycle.

Features of the NCO include:

- 20-Bit Increment Function
- Fixed Duty Cycle (FDC) mode
- Pulse Frequency (PF) mode
- Output Pulse-Width Control
- Multiple Clock Input Sources
- Output Polarity Control
- Interrupt Capability

The following figure is a simplified block diagram of the NCO module.

**Figure 25-1. Numerically Controlled Oscillator Module Simplified Block Diagram**




**Note 1:** The increment registers are double-buffered to allow for value changes to be made without first disabling the NCO module. The full increment value is loaded into the buffer registers on the second rising edge of the NCOx_clk signal that occurs immediately after a write to the NCOxINCL register. The buffers are not user-accessible and are shown here for reference.

chanhvinh.com



## 25.1 NCO Operation

The NCO operates by repeatedly adding a fixed value to an accumulator. Additions occur at the input clock rate. The accumulator will overflow with a carry periodically, which is the raw NCO output (NCO_overflow). This effectively reduces the input clock by the ratio of the addition value to the maximum accumulator value. See the following equation.

**Equation 25-1. NCO Overflow Frequency**

$$F_{OVERFLOW} = \frac{NCO\ Clock\ Frequency \times Increment\ Value}{2^{20}}$$

It is apparent from the equation that there is a linear relationship between the increment value and the overflow frequency. This linear advantage over divide-by-n timers comes at the cost of output jitter. However, the jitter is always plus or minus one NCO clock period that occurs periodically, depending on the division remainder. For example, there is no jitter when there is no division remainder, whereas a division remainder of 0.5 will result in a jitter frequency one half of the overflow frequency.

### 25.1.1 NCO Clock Sources

The NCO can be clocked from a variety of sources including the system clock, internal timers, and other peripherals. The NCO clock source is selected by configuring the CKS bits.

### 25.1.2 Accumulator

The accumulator is a 20-bit register. Read and write access to the accumulator is available through three registers:

- NCOxACCL
- NCOxACCH
- NCOxACCU

### 25.1.3 Adder

The NCO adder is a full adder, which operates synchronously from the source clock. The addition of the previous result and the increment value replaces the accumulator value on the rising edge of each input clock.

### 25.1.4 Increment Registers

The increment value is stored in three registers making up a 20-bit word. In order of LSB to MSB, they are:

- NCOxINCL
- NCOxINCH
- NCOxINCU

The increment registers are readable and writable and are double-buffered to allow value changes to be made without first disabling the NCO module.

When the NCO module is enabled, the NCOxINCU and NCOxINCH registers will be written first, then the NCOxINCL register. Writing to the NCOxINCL register initiates the increment buffer registers to be loaded simultaneously on the second rising edge of the NCO_clk signal.

When the NCO module is disabled, the increment buffers are loaded immediately after a write to the increment registers.

**Important:** The increment buffer registers are not user-accessible.

chanhvinh.com

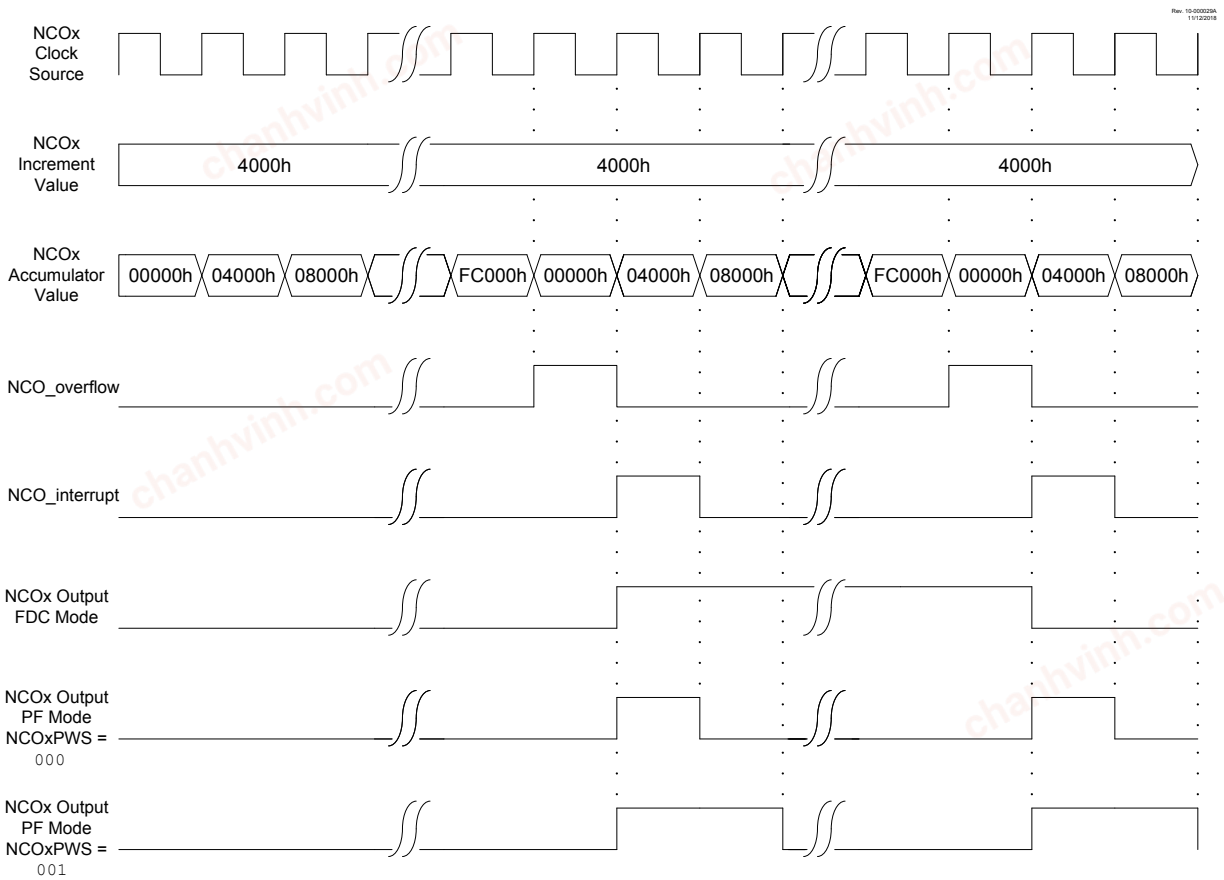


## 25.2 Fixed Duty Cycle Mode

In Fixed Duty Cycle (FDC) mode, every time the accumulator overflows, the output is toggled. This provides a 50% duty cycle at half the $F_{OVERFLOW}$ frequency, provided that the increment value remains constant. For more information, see the figure below.

The FDC mode is selected by clearing the PFM bit.

**Figure 25-2. FDC Output Mode Timing Diagram**




## 25.3 Pulse Frequency Mode

In Pulse Frequency (PF) mode, the output becomes active on the rising clock edge immediately following the overflow event, and goes inactive 1 to 128 clock periods later, determined by the PWS bits. This provides a pulsed output at the $F_{OVERFLOW}$ frequency. For more information, refer to the figure above.

**Important:** When the selected pulse width is greater than the accumulator overflow time frame, then the NCO output does not toggle.

The level of the Active and Inactive states is determined by the POL bit.

PF mode is selected by setting the PFM bit.

chanhvinh.com



## 25.4 Output Polarity Control

The last stage in the NCO module is the output polarity. The POL bit selects the output polarity. The active level of the Pulse Frequency mode is high true when the POL bit is cleared.

Changing the polarity while the interrupts are enabled will cause an interrupt for the resulting output transition.

The NCO output signal (NCOx_out) is available by internal routing to several other peripherals.

## 25.5 Interrupts

When the accumulator overflows, the NCO Interrupt Flag bit, NCOxIF, in the associated PIR register is set. To enable interrupt service on this event, the following bits must be set:

- EN bit
- NCOxIE bit in the associated PIE register
- Peripheral and Global Interrupt Enable bits

The interrupt must be cleared by software by clearing the NCOxIF bit in the Interrupt Service Routine.

## 25.6 Effects of a Reset

All of the NCO registers are cleared to zero as the result of any Reset.

## 25.7 Operation in Sleep

The NCO module operates independently from the system clock and will continue to run during Sleep, provided that the clock source selected remains active.

The HFINTOSC remains active during Sleep when the NCO module is enabled and the 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 NCO clock source, when the NCO is enabled, the CPU will go Idle during Sleep, but the NCO will continue to operate and the HFINTOSC will remain active.

With a clock running, it will have a direct effect on the Sleep mode current.

## 25.8 Register Definitions: NCO

Long bit name prefixes for the NCO peripherals are shown in the table below. Refer to the **“Long Bit Names”** section in the **“Register and Bit Naming Conventions”** chapter for more information.

**Table 25-1. NCO Long Bit Name Prefixes**

| Peripheral | Bit Name Prefix |
|---|---|
| NCO1 | NCO1 |

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



### 25.8.1 NCOxCON

**Name:** NCOxCON
**Offset:** 0x0592

NCO Control Register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | EN | | OUT | POL | | | | PFM |
| Access | R/W | | R | R/W | | | | R/W |
| Reset | 0 | | 0 | 0 | | | | 0 |

**Bit 7 – EN** NCO Enable

| Value | Description |
|---|---|
| 1 | NCO module is enabled |
| 0 | NCO module is disabled |

**Bit 5 – OUT** NCO Output

Displays the current logic level of the NCO module output.

**Bit 4 – POL** NCO Polarity

| Value | Description |
|---|---|
| 1 | NCO output signal is inverted |
| 0 | NCO output signal is not inverted |

**Bit 0 – PFM** NCO Pulse Frequency Mode

| Value | Description |
|---|---|
| 1 | NCO operates in Pulse Frequency mode. Output frequency is $F_{OVERFLOW}$. |
| 0 | NCO operates in Fixed Duty Cycle mode. Output frequency is $F_{OVERFLOW}$ divided by 2. |

chanhvinh.com

chanhvinh.com



### 25.8.2 NCOxCLK

**Name:** NCOxCLK
**Offset:** 0x0593

NCO Input Clock Control Register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PWS[2:0] | | | | CKS[3:0] | | | |
| Access | R/W | R/W | R/W | | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | | 0 | 0 | 0 | 0 |

**Bits 7:5 – PWS[2:0]** NCO Output Pulse-Width Select(1)

| Value | Description |
|---|---|
| 111 | NCO output is active for 128 input clock periods |
| 110 | NCO output is active for 64 input clock periods |
| 101 | NCO output is active for 32 input clock periods |
| 100 | NCO output is active for 16 input clock periods |
| 011 | NCO output is active for 8 input clock periods |
| 010 | NCO output is active for 4 input clock periods |
| 001 | NCO output is active for 2 input clock periods |
| 000 | NCO output is active for 1 input clock periods |

**Bits 3:0 – CKS[3:0]** NCO Clock Source Select

| CKS Value | Clock Source NCO1 | Active in Sleep |
|---|---|---|
| 1111 | Reserved | - |
| 1110 | CLC4_OUT | No |
| 1101 | CLC3_OUT | No |
| 1100 | CLC2_OUT | No |
| 1011 | CLC1_OUT | No |
| 1010 | TMR6_Postscaled_OUT | No |
| 1001 | TMR4_Postscaled_OUT | No |
| 1000 | TMR2_Postscaled_OUT | No |
| 0111 | CLKR_OUT | No |
| 0110 | EXTOSC | Yes |
| 0101 | SOSC | Yes |
| 0100 | MFINTOSC (32 kHz) | Yes |
| 0011 | MFINTOSC (500 kHz) | Yes |
| 0010 | LFINTOSC | Yes |
| 0001 | HFINTOSC | Yes |
| 0000 | $F_{OSC}$ | No |

**Note:**
1. PWS applies only when operating in Pulse Frequency mode.

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



### 25.8.3 NCOxACC

**Name:** NCOxACC
**Offset:** 0x058C

NCO Accumulator Register

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | ACC[19:16] | | | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | ACC[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ACC[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 19:0 – ACC[19:0]** Accumulated sum of NCO additions

**Notes:**

1. The individual bytes in this multibyte register can be accessed with the following register names:
   - NCOxACCU: Accesses the upper byte ACC[23:16]
   - NCOxACCH: Accesses the high byte ACC[15:8]
   - NCOxACCL: Accesses the low byte ACC[7:0].
2. The accumulator spans registers NCOxACCU:NCOxACCH:NCOxACCL. The 24 bits are reserved, but not all are used. This register updates in real-time, asynchronously to the CPU; there is no provision to ensure atomic access to this 24-bit space using an 8-bit bus. Writing to this register while the module is operating will produce undefined results.

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



### 25.8.4 NCOxINC

**Name:** NCOxINC
**Offset:** 0x058F

NCO Increment Register

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | INC[19:16] | | | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | INC[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | INC[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |

**Bits 19:0 – INC[19:0]** Value by which the NCOxACC is increased by each NCO clock

**Notes:**

1. The individual bytes in this multibyte register can be accessed with the following register names:
   – NCOxINCU: Accesses the upper byte INC[19:16]
   – NCOxINCH: Accesses the high byte INC[15:8]
   – NCOxINCL: Accesses the low byte INC[7:0].
2. The logical increment spans NCOxINCU:NCOxINCH:NCOxINCL.
3. NCOxINC is double-buffered as INCBUF:
   – INCBUF is updated on the next falling edge of NCOxCLK after writing to NCOxINCL
   – NCOxINCU and NCOxINCH will be written prior to writing NCOxINCL.

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



## 25.9 Register Summary - NCO

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 ... 0x058B | Reserved | | | | | | | | | |
| 0x058C | NCO1ACC | 7:0 | ACC[7:0] | | | | | | | |
| | | 15:8 | ACC[15:8] | | | | | | | |
| | | 23:16 | | | | | ACC[19:16] | | | |
| 0x058F | NCO1INC | 7:0 | INC[7:0] | | | | | | | |
| | | 15:8 | INC[15:8] | | | | | | | |
| | | 23:16 | | | | | INC[19:16] | | | |
| 0x0592 | NCO1CON | 7:0 | EN | | OUT | POL | | | | PFM |
| 0x0593 | NCO1CLK | 7:0 | PWS[2:0] | | | | CKS[3:0] | | | |

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



# 26. CWG - Complementary Waveform Generator Module

The Complementary Waveform Generator (CWG) produces half-bridge, full-bridge, and steering of PWM waveforms. It is backwards compatible with previous CCP 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
- 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 option
  - Auto-shutdown pin override control

## 26.1 Fundamental Operation

The CWG generates two output waveforms from the selected input source.

The off-to-on transition of each output can be delayed from the on-to-off transition of the other output, thereby creating a time delay immediately where neither output is driven. This is referred to as dead time and is covered in the Dead-Band Control section.

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. This is referred to as auto-shutdown and is covered in the Auto-Shutdown section.

## 26.2 Operating Modes

The CWG module can operate in six different modes, as specified by the MODE bits:

- Half Bridge mode
- Push-Pull mode
- Asynchronous Steering mode
- Synchronous Steering mode
- Full Bridge mode, Forward
- Full Bridge mode, Reverse

All modes accept a single pulse input, and provide up to four outputs as described in the following sections.

All modes include auto-shutdown control as described in the Auto-Shutdown section.

**Important:** Except as noted for Full Bridge mode, mode changes must only be performed while EN = 0.

chanhvinh.com

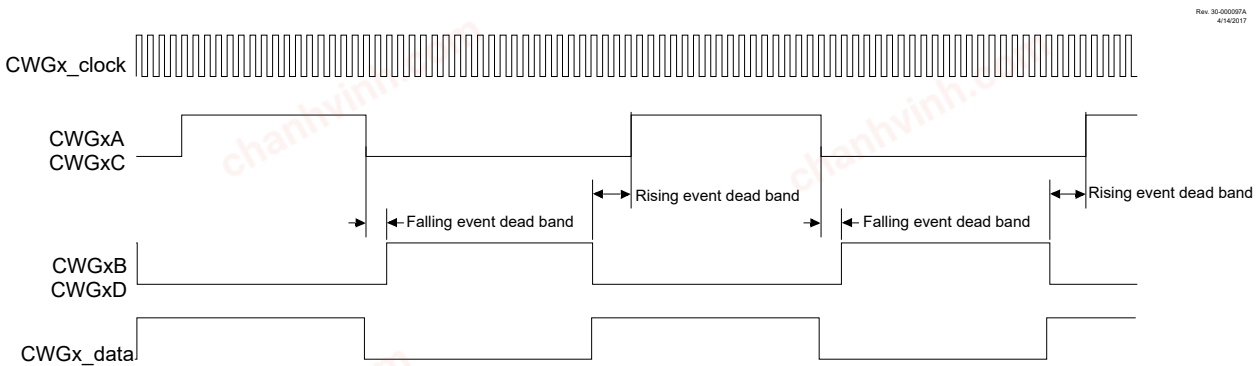


### 26.2.1 Half Bridge Mode

In Half Bridge mode, two output signals are generated as true and inverted versions of the input as illustrated in Figure 26-1. A nonoverlap (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 the Dead-Band Control section. The output steering feature cannot be used in this mode. A basic block diagram of this mode is shown in Figure 26-2.

The unused outputs CWGxC and CWGxD drive similar signals as CWGxA and CWGxB, with polarity independently controlled by the POLC and POLD bits, respectively.

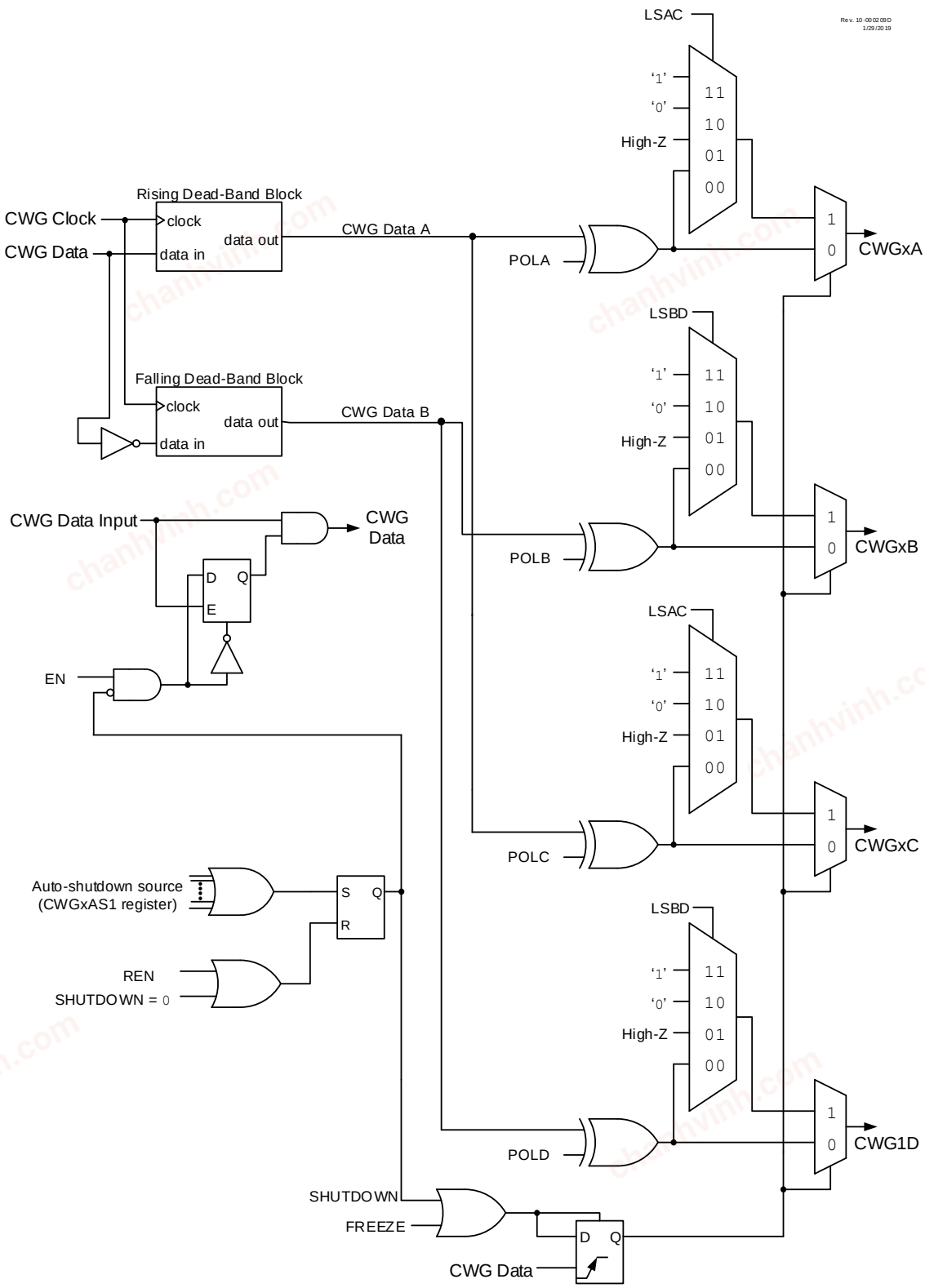
**Figure 26-1. CWG Half Bridge Mode Operation**



chanhvinh.com



**Figure 26-2. Simplified CWG Block Diagram (Half Bridge Mode, MODE = 'b100)**



chanhvinh.com



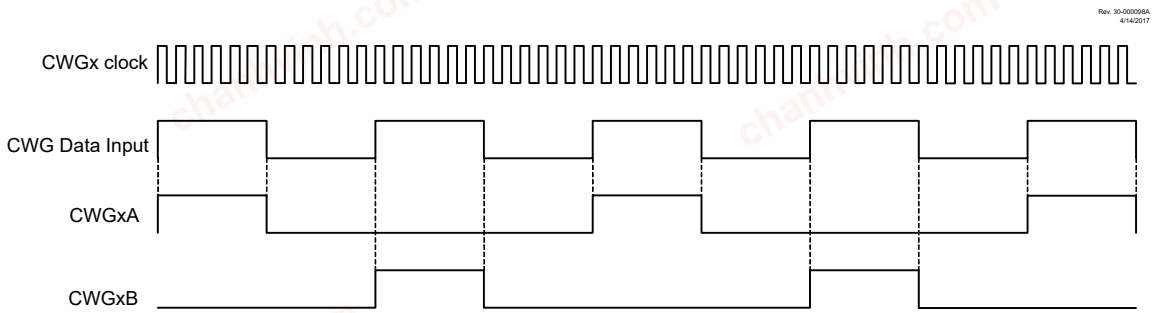
### 26.2.2 Push-Pull Mode

In Push-Pull mode, two output signals are generated, alternating copies of the input as illustrated in Figure 26-3. This alternation creates the Push-Pull effect required for driving some transformer-based power supply designs. Steering modes are not used in Push-Pull mode. A basic block diagram for the Push-Pull mode is shown in Figure 26-4.

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, respectively.

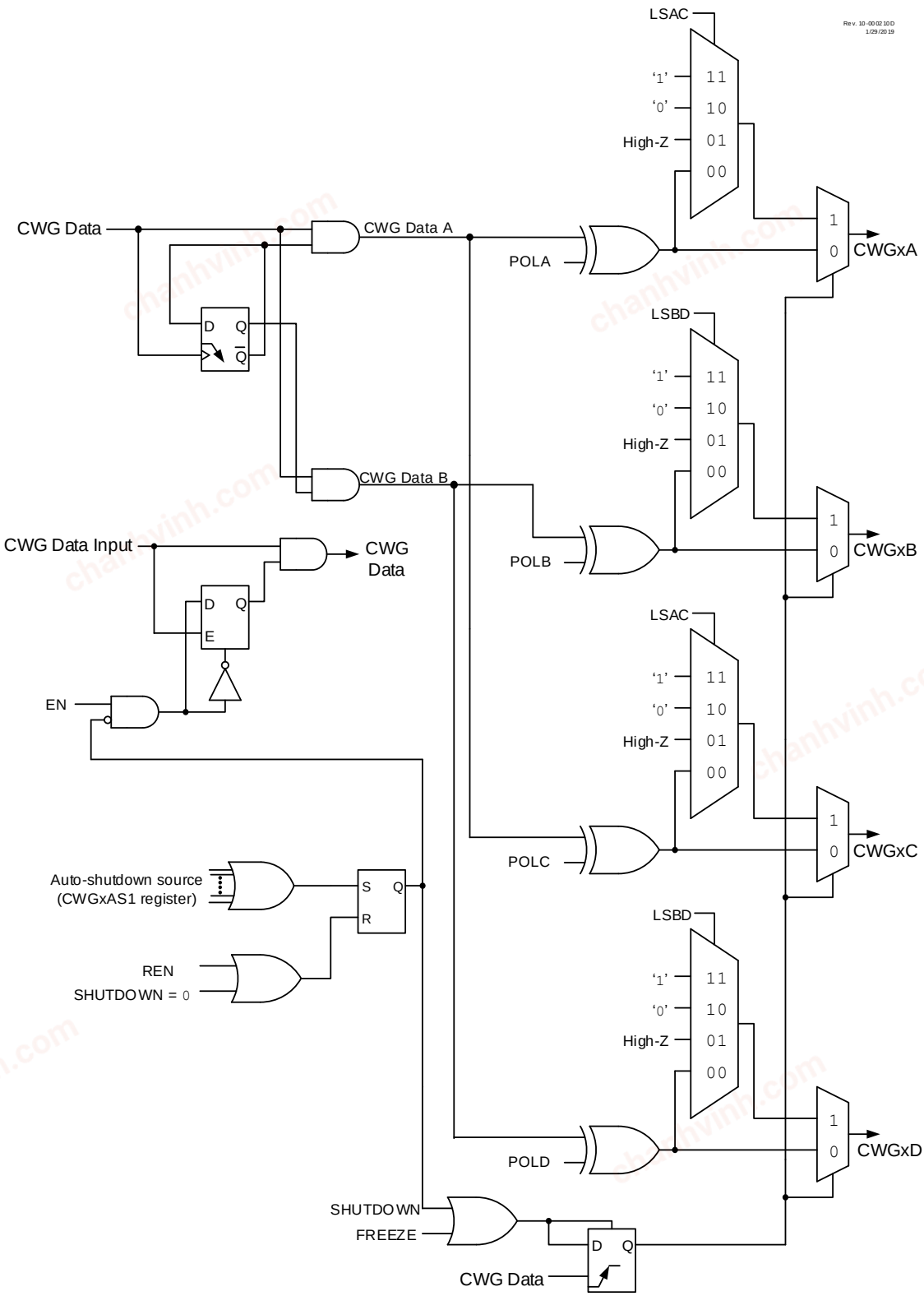
**Figure 26-3. CWG Push-Pull Mode Operation**



chanhvinh.com



**Figure 26-4. Simplified CWG Block Diagram (Push-Pull Mode, MODE = 'b101)**



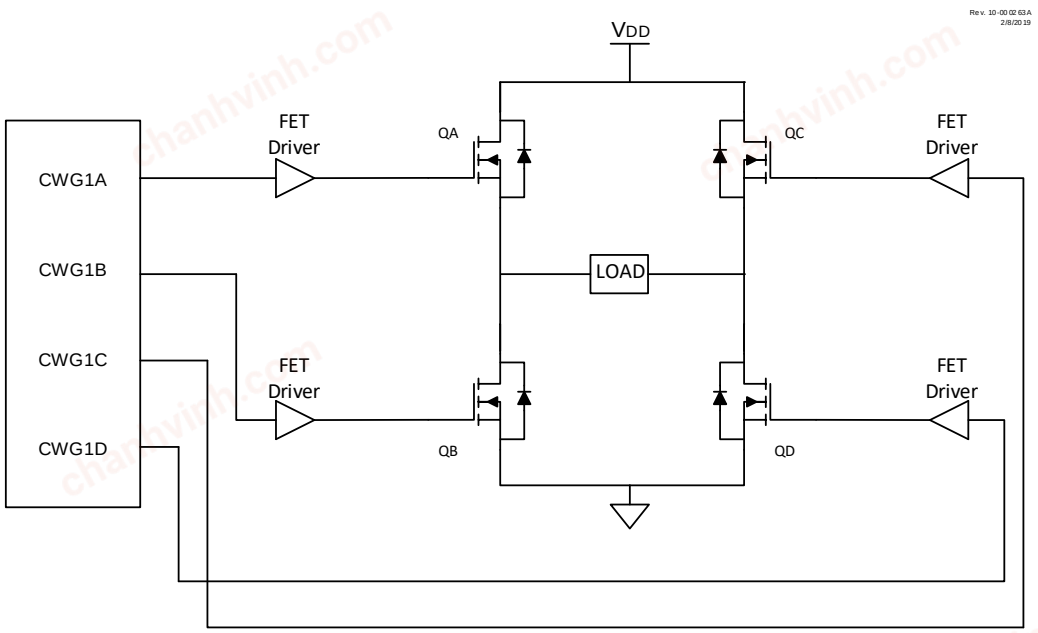
chanhvinh.com



### 26.2.3 Full Bridge Mode

In Forward and Reverse Full Bridge modes, three outputs drive static values while the fourth is modulated by the input data signal. The mode selection may be toggled between forward and reverse by toggling the MODE[0] bit of the CWGxCON0 register while keeping the MODE[2:1] bits static, without disabling the CWG module. When connected, as shown in Figure 26-5, the outputs are appropriate for a full-bridge motor driver. Each CWG output signal has independent polarity control, so the circuit can be adapted to high-active and low-active drivers. A simplified block diagram for the Full Bridge modes is shown in Figure 26-6.

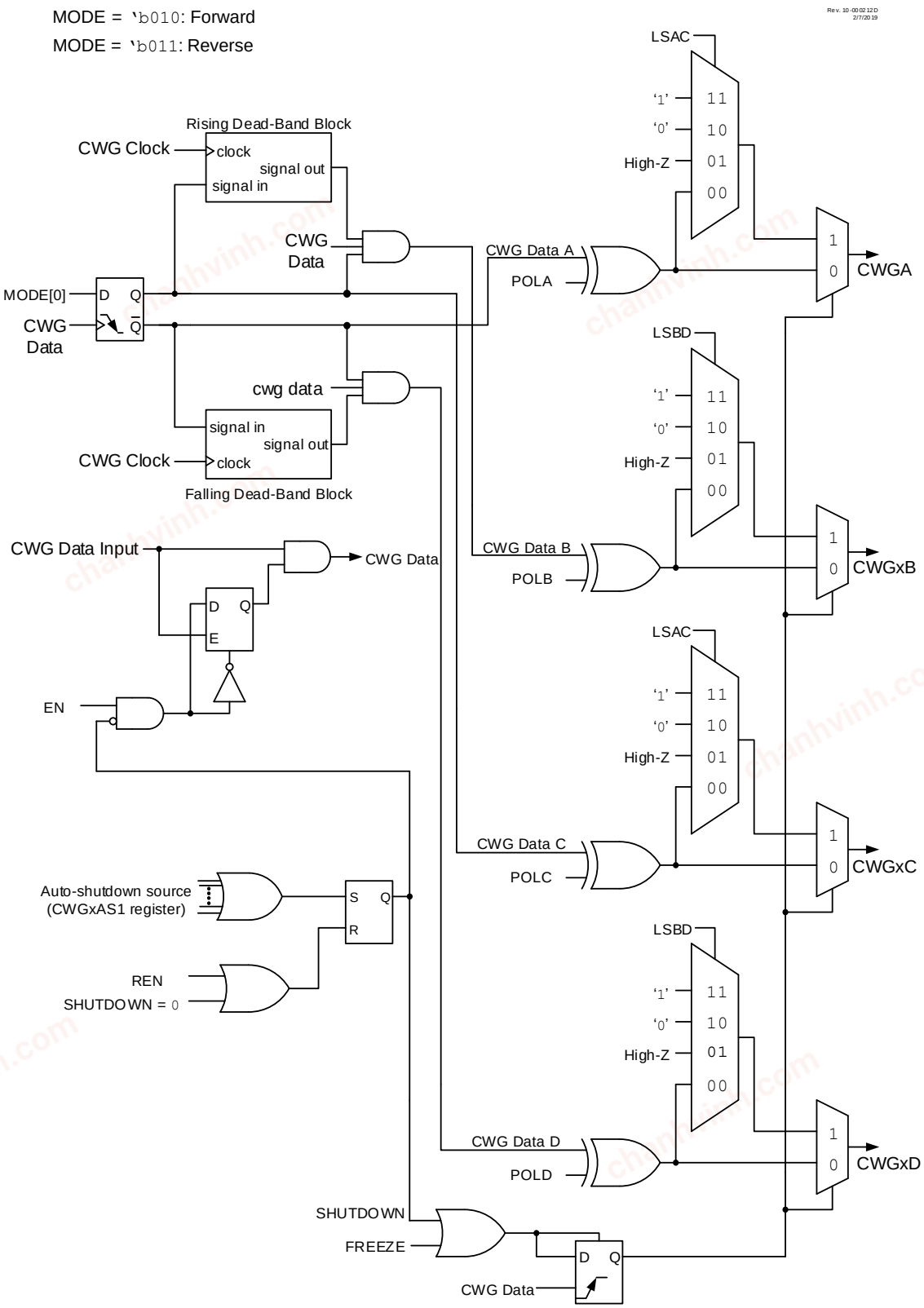
**Figure 26-5. Example of Full-Bridge Application**



chanhvinh.com



**Figure 26-6. Simplified CWG Block Diagram (Forward and Reverse Full Bridge Modes)**



chanhvinh.com

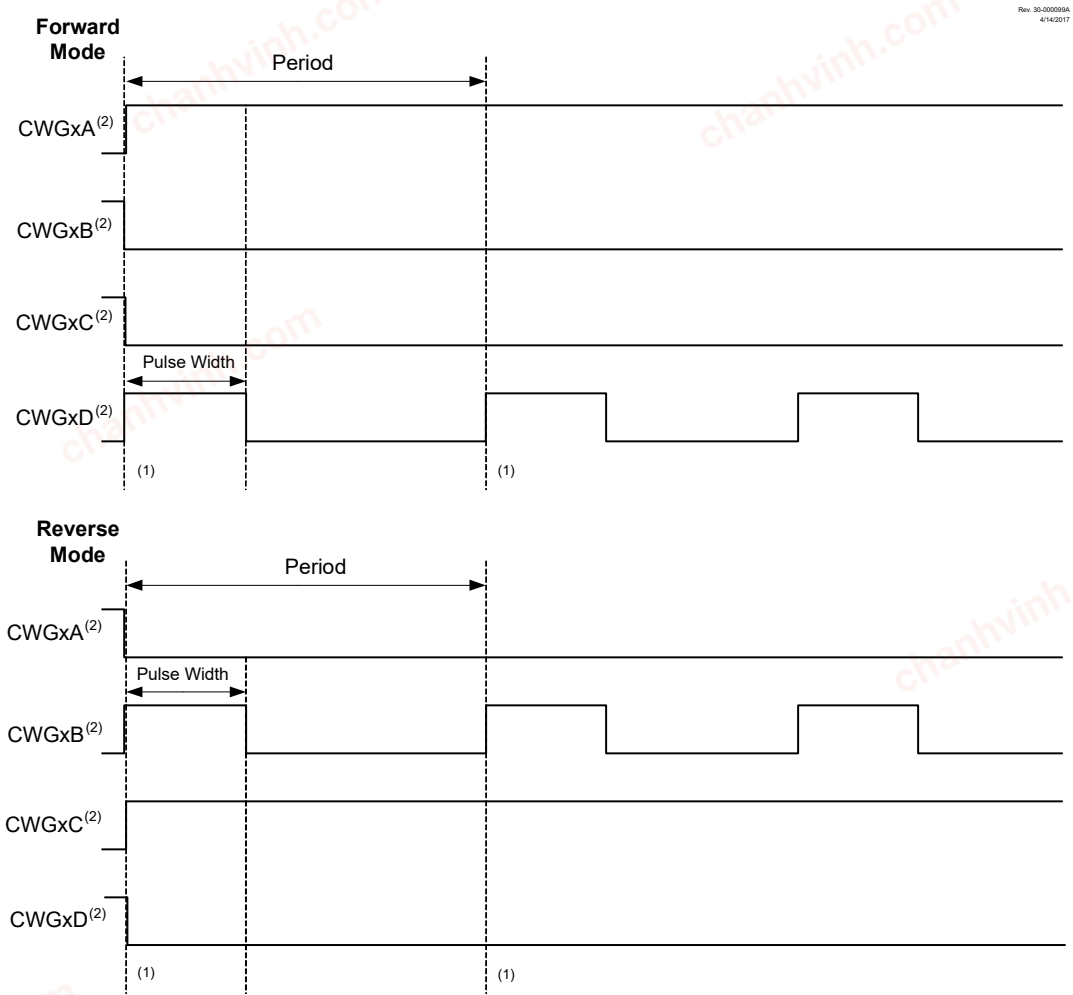


In Forward Full Bridge mode (MODE = 'b010), CWGxA is driven to its Active state, CWGxB and CWGxC are driven to their Inactive state, and CWGxD is modulated by the input signal, as shown in Figure 26-7.

In Reverse Full Bridge mode (MODE = 'b011), CWGxC is driven to its Active state, CWGxA and CWGxD are driven to their Inactive states, and CWGxB is modulated by the input signal, as shown in Figure 26-7.

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 the Dead-Band Control section, with additional details in the Rising Edge and Reverse Dead Band and Falling Edge and Forward Dead Band sections. Steering modes are not used with either of the Full Bridge modes.

**Figure 26-7. Example of Full-Bridge Output**




**Notes:**

1. A rising CWG data input creates a rising event on the modulated output.
2. Output signals shown as active-high; all POLy bits are clear.

### 26.2.3.1 Direction Change in Full Bridge Mode

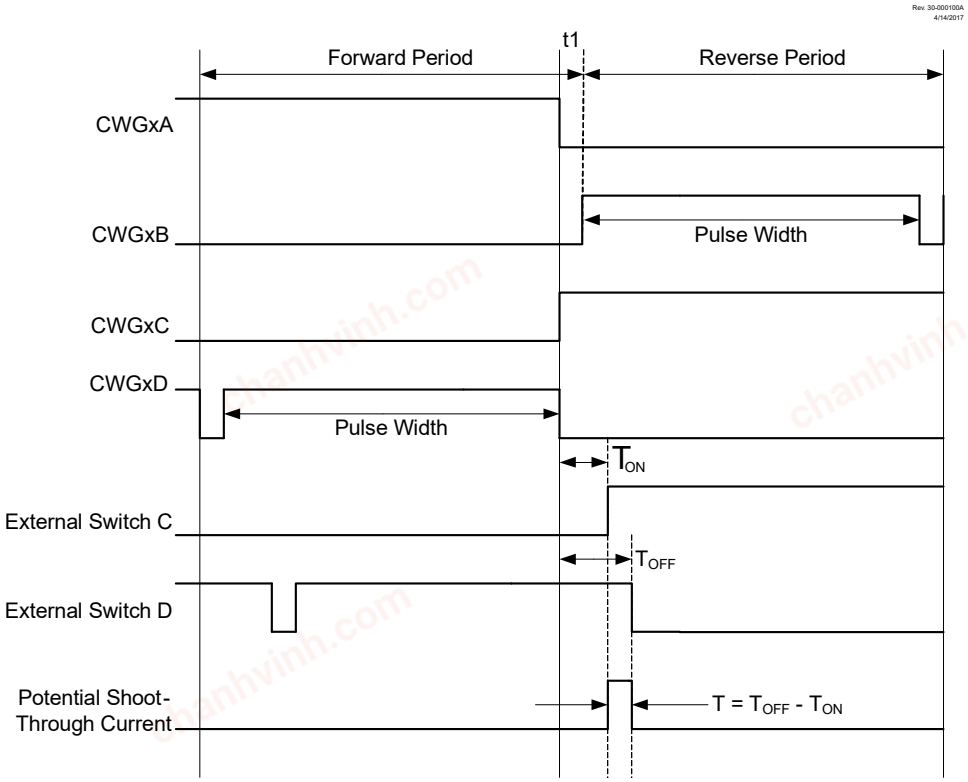
In Full Bridge mode, changing the MODE[0] bit controls the forward/reverse direction. Direction changes occur on the next rising edge of the modulated input. The sequence, described as follows, is illustrated in Figure 26-8.

1. The associated active output CWGxA and the inactive output CWGxC are switched to drive in the opposite direction.
2. The previously modulated output CWGxD is switched to the Inactive state, and the previously inactive output CWGxB begins to modulate.
3. CWG modulation resumes after the direction-switch dead band has elapsed.

chanhvinh.com



**Figure 26-8. Example of PWM Direction Change at Near 100% Duty Cycle**




### 26.2.3.2 Dead-Band Delay in Full Bridge Mode

Dead-band delay is important when either of the following conditions is true:

- The direction of the CWG output changes when the duty cycle of the data input is at or near 100%.
- The turn-off time of the power switch, including the power device and driver circuit, is greater than the turn-on time.

The dead-band delay is inserted only when changing directions, and only the modulated output is affected. The statically-configured outputs (CWGxA and CWGxC) are not afforded dead band, and switch essentially simultaneously.

Figure 26-8 shows an example of the CWG outputs changing directions from forward to reverse, at near 100% duty cycle. In this example, at time t1, the output of CWGxA and CWGxD becomes inactive, while the output of CWGxC becomes active. Since the turn-off time of the power devices is longer than the turn-on time, a shoot-through current will flow through the power devices QC and QD for the duration of 'T'. The same phenomenon will occur to power devices QA and QB for the CWG direction change from reverse to forward.

When changing the CWG direction at high duty cycle is required for an application, two possible solutions for eliminating the shoot-through current are:

1. Reduce the CWG duty cycle for one period before changing directions.
2. Use switch drivers that can drive the switches off faster than they can drive them on.

## 26.2.4 Steering Modes

In both Synchronous and Asynchronous Steering modes, the CWG Data can be steered to any combination of four CWG outputs. A fixed value will be presented on all the outputs not used for the PWM output. Each output has independent polarity, steering, and shutdown options. Dead-band control is not used in either Steering mode.

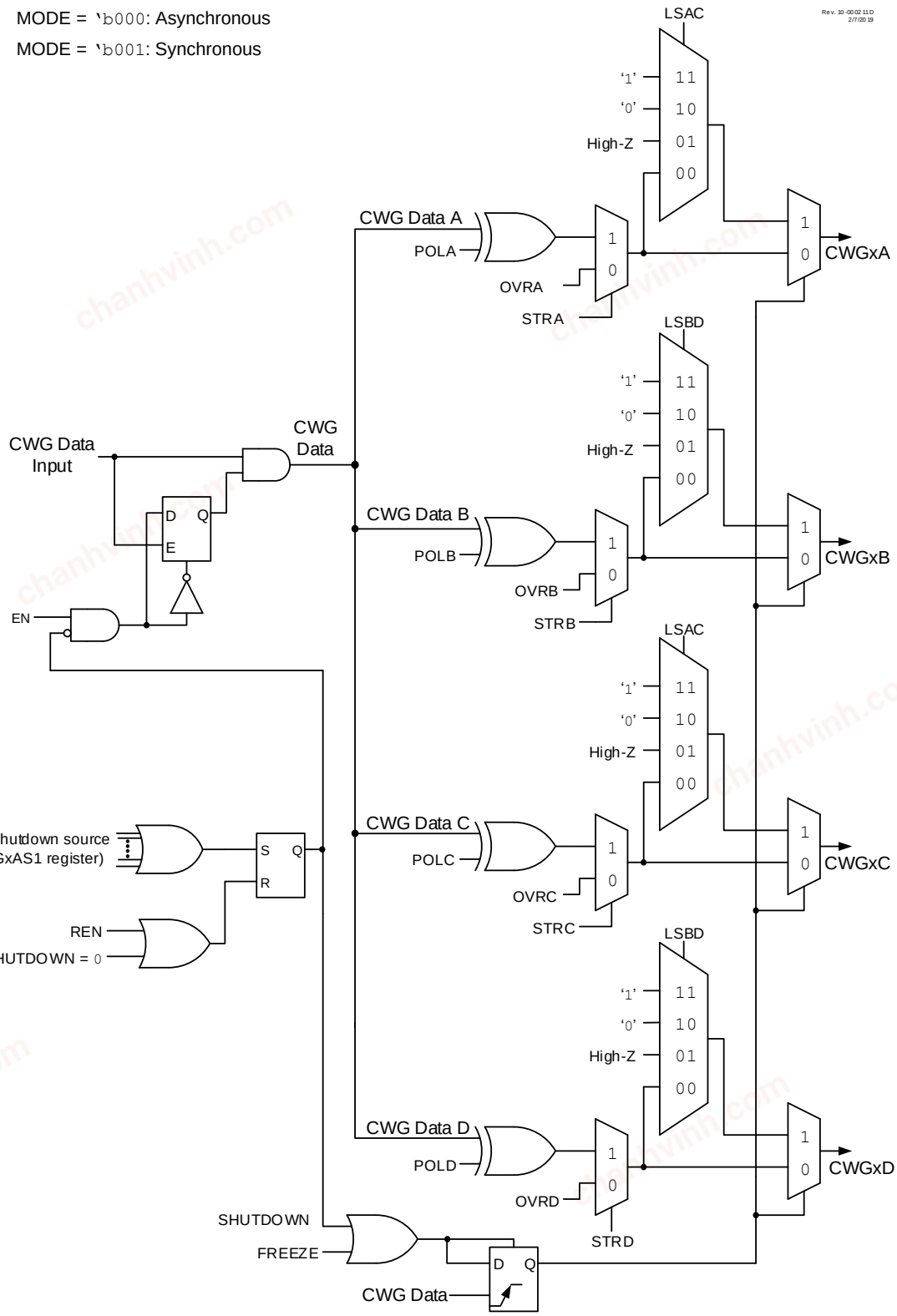
For example, when STRA = 0, the corresponding pin is held at the level defined by OVRA. When STRA = 1, the pin is driven by the CWG Data signal. The POLy bits control the signal polarity only when STRy = 1.

The CWG auto-shutdown operation also applies in Steering modes as described in the Auto-Shutdown section. An auto-shutdown event will only affect pins that have STRy = 1.

chanhvinh.com



**Figure 26-9. Simplified CWG Block Diagram (Output Steering Modes)**



chanhvinh.com

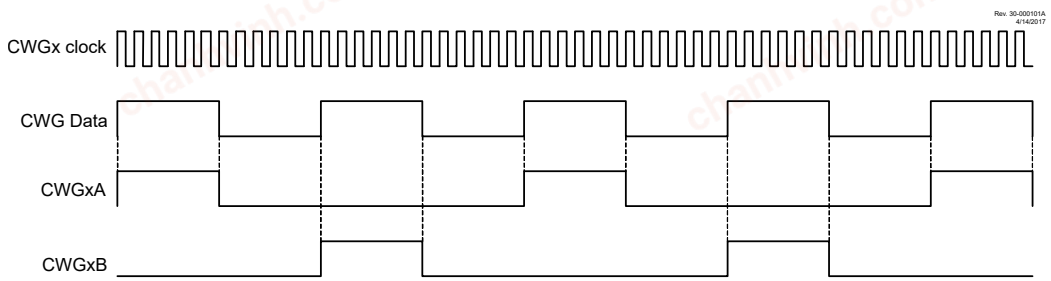


#### 26.2.4.1 Synchronous Steering Mode

In Synchronous Steering mode (MODE = 'b001), the changes to steering selection registers take effect on the next rising edge of CWG Data (see the figure below). In Synchronous Steering mode, the output will always produce a complete waveform.

**Important:** Only the STRx bits are synchronized; the OVRx bits are not synchronized.

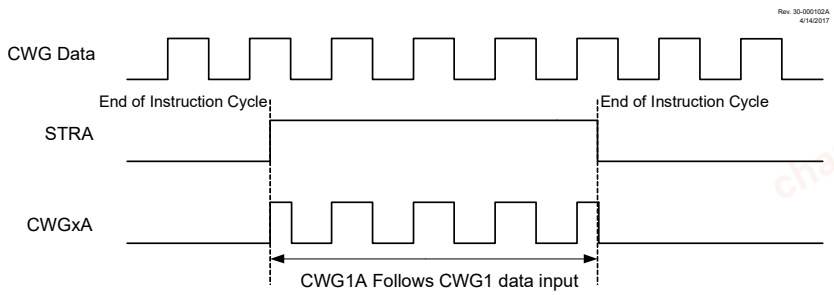
**Figure 26-10. Example of Synchronous Steering (MODE = 'b001)**




#### 26.2.4.2 Asynchronous Steering Mode

In Asynchronous mode (MODE = 'b000), steering takes effect at the end of the instruction cycle that writes to STRx. In Asynchronous Steering mode, the output signal may be an incomplete waveform (see the figure below). This operation may be useful when the user firmware needs to immediately remove a signal from the output pin.

**Figure 26-11. Example of Asynchronous Steering (MODE = 'b000)**




#### 26.2.4.3 Start-Up Considerations

The application hardware must use the proper external pull-up and/or pull-down resistors on the CWG output pins. This is required because all I/O pins are forced to high-impedance at Reset.

The Polarity Control (POLy) bits allow the user to choose whether the output signals are active-high or active-low.

## 26.3 Clock Source

The clock source is used to drive the dead-band timing circuits. The CWG module allows the following clock sources to be selected:

- $F_{OSC}$ (system clock)
- HFINTOSC

When the HFINTOSC is selected, the HFINTOSC will be kept running during Sleep. Therefore, the CWG modes requiring dead band can operate in Sleep, provided that the CWG data input is also active during Sleep. The clock sources are selected using the CS bit. The system clock $F_{OSC}$ is disabled in Sleep and thus dead-band control cannot be used.

chanhvinh.com



## 26.4 Selectable Input Sources

The CWG generates the output waveforms from the input sources which are selected with the ISM bits. Refer to the CWGxISM register for more details.

## 26.5 Output Control

### 26.5.1 CWG Output

Each CWG output can be routed to a Peripheral Pin Select (PPS) output via the RxyPPS register. Refer to the **"PPS - Peripheral Pin Select Module"** chapter for more details.

### 26.5.2 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 POLy bits. Auto-shutdown and steering options are unaffected by polarity.

## 26.6 Dead-Band Control

The dead-band control provides nonoverlapping complementary outputs to prevent shoot-through current when the outputs switch. 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 direction change dead band in Full Bridge mode. The other is used for the falling edge of the input source control in Half Bridge mode or for forward direction change 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.

### 26.6.1 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 26-1.

### 26.6.2 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 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.

## 26.7 Rising Edge and Reverse Dead Band

In Half Bridge mode, the rising edge dead band delays the turn-on of the CWGxA output after the rising edge of the CWG data input. In Full Bridge mode, the reverse dead-band delay is only inserted when changing directions from Forward mode to Reverse mode, and only the modulated output, CWGxB, is affected.

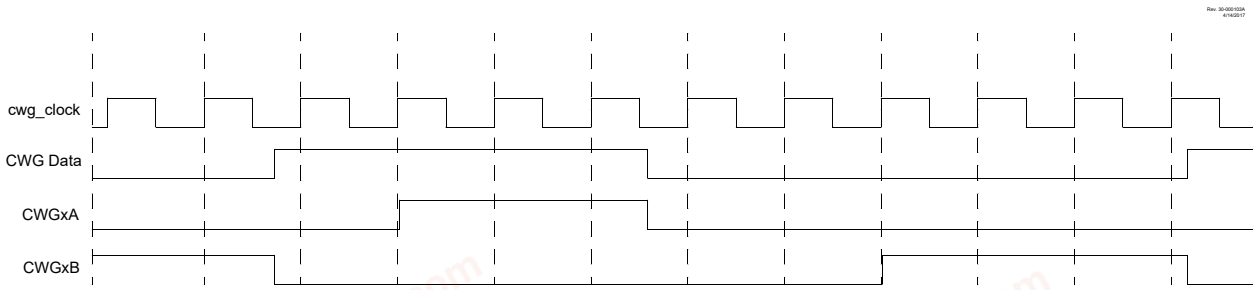
The CWGxDBR register determines the duration of the dead-band interval on the rising edge of the input source signal. This duration is from 0 to 64 periods of the CWG clock. The following figure illustrates different dead-band delays for rising and falling CWG Data events.

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



**Figure 26-12. Dead-Band Operation, CWGxDBR = 0x01, CWGxDBF = 0x02**




Dead band is always initiated on the edge of the input source signal. A count of zero indicates that no dead band is present.

If the input source signal reverses polarity before the dead-band count is completed, then no signal will be seen on the respective output.

The CWGxDBR register value is double-buffered. When EN = 0, the buffer is loaded when CWGxDBR is written. When EN = 1, the buffer will be loaded at the rising edge following the first falling edge of the CWG Data, after the LD bit is set.

## 26.8 Falling Edge and Forward Dead Band

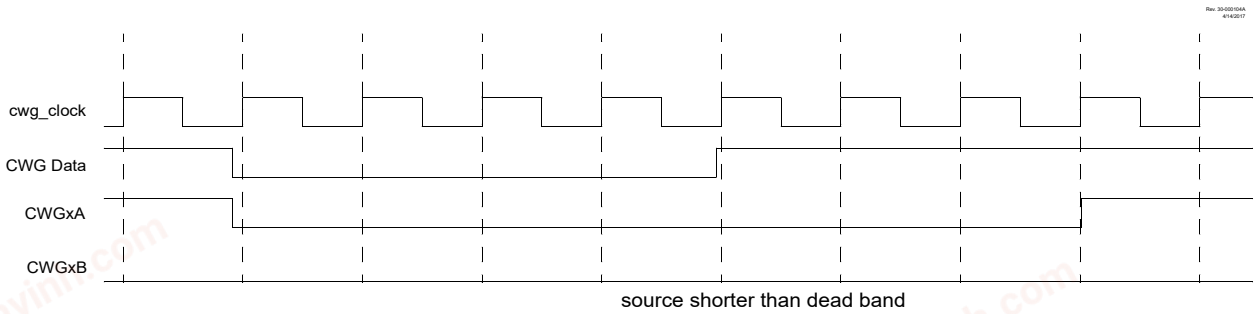
In Half Bridge mode, the falling edge dead band delays the turn-on of the CWGxB output at the falling edge of the CWG data input. In Full Bridge mode, the forward dead-band delay is only inserted when changing directions from Reverse mode to Forward mode, and only the modulated output, CWGxD, is affected.

The CWGxDBF register determines the duration of the dead-band interval on the falling edge of the input source signal. This duration is from 0 to 64 periods of the CWG clock.

Dead-band delay is always initiated on the edge of the input source signal. A count of zero indicates that no dead band is present.

If the input source signal reverses polarity before the dead-band count is completed, then no signal will be seen on the respective output.

**Figure 26-13. Dead-Band Operation, CWGxDBR = 0x03, CWGxDBF = 0x06, Source Shorter Than Dead Band**




The CWGxDBF register value is double-buffered. When EN = 0, the buffer is loaded when CWGxDBF is written. When EN = 1, the buffer will be loaded at the rising edge following the first falling edge of the data input after the LD bit is set.

## 26.9 Dead-Band Jitter

When the rising and falling edges of the input source are asynchronous to the CWG clock, it creates jitter in the dead-band time delay. The maximum jitter is equal to one CWG clock period. Refer to the equations below for more details.

chanhvinh.com



**Equation 26-1. Dead-Band Delay Time Calculation**

$$T_{DEAD-BAND_MIN} = \frac{1}{F_{CWG_CLOCK}} \bullet DBx$$

$$T_{DEAD-BAND_MAX} = \frac{1}{F_{CWG_CLOCK}} \bullet (DBx + 1)$$

$$T_{JITTER} = T_{DEAD-BAND_MAX} - T_{DEAD-BAND_MIN}$$

$$T_{JITTER} = \frac{1}{F_{CWG_CLOCK}}$$

$$T_{DEAD-BAND_MAX} = T_{DEAD-BAND_MIN} + T_{JITTER}$$

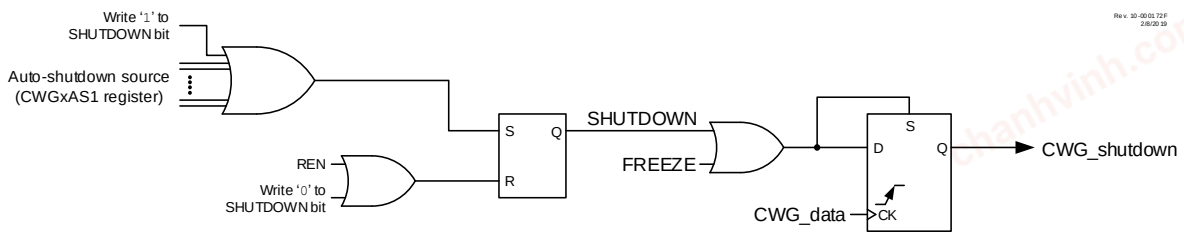
**Dead-Band Delay Example Calculation**

$DBx = 0x0A = 10$
$F_{CWG_CLOCK} = 8\ MHz$
$T_{JITTER} = \frac{1}{8\ MHz} = 125\ ns$
$T_{DEAD-BAND_MIN} = 125\ ns \bullet 10 = 1.25\ \mu s$
$T_{DEAD-BAND_MAX} = 1.25\ \mu s + 0.125\ \mu s = 1.37\ \mu s$

## 26.10 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 the following figure.

**Figure 26-14. CWG Shutdown Block Diagram**




### 26.10.1 Shutdown

The Shutdown state can be entered by either of the following two methods:

- Software Generated
- External Input

### 26.10.2 Software Generated Shutdown

Setting the SHUTDOWN bit 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. The SHUTDOWN bit indicates when a Shutdown condition exists. The bit may be set or cleared in software or by hardware.

### 26.10.3 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. The override levels are selected by the LSBD and LSAC bits. Several input sources can be selected to cause a Shutdown condition. All input sources are active-low. The shutdown input sources are individually enabled by the ASyE bits.

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



**Important:** 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.

### 26.10.4 Pin Override Levels

The levels driven to the CWG outputs during an auto-shutdown event are controlled by the LSBD and LSAC bits. The LSBD bits control CWGxB/D output levels, while the LSAC bits control the CWGxA/C output levels.

### 26.10.5 Auto-Shutdown Interrupts

When an auto-shutdown event occurs, either by software or hardware setting SHUTDOWN, the CWGxIF flag bit of the PIRx register is set.

## 26.11 Auto-Shutdown Restart

After an auto-shutdown event has occurred, there are two ways to resume operation:

- Software controlled
- Auto-restart

In either case, the shutdown source must be cleared before the restart can take place. That is, either the Shutdown condition must be removed, or the corresponding ASyE bit must be cleared.

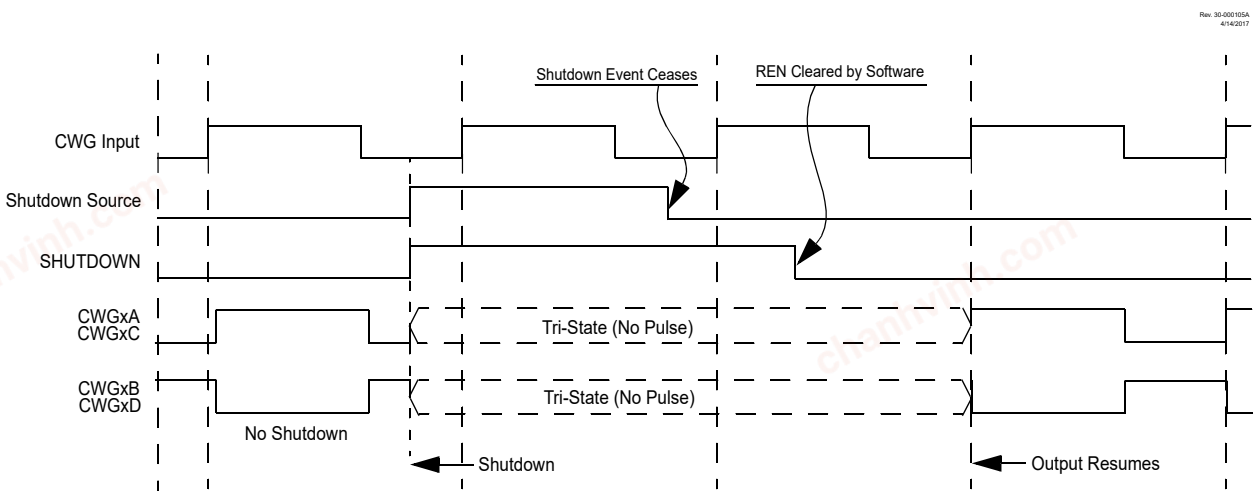
### 26.11.1 Software-Controlled Restart

When the REN bit is clear (REN = 0), the CWG module must be restarted after an auto-shutdown event through software.

Once all auto-shutdown sources are removed, the software must clear the SHUTDOWN bit. Once SHUTDOWN is cleared, the CWG module will resume operation upon the first rising edge of the CWG data input.

**Important:** The SHUTDOWN bit cannot be cleared in software if the Auto-Shutdown condition is still present.

**Figure 26-15. Shutdown Functionality, Auto-Restart Disabled (REN = 0, LSAC = 'b01, LSBD = 'b01)**




### 26.11.2 Auto-Restart

When the REN bit is set (REN = 1), the CWG module will restart from the Shutdown state automatically.

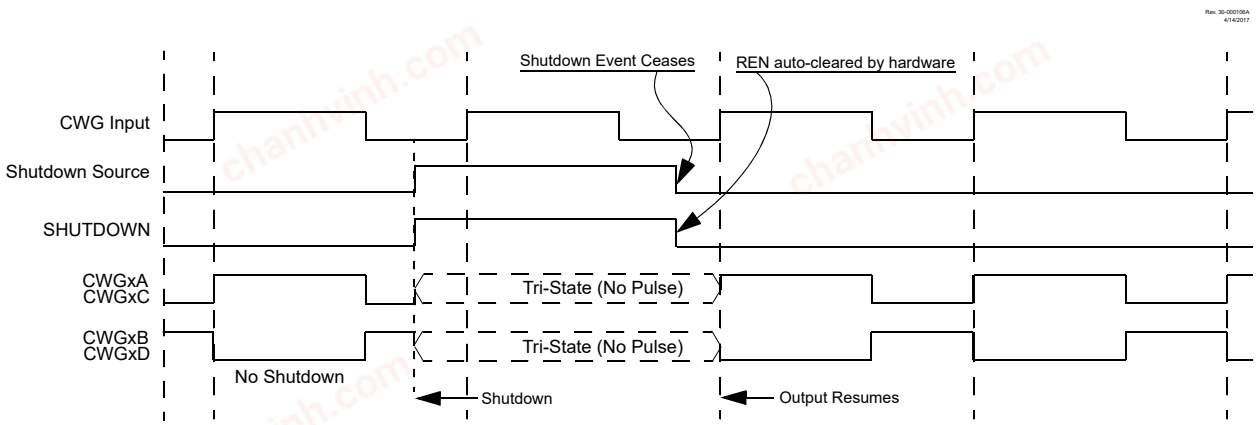
chanhvinh.com



Once all Auto-Shutdown conditions are removed, the hardware will automatically clear the SHUTDOWN bit. Once SHUTDOWN is cleared, the CWG module will resume operation upon the first rising edge of the CWG data input.

**Important:** The SHUTDOWN bit cannot be cleared in software if the Auto-Shutdown condition is still present.

**Figure 26-16. Shutdown Functionality, Auto-Restart Enabled (REN = 1, LSAC = 'b01, LSBD = 'b01)**




## 26.12 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.

## 26.13 Configuring the CWG

1. Ensure that the TRIS control bits corresponding to CWG outputs are set so that all are configured as inputs, ensuring that the outputs are inactive during setup. External hardware must ensure that pin levels are held to safe levels.
2. Clear the EN bit, if not already cleared.
3. Configure the MODE bits to set the output operating mode.
4. Configure the POLy bits to set the output polarities.
5. Configure the ISM bits to select the data input source.
6. If a Steering mode is selected, configure the STRy bits to select the desired output on the CWG outputs.
7. Configure the LSBD and LSAC bits to select the Auto-Shutdown Output Override states (this is necessary even if not using auto-shutdown, because start-up will be from a Shutdown state).
8. If auto-restart is desired, set the REN bit.
9. If auto-shutdown is desired, configure the ASyE bits to select the shutdown source.
10. Set the desired rising and falling dead-band times with the CWGxDBR and CWGxDBF registers.
11. Select the clock source with the CS bit.

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



12. Set the EN bit to enable the module.
13. Clear the TRIS bits that correspond to the CWG outputs to set them as outputs.

If auto-restart is to be used, set the REN bit and the SHUTDOWN bit will be cleared automatically. Otherwise, clear the SHUTDOWN bit in software to start the CWG.

## 26.14 Register Definitions: CWG Control

Long bit name prefixes for the CWG peripherals are shown in the table below. Refer to the **"Long Bit Names"** section in the **"Register and Bit Naming Conventions"** chapter for more information.

**Table 26-1. CWG Long Bit Name Prefixes**

| Peripheral | Bit Name Prefix |
|---|---|
| CWG1 | CWG1 |

chanhvinh.com

chanhvinh.com



### 26.14.1 CWGxCON0

**Name:** CWGxCON0
**Offset:** 0x0610

CWG Control Register 0

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | EN | LD | | | | MODE[2:0] | | |
| Access | R/W | R/W/HC | | | | R/W | R/W | R/W |
| Reset | 0 | 0 | | | | 0 | 0 | 0 |

**Bit 7 – EN** CWG Enable

| Value | Description |
|---|---|
| 1 | Module is enabled |
| 0 | Module is disabled |

**Bit 6 – LD** CWG1 Load Buffers(1)

| Value | Description |
|---|---|
| 1 | Dead-band count buffers to be loaded on CWG data rising edge, following first falling edge after this bit is set |
| 0 | Buffers remain unchanged |

**Bits 2:0 – MODE[2:0]** CWG Mode

| Value | Description |
|---|---|
| 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 Asynchronous Steering mode |

**Note:**

1. This bit can only be set after EN = 1; it cannot be set in the same cycle when EN is set.

chanhvinh.com

chanhvinh.com



### 26.14.2 CWGxCON1

**Name:** CWGxCON1
**Offset:** 0x0611

CWG Control Register 1

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | IN | | POLD | POLC | POLB | POLA |
| Access | | | R | | R/W | R/W | R/W | R/W |
| Reset | | | x | | 0 | 0 | 0 | 0 |

**Bit 5 – IN** CWG Input Value (read-only)

| Value | Description |
|---|---|
| 1 | CWG data input is a logic '1' |
| 0 | CWG data input is a logic '0' |

**Bits 0, 1, 2, 3 – POLy** CWG Output 'y' Polarity

| Value | Description |
|---|---|
| 1 | Signal output is inverted polarity |
| 0 | Signal output is normal polarity |

chanhvinh.com

chanhvinh.com



### 26.14.3 CWGxCLK

**Name:** CWGxCLK
**Offset:** 0x060C

CWG Clock Input Selection Register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | CS |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

**Bit 0 – CS** CWG Clock Source Selection Select

| Value | Description |
|---|---|
| 1 | HFINTOSC (remains operating during Sleep) |
| 0 | $F_{OSC}$ |

chanhvinh.com



### 26.14.4 CWGxISM

**Name:** CWGxISM
**Offset:** 0x060D

CWGx Input Selection Register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | ISM[4:0] | | | | |
| Access | | | | R/W | R/W | R/W | R/W | R/W |
| Reset | | | | 0 | 0 | 0 | 0 | 0 |

**Bits 4:0 – ISM[4:0]** CWG Data Input Source Select

| ISM | Input Selection |
|---|---|
| | CWG1 |
| 11111-10010 | Reserved |
| 10001 | CLC4_OUT |
| 10000 | CLC3_OUT |
| 01111 | CLC2_OUT |
| 01110 | CLC1_OUT |
| 01101 | C2_OUT |
| 01100 | C1_OUT |
| 01011 | NCO1_OUT |
| 01010 | PWM4S1P2_OUT |
| 01001 | PWM4S1P1_OUT |
| 01000 | PWM3S1P2_OUT |
| 00111 | PWM3S1P1_OUT |
| 00110 | PWM2S1P2_OUT |
| 00101 | PWM2S1P1_OUT |
| 00100 | PWM1S1P2_OUT |
| 00011 | PWM1S1P1_OUT |
| 00010 | CCP2_OUT |
| 00001 | CCP1_OUT |
| 00000 | Pin selected by CWG1PPS |

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



### 26.14.5 CWGxSTR

**Name:** CWGxSTR
**Offset:** 0x0614

CWG Steering Control Register[1]

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | OVRD | OVRC | OVRB | OVRA | STRD | STRC | STRB | STRA |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 4, 5, 6, 7 – OVRy** Steering Data OVR'y'

| Value | Condition | Description |
|---|---|---|
| x | STRy = 1 | CWGx'y' output has the CWG data input waveform with polarity control from POLy bit |
| 1 | STRy = 0 and POLy = x | CWGx'y' output is high |
| 0 | STRy = 0 and POLy = x | CWGx'y' output is low |

**Bits 0, 1, 2, 3 – STRy** STR'y' Steering Enable[2]

| Value | Description |
|---|---|
| 1 | CWGx'y' output has the CWG data input waveform with polarity control from the POLy bit |
| 0 | CWGx'y' output is assigned to value of the OVRy bit |

**Notes:**

1. The bits in this register apply only when MODE = 'b00x (CWGxCON0, Steering modes).
2. This bit is double-buffered when MODE = 'b001.

chanhvinh.com



### 26.14.6 CWGxAS0

**Name:** CWGxAS0
**Offset:** 0x0612

CWG Auto-Shutdown Control Register 0

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SHUTDOWN | REN | LSBD[1:0] | | LSAC[1:0] | | | |
| Access | R/W/HS/HC | R/W | R/W | R/W | R/W | R/W | | |
| Reset | 0 | 0 | 0 | 1 | 0 | 1 | | |

**Bit 7 – SHUTDOWN** Auto-Shutdown Event Status[1,2]

| Value | Description |
|---|---|
| 1 | An Auto-Shutdown state is in effect |
| 0 | No auto-shutdown event has occurred |

**Bit 6 – REN** Auto-Restart Enable

| Value | Description |
|---|---|
| 1 | Auto-restart is enabled |
| 0 | Auto-restart is disabled |

**Bits 5:4 – LSBD[1:0]** CWGxB and CWGxD Auto-Shutdown State Control

| Value | Description |
|---|---|
| 11 | A logic '1' is placed on CWGxB/D when an auto-shutdown event occurs |
| 10 | A logic '0' is placed on CWGxB/D when an auto-shutdown event occurs |
| 01 | Pin is tri-stated on CWGxB/D when an auto-shutdown event occurs |
| 00 | The Inactive state of the pin, including polarity, is placed on CWGxB/D after the required dead-band interval when an auto-shutdown event occurs |

**Bits 3:2 – LSAC[1:0]** CWGxA and CWGxC Auto-Shutdown State Control

| Value | Description |
|---|---|
| 11 | A logic '1' is placed on CWGxA/C when an auto-shutdown event occurs |
| 10 | A logic '0' is placed on CWGxA/C when an auto-shutdown event occurs |
| 01 | Pin is tri-stated on CWGxA/C when an auto-shutdown event occurs |
| 00 | The Inactive state of the pin, including polarity, is placed on CWGxA/C after the required dead-band interval when an auto-shutdown event occurs |

**Notes:**

1. This bit may be written while EN = 0, to place the outputs into the shutdown configuration.
2. The outputs will remain in Auto-Shutdown state until the next rising edge of the CWG data input after this bit is cleared.

chanhvinh.com



### 26.14.7 CWGxAS1

**Name:** CWGxAS1
**Offset:** 0x0613

CWG Auto-Shutdown Control Register 1

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | AS7E | AS6E | AS5E | AS4E | AS3E | AS2E | AS1E | AS0E |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 0, 1, 2, 3, 4, 5, 6, 7 – ASyE** CWG Auto-Shutdown Source Enable(1,2)

| ASyE | Auto-Shutdown Source |
|---|---|
| | CWG1 |
| AS7E | CLC4_OUT |
| AS6E | CLC2_OUT |
| AS5E | C2_OUT |
| AS4E | C1_OUT |
| AS3E | TMR6_Postscaled_OUT |
| AS2E | TMR4_Postscaled_OUT |
| AS1E | TMR2_Postscaled_OUT |
| AS0E | Pin selected by CWG1PPS |

**Notes:**

1. This bit may be written while EN = 0, to place the outputs into the shutdown configuration.
2. The outputs will remain in Auto-Shutdown state until the next rising edge of the CWG data input after this bit is cleared.

chanhvinh.com

chanhvinh.com



### 26.14.8 CWGxDBR

**Name:** CWGxDBR
**Offset:** 0x060E

CWG Rising Dead-Band Count Register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | DBR[5:0] | | | | | |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | x | x | x | x | x | x |

**Bits 5:0 – DBR[5:0]** CWG Rising Edge-Triggered Dead-Band Count
Reset States: POR/BOR = xxxxxx
All Other Resets = uuuuuu

| Value | Description |
|---|---|
| n | Dead band is active no less than n and no more than n+1 CWG clock periods after the rising edge |
| 0 | 0 CWG clock periods. Dead-band generation is bypassed. |

chanhvinh.com



### 26.14.9 CWGxDBF

**Name:** CWGxDBF
**Offset:** 0x060F

CWG Falling Dead-Band Count Register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | DBF[5:0] | | | | | |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | x | x | x | x | x | x |

**Bits 5:0 – DBF[5:0]** CWG Falling Edge-Triggered Dead-Band Count
Reset States: POR/BOR = xxxxxx
All Other Resets = uuuuuu

| Value | Description |
|---|---|
| n | Dead band is active no less than n and no more than n+1 CWG clock periods after the falling edge |
| 0 | 0 CWG clock periods. Dead-band generation is bypassed. |

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



## 26.15 Register Summary - CWG

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 ... 0x060B | Reserved | | | | | | | | | |
| 0x060C | CWG1CLK | 7:0 | | | | | | | | CS |
| 0x060D | CWG1ISM | 7:0 | | | | ISM[4:0] | | | | |
| 0x060E | CWG1DBR | 7:0 | | | DBR[5:0] | | | | | |
| 0x060F | CWG1DBF | 7:0 | | | DBF[5:0] | | | | | |
| 0x0610 | CWG1CON0 | 7:0 | EN | LD | | | | MODE[2:0] | | |
| 0x0611 | CWG1CON1 | 7:0 | | | IN | | POLD | POLC | POLB | POLA |
| 0x0612 | CWG1AS0 | 7:0 | SHUTDOWN | REN | LSBD[1:0] | | LSAC[1:0] | | | |
| 0x0613 | CWG1AS1 | 7:0 | AS7E | AS6E | AS5E | AS4E | AS3E | AS2E | AS1E | AS0E |
| 0x0614 | CWG1STR | 7:0 | OVRD | OVRC | OVRB | OVRA | STRD | STRC | STRB | STRA |

chanhvinh.com

chanhvinh.com



# 27. CCP - Capture/Compare/PWM Module

The Capture/Compare/PWM module is a peripheral that 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.

Each individual CCP module can select the timer source that controls the module. The default timer selection is Timer1 when using Capture/Compare mode and Timer2 when using PWM mode in the CCPx module.

Note that the Capture/Compare mode operation is described with respect to Timer1 and the PWM mode operation is described with respect to Timer2 in the following sections.

The Capture and Compare functions are identical for all CCP modules.

**Important:** In devices with more than one CCP module, it is very important to pay close attention to the register names used. Throughout this section, the prefix “CCPx” is used as a generic replacement for specific numbering. A number placed where the “x” is in the prefix is used to distinguish between separate modules. For example, CCP1CON and CCP2CON control the same operational aspects of two completely different CCP modules.

## 27.1 CCP Module Configuration

Each Capture/Compare/PWM module is associated with a control register (CCPxCON), a capture input selection register (CCPxCAP) and a data register (CCPRx). The data register, in turn, is comprised of two 8-bit registers: CCPRxL (low byte) and CCPRxH (high byte).

### 27.1.1 CCP Modules and Timer Resources

The CCP modules utilize Timers 1 through 6 that vary with the selected mode. Various timers are available to the CCP modules in Capture, Compare or PWM modes, as shown in the table below.

**Table 27-1. CCP Mode - Timer Resources**

| CCP Mode | Timer Resource |
|---|---|
| Capture | Timer1, Timer3 |
| Compare | |
| PWM | Timer2, Timer4, Timer6 |

The assignment of a particular timer to a module is selected as shown in the **“Capture, Compare, and PWM Timers Selection”** chapter. All of the modules may be active at once and may share the same timer resource if they are configured to operate in the same mode (Capture/Compare or PWM) at the same time.

### 27.1.2 Open-Drain Output Option

When operating in Output mode (the Compare or PWM modes), the drivers for the CCPx pins can be optionally configured as open-drain outputs. This feature allows the voltage level on the pin to be pulled to a higher level through an external pull-up resistor and allows the output to communicate with external circuits without the need for additional level shifters.

## 27.2 Capture Mode

Capture mode makes use of the 16-bit odd numbered timer resources (Timer1, Timer3, etc.). When an event occurs on the capture source, the 16-bit CCPRx register captures and stores the 16-bit value of the TMRx register. An event is defined as one of the following and is configured by the MODE bits:

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com

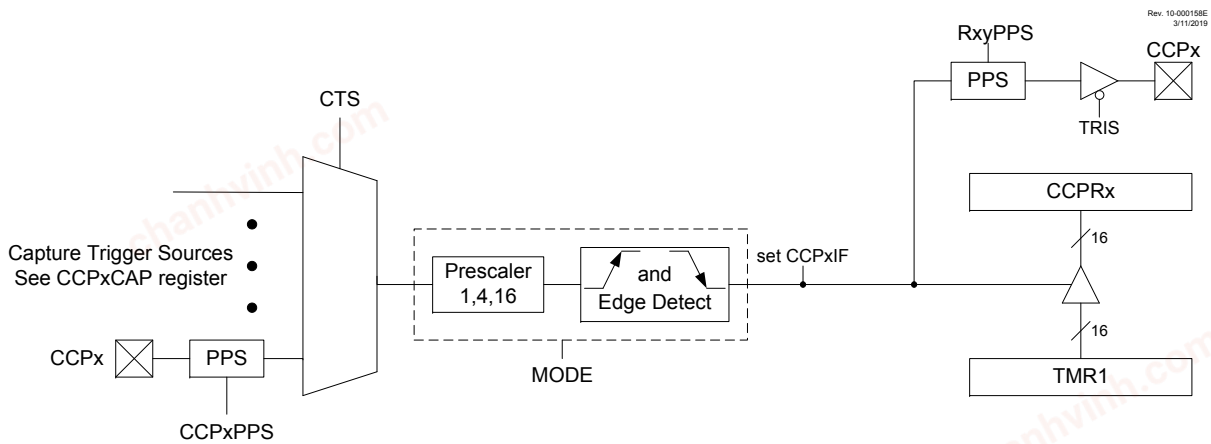


- Every falling edge of CCPx input
- Every rising edge of CCPx input
- Every 4th rising edge of CCPx input
- Every 16th rising edge of CCPx input
- Every edge of CCPx input (rising or falling)

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 CCPRx register is read, the old captured value is overwritten by the new captured value. The following figure shows a simplified diagram of the capture operation.

**Important:** If an event occurs during a 2-byte read, the high and low-byte data will be from different events. It is recommended while reading the CCPRx register pair to either disable the module or read the register pair twice for data integrity.

**Figure 27-1. Capture Mode Operation Block Diagram**




### 27.2.1 Capture Sources

The capture source is selected with the CTS bits.

In Capture mode, the CCPx pin must be configured as an input by setting the associated TRIS control bit.

**Important:** If the CCPx pin is configured as an output, a write to the port can cause a capture event.

### 27.2.2 Timer1 Mode for Capture

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 the **"TMR1 - Timer1 Module with Gate Control"** chapter for more information on configuring Timer1.

### 27.2.3 Software Interrupt Mode

When the Capture mode is changed, a false capture interrupt may be generated. The user will keep the CCPxIE Interrupt Enable bit of the PIEx register clear to avoid false interrupts. Additionally, the user will clear the CCPxIF Interrupt Flag bit of the PIRx register following any change in Operating mode.

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



**Important:** Clocking Timer1 from the system clock ($F_{OSC}$) must not be used in Capture mode. 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.

### 27.2.4 CCP Prescaler

There are four prescaler settings specified by the MODE bits. Whenever the CCP module is turned off, or the CCP module is not in Capture mode, the prescaler counter is cleared. Any Reset will clear the prescaler counter.

Switching from one capture prescaler to another does not clear the prescaler and may generate a false interrupt. To avoid this unexpected operation, turn the module off by clearing the CCPxCON register before changing the prescaler. The example below demonstrates the code to perform this function.

**Example 27-1. Changing between Capture Prescalers**

```
BANKSEL CCP1CON         ;only needed when CCP1CON is not in ACCESS space
CLRF    CCP1CON         ;Turn CCP module off
MOVLW   NEW_CAPT_PS     ;CCP ON and Prescaler select → W
MOVWF   CCP1CON         ;Load CCP1CON with this value
```

### 27.2.5 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.

## 27.3 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 odd numbered Timer resources (Timer1, Timer3, etc.). The 16-bit value of the CCPRx register is constantly compared against the 16-bit value of the TMRx register. When a match occurs, one of the following events can occur:

- Toggle the CCPx output and clear TMRx
- Toggle the CCPx output without clearing TMRx
- Set the CCPx output
- Clear the CCPx output
- Generate a Pulse output
- Generate a Pulse output and clear TMRx

The action on the pin is based on the value of the MODE control bits.

All Compare modes can generate an interrupt. When MODE = `'b0001` or `'b1011`, the CCP resets the TMRx register.

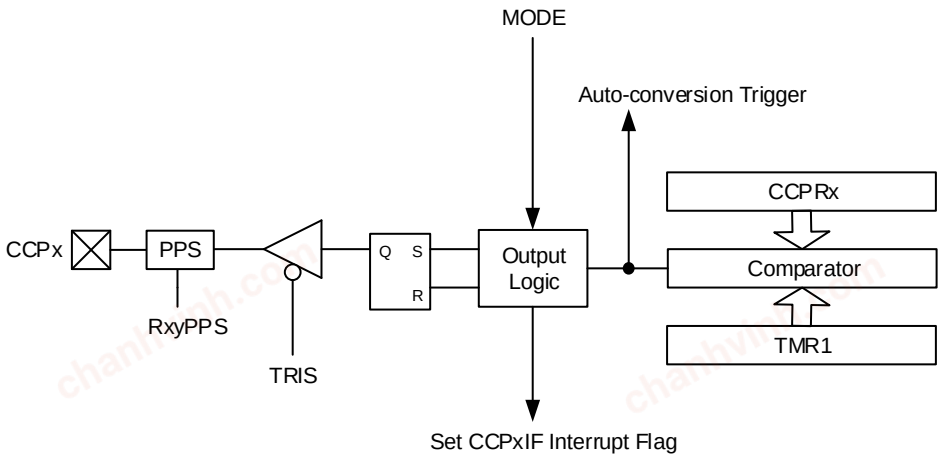
The following figure shows a simplified diagram of the compare operation.

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



**Figure 27-2. Compare Mode Operation Block Diagram**




### 27.3.1 CCPx Pin Configuration

The CCPx pin must be configured as an output in software by clearing the associated TRIS bit and defining the appropriate output pin through the RxyPPS registers. See the **"PPS - Peripheral Pin Select Module"** section for more details.

The CCP output can also be used as an input for other peripherals.

**Important:** Clearing the CCPxCON register will force the CCPx compare output latch to the default low level. This is not the PORT I/O data latch.

### 27.3.2 Timer1 Mode for Compare

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 the "**TMR1 - Timer1 Module with Gate Control**" section for more information on configuring Timer1.

**Important:** Clocking Timer1 from the system clock ($F_{OSC}$) must not be used in Compare mode. For Compare 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.

### 27.3.3 Compare During Sleep

Since $F_{OSC}$ is shut down during Sleep mode, the Compare mode will not function properly during Sleep, unless the timer is running. The device will wake on interrupt (if enabled).

## 27.4 PWM Overview

Pulse-Width Modulation (PWM) is a scheme that controls 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.

© 2023 Microchip Technology Inc. and its subsidiaries

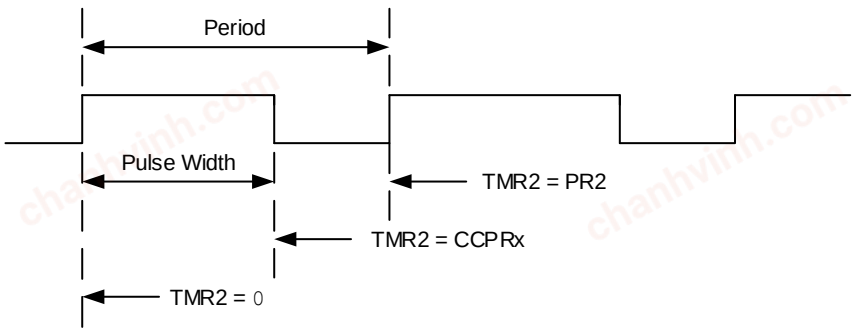
chanhvinh.com



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 power 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. The figure below shows a typical waveform of the PWM signal.

**Figure 27-3. CCP PWM Output Signal**




### 27.4.1 Standard PWM Operation

The standard PWM function described in this section is available and identical for all CCP modules. It generates a Pulse-Width Modulation (PWM) signal on the CCPx pin with up to ten bits of resolution. The period, duty cycle, and resolution are controlled by the following registers:

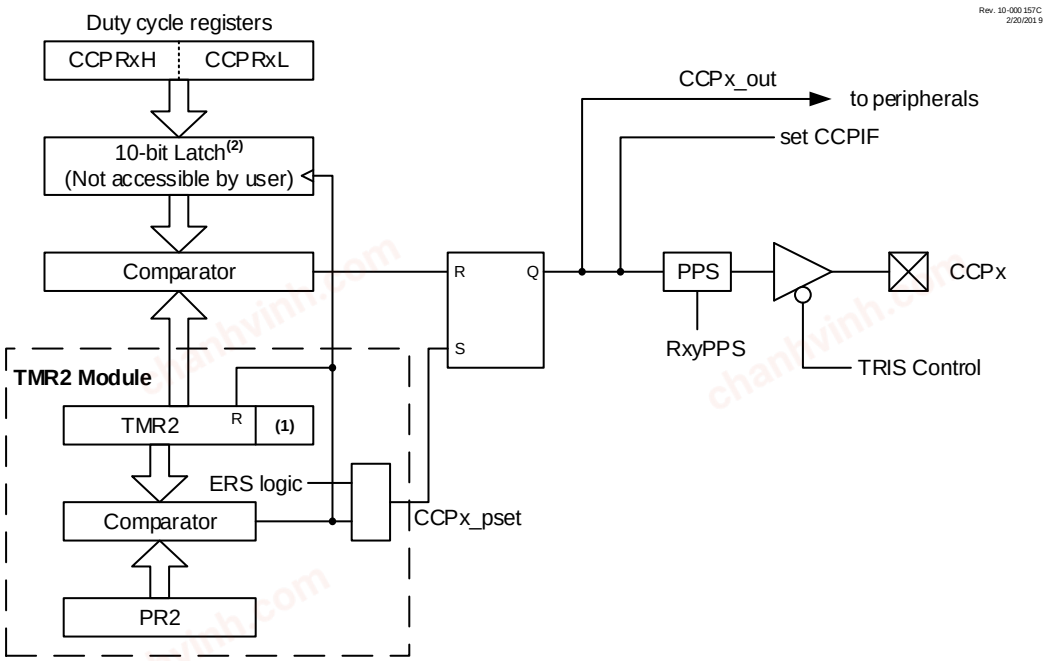
- Even numbered TxPR registers (T2PR, T4PR, etc.)
- Even numbered TxCON registers (T2CON, T4CON, etc.)
- 16-bit CCPRx registers
- CCPxCON registers

It is required to have $F_{OSC}/4$ as the clock input to TxTMR for correct PWM operation. The following figure shows a simplified block diagram of the PWM operation.

chanhvinh.com



**Figure 27-4. Simplified PWM Block Diagram**




**Notes**: 1. An 8-bit timer is concatenated with two bits generated by Fosc or two bits of the internal prescaler to create 10-bit time base.
2. The alignment of the 10 bits from the CCPR register is determined by the CCPxFMT bit.

**Important:** The corresponding TRIS bit must be cleared to enable the PWM output on the CCPx pin.

### 27.4.2 Setup for PWM Operation

The following steps illustrate how to configure the CCP module for standard PWM operation:

1. Select the desired output pin with the RxyPPS control to select CCPx as the source. Disable the selected pin output driver by setting the associated TRIS bit. The output will be enabled later at the end of the PWM setup.
2. Load the selected timer TxPR period register with the PWM period value.
3. Configure the CCP module for the PWM mode by loading the CCPxCON register with the appropriate values.
4. Load the CCPRx register with the PWM duty cycle value and configure the FMT bit to set the proper register alignment.
5. Configure and start the selected timer:
   - Clear the TMRxIF Interrupt Flag bit of the PIRx register. See the Note below.
   - Select the timer clock source to be as $F_{OSC}/4$. This is required for correct operation of the PWM module.
   - Configure the TxCKPS bits of the TxCON register with the desired timer prescale value.
   - Enable the timer by setting the TxON bit.
6. Enable the PWM output:
   - Wait until the timer overflows and the TMRxIF bit of the PIRx register is set. See the Note below.
   - Enable the CCPx pin output driver by clearing the associated TRIS bit.

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



**Important:** 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.

### 27.4.3 Timer2 Timer Resource

The PWM Standard mode makes use of the 8-bit Timer2 timer resources to specify the PWM period.

### 27.4.4 PWM Period

The PWM period is specified by the T2PR register of Timer2. The PWM period can be calculated using the formula in the equation below.

**Equation 27-1. PWM Period**

$$PWM\ Period = [(T2PR + 1)] \bullet 4 \bullet T_{OSC} \bullet (TMR2\ Prescale\ Value)$$

where $T_{OSC} = 1/F_{OSC}$

When T2TMR is equal to T2PR, the following three events occur on the next increment event:

- T2TMR 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 transferred from the CCPRx register into a 10-bit buffer

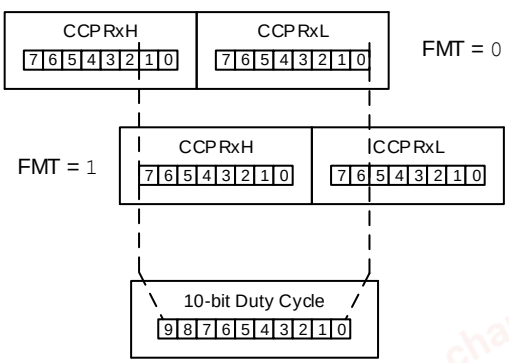
**Important:** The Timer postscaler (see the "**Timer2 Interrupt**" section) is not used in the determination of the PWM frequency.

### 27.4.5 PWM Duty Cycle

The PWM duty cycle is specified by writing a 10-bit value to the CCPRx register. The alignment of the 10-bit value is determined by the FMT bit (see Figure 27-5). The CCPRx register can be written to at any time. However, the duty cycle value is not latched onto the 10-bit buffer until after a match between T2PR and T2TMR.

The equations below are used to calculate the PWM pulse width and the PWM duty cycle ratio.

**Figure 27-5. PWM 10-Bit Alignment**




**Equation 27-2. Pulse Width**

$$Pulse\ Width = (CCPRxH{:}CCPRxL\ register\ value) \bullet T_{OSC} \bullet (TMR2\ Prescale\ Value)$$

**Equation 27-3. Duty Cycle**

$$DutyCycleRatio = \frac{(CCPRxH{:}CCPRxL\ register\ value)}{4(T2PR + 1)}$$

chanhvinh.com



The CCPRx register is used to double buffer the PWM duty cycle. This double buffering is essential for glitchless PWM operation.

The 8-bit timer T2TMR register is concatenated with either the 2-bit internal system clock ($F_{OSC}$), or two bits of the prescaler, to create the 10-bit time base. The system clock is used if the Timer2 prescaler is set to 1:1.

When the 10-bit time base matches the CCPRx register, then the CCPx pin is cleared (see Figure 27-4).

### 27.4.6 PWM Resolution

The resolution determines the number of available duty cycles for a given period. For example, a 10-bit resolution will result in 1024 discrete duty cycles, whereas an 8-bit resolution will result in 256 discrete duty cycles.

The maximum PWM resolution is 10 bits when T2PR is `0xFF`. The resolution is a function of the T2PR register value, as shown below.

**Equation 27-4. PWM Resolution**

$$Resolution = \frac{\log[4(T2PR + 1)]}{\log(2)} bits$$

**Important:** If the pulse-width value is greater than the period, the assigned PWM pin(s) will remain unchanged.

**Table 27-2. Example PWM Frequencies and Resolutions ($F_{OSC}$ = 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 |
| T2PR Value | 0xFF | 0xFF | 0xFF | 0x3F | 0x1F | 0x17 |
| Maximum Resolution (bits) | 10 | 10 | 10 | 8 | 7 | 6.6 |

**Table 27-3. Example PWM Frequencies and Resolutions ($F_{OSC}$ = 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 |
| T2PR Value | 0x65 | 0x65 | 0x65 | 0x19 | 0x0C | 0x09 |
| Maximum Resolution (bits) | 8 | 8 | 8 | 6 | 5 | 5 |

### 27.4.7 Operation in Sleep Mode

In Sleep mode, the T2TMR register will not increment and the state of the module will not change. If the CCPx pin is driving a value, it will continue to drive that value. When the device wakes up, T2TMR will continue from the previous state.

### 27.4.8 Changes in System Clock Frequency

The PWM frequency is derived from the system clock frequency. Any changes in the system clock frequency will result in changes to the PWM frequency. See the **"OSC - Oscillator Module (with Fail-Safe Clock Monitor)"** chapter for additional details.

### 27.4.9 Effects of Reset

Any Reset will force all ports to Input mode and the CCP registers to their Reset states.

## 27.5 Register Definitions: CCP Control

Long bit name prefixes for the CCP peripherals are shown in the following table. Refer to the **"Long Bit Names"** section in the **"Register and Bit Naming Conventions"** chapter for more information.

chanhvinh.com



**Table 27-4. CCP Long Bit Name Prefixes**

| Peripheral | Bit Name Prefix |
|---|---|
| CCP1 | CCP1 |
| CCP2 | CCP2 |

chanhvinh.com

chanhvinh.com



### 27.5.1 CCPxCON

**Name:** CCPxCON
**Offset:** 0x040E,0x0412

CCP Control Register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | EN | | OUT | FMT | MODE[3:0] | | | |
| Access | R/W | | R | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | | x | 0 | 0 | 0 | 0 | 0 |

**Bit 7 – EN** CCP Module Enable

| Value | Description |
|---|---|
| 1 | CCP is enabled |
| 0 | CCP is disabled |

**Bit 5 – OUT** CCP Output Data (read-only)

**Bit 4 – FMT** CCPxRH:L Value Alignment (PWM mode)

| Value | Condition | Description |
|---|---|---|
| x | Capture mode | Not used |
| x | Compare mode | Not used |
| 1 | PWM mode | Left aligned format |
| 0 | PWM mode | Right aligned format |

**Bits 3:0 – MODE[3:0]** CCP Mode Select

**Table 27-5. CCPx Mode Select**

| MODE Value | Operating Mode | Operation | Set CCPxIF |
|---|---|---|---|
| 11xx | PWM | PWM operation | Yes |
| 1011 | Compare | Pulse output; clear TMR1(2) | Yes |
| 1010 | | Pulse output | Yes |
| 1001 | | Clear output(1) | Yes |
| 1000 | | Set output(1) | Yes |
| 0111 | Capture | Every 16th rising edge of CCPx input | Yes |
| 0110 | | Every 4th rising edge of CCPx input | Yes |
| 0101 | | Every rising edge of CCPx input | Yes |
| 0100 | | Every falling edge of CCPx input | Yes |
| 0011 | | Every edge of CCPx input | Yes |
| 0010 | Compare | Toggle output | Yes |
| 0001 | | Toggle output; clear TMR1(2) | Yes |
| 0000 | Disabled | | — |

**Notes:**

1. The set and clear operations of the Compare mode are reset by setting MODE = 'b0000 or EN = 0.
2. When MODE = 'b0001 or 'b1011, then the timer associated with the CCP module is cleared. TMR1 is the default selection for the CCP module, so it is used for indication purposes only.

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



### 27.5.2 CCPxCAP

**Name:** CCPxCAP
**Offset:** 0x040F,0x0413

Capture Trigger Input Selection Register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | CTS[3:0] | | | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

**Bits 3:0 – CTS[3:0]** Capture Trigger Input Selection

**Table 27-6. Capture Trigger Sources**

| CTS Value | Source |
|---|---|
| 1111-1000 | Reserved |
| 0111 | CLC4_OUT |
| 0110 | CLC3_OUT |
| 0101 | CLC2_OUT |
| 0100 | CLC1_OUT |
| 0011 | IOC Interrupt |
| 0010 | C2_OUT |
| 0001 | C1_OUT |
| 0000 | Pin selected by CCPxPPS |

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



### 27.5.3 CCPRx

**Name:** CCPRx
**Offset:** 0x040C,0x0410

Capture/Compare/Pulse-Width Register

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | CCPR[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | x | x | x | x | x | x | x | x |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CCPR[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | x | x | x | x | x | x | x | x |

**Bits 15:0 – CCPR[15:0]** Capture/Compare/Pulse-Width
Reset States: POR/BOR = xxxxxxxxxxxxxxxx
All other Resets = uuuuuuuuuuuuuuuu

**Notes:** The individual bytes in this multibyte register can be accessed with the following register names:

- When MODE = Capture or Compare
  - CCPRxH: Accesses the high byte CCPR[15:8]
  - CCPRxL: Accesses the low byte CCPR[7:0]
- When MODE = PWM and FMT = 0
  - CCPRx[15:10]: Not used
  - CCPRxH[1:0]: Accesses the two Most Significant bits CCPR[9:8]
  - CCPRxL: Accesses the eight Least Significant bits CCPR[7:0]
- When MODE = PWM and FMT = 1
  - CCPRxH: Accesses the eight Most Significant bits CCPR[9:2]
  - CCPRxL[7:6]: Accesses the two Least Significant bits CCPR[1:0]
  - CCPRx[5:0]: Not used

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



## 27.6 Register Summary - CCP Control

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 ... 0x040B | Reserved | | | | | | | | | |
| 0x040C | CCPR1 | 7:0 | CCPR[7:0] | | | | | | | |
| | | 15:8 | CCPR[15:8] | | | | | | | |
| 0x040E | CCP1CON | 7:0 | EN | | OUT | FMT | MODE[3:0] | | | |
| 0x040F | CCP1CAP | 7:0 | | | | | CTS[3:0] | | | |
| 0x0410 | CCPR2 | 7:0 | CCPR[7:0] | | | | | | | |
| | | 15:8 | CCPR[15:8] | | | | | | | |
| 0x0412 | CCP2CON | 7:0 | EN | | OUT | FMT | MODE[3:0] | | | |
| 0x0413 | CCP2CAP | 7:0 | | | | | CTS[3:0] | | | |

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



# 28. Capture, Compare, and PWM Timers Selection

Each of these modules has an independent timer selection which can be accessed using the timer selection register. The default timer selection is Timer1 for capture or compare functions and Timer2 for PWM functions.

## 28.1 Register Definitions: Capture, Compare, and PWM Timers Selection

chanhvinh.com

chanhvinh.com



### 28.1.1 CCPTMRS0

**Name:** CCPTMRS0
**Offset:** 0x041F

CCP Timers Selection Register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | C2TSEL[1:0] | | C1TSEL[1:0] | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 1 | 0 | 1 |

**Bits 0:1, 2:3 – CnTSEL** CCPn Timer Selection

| CnTSEL Value | Capture/Compare | PWM |
|---|---|---|
| 11 | Reserved | Timer6 |
| 10 | Timer3 | Timer4 |
| 01 | Timer1 | Timer2 |
| 00 | Reserved | |

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



## 28.2 Register Summary - Capture, Compare, and PWM Timers Selection

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 ... 0x041E | Reserved | | | | | | | | | |
| 0x041F | CCPTMRS0 | 7:0 | | | | | C2TSEL[1:0] | | C1TSEL[1:0] | |

chanhvinh.com

chanhvinh.com



# 29. PWM - Pulse-Width Modulator with Compare

This module is a 16-bit Pulse-Width Modulator (PWM) with a compare feature and multiple outputs. The outputs are grouped in slices where each slice has two outputs. There can be up to four slices in each PWM module. The EN bit enables the PWM operation for all slices simultaneously. The prescale counter, postscale counter, and all internal logic is held in Reset while the EN bit is low.

Features of this module include the following:

- Five main operating modes:
  - Left Aligned
  - Right Aligned
  - Center-Aligned
  - Variable Aligned
  - Compare
    - Pulsed
    - Toggled
- Push-pull operation (available in Left and Right Aligned modes only)
- Independent 16-bit period timer
- Programmable clock sources
- Programmable trigger sources for synchronous duty cycle and period changes
- Programmable synchronous/asynchronous Reset sources
- Programmable Reset source polarity control
- Programmable PWM output polarity control
- Up to four two-output slices per module

Block diagrams of each PWM mode are shown in their respective sections.

## 29.1 Output Slices

A PWM module can have up to four output slices. An output slice consists of two PWM outputs, PWMx_SaP1_out and PWMx_SaP2_out. Both share the same operating mode. However, other slices may operate in a different mode. PWMx_SaP1_out and PWMx_SaP2_out have independent duty cycles which are set with the respective P1 and P2 parameter registers.

### 29.1.1 Output Polarity

The polarity for the PWMx_SaP1_out and PWMx_SaP2_out is controlled with the respective POL1 and POL2 bits. Setting the polarity bit inverts the output Active state to Low True. Toggling the polarity bit toggles the output whether or not the PWM module is enabled.

### 29.1.2 Operating Modes

Each output slice can operate in one of six modes selected with the MODE bits. The Left and Right Aligned modes can also be operated in Push-Pull mode by setting the PPEN bit. The following sections provide more details on each mode, including block diagrams.

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



### 29.1.2.1 Left Aligned Mode

In Left Aligned mode, the active part of the duty cycle is at the beginning of the period. The outputs start active and stay active for the number of prescaled PWM clock periods specified by the P1 and P2 parameter registers, then go inactive for the remainder of the period. Block and timing diagrams follow.

**Figure 29-1. Left-Aligned Block Diagram**

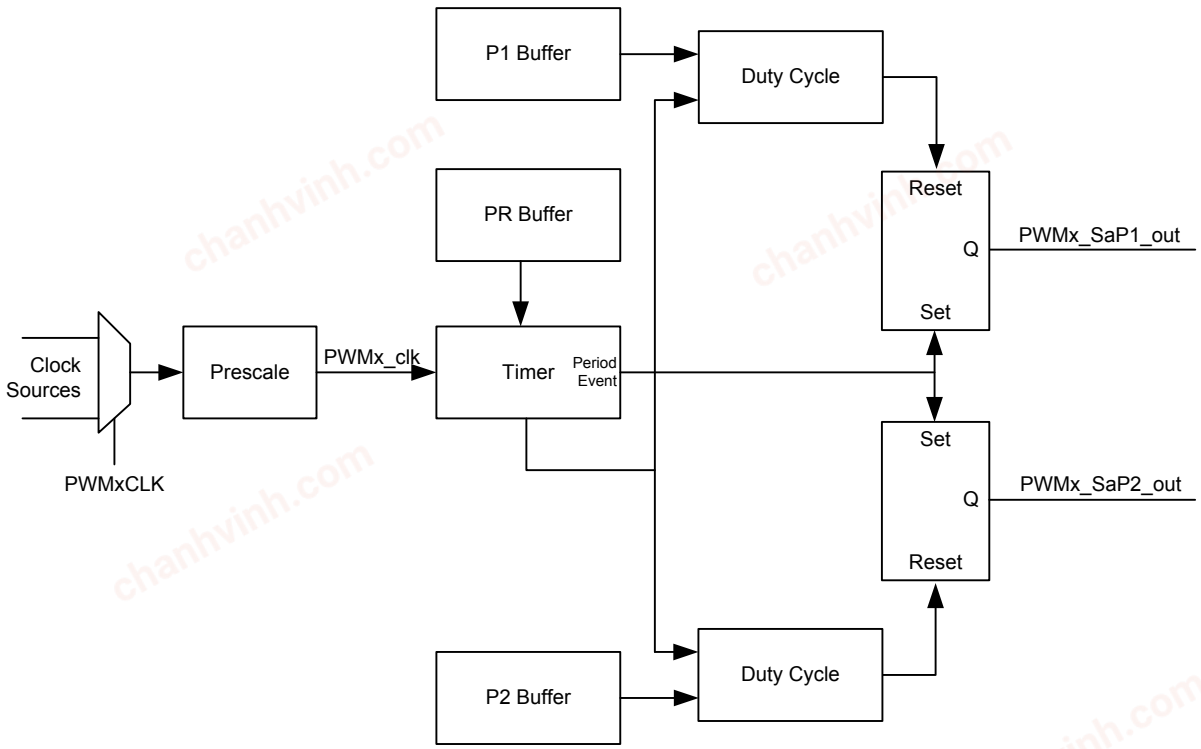
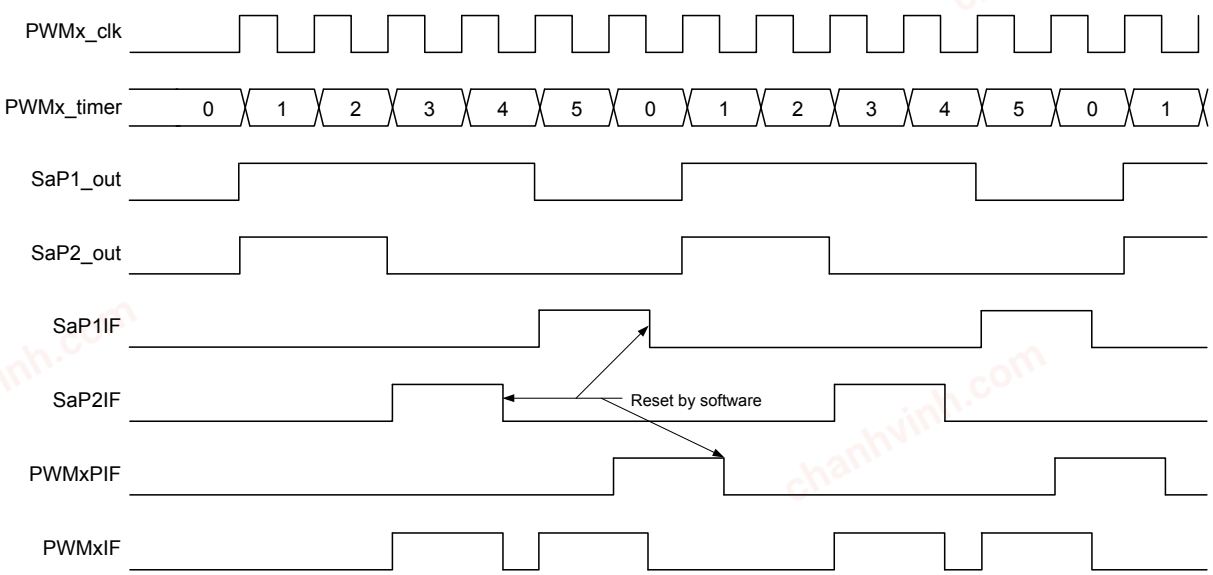



**Figure 29-2. Left-Aligned Timing Diagram**




**Note:** MODE = `'b000`, PR = 5, P1 = 4, P2 = 2.

chanhvinh.com



### 29.1.2.2 Right Aligned Mode

In Right Aligned mode, the active part of the duty cycle is at the end of the period. The outputs start in the Inactive state and then go Active the number of prescaled PWM clock periods specified by the P1 and P2 parameter registers before the end of the period. Block and timing diagrams follow.

**Figure 29-3. Right-Aligned Block Diagram**

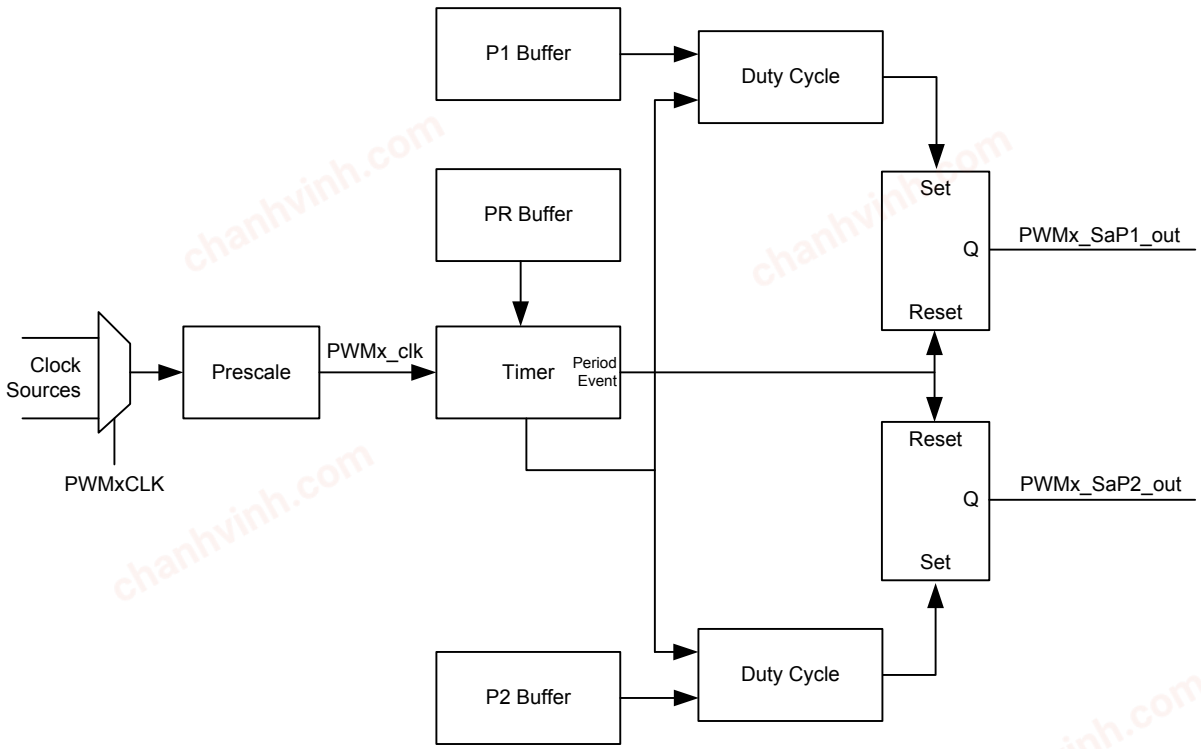
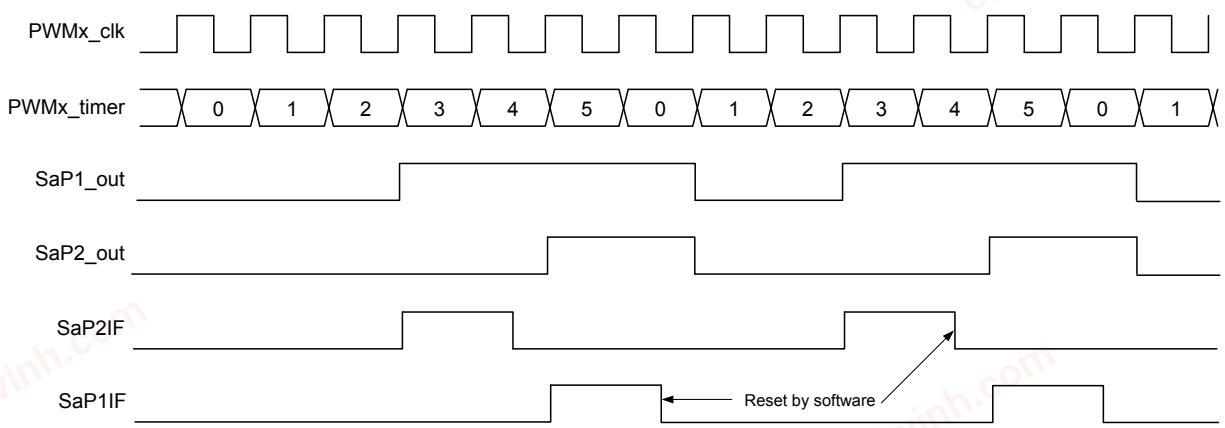



**Figure 29-4. Right-Aligned Timing Diagram**




**Note:** MODE = `'b001`, PR = 5, P1 = 4, P2 = 2.

chanhvinh.com



### 29.1.2.3 Center-Aligned Mode

In Center-Aligned mode, the active duty cycle is centered in the period. The period for this mode is twice that of other modes, as shown in the following equation.

**Equation 29-1. Center-Aligned Period**

$$Period = \frac{(PR + 1) \times 2}{F_{PWMx_clk}}$$

The parameter register specifies the number of PWM clock periods that the output goes Active before the period center. The output goes inactive the same number of prescaled PWM clock periods after the period center. Block and timing diagrams follow.

**Figure 29-5. Center-Aligned Block Diagram**

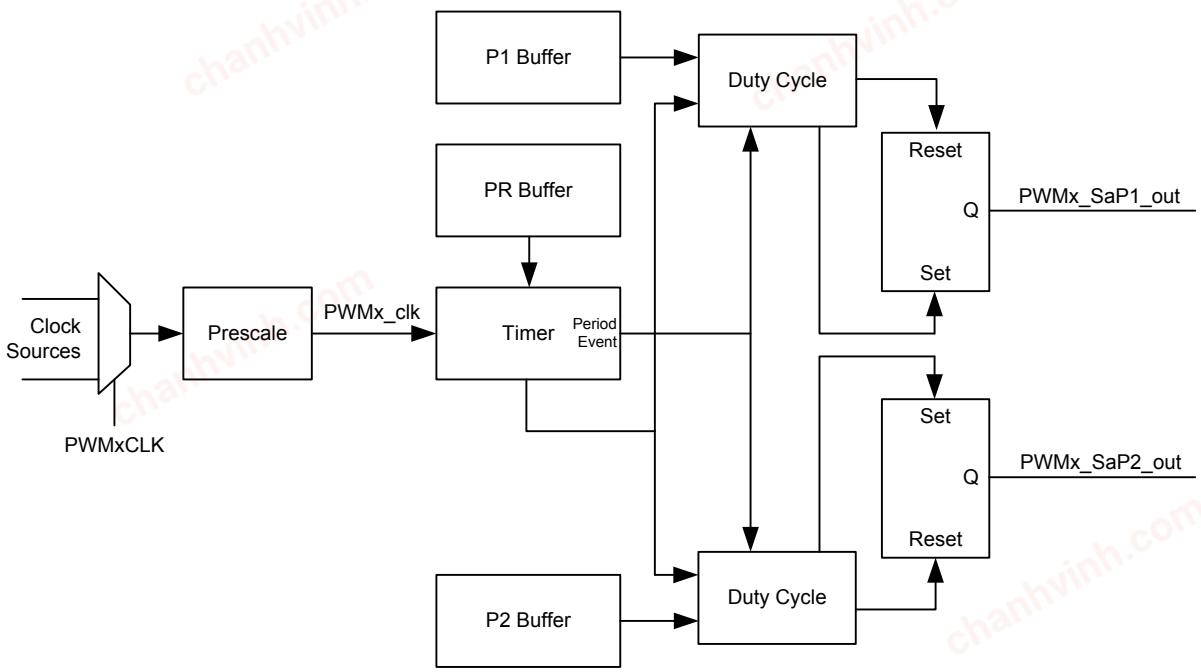
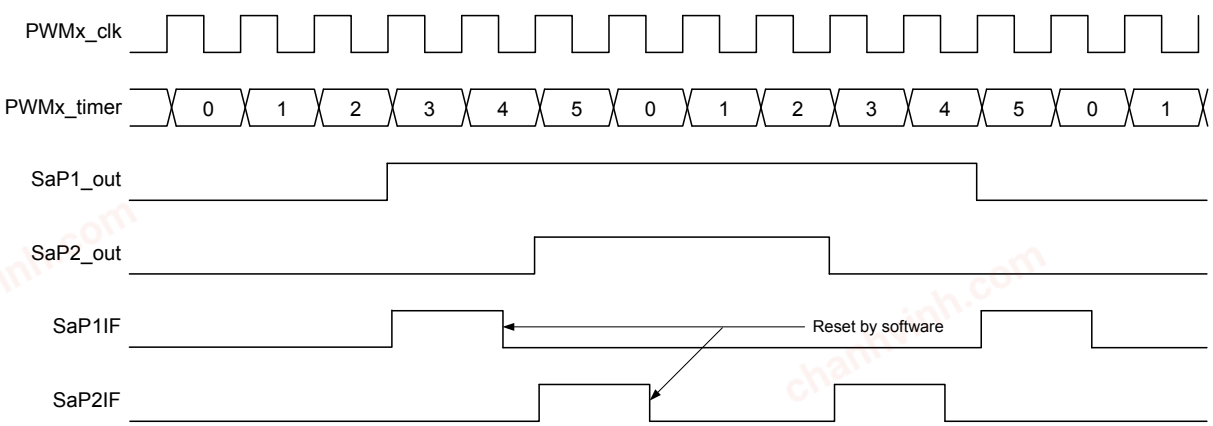



**Figure 29-6. Center-Aligned Timing Diagram**




**Note:** MODE = `'b010`, PR = 5, P1 = 4, P2 = 2.

chanhvinh.com



### 29.1.2.4 Variable Alignment Mode

In Variable Alignment mode, the active part of the duty cycle starts when the parameter 1 value (P1) matches the timer and ends when the parameter 2 value (P2) matches the timer. Both outputs are identical because both parameter values are used for the same duty cycle. Block and timing diagrams follow.

**Figure 29-7. Variable Alignment Block Diagram**

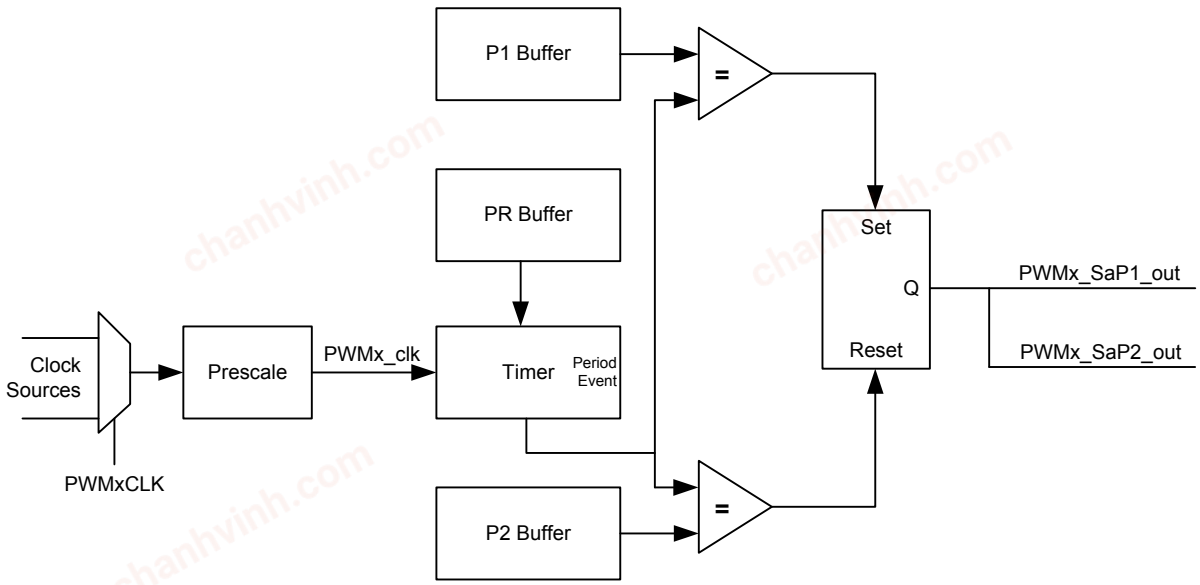
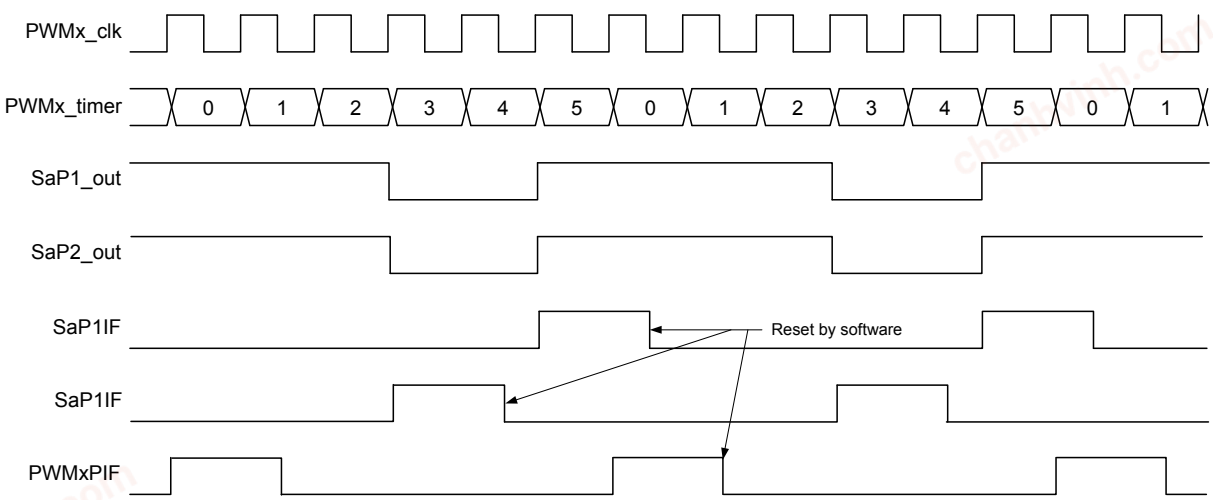



**Figure 29-8. Variable Alignment Timing Diagram**




**Note:** MODE = `'b011`, PR = 5, P1 = 4, P2 = 2.

chanhvinh.com



### 29.1.2.5 Compare Modes

In the Compare modes, the PWM timer is compared to the P1 and P2 parameter values. When a match occurs, the output is either pulsed or toggled. In Pulsed Compare mode, the duty cycle is always one prescaled PWM clock period. In Toggle Compare mode, the duty cycle is always one full PWM period. Refer to the following sections for more details.

#### 29.1.2.5.1 Pulsed Compare Mode

In Pulsed Compare mode, the duty cycle is one prescaled PWM clock period that starts when the timer matches the parameter value and ends one prescaled PWM clock period later. The outputs start in the Inactive state and then go Active during the duty cycle. Block and timing diagrams follow.

**Figure 29-9. Pulsed Compare Block Diagram**

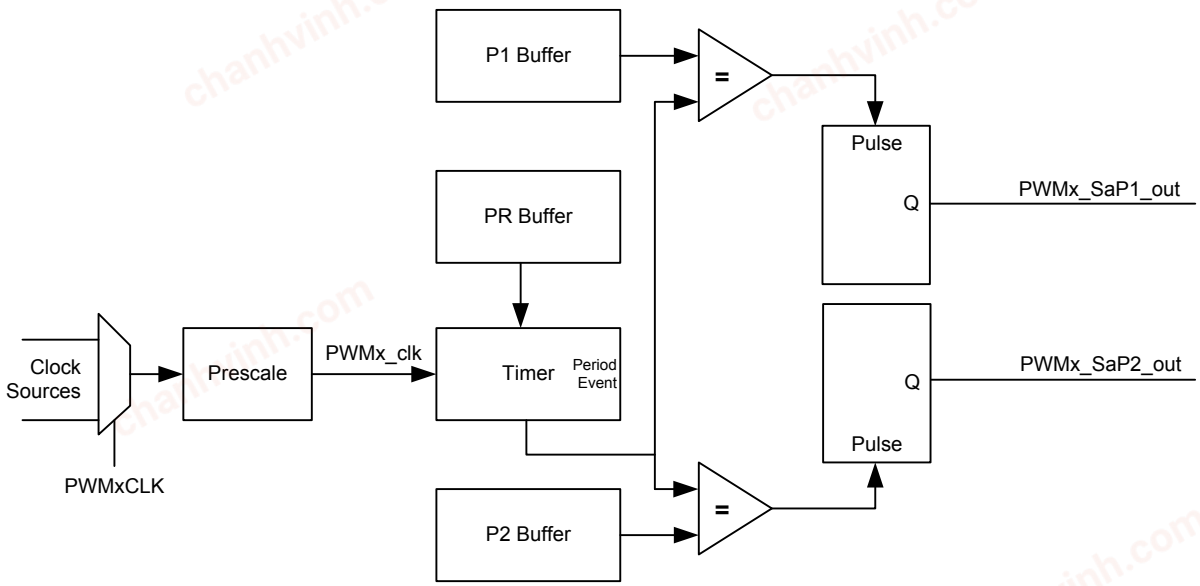
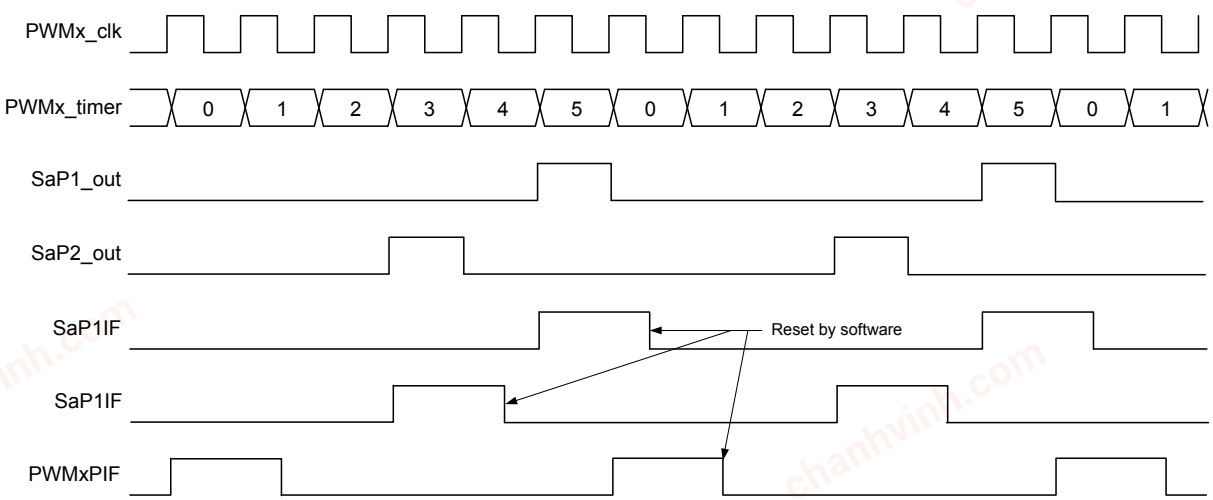



**Figure 29-10. Pulsed Compare Timing Diagram**




**Note:** MODE = `'b100`, PR = 5, P1 = 4, P2 = 2.

chanhvinh.com



#### 29.1.2.5.2 Toggled Compare

In Toggled Compare mode, the duty cycle is alternating full PWM periods. The output goes Active when the PWM timer matches the P1 or P2 parameter value and goes Inactive in the next period at the same match point. Block and timing diagrams follow.

**Figure 29-11. Toggled Compare Block Diagram**

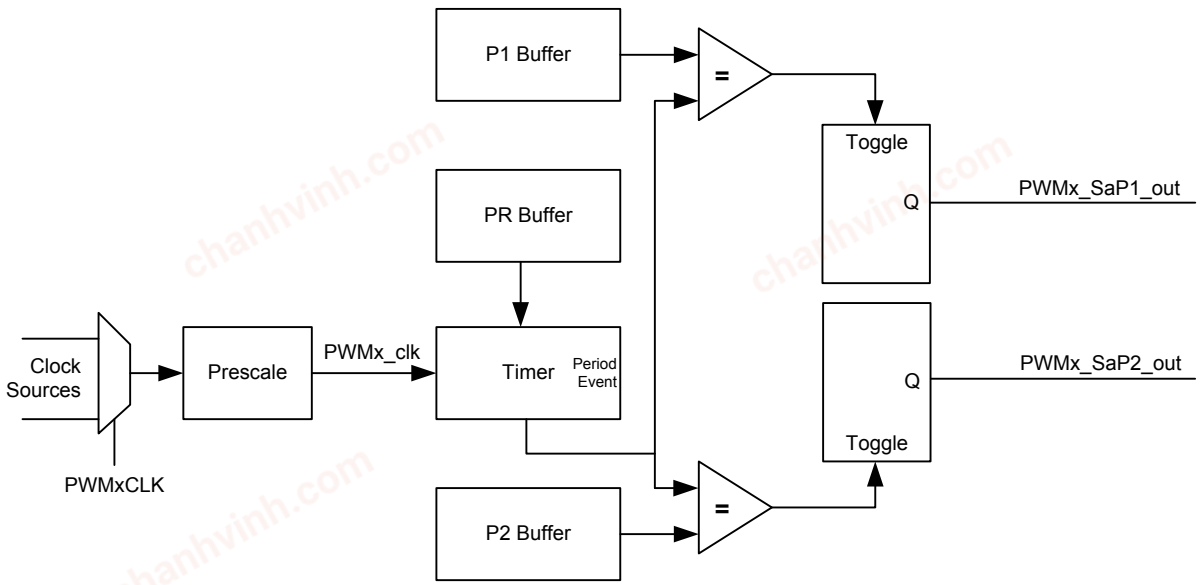
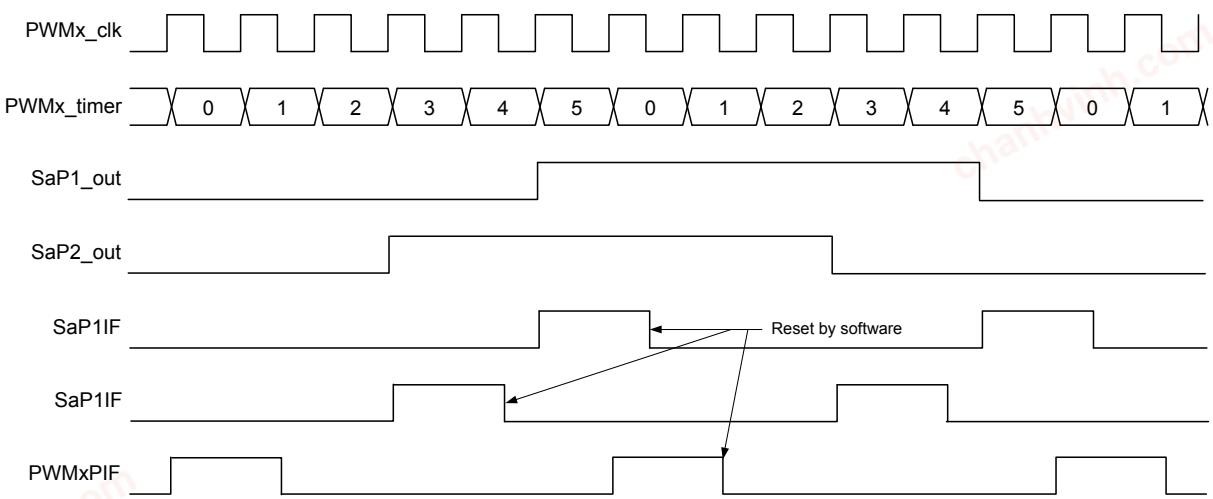



**Figure 29-12. Toggled Compare Timing Diagram**




**Note:** MODE = `'b101`, PR = 5, P1 = 4, P2 = 2.

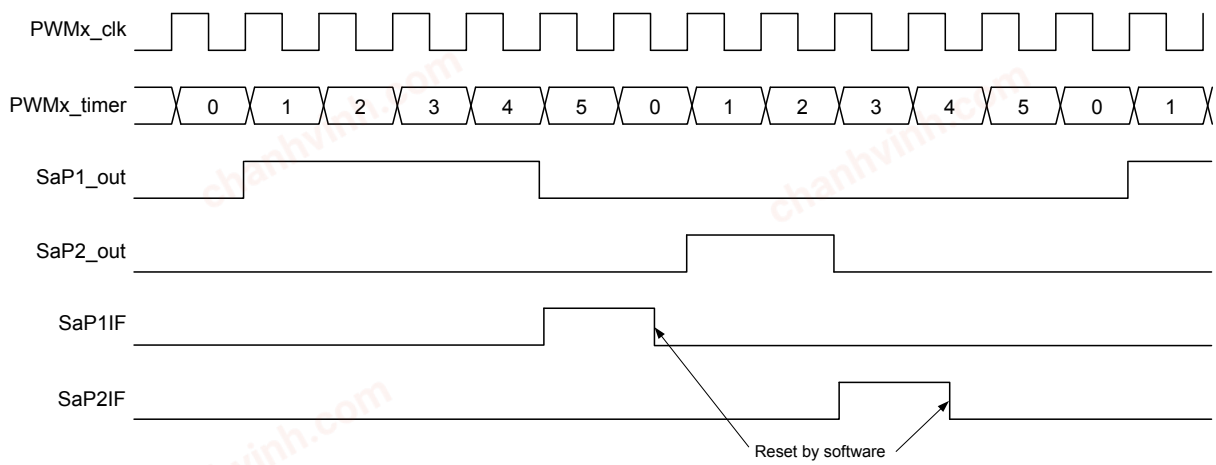
chanhvinh.com



### 29.1.3 Push-Pull Mode

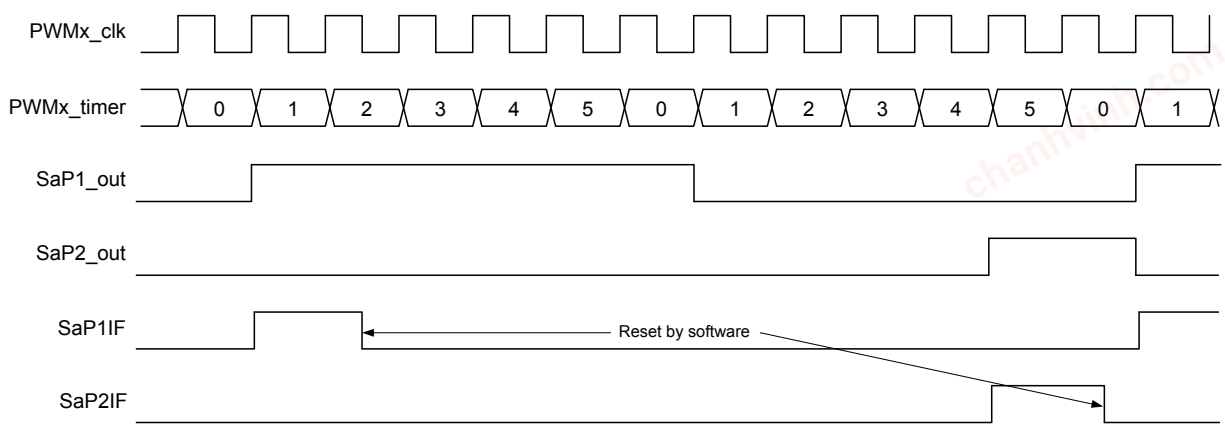
The Push-Pull mode is enabled by setting the PPEN bit. Push-Pull operates only in the Left Aligned and Right Aligned modes. In the Push-Pull mode, the outputs are Active every other PWM period. PWMx_SaP1_out is Active when the PWMx_SaP2_out is not and the PWMx_SaP2_out is Active when the PWMx_SaP1_out is not. When the parameter value (P1 or P2) is greater than the period value (PR), then the corresponding output is Active for one full PWM period. The following figures illustrate timing examples of Left and Right Aligned Push-Pull modes.

**Figure 29-13. Left Aligned Push-Pull Mode Timing Diagram**




**Note:** MODE = `'b000`, PR = 5, P1 = 4, P2 = 2, PPEN = `1`.

**Figure 29-14. Right Aligned Push-Pull Mode Timing Diagram**




**Note:** MODE = `'b001`, PR = 5, P1 = 6, P2 = 2, PPEN = `1`.

## 29.2 Period Timer

All slices in a PWM instance operate with the same period. The value written to the PWMxPR register is one less than the number of prescaled PWM clock periods (PWM_clk) in the PWM period.

The PWMxPR register is double-buffered. When the PWM is operating, writes to the PWMxPR register are transferred to the period buffer only after the LD bit is set or an external load event occurs. The transfer occurs at the next period Reset event. If the LD bit is set less than three PWM clock periods before the end of the period, then the transfer may be one full period later.

Loading the buffers of multiple PWM instances can be coordinated using the PWMLOAD register. See the Buffered Period and Parameter Registers section for more details.

chanhvinh.com



## 29.3 Clock Sources

The time base for the PWM period prescaler is selected with the CLK bits. Changes take effect immediately when written. Clearing the EN bit before making clock source changes is recommended to avoid unexpected behavior.

### 29.3.1 Clock Prescaler

The PWM clock frequency can be reduced with the clock prescaler. There are 256 prescale selections from 1:1 to 1:256.

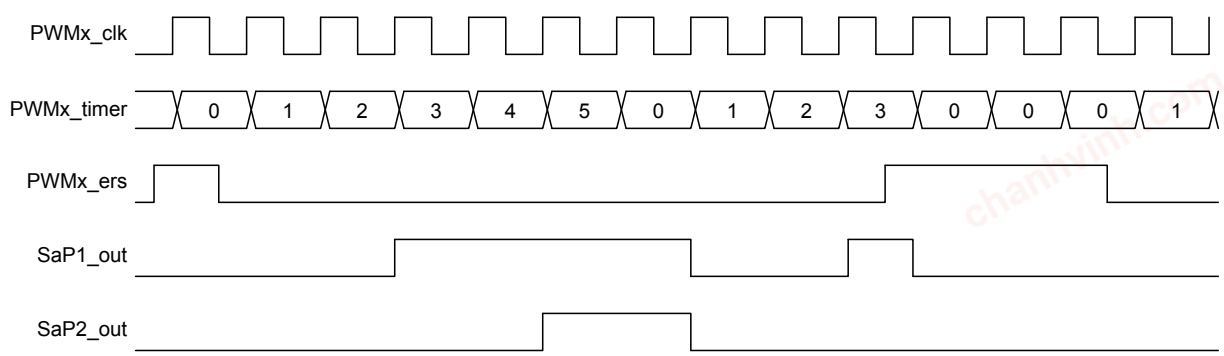
The CPRE bits select the prescale value. Changes to the prescale value take effect immediately. Clearing the EN bit before making prescaler changes is recommended to avoid unexpected behavior. The prescale counter is reset when the EN bit is cleared.

## 29.4 External Period Resets

The period timer can be reset and held at zero by a logic level from one of various sources. The Reset event also resets the postscaler counter. The resetting source is selected with the ERS bits.

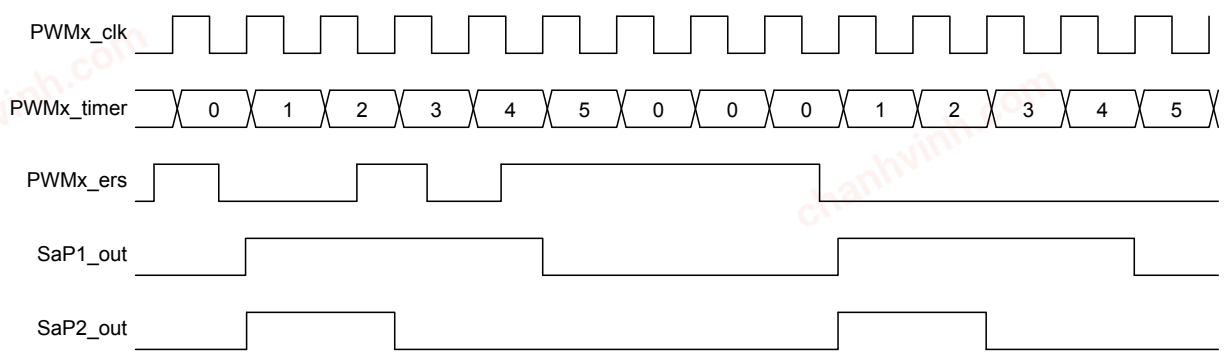
The Reset can be configured with the ERSNOW bit to occur on either the next PWM clock or the next PWM period Reset event. When the ERSNOW bit is set, then the Reset will occur on the next PWM clock. When the ERSNOW bit is cleared, then the Reset will be held off until the timer resets at the end of the period. The difference between a normal period Reset and an ERS Reset is that once the timer is reset, it is held at zero until the ERS signal goes false. The following timing diagrams illustrate the two types of external Reset.

**Figure 29-15. Right Aligned Mode with ERSNOW = 1**




**Note:** PR = 5, P1 = 4, P2 = 2.

**Figure 29-16. Left Aligned Mode with ERSNOW = 0**




**Note:** PR = 5, P1 = 4, P2 = 2.

chanhvinh.com



## 29.5 Buffered Period and Parameter Registers

The PWMxPR, PWMxSaP1 and PWMxSaP2 registers are double-buffered. The PWM module operates on the buffered copies. The values in all these registers are copied to the buffer registers when the PWM module is enabled.

Changes to the PWMxPR, PWMxSaP1 and PWMxSaP2 registers do not affect the buffer registers while the PWM is operating until either software sets the LD bit or an external load event occurs. For all operating modes except Center-Aligned, the values are copied to the buffer registers when the PWM timer is reloaded at the end of the period in which the load request occurred. In the Center-Aligned mode, the buffer update occurs on every other period Reset event because one full center-aligned period uses two period cycles. Load requests occurring three or less clocks before the end of the period may not be serviced until the following period.

A list of external load trigger sources is shown in the PWMxLDS register. Software can set the LD bits of multiple PWM instances simultaneously with the PWMLOAD register.

**Important:** No changes are allowed after the LD bit is set until after the LD bit is cleared by hardware. Unexpected behavior may result if the LD bit is cleared by software.

## 29.6 Synchronizing Multiple PWMs

To synchronize multiple PWMs, the PWMEN register is used to enable selected PWMs simultaneously. The bits in the PWMEN register are mirror copies of the EN bit of every PWM in the device. Setting or clearing the EN bits in the PWMEN register enables or disables all the corresponding PWMs simultaneously.

## 29.7 Interrupts

Each PWM instance has a period interrupt and interrupts associated with the mode and parameter settings.

### 29.7.1 Period Interrupt

The period interrupt occurs when the PWMx timer value matches the PR value, thereby also resetting the PWMx timer. Refer to Figure 29-2 for a timing example. The period interrupt is indicated with the PWMxPIF flag bit in one of the PIR registers and is set whether or not the interrupt is enabled. This flag must be reset by software. The PWMxPIF interrupt is enabled with the PWMxPIE bit in the corresponding PIE register.

#### 29.7.1.1 Period Interrupt Postscaler

The frequency of the period interrupt events can be reduced with the period interrupt postscaler. A postscaler counter suppresses period interrupts until the postscale count is reached. Only one PWM period interrupt is generated for every postscale counts. There are 256 postscale selections from 1:1 to 1:256.

The PIPOS bits select the postscale value. Changes to the postscale value take effect immediately. Clearing the EN bit before making postscaler changes is recommended to avoid unexpected behavior. The postscale counter is reset when the EN bit is cleared.

### 29.7.2 Parameter Interrupts

The P1 and P2 parameters in each slice have interrupts that occur depending on the selected mode. The individual parameter interrupts are indicated in the PWMxGIR register and enabled by the corresponding bits in the PWMxGIE register.

A timing example is shown in Figure 29-2. Refer to the timing diagrams of each of the other modes for more details.

All the enabled PWMxGIR interrupts of one PMW instance are OR'd together into the PWMxIF bit in one of the PIR registers. The PWMxIF bit is read-only. When any of the PWMxGIR bits are set then the PWMxIF bit is true. All PWMxGIF flags must be reset to clear the PWMxIF bit. The PWMxIF interrupt is enabled with the PWMxIE bit in the corresponding PIE register.

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



## 29.8 Operation During Sleep

The PWM module operates in Sleep only if the PWM clock is Active. Some internal clock sources are automatically enabled to operate in Sleep when a peripheral using them is enabled. Those clock sources are identified in the clock source table shown in the PWMxCLK clock source selection register.

## 29.9 Register Definitions: PWM Control

Long bit name prefixes for the PWM peripherals are shown in the table below. Refer to the **"Long Bit Names"** section in the **"Register and Bit Naming Conventions"** chapter for more information.

**Table 29-1. PWM Bit Name Prefixes**

| Peripheral | Bit Name Prefix |
|---|---|
| PWM1 | PWM1 |
| PWM2 | PWM2 |
| PWM3 | PWM3 |
| PWM4 | PWM4 |

chanhvinh.com

chanhvinh.com



### 29.9.1 PWMxERS

**Name:** PWMxERS
**Offset:** 0x048C,0x050C,0x098C,0x0A0C

PWMx External Reset Source

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | ERS[3:0] | | | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

**Bits 3:0 – ERS[3:0]** External Reset Source Select

| ERS | Reset Source | | | |
|---|---|---|---|---|
| | PWM1 | PWM2 | PWM3 | PWM4 |
| 1111-1110 | Reserved (ERS Disabled) | | | |
| 1101 | CLC4_OUT | | | |
| 1100 | CLC3_OUT | | | |
| 1011 | CLC2_OUT | | | |
| 1010 | CLC1_OUT | | | |
| 1001 | PWM4S1P2_OUT | | | Reserved (ERS Disabled) |
| 1000 | PWM4S1P1_OUT | | | Reserved (ERS Disabled) |
| 0111 | PWM3S1P2_OUT | | Reserved (ERS Disabled) | PWM3S1P2_OUT |
| 0110 | PWM3S1P1_OUT | | Reserved (ERS Disabled) | PWM3S1P1_OUT |
| 0101 | PWM2S1P2_OUT | Reserved (ERS Disabled) | PWM2S1P2_OUT | |
| 0100 | PWM2S1P1_OUT | Reserved (ERS Disabled) | PWM2S1P1_OUT | |
| 0011 | Reserved (ERS Disabled) | PWM1S1P2_OUT | | |
| 0010 | Reserved (ERS Disabled) | PWM1S1P1_OUT | | |
| 0001 | PWM1ERSPPS | PWM2ERSPPS | PWM3ERSPPS | PWM4ERSPPS |
| 0000 | ERS Disabled | | | |

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



### 29.9.2 PWMxCLK

**Name:** PWMxCLK
**Offset:** 0x048D,0x050D,0x098D,0x0A0D

PWMx Clock Source

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | CLK[3:0] | | | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

**Bits 3:0 – CLK[3:0]** PWM Clock Source Select

| CLK | Source | Operates in Sleep |
|---|---|---|
| 1111 | Reserved | N/A |
| 1110 | CLC4_OUT | Yes(1) |
| 1101 | CLC3_OUT | Yes(1) |
| 1100 | CLC2_OUT | Yes(1) |
| 1011 | CLC1_OUT | Yes(1) |
| 1010 | NCO1_OUT | Yes(1) |
| 1001 | CLKR_OUT | Yes(1) |
| 1000 | EXTOSC | Yes |
| 0111 | SOSC | Yes |
| 0110 | MFINTOSC (32 kHz) | Yes |
| 0101 | MFINTOSC (500 kHz) | Yes |
| 0100 | LFINTOSC | Yes |
| 0011 | HFINTOSC | Yes |
| 0010 | $F_{OSC}$ | No |
| 0001 | PWMIN1PPS | Yes(1) |
| 0000 | PWMIN0PPS | Yes(1) |

**Note:** Operation during Sleep is possible if the clock supplying the source peripheral operates in Sleep.

© 2023 Microchip Technology Inc.
and its subsidiaries

chanhvinh.com



### 29.9.3 PWMxLDS

**Name:** PWMxLDS
**Offset:** 0x048E,0x050E,0x098E,0x0A0E

PWMx Auto-load Trigger Source Select Register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | LDS[2:0] | | |
| Access | | | | | | R/W | R/W | R/W |
| Reset | | | | | | 0 | 0 | 0 |

**Bits 2:0 – LDS[2:0]** Auto-load Trigger Source Select

| LDS | Source |
|---|---|
| 111 | Reserved |
| 110 | CLC4_OUT |
| 101 | CLC3_OUT |
| 100 | CLC2_OUT |
| 011 | CLC1_OUT |
| 010 | PWMIN1PPS |
| 001 | PWMIN0PPS |
| 000 | Auto-load Disabled |

chanhvinh.com

chanhvinh.com



### 29.9.4 PWMxPR

**Name:** PWMxPR
**Offset:** 0x048F,0x050F,0x098F,0x0A0F

PWMx Period Register

Determines the PWMx period

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | PR[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PR[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 15:0 – PR[15:0]** PWM Period
Number of PWM clocks periods in the PWM period

**Notes:** The individual bytes in this multibyte register can be accessed with the following register names:

- PWMxPRH: Accesses the high byte PR[15:8]
- PWMxPRL: Accesses the low byte PR[7:0]

chanhvinh.com



### 29.9.5 PWMxCPRE

**Name:** PWMxCPRE
**Offset:** 0x0491,0x0511,0x0991,0x0A11

PWMx Clock Prescaler Register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CPRE[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 7:0 – CPRE[7:0]** PWM Clock Prescale Value

| Value | Description |
|---|---|
| n | PWM clock is prescaled by n+1 |

chanhvinh.com



### 29.9.6 PWMxPIPOS

**Name:** PWMxPIPOS
**Offset:** 0x0492,0x0512,0x0992,0x0A12

PWMx Period Interrupt Postscaler Register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PIPOS[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 7:0 – PIPOS[7:0]** Period Interrupt Postscale Value

| Value | Description |
|---|---|
| n | Period interrupt occurs after n+1 period events |

chanhvinh.com



### 29.9.7 PWMxGIR

**Name:** PWMxGIR
**Offset:** 0x0493,0x0513,0x0993,0x0A13

PWMx Interrupt Register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | S1P2 | S1P1 |
| Access | | | | | | | R/W/HS | R/W/HS |
| Reset | | | | | | | 0 | 0 |

**Bit 1 – SaP2** Slice “a” Parameter 2 Interrupt Flag

| Value | Mode | Description |
|---|---|---|
| 1 | Variable Aligned or Compare | Compare match between P2 and PWM counter has occurred |
| 1 | Center-Aligned | PWMx_SaP2_out has changed |
| 1 | Right Aligned | Left edge of PWMx_SaP2_out pulse has occurred |
| 1 | Left Aligned | Right edge of PWMx_SaP2_out pulse has occurred |
| 0 | All | Interrupt event has not occurred |

**Bit 0 – SaP1** Slice “a” Parameter 1 Interrupt Flag

| Value | Mode | Description |
|---|---|---|
| 1 | Variable Aligned or Compare | Compare match between P1 and PWM counter has occurred |
| 1 | Center-Aligned | PWMx_SaP1_out has changed |
| 1 | Right Aligned | Left edge of PWMx_SaP1_out pulse has occurred |
| 1 | Left Aligned | Right edge of PWMx_SaP1_out pulse has occurred |
| 0 | All | Interrupt event has not occurred |

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



### 29.9.8 PWMxGIE

**Name:** PWMxGIE
**Offset:** 0x0494,0x0514,0x0994,0x0A14

PWMx Interrupt Enable Register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | S1P2 | S1P1 |
| Access | | | | | | | R/W | R/W |
| Reset | | | | | | | 0 | 0 |

**Bit 1 – SaP2** Slice “a” Parameter 2 Interrupt Enable

| Value | Description |
|---|---|
| 1 | Slice “a” Parameter 2 match interrupt is enabled |
| 0 | Slice “a” Parameter 2 match interrupt is not enabled |

**Bit 0 – SaP1** Slice “a” Parameter 1 Interrupt Enable

| Value | Description |
|---|---|
| 1 | Slice “a” Parameter 1 match interrupt is enabled |
| 0 | Slice “a” Parameter 1 match interrupt is not enabled |

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



### 29.9.9 PWMxCON

**Name:** PWMxCON
**Offset:** 0x0495,0x0515,0x0995,0x0A15

PWM Control Register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | EN | | | | | LD | ERSPOL | ERSNOW |
| Access | R/W | | | | | R/W/HC | R/W | R/W |
| Reset | 0 | | | | | 0 | 0 | 0 |

**Bit 7 – EN** PWM Module Enable

| Value | Description |
|---|---|
| 1 | PWM module is enabled |
| 0 | PWM module is disabled. The prescaler, postscaler, and all internal logic is reset. Outputs go to their default states. Register values remain unchanged. |

**Bit 2 – LD** Reload Registers

Reload the period and duty cycle registers on the next period event

| Value | Description |
|---|---|
| 1 | Reload PR/P1/P2 registers |
| 0 | Reload not enabled or reload complete |

**Bit 1 – ERSPOL** External Reset Polarity Select

| Value | Description |
|---|---|
| 1 | External Reset input is active-low |
| 0 | External Reset input is active-high |

**Bit 0 – ERSNOW** External Reset Mode Select

Determines when an external Reset event takes effect.

| Value | Description |
|---|---|
| 1 | Stop counter on the next PWM clock. Output goes to the Inactive state. |
| 0 | Stop counter at the end of the period. Output goes to the Inactive state. |

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



### 29.9.10 PWMxSaCFG

**Name:** PWMxSaCFG

PWM Slice “a” Configuration Register(1)

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | POL2 | POL1 | | | PPEN | MODE[2:0] | | |
| Access | R/W | R/W | | | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | | | 0 | 0 | 0 | 0 |

**Bit 7 – POL2** PWM Slice “a” Parameter 2 Output Polarity

| Value | Description |
|---|---|
| 1 | PWMx_SaP2_out is low true |
| 0 | PWMx_SaP2_out is high true |

**Bit 6 – POL1** PWM Slice “a” Parameter 1 Output Polarity

| Value | Description |
|---|---|
| 1 | PWMx_SaP1_out is low true |
| 0 | PWMx_SaP1_out is high true |

**Bit 3 – PPEN** Push-Pull Mode Enable

Each period the output alternates between PWMx_SaP1_out and PWMx_SaP2_out. Only Left and Right Aligned modes are supported. Other modes may exhibit unexpected results.

| Value | Description |
|---|---|
| 1 | PWMx Slice “a” Push-Pull mode is enabled |
| 0 | PWMx Slice “a” Push-Pull mode is not enabled |

**Bits 2:0 – MODE[2:0]** PWM Module Slice “a” Operating Mode Select

Selects operating mode for both PWMx_SaP1_out and PWMx_SaP2_out

| Value | Description |
|---|---|
| 11x | Reserved. Outputs go to Reset state. |
| 101 | Compare mode: Toggle PWMx_SaP1_out and PWMx_SaP2_out on PWM timer match with corresponding parameter register |
| 100 | Compare mode: Set PWMx_SaP1_out and PWMx_SaP2_out high on PWM timer match with corresponding parameter register |
| 011 | Variable Aligned mode |
| 010 | Center-Aligned mode |
| 001 | Right Aligned mode |
| 000 | Left Aligned mode |

**Note:**

1. Changes to this register must be done only when the EN bit is cleared.

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



### 29.9.11 PWMxSaP1

**Name:** PWMxSaP1

PWM Slice “a” Parameter 1 Register

Determines the active period of slice “a”, parameter 1 output

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | P1[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | P1[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 15:0 – P1[15:0]** Parameter 1 Value

| Value | Mode | Description |
|---|---|---|
| n | Compare | Compare match event occurs when PWMx timer = n (Refer to MODE selections) |
| n | Variable Aligned | PWMx_SaP1_out and PWMx_SaP2 both go high when PWMx timer = n |
| n | Center-Aligned | PWMx_SaP1_out is high 2*n PWMx clock periods centered around PWMx period event |
| n | Right Aligned | PWMx_SaP1_out is high n PWMx clock periods at end of PWMx period |
| n | Left Aligned | PWMx_SaP1_out is high n PWMx clock periods at beginning of PWMx period |

**Notes:** The individual bytes in this multibyte register can be accessed with the following register names:

- PWMxSaP1H: Accesses the high byte P1[15:8]
- PWMxSaP1L: Accesses the low byte P1[7:0]

chanhvinh.com



### 29.9.12 PWMxSaP2

**Name:** PWMxSaP2

PWM Slice “a” Parameter 2 Register

Determines the active period of slice “a”, parameter 2 output

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | P2[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | P2[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 15:0 – P2[15:0]** Parameter 2 Value

| Value | Mode | Description |
|---|---|---|
| n | Compare | Compare match event occurs when PWMx timer = n (Refer to MODE selections) |
| n | Variable Aligned | PWMx_SaP1_out and PWMx_SaP2 both go low when PWMx timer = n |
| n | Center-Aligned | PWMx_SaP2_out is high 2*n PWMx clock periods centered around PWMx period event |
| n | Right Aligned | PWMx_SaP2_out is high n PWMx clock periods at end of PWMx period |
| n | Left Aligned | PWMx_SaP2_out is high n PWMx clock periods at beginning of PWMx period |

**Notes:** The individual bytes in this multibyte register can be accessed with the following register names:

- PWMxSaP2H: Accesses the high byte P2[15:8]
- PWMxSaP2L: Accesses the low byte P2[7:0]

chanhvinh.com



### 29.9.13 PWMLOAD

**Name:** PWMLOAD
**Offset:** 0x049E

Mirror copies of all PWMxLD bits

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | MPWM4LD | MPWM3LD | MPWM2LD | MPWM1LD |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

**Bits 0, 1, 2, 3 – MPWMxLD** Mirror copy of PWMxLD bit

Mirror copies of all PWMxLD bits can be set simultaneously to synchronize the load event across all PWMs

| Value | Description |
|---|---|
| 1 | PWMx parameter and period values will be transferred to their buffer registers at the next period Reset event |
| 0 | There are no PWMx period and parameter value transfers pending |

chanhvinh.com

chanhvinh.com



### 29.9.14 PWMEN

**Name:** PWMEN
**Offset:** 0x049F

Mirror copies of all PWMxEN bits

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | MPWM4EN | MPWM3EN | MPWM2EN | MPWM1EN |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

**Bits 0, 1, 2, 3 – MPWMxEN** Mirror copy of PWMxEN bit

Mirror copies of all PWMxEN bits can be set simultaneously to synchronize the enable event across all PWMs

| Value | Description |
|---|---|
| 1 | PWMx is enabled |
| 0 | PWMx is not enabled |

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



## 29.10 Register Summary - PWM

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 ... 0x048B | Reserved | | | | | | | | | |
| 0x048C | PWM1ERS | 7:0 | | | | | ERS[3:0] | | | |
| 0x048D | PWM1CLK | 7:0 | | | | | CLK[3:0] | | | |
| 0x048E | PWM1LDS | 7:0 | | | | | | LDS[2:0] | | |
| 0x048F | PWM1PR | 7:0 | PR[7:0] | | | | | | | |
| | | 15:8 | PR[15:8] | | | | | | | |
| 0x0491 | PWM1CPRE | 7:0 | CPRE[7:0] | | | | | | | |
| 0x0492 | PWM1PIPOS | 7:0 | PIPOS[7:0] | | | | | | | |
| 0x0493 | PWM1GIR | 7:0 | | | | | | | S1P2 | S1P1 |
| 0x0494 | PWM1GIE | 7:0 | | | | | | | S1P2 | S1P1 |
| 0x0495 | PWM1CON | 7:0 | EN | | | | | LD | ERSPOL | ERSNOW |
| 0x0496 | PWM1S1CFG | 7:0 | POL2 | POL1 | | | PPEN | MODE[2:0] | | |
| 0x0497 | PWM1S1P1 | 7:0 | P1[7:0] | | | | | | | |
| | | 15:8 | P1[15:8] | | | | | | | |
| 0x0499 | PWM1S1P2 | 7:0 | P2[7:0] | | | | | | | |
| | | 15:8 | P2[15:8] | | | | | | | |
| 0x049B ... 0x049D | Reserved | | | | | | | | | |
| 0x049E | PWMLOAD | 7:0 | | | | | MPWM4LD | MPWM3LD | MPWM2LD | MPWM1LD |
| 0x049F | PWMEN | 7:0 | | | | | MPWM4EN | MPWM3EN | MPWM2EN | MPWM1EN |
| 0x04A0 ... 0x050B | Reserved | | | | | | | | | |
| 0x050C | PWM2ERS | 7:0 | | | | | ERS[3:0] | | | |
| 0x050D | PWM2CLK | 7:0 | | | | | CLK[3:0] | | | |
| 0x050E | PWM2LDS | 7:0 | | | | | | LDS[2:0] | | |
| 0x050F | PWM2PR | 7:0 | PR[7:0] | | | | | | | |
| | | 15:8 | PR[15:8] | | | | | | | |
| 0x0511 | PWM2CPRE | 7:0 | CPRE[7:0] | | | | | | | |
| 0x0512 | PWM2PIPOS | 7:0 | PIPOS[7:0] | | | | | | | |
| 0x0513 | PWM2GIR | 7:0 | | | | | | | S1P2 | S1P1 |
| 0x0514 | PWM2GIE | 7:0 | | | | | | | S1P2 | S1P1 |
| 0x0515 | PWM2CON | 7:0 | EN | | | | | LD | ERSPOL | ERSNOW |
| 0x0516 | PWM2S1CFG | 7:0 | POL2 | POL1 | | | PPEN | MODE[2:0] | | |
| 0x0517 | PWM2S1P1 | 7:0 | P1[7:0] | | | | | | | |
| | | 15:8 | P1[15:8] | | | | | | | |
| 0x0519 | PWM2S1P2 | 7:0 | P2[7:0] | | | | | | | |
| | | 15:8 | P2[15:8] | | | | | | | |
| 0x051B ... 0x098B | Reserved | | | | | | | | | |
| 0x098C | PWM3ERS | 7:0 | | | | | ERS[3:0] | | | |
| 0x098D | PWM3CLK | 7:0 | | | | | CLK[3:0] | | | |
| 0x098E | PWM3LDS | 7:0 | | | | | | LDS[2:0] | | |
| 0x098F | PWM3PR | 7:0 | PR[7:0] | | | | | | | |
| | | 15:8 | PR[15:8] | | | | | | | |
| 0x0991 | PWM3CPRE | 7:0 | CPRE[7:0] | | | | | | | |
| 0x0992 | PWM3PIPOS | 7:0 | PIPOS[7:0] | | | | | | | |
| 0x0993 | PWM3GIR | 7:0 | | | | | | | S1P2 | S1P1 |
| 0x0994 | PWM3GIE | 7:0 | | | | | | | S1P2 | S1P1 |
| 0x0995 | PWM3CON | 7:0 | EN | | | | | LD | ERSPOL | ERSNOW |
| 0x0996 | PWM3S1CFG | 7:0 | POL2 | POL1 | | | PPEN | MODE[2:0] | | |

chanhvinh.com



.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0997 | PWM3S1P1 | 7:0 | P1[7:0] | | | | | | | |
| | | 15:8 | P1[15:8] | | | | | | | |
| 0x0999 | PWM3S1P2 | 7:0 | P2[7:0] | | | | | | | |
| | | 15:8 | P2[15:8] | | | | | | | |
| 0x099B ... 0x0A0B | Reserved | | | | | | | | | |
| 0x0A0C | PWM4ERS | 7:0 | | | | | ERS[3:0] | | | |
| 0x0A0D | PWM4CLK | 7:0 | | | | | CLK[3:0] | | | |
| 0x0A0E | PWM4LDS | 7:0 | | | | | | LDS[2:0] | | |
| 0x0A0F | PWM4PR | 7:0 | PR[7:0] | | | | | | | |
| | | 15:8 | PR[15:8] | | | | | | | |
| 0x0A11 | PWM4CPRE | 7:0 | CPRE[7:0] | | | | | | | |
| 0x0A12 | PWM4PIPOS | 7:0 | PIPOS[7:0] | | | | | | | |
| 0x0A13 | PWM4GIR | 7:0 | | | | | | | S1P2 | S1P1 |
| 0x0A14 | PWM4GIE | 7:0 | | | | | | | S1P2 | S1P1 |
| 0x0A15 | PWM4CON | 7:0 | EN | | | | | LD | ERSPOL | ERSNOW |
| 0x0A16 | PWM4S1CFG | 7:0 | POL2 | POL1 | | | PPEN | MODE[2:0] | | |
| 0x0A17 | PWM4S1P1 | 7:0 | P1[7:0] | | | | | | | |
| | | 15:8 | P1[15:8] | | | | | | | |
| 0x0A19 | PWM4S1P2 | 7:0 | P2[7:0] | | | | | | | |
| | | 15:8 | P2[15:8] | | | | | | | |

chanhvinh.com

chanhvinh.com



# 30. CLC - Configurable Logic Cell

The Configurable Logic Cell (CLC) module provides programmable logic that operates outside the speed limitations of software execution. The logic cell takes up to 256 input signals and, through the use of configurable gates, reduces those inputs to four logic lines that drive one of eight selectable single-output logic functions.

Input sources are a combination of the following:

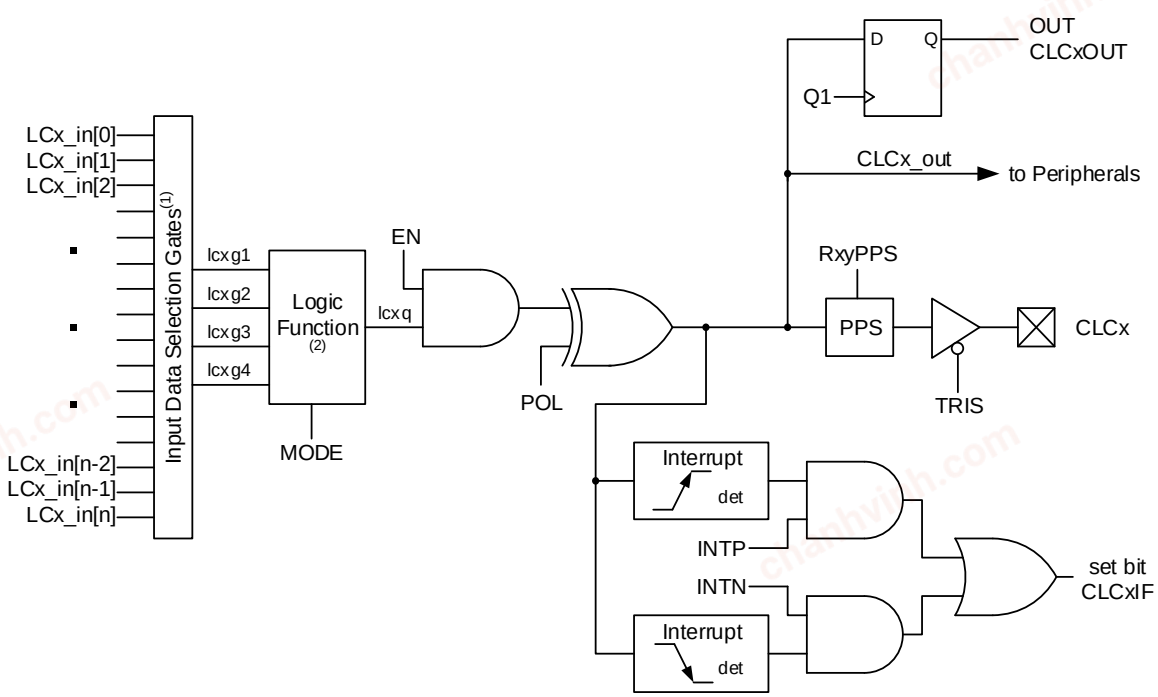
- I/O pins
- Internal clocks
- Peripherals
- Register bits

The output can be directed internally to peripherals and to an output pin.

The following figure is a simplified diagram showing signal flow through the CLC. Possible configurations include:

- Combinatorial Logic
  - AND
  - NAND
  - AND-OR
  - AND-OR-INVERT
  - OR-XOR
  - OR-XNOR
- Latches
  - SR
  - Clocked D with Set and Reset
  - Transparent D with Set and Reset

**Figure 30-1. CLC Simplified Block Diagram**



chanhvinh.com



**Notes:**

1. See Figure 30-2 for input data selection and gating.
2. See Figure 30-3 for programmable logic functions.

## 30.1 CLC Setup

Programming the CLC 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 set up at run time by writing to the corresponding CLC Special Function Registers. This has the added advantage of permitting logic reconfiguration on-the-fly during program execution.

### 30.1.1 Data Selection

Data inputs are selected with CLCnSEL0 through CLCnSEL3 registers.

**Important:** Data selections are undefined at power-up.

Depending on the number of bits implemented in the CLCnSELy registers, there can be as many as 256 sources available as inputs to the configurable logic. Four multiplexers are used to independently select these inputs to pass on to the next stage as indicated on the left side of the following diagram.

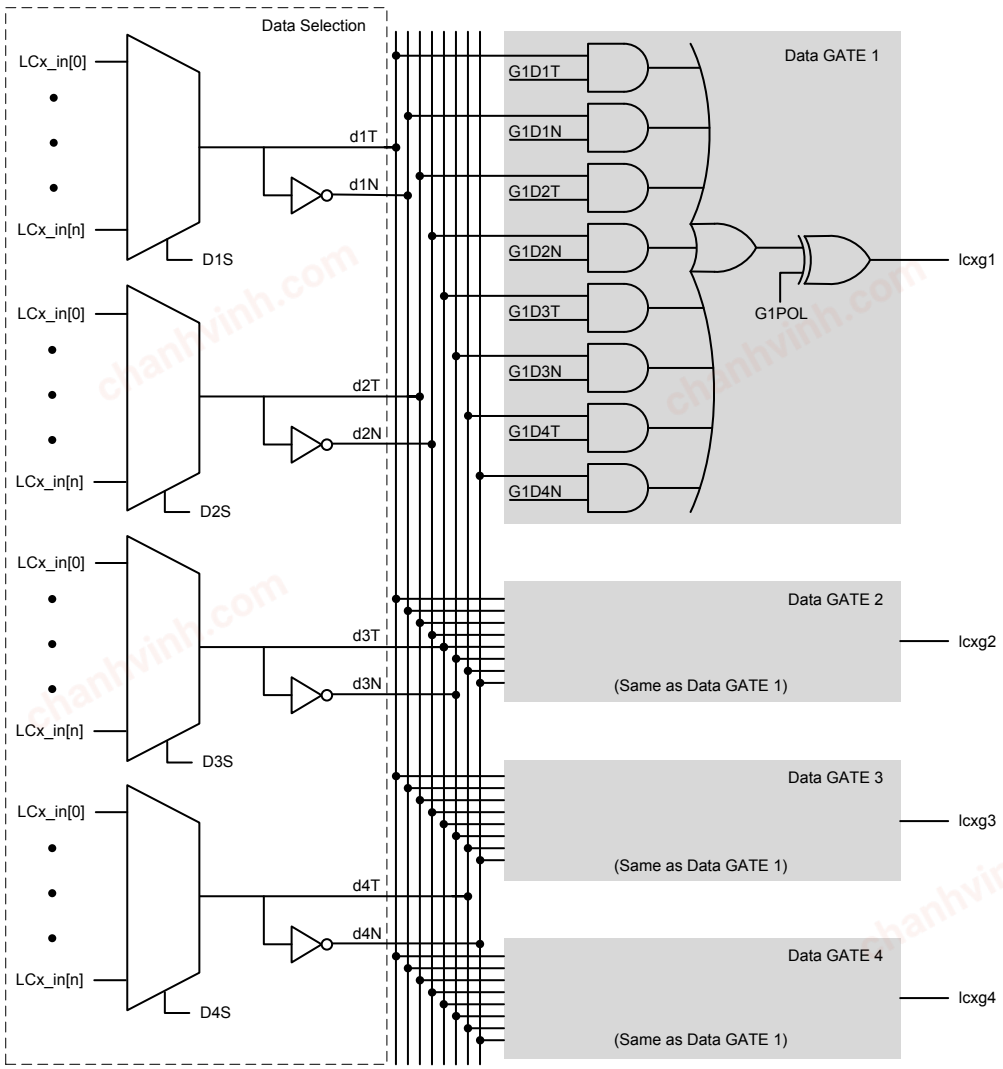
Data inputs in the figure are identified by a generic numbered input name.

chanhvinh.com

chanhvinh.com



**Figure 30-2. Input Data Selection and Gating**




**Note:** All controls are undefined at power-up.

The CLC Input Selection table correlates the generic input name to the actual signal for each CLC module. The table column labeled 'DyS Value' indicates the MUX selection code for the selected data input. DyS is an abbreviation for the MUX select input codes, D1S through D4S, where 'y' is the gate number.

### 30.1.2 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.

The gate stage is more than just signal direction. The gate can be configured to direct each input signal as inverted or noninverted 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 AND of all enabled data inputs. When the inputs and output are not inverted, the gate is an OR or all enabled inputs.

Table 30-1 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 '0' or '1', depending on the gate output polarity bit.

chanhvinh.com



**Table 30-1. Data Gating Logic**

| CLCnGLSy | GyPOL | 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 '0', regardless of the other inputs, but may emit logic glitches (transient-induced pulses). If the output of the channel must be '0' or '1', the recommended method is to set all gate bits to '0' 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: CLCnGLS0
- Gate 2: CLCnGLS1
- Gate 3: CLCnGLS2
- Gate 4: CLCnGLS3

**Note:** 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, except when the data enables correspond to the enables for that gate.

### 30.1.3 Logic Function

There are eight available logic functions including:

- AND-OR
- OR-XOR
- AND
- SR 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 the following diagram. 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 CLC itself.

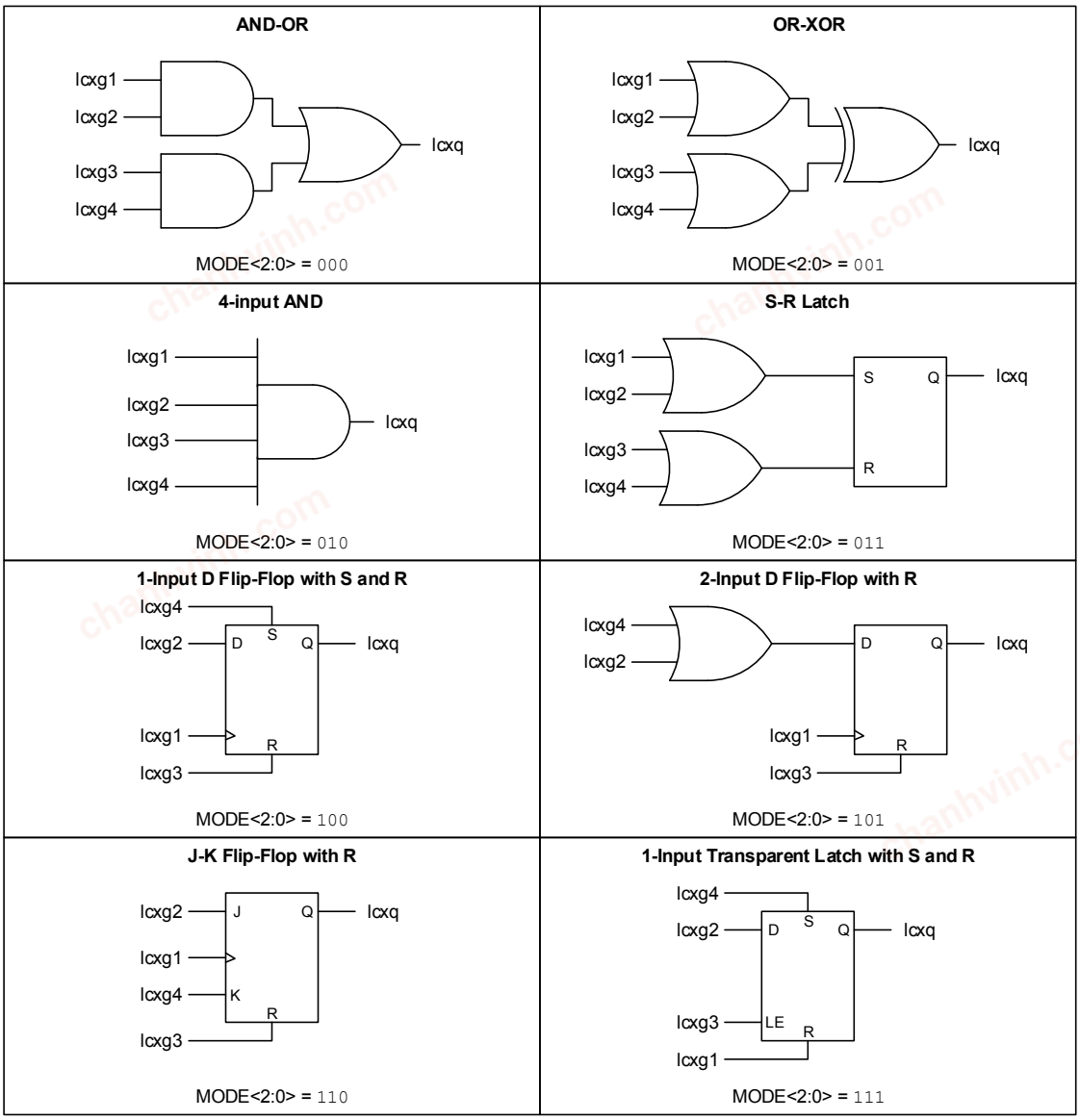
© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



**Figure 30-3. Programmable Logic Functions**

Rev. 10-000122B
9/13/2016




### 30.1.4 Output Polarity

The last stage in the Configurable Logic Cell is the output polarity. Setting the POL bit 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 CLC 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 register will be set when either edge detector is triggered and its associated enable bit is set. The INTP bit enables rising edge interrupts and the INTN bit enables falling edge interrupts.

chanhvinh.com



To fully enable the interrupt, set the following bits:

- The CLCxIE bit of the respective PIE register
- The INTP bit (for a rising edge detection)
- The INTN bit (for a falling edge detection)

If priority interrupts are not used:

1. Clear the IPEN bit of the INTCON register.
2. Set the GIE bit of the INTCON register.
3. Set the GIEL bit of the INTCON register.

If the CLC is a high-priority interrupt:

1. Set the IPEN bit of the INTCON register.
2. Set the CLCxIP bit of the respective IPR register.
3. Set the GIEH bit of the INTCON register.

If the CLC is a low-priority interrupt:

1. Set the IPEN bit of the INTCON register.
2. Clear the CLCxIP bit of the respective IPR register.
3. Set the GIEL bit of the INTCON register.

The CLCxIF bit of the respective PIR 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.

## 30.3 Effects of a Reset

The CLCnCON register is cleared to '0' as the result of a Reset. All other selection and gating values remain unchanged.

## 30.4 Output Mirror Copies

Mirror copies of all CLCxOUT bits are contained in the CLCDATA register. Reading this register reads the outputs of all CLCs simultaneously. This prevents any reading skew introduced by testing or reading the OUT bits in the individual CLCnCON registers.

## 30.5 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 both the system clock and as a CLC input source then, 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.

## 30.6 CLC Setup Steps

These steps need to be followed when setting up the CLC:

1. Disable the CLC by clearing the EN bit.
2. Select the desired inputs using the CLCnSEL0 through CLCnSEL3 registers.
3. Clear any ANSEL bits associated with CLC input pins.
4. Set all TRIS bits associated with inputs. However, a CLC input will also operate if the pin is configured as an output, in which case the TRIS bits must be cleared.

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



5. Enable the chosen inputs through the four gates using the CLCnGLS0 through CLCnGLS3 registers.
6. Select the gate output polarities with the GyPOL bits.
7. Select the desired logic function with the MODE bits.
8. Select the desired polarity of the logic output with the POL bit (this step may be combined with the previous gate output polarity step).
9. If driving a device pin, configure the associated pin PPS control register and also clear the TRIS bit corresponding to that output.
10. Configure the interrupts (optional). See the CLC Interrupts section.
11. Enable the CLC by setting the EN bit.

## 30.7 Register Overlay

All CLCs in this device share the same set of registers. Only one CLC instance is accessible at a time. The value in the CLCSELECT register is one less than the selected CLC instance. For example, a CLCSELECT value of '0' selects CLC1.

## 30.8 Register Definitions: Configurable Logic Cell

chanhvinh.com

chanhvinh.com



### 30.8.1 CLCSELECT

**Name:** CLCSELECT
**Offset:** 0x0696

CLC Instance Selection Register

Selects which CLC instance is accessed by the CLC registers

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | SLCT[3:0] | | | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

**Bits 3:0 – SLCT[3:0]** CLC instance selection

| Value | Description |
|---|---|
| n | Shared CLC registers of instance n+1 are selected for read and write operations |

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



### 30.8.2 CLCnCON

**Name:** CLCnCON
**Offset:** 0x068C

Configurable Logic Cell Control Register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | EN | | OUT | INTP | INTN | MODE[2:0] | | |
| Access | R/W | | R | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 7 – EN** CLC Enable

| Value | Description |
|---|---|
| 1 | Configurable logic cell is enabled and mixing signals |
| 0 | Configurable logic cell is disabled and has logic zero output |

**Bit 5 – OUT** Logic cell output data, after LCPOL. Sampled from CLCxOUT.

**Bit 4 – INTP** Configurable Logic Cell Positive Edge Going Interrupt Enable

| Value | Description |
|---|---|
| 1 | CLCxIF will be set when a rising edge occurs on CLCxOUT |
| 0 | Rising edges on CLCxOUT have no effect on CLCxIF |

**Bit 3 – INTN** Configurable Logic Cell Negative Edge Going Interrupt Enable

| Value | Description |
|---|---|
| 1 | CLCxIF will be set when a falling edge occurs on CLCxOUT |
| 0 | Falling edges on CLCxOUT have no effect on CLCxIF |

**Bits 2:0 – MODE[2:0]** Configurable Logic Cell Functional Mode Selection

| Value | Description |
|---|---|
| 111 | Cell is 1-input transparent latch with Set and Reset |
| 110 | Cell is J-K flip-flop with Reset |
| 101 | Cell is 2-input D flip-flop with Reset |
| 100 | Cell is 1-input D flip-flop with Set and Reset |
| 011 | Cell is SR latch |
| 010 | Cell is 4-input AND |
| 001 | Cell is OR-XOR |
| 000 | Cell is AND-OR |

chanhvinh.com

chanhvinh.com



### 30.8.3 CLCnPOL

**Name:** CLCnPOL
**Offset:** 0x068D

Signal Polarity Control Register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | POL | | | | G4POL | G3POL | G2POL | G1POL |
| Access | R/W | | | | R/W | R/W | R/W | R/W |
| Reset | 0 | | | | x | x | x | x |

**Bit 7 – POL** CLCxOUT Output Polarity Control

| Value | Description |
|---|---|
| 1 | The output of the logic cell is inverted |
| 0 | The output of the logic cell is not inverted |

**Bits 0, 1, 2, 3 – GyPOL** Gate Output Polarity Control
Reset States: POR/BOR = xxxx
All Other Resets = uuuu

| Value | Description |
|---|---|
| 1 | The gate output is inverted when applied to the logic cell |
| 0 | The output of the gate is not inverted |

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



### 30.8.4 CLCnSEL0

**Name:** CLCnSEL0
**Offset:** 0x068E

Generic CLCn Data 1 Select Register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | D1S[6:0] | | | | | | |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | x | x | x | x | x | x | x |

**Bits 6:0 – D1S[6:0]** CLCn Data1 Input Selection

**Table 30-2. CLC Input Selection**

| DyS | Input Source | DyS (cont.) | Input Source (cont.) |
|---|---|---|---|
| [0] 0000 0000 | CLCIN0PPS | [32] 0010 0000 | C2_OUT |
| [1] 0000 0001 | CLCIN1PPS | [33] 0010 0001 | ZCD_OUT |
| [2] 0000 0010 | CLCIN2PPS | [34] 0010 0010 | IOCIF |
| [3] 0000 0011 | CLCIN3PPS | [35] 0010 0011 | CLC1_OUT |
| [4] 0000 0100 | $F_{OSC}$ | [36] 0010 0100 | CLC2_OUT |
| [5] 0000 0101 | HFINTOSC | [37] 0010 0101 | CLC3_OUT |
| [6] 0000 0110 | LFINTOSC | [38] 0010 0110 | CLC4_OUT |
| [7] 0000 0111 | MFINTOSC (500 kHz) | [39] 0010 0111 | TX1/CK1 |
| [8] 0000 1000 | MFINTOSC (32 kHz) | [40] 0010 1000 | TX2/CK2 |
| [9] 0000 1001 | — | [41] 0010 1001 | SDA1/SDO1 |
| [10] 0000 1010 | SOSC | [42] 0010 1010 | SCL1/SCK1 |
| [11] 0000 1011 | EXTOSC | [43] 0010 1011 | SDA2/SDO2 |
| [12] 0000 1100 | ADCRC | [44] 0010 1100 | SCL2/SCK2 |
| [13] 0000 1101 | CLKR_OUT | [45] 0010 1101 | CWG1A_OUT |
| [14] 0000 1110 | TMR0_Overflow | [46] 0010 1110 | CWG1B_OUT |
| [15] 0000 1111 | TMR1_Overflow | ... | — |
| [16] 0001 0000 | TMR2_Postscaled_OUT | ... | — |
| [17] 0001 0001 | TMR3_Overflow | ... | — |
| [18] 0001 0010 | TMR4_Postscaled_OUT | ... | — |
| [19] 0001 0011 | TMR6_Postscaled_OUT | ... | — |
| [20] 0001 0100 | CCP1_OUT | ... | — |
| [21] 0001 0101 | CCP2_OUT | ... | — |
| [22] 0001 0110 | PWM1S1P1_OUT | ... | — |
| [23] 0001 0111 | PWM1S1P2_OUT | ... | — |
| [24] 0001 1000 | PWM2S1P1_OUT | ... | — |
| [25] 0001 1001 | PWM2S1P2_OUT | ... | — |
| [26] 0001 1010 | PWM3S1P1_OUT | ... | — |
| [27] 0001 1011 | PWM3S1P2_OUT | ... | — |
| [28] 0001 1100 | PWM4S1P1_OUT | ... | — |
| [29] 0001 1101 | PWM4S1P2_OUT | ... | — |
| [30] 0001 1110 | NCO1_OUT | ... | — |
| [31] 0001 1111 | C1_OUT | [127] 0111 1111 | — |

Reset States: POR/BOR = xxxxxxx
All Other Resets = uuuuuuu

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



### 30.8.5 CLCnSEL1

**Name:** CLCnSEL1
**Offset:** 0x068F

Generic CLCn Data 1 Select Register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | D2S[6:0] | | | | | | |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | x | x | x | x | x | x | x |

**Bits 6:0 – D2S[6:0]** CLCn Data2 Input Selection
Reset States: POR/BOR = xxxxxxx
All Other Resets = uuuuuuu

| Value | Description |
|---|---|
| n | Refer to the CLC Input Selection table for input selections |

chanhvinh.com



### 30.8.6 CLCnSEL2

**Name:** CLCnSEL2
**Offset:** 0x0690

Generic CLCn Data 1 Select Register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | D3S[6:0] | | | | | | |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | x | x | x | x | x | x | x |

**Bits 6:0 – D3S[6:0]** CLCn Data3 Input Selection
Reset States: POR/BOR = xxxxxxx
All Other Resets = uuuuuuu

| Value | Description |
|---|---|
| n | Refer to the CLC Input Selection table for input selections |

chanhvinh.com



### 30.8.7 CLCnSEL3

**Name:** CLCnSEL3
**Offset:** 0x0691

Generic CLCn Data 4 Select Register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | D4S[6:0] | | | | | | |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | x | x | x | x | x | x | x |

**Bits 6:0 – D4S[6:0]** CLCn Data4 Input Selection
Reset States: POR/BOR = xxxxxxx
All Other Resets = uuuuuuu

| Value | Description |
|---|---|
| n | Refer to the CLC Input Selection table for input selections |

chanhvinh.com



### 30.8.8 CLCnGLS0

**Name:** CLCnGLS0
**Offset:** 0x0692

CLCn Gate1 Logic Select Register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | G1D4T | G1D4N | G1D3T | G1D3N | G1D2T | G1D2N | G1D1T | G1D1N |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | x | x | x | x | x | x | x | x |

**Bits 1, 3, 5, 7 – G1DyT** dyT: Gate1 Data 'y' True (noninverted)
Reset States: POR/BOR = xxxx
All Other Resets = uuuu

| Value | Description |
|---|---|
| 1 | dyT is gated into g1 |
| 0 | dyT is not gated into g1 |

**Bits 0, 2, 4, 6 – G1DyN** dyN: Gate1 Data 'y' Negated (inverted)
Reset States: POR/BOR = xxxx
All Other Resets = uuuu

| Value | Description |
|---|---|
| 1 | dyN is gated into g1 |
| 0 | dyN is not gated into g1 |

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



### 30.8.9 CLCnGLS1

**Name:** CLCnGLS1
**Offset:** 0x0693

CLCn Gate2 Logic Select Register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | G2D4T | G2D4N | G2D3T | G2D3N | G2D2T | G2D2N | G2D1T | G2D1N |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | x | x | x | x | x | x | x | x |

**Bits 1, 3, 5, 7 – G2DyT** dyT: Gate2 Data 'y' True (noninverted)
Reset States: POR/BOR = xxxx
All Other Resets = uuuu

| Value | Description |
|---|---|
| 1 | dyT is gated into g2 |
| 0 | dyT is not gated into g2 |

**Bits 0, 2, 4, 6 – G2DyN** dyN: Gate2 Data 'y' Negated (inverted)
Reset States: POR/BOR = xxxx
All Other Resets = uuuu

| Value | Description |
|---|---|
| 1 | dyN is gated into g2 |
| 0 | dyN is not gated into g2 |

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



### 30.8.10 CLCnGLS2

**Name:** CLCnGLS2
**Offset:** 0x0694

CLCn Gate3 Logic Select Register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | G3D4T | G3D4N | G3D3T | G3D3N | G3D2T | G3D2N | G3D1T | G3D1N |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | x | x | x | x | x | x | x | x |

**Bits 1, 3, 5, 7 – G3DyT** dyT: Gate3 Data 'y' True (noninverted)
Reset States: POR/BOR = xxxx
All Other Resets = uuuu

| Value | Description |
|---|---|
| 1 | dyT is gated into g3 |
| 0 | dyT is not gated into g3 |

**Bits 0, 2, 4, 6 – G3DyN** dyN: Gate3 Data 'y' Negated (inverted)
Reset States: POR/BOR = xxxx
All Other Resets = uuuu

| Value | Description |
|---|---|
| 1 | dyN is gated into g3 |
| 0 | dyN is not gated into g3 |

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



### 30.8.11 CLCnGLS3

**Name:** CLCnGLS3
**Offset:** 0x0695

CLCn Gate4 Logic Select Register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | G4D4T | G4D4N | G4D3T | G4D3N | G4D2T | G4D2N | G4D1T | G4D1N |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | x | x | x | x | x | x | x | x |

**Bits 1, 3, 5, 7 – G4DyT** dyT: Gate4 Data 'y' True (noninverted)
Reset States: POR/BOR = xxxx
All Other Resets = uuuu

| Value | Description |
|---|---|
| 1 | dyT is gated into g4 |
| 0 | dyT is not gated into g4 |

**Bits 0, 2, 4, 6 – G4DyN** dyN: Gate4 Data 'y' Negated (inverted)
Reset States: POR/BOR = xxxx
All Other Resets = uuuu

| Value | Description |
|---|---|
| 1 | dyN is gated into g4 |
| 0 | dyN is not gated into g4 |

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



### 30.8.12 CLCDATA

**Name:** CLCDATA
**Offset:** 0x0697

CLC Data Output Register

Mirror copy of CLC outputs

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | CLC4OUT | CLC3OUT | CLC2OUT | CLC1OUT |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

**Bits 0, 1, 2, 3 – CLCxOUT** Mirror copy of CLCx_out

| Value | Description |
|---|---|
| 1 | CLCx_out is 1 |
| 0 | CLCx_out is 0 |

chanhvinh.com

chanhvinh.com



## 30.9 Register Summary - CLC Control

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 ... 0x068B | Reserved | | | | | | | | | |
| 0x068C | CLCnCON | 7:0 | EN | | OUT | INTP | INTN | MODE[2:0] | | |
| 0x068D | CLCnPOL | 7:0 | POL | | | | G4POL | G3POL | G2POL | G1POL |
| 0x068E | CLCnSEL0 | 7:0 | | D1S[6:0] | | | | | | |
| 0x068F | CLCnSEL1 | 7:0 | | D2S[6:0] | | | | | | |
| 0x0690 | CLCnSEL2 | 7:0 | | D3S[6:0] | | | | | | |
| 0x0691 | CLCnSEL3 | 7:0 | | D4S[6:0] | | | | | | |
| 0x0692 | CLCnGLS0 | 7:0 | G1D4T | G1D4N | G1D3T | G1D3N | G1D2T | G1D2N | G1D1T | G1D1N |
| 0x0693 | CLCnGLS1 | 7:0 | G2D4T | G2D4N | G2D3T | G2D3N | G2D2T | G2D2N | G2D1T | G2D1N |
| 0x0694 | CLCnGLS2 | 7:0 | G3D4T | G3D4N | G3D3T | G3D3N | G3D2T | G3D2N | G3D1T | G3D1N |
| 0x0695 | CLCnGLS3 | 7:0 | G4D4T | G4D4N | G4D3T | G4D3N | G4D2T | G4D2N | G4D1T | G4D1N |
| 0x0696 | CLCSELECT | 7:0 | | | | | SLCT[3:0] | | | |
| 0x0697 | CLCDATA | 7:0 | | | | | CLC4OUT | CLC3OUT | CLC2OUT | CLC1OUT |

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



# 31. MSSP - Host Synchronous Serial Port Module

The Host 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 ($I^2C$)

The SPI interface can operate in Host or Client mode and supports the following features:

- Selectable clock parity
- Client select synchronization (Client mode only)
- Daisy-chain connection of client devices

The $I^2C$ interface can operate in Host or Client mode and supports the following modes and features:

- Byte NACKing (Client mode)
- Limited multi-host 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

## 31.1 SPI Mode Overview

The Serial Peripheral Interface (SPI) is a synchronous serial data communication bus that operates in Full Duplex mode. Devices communicate in a host/client environment where the host device initiates the communication. A client device is selected for communication using the Client Select feature.

The SPI bus specifies four signal connections:

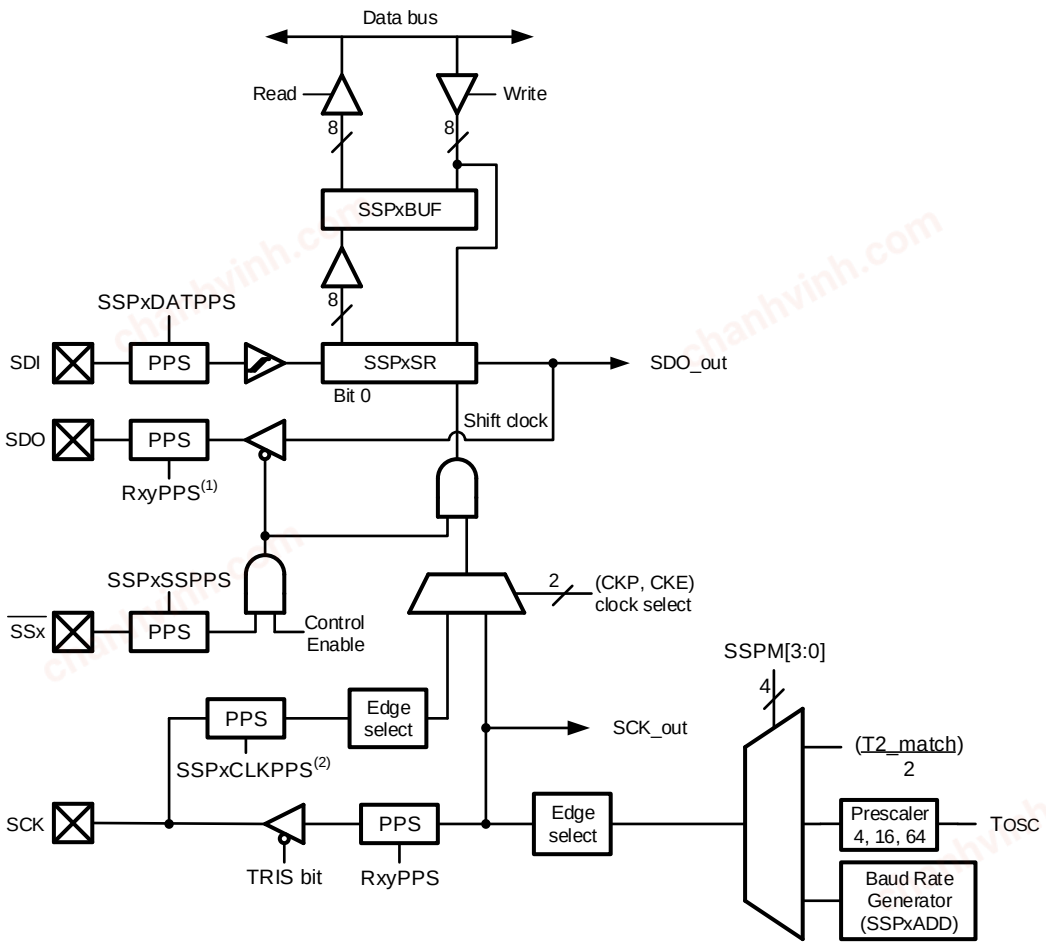
- Serial Clock (SCK)
- Serial Data Out (SDO)
- Serial Data In (SDI)
- Client Select ($\overline{SS}$)

Figure 31-1 shows the block diagram of the MSSP module when operating in SPI mode.

chanhvinh.com



**Figure 31-1. MSSP Block Diagram (SPI Mode)**




**Notes:** 1. Output selection for Host mode.
2. Input selection for Client and Host modes.

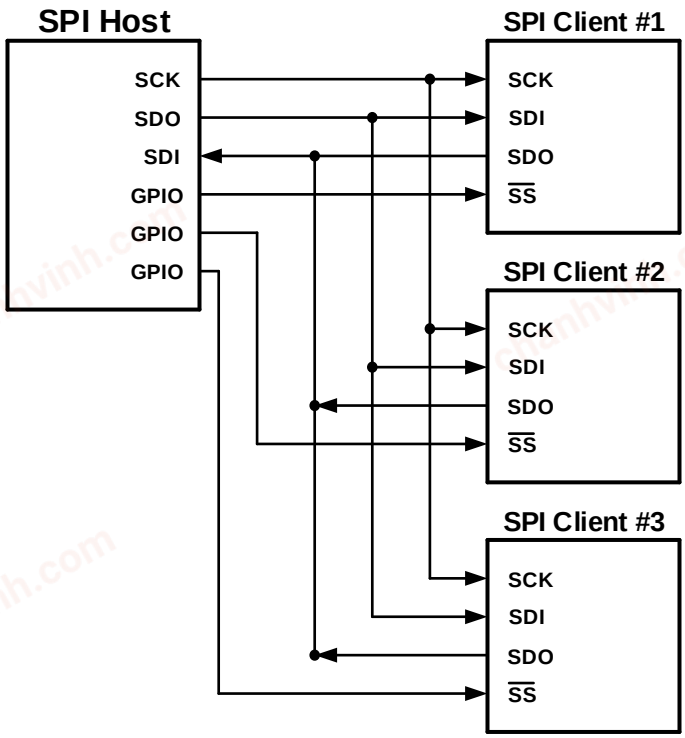
The SPI bus operates with a single host device and one or more client devices. When multiple client devices are used, an independent Client Select connection is required from the host device to each client device. The host selects only one client at a time. Most client devices have tri-state outputs, so their output signal appears disconnected from the bus when they are not selected.

Figure 31-2 shows a typical connection between a host device and multiple client devices.

chanhvinh.com



**Figure 31-2. SPI Host and Multiple Client Connection**



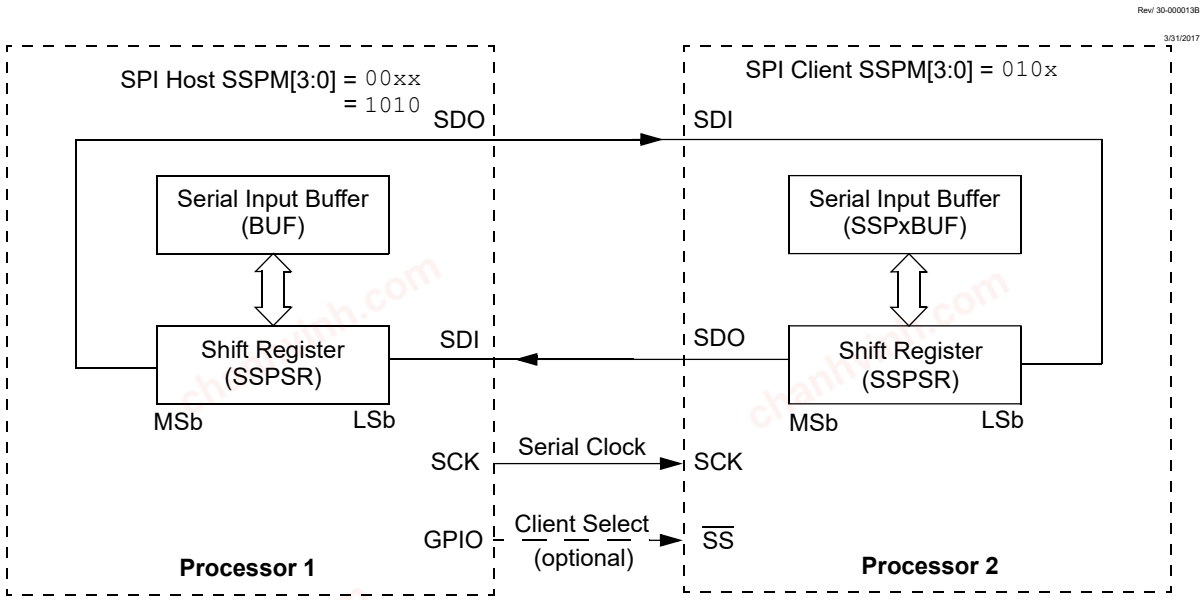

Transmissions involve two shift registers, eight bits in size: One in the host and one in the client. Data is always shifted out one bit at a time, with the Most Significant bit (MSb) shifted out first. At the same time, a new Least Significant bit (LSb) is shifted into the same register.

Figure 31-3 shows a typical connection between two processors configured as host and client devices.

chanhvinh.com



**Figure 31-3. SPI Host/Client Connection**




Data is shifted out of both shift registers on the programmed clock edge and latched on the opposite edge of the clock.

The host device transmits information out on its SDO output pin, which is connected to and received by the client's SDI input pin. The client device transmits information out on its SDO output pin, which is connected to and received by the host's SDI input pin.

To begin communication, the host device transmits both the MSb from its shift register and the clock signal. Both the host and client devices need to be configured for the same clock polarity. During each SPI clock cycle, a full-duplex data transmission occurs. This means that while the host device is sending out the MSb from its shift register (on its SDO pin) and the client device is reading this bit and saving it as the LSb of its shift register, the client device is also sending out the MSb from its shift register (on its SDO pin) and the host device is reading this bit and saving it as the LSb of its shift register.

After eight bits have been shifted out, the host and client 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:

- Host sends useful data and client sends dummy data.
- Host sends useful data and client sends useful data.
- Host sends dummy data and client sends useful data.

Transmissions must be performed in multiples of eight clock cycles. When there is no more data to be transmitted, the host stops sending the clock signal and it deselects the client.

Every client device connected to the bus that has not been selected through its Client Select line must disregard the clock and transmission signals and must not transmit out any data of its own.

### 31.1.1 SPI Mode Registers

The MSSP module has six registers for SPI mode operation. They are:

- MSSP Status Register (SSPxSTAT)
- MSSP Control Register 1 (SSPxCON1)
- MSSP Control Register 3 (SSPxCON3)
- MSSP Data Buffer Register (SSPxBUF)

chanhvinh.com



- MSSP Address Register (SSPxADD)
- MSSP Shift (SSPSR) register (not directly accessible)

SSPxCON1 and SSPxSTAT are the control and status registers for 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.

One of the five SPI Host modes uses the SSPxADD value to determine the Baud Rate Generator clock frequency. More information on the Baud Rate Generator is available in 31.3. 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.

### 31.1.2 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:

- Host mode (SCK is the clock output)
- Client 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 (Host mode only)
- Client Select mode (Client mode only)

To enable the serial port, the SSP Enable (SSPEN) bit must be set. To reset or reconfigure SPI mode, clear the SSPEN bit, re-initialize the SSPxCONy registers and then set the SSPEN bit. The SDI, SDO, SCK and $\overline{SS}$ serial port pins are selected with the PPS controls. 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 the corresponding TRIS bit set
- SDO must have the corresponding TRIS bit cleared
- SCK (Host mode) must have the corresponding TRIS bit cleared
- SCK (Client mode) must have the corresponding TRIS bit set
- The RxyPPS and SSPxCLKPPS controls must select the same pin
- $\overline{SS}$ must have the 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 Status (BF) bit and the MSSP Interrupt Flag (SSPxIF) bit are set. This double-buffering of the received data 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 (WCOL) bit 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 must be read before the next byte of data to be transferred is written to the SSPxBUF. The BF bit 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. 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.

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



**Important:** The SSPSR is not directly readable or writable and can only be accessed by addressing the SSPxBUF register.

#### 31.1.2.1 SPI Host Mode

The Host can initiate the data transfer at any time because it controls the SCK line. The Host determines when the client (Processor 2, Figure 31-3) is to broadcast data by the software protocol.

In Host 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 may 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 (interrupts and Status bits appropriately set).

The clock polarity is selected by appropriately programming the Clock Polarity Select (CKP) and SPI Clock Edge Select (CKE) bits. Figure 31-4 shows the four clocking configurations. When the CKE bit is set, the SDO data is valid one half of a clock cycle before a clock edge appears on SCK, and transmission occurs on the transition from the Active to Idle clock state. When CKE is clear, the SDO data is valid at the same time as the clock edge appears on SCK, and transmission occurs on the transition from the Idle to Active clock states.

The SPI Data Input Sample (SMP) bit determines when the SDI input is sampled. When SMP is set, input data is sampled at the end of the data output time. When SMP is clear, input data is sampled at the middle of the data output time.

The SPI clock rate (bit rate) is user-programmable to be one of the following:

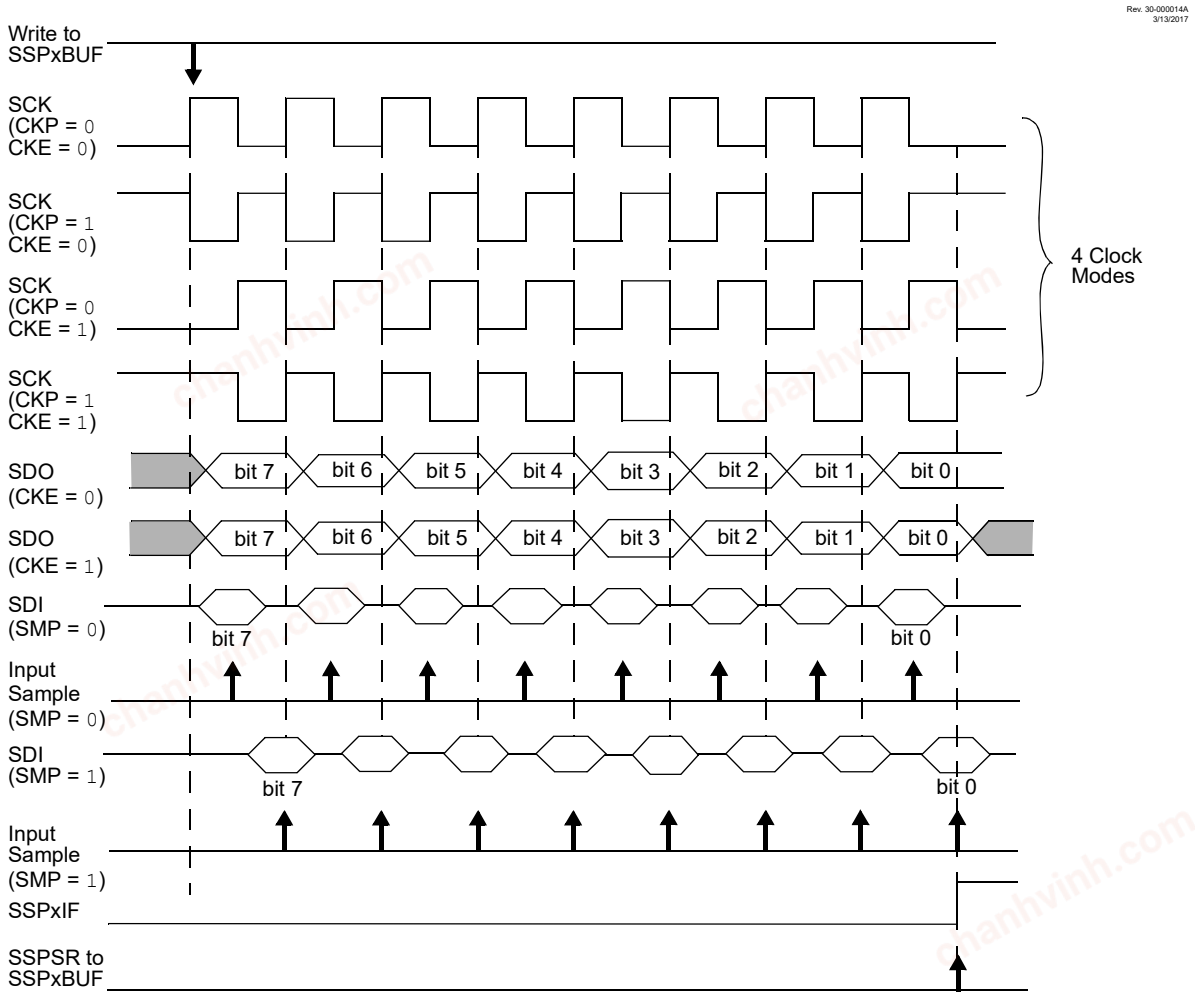
- $F_{OSC}/4$ (or $T_{CY}$)
- $F_{OSC}/16$ (or 4 * $T_{CY}$)
- $F_{OSC}/64$ (or 16 * $T_{CY}$)
- Timer2 output/2
- $F_{OSC}/(4 * (SSPxADD + 1))$

**Important:** In Host 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 SSPxCLKPPS register.

chanhvinh.com



**Figure 31-4. SPI Mode Waveform (Host Mode)**




### 31.1.2.2 SPI Client Mode

In Client 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 Client 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.

While in Client 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 Electrical Specifications.

While in Sleep mode, the client 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.

### 31.1.2.3 Daisy-Chain Configuration

The SPI bus can sometimes be connected in a daisy-chain configuration. The first client output is connected to the second client input, the second client output is connected to the third client input, and so on. The final client output is connected to the host input. Each client 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 Client Select line from the host device.

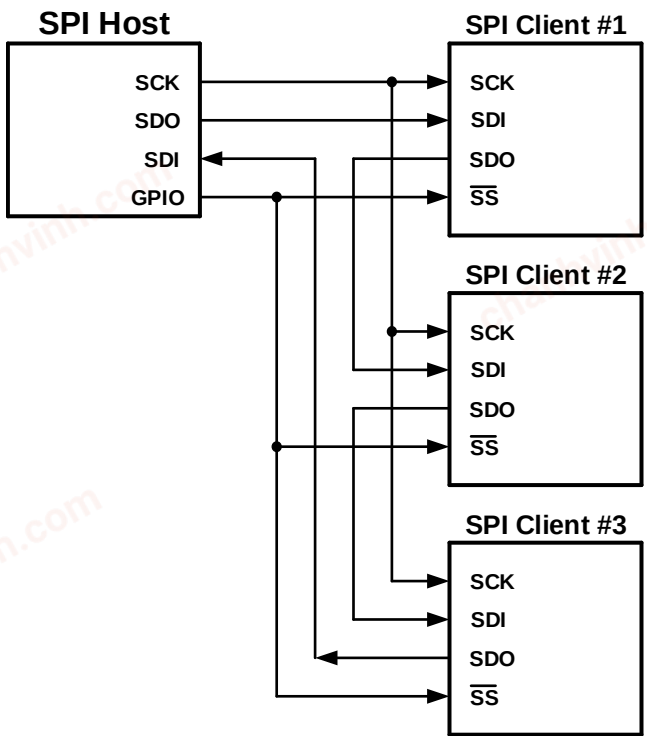
In a daisy-chain configuration, only the most recent byte on the bus is required by the client. Setting the Buffer Overwrite Enable (BOEN) bit will enable writes to the SSPxBUF register, even if the previous byte has not been read. This allows the software to ignore data that may not apply to it.

chanhvinh.com



Figure 31-5 shows the block diagram of a typical daisy-chain connection when operating in SPI mode.

**Figure 31-5. SPI Daisy-Chain Connection**




### 31.1.2.4 Client Select Synchronization

The Client Select can also be used to synchronize communication (see Figure 31-6). The Client Select line is held high until the host device is ready to communicate. When the Client Select line is pulled low, the client knows that a new transmission is starting.

If the client fails to receive the communication properly, it will be reset at the end of the transmission, when the Client Select line returns to a High state. The client is then ready to receive a new transmission when the Client Select line is pulled low again. If the Client Select line is not used, there is a risk that the client will eventually become out of sync with the host. If the client misses a bit, it will always be one bit off in future transmissions. Use of the Client Select line allows the client and host to align themselves at the beginning of each transmission.

The $\overline{SS}$ pin allows a Synchronous Client mode. The SPI must be in Client mode with $\overline{SS}$ pin control enabled (MSSP Mode Select (SSPM) bits = 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.

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.

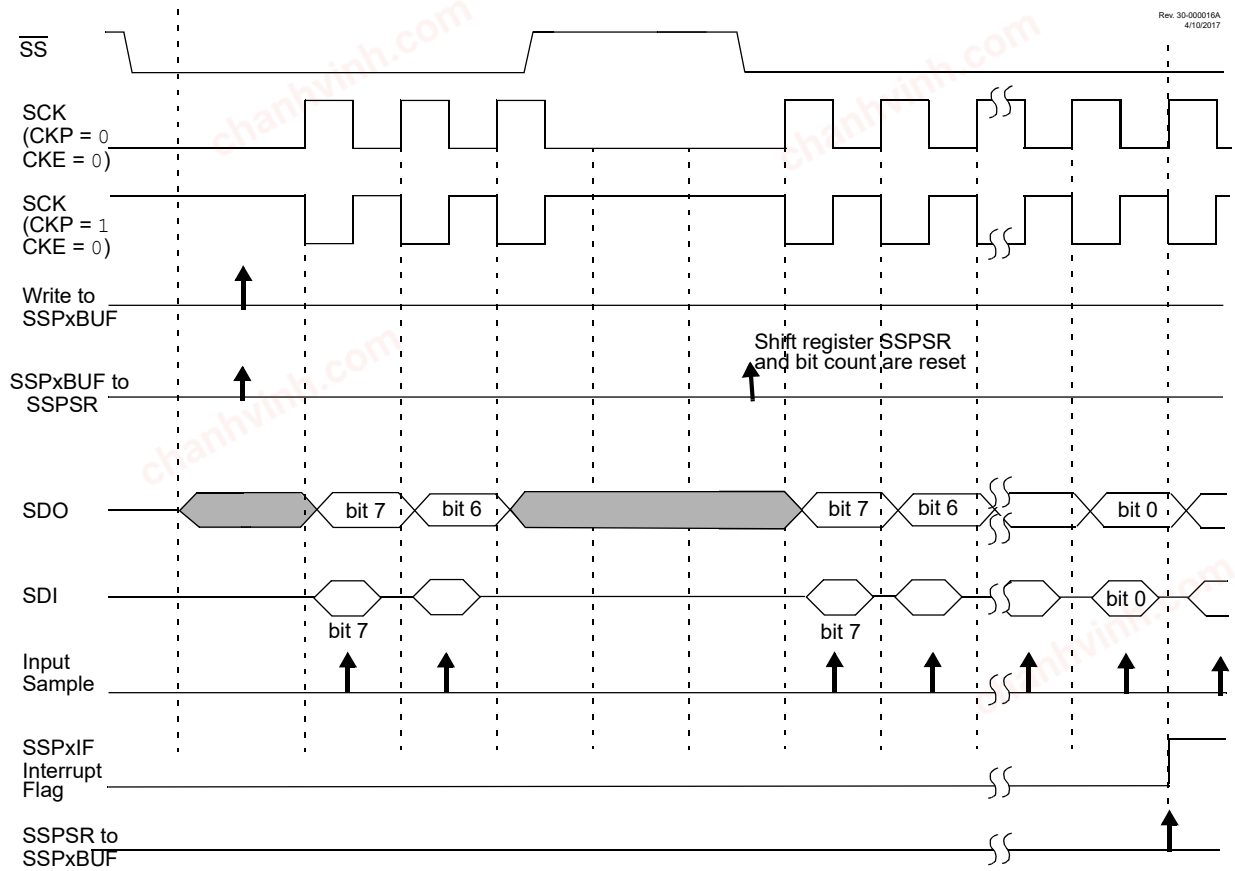
chanhvinh.com



**Important:**

1. When the SPI is in Client mode with $\overline{SS}$ pin control enabled (SSPM = 0100), the SPI module will reset if the $\overline{SS}$ pin is set to $V_{DD}$.
2. When the SPI is used in Client mode with CKE set, the user must enable $\overline{SS}$ pin control (see Figure 31-8). If CKE is clear, $\overline{SS}$ pin control is optional (see Figure 31-7).
3. While operated in SPI Client mode, the SMP bit must remain clear.

**Figure 31-6. Client Select Synchronous Waveform**



chanhvinh.com



**Figure 31-7. SPI Mode Waveform (Client Mode with CKE = 0)**

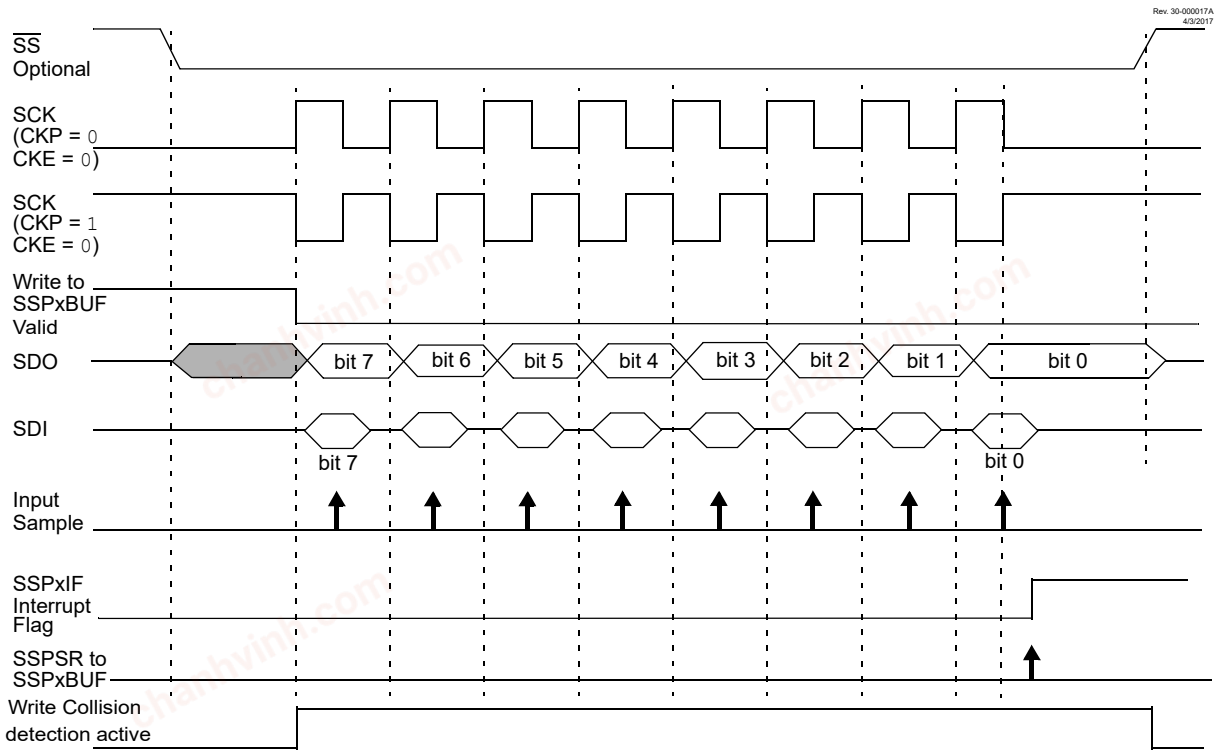
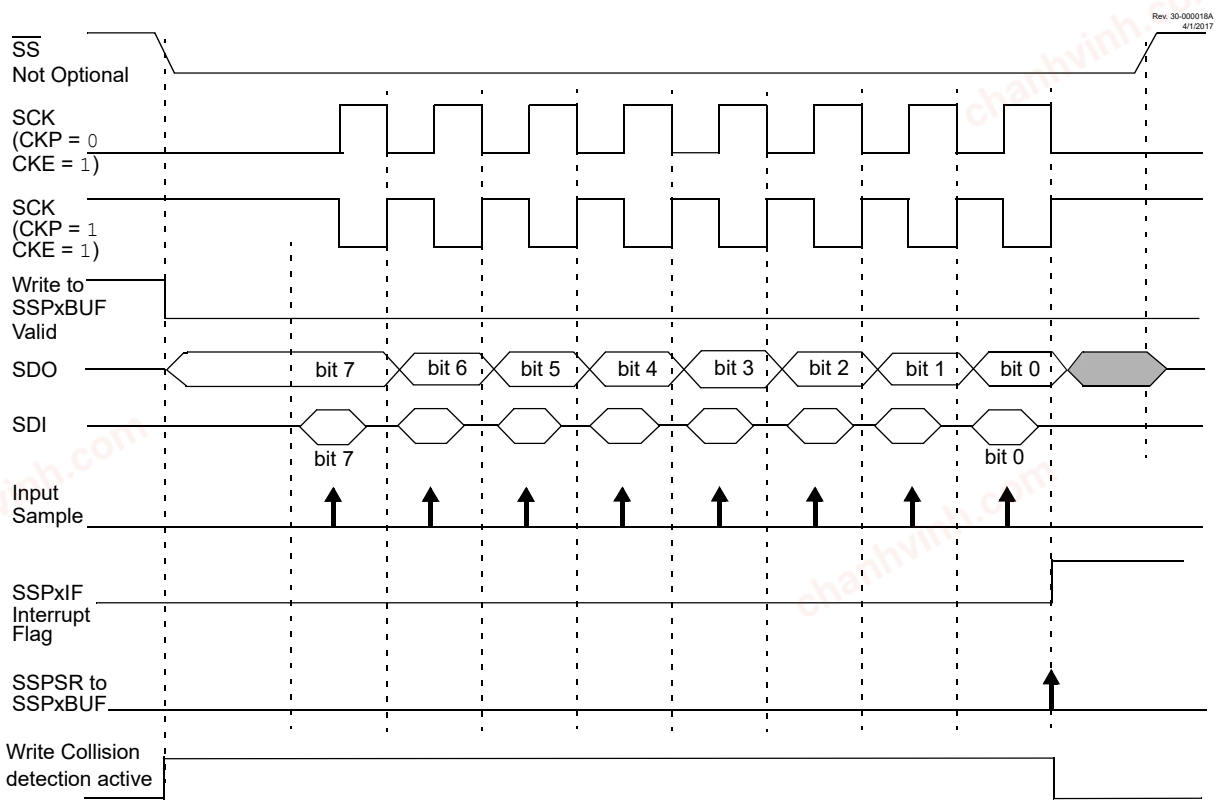



**Figure 31-8. SPI Mode Waveform (Client Mode with CKE = 1)**



chanhvinh.com



#### 31.1.2.5 SPI Operation in Sleep Mode

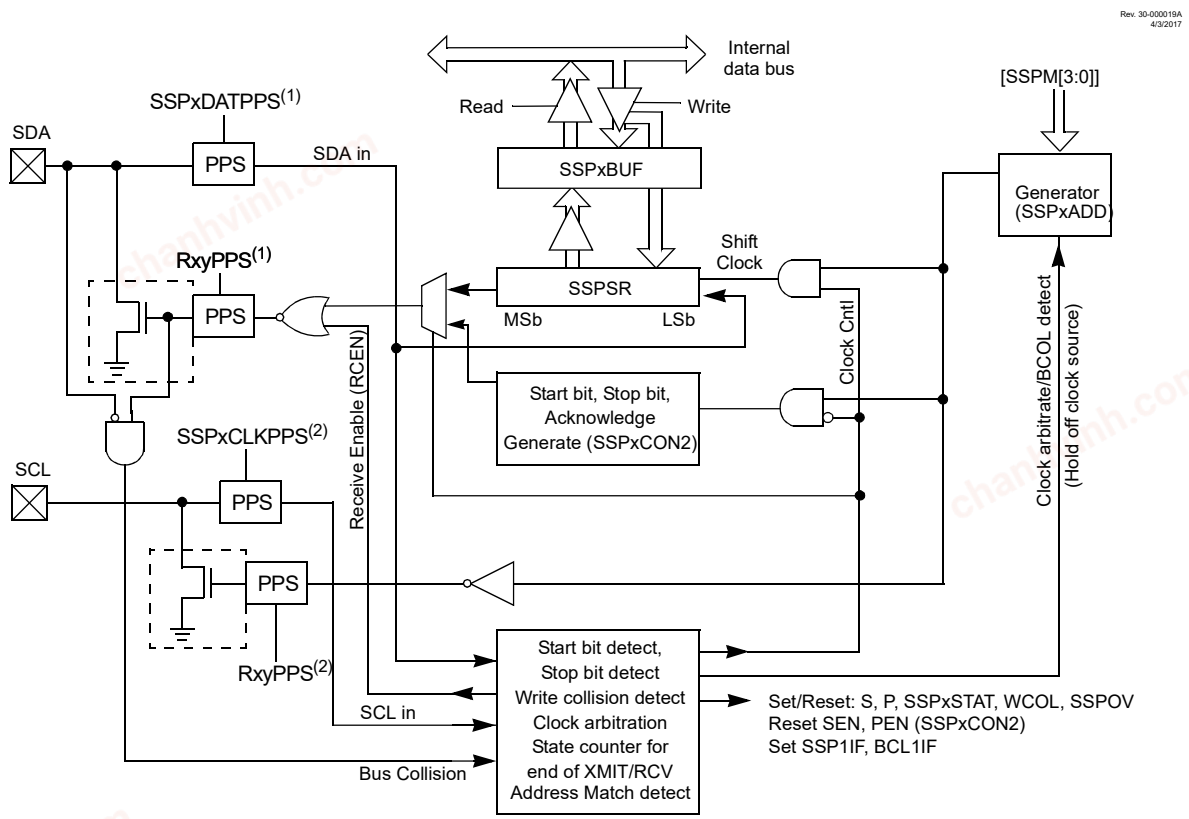
In SPI Host mode, when the Sleep mode is selected, all module clocks are halted and the transmission/reception will remain in that state until the device wakes. After the device returns to Run mode, the module will resume transmitting and receiving data.

In SPI Client 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.

## 31.2 I²C Mode Overview

The Inter-Integrated Circuit ($I^2C$) bus is a multi-host serial data communication bus. Devices communicate in a host/client environment where the host devices initiate the communication. A client device is controlled through addressing. Figure 31-9 and Figure 31-10 show block diagrams of the $I^2C$ Host and Client modes, respectively.

**Figure 31-9. MSSP Block Diagram ($I^2C$ Host Mode)**



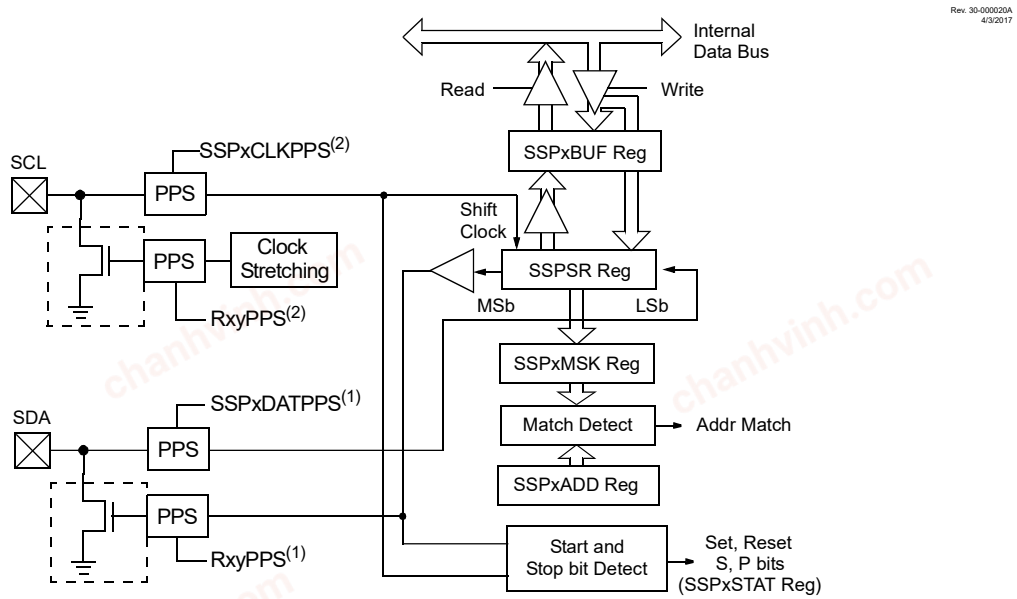

**Notes:** 1. SDA pin selections must be the same for input and output.

2. SCL pin selections must be the same for input and output.

chanhvinh.com



**Figure 31-10. MSSP Block Diagram (I²C Client Mode)**




**Notes:** 1. SDA pin selections must be the same for input and output.
2. SCL pin selections must be the same for input and output.

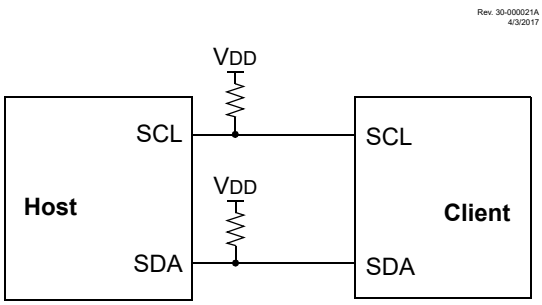
The I²C bus specifies two signal connections:

- Serial Clock (SCL)
- Serial Data (SDA)

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 31-11 shows a typical connection between two processors configured as host and client devices.

**Figure 31-11. I²C Host/Client Connection**




The I²C bus can operate with one or more host devices and one or more client devices.

There are four potential modes of operation for a given device:

- Host Transmit mode (host is transmitting data to a client)
- Host Receive mode (host is receiving data from a client)
- Client Transmit mode (client is transmitting data to a host)
- Client Receive mode (client is receiving data from the host)

To begin communication, the host device transmits a Start condition followed by the address byte of the client it intends to communicate with. A Start condition is indicated by a high-to-low transition of the SDA line while the SCL

chanhvinh.com



line is held high. Address and data bytes are sent out, MSb first. This is followed by a single Read/$\overline{\text{Write}}$ Information (R/$\overline{\text{W}}$) bit, which determines whether the host intends to transmit to or receive data from the client device. The R/$\overline{\text{W}}$ bit is sent out as a logical one when the host intends to read data from the client, and is sent out as a logical zero when it intends to write data to the client.

If the requested client exists on the bus, it will respond with an Acknowledge sequence, otherwise known as an $\overline{\text{ACK}}$. The Acknowledge sequence is an active-low signal, which holds the SDA line low to indicate to the transmitter that the client device has received the transmitted data and is ready to receive more. The host then continues to either transmit to or receive data from the client.

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 conditions.

If the host intends to write to the client, then it repeatedly sends out a byte of data, with the client responding after each byte with an $\overline{\text{ACK}}$ sequence. In this example, the host device is in Host Transmit mode and the client is in Client Receive mode.

If the host intends to read from the client, then it repeatedly receives a byte of data from the client, and responds after each byte with an $\overline{\text{ACK}}$ sequence. In this example, the host device is in Host Receive mode and the client is in Client Transmit mode.

On the last byte of data communicated, the host device may end the transmission by sending a Stop condition. If the host device is in Receive mode, it sends the Stop condition in place of the last $\overline{\text{ACK}}$ sequence. A Stop condition is indicated by a low-to-high transition of the SDA line while the SCL line is held high.

In some cases, the host may want to maintain control of the bus and re-initiate another transmission. If so, the host device may send a Restart condition in place of the Stop condition or last $\overline{\text{ACK}}$ sequence when it is in Receive mode.

The $I^2C$ bus specifies three message protocols:

- Single message where a host writes data to a client.
- Single message where a host reads data from a client.
- Combined message where a host initiates a minimum of two writes, or two reads, or a combination of writes and reads, to one or more clients.

### 31.2.1 $I^2C$ Mode Registers

The MSSP module has eight registers for $I^2C$ operation.

These are:

- MSSP Status Register (SSPxSTAT)
- MSSP Control 1 Register (SSPxCON1)
- MSSP Control 2 Register (SSPxCON2)
- MSSP Control 3 Register (SSPxCON3)
- Serial Receive/Transmit Buffer Register (SSPxBUF)
- MSSP Address Register (SSPxADD)
- $I^2C$ Client Address Mask Register (SSPxMSK)
- MSSP Shift (SSPSR) register – not directly accessible

SSPxCON1, SSPxCON2, SSPxCON3 and SSPxSTAT are the Control and Status registers in $I^2C$ mode operation. The SSPxCON1, SSPxCON2 and SSPxCON3 registers are readable and writable. The lower six bits of the SSPxSTAT are read-only. The upper two bits of the SSPxSTAT are read/write. SSPxMSK holds the client address mask value used in address comparison. SSPxADD contains the client device address when the MSSP is configured in $I^2C$ Client mode. When the MSSP is configured in Host mode, SSPxADD acts as the Baud Rate Generator reload value.

SSPSR is the shift register used for shifting data in or out. SSPxBUF is the buffer register to which data bytes are written to or read from. In receive operations, SSPSR and SSPxBUF together, create a double-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 double-buffered. A write to SSPxBUF will write to both SSPxBUF and SSPSR.

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



### 31.2.2 I²C Mode Operation

All MSSP I²C communication is byte oriented and shifted out MSb first. Eight 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.

#### 31.2.2.1 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/NXP I²C Specification.

**Table 31-1. I²C Terminology**

| Term | Description |
|---|---|
| Transmitter | The device that shifts data out onto the bus |
| Receiver | The device that shifts data in from the bus |
| Host | The device that initiates a transfer, generates clock signals, and terminates a transfer |
| Client | The device addressed by the host |
| Multi-Host | A bus with more than one device that can initiate data transfers |
| Arbitration | Procedure to ensure that only one host 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 host is controlling the bus, and both SDA and SCL lines are high |
| Active | Any time one or more host devices are controlling the bus |
| Addressed Client | Client device that has received a matching address and is actively being clocked by a host |
| Matching Address | Address byte that is clocked into a client that matches the value stored in SSPxADD |
| Write Request | Client receives a matching address with the R/$\overline{W}$ bit clear, and is ready to clock in data |
| Read Request | Host sends an address byte with the R/$\overline{W}$ bit set, indicating that it wishes to clock data out of the client. This data is the next and all following bytes until a Restart or Stop. |
| Clock Stretching | When a device on the bus hold 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 |

#### 31.2.2.2 Byte Format

All communication in I²C is done in 9-bit segments. A byte is sent from a host to a client or vice versa, followed by an Acknowledge sequence 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 the Acknowledge value on the next clock pulse.

The clock signal, SCL, is provided by the host. 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, such as a Start or Stop condition.

#### 31.2.2.3 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 must be configured as inputs by setting the appropriate TRIS bits.

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



**Important:** 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 SSPxDATPPS registers. The SCL input is selected with the SSPxCLKPPS 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.

#### 31.2.2.4 SDA Hold Time

The hold time of the SDA pin is selected by the SDA Hold Time Selection (SDAHT) bit. 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 buses with large capacitance.

#### 31.2.2.5 Clock Stretching

Clock stretching occurs when a device on the bus holds the SCL line low, effectively pausing communication. The client may stretch the clock to allow more time to handle data or prepare a response for the host device. A host 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 client is invisible to the host software and handled by the hardware that generates SCL.

The CKP bit 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.

#### 31.2.2.6 Arbitration

Each host device must monitor the bus for Start and Stop conditions. 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 host 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 SDA float) and a second transmitter holds it to a logical zero (pulls SDA 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 host 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.

Client Transmit mode can also be arbitrated, when a host addresses multiple clients, but this is less common.

#### 31.2.2.7 Start Condition

The $I^2C$ 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 host and signifies the transition of the bus from an Idle to an Active state. Figure 31-12 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 $I^2C$ Specification that states no bus collision can occur on a Start.

#### 31.2.2.8 Stop Condition

A Stop condition is a transition of the SDA line from Low-to-High state while the SCL line is high.

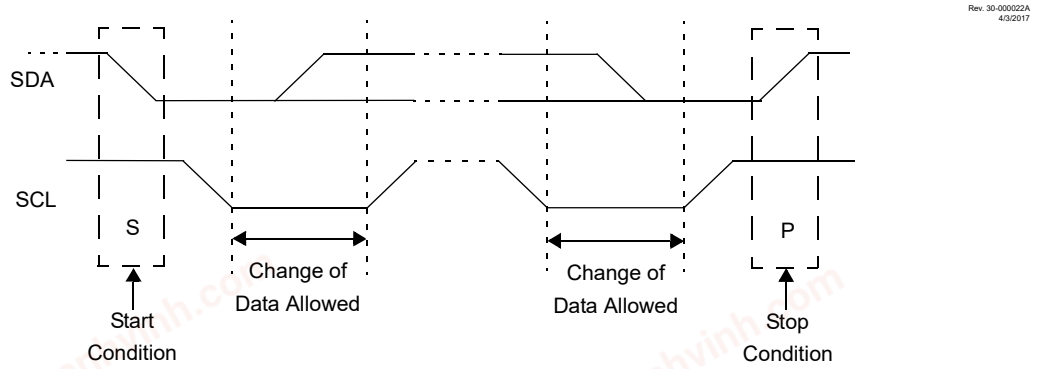
**Important:** At least one SCL low time must appear before a Stop is valid, therefore, if the SDA line goes low then high again while the SCL line stays high, only the Start condition is detected.

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



**Figure 31-12. I²C Start and Stop Conditions**




### 31.2.2.9 Start/Stop Condition Interrupt Masking

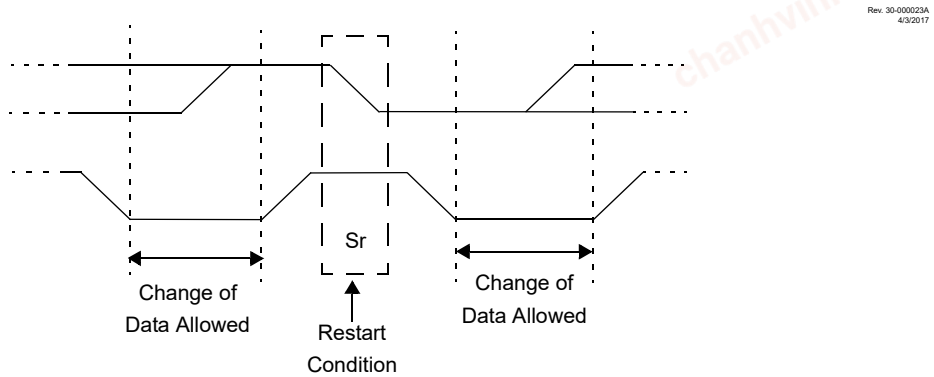
The Start Condition Interrupt Enable (SCIE) and Stop Condition Interrupt Enable (PCIE) bits can enable the generation of an interrupt in Client modes that do not typically support this function. These bits will have no effect in Client modes where interrupt on Start and Stop detect are already enabled.

### 31.2.2.10 Restart Condition

A Restart condition is valid any time that a Stop is valid. A host can issue a Restart if it wishes to hold the bus after terminating the current transfer. A Restart has the same effect on the client that a Start would, resetting all client logic and preparing it to clock in an address. The host may want to address the same or another client. Figure 31-13 shows the waveform for a Restart condition.

In 10-bit Addressing Client mode, a Restart is required for the host to clock data out of the addressed client. Once a client has been fully addressed, matching both high and low address bytes, the host can issue a Restart and the high address byte with the R/$\overline{W}$ bit set. The client logic will then hold the clock and prepare to clock out data.

**Figure 31-13. I²C Restart Condition**




### 31.2.2.11 Acknowledge Sequence

The ninth SCL pulse for any transferred byte in I²C is dedicated as an Acknowledge sequence ($\overline{ACK}$). 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 ($\overline{ACK}$) 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 $\overline{ACK}$ is placed in the Acknowledge Status (ACKSTAT) bit.

The client software, when the Address Hold Enable (AHEN) and Data Hold Enable (DHEN) bits are set, allows the user to select the $\overline{ACK}$ value sent back to the transmitter. The Acknowledge Data (ACKDT) bit is set/cleared to determine the response.

The client hardware will generate an $\overline{ACK}$ response under most circumstances. However, if the BF bit or the Receive Overflow Indicator (SSPOV) bit are set when a byte is received then the $\overline{ACK}$ will not be sent by the client.

chanhvinh.com



When the module is addressed, after the eighth falling edge of SCL on the bus, the Acknowledge Time Status (ACKTIM) bit is set. The ACKTIM bit indicates the acknowledge time of the active bus. The ACKTIM bit is only active when either the AHEN bit or DHEN bit is enabled.

### 31.2.3 I²C Client Mode Operation

The MSSP Client mode operates in one of four modes selected by the MSSP Mode Select (SSPM) bits. 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 condition interrupts operate the same as the other modes with SSPxIF additionally getting set upon detection of a Start, Restart, or Stop condition.

#### 31.2.3.1 Client Mode Addresses

The SSPxADD register contains the Client 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 SSPxMSK register affects the address matching process. See 31.2.3.5. SSP Mask Register for more information.

##### 31.2.3.1.1 I²C Client 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.

##### 31.2.3.1.2 I²C Client 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 MSbs of the 10-bit address and stored in bits 2 and 1 of the SSPxADD register.

After the acknowledge of the high byte the Update Address (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 client is addressed and clocking in the high address with the R/$\overline{W}$ bit set. The client hardware will then acknowledge the read request and prepare to clock out data. This is only valid for a client after it has received a complete high and low address byte match.

#### 31.2.3.2 Clock Stretching

Clock stretching occurs when a device on the bus holds the SCL line low, effectively pausing communication. The client may stretch the clock to allow more time to handle data or prepare a response for the host device. A host 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 client is invisible to the host software and handled by the hardware that generates SCL.

The CKP bit 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.

##### 31.2.3.2.1 Normal Clock Stretching

Following an $\overline{ACK}$ if the R/$\overline{W}$ bit is set (a read request), the client hardware will clear CKP. This allows the client time to update SSPxBUF with data to transfer to the host. If the Stretch Enable (SEN) bit is set, the client hardware will always stretch the clock after the $\overline{ACK}$ sequence. Once the client is ready; CKP is set by software and communication resumes.

##### 31.2.3.2.2 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.

##### 31.2.3.2.3 Byte NACKing

When the AHEN bit is set, CKP is cleared by hardware after the eighth falling edge of SCL for a received matching address byte. When the DHEN bit is set, CKP is cleared after the eighth falling edge of SCL for received data.

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



Stretching after the eighth falling edge of SCL allows the client to look at the received address or data and decide if it wants to acknowledge ($\overline{ACK}$) the received address or data, or not acknowledge (NACK) the address or data.

#### 31.2.3.3 Clock Synchronization and the CKP Bit

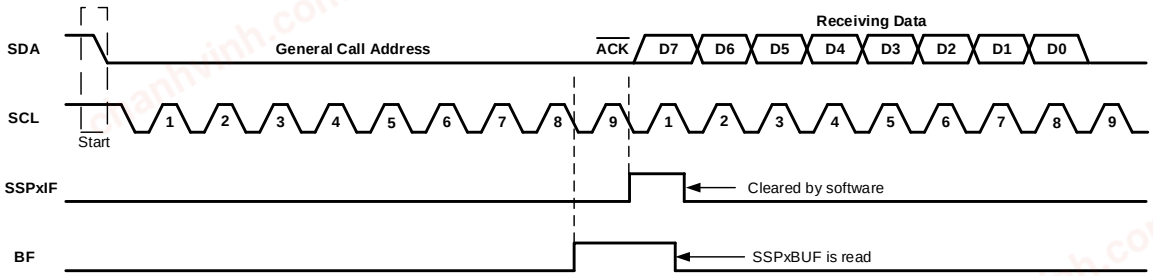
Any time the CKP bit is cleared, the module will wait for the SCL line to go low and then hold it. However, clearing the CKP bit will not assert the SCL output low until the SCL output is already sampled low. Therefore, the CKP bit will not assert the SCL line until an external $I^2C$ host device has already asserted the SCL line. The SCL output will remain low until the CKP bit is set and all other devices on the $I^2C$ bus have released SCL.

#### 31.2.3.4 General Call Address Support

The addressing procedure for the $I^2C$ bus is such that the first byte after the Start condition usually determines which device will be the client addressed by the host device. The exception is the General Call address that can address all devices. When this address is used, all devices must, in theory, respond with an $\overline{ACK}$.

The general call address is a reserved address in the $I^2C$ protocol, defined as address 0x00. When the General Call Enable (GCEN) bit is set, the client module will automatically $\overline{ACK}$ the reception of this address regardless of the value stored in SSPxADD. After the client clocks in an address of all zeros with the R/$\overline{W}$ bit clear, an interrupt is generated and client software can read SSPxBUF and respond. Figure 31-14 shows a General Call reception sequence.

**Figure 31-14. Client Mode General Call Address Sequence**




In 10-bit Address mode, the UA bit will not be set on the reception of the general call address. The client will prepare to receive the second byte as data, just as it would in 7-bit mode.

If the AHEN bit is set, just as with any other address reception, the client hardware will stretch the clock after the eighth falling edge of SCL. The client must then set its Acknowledge Sequence Enable (ACKEN) bit and release the clock.

#### 31.2.3.5 SSP Mask Register

The MSSP Mask (SSPxMSK) register is available in $I^2C$ Client 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 MSSP operation until written with a mask value.

SSPxMSK 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 MSSP mask has no effect during the reception of the first (high) byte of the address.

#### 31.2.3.6 Client Reception

When the R/$\overline{W}$ bit of a matching received address byte is clear, the R/$\overline{W}$ bit is cleared. The received address is loaded into the SSPxBUF register and acknowledged.

When the Overflow condition exists for a received address, a Not Acknowledge (NACK) is transmitted and the Receive Overflow Indicator (SSPOV) bit is set. The Buffer Override Enable (BOEN) bit modifies this operation.

chanhvinh.com



An MSSP interrupt is generated for each transferred data byte. The SSPxIF flag bit must be cleared by software.

When the SEN bit is set, SCL will be held low (clock stretch) following each received byte. The clock must be released by setting the CKP bit, except sometimes in 10-bit mode. See 31.2.3.2.2. 10-Bit Addressing Mode for more details.

#### 31.2.3.6.1 7-Bit Addressing Reception

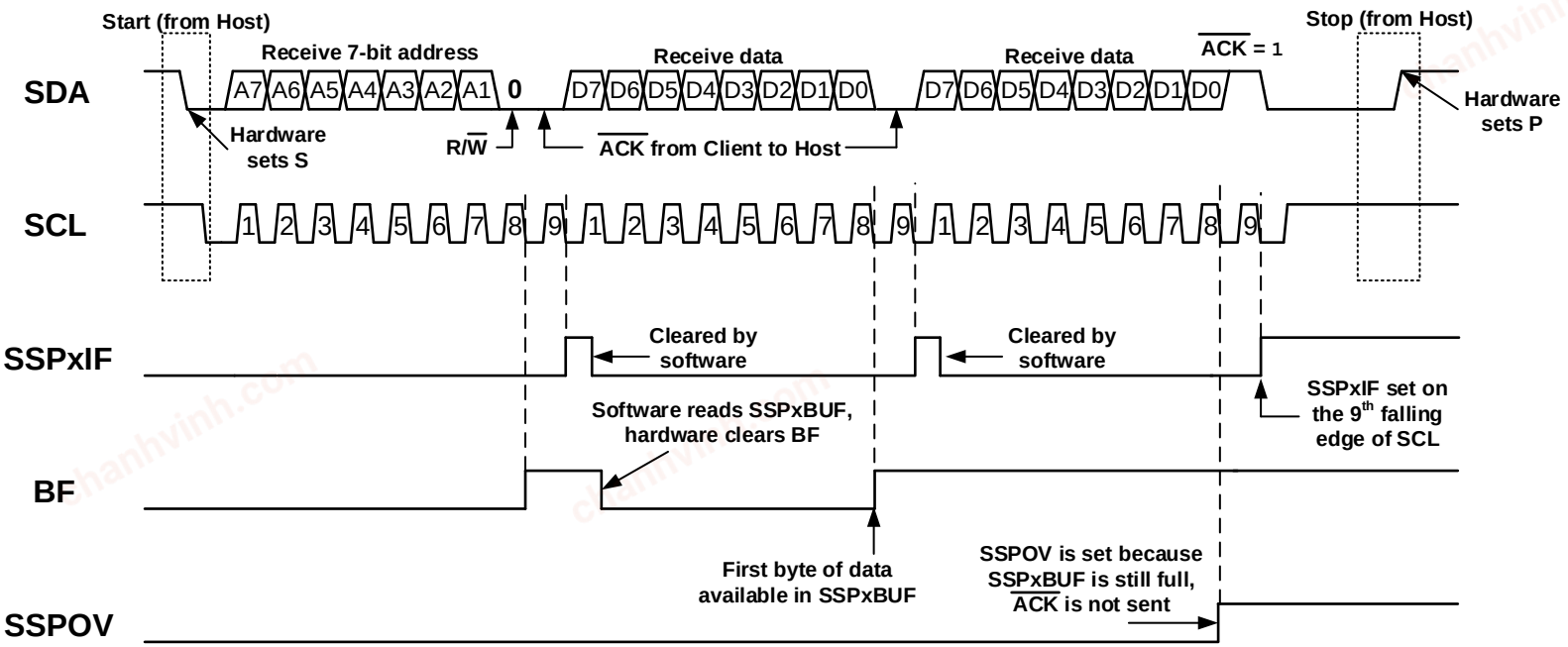
This section describes a standard sequence of events for the MSSP module configured as an $I^2C$ client in 7-bit Addressing mode. Figure 31-15 and Figure 31-16 are used as a visual reference for this description.

This is a step by step process of what typically must be done to accomplish $I^2C$ communication.

1. Start condition detected.
2. The Start (S) bit is set; SSPxIF is set if the Start Condition Interrupt Enable (SCIE) bit is set.
3. Matching address with R/$\overline{W}$ bit clear is received.
4. The client pulls SDA low, sending an $\overline{ACK}$ to the host, 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; Client software sets the CKP bit to release the SCL line.
8. The host clocks out a data byte.
9. Client drives SDA low, sending an $\overline{ACK}$ to the host, 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 host.
13. Host sends Stop condition, setting the Stop (P) bit, and the bus goes Idle.

chanhvinh.com

**Figure 31-15. I²C Client, 7-Bit Address, Reception (SEN = 0, AHEN = 0, DHEN = 0)**

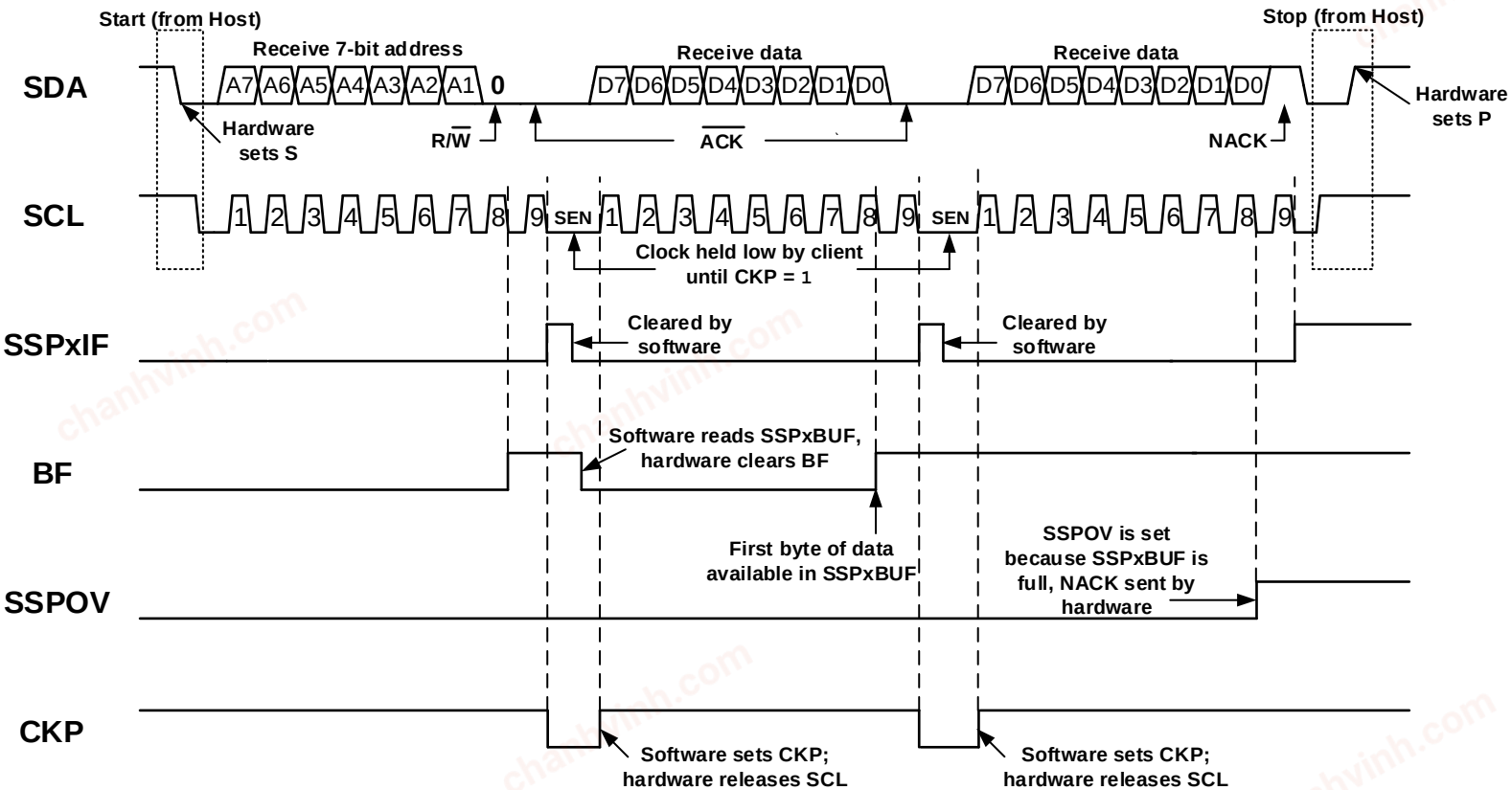





© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com

**Figure 31-16. I²C Client, 7-Bit Address, Reception (SEN = 1, AHEN = 0, DHEN = 0)**






© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



#### 31.2.3.6.2 7-Bit Reception with AHEN and DHEN

Client 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 client software to decide whether it wants to $\overline{ACK}$ 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 client software to use these options for $I^2C$ communication. Figure 31-17 displays a module using both address and data holding. Figure 31-18 includes the operation with the SEN bit set.

1. The Start (S) bit is set; SSPxIF is set if SCIE is set.
2. Matching address with the R/$\overline{W}$ bit clear is clocked in. SSPxIF is set and CKP cleared after the eighth falling edge of SCL.
3. Software clears the SSPxIF.
4. Client can look at the ACKTIM bit to determine if the SSPxIF was after or before the $\overline{ACK}$.
5. Client reads the address value from SSPxBUF, clearing the BF flag.
6. Client transmits an $\overline{ACK}$ to the host by clearing ACKDT.
7. Client releases the clock by setting CKP.
8. SSPxIF is set after an $\overline{ACK}$, not after a NACK.
9. If SEN = 1, the client hardware will stretch the clock after the $\overline{ACK}$.
10. Client clears SSPxIF.

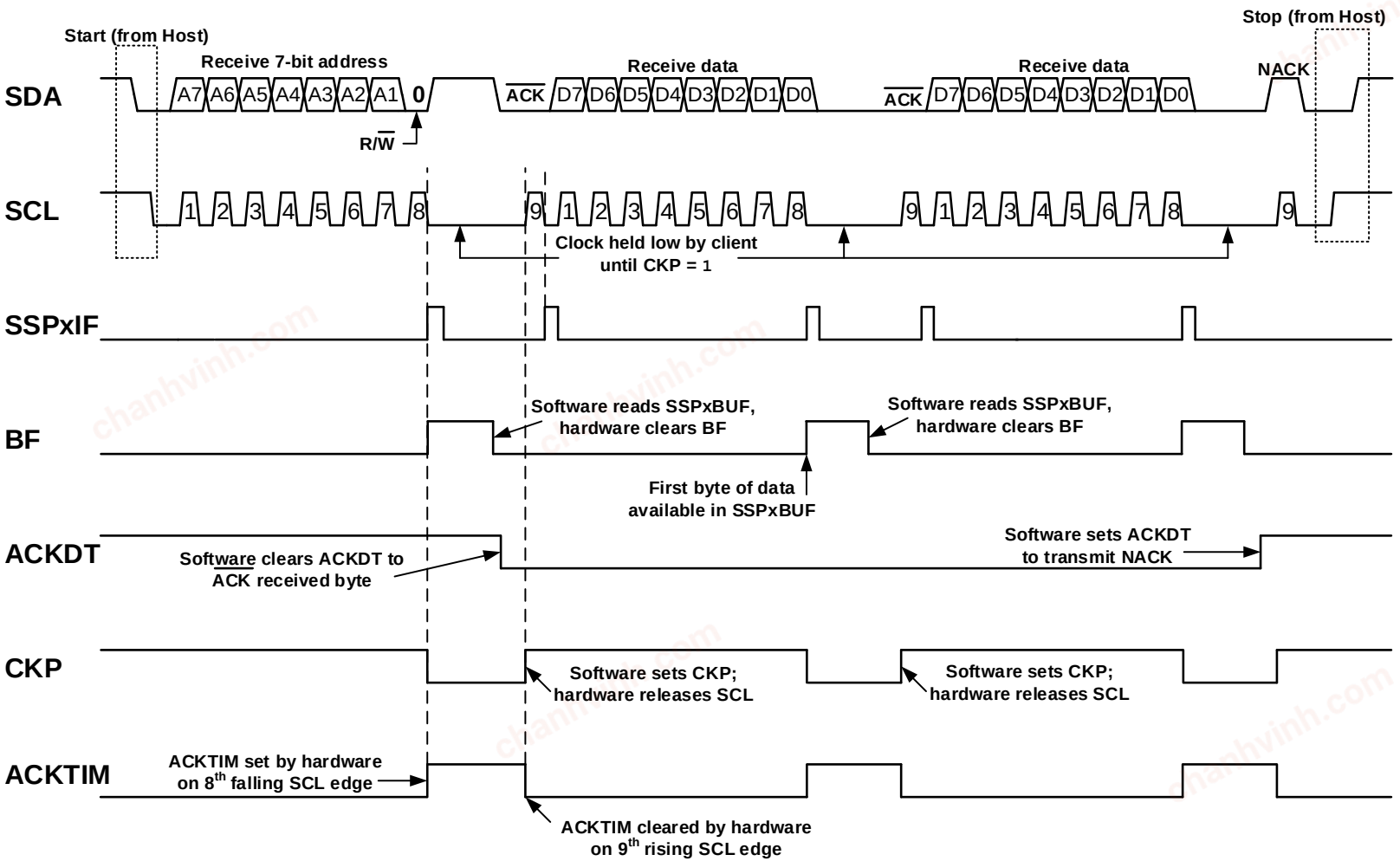
**Important:** SSPxIF is still set after the ninth falling edge of SCL even if there is no clock stretching and BF has been cleared. Only if a NACK is sent to the host is SSPxIF not set.

11. SSPxIF is set and CKP cleared after eighth falling edge of SCL for a received data byte.
12. Client looks at the ACKTIM bit to determine the source of the interrupt.
13. Client 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 client sending a NACK, or the host sending a Stop condition. If a Stop is sent and the Stop Condition Interrupt Enable (PCIE) bit is clear, the client will only know by polling the Stop (P) bit.

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com

**Figure 31-17. I²C Client, 7-Bit Address, Reception (SEN = 0, AHEN = 1, DHEN = 1)**

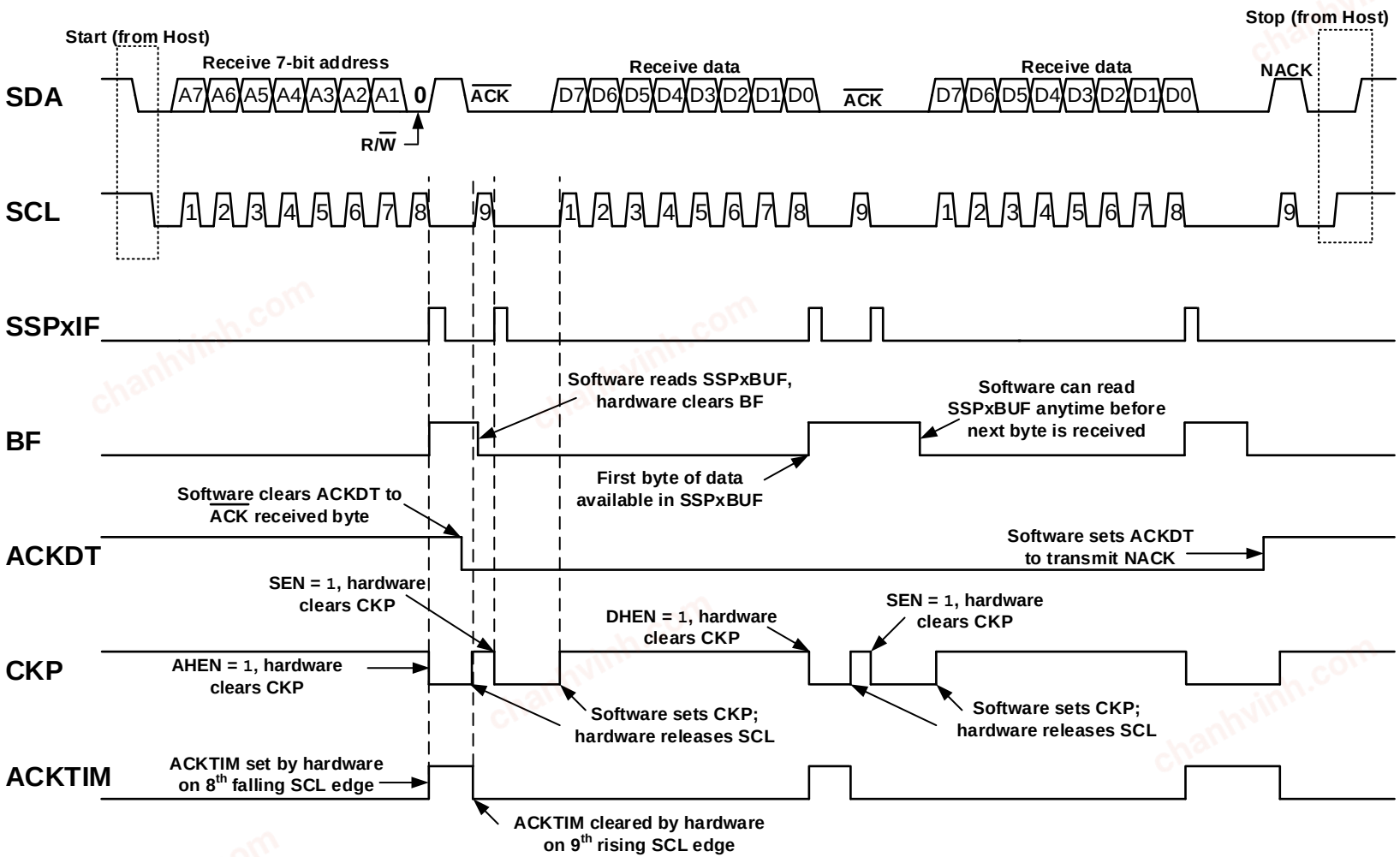





© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com

**Figure 31-18. I²C Client, 7-Bit Address, Reception (SEN = 1, AHEN = 1, DHEN = 1)**






© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



#### 31.2.3.6.3 Client Mode 10-Bit Address Reception

This section describes a standard sequence of events for the MSSP module configured as an $I^2C$ client in 10-bit Addressing mode. Figure 31-19 shows a standard waveform for a client receiver in 10-bit Addressing mode with clock stretching enabled.

This is a step-by-step process of what must be done by the client software to accomplish $I^2C$ communication.

1. Bus starts Idle.
2. Host sends Start condition; S bit is set; SSPxIF is set if SCIE is set.
3. Host sends matching high address with the R/$\overline{W}$ bit clear; the UA bit is set.
4. Client 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. Client loads low address into SSPxADD, releasing SCL.
8. Host sends matching low address byte to the client; UA bit is set.

**Important:** Updates to the SSPxADD register are not allowed until after the $\overline{ACK}$ sequence.

9. Client sends $\overline{ACK}$ and SSPxIF is set.

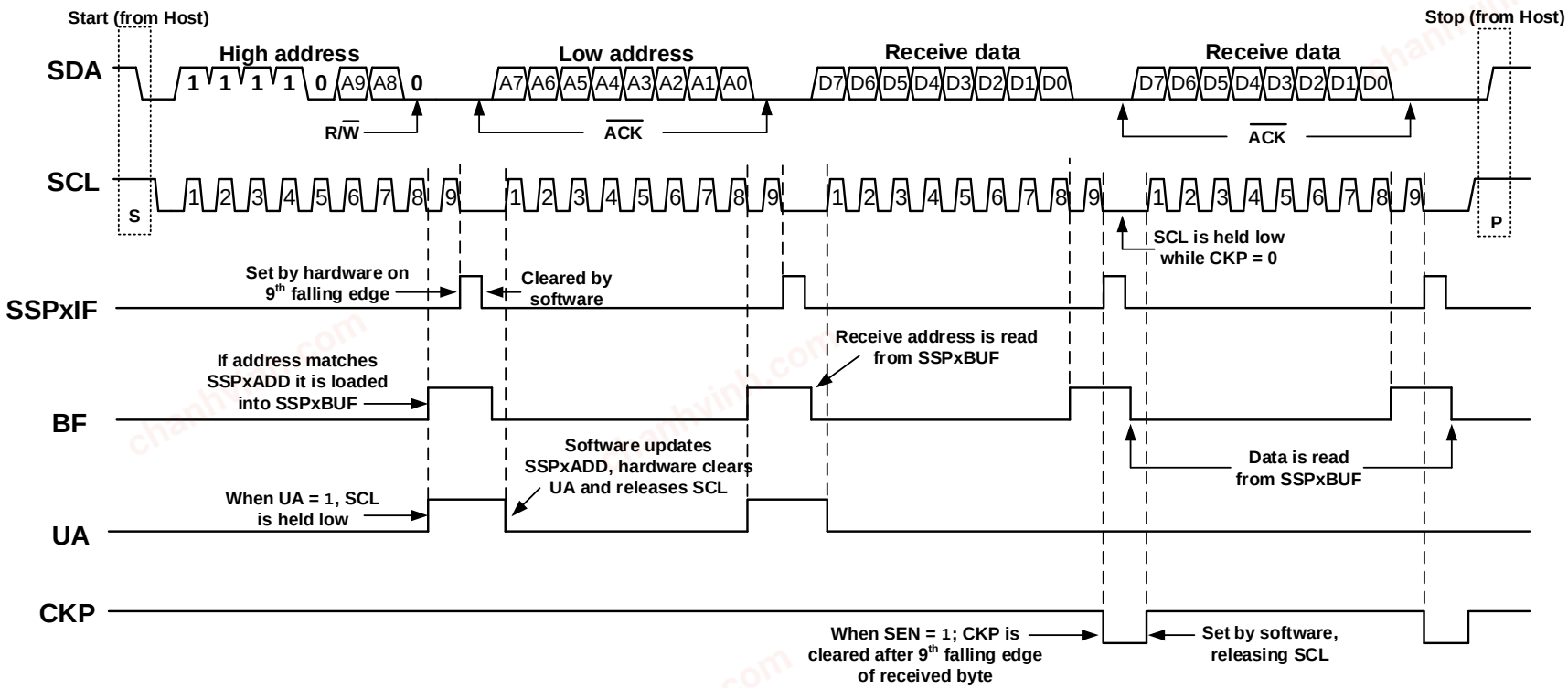
**Important:** If the low address does not match, SSPxIF and UA are still set so that the client software can set SSPxADD back to the high address. BF is not set because there is no match. CKP is unaffected.

10. Client clears SSPxIF.
11. Client reads the received matching address from SSPxBUF, clearing BF.
12. Client loads high address into SSPxADD.
13. Host clocks a data byte to the client and clocks out the client $\overline{ACK}$ on the ninth SCL pulse; SSPxIF is set.
14. If the SEN bit is set, CKP is cleared by hardware and the clock is stretched.
15. Client clears SSPxIF.
16. Client reads the received byte from SSPxBUF, clearing BF.
17. If SEN is set the client software sets CKP to release the SCL.
18. Steps 13-17 repeat for each received byte.
19. Host sends Stop to end the transmission.

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com

**Figure 31-19. I²C Client, 10-Bit Address, Reception (SEN = 1, AHEN = 0, DHEN = 0)**






© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



#### 31.2.3.6.4 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 31-20 can be used as a reference of a client in 10-bit addressing with AHEN set.

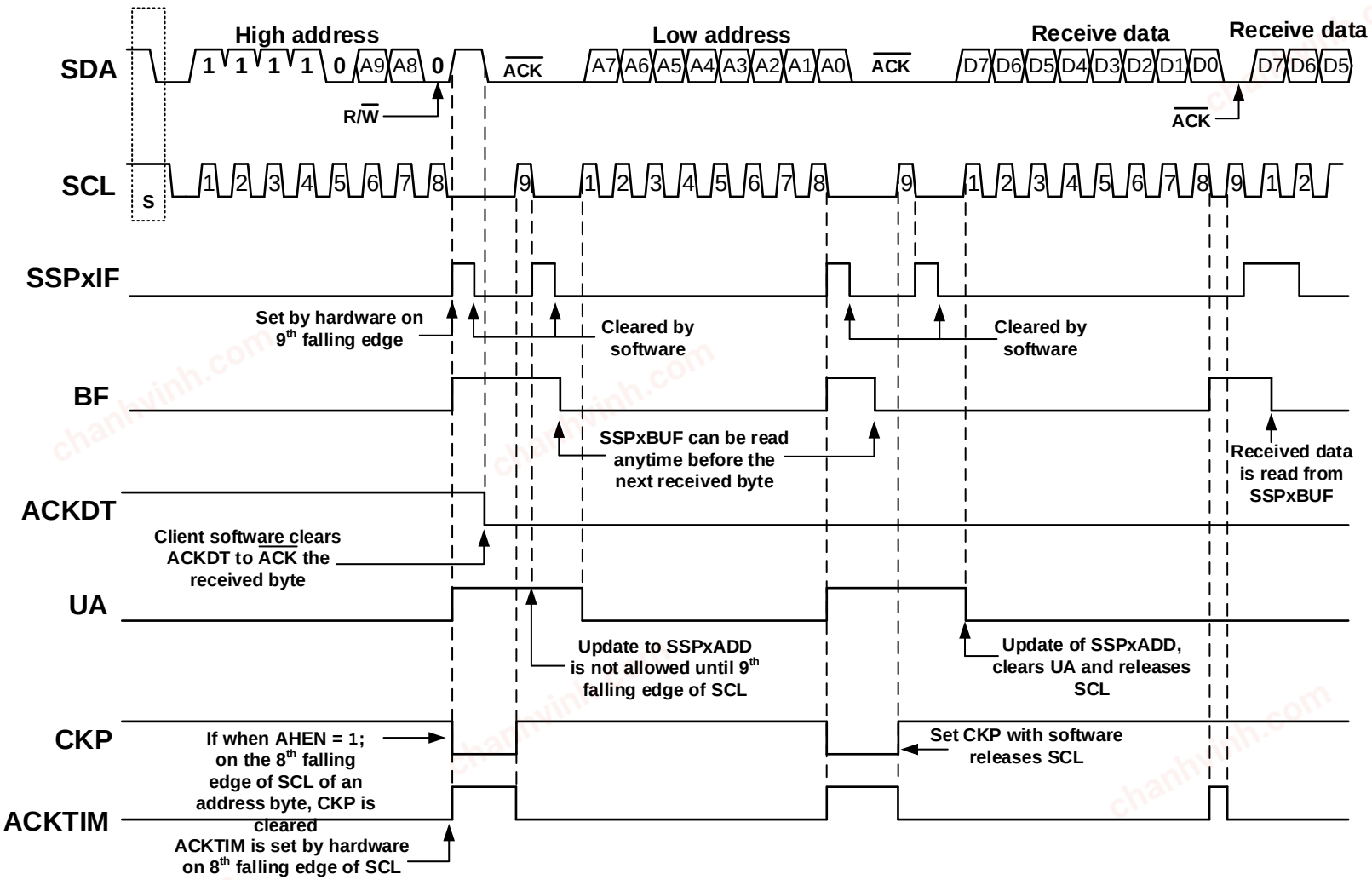
Figure 31-21 shows a standard waveform for a client transmitter in 10-bit Addressing mode.

chanhvinh.com

chanhvinh.com



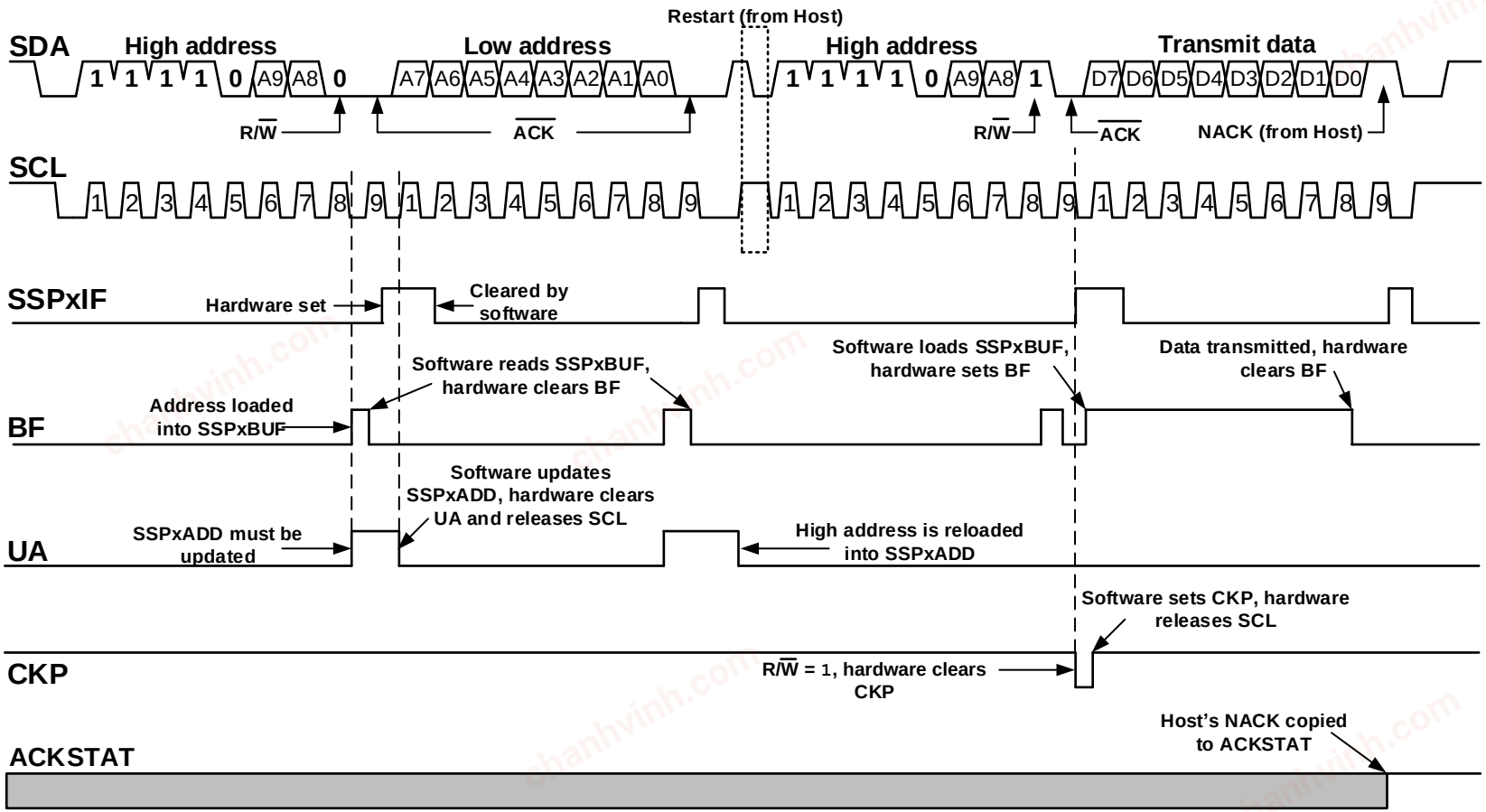
**Figure 31-20. I²C Client, 10-Bit Address, Reception (SEN = 0, AHEN = 1, DHEN = 0)**




© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com

**Figure 31-21. I²C Client, 10-Bit Address, Transmission (SEN = 0, AHEN = 0, DHEN = 0)**




© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



#### 31.2.3.7 Client Transmission

When the R/$\overline{W}$ bit of the incoming address byte is set and an address match occurs, the R/$\overline{W}$ bit is set. The received address is loaded into the SSPxBUF register, and an $\overline{ACK}$ pulse is sent by the client on the ninth bit.

Following the $\overline{ACK}$, client hardware clears the CKP bit and the SCL pin is held low (see 31.2.2.5. Clock Stretching for more details). By stretching the clock, the host will be unable to assert another clock pulse until the client 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 will be released by setting the CKP bit. 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 host receiver is latched on the rising edge of the ninth SCL input pulse. This $\overline{ACK}$ value is copied to the ACKSTAT bit. If ACKSTAT is set (NACK), then the data transfer is complete. In this case, when the NACK is latched by the client, the client goes Idle and waits for another occurrence of a Start condition. 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.

##### 31.2.3.7.1 Client Mode Bus Collision

A client receives a read request and begins shifting data out on the SDA line. If a bus collision is detected and the Client Mode Bus Collision Detect Enable (SBCDE) bit is set, the Bus Collision Interrupt Flag (BCLxIF) bit of the PIRx register is set. Once a bus collision is detected, the client goes Idle and waits to be addressed again. User software can use the BCLxIF bit to handle a client bus collision.

##### 31.2.3.7.2 7-Bit Transmission

A host device can transmit a read request to a client, and then clock data out of the clien. The list below outlines what software for a client will need to do to accomplish a standard transmission. Figure 31-22 can be used as a reference to this list.

1. Host sends a Start condition.
2. The Start (S) bit is set; SSPxIF is set if SCIE is set.
3. Matching address with R/$\overline{W}$ bit set is received by the Client, setting SSPxIF bit.
4. Client hardware generates an $\overline{ACK}$ and sets SSPxIF.
5. The SSPxIF bit is cleared by software.
6. Software reads the received address from SSPxBUF, clearing BF.
7. R/$\overline{W}$ is set so CKP was automatically cleared after the $\overline{ACK}$.
8. The client software loads the transmit data into SSPxBUF.
9. CKP bit is set by software, releasing SCL, allowing the host to clock the data out of the client.
10. SSPxIF is set after the $\overline{ACK}$ response from the host is loaded into the ACKSTAT bit.
11. SSPxIF bit is cleared.
12. The client software checks the ACKSTAT bit to see if the host wants to clock out more data.

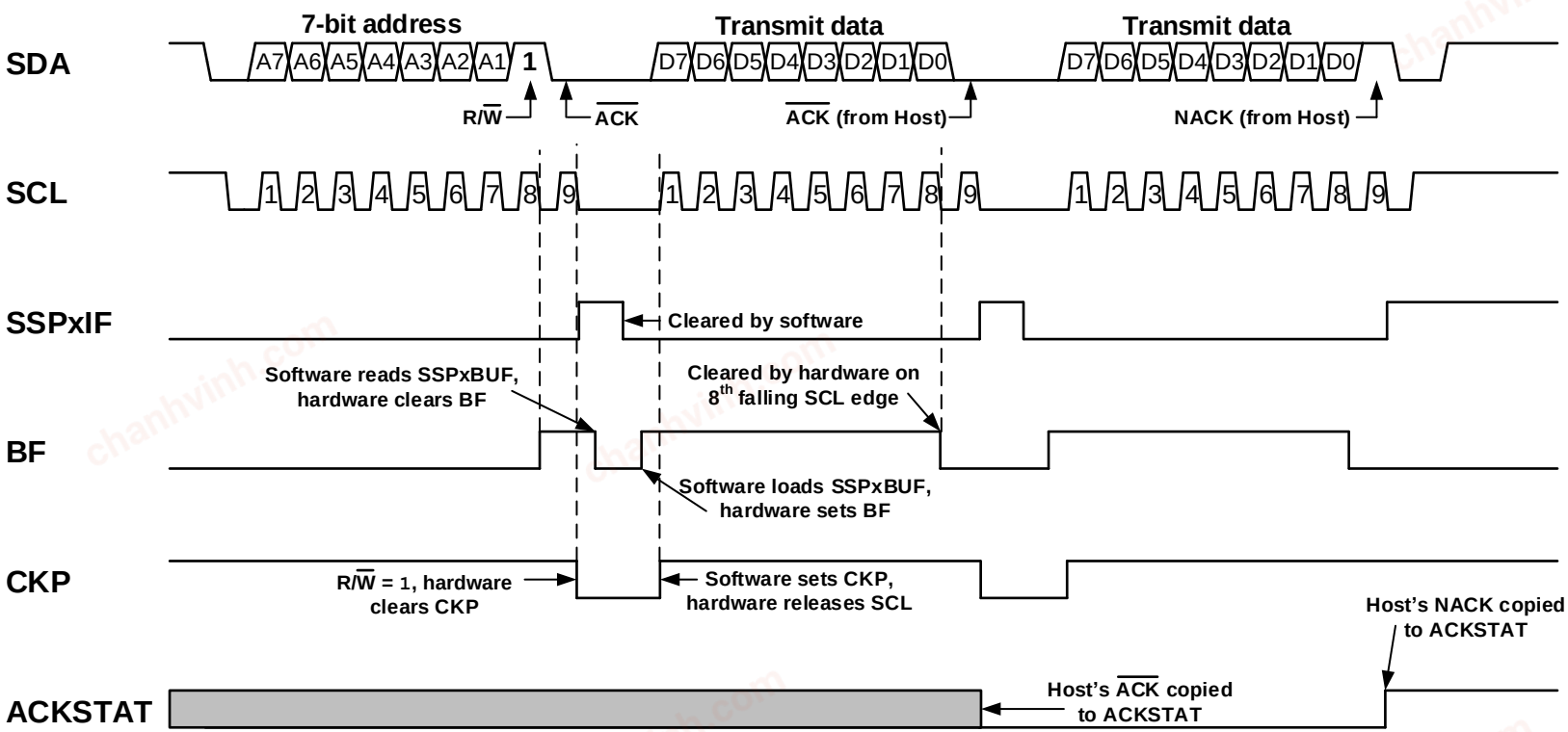
**Important:**

1. If the host $\overline{ACK}$s then the clock will be stretched.
2. ACKSTAT is the only bit updated on the rising edge of the ninth SCL clock instead of the falling edge.

13. Steps 9-13 are repeated for each transmitted byte.
14. If the host sends a not $\overline{ACK}$; the clock is not held, but SSPxIF is still set.
15. The host sends a Restart condition or a Stop.

chanhvinh.com

**Figure 31-22. I²C Client, 7-Bit Address, Transmission (AHEN = 0)**






© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



#### 31.2.3.7.3 7-Bit Transmission with Address Hold Enabled

Setting the AHEN bit enables additional clock stretching and interrupt generation after the eighth falling edge of a received matching address. Once a matching address has been clocked in, CKP is cleared and the SSPxIF interrupt is set.

Figure 31-23 displays a standard waveform of a 7-bit address client transmission with AHEN enabled.

1. Bus starts Idle.
2. Host sends Start condition; the S bit is set; SSPxIF is set if SCIE is set.
3. Host sends matching address with the 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. Client software clears SSPxIF.
5. Client software reads the ACKTIM, R/$\overline{W}$ and D/$\overline{A}$ bits to determine the source of the interrupt.
6. Client reads the address value from the SSPxBUF register, clearing the BF bit.
7. Client software decides from this information if it wishes to $\overline{ACK}$ or NACK and sets the ACKDT bit accordingly.
8. Client software sets the CKP bit, releasing SCL.
9. Host clocks in the $\overline{ACK}$ value from the client.
10. Client hardware automatically clears the CKP bit and sets SSPxIF after $\overline{ACK}$ if the R/$\overline{W}$ bit is set.
11. Client software clears SSPxIF.
12. Client loads value to transmit to the host into SSPxBUF, setting the BF bit.

**Important:** SSPxBUF cannot be loaded until after the $\overline{ACK}$.

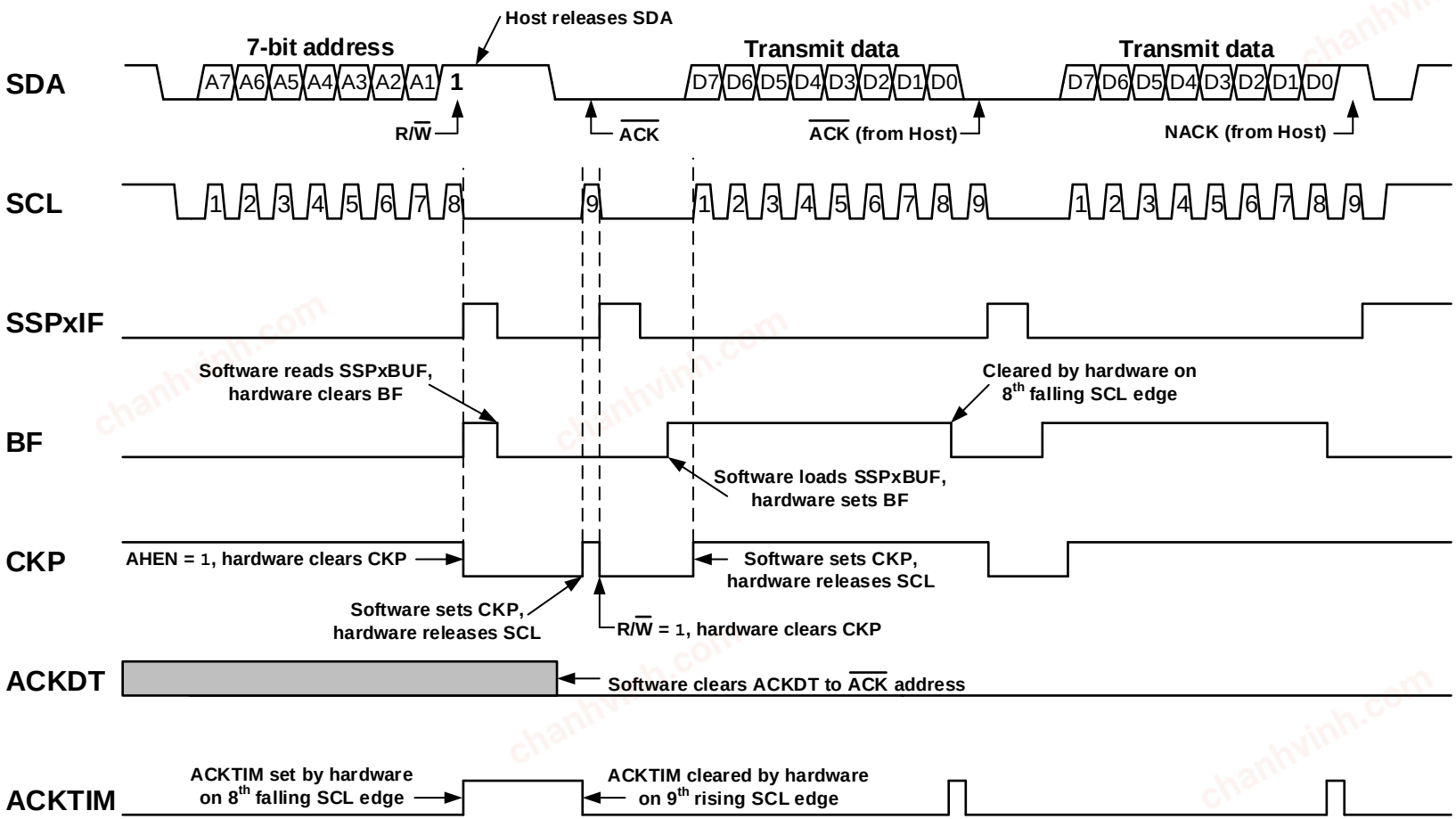
13. Client software sets the CKP bit, releasing the clock.
14. Host clocks out the data from the client and sends an $\overline{ACK}$ value on the ninth SCL pulse.
15. Client hardware copies the $\overline{ACK}$ value into the ACKSTAT bit.
16. Steps 10-15 are repeated for each byte transmitted to the host from the client.
17. If the host sends a not $\overline{ACK}$, the client releases the bus allowing the host to send a Stop and end the communication.

**Important:** Host must send a not $\overline{ACK}$ on the last byte to ensure that the client releases the SCL line to receive a Stop.

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com

**Figure 31-23. I²C Client, 7-Bit Address, Transmission (AHEN = 1)**






© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



### 31.2.4 I²C Host Mode

Host mode is enabled by configuring the appropriate SSPM bits and setting the SSPEN bit. In Host mode, the SDA and SCL pins must be configured as inputs. The MSSP peripheral hardware will override the output driver TRIS controls when necessary to drive the pins low.

Host 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 Host mode, user code conducts all I²C bus operations based on Start and Stop 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 MSSP Interrupt Flag (SSPxIF) bit to be set (MSSP interrupt, if enabled):

- Start condition detected
- Stop condition detected
- Data transfer byte transmitted/received
- Acknowledge transmitted/received
- Repeated Start generated

**Important:**

1. The MSSP module, when configured in I²C Host 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, SSPxBUF will not be written to and the Write Collision Detect (WCOL) bit will be set, indicating that a write to SSPxBUF did not occur.
2. Host 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.

#### 31.2.4.1 I²C Host Mode Operation

The host 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 Host Transmitter mode, serial data is output through SDA, while SCL outputs the serial clock. The first byte transmitted contains the client address of the receiving device (seven bits) and the R/$\overline{W}$ bit. In this case, the R/$\overline{W}$ 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 Host Receive mode, the first byte transmitted contains the client address of the transmitting device (seven bits) and the R/$\overline{W}$ bit. In this case, the R/$\overline{W}$ bit will be logic '1'. Thus, the first byte transmitted is a 7-bit client 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 sequence 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 31.3. Baud Rate Generator for more details.

##### 31.2.4.1.1 Clock Arbitration

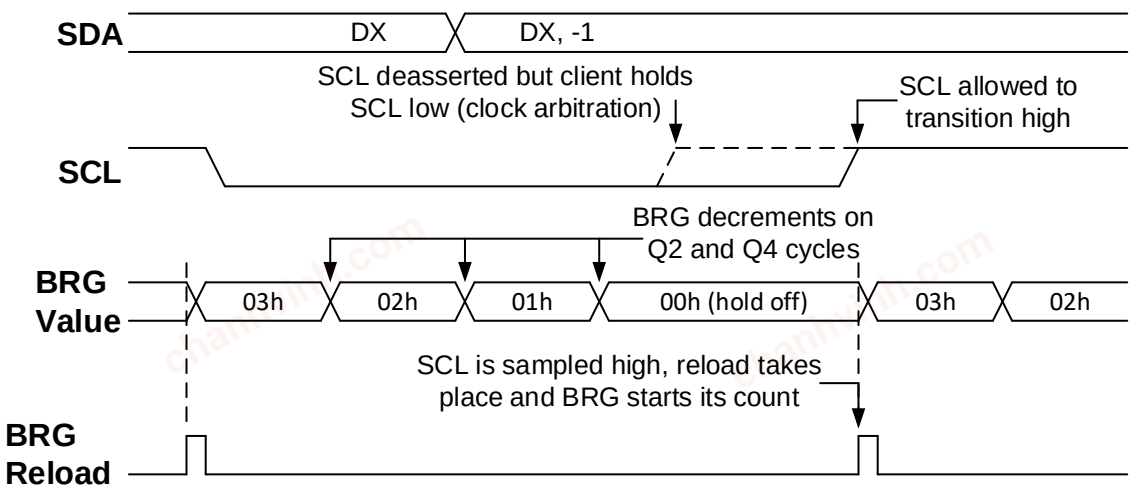
Clock arbitration occurs when the host, during any receive, transmit or Repeated Start/Stop condition, releases the SCL pin (SCL allowed to float high). When the SCL pin is allowed to float high, the Baud Rate Generator (BRG) is suspended from counting until the SCL pin is actually sampled high. When the SCL pin is sampled high, the Baud Rate Generator is reloaded with the contents of SSPxADD 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 as shown in Figure 31-24.

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



**Figure 31-24. Baud Rate Generator Timing with Clock Arbitration**




#### 31.2.4.1.2 WCOL Status Flag

If the user writes the SSPxBUF when a Start, Restart, Stop, Receive or Transmit sequence is in progress, the Write Collision Detect (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.

**Important:** Because queuing of events is not allowed, writing to the lower five bits of SSPxCON2 is disabled until the Start condition is complete.

#### 31.2.4.1.3 I²C Host Mode Start Condition Timing

To initiate a Start condition (see Figure 31-25), the user sets the Start Condition Enable (SEN) bit. If the SDA and SCL pins are sampled high, the Baud Rate Generator is reloaded with the contents of SSPxADD and starts its count. If SCL and SDA are both sampled high when the Baud Rate Generator times out ($T_{BRG}$), 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 Start (S) bit to be set. Following this, the Baud Rate Generator is reloaded with the contents of SSPxADD and resumes its count. When the Baud Rate Generator times out ($T_{BRG}$), the SEN bit will be automatically cleared by hardware; the Baud Rate Generator is suspended, leaving the SDA line held low and the Start condition is complete.

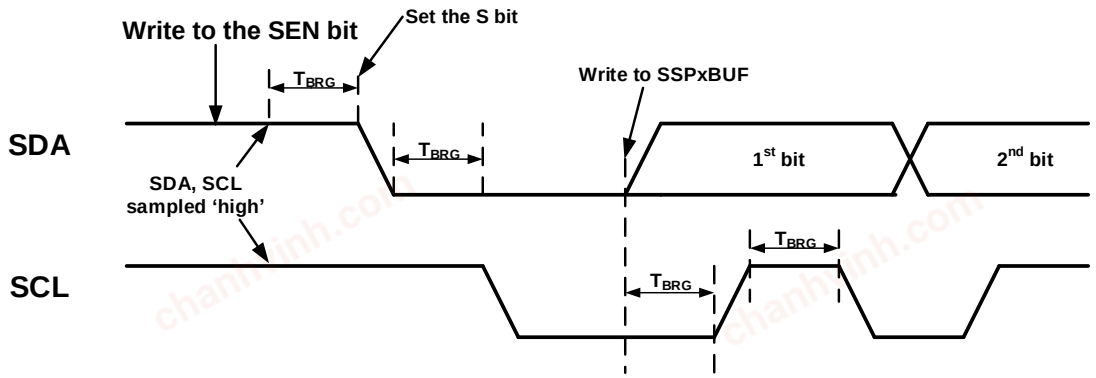
**Important:**

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 (BCLxIF) is set, the Start condition is aborted and the I²C module is reset into its Idle state.
2. The Philips I²C Specification states that a bus collision cannot occur on a Start.

chanhvinh.com



**Figure 31-25. First Start Bit Timing**




#### 31.2.4.1.4 I²C Host Mode Repeated Start Condition Timing

A Repeated Start condition (see Figure 31-26) occurs when the Repeated Start Condition Enable (RSEN) bit is programmed high and the host state machine is Idle. 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 ($T_{BRG}$). 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 remain high for one $T_{BRG}$. Module hardware then pulls the SDA line low (while SCL remains high) for one $T_{BRG}$, and then pulls the SCL line low. Following this, the RSEN bit will be automatically 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 will be set. The SSPxIF bit will not be set until the Baud Rate Generator has timed out.

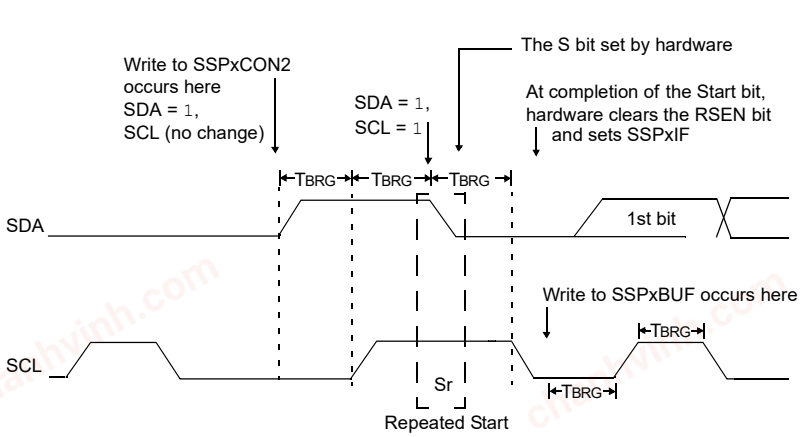
**Important:**

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 host is attempting to transmit a data '1'.

chanhvinh.com



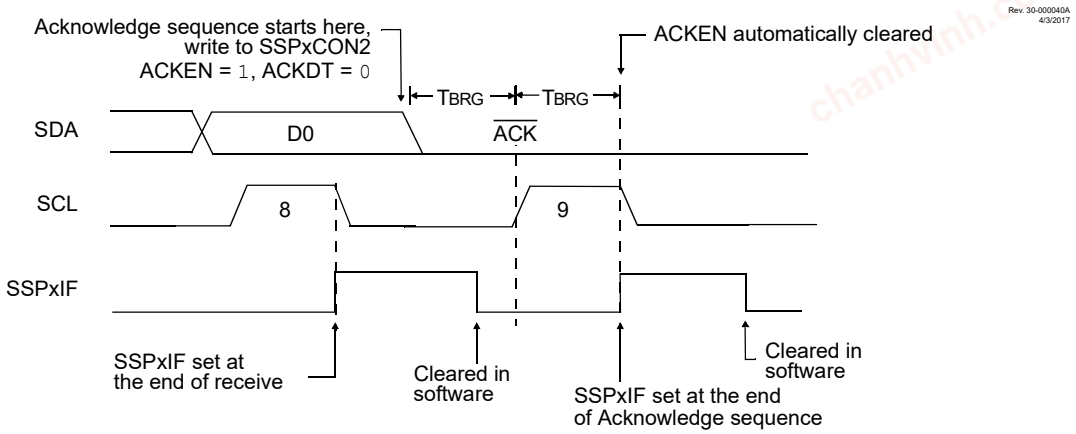
**Figure 31-26. Repeated Start Condition Waveform**




### 31.2.4.1.5 Acknowledge Sequence Timing

An Acknowledge sequence (see Figure 31-27) is enabled by setting the Acknowledge Sequence Enable (ACKEN) bit. When this bit is set, the SCL pin is pulled low and the contents of the Acknowledge Data (ACKDT) bit are presented on the SDA pin. If the user wishes to generate an Acknowledge, then the ACKDT bit must be cleared. If not, the user must set the ACKDT bit before starting an Acknowledge sequence. The Baud Rate Generator then counts for one rollover period ($T_{BRG}$) and the SCL pin is deasserted (pulled high). When the SCL pin is sampled high (clock arbitration), the Baud Rate Generator counts for $T_{BRG}$. 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 31-27. Acknowledge Sequence Waveform**




**Note:** TBRG = one Baud Rate Generator period.

**Acknowledge Write Collision**

If the user writes the SSPxBUF when an Acknowledge sequence is in progress, then the WCOL bit is set and the contents of the buffer are unchanged (the write does not occur).

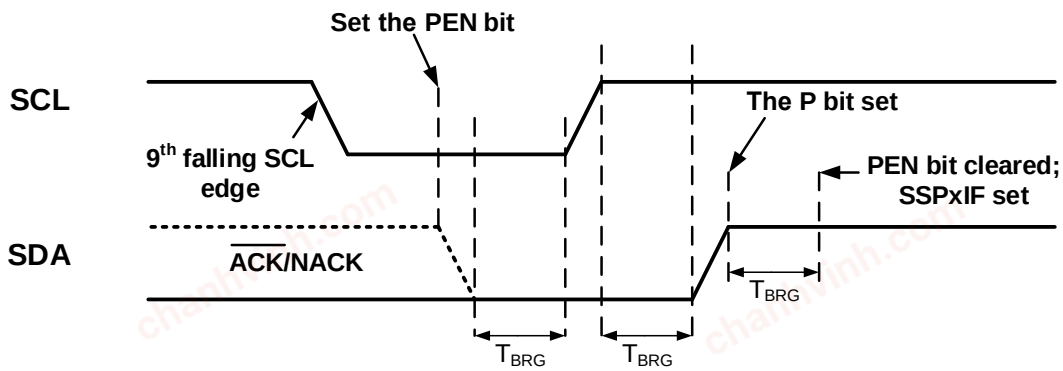
### 31.2.4.1.6 Stop Condition Timing

A Stop condition (see Figure 31-28) is asserted on the SDA pin at the end of a receive/transmit by setting the Stop Condition Enable (PEN) bit. 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 host 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 $T_{BRG}$ (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 is set. One $T_{BRG}$ later, the PEN bit is cleared and the SSPxIF bit is set.

chanhvinh.com



**Figure 31-28. Stop Condition in Receive or Transmit Mode**




**Write Collision on Stop**
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).

#### 31.2.4.1.7 Sleep Operation
While in Sleep mode, the $I^2C$ client 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).

#### 31.2.4.1.8 Effects of a Reset
A Reset disables the MSSP module and terminates the current transfer.

### 31.2.4.2 $I^2C$ Host 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 Status (BF) bit 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 ($T_{BRG}$). Data must be valid before SCL is released high. When the SCL pin is released high, it is held that way for $T_{BRG}$. 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 host releases SDA. This allows the client device being addressed to respond with an $\overline{ACK}$ sequence during the ninth bit time if an address match occurred, or if data was received properly. The status of $\overline{ACK}$ is written into the Acknowledge Status (ACKSTAT) bit on the rising edge of the ninth clock. If the host receives an $\overline{ACK}$, the ACKSTAT bit is cleared. If a NACK is received, ACKSTAT is set. After the ninth clock, the SSPxIF bit is set and the host clock (Baud Rate Generator) is suspended until the next data byte is loaded into the SSPxBUF, leaving SCL low and SDA unchanged (see Figure 31-29).

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/$\overline{W}$ bit are completed. On the falling edge of the eighth clock, the host will release the SDA pin, allowing the client to respond with an $\overline{ACK}$. On the falling edge of the ninth clock, the host will sample the SDA pin to see if the address was recognized by a client. The status of the $\overline{ACK}$ bit is loaded into the ACKSTAT bit.

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.

#### 31.2.4.2.1 BF Status Flag
In Transmit mode, the Buffer Full Status (BF) bit is set when the CPU writes to SSPxBUF, and is cleared when all eight bits are shifted out.

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



#### 31.2.4.2.2 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 Write Collision Detect (WCOL) bit is set and the contents of the buffer are unchanged (the write does not occur).

The WCOL bit must be cleared by software before the next transmission.

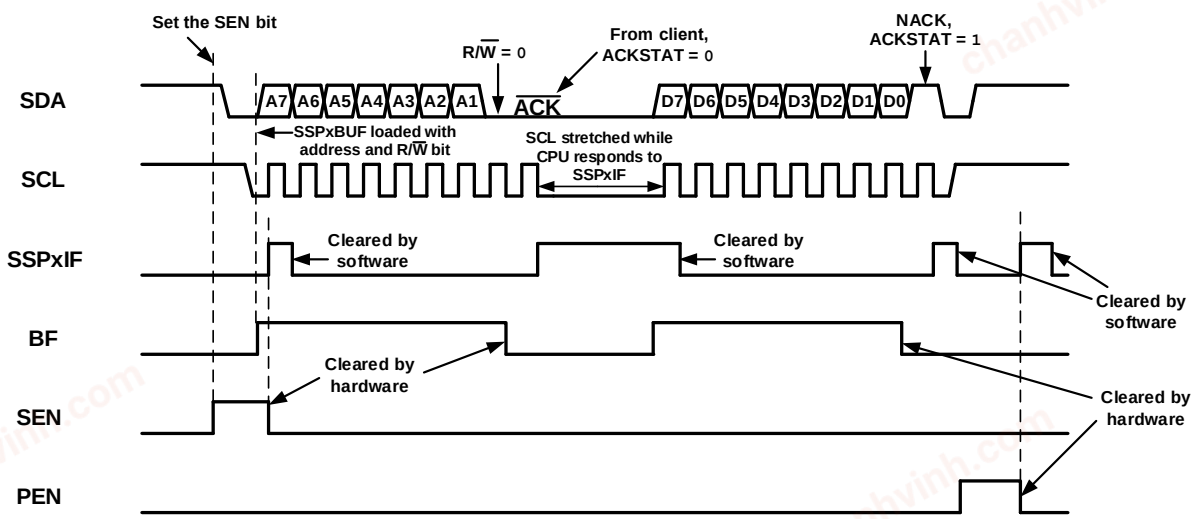
#### 31.2.4.2.3 ACKSTAT Status Flag

In Transmit mode, the Acknowledge Status (ACKSTAT) bit is cleared when the client has sent an Acknowledge ($\overline{ACK}$ = 0), and is set when the client issues a NACK. A client sends an $\overline{ACK}$ when it has recognized its address (including a General Call), or when the client has properly received its data.

#### 31.2.4.2.4 Typical Transmit Sequence

1. The Host generates a Start condition by setting the SEN bit.
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. Software loads the SSPxBUF with the client address and the R/$\overline{W}$ bit. In Host Transmit mode, the R/$\overline{W}$ value is zero.
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 $\overline{ACK}$ value from the client device and writes its into the ACKSTAT bit.
8. The MSSP module generates an interrupt at the end of the ninth clock cycle by setting the SSPxIF bit.
9. Software 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 $\overline{ACK}$ bit from the client device and writes its value into the ACKSTAT bit.
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, respectively. An Interrupt is generated once the Stop/Restart condition is complete.

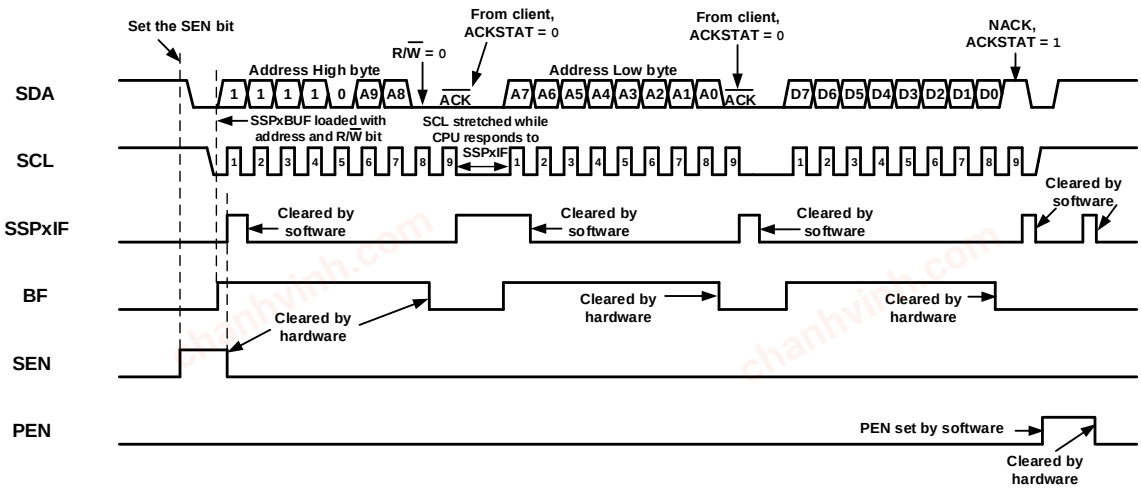
**Figure 31-29. I²C Host Mode Waveform (Transmission, 7-Bit Address)**



chanhvinh.com



**Figure 31-30. I²C Host Mode Waveform (Transmission, 10-Bit Address)**




### 31.2.4.3 I²C Host Mode Reception

Host mode reception (see Figure 31-31) is enabled by setting the Receive Enable (RCEN) bit.

**Important:** 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 SSPxSR. After the falling edge of the eighth clock all the following events occur:

- RCEN is automatically cleared by hardware.
- The contents of the SSPxSR are loaded into the SSPxBUF.
- The BF flag bit is set.
- The SSPxIF flag bit is set.
- The Baud Rate Generator is suspended from counting.
- The SCL pin is held 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 Host can then send an Acknowledge sequence at the end of reception by setting the Acknowledge Sequence Enable (ACKEN) bit.

#### 31.2.4.3.1 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.

#### 31.2.4.3.2 SSPOV Status Flag

In receive operation, the SSPOV bit is set when eight bits are received into SSPxSR while the BF flag bit is already set from a previous reception.

#### 31.2.4.3.3 WCOL Status Flag

If the user writes the SSPxBUF when a receive is already in progress (i.e., SSPxSR is still shifting in a data byte), the WCOL bit is set and the contents of the buffer are unchanged (the write does not occur).

chanhvinh.com



#### 31.2.4.3.4 Typical Receive Sequence

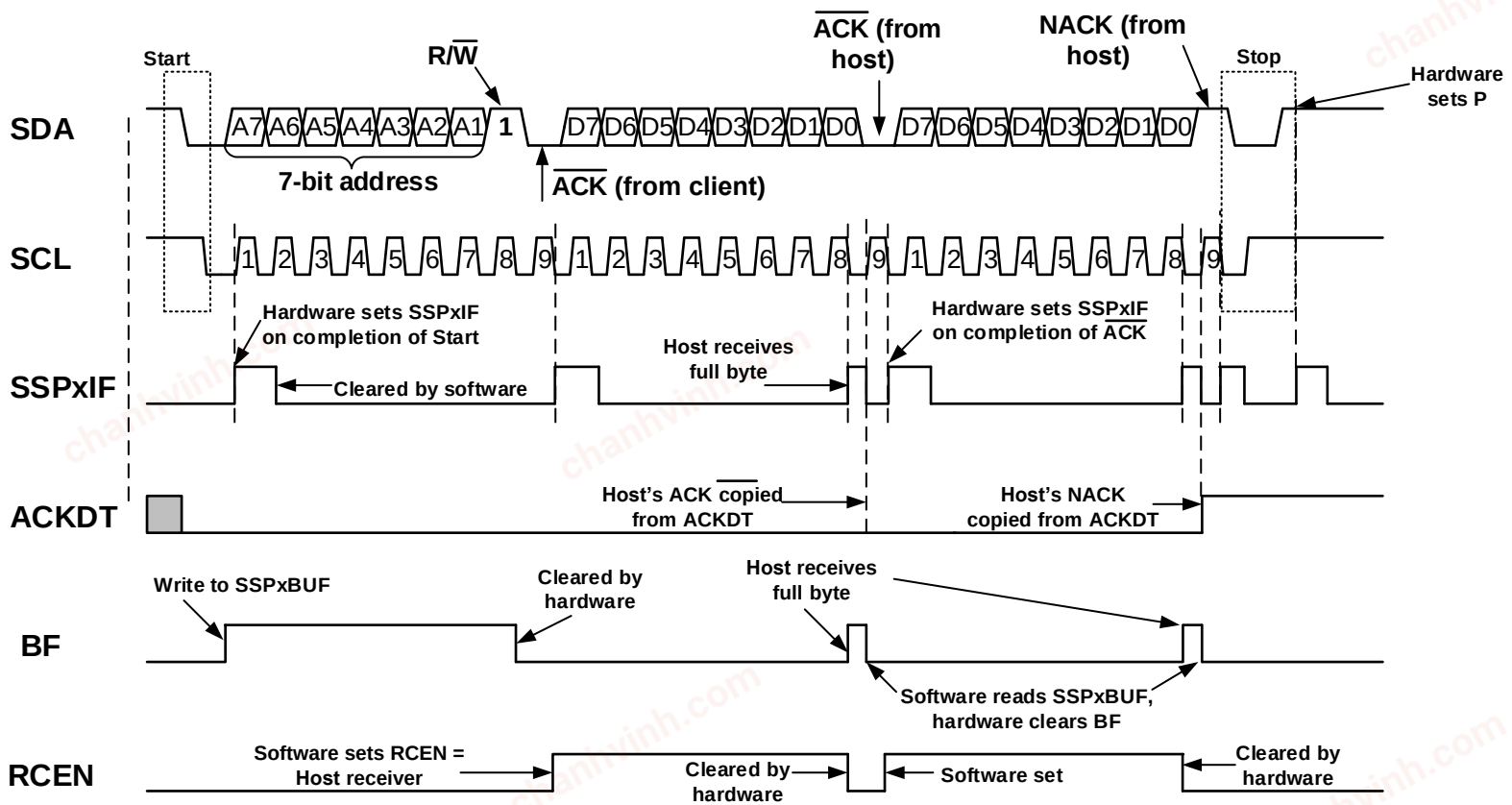
1. The Host generates a Start condition by setting the SEN bit.
2. SSPxIF is set by hardware on completion of the Start.
3. SSPxIF is cleared by software.
4. Software writes SSPxBUF with the client address to transmit and the R/$\overline{W}$ bit set.
5. Address is shifted out the SDA pin until all eight bits are transmitted. Transmission begins as soon as SSPxBUF is written to.
6. The MSSP module shifts in the $\overline{ACK}$ value from the client device and writes it into the ACKSTAT bit.
7. The MSSP module generates an interrupt at the end of the ninth clock cycle by setting the SSPxIF bit.
8. Software sets the RCEN bit and the host clocks in a byte from the client.
9. After the eighth falling edge of SCL, SSPxIF and BF are set.
10. Host clears SSPxIF and reads the received byte from SSPxBUF, which clears BF.
11. Host clears the ACKDT bit and initiates the $\overline{ACK}$ sequence by setting the ACKEN bit.
12. Host's $\overline{ACK}$ is clocked out to the client and SSPxIF is set.
13. User clears SSPxIF.
14. Steps 8-13 are repeated for each received byte from the client.
15. Host sends a NACK or Stop to end communication.

chanhvinh.com

chanhvinh.com



**Figure 31-31. I²C Host Mode Waveform (Reception, 7-Bit Address)**

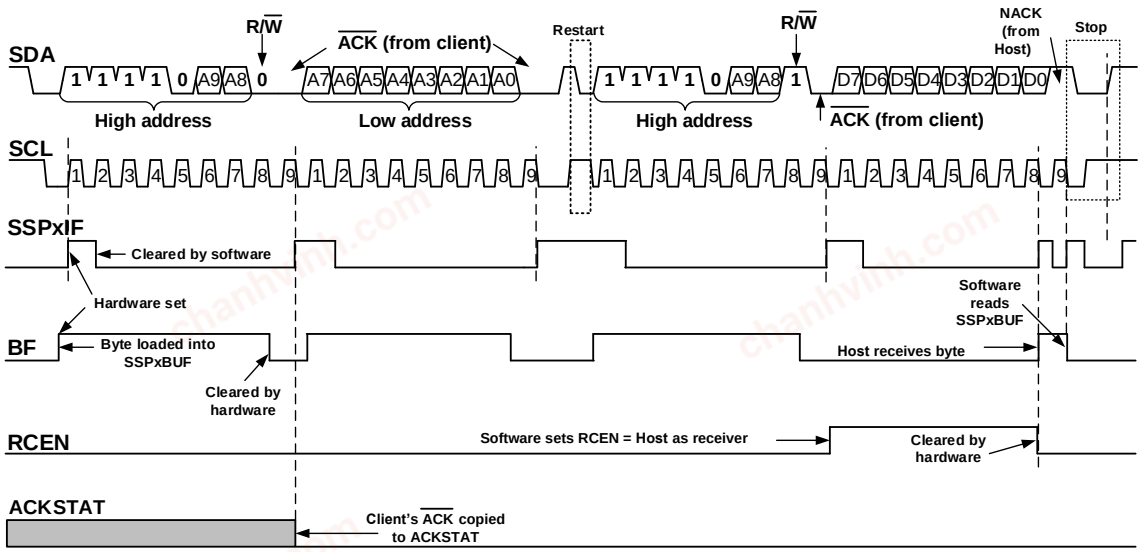



© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



**Figure 31-32. I²C Host Mode Waveform (Reception, 10-Bit Address)**




### 31.2.5 Multi-Host Mode

In Multi-Host 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 is set, or the bus is Idle, with both the S and P bits cleared. When the bus is busy, enabling the MSSP interrupt will generate an interrupt when the Stop condition occurs.

In Multi-Host 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 BCLxIF bit.

The states where arbitration can be lost are:

- Address Transfer
- Data Transfer
- A Start Condition
- A Repeated Start Condition
- An Acknowledge Condition

#### 31.2.5.1 Multi-Host Communication, Bus Collision and Bus Arbitration

Multi-Host mode support is achieved by bus arbitration. When the host outputs address/data bits onto the SDA pin, arbitration takes place when the host outputs a '1' on SDA, by letting SDA float high and another host asserts a '0'. When the SCL pin floats high, data may 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 host will set the Bus Collision Interrupt Flag (BCLxIF) and reset the I²C port to its Idle state (see Figure 31-33).

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 software services the bus collision Interrupt Service Routine, and if the I²C bus is free, software 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 software services the bus collision Interrupt Service Routine, and if the I²C bus is free, software can resume communication by asserting a Start condition.

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com

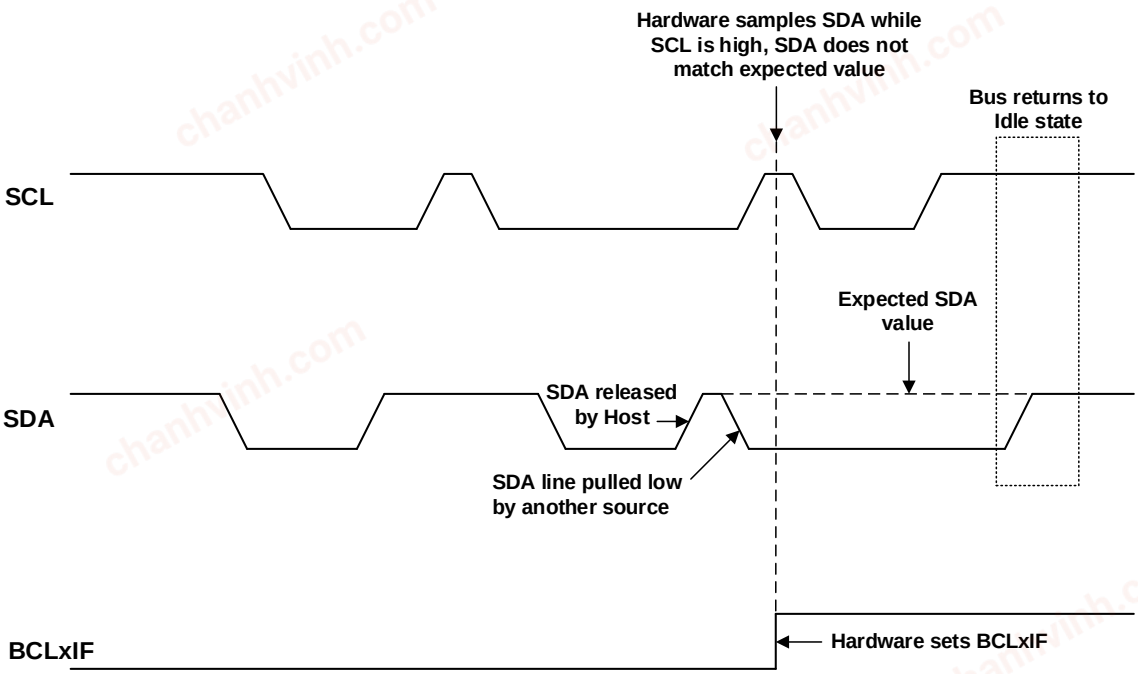


The host will continue to monitor the SDA and SCL pins. If a Stop condition occurs, the SSPxIF bit will be set.

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-Host mode, the interrupt generation on the detection of Start and Stop conditions allows the determination of when the bus is free. Control of the $I^2C$ bus can be taken when the P bit is set, or the bus is Idle and the S and P bits are cleared.

**Figure 31-33. Bus Collision Timing for Transmit and Acknowledge**




#### 31.2.5.1.1 Bus Collision During a Start Condition

During a Start condition, a bus collision occurs if:

1. SDA or SCL are sampled low at the beginning of the Start condition (see Figure 31-34).
2. SCL is sampled low before SDA is asserted low (see Figure 31-35).

During a Start condition, both the SDA and the SCL pins are monitored.

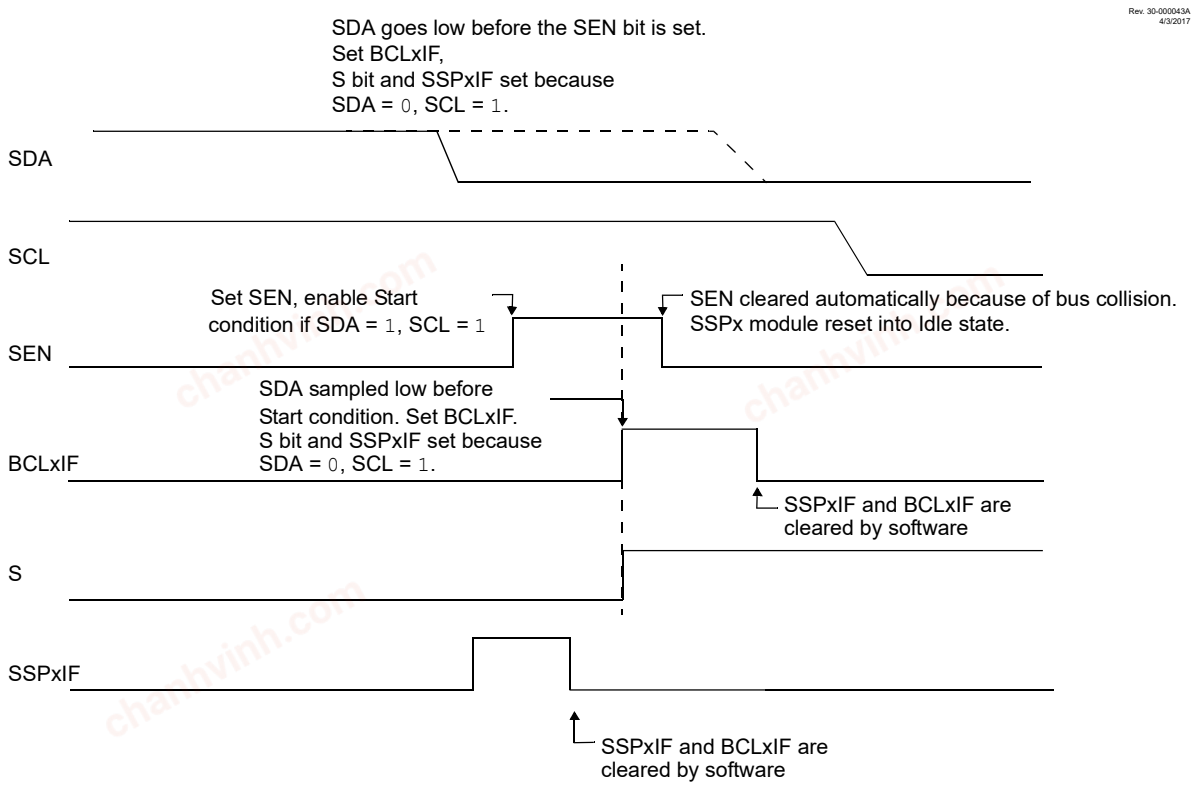
If the SDA pin is already low, or the SCL pin is already low, then all of the following occur:

- the Start condition is aborted,
- the BCLxIF flag is set and
- the MSSP module is reset to its Idle state (see Figure 31-34).

chanhvinh.com

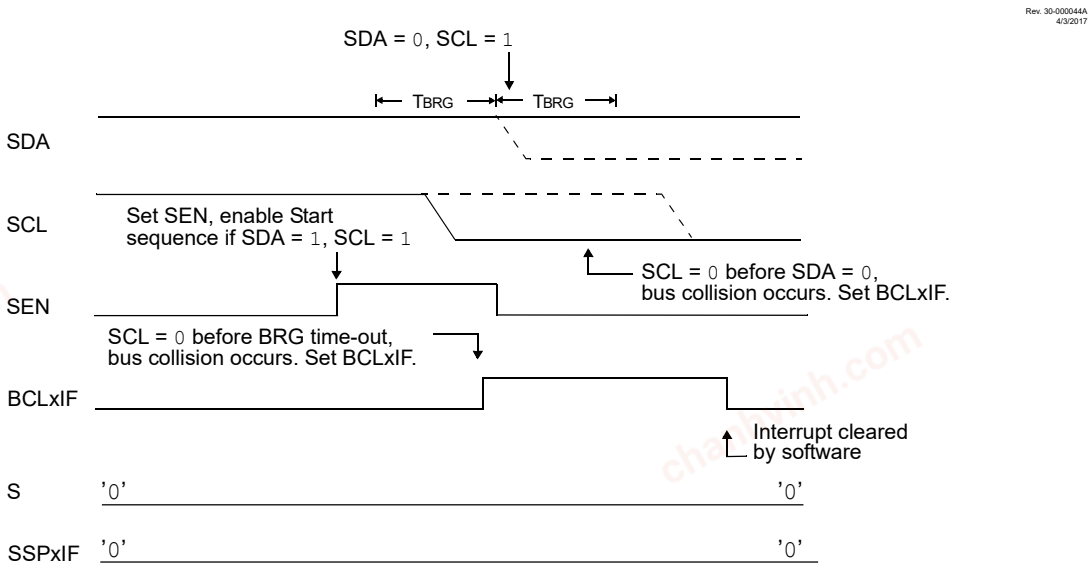


**Figure 31-34. Bus Collision During Start Condition (SDA Only)**




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 host is attempting to drive a data '1' during the Start condition.

**Figure 31-35. Bus Collision During Start Condition (SCL = 0)**



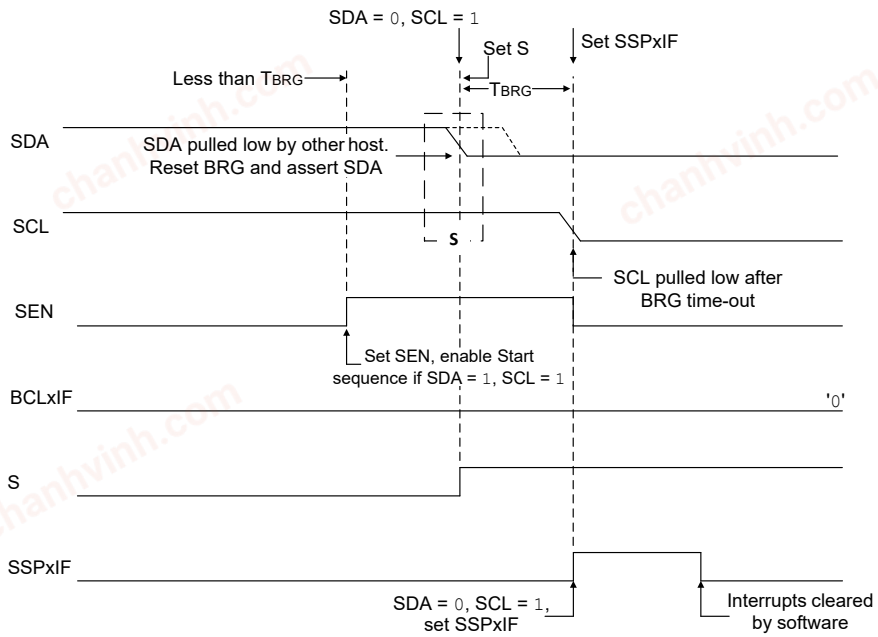

If the SDA pin is sampled low during this count, the BRG is reset and the SDA line is asserted early (see Figure 31-36). If, however, a '1' is sampled on the SDA pin, the SDA pin is asserted low at the end of the BRG count. The

chanhvinh.com



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.

**Figure 31-36. BRG Reset Due to SDA Arbitration During Start Condition**




**Important:** The reason that a bus collision is not a factor during a Start condition is that no two bus hosts can assert a Start condition at the exact same time. Therefore, one host will always assert SDA before the other. This condition does not cause a bus collision because the two hosts 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.

#### 31.2.5.1.2 Bus Collision During a Repeated Start Condition

During a Repeated Start condition, a bus collision occurs if:

1. A low level is sampled on SDA when SCL goes from low level to high level (see Figure 31-37).
2. SCL goes low before SDA is asserted low, indicating that another host is attempting to transmit a data '1' (see Figure 31-38).

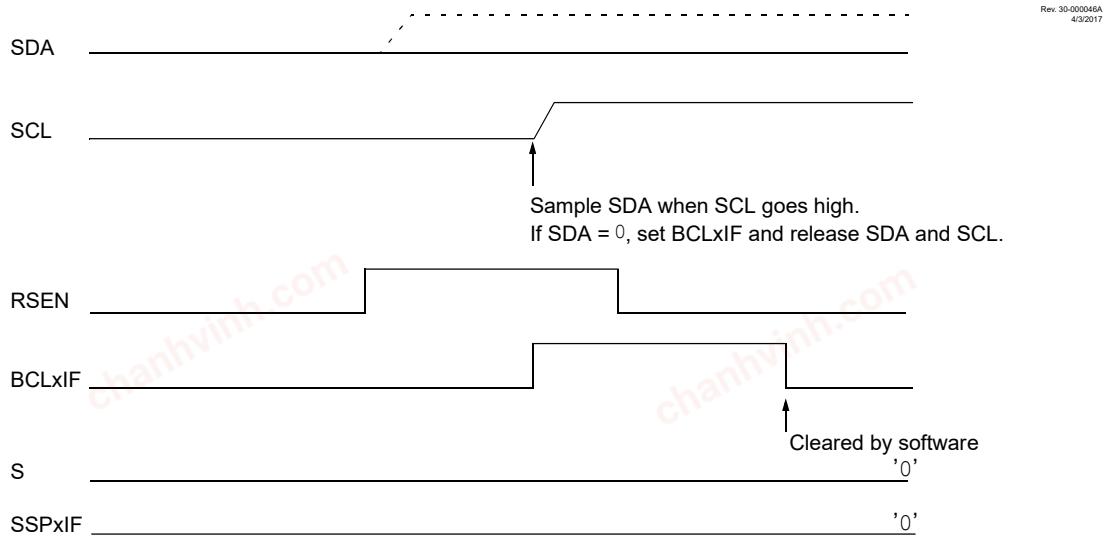
When the user releases SDA and the pin is allowed to float high, the BRG is loaded with SSPxADD and counts down to zero. The SCL pin is then deasserted and when sampled high, the SDA pin is sampled.

If SDA is low, a bus collision has occurred (i.e., another host is attempting to transmit a data '0', see Figure 31-37). 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 hosts can assert SDA at exactly the same time.

chanhvinh.com



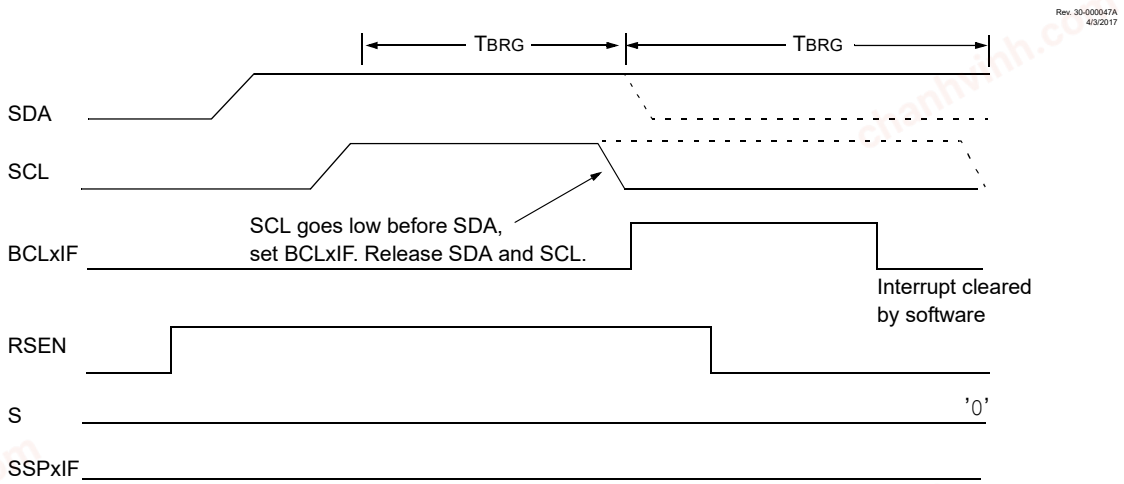
**Figure 31-37. Bus Collision During a Repeated Start Condition (Case 1)**




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 host is attempting to transmit a data '1' during the Repeated Start condition (see Figure 31-38).

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 31-38. Bus Collision During Repeated Start Condition (Case 2)**




#### 31.2.5.1.3 Bus Collision During a Stop Condition

Bus collision occurs during a Stop condition if:

1. After the SDA pin has been deasserted and allowed to float high, SDA is sampled low after the BRG has timed out (see Figure 31-39).
2. After the SCL pin is deasserted, SCL is sampled low before SDA goes high (see Figure 31-40).

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 host attempting to drive a data '0' (see Figure 31-39). If the SCL pin is sampled low before SDA is allowed to float high, a bus collision occurs. This is another case of another host attempting to drive a data '0' (Figure 31-40).

chanhvinh.com



**Figure 31-39. Bus Collision During a Stop Condition (Case 1)**

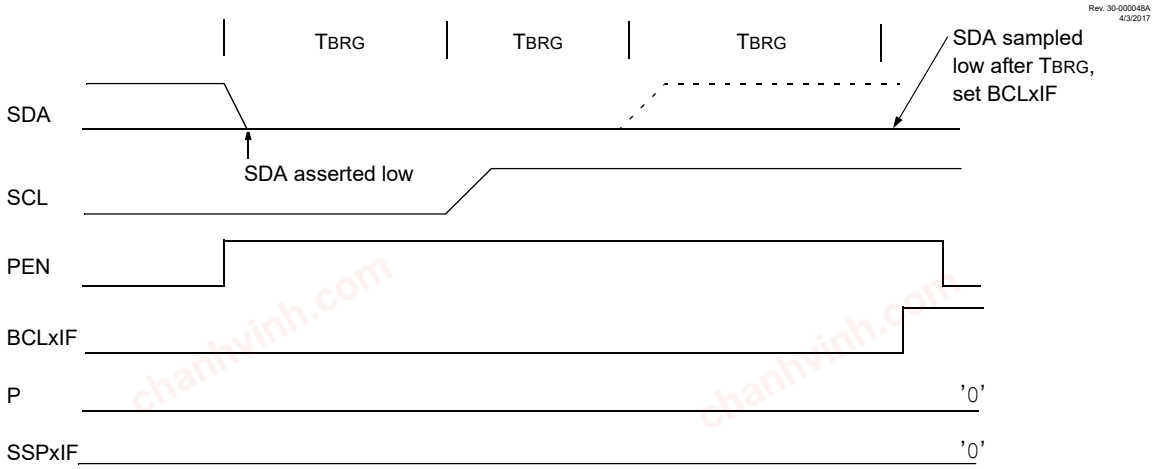
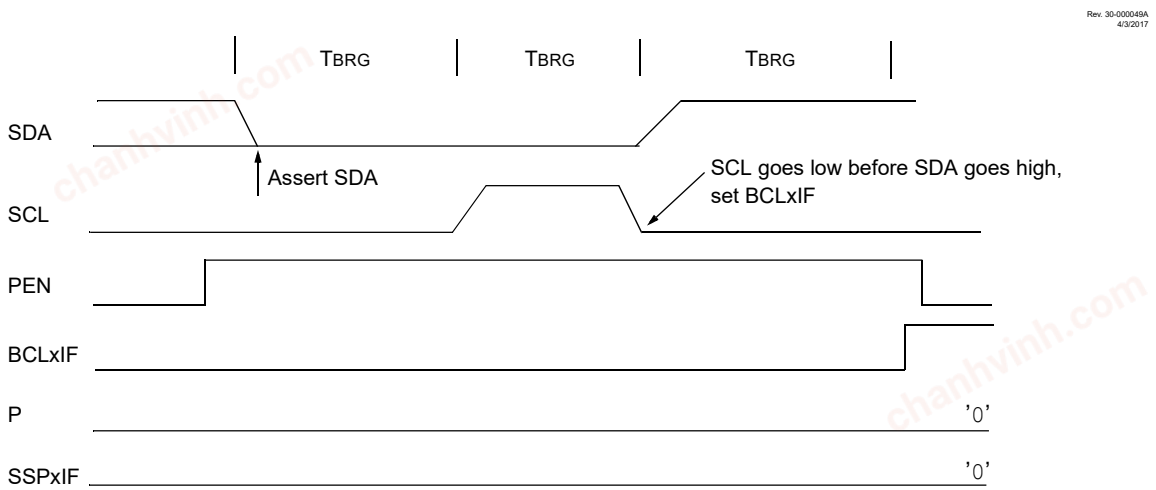



**Figure 31-40. Bus Collision During a Stop Condition (Case 2)**




## 31.3 Baud Rate Generator

The MSSP module has a Baud Rate Generator (BRG) available for clock generation in both $I^2C$ and SPI Host modes. The BRG reload value is placed in the SSPxADD register. When a write occurs to SSPxBUF, the BRG will automatically begin counting down. Example 31-1 shows how the value for SSPxADD is calculated.

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 Reload signal, shown in Figure 31-41, triggers the value from SSPxADD to be loaded into the BRG counter. This occurs twice for each oscillation of the module clock line.

Table 31-2 illustrates clock rates based on instruction cycles and the BRG value loaded into SSPxADD.

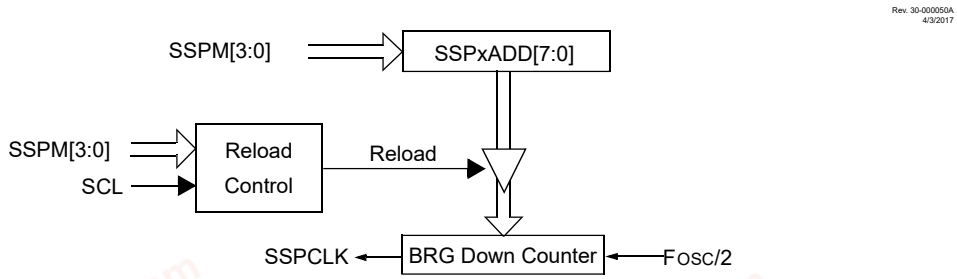
**Example 31-1. MSSP Baud Rate Generator Frequency Equation**

$$F_{CLOCK} = \frac{F_{OSC}}{4 \times (SSPxADD + 1)}$$

chanhvinh.com



**Figure 31-41. Baud Rate Generator Block Diagram**




**Important:** Values of `0x00`, `0x01` and `0x02` are not valid for SSPxADD when used as a Baud Rate Generator for $I^2C$. This is an implementation limitation.

**Table 31-2. MSSP Clock Rate w/BRG**

| $F_{OSC}$ | $F_{CY}$ | 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 the **"Electrical Specifications"** chapter, Internal Oscillator Parameters, to ensure the system is designed to support all requirements.

## 31.4 Register Definitions: MSSP Control

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



### 31.4.1 SSPxBUF

**Name:** SSPxBUF
**Offset:** 0x078C,0x0796

MSSP Data Buffer Register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | BUF[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | x | x | x | x | x | x | x | x |

**Bits 7:0 – BUF[7:0]** MSSP Input and Output Data Buffer bits

chanhvinh.com

chanhvinh.com



### 31.4.2 SSPxADD

**Name:** SSPxADD
**Offset:** 0x078D,0x0797

MSSP Baud Rate Divider and Address Register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ADD[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 7:0 – ADD[7:0]**

- SPI and $I^2C$ Host: Baud rate divider
- $I^2C$ Client: Address bits

| Value | Mode | Description |
|---|---|---|
| 11111111 – 00000011 | SPI and $I^2C$ Host | Baud rate divider. SCK/SCL pin clock period = ((n + 1) * 4)/$F_{OSC}$. Values less than 3 are not valid. |
| xxxxx11x – xxxxx00x | $I^2C$ 10-bit Client MS Address | Bits [7:3] and Bit 0 are not used and are don't care. Bits [2:1] are bits [9:8] of the 10-bit Client Most Significant Address. |
| 11111111 – 00000000 | $I^2C$ 10-bit Client LS Address | Bits [7:0] of 10-bit Client Least Significant Address |
| 1111111x – 0000000x | $I^2C$ 7-bit Client | Bit 0 is not used and is don't care. Bits [7:1] are the 7-bit Client Address. |

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



### 31.4.3 SSPxMSK

**Name:** SSPxMSK
**Offset:** 0x078E,0x0798

MSSP Address Mask Register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | MSK[6:0] | | | | | | | MSK0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

**Bits 7:1 – MSK[6:0]** Mask bits

| Value | Mode | Description |
|---|---|---|
| 1 | $I^2C$ Client | The received address bit n is compared to SSPxADD bit n to detect $I^2C$ address match |
| 0 | $I^2C$ Client | The received address bit n is not used to detect $I^2C$ address match |

**Bit 0 – MSK0** Mask bit for $I^2C$ 10-bit Client mode

| Value | Mode | Description |
|---|---|---|
| x | SPI or $I^2C$ 7-bit | This bit is not used |
| 1 | $I^2C$ 10-bit Client | The received address bit 0 is compared to SSPxADD bit 0 to detect $I^2C$ address match |
| 0 | $I^2C$ 10-bit Client | The received address bit 0 is not used to detect $I^2C$ address match |

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



### 31.4.4 SSPxSTAT

**Name:** SSPxSTAT
**Offset:** 0x078F,0x0799

MSSP Status Register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SMP | CKE | D/$\overline{A}$ | P | S | R/$\overline{W}$ | UA | BF |
| Access | R/W | R/W | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 7 – SMP** Slew Rate Control bit

| Value | Mode | Description |
|---|---|---|
| 1 | SPI Host | Input data is sampled at the end of data output time |
| 0 | SPI Host | Input data is sampled at the middle of data output time |
| 0 | SPI Client | Bit must be cleared in SPI Client mode |
| 1 | $I^2C$ | Slew rate control is disabled for Standard Speed mode (100 kHz and 1 MHz) |
| 0 | $I^2C$ | Slew rate control is enabled for High Speed mode (400 kHz) |

**Bit 6 – CKE** SPI: Clock Select bit(4); $I^2C$: SMBus Select bit

| Value | Mode | Description |
|---|---|---|
| 1 | SPI | Transmit occurs on the transition from Active to Idle clock state |
| 0 | SPI | Transmit occurs on the transition from Idle to Active clock state |
| 1 | $I^2C$ | Enables SMBus-specific inputs |
| 0 | $I^2C$ | Disables SMBus-specific inputs |

**Bit 5 – D/$\overline{A}$** Data/$\overline{Address}$ bit

| Value | Mode | Description |
|---|---|---|
| x | SPI or $I^2C$ Host | This bit is not used |
| 1 | $I^2C$ Client | Indicates that the last byte received or transmitted was data |
| 0 | $I^2C$ Client | Indicates that the last byte received or transmitted was address |

**Bit 4 – P** Stop bit(1)

| Value | Mode | Description |
|---|---|---|
| x | SPI | This bit is not used |
| 1 | $I^2C$ | Stop bit was detected last |
| 0 | $I^2C$ | Stop bit was not detected last |

**Bit 3 – S** Start bit(1)

| Value | Mode | Description |
|---|---|---|
| x | SPI | This bit is not used |
| 1 | $I^2C$ | Start bit was detected last |
| 0 | $I^2C$ | Start bit was not detected last |

**Bit 2 – R/$\overline{W}$** Read/$\overline{Write}$ Information bit(2,3)

| Value | Mode | Description |
|---|---|---|
| x | SPI | This bit is not used |
| 1 | $I^2C$ Client | Read |
| 0 | $I^2C$ Client | Write |
| 1 | $I^2C$ Host | Transmit is in progress |
| 0 | $I^2C$ Host | Transmit is not in progress |

**Bit 1 – UA** Update Address bit (10-bit $I^2C$ Client mode only)

| Value | Mode | Description |
|---|---|---|
| x | All other modes | This bit is not used |

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



| Value | Mode | Description |
|---|---|---|
| 1 | $I^2C$ 10-bit Client | Indicates that the user needs to update the address in the SSPxADD register |
| 0 | $I^2C$ 10-bit Client | Address does not need to be updated |

**Bit 0 – BF** Buffer Full Status bit(5)

| Value | Mode | Description |
|---|---|---|
| 1 | $I^2C$ Transmit | Transmit in progress, SSPxBUF is full |
| 0 | $I^2C$ Transmit | Transmit complete; SSPxBUF is empty |
| 1 | SPI and $I^2C$ Receive | Receive complete, SSPxBUF is full |
| 0 | SPI and $I^2C$ Receive | Receive not complete, SSPxBUF is empty |

**Notes:**

1. This bit is cleared on Reset and when SSPEN is cleared.
2. In $I^2C$ Client mode, this bit holds the R/$\overline{W}$ 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 $\overline{ACK}$ bit.
3. ORing this bit with SEN, RSEN, PEN, RCEN or ACKEN will indicate if the MSSP is in Active mode.
4. Polarity of clock state is set by the CKP bit.
5. $I^2C$ receive status does not include $\overline{ACK}$ and Stop bits.

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



### 31.4.5 SSPxCON1

**Name:** SSPxCON1
**Offset:** 0x0790,0x079A

MSSP Control Register 1

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | WCOL | SSPOV | SSPEN | CKP | SSPM[3:0] | | | |
| Access | R/W/HS | R/W/HS | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 7 – WCOL** Write Collision Detect bit

| Value | Mode | Description |
|---|---|---|
| x | Host or Client receive | This bit is not used |
| 1 | SPI or I²C Host or Client transmit | The SSPxBUF register is written while it is still transmitting the previous word (must be cleared in software) |
| 0 | SPI or I²C Host or Client transmit | No collision |

**Bit 6 – SSPOV** Receive Overflow Indicator bit(1)

| Value | Mode | Description |
|---|---|---|
| x | SPI Host or I²C Host transmit | This bit is not used |
| 1 | SPI Client | A byte is received while the SSPxBUF register is still holding the previous byte. Data contained in the shift register will be discarded. The user must read SSPxBUF, even if only transmitting data, to avoid setting overflow (must be cleared in software). |
| 1 | I²C Receive | A byte is received while the SSPxBUF register is still holding the previous byte (must be cleared in software) |
| 0 | SPI Client or I²C Receive | No overflow |

**Bit 5 – SSPEN** Host Synchronous Serial Port Enable bit.(2)

| Value | Description |
|---|---|
| 1 | Enables the serial port |
| 0 | Disables serial port and configures these pins as I/O PORT pins |

**Bit 4 – CKP** SCK Release Control bit

| Value | Mode | Description |
|---|---|---|
| x | I²C Host | This bit is not used |
| 1 | SPI | Idle state for the clock is a high level |
| 0 | SPI | Idle state for the clock is a low level |
| 1 | I²C Client | Releases clock |
| 0 | I²C Client | Holds clock low (clock stretch), used to ensure data setup time |

**Bits 3:0 – SSPM[3:0]** Host Synchronous Serial Port Mode Select bits(4)

| Value | Description |
|---|---|
| 1111 | I²C Client mode: 10-bit address with Start and Stop bit interrupts enabled |
| 1110 | I²C Client mode: 7-bit address with Start and Stop bit interrupts enabled |
| 1101 | Reserved - do not use |
| 1100 | Reserved - do not use |
| 1011 | I²C Firmware Controlled Host mode (client Idle) |
| 1010 | SPI Host mode: Clock = $F_{OSC}/(4*(SSPxADD+1))$. SSPxADD must be greater than 0.(3) |
| 1001 | Reserved - do not use |
| 1000 | I²C Host mode: Clock = $F_{OSC}/(4 * (SSPxADD + 1))$ |
| 0111 | I²C Client mode: 10-bit address |
| 0110 | I²C Client mode: 7-bit address |

chanhvinh.com



| Value | Description |
|---|---|
| 0101 | SPI Client mode: Clock = SCKx pin. $\overline{SS}$x pin control is disabled |
| 0100 | SPI Client mode: Clock = SCKx pin. $\overline{SS}$x pin control is enabled |
| 0011 | SPI Host mode: Clock = TMR2 output/2 |
| 0010 | SPI Host mode: Clock = $F_{OSC}$/64 |
| 0001 | SPI Host mode: Clock = $F_{OSC}$/16 |
| 0000 | SPI Host mode: Clock = $F_{OSC}$/4 |

**Notes:**

1. In Host mode, the overflow bit is not set since each new reception (and transmission) is initiated by writing to the SSPxBUF register.
2. When enabled, these pins must be properly configured as inputs or outputs.
3. SSPxADD = 0 is not supported.
4. Bit combinations not specifically listed here are either reserved or implemented in $I^2C$ mode only.

chanhvinh.com

chanhvinh.com



### 31.4.6 SSPxCON2

**Name:** SSPxCON2
**Offset:** 0x0791,0x079B

MSSP Control Register 2

Control Register for I²C Operation Only

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | GCEN | ACKSTAT | ACKDT | ACKEN | RCEN | PEN | RSEN | SEN |
| Access | R/W | R//HC/HS | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 7 – GCEN** General Call Enable bit (Client mode only)

| Value | Mode | Description |
|---|---|---|
| x | Host mode | Don't care |
| 1 | Client mode | General Call is enabled |
| 0 | Client mode | General Call is not enabled |

**Bit 6 – ACKSTAT** Acknowledge Status bit (Host Transmit mode only)

| Value | Description |
|---|---|
| 1 | Acknowledge was not received from client |
| 0 | Acknowledge was received from client |

**Bit 5 – ACKDT** Acknowledge Data bit (Host Receive mode only)(1)

| Value | Description |
|---|---|
| 1 | Not Acknowledge |
| 0 | Acknowledge |

**Bit 4 – ACKEN** Acknowledge Sequence Enable bit(2)

| Value | Description |
|---|---|
| 1 | Initiates Acknowledge sequence on SDAx and SCLx pins and transmits ACKDT data bit; automatically cleared by hardware |
| 0 | Acknowledge sequence is Idle |

**Bit 3 – RCEN** Receive Enable bit (Host Receive mode only)(2)

| Value | Description |
|---|---|
| 1 | Enables Receive mode for I²C |
| 0 | Receive is Idle |

**Bit 2 – PEN** Stop Condition Enable bit (Host mode only)(2)

| Value | Description |
|---|---|
| 1 | Initiates Stop condition on SDAx and SCLx pins; automatically cleared by hardware |
| 0 | Stop condition is Idle |

**Bit 1 – RSEN** Repeated Start Condition Enable bit (Host mode only)(2)

| Value | Description |
|---|---|
| 1 | Initiates Repeated Start condition on SDAx and SCLx pins; automatically cleared by hardware |
| 0 | Repeated Start condition is Idle |

**Bit 0 – SEN** Start Condition Enable bit(2)

| Value | Mode | Description |
|---|---|---|
| 1 | Host | Initiates Start condition on SDAx and SCLx pins; automatically cleared by hardware |
| 0 | Host | Start condition is Idle |
| 1 | Client | Clock stretching is enabled |
| 0 | Client | Clock stretching is disabled |

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



**Notes:**

1. The value that will be transmitted when the user initiates an Acknowledge sequence at the end of a receive.
2. If the $I^2C$ module is active, these bits may not be set (no spooling) and the SSPxBUF may not be written (or writes to the SSPxBUF are disabled).

chanhvinh.com

chanhvinh.com



### 31.4.7 SSPxCON3

**Name:** SSPxCON3
**Offset:** 0x0792,0x079C

MSSP Control Register 3

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ACKTIM | PCIE | SCIE | BOEN | SDAHT | SBCDE | AHEN | DHEN |
| Access | R/HS/HC | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 7 – ACKTIM** Acknowledge Time Status bit

| Value | Mode | Description |
|---|---|---|
| x | SPI or I²C Host | This bit is not used |
| 1 | I²C Client and AHEN = 1 or DHEN = 1 | Eighth falling edge of SCL has occurred and the ACK/NACK state is Active |
| 0 | I²C Client | ACK/NACK state is not Active. Transitions low on ninth rising edge of SCL. |

**Bit 6 – PCIE** Stop Condition Interrupt Enable bit

| Value | Mode | Description |
|---|---|---|
| x | SPI or SSPM = 1111 or 1110 | This bit is not used |
| 1 | SSPM ≠ 1111 and SSPM ≠ 1110 | Enable interrupt on detection of Stop condition |
| 0 | SSPM ≠ 1111 and SSPM ≠ 1110 | Stop detection interrupts are disabled |

**Bit 5 – SCIE** Start Condition Interrupt Enable bit

| Value | Mode | Description |
|---|---|---|
| x | SPI or SSPM = 1111 or 1110 | This bit is not used |
| 1 | SSPM ≠ 1111 and SSPM ≠ 1110 | Enable interrupt on detection of Start condition |
| 0 | SSPM ≠ 1111 and SSPM ≠ 1110 | Start detection interrupts are disabled |

**Bit 4 – BOEN** Buffer Overwrite Enable bit(1)

| Value | Mode | Description |
|---|---|---|
| 1 | SPI | SSPxBUF is updated every time a new data byte is available, ignoring the BF bit |
| 0 | SPI | If a new byte is receive with BF set then SSPOV is set and SSPxBUF is not updated |
| 1 | I²C | SSPxBUF is updated every time a new data byte is available, ignoring the SSPOV effect on updating the buffer |
| 0 | I²C | SSPxBUF is only updated when SSPOV is clear |

**Bit 3 – SDAHT** SDA Hold Time Selection bit

| Value | Mode | Description |
|---|---|---|
| x | SPI | Not used in SPI mode |
| 1 | I²C | Minimum of 300 ns hold time on SDA after the falling edge of SCL |
| 0 | I²C | Minimum of 100 ns hold time on SDA after the falling edge of SCL |

**Bit 2 – SBCDE** Client Mode Bus Collision Detect Enable bit
Unused in Host mode.

| Value | Mode | Description |
|---|---|---|
| x | SPI or I²C Host | This bit is not used |
| 1 | I²C Client | Bus Collision detection is enabled |
| 0 | I²C Client | Bus Collision detection is not enabled |

**Bit 1 – AHEN** Address Hold Enable bit

| Value | Mode | Description |
|---|---|---|
| x | SPI or I²C Host | This bit is not used |

chanhvinh.com



| Value | Mode | Description |
|---|---|---|
| 1 | I²C Client | Address hold is enabled. As a result, CKP is cleared after the eighth falling SCL edge of an address byte reception. Software must set the CKP bit to resume operation. |
| 0 | I²C Client | Address hold is not enabled |

**Bit 0 – DHEN** Data Hold Enable bit

| Value | Mode | Description |
|---|---|---|
| x | SPI or I²C Host | This bit is not used |
| 1 | I²C Client | Data hold is enabled. As a result, CKP is cleared after the eighth falling SCL edge of a data byte reception. Software must set the CKP bit to resume operation. |
| 0 | I²C Client | Data hold is not enabled |

**Note:**

1. For daisy-chained SPI operation; allows the user to ignore all except the last received byte. SSPOV is still set when a new byte is received and BF = 1, but hardware continues to write the most recent byte to SSPxBUF.

chanhvinh.com

chanhvinh.com



## 31.5 Register Summary - MSSP Control

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 ... 0x078B | Reserved | | | | | | | | | |
| 0x078C | SSP1BUF | 7:0 | BUF[7:0] | | | | | | | |
| 0x078D | SSP1ADD | 7:0 | ADD[7:0] | | | | | | | |
| 0x078E | SSP1MSK | 7:0 | MSK[6:0] | | | | | | | MSK0 |
| 0x078F | SSP1STAT | 7:0 | SMP | CKE | D/$\overline{A}$ | P | S | R/$\overline{W}$ | UA | BF |
| 0x0790 | SSP1CON1 | 7:0 | WCOL | SSPOV | SSPEN | CKP | SSPM[3:0] | | | |
| 0x0791 | SSP1CON2 | 7:0 | GCEN | ACKSTAT | ACKDT | ACKEN | RCEN | PEN | RSEN | SEN |
| 0x0792 | SSP1CON3 | 7:0 | ACKTIM | PCIE | SCIE | BOEN | SDAHT | SBCDE | AHEN | DHEN |
| 0x0793 ... 0x0795 | Reserved | | | | | | | | | |
| 0x0796 | SSP2BUF | 7:0 | BUF[7:0] | | | | | | | |
| 0x0797 | SSP2ADD | 7:0 | ADD[7:0] | | | | | | | |
| 0x0798 | SSP2MSK | 7:0 | MSK[6:0] | | | | | | | MSK0 |
| 0x0799 | SSP2STAT | 7:0 | SMP | CKE | D/$\overline{A}$ | P | S | R/$\overline{W}$ | UA | BF |
| 0x079A | SSP2CON1 | 7:0 | WCOL | SSPOV | SSPEN | CKP | SSPM[3:0] | | | |
| 0x079B | SSP2CON2 | 7:0 | GCEN | ACKSTAT | ACKDT | ACKEN | RCEN | PEN | RSEN | SEN |
| 0x079C | SSP2CON3 | 7:0 | ACKTIM | PCIE | SCIE | BOEN | SDAHT | SBCDE | AHEN | DHEN |

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



# 32. EUSART - Enhanced Universal Synchronous Asynchronous Receiver Transmitter

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 host 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 host
- Half-duplex synchronous client
- 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 32-1 and Figure 32-2.

The operation of the EUSART module consists of six registers:

- Transmit Status and Control (TXxSTA)
- Receive Status and Control (RCxSTA)
- Baud Rate Control (BAUDxCON)
- Baud Rate Value (SPxBRG)
- Receive Data Register (RCxREG)
- Transmit Data Register (TXxREG)

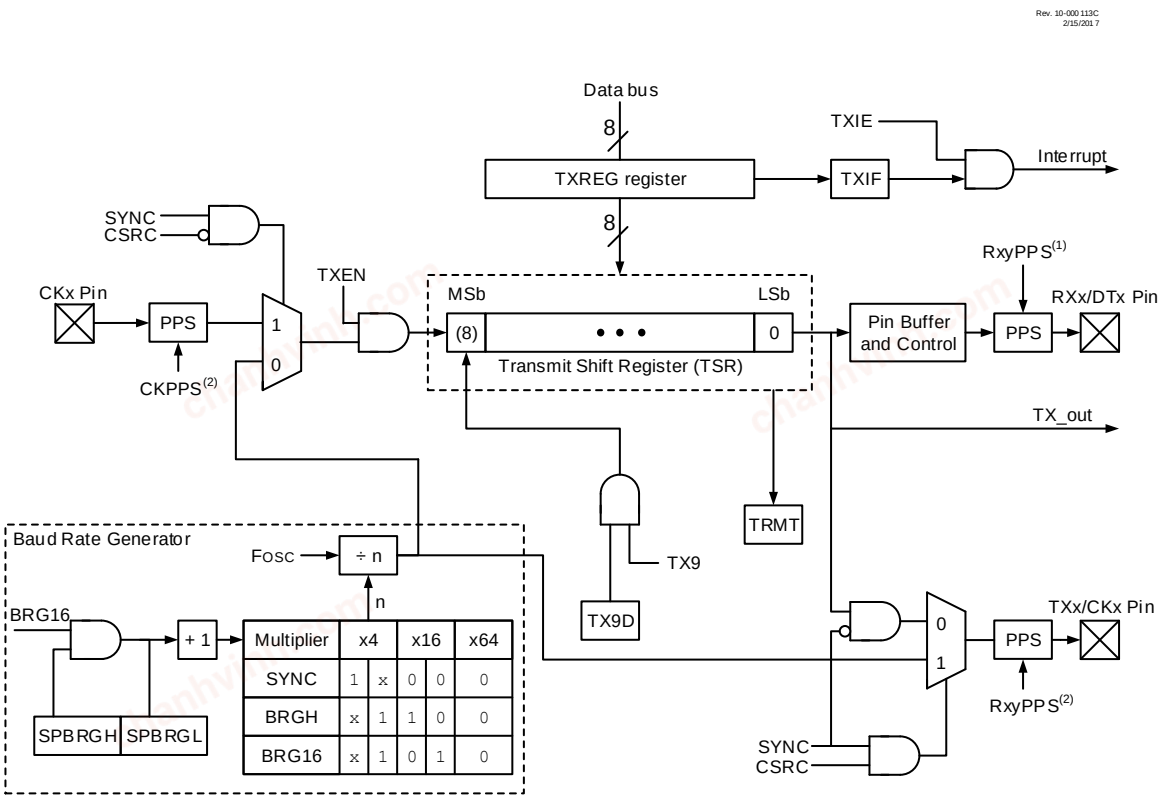
The RXx/DTx and TXx/CKx input pins are selected with the RXxPPS and TXxPPS registers, respectively. TXx, CKx, and DTx output pins are selected with each pin's RxyPPS register. Since the RX input is coupled with the DT output in Synchronous mode, it is the user's responsibility to select the same pin for both of these functions when operating in Synchronous mode. The EUSART control logic will control the data direction drivers automatically.

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



**Figure 32-1. EUSART Transmit Block Diagram**

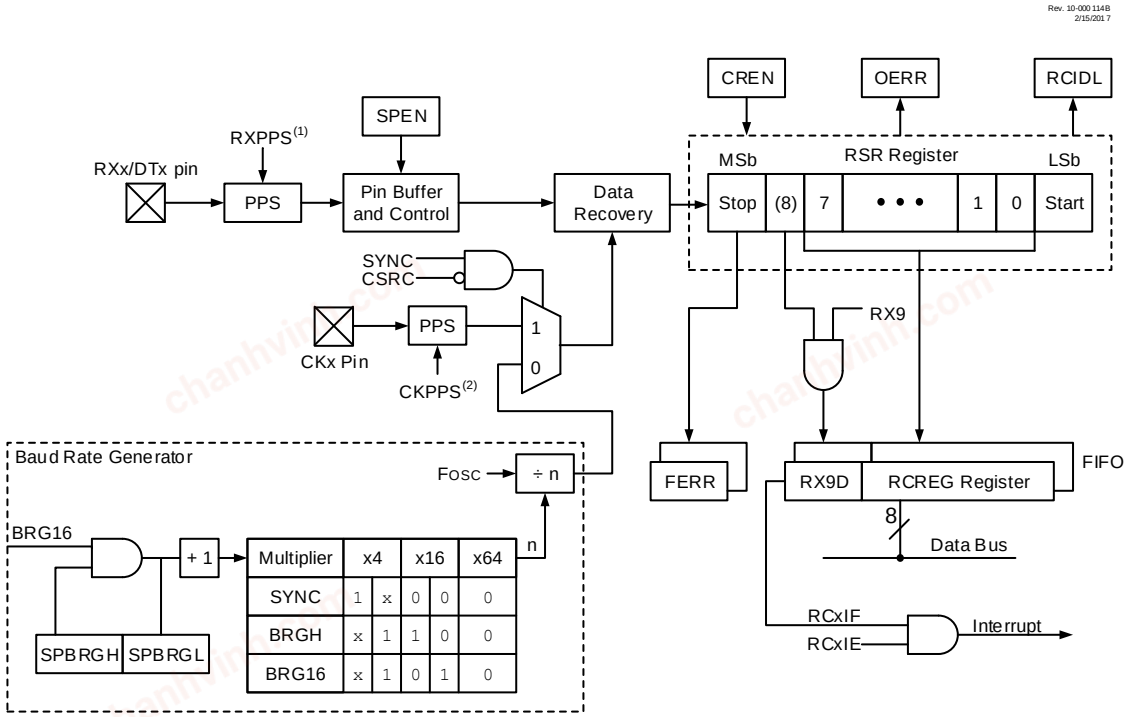



**Notes:** 1. In Synchronous mode, the DT output and RX input PPS selections will enable the same pin.
2. In Host Synchronous mode, the TX output and CK input PPS selections will enable the same pin.

chanhvinh.com



**Figure 32-2. EUSART Receive Block Diagram**




**Notes:** 1. In Synchronous mode, the DT output and RX input PPS selections will enable the same pin.
2. In Host Synchronous mode, the TX output and CK input PPS selections will enable the same pin.

## 32.1 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 32-2 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.

### 32.1.1 EUSART Asynchronous Transmitter

Figure 32-1 is a simplified representation of the transmitter. 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.

#### 32.1.1.1 Enabling the Transmitter

The EUSART transmitter is enabled for asynchronous operations by configuring the following three control bits:

- The Transmit Enable (TXEN) bit is set to '1' to enable the transmitter circuitry of the EUSART
- The EUSART Mode Select (SYNC) bit is set to '0' to configure the EUSART for asynchronous operation

chanhvinh.com



- The Serial Port Enable (SPEN) bit is set to '1' to enable the EUSART interface and to enable automatically the output drivers for the RxyPPS selected as the TXx/CKx output

All other EUSART control bits are assumed to be in their default state.

If the TXx/CKx pin is shared with an analog peripheral, the analog I/O function must be disabled by clearing the corresponding ANSEL bit.

**Important:** The TXxIF Transmitter Interrupt Flag in the PIRx register is set when the TXEN enable bit is set and the Transmit Shift Register (TSR) is Idle.

#### 32.1.1.2 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.

#### 32.1.1.3 Transmit Data Polarity

The polarity of the transmit data can be controlled with the Clock/Transmit Polarity Select (SCKP) bit. 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 32.4.1.2. Clock Polarity for more details.

#### 32.1.1.4 Transmit Interrupt Flag

The EUSART Transmit Interrupt Flag (TXxIF) bit of the PIRx register is set whenever the EUSART transmitter is enabled and no character is being held for transmission in the TXxREG. In other words, the TXxIF bit is only cleared when the TSR is busy with a character and a new character has been queued for transmission in the TXxREG. The TXxIF flag bit is not cleared immediately upon writing TXxREG. TXxIF becomes valid in the second instruction cycle following the write execution. Polling TXxIF immediately following the TXxREG write will return invalid results. The TXxIF bit is read-only, it cannot be set or cleared by software.

The TXxIF interrupt can be enabled by setting the EUSART Transmit Interrupt Enable (TXxIE) bit of the PIEx register. However, the TXxIF flag bit will be set whenever the TXxREG is empty, regardless of the state of TXxIE enable bit.

To use interrupts when transmitting data, set the TXxIE bit only when there is more data to send. Clear the TXxIE interrupt enable bit upon writing the last character of the transmission to the TXxREG.

#### 32.1.1.5 TSR Status

The Transmit Shift Register Status (TRMT) bit indicates the status of the TSR register. This is a read-only bit. The TRMT bit is set when the TSR register is empty and is cleared when a character is transferred to the TSR register from the TXxREG. The TRMT bit remains clear until all bits have been shifted out of the TSR register. No interrupt logic is tied to this bit, so the user needs to poll this bit to determine the TSR status.

**Important:** The TSR register is not mapped in data memory, so it is not available to the user.

#### 32.1.1.6 Transmitting 9-Bit Characters

The EUSART supports 9-bit character transmissions. When the 9-Bit Transmit Enable (TX9) bit is set, the EUSART will shift nine bits out for each character transmitted. The TX9D bit 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 register immediately after the TXxREG is written.

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



A special 9-bit Address mode is available for use with multiple receivers. See 32.1.2.7. Address Detection for more information on the Address mode.

### 32.1.1.7 Asynchronous Transmission Setup

1. Initialize the SPxBRGH:SPxBRGL register pair and the BRGH and BRG16 bits to achieve the desired baud rate (see 32.3. EUSART Baud Rate Generator (BRG)).
2. Select the transmit output pin by writing the appropriate value to the RxyPPS register.
3. Enable the asynchronous serial port by clearing the SYNC bit and setting the SPEN bit.
4. If 9-bit transmission is desired, set the TX9 control bit. That will indicate that the eight Least Significant data bits are an address when the receiver is set for address detection.
5. Set SCKP bit if inverted transmit is desired.
6. Enable the transmission by setting the TXEN control bit. This will cause the TXxIF interrupt bit to be set.
7. If interrupts are desired, set the TXxIE interrupt enable bit of the PIEx register.
8. An interrupt will occur immediately provided that the GIE and PEIE bits of the INTCON register are also set.
9. If 9-bit transmission is selected, the ninth bit will be loaded into the TX9D data bit.
10. Load 8-bit data into the TXxREG register. This will start the transmission.

**Figure 32-3. Asynchronous Transmission**

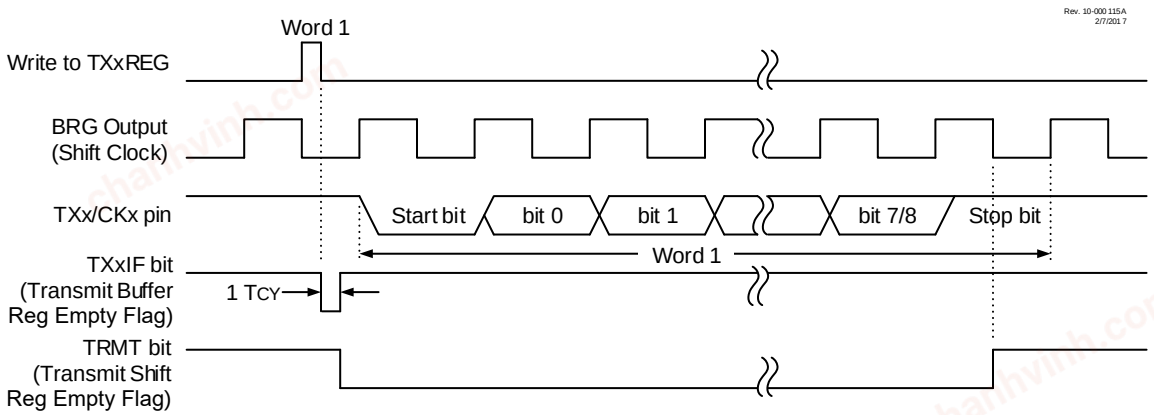
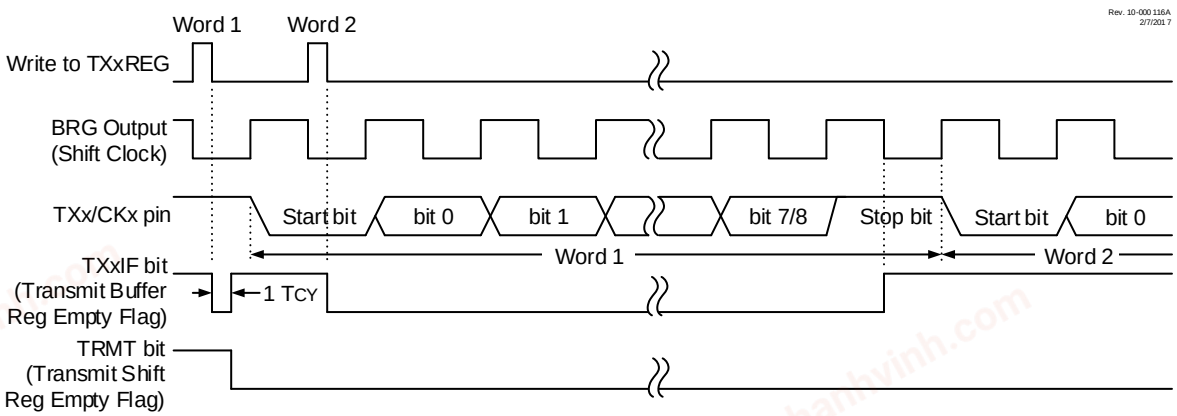



**Figure 32-4. Asynchronous Transmission (Back-to-Back)**




## 32.1.2 EUSART Asynchronous Receiver

The Asynchronous mode is typically used in RS-232 systems. A simplified representation of the receiver is shown in Figure 32-2. The data is received on the RXx/DTx pin and drives the data recovery block. The data recovery block is actually a high-speed shifter operating at 16 times the baud rate, whereas the serial Receive Shift Register (RSR) operates at the bit rate. When all 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

chanhvinh.com



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.

#### 32.1.2.1 Enabling the Receiver

The EUSART receiver is enabled for asynchronous operation by configuring the following three control bits:

- The Continuous Receive Enable (CREN) bit is set to '1' to enables the receiver circuitry of the EUSART
- The EUSART Mode Select (SYNC) bit is set to '0' to configure the EUSART for asynchronous operation
- The Serial Port Enable (SPEN) bit is set to '1' to enable the EUSART interface

All other EUSART control bits are assumed to be in their default state.

The user must set the RXxPPS register to select the RXx/DTx I/O pin and set the corresponding TRIS bit to configure the pin as an input.

**Important:** If the RX/DT function is on an analog pin, the corresponding ANSEL bit must be cleared for the receiver to function.

#### 32.1.2.2 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 32.1.2.4. 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 EUSART Receive Interrupt Flag (RCxIF) bit of the PIRx register is set. The top character in the FIFO is transferred out of the FIFO by reading the RCxREG register.

**Important:** If the receive FIFO is overrun, no additional characters will be received until the Overrun condition is cleared. See 32.1.2.4. Receive Framing Error for more information.

#### 32.1.2.3 Receive Interrupts

The EUSART Receive Interrupt Flag (RCxIF) bit of the PIRx register is set whenever the EUSART receiver is enabled and there is an unread character in the receive FIFO. The RCxIF Interrupt Flag bit is read-only, it cannot be set or cleared by software.

RCxIF interrupts are enabled by setting all of the following bits:

- RCxIE, Interrupt Enable bit of the PIEx register
- PEIE, Peripheral Interrupt Enable bit of the INTCON register
- GIE, Global Interrupt Enable bit of the INTCON register

The RCxIF Interrupt Flag bit will be set when there is an unread character in the FIFO, regardless of the state of interrupt enable bits.

#### 32.1.2.4 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 Framing Error (FERR) bit. 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 register.

chanhvinh.com



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, which resets the EUSART. Clearing the CREN bit does not affect the FERR bit. A framing error by itself does not generate an interrupt.

**Important:** If all receive characters in the receive FIFO have framing errors, repeated reads of the RCxREG register will not clear the FERR bit.

#### 32.1.2.5 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 Overrun Error (OERR) bit 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 or by resetting the EUSART by clearing the SPEN bit.

#### 32.1.2.6 Receiving 9-Bit Characters

The EUSART supports 9-bit character reception. When the 9-Bit Receive Enable (RX9) bit is set, the EUSART will shift nine bits into the RSR for each character received. The RX9D bit 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 register.

#### 32.1.2.7 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 Address Detect Enable (ADDEN) bit.

Address detection requires 9-bit character reception. When address detection is enabled, only characters with the ninth data bit set will be transferred to the receive FIFO buffer, thereby setting the RCxIF interrupt bit. All other characters will be ignored.

Upon receiving an address character, user software determines if the address matches its own. Upon address match, user software must disable address detection by clearing the ADDEN bit before the next Stop bit occurs. When user software detects the end of the message, determined by the message protocol used, software places the receiver back into the Address Detection mode by setting the ADDEN bit.

#### 32.1.2.8 Asynchronous Reception Setup

1. Initialize the SPxBRGH:SPxBRGL register pair and the BRGH and BRG16 bits to achieve the desired baud rate (see 32.3. EUSART Baud Rate Generator (BRG)).
2. Set the RXxPPS register to select the RXx/DTx input pin.
3. Clear the ANSEL bit for the RXx pin (if applicable).
4. Enable the serial port by setting the SPEN bit. The SYNC bit must be cleared for asynchronous operation.
5. If interrupts are desired, set the RCxIE bit of the PIEx register and the GIE and PEIE bits of the INTCON register.
6. If 9-bit reception is desired, set the RX9 bit.
7. Enable reception by setting the CREN bit.
8. The RCxIF Interrupt Flag bit will be set when a character is transferred from the RSR to the receive buffer. An interrupt will be generated if the RCxIE interrupt enable bit was also set.
9. Read the RCxSTA register to get the Error flags and, if 9-bit data reception is enabled, the ninth data bit.
10. Get the received eight Least Significant data bits from the receive buffer by reading the RCxREG register.
11. If an overrun occurred, clear the OERR flag by clearing the CREN receiver enable bit.

#### 32.1.2.9 9-Bit Address Detection Mode Setup

This mode is typically used in RS-485 systems. To set up an Asynchronous Reception with Address Detect Enable, follow these steps:

1. Initialize the SPxBRGH:SPxBRGL register pair and the BRGH and BRG16 bits to achieve the desired baud rate (see 32.3. EUSART Baud Rate Generator (BRG)).

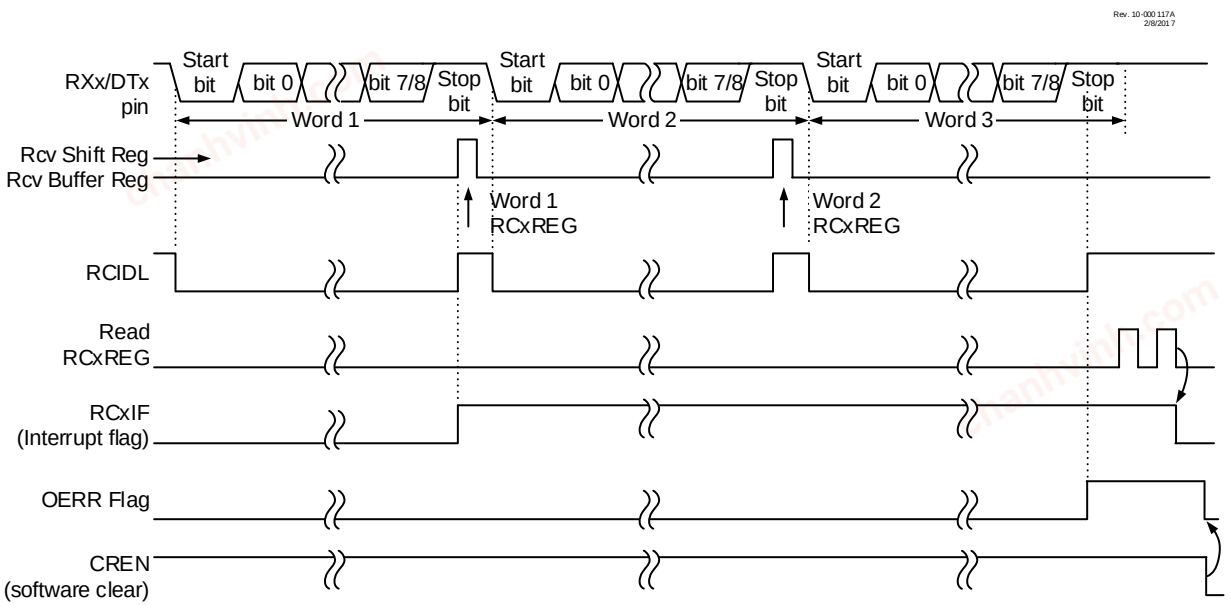
© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



2. Set the RXxPPS register to select the RXx input pin.
3. Clear the ANSEL bit for the RXx pin (if applicable).
4. Enable the serial port by setting the SPEN bit. The SYNC bit must be cleared for asynchronous operation.
5. If interrupts are desired, set the RCxIE bit of the PIEx register and the GIE and PEIE bits of the INTCON register.
6. Enable 9-bit reception by setting the RX9 bit.
7. Enable address detection by setting the ADDEN bit.
8. Enable reception by setting the CREN bit.
9. The RCxIF Interrupt Flag bit will be set when a character with the ninth bit set is transferred from the RSR to the receive buffer. An interrupt will be generated if the RCxIE interrupt enable bit is also set.
10. Read the RCxSTA register to get the Error flags. The ninth data bit will always be set.
11. 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.
12. If an overrun occurred, clear the OERR flag by clearing the CREN receiver enable bit.
13. If the device has been addressed, clear the ADDEN bit to allow all received data into the receive buffer and generate interrupts.

**Figure 32-5. Asynchronous Reception**




**Note:** This timing diagram shows three bytes appearing on the RXx input. The OERR flag is set because the RCxREG register is not read before the third word is received.

## 32.2 Clock Accuracy with Asynchronous Operation

The factory calibrates the internal oscillator block output (INTOSC). However, the INTOSC frequency may drift as $V_{DD}$ 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.

The other method adjusts the value in the Baud Rate Generator. This can be done automatically with the Auto-Baud Detect feature (see 32.3.1. 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.

chanhvinh.com



## 32.3 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 operations. By default, the BRG operates in 8-bit mode. Setting the BRG16 bit 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 and the BRG16 bits. In Synchronous mode, the BRGH bit is ignored.

Table 32-1 contains the formulas for determining the baud rate. Equation 32-1 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 and are shown in the table below. 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. The BRGH bit is used to achieve very high baud rates.

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, check the status of the Receive Idle Flag (RCIDL) bit to make sure the receive operation is idle before changing the system clock.

**Equation 32-1. Calculating Baud Rate Error**

For a device with $F_{OSC}$ of 16 MHz, desired baud rate of 9600, Asynchronous mode, 8-bit BRG:

$$DesiredBaudrate = \frac{F_{OSC}}{64 \times (SPxBRG + 1)}$$

Solving for SPxBRG:

$$SPxBRG = \frac{F_{OSC}}{64 \times DesiredBaudrate} - 1$$

$$SPxBRG = \frac{16000000}{64 \times 9600} - 1$$

$$SPxBRG = 25.042 \simeq 25$$

$$CalculatedBaudrate = \frac{16000000}{64 \times (25 + 1)}$$

$$CalculatedBaudrate = 9615$$

$$Error = \frac{CalculatedBaudrate - DesiredBaudrate}{DesiredBaudrate}$$

$$Error = \frac{9615 - 9600}{9600}$$

$$Error = 0.16\,\%$$

**Table 32-1. 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 | |

chanhvinh.com



| .....continued | | | | |
|---|---|---|---|---|
| Configuration Bits | | | BRG/EUSART Mode | Baud Rate Formula |
| SYNC | BRG16 | BRGH | | |
| 0 | 1 | 1 | 16-bit/Asynchronous | $F_{OSC}/[4\ (n+1)]$ |
| 1 | 0 | x | 8-bit/Synchronous | |
| 1 | 1 | x | 16-bit/Synchronous | |

**Note:** x = Don't care, n = value of SPxBRGH:SPxBRGL register pair.

**Table 32-2. Sample Baud Rates for Asynchronous Modes**

| Baud Rate | SYNC = 0, BRGH = 0, BRG16 = 0 | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | $F_{OSC}$ = 32.000 MHz | | | $F_{OSC}$ = 20.000 MHz | | | $F_{OSC}$ = 18.432 MHz | | | $F_{OSC}$ = 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 | 1200 | 0.00 | 143 |
| 2400 | 2404 | 0.16 | 207 | 2404 | 0.16 | 129 | 2400 | 0.00 | 119 | 2400 | 0.00 | 71 |
| 9600 | 9615 | 0.16 | 51 | 9470 | -1.36 | 32 | 9600 | 0.00 | 29 | 9600 | 0.00 | 17 |
| 10417 | 10417 | 0.00 | 47 | 10417 | 0.00 | 29 | 10286 | -1.26 | 27 | 10165 | -2.42 | 16 |
| 19.2k | 19.23k | 0.16 | 25 | 19.53k | 1.73 | 15 | 19.20k | 0.00 | 14 | 19.20k | 0.00 | 8 |
| 57.6k | 55.55k | -3.55 | 3 | — | — | — | 57.60k | 0.00 | 7 | 57.60k | 0.00 | 2 |
| 115.2k | — | — | — | — | — | — | — | — | — | — | — | — |

| Baud Rate | SYNC = 0, BRGH = 0, BRG16 = 0 | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | $F_{OSC}$ = 8.000 MHz | | | $F_{OSC}$ = 4.000 MHz | | | $F_{OSC}$ = 3.6864 MHz | | | $F_{OSC}$ = 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 | — | — | — | — | — | — | — | — | — | — | — | — |

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



| Baud Rate | SYNC = 0, BRGH = 1, BRG16 = 0 | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | $F_{OSC}$ = 32.000 MHz | | | $F_{OSC}$ = 20.000 MHz | | | $F_{OSC}$ = 18.432 MHz | | | $F_{OSC}$ = 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 | 207 | 9615 | 0.16 | 129 | 9600 | 0.00 | 119 | 9600 | 0.00 | 71 |
| 10417 | 10417 | 0.00 | 191 | 10417 | 0.00 | 119 | 10378 | -0.37 | 110 | 10473 | 0.53 | 65 |
| 19.2k | 19.23k | 0.16 | 103 | 19.23k | 0.16 | 64 | 19.20k | 0.00 | 59 | 19.20k | 0.00 | 35 |
| 57.6k | 57.14k | -0.79 | 34 | 56.82k | -1.36 | 21 | 57.60k | 0.00 | 19 | 57.60k | 0.00 | 11 |
| 115.2k | 117.64k | 2.12 | 16 | 113.64k | -1.36 | 10 | 115.2k | 0.00 | 9 | 115.2k | 0.00 | 5 |

| Baud Rate | SYNC = 0, BRGH = 1, BRG16 = 0 | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | $F_{OSC}$ = 8.000 MHz | | | $F_{OSC}$ = 4.000 MHz | | | $F_{OSC}$ = 3.6864 MHz | | | $F_{OSC}$ = 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 | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | $F_{OSC}$ = 32.000 MHz | | | $F_{OSC}$ = 20.000 MHz | | | $F_{OSC}$ = 18.432 MHz | | | $F_{OSC}$ = 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 | 6666 | 300.0 | -0.01 | 4166 | 300.0 | 0.00 | 3839 | 300.0 | 0.00 | 2303 |
| 1200 | 1200 | -0.02 | 3332 | 1200 | -0.03 | 1041 | 1200 | 0.00 | 959 | 1200 | 0.00 | 575 |
| 2400 | 2401 | -0.04 | 832 | 2399 | -0.03 | 520 | 2400 | 0.00 | 479 | 2400 | 0.00 | 287 |
| 9600 | 9615 | 0.16 | 207 | 9615 | 0.16 | 129 | 9600 | 0.00 | 119 | 9600 | 0.00 | 71 |
| 10417 | 10417 | 0.00 | 191 | 10417 | 0.00 | 119 | 10378 | -0.37 | 110 | 10473 | 0.53 | 65 |
| 19.2k | 19.23k | 0.16 | 103 | 19.23k | 0.16 | 64 | 19.20k | 0.00 | 59 | 19.20k | 0.00 | 35 |

© 2023 Microchip Technology Inc.
and its subsidiaries

chanhvinh.com



| | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 57.6k | 57.14k | -0.79 | 34 | 56.818 | -1.36 | 21 | 57.60k | 0.00 | 19 | 57.60k | 0.00 | 11 |
| 115.2k | 117.6k | 2.12 | 16 | 113.636 | -1.36 | 10 | 115.2k | 0.00 | 9 | 115.2k | 0.00 | 5 |

| Baud Rate | SYNC = 0, BRGH = 0, BRG16 = 1 | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | $F_{OSC}$ = 8.000 MHz | | | $F_{OSC}$ = 4.000 MHz | | | $F_{OSC}$ = 3.6864 MHz | | | $F_{OSC}$ = 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 | — | — | — |

| Baud Rate | SYNC = 0, BRGH = 1, BRG16 = 1 or SYNC = 1, BRG16 = 1 | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | $F_{OSC}$ = 32.000 MHz | | | $F_{OSC}$ = 20.000 MHz | | | $F_{OSC}$ = 18.432 MHz | | | $F_{OSC}$ = 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 | 26666 | 300.0 | 0.00 | 16665 | 300.0 | 0.00 | 15359 | 300.0 | 0.00 | 9215 |
| 1200 | 1200 | 0.00 | 6666 | 1200 | -0.01 | 4166 | 1200 | 0.00 | 3839 | 1200 | 0.00 | 2303 |
| 2400 | 2400 | 0.01 | 3332 | 2400 | 0.02 | 2082 | 2400 | 0.00 | 1919 | 2400 | 0.00 | 1151 |
| 9600 | 9604 | 0.04 | 832 | 9597 | -0.03 | 520 | 9600 | 0.00 | 479 | 9600 | 0.00 | 287 |
| 10417 | 10417 | 0.00 | 767 | 10417 | 0.00 | 479 | 10425 | 0.08 | 441 | 10433 | 0.16 | 264 |
| 19.2k | 19.18k | -0.08 | 416 | 19.23k | 0.16 | 259 | 19.20k | 0.00 | 239 | 19.20k | 0.00 | 143 |
| 57.6k | 57.55k | -0.08 | 138 | 57.47k | -0.22 | 86 | 57.60k | 0.00 | 79 | 57.60k | 0.00 | 47 |
| 115.2k | 115.9k | 0.64 | 68 | 116.3k | 0.94 | 42 | 115.2k | 0.00 | 39 | 115.2k | 0.00 | 23 |

| Baud Rate | SYNC = 0, BRGH = 1, BRG16 = 1 or SYNC = 1, BRG16 = 1 | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | $F_{OSC}$ = 8.000 MHz | | | $F_{OSC}$ = 4.000 MHz | | | $F_{OSC}$ = 3.6864 MHz | | | $F_{OSC}$ = 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 |

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



| | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 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 | — | — | — |

### 32.3.1 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 Auto-Baud Detect Enable (ABDEN) bit 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 register begins counting up using the BRG counter clock as shown in Figure 32-6. The fifth rising edge will occur on the RXx pin at the end of the eighth bit period. At that time, an accumulated value totaling the proper BRG period is left in the SPxBRGH:SPxBRGL register pair, the ABDEN bit is automatically cleared, and the RCxIF interrupt flag is set. The value in the RCxREG register needs to be read to clear the RCxIF interrupt. RCxREG content may 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 32-3. 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 1/8$^{th}$ the BRG base clock rate. The resulting byte measurement is the average bit time when clocked at full speed.

**Notes:**

1. If the Wake-Up Enable (WUE) bit is set with the ABDEN bit, auto-baud detection will occur on the byte following the Break character (see 32.3.3. 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 32-3. BRG Counter Clock Rates**

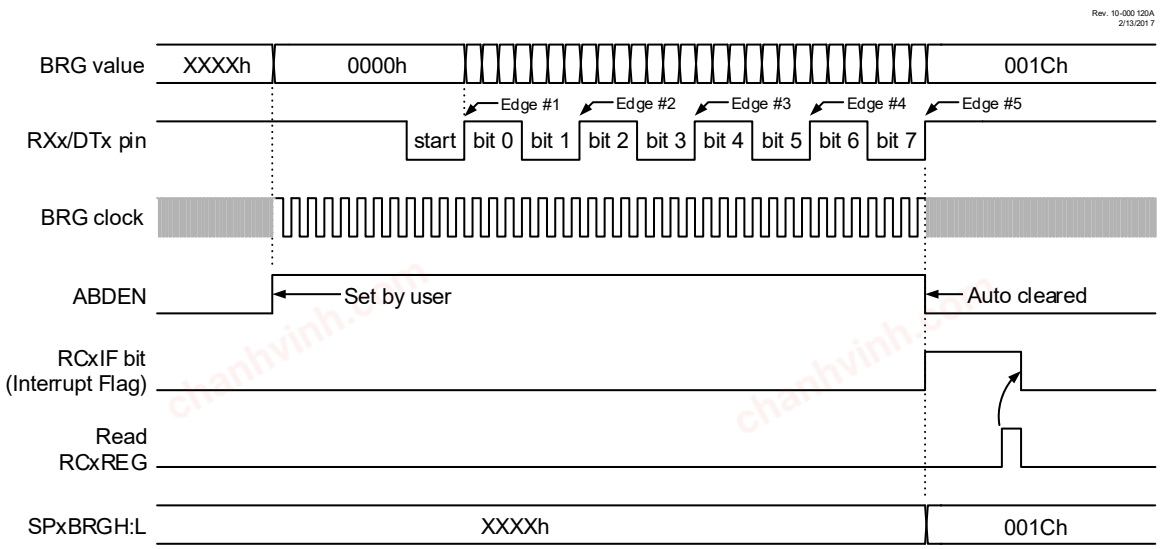
| BRG16 | BRGH | BRG Base Clock | BRG ABD Clock |
|---|---|---|---|
| 1 | 1 | $F_{OSC}/4$ | $F_{OSC}/32$ |
| 1 | 0 | $F_{OSC}/16$ | $F_{OSC}/128$ |
| 0 | 1 | $F_{OSC}/16$ | $F_{OSC}/128$ |
| 0 | 0 | $F_{OSC}/64$ | $F_{OSC}/512$ |

**Note:** During the ABD sequence, the SPxBRGL and SPxBRGH registers are both used as a 16-bit counter, independent of the BRG16 setting.

chanhvinh.com



**Figure 32-6. Automatic Baud Rate Calibration**




### 32.3.2 Auto-Baud Overflow

During the course of automatic baud detection, the Auto-Baud Detect Overflow (ABDOVF) bit will be set if the baud rate counter overflows before the fifth rising edge is detected on the RXx 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. After the ABDOVF bit has been set, the counter continues to count until the fifth rising edge is detected on the RXx pin. Upon detecting the fifth RX edge, the hardware will set the RCxIF interrupt flag and clear the ABDEN bit. The RCxIF flag can be subsequently cleared by reading the RCxREG register. The ABDOVF bit can be cleared by software directly.

To terminate the auto-baud process before the RCxIF flag is set, clear the ABDEN bit then clear the ABDOVF bit. The ABDOVF bit will remain set if the ABDEN bit is not cleared first.

### 32.3.3 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. 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 RCxIF interrupt coincident with the wake-up event. The interrupt is generated synchronously to the Q clocks in normal CPU operating modes as shown in Figure 32-7, and asynchronously if the device is in Sleep mode, as shown in Figure 32-8. 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.

#### 32.3.3.1 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 nonzero character is received, the low time from the Start bit to the first rising edge will be interpreted

© 2023 Microchip Technology Inc.
and its subsidiaries

chanhvinh.com



as the wake-up event. The remaining bits in the character will be received as a fragmented character and subsequent characters can result in framing or overrun errors.

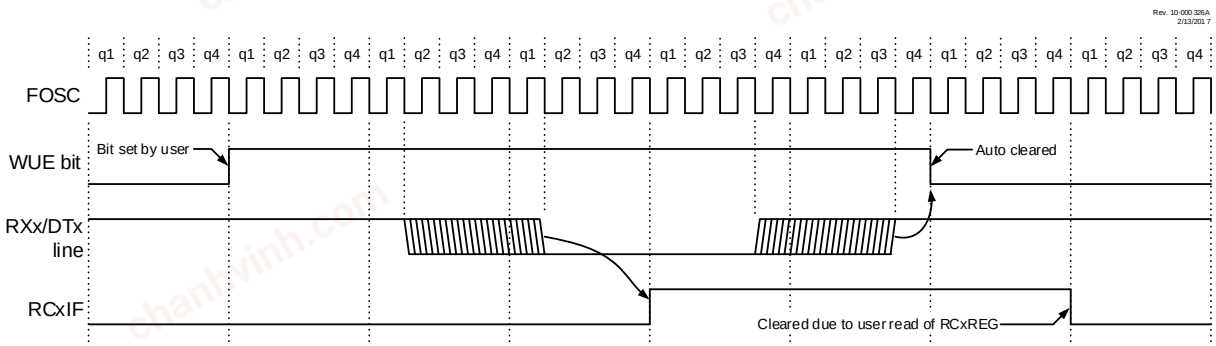
Therefore, the initial character in the transmission must be all '0's. This must be 10 or more bit times, 13-bit times recommended for LIN bus, or any number of bit times for standard RS-232 devices.

**WUE Bit**

The wake-up event causes a receive interrupt by setting the RCxIF 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.

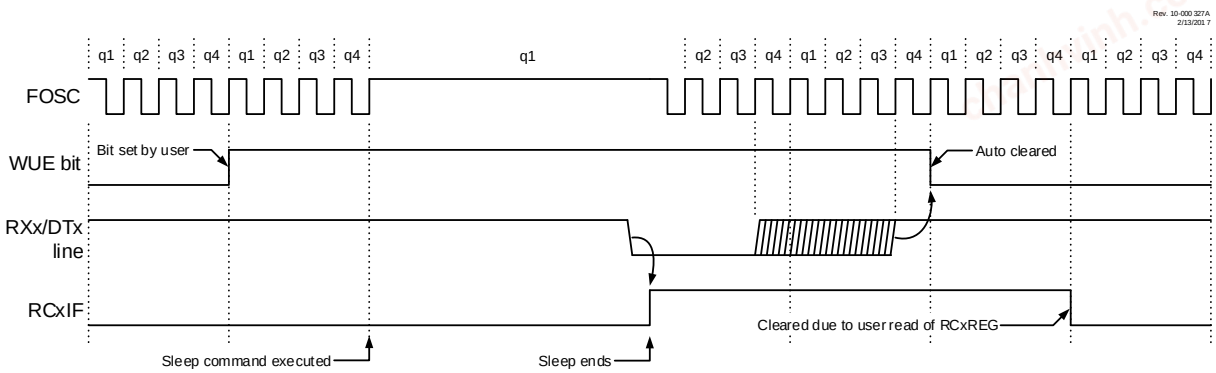
To ensure that no actual data is lost, check the RCIDL bit to verify that a receive operation is not in process before setting the WUE bit. If a receive operation is not occurring, the WUE bit may then be set just prior to entering the Sleep mode.

**Figure 32-7. Auto-Wake-Up (WUE) Bit Timing During Normal Operation**




**Note:** The EUSART remains in Idle while the WUE bit is set.

**Figure 32-8. Auto-Wake-Up (WUE) Bit Timings During Sleep**




**Note:** The EUSART remains in Idle while the WUE bit is set.

### 32.3.4 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 Send Break Character (SENDB) and Transmit Enable (TXEN) bits. 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).

chanhvinh.com



The Transmit Shift Register Status (TRMT) bit indicates when the transmit operation is Active or Idle, just as it does during normal transmission. See Figure 32-9 for more details.

#### 32.3.4.1 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 host.

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 TXxIF, the next data byte can be written to TXxREG.

### 32.3.5 Receiving a Break Character

The EUSART module can receive a Break character in two ways.

The first method to detect a Break character uses the Framing Error (FERR) bit and the received data as indicated by RCxREG. The Baud Rate Generator is assumed to have been initialized to the expected baud rate.

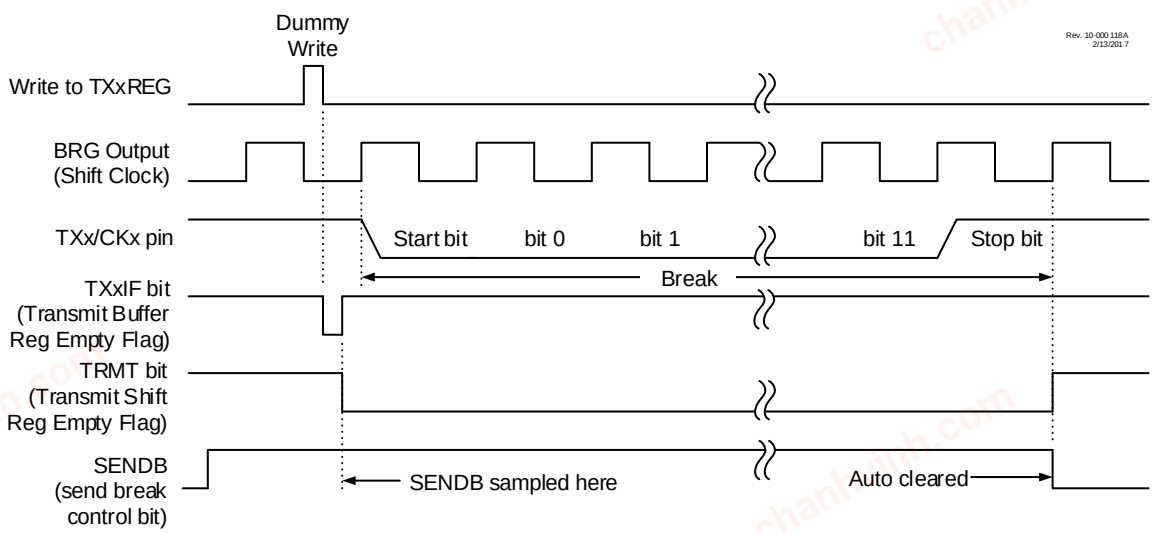
A Break character has been received when all three of the following conditions are true:

- RCxIF bit is set
- FERR bit is set
- RCxREG = 00h

The second method uses the Auto-Wake-Up feature described in 32.3.3. Auto-Wake-Up on Break. By enabling this feature, the EUSART will sample the next two transitions on RX/DT, cause an RCxIF interrupt, and receive the next data byte followed by another interrupt.

Note that following a Break character, the user will typically want to enable the Auto-Baud Detect feature. For both methods, the user can set the ABDEN bit before placing the EUSART in Sleep mode.

**Figure 32-9. Send Break Character Sequence**




## 32.4 EUSART Synchronous Mode

Synchronous serial communications are typically used in systems with a single host and one or more clients. The host device contains the necessary circuitry for baud rate generation and supplies the clock for all devices in the system. Client devices can take advantage of the host clock by eliminating the internal clock generation circuitry.

chanhvinh.com



There are two signal lines in Synchronous mode: A bidirectional data line (DT) and a clock line (CK). The clients use the external clock supplied by the host 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 host and client devices can receive and transmit data but not both simultaneously. The EUSART can operate as either a host or client device.

Start and Stop bits are not used in synchronous transmissions.

### 32.4.1 Synchronous Host Mode

The following bits are used to configure the EUSART for synchronous host operation:

- The SYNC bit is set to '1' to configure the EUSART for synchronous operation
- The Clock Source Select (CSRC) bit is set to '1' to configure the EUSART as the host
- The Single Receive Enable (SREN) bit is set to '0' for transmit; SREN = 1 for receive (recommended setting to receive 1 byte)
- The Continuous Receive Enable (CREN) bit is set to '0' for transmit; CREN = 1 to receive continuously
- The SPEN bit is set to '1' to enable the EUSART interface



**Important:** Clearing the SREN and CREN bits ensure that the device is in the Transmit mode, otherwise the device will be configured to receive.

#### 32.4.1.1 Host Clock

Synchronous data transfers use a separate clock line, which is synchronous with the data. A device configured as a host transmits the clock on the TX/CK line. The TXx/CKx pin output driver is automatically enabled when the EUSART is configured for synchronous transmit or receive operation. Serial data bits change on the leading edge to ensure they are valid at the trailing edge of each clock. One clock cycle is generated for each data bit. Only as many clock cycles are generated as there are data bits.

#### 32.4.1.2 Clock Polarity

A clock polarity option is provided for Microwire compatibility. Clock polarity is selected with the Clock/Transmit Polarity Select (SCKP) bit. 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.

#### 32.4.1.3 Synchronous Host Transmission

Data is transferred out of the device on the RXx/DTx pin. The RXx/DTx and TXx/CKx pin output drivers are automatically enabled when the EUSART is configured for synchronous host 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 host 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.

#### 32.4.1.4 Synchronous Host Transmission Setup

1. Initialize the SPxBRGH;SPxBRGL register pair and the BRG16 bit to achieve the desired baud rate (see 32.3. EUSART Baud Rate Generator (BRG)).
2. Select the transmit output pin by writing the appropriate values to the RxyPPS register and RXxPPS register. Both selections may enable the same pin.
3. Select the clock output pin by writing the appropriate values to the RxyPPS register and TXxPPS register. Both selections may enable the same pin.

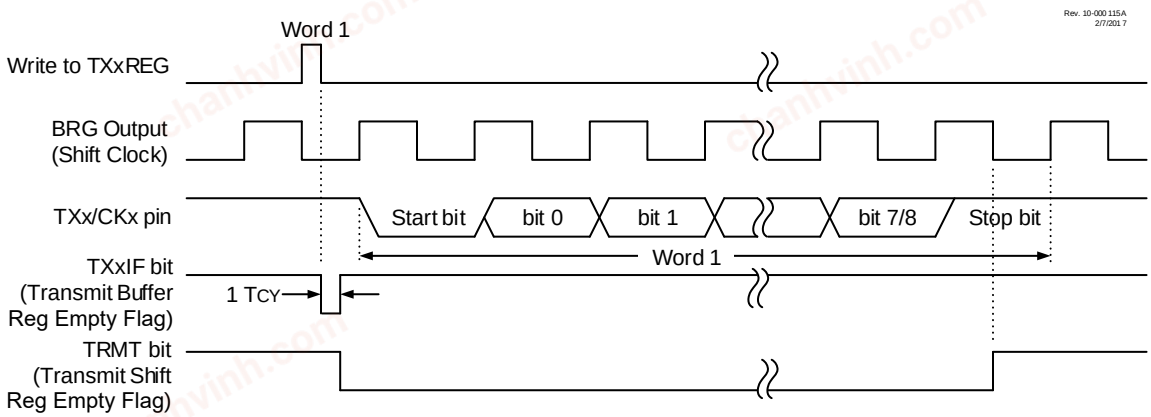
© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



4. Enable the synchronous host serial port by setting bits SYNC, SPEN and CSRC.
5. Disable Receive mode by clearing the SREN and CREN bits.
6. Enable Transmit mode by setting the TXEN bit.
7. If 9-bit transmission is desired, set the TX9 bit.
8. If interrupts are desired, set the TXxIE bit of the PIEx register and the GIE and PEIE bits of the INTCON register.
9. If 9-bit transmission is selected, the ninth bit will be loaded in the TX9D bit.
10. Start transmission by loading data to the TXxREG register.

**Figure 32-10. Synchronous Transmission**




### 32.4.1.5 Synchronous Host Reception

Data is received at the RXx/DTx pin. The RXx/DTx pin output driver is automatically disabled when the EUSART is configured for synchronous host receive operation.

In Synchronous mode, reception is enabled by setting either the Single Receive Enable (SREN) bit or the Continuous Receive Enable (CREN) bit.

When SREN is set and CREN is clear, only as many clock cycles are generated as there are data bits in a single character. The SREN bit is automatically cleared at the completion of one character. When CREN is set, clocks are continuously generated until CREN is cleared. If CREN is cleared in the middle of a character the CK clock stops immediately and the partial character is discarded. If SREN and CREN are both set, then SREN is cleared at the completion of the first character and CREN takes precedence.

To initiate reception, set either SREN or CREN. Data is sampled at the RXx/DTx pin on the trailing edge of the TX/CK clock pin and is shifted into the Receive Shift Register (RSR). When a complete character is received into the RSR, the RCxIF bit is set and the character is automatically transferred to the two character receive FIFO. The eight Least Significant bits of the top character in the receive FIFO are available in RCxREG. The RCxIF 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.

### 32.4.1.6 Client Clock

Synchronous data transfers use a separate clock line, which is synchronous with the data. A device configured as a client receives the clock on the TX/CK line. The TXx/CKx pin output driver is automatically disabled when the device is configured for synchronous client 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 may be received as there are data bits.

**Important:** If the device is configured as a client and the TX/CK function is on an analog pin, the corresponding ANSEL bit must be cleared.

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



#### 32.4.1.7 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 the FIFO is accessed. When this happens the Overrun Error (OERR) bit 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 or by resetting the EUSART by clearing the SPEN bit.

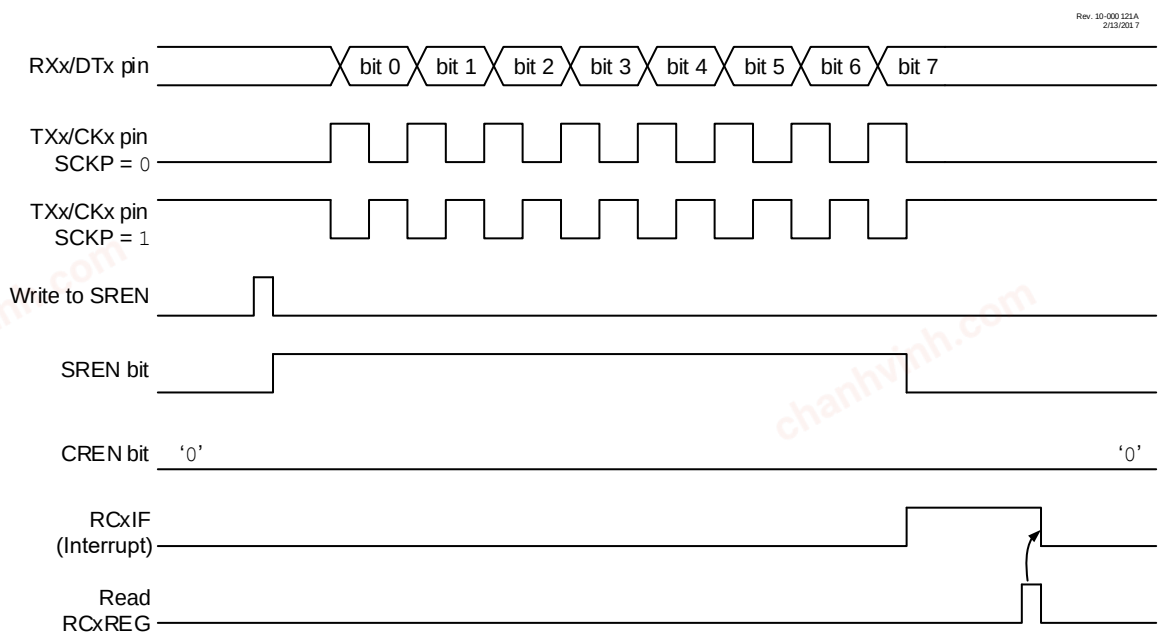
#### 32.4.1.8 Receiving 9-Bit Characters

The EUSART supports 9-bit character reception. When the 9-Bit Receive Enable (RX9) bit is set, the EUSART will shift nine bits into the RSR for each character received. The RX9D bit 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 register.

#### 32.4.1.9 Synchronous Host Reception Setup

1. Initialize the SPxBRGH:SPxBRGL register pair and set or clear the BRG16 bit, as required, to achieve the desired baud rate.
2. Select the receive input pin by writing the appropriate values to the RxyPPS and RXxPPS registers. Both selections may enable the same pin.
3. Select the clock output pin by writing the appropriate values to the RxyPPS and TXxPPS registers. Both selections may enable the same pin.
4. Clear the ANSEL bit for the RXx pin (if applicable).
5. Enable the synchronous host serial port by setting the SYNC, SPEN and CSRC bits.
6. Ensure that the CREN and SREN bits are cleared.
7. If interrupts are desired, set the RCxIE bit of the PIEx register and the GIE and PEIE bits of the INTCON register.
8. If 9-bit reception is desired, set the RX9 bit.
9. Start reception by setting the SREN bit, or for continuous reception set the CREN bit.
10. The RCxIF Interrupt Flag bit will be set when reception of a character is complete. An interrupt will be generated if the RCxIE enable bit was set.
11. Read the RCxSTA register to get the ninth bit (if enabled) and determine if any error occurred during reception.
12. Read the 8-bit received data by reading the RCxREG register.
13. If an overrun error occurs, clear the error by either clearing the CREN bit or by clearing the SPEN bit which resets the EUSART.

**Figure 32-11. Synchronous Reception (Host Mode, SREN)**



chanhvinh.com



### 32.4.2 Synchronous Client Mode

The following bits are used to configure the EUSART for synchronous client operation:

- SYNC = 1 (configures the EUSART for synchronous operation)
- CSRC = 0 (configures the EUSART as a client)
- SREN = 0 (for transmit); SREN = 1 (for single byte receive)
- CREN = 0 (for transmit); CREN = 1 (recommended setting for continuous receive)
- SPEN = 1 (enables the EUSART)

**Important:** Clearing the SREN and CREN bits ensure that the device is in Transmit mode, otherwise the device will be configured to receive.

#### 32.4.2.1 EUSART Synchronous Client Transmit

The operation of the Synchronous Host and Client modes are identical (see 32.4.1.3. Synchronous Host 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 TXxIF bit will not be set.
4. After the first character has been shifted out of TSR, the TXxREG register will transfer the second character to the TSR and the TXxIF bit will now be set.
5. If the PEIE and TXxIE bits are set, the interrupt will wake the device from Sleep and execute the next instruction. If the GIE bit is also set, the program will call the Interrupt Service Routine.

#### 32.4.2.2 Synchronous Client Transmission Setup

1. Set the SYNC and SPEN bits and clear the CSRC bit.
2. Select the transmit output pin by writing the appropriate values to the RxyPPS register and RXxPPS register. Both selections may enable the same pin.
3. Select the clock input pin by writing the appropriate value to the TXxPPS register.
4. Clear the ANSEL bit for the CKx pin (if applicable).
5. Clear the CREN and SREN bits.
6. If interrupts are desired, set the TXxIE bit of the PIEx register and the GIE and PEIE bits of the INTCON register.
7. If 9-bit transmission is desired, set the TX9 bit.
8. Enable transmission by setting the TXEN bit.
9. If 9-bit transmission is selected, insert the Most Significant bit into the TX9D bit.
10. Prepare for transmission by writing the eight Least Significant bits to the TXxREG register. The word will be transmitted in response to the Host clocks at the CKx pin.

#### 32.4.2.3 EUSART Synchronous Client Reception

The operation of the Synchronous Host and Client modes is identical (see 32.4.1.5. Synchronous Host 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 Client mode

A character may be received while in Sleep mode by setting the CREN bit prior to entering Sleep. Once the word is received, the RSR register will transfer the data to the RCxREG register. If the RCxIE enable bit is set, the interrupt generated will wake the device from Sleep and execute the next instruction. If the GIE bit is also set, the program will branch to the interrupt vector.

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



#### 32.4.2.4 Synchronous Client Reception Setup

1. Set the SYNC and SPEN bits and clear the CSRC bit.
2. Select the receive input pin by writing the appropriate value to the RXxPPS register.
3. Select the clock input pin by writing the appropriate values to the TXxPPS register.
4. Clear the ANSEL bit for both the TXx/CKx and RXx/DTx pins (if applicable).
5. If interrupts are desired, set the RCxIE bit of the PIEx register and the GIE and PEIE bits of the INTCON register.
6. If 9-bit reception is desired, set the RX9 bit.
7. Set the CREN bit to enable reception.
8. The RCxIF bit will be set when reception is complete. An interrupt will be generated if the RCxIE bit was set.
9. If 9-bit mode is enabled, retrieve the Most Significant bit from the RX9D bit.
10. Retrieve the eight Least Significant bits from the receive FIFO by reading the RCxREG register.
11. If an overrun error occurs, clear the error by either clearing the CREN bit or by clearing the SPEN bit which resets the EUSART.

## 32.5 EUSART Operation During Sleep

The EUSART will remain active during Sleep only in the Synchronous Client 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 Client mode uses an externally generated clock to run the Transmit and Receive Shift registers.

### 32.5.1 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 Client Reception (see 32.4.2.4. Synchronous Client Reception Setup).
- If interrupts are desired, set the RCxIE bit of the PIEx register and the GIE and PEIE bits of the INTCON register.
- The RCxIF 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 RXx/DTx and TXx/CKx pins, respectively. When the data word has been completely clocked in by the external device, the RCxIF Interrupt Flag bit of the PIRx 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 (ISR) will be called.

### 32.5.2 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 client transmission (see 32.4.2.2. Synchronous Client Transmission Setup).
- The TXxIF interrupt flag must be cleared by writing the output data to the TXxREG, thereby filling the TSR and transmit buffer.
- The TXxIE interrupt enable bits of the PIEx register and PEIE of the INTCON register must be written to '`1`'.
- If interrupts are desired, set the GIE bit of the INTCON register.

Upon entering Sleep mode, the device will be ready to accept clocks on the TXx/CKx pin and transmit data on the RXx/DTx pin. When the data word in the TSR register has been completely clocked out by the external device, the pending byte in the TXxREG will transfer to the TSR and the TXxIF flag will be set. Thereby, waking the processor from Sleep. At this point, the TXxREG is available to accept another character for transmission. Writing TXxREG will clear the TXxIF 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 (ISR) will be called.

© 2023 Microchip Technology Inc.
and its subsidiaries

chanhvinh.com



## 32.6 Register Definitions: EUSART Control

chanhvinh.com

chanhvinh.com



### 32.6.1 TXxSTA

**Name:** TXxSTA
**Offset:** 0x0711,0x071B

Transmit Status and Control Register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CSRC | TX9 | TXEN | SYNC | SENDB | BRGH | TRMT | TX9D |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |

**Bit 7 – CSRC** Clock Source Select

| Value | Condition | Description |
|---|---|---|
| 1 | SYNC = 1 | Host mode (clock generated internally from BRG) |
| 0 | SYNC = 1 | Client mode (clock from external source) |
| X | SYNC = 0 | Don't care |

**Bit 6 – TX9** 9-Bit Transmit Enable

| Value | Description |
|---|---|
| 1 | Selects 9-bit transmission |
| 0 | Selects 8-bit transmission |

**Bit 5 – TXEN** Transmit Enable
Enables transmitter(1)

| Value | Description |
|---|---|
| 1 | Transmit enabled |
| 0 | Transmit disabled |

**Bit 4 – SYNC** EUSART Mode Select

| Value | Description |
|---|---|
| 1 | Synchronous mode |
| 0 | Asynchronous mode |

**Bit 3 – SENDB** Send Break Character

| Value | Condition | Description |
|---|---|---|
| 1 | SYNC = 0 | Send Sync Break on next transmission (cleared by hardware upon completion) |
| 0 | SYNC = 0 | Sync Break transmission disabled or completed |
| X | SYNC = 1 | Don't care |

**Bit 2 – BRGH** High Baud Rate Select

| Value | Condition | Description |
|---|---|---|
| 1 | SYNC = 0 | High speed, if BRG16 = 1, baud rate is baudclk/4; else baudclk/16 |
| 0 | SYNC = 0 | Low speed |
| X | SYNC = 1 | Don't care |

**Bit 1 – TRMT** Transmit Shift Register (TSR) Status

| Value | Description |
|---|---|
| 1 | TSR is empty |
| 0 | TSR is not empty |

**Bit 0 – TX9D** Ninth Bit of Transmit Data
Can be address/data bit or a parity bit.

**Note:** 1. The SREN and CREN bits override TXEN in Sync mode.

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



### 32.6.2 RCxSTA

**Name:** RCxSTA
**Offset:** 0x0710,0x071A

Receive Status and Control Register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SPEN | RX9 | SREN | CREN | ADDEN | FERR | OERR | RX9D |
| Access | R/W | R/W | R/W/HC | R/W | R/W | R | R/HC | R/HC |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 7 – SPEN** Serial Port Enable

| Value | Description |
|---|---|
| 1 | Serial port enabled |
| 0 | Serial port disabled (held in Reset) |

**Bit 6 – RX9** 9-Bit Receive Enable

| Value | Description |
|---|---|
| 1 | Selects 9-bit reception |
| 0 | Selects 8-bit reception |

**Bit 5 – SREN** Single Receive Enable

Controls reception. This bit is cleared by hardware when reception is complete

| Value | Condition | Description |
|---|---|---|
| 1 | SYNC = 1 AND CSRC = 1 | Start single receive |
| 0 | SYNC = 1 AND CSRC = 1 | Single receive is complete |
| X | SYNC = 0 OR CSRC = 0 | Don't care |

**Bit 4 – CREN** Continuous Receive Enable

| Value | Condition | Description |
|---|---|---|
| 1 | SYNC = 1 | Enables continuous receive until enable bit CREN is cleared (CREN overrides SREN) |
| 0 | SYNC = 1 | Disables continuous receive |
| 1 | SYNC = 0 | Enables receiver |
| 0 | SYNC = 0 | Disables receiver |

**Bit 3 – ADDEN** Address Detect Enable

| Value | Condition | Description |
|---|---|---|
| 1 | SYNC = 0 AND RX9 = 1 | The receive buffer is loaded and the interrupt occurs only when the ninth received bit is set |
| 0 | SYNC = 0 AND RX9 = 1 | All bytes are received and interrupt always occurs. Ninth bit can be used as parity bit |
| X | RX9 = 0 OR SYNC = 1 | Don't care |

**Bit 2 – FERR** Framing Error

| Value | Description |
|---|---|
| 1 | Unread byte in RCxREG has a framing error |
| 0 | Unread byte in RCxREG does not have a framing error |

**Bit 1 – OERR** Overrun Error

| Value | Description |
|---|---|
| 1 | Overrun error (can be cleared by clearing either SPEN or CREN bit) |
| 0 | No overrun error |

**Bit 0 – RX9D** Ninth bit of Received Data

This can be address/data bit or a parity bit which is determined by user firmware.

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



### 32.6.3 BAUDxCON

**Name:** BAUDxCON
**Offset:** 0x0712,0x071C

Baud Rate Control Register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ABDOVF | RCIDL | | SCKP | BRG16 | | WUE | ABDEN |
| Access | R | R | | R/W | R/W | | R/W | R/W |
| Reset | 0 | 1 | | 0 | 0 | | 0 | 0 |

**Bit 7 – ABDOVF** Auto-Baud Detect Overflow

| Value | Condition | Description |
|---|---|---|
| 1 | SYNC = 0 | Auto-baud timer overflowed |
| 0 | SYNC = 0 | Auto-baud timer did not overflow |
| X | SYNC = 1 | Don't care |

**Bit 6 – RCIDL** Receive Idle Flag

| Value | Condition | Description |
|---|---|---|
| 1 | SYNC = 0 | Receiver is Idle |
| 0 | SYNC = 0 | Start bit has been received and the receiver is receiving |
| X | SYNC = 1 | Don't care |

**Bit 4 – SCKP** Clock/Transmit Polarity Select

| Value | Condition | Description |
|---|---|---|
| 1 | SYNC = 0 | Idle state for transmit (TX) is a low level (transmit data inverted) |
| 0 | SYNC = 0 | Idle state for transmit (TX) is a high level (transmit data is noninverted) |
| 1 | SYNC = 1 | Data is clocked on rising edge of the clock |
| 0 | SYNC = 1 | Data is clocked on falling edge of the clock |

**Bit 3 – BRG16** 16-bit Baud Rate Generator Select

| Value | Description |
|---|---|
| 1 | 16-bit Baud Rate Generator is used |
| 0 | 8-bit Baud Rate Generator is used |

**Bit 1 – WUE** Wake-Up Enable

| Value | Condition | Description |
|---|---|---|
| 1 | SYNC = 0 | Receiver is waiting for a falling edge. Upon falling edge, no character will be received and the RCxIF flag will be set. WUE will automatically clear after RCxIF is set. |
| 0 | SYNC = 0 | Receiver is operating normally |
| X | SYNC = 1 | Don't care |

**Bit 0 – ABDEN** Auto-Baud Detect Enable

| Value | Condition | Description |
|---|---|---|
| 1 | SYNC = 0 | Auto-Baud Detect mode is enabled (clears when auto-baud is complete) |
| 0 | SYNC = 0 | Auto-Baud Detect is complete or mode is disabled |
| X | SYNC = 1 | Don't care |

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



### 32.6.4 RCxREG

**Name:** RCxREG
**Offset:** 0x070C,0x0716

Receive Data Register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RCREG[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 7:0 – RCREG[7:0]** Receive data

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



### 32.6.5 TXxREG

**Name:** TXxREG
**Offset:** 0x070D,0x0717

Transmit Data Register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TXREG[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 7:0 – TXREG[7:0]** Transmit Data

© 2023 Microchip Technology Inc.
and its subsidiaries

chanhvinh.com



### 32.6.6 SPxBRG

**Name:** SPxBRG
**Offset:** 0x070E,0x0718

EUSART Baud Rate Generator

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | SPBRG[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SPBRG[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 15:0 – SPBRG[15:0]** Baud Rate Register

**Notes:** The individual bytes in this multibyte register can be accessed with the following register names:

- SPxBRGH: Accesses the high byte SPBRG[15:8]
- SPxBRGL: Accesses the low byte SPBRG[7:0]

chanhvinh.com

chanhvinh.com



## 32.7 Register Summary - EUSART

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 ... 0x070B | Reserved | | | | | | | | | |
| 0x070C | RC1REG | 7:0 | RCREG[7:0] | | | | | | | |
| 0x070D | TX1REG | 7:0 | TXREG[7:0] | | | | | | | |
| 0x070E | SP1BRG | 7:0 | SPBRG[7:0] | | | | | | | |
| | | 15:8 | SPBRG[15:8] | | | | | | | |
| 0x0710 | RC1STA | 7:0 | SPEN | RX9 | SREN | CREN | ADDEN | FERR | OERR | RX9D |
| 0x0711 | TX1STA | 7:0 | CSRC | TX9 | TXEN | SYNC | SENDB | BRGH | TRMT | TX9D |
| 0x0712 | BAUD1CON | 7:0 | ABDOVF | RCIDL | | SCKP | BRG16 | | WUE | ABDEN |
| 0x0713 ... 0x0715 | Reserved | | | | | | | | | |
| 0x0716 | RC2REG | 7:0 | RCREG[7:0] | | | | | | | |
| 0x0717 | TX2REG | 7:0 | TXREG[7:0] | | | | | | | |
| 0x0718 | SP2BRG | 7:0 | SPBRG[7:0] | | | | | | | |
| | | 15:8 | SPBRG[15:8] | | | | | | | |
| 0x071A | RC2STA | 7:0 | SPEN | RX9 | SREN | CREN | ADDEN | FERR | OERR | RX9D |
| 0x071B | TX2STA | 7:0 | CSRC | TX9 | TXEN | SYNC | SENDB | BRGH | TRMT | TX9D |
| 0x071C | BAUD2CON | 7:0 | ABDOVF | RCIDL | | SCKP | BRG16 | | WUE | ABDEN |

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



# 33. ADC - Analog-to-Digital Converter with Computation Module

The Analog-to-Digital Converter with Computation module allows conversion of both single-ended and differential analog input signals to a 12-bit binary representation of that signal. This device uses analog inputs that are multiplexed into a single Sample-and-Hold circuit. The output of the Sample-and-Hold (S/H) circuit is connected to the input of the converter. The converter generates a 12-bit binary result via successive approximation and stores the conversion result into the ADC result registers. In single-ended conversions, the ADC measures the voltage between the selected analog input and $V_{SS}$ (0V). In differential conversions, the ADC measures the voltage difference between two selected analog inputs. The selected ADC input channels can either be from an internal source, such as the Fixed Voltage Reference (FVR), or from external analog input pins. Additionally, the following features are provided within the ADC module:

- Acquisition Timer
- Hardware Capacitive Voltage Divider (CVD) support:
  - Precharge timer
  - Adjustable Sample-and-Hold capacitor array
  - Guard ring digital output drive
- Automatic Repeat and Sequencing:
  - Automated double sample conversion for CVD
  - Two sets of Result registers (Current Result and Previous Result)
  - Auto-conversion trigger
  - Internal retrigger
- Channel Grouping:
  - Allows multiple input channels to be grouped together into a single input channel
- Computation Features:
  - Averaging and low-pass filter functions
  - Reference comparison
  - 2-level threshold comparison
  - Selectable interrupts

Figure 33-1 shows the block diagram of the ADC.

The ADC voltage reference is software selectable to be either internally generated or externally supplied.

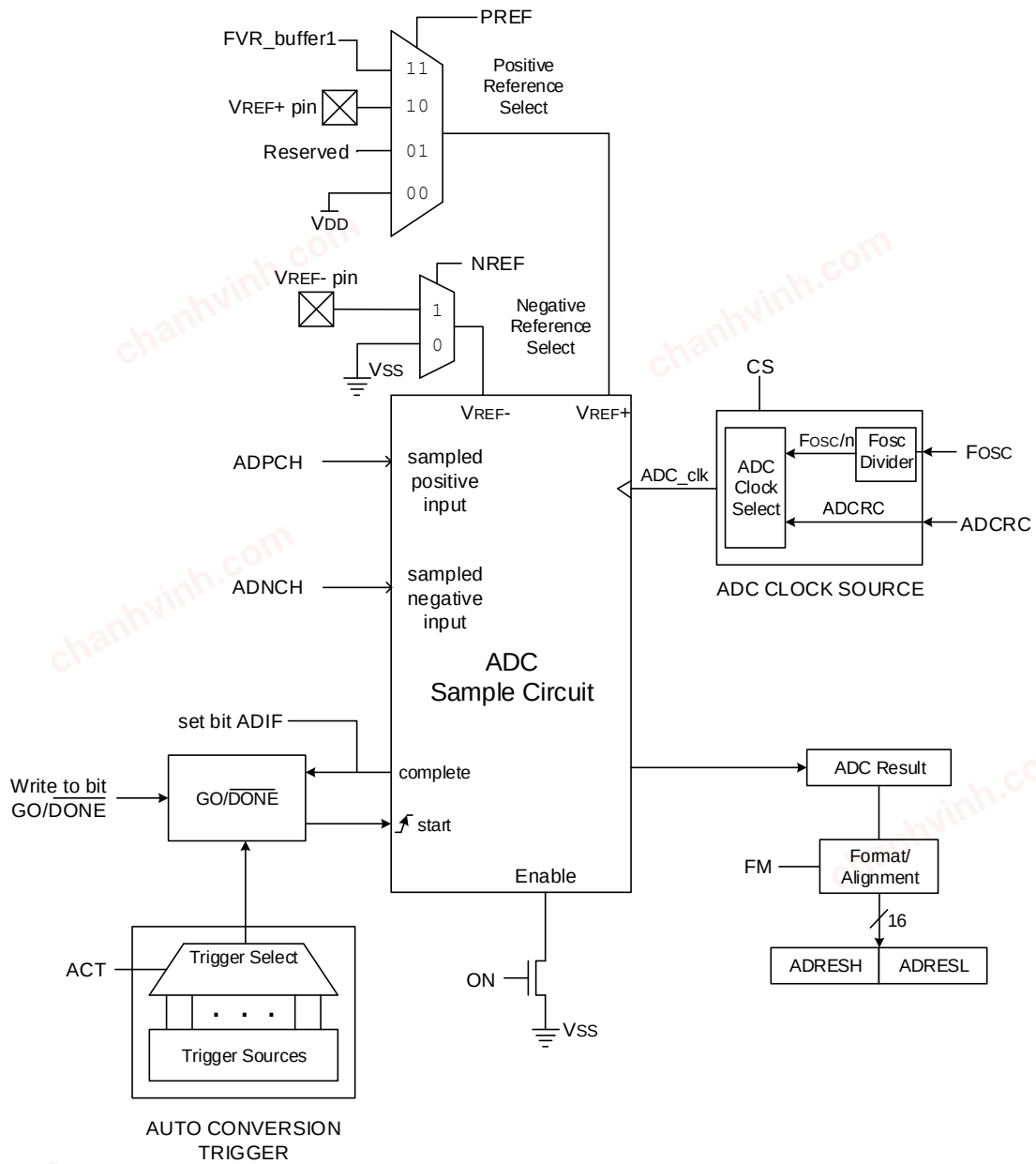
The ADC can generate an interrupt upon completion of a conversion and upon threshold comparison. These interrupts can be used to wake up the device from Sleep.

chanhvinh.com

chanhvinh.com



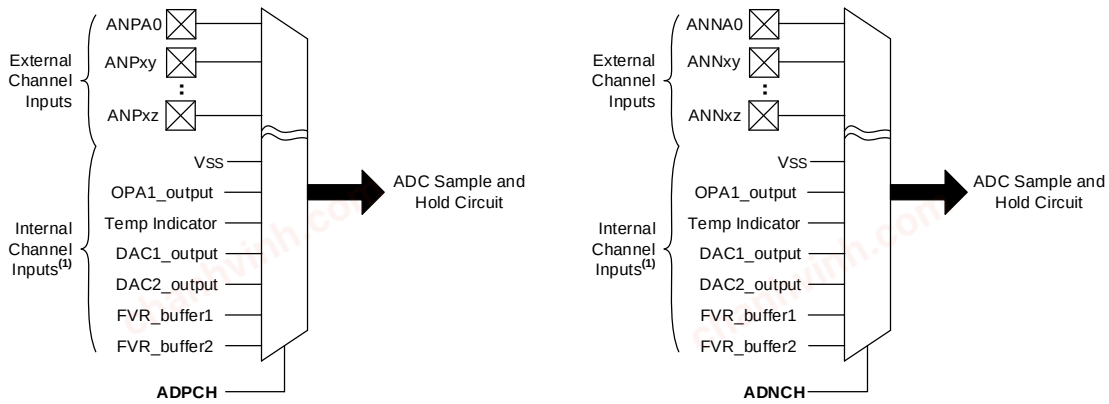
**Figure 33-1. ADC Block Diagram**



chanhvinh.com



**Positive/Negative Input Selection Multiplexers**




**Note 1:** The internal input channel selections vary. The inputs shown are all possible input selections. Please refer to the "**ADC Positive Input Channel Selection**" and "**ADC Negative Input Channel Selection**" tables for device-specific selection options.

## 33.1 ADC Configuration

When configuring the ADC the following functions must be considered:

- ADC Input Configuration
- Port Configuration
- Channel Selection
- ADC Voltage Reference Selection
- ADC Conversion Clock Source
- Interrupt Control
- Result Formatting
- Conversion Trigger Selection
- ADC Acquisition Time
- ADC Precharge Time
- Additional Sample-and-Hold Capacitor
- Single/Double Sample Conversion
- Guard Ring Outputs

### 33.1.1 ADC Input Configuration

The ADC Input Configuration (IC) bit is used to determine whether the ADC uses both positive and negative input channels in differential operations, or only the positive input channel in single-ended operations.

When IC is set (IC = 1), the ADC operates in Differential mode. In Differential mode, the ADC calculates the difference between a positive analog input signal and a negative analog input signal with respect to the ADC reference voltage ($V_{REF}$), and the conversion result may be a positive or negative value. The positive input signal is selected by the ADC Positive Channel Selection (ADPCH) register, and the negative input signal is selected by the ADC Negative Channel Selection (ADNCH) register. The ADC reference voltage ($V_{REF}$) is the difference between the ADC Positive Voltage Reference ($V_{REF}$+) input and the ADC Negative Voltage Reference ($V_{REF}$-) input.

chanhvinh.com



**Equation 33-1. Differential Conversion Result**

$$Conversion\ result = \frac{IN_{ADPCH} - IN_{ADNCH}}{V_{REF}+ - V_{REF}-}$$

When IC is clear (IC = 0), the ADC operates in Single-Ended mode. In Single-Ended mode, the ADC converts the positive input signal with respect to $V_{REF}$, and the conversion result will be a positive value. The positive input is selected by ADPCH, similar to Differential mode; however, in Single-Ended mode, ADC hardware ignores any negative input selections selected by ADNCH. In Single-Ended mode, $V_{REF}$ is the difference between $V_{REF}$**+** and $V_{SS}$.

**Equation 33-2. Single-Ended Conversion Result**

$$Conversion\ result = \frac{IN_{ADPCH}}{V_{REF}+ - V_{SS}}$$

### 33.1.2 Port Configuration

The ADC will convert the voltage level on a pin, whether or not the ANSEL bit is set. When converting analog signals, the I/O pin may be configured for analog by setting the associated TRIS and ANSEL bits. Refer to the **"I/O Ports"** chapter for more information.

**Important:** Analog voltages on any pin defined as a digital input may cause the input buffer to conduct excess current.

### 33.1.3 Channel Selection

The ADPCH and ADNCH registers determine which input channels are connected to the Sample-and-Hold circuit for conversion. When switching channels, it is recommended to have some acquisition time (ADACQ register) before starting the next conversion. Refer to the ADC Operation section for more information.

**Important:** To reduce the chance of measurement error, it is recommended to discharge the Sample-and-Hold capacitor when switching between ADC channels by starting a conversion on a channel connected to $V_{SS}$ and terminating the conversion after the acquisition time has elapsed. If the ADC does not have a dedicated $V_{SS}$ input channel, the $V_{SS}$ selection through the DAC output channel can be used. If the DAC is in use (or the device does not have a DAC), a free input channel can be connected to $V_{SS}$, and can be used in place of the DAC.

#### 33.1.3.1 Channel Grouping

Channel grouping allows multiple, simultaneous input connections to the ADC. The ADC Channel Group Selection (ADCGxp, x = Group number, p = PORT) registers are used to enable each I/O port's analog input channels. A channel group includes all enabled inputs from each of the group's selection registers. All of the group's input signals are wire-OR'd into a single ADC positive input channel, ADCGx, which can be selected by the ADC Positive Input Channel Selection (PCH) bits.

The example below illustrates the configuration of one channel group.

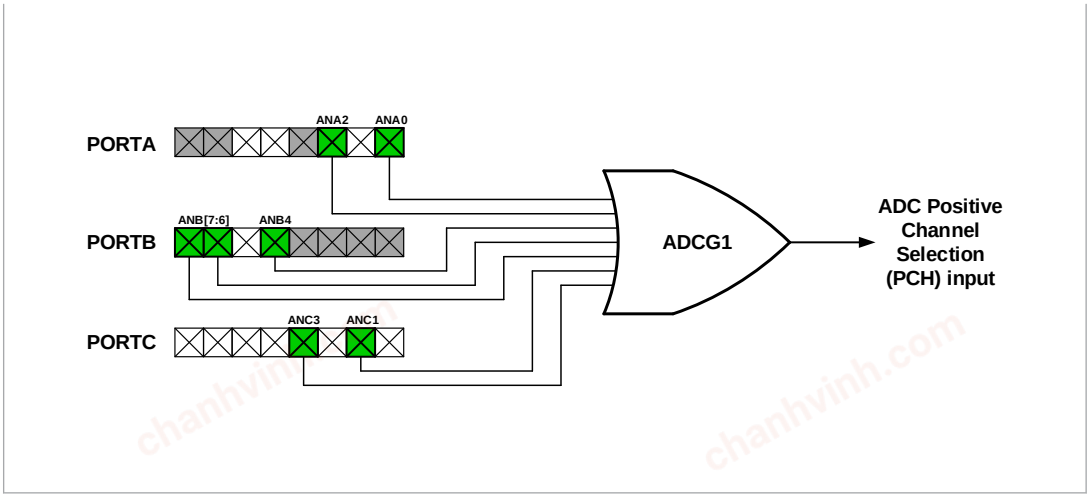
**Example 33-1. ADC Group Example (20-Pin Device)**

```
ADCG1A = 0x05;      // Include ANA0 and ANA2 in Group 1
ADCG1B = 0xB0;      // Include ANB4, ANB6, and ANB7 in Group 1
ADCG1C = 0x0A;      // Include ANC1 and ANC3 in Group 1
```

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com






### 33.1.4 ADC Voltage Reference

The PREF bits provide control of the positive voltage reference. The NREF bit provides control of the negative voltage reference. Refer to the ADREF register for the list of available positive and negative sources.

### 33.1.5 Conversion Clock

The conversion clock source is selected with the CS bit. When CS = 1, the ADC clock source is an internal fixed-frequency clock referred to as ADCRC. When CS = 0, the ADC clock source is derived from $F_{OSC}$.

**Important:** When CS = 0, the clock can be divided using the ADCLK register to meet the ADC clock period requirements.

The time to complete one bit conversion is defined as the $T_{AD}$. Refer to Figure 33-2 for the complete timing details of the ADC conversion.

For correct conversion, the appropriate $T_{AD}$ specification must be met. Refer to the ADC Timing Specifications table in the **"Electrical Specifications"** chapter of the device data sheet for more details. The table below gives examples of appropriate ADC clock selections.

**Table 33-1. ADC Clock Period ($T_{AD}$) vs. Device Operating Frequencies(3,5)**

| ADC Clock Source | ADCLK(2) | ADC Clock Period ($T_{AD}$) for Different Device Frequency ($F_{OSC}$) | | | | | |
|---|---|---|---|---|---|---|---|
| | | 32 MHz(1) | 20 MHz(1) | 16 MHz(1) | 8 MHz | 4 MHz | 1 MHz |
| $F_{OSC}$/2 | 'b000000(2) | 62.5 ns(4) | 100 ns(4) | 125 ns(4) | 250 ns(4) | 500 ns(4) | 2.0 µs |
| $F_{OSC}$/4 | 'b000001 | 125 ns(4) | 200 ns(4) | 250 ns(4) | 500 ns(4) | 1.0 µs | 4.0 µs |
| $F_{OSC}$/6 | 'b000010(2) | 187.5 ns(4) | 300 ns(4) | 375 ns(4) | 750 ns(4) | 1.5 µs | 6.0 µs |
| $F_{OSC}$/8 | 'b000011 | 250 ns(4) | 400 ns(4) | 500 ns(4) | 1.0 µs | 2.0 µs | 8.0 µs |
| ... | ... | ... | ... | ... | ... | ... | ... |
| $F_{OSC}$/16 | 'b000111 | 500 ns(4) | 800 ns(4) | 1.0 µs | 2.0 µs | 4.0 µs | 16.0 µs(4) |
| ... | ... | ... | ... | ... | ... | ... | ... |
| $F_{OSC}$/32 | 'b001111 | 1.0 µs | 1.6 µs | 2.0 µs | 4.0 µs | 8.0 µs | 32.0 µs(4) |
| ... | ... | ... | ... | ... | ... | ... | ... |
| $F_{OSC}$/64 | 'b0111111 | 2.0 µs | 3.2 µs | 4.0 µs | 8.0 µs | 16.0 µs(4) | 64.0 µs(4) |

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



.....continued

| ADC Clock Source | ADCLK(2) | ADC Clock Period ($T_{AD}$) for Different Device Frequency ($F_{OSC}$) | | | | | |
|---|---|---|---|---|---|---|---|
| | | 32 MHz(1) | 20 MHz(1) | 16 MHz(1) | 8 MHz | 4 MHz | 1 MHz |
| ... | ... | ... | ... | ... | ... | ... | ... |
| $F_{OSC}$/128 | 'b111111 | 4.0 µs | 6.4 µs | 8.0 µs | 16.0 µs(4) | 32.0 µs(4) | 128.0 µs(4) |
| ADCRC | CS = 1 | 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 | 1.0-6.0 µs |

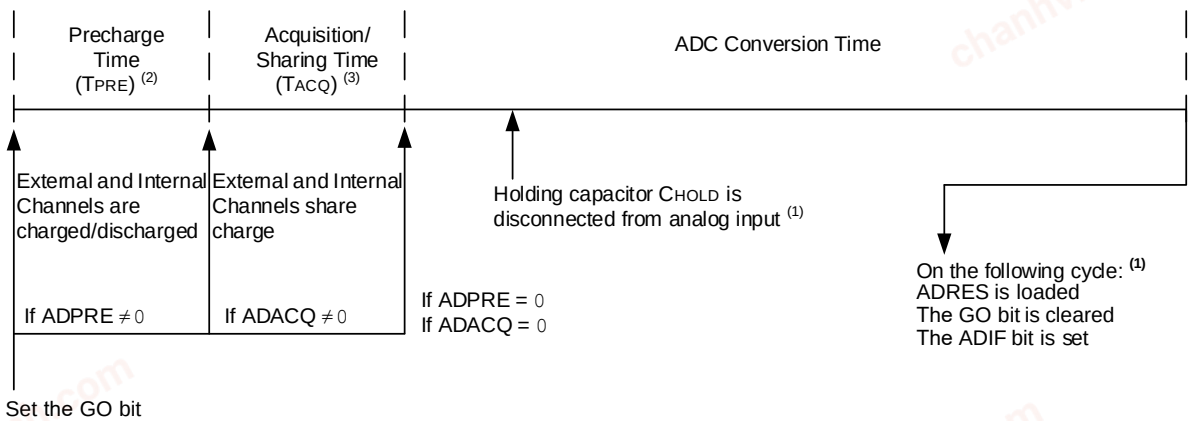
**Notes:**

1. Frequencies above 8 MHz are not recommended when using $F_{OSC}$ as the clock source.
2. For accurate ADC results, loading the ADCLK register with even-numbered values (e.g. 0x00, 0x02, 0x04, etc.) is not recommended.
3. Refer to the **"Electrical Specifications"** chapter of the device data sheet to see the $T_{AD}$ parameter for the ADCRC source typical $T_{AD}$ value.
4. These values violate the required $T_{AD}$ time.
5. The ADC clock period ($T_{AD}$) and total ADC conversion time can be minimized when the ADC clock is derived from the system clock $F_{OSC}$. However, the ADCRC oscillator source must be used when conversions are to be performed with the device in Sleep mode.

**Important:**

- Except for the ADCRC clock source, any changes in the system clock frequency will change the ADC clock frequency, which may adversely affect the ADC result.
- The internal control logic of the ADC runs off of the clock selected by the CS bit. When the CS bit is set to '1' (ADC runs on ADCRC), there may be unexpected delays in operation when setting the ADC control bits.

**Figure 33-2. Analog-to-Digital Conversion Cycles**




**Notes:**

1. Refer to the ADC Conversion Timing Specifications table in the **"Electrical Specifications"** chapter of the device data sheet for more details.
2. Refer to the ADPRE register for more details.
3. Refer to the ADACQ register for more details.

### 33.1.6 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 PIRx register. The ADC interrupt enable is the ADIE bit in the PIEx register. The ADIF bit must be cleared by software.

chanhvinh.com



**Important:**
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 ADCRC oscillator is selected.

The ADC Interrupt can be generated while the device is operating or while in Sleep. While the device is operating in Sleep mode:

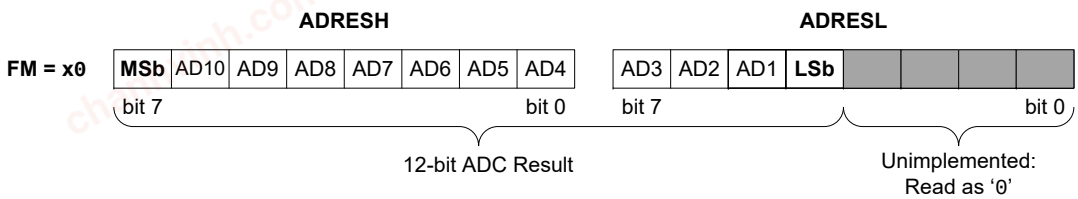
- If ADIE = 1, PEIE = 1, and GIE = 0: An interrupt will wake the device from Sleep. Upon waking from Sleep, the instructions following the SLEEP instruction are executed. The Interrupt Service Routine is not executed.
- If ADIE = 1, PEIE = 1, and GIE = 1: An interrupt will wake the device from Sleep. Upon waking from Sleep, the instruction following the SLEEP instruction is always executed. Then the execution will switch to the Interrupt Service Routine.

### 33.1.7 Result Formatting

The ADC Results Format/Alignment Selection (FM) and the ADC Input Configuration (IC) bits are used to determine the format and alignment of the conversion result.

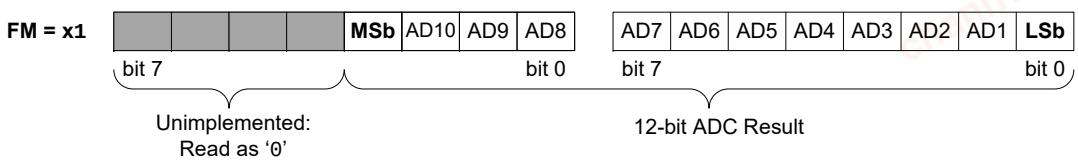
When IC = '0' and FM = 'x0', the conversion result is left justified, as shown in Figure 33-3.

**Figure 33-3. 12-Bit ADC Conversion Result Alignment/Format: IC = 0, FM = x0**



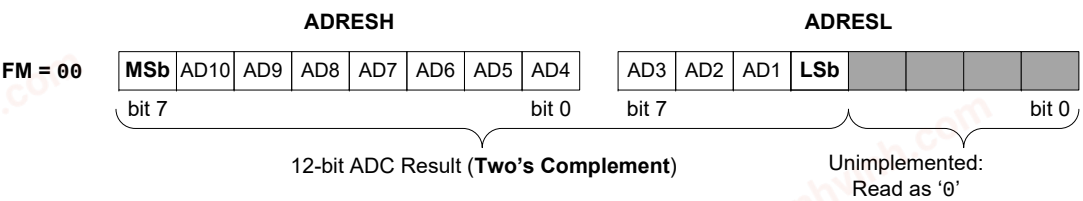

When IC = '0' and FM = 'x1', the conversion result is right justified, as shown in Figure 33-4.

**Figure 33-4. 12-Bit ADC Conversion Result Alignment/Format: IC = 0, FM = x1**




When IC = '1', and FM = '00', the conversion result is in left justified alignment and Two's Complement format, as shown in Figure 33-5.

**Figure 33-5. 12-Bit ADC Conversion Result Alignment/Format: IC = 1, FM = 00**

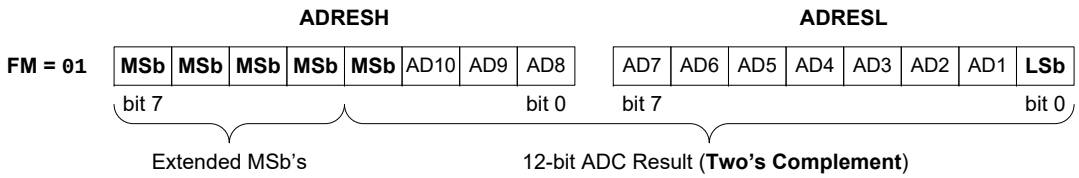



When IC = '1' and FM = '01', the conversion result is in right justified alignment and Two's Complement format, as shown in Figure 33-6.

chanhvinh.com

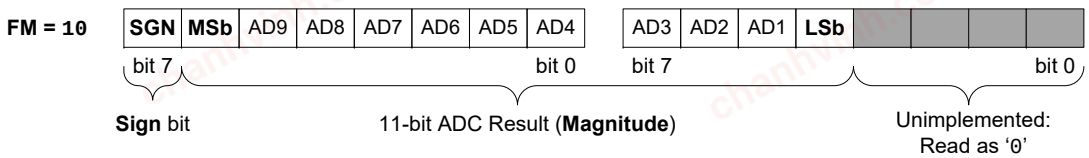


**Figure 33-6. 12-Bit ADC Conversion Result Alignment/Format: IC = 1, FM = 01**



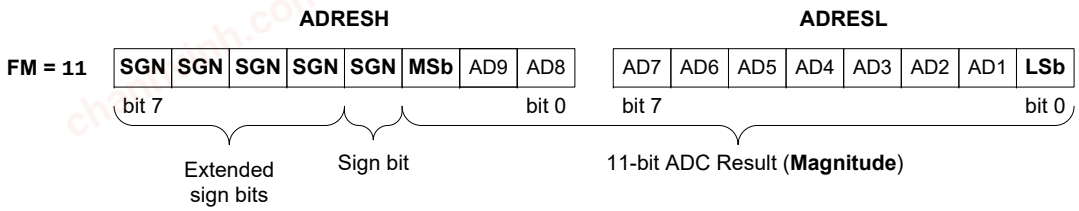

When IC = '1' and FM = '10', the conversion result is in left justified alignment and sign/magnitude format, as shown in Figure 33-7.

**Figure 33-7. 12-Bit ADC Conversion Result Alignment/Format: IC = 1, FM = 10**




When IC = '1' and FM = '11', the conversion result is in right justified alignment and sign/magnitude format, as shown in Figure 33-8.

**Figure 33-8. 12-Bit ADC Conversion Result Alignment/Format: IC = 1, FM = 11**




**Important:** Writes to the ADRES register pair are always right justified, regardless of the selected format mode. Therefore, a data read after writing to ADRES when FM = 00 or 10 will be shifted left three places.

### 33.1.7.1 Sign/Magnitude and Two's Complement Result

When performing differential measurements, the conversion results can be presented in either a sign/magnitude or two's complement format.

In sign/magnitude format, the conversion result is a binary representation of the input, with a sign bit determining the polarity. Equation 33-3 can be used to calculate the full-scale range of ADC result values in sign/magnitude format:

**Equation 33-3. ADC Range in Sign/Magnitude Result Format**

$$n-bit\ range\ in\ sign/magnitude\ format\ =\ \pm 2^{n\ -\ 1}\ -\ 1$$

*where*:

$n =$ *ADC resolution*

*Examples*:

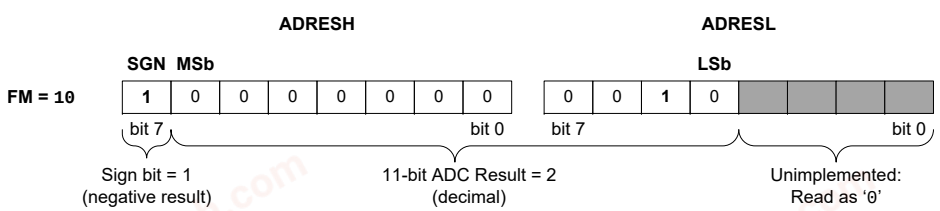
$$12-bit\ ADC\ =\ \pm 2^{12\ -\ 1}\ -\ 1\ =\ \pm 2047$$

In Example 33-2, the right justified ADRES value is '0x8020'. The 11-bit sign/magnitude formatted result shows an absolute value of '2' (decimal), with bit 7 of the ADRESH register representing the sign bit. When the sign bit = '0', the result is positive, and when the sign bit = '1', the result is negative. For this example, the sign bit = '1', so the sign/magnitude formatted decimal result value is '-2'.

chanhvinh.com



**Example 33-2. 12-Bit Sign/Magnitude Result Format**




**ADRES = 0x8020 = -2 (decimal)**

In two's complement format, the MSb of the conversion result determines the polarity. Equation 33-4 can be used to calculate the full-scale range of ADC result values in two's complement format:

**Equation 33-4. ADC Range in Two's Complement Result Format**

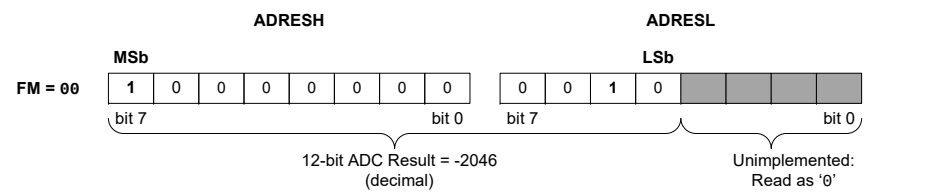
$$n-bit\ range\ in\ two's\ complement\ format\ =\ \left(-2^{n\ -\ 1}\ -\ 1\right),\left(2^{n\ -\ 1}\ -\ 2\right)$$

$where$:
$n\ =\ ADC\ resolution$
$Example$:

$$12-bit\ ADC\ =\ \left(-2^{12\ -\ 1}\ -\ 1\right)\ =\ -2047\left(low\ range\right)$$
$$=\left(2^{12\ -\ 1}\ -\ 2\right)\ =\ 2046\ \left(high\ range\right)$$

In Example 33-3, the right justified ADRES result is '0x8020', which matches the ADRES result in Example 33-2. The 12-bit two's complement result shows a value of '-2046' (decimal), with bit 7 of the ADRESH register representing the MSb of the conversion result.

**Example 33-3. 12-Bit Two's Complement Result Format**




**ADRES = 0x8020 = -2046 (decimal)**

## 33.2 ADC Operation

### 33.2.1 Starting a Conversion

To enable the ADC module, the ON bit must be set to '1'. A conversion may be started by any of the following:

chanhvinh.com



- Software setting the GO bit to '1'
- An external trigger (source selected by ADACT)
- A Continuous-mode retrigger (see the Continuous Sampling Mode section for more details)

**Important:** The GO bit must not be set in the same instruction that turns on the ADC. Refer to the ADC Conversion Procedure (Basic Mode) section for more details.

### 33.2.2 Completion of a Conversion

When any individual conversion is complete, the existing value in ADRES is written into ADPREV (if PSIS = 0) and the new conversion results appear in ADRES. When the conversion completes, the ADC module will:

- Clear the GO bit (unless the CONT bit is set)
- Set the ADIF Interrupt Flag bit
- Set the MATH bit
- Update ADACC

After every conversion when DSEN = 0, or after every other conversion when DSEN = 1, the following events occur:

- ADERR is calculated
- ADC Channel Threshold Interrupt (ADTIF) is set if ADERR calculation meets threshold comparison

### 33.2.3 ADC Operation During Sleep

The ADC module can operate during Sleep. This requires the ADC clock source to be set to the ADCRC option. When the ADCRC oscillator source is selected, the ADC waits one additional instruction before starting the conversion. This allows the SLEEP instruction to be executed, which can reduce system noise during the conversion. If the ADC interrupt is enabled, the device will wake up from Sleep when the conversion completes. If the ADC interrupt is disabled, the device remains in Sleep and the ADC module is turned off after the conversion completes, although the ON bit remains set.

### 33.2.4 External Trigger During Sleep

If the external trigger is received during Sleep while the ADC clock source is set to the ADCRC, the ADC module will perform the conversion and set the ADIF bit upon completion.

If an external trigger is received when the ADC clock source is something other than ADCRC, the trigger will be recorded, but the conversion will not begin until the device exits Sleep.

### 33.2.5 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 bit is set by hardware.

The auto-conversion trigger source is selected with the ACT bits.

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

### 33.2.6 ADC Conversion Procedure (Basic Mode)

This is an example procedure for using the ADC to perform an Analog-to-Digital Conversion:

1. Configure Port:
   a. Disable pin output driver (refer to the TRISx register)
   b. Configure pin as analog (refer to the ANSELx register)
2. Configure the ADC module:
   a. Select ADC conversion clock
   b. Configure voltage reference
   c. Select ADC input channel

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



d. Configure precharge (ADPRE) and acquisition (ADACQ) time period
e. Turn on ADC module

3. Configure ADC interrupt (optional):
   a. Clear ADC interrupt flag
   b. Enable ADC interrupt
   c. Enable global interrupt (GIE bit)(1)
4. If ADACQ != 0, software must wait the required acquisition time(2).
5. Start conversion by setting the GO bit.
6. Wait for ADC conversion to complete by one of the following:
   – Polling the GO bit
   – Waiting for the ADC interrupt (if interrupt is enabled)
7. Read ADC Result.
8. Clear the ADC interrupt flag (if interrupt is enabled).

**Notes:**

1. With global interrupts disabled (GIE = 0), the device will wake from Sleep, but will not enter an Interrupt Service Routine.
2. Refer to the ADC Acquisition Requirements section for more details.

**Example 33-4. ADC Conversion (Differential Inputs)**

```
/*This code block configures the ADC
         for polling, VDD and VSS references, ADCRC oscillator.
         Conversion start & polling for completion
         are included.
 */
   void main()
{
   initializeSystem();     //System Initialize

   //Setup ADC
   ADCON0bits.FM = 10;     //Right justify, sign/magnitude
   ADCON0bits.CS = 1;      //ADCRC Clock
   ADCON0bits.IC = 1;      //Differential mode
   ADPCH = 0x00;           //RA0 is positive input
   ADNCH = 0x01;           //RA1 is negative input
   TRISAbits.TRISA0 = 1;   //Set RA0 to input
   TRISAbits.TRISA1 = 1;   //Set RA1 to input
   ANSELAbits.ANSELA0 = 1; //Set RA0 to analog
   ANSELAbits.ANSELA1 = 1; //Set RA1 to analog
   ADACQ = 32;             //Set acquitisition time
   ADCON0bits.ON = 1;      //Turn ADC On

   while (1)
   {
      ADCON0bits.GO = 1;     //Start conversion
      while (ADCON0bits.GO); //Wait for conversion done
      resultHigh = ADRESH;   //Read result
      resultLow = ADRESL;    //Read result
   }
}
```

## 33.3 ADC Acquisition Requirements

For the ADC to meet its specified accuracy, the charge holding capacitor ($C_{HOLD}$) must be allowed to fully charge to the input channel voltage level. The analog input model is shown in Figure 33-9. The source impedance ($R_S$) and the internal sampling switch ($R_{SS}$) impedance directly affect the time required to charge the capacitor $C_{HOLD}$. The sampling switch ($R_{SS}$) impedance varies over the device voltage ($V_{DD}$). 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 time must be completed before the conversion can

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



be started. To calculate the minimum acquisition time, Equation 33-5 may be used. This equation assumes an error of 1/2 LSb. The 1/2 LSb error is the maximum error allowed for the ADC to meet its specified resolution.

**Equation 33-5. Acquisition Time Example**

Assumptions: Temperature = 50°C; External impedance = 10 kΩ; $V_{DD}$ = 5.0V

$T_{ACQ}$ = Amplifier Settling Time + Hold Capacitor Charging Time + Temperature Coefficient

$$T_{ACQ} = T_{AMP} + T_C + T_{COFF}$$

$$T_{ACQ} = 2\,\mu s + T_C + \left[\left(Temperature - 25°C\right)\left(0.05\,\mu s/°C\right)\right]$$

The value for $T_C$ can be approximated with the following equations:

$$V_{APPLIED}\left(1 - \frac{1}{\left(2^{n+1}\right) - 1}\right) = V_{CHOLD}$$ ; [1] $V_{CHOLD}$ charged to within ½ LSb

$$V_{APPLIED}\left(1 - e^{\frac{-T_C}{RC}}\right) = V_{CHOLD}$$ ; [2] $V_{CHOLD}$ charge response to $V_{APPLIED}$

$$V_{APPLIED}\left(1 - e^{\frac{-T_C}{RC}}\right) = V_{APPLIED}\left(1 - \frac{1}{\left(2^{n+1}\right) - 1}\right)$$ ; Combining [1] and [2]

**Note:** Where n = ADC resolution in bits

Solving for $T_C$:

$$T_C = -C_{HOLD}(R_{IC} + R_{SS} + R_S)\ ln\ (1/8191)$$

$$T_C = -28\,pF(1\,k\Omega + 7\,k\Omega + 10\,k\Omega)\ ln\ (0.0001221)$$

$$T_C = 4.54\ \mu s$$

Therefore:

$$T_{ACQ} = 2\ \mu s + 4.54\ \mu s + \left[\left(50°C - 25°C\right)\left(0.05\,\mu s/°C\right)\right]$$

$$T_{ACQ} = 7.79\,\mu s$$

**Important:**

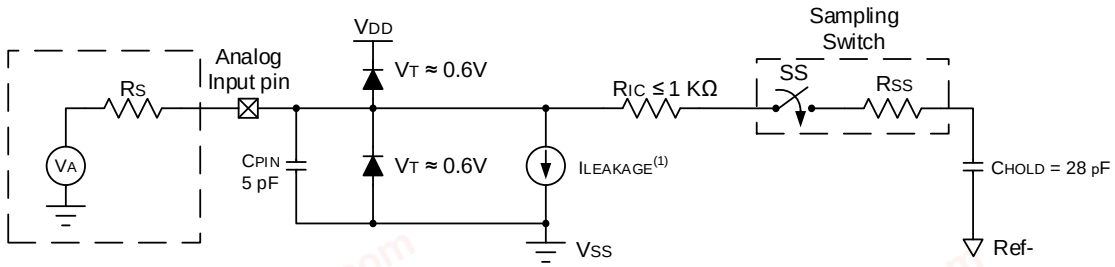
- The reference voltage ($V_{REF}$) has no effect on the equation, since it cancels itself out
- The charge holding capacitor ($C_{HOLD}$) is not discharged after each conversion
- The maximum recommended impedance for analog sources is 10 kΩ. This is required to meet the pin leakage specification.

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com

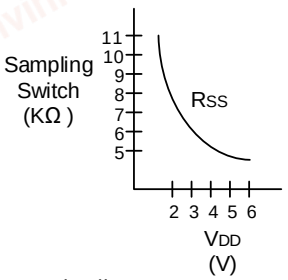


**Figure 33-9. Analog Input Model**



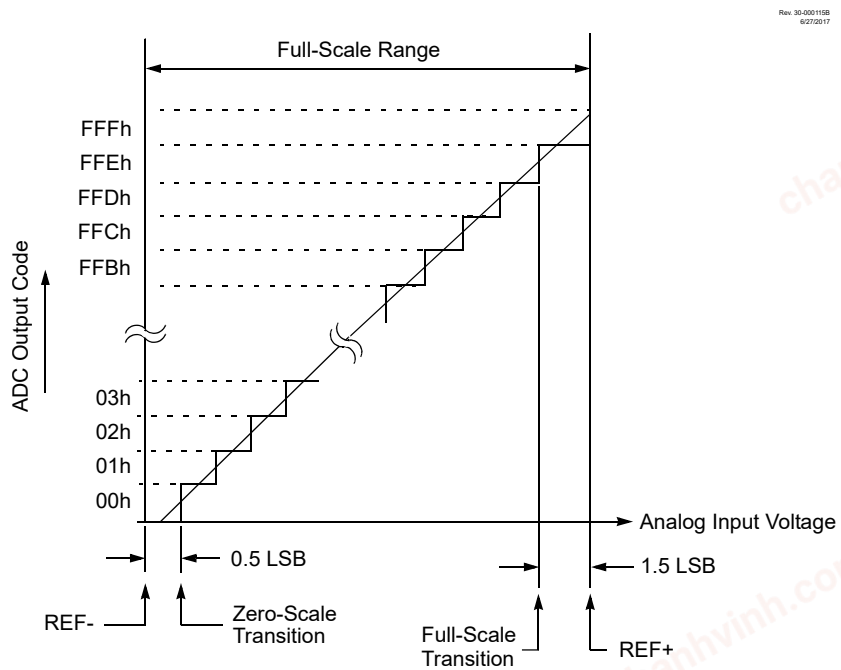

**Legend:**

| | | |
|---|---|---|
| CPIN | = | Input Capacitance |
| ILEAKAGE | = | Leakage Current at the pin due to various junctions |
| RIC | = | Interconnect Resistance |
| RS | = | Source Impedance |
| VA | = | Analog Voltage |
| VT | = | Diode Forward Voltage |
| SS | = | Sampling Switch |
| RSS | = | Resistance of the Sampling Switch |
| CHOLD | = | Sample/Hold Capacitance |




**Note:**
1. Refer to the **"Electrical Specifications"** chapter of the device data sheet for more details.

**Figure 33-10. ADC Transfer Function**



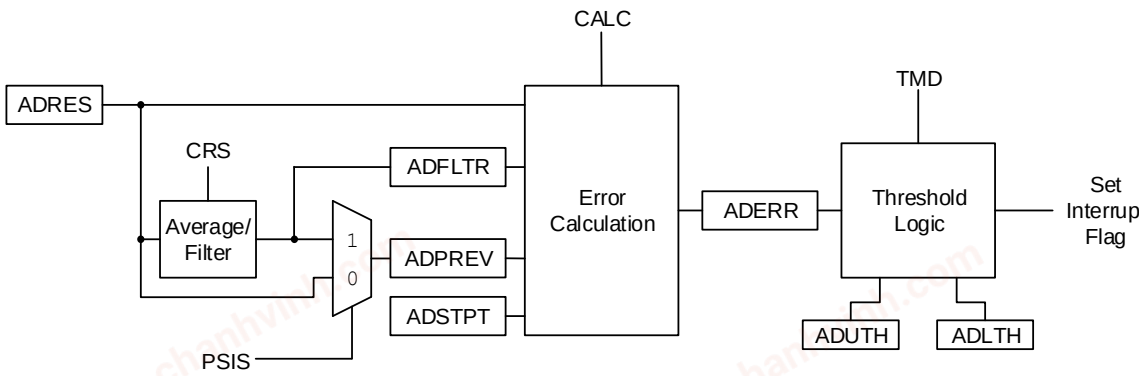

## 33.4 Computation Operation

The ADC module hardware is equipped with post-conversion computation features. These features provide post-processing functions such as digital filtering/averaging and threshold comparison. Based on computation results, the module can be configured to take additional samples or stop conversions and an interrupt may be asserted.

chanhvinh.com



**Figure 33-11. Computational Features Simplified Block Diagram**




The operation of the ADC computational features is controlled by the MD bits.

The module can be operated in one of five modes:

- **Basic:** This is a Legacy mode. In this mode, ADC conversion occurs on single (DSEN = 0) or double (DSEN = 1) samples. ADIF is set after each conversion is complete. ADTIF is set according to the Calculation mode.
- **Accumulate:** With each trigger, the ADC conversion result is added to the accumulator and ADCNT increments. ADIF is set after each conversion. ADTIF is set according to the Calculation mode.
- **Average:** With each trigger, the ADC conversion result is added to the accumulator. When the RPT number of samples have been accumulated, a threshold test is performed. Upon the next trigger, the accumulator is cleared. For the subsequent tests, additional RPT samples are required to be accumulated.
- **Burst Average:** At the trigger, the accumulator is cleared. The ADC conversion results are then collected repetitively until RPT samples are accumulated and finally the threshold is tested.
- **Low-Pass Filter (LPF):** With each trigger, the ADC conversion result is sent through a filter. When RPT samples have occurred, a threshold test is performed. Every trigger after that, the ADC conversion result is sent through the filter and another threshold test is performed.

The five modes are summarized in the following table.

chanhvinh.com



**Table 33-2. Computation Modes**

| Mode | MD | Register Clear Event | Value after Cycle(1) Completion | | Threshold Operations | | | Value at ADTIF Interrupt | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | ADACC and CNT | ADACC | ADCNT | Retrigger | Threshold Test | Interrupt | AOV | ADFLTR | ADCNT |
| Basic | 0 | ACLR = 1 | Unchanged | Unchanged | No | Every Sample | If threshold=true | N/A | N/A | count |
| Accumulate | 1 | ACLR = 1 | S1 + ADACC or (S2-S1)(2) + ADACC | If (ADCNT = 0xFF): ADCNT, otherwise: ADCNT+1 | No | Every Sample | If threshold=true | ADACC Overflow | ADACC/$2^{CRS}$ | count |
| Average | 2 | ACLR = 1 or ADCNT ≥ ADRPT at GO set or retrigger | S1 + ADACC or (S2-S1) + ADACC | If (ADCNT = 0xFF): ADCNT, otherwise: ADCNT+1 | No | If ADCNT ≥ ADRPT | If threshold=true | ADACC Overflow | ADACC/$2^{CRS}$ | count |
| Burst Average | 3 | ACLR = 1 or at GO set or retrigger | Each repetition: same as Average End with sum of all samples | Each repetition: same as Average End with ADCNT = ADRPT | Repeat while ADCNT < ADRPT | If ADCNT ≥ ADRPT | If threshold=true | ADACC Overflow | ADACC/$2^{CRS}$ | ADRPT |
| Low-pass Filter | 4 | ACLR = 1 | S1 + ADACC-ADACC/ $2^{CRS}$ or (S2-S1) + ADACC-ADACC/ $2^{CRS}$ | If (ADCNT = 0xFF): ADCNT, otherwise: ADCNT+1 | No | If ADCNT ≥ ADRPT | If threshold=true | ADACC Overflow | ADACC/$2^{CRS}$ (Filtered Value) | count |

**Notes:**

1. When DSEN = 0, Cycle means one conversion. When DSEN = 1, Cycle means two conversions.
2. S1 and S2 are abbreviations for Sample 1 and Sample 2, respectively. When DSEN = 0, S1 = ADRES; when DSEN = 1, S1 = ADPREV and S2 = ADRES.

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



### 33.4.1 Digital Filter/Average

The digital filter/average module consists of an accumulator with data feedback options, and control logic to determine when threshold tests need to be applied. The accumulator can be accessed through the ADACC register.

Upon each trigger event (the GO bit set or external event trigger), the ADC conversion result is added to or subtracted from the accumulator. If the accumulated value exceeds $2^{(accumulator_width)}$-1 = $2^{18}$-1 = 262143, the AOV overflow bit is set.

The number of samples to be accumulated is determined by the ADRPT (ADC Repeat Setting) register. Each time a sample is added to the accumulator, the ADCNT register is incremented. Once the ADRPT samples are accumulated (ADCNT = ADRPT), the accumulator may be cleared automatically depending on ADC Operation mode. An accumulator clear command can be issued in software by setting the ACLR bit. Setting the ACLR bit will also clear the AOV (Accumulator Overflow) bit, as well as the ADCNT register. The ACLR bit is cleared by the hardware when accumulator clearing action is complete.

**Important:** When ADC is operating from ADCRC, up to five ADCRC clock cycles are required to execute the ADACC clearing operation.

The CRS bits control the data shift on the accumulator result, which effectively divides the value in the accumulator registers. For the Accumulate mode of the digital filter, the shift provides a simple scaling operation. For the Average/ Burst Average mode, the calculated average is only accurate when the number of samples agrees with the number of bits shifted. For the Low-Pass Filter mode, the shift is an integral part of the filter, and determines the cutoff frequency of the filter. Table 33-3 shows the -3 dB cutoff frequency in ωT (radians) and the highest signal attenuation obtained by this filter at Nyquist frequency (ωT = π).

**Table 33-3. Low-Pass Filter -3 dB Cutoff Frequency**

| CRS | ωT (radians) @ -3 dB Frequency | dB @ $F_{Nyquist}$ = 1/(2T) |
|---|---|---|
| 1 | 0.72 | -9.5 |
| 2 | 0.284 | -16.9 |
| 3 | 0.134 | -23.5 |
| 4 | 0.065 | -29.8 |
| 5 | 0.032 | -36.0 |
| 6 | 0.016 | -42.0 |

### 33.4.2 Basic Mode

Basic mode (MD = 'b000) disables all additional computation features. In this mode, no accumulation occurs but threshold error comparison is performed. Double sampling, Continuous mode, and all CVD features are still available, but no digital filter/average calculations are performed.

### 33.4.3 Accumulate Mode

In Accumulate mode (MD = 'b001), after every conversion, the ADC result is added to the ADACC register. The ADACC register is right-shifted by the value of the CRS bits. This right-shifted value is copied into the ADFLTR register. The Formatting mode does not affect the right-justification of the ADACC or ADFLTR values. Upon each sample, ADCNT is incremented, counting the number of samples accumulated. After each sample and accumulation, the ADFLTR value has a threshold comparison performed on it (see the Threshold Comparison section) and the ADTIF interrupt may trigger.

### 33.4.4 Average Mode

In Average mode (MD = 'b010), the ADACC registers accumulate with each ADC sample, much as in Accumulate mode, and the ADCNT register increments with each sample. The ADFLTR register is also updated with the right-shifted value of the ADACC register. The value of the CRS bits governs the number of right shifts. However, in Average mode, the threshold comparison is performed upon ADCNT being greater than or equal to a user-defined

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



ADRPT value. In this mode, when ADRPT = $2^{CRS}$, the final accumulated value will be divided by the number of samples, allowing for a threshold comparison operation on the average of all gathered samples.

### 33.4.5 Burst Average Mode

The Burst Average mode (MD = 'b011) acts the same as the Average mode in most respects. The one way it differs is that it continuously retriggers ADC sampling until the CNT value is equal to RPT, even if Continuous Sampling mode (see Continuous Sampling Mode) is not enabled. This provides a threshold comparison on the average of a short burst of ADC samples.

### 33.4.6 Low-Pass Filter Mode

The Low-Pass Filter mode (MD = 'b100) acts similarly to the Average mode in how it handles samples; it accumulates samples until the CNT value is greater than or equal to RPT, then triggers a threshold comparison. But, instead of a simple average, it performs a low-pass filter operation on all of the samples, reducing the effect of high-frequency noise on the total, then performs a threshold comparison on the results. In this mode, the CRS bits determine the cutoff frequency of the low-pass filter (as demonstrated by the Digital Filter/Average section). Refer to the Computation Operation section for a more detailed description of the mathematical operation.

For more information about Low-Pass Filter mode, refer to the following Microchip application note, available in the corporate website (www.microchip.com):

- AN2749, *"PIC18 12-bit ADCC in Low-Pass Filter Mode"*

### 33.4.7 Threshold Comparison

At the end of each computation:

- The conversion results are captured at the end-of-conversion.
- The error (ADERR) is calculated based on a difference calculation which is selected by the CALC bits. The value can be one of the following calculations:
  - The first derivative of single measurements
  - The CVD result when double sampling is enabled
  - The current result vs. setpoint value in the ADSTPT register
  - The current result vs. the filtered/average result
  - The first derivative of the filtered/average value
  - Filtered/average value vs. setpoint value in the ADSTPT register
- The result of the calculation (ADERR) is compared to the upper and lower thresholds, ADUTH and ADLTH registers, to set the UTHR and LTHR Status bits. The threshold logic is selected by the TMD bits. The threshold trigger option can be one of the following:
  - Never interrupt
  - Error is less than lower threshold
  - Error is greater than or equal to lower threshold
  - Error is between thresholds (inclusive)
  - Error is outside of thresholds
  - Error is less than or equal to upper threshold
  - Error is greater than upper threshold
  - Always interrupt regardless of threshold test results
  - If the Threshold condition is met, the channel threshold interrupt flag ADTIF is set.

**Important:**

- The threshold tests are signed operations.
- If the AOV bit is set, a threshold interrupt is signaled. It is good practice for threshold interrupt handlers to verify the validity of the threshold by checking the AOV bit.

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



### 33.4.8 Repetition and Sampling Options

#### 33.4.8.1 Continuous Sampling Mode

Setting the CONT bit automatically retriggers a new conversion cycle after updating the ADACC register. That means the GO bit remains set to generate automatic retriggering. If SOI = 1, a Threshold Interrupt condition will clear the GO bit and the conversion will stop.

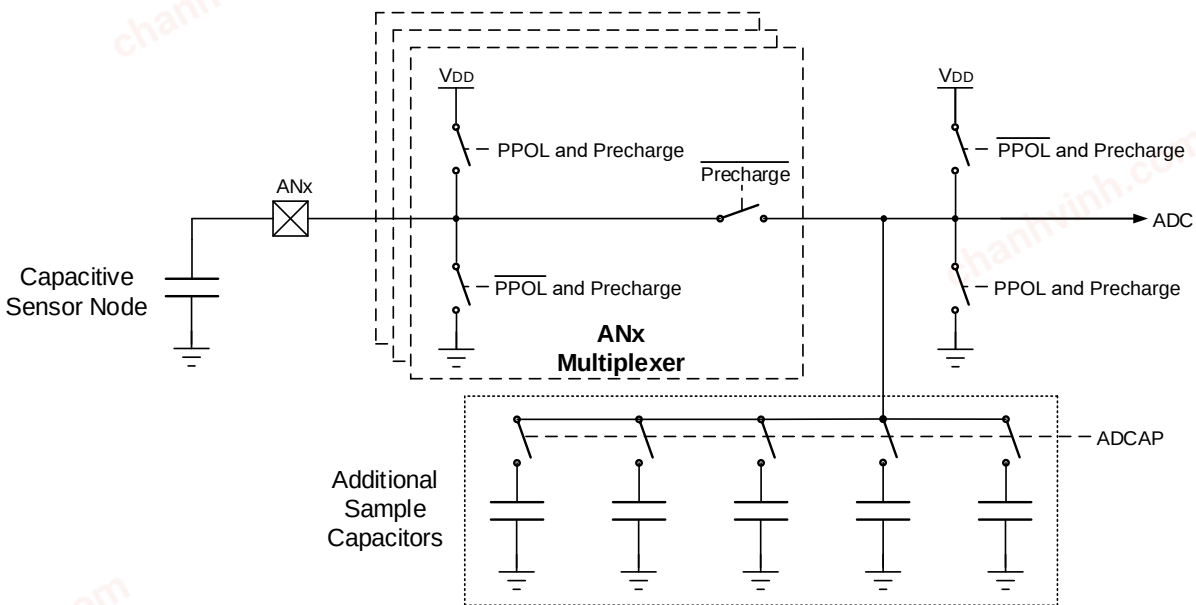
#### 33.4.8.2 Double Sample Conversion

Double sampling is enabled by setting the DSEN bit. When this bit is set, two conversions are required before the module calculates the threshold error. Each conversion must be triggered separately when CONT = 0, but will repeat automatically form a single trigger when CONT = 1. The first conversion will set the MATH bit and update the ADACC register, but will not calculate ADERR or trigger ADTIF. When the second conversion completes, the first value is transferred to ADPREV (depending on the setting of PSIS) and the value of the second conversion is placed into ADRES. Only upon the completion of the second conversion is ADERR calculated and ADTIF triggered (depending on the value of CALC).

### 33.4.9 Capacitive Voltage Divider (CVD) Features

The ADC module contains several features that allow the user to perform a relative capacitance measurement on any ADC channel using the internal ADC Sample-and-Hold capacitance as a reference. This relative capacitance measurement can be used to implement capacitive touch or proximity sensing applications. The following figure shows the basic block diagram of the CVD portion of the ADC module.

**Figure 33-12. Hardware Capacitive Voltage Divider Block Diagram**




This is an example to configure ADC for CVD operation:

1. Configure Port:
   a. Disable pin output driver (refer to the TRISx register)
   b. Configure pin as analog (refer to the ANSELx register)
2. Configure the ADC module:
   a. Select ADC conversion clock
   b. Configure voltage reference
   c. Select ADC input channel
   d. Configure precharge (ADPRE) and acquisition (ADACQ) time period
   e. Select precharge polarity (PPOL)
   f. Enable Double Sampling (DSEN)

© 2023 Microchip Technology Inc.
and its subsidiaries

chanhvinh.com



g. Turn on ADC module

3. Configure ADC interrupt (optional):
   a. Clear ADC interrupt flag
   b. Enable ADC interrupt
   c. Enable global interrupt (GIE bit)(1)
4. Start double sample conversion by setting the GO bit.
5. Wait for ADC conversion to complete by one of the following:
   – Polling the GO bit
   – Waiting for the ADC interrupt (if interrupt is enabled)
6. Second ADC conversion depends on the state of CONT:
   a. If CONT = 1, both conversion will repeat automatically form a single trigger.
   b. If CONT = 0, each conversion must be triggered separately.
7. ADERR register contains the CVD result
8. Clear the ADC interrupt flag (if interrupt is enabled).

**Note:**

1. With global interrupts disabled (GIE = 0), the device will wake from Sleep, but will not enter an Interrupt Service Routine.

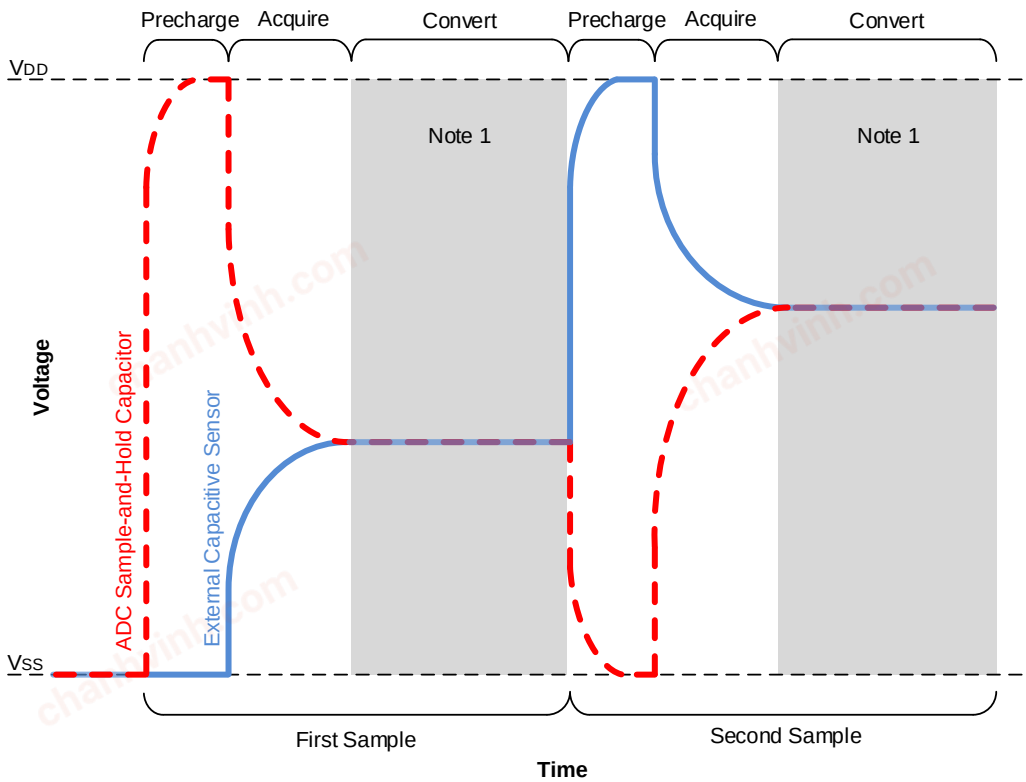
#### 33.4.9.1 CVD Operation

A CVD operation begins with the ADC's internal Sample-and-Hold capacitor ($C_{HOLD}$) being disconnected from the path, which connects it to the external capacitive sensor node. While disconnected, $C_{HOLD}$ is precharged to $V_{DD}$ or discharged to $V_{SS}$. If the PCSC bit is clear, the sensor node is either discharged or charged to $V_{SS}$ or $V_{DD}$, respectively to the opposite level of $C_{HOLD}$. If PCSC is set, the external capacitive sensor node receives no precharge. When the precharge phase is complete, the $V_{DD}/V_{SS}$ bias paths for the two nodes are disconnected and the paths between $C_{HOLD}$ and the external sensor node is reconnected, at which time the acquisition phase of the CVD operation begins. During acquisition, a capacitive voltage divider is formed between the precharged $C_{HOLD}$ and sensor nodes, which results in a final voltage level setting on $C_{HOLD}$, which is determined by the capacitances and precharge levels of the two nodes. After acquisition, the ADC converts the voltage level on $C_{HOLD}$. This process is then repeated with the selected precharge levels inverted for both the $C_{HOLD}$ and the sensor nodes. The waveform for two CVD measurements, which is known as differential CVD measurement, is shown in the following figure.

chanhvinh.com



**Figure 33-13. Differential CVD Measurement Waveform**




**Note 1:** External Capacitive Sensor voltage during the conversion phase may vary as per the configuration of the corresponding pin.

### 33.4.9.2 Precharge Control

The precharge stage is the period of time that brings the external channel and internal Sample-and-Hold capacitor to known voltage levels. Precharge is enabled by writing a nonzero value to the ADPRE register. This stage is initiated when an ADC conversion begins, either from setting the GO bit, a Special Event Trigger, or a conversion restart from the computation functionality. If the ADPRE register is cleared when an ADC conversion begins, this stage is skipped.

The Precharge Sample Capacitor Only (PCSC) bit can be used to disable the precharge stage to the external channel.

During the precharge time, $C_{HOLD}$ is disconnected from the outer portion of the sample path that leads to the external capacitive sensor and is connected to either $V_{DD}$ or $V_{SS}$, depending on the value of the PPOL bit. At the same time, when PCSC is clear (PCSC = 0), the port pin logic of the selected analog channel is overridden to drive a digital high or low out, to precharge the outer portion of the ADC's sample path, which includes the external sensor. The output polarity of this override is determined by the PPOL bit such that the external sensor cap is charged opposite that of the internal $C_{HOLD}$ cap. If PCSC is set (PCSC = 1), the outer portion of the ADC's sample path is disconnected, preventing the precharge from occurring on the external channel. The amount of time for precharge is controlled by the ADPRE register.

**Important:** The external charging overrides the TRIS/LAT/Guard outputs setting of the respective I/O pin. If there is a device attached to this pin, the PCSC bit will be set, or precharge will not be used.

chanhvinh.com



### 33.4.9.3 Acquisition Control for CVD (ADPRE > 0)

The acquisition stage allows time for the voltage on the internal Sample-and-Hold capacitor to charge or discharge from the selected analog channel. This acquisition time is controlled by the ADACQ register. The acquisition stage begins when precharge stage ends.

At the start of the acquisition stage, the port pin logic of the selected analog channel is overridden to turn off the digital high/low output drivers so they do not affect the final result of the charge averaging. Also, the selected ADC channel is connected to $C_{HOLD}$. This allows charge averaging to proceed between the precharged channel and the $C_{HOLD}$ capacitor.

**Important:** When ADPRE > 0, setting ADACQ to '0' will set a maximum acquisition time. When precharge is disabled, setting ADACQ to '0' will disable hardware acquisition time control.

### 33.4.9.4 Guard Ring Outputs

Figure 33-14 shows a typical guard ring circuit. $C_{GUARD}$ represents the capacitance of the guard ring trace placed on the PCB. The user selects values for $R_A$ and $R_B$ that will create a voltage profile on $C_{GUARD}$, which will match the selected acquisition channel.

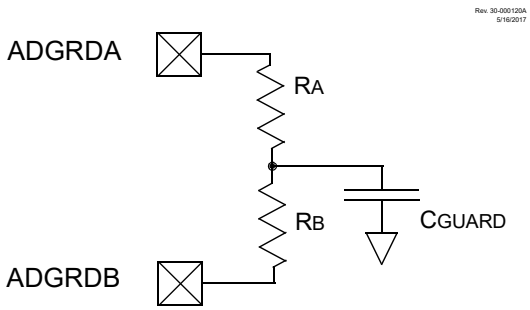
The purpose of the guard ring is to generate a signal in phase with the CVD sensing signal to minimize the effects of the parasitic capacitance on sensing electrodes. It also can be used as a mutual drive for mutual capacitive sensing. For more information about active guard and mutual drive, refer to the following Microchip application note, available on the corporate website (www.microchip.com):

- AN1478, *"mTouch™ Sensing Solution Acquisition Methods Capacitive Voltage Divider"*

The ADC has two guard ring drive outputs, ADGRDA and ADGRDB. These outputs are routed through PPS controls to I/O pins. Refer to the **"Peripheral Pin Select (PPS) Module"** chapter for more details. The polarity of these outputs is controlled by the GPOL and IPEN bits.

At the start of the first precharge stage, both outputs are set to match the GPOL bit. Once the acquisition stage begins, ADGRDA changes polarity, while ADGRDB remains unchanged. When performing a double sample conversion, setting the IPEN bit causes both guard ring outputs to transition to the opposite polarity of GPOL at the start of the second precharge stage, and ADGRDA toggles again for the second acquisition. For more information on the timing of the guard ring output, refer to Figure 33-15.

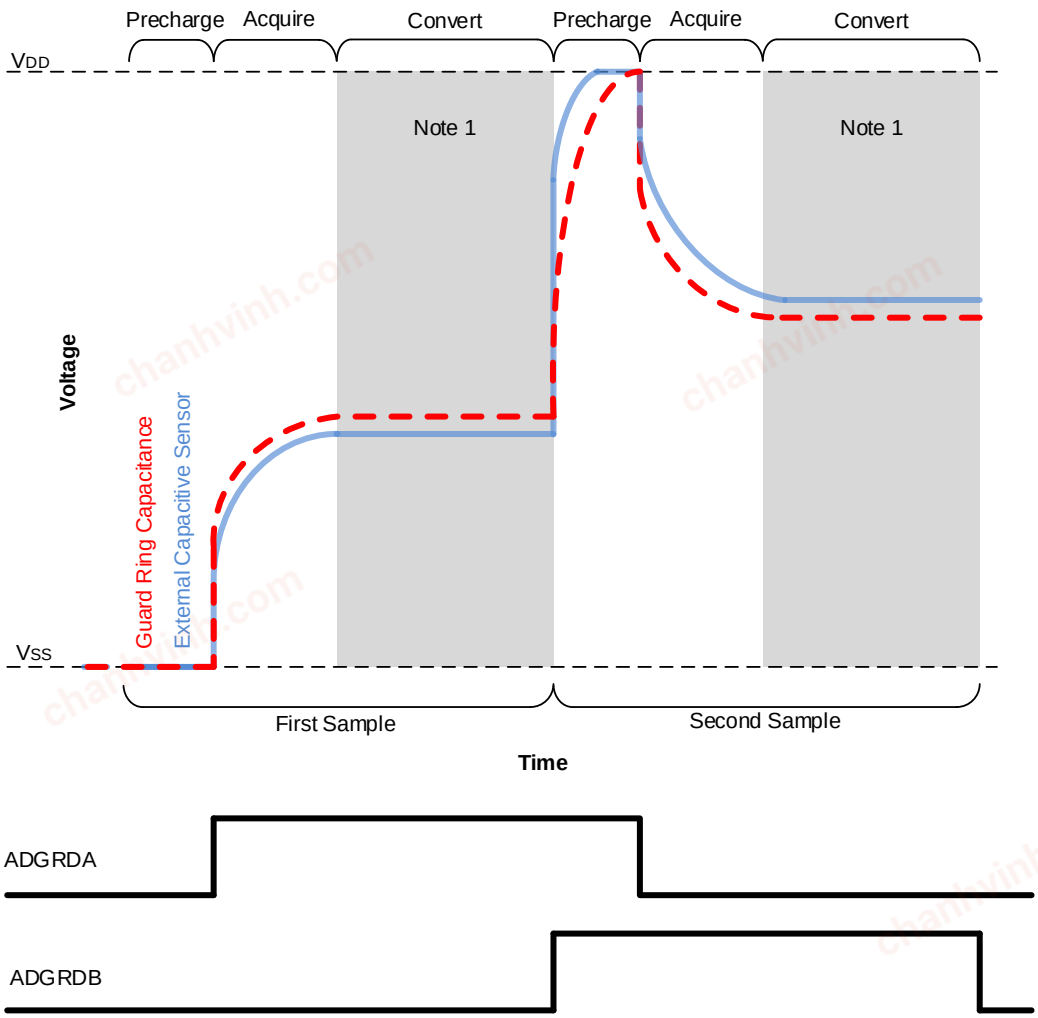
**Figure 33-14. Guard Ring Circuit**



chanhvinh.com



**Figure 33-15. Differential CVD with Guard Ring Output Waveform**




**Note 1:** External Capacitive Sensor voltage during the conversion phase may vary as per the configuration of the corresponding pin.

#### 33.4.9.5 Additional Sample-and-Hold Capacitance

Additional capacitance can be added in parallel with the internal Sample-and-Hold capacitor ($C_{HOLD}$) by using the ADCAP register. This register selects a digitally programmable capacitance that is added to the ADC conversion bus, increasing the effective internal capacitance of the Sample-and-Hold capacitor in the ADC module. This is used to improve the match between internal and external capacitance for a better sensing performance. The additional capacitance does not affect analog performance of the ADC because it is not connected during conversion.

## 33.5 Register Definitions: ADC Control

Long bit name prefixes for the ADC peripherals are shown in the following table. Refer to the **"Long Bit Names"** section in the **"Register and Bit Naming Conventions"** chapter for more information.

**Table 33-4. ADC Long Bit Name Prefixes**

| Peripheral | Bit Name Prefix |
|---|---|
| ADC | AD |

chanhvinh.com



### 33.5.1 ADCON0

**Name:** ADCON0
**Offset:** 0x1D26

ADC Control Register 0

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ON | CONT | | CS | FM[1:0] | | IC | GO |
| Access | R/W | R/W | | R/W | R/W | R/W | R/W | R/W/HC/HS |
| Reset | 0 | 0 | | 0 | 0 | 0 | 1 | 0 |

**Bit 7 – ON** ADC Enable

| Value | Description |
|---|---|
| 1 | ADC is enabled |
| 0 | ADC is disabled |

**Bit 6 – CONT** ADC Continuous Operation Enable

| Value | Description |
|---|---|
| 1 | GO is retriggered upon completion of each conversion trigger until ADTIF is set (if SOI is set) or until GO is cleared (regardless of the value of SOI) |
| 0 | ADC is cleared upon completion of each conversion trigger |

**Bit 4 – CS** ADC Clock Selection

| Value | Description |
|---|---|
| 1 | Clock supplied from ADCRC dedicated oscillator |
| 0 | Clock supplied by $F_{OSC}$, divided according to the ADCLK register |

**Bits 3:2 – FM[1:0]** ADC Results Format/Alignment Selection

| Value | Condition | Description |
|---|---|---|
| x1 | IC = 0 | ADRES and ADPREV data are right justified |
| x0 | IC = 0 | ADRES and ADPREV data are left justified, zero-filled |
| 11 | IC = 1 | ADRES and ADPREV data are right justified, sign bit |
| 10 | IC = 1 | ADRES and ADPREV data are left justified, sign bit, zero-filled |
| 01 | IC = 1 | ADRES and ADPREV data are right justified, two's complement |
| 00 | IC = 1 | ADRES and ADPREV data are left justified, two's complement, zero-filled |

**Bit 1 – IC** ADC Input Configuration

| Value | Description |
|---|---|
| 1 | ADC is operating in Differential mode |
| 0 | ADC is operating in Single-Ended mode |

**Bit 0 – GO** ADC Conversion Status(1,2)

| Value | Description |
|---|---|
| 1 | ADC conversion cycle in progress. Setting this bit starts an ADC conversion cycle. The bit is cleared by hardware as determined by the CONT bit. |
| 0 | ADC conversion completed/not in progress |

**Notes:**

1. This bit requires the ON bit to be set.
2. If cleared by software while a conversion is in progress, the results of the conversion up to this point will be transferred to ADRES and the state machine will be reset, but the ADIF interrupt flag bit will not be set; filter and threshold operations will not be performed.

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



### 33.5.2 ADCON1

**Name:** ADCON1
**Offset:** 0x1D27

ADC Control Register 1

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PPOL | IPEN | GPOL | | | | PCSC | DSEN |
| Access | R/W | R/W | R/W | | | | R/W | R/W |
| Reset | 0 | 0 | 0 | | | | 0 | 0 |

**Bit 7 – PPOL** Precharge Polarity
Action During 1st Precharge Stage

| Value | Condition | Description |
|---|---|---|
| x | ADPRE = 0 | Bit has no effect |
| 1 | ADPRE > 0 | External analog I/O pin is connected to $V_{DD}$<br>Internal AD sampling capacitor ($C_{HOLD}$) is connected to $V_{SS}$ |
| 0 | ADPRE > 0 | External analog I/O pin is connected to $V_{SS}$<br>Internal AD sampling capacitor ($C_{HOLD}$) is connected to $V_{DD}$ |

**Bit 6 – IPEN** A/D Inverted Precharge Enable

| Value | Condition | Description |
|---|---|---|
| x | DSEN = 0 | Bit has no effect |
| 1 | DSEN = 1 | The precharge and guard signals in the second conversion cycle are the opposite polarity of the first cycle |
| 0 | DSEN = 1 | Both Conversion cycles use the precharge and guards specified by PPOL and GPOL |

**Bit 5 – GPOL** Guard Ring Polarity Selection

| Value | Description |
|---|---|
| 1 | ADC guard Ring outputs start as digital high during Precharge stage |
| 0 | ADC guard Ring outputs start as digital low during Precharge stage |

**Bit 1 – PCSC** Precharge Sample Capacitor Only

| Value | Description |
|---|---|
| 1 | Precharge only applies to the internal sampling capacitor |
| 0 | Precharge applies to both the internal sampling capacitor and the external I/O pin |

**Bit 0 – DSEN** Double-Sample Enable

| Value | Description |
|---|---|
| 1 | Two conversions are processed as a pair. The selected computation is performed after every second conversion. |
| 0 | Selected computation is performed after every conversion |

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



### 33.5.3 ADCON2

**Name:** ADCON2
**Offset:** 0x1D28

ADC Control Register 2

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PSIS | CRS[2:0] | | | ACLR | MD[2:0] | | |
| Access | R/W | R/W | R/W | R/W | R/W/HC | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 7 – PSIS** ADC Previous Sample Input Select

| Value | Description |
|---|---|
| 1 | ADFLTR is transferred to ADPREV at the start of conversion |
| 0 | ADRES is transferred to ADPREV at the start of conversion |

**Bits 6:4 – CRS[2:0]** ADC Accumulated Calculation Right Shift Select

| Value | Condition | Description |
|---|---|---|
| 1 to 6 | MD = 'b100 | Low-pass filter time constant is $2^{CRS}$, filter gain is 1:1(2) |
| 1 to 6 | MD = 'b011 to 'b001 | The accumulated value is right-shifted by CRS (divided by $2^{CRS}$)(1,2) |
| x | MD = 'b000 or 'b111 | These bits are ignored |

**Bit 3 – ACLR** A/D Accumulator Clear Command(3)

| Value | Description |
|---|---|
| 1 | Registers ADACC, ADCNT and the AOV bit are cleared |
| 0 | Clearing action is complete (or not started) |

**Bits 2:0 – MD[2:0]** ADC Operating Mode Selection(4)

| Value | Description |
|---|---|
| 111-101 | Reserved |
| 100 | Low-Pass Filter mode |
| 011 | Burst Average mode |
| 010 | Average mode |
| 001 | Accumulate mode |
| 000 | Basic (Legacy) mode |

**Notes:**

1. To correctly calculate an average, the number of samples (set in ADRPT) must be $2^{CRS}$.
2. CRS = 'b111 and 'b000 are reserved.
3. This bit is cleared by hardware when the accumulator operation is complete; depending on oscillator selections, the delay may be many instructions.
4. See the **"Computation Operation"** section for full mode descriptions.

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



### 33.5.4 ADCON3

**Name:** ADCON3
**Offset:** 0x1D29

ADC Control Register 3

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | CALC[2:0] | | | SOI | TMD[2:0] | | |
| Access | | R/W | R/W | R/W | R/W/HC | R/W | R/W | R/W |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 6:4 – CALC[2:0]** ADC Error Calculation Mode Select

**Table 33-5. ADC Error Calculation Mode**

| CALC | ADERR | | Application |
|---|---|---|---|
| | DSEN = 0 Single-Sample Mode | DSEN = 1 CVD Double-Sample Mode(1) | |
| 111 | Reserved | Reserved | Reserved |
| 110 | Reserved | Reserved | Reserved |
| 101 | ADFLTR-ADSTPT | ADFLTR-ADSTPT | Average/filtered value vs. setpoint |
| 100 | ADPREV-ADFLTR | ADPREV-ADFLTR | First derivative of filtered value(3) (negative) |
| 011 | Reserved | Reserved | Reserved |
| 010 | ADRES-ADFLTR | (ADRES-ADPREV)-ADFLTR | Actual result vs. averaged/filtered value |
| 001 | ADRES-ADSTPT | (ADRES-ADPREV)-ADSTPT | Actual result vs. setpoint |
| 000 | ADRES-ADPREV | ADRES-ADPREV | First derivative of single measurement(2) |
| | | | Actual CVD result(2) |

**Notes:**
1. When DSEN = 1 and PSIS = 0, ADERR is computed only after every second sample.
2. When PSIS = 0.
3. When PSIS = 1.

**Bit 3 – SOI** ADC Stop-on-Interrupt

| Value | Condition | Description |
|---|---|---|
| x | CONT = 0 | This bit is not used |
| 1 | CONT = 1 | GO is cleared when the Threshold conditions are met, otherwise the conversion is retriggered |
| 0 | CONT = 1 | GO is not cleared by hardware, must be cleared by software to stop retriggers |

**Bits 2:0 – TMD[2:0]** Threshold Interrupt Mode Select

| Value | Description |
|---|---|
| 111 | Interrupt regardless of threshold test results |
| 110 | Interrupt if ADERR > ADUTH |
| 101 | Interrupt if ADERR ≤ ADUTH |
| 100 | Interrupt if ADERR < ADLTH or ADERR > ADUTH |
| 011 | Interrupt if ADERR > ADLTH and ADERR < ADUTH |
| 010 | Interrupt if ADERR ≥ ADLTH |
| 001 | Interrupt if ADERR < ADLTH |
| 000 | Never interrupt |

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



### 33.5.5 ADSTAT

**Name:** ADSTAT
**Offset:** 0x1D2A

ADC Status Register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | AOV | UTHR | LTHR | MATH | | STAT[2:0] | | |
| Access | R/C/HS/HC | R | R | R/W/HS | | R | R | R |
| Reset | 0 | 0 | 0 | 0 | | 0 | 0 | 0 |

**Bit 7 – AOV** ADC Accumulator Overflow

| Value | Description |
|---|---|
| 1 | ADACC or ADFLTR or ADERR registers have overflowed |
| 0 | ADACC, ADFLTR and ADERR registers have not overflowed |

**Bit 6 – UTHR** ADC Module Greater-than Upper Threshold Flag

| Value | Description |
|---|---|
| 1 | ADERR > ADUTH |
| 0 | ADERR ≤ ADUTH |

**Bit 5 – LTHR** ADC Module Less-than Lower Threshold Flag

| Value | Description |
|---|---|
| 1 | ADERR < ADLTH |
| 0 | ADERR ≥ ADLTH |

**Bit 4 – MATH** ADC Module Computation Status
ADC Module Computation Status(1)

| Value | Description |
|---|---|
| 1 | Registers ADACC, ADFLTR, ADUTH, ADLTH and the AOV bit are updating or have already updated |
| 0 | Associated registers/bits have not changed since this bit was last cleared |

**Bits 2:0 – STAT[2:0]** ADC Module Cycle Multi-Stage Status

| Value | Description |
|---|---|
| 111 | ADC module is in 2nd conversion stage |
| 110 | ADC module is in 2nd acquisition stage |
| 101 | ADC module is in 2nd precharge stage |
| 100 | ADC computation is suspended between 1st and 2nd sample; the computation results are incomplete and awaiting data from the 2nd sample(2,3) |
| 011 | ADC module is in 1st conversion stage |
| 010 | ADC module is in 1st acquisition stage |
| 001 | ADC module is in 1st precharge stage |
| 000 | ADC module is not converting |

**Notes:**

1. MATH bit cannot be cleared by software while STAT = 'b100.
2. If ADC clock source is ADCRC, and $F_{OSC}$ < ADCRC, the indicated status may not be valid.
3. STAT = 'b100 appears between the two triggers when DSEN = 1 and CONT = 0.

© 2023 Microchip Technology Inc.
and its subsidiaries

chanhvinh.com



### 33.5.6 ADCLK

**Name:** ADCLK
**Offset:** 0x1D2D

ADC Clock divider Register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | CS[5:0] | | | | | |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 5:0 – CS[5:0]** ADC Clock divider Select

| Value | Description |
|---|---|
| n | ADC Clock frequency = $F_{OSC}/(2*(n+1))$ |

**Note:** ADC Clock divider is only available if $F_{OSC}$ is selected as the ADC clock source (CS = 0).

chanhvinh.com



### 33.5.7 ADREF

**Name:** ADREF
**Offset:** 0x1D2B

ADC Reference Selection Register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | NREF | | | PREF[1:0] | |
| Access | | | | R/W | | | R/W | R/W |
| Reset | | | | 0 | | | 0 | 0 |

**Bit 4 – NREF** ADC Negative Voltage Reference Selection

| Value | Description |
|---|---|
| 1 | $V_{REF}$- is connected to external $V_{REF}$- |
| 0 | $V_{REF}$- is connected to $AV_{SS}$ |

**Bits 1:0 – PREF[1:0]** ADC Positive Voltage Reference Selection

| Value | Description |
|---|---|
| 11 | $V_{REF}$+ is connected to internal Fixed Voltage Reference (FVR) module |
| 10 | $V_{REF}$+ is connected to external $V_{REF}$+ |
| 01 | Reserved |
| 00 | $V_{REF}$+ is connected to $V_{DD}$ |

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



### 33.5.8 ADPCH

**Name:** ADPCH
**Offset:** 0x1D1F

ADC Positive Channel Selection Register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | PCH[5:0] | | | | | |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 5:0 – PCH[5:0]** ADC Positive Input Channel Selection

**Table 33-6. ADC Positive Input Channel Selections**

| PCH | ADC Positive Channel Input |
|---|---|
| 111111 | Fixed Voltage Reference (FVR) Buffer 2(1) |
| 111110 | Fixed Voltage Reference (FVR) Buffer 1(1) |
| 111101 | DAC2OUT |
| 111100 | DAC1OUT |
| 111011 | Temperature Indicator(2) |
| 111010 | $V_{SS}$ (Analog Ground) |
| 111001 | OPA1OUT |
| 111000-100100 | Reserved. No channel connected. |
| 100011 | ADCG1 |
| 100010 | RE2(3) |
| 100001 | RE1(3) |
| 100000 | RE0(3) |
| 011111 | RD7(3) |
| 011110 | RD6(3) |
| 011101 | RD5(3) |
| 011100 | RD4(3) |
| 011011 | RD3(3) |
| 011010 | RD2(3) |
| 011001 | RD1(3) |
| 011000 | RD0(3) |
| 010111 | RC7 |
| 010110 | RC6 |
| 010101 | RC5 |
| 010100 | RC4 |
| 010011 | RC3 |
| 010010 | RC2 |
| 010001 | RC1 |
| 010000 | RC0 |
| 001111 | RB7 |
| 001110 | RB6 |
| 001101 | RB5 |
| 001100 | RB4 |
| 001011 | RB3 |
| 001010 | RB2 |

chanhvinh.com



.....continued

| PCH | ADC Positive Channel Input |
|---|---|
| 001001 | RB1 |
| 001000 | RB0 |
| 000111 | RA7 |
| 000110 | RA6 |
| 000101 | RA5 |
| 000100 | RA4 |
| 000011 | RA3 |
| 000010 | RA2 |
| 000001 | RA1 |
| 000000 | RA0 |

**Notes:**

1. Refer to the **"Fixed Voltage Reference Module"** chapter for more details.
2. Refer to the **"Temperature Indicator Module"** chapter for more details.
3. 40/44-pin devices only.

chanhvinh.com

chanhvinh.com



### 33.5.9 ADNCH

**Name:** ADNCH
**Offset:** 0x1D20

ADC Negative Channel Selection Register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | NCH[5:0] | | | | | |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 5:0 – NCH[5:0]** ADC Negative Input Channel Selection

**Table 33-7. ADC Negative Input Channel Selections**

| NCH | ADC Negative Channel Input |
|---|---|
| 111111 | Fixed Voltage Reference (FVR) Buffer 2(1) |
| 111110 | Fixed Voltage Reference (FVR) Buffer 1(1) |
| 111101 | DAC2_OUT(2) |
| 111100 | DAC1_OUT(2) |
| 111011 | Temperature Indicator(3) |
| 111010 | $V_{SS}$ (Analog Ground) |
| 111001 | OPA1_OUT |
| 111000-100010 | Reserved. No channel connected. |
| 100001 | RE1(4) |
| 100000 | Reserved. No channel connected. |
| 011111 | RD7(4) |
| 011110 | Reserved. No channel connected. |
| 011101 | RD5(4) |
| 011100 | Reserved. No channel connected. |
| 011011 | RD3(4) |
| 011010 | RD2(4) |
| 011001 | Reserved. No channel connected. |
| 011000 | Reserved. No channel connected. |
| 010111 | Reserved. No channel connected. |
| 010110 | RC6 |
| 010101 | Reserved. No channel connected. |
| 010100 | Reserved. No channel connected. |
| 010011 | RC3 |
| 010010 | Reserved. No channel connected. |
| 010001 | RC1 |
| 010000 | Reserved. No channel connected. |
| 001111 | Reserved. No channel connected. |
| 001110 | RB6 |
| 001101 | Reserved. No channel connected. |
| 001100 | RB4 |
| 001011 | Reserved. No channel connected. |
| 001010 | RB2 |
| 001001 | Reserved. No channel connected. |
| 001000 | RB0 |

© 2023 Microchip Technology Inc.
and its subsidiaries

chanhvinh.com



.....continued

| NCH | ADC Negative Channel Input |
|---|---|
| 000111 | Reserved. No channel connected. |
| 000110 | RA6 |
| 000101 | RA5 |
| 000100 | Reserved. No channel connected. |
| 000011 | RA3 |
| 000010 | Reserved. No channel connected. |
| 000001 | RA1 |
| 000000 | Reserved. No channel connected. |

**Notes:**

1. Refer to the **"Fixed Voltage Reference Module"** chapter for more details.
2. Refer to the **"Digital-to-Analog Converter Module"** chapter for more details.
3. Refer to the **"Temperature Indicator Module"** chapter for more details.
4. 40/44-pin devices only.

chanhvinh.com

chanhvinh.com



### 33.5.10 ADPRE

**Name:** ADPRE
**Offset:** 0x1D24

ADC Precharge Time Control Register

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | PRE[12:8] | | | | |
| Access | | | | R/W | R/W | R/W | R/W | R/W |
| Reset | | | | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PRE[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 12:0 – PRE[12:0]** Precharge Time Select

**Table 33-8. Precharge Time**

| ADPRE | Precharge Time | |
|---|---|---|
| | CS != ADCRC | CS = ADCRC |
| 1 1111 1111 1111 | 8191 clocks of $F_{OSC}$ | 8191 clocks of ADCRC |
| 1 1111 1111 1110 | 8190 clocks of $F_{OSC}$ | 8190 clocks of ADCRC |
| 1 1111 1111 1101 | 8189 clocks of $F_{OSC}$ | 8189 clocks of ADCRC |
| ... | ... | ... |
| 0 0000 0000 0010 | 2 clocks of $F_{OSC}$ | 2 clocks of ADCRC |
| 0 0000 0000 0001 | 1 clocks of $F_{OSC}$ | 1 clocks of ADCRC |
| 0 0000 0000 0000 | Not included in the data conversion cycle | |

**Notes:** The individual bytes in this multibyte register can be accessed with the following register names:

- ADPREH: Accesses the high byte ADPRE[12:8]
- ADPREL: Accesses the low byte ADPRE[7:0]

chanhvinh.com



### 33.5.11 ADACQ

**Name:** ADACQ
**Offset:** 0x1D21

ADC Acquisition Time Control Register

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | ACQ[12:8] | | | | |
| Access | | | | R/W | R/W | R/W | R/W | R/W |
| Reset | | | | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ACQ[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 12:0 – ACQ[12:0]** Acquisition (charge share time) Select

**Table 33-9. Acquisition Time**

| ADACQ | Acquisition Time | |
|---|---|---|
| | CS != ADCRC | CS = ADCRC |
| 1 1111 1111 1111 | 8191 clocks of $F_{OSC}$ | 8191 clocks of ADCRC |
| 1 1111 1111 1110 | 8190 clocks of $F_{OSC}$ | 8190 clocks of ADCRC |
| 1 1111 1111 1101 | 8189 clocks of $F_{OSC}$ | 8189 clocks of ADCRC |
| ... | ... | ... |
| 0 0000 0000 0010 | 2 clocks of $F_{OSC}$ | 2 clocks of ADCRC |
| 0 0000 0000 0001 | 1 clocks of $F_{OSC}$ | 1 clocks of ADCRC |
| 0 0000 0000 0000 | Not included in the data conversion cycle(1) | |

**Note:**
1. If ADPRE is not equal to '0', then ACQ = 0 means Acquisition Time is 8192 clocks of $F_{OSC}$ or ADCRC.

**Notes:** The individual bytes in this multibyte register can be accessed with the following register names:
- ADACQH: Accesses the high byte ADACQ[12:8]
- ADACQL: Accesses the low byte ADACQ[7:0]

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



### 33.5.12 ADCAP

**Name:** ADCAP
**Offset:** 0x1D23

ADC Additional Sample Capacitor Selection Register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | CAP[4:0] | | | | |
| Access | | | | R/W | R/W | R/W | R/W | R/W |
| Reset | | | | 0 | 0 | 0 | 0 | 0 |

**Bits 4:0 – CAP[4:0]** ADC Additional Sample Capacitor Selection

| Value | Description |
|---|---|
| 1 to 31 | Number of pF in the additional capacitance |
| 0 | No additional capacitance |

chanhvinh.com



### 33.5.13 ADRPT

**Name:** ADRPT
**Offset:** 0x1D1A

ADC Repeat Setting Register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RPT[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 7:0 – RPT[7:0]** ADC Repeat Threshold
Determines the number of times that the ADC is triggered for a threshold check. When CNT reaches this value, the error threshold is checked. Used when the computation mode is Low-Pass Filter, Burst Average, or Average. See the **"Computation Operation"** section for more details.

chanhvinh.com



### 33.5.14 ADCNT

**Name:** ADCNT
**Offset:** 0x1D19

ADC Repeat Counter Register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CNT[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 7:0 – CNT[7:0]** ADC Repeat Count
Counts the number of times that the ADC is triggered before the threshold is checked. When this value reaches RPT, the threshold is checked. Used when the computation mode is Low-Pass Filter, Burst Average, or Average. See the **"Computation Operation"** section for more details.

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



### 33.5.15 ADFLTR

**Name:** ADFLTR
**Offset:** 0x1D14

ADC Filter Register

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | FLTR[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | x | x | x | x | x | x | x | x |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | FLTR[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | x | x | x | x | x | x | x | x |

**Bits 15:0 – FLTR[15:0]** ADC Filter Output - Signed two's complement
In Accumulate, Average, and Burst Average mode, this is equal to ACC right shifted by the CRS bits. In LPF mode, this is the output of the Low-Pass Filter.

**Notes:** The individual bytes in this multibyte register can be accessed with the following register names:

- ADFLTRH: Accesses the high byte ADFLTR[15:8]
- ADFLTRL: Accesses the low byte ADFLTR[7:0]

chanhvinh.com

chanhvinh.com



### 33.5.16 ADRES

**Name:** ADRES
**Offset:** 0x1D1D

ADC Result Register

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | RES[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RES[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 15:0 – RES[15:0]** ADC Sample Result

**Notes:** The individual bytes in this multibyte register can be accessed with the following register names:

- ADRESH: Accesses the high byte ADRES[15:18]
- ADRESL: Accesses the low byte ADRES[7:0]

chanhvinh.com



### 33.5.17 ADPREV

**Name:** ADPREV
**Offset:** 0x1D1B

ADC Previous Result Register

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | PREV[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PREV[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 15:0 – PREV[15:0]** Previous ADC Result

| Value | Condition | Description |
|---|---|---|
| n | PSIS = 1 | n = ADFLTR value at the start of current ADC conversion |
| n | PSIS = 0 | n = ADRES at the start of current ADC conversion(1) |

**Notes:**

1. If PSIS = 0, ADPREV is formatted the same way as ADRES is, depending on the FM bits.
2. The individual bytes in this multibyte register can be accessed with the following register names:
   – ADPREVH: Accesses ADPREV[15:8]
   – ADPREVL: Accesses ADPREV[7:0].

chanhvinh.com

chanhvinh.com



### 33.5.18 ADACC

**Name:** ADACC
**Offset:** 0x1D16

ADC Accumulator Register(1)

See the **"Computation Operation"** section for more details.

**Important:** This register contains signed two's complement accumulator value and the upper unused bits contain copies of the sign bit.

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | ACC[17:16] | |
| Access | | | | | | | R/W | R/W |
| Reset | | | | | | | x | x |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | ACC[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | x | x | x | x | x | x | x | x |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ACC[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | x | x | x | x | x | x | x | x |

**Bits 17:0 – ACC[17:0]** ADC Accumulator - Signed two's complement

**Notes:**

1. This register can only be written when GO = 0.
2. The individual bytes in this multibyte register can be accessed with the following register names:
   - ADACCU: Accesses the upper byte ADACC[17:16]
   - ADACCH: Accesses the high byte ADACC[15:8]
   - ADACCL: Accesses the low byte ADACC[7:0].

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



### 33.5.19 ADSTPT

**Name:** ADSTPT
**Offset:** 0x1D12

ADC Threshold Setpoint Register
Depending on CALC, it may be used to determine ADERR.

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | STPT[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | STPT[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 15:0 – STPT[15:0]** ADC Threshold Setpoint - Signed two's complement

**Notes:** The individual bytes in this multibyte register can be accessed with the following register names:

- ADSTPTH: Accesses the high byte ADSTPT[15:8]
- ADSTPTH: Accesses the low byte ADSTPT[7:0]

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



### 33.5.20 ADERR

**Name:** ADERR
**Offset:** 0x1D10

ADC Setpoint Error Register
ADC Setpoint Error calculation is determined by the CALC bits.

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | ERR[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | x | x | x | x | x | x | x | x |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ERR[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | x | x | x | x | x | x | x | x |

**Bits 15:0 – ERR[15:0]** ADC Setpoint Error - Signed two's complement

**Notes:** The individual bytes in this multibyte register can be accessed with the following register names:
- ADERRH: Accesses the high byte ADERR[15:8]
- ADERRL: Accesses the low byte ADERR[7:0]

chanhvinh.com

chanhvinh.com



### 33.5.21 ADLTH

**Name:** ADLTH
**Offset:** 0x1D0C

ADC Lower Threshold Register
ADLTH and ADUTH are compared with ADERR to set the UTHR and LTHR bits. Depending on the setting of the TMD bits, an interrupt may be triggered by the results of this comparison.

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | LTH[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | LTH[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 15:0 – LTH[15:0]** ADC Lower Threshold - Signed two's complement

**Notes:** The individual bytes in this multibyte register can be accessed with the following register names:

- ADLTHH: Accesses the high byte ADLTH[15:8]
- ADLTHL: Accesses the low byte ADLTH[7:0]

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



### 33.5.22 ADUTH

**Name:** ADUTH
**Offset:** 0x1D0E

ADC Upper Threshold Register
ADLTH and ADUTH are compared with ADERR to set the UTHR and LTHR bits. Depending on the setting of the TMD bits, an interrupt may be triggered by the results of this comparison.

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | UTH[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | UTH[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 15:0 – UTH[15:0]** ADC Upper Threshold - Signed two's complement

**Notes:** The individual bytes in this multibyte register can be accessed with the following register names:

- ADUTHH: Accesses the high byte ADUTH[15:8]
- ADUTHL: Accesses the low byte ADUTH[7:0]

chanhvinh.com



### 33.5.23 ADACT

**Name:** ADACT
**Offset:** 0x1D2C

ADC Auto-Conversion Trigger Source Selection Register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | ACT[4:0] | | | | |
| Access | | | | R/W | R/W | R/W | R/W | R/W |
| Reset | | | | 0 | 0 | 0 | 0 | 0 |

**Bits 4:0 – ACT[4:0]** Auto-Conversion Trigger Select

**Table 33-10. ADC Auto-Conversion Trigger Sources**

| ACT | Auto-Conversion Trigger Source |
|---|---|
| 11111 | Software write to ADPCH |
| 11110 | Software read of ADRESH |
| 11101 | Software read of ADERRH |
| 11100-11010 | Reserved |
| 11001 | CLC4_OUT |
| 11000 | CLC3_OUT |
| 10111 | CLC2_OUT |
| 10110 | CLC1_OUT |
| 10101 | Interrupt-on-change Interrupt Flag |
| 10100 | C2_OUT |
| 10011 | C1_OUT |
| 10010 | NCO1_OUT |
| 10001 | PWM4S1P2_OUT |
| 10000 | PWM4S1P1_OUT |
| 01111 | PWM3S1P2_OUT |
| 01110 | PWM3S1P1_OUT |
| 01101 | PWM2S1P2_OUT |
| 01100 | PWM2S1P1_OUT |
| 01011 | PWM1S1P2_OUT |
| 01010 | PWM1S1P1_OUT |
| 01001 | CCP2_OUT |
| 01000 | CCP1_OUT |
| 00111 | TMR6_postscaled_OUT |
| 00110 | TMR4_postscaled_OUT |
| 00101 | TMR3_overflow |
| 00100 | TMR2_postscaled_OUT |
| 00011 | TMR1_overflow |
| 00010 | TMR0_overflow |
| 00001 | Pin selected by ADACTPPS |
| 00000 | External Trigger Disabled |

© 2023 Microchip Technology Inc.
and its subsidiaries

chanhvinh.com



### 33.5.24 ADCGxA

**Name:** ADCGxA
**Offset:** 0x1D2E

ADC Channel Group Selection Port A

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CGA7 | CGA6 | CGA5 | CGA4 | CGA3 | CGA2 | CGA1 | CGA0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 0, 1, 2, 3, 4, 5, 6, 7 – CGAn** Channel Group Selection Enable on RA Pins

**Note:** Refer to the **"Pin Allocation Table"** for details about available pins per port.

© 2023 Microchip Technology Inc.
and its subsidiaries

chanhvinh.com



### 33.5.25 ADCGxB

**Name:** ADCGxB
**Offset:** 0x1D2F

ADC Channel Group Selection Port B

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CGB7 | CGB6 | CGB5 | CGB4 | CGB3 | CGB2 | CGB1 | CGB0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 0, 1, 2, 3, 4, 5, 6, 7 – CGBn** Channel Group Selection Enable on RB Pins

**Note:** Refer to the **"Pin Allocation Table"** for details about available pins per port.

chanhvinh.com



### 33.5.26 ADCGxC

**Name:** ADCGxC
**Offset:** 0x1D30

ADC Channel Group Selection Port C

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CGC7 | CGC6 | CGC5 | CGC4 | CGC3 | CGC2 | CGC1 | CGC0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 0, 1, 2, 3, 4, 5, 6, 7 – CGCn** Channel Group Selection Enable on RC Pins

**Note:** Refer to the **"Pin Allocation Table"** for details about available pins per port.

chanhvinh.com



### 33.5.27 ADCGxD

**Name:** ADCGxD
**Offset:** 0x1D31

ADC Channel Group Selection Port D

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CGD7 | CGD6 | CGD5 | CGD4 | CGD3 | CGD2 | CGD1 | CGD0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 0, 1, 2, 3, 4, 5, 6, 7 – CGDn** Channel Group Selection Enable on RD Pins

**Note:** Refer to the **"Pin Allocation Table"** for details about available pins per port.

chanhvinh.com



### 33.5.28 ADCGxE

**Name:** ADCGxE
**Offset:** 0x1D32

ADC Channel Group Selection Port E

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | CGE2 | CGE1 | CGE0 |
| Access | | | | | | R/W | R/W | R/W |
| Reset | | | | | | 0 | 0 | 0 |

**Bits 0, 1, 2 – CGEn** Channel Group Selection Enable on RE Pins

**Note:** Refer to the **"Pin Allocation Table"** for details about available pins per port.

chanhvinh.com



## 33.6 Register Summary - ADC

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 ... 0x1D0B | Reserved | | | | | | | | | |
| 0x1D0C | ADLTH | 7:0 | LTH[7:0] | | | | | | | |
| | | 15:8 | LTH[15:8] | | | | | | | |
| 0x1D0E | ADUTH | 7:0 | UTH[7:0] | | | | | | | |
| | | 15:8 | UTH[15:8] | | | | | | | |
| 0x1D10 | ADERR | 7:0 | ERR[7:0] | | | | | | | |
| | | 15:8 | ERR[15:8] | | | | | | | |
| 0x1D12 | ADSTPT | 7:0 | STPT[7:0] | | | | | | | |
| | | 15:8 | STPT[15:8] | | | | | | | |
| 0x1D14 | ADFLTR | 7:0 | FLTR[7:0] | | | | | | | |
| | | 15:8 | FLTR[15:8] | | | | | | | |
| 0x1D16 | ADACC | 7:0 | ACC[7:0] | | | | | | | |
| | | 15:8 | ACC[15:8] | | | | | | | |
| | | 23:16 | | | | | | | ACC[17:16] | |
| 0x1D19 | ADCNT | 7:0 | CNT[7:0] | | | | | | | |
| 0x1D1A | ADRPT | 7:0 | RPT[7:0] | | | | | | | |
| 0x1D1B | ADPREV | 7:0 | PREV[7:0] | | | | | | | |
| | | 15:8 | PREV[15:8] | | | | | | | |
| 0x1D1D | ADRES | 7:0 | RES[7:0] | | | | | | | |
| | | 15:8 | RES[15:8] | | | | | | | |
| 0x1D1F | ADPCH | 7:0 | | | PCH[5:0] | | | | | |
| 0x1D20 | ADNCH | 7:0 | | | NCH[5:0] | | | | | |
| 0x1D21 | ADACQ | 7:0 | ACQ[7:0] | | | | | | | |
| | | 15:8 | | | | ACQ[12:8] | | | | |
| 0x1D23 | ADCAP | 7:0 | | | | CAP[4:0] | | | | |
| 0x1D24 | ADPRE | 7:0 | PRE[7:0] | | | | | | | |
| | | 15:8 | | | | PRE[12:8] | | | | |
| 0x1D26 | ADCON0 | 7:0 | ON | CONT | | CS | FM[1:0] | | IC | GO |
| 0x1D27 | ADCON1 | 7:0 | PPOL | IPEN | GPOL | | | | PCSC | DSEN |
| 0x1D28 | ADCON2 | 7:0 | PSIS | CRS[2:0] | | | ACLR | MD[2:0] | | |
| 0x1D29 | ADCON3 | 7:0 | | CALC[2:0] | | | SOI | TMD[2:0] | | |
| 0x1D2A | ADSTAT | 7:0 | AOV | UTHR | LTHR | MATH | | STAT[2:0] | | |
| 0x1D2B | ADREF | 7:0 | | | | NREF | | | PREF[1:0] | |
| 0x1D2C | ADACT | 7:0 | | | | ACT[4:0] | | | | |
| 0x1D2D | ADCLK | 7:0 | | | CS[5:0] | | | | | |
| 0x1D2E | ADCG1A | 7:0 | CGA7 | CGA6 | CGA5 | CGA4 | CGA3 | CGA2 | CGA1 | CGA0 |
| 0x1D2F | ADCG1B | 7:0 | CGB7 | CGB6 | CGB5 | CGB4 | CGB3 | CGB2 | CGB1 | CGB0 |
| 0x1D30 | ADCG1C | 7:0 | CGC7 | CGC6 | CGC5 | CGC4 | CGC3 | CGC2 | CGC1 | CGC0 |
| 0x1D31 | ADCG1D | 7:0 | CGD7 | CGD6 | CGD5 | CGD4 | CGD3 | CGD2 | CGD1 | CGD0 |
| 0x1D32 | ADCG1E | 7:0 | | | | | | CGE2 | CGE1 | CGE0 |

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com

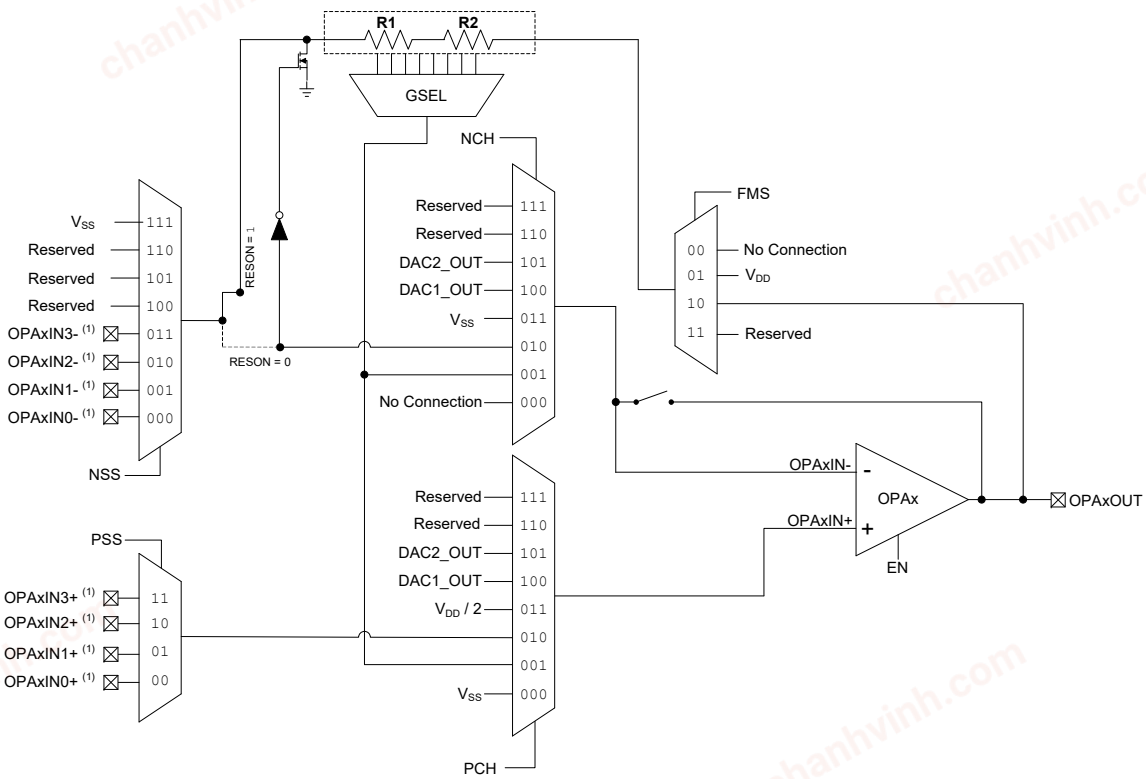


# 34. OPA - Operational Amplifier

The Operational Amplifier (OPA) module features a standard general purpose three-terminal device with programmable gain options, adjustable input offset voltage and hardware override control capabilities. The OPA module has the following features:

- 2.3 MHz Gain Bandwidth
  - Dedicated external output (OPAxOUT)
  - Multiple noninverting input pins available (OPAxIN+)
  - Multiple inverting input pins available (OPAxIN-)
- Programmable Gain Options Using Built-in Internal Resistor Ladder
- Configurable Positive and Negative Source Selections
- Hardware Controlled Drive with Override Controls
  - Forced Unity Gain mode
  - Forced Rail Drive mode
- Programmable Input Offset Voltage Calibration
- Internal Connection to the ADC Module
  - Allows OPA to be used as programmable gain amplifier for the ADC Input

**Figure 34-1. Operational Amplifier Module Block Diagram**




**Note:**

1. Refer to the **"Pin Allocation Table"** for details about OPAxIN- and OPAxIN+ availability per port.

chanhvinh.com



## 34.1 OPA Module Control

The OPA module is enabled by setting the EN bit of the corresponding operational amplifier Configuration register. Once enabled, the OPA module forces the output driver of the output pin (OPAxOUT) into tri-state. Forcing the output pin into tri-state prevents contention between the digital PORT driver of the pin and the output of the OPA during operation. The Software Output Control (SOC) bits are used to select the OPA mode of operation when hardware controlled override is not being used (OREN = 0).

### 34.1.1 Unity Gain Mode

The OPA module can be configured to operate in Unity Gain mode either in software or hardware, depending on the configuration of the Override Enable (OREN) bit. If the OREN bit has not been set, Unity Gain mode can be enabled in software by setting the Unity Gain Enable (UG) bit. Once Unity Gain mode has been enabled, the output of the OPA will be connected internally to the inverting input and the OPA will operate with unity gain feedback. If hardware-controlled override has been enabled by setting the OREN bit, the OPAxHWC register may be used to configure the OPA mode of operation, depending on the status of the override source. Refer to the Hardware Override Control section for more information.



**Tip:** Operating the OPA module in Unity Gain mode relinquishes the need for an external inverting input pin (OPAxIN-), since it connects internally to the OPA output, which allows that pin to be used for general purpose I/O.

### 34.1.2 Programmable Source Selection

The noninverting input source of the OPA module is selected using the Op Amp Noninverting Input Channel Selection (PCH) bits and can be connected to multiple internal sources or to an external input pin (OPAxIN+). If an external pin is chosen as the noninverting input source, the Positive Source Selection (PSS) bits may then be used to select from the available OPA noninverting input pins for the device.

The inverting input source of the OPA module is selected using the Op Amp Inverting Input Channel Selection (NCH) bits and can be connected to multiple internal sources or to an external input pin (OPAxIN-). If an external pin is chosen as the inverting input source, the Negative Source Selection (NSS) bits may then be used to select from the available inverting OPA input pins for the device.

### 34.1.3 Programmable Gain Options

The gain of the OPA module can be controlled either using external components to provide feedback or internally, using a built-in resistor ladder. When using the built-in programmable gain options of the OPA module, the internal resistor ladder must be enabled by setting the Resister Ladder Enable (RESON) bit. Once enabled, the nominal gain of the amplifier can be selected using the Op Amp Gain Selection (GSEL) bits.

The internal sources to the operational amplifier must be configured based on the OPA mode of operation, when using the built-in programmable gain options. The PCH and NCH bits can be used to connect the internal resistor ladder to the positive or negative input of the operational amplifier, respectively, creating the feedback network needed to control the gain of the circuit.

If the OPA module is being used in the noninverting configuration, the NCH bits may be configured so that the internal resistor ladder is connected to the negative input of the operational amplifier. If the OPA module is being used in an inverting configuration, the PCH bits may be configured so that the internal resistor ladder is connected to the positive input of the operational amplifier.

## 34.2 Hardware Override Control

The OPA mode of operation can be switched core independently, using the hardware override control feature built into the peripheral. Hardware override control is enabled by setting the OREN bit and selecting an override source using the ORS bits. The OPA mode of operation is determined based on the status of the selected override signal. The Hardware Override Control Configuration bits (HWCH and HWCL) are used to select the OPA mode of operation, when the override source is high or low, respectively. The ORPOL bit can be used to invert the hardware

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



controlled override input, meaning that when ORPOL = 1, the HWCH bits will determine the OPA mode of operation when the override source is low, and the HWCL bits will determine the OPA mode of operation when the override source is high. The hardware override control can be used to switch between the following OPA configurations:

- Basic Operation with User Defined Feedback
  - Feedback is based on the configuration of the internal gain options (GSEL) or external components, depending upon peripheral setup.
- Unity Gain Mode
- Rail Drive Mode
  - Forces the operational amplifier output to be driven to $V_{DD}$ or $V_{SS}$, depending on the status of the override source and the configured override polarity.

## 34.3 Input Offset Voltage

Input offset voltage is a measure of the voltage difference between the noninverting and inverting input sources in a closed loop circuit, with the operational amplifier operating in its linear region. The offset voltage will appear as a DC offset in the output equal to the input offset voltage, multiplied by the gain of the circuit. The input offset voltage is also affected by the Common-mode voltage. The OPA module is factory calibrated to minimize the input offset voltage.

### 34.3.1 Offset Calibration

The OPAxOFFSET register can be used to recalibrate or adjust the input offset voltage from the factory calibration. This can be accomplished by using the DAC module or an external constant-voltage source, in conjunction with the ADC module. The OPA input offset voltage can be re-calibrated using the following steps:

- Configure the DAC module to be used as a constant voltage reference connected to the noninverting input (OPAxIN+) of the OPA module.
  - An external constant voltage reference can be used, instead of the DAC, by connecting it to one of the external noninverting input pins.
- Configure the OPA module to operate in Unity Gain mode by setting the UG bit.
  - Use the PCH bits to connect the calibration source (either the DAC module or an external voltage reference) to the noninverting input (OPAxIN+) of the OPA.
  - If using an external voltage reference, the PSS must also be used to select which OPAxIN+ pin will be connected to the noninverting input.
- Perform an ADC conversion to measure the voltage of the selected calibration source. The value read by the ADC during this conversion will serve as the calibration target.
- Use the ADC Positive Channel Selection (ADPCH) register to select the OPA output (OPAxOUT), and then measure the output voltage of the OPA module using the ADC.
- The difference between the measured value of the calibration target and the measured value of the OPA output can be used to determine the value needed to calibrate the OPA input offset voltage using the OPAxOFFSET register.

**Important:** The OPA input offset voltage is factory calibrated, and any data written to the OPAxOFFSET register will adjust the input offset voltage from the factory calibrated value. The factory calibrated input offset voltage will be restored on a Reset event, overwriting any previous data that may have been written to the register.

## 34.4 OPA Operation with ADC

The OPA module provides internal connections directly to the ADC, allowing it to be used for analog signal conditioning before a signal is converted by the ADC. In this mode of operation, the output of the OPA (OPAxOUT) will connect internally to the input of the ADC, and any ADC conversions will be performed on that signal. When using this mode of operation, the ADPCH register of the ADC module may be used to select one of the available

chanhvinh.com



noninverting OPA input pins (OPAxIN+) Both the ADC and the OPA module must be configured accordingly to use this mode of operation.

**Tip:**

1. When using the OPA module with the ADC in this mode of operation, the OPA input pin selection determined using the ADC input channel selection registers will take precedence over the input pin selection using the PSS and NSS bits.
2. Although the output of the OPA (OPAxOUT) is connected internally to the ADC in this mode of operation, the OPAxOUT pin cannot be used as a general purpose I/O at this time.

## 34.5 Register Definitions: Operational Amplifier

Long bit name prefixes for the OPA peripherals are shown in the table below. Refer to the **"Long Bit Names"** section in the **"Register and Bit Naming Conventions"** chapter for more information.

**Table 34-1. Operational Amplifier Bit Name Prefixes**

| Peripheral | Bit Name Prefix |
|---|---|
| OPA1 | OPA1 |

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



### 34.5.1 OPAxCON0

**Name:** OPAxCON0
**Offset:** 0x090C

Operational Amplifier Control Register 0

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | EN | | CPON | | UG | | SOC[1:0] | |
| Access | R/W | | R/W | | R/W | | R/W | R/W |
| Reset | 0 | | 0 | | 0 | | 0 | 0 |

**Bit 7 – EN** Operational Amplifier Enable

| Value | Description |
|---|---|
| 1 | Operational amplifier is enabled |
| 0 | Operational amplifier is disabled and consumes no active power |

**Bit 5 – CPON** Charge Pump On Control

| Value | Description |
|---|---|
| 1 | OPA Charge Pump on |
| 0 | OPA Charge Pump off (Low Power mode) |

**Bit 3 – UG** Operational Amplifier Unity Gain Select

| Value | Description |
|---|---|
| 1 | Operational amplifier output is connected to inverting input, OPAxIN- input pins are available for general purpose I/O |
| 0 | Inverting input is connected to designated OPAxIN- pin |

**Bits 1:0 – SOC[1:0]** Software Output Control

| Value | Description |
|---|---|
| 11 | Reserved |
| 10 | Drive output to $V_{DD}$ |
| 01 | Drive output to $V_{SS}$ |
| 00 | Basic Operation; Operational amplifier configuration with user-defined or unity-gain feedback |

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



### 34.5.2 OPAxCON1

**Name:** OPAxCON1
**Offset:** 0x090D

Operational Amplifier Control Register 1

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | GSEL[2:0] | | | RESON | NSS[2:0] | | |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 6:4 – GSEL[2:0]** Operational Amplifier Gain Selection

**Table 34-2. Operational Amplifier Internal Resistor Ladder Selections**

| GSEL[2:0] | R1 | R2 | Inverting (R2/R1) | Noninverting (1 + R2/R1) |
|---|---|---|---|---|
| 111 | 1R | 15R | 15 | 16 |
| 110 | 2R | 14R | 7 | 8 |
| 101 | 4R | 12R | 3 | 4 |
| 100 | 6R | 10R | 5/3 | 8/3 |
| 011 | 8R | 8R | 1 | 2 |
| 010 | 12R | 4R | 1/3 | 4/3 |
| 001 | 14R | 2R | 1/7 | 8/7 |
| 000 | 15R | 1R | 1/15 | 16/15 |
| **Note:** R = 20 kΩ nominal | | | | |

**Bit 3 – RESON** Resistor Ladder Enable

| Value | Description |
|---|---|
| 1 | Internal Resistor Ladder is enabled; OPA input is connected to the resistor ladder allowing GSEL to be used to control programmable gain. |
| 0 | Internal Resistor Ladder is disabled. External feedback to the OPA is required unless operating in Unity Gain mode. |

**Bits 2:0 – NSS[2:0]** Negative Source Selection

| Value | Description |
|---|---|
| 111 | $V_{SS}$ |
| 110 | Reserved |
| 101 | Reserved |
| 100 | Reserved |
| 011 | OPAxIN3- |
| 010 | OPAxIN2- |
| 001 | OPAxIN1- |
| 000 | OPAxIN0- |

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



### 34.5.3 OPAxCON2

**Name:** OPAxCON2
**Offset:** 0x090E

Operational Amplifier Control Register 2

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | NCH[2:0] | | | | PCH[2:0] | | |
| Access | | R/W | R/W | R/W | | R/W | R/W | R/W |
| Reset | | 0 | 0 | 0 | | 0 | 0 | 0 |

**Bits 6:4 – NCH[2:0]** Operational Amplifier Inverting Input Channel Selection

| Value | Description |
|---|---|
| 111 | Reserved |
| 110 | Reserved |
| 101 | DAC2_OUT |
| 100 | DAC1_OUT |
| 011 | Reserved |
| 010 | OPAxIN- (NSS) |
| 001 | Internal Resistor Ladder (GSEL) |
| 000 | No Connection |

**Bits 2:0 – PCH[2:0]** Operational Amplifier Noninverting Input Channel Selection

| Value | Description |
|---|---|
| 111 | Reserved |
| 110 | Reserved |
| 101 | DAC2_OUT |
| 100 | DAC1_OUT |
| 011 | $V_{DD}$ / 2 |
| 010 | OPAxIN+ (PSS) |
| 001 | Internal Resistor Ladder (GSEL) |
| 000 | $V_{SS}$ |

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



### 34.5.4 OPAxCON3

**Name:** OPAxCON3
**Offset:** 0x090F

Operational Amplifier Control Register 3

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | FMS[1:0] | | | | | | PSS[1:0] | |
| Access | R/W | R/W | | | | | R/W | R/W |
| Reset | 0 | 0 | | | | | 0 | 0 |

**Bits 7:6 – FMS[1:0]** Feedback Mode Selection

| Value | Description |
|---|---|
| 11 | Reserved |
| 10 | Operational Amplifier Output Pin (OPAxOUT) |
| 01 | $V_{DD}$ |
| 00 | No Connection |

**Bits 1:0 – PSS[1:0]** Positive Source Selection

| Value | Description |
|---|---|
| 11 | OPAxIN3+ |
| 10 | OPAxIN2+ |
| 01 | OPAxIN1+ |
| 00 | OPAxIN0+ |

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



### 34.5.5 OPAxHWC

**Name:** OPAxHWC
**Offset:** 0x0910

Operational Amplifier Hardware Control Options Register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | OREN | HWCH[2:0] | | | ORPOL | HWCL[2:0] | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 7 – OREN** Override Enable

| Value | Description |
|---|---|
| 1 | Hardware Override Control is enabled. OPA mode of operation is configured using the HWCH / HWCL bits. |
| 0 | Hardware Override Control is disabled. OPA mode of operation must be configured in software. |

**Bits 6:4 – HWCH[2:0]** Hardware Control Configuration High

| Value | Description |
|---|---|
| 111 | Rail Drive to $V_{DD}$ |
| 110 | Reserved |
| 101 | Reserved |
| 100 | Basic OPA configuration with unity-gain feedback |
| 011 | Reserved |
| 010 | Reserved |
| 001 | Reserved |
| 000 | Basic OPA configuration with user-defined feedback |

**Bit 3 – ORPOL** Override Source Polarity

| Value | Description |
|---|---|
| 1 | Hardware Control Input is Inverted (Active-Low) |
| 0 | Hardware Control Input is not Inverted (Active-High) |

**Bits 2:0 – HWCL[2:0]** Hardware Control Configuration Low

| Value | Description |
|---|---|
| 111 | Rail Drive to $V_{SS}$ |
| 110 | Reserved |
| 101 | Reserved |
| 100 | Basic OPA configuration with unity-gain feedback |
| 011 | Reserved |
| 010 | Reserved |
| 001 | Reserved |
| 000 | Basic OPA configuration with user-defined feedback |

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



### 34.5.6 OPAxOFFSET

**Name:** OPAxOFFSET
**Offset:** 0x0911

Operational Amplifier Input Offset Adjustment Register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | OFFSET[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | m | m | m | m | m | m | m | m |

**Bits 7:0 – OFFSET[7:0]** Operational Amplifier Input Offset Calibration



**Important:** If written by the user, the factory calibrated value of this register will be replaced and can only be restored on a Reset.

**Note:** The Reset value ‘m’ is determined by device default locations for that input.

chanhvinh.com



### 34.5.7 OPAxORS

**Name:** OPAxORS
**Offset:** 0x0912

Operational Amplifier Override Source Selection Register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | ORS[4:0] | | | | |
| Access | | | | R/W | R/W | R/W | R/W | R/W |
| Reset | | | | 0 | 0 | 0 | 0 | 0 |

**Bits 4:0 – ORS[4:0]** Operational Amplifier Output Override Source Selection

| Value | Description |
|---|---|
| 11111 - 11011 | Reserved |
| 11010 | CLC4_OUT |
| 11001 | CLC3_OUT |
| 11000 | CLC2_OUT |
| 10111 | CLC1_OUT |
| 10110 | ZCD_OUT |
| 10101 | C2_OUT |
| 10100 | C1_OUT |
| 10011 | NCO1_OUT |
| 10010 | PWM4_S1P2 |
| 10001 | PWM4_S1P1 |
| 10000 | PWM3_S1P2 |
| 01111 | PWM3_S1P1 |
| 01110 | PWM2_S1P2 |
| 01101 | PWM2_S1P1 |
| 01100 | PWM1_S1P2 |
| 01011 | PWM1_S1P1 |
| 01010 | CCP2_OUT |
| 01001 | CCP1_OUT |
| 01000 | TMR6_postscaled_OUT |
| 00111 | TMR4_postscaled_OUT |
| 00110 | TMR3_overflow_OUT |
| 00101 | TMR2_postscaled_OUT |
| 00100 | TMR1_overflow_OUT |
| 00011 | TMR0_overflow_OUT |
| 00010 | SOSC |
| 00001 | LFINTOSC |
| 00000 | OPA1PPS |

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



## 34.6 Register Summary - Operational Amplifier

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 ... 0x090B | Reserved | | | | | | | | | |
| 0x090C | OPA1CON0 | 7:0 | EN | | CPON | | UG | | SOC[1:0] | |
| 0x090D | OPA1CON1 | 7:0 | | GSEL[2:0] | | | RESON | NSS[2:0] | | |
| 0x090E | OPA1CON2 | 7:0 | | NCH[2:0] | | | | PCH[2:0] | | |
| 0x090F | OPA1CON3 | 7:0 | FMS[1:0] | | | | | | PSS[1:0] | |
| 0x0910 | OPA1HWC | 7:0 | OREN | HWCH[2:0] | | | ORPOL | HWCL[2:0] | | |
| 0x0911 | OPA1OFFSET | 7:0 | OFFSET[7:0] | | | | | | | |
| 0x0912 | OPA1ORS | 7:0 | | | | ORS[4:0] | | | | |

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



# 35. DAC - Digital-to-Analog Converter Module

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

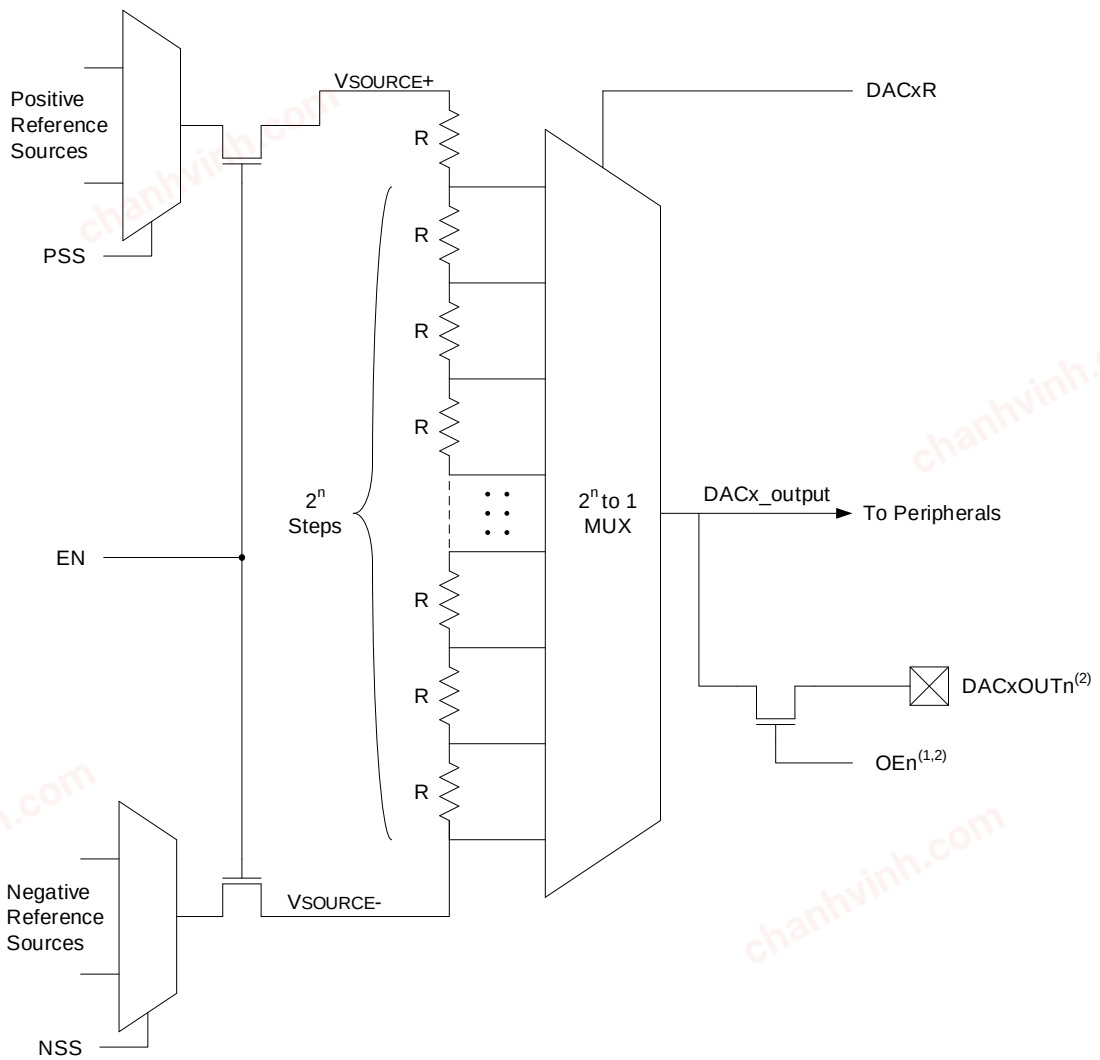
The positive and negative input references (DACxREF+ and DACxREF-) can each be selected from several sources.

The output of the DAC (DAC1OUTx) can be selected as a reference voltage to several other peripherals or routed to output pins.

The Digital-to-Analog Converter (DAC) is enabled by setting the EN bit.

**Important:** This family of devices has two DAC modules. The DAC1 module has a buffered output that can be connected to any of the designated DAC output pins. The DAC2 module has no output pins or buffer, and the output is only connected internally to the ADC, CMP, and OPA modules.

**Figure 35-1. Digital-to-Analog Converter Block Diagram**




**Notes:**
1. The output enable bits are configured so that they act as a “one-hot” system, meaning only one DAC output can be enabled at a time.
2. DAC2 has no output buffer; the output from DAC2 is only connected internally to the specified peripherals.

chanhvinh.com



## 35.1 Output Voltage Selection

The DAC has $2^n$ voltage level ranges, where n is the number of bits in DACR. Each level is determined by the DACxR bits. The DAC output voltage can be determined by using Equation 35-1.

**Equation 35-1. DAC Output Equation**

$$DACx_output = \left( (V_{REF+} - V_{REF-}) \times \frac{DACR}{2^n} \right) + V_{REF-}$$

## 35.2 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 the **"Electrical Specifications"** chapter for each respective device.

## 35.3 Buffered DAC Output Range Selection

The DAC offers selectable output ranges that improve the output performance of the buffered DAC output (DAC1OUTx). Range selection allows module hardware to optimize the DAC buffer output by biasing the reference voltages towards either DACxREF+ (high range) or DACxREF- (low range). Range selection can be done automatically or through software control.

The DAC Buffer Automatic Range Select Enable ($\overline{\text{DACAUTOEN}}$) bit of the Configuration Words is used to select either user software-controlled ranging or automatic ranging via hardware control.

When $\overline{\text{DACAUTOEN}}$ is set ($\overline{\text{DACAUTOEN}}$ = 1), the range is determined by the Buffer Reference Range Selection (REFRNG) bit in user software. When REFRNG is set (REFRNG = 1), the high range (($V_{SS}$ + 1.0V) through $V_{DD}$) is selected as the voltage reference range. When REFRNG is clear (REFRNG = 0), the low range ($V_{SS}$ through ($V_{DD}$ - 1.0V) is selected as the reference range.

When $\overline{\text{DACAUTOEN}}$ is clear ($\overline{\text{DACAUTOEN}}$ = 0), module hardware monitors the DACxDATL register and automatically selects the appropriate range based on the DACxDAT value.

**Important:** To ensure the most accurate results, it is highly recommended to do the following:

- Enable the DAC Auto-Ranging feature in the Configuration Words ($\overline{\text{DACAUTOEN}}$ = 0)
- Set the Charge Pump to Auto mode (CPCONbits.CPON = 'b10)
- Wait the required settling time when changing the DACxDATL values (see the "**Electrical Specifications**" section)

This allows module hardware to continuously monitor the DAC output and $V_{DD}$ levels to ensure a stable, accurate result with little software overhead.

## 35.4 Operation During Sleep

When the device wakes from Sleep through an interrupt or a WWDT Time-out Reset, the contents of the DACxCON and DACxDATL registers are not affected. To minimize current consumption in Sleep mode, the voltage reference will be disabled.

## 35.5 Effects of a Reset

A device Reset affects the following:

chanhvinh.com



- The DAC module is disabled
- The DAC output voltage is removed from the DACxOUTn pin(s)
- The DACxR bits are cleared

## 35.6 Register Definitions: DAC Control

Long bit name prefixes for the DAC are shown in the table below. Refer to the **"Long Bit Names"** section in the **"Register and Bit Naming Conventions"** chapter for more information.

**Table 35-1. DAC Long Bit Name Prefixes**

| Peripheral | Bit Name Prefix |
|---|---|
| DAC1 | DAC1 |
| DAC2 | DAC2 |

chanhvinh.com

chanhvinh.com



### 35.6.1 DACxCON

**Name:** DACxCON
**Offset:** 0x088C

Digital-to-Analog Converter Control Register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | EN | REFRNG | OE[1:0] | | PSS[1:0] | | | NSS |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | | 0 |

**Bit 7 – EN** DAC Enable

| Value | Description |
|---|---|
| 1 | DAC is enabled |
| 0 | DAC is disabled |

**Bit 6 – REFRNG** Buffer Reference Range Selection(1)

| Value | Condition | Description |
|---|---|---|
| x | $\overline{\text{DACAUTOEN}}$ = 0 | Automatic ranging enabled; the REFRNG bit is ignored |
| 1 | $\overline{\text{DACAUTOEN}}$ = 1 | Range optimized for voltages from ($V_{SS}$ + 1.0V) through $V_{DD}$ |
| 0 | $\overline{\text{DACAUTOEN}}$ = 1 | Range optimized for voltages from $V_{SS}$ through ($V_{DD}$ - 1.0V) |

**Bits 5:4 – OE[1:0]** DAC Output Enable

| OE | DAC1 |
|---|---|
| 11 | DAC1OUTx is disabled |
| 10 | DAC1OUT2 is enabled on pin RB7 only |
| 01 | DAC1OUT1 is enabled on pin RA2 only |
| 00 | DAC1OUTx is disabled |

**Bits 3:2 – PSS[1:0]** DAC Positive Reference Selection

| PSS | DAC Positive Reference |
|---|---|
| 11 | OPA1OUT |
| 10 | FVR Buffer 2 |
| 01 | $V_{REF}$+ |
| 00 | $V_{DD}$ |

**Bit 0 – NSS** DAC Negative Reference Selection

| NSS | DAC Negative Reference |
|---|---|
| 1 | $V_{REF}$- |
| 0 | $V_{SS}$ |

**Note:**

1. The REFRNG bit only applies to DAC1OUT1/2.

chanhvinh.com



### 35.6.2 DACxCON

**Name:** DACxCON
**Offset:** 0x0890

**Important:** This instance of the DAC module has no output pins or buffer; the output of this DAC is only connected internally to be used with the Comparator and OPA modules.

Digital-to-Analog Converter Control Register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | EN | | | | PSS[1:0] | | | NSS |
| Access | R/W | | | | R/W | R/W | | R/W |
| Reset | 0 | | | | 0 | 0 | | 0 |

**Bit 7 – EN** DAC Enable

| Value | Description |
|---|---|
| 1 | DAC is enabled |
| 0 | DAC is disabled |

**Bits 3:2 – PSS[1:0]** DAC Positive Reference Selection

| PSS | DAC Positive Reference |
|---|---|
| 11 | OPA1OUT |
| 10 | FVR Buffer 2 |
| 01 | $V_{REF}+$ |
| 00 | $V_{DD}$ |

**Bit 0 – NSS** DAC Negative Reference Selection

| NSS | DAC Negative Reference |
|---|---|
| 1 | $V_{REF}-$ |
| 0 | $V_{SS}$ |

chanhvinh.com



### 35.6.3 DACxDATL

**Name:** DACxDATL
**Offset:** 0x088D

Digital-to-Analog Converter Data Register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | DACxR[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 7:0 – DACxR[7:0]** Data Input Bits for DAC Value

chanhvinh.com



### 35.6.4 DACxDATL

**Name:** DACxDATL
**Offset:** 0x0891

Digital-to-Analog Converter Data Register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | DACxR[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 7:0 – DACxR[7:0]** Data Input Bits for DAC Value

chanhvinh.com



## 35.7 Register Summary - DAC

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 ... 0x088B | Reserved | | | | | | | | | |
| 0x088C | DAC1CON | 7:0 | EN | REFRNG | OE[1:0] | | PSS[1:0] | | | NSS |
| 0x088D | DAC1DATL | 7:0 | DAC1R[7:0] | | | | | | | |
| 0x088E ... 0x088F | Reserved | | | | | | | | | |
| 0x0890 | DAC2CON | 7:0 | EN | | | | PSS[1:0] | | | NSS |
| 0x0891 | DAC2DATL | 7:0 | DAC2R[7:0] | | | | | | | |

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



# 36. CMP - 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.

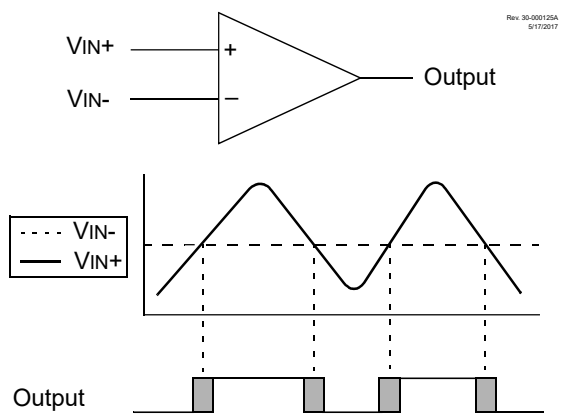
The analog comparator module includes the following features:

- Programmable input selection
- Programmable output polarity
- Rising/falling output edge interrupts
- Wake-up from Sleep
- Selectable voltage reference
- ADC auto-trigger
- Inter-connections with other available modules (e.g., timer clocks)

## 36.1 Comparator Overview

A single comparator is shown in Figure 36-1 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.

**Figure 36-1. Single Comparator**



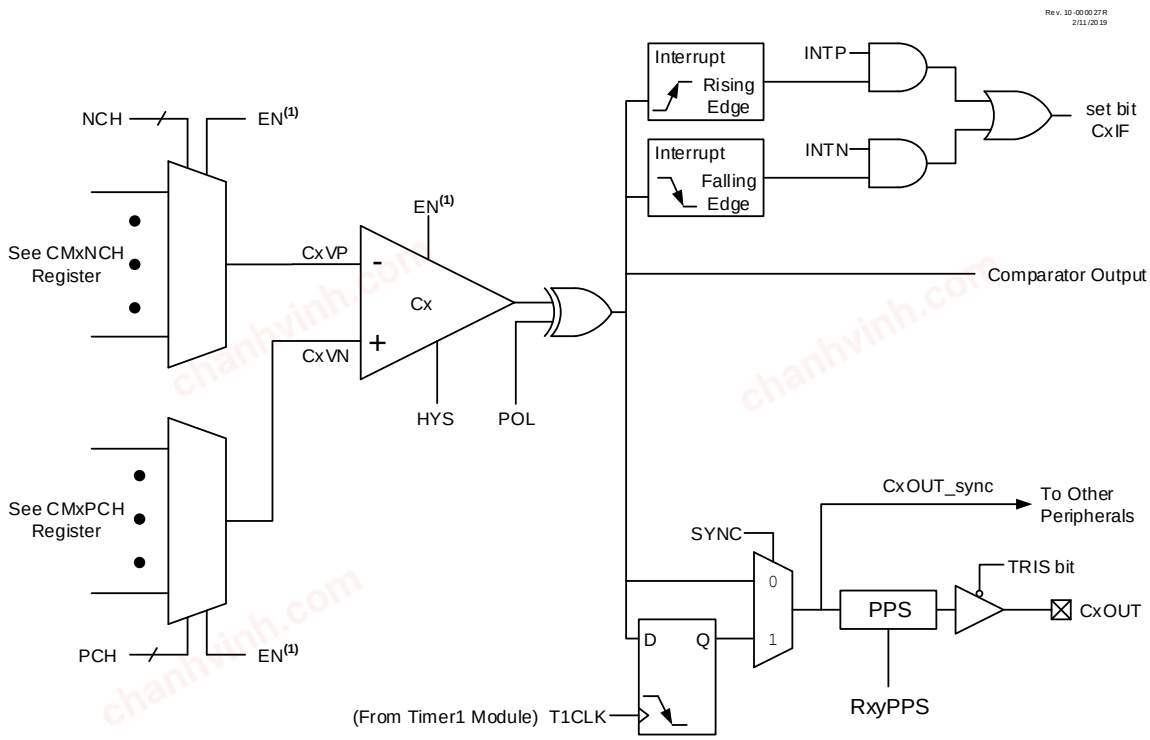

**Note:**

1. The black areas of the output of the comparator represent the uncertainty due to input offsets and response time.

chanhvinh.com



**Figure 36-2. Comparator Module Simplified Block Diagram**




**Note 1:** When EN = 0, all multiplexer inputs are disconnected and the Comparator will produce a '0' at the output.

## 36.2 Comparator Control

Each comparator has two control registers: CMxCON0 and CMxCON1.

The CMxCON0 register contains Control and Status bits for the following:

- Enable
- Output
- Output Polarity
- Hysteresis Enable
- Timer1 Output Synchronization

The CMxCON1 register contains Control bits for the following:

- Interrupt on Positive/Negative Edge Enables

The CMxPCH and CMxNCH registers are used to select the positive and negative input channels, respectively.

### 36.2.1 Comparator Enable

Setting the EN bit enables the comparator for operation. Clearing the EN bit disables the comparator, resulting in minimum current consumption.

### 36.2.2 Comparator Output

The output of the comparator can be monitored in two different registers. Each output can be read individually by reading the OUT bit. Outputs of all the comparators can be collectively accessed by reading the CMOUT register.

chanhvinh.com



The comparator output can also be routed to an external pin through the RxyPPS register. Refer to the **"PPS - Peripheral Pin Select Module"** chapter for more details. The corresponding TRIS bit must be clear to enable the pin as an output.

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

### 36.2.3 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 POL bit. Clearing the POL bit results in a noninverted output. Table 36-1 shows the Output state versus Input conditions, including polarity control.

**Table 36-1. Comparator Output State vs. Input Conditions**

| Input Condition | POL | OUT |
|---|---|---|
| CxVn > CxVp | 0 | 0 |
| CxVn < CxVp | 0 | 1 |
| CxVn > CxVp | 1 | 1 |
| CxVn < CxVp | 1 | 0 |

## 36.3 Comparator Output Synchronization

The output from a comparator can be synchronized with Timer1 by setting the SYNC bit.

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. A simplified block diagram of the comparator module is shown in Figure 36-2. Refer to the **"TMR1 - Timer1 Module with Gate Control"** chapter for more details.

## 36.4 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 HYS bit.

See the **"Comparator Specifications"** section for more information.

## 36.5 Comparator Interrupt

An interrupt can be generated for every rising or falling edge of the comparator output.

When either edge detector is triggered and its associated enable bit is set (INTP and/or INTN bits), the Corresponding Interrupt Flag bit (CxIF bit of the respective PIR register) will be set.

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

- EN bit
- INTP bit (for a rising edge detection)
- INTN bit (for a falling edge detection)
- CxIE bit of the respective PIE register
- GIE bit of the INTCON0 register

The associated interrupt flag bit, CxIF bit of the respective PIR register, must be cleared in software to successfully detect another edge.

chanhvinh.com



**Important:** Although a comparator is disabled, an interrupt will be generated by changing the output polarity with the POL bit.

## 36.6 Comparator Positive Input Selection

Configuring the PCH bits direct an internal voltage reference or an analog pin to the noninverting input of the comparator.

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

## 36.7 Comparator Negative Input Selection

The NCH bits direct an analog input pin, internal reference voltage or analog ground to the inverting input of the comparator.

**Important:** 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.

## 36.8 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 the **"Comparator Specifications"** and **"Fixed Voltage Reference (FVR) Specifications"** sections for more details.

## 36.9 Analog Input Connection Considerations

A simplified circuit for an analog input is shown in Figure 36-3. Since the analog input pins share their connection with a digital input, they have reverse biased ESD protection diodes to $V_{DD}$ and $V_{SS}$. The analog input, therefore, must be between $V_{SS}$ and $V_{DD}$. If the input voltage deviates from this range by more than 0.6V in either direction, one of the diodes is forward biased and abnormal behavior 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, will have very little leakage current to minimize corrupting the result.

**Notes:**

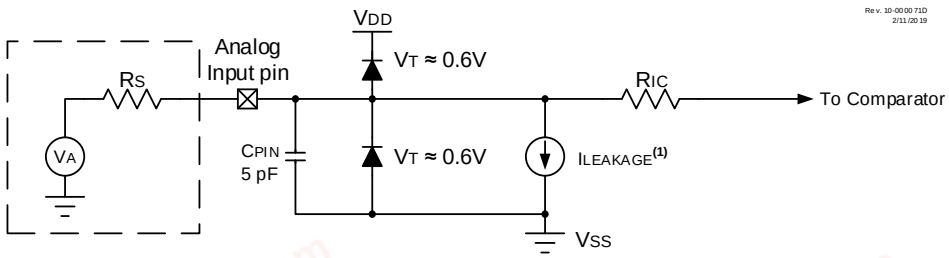
1. When reading a PORT register, all pins configured as analog inputs will read as a '0'. Pins configured as digital inputs will convert as an analog input, according to the input specification.
2. Analog levels on any pin defined as a digital input may cause the input buffer to consume more current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

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



**Figure 36-3. Analog Input Model**




**Legend:** CPIN = Input Capacitance
ILEAKAGE = Leakage Current at the pin due to various junctions
RIC = Interconnect Resistance
RS = Source Impedance
VA = Analog Voltage
VT = Diode Forward Voltage

**Note:**
1. See the **"Electrical Specifications"** chapter.

## 36.10 Operation in Sleep Mode

The comparator module can operate during Sleep. A comparator interrupt will wake the device from Sleep. The CxIE bits of the respective PIE register must be set to enable comparator interrupts.

The comparator clock source is based on the Timer1 clock source. If the Timer1 clock source is either the system clock ($F_{OSC}$) or the instruction clock ($F_{OSC}/4$), Timer1 will not operate during Sleep, and synchronized comparator outputs will not operate.

## 36.11 ADC Auto-Trigger Source

The output of the comparator module can be used to trigger an ADC conversion. When the ADACT register is set to trigger on a comparator output, an ADC conversion will trigger when the comparator output goes high.

## 36.12 Register Definitions: Comparator Control

Long bit name prefixes for the Comparator peripherals are shown in the table below. Refer to the **“Long Bit Names”** section in the **“Register and Bit Naming Conventions”** chapter for more information.

**Table 36-2. Comparator Long Bit Name Prefixes**

| Peripheral | Bit Name Prefix |
|---|---|
| C1 | C1 |
| C2 | C2 |

chanhvinh.com



### 36.12.1 CMxCON0

**Name:** CMxCON0
**Offset:** 0x080C,0x0810

Comparator Control Register 0

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | EN | OUT | | POL | | | HYS | SYNC |
| Access | R/W | R | | R/W | | | R/W | R/W |
| Reset | 0 | 0 | | 0 | | | 0 | 0 |

**Bit 7 – EN** Comparator Enable

| Value | Description |
|---|---|
| 1 | Comparator is enabled |
| 0 | Comparator is disabled and consumes no active power |

**Bit 6 – OUT** Comparator Output

| Value | Condition | Description |
|---|---|---|
| 1 | If POL = 0 (noninverted polarity): | CxVP > CxVN |
| 0 | If POL = 0 (noninverted polarity): | CxVP < CxVN |
| 1 | If POL = 1 (inverted polarity): | CxVP < CxVN |
| 0 | If POL = 1 (inverted polarity): | CxVP > CxVN |

**Bit 4 – POL** Comparator Output Polarity Select

| Value | Description |
|---|---|
| 1 | Comparator output is inverted |
| 0 | Comparator output is not inverted |

**Bit 1 – HYS** Comparator Hysteresis Enable

| Value | Description |
|---|---|
| 1 | Comparator hysteresis enabled |
| 0 | Comparator hysteresis disabled |

**Bit 0 – SYNC** Comparator Output Synchronous Mode

| Value | Description |
|---|---|
| 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 |

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



### 36.12.2 CMxCON1

**Name:** CMxCON1
**Offset:** 0x080D,0x0811

Comparator Control Register 1

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | INTP | INTN |
| Access | | | | | | | R/W | R/W |
| Reset | | | | | | | 0 | 0 |

**Bit 1 – INTP** Comparator Interrupt on Positive-Going Edge Enable

| Value | Description |
|---|---|
| 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 0 – INTN** Comparator Interrupt on Negative-Going Edge Enable

| Value | Description |
|---|---|
| 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 |

chanhvinh.com

chanhvinh.com



### 36.12.3 CMxNCH

**Name:** CMxNCH
**Offset:** 0x080E,0x0812

Comparator Inverting Channel Select Register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | NCH[2:0] | | |
| Access | | | | | | R/W | R/W | R/W |
| Reset | | | | | | 0 | 0 | 0 |

**Bits 2:0 – NCH[2:0]** Comparator Inverting Input Channel Select

| NCH | Negative Input Sources |
|---|---|
| 111 | $V_{SS}$ |
| 110 | FVR_Buffer2 |
| 101 | NCH not connected |
| 100 | NCH not connected |
| 011 | CxIN3- |
| 010 | CxIN2- |
| 001 | CxIN1- |
| 000 | CxIN0- |

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



### 36.12.4 CMxPCH

**Name:** CMxPCH
**Offset:** 0x080F,0x0813

Comparator Noninverting Channel Select Register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | PCH[2:0] | | |
| Access | | | | | | R/W | R/W | R/W |
| Reset | | | | | | 0 | 0 | 0 |

**Bits 2:0 – PCH[2:0]** Comparator Noninverting Input Channel Select

| PCH | Positive Input Sources |
|---|---|
| 111 | $V_{SS}$ |
| 110 | FVR_Buffer2 |
| 101 | PCH not connected |
| 100 | DAC1OUT |
| 011 | PCH not connected |
| 010 | PCH not connected |
| 001 | CxIN1+ |
| 000 | CxIN0+ |

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



### 36.12.5 CMOUT

**Name:** CMOUT
**Offset:** 0x081F

Comparator Output Register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | C2OUT | C1OUT |
| Access | | | | | | | R | R |
| Reset | | | | | | | 0 | 0 |

**Bits 0, 1 – CxOUT** Mirror copy of the CMxCON0.OUT

chanhvinh.com

chanhvinh.com



## 36.13 Register Summary - Comparator

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 ... 0x080B | Reserved | | | | | | | | | |
| 0x080C | CM1CON0 | 7:0 | EN | OUT | | POL | | | HYS | SYNC |
| 0x080D | CM1CON1 | 7:0 | | | | | | | INTP | INTN |
| 0x080E | CM1NCH | 7:0 | | | | | | NCH[2:0] | | |
| 0x080F | CM1PCH | 7:0 | | | | | | PCH[2:0] | | |
| 0x0810 | CM2CON0 | 7:0 | EN | OUT | | POL | | | HYS | SYNC |
| 0x0811 | CM2CON1 | 7:0 | | | | | | | INTP | INTN |
| 0x0812 | CM2NCH | 7:0 | | | | | | NCH[2:0] | | |
| 0x0813 | CM2PCH | 7:0 | | | | | | PCH[2:0] | | |
| 0x0814 ... 0x081E | Reserved | | | | | | | | | |
| 0x081F | CMOUT | 7:0 | | | | | | | C2OUT | C1OUT |

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



# 37. FVR - Fixed Voltage Reference

The Fixed Voltage Reference (FVR) is a stable voltage reference, independent of $V_{DD}$, with 1.024V, 2.048V or 4.096V selectable output levels. The output of the FVR can be configured to supply a reference voltage to analog peripherals such as those listed below.

- ADC input channel
- ADC positive reference
- Comparator input
- Digital-to-Analog Converter (DAC)

The FVR can be enabled by setting the EN bit to '1'.

**Note:** Fixed Voltage Reference output cannot exceed $V_{DD}$.

## 37.1 Independent Gain Amplifiers

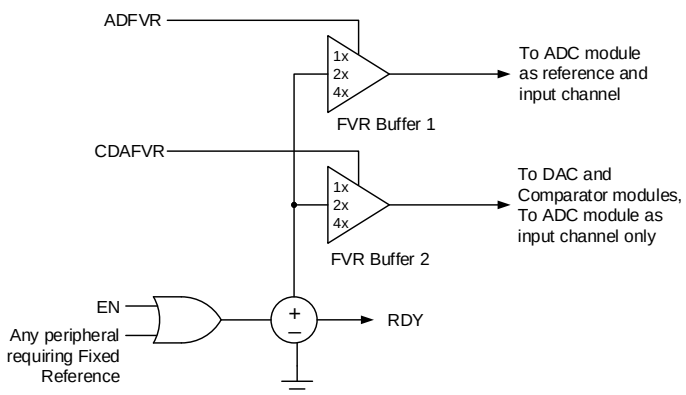
The output of the FVR is routed through two independent programmable gain amplifiers. Each amplifier can be programmed for a gain of 1x, 2x or 4x, to produce the three possible voltage levels.

The ADFVR bits are used to enable and configure the gain amplifier settings for the reference supplied to the ADC module. Refer to the **"ADCC - Analog-to-Digital Converter with Computation Module"** chapter for additional information.

The CDAFVR bits are used to enable and configure the gain amplifier settings for the reference supplied to the DAC and comparator modules. Refer to the **"DAC - Digital-to-Analog Converter Module"** and **"CMP - Comparator Module"** chapters for additional information.

Refer to the figure below for the block diagram of the FVR module.

**Figure 37-1. Fixed Voltage Reference Block Diagram**




## 37.2 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 RDY bit will be set.

## 37.3 Register Definitions: FVR

Long bit name prefixes for the FVR peripherals are shown in the following table. Refer to the **"Long Bit Names"** section in the **"Register and Bits Naming Conventions"** chapter for more information.

chanhvinh.com



**Table 37-1. FVR Long Bit Name Prefixes**

| Peripheral | Bit Name Prefix |
|---|---|
| FVR | FVR |

chanhvinh.com

chanhvinh.com



### 37.3.1 FVRCON

**Name:** FVRCON
**Offset:** 0x020C

FVR Control Register

**Important:** This register is shared between the Fixed Voltage Reference (FVR) module and the temperature indicator module.

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | EN | RDY | TSEN | TSRNG | CDAFVR[1:0] | | ADFVR[1:0] | |
| Access | R/W | R | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | q | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 7 – EN** Fixed Voltage Reference Enable

| Value | Description |
|---|---|
| 1 | Enables module |
| 0 | Disables module |

**Bit 6 – RDY** Fixed Voltage Reference Ready Flag

| Value | Description |
|---|---|
| 1 | Fixed Voltage Reference output is ready for use |
| 0 | Fixed Voltage Reference output is not ready for use or not enabled |

**Bit 5 – TSEN** Temperature Indicator Enable

| Value | Description |
|---|---|
| 1 | Temperature Indicator is enabled |
| 0 | Temperature Indicator is disabled |

**Bit 4 – TSRNG** Temperature Indicator Range Selection

| Value | Description |
|---|---|
| 1 | $V_{OUT} = 3V_T$ (High Range) |
| 0 | $V_{OUT} = 2V_T$ (Low Range) |

**Bits 3:2 – CDAFVR[1:0]** FVR Buffer 2 Gain Selection(1)

| Value | Description |
|---|---|
| 11 | FVR Buffer 2 Gain is 4x, (4.096V)(3) |
| 10 | FVR Buffer 2 Gain is 2x, (2.048V)(3) |
| 01 | FVR Buffer 2 Gain is 1x, (1.024V) |
| 00 | FVR Buffer 2 is OFF |

**Bits 1:0 – ADFVR[1:0]** FVR Buffer 1 Gain Selection(2)

| Value | Description |
|---|---|
| 11 | FVR Buffer 1 Gain is 4x, (4.096V)(3) |
| 10 | FVR Buffer 1 Gain is 2x, (2.048V)(3) |
| 01 | FVR Buffer 1 Gain is 1x, (1.024V) |
| 00 | FVR Buffer 1 is OFF |

**Notes:**

1. This output goes to the DAC and comparator modules, and to the ADC module as an input channel only.
2. This output goes to the ADC module as a reference and an input channel.
3. Fixed Voltage Reference output cannot exceed $V_{DD}$.

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



## 37.4 Register Summary - FVR

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 ... 0x020B | Reserved | | | | | | | | | |
| 0x020C | FVRCON | 7:0 | EN | RDY | TSEN | TSRNG | CDAFVR[1:0] | | ADFVR[1:0] | |

chanhvinh.com

chanhvinh.com



# 38. Temperature Indicator Module

This family of devices is equipped with a temperature circuit designed to measure the operating temperature of the silicon die. The temperature indicator module provides a temperature-dependent voltage that can be measured by the internal Analog-to-Digital Converter.

The circuit's range of operating temperature falls between -40℃ and +125℃. 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.

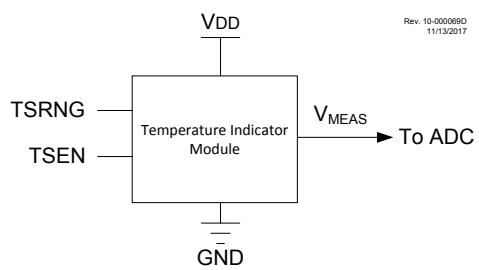
Microchip tests each device during manufacturing and provides the gain, offset, and Temperature Indicator ADC values at 90°C, for both High and Low Range operation.

## 38.1 Module Operation

The temperature indicator module consists of a temperature-sensing circuit that provides a voltage to the device ADC. The analog voltage output varies inversely to the device temperature. The output of the temperature indicator is referred to as $V_{MEAS}$.

The following figure shows a simplified block diagram of the temperature indicator module.

**Figure 38-1. Temperature Indicator Module Block Diagram**




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 the **"ADCC - Analog-to-Digital Converter with Computation Module"** chapter for more details.

The ON/OFF bit for the module is located in the FVRCON register. The circuit is enabled by setting the TSEN bit. When the module is disabled, the circuit draws no current. Refer to the **"FVR - Fixed Reference Voltage"** chapter for more details.

### 38.1.1 Temperature Indicator Range

The temperature indicator circuit operates in either high or low range. The high range, selected by setting the TSRNG bit, provides a wider output voltage. This provides more resolution over the temperature range. High 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. The low range generates a lower sensor voltage and thus, a lower $V_{DD}$ voltage is needed to operate the circuit.

The output voltage of the sensor is the highest value at -40℃ and the lowest value at +125℃.

- **High Range:** The high range is used in applications with the reference for the ADC, $V_{REF}$ = 2.048V. This range may not be suitable for battery-powered applications.
- **Low Range:** This mode is useful in applications in which the $V_{DD}$ is too low for high-range operation. The $V_{DD}$ in this mode can be as low as 1.8V. However, $V_{DD}$ must be at least 0.5V higher than the maximum sensor voltage depending on the expected low operating temperature.

chanhvinh.com



**Important:** The standard parameters for the Temperature Indicator for both high range and low range are stored in the DIA table. Refer to the DIA table in the **"Memory Organization"** chapter for more details. Additionally, the Temperature Indicator sensitivity parameter ($M_V$) for both high range and low range is located in the **"Electrical Specifications"** section.

### 38.1.2 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 the device 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.

The following table shows the recommended minimum $V_{DD}$ vs. Range setting.

**Table 38-1. Recommended $V_{DD}$ vs. Range**

| Min. $V_{DD}$, TSRNG = 1 (High Range) | Min. $V_{DD}$, TSRNG = 0 (Low Range) |
|---|---|
| ≥ 2.5 | ≥ 1.8 |

## 38.2 Temperature Calculation

This section describes the steps involved in calculating the die temperature, $T_{MEAS}$:

1. Obtain the ADC count value of the measured analog voltage: The analog output voltage, $V_{MEAS}$, is converted to a digital count value by the Analog-to-Digital Converter (ADC) and is referred to as $ADC_{MEAS}$.
2. Obtain the Gain value from the DIA table. This parameter is TSLR1 for the low range setting or TSHR1 for the high range setting of the temperature indicator module. Refer to the DIA table in the **"Memory Organization"** chapter for more details.
3. Obtain the Offset value from the DIA table. This parameter is TSLR3 for the low range setting or TSHR3 for the high range setting of the temperature indicator module. Refer to the DIA table in the **"Memory Organization"** chapter for more details.

The following equation provides an estimate for the die temperature based on the above parameters:

**Equation 38-1. Sensor Temperature (in ℃)**

$$T_{MEAS} = \frac{\frac{(ADC_{MEAS} \times Gain)}{256} + Offset}{10}$$

Where:
$ADC_{MEAS}$ = ADC reading at temperature being estimated

Gain = Gain value stored in the DIA table

Offset = Offset value stored in the DIA table

**Note:** It is recommended to take the average of ten measurements of $ADC_{MEAS}$ to reduce noise and improve accuracy.

**Example 38-1. Temperature Calculation (C)**

```
// offset is int16_t data type
// gain is int16_t data type
// ADC_MEAS is uint16_t data type
// Temp_in_C is int24_t data type

ADC_MEAS = ((ADRESH << 8) + ADRESL);      // Store the ADC Result
Temp_in_C = (int24_t)(ADC_MEAS) * gain;   // Multiply the ADC Result by
                                          // Gain and store the result
                                          // in a signed variable
Temp_in_C = Temp_in_C / 256;              // Divide (ADC Result * Gain) by 256
Temp_in_C = Temp_in_C + offset;           // Add (Offset) to the result
```

chanhvinh.com



```
Temp_in_C = Temp_in_C / 10;                // Divide the result by 10 and store
                                           // the calculated temperature
```

**Important:** If the application requires more precise temperature measurement, additional calibrations steps will be necessary. For these applications, two-point or three-point calibration is recommended. For additional information on two-point calibration method, refer to the following Microchip application note, available at the corporate website (www.microchip.com):

- AN2798, *"Using the PIC16F/PIC18F Ground Referenced Temperature Indicator Module"*

## 38.3 ADC Acquisition Time

To ensure accurate temperature measurements, the user must wait a certain minimum acquisition time (parameter TS01) after the temperature indicator output is selected as ADC input. This is required for the ADC sampling circuit to settle before the conversion is performed.

**Note:** Parameter TS01 can be found in the Temperature Indicator Requirements table of the **"Electrical Specifications"** chapter.

## 38.4 Register Definitions: Temperature Indicator

chanhvinh.com



### 38.4.1 FVRCON

**Name:** FVRCON
**Offset:** 0x020C

FVR Control Register

**Important:** This register is shared between the Fixed Voltage Reference (FVR) module and the temperature indicator module.

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | EN | RDY | TSEN | TSRNG | CDAFVR[1:0] | | ADFVR[1:0] | |
| Access | R/W | R | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | q | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 7 – EN** Fixed Voltage Reference Enable

| Value | Description |
|---|---|
| 1 | Enables module |
| 0 | Disables module |

**Bit 6 – RDY** Fixed Voltage Reference Ready Flag

| Value | Description |
|---|---|
| 1 | Fixed Voltage Reference output is ready for use |
| 0 | Fixed Voltage Reference output is not ready for use or not enabled |

**Bit 5 – TSEN** Temperature Indicator Enable

| Value | Description |
|---|---|
| 1 | Temperature Indicator is enabled |
| 0 | Temperature Indicator is disabled |

**Bit 4 – TSRNG** Temperature Indicator Range Selection

| Value | Description |
|---|---|
| 1 | $V_{OUT} = 3V_T$ (High Range) |
| 0 | $V_{OUT} = 2V_T$ (Low Range) |

**Bits 3:2 – CDAFVR[1:0]** FVR Buffer 2 Gain Selection(1)

| Value | Description |
|---|---|
| 11 | FVR Buffer 2 Gain is 4x, (4.096V)(3) |
| 10 | FVR Buffer 2 Gain is 2x, (2.048V)(3) |
| 01 | FVR Buffer 2 Gain is 1x, (1.024V) |
| 00 | FVR Buffer 2 is OFF |

**Bits 1:0 – ADFVR[1:0]** FVR Buffer 1 Gain Selection(2)

| Value | Description |
|---|---|
| 11 | FVR Buffer 1 Gain is 4x, (4.096V)(3) |
| 10 | FVR Buffer 1 Gain is 2x, (2.048V)(3) |
| 01 | FVR Buffer 1 Gain is 1x, (1.024V) |
| 00 | FVR Buffer 1 is OFF |

**Notes:**

1. This output goes to the DAC and comparator modules, and to the ADC module as an input channel only.
2. This output goes to the ADC module as a reference and an input channel.
3. Fixed Voltage Reference output cannot exceed $V_{DD}$.

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



## 38.5 Register Summary - Temperature Indicator

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 ... 0x020B | Reserved | | | | | | | | | |
| 0x020C | FVRCON | 7:0 | EN | RDY | TSEN | TSRNG | CDAFVR[1:0] | | ADFVR[1:0] | |

chanhvinh.com

chanhvinh.com

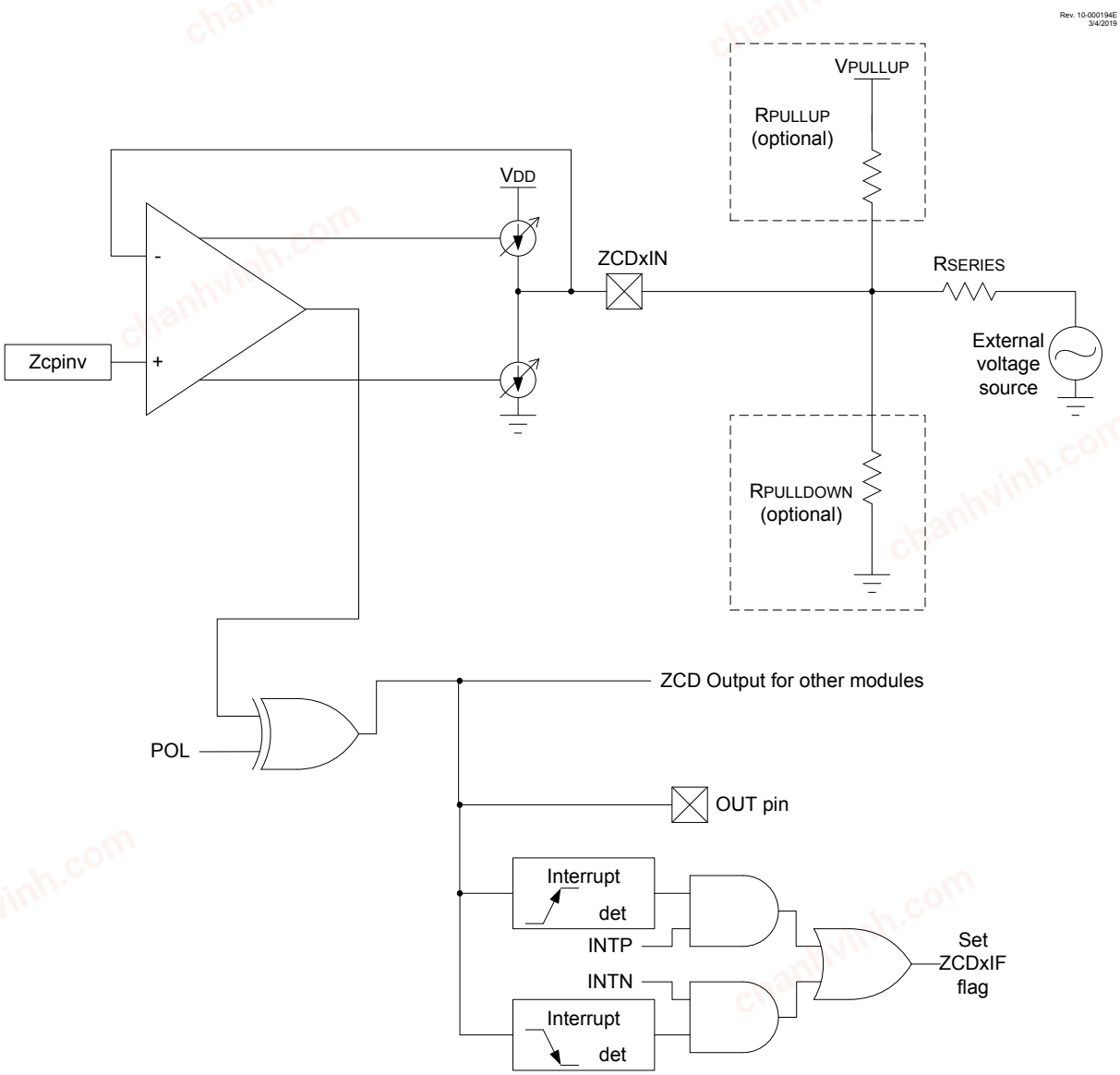


# 39. ZCD - Zero-Cross Detection Module

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, $Z_{CPINV}$, 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 following simplified block diagram.

**Figure 39-1. Simplified ZCD Block Diagram**




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

chanhvinh.com



- Dimmer phase delayed drive
- Low EMI cycle switching

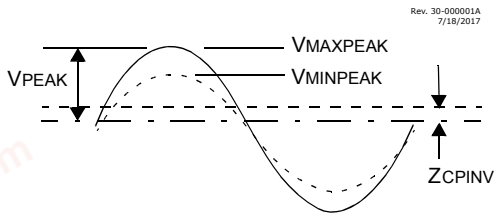
## 39.1 External Resistor Selection

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 less than the maximum input current (ZC02). Refer to the **"Electrical Specifications"** chapter for more details. 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 39-1. External Resistor**

$$R_{SERIES} = \frac{V_{PEAK}}{I_{ZCD}}$$

**Figure 39-2. External Voltage Source**




## 39.2 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 OUT bit is set when the current sink is active, and cleared when the current source is active. The OUT bit is affected by the polarity bit.

The OUT signal can also be used as input to other modules. This is controlled by the registers of the corresponding module.

## 39.3 ZCD Logic Polarity

The POL bit inverts the OUT bit relative to the current source and sink output. When the POL bit is set, a OUT high indicates that the current source is active, and a low output indicates that the current sink is active. The POL bit affects the ZCD interrupts.

## 39.4 ZCD Interrupts

An interrupt will be generated upon a change in the ZCD logic output when the appropriate interrupt enables are set. The ZCD module has a rising edge detector and a falling edge detector.

The ZCDIF bit of the PIRx register will be set when either edge detector is triggered and its associated enable bit is set. The INTP enables rising edge interrupts and the INTN bit enables falling edge interrupts.

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

- ZCDIE bit of the PIEx register
- INTP bit for rising edge detection
- INTN bit for falling edge detection
- GIEL and GIE bits of the INTCON0 register

Changing the POL bit will cause an interrupt, regardless of the level of the SEN bit.

The ZCDIF bit of the PIRx 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.

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



## 39.5 Correction for $Z_{CPINV}$ Offset

The actual voltage at which the ZCD switches is the reference voltage at the noninverting 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.

### 39.5.1 Correction by AC Coupling

When the external voltage source is sinusoidal, the effects of the $Z_{CPINV}$ 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 less than the maximum input current (ZC02). Refer to the **"Electrical Specifications"** chapter for more details. Next, arbitrarily select a suitably large nonpolar capacitor and compute its reactance, Xc, at the external voltage source frequency. Finally, compute the series resistor, capacitor peak voltage, and phase shift using the formulas shown below.

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 $V_{DD}$ or GND with a high-impedance resistor.

**Note:** In this example, the impedance value is calculated for a peak current of 300 µA.

**Equation 39-2. R-C Equations**

$V_{PEAK}$ = 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

$T_\Phi$ = time ZC event occurs before actual zero-crossing

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

$$X_C = \frac{1}{2\pi f C}$$

$$R = \sqrt{Z^2 - {X_C}^2}$$

$$V_C = X_C\left(3 \times 10^{-4}\right)$$

$$\Phi = \tan^{-1}\left(\frac{X_C}{R}\right)$$

$$T_\Phi = \frac{\Phi}{2\pi f}$$

**Equation 39-3. R-C Calculation Example**

$$V_{rms} = 120$$

$$V_{PEAK} = V_{rms} \times \sqrt{2} = 169.7$$

$$f = 60\ Hz$$

chanhvinh.com



$C = 0.1\ \mu F$

$$Z = \frac{V_{PEAK}}{3 \times 10^{-4}} = \frac{169.7}{3 \times 10^{-4}} = 565.7\ k\Omega$$

$$X_C = \frac{1}{2\pi f C} = \frac{1}{2\pi \times 60 \times 10^{-7}} = 26.53\ k\Omega$$

$$R = \sqrt{Z^2 - {X_C}^2} = 565.1\ k\Omega\ (computed)$$

$$R_a = 560\ k\Omega\ (used)$$

$$Z_R = \sqrt{{R_a}^2 + {X_C}^2} = 560.6\ k\Omega$$

$$I_{PEAK} = \frac{V_{PEAK}}{Z_R} = 302.7 \times 10^{-6} A$$

$$V_C = X_C \times I_{PEAK} = 8.0\ V$$

$$\Phi = \tan^{-1}\left(\frac{X_C}{R}\right) = 0.047\ radians$$

$$T_\Phi = \frac{\Phi}{2\pi f} = 125.6\ \mu s$$

### 39.5.2 Correction by Offset Current

When the waveform is varying relative to $V_{SS}$, 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 $V_{DD}$, 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 below.

**Equation 39-4. ZCD Event Offset**

When External Voltage source is relative to $V_{SS}$:

$$T_{offset} = \frac{\sin^{-1}\left(\frac{Z_{CPINV}}{V_{PEAK}}\right)}{2\pi f}$$

When External Voltage source is relative to $V_{DD}$:

$$T_{offset} = \frac{\sin^{-1}\left(\frac{V_{DD} - Z_{CPINV}}{V_{PEAK}}\right)}{2\pi f}$$

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 $Z_{CPINV}$ switching voltage. The pull-up or pull-down value can be determined with the equations shown below.

**Equation 39-5. ZCD Pull-up/Pull-down Resistor**

When External Voltage source is relative to $V_{SS}$:

$$R_{pullup} = \frac{R_{SERIES}\left(V_{pullup} - Z_{CPINV}\right)}{Z_{CPINV}}$$

When External Voltage source is relative to $V_{DD}$:

chanhvinh.com



$$R_{pulldown} = \frac{R_{SERIES}(Z_{CPINV})}{(V_{DD} - Z_{CPINV})}$$

## 39.6 Handling $V_{PEAK}$ 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 specified by ZC02 and above a reasonable minimum range depending on the application. The compensating pull-up for this series resistance can be determined with the equations shown in Equation 39-5 because the pull-up value is independent from the peak voltage.

**Tip:** It is recommended that the maximum peak voltage be no more than six times the minimum peak voltage.

## 39.7 Operation During Sleep

The ZCD current sources and interrupts are unaffected by Sleep.

## 39.8 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 $\overline{ZCD}$ Configuration bit is cleared, the ZCD circuit will be active at POR. When the $\overline{ZCD}$ Configuration bit is set, the SEN bit must be set to enable the ZCD module.

## 39.9 Disabling the ZCD Module

The ZCD module can be disabled in two ways:

1. The $\overline{ZCD}$ Configuration bit disables the ZCD module when set. When this is the case, then the ZCD module will be enabled by setting the SEN bit. When the $\overline{ZCD}$ bit is clear, the ZCD is always enabled and the SEN bit has no effect.
2. The ZCD can also be disabled using the ZCDMD bit of the PMDx register. This is subject to the status of the $\overline{ZCD}$ bit.

## 39.10 Register Definitions: ZCD Control

Long bit name prefixes for the ZCD peripherals are shown in the table below. Refer to the **"Long Bit Names"** section of the **"Register and Bit Naming Conventions"** chapter for more information.

**Table 39-1. ZCD Long Bit Name Prefixes**

| Peripheral | Bit Name Prefix |
|---|---|
| ZCD | ZCD |

chanhvinh.com



### 39.10.1 ZCDCON

**Name:** ZCDCON
**Offset:** 0x021F

Zero-Cross Detect Control Register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SEN | | OUT | POL | | | INTP | INTN |
| Access | R/W | | R | R/W | | | R/W | R/W |
| Reset | 0 | | x | 0 | | | 0 | 0 |

**Bit 7 – SEN** Zero-Cross Detect Software Enable
This bit is ignored when the ZCD fuse is cleared.

| Value | Condition | Description |
|---|---|---|
| X | ZCD Config fuse = 0 | Zero-cross detect is always enabled. This bit is ignored. |
| 1 | ZCD Config fuse = 1 | Zero-cross detect is enabled. ZCD pin is forced to output to source and sink current. |
| 0 | ZCD Config fuse = 1 | Zero-cross detect is disabled. ZCD pin operates according to PPS and TRIS controls. |

**Bit 5 – OUT** Zero-Cross Detect Data Output

| Value | Condition | Description |
|---|---|---|
| 1 | POL = 0 | ZCD pin is sinking current |
| 0 | POL = 0 | ZCD pin is sourcing current |
| 1 | POL = 1 | ZCD pin is sourcing current |
| 0 | POL = 1 | ZCD pin is sinking current |

**Bit 4 – POL** Zero-Cross Detect Polarity

| Value | Description |
|---|---|
| 1 | ZCD logic output is inverted |
| 0 | ZCD logic output is not inverted |

**Bit 1 – INTP** Zero-Cross Detect Positive-Going Edge Interrupt Enable

| Value | Description |
|---|---|
| 1 | The ZCDIF bit is set on low-to-high ZCD_output transition |
| 0 | The ZCDIF bit is unaffected by low-to-high ZCD_output transition |

**Bit 0 – INTN** Zero-Cross Detect Negative-Going Edge Interrupt Enable

| Value | Description |
|---|---|
| 1 | The ZCDIF bit is set on high-to-low ZCD_output transition |
| 0 | The ZCDIF bit is unaffected by high-to-low ZCD_output transition |

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



## 39.11 Register Summary - ZCD

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 ... 0x021E | Reserved | | | | | | | | | |
| 0x021F | ZCDCON | 7:0 | SEN | | OUT | POL | | | INTP | INTN |

chanhvinh.com

chanhvinh.com



# 40. Charge Pump

This family of devices offers a dedicated charge pump, which is controlled through the Charge Pump Control (CPCON) register. The primary purpose of the charge pump is to supply a constant voltage to the gates of transistor devices contained in analog peripherals, signal and reference input pass-gates, and to prevent degradation of transistor performance at low operating voltages.

The charge pump offers the following modes:

- Manually Enabled
- Automatically Enabled
- Disabled

## 40.1 Manually Enabled

The charge pump can be manually enabled via the Charge Pump Enable (CPON) bits. When the CPON bits are configured as '11', the charge pump is enabled. In this case, the charge pump provides additional voltage to all analog systems, regardless of $V_{DD}$ levels, but also consumes additional current.

## 40.2 Automatically Enabled

The charge pump can also be enabled automatically. This allows the application to determine when to enable the charge pump. If the charge pump is enabled while $V_{DD}$ levels are above a sufficient threshold, the charge pump does not improve analog performance, but also consumes additional current. Allowing hardware to monitor $V_{DD}$ and determine when to enable the charge pump prevents unnecessary current consumption.

When the CPON bits are configured as '10', charge pump hardware monitors $V_{DD}$ and compares the $V_{DD}$ levels to a reference voltage threshold ($V_{AUTO}$), which is set to 4.6V. When hardware detects a $V_{DD}$ level lower than the threshold, the charge pump is automatically enabled. If $V_{DD}$ returns to a level above the threshold, hardware automatically disables the charge pump.

When the CPON bits are configured as '01', charge pump hardware waits for an analog peripheral, such as the ADC, to be enabled before monitoring $V_{DD}$. In this case, charge pump hardware monitors all analog peripherals, and once an analog peripheral is enabled, hardware begins to compare $V_{DD}$ to $V_{AUTO}$. When hardware detects a $V_{DD}$ level lower than the threshold, hardware enables the charge pump. If $V_{DD}$ returns to a level above the threshold, or if the analog peripheral is disabled, the charge pump is automatically disabled.

## 40.3 Disabled

The charge pump is disabled by default (CPON = 00). Clearing the CPON bits will disable the charge pump.

## 40.4 Charge Pump Oscillator

The Charge Pump Oscillator Selection (CPOS) bit selects the charge pump oscillator source. The CPOS bit allows the user to select between the charge pump's internal oscillator or the oscillator driving the ADC.

When CPOS is set (CPOS = 1), the charge pump utilizes its internal oscillator. The charge pump's internal oscillator provides a very steady output voltage, but at the expense of higher operating current.

When CPOS is clear (CPOS = 0), and the ADGO bit is clear (GO = 0), the charge pump is clock by the ADCRC. When ADGO is set (GO = 1), the charge pump is clocked by a derivative of the $F_{OSC}$ (as determined by the ADCLK register). This allows the charge pump to operate at a lower current when the ADC is not converting, while offering higher performance when the ADC is converting.

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



## 40.5 Charge Pump Threshold

The Charge Pump Threshold (CPT) bit indicates whether or not $V_{DD}$ is at an acceptable operating level. Charge pump hardware compares $V_{DD}$ to the threshold voltage ($V_{AUTO}$), which is set at 4.6V. If $V_{DD}$ is above $V_{AUTO}$, the CPT bit is set (CPT = 1). If $V_{DD}$ is below $V_{AUTO}$, CPT is clear (CPT = 0).

## 40.6 Charge Pump Ready

The Charge Pump Ready Status (CPRDY) bit indicates whether or not the charge pump is ready for use. When CPRDY is set (CPRDY = 1), the charge pump has reached a steady-state operation and is ready for use. When CPRDY is clear (CPRDY = 0), the charge pump is either in the OFF state or has not reached a steady-state operation.

## 40.7 Register Definitions: Charge Pump

chanhvinh.com

chanhvinh.com



### 40.7.1 CPCON

**Name:** CPCON
**Offset:** 0x020D

Charge Pump Control Register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CPON[1:0] | | CPOS | | | CPREQ | CPT | CPRDY |
| Access | R/W | R/W | R/W | | | R | R | R |
| Reset | 0 | 0 | 0 | | | 0 | 0 | 0 |

**Bits 7:6 – CPON[1:0]**

Charge Pump Enable

| Value | Description |
|---|---|
| 11 | Charge pump is enabled |
| 10 | Charge pump is automatically enabled when $V_{DD} < V_{AUTO}$ ($V_{AUTO}$ = 4.6V) |
| 01 | Charge pump is automatically enabled when an analog peripheral is enabled (ADCON0.ON = 1) and $V_{DD} < V_{AUTO}$ |
| 00 | Charge pump is disabled |

**Bit 5 – CPOS**

Charge Pump Oscillator Selection

| Value | Condition | Description |
|---|---|---|
| 1 | | Charge pump clock is the internal charge pump oscillator |
| 0 | When ADC GO bit = 1 | Charge pump clock is FOSC (oscillator frequency determined by ADCLK) |
| 0 | When ADC GO bit = 0 | Charge pump clock is the ADCRC |

**Bit 2 – CPREQ**

Charge Pump Request Status

| Value | Description |
|---|---|
| 1 | Charge pump has been requested by an analog peripheral |
| 0 | Charge pump has not been requested by an analog peripheral |

**Bit 1 – CPT**

Charge Pump Threshold

| Value | Description |
|---|---|
| 1 | $V_{DD}$ is above the charge pump auto-enable threshold ($V_{AUTO}$) |
| 0 | $V_{DD}$ is below the charge pump auto-enable threshold ($V_{AUTO}$) |

**Bit 0 – CPRDY**

Charge Pump Ready Status

| Value | Description |
|---|---|
| 1 | Charge pump has reached a steady-state operation |
| 0 | Charge pump is off or has not reached a steady-state operation |

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



## 40.8 Register Summary - Charge Pump

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00<br>...<br>0x020C | Reserved | | | | | | | | | |
| 0x020D | CPCON | 7:0 | CPON[1:0] | | CPOS | | | CPREQ | CPT | CPRDY |

chanhvinh.com

chanhvinh.com



# 41. Instruction Set Summary

The PIC16F171 devices incorporate the standard set of 50 PIC16 core instructions. 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 41-3 lists the instructions recognized by the XC8 assembler.

All instructions are executed within a single instruction cycle, with the following exceptions, which may take two or three cycles:

- Subroutine entry 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 four 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.

## 41.1 Read-Modify-Write Operations

Any instruction that specifies a file register as part of the instruction performs a Read-Modify-Write (RMW) operation. The register is read, the data is modified, and the result is stored according to either the Working (W) register, or the originating file register, depending on the state of the destination designator 'd' (see Table 41-1 for more information). A read operation is performed on a register even if the instruction writes to that register.

**Table 41-1. 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`).<br>The assembler will generate code with x = `0`.<br>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. |
| n | FSR or INDF number. (0-1) |
| mm | Pre/post increment/decrement mode selection |

**Table 41-2. Abbreviation Descriptions**

| Field | Description |
|---|---|
| PC | Program Counter |

chanhvinh.com



.....continued

| Field | Description |
|---|---|
| $\overline{TO}$ | Time-Out bit |
| C | Carry bit |
| DC | Digit Carry bit |
| Z | Zero bit |
| $\overline{PD}$ | Power-Down bit |

## 41.2 Standard Instruction Set

**Table 41-3. Instruction Set**

| Mnemonic, Operands | | Description | Cycles | 14-Bit Opcode | | | | Status Affected | Notes |
|---|---|---|---|---|---|---|---|---|---|
| | | | | MSb | | | LSb | | |
| **BYTE-ORIENTED OPERATIONS** | | | | | | | | | |
| ADDWF | f, d | Add WREG and f | 1 | 00 | 0111 | dfff | ffff | C, DC, Z | 2 |
| ADDWFC | f, d | Add WREG and Carry bit to f | 1 | 11 | 1101 | dfff | ffff | C, DC, Z | 2 |
| ANDWF | f, d | AND WREG 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 |
| CLRW | – | Clear WREG | 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 WREG with f | 1 | 00 | 0100 | dfff | ffff | Z | 2 |
| MOVF | f, d | Move f | 1 | 00 | 1000 | dfff | ffff | Z | 2 |
| MOVWF | f | Move WREG to f | 1 | 00 | 0000 | 1fff | ffff | None | 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 |

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



.....continued

| Mnemonic, Operands | | Description | Cycles | 14-Bit Opcode | | | | Status Affected | Notes |
|---|---|---|---|---|---|---|---|---|---|
| | | | | MSb | | | LSb | | |
| SUBWF | f, d | Subtract WREG from f | 1 | 00 | 0010 | dfff | ffff | C, DC, Z | 2 |
| SUBWFB | f, d | Subtract WREG from f with Borrow | 1 | 11 | 1011 | dfff | ffff | C, DC, Z | 2 |
| SWAPF | f, d | Swap nibbles in f | 1 | 00 | 1110 | dfff | ffff | None | 2 |
| XORWF | f, d | Exclusive OR WREG 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 | None | 1, 2 |
| INCFSZ | f, d | Increment f, Skip if 0 | 1(2) | 00 | 1111 | dfff | ffff | None | 1, 2 |
| **BIT-ORIENTED FILE REGISTER OPERATIONS** | | | | | | | | | |
| BCF | f, b | Bit Clear f | 1 | 01 | 00bb | bfff | ffff | None | 2 |
| BSF | f, b | Bit Set f | 1 | 01 | 01bb | bfff | ffff | None | 2 |
| **BIT-ORIENTED SKIP OPERATIONS** | | | | | | | | | |
| BTFSC | f, b | Bit Test f, Skip if Clear | 1(2) | 01 | 10bb | bfff | ffff | None | 1, 2 |
| BTFSS | f, b | Bit Test f, Skip if Set | 1(2) | 1010 | 11bb | bfff | ffff | None | 1, 2 |
| **LITERAL OPERATIONS** | | | | | | | | | |
| ADDLW | k | Add literal and WREG | 1 | 11 | 1110 | kkkk | kkkk | C, DC, Z | |
| ANDLW | k | AND literal with WREG | 1 | 11 | 1001 | kkkk | kkkk | Z | |
| IORLW | k | Inclusive OR literal with WREG | 1 | 11 | 1000 | kkkk | kkkk | Z | |
| MOVLB | k | Move literal to BSR | 1 | 00 | 000 | 0k | kkkk | None | |
| MOVLP | k | Move literal to PCLATH | 1 | 11 | 0001 | 1kkk | kkkk | None | |
| MOVLW | k | Move literal to W | 1 | 11 | 0000 | kkkk | kkkk | None | |
| 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 | |
| **CONTROL OPERATIONS** | | | | | | | | | |

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



.....continued

| Mnemonic, Operands | | Description | Cycles | 14-Bit Opcode | | | | Status Affected | Notes |
|---|---|---|---|---|---|---|---|---|---|
| | | | | MSb | | | LSb | | |
| BRA | k | Relative Branch | 2 | 11 | 001k | kkkk | kkkk | None | |
| BRW | — | Relative Branch with WREG | 2 | 00 | 0000 | 0000 | 1011 | None | |
| CALL | k | Call Subroutine | 2 | 10 | 0kkk | kkkk | kkkk | None | |
| CALLW | — | Call Subroutine with WREG | 2 | 00 | 0000 | 0000 | 1010 | None | |
| GOTO | k | Go to address | 2 | 10 | 1kkk | kkkk | kkkk | None | |
| RETFIE | k | Return from interrupt | 2 | 00 | 0000 | 0000 | 1001 | None | |
| RETLW | k | Return with literal in WREG | 2 | 11 | 0100 | kkkk | kkkk | None | |
| RETURN | — | Return from Subroutine | 2 | 00 | 0000 | 0000 | 1000 | None | |
| **INHERENT OPERATIONS** | | | | | | | | | |
| CLRWDT | — | Clear Watchdog Timer | 1 | 00 | 0000 | 0110 | 0100 | $\overline{TO}$, $\overline{PD}$ | |
| NOP | — | No Operation | 1 | 00 | 0000 | 0000 | 0000 | None | |
| RESET | — | Software device Reset | 1 | 00 | 0000 | 0000 | 0001 | None | |
| SLEEP | — | Go into Standby mode | 1 | 00 | 0000 | 0110 | 0011 | $\overline{TO}$, $\overline{PD}$ | |
| TRIS | f | Load TRIS register with WREG | 1 | 00 | 0000 | 0110 | 0fff | None | |
| **C-COMPILER OPTIMIZED** | | | | | | | | | |
| ADDFSR | n, k | Add Literal k to FSRn | 1 | 11 | 0001 | 0nkk | kkkk | None | |
| MOVIW | n, mm | Move Indirect FSRn to WREG with pre/post inc/dec modifier, mm | 1 | 00 | 0000 | 0001 | 0nmm | Z | 2, 3 |
| | k[n] | Move INDFn to WREG, Indexed Indirect. | 1 | 11 | 1111 | 0nkk | kkkk | Z | 2 |
| MOVWI | n, mm | Move WREG to Indirect FSRn with pre/post inc/dec modifier, mm | 1 | 00 | 0000 | 0001 | 1nmm | None | 2, 3 |
| | k[n] | Move WREG to INDFn, Indexed Indirect. | 1 | 11 | 1111 | 1nkk | kkkk | None | 2 |

**Notes:**

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. Details on `MOVIW` and `MOVWI` instruction descriptions are available in the next section.

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



### 41.2.1 Standard Instruction Set

| ADDFSR | Add Literal to FSRn |
|---|---|
| Syntax: | [ *label* ] ADDFSR FSRn, k |
| Operands: | -32 ≤ k ≤ 31;<br>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.<br>FSRn is limited to the range `0000h-FFFFh`. Moving beyond these bounds will cause the FSR to wrap-around. |

| ADDLW | Add Literal to W |
|---|---|
| Syntax: | [ *label* ] ADDLW k |
| Operands: | 0 ≤ k ≤ 255 |
| Operation: | (W) + k → (W) |
| Status Affected: | C, DC, Z |
| Description: | The contents of W are added to the 8-bit literal ‘k’ and the result is placed in W. |

| ADDWF | Add W to f |
|---|---|
| Syntax: | [ *label* ] ADDWF f, d |
| Operands: | 0 ≤ f ≤ 127<br>d ∈ [0,1] |
| Operation: | (W) + (f) → dest |
| Status Affected: | C, DC, Z |
| Description: | Add the contents of the W register with register ‘f’. If ‘d’ is ‘`0`’, the result is stored in the W register. If ‘d’ is ‘`1`’, the result is stored back in register ‘f’. |

| ADDWFC | Add W and Carry Bit to f |
|---|---|
| Syntax: | [ *label* ] ADDWFC f {,d} |
| Operands: | 0 ≤ f ≤ 127<br>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.<br>If ‘d’ is ‘`1`’, the result is placed in data memory location ‘f’. |

| ANDLW | AND Literal with W |
|---|---|
| Syntax: | [ *label* ] ANDLW k |
| Operands: | 0 ≤ k ≤ 255 |
| Operation: | (W) .AND. k → (W) |

© 2023 Microchip Technology Inc.
and its subsidiaries

chanhvinh.com



| .....continued | |
|---|---|
| **ANDLW** | **AND Literal with W** |
| Status Affected: | Z |
| Description: | The contents of W are AND'ed with the 8-bit literal 'k'.<br>The result is placed in W. |

| ANDWF | AND W with f |
|---|---|
| Syntax: | [ *label* ] ANDWF f, d |
| Operands: | 0 ≤ f ≤ 127<br>d ∈ [0,1] |
| Operation: | (W) .AND. (f) → dest |
| Status Affected: | Z |
| Description: | AND the W register with register 'f'. If 'd' is '0', the result is stored in the W register.<br>If 'd' is '1', the result is stored back in register 'f'. |

| ASRF | Arithmetic Right Shift |
|---|---|
| Syntax: | [ *label* ] ASRF f, d |
| Operands: | 0 ≤ f ≤ 127<br>d ∈ [0,1] |
| Operation: | (f[7]) → dest[7]<br>(f[7:1]) → dest[6:0]<br>(f[0]) → C |
| Status Affected: | C, Z |
| Description: | The contents of register 'f' are shifted one bit to the right through the Carry flag.<br>The MSb remains unchanged.<br>If 'd' is '0', the result is placed in W.<br>If 'd' is '1', the result is stored back in register 'f'.<br>**Register f → C** |

| BCF | Bit Clear f |
|---|---|
| Syntax: | [ *label* ] BCF f, b |
| Operands: | 0 ≤ f ≤ 127<br>0 ≤ b ≤ 7 |
| Operation: | 0 → f[b] |
| Status Affected: | None |
| Description: | Bit 'b' in register 'f' is cleared. |

| BRA | Relative Branch |
|---|---|
| Syntax: | [ *label* ] BRA label<br>[ *label* ] BRA $+k |
| Operands: | -256 ≤ label - PC + ≤ 255<br>-256 ≤ k ≤ 255 |

chanhvinh.com



| .....continued | |
|---|---|
| **BRA** | **Relative Branch** |
| Operation: | (PC) + 1 + k → PC |
| Status Affected: | None |
| Description: | Add the signed 9-bit literal 'k' to the PC.<br>Since the PC will have incremented to fetch the next instruction, the new address will be PC + 1 + k.<br>This instruction is a two-cycle instruction. This branch has a limited range. |

| **BRW** | **Relative Branch with W** |
|---|---|
| Syntax: | [ *label* ] BRW |
| Operands: | None |
| Operation: | (PC) + (W) → PC |
| Status Affected: | None |
| Description: | Add the contents of W (unsigned) to the PC.<br>Since the PC will have incremented to fetch the next instruction, the new address will be PC + 1 + (W).<br>This instruction is a two-cycle instruction. |

| **BSF** | **Bit Set f** |
|---|---|
| Syntax: | [ *label* ] BSF f, b |
| Operands: | 0 ≤ f ≤ 127<br>0 ≤ b ≤ 7 |
| Operation: | 1 → (f[b]) |
| Status Affected: | None |
| Description: | Bit 'b' in register 'f' is set. |

| **BTFSC** | **Bit Test File, Skip If Clear** |
|---|---|
| Syntax: | [ *label* ] BTFSC f, b |
| Operands: | 0 ≤ f ≤ 127<br>0 ≤ b ≤ 7 |
| Operation: | skip if (f[b]) = 0 |
| Status Affected: | None |
| Description: | If bit 'b' in register 'f' is '1', the next instruction is executed.<br>If bit 'b', in register 'f', is '0', the next instruction is discarded,<br>and a NOP is executed instead, making this a two-cycle instruction. |

| **BTFSS** | **Bit Test File, Skip If Set** |
|---|---|
| Syntax: | [ *label* ] BTFSS f, b |
| Operands: | 0 ≤ f ≤ 127<br>0 ≤ b < 7 |
| Operation: | skip if (f[b]) = 1 |

chanhvinh.com



| .....continued | |
|---|---|
| **BTFSS** | **Bit Test File, Skip If Set** |
| Status Affected: | None |
| Description: | If bit 'b' in register 'f' is '0', the next instruction is executed.<br>If bit 'b' is '1', then the next instruction is discarded,<br>and a NOP is executed instead, making this a two-cycle instruction. |

| CALL | Subroutine Call |
|---|---|
| Syntax: | [ *label* ] CALL k |
| Operands: | 0 ≤ k ≤ 2047 |
| Operation: | (PC) + 1 → TOS,<br>k → PC[10:0],<br>(PCLATH[6:3]) → PC[14:11] |
| Status Affected: | None |
| Description: | Call Subroutine. First, return address (PC + 1) is pushed onto the stack.<br>The 11-bit immediate address is loaded into PC bits [10:0].<br>The upper bits of the PC are loaded from PCLATH.<br>CALL is a two-cycle instruction. |

| CALLW | Subroutine Call with W |
|---|---|
| Syntax: | [ *label* ] CALLW |
| Operands: | None |
| Operation: | (PC) + 1 → TOS,<br>(W) → PC[7:0],<br>(PCLATH[6:0]) → PC[14:8] |
| Status Affected: | None |
| Description: | Subroutine call with W.<br>First, the return address (PC + 1) is pushed onto the return stack.<br>Then, the contents of W is loaded into PC[7:0], and the contents of PCLATH into PC[14:8].<br>CALLW is a two-cycle instruction. |

| CLRF | Clear f |
|---|---|
| Syntax: | [ *label* ] CLRF f |
| Operands: | 0 ≤ f ≤ 127 |
| Operation: | 000h → f<br>1 → Z |
| Status Affected: | Z |
| Description: | The contents of register 'f' are cleared and the Z bit is set. |

| CLRW | Clear W |
|---|---|
| Syntax: | [ *label* ] CLRW |
| Operands: | None |

© 2023 Microchip Technology Inc.
and its subsidiaries

chanhvinh.com



| .....continued | |
|---|---|
| **CLRW** | **Clear W** |
| Operation: | 00h → (W)<br>1 → Z |
| Status Affected: | Z |
| Description: | W register is cleared. Zero (Z) bit is set. |

| CLRWDT | Clear Watchdog Timer |
|---|---|
| Syntax: | [ *label* ] CLRWDT |
| Operands: | None |
| Operation: | 00h → WDT,<br>00h → WDT prescaler,<br>1 → $\overline{TO}$,<br>1 → $\overline{PD}$ |
| Status Affected: | $\overline{TO}$, $\overline{PD}$ |
| Description: | The `CLRWDT` instruction resets the Watchdog Timer.<br>It also resets the prescaler of the WDT.<br>Status bits, $\overline{TO}$ and $\overline{PD}$, are set. |

| COMF | Complement f |
|---|---|
| Syntax: | [ *label* ] COMF f, d |
| Operands: | 0 ≤ f ≤ 127<br>d ∈ [0,1] |
| Operation: | ($\bar{f}$) → dest |
| Status Affected: | Z |
| Description: | The contents of register ‘f’ are complemented.<br>If ‘d’ is ‘`0`’, the result is stored in W.<br>If ‘d’ is ‘`1`’, the result is stored back in register ‘f’. |

| DECF | Decrement f |
|---|---|
| Syntax: | [ *label* ] DECF f, d |
| Operands: | 0 ≤ f ≤ 127<br>d ∈ [0,1] |
| Operation: | (f) – 1 → dest |
| Status Affected: | Z |
| Description: | Decrement register ‘f’.<br>If ‘d’ is ‘`0`’, the result is stored in the W register.<br>If ‘d’ is ‘`1`’, the result is stored back in register ‘f’. |

| DECFSZ | Decrement f, Skip If 0 |
|---|---|
| Syntax: | [ *label* ] DECFSZ f, d |

© 2023 Microchip Technology Inc.
and its subsidiaries

chanhvinh.com



| .....continued | |
|---|---|
| **DECFSZ** | **Decrement f, Skip If 0** |
| Operands: | 0 ≤ f ≤ 127<br>d ∈ [0,1] |
| Operation: | (f) – 1 → dest,<br>skip if result = 0 |
| Description: | The contents of register 'f' are decremented.<br>If 'd' is '0', the result is placed in the W register.<br>If 'd' is '1', the result is placed back in register 'f'.<br>If the result is '1', the next instruction is executed.<br>If the result is '0', then a NOP is executed instead,<br>making it a two-cycle instruction. |

| **GOTO** | **Unconditional Branch** |
|---|---|
| Syntax: | [ *label* ] GOTO k |
| Operands: | 0 ≤ k ≤ 2047 |
| Operation: | k → PC[10:0]<br>PCLATH[6:3] → PC[14:11] |
| Status Affected: | None |
| Description: | GOTO is an unconditional branch.<br>The 11-bit immediate value is loaded into PC bits [10:0].<br>The upper bits of PC are loaded from PCLATH[4:3].<br>GOTO is a two-cycle instruction. |

| **INCF** | **Increment f** |
|---|---|
| Syntax: | [ *label* ] INCF f, d |
| Operands: | 0 ≤ f ≤ 127<br>d ∈ [0,1] |
| Operation: | (f) + 1 → dest |
| Status Affected: | Z |
| Description: | The contents of register 'f' are incremented.<br>If 'd' is '0', the result is placed in the W register.<br>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 ≤ f ≤ 127<br>d ∈ [0,1] |
| Operation: | (f) + 1 → dest,<br>skip if result = 0 |
| Status Affected: | None |

chanhvinh.com



.....continued

| INCFSZ | Increment f, Skip If 0 |
|---|---|
| Description: | The contents of register ‘f’ are incremented.<br>If ‘d’ is ‘0’, the result is placed in the W register.<br>If ‘d’ is ‘1’, the result is placed back in register ‘f’.<br>If the result is ‘1’, the next instruction is executed.<br>If the result is ‘0’, a NOP is executed instead,<br>making it a two-cycle instruction. |

| IORLW | Inclusive OR Literal with W |
|---|---|
| Syntax: | [ *label* ] IORLW k |
| Operands: | 0 ≤ k ≤ 255 |
| Operation: | (W) .OR. k → (W) |
| Status Affected: | Z |
| Description: | The contents of W are ORed with the 8-bit literal ‘k’.<br>The result is placed in W. |

| IORWF | Inclusive OR W with f |
|---|---|
| Syntax: | IORWF f, d |
| Operands: | 0 ≤ f ≤ 127<br>d ∈ [0,1] |
| Operation: | (W) .OR. (f) → dest |
| Status Affected: | Z |
| Description: | Inclusive OR the W register with register ‘f’.<br>If ‘d’ is ‘0’, the result is placed in the W register.<br>If ‘d’ is ‘1’, the result is placed back in register ‘f’. |

| LSLF | Logical Left Shift |
|---|---|
| Syntax: | [ *label* ] LSLF f {,d} |
| Operands: | 0 ≤ f ≤ 127<br>d ∈ [0,1] |
| Operation: | (f[7]) → C<br>(f[6:0]) → dest[7:1]<br>0 → dest[0] |
| Status Affected: | C, Z |
| Description: | The contents of register ‘f’ are shifted one bit to the left through the Carry flag.<br>A ‘0’ is shifted into the LSb.<br>If ‘d’ is ‘0’, the result is placed in W.<br>If ‘d’ is ‘1’, the result is stored back in register ‘f’.<br>**C ← Register f ← 0** |

| LSRF | Logical Right Shift |
|---|---|
| Syntax: | [ *label* ] LSRF f {,d} |

© 2023 Microchip Technology Inc.
and its subsidiaries

chanhvinh.com



| .....continued | |
|---|---|
| **LSRF** | **Logical Right Shift** |
| Operands: | 0 ≤ f ≤ 127<br>d ∈ [0,1] |
| Operation: | 0 → dest[7]<br>(f[7:1]) → dest[6:0],<br>(f[0]) → C |
| Status Affected: | C, Z |
| Description: | The contents of register 'f' are shifted one bit to the right through the Carry flag.<br>A '0' is shifted into the MSb.<br>If 'd' is '0', the result is placed in W.<br>If 'd' is '1', the result is stored back in register 'f'.<br>**0 → Register f → C** |

| **MOVF** | **Move f** | |
|---|---|---|
| Syntax: | [ *label* ] MOVF f, d | |
| Operands: | 0 ≤ f ≤ 127<br>d ∈ [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.<br>If d = 0, destination is W register.<br>If d = 1, the destination is file register f itself.<br>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<br>W = value in FSR register<br>Z = 1 | |

| **MOVIW** | **Move INDFn to W** |
|---|---|
| Syntax: | [ *label* ] MOVIW ++FSRn<br>[ *label* ] MOVIW --FSRn<br>[ *label* ] MOVIW FSRn++<br>[ *label* ] MOVIW FSRn--<br>[ *label* ] MOVIW k[FSRn] |
| Operands: | n ∈ [0,1]<br>mm ∈ [00,01,10,11]<br>-32 ≤ k ≤ 31 |

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



.....continued

| MOVIW | Move INDFn to W |
|---|---|
| Operation: | INDFn → (W)<br>Effective address is determined by<br>• FSR + 1 (preincrement)<br>• FSR - 1 (predecrement)<br>• FSR + k (relative offset)<br>After the Move, the FSR value will be either:<br>• FSR + 1 (all increments)<br>• FSR - 1 (all decrements)<br>• 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. The INDFn registers are not physical registers. Any instruction that accesses an INDFn register actually accesses the register at the address specified by the FSRn. FSRn is limited to the range 0000h - FFFFh. Incrementing/decrementing it beyond these bounds will cause it to wrap-around. |

| MOVLB | Move Literal to BSR |
|---|---|
| Syntax: | [ *label* ] MOVLB k |
| Operands: | 0 ≤ k ≤ 127 |
| Operation: | k → BSR |
| Status Affected: | None |
| Description: | The 6-bit literal ‘k’ is loaded into the Bank Select Register (BSR). |

| MOVLP | Move Literal to PCLATH |
|---|---|
| Syntax: | [ *label* ] MOVLP k |
| Operands: | 0 ≤ k ≤ 127 |
| Operation: | k → PCLATH |
| Status Affected: | None |
| Description: | The 7-bit literal ‘k’ is loaded into the PCLATH register. |

| MOVLW | Move Literal to W |
|---|---|
| Syntax: | [ *label* ] MOVLW k |
| Operands: | 0 ≤ k ≤ 255 |

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



| .....continued | | | | |
|---|---|---|---|---|
| **MOVLW** | **Move Literal to W** | | | |
| Operation: | k → (W) | | | |
| Status Affected: | None | | | |
| Description: | The 8-bit literal ‘k’ is loaded into W register.<br>The “don’t cares” will assemble as ‘0’s. | | | |
| Words: | 1 | | | |
| Cycles: | 1 | | | |

| Example: | MOVLW | 5Ah |
|---|---|---|
| After Instruction<br>W = 5Ah | | |

| **MOVWF** | **Move W to f** | | | |
|---|---|---|---|---|
| Syntax: | [ *label* ] MOVWF f | | | |
| Operands: | 0 ≤ f ≤ 127 | | | |
| Operation: | (W) → f | | | |
| Status Affected: | None | | | |
| Description: | Move data from W to register ‘f’. | | | |
| Words: | 1 | | | |
| Cycles: | 1 | | | |

| Example: | MOVWF | LATA |
|---|---|---|
| Before Instruction<br>LATA = FFh<br>W = 4Fh<br>After Instruction<br>LATA = 4Fh<br>W = 4Fh | | |

| **MOVWI** | **Move W to INDFn** |
|---|---|
| Syntax: | [ *label* ] MOVWI ++FSRn<br>[ *label* ] MOVWI --FSRn<br>[ *label* ] MOVWI FSRn++<br>[ *label* ] MOVWI FSRn--<br>[ *label* ] MOVWI k[FSRn] |
| Operands: | n ∈ [0,1]<br>mm ∈ [00,01,10,11]<br>-32 ≤ k ≤ 31 |

© 2023 Microchip Technology Inc.
and its subsidiaries

chanhvinh.com



.....continued

| MOVWI | Move W to INDFn | | |
|---|---|---|---|
| Operation: | (W) → INDFn<br>Effective address is determined by<br>• FSR + 1 (preincrement)<br>• FSR - 1 (predecrement)<br>• FSR + k (relative offset)<br>After the Move, the FSR value will be either:<br>• FSR + 1 (all increments)<br>• FSR - 1 (all decrements)<br>• 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.<br>The INDFn registers are not physical registers.<br>Any instruction that accesses an INDFn register actually accesses the register at the address specified by the FSRn.<br>FSRn is limited to the range 0000h-FFFFh.<br>Incrementing/decrementing it beyond these bounds will cause it to wrap-around.<br>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 |
|---|---|
| None. | |

| RESET | Software Reset |
|---|---|
| Syntax: | [ *label* ] RESET |
| Operands: | None |
| Operation: | Execute a device Reset.<br>Resets the $\overline{RI}$ flag of the PCON register. |

chanhvinh.com



| .....continued | |
|---|---|
| **RESET** | **Software Reset** |
| Status Affected: | None |
| Description: | This instruction provides a way to execute a hardware Reset by software. |

| RETFIE | Return from Interrupt | | | |
|---|---|---|---|---|
| Syntax: | [ *label* ] RETFIE k | | | |
| Operands: | None | | | |
| Operation: | (TOS) → PC,<br>1 → GIE | | | |
| Status Affected: | None | | | |
| Description: | Return from Interrupt.<br>Stack is POPed and Top-of-Stack (TOS) is loaded in the PC.<br>Interrupts are enabled by setting the Global Interrupt Enable bit, GIE (INTCON[7]).<br>This is a two-cycle instruction. | | | |
| Words: | 1 | | | |
| Cycles: | 2 | | | |

| Example: | `RETFIE` |
|---|---|
| After Interrupt<br>PC = TOS<br>GIE = `1` | |

| RETLW | Return Literal to W | | | |
|---|---|---|---|---|
| Syntax: | [ *label* ] RETLW k | | | |
| Operands: | 0 ≤ k ≤ 255 | | | |
| Operation: | k → (W),<br>(TOS) → PC, | | | |
| Status Affected: | None | | | |
| Description: | The W register is loaded with the 8-bit literal ‘k’.<br>The program counter is loaded from the top of the stack (the return address).<br>This is a two-cycle instruction. | | | |
| Words: | 1 | | | |
| Cycles: | 2 | | | |

© 2023 Microchip Technology Inc.
and its subsidiaries

chanhvinh.com



Example:

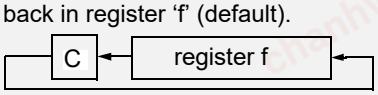
```
      CALL TABLE            ; W contains table
; offset value
; W now has
; table value
   :
TABLE
ADDWF PC            ; W = offset
RETLW k1            ; Begin table
RETLW k2            ;
:
:
RETLW kn            ; End of table
```

Before Instruction
W = 07h

After Instruction

W = value of k8

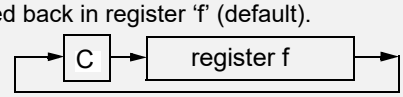
| RETURN | Return from Subroutine | | | |
|---|---|---|---|---|
| Syntax: | [ *label* ] RETURN | | | |
| Operands: | None | | | |
| Operation: | (TOS) → PC, | | | |
| Status Affected: | None | | | |
| Encoding: | 0000 | 0000 | 0001 | 001s |
| Description: | Return from subroutine.<br>The stack is POPped and the top of the stack (TOS) is loaded into the Program Counter.<br>This is a two-cycle instruction. | | | |

| RLF | Rotate Left f through Carry | | | |
|---|---|---|---|---|
| Syntax: | [ *label* ] RLF f, d | | | |
| Operands: | 0 ≤ f ≤ 127<br>d ∈ [0,1] | | | |
| Operation: | (f[n]) → dest[n + 1],<br>(f[7]) → C,<br>(C) → dest[0] | | | |
| Status Affected: | C | | | |
| Encoding: | 0011 | 01da | ffff | ffff |
| Description: | The contents of register ‘f’ are rotated one bit to the left through the Carry flag.<br>If ‘d’ is ‘0’, the result is placed in W.<br>If ‘d’ is ‘1’, the result is stored back in register ‘f’ (default).<br> | | | |
| Words: | 1 | | | |
| Cycles: | 1 | | | |

chanhvinh.com



| | | |
|---|---|---|
| Example: | RLF | REG1, 0 |
| Before Instruction<br>REG1 = 1110 0110<br>C = 0<br>After Instruction<br>REG = 1110 0110<br>W = 1100 1100<br>C = 1 | | |

| RRF | Rotate Right f through Carry |
|---|---|
| Syntax: | [ *label* ] RRF f, d |
| Operands: | 0 ≤ f ≤ 127<br>d ∈ [0,1] |
| Operation: | (f[n]) → dest[n – 1],<br>(f[0]) → C,<br>(C) → dest[7] |
| Status Affected: | C |
| Description: | The contents of register ‘f’ are rotated one bit to the right through the Carry flag.<br>If ‘d’ is ‘0’, the result is placed in W.<br>If ‘d’ is ‘1’, the result is placed back in register ‘f’ (default).<br> |

| SLEEP | Enter Sleep Mode |
|---|---|
| Syntax: | [ *label* ] SLEEP |
| Operands: | None |
| Operation: | 00h → WDT,<br>0 → WDT prescaler,<br>1 → $\overline{TO}$,<br>0 → $\overline{PD}$ |
| Status Affected: | $\overline{TO}$, $\overline{PD}$ |
| Description: | The Power-Down ($\overline{PD}$) Status bit is cleared.<br>The Time-Out ($\overline{TO}$) Status bit is set.<br>Watchdog Timer and its prescaler are cleared. |

| SUBLW | Subtract W from Literal |
|---|---|
| Syntax: | [ *label* ] SUBLW k |
| Operands: | 0 ≤ k ≤ 255 |
| Operation: | k – (W) → (W) |
| Status Affected: | C, DC, Z |

© 2023 Microchip Technology Inc.
and its subsidiaries

chanhvinh.com



| .....continued | |
|---|---|
| **SUBLW** | **Subtract W from Literal** |
| Description | The W register is subtracted (two's complement method) from the 8-bit literal 'k'.<br>The result is placed in the W register.<br>C = 0, W > k<br>C = 1, W ≤ k<br>DC = 0, W[3:0] > k[3:0]<br>DC = 1, W[3:0] ≤ k[3:0] |

| **SUBWF** | **Subtract W from f** |
|---|---|
| Syntax: | [ *label* ] SUBWF f, d |
| Operands: | 0 ≤ f ≤ 127<br>d ∈ [0,1] |
| Operation: | (f) - (W) → (dest) |
| Status Affected: | C, DC, Z |
| Description | Subtract (two's complement method) W register from register 'f'.<br>If 'd' is '0', the result is stored in the W register.<br>If 'd' is '1', the result is stored back in register 'f.<br>C = 0, W > f<br>C = 1, W ≤ f<br>DC = 0, W[3:0] > f[3:0]<br>DC = 1, W[3:0] ≤ f[3:0] |

| **SUBWFB** | **Subtract W from f with Borrow** |
|---|---|
| Syntax: | [ *label* ] SUBFWB f {,d} |
| Operands: | 0 ≤ f ≤ 127<br>d ∈ [0,1] |
| Operation: | (W) – (f) – ($\overline{B}$) → dest |
| Status Affected: | C, DC, Z |
| Description: | Subtract W and the Borrow flag (Carry) from register 'f' (two's complement method).<br>If 'd' is '0', the result is stored in W.<br>If 'd' is '1', the result is stored back in register 'f'. |

| **SWAPF** | **Swap Nibbles in f** |
|---|---|
| Syntax: | [ *label* ] SWAPF f, d |
| Operands: | 0 ≤ f ≤ 127<br>d ∈ [0,1] |
| Operation: | (f[3:0]) → dest[7:4],<br>(f[7:4]) → dest[3:0] |
| Status Affected: | None |

chanhvinh.com



| .....continued | |
|---|---|
| **SWAPF** | **Swap Nibbles in f** |
| Description: | The upper and lower nibbles of register ‘f’ are exchanged.<br>If ‘d’ is ‘0’, the result is placed in W.<br>If ‘d’ is ‘1’, the result is placed in register ‘f’ (default). |

| **TRIS** | **Load TRIS Register with W** |
|---|---|
| Syntax: | [ *label* ] TRIS f |
| Operands: | 5 ≤ f ≤ 7 |
| Operation: | (W) → TRIS register ‘f’ |
| Status Affected: | None |
| Description: | Move data from W register to TRIS register.<br>When ‘f’ = 5, TRISA is loaded.<br>When ‘f’ = 6, TRISB is loaded.<br>When ‘f’ = 7, TRISC is loaded. |

| **XORLW** | **Exclusive OR Literal with W** |
|---|---|
| Syntax: | [ *label* ] XORLW k |
| Operands: | 0 ≤ k ≤ 255 |
| Operation: | (W) .XOR. k → (W) |
| Status Affected: | Z |
| Description: | The contents of W are XORed with the 8-bit literal ‘k’.<br>The result is placed in W. |

| **XORWF** | **Exclusive OR W with f** |
|---|---|
| Syntax: | [ *label* ] XORWF f, d |
| Operands: | 0 ≤ f ≤ 127<br>d ∈ [0,1] |
| Operation: | (W) .XOR. (f) → dest |
| Status Affected: | Z |
| Description: | Exclusive OR the contents of the W register with register ‘f’.<br>If ‘d’ is ‘0’, the result is stored in the W register.<br>If ‘d’ is ‘1’, the result is stored back in register ‘f’. |

© 2023 Microchip Technology Inc.
and its subsidiaries

chanhvinh.com



# 42. ICSP™ - In-Circuit Serial Programming™

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
- $\overline{MCLR}/V_{PP}$
- $V_{DD}$
- $V_{SS}$

In Program/Verify mode, the program memory, User IDs and the Configuration bits 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 **"Family Programming Specification"**.

## 42.1 High-Voltage Programming Entry Mode

The device is placed into High-Voltage Programming Entry mode by holding the ICSPCLK and ICSPDAT pins low, then raising the voltage on $\overline{MCLR}/V_{PP}$ to $V_{IH}$.

## 42.2 Low-Voltage Programming Entry Mode

The Low-Voltage Programming Entry mode allows the PIC® Flash MCUs to be programmed using $V_{DD}$ only, without high voltage. When the LVP Configuration bit is set to '1', the low-voltage ICSP programming entry is enabled. To disable the Low-Voltage ICSP mode, the LVP bit must be programmed to '0'.

Entry into the Low-Voltage Programming Entry mode requires the following steps:

1. $\overline{MCLR}$ is brought to $V_{IL}$.
2. A 32-bit key sequence is presented on ICSPDAT, while clocking ICSPCLK.

Once the key sequence is complete, $\overline{MCLR}$ must be held at $V_{IL}$ for as long as Program/Verify mode is to be maintained.

If low-voltage programming is enabled (LVP = 1), the $\overline{MCLR}$ Reset function is automatically enabled and cannot be disabled. See the *$\overline{MCLR}$* section for more information.

The LVP bit can only be reprogrammed to '0' by using the High-Voltage Programming mode.

## 42.3 Common Programming Interfaces

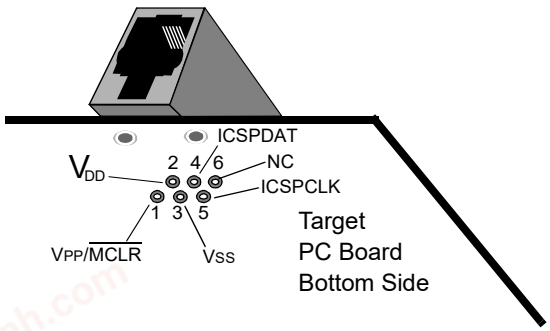
Connection to a target device is typically done through an ICSP header. A commonly found connector on development tools is the RJ-11 in the 6P6C (6-pin, 6-connector) configuration. See Figure 42-1.

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



**Figure 42-1. ICD RJ-11 Style Connector Interface**




Pin Description

1 = $V_{PP}/\overline{MCLR}$

2 = $V_{DD}$ Target

3 = $V_{SS}$ (ground)

4 = ICSPDAT

5 = ICSPCLK

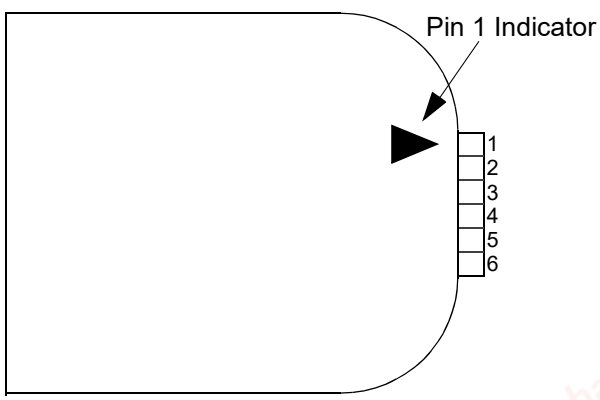
6 = No Connect

Another connector often found in use with the PICkit™ programmers is a standard 6-pin header with 0.1 inch spacing. Refer to Figure 42-2.

For additional interface recommendations, refer to the 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 42-3 for more information.

**Figure 42-2. PICkit™ Programmer Style Connector Interface**



chanhvinh.com



Pin Description[1]:

1 = $V_{PP}$/$\overline{MCLR}$

2 = $V_{DD}$ Target

3 = $V_{SS}$ (ground)

4 = ICSPDAT

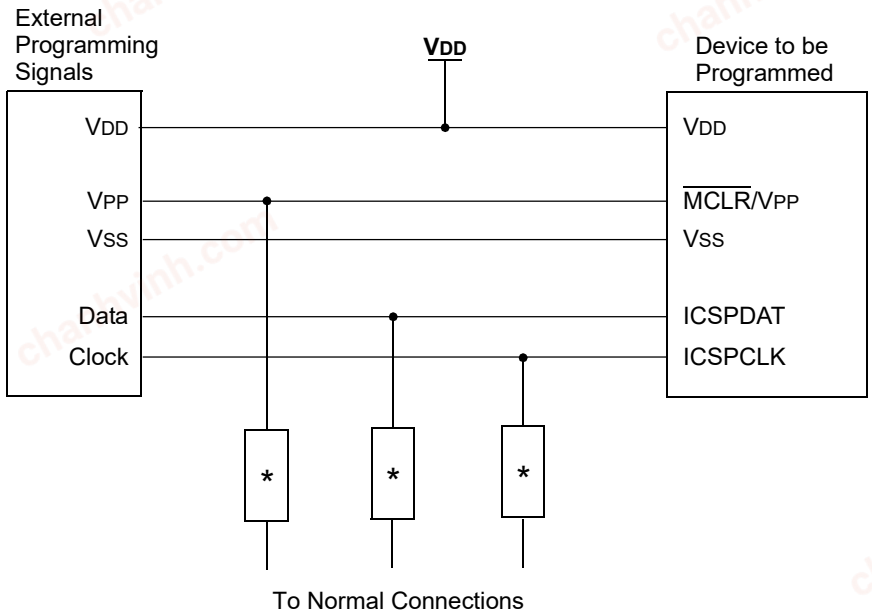
5 = ICSPCLK

6 = No Connect

**Note:**

1. The 6-pin header (0.100" spacing) accepts 0.025" square pins.

**Figure 42-3. Typical Connection for ICSP™ Programming**




* Isolation devices (as required).

chanhvinh.com



# 43. Register Summary

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | INDF0 | 7:0 | INDF0[7:0] | | | | | | | |
| 0x01 | INDF1 | 7:0 | INDF1[7:0] | | | | | | | |
| 0x02 | PCL | 7:0 | PCL[7:0] | | | | | | | |
| 0x03 | STATUS | 7:0 | | | | $\overline{TO}$ | $\overline{PD}$ | Z | DC | C |
| 0x04 | FSR0 | 7:0 | FSR0[7:0] | | | | | | | |
| | | 15:8 | FSR0[15:8] | | | | | | | |
| 0x06 | FSR1 | 7:0 | FSR1[7:0] | | | | | | | |
| | | 15:8 | FSR1[15:8] | | | | | | | |
| 0x08 | BSR | 7:0 | | | BSR[5:0] | | | | | |
| 0x09 | WREG | 7:0 | WREG[7:0] | | | | | | | |
| 0x0A | PCLATH | 7:0 | | PCLATH[6:0] | | | | | | |
| 0x0B | INTCON | 7:0 | GIE | PEIE | | | | | | INTEDG |
| 0x0C | PORTA | 7:0 | RA7 | RA6 | RA5 | RA4 | RA3 | RA2 | RA1 | RA0 |
| 0x0D | PORTB | 7:0 | RB7 | RB6 | RB5 | RB4 | RB3 | RB2 | RB1 | RB0 |
| 0x0E | PORTC | 7:0 | RC7 | RC6 | RC5 | RC4 | RC3 | RC2 | RC1 | RC0 |
| 0x0F | PORTD | 7:0 | RD7 | RD6 | RD5 | RD4 | RD3 | RD2 | RD1 | RD0 |
| 0x10 | PORTE | 7:0 | | | | | RE3 | RE2 | RE1 | RE0 |
| 0x11 | Reserved | | | | | | | | | |
| 0x12 | TRISA | 7:0 | TRISA7 | TRISA6 | TRISA5 | TRISA4 | TRISA3 | TRISA2 | TRISA1 | TRISA0 |
| 0x13 | TRISB | 7:0 | TRISB7 | TRISB6 | TRISB5 | TRISB4 | TRISB3 | TRISB2 | TRISB1 | TRISB0 |
| 0x14 | TRISC | 7:0 | TRISC7 | TRISC6 | TRISC5 | TRISC4 | TRISC3 | TRISC2 | TRISC1 | TRISC0 |
| 0x15 | TRISD | 7:0 | TRISD7 | TRISD6 | TRISD5 | TRISD4 | TRISD3 | TRISD2 | TRISD1 | TRISD0 |
| 0x16 | TRISE | 7:0 | | | | | Reserved | TRISE2 | TRISE1 | TRISE0 |
| 0x17 | Reserved | | | | | | | | | |
| 0x18 | LATA | 7:0 | LATA7 | LATA6 | LATA5 | LATA4 | LATA3 | LATA2 | LATA1 | LATA0 |
| 0x19 | LATB | 7:0 | LATB7 | LATB6 | LATB5 | LATB4 | LATB3 | LATB2 | LATB1 | LATB0 |
| 0x1A | LATC | 7:0 | LATC7 | LATC6 | LATC5 | LATC4 | LATC3 | LATC2 | LATC1 | LATC0 |
| 0x1B | LATD | 7:0 | LATD7 | LATD6 | LATD5 | LATD4 | LATD3 | LATD2 | LATD1 | LATD0 |
| 0x1C | LATE | 7:0 | | | | | | LATE2 | LATE1 | LATE0 |
| 0x1D ... 0x8B | Reserved | | | | | | | | | |
| 0x8C | PIR0 | 7:0 | | | TMR0IF | IOCIF | | | | INTF |
| 0x8D | PIR1 | 7:0 | TMR1GIF | TMR1IF | OSFIF | CSWIF | ACTIF | SCANIF | CRCIF | NVMIF |
| 0x8E | PIR2 | 7:0 | | CCP2IF | CCP1IF | TMR6IF | TMR4IF | TMR2IF | TMR3GIF | TMR3IF |
| 0x8F | PIR3 | 7:0 | PWM4IF | PWM4PIF | PWM3IF | PWM3PIF | PWM2IF | PWM2PIF | PWM1IF | PWM1PIF |
| 0x90 | PIR4 | 7:0 | RC1IF | TX1IF | CLC4IF | CLC3IF | CLC2IF | CLC1IF | CWG1IF | NCO1IF |
| 0x91 | PIR5 | 7:0 | CM2IF | CM1IF | BCL2IF | SSP2IF | BCL1IF | SSP1IF | RC2IF | TX2IF |
| 0x92 | PIR6 | 7:0 | | | | | | ZCDIF | ADTIF | ADIF |
| 0x93 ... 0x95 | Reserved | | | | | | | | | |
| 0x96 | PIE0 | 7:0 | | | TMR0IE | IOCIE | | | | INTE |
| 0x97 | PIE1 | 7:0 | TMR1GIE | TMR1IE | OSFIE | CSWIE | ACTIE | SCANIE | CRCIE | NVMIE |
| 0x98 | PIE2 | 7:0 | | CCP2IE | CCP1IE | TMR6IE | TMR4IE | TMR2IE | TMR3GIE | TMR3IE |
| 0x99 | PIE3 | 7:0 | PWM4IE | PWM4PIE | PWM3IE | PWM3PIE | PWM2IE | PWM2PIE | PWM1IE | PWM1PIE |
| 0x9A | PIE4 | 7:0 | RC1IE | TX1IE | CLC4IE | CLC3IE | CLC2IE | CLC1IE | CWG1IE | NCO1IE |
| 0x9B | PIE5 | 7:0 | CM2IE | CM1IE | BCL2IE | SSP2IE | BCL1IE | SSP1IE | RC2IE | TX2IE |
| 0x9C | PIE6 | 7:0 | | | | | | ZCDIE | ADTIE | ADIE |
| 0x9D ... 0x010B | Reserved | | | | | | | | | |
| 0x010C | PMD0 | 7:0 | TMR0MD | CLKRMD | IOCMD | ACTMD | SYSCMD | SCANMD | CRCMD | NVMMD |
| 0x010D | PMD1 | 7:0 | PWM1MD | CCP2MD | CCP1MD | TMR6MD | TMR4MD | TMR2MD | TMR3MD | TMR1MD |
| 0x010E | PMD2 | 7:0 | CLC3MD | CLC2MD | CLC1MD | CWG1MD | NCO1MD | PWM4MD | PWM3MD | PWM2MD |
| 0x010F | PMD3 | 7:0 | CM2MD | CM1MD | FVRMD | MSSP2MD | MSSP1MD | UART2MD | UART1MD | CLC4MD |

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0110 | PMD4 | 7:0 | | | | ZCDMD | OPA1MD | DAC2MD | DAC1MD | ADCMD |
| 0x0111 ... 0x018B | Reserved | | | | | | | | | |
| 0x018C | WDTCON0 | 7:0 | | | PS[4:0] | | | | | SEN |
| 0x018D | WDTCON1 | 7:0 | | CS[2:0] | | | | WINDOW[2:0] | | |
| 0x018E | WDTPSL | 7:0 | PSCNTL[7:0] | | | | | | | |
| 0x018F | WDTPSH | 7:0 | PSCNTH[7:0] | | | | | | | |
| 0x0190 | WDTTMR | 7:0 | TMR[4:0] | | | | | STATE | PSCNT[17:16] | |
| 0x0191 | BORCON | 7:0 | SBOREN | | | | | | | BORRDY |
| 0x0192 | PCON0 | 7:0 | STKOVF | STKUNF | | RWDT | RMCLR | RI | POR | BOR |
| 0x0193 | PCON1 | 7:0 | | | | | | | MEMV | |
| 0x0194 ... 0x019B | Reserved | | | | | | | | | |
| 0x019C | TMR0L | 7:0 | TMR0L[7:0] | | | | | | | |
| 0x019D | TMR0H | 7:0 | TMR0H[7:0] | | | | | | | |
| 0x019E | T0CON0 | 7:0 | EN | | OUT | MD16 | OUTPS[3:0] | | | |
| 0x019F | T0CON1 | 7:0 | CS[2:0] | | | ASYNC | CKPS[3:0] | | | |
| 0x01A0 ... 0x020B | Reserved | | | | | | | | | |
| 0x020C | FVRCON | 7:0 | EN | RDY | TSEN | TSRNG | CDAFVR[1:0] | | ADFVR[1:0] | |
| 0x020D | CPCON | 7:0 | CPON[1:0] | | CPOS | | | CPREQ | CPT | CPRDY |
| 0x020E ... 0x021E | Reserved | | | | | | | | | |
| 0x021F | ZCDCON | 7:0 | SEN | | OUT | POL | | | INTP | INTN |
| 0x0220 ... 0x028B | Reserved | | | | | | | | | |
| 0x028C | CPUDOZE | 7:0 | IDLEN | DOZEN | ROI | DOE | | DOZE[2:0] | | |
| 0x028D | OSCCON1 | 7:0 | | NOSC[2:0] | | | NDIV[3:0] | | | |
| 0x028E | OSCCON2 | 7:0 | | COSC[2:0] | | | CDIV[3:0] | | | |
| 0x028F | OSCCON3 | 7:0 | CSWHOLD | SOSCPWR | | ORDY | NOSCR | | | |
| 0x0290 | OSCSTAT | 7:0 | EXTOR | HFOR | MFOR | LFOR | SOR | ADOR | SFOR | PLLR |
| 0x0291 | OSCEN | 7:0 | EXTOEN | HFOEN | MFOEN | LFOEN | SOSCEN | ADOEN | | PLLEN |
| 0x0292 | OSCTUNE | 7:0 | | | TUN[5:0] | | | | | |
| 0x0293 | OSCFRQ | 7:0 | | | | | | FRQ[2:0] | | |
| 0x0294 | ACTCON | 7:0 | ACTEN | ACTUD | | | ACTLOCK | | ACTORS | |
| 0x0295 | Reserved | | | | | | | | | |
| 0x0296 | CLKRCON | 7:0 | EN | | | DC[1:0] | | DIV[2:0] | | |
| 0x0297 | CLKRCLK | 7:0 | | | | | CLK[3:0] | | | |
| 0x0298 ... 0x030B | Reserved | | | | | | | | | |
| 0x030C | TMR1 | 7:0 | TMR1[7:0] | | | | | | | |
| | | 15:8 | TMR1[15:8] | | | | | | | |
| 0x030E | T1CON | 7:0 | | | CKPS[1:0] | | | SYNC | RD16 | ON |
| 0x030F | T1GCON | 7:0 | GE | GPOL | GTM | GSPM | GGO/DONE | GVAL | | |
| 0x0310 | T1GATE | 7:0 | | | | GSS[4:0] | | | | |
| 0x0311 | T1CLK | 7:0 | | | | CS[4:0] | | | | |
| 0x0312 | TMR3 | 7:0 | TMR3[7:0] | | | | | | | |
| | | 15:8 | TMR3[15:8] | | | | | | | |
| 0x0314 | T3CON | 7:0 | | | CKPS[1:0] | | | SYNC | RD16 | ON |
| 0x0315 | T3GCON | 7:0 | GE | GPOL | GTM | GSPM | GGO/DONE | GVAL | | |
| 0x0316 | T3GATE | 7:0 | | | | GSS[4:0] | | | | |
| 0x0317 | T3CLK | 7:0 | | | | CS[4:0] | | | | |

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0318 ... 0x038B | Reserved | | | | | | | | | |
| 0x038C | T2TMR | 7:0 | T2TMR[7:0] | | | | | | | |
| 0x038D | T2PR | 7:0 | T2PR[7:0] | | | | | | | |
| 0x038E | T2CON | 7:0 | ON | | CKPS[2:0] | | OUTPS[3:0] | | | |
| 0x038F | T2HLT | 7:0 | PSYNC | CPOL | CSYNC | MODE[4:0] | | | | |
| 0x0390 | T2CLKCON | 7:0 | | | | | CS[3:0] | | | |
| 0x0391 | T2RST | 7:0 | | | | RSEL[4:0] | | | | |
| 0x0392 | T4TMR | 7:0 | T4TMR[7:0] | | | | | | | |
| 0x0393 | T4PR | 7:0 | T4PR[7:0] | | | | | | | |
| 0x0394 | T4CON | 7:0 | ON | | CKPS[2:0] | | OUTPS[3:0] | | | |
| 0x0395 | T4HLT | 7:0 | PSYNC | CPOL | CSYNC | MODE[4:0] | | | | |
| 0x0396 | T4CLKCON | 7:0 | | | | | CS[3:0] | | | |
| 0x0397 | T4RST | 7:0 | | | | RSEL[4:0] | | | | |
| 0x0398 | T6TMR | 7:0 | T6TMR[7:0] | | | | | | | |
| 0x0399 | T6PR | 7:0 | T6PR[7:0] | | | | | | | |
| 0x039A | T6CON | 7:0 | ON | | CKPS[2:0] | | OUTPS[3:0] | | | |
| 0x039B | T6HLT | 7:0 | PSYNC | CPOL | CSYNC | MODE[4:0] | | | | |
| 0x039C | T6CLKCON | 7:0 | | | | | CS[3:0] | | | |
| 0x039D | T6RST | 7:0 | | | | RSEL[4:0] | | | | |
| 0x039E ... 0x040B | Reserved | | | | | | | | | |
| 0x040C | CCPR1 | 7:0 | CCPR[7:0] | | | | | | | |
| | | 15:8 | CCPR[15:8] | | | | | | | |
| 0x040E | CCP1CON | 7:0 | EN | | OUT | FMT | MODE[3:0] | | | |
| 0x040F | CCP1CAP | 7:0 | | | | | CTS[3:0] | | | |
| 0x0410 | CCPR2 | 7:0 | CCPR[7:0] | | | | | | | |
| | | 15:8 | CCPR[15:8] | | | | | | | |
| 0x0412 | CCP2CON | 7:0 | EN | | OUT | FMT | MODE[3:0] | | | |
| 0x0413 | CCP2CAP | 7:0 | | | | | CTS[3:0] | | | |
| 0x0414 ... 0x041E | Reserved | | | | | | | | | |
| 0x041F | CCPTMRS0 | 7:0 | | | | | C2TSEL[1:0] | | C1TSEL[1:0] | |
| 0x0420 ... 0x048B | Reserved | | | | | | | | | |
| 0x048C | PWM1ERS | 7:0 | | | | | ERS[3:0] | | | |
| 0x048D | PWM1CLK | 7:0 | | | | | CLK[3:0] | | | |
| 0x048E | PWM1LDS | 7:0 | | | | | | LDS[2:0] | | |
| 0x048F | PWM1PR | 7:0 | PR[7:0] | | | | | | | |
| | | 15:8 | PR[15:8] | | | | | | | |
| 0x0491 | PWM1CPRE | 7:0 | CPRE[7:0] | | | | | | | |
| 0x0492 | PWM1PIPOS | 7:0 | PIPOS[7:0] | | | | | | | |
| 0x0493 | PWM1GIR | 7:0 | | | | | | | S1P2 | S1P1 |
| 0x0494 | PWM1GIE | 7:0 | | | | | | | S1P2 | S1P1 |
| 0x0495 | PWM1CON | 7:0 | EN | | | | | LD | ERSPOL | ERSNOW |
| 0x0496 | PWM1S1CFG | 7:0 | POL2 | POL1 | | | PPEN | MODE[2:0] | | |
| 0x0497 | PWM1S1P1 | 7:0 | P1[7:0] | | | | | | | |
| | | 15:8 | P1[15:8] | | | | | | | |
| 0x0499 | PWM1S1P2 | 7:0 | P2[7:0] | | | | | | | |
| | | 15:8 | P2[15:8] | | | | | | | |
| 0x049B ... 0x049D | Reserved | | | | | | | | | |
| 0x049E | PWMLOAD | 7:0 | | | | | MPWM4LD | MPWM3LD | MPWM2LD | MPWM1LD |
| 0x049F | PWMEN | 7:0 | | | | | MPWM4EN | MPWM3EN | MPWM2EN | MPWM1EN |

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x04A0 ... 0x050B | Reserved | | | | | | | | | |
| 0x050C | PWM2ERS | 7:0 | | | | | ERS[3:0] | | | |
| 0x050D | PWM2CLK | 7:0 | | | | | CLK[3:0] | | | |
| 0x050E | PWM2LDS | 7:0 | | | | | | LDS[2:0] | | |
| 0x050F | PWM2PR | 7:0 | PR[7:0] | | | | | | | |
| | | 15:8 | PR[15:8] | | | | | | | |
| 0x0511 | PWM2CPRE | 7:0 | CPRE[7:0] | | | | | | | |
| 0x0512 | PWM2PIPOS | 7:0 | PIPOS[7:0] | | | | | | | |
| 0x0513 | PWM2GIR | 7:0 | | | | | | | S1P2 | S1P1 |
| 0x0514 | PWM2GIE | 7:0 | | | | | | | S1P2 | S1P1 |
| 0x0515 | PWM2CON | 7:0 | EN | | | | | LD | ERSPOL | ERSNOW |
| 0x0516 | PWM2S1CFG | 7:0 | POL2 | POL1 | | | PPEN | MODE[2:0] | | |
| 0x0517 | PWM2S1P1 | 7:0 | P1[7:0] | | | | | | | |
| | | 15:8 | P1[15:8] | | | | | | | |
| 0x0519 | PWM2S1P2 | 7:0 | P2[7:0] | | | | | | | |
| | | 15:8 | P2[15:8] | | | | | | | |
| 0x051B ... 0x058B | Reserved | | | | | | | | | |
| 0x058C | NCO1ACC | 7:0 | ACC[7:0] | | | | | | | |
| | | 15:8 | ACC[15:8] | | | | | | | |
| | | 23:16 | | | | | ACC[19:16] | | | |
| 0x058F | NCO1INC | 7:0 | INC[7:0] | | | | | | | |
| | | 15:8 | INC[15:8] | | | | | | | |
| | | 23:16 | | | | | INC[19:16] | | | |
| 0x0592 | NCO1CON | 7:0 | EN | | OUT | POL | | | | PFM |
| 0x0593 | NCO1CLK | 7:0 | | PWS[2:0] | | | CKS[3:0] | | | |
| 0x0594 ... 0x060B | Reserved | | | | | | | | | |
| 0x060C | CWG1CLK | 7:0 | | | | | | | | CS |
| 0x060D | CWG1ISM | 7:0 | | | | ISM[4:0] | | | | |
| 0x060E | CWG1DBR | 7:0 | | | DBR[5:0] | | | | | |
| 0x060F | CWG1DBF | 7:0 | | | DBF[5:0] | | | | | |
| 0x0610 | CWG1CON0 | 7:0 | EN | LD | | | | MODE[2:0] | | |
| 0x0611 | CWG1CON1 | 7:0 | | | IN | | POLD | POLC | POLB | POLA |
| 0x0612 | CWG1AS0 | 7:0 | SHUTDOWN | REN | LSBD[1:0] | | LSAC[1:0] | | | |
| 0x0613 | CWG1AS1 | 7:0 | AS7E | AS6E | AS5E | AS4E | AS3E | AS2E | AS1E | AS0E |
| 0x0614 | CWG1STR | 7:0 | OVRD | OVRC | OVRB | OVRA | STRD | STRC | STRB | STRA |
| 0x0615 ... 0x068B | Reserved | | | | | | | | | |
| 0x068C | CLCnCON | 7:0 | EN | | OUT | INTP | INTN | MODE[2:0] | | |
| 0x068D | CLCnPOL | 7:0 | POL | | | | G4POL | G3POL | G2POL | G1POL |
| 0x068E | CLCnSEL0 | 7:0 | | D1S[6:0] | | | | | | |
| 0x068F | CLCnSEL1 | 7:0 | | D2S[6:0] | | | | | | |
| 0x0690 | CLCnSEL2 | 7:0 | | D3S[6:0] | | | | | | |
| 0x0691 | CLCnSEL3 | 7:0 | | D4S[6:0] | | | | | | |
| 0x0692 | CLCnGLS0 | 7:0 | G1D4T | G1D4N | G1D3T | G1D3N | G1D2T | G1D2N | G1D1T | G1D1N |
| 0x0693 | CLCnGLS1 | 7:0 | G2D4T | G2D4N | G2D3T | G2D3N | G2D2T | G2D2N | G2D1T | G2D1N |
| 0x0694 | CLCnGLS2 | 7:0 | G3D4T | G3D4N | G3D3T | G3D3N | G3D2T | G3D2N | G3D1T | G3D1N |
| 0x0695 | CLCnGLS3 | 7:0 | G4D4T | G4D4N | G4D3T | G4D3N | G4D2T | G4D2N | G4D1T | G4D1N |
| 0x0696 | CLCSELECT | 7:0 | | | | | SLCT[3:0] | | | |
| 0x0697 | CLCDATA | 7:0 | | | | | CLC4OUT | CLC3OUT | CLC2OUT | CLC1OUT |
| 0x0698 ... 0x070B | Reserved | | | | | | | | | |

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x070C | RC1REG | 7:0 | RCREG[7:0] | | | | | | | |
| 0x070D | TX1REG | 7:0 | TXREG[7:0] | | | | | | | |
| 0x070E | SP1BRG | 7:0 | SPBRG[7:0] | | | | | | | |
| | | 15:8 | SPBRG[15:8] | | | | | | | |
| 0x0710 | RC1STA | 7:0 | SPEN | RX9 | SREN | CREN | ADDEN | FERR | OERR | RX9D |
| 0x0711 | TX1STA | 7:0 | CSRC | TX9 | TXEN | SYNC | SENDB | BRGH | TRMT | TX9D |
| 0x0712 | BAUD1CON | 7:0 | ABDOVF | RCIDL | | SCKP | BRG16 | | WUE | ABDEN |
| 0x0713 ... 0x0715 | Reserved | | | | | | | | | |
| 0x0716 | RC2REG | 7:0 | RCREG[7:0] | | | | | | | |
| 0x0717 | TX2REG | 7:0 | TXREG[7:0] | | | | | | | |
| 0x0718 | SP2BRG | 7:0 | SPBRG[7:0] | | | | | | | |
| | | 15:8 | SPBRG[15:8] | | | | | | | |
| 0x071A | RC2STA | 7:0 | SPEN | RX9 | SREN | CREN | ADDEN | FERR | OERR | RX9D |
| 0x071B | TX2STA | 7:0 | CSRC | TX9 | TXEN | SYNC | SENDB | BRGH | TRMT | TX9D |
| 0x071C | BAUD2CON | 7:0 | ABDOVF | RCIDL | | SCKP | BRG16 | | WUE | ABDEN |
| 0x071D ... 0x078B | Reserved | | | | | | | | | |
| 0x078C | SSP1BUF | 7:0 | BUF[7:0] | | | | | | | |
| 0x078D | SSP1ADD | 7:0 | ADD[7:0] | | | | | | | |
| 0x078E | SSP1MSK | 7:0 | MSK[6:0] | | | | | | | MSK0 |
| 0x078F | SSP1STAT | 7:0 | SMP | CKE | D/$\overline{A}$ | P | S | R/$\overline{W}$ | UA | BF |
| 0x0790 | SSP1CON1 | 7:0 | WCOL | SSPOV | SSPEN | CKP | SSPM[3:0] | | | |
| 0x0791 | SSP1CON2 | 7:0 | GCEN | ACKSTAT | ACKDT | ACKEN | RCEN | PEN | RSEN | SEN |
| 0x0792 | SSP1CON3 | 7:0 | ACKTIM | PCIE | SCIE | BOEN | SDAHT | SBCDE | AHEN | DHEN |
| 0x0793 ... 0x0795 | Reserved | | | | | | | | | |
| 0x0796 | SSP2BUF | 7:0 | BUF[7:0] | | | | | | | |
| 0x0797 | SSP2ADD | 7:0 | ADD[7:0] | | | | | | | |
| 0x0798 | SSP2MSK | 7:0 | MSK[6:0] | | | | | | | MSK0 |
| 0x0799 | SSP2STAT | 7:0 | SMP | CKE | D/$\overline{A}$ | P | S | R/$\overline{W}$ | UA | BF |
| 0x079A | SSP2CON1 | 7:0 | WCOL | SSPOV | SSPEN | CKP | SSPM[3:0] | | | |
| 0x079B | SSP2CON2 | 7:0 | GCEN | ACKSTAT | ACKDT | ACKEN | RCEN | PEN | RSEN | SEN |
| 0x079C | SSP2CON3 | 7:0 | ACKTIM | PCIE | SCIE | BOEN | SDAHT | SBCDE | AHEN | DHEN |
| 0x079D ... 0x080B | Reserved | | | | | | | | | |
| 0x080C | CM1CON0 | 7:0 | EN | OUT | | POL | | | HYS | SYNC |
| 0x080D | CM1CON1 | 7:0 | | | | | | | INTP | INTN |
| 0x080E | CM1NCH | 7:0 | | | | | | NCH[2:0] | | |
| 0x080F | CM1PCH | 7:0 | | | | | | PCH[2:0] | | |
| 0x0810 | CM2CON0 | 7:0 | EN | OUT | | POL | | | HYS | SYNC |
| 0x0811 | CM2CON1 | 7:0 | | | | | | | INTP | INTN |
| 0x0812 | CM2NCH | 7:0 | | | | | | NCH[2:0] | | |
| 0x0813 | CM2PCH | 7:0 | | | | | | PCH[2:0] | | |
| 0x0814 ... 0x081E | Reserved | | | | | | | | | |
| 0x081F | CMOUT | 7:0 | | | | | | | C2OUT | C1OUT |
| 0x0820 ... 0x088B | Reserved | | | | | | | | | |
| 0x088C | DAC1CON | 7:0 | EN | REFRNG | OE[1:0] | | PSS[1:0] | | | NSS |
| 0x088D | DAC1DATL | 7:0 | DAC1R[7:0] | | | | | | | |

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x088E ... 0x088F | Reserved | | | | | | | | | |
| 0x0890 | DAC2CON | 7:0 | EN | | | | PSS[1:0] | | | NSS |
| 0x0891 | DAC2DATL | 7:0 | DAC2R[7:0] | | | | | | | |
| 0x0892 ... 0x090B | Reserved | | | | | | | | | |
| 0x090C | OPA1CON0 | 7:0 | EN | | CPON | | UG | | SOC[1:0] | |
| 0x090D | OPA1CON1 | 7:0 | | GSEL[2:0] | | | RESON | NSS[2:0] | | |
| 0x090E | OPA1CON2 | 7:0 | | NCH[2:0] | | | | PCH[2:0] | | |
| 0x090F | OPA1CON3 | 7:0 | FMS[1:0] | | | | | | PSS[1:0] | |
| 0x0910 | OPA1HWC | 7:0 | OREN | HWCH[2:0] | | | ORPOL | HWCL[2:0] | | |
| 0x0911 | OPA1OFFSET | 7:0 | OFFSET[7:0] | | | | | | | |
| 0x0912 | OPA1ORS | 7:0 | | | | ORS[4:0] | | | | |
| 0x0913 ... 0x098B | Reserved | | | | | | | | | |
| 0x098C | PWM3ERS | 7:0 | | | | | ERS[3:0] | | | |
| 0x098D | PWM3CLK | 7:0 | | | | | CLK[3:0] | | | |
| 0x098E | PWM3LDS | 7:0 | | | | | | LDS[2:0] | | |
| 0x098F | PWM3PR | 7:0 | PR[7:0] | | | | | | | |
| | | 15:8 | PR[15:8] | | | | | | | |
| 0x0991 | PWM3CPRE | 7:0 | CPRE[7:0] | | | | | | | |
| 0x0992 | PWM3PIPOS | 7:0 | PIPOS[7:0] | | | | | | | |
| 0x0993 | PWM3GIR | 7:0 | | | | | | | S1P2 | S1P1 |
| 0x0994 | PWM3GIE | 7:0 | | | | | | | S1P2 | S1P1 |
| 0x0995 | PWM3CON | 7:0 | EN | | | | | LD | ERSPOL | ERSNOW |
| 0x0996 | PWM3S1CFG | 7:0 | POL2 | POL1 | | | PPEN | MODE[2:0] | | |
| 0x0997 | PWM3S1P1 | 7:0 | P1[7:0] | | | | | | | |
| | | 15:8 | P1[15:8] | | | | | | | |
| 0x0999 | PWM3S1P2 | 7:0 | P2[7:0] | | | | | | | |
| | | 15:8 | P2[15:8] | | | | | | | |
| 0x099B ... 0x0A0B | Reserved | | | | | | | | | |
| 0x0A0C | PWM4ERS | 7:0 | | | | | ERS[3:0] | | | |
| 0x0A0D | PWM4CLK | 7:0 | | | | | CLK[3:0] | | | |
| 0x0A0E | PWM4LDS | 7:0 | | | | | | LDS[2:0] | | |
| 0x0A0F | PWM4PR | 7:0 | PR[7:0] | | | | | | | |
| | | 15:8 | PR[15:8] | | | | | | | |
| 0x0A11 | PWM4CPRE | 7:0 | CPRE[7:0] | | | | | | | |
| 0x0A12 | PWM4PIPOS | 7:0 | PIPOS[7:0] | | | | | | | |
| 0x0A13 | PWM4GIR | 7:0 | | | | | | | S1P2 | S1P1 |
| 0x0A14 | PWM4GIE | 7:0 | | | | | | | S1P2 | S1P1 |
| 0x0A15 | PWM4CON | 7:0 | EN | | | | | LD | ERSPOL | ERSNOW |
| 0x0A16 | PWM4S1CFG | 7:0 | POL2 | POL1 | | | PPEN | MODE[2:0] | | |
| 0x0A17 | PWM4S1P1 | 7:0 | P1[7:0] | | | | | | | |
| | | 15:8 | P1[15:8] | | | | | | | |
| 0x0A19 | PWM4S1P2 | 7:0 | P2[7:0] | | | | | | | |
| | | 15:8 | P2[15:8] | | | | | | | |
| 0x0A1B ... 0x1C8B | Reserved | | | | | | | | | |
| 0x1C8C | NVMADR | 7:0 | NVMADR[7:0] | | | | | | | |
| | | 15:8 | | NVMADR[14:8] | | | | | | |
| 0x1C8E | NVMDAT | 7:0 | NVMDAT[7:0] | | | | | | | |
| | | 15:8 | | | NVMDAT[13:8] | | | | | |
| 0x1C90 | NVMCON1 | 7:0 | | NVMREGS | LWLO | FREE | WRERR | WREN | WR | RD |

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x1C91 | NVMCON2 | 7:0 | NVMCON2[7:0] | | | | | | | |
| 0x1C92 | SCANHADR | 7:0 | SCANHADRL[7:0] | | | | | | | |
| | | 15:8 | SCANHADRH[7:0] | | | | | | | |
| 0x1C94 | Reserved | | | | | | | | | |
| 0x1C95 | SCANLADR | 7:0 | SCANLADRL[7:0] | | | | | | | |
| | | 15:8 | SCANLADRH[7:0] | | | | | | | |
| 0x1C97 | Reserved | | | | | | | | | |
| 0x1C98 | SCANCON0 | 7:0 | EN | SGO | BUSY | DABORT | INTM | | MD[1:0] | |
| 0x1C99 | SCANTRIG | 7:0 | | | | | TSEL[3:0] | | | |
| 0x1C9A | CRCDATA | 7:0 | CRCDATAL[7:0] | | | | | | | |
| | | 15:8 | CRCDATAH[7:0] | | | | | | | |
| | | 23:16 | CRCDATAU[7:0] | | | | | | | |
| | | 31:24 | CRCDATAT[7:0] | | | | | | | |
| 0x1C9E | CRCOUT | 7:0 | CRCOUTL[7:0] | | | | | | | |
| | | 15:8 | CRCOUTH[7:0] | | | | | | | |
| | | 23:16 | CRCOUTU[7:0] | | | | | | | |
| | | 31:24 | CRCOUTT[7:0] | | | | | | | |
| 0x1C9E | CRCSHIFT | 7:0 | CRCSHIFTL[7:0] | | | | | | | |
| | | 15:8 | CRCSHIFTH[7:0] | | | | | | | |
| | | 23:16 | CRCSHIFTU[7:0] | | | | | | | |
| | | 31:24 | CRCSHIFTT[7:0] | | | | | | | |
| 0x1C9E | CRCXOR | 7:0 | CRCXORL[7:0] | | | | | | | |
| | | 15:8 | CRCXORH[7:0] | | | | | | | |
| | | 23:16 | CRCXORU[7:0] | | | | | | | |
| | | 31:24 | CRCXORT[7:0] | | | | | | | |
| 0x1CA2 | CRCCON0 | 7:0 | EN | GO | BUSY | ACCM | SETUP[1:0] | | SHIFTM | FULL |
| 0x1CA3 | CRCCON1 | 7:0 | | | | PLEN[4:0] | | | | |
| 0x1CA4 | CRCCON2 | 7:0 | | | | DLEN[4:0] | | | | |
| 0x1CA5 ... 0x1D0B | Reserved | | | | | | | | | |
| 0x1D0C | ADLTH | 7:0 | LTH[7:0] | | | | | | | |
| | | 15:8 | LTH[15:8] | | | | | | | |
| 0x1D0E | ADUTH | 7:0 | UTH[7:0] | | | | | | | |
| | | 15:8 | UTH[15:8] | | | | | | | |
| 0x1D10 | ADERR | 7:0 | ERR[7:0] | | | | | | | |
| | | 15:8 | ERR[15:8] | | | | | | | |
| 0x1D12 | ADSTPT | 7:0 | STPT[7:0] | | | | | | | |
| | | 15:8 | STPT[15:8] | | | | | | | |
| 0x1D14 | ADFLTR | 7:0 | FLTR[7:0] | | | | | | | |
| | | 15:8 | FLTR[15:8] | | | | | | | |
| 0x1D16 | ADACC | 7:0 | ACC[7:0] | | | | | | | |
| | | 15:8 | ACC[15:8] | | | | | | | |
| | | 23:16 | | | | | | | ACC[17:16] | |
| 0x1D19 | ADCNT | 7:0 | CNT[7:0] | | | | | | | |
| 0x1D1A | ADRPT | 7:0 | RPT[7:0] | | | | | | | |
| 0x1D1B | ADPREV | 7:0 | PREV[7:0] | | | | | | | |
| | | 15:8 | PREV[15:8] | | | | | | | |
| 0x1D1D | ADRES | 7:0 | RES[7:0] | | | | | | | |
| | | 15:8 | RES[15:8] | | | | | | | |
| 0x1D1F | ADPCH | 7:0 | | | PCH[5:0] | | | | | |
| 0x1D20 | ADNCH | 7:0 | | | NCH[5:0] | | | | | |
| 0x1D21 | ADACQ | 7:0 | ACQ[7:0] | | | | | | | |
| | | 15:8 | | | | ACQ[12:8] | | | | |
| 0x1D23 | ADCAP | 7:0 | | | | CAP[4:0] | | | | |
| 0x1D24 | ADPRE | 7:0 | PRE[7:0] | | | | | | | |
| | | 15:8 | | | | PRE[12:8] | | | | |
| 0x1D26 | ADCON0 | 7:0 | ON | CONT | | CS | FM[1:0] | | IC | GO |
| 0x1D27 | ADCON1 | 7:0 | PPOL | IPEN | GPOL | | | | PCSC | DSEN |

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x1D28 | ADCON2 | 7:0 | PSIS | | CRS[2:0] | | ACLR | | MD[2:0] | |
| 0x1D29 | ADCON3 | 7:0 | | | CALC[2:0] | | SOI | | TMD[2:0] | |
| 0x1D2A | ADSTAT | 7:0 | AOV | UTHR | LTHR | MATH | | | STAT[2:0] | |
| 0x1D2B | ADREF | 7:0 | | | | NREF | | | PREF[1:0] | |
| 0x1D2C | ADACT | 7:0 | | | | ACT[4:0] | | | | |
| 0x1D2D | ADCLK | 7:0 | | | CS[5:0] | | | | | |
| 0x1D2E | ADCG1A | 7:0 | CGA7 | CGA6 | CGA5 | CGA4 | CGA3 | CGA2 | CGA1 | CGA0 |
| 0x1D2F | ADCG1B | 7:0 | CGB7 | CGB6 | CGB5 | CGB4 | CGB3 | CGB2 | CGB1 | CGB0 |
| 0x1D30 | ADCG1C | 7:0 | CGC7 | CGC6 | CGC5 | CGC4 | CGC3 | CGC2 | CGC1 | CGC0 |
| 0x1D31 | ADCG1D | 7:0 | CGD7 | CGD6 | CGD5 | CGD4 | CGD3 | CGD2 | CGD1 | CGD0 |
| 0x1D32 | ADCG1E | 7:0 | | | | | | CGE2 | CGE1 | CGE0 |
| 0x1D33 ... 0x1D8B | Reserved | | | | | | | | | |
| 0x1D8C | RA0PPS | 7:0 | | | RA0PPS[5:0] | | | | | |
| 0x1D8D | RA1PPS | 7:0 | | | RA1PPS[5:0] | | | | | |
| 0x1D8E | RA2PPS | 7:0 | | | RA2PPS[5:0] | | | | | |
| 0x1D8F | RA3PPS | 7:0 | | | RA3PPS[5:0] | | | | | |
| 0x1D90 | RA4PPS | 7:0 | | | RA4PPS[5:0] | | | | | |
| 0x1D91 | RA5PPS | 7:0 | | | RA5PPS[5:0] | | | | | |
| 0x1D92 | RA6PPS | 7:0 | | | RA6PPS[5:0] | | | | | |
| 0x1D93 | RA7PPS | 7:0 | | | RA7PPS[5:0] | | | | | |
| 0x1D94 | RB0PPS | 7:0 | | | RB0PPS[5:0] | | | | | |
| 0x1D95 | RB1PPS | 7:0 | | | RB1PPS[5:0] | | | | | |
| 0x1D96 | RB2PPS | 7:0 | | | RB2PPS[5:0] | | | | | |
| 0x1D97 | RB3PPS | 7:0 | | | RB3PPS[5:0] | | | | | |
| 0x1D98 | RB4PPS | 7:0 | | | RB4PPS[5:0] | | | | | |
| 0x1D99 | RB5PPS | 7:0 | | | RB5PPS[5:0] | | | | | |
| 0x1D9A | RB6PPS | 7:0 | | | RB6PPS[5:0] | | | | | |
| 0x1D9B | RB7PPS | 7:0 | | | RB7PPS[5:0] | | | | | |
| 0x1D9C | RC0PPS | 7:0 | | | RC0PPS[5:0] | | | | | |
| 0x1D9D | RC1PPS | 7:0 | | | RC1PPS[5:0] | | | | | |
| 0x1D9E | RC2PPS | 7:0 | | | RC2PPS[5:0] | | | | | |
| 0x1D9F | RC3PPS | 7:0 | | | RC3PPS[5:0] | | | | | |
| 0x1DA0 | RC4PPS | 7:0 | | | RC4PPS[5:0] | | | | | |
| 0x1DA1 | RC5PPS | 7:0 | | | RC5PPS[5:0] | | | | | |
| 0x1DA2 | RC6PPS | 7:0 | | | RC6PPS[5:0] | | | | | |
| 0x1DA3 | RC7PPS | 7:0 | | | RC7PPS[5:0] | | | | | |
| 0x1DA4 | RD0PPS | 7:0 | | | RD0PPS[5:0] | | | | | |
| 0x1DA5 | RD1PPS | 7:0 | | | RD1PPS[5:0] | | | | | |
| 0x1DA6 | RD2PPS | 7:0 | | | RD2PPS[5:0] | | | | | |
| 0x1DA7 | RD3PPS | 7:0 | | | RD3PPS[5:0] | | | | | |
| 0x1DA8 | RD4PPS | 7:0 | | | RD4PPS[5:0] | | | | | |
| 0x1DA9 | RD5PPS | 7:0 | | | RD5PPS[5:0] | | | | | |
| 0x1DAA | RD6PPS | 7:0 | | | RD6PPS[5:0] | | | | | |
| 0x1DAB | RD7PPS | 7:0 | | | RD7PPS[5:0] | | | | | |
| 0x1DAC | RE0PPS | 7:0 | | | RE0PPS[5:0] | | | | | |
| 0x1DAD | RE1PPS | 7:0 | | | RE1PPS[5:0] | | | | | |
| 0x1DAE | RE2PPS | 7:0 | | | RE2PPS[5:0] | | | | | |
| 0x1DAF ... 0x1E0B | Reserved | | | | | | | | | |
| 0x1E0C | PPSLOCK | 7:0 | | | | | | | | PPSLOCKED |
| 0x1E0D | INTPPS | 7:0 | | | PORT[2:0] | | | PIN[2:0] | | |
| 0x1E0E | T0CKIPPS | 7:0 | | | PORT[2:0] | | | PIN[2:0] | | |
| 0x1E0F | T1CKIPPS | 7:0 | | | PORT[2:0] | | | PIN[2:0] | | |
| 0x1E10 | T1GPPS | 7:0 | | | PORT[2:0] | | | PIN[2:0] | | |
| 0x1E11 | T3CKIPPS | 7:0 | | | PORT[2:0] | | | PIN[2:0] | | |
| 0x1E12 | T3GPPS | 7:0 | | | PORT[2:0] | | | PIN[2:0] | | |

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x1E13 ... 0x1E18 | Reserved | | | | | | | | | |
| 0x1E19 | T2INPPS | 7:0 | | | | PORT[2:0] | | | PIN[2:0] | |
| 0x1E1A | T4INPPS | 7:0 | | | | PORT[2:0] | | | PIN[2:0] | |
| 0x1E1B | T6INPPS | 7:0 | | | | PORT[2:0] | | | PIN[2:0] | |
| 0x1E1C ... 0x1E1D | Reserved | | | | | | | | | |
| 0x1E1E | CCP1PPS | 7:0 | | | | PORT[2:0] | | | PIN[2:0] | |
| 0x1E1F | CCP2PPS | 7:0 | | | | PORT[2:0] | | | PIN[2:0] | |
| 0x1E20 ... 0x1E23 | Reserved | | | | | | | | | |
| 0x1E24 | PWMIN0PPS | 7:0 | | | | PORT[2:0] | | | PIN[2:0] | |
| 0x1E25 | PWMIN1PPS | 7:0 | | | | PORT[2:0] | | | PIN[2:0] | |
| 0x1E26 | PWM1ERSPPS | 7:0 | | | | PORT[2:0] | | | PIN[2:0] | |
| 0x1E27 | PWM2ERSPPS | 7:0 | | | | PORT[2:0] | | | PIN[2:0] | |
| 0x1E28 | PWM3ERSPPS | 7:0 | | | | PORT[2:0] | | | PIN[2:0] | |
| 0x1E29 | PWM4ERSPPS | 7:0 | | | | PORT[2:0] | | | PIN[2:0] | |
| 0x1E2A ... 0x1E38 | Reserved | | | | | | | | | |
| 0x1E39 | CWG1PPS | 7:0 | | | | PORT[2:0] | | | PIN[2:0] | |
| 0x1E3A ... 0x1E3C | Reserved | | | | | | | | | |
| 0x1E3D | CLCIN0PPS | 7:0 | | | | PORT[2:0] | | | PIN[2:0] | |
| 0x1E3E | CLCIN1PPS | 7:0 | | | | PORT[2:0] | | | PIN[2:0] | |
| 0x1E3F | CLCIN2PPS | 7:0 | | | | PORT[2:0] | | | PIN[2:0] | |
| 0x1E40 | CLCIN3PPS | 7:0 | | | | PORT[2:0] | | | PIN[2:0] | |
| 0x1E41 | RX1PPS | 7:0 | | | | PORT[2:0] | | | PIN[2:0] | |
| 0x1E42 | CK1PPS | 7:0 | | | | PORT[2:0] | | | PIN[2:0] | |
| 0x1E43 | RX2PPS | 7:0 | | | | PORT[2:0] | | | PIN[2:0] | |
| 0x1E44 | CK2PPS | 7:0 | | | | PORT[2:0] | | | PIN[2:0] | |
| 0x1E45 ... 0x1E46 | Reserved | | | | | | | | | |
| 0x1E47 | SSP1CLKPPS | 7:0 | | | | PORT[2:0] | | | PIN[2:0] | |
| 0x1E48 | SSP1DATPPS | 7:0 | | | | PORT[2:0] | | | PIN[2:0] | |
| 0x1E49 | SSP1SSPPS | 7:0 | | | | PORT[2:0] | | | PIN[2:0] | |
| 0x1E4A | SSP2CLKPPS | 7:0 | | | | PORT[2:0] | | | PIN[2:0] | |
| 0x1E4B | SSP2DATPPS | 7:0 | | | | PORT[2:0] | | | PIN[2:0] | |
| 0x1E4C | SSP2SSPPS | 7:0 | | | | PORT[2:0] | | | PIN[2:0] | |
| 0x1E4D ... 0x1E4F | Reserved | | | | | | | | | |
| 0x1E50 | ADACTPPS | 7:0 | | | | PORT[2:0] | | | PIN[2:0] | |
| 0x1E51 ... 0x1E52 | Reserved | | | | | | | | | |
| 0x1E53 | OPA1PPS | 7:0 | | | | PORT[2:0] | | | PIN[2:0] | |
| 0x1E54 ... 0x1E8B | Reserved | | | | | | | | | |
| 0x1E8C | ANSELA | 7:0 | ANSELA7 | ANSELA6 | ANSELA5 | ANSELA4 | ANSELA3 | ANSELA2 | ANSELA1 | ANSELA0 |
| 0x1E8D | WPUA | 7:0 | WPUA7 | WPUA6 | WPUA5 | WPUA4 | WPUA3 | WPUA2 | WPUA1 | WPUA0 |
| 0x1E8E | ODCONA | 7:0 | ODCA7 | ODCA6 | ODCA5 | ODCA4 | ODCA3 | ODCA2 | ODCA1 | ODCA0 |
| 0x1E8F | SLRCONA | 7:0 | SLRA7 | SLRA6 | SLRA5 | SLRA4 | SLRA3 | SLRA2 | SLRA1 | SLRA0 |

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x1E90 | INLVLA | 7:0 | INLVLA7 | INLVLA6 | INLVLA5 | INLVLA4 | INLVLA3 | INLVLA2 | INLVLA1 | INLVLA0 |
| 0x1E91 | IOCAP | 7:0 | IOCAP7 | IOCAP6 | IOCAP5 | IOCAP4 | IOCAP3 | IOCAP2 | IOCAP1 | IOCAP0 |
| 0x1E92 | IOCAN | 7:0 | IOCAN7 | IOCAN6 | IOCAN5 | IOCAN4 | IOCAN3 | IOCAN2 | IOCAN1 | IOCAN0 |
| 0x1E93 | IOCAF | 7:0 | IOCAF7 | IOCAF6 | IOCAF5 | IOCAF4 | IOCAF3 | IOCAF2 | IOCAF1 | IOCAF0 |
| 0x1E94 ... 0x1E95 | Reserved | | | | | | | | | |
| 0x1E96 | ANSELB | 7:0 | ANSELB7 | ANSELB6 | ANSELB5 | ANSELB4 | ANSELB3 | ANSELB2 | ANSELB1 | ANSELB0 |
| 0x1E97 | WPUB | 7:0 | WPUB7 | WPUB6 | WPUB5 | WPUB4 | WPUB3 | WPUB2 | WPUB1 | WPUB0 |
| 0x1E98 | ODCONB | 7:0 | ODCB7 | ODCB6 | ODCB5 | ODCB4 | ODCB3 | ODCB2 | ODCB1 | ODCB0 |
| 0x1E99 | SLRCONB | 7:0 | SLRB7 | SLRB6 | SLRB5 | SLRB4 | SLRB3 | SLRB2 | SLRB1 | SLRB0 |
| 0x1E9A | INLVLB | 7:0 | INLVLB7 | INLVLB6 | INLVLB5 | INLVLB4 | INLVLB3 | INLVLB2 | INLVLB1 | INLVLB0 |
| 0x1E9B | IOCBP | 7:0 | IOCBP7 | IOCBP6 | IOCBP5 | IOCBP4 | IOCBP3 | IOCBP2 | IOCBP1 | IOCBP0 |
| 0x1E9C | IOCBN | 7:0 | IOCBN7 | IOCBN6 | IOCBN5 | IOCBN4 | IOCBN3 | IOCBN2 | IOCBN1 | IOCBN0 |
| 0x1E9D | IOCBF | 7:0 | IOCBF7 | IOCBF6 | IOCBF5 | IOCBF4 | IOCBF3 | IOCBF2 | IOCBF1 | IOCBF0 |
| 0x1E9E ... 0x1E9F | Reserved | | | | | | | | | |
| 0x1EA0 | ANSELC | 7:0 | ANSELC7 | ANSELC6 | ANSELC5 | ANSELC4 | ANSELC3 | ANSELC2 | ANSELC1 | ANSELC0 |
| 0x1EA1 | WPUC | 7:0 | WPUC7 | WPUC6 | WPUC5 | WPUC4 | WPUC3 | WPUC2 | WPUC1 | WPUC0 |
| 0x1EA2 | ODCONC | 7:0 | ODCC7 | ODCC6 | ODCC5 | ODCC4 | ODCC3 | ODCC2 | ODCC1 | ODCC0 |
| 0x1EA3 | SLRCONC | 7:0 | SLRC7 | SLRC6 | SLRC5 | SLRC4 | SLRC3 | SLRC2 | SLRC1 | SLRC0 |
| 0x1EA4 | INLVLC | 7:0 | INLVLC7 | INLVLC6 | INLVLC5 | INLVLC4 | INLVLC3 | INLVLC2 | INLVLC1 | INLVLC0 |
| 0x1EA5 | IOCCP | 7:0 | IOCCP7 | IOCCP6 | IOCCP5 | IOCCP4 | IOCCP3 | IOCCP2 | IOCCP1 | IOCCP0 |
| 0x1EA6 | IOCCN | 7:0 | IOCCN7 | IOCCN6 | IOCCN5 | IOCCN4 | IOCCN3 | IOCCN2 | IOCCN1 | IOCCN0 |
| 0x1EA7 | IOCCF | 7:0 | IOCCF7 | IOCCF6 | IOCCF5 | IOCCF4 | IOCCF3 | IOCCF2 | IOCCF1 | IOCCF0 |
| 0x1EA8 ... 0x1EA9 | Reserved | | | | | | | | | |
| 0x1EAA | ANSELD | 7:0 | ANSELD7 | ANSELD6 | ANSELD5 | ANSELD4 | ANSELD3 | ANSELD2 | ANSELD1 | ANSELD0 |
| 0x1EAB | WPUD | 7:0 | WPUD7 | WPUD6 | WPUD5 | WPUD4 | WPUD3 | WPUD2 | WPUD1 | WPUD0 |
| 0x1EAC | ODCOND | 7:0 | ODCD7 | ODCD6 | ODCD5 | ODCD4 | ODCD3 | ODCD2 | ODCD1 | ODCD0 |
| 0x1EAD | SLRCOND | 7:0 | SLRD7 | SLRD6 | SLRD5 | SLRD4 | SLRD3 | SLRD2 | SLRD1 | SLRD0 |
| 0x1EAE | INLVLD | 7:0 | INLVLD7 | INLVLD6 | INLVLD5 | INLVLD4 | INLVLD3 | INLVLD2 | INLVLD1 | INLVLD0 |
| 0x1EAF ... 0x1EB3 | Reserved | | | | | | | | | |
| 0x1EB4 | ANSELE | 7:0 | | | | | | ANSELE2 | ANSELE1 | ANSELE0 |
| 0x1EB5 | WPUE | 7:0 | | | | | WPUE3 | WPUE2 | WPUE1 | WPUE0 |
| 0x1EB6 | ODCONE | 7:0 | | | | | | ODCE2 | ODCE1 | ODCE0 |
| 0x1EB7 | SLRCONE | 7:0 | | | | | | SLRE2 | SLRE1 | SLRE0 |
| 0x1EB8 | INLVLE | 7:0 | | | | | INLVLE3 | INLVLE2 | INLVLE1 | INLVLE0 |
| 0x1EB9 | IOCEP | 7:0 | | | | | IOCEP3 | | | |
| 0x1EBA | IOCEN | 7:0 | | | | | IOCEN3 | | | |
| 0x1EBB | IOCEF | 7:0 | | | | | IOCEF3 | | | |
| 0x1EBC ... 0x1EE2 | Reserved | | | | | | | | | |
| 0x1EE3 | RB1I2C | 7:0 | | SLEW | PU[1:0] | | | | TH[1:0] | |
| 0x1EE4 | RB2I2C | 7:0 | | SLEW | PU[1:0] | | | | TH[1:0] | |
| 0x1EE5 ... 0x1EEA | Reserved | | | | | | | | | |
| 0x1EEB | RC3I2C | 7:0 | | SLEW | PU[1:0] | | | | TH[1:0] | |
| 0x1EEC | RC4I2C | 7:0 | | SLEW | PU[1:0] | | | | TH[1:0] | |
| 0x1EED | Reserved | | | | | | | | | |
| 0x1EEE | RD0I2C | 7:0 | | SLEW | PU[1:0] | | | | TH[1:0] | |
| 0x1EEF | RD1I2C | 7:0 | | SLEW | PU[1:0] | | | | TH[1:0] | |

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x1EF0 ... 0x8004 | Reserved | | | | | | | | | |
| 0x8005 | REVISIONID | 7:0 | MJRREV[1:0] | | MNRREV[5:0] | | | | | |
| | | 15:8 | | | Reserved | Reserved | MJRREV[5:2] | | | |
| 0x8006 | DEVICEID | 7:0 | DEV[7:0] | | | | | | | |
| | | 15:8 | | | Reserved | Reserved | DEV[11:8] | | | |
| 0x8007 | CONFIG1 | 7:0 | | RSTOSC[2:0] | | | | FEXTOSC[2:0] | | |
| | | 15:8 | | | FCMEN | VDDAR | CSWEN | | | CLKOUTEN |
| 0x8008 | CONFIG2 | 7:0 | BOREN[1:0] | | LPBOREN | | | PWRTS[1:0] | | MCLRE |
| | | 15:8 | | | DEBUG | STVREN | PPS1WAY | ZCD | BORV | DACAUTOEN |
| 0x8009 | CONFIG3 | 7:0 | | WDTE[1:0] | | WDTCPS[4:0] | | | | |
| | | 15:8 | | | | WDTCCS[2:0] | | WDTCWS[2:0] | | |
| 0x800A | CONFIG4 | 7:0 | WRTAPP | | | SAFEN | BBEN | BBSIZE[2:0] | | |
| | | 15:8 | | | LVP | | WRTSAF | WRTD | WRTC | WRTB |
| 0x800B | CONFIG5 | 7:0 | | | | | | | CPD | CP |
| | | 15:8 | | | | | | | | |

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



# 44. Electrical Specifications

## 44.1 Absolute Maximum Ratings(†)

| Parameter | | Rating |
|---|---|---|
| **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:** | | -0.3V to +6.5V |
| • **on $\overline{MCLR}$ pin:** | | -0.3V to +9.0V |
| • **on all other pins:** | | -0.3V to ($V_{DD}$ + 0.3V) |
| **Maximum current(1)** | | |
| • **on $V_{SS}$ pin** | -40°C ≤ $T_A$ ≤ +85°C | 300 mA |
| | 85°C < $T_A$ ≤ +125°C | 120 mA |
| • **on $V_{DD}$ pin** | -40°C ≤ $T_A$ ≤ +85°C | 250 mA |
| | 85°C < $T_A$ ≤ +125°C | 85 mA |
| • **on any standard I/O pin** | | ±25 mA |
| **Clamp current, $I_K$ ($V_{PIN}$ < 0 or $V_{PIN}$ > $V_{DD}$)** | | ±20 mA |
| **Total power dissipation(2)** | | 800 mW |

**Notes:**

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 44.3.6. Thermal Characteristics to calculate device specifications.
2. Power dissipation is calculated as follows:
   $P_{DIS} = V_{DD} \times \{I_{DD} - \Sigma I_{OH}\} + \Sigma \{(V_{DD} - V_{OH}) \times I_{OH}\} + \Sigma (V_{OI} \times I_{OL})$
3. Internal Power Dissipation is calculated as follows:
   $P_{INTERNAL} = I_{DD} \times V_{DD}$

   where $I_{DD}$ is current to run the chip alone without driving any load on the output pins.
4. I/O Power Dissipation is calculated as follows:
   $P_{I/O} = \Sigma(I_{OL}*V_{OL})+\Sigma(I_{OH}*(V_{DD}-V_{OH}))$
5. Derated Power is calculated as follows:
   $P_{DER} = PD_{MAX}(T_J-T_A)/\theta_{JA}$

   where $T_A$ = Ambient Temperature, $T_J$ = Junction Temperature.

**CAUTION**

**Notice:** Stresses above those listed under the *Absolute Maximum Ratings* section 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.

## 44.2 Standard Operating Conditions

The standard operating conditions for any device are defined as:

chanhvinh.com



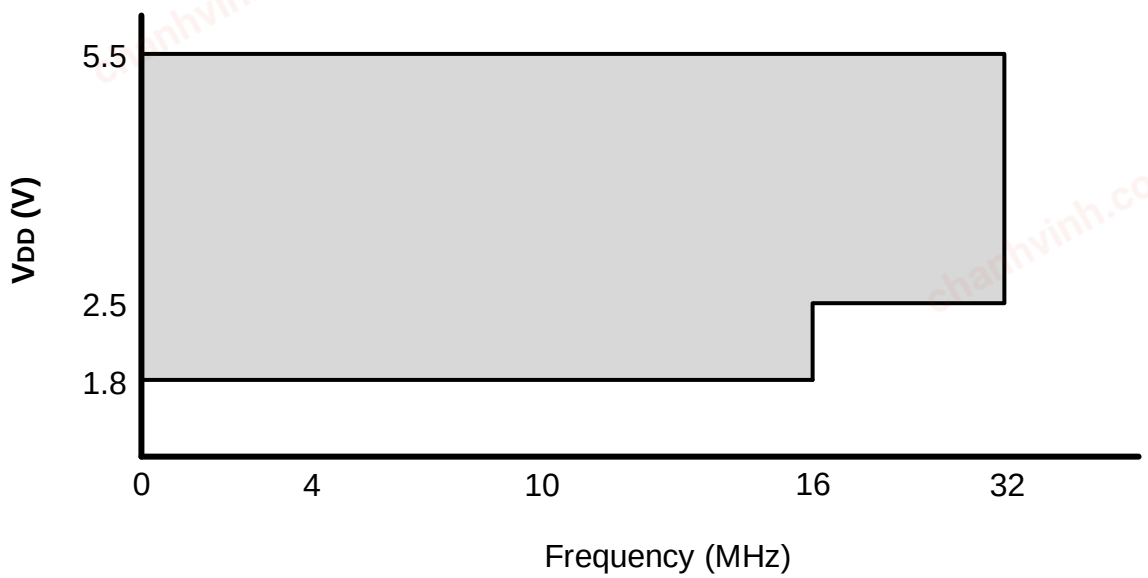
| Parameter | Condition |
|---|---|
| Operating Voltage: | $V_{DDMIN} \le V_{DD} \le V_{DDMAX}$ |
| Operating Temperature: | $T_{AMIN} \le T_A \le T_{AMAX}$ |

| Parameter | | Ratings |
|---|---|---|
| **$V_{DD}$ — Operating Supply Voltage(1)** | | |
| | $V_{DDMIN}$ ($F_{OSC} \le 16$ MHz) | +1.8V |
| | $V_{DDMIN}$ ($F_{OSC} \le 32$ 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 **D002**, DC Characteristics: Supply Voltage.

**Figure 44-1. Voltage Frequency Graph, -40°C ≤ $T_A$ ≤ +125°C**




**Notes:**

1. The shaded region indicates the permissible combinations of voltage and frequency.
2. Refer to the **"External Clock/Oscillator Timing Requirements"** section for each Oscillator mode's supported frequencies.

chanhvinh.com



## 44.3 DC Characteristics

### 44.3.1 Supply Voltage

**Table 44-1.**

| Standard Operating Conditions (unless otherwise stated) | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Param. No.** | **Sym.** | **Characteristic** | **Min.** | **Typ.†** | **Max.** | **Units** | **Conditions** |
| **Supply Voltage** | | | | | | | |
| D002 | $V_{DD}$ | | 1.8 | — | 5.5 | V | $F_{OSC}$ ≤ 16 MHz |
| | | | 2.5 | — | 5.5 | V | $F_{OSC}$ > 16 MHz |
| **RAM Data Retention(1)** | | | | | | | |
| D003 | $V_{DR}$ | | 1.7 | — | — | V | Device in Sleep mode |
| **Power-on Reset Release Voltage(2)** | | | | | | | |
| D004 | $V_{POR}$ | | — | 1.6 | — | V | BOR and LPBOR disabled(3) |
| **Power-on Reset Rearm Voltage(2)** | | | | | | | |
| D005 | $V_{PORR}$ | | — | 1 | — | V | BOR and LPBOR disabled(3) |
| **$V_{DD}$ Rise Rate to ensure internal Power-on Reset signal(2)** | | | | | | | |
| D006 | $S_{VDD}$ | | 0.05 | — | — | V/ms | BOR and LPBOR disabled(3) |

† Data in “Typ.” column is at 3.0V, 25℃ unless otherwise stated. These parameters are for design guidance only and are not tested.

**Notes:**

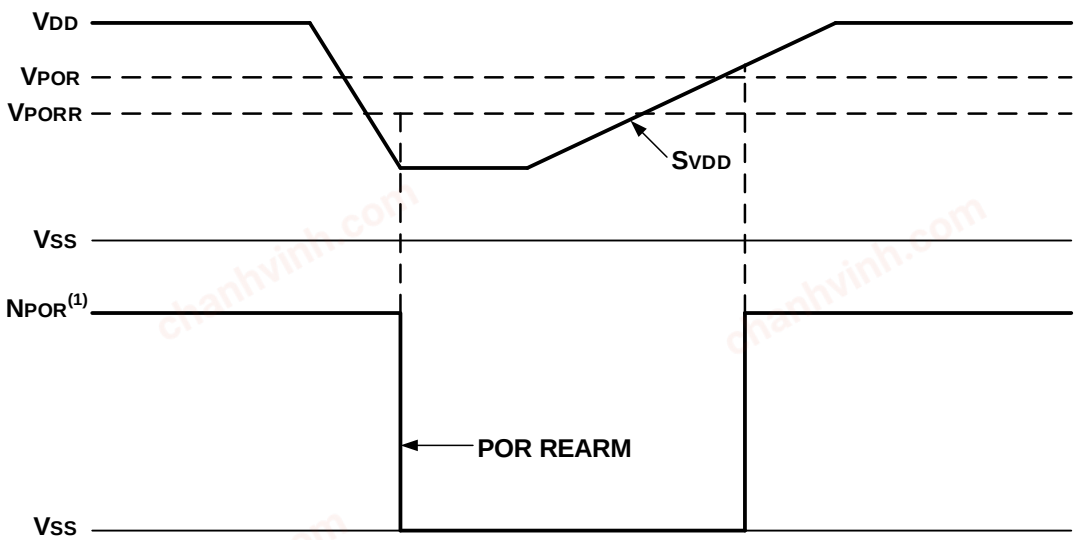
1. This is the limit to which $V_{DD}$ can be lowered in Sleep mode without losing RAM data.
2. See the following figure, POR and POR REARM with Slow Rising $V_{DD}$.
3. See the **“Reset, WDT, Oscillator Start-up Timer, Brown-Out Reset and Low-Power Brown-Out Reset Specifications”** section for BOR and LPBOR trip point information.

chanhvinh.com

chanhvinh.com



**Figure 44-2. POR and POR Rearm with Slow Rising $V_{DD}$**




**Note:**

1. When $N_{POR}$ is low, the device is held in Reset.

### 44.3.2 Supply Current ($I_{DD}$)(1,2)

**Table 44-2.**

| Standard Operating Conditions (unless otherwise stated) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| Param. No. | Sym. | Device Characteristics | Min. | Typ.† | Max. | Units | Conditions | |
| | | | | | | | $V_{DD}$ | Note |
| D100 | $I_{DD_{XT4}}$ | XT = 4 MHz | — | 660 | 859 | µA | 3.0V | |
| D100A | $I_{DD_{XT4}}$ | XT = 4 MHz | — | 504 | 730 | µA | 3.0V | PMD bits all 1 |
| D101 | $I_{DD_{HFO16}}$ | HFINTOSC = 16 MHz | — | 2.4 | 3.2 | mA | 3.0V | |
| D101A | $I_{DD_{HFO16}}$ | HFINTOSC = 16 MHz | — | 1.8 | 2.5 | mA | 3.0V | PMD bits all 1 |
| D102 | $I_{DD_{HFOPLL}}$ | HFINTOSC = 32 MHz | — | 4.7 | 6.1 | mA | 3.0V | |
| D102A | $I_{DD_{HFOPLL}}$ | HFINTOSC = 32 MHz | — | 3.3 | 4.6 | mA | 3.0V | PMD bits all 1 |
| D103 | $I_{DD_{HSPLL64}}$ | HS+PLL = 32 MHz | — | 4.9 | 6.0 | mA | 3.0V | |
| D103A | $I_{DD_{HSPLL64}}$ | HS+PLL = 32 MHz | — | 3.7 | 5.0 | mA | 3.0V | PMD bits all 1 |

chanhvinh.com



.....continued

| Standard Operating Conditions (unless otherwise stated) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| Param. No. | Sym. | Device Characteristics | Min. | Typ.† | Max. | Units | Conditions | |
| | | | | | | | $V_{DD}$ | Note |
| D104 | $I_{DD_{IDLE}}$ | Idle mode, HFINTOSC = 16 MHz | — | 1.4 | 2.7 | mA | 3.0V | |
| D105 | $I_{DD_{DOZE}}$ (3) | Doze mode, HFINTOSC = 16 MHz, Doze Ratio = 16 | — | 1.5 | 2.8 | mA | 3.0V | |

† Data in "Typ." column is at 3.0V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

**Notes:**

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 are outputs driven low; $\overline{MCLR}$ = $V_{DD}$; 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.
3. $I_{DD_{DOZE}}$ = [$I_{DD_{IDLE}}$*(N-1)/N] + $I_{DD_{HFO}}$16/N where N = Doze Ratio (see the **CPUDOZE** register).
4. PMD bits are all in the Default state, no modules are disabled.

### 44.3.3 Power-Down Current ($I_{PD}$)(1,2,3)

**Table 44-3.**

| Standard Operating Conditions (unless otherwise stated) | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| Param. No. | Sym. | Device Characteristics | Min. | Typ.† | Max. +85°C | Max. +125°C | Units | Conditions | |
| | | | | | | | | $V_{DD}$ | Note |
| D200 | $I_{PD}$ | $I_{PD}$ Base | — | 0.7 | 4 | 12 | µA | 3.0V | |
| D201 | $I_{PD_WDT}$ | Low-Frequency Internal Oscillator/WDT | — | 0.75 | 5 | 13 | µA | 3.0V | |
| D202 | $I_{PD_SOSC}$ | Secondary Oscillator ($S_{OSC}$) | — | 6 | 9.7 | 20 | µA | 3.0V | |
| D203 | $I_{PD_LPBOR}$ | Low-Power Brown-out Reset (LPBOR) | — | 0.8 | 4.3 | 7.3 | µA | 3.0V | |
| D204 | $I_{PD_FVR_BUF1}$ | FVR Buffer 1 (ADC) | — | 178 | 300 | 360 | µA | 3.0V | |
| D204A | $I_{PD_FVR_BUF2}$ | FVR Buffer 2 (DAC/CMP) | — | 43 | 73 | 92 | µA | 3.0V | |
| D205 | $I_{PD_BOR}$ | Brown-out Reset (BOR) | — | 31 | 49 | 68 | µA | 3.0V | |
| D207 | $I_{PD_ADCA}$ | ADC - Active | — | — | 4 | 12 | µA | 3.0V | ADC is not converting **(Note 4)** |

chanhvinh.com



.....continued

| Standard Operating Conditions (unless otherwise stated) | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| Param. No. | Sym. | Device Characteristics | Min. | Typ.† | Max. +85°C | Max. +125°C | Units | Conditions | |
| | | | | | | | | $V_{DD}$ | Note |
| D208 | $I_{PD_CMP}$ | Comparator1 | — | 30 | 49 | 68 | µA | 3.0V | |
| D208A | $I_{PD_CMP}$ | Comparator2 | — | 30 | 45 | 60 | µA | 3.0V | |

* 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.

**Notes:**

1. The peripheral current is the sum of the base $I_{DD}$ and the additional current consumed when this peripheral is enabled. The peripheral Δ current can be determined by subtracting the base $I_{DD}$ or $I_{PD}$ current from this limit. Max. values will 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 $V_{SS}$.
3. All peripheral currents listed are on a per-peripheral basis if more than one instance of a peripheral is available.
4. ADC clock source is ADCRC.

### 44.3.4 I/O Ports

**Table 44-4.**

| Standard Operating Conditions (unless otherwise stated) | | | | | | | |
|---|---|---|---|---|---|---|---|
| Param. No. | Sym. | Device Characteristics | Min. | Typ.† | Max. | Units | Conditions |
| **Input Low Voltage** | | | | | | | |
| | $V_{IL}$ | I/O PORT: | | | | | |
| D300 | | • with TTL buffer | — | — | 0.8 | V | $4.5V \le V_{DD} \le 5.5V$ |
| D301 | | | — | — | $0.15\ V_{DD}$ | V | $1.8V \le V_{DD} < 4.5V$ |
| D302 | | • with Schmitt Trigger buffer | — | — | $0.2\ V_{DD}$ | V | $2.0V \le V_{DD} \le 5.5V$ |
| D303 | | • with $I^2C$ levels | — | — | $0.3\ V_{DD}$ | V | $1.8V \le V_{DD} \le 5.5V$ |
| D304 | | • with SMBus 2.0 | — | — | 0.8 | V | $2.7V \le V_{DD} \le 5.5V$ |
| D305 | | • with SMBus 3.0 | — | — | 0.8 | V | |
| D306 | | $\overline{MCLR}$ | — | — | $0.2V_{DD}$ | V | |
| **High Low Voltage** | | | | | | | |

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



.....continued

| Standard Operating Conditions (unless otherwise stated) | | | | | | | |
|---|---|---|---|---|---|---|---|
| Param. No. | Sym. | Device Characteristics | Min. | Typ.† | Max. | Units | Conditions |
| | $V_{IH}$ | I/O PORT: | | | | | |
| D320 | | • with TTL buffer | 2.0 | — | — | V | $4.5V \le V_{DD} \le 5.5V$ |
| D321 | | | 0.25 $V_{DD}$ + 0.8 | — | — | V | $1.8V \le V_{DD} < 4.5V$ |
| D322 | | • with Schmitt Trigger buffer | 0.8 $V_{DD}$ | — | — | V | $2.0V \le V_{DD} \le 5.5V$ |
| D323 | | • with $I^2C$ levels | 0.7 $V_{DD}$ | — | — | V | |
| D324 | | • with SMBus 2.0 | 2.1 | — | — | V | $2.7V \le V_{DD} \le 5.5V$ |
| D325 | | • with SMBus 3.0 | 1.35 | — | — | V | |
| D326 | | $\overline{MCLR}$ | $0.7V_{DD}$ | — | — | V | |
| **Input Leakage Current(1)** | | | | | | | |
| D340 | $I_{IL}$ | I/O PORTS | — | ±5 | ±125 | nA | $V_{SS} \le V_{PIN} \le V_{DD}$, Pin at high-impedance, 85°C |
| D341 | | | — | ±5 | ±1000 | nA | $V_{SS} \le V_{PIN} \le V_{DD}$, Pin at high-impedance, 125°C |
| D342 | | $\overline{MCLR}$(2) | — | ±50 | ±200 | nA | $V_{SS} \le V_{PIN} \le V_{DD}$, Pin at high-impedance, 85°C |
| **Weak Pull-up Current** | | | | | | | |
| D350 | $I_{PUR}$ | | 80 | 140 | 200 | µA | $V_{DD}$ = 3.0V, $V_{PIN} = V_{SS}$ |
| **Output Low Voltage** | | | | | | | |
| D360 | $V_{OL}$ | I/O PORTS | — | — | 0.6 | V | $I_{OL}$ = 10.0 mA, $V_{PIN}$ = 3.0V |
| **Output High Voltage** | | | | | | | |
| D370 | $V_{OH}$ | I/O PORTS | $V_{DD}$ - 0.7 | — | — | V | $I_{OH}$ = 6.0 mA, $V_{PIN}$ = 3.0V |
| **All I/O Pins** | | | | | | | |
| D380 | $C_{IO}$ | | — | 5 | 50 | pF | |

† Data in "Typ" column is at 3.0V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

**Notes:**

1. Negative current is defined as current sourced by the pin.
2. The leakage current on the $\overline{MCLR}$ 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.

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



### 44.3.5 Memory Programming Specifications

**Table 44-5.**

| Standard Operating Conditions (unless otherwise stated) | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Param No.** | **Sym.** | **Device Characteristics** | **Min.** | **Typ†** | **Max.** | **Units** | **Conditions** |
| **Data EEPROM Memory Specifications** | | | | | | | |
| MEM20 | $E_D$ | DataEE Byte Endurance | 100k | — | — | E/W | -40°C ≤ $T_A$ ≤ +85°C |
| MEM21 | $T_{D_RET}$ | Characteristic Retention | — | 40 | — | Year | Provided no other specifications are violated |
| MEM23 | $V_{D_RW}$ | $V_{DD}$ for Read or Erase/Write operation | $V_{DDMIN}$ | — | $V_{DDMAX}$ | V | |
| MEM24 | $T_{D_BEW}$ | Byte Erase and Write Cycle Time | — | — | 11 | ms | |
| **Program Flash Memory Specifications** | | | | | | | |
| MEM30 | $E_P$ | Flash Memory Cell Endurance | 10k | — | — | E/W | -40°C ≤ $T_A$ ≤ +85°C **(Note 1)** |
| MEM32 | $T_{P_RET}$ | Characteristic Retention | — | 40 | — | Year | Provided no other specifications are violated |
| MEM33 | $V_{P_RD}$ | $V_{DD}$ for Read operation | $V_{DDMIN}$ | — | $V_{DDMAX}$ | V | |
| MEM34 | $V_{P_REW}$ | $V_{DD}$ for Row Erase or Write operation | $V_{DDMIN}$ | — | $V_{DDMAX}$ | V | |
| MEM35 | $T_{P_REW}$ | Self-Timed Page Write | — | — | — | ms | |
| MEM36 | $T_{SE}$ | Self-Timed Page Erase | — | — | — | ms | |
| MEM37 | $T_{P_WRD}$ | Self-Timed Word Write | — | — | — | µs | |

† 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. Flash Memory Cell Endurance for the Flash memory is defined as: One Row Erase operation and one Self-Timed Write.

### 44.3.6 Thermal Characteristics

**Table 44-6.**

| Standard Operating Conditions (unless otherwise stated) | | | | | |
|---|---|---|---|---|---|
| **Param No.** | **Sym.** | **Characteristic** | **Typ.** | **Units** | **Conditions** |
| TH01 | $\theta_{JA}$ | Thermal Resistance Junction to Ambient | — | °C/W | 28-pin SPDIP package |
| | | | — | °C/W | 28-pin SOIC package |
| | | | — | °C/W | 28-pin SSOP package |
| | | | — | °C/W | 28-pin VQFN 4x4 mm package |
| | | | — | °C/W | 40-pin PDIP package |
| | | | — | °C/W | 40-pin VQFN 5x5 mm package |
| | | | — | °C/W | 44-pin TQFP 10x10 mm package |

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



.....continued

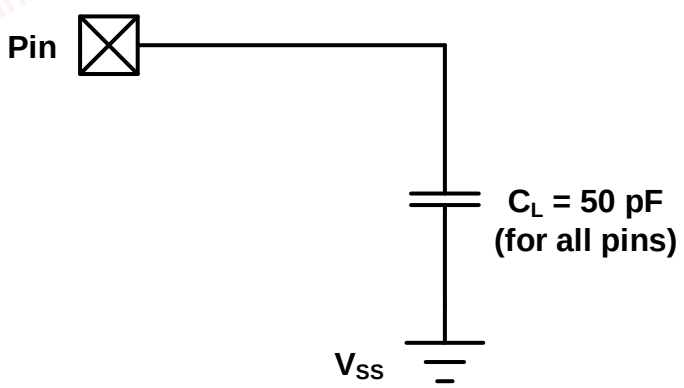
| Standard Operating Conditions (unless otherwise stated) | | | | | |
|---|---|---|---|---|---|
| Param No. | Sym. | Characteristic | Typ. | Units | Conditions |
| TH02 | $T_{JMAX}$ | Maximum Junction Temperature | 150 | °C | |
| TH03 | PD | Power Dissipation | — | W | $PD = P_{INTERNAL}+P_{I/O}$ |
| TH04 | $P_{INTERNAL}$ | Internal Power Dissipation | — | W | $P_{INTERNAL} = I_{DD} \times V_{DD}$ (1) |
| TH05 | $P_{I/O}$ | I/O Power Dissipation | — | W | $P_{I/O} = \Sigma(I_{OL}*V_{OL})+\Sigma(I_{OH}*(V_{DD}-V_{OH}))$ |
| TH06 | $P_{DER}$ | Derated Power | — | W | $P_{DER} = PD_{MAX} (T_J-T_A)/\theta_{JA}$ (2) |

**Notes:**

1. $I_{DD}$ is current to run the chip alone without driving any load on the output pins.
2. $T_A$ = Ambient Temperature, $T_J$ = Junction Temperature.

## 44.4 AC Characteristics

**Figure 44-3. Load Conditions**



chanhvinh.com



### 44.4.1 External Clock/Oscillator Timing Requirements

**Figure 44-4. Clock Timing**

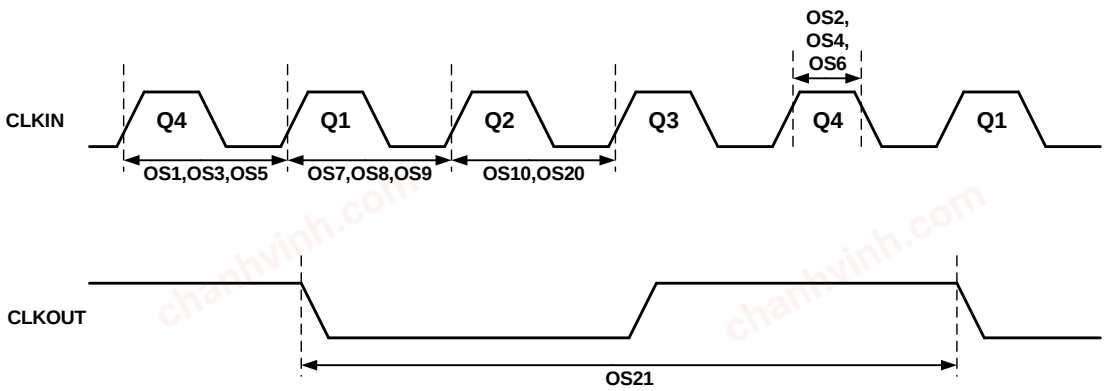



**Table 44-7.**

| Standard Operating Conditions (unless otherwise stated) | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Param No.** | **Sym.** | **Characteristic** | **Min.** | **Typ. †** | **Max.** | **Units** | **Conditions** |
| **ECL Oscillator** | | | | | | | |
| OS1 | $F_{ECL}$ | Clock Frequency | — | — | 1 | MHz | |
| OS2 | $T_{ECL_DC}$ | Clock Duty Cycle | 40 | — | 60 | % | |
| **ECH Oscillator** | | | | | | | |
| OS5 | $F_{ECH}$ | Clock Frequency | — | — | 32 | MHz | $V_{DD} \geq 2.7V$ |
| | | | — | — | 16 | MHz | $V_{DD} < 2.7V$ |
| OS6 | $T_{ECH_DC}$ | Clock Duty Cycle | 40 | — | 60 | % | |
| **LP Oscillator** | | | | | | | |
| OS7 | $F_{LP}$ | Clock Frequency | — | — | 100 | kHz | **(Note 4)** |
| **XT Oscillator** | | | | | | | |
| OS8 | $F_{XT}$ | Clock Frequency | — | — | 4 | MHz | **(Note 4)** |
| **HS Oscillator** | | | | | | | |
| OS9 | $F_{HS}$ | Clock Frequency | — | — | 20 | MHz | $V_{DD} > 2.5V$ **(Note 4)** |
| **Secondary Oscillator** | | | | | | | |
| OS10 | $F_{SEC}$ | Clock Frequency | — | 32.768 | — | kHz | **(Note 4)** |
| **System Oscillator** | | | | | | | |
| OS20 | $F_{OSC}$ | System Clock Frequency | — | — | 32 | MHz | **(Note 2, Note 3)** |
| OS21 | $F_{CY}$ | Instruction Frequency | — | $F_{OSC}/4$ | — | MHz | |
| OS22 | $T_{CY}$ | Instruction Period | 125 | $1/F_{CY}$ | — | ns | |

© 2023 Microchip Technology Inc.
and its subsidiaries

chanhvinh.com



.....continued

| Standard Operating Conditions (unless otherwise stated) | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Param No.** | **Sym.** | **Characteristic** | **Min.** | **Typ. †** | **Max.** | **Units** | **Conditions** |

**Notes:**

1. Instruction cycle period (TCY) equals four times the input oscillator time base period. All specified values are based on characterization data for that particular oscillator type under standard operating conditions with the device executing code. Exceeding these specified limits may result in an unstable oscillator operation and/or higher than expected current consumption. All devices are tested to operate at “min” values with an external clock applied to OSC1 pin. When an external clock input is used, the “max” cycle time limit is “DC” (no clock) for all devices.
2. The system clock frequency ($F_{OSC}$) is selected by the “main clock switch controls” as described in the **“Power Saving Operation Modes”** section.
3. The system clock frequency ($F_{OSC}$) must meet the voltage requirements defined in the **“Standard Operating Conditions”** section.
4. LP, XT and HS oscillator modes require an appropriate crystal or resonator to be connected to the device. For clocking the device with the external square wave, one of the EC mode selections must be used.

### 44.4.2 Internal Oscillator Parameters(1)

**Table 44-8.**

| Standard Operating Conditions (unless otherwise stated) | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Param No.** | **Sym.** | **Characteristic** | **Min.** | **Typ. †** | **Max.** | **Units** | **Conditions** |
| OS50 | $F_{HFOSC}$ | Precision Calibrated HFINTOSC Frequency | — | 4<br>8<br>16<br>32 | — | MHz | (**Note 2**) |
| OS51 | $F_{HFOSCLP}$ | Low-Power Optimized HFINTOSC Frequency | —<br>—<br>—<br>— | 1<br>2<br>1<br>2 | —<br>—<br>—<br>— | MHz<br>MHz<br>MHz<br>MHz | -40°C to 85°C<br>-40°C to 85°C<br>-40°C to 125°C<br>-40°C to 125°C |
| OS52 | $F_{MFOSC}$ | Internal Calibrated MFINTOSC Frequency | — | 500 | — | kHz | |
| OS53* | $F_{LFOSC}$ | Internal LFINTOSC Frequency | — | 31 | — | kHz | |
| OS54* | $T_{HFOSCST}$ | HFINTOSC Wake-up from Sleep Start-up Time | — | — | — | µs | |
| OS56 | $T_{LFOSCST}$ | LFINTOSC Wake-up from Sleep Start-up Time | — | — | — | ms | |

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



.....continued

| Standard Operating Conditions (unless otherwise stated) | | | | | | | |
|---|---|---|---|---|---|---|---|
| Param No. | Sym. | Characteristic | Min. | Typ. † | Max. | Units | Conditions |

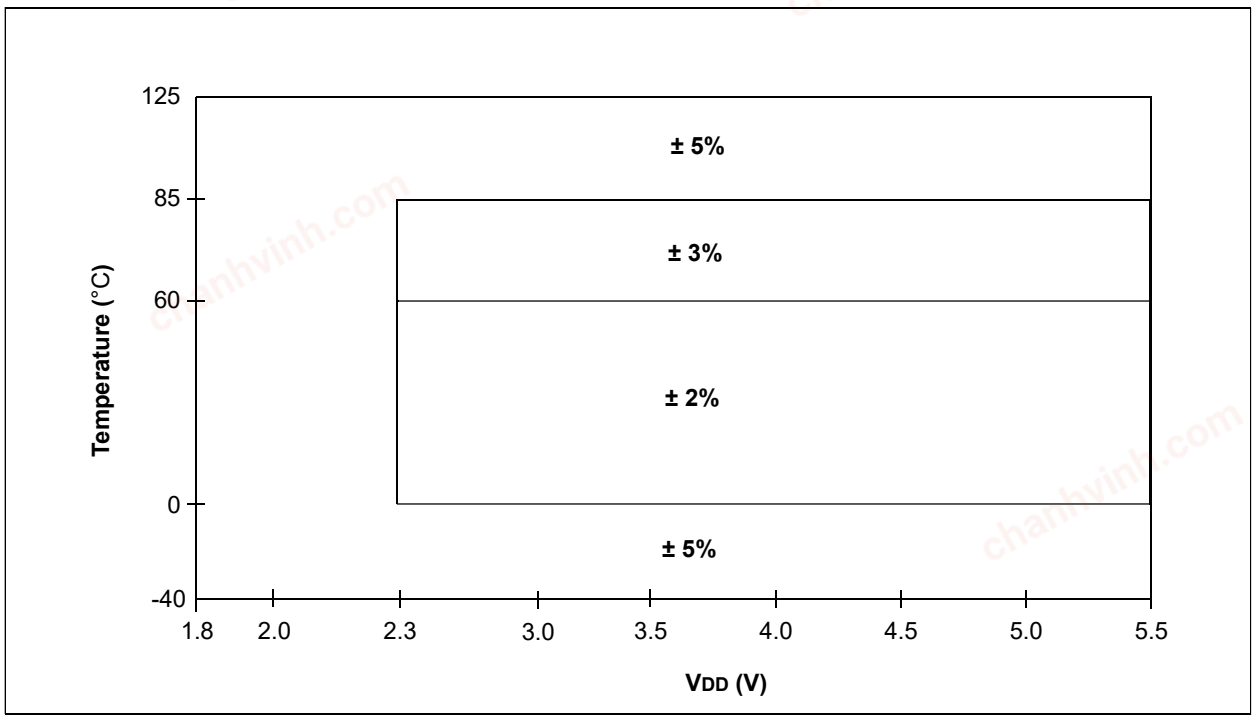
* These parameters are characterized but not tested.

† Data in “Typ” column is at 3.0V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

**Notes:**

1. To ensure these oscillator frequency tolerances, $V_{DD}$ and $V_{SS}$ must be capacitively decoupled as close to the device as possible. 0.1 µF and 0.01 µF values in parallel are recommended.
2. See the figure below.

**Figure 44-5. Precision Calibrated HFINTOSC Frequency Accuracy Over Device $V_{DD}$ and Temperature**



chanhvinh.com



### 44.4.3 I/O and CLKOUT Timing Specifications

**Figure 44-6. CLKOUT and I/O Timing**

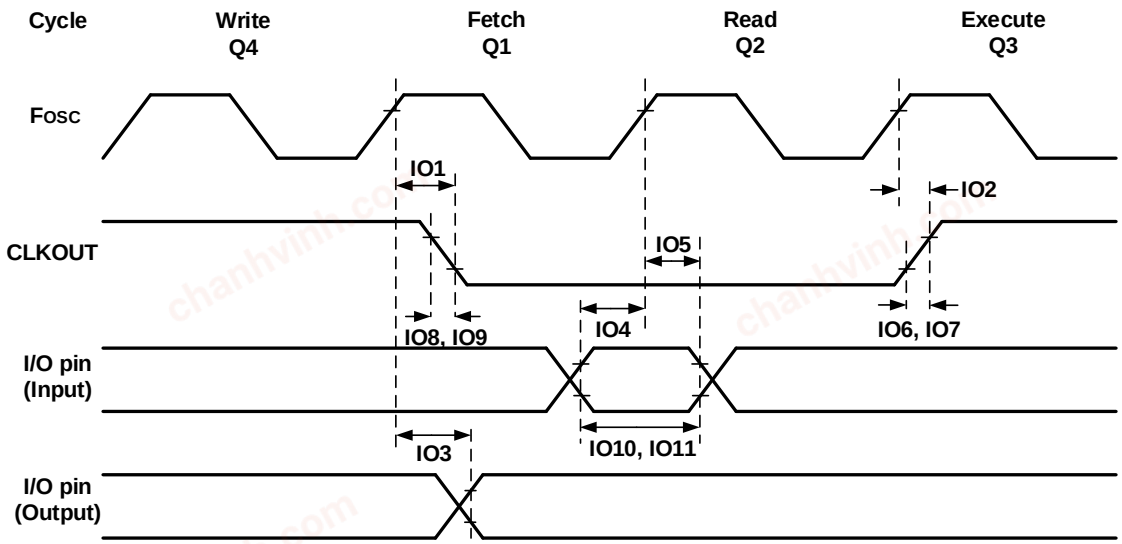



**Table 44-9. I/O and CLKOUT Timing Specifications**

| Standard Operating Conditions (unless otherwise stated) | | | | | | | |
|---|---|---|---|---|---|---|---|
| Param No. | Sym. | Characteristic | Min. | Typ. † | Max. | Units | Conditions |
| IO1* | $T_{CLKOUTH}$ | CLKOUT rising edge delay (rising edge $F_{OSC}$ (Q1 cycle) to falling edge CLKOUT | — | — | 70 | ns | |
| IO2* | $T_{CLKOUTL}$ | CLKOUT falling edge delay (rising edge $F_{OSC}$ (Q3 cycle) to rising edge CLKOUT | — | — | 72 | ns | |
| IO3* | $T_{IO_VALID}$ | Port output valid time (rising edge $F_{OSC}$ (Q1 cycle) to port valid) | — | 50 | 70 | ns | |
| IO4* | $T_{IO_SETUP}$ | Port input setup time (Setup time before rising edge $F_{OSC}$ – Q2 cycle) | 20 | — | — | ns | |
| IO5* | $T_{IO_HOLD}$ | Port input hold time (Hold time after rising edge $F_{OSC}$ – Q2 cycle) | 50 | — | — | ns | |
| IO6* | $T_{IOR_SLREN}$ | Port I/O rise time, slew rate enabled | — | 25 | — | ns | $V_{DD}$ = 3.0V |
| IO7* | $T_{IOR_SLRDIS}$ | Port I/O rise time, slew rate disabled | — | 5 | — | ns | $V_{DD}$ = 3.0V |
| IO8* | $T_{IOF_SLREN}$ | Port I/O fall time, slew rate enabled | — | 25 | — | ns | $V_{DD}$ = 3.0V |
| IO9* | $T_{IOF_SLRDIS}$ | Port I/O fall time, slew rate disabled | — | 5 | — | ns | $V_{DD}$ = 3.0V |
| IO10* | $T_{INT}$ | INT pin high or low time to trigger an interrupt | 25 | — | — | ns | |
| IO11* | $T_{IOC}$ | Interrupt-on-Change minimum high or low time to trigger interrupt | 25 | — | — | ns | |

* These parameters are characterized but not tested.

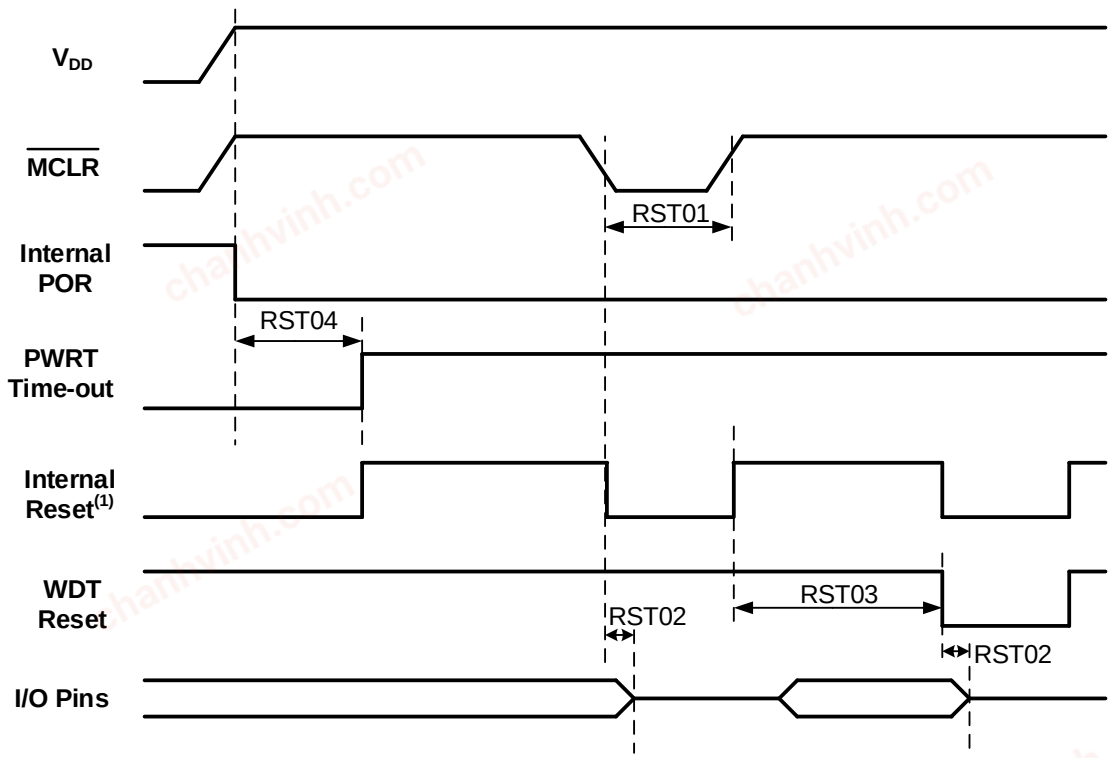
© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



### 44.4.4 Reset, WDT, Power-up Timer, and Brown-Out Reset Specifications

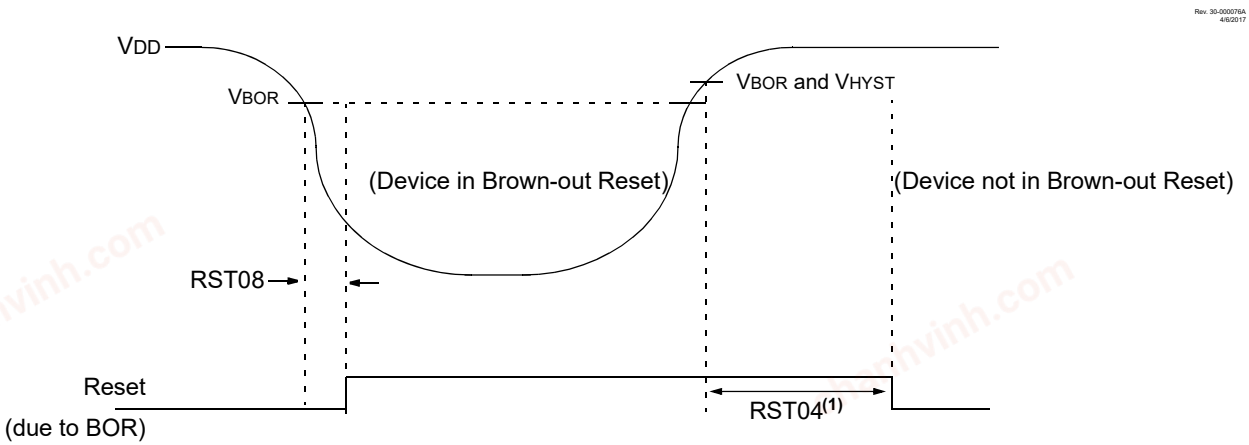
**Figure 44-7. Reset, Watchdog Timer, and Power-up Timer Timing**




**Note:**

1. Asserted low.

**Figure 44-8. Brown-out Reset Timing and Characteristics**




**Note:**

1. Only if $\overline{\text{PWRTE}}$ bit in the Configuration Word register is programmed to '1'; 2 ms delay if $\overline{\text{PWRTE}}$ = 0.

chanhvinh.com



**Table 44-10.**

| Standard Operating Conditions (unless otherwise stated) | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Param No.** | **Sym.** | **Characteristic** | **Min.** | **Typ. †** | **Max.** | **Units** | **Conditions** |
| RST01* | $T_{MCLR}$ | $\overline{MCLR}$ Pulse-Width Low to ensure Reset | — | — | — | µs | |
| RST02* | $T_{IOZ}$ | I/O high-impedance from Reset detection | — | — | 2 | µs | |
| RST03 | $T_{WDT}$ | Watchdog Timer Time-out Period | — | 16 | — | ms | WDTCPS = 00100 |
| RST04* | $T_{PWRT}$ | Power-up Timer Period | — | 65 | — | ms | |
| RST05 | $T_{OST}$ | Oscillator Start-up Timer Period(1,2) | — | 1024 | — | $T_{OSC}$ | |
| RST06 | $V_{BOR}$ | Brown-out Reset Voltage | — | 2.85 | — | V | BORV = 00 |
| | | | — | 2.7 | — | V | BORV = 01 |
| | | | — | 2.45 | — | V | BORV = 10 |
| | | | — | 1.9 | — | V | BORV = 11 |
| RST07 | $V_{BORHYS}$ | Brown-out Reset Hysteresis | — | 60 | — | mV | BORV = 00 |
| RST08 | $T_{BORDC}$ | Brown-out Reset Response Time | — | 3 | — | µs | |
| RST09 | $V_{LPBOR}$ | Low-Power Brown-out Reset Voltage | 1.8 | 1.9 | 2.2 | V | |

* 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.

**Notes:**

1. By design, the Oscillator Start-up Timer (OST) counts the first 1024 cycles, independent of frequency.
2. To ensure these voltage tolerances, $V_{DD}$ and $V_{SS}$ must be capacitively decoupled as close to the device as possible. 0.1 µF and 0.01 µF values in parallel are recommended.

### 44.4.5 Analog-to-Digital Converter (ADC) Accuracy Specifications(1,2)

**Table 44-11.**

| Standard Operating Conditions (unless otherwise stated)<br>$V_{DD}$ = 3.0V, $T_A$ = 25°C, $T_{AD}$ = 1 µs, $F_{OSC} \leq$ 8 MHz, $ADC_{VREF} = V_{DD}/V_{SS}$, Differential Mode | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Param No.** | **Sym.** | **Characteristic** | **Min.** | **Typ. †** | **Max.** | **Units** | **Conditions** |
| AD01 | $N_R$ | Resolution | — | — | 12 | bit | |
| AD02 | $E_{IL}$ | Integral Nonlinearity Error | -6 | ±0.5 | +6 | LSb | $ADC_{REF}$+ = 3.0V, $ADC_{REF}$- = 0V |
| AD03 | $E_{DL}$ | Differential Nonlinearity Error | -1 | ±0.5 | +2.5 | LSb | $ADC_{REF}$+ = 3.0V, $ADC_{REF}$- = 0V |
| AD04 | $E_{OFF}$ | Offset Error | -11.5 | ±3 | +11.5 | LSb | $ADC_{REF}$+ = 3.0V, $ADC_{REF}$- = 0V |
| AD05 | $E_{GN}$ | Gain Error | -4 | ±1 | +4 | LSb | $ADC_{REF}$+ = 3.0V, $ADC_{REF}$- = 0V |

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



.....continued

| Standard Operating Conditions (unless otherwise stated)<br>$V_{DD}$ = 3.0V, $T_A$ = 25°C, $T_{AD}$ = 1 µs, $F_{OSC}$ ≤ 8 MHz, $ADC_{VREF}$ = $V_{DD}/V_{SS}$, Differential Mode | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Param No.** | **Sym.** | **Characteristic** | **Min.** | **Typ. †** | **Max.** | **Units** | **Conditions** |
| AD06 | $V_{ADREF}$ | ADC Reference Voltage ($AD_{REF}$+ - $AD_{REF}$-) | 1.8 | — | $V_{DD}$ | V | |
| AD07 | $V_{AIN}$ | Full-Scale Range | $V_{SS}$ | — | $AD_{REF}$+ | V | |
| AD08 | $Z_{AIN}$ | Recommended Impedance of Analog Voltage Source | — | 1 | — | kΩ | |
| AD09 | $R_{VREF}$ | ADC Voltage Reference Ladder Impedance | — | 50 | — | kΩ | **(Note 3)** |

* 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.

**Notes:**

1. Total Absolute Error is the sum of the offset, gain and integral nonlinearity (INL) errors.
2. This is the impedance seen by the $V_{REF}$ pads when the external reference pads are selected.

### 44.4.6 Analog-to-Digital Converter (ADC) Conversion Timing Specifications

**Table 44-12.**

| Standard Operating Conditions (unless otherwise stated) | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Param No.** | **Sym.** | **Characteristic** | **Min.** | **Typ. †** | **Max.** | **Units** | **Conditions** |
| AD20 | $T_{AD}$ | ADC Clock Period | 0.5 | — | 9 | µs | Using $F_{OSC}$ as the ADC clock source ADOCS = 0 |
| | | | — | 2 | — | µs | Using ADCRC as the ADC clock source ADOCS = 1 |
| AD21 | $T_{CNV}$ | Conversion Time | — | 14 $T_{AD}$+2$T_{CY}$ | — | — | Using $F_{OSC}$ as the ADC clock source ADOCS = 0 |
| | | | — | 16 $T_{AD}$+2$T_{CY}$ | — | — | Using ADCRC as the ADC clock source ADOCS = 1 |
| AD22 | $T_{HCD}$ | Sample-and-Hold Capacitor Disconnect Time | — | 2 $T_{AD}$+1$T_{CY}$ | — | — | Using $F_{OSC}$ as the ADC clock source ADOCS = 0 |
| | | | — | 3 $T_{AD}$+2$T_{CY}$ | — | — | Using ADCRC as the ADC clock source ADOCS = 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.

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



**Figure 44-9. ADC Conversion Timing (ADC Clock $F_{OSC}$-Based)**

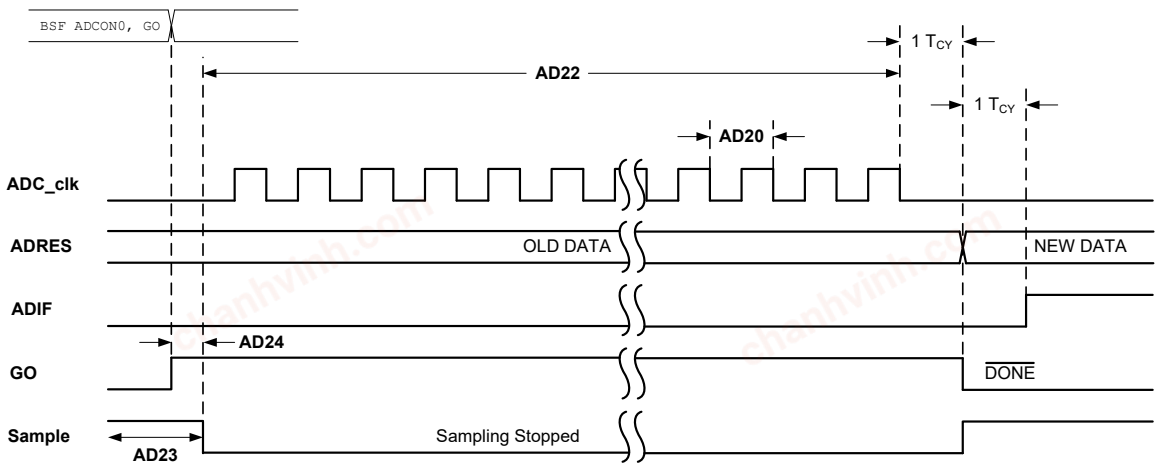
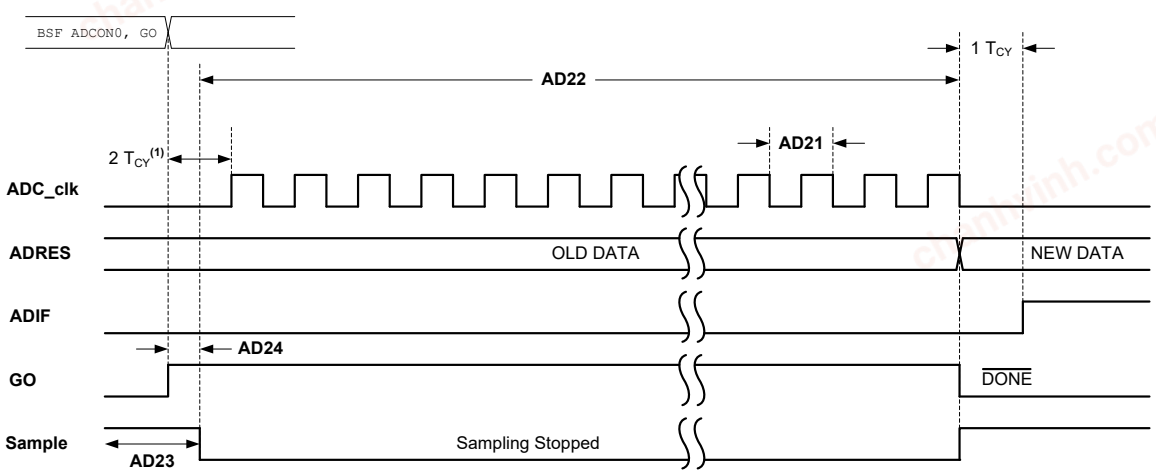



**Figure 44-10. ADC Conversion Timing (ADC Clock from ADCRC)**




**Note:**

1. If the ADC clock source is selected as ADCRC, a time of $T_{CY}$ is added before the ADC clock starts. This allows the `SLEEP` instruction to be executed.

### 44.4.7 Operational Amplifier Specifications

**Table 44-13.**

| Standard Operating Conditions (unless otherwise stated)<br>$V_{DD}$ = 3.0V, $T_A$ = 25°C | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Param No.** | **Sym.** | **Characteristic** | **Min.** | **Typ. †** | **Max.** | **Units** | **Conditions** |
| OPA01* | GBWP | Gain Bandwidth Product | — | — | 2.3 | MHz | |

chanhvinh.com



.....continued

| Standard Operating Conditions (unless otherwise stated)<br>$V_{DD}$ = 3.0V, $T_A$ = 25°C | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Param No.** | **Sym.** | **Characteristic** | **Min.** | **Typ. †** | **Max.** | **Units** | **Conditions** |
| OPA02* | $T_{ON}$ | Turn-on Time | — | 0.9 | — | µs | |
| OPA03* | $P_M$ | Phase Margin | — | 68 | — | degrees | |
| OPA04* | $S_R$ | Slew Rate | — | 9 | — | V/µs | |
| OPA05* | $O_{FF}$ | Offset | — | ±1 | — | mV | $V_{DD}$/2 |
| OPA06* | CMRR | Common-Mode Rejection Ratio | 45 | 73 | — | dB | Rail to Rail |
| OPA07* | $A_{OL}$ | Open-Loop Gain | — | 93 | — | dB | Open-Loop Gain (Phase) |
| OPA08* | $V_{ICM}$ | Input Common-Mode Voltage | $V_{SS}$ | — | $V_{DD}$ | V | Charge Pump On; $V_{DD}$ > 2.5V |
| | | | — | — | — | | Charge Pump Off; $V_{DD}$ > 2.5V |
| OPA09* | $E_{ni}$ | Input Noise Voltage | — | 160 | — | µV | |
| OPA10* | $e_{ni}$ | Input Noise Voltage Density | — | 28 | — | nV/√(Hz) | 10 kHz |
| OPA11* | PSRR | Power Supply Rejection Ratio | — | 76 | — | dB | |
| OPA12* | ISC | Short Circuit Current | — | 10 | — | mA | |
| OPA13* | $V_{OS}$ | Voltage Output Swing | (0.15) * $V_{DD}$ | — | (0.85) * $V_{DD}$ | V | |
| OPA14* | $I_O$ | Maximum Output Current | — | 0.5 | — | nA | |

* 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.

### 44.4.8 8-Bit DAC Specifications

**Table 44-14.**

| Standard Operating Conditions (unless otherwise stated)<br>$V_{DD}$ = 3.0V, $T_A$ = 25°C | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Param No.** | **Sym.** | **Characteristic** | **Min.** | **Typ. †** | **Max.** | **Units** | **Conditions** |
| DAC01* | $C_{LSB}$ | Step Size | — | ($V_{DACREF^+}$-$V_{DACREF^-}$)/256 | — | V | |
| DAC02 | $C_{INL}$ | Integral Error | — | — | ±1.4 | LSb | |
| DAC03 | $C_{DNL}$ | Differential Error | — | — | ±1 | LSb | |
| DAC04 | $C_{OFF}$ | Offset Error | — | — | ±1 | LSb | |
| DSB05 | $C_{GN}$ | Gain Error | — | — | ±1.5 | LSb | |
| DSB06* | $C_R$ | Unit Resistor Value | — | — | — | kΩ | |

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



.....continued

| Standard Operating Conditions (unless otherwise stated)<br>$V_{DD}$ = 3.0V, $T_A$ = 25°C | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Param No.** | **Sym.** | **Characteristic** | **Min.** | **Typ. †** | **Max.** | **Units** | **Conditions** |
| DSB07* | $C_{ST}$ | Settling Time(1) | — | — | 10 | µs | |
| DSB08* | $C_{ACC}$ | Absolute Accuracy | — | — | ±4 | LSb | |

* 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. Settling time measured while DACR[7:0] transitions from 'b00000000 to 'b11111111.

### 44.4.9 Comparator Specifications

**Table 44-15.**

| Standard Operating Conditions (unless otherwise stated)<br>$V_{DD}$ = 3.0V, $T_A$ = 25°C | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Param No.** | **Sym.** | **Characteristic** | **Min.** | **Typ. †** | **Max.** | **Units** | **Conditions** |
| CM01 | $V_{IOFF}$ | Input Offset Voltage | — | — | ±50 | mV | $V_{ICM} = V_{DD}/2$ |
| CM02 | $V_{ICM}$ | Input Common-Mode Range | GND | — | $V_{DD}$ | V | |
| CM03 | CMRR | Common-Mode Input Rejection Ratio | — | 50 | — | dB | |
| CM04 | $V_{HYST}$ | Comparator Hysteresis | 10 | 25 | 45 | mV | |
| CM05 | $T_{RESP}$(1) | Response Time, Rising Edge | — | 300 | 600 | ns | |
| | | Response Time, Falling Edge | — | 220 | 500 | ns | |

* These parameters are characterized but not tested.

† Data in "Typ" column is at 3.0V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

**Note:**

1. Response time measured with one comparator input at $V_{DD}/2$, while the other input transitions from $V_{SS}$ to $V_{DD}$.

### 44.4.10 Zero-Cross Detect (ZCD) Specifications

**Table 44-16.**

| Standard Operating Conditions (unless otherwise stated)<br>$V_{DD}$ = 3.0V, $T_A$ = 25°C | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Param No.** | **Sym.** | **Characteristic** | **Min.** | **Typ. †** | **Max.** | **Units** | **Conditions** |
| ZC01 | $V_{PINZC}$ | Voltage on Zero-Cross Pin | — | 0.7 | — | V | |

chanhvinh.com



.....continued

| Standard Operating Conditions (unless otherwise stated)<br>$V_{DD}$ = 3.0V, $T_A$ = 25°C | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Param No.** | **Sym.** | **Characteristic** | **Min.** | **Typ. †** | **Max.** | **Units** | **Conditions** |
| ZC02 | $I_{ZCD_MAX}$ | Maximum source or sink current | — | — | 600 | µA | |
| ZC03 | $T_{RESPH}$ | Response Time, Rising Edge | — | 1 | — | µs | |
| | $T_{RESPL}$ | Response Time, Falling Edge | — | 1 | — | µs | |

† Data in “Typ” column is at 3.0V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

### 44.4.11 Fixed Voltage Reference (FVR) Specifications

**Table 44-17.**

| Standard Operating Conditions (unless otherwise stated) | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Param No.** | **Sym.** | **Characteristic** | **Min.** | **Typ. †** | **Max.** | **Units** | **Conditions** |
| FVR01 | $V_{FVR}1$ | 1x Gain (1.024V) | -4 | — | -4 | % | $V_{DD} \geq$ 2.5V, -40°C to 85°C |
| FVR02 | $V_{FVR}2$ | 2x Gain (2.048V) | -4 | — | +4 | % | $V_{DD} \geq$ 2.5V, -40°C to 85°C |
| FVR03 | $V_{FVR}4$ | 4x Gain (4.096V) | -5 | — | +5 | % | $V_{DD} \geq$ 4.75V, -40°C to 85°C |
| FVR04 | $T_{FVRST}$ | FVR Start-up Time | — | 25 | — | µs | |

### 44.4.12 Temperature Indicator Requirements

**Table 44-18.**

| Standard Operating Conditions (unless otherwise stated) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **Param No.** | **Sym.** | **Characteristic** | | **Min.** | **Typ. †** | **Max.** | **Units** | **Conditions** |
| TS01* | $T_{ACQMIN}$ | Minimum ADC Acquisition Time Delay | | — | 25 | — | µs | |
| TS02* | $M_V$ | Voltage Sensitivity | High Range | — | -3.75 | — | mV/°C | TSRNG = 1 |
| | | | Low Range | — | -2.75 | — | mV/°C | TSRNG = 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.

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



### 44.4.13 Timer0 and Timer1 External Clock Requirements

**Table 44-19.**

| Standard Operating Conditions (unless otherwise stated)<br>Operating Temperature: -40°C≤$T_A$≤+125°C | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **Param No.** | **Sym.** | **Characteristic** | | **Min.** | **Typ. †** | **Max.** | **Units** | **Conditions** |
| 40* | $T_T0H$ | T0CKI High Pulse-Width | No Prescaler | $0.5T_{CY}$+20 | — | — | ns | |
| | | | With Prescaler | 10 | — | — | ns | |
| 41* | $T_T0L$ | T0CKI Low Pulse-Width | No Prescaler | $0.5T_{CY}$+20 | — | — | ns | |
| | | | With Prescaler | 10 | — | — | ns | |
| 42* | $T_T0P$ | T0CKI Period | | Greater of: 20 or ($T_{CY}$+40)/N | — | — | ns | N = Prescale value |
| 45* | $T_T1H$ | T1CKI High Time | Synchronous, No Prescaler | $0.5T_{CY}$+20 | — | — | ns | |
| | | | Synchronous, with Prescaler | 15 | — | — | ns | |
| | | | Asynchronous | 30 | — | — | ns | |
| 46* | $T_T1L$ | T1CKI Low Time | Synchronous, No Prescaler | $0.5T_{CY}$+20 | — | — | ns | |
| | | | Synchronous, with Prescaler | 15 | — | — | ns | |
| | | | Asynchronous | 30 | — | — | ns | |
| 47* | $T_T1P$ | T1CKI Input Period | Synchronous | Greater of: 30 or ($T_{CY}$+40)/N | — | — | ns | N = Prescale value |
| | | | Asynchronous | 60 | — | — | ns | |
| 49* | $TCKEZ_{TMR}1$ | 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.

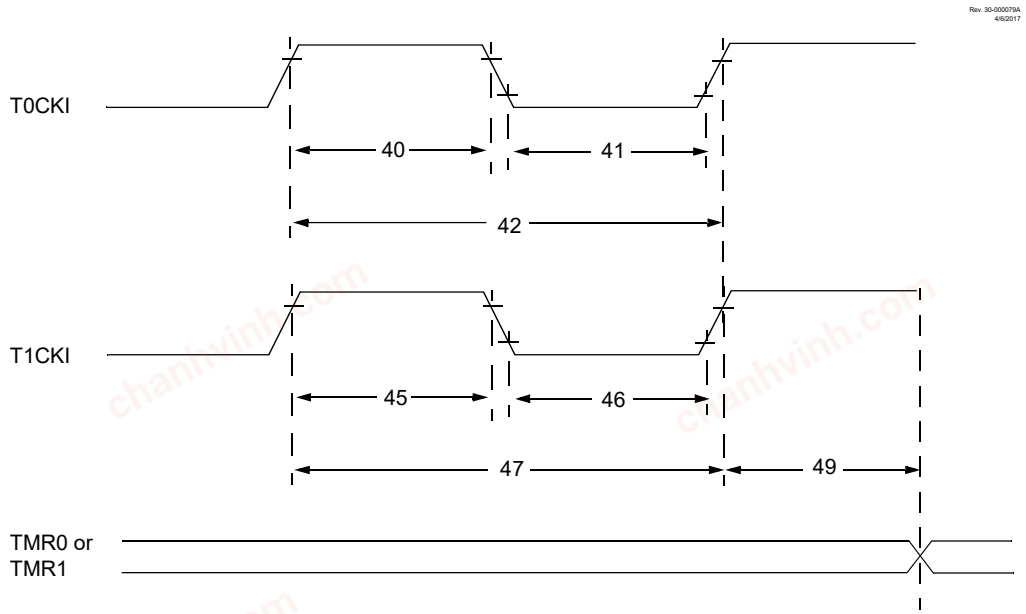
† Data in “Typ” column is at 3.0V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

chanhvinh.com

chanhvinh.com



**Figure 44-11. Timer0 and Timing1 External Clock Timings**




### 44.4.14 Capture/Compare/PWM Requirements (CCP)

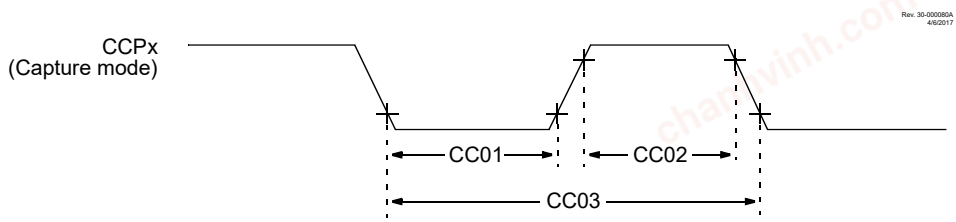
**Table 44-20.**

| Standard Operating Conditions (unless otherwise stated)<br>Operating Temperature: -40°C ≤ $T_A$ ≤ +125°C | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **Param No.** | **Sym.** | **Characteristic** | | **Min.** | **Typ. †** | **Max.** | **Units** | **Conditions** |
| CC01* | $T_{CC}L$ | CCPx Input Low Time | No Prescaler | $0.5T_{CY}$+20 | — | — | ns | |
| | | | With Prescaler | 20 | — | — | ns | |
| CC02* | $T_{CC}H$ | CCPx Input High Time | No Prescaler | $0.5T_{CY}$+20 | — | — | ns | |
| | | | With Prescaler | 20 | — | — | ns | |
| CC03* | $T_{CC}P$ | CCPx Input Period | | $(3T_{CY}+40)/N$ | — | — | ns | N = Prescale value |

* These parameters are characterized but not tested.

† Data in "Typ" column is at 3.0V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

**Figure 44-12. Capture/Compare/PWM Timings (CCP)**




**Note:** Refer to Figure 44-3 for load conditions.

chanhvinh.com

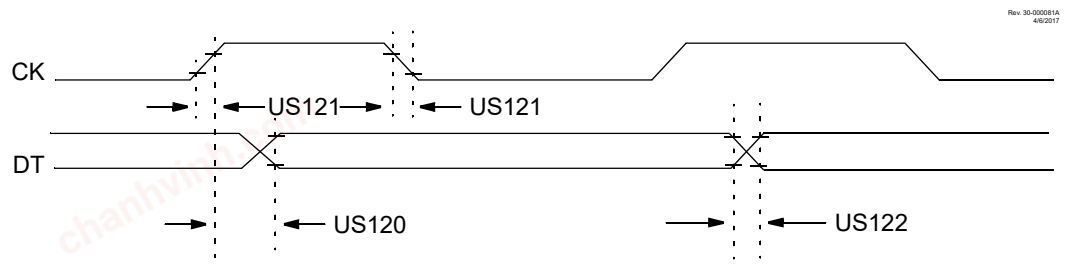


### 44.4.15 EUSART Synchronous Transmission Requirements

**Table 44-21.**

| Standard Operating Conditions (unless otherwise stated) | | | | | | |
|---|---|---|---|---|---|---|
| Param No. | Sym. | Characteristic | Min. | Max. | Units | Conditions |
| US120 | $T_{CK}H2_{DT}V$ | SYNC XMIT (Host and Client)<br>Clock high to data-out valid | — | — | ns | $3.0V \le V_{DD} \le 5.5V$ |
| | | | — | — | ns | $1.8V \le V_{DD} \le 5.5V$ |
| US121 | $T_{CKRF}$ | Clock out rise time and fall time<br>(Host mode) | — | — | ns | $3.0V \le V_{DD} \le 5.5V$ |
| | | | — | — | ns | $1.8V \le V_{DD} \le 5.5V$ |
| US122 | $T_{DTRF}$ | Data-out rise time and fall time | — | — | ns | $3.0V \le V_{DD} \le 5.5V$ |
| | | | — | — | ns | $1.8V \le V_{DD} \le 5.5V$ |

**Figure 44-13. EUSART Synchronous Transmission (Host/Client) Timing**



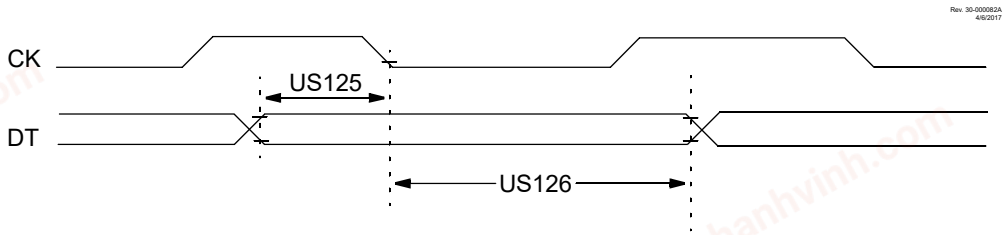

**Note:** Refer to Figure 44-3 for load conditions.

### 44.4.16 EUSART Synchronous Receive Requirements

**Table 44-22.**

| Standard Operating Conditions (unless otherwise stated) | | | | | | |
|---|---|---|---|---|---|---|
| Param No. | Sym. | Characteristic | Min. | Max. | Units | Conditions |
| US125 | $T_{DT}V2_{CKL}$ | SYNC RCV (Host and Client)<br>Data-setup before CK ↓ (DT hold time) | — | — | ns | |
| US126 | $T_{CK}L2_{DTL}$ | Data-hold after CK ↓ (DT hold time) | — | — | ns | |

**Figure 44-14. EUSART Synchronous Receive (Host/Client) Timing**




**Note:** Refer to Figure 44-3 for load conditions.

chanhvinh.com



### 44.4.17 SPI Mode Requirements

**Table 44-23. SPI Mode**

| Standard Operating Conditions (unless otherwise stated) | | | | | | | |
|---|---|---|---|---|---|---|---|
| Param No. | Sym. | Characteristic | Min. | Typ. † | Max. | Units | Conditions |
| SP70* | $T_{SS}L2_{SC}H$, $T_{SS}L2_{SC}L$ | SDO to SCK↓ or SCK↑ input | 2.25*$T_{CY}$ | — | — | ns | |
| SP71* | $T_{SC}H$ | SCK output high time | $T_{CY}$ + 20 | — | 0.5 $T_{SCK}$ + 12 | ns | |
| SP72* | $T_{SC}L$ | SCK output low time | $T_{CY}$ + 20 | — | 0.5 $T_{SCK}$ + 12 | ns | |
| SP73* | $T_{DI}V2_{SC}H$, $T_{DI}V2_{SC}L$ | Setup time of SDI data input to SCK edge | 85 | — | — | ns | |
| SP74* | $T_{SC}H2_{DI}L$, $T_{SC}L2_{DI}L$ | Hold time of SDI data input to SCK edge | 0 | — | — | ns | |
| | | Hold time of SDI data input to final SCK | 0.5 $T_{SCK}$ | | | ns | CKE = 0, SMP = 1 |
| SP75* | $T_{DO}R$ | SDO data output rise time | — | 10 | 25 | ns | $C_L$ = 50 pF |
| SP76* | $T_{DO}F$ | SDO data output fall time | — | 10 | 25 | ns | $C_L$ = 50 pF |
| SP78* | $T_{SC}R$ | SCK output rise time | — | 10 | 25 | ns | $C_L$ = 50 pF |
| SP79* | $T_{SC}F$ | SCK output fall time | — | 10 | 25 | ns | $C_L$ = 50 pF |
| SP80* | $T_{SC}H2_{DO}V$, $T_{SC}L2_{DO}V$ | SDO data output valid after SCK edge | — | — | — | ns | $C_L$ = 50 pF |
| SP81* | $T_{DO}V2_{SC}H$, $T_{DO}V2_{SC}L$ | SDO data output valid to first SCK edge | 1 $T_{CY}$ | — | — | ns | $C_L$ = 50 pF<br>CKE = 1 |
| SP82* | $T_{SS}L2_{DO}V$ | SDO data output valid after $\overline{SS}$↓ edge | — | — | 50 | ns | $C_L$ = 20 pF |
| SP83* | $T_{SC}H2_{SS}H$, $T_{SC}L2_{SS}H$ | $\overline{SS}$ ↑ after last SCK edge | 1.5 $T_{CY}$ + 40 | — | — | ns | |

* These parameters are characterized but not tested.

† Data in "Typ" column is at 3.0V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

**Note:**

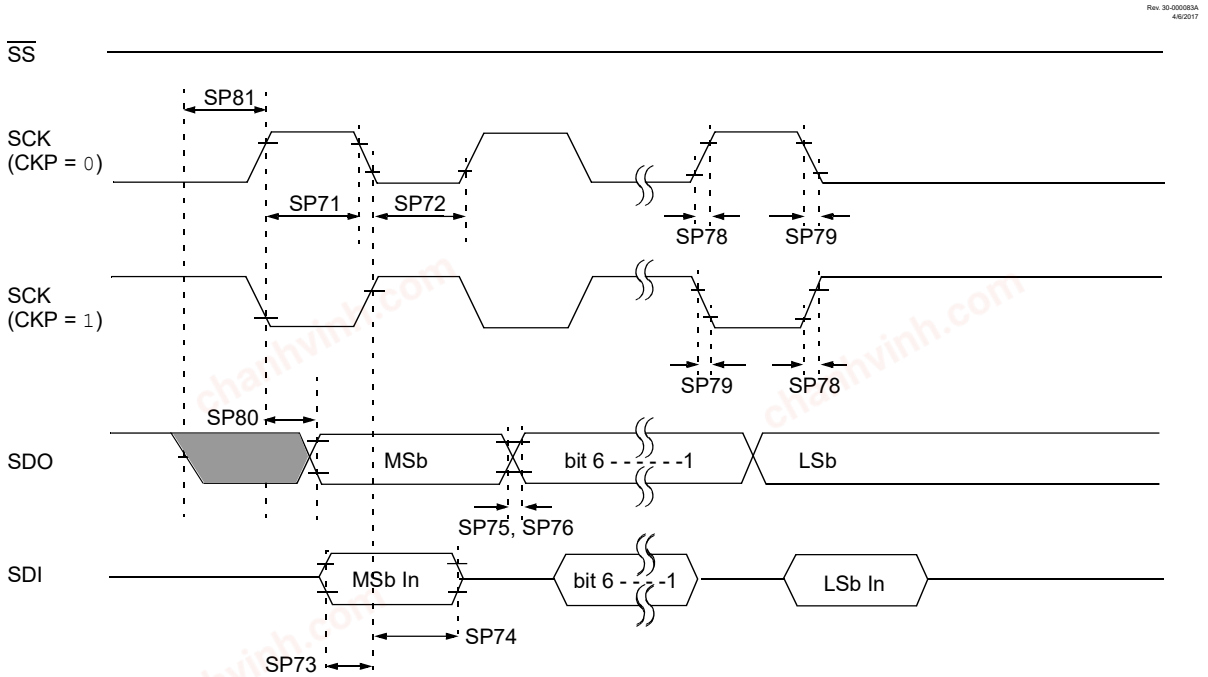
1. The SMP bit in the SSPxSTAT register must be set and the slew rate control must be disabled on the clock and data pins (clear the corresponding bits in SLRCONx register) for SPI to operate over 4 MHz.

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com

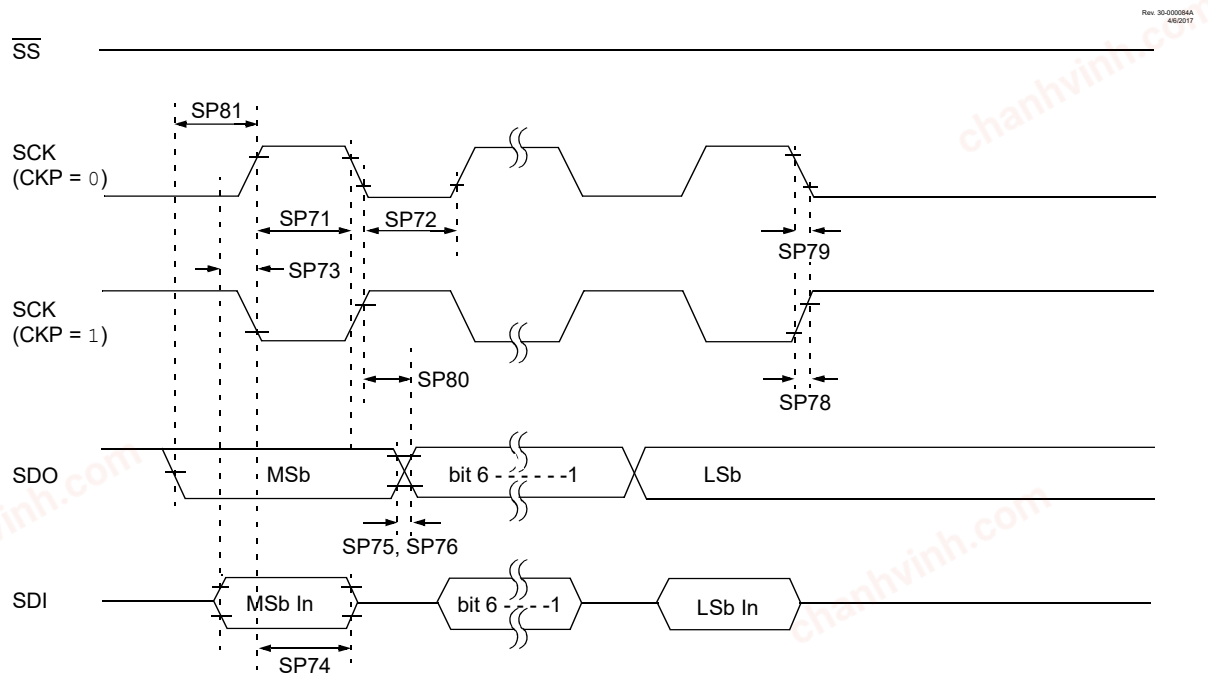


**Figure 44-15. SPI Host Mode Timing (CKE = 0, SMP = 0)**




**Note:** Refer to Figure 44-3 for load conditions.

**Figure 44-16. SPI Host Mode Timing (CKE = 1, SMP = 1)**

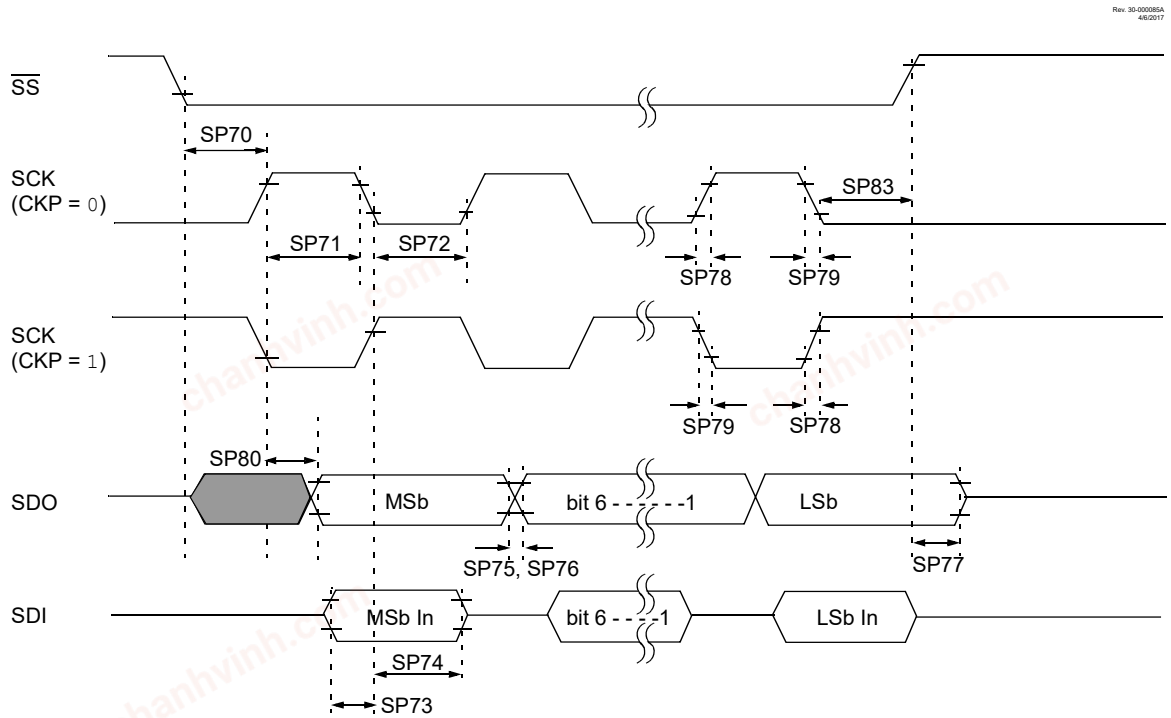



**Note:** Refer to Figure 44-3 for load conditions.

chanhvinh.com

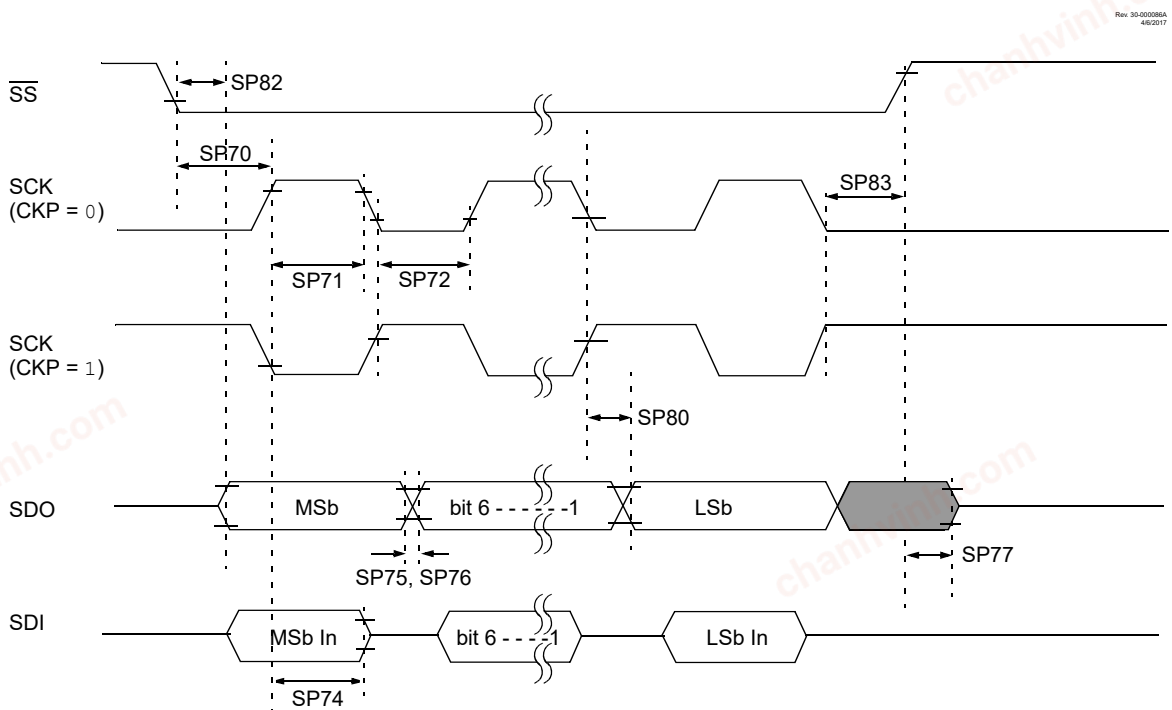


**Figure 44-17. SPI Client Mode Timing (CKE = 0)**




**Note:** Refer to Figure 44-3 for load conditions.

**Figure 44-18. SPI Client Mode Timing (CKE = 1)**




**Note:** Refer to Figure 44-3 for load conditions.

chanhvinh.com

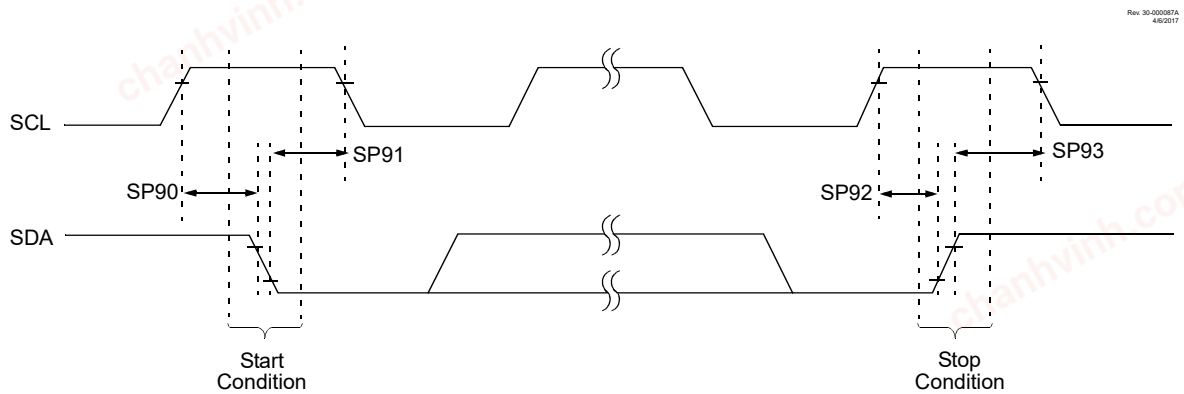


### 44.4.18 I²C Bus Start/Stop Bits Requirements

**Table 44-24.**

| Standard Operating Conditions (unless otherwise stated) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **Param. No.** | **Sym.** | **Characteristic** | | **Min.** | **Typ. †** | **Max.** | **Units** | **Conditions** |
| SP90* | $T_{SU:STA}$ | Start condition Setup time | 100 kHz mode | 4700 | — | — | ns | Only relevant for Repeated Start condition |
| | | | 400 kHz mode | 600 | — | — | | |
| SP91* | $T_{HD:STA}$ | Start condition Hold time | 100 kHz mode | 4000 | — | — | ns | After this period, the first clock pulse is generated |
| | | | 400 kHz mode | 600 | — | — | | |
| SP92* | $T_{SU:STO}$ | Stop condition Setup time | 100 kHz mode | 4000 | — | — | ns | |
| | | | 400 kHz mode | 600 | — | — | | |
| SP93* | $T_{HD:STO}$ | Stop condition Hold time | 100 kHz mode | 4700 | — | — | ns | |
| | | | 400 kHz mode | 1300 | — | — | | |
| * These parameters are characterized but not tested. | | | | | | | | |

**Figure 44-19. I²C Bus Start/Stop Bits Timing**




**Note:** Refer to Figure 44-3 for load conditions.

### 44.4.19 I²C Bus Data Requirements

**Table 44-25.**

| Standard Operating Conditions (unless otherwise stated) | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Param. No.** | **Sym.** | **Characteristic** | | **Min.** | **Max.** | **Units** | **Conditions** |
| SP100* | $T_{HIGH}$ | Clock high time | 100 kHz mode | 4000 | — | ns | Device must operate at a minimum of 1.5 MHz |
| | | | 400 kHz mode | 600 | — | ns | Device must operate at a minimum of 10 MHz |

chanhvinh.com



.....continued

| Standard Operating Conditions (unless otherwise stated) | | | | | | | |
|---|---|---|---|---|---|---|---|
| Param. No. | Sym. | Characteristic | | Min. | Max. | Units | Conditions |
| SP101* | $T_{LOW}$ | Clock low time | 100 kHz mode | 4700 | — | ns | Device must operate at a minimum of 1.5 MHz |
| | | | 400 kHz mode | 1300 | — | ns | Device must operate at a minimum of 10 MHz |
| SP102* | $T_R$ | SDA and SCL rise time | 100 kHz mode | — | 1000 | ns | |
| | | | 400 kHz mode | 20 | 300 | ns | $C_B$ is specified to be from 10-400 pF |
| SP103* | $T_F$ | SDA and SCL fall time | 100 kHz mode | — | 250 | ns | |
| | | | 400 kHz mode | 20 × ($V_{DD}$/ 5.5V) | 250 | ns | $C_B$ is specified to be from 10-400 pF |
| SP106* | $T_{HD:DAT}$ | Data input hold time | 100 kHz mode | 0 | — | ns | |
| | | | 400 kHz mode | 0 | — | ns | |
| SP107* | $T_{SU:DAT}$ | Data input setup time | 100 kHz mode | 250 | — | ns | **(Note 2)** |
| | | | 400 kHz mode | 100 | — | ns | |
| SP109* | $T_{AA}$ | Output valid from clock | 100 kHz mode | — | 3450 | ns | **(Note 1)** |
| | | | 400 kHz mode | — | 900 | ns | |
| SP110* | $T_{BUF}$ | Bus free time | 100 kHz mode | 4700 | — | ns | Time the bus must be free before a new transmission can start |
| | | | 400 kHz mode | 1300 | — | ns | |
| SP111 | $C_B$ | Bus capacitive loading | 100 kHz mode | — | 400 | pF | |
| | | | 400 kHz mode | — | 400 | pF | |

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com

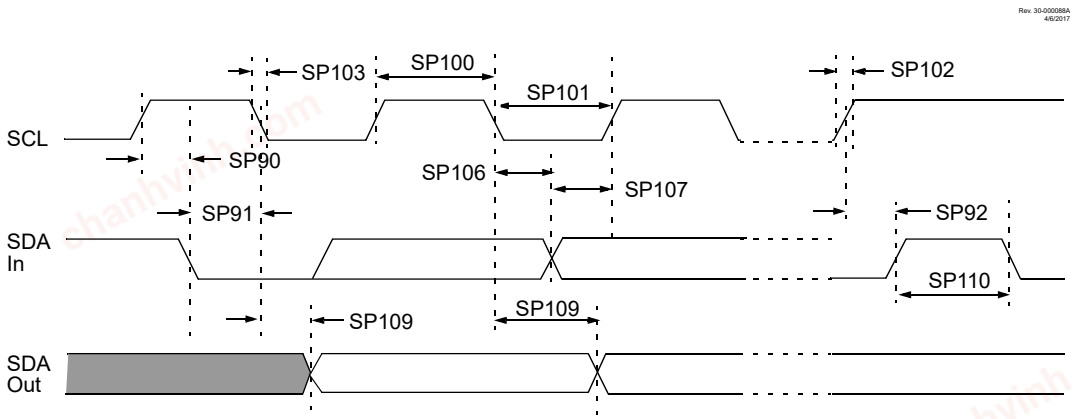


| .....continued | | | | | | |
|---|---|---|---|---|---|---|
| **Standard Operating Conditions (unless otherwise stated)** | | | | | | |
| **Param. No.** | **Sym.** | **Characteristic** | **Min.** | **Max.** | **Units** | **Conditions** |
| * These parameters are characterized but not tested. | | | | | | |

**Notes:**

1. As a transmitter, the device must provide this internal minimum delay time to bridge the undefined region (min. 300 ns) of the falling edge of SCL to avoid unintended generation of Start or Stop conditions.
2. A Fast mode (400 kHz) $I^2C$ bus device can be used in a Standard mode (100 kHz) $I^2C$ bus system, but the requirement $T_{SU:DAT} \geq 250$ ns must then be met. This will automatically be the case if the device does not stretch the low period of the SCL signal. If such a device does stretch the low period of the SCL signal, it must output the next data bit to the SDA line TR max. + $T_{SU:DAT}$ = 1000 + 250 = 1250 ns (according to the Standard mode $I^2C$ bus specification), before the SCL line is released.
3. Using internal $I^2C$ pull-ups. For greater bus capacitance use external pull-ups.

**Figure 44-20. $I^2C$ Bus Data Timing**




**Note:** Refer to Figure 44-3 for load conditions.

## 44.4.20 Configurable Logic Cell (CLC) Characteristics

**Table 44-26.**

| **Standard Operating Conditions (unless otherwise stated)**<br>**Operating Temperature: -40°C ≤ $T_A$ ≤ +125°C** | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **Param No.** | **Sym.** | **Characteristic** | | **Min.** | **Typ. †** | **Max.** | **Units** | **Conditions** |
| CLC01* | $T_{CLCIN}$ | CLC input time | | — | 7 | IO5 | ns | **(Note 1)** |
| CLC02* | $T_{CLC}$ | CLC module input to output propagation time | | — | 24 | — | ns | $V_{DD}$ = 1.8V |
| | | | | — | 12 | — | ns | $V_{DD}$ > 3.6V |
| CLC03* | $T_{CLCOUT}$ | CLC output time | Rise Time | — | IO6 | — | — | **(Note 1)** |
| | | | Fall Time | — | IO8 | — | — | **(Note 1)** |
| CLC04* | $F_{CLCMAX}$ | CLC maximum switching frequency | | — | — | OS20 | — | |

chanhvinh.com



.....continued

| Standard Operating Conditions (unless otherwise stated)<br>Operating Temperature: -40°C ≤ $T_A$ ≤ +125°C | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Param No.** | **Sym.** | **Characteristic** | **Min.** | **Typ. †** | **Max.** | **Units** | **Conditions** |

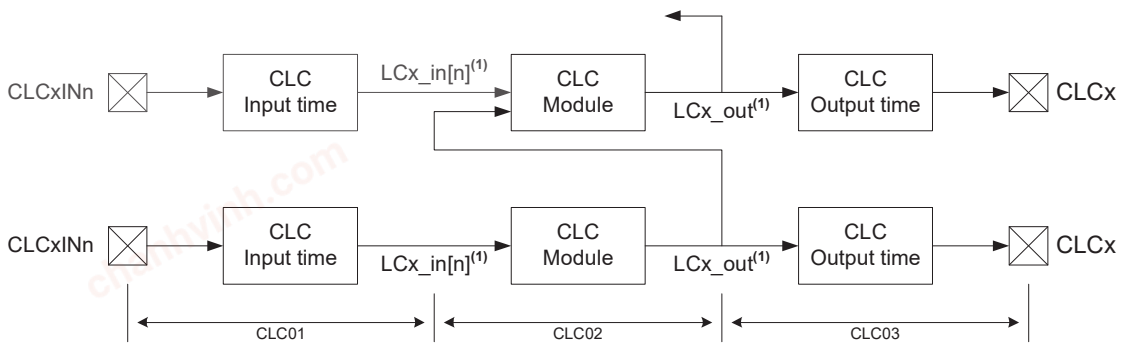
* These parameters are characterized but not tested.

† 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. See the **"I/O and CLKOUT Timing Specifications"** section for IO5, IO6 and IO8 rise and fall times.

**Figure 44-21. CLC Propagation Timing**



chanhvinh.com



# 45. DC and AC Characteristics Graphs and Tables

Graphs and tables are not available at this time.

chanhvinh.com

chanhvinh.com



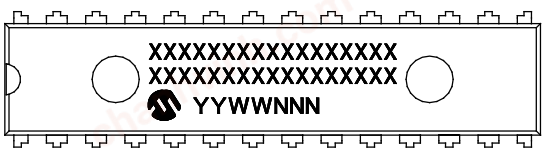
# 46. Packaging Information

Package Marking Information

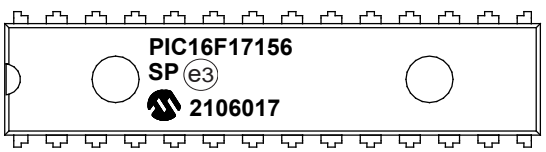
| | | |
|---|---|---|
| **Legend:** | XX...X | Customer-specific information or Microchip part number |
| | Y | Year code (last digit of calendar year) |
| | YY | Year code (last 2 digits of calendar year) |
| | WW | Week code (week of January 1 is week ‘01’) |
| | NNN | Alphanumeric traceability code |
| | (e3) | Pb-free JEDEC® designator for Matte Tin (Sn) |

**Note**: In the event the full Microchip part number cannot be marked on one line, it will be carried over to the next line, thus limiting the number of available characters for customer-specific information.

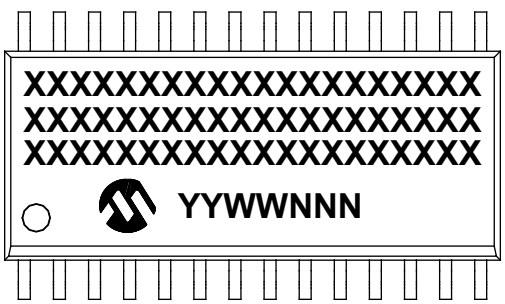
28-Lead SPDIP (.300”)



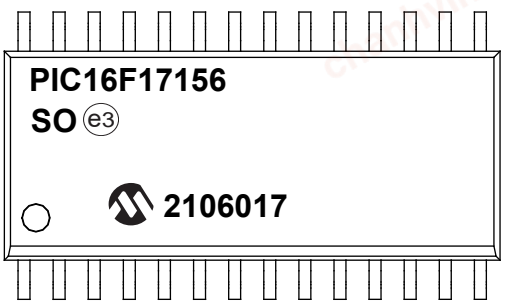

Example




28-Lead SOIC (7.50 mm)



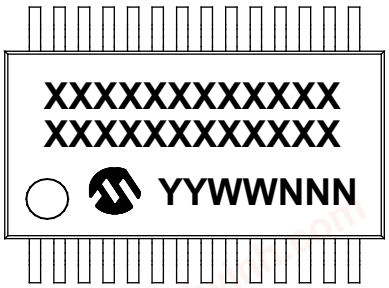

Example



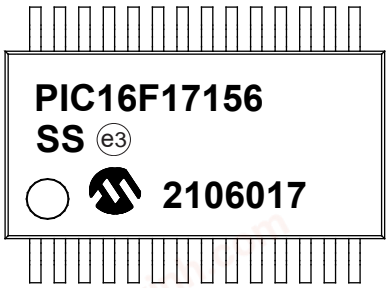
chanhvinh.com



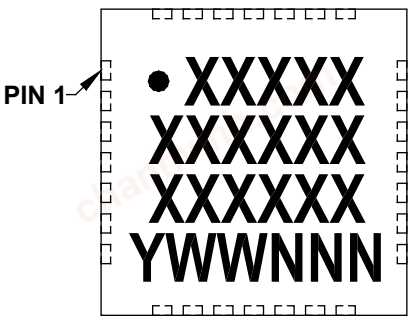
28-Lead SSOP (5.30 mm)



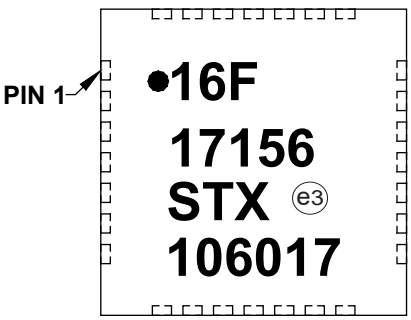

Example



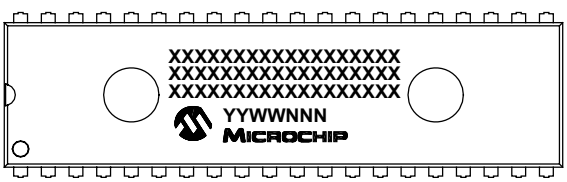

28-Lead VQFN (4x4x1 mm)



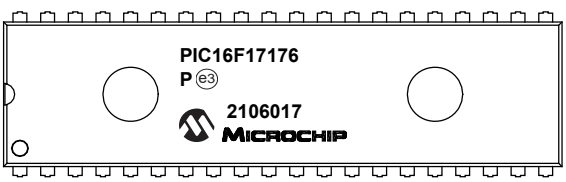

Example




40-Lead PDIP (600 mil)



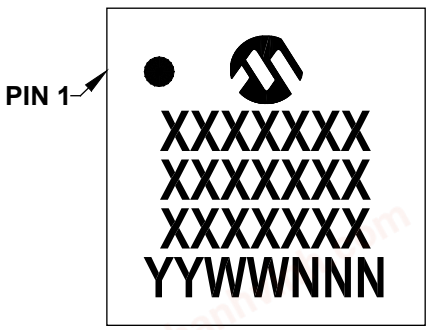

Example



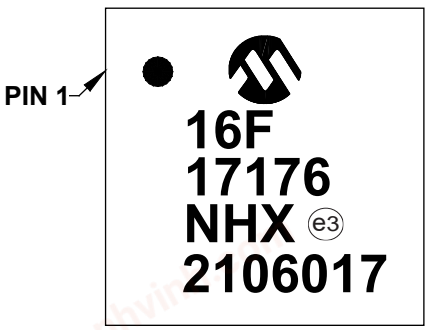
chanhvinh.com



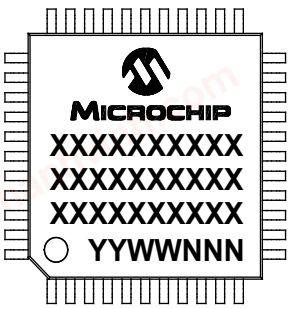
40-Lead VQFN (5x5x0.9 mm)



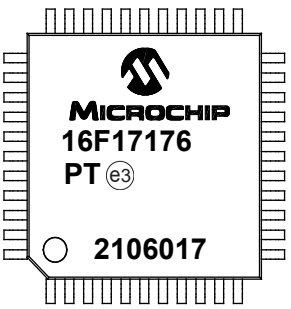

Example




44-Lead TQFP (10x10x1 mm)




Example




## 46.1 Package Details

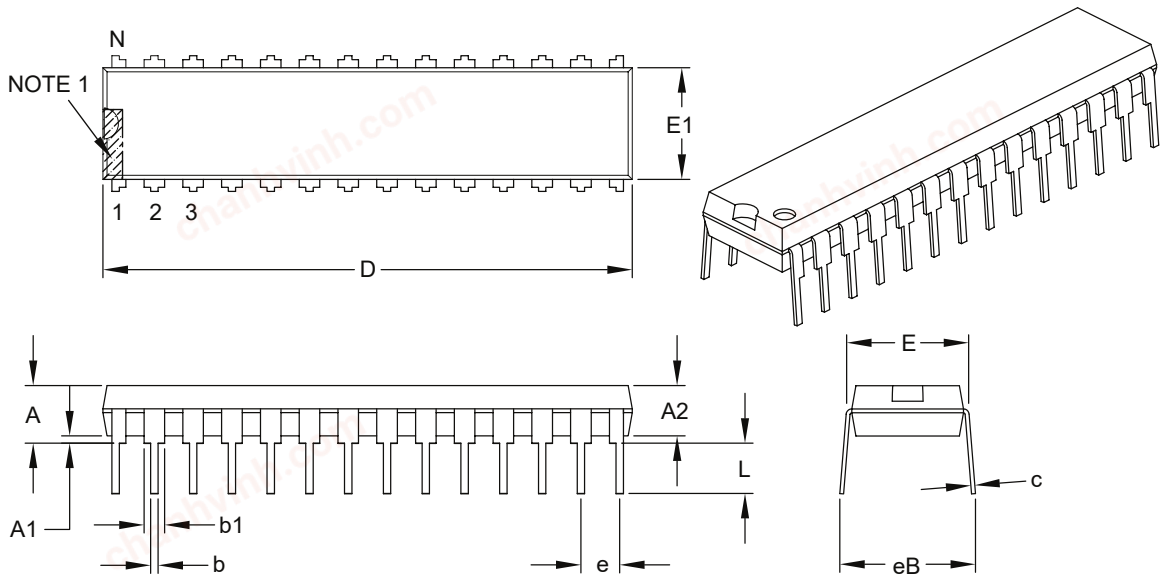
The following sections give the technical details of the packages.

chanhvinh.com



## 28-Lead Skinny Plastic Dual In-Line (SP) – 300 mil Body [SPDIP]

**Note:** For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging




| Units | | INCHES | | |
|---|---|---|---|---|
| Dimension Limits | | MIN | NOM | MAX |
| Number of Pins | N | | 28 | |
| Pitch | e | | .100 BSC | |
| Top to Seating Plane | A | – | – | .200 |
| Molded Package Thickness | A2 | .120 | .135 | .150 |
| Base to Seating Plane | A1 | .015 | – | – |
| Shoulder to Shoulder Width | E | .290 | .310 | .335 |
| Molded Package Width | E1 | .240 | .285 | .295 |
| Overall Length | D | 1.345 | 1.365 | 1.400 |
| Tip to Seating Plane | L | .110 | .130 | .150 |
| Lead Thickness | c | .008 | .010 | .015 |
| Upper Lead Width | b1 | .040 | .050 | .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 within 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.

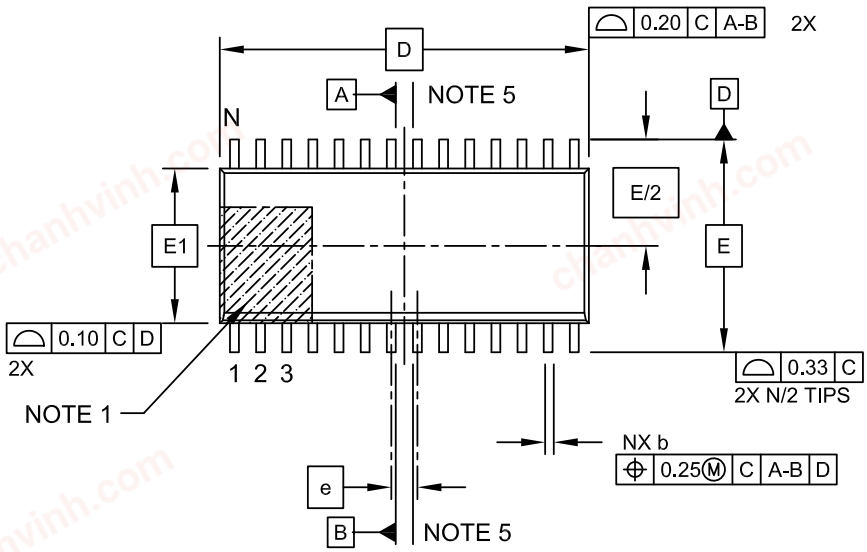
Microchip Technology Drawing C04-070B

chanhvinh.com

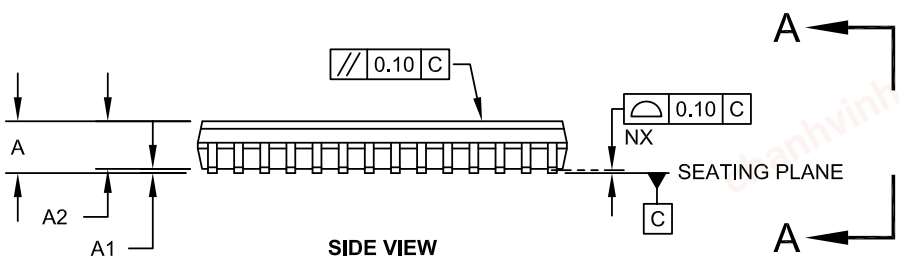


### 28-Lead Plastic Small Outline (SO) - Wide, 7.50 mm Body [SOIC]

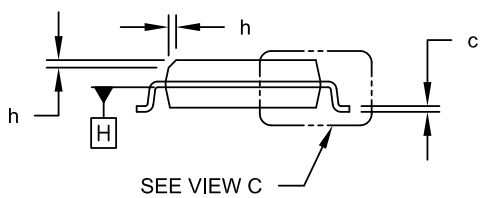
**Note:** For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging




TOP VIEW




SIDE VIEW




VIEW A-A

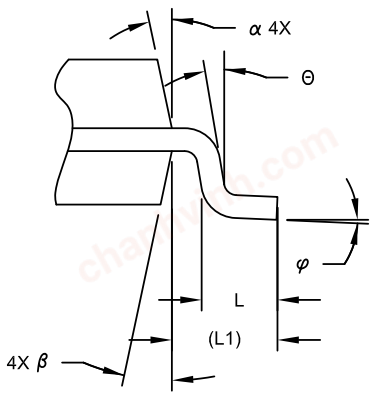
Microchip Technology Drawing C04-052C Sheet 1 of 2

chanhvinh.com

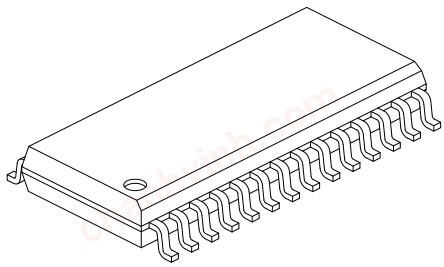


## 28-Lead Plastic Small Outline (SO) - Wide, 7.50 mm Body [SOIC]

**Note:** For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging




VIEW C



| Units | | MILLIMETERS | | |
|---|---|---|---|---|
| Dimension Limits | | MIN | NOM | MAX |
| Number of Pins | N | | 28 | |
| 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 | | 17.90 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 | 0.18 | - | 0.33 |
| Lead Width | b | 0.31 | - | 0.51 |
| Mold Draft Angle Top | α | 5° | - | 15° |
| Mold Draft Angle Bottom | β | 5° | - | 15° |

**Notes:**

1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. § Significant Characteristic
3. 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.
4. 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.
5. Datums A & B to be determined at Datum H.

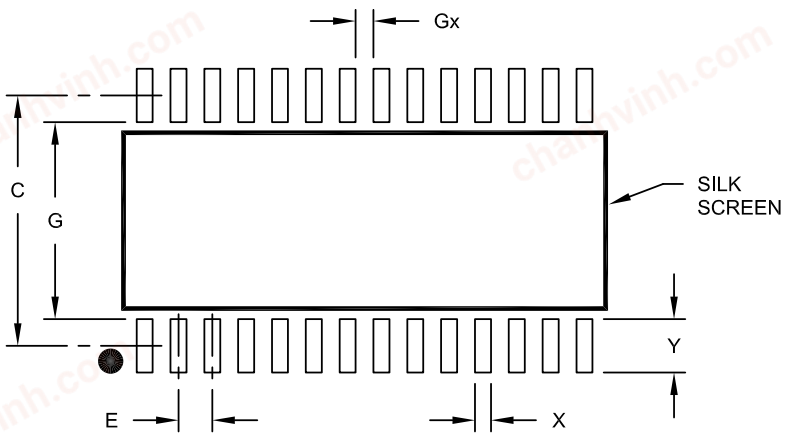
Microchip Technology Drawing C04-052C Sheet 2 of 2

chanhvinh.com



28-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

| Units | | MILLIMETERS | | |
|---|---|---|---|---|
| Dimension Limits | | MIN | NOM | MAX |
| Contact Pitch | E | | 1.27 BSC | |
| Contact Pad Spacing | C | | 9.40 | |
| Contact Pad Width (X28) | X | | | 0.60 |
| Contact Pad Length (X28) | Y | | | 2.00 |
| Distance Between Pads | Gx | 0.67 | | |
| Distance Between Pads | G | 7.40 | | |

Notes:

1. Dimensioning and tolerancing per ASME Y14.5M

BSC: Basic Dimension. Theoretically exact value shown without tolerances.

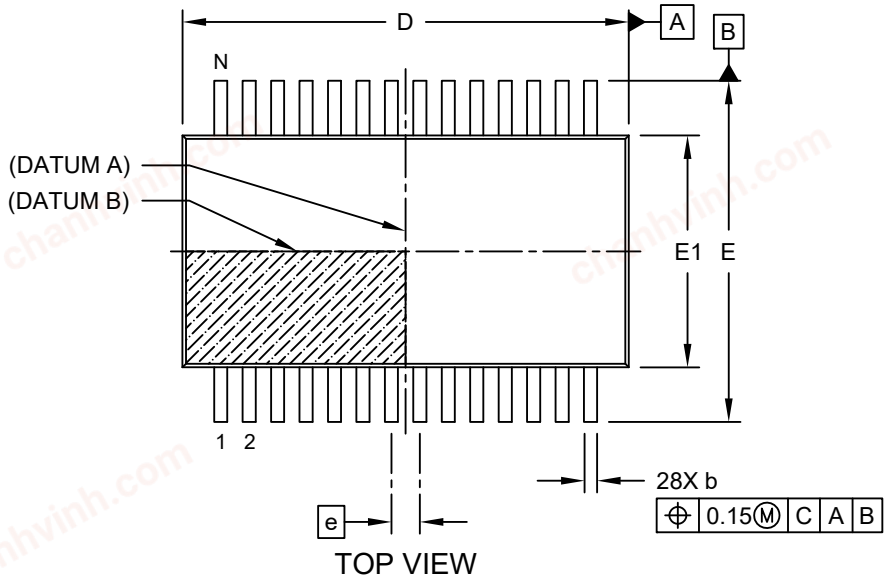
Microchip Technology Drawing No. C04-2052A

chanhvinh.com

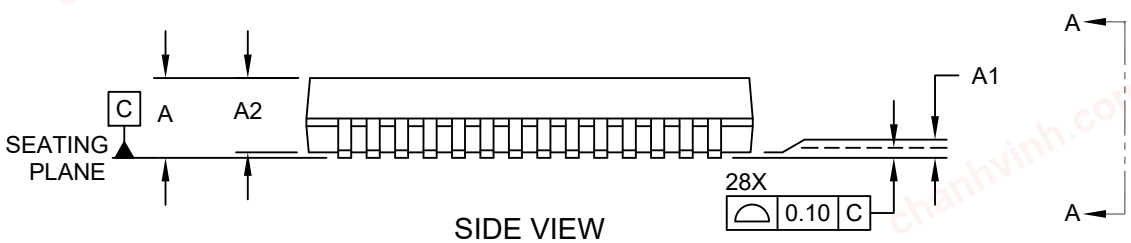


## 28-Lead Plastic Shrink Small Outline (SS) - 5.30 mm Body [SSOP]

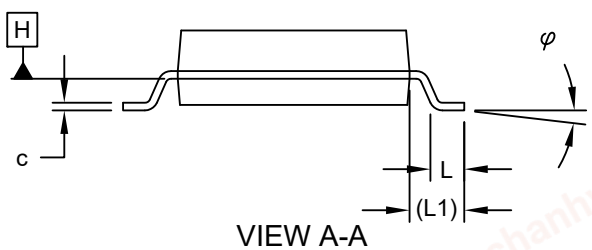
**Note:** For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging




TOP VIEW




SIDE VIEW




VIEW A-A

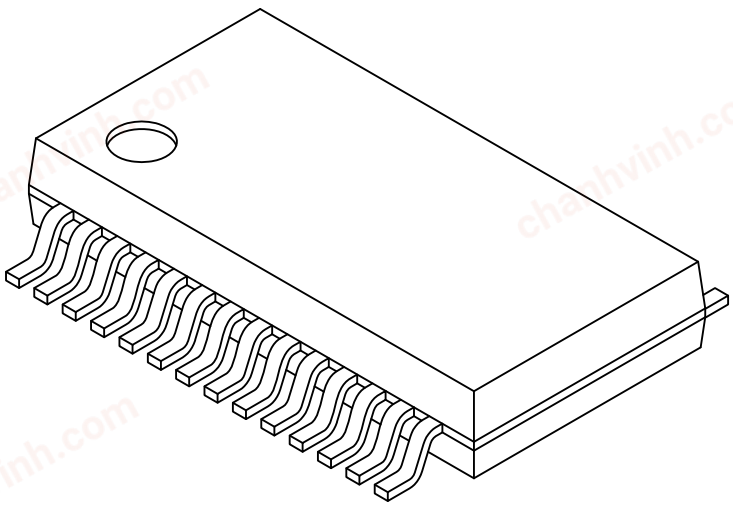
Microchip Technology Drawing C04-073 Rev C Sheet 1 of 2

chanhvinh.com



## 28-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



| Units | | MILLIMETERS | | |
|---|---|---|---|---|
| Dimension Limits | | MIN | NOM | MAX |
| Number of Pins | N | | 28 | |
| 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 | 9.90 | 10.20 | 10.50 |
| 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 0.20mm 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 C04-073 Rev C Sheet 2 of 2

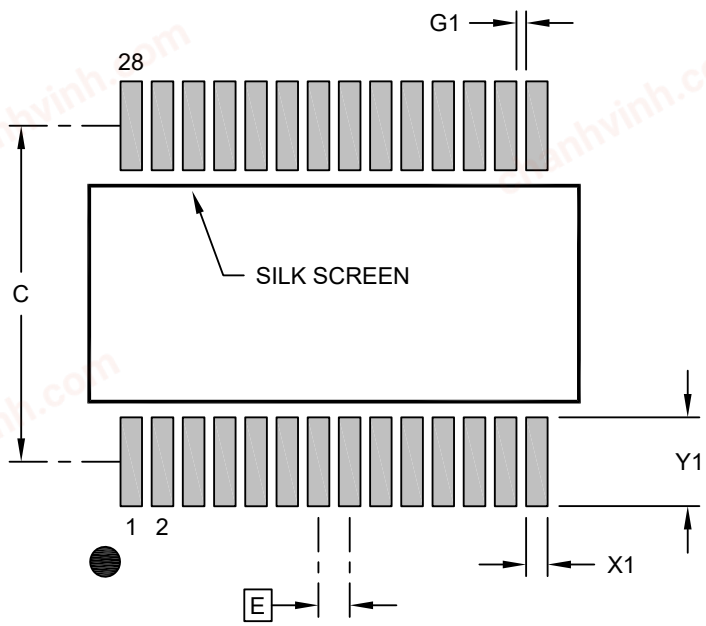
© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



## 28-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




RECOMMENDED LAND PATTERN

| Units | | MILLIMETERS | | |
|---|---|---|---|---|
| Dimension Limits | | MIN | NOM | MAX |
| Contact Pitch | E | | 0.65 BSC | |
| Contact Pad Spacing | C | | 7.00 | |
| Contact Pad Width (X28) | X1 | | | 0.45 |
| Contact Pad Length (X28) | Y1 | | | 1.85 |
| Contact Pad to Center Pad (X26) | G1 | 0.20 | | |

Notes:

1. Dimensioning and tolerancing per ASME Y14.5M
   BSC: Basic Dimension. Theoretically exact value shown without tolerances.
2. For best soldering results, thermal vias, if used, should be filled or tented to avoid solder loss during reflow process

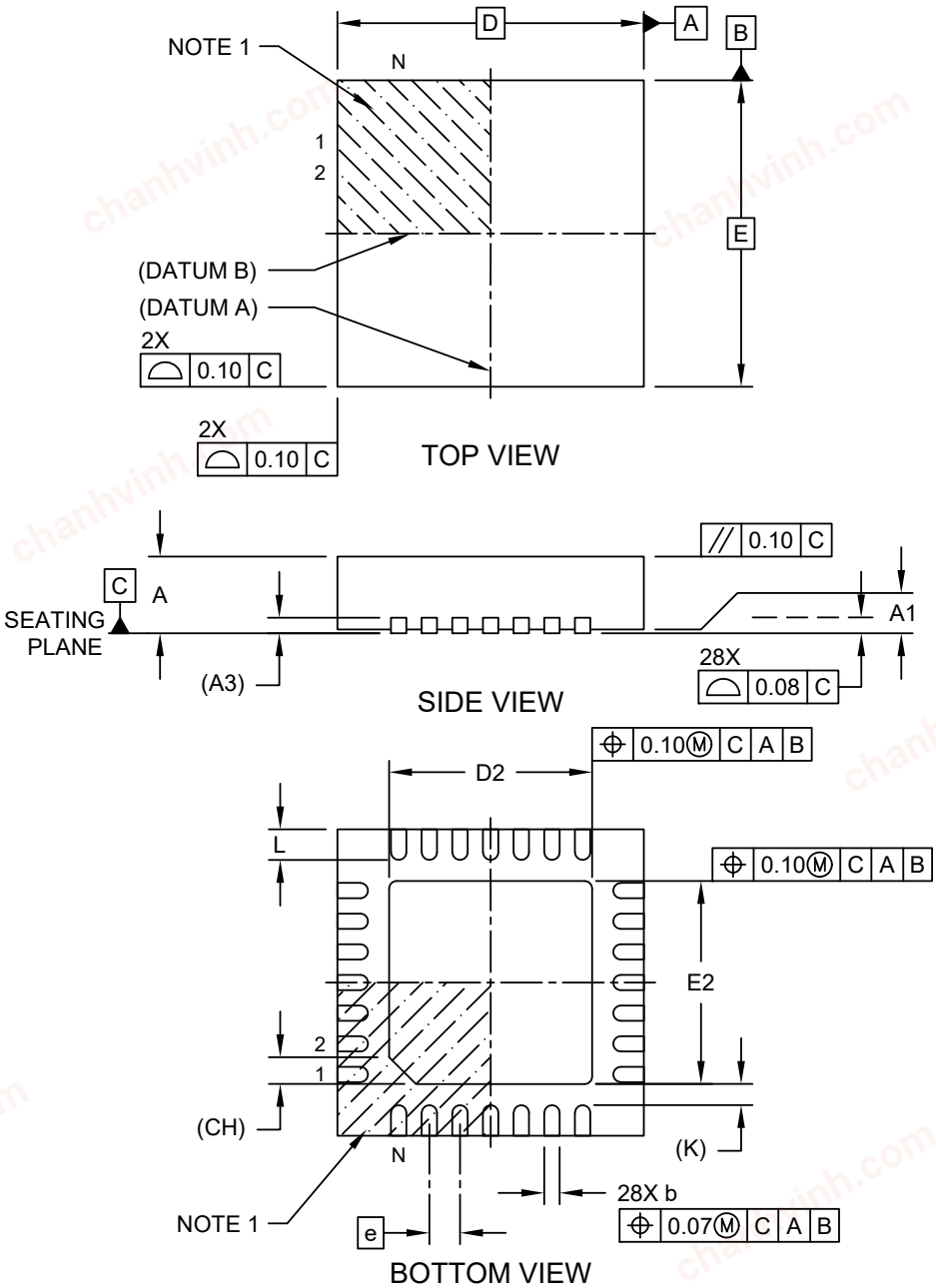
Microchip Technology Drawing C04-2073 Rev B

chanhvinh.com



### 28-Lead Very Thin Plastic Quad Flat, No Lead (STX) - 4x4x1.0 mm Body [VQFN] With 2.65x2.65 mm Exposed Pad

**Note:** For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging



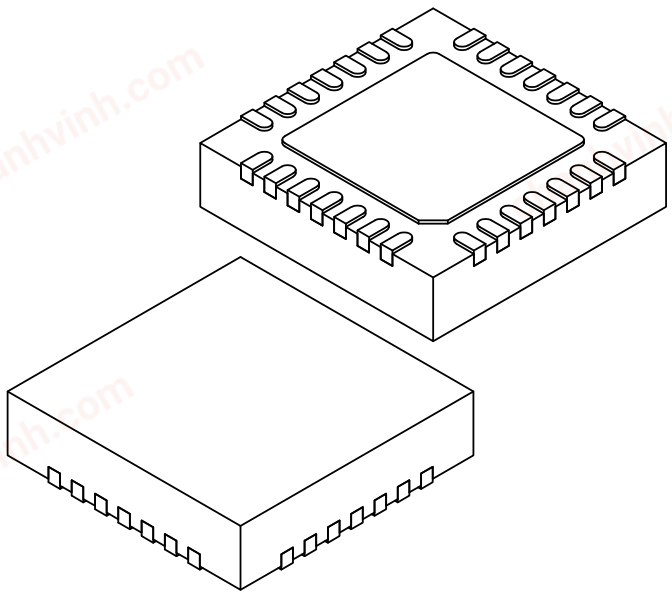

Microchip Technology Drawing C04-456 Rev C Sheet 1 of 2

chanhvinh.com



## 28-Lead Very Thin Plastic Quad Flat, No Lead (STX) - 4x4x1.0 mm Body [VQFN] With 2.65x2.65 mm Exposed Pad

**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 | | 28 | |
| Pitch | e | | 0.40 BSC | |
| Overall Height | A | 0.80 | 0.90 | 1.00 |
| Standoff | A1 | 0.00 | 0.02 | 0.05 |
| Terminal Thickness | A3 | | 0.203 REF | |
| Overall Length | D | | 4.00 BSC | |
| Exposed Pad Length | D2 | 2.55 | 2.65 | 2.75 |
| Overall Width | E | | 4.00 BSC | |
| Exposed Pad Width | E2 | 2.55 | 2.65 | 2.75 |
| Exposed Pad Corner Chamfer | CH | | 0.35 REF | |
| Terminal Width | b | 0.15 | 0.20 | 0.25 |
| Terminal Length | L | 0.30 | 0.40 | 0.50 |
| Terminal-to-Exposed-Pad | K | | 0.275 REF | |

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 C04-456 Rev C Sheet 2 of 2

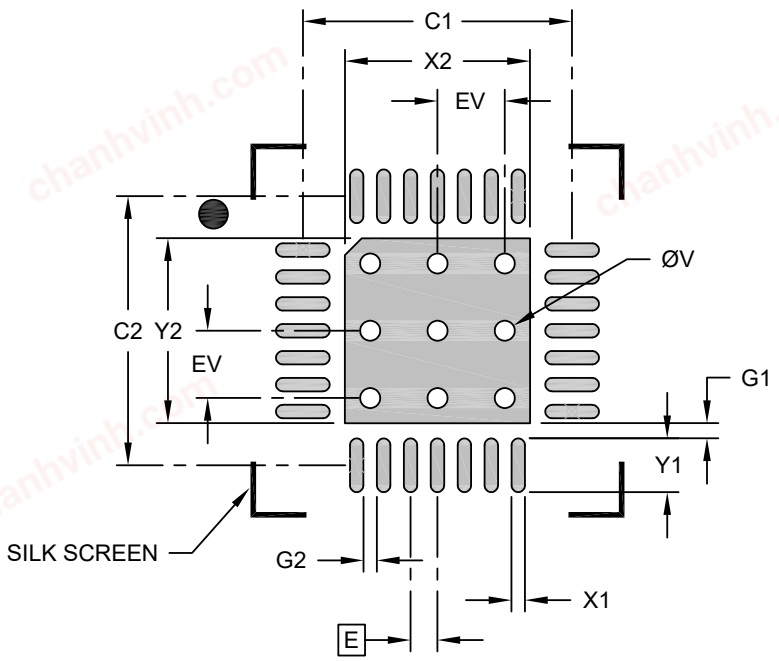
© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



### 28-Lead Very Thin Plastic Quad Flat, No Lead (STX) - 4x4x1.0 mm Body [VQFN] With 2.65x2.65 mm Exposed Pad

**Note:** For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging




RECOMMENDED LAND PATTERN

| Units | | MILLIMETERS | | |
|---|---|---|---|---|
| Dimension Limits | | MIN | NOM | MAX |
| Contact Pitch | E | | 0.40 BSC | |
| Optional Center Pad Width | X2 | | | 2.75 |
| Optional Center Pad Length | Y2 | | | 2.75 |
| Contact Pad Spacing | C1 | | 4.00 | |
| Contact Pad Spacing | C2 | | 4.00 | |
| Contact Pad Width (X28) | X1 | | | 0.20 |
| Contact Pad Length (X28) | Y1 | | | 0.80 |
| Contact Pad to Center Pad (X28) | G1 | 0.23 | | |
| Contact Pad to Contact Pad (X24) | G2 | 0.20 | | |
| Thermal Via Diameter | V | | 0.30 | |
| Thermal Via Pitch | EV | | 1.00 | |

Notes:

1. Dimensioning and tolerancing per ASME Y14.5M
   BSC: Basic Dimension. Theoretically exact value shown without tolerances.
2. For best soldering results, thermal vias, if used, should be filled or tented to avoid solder loss during reflow process

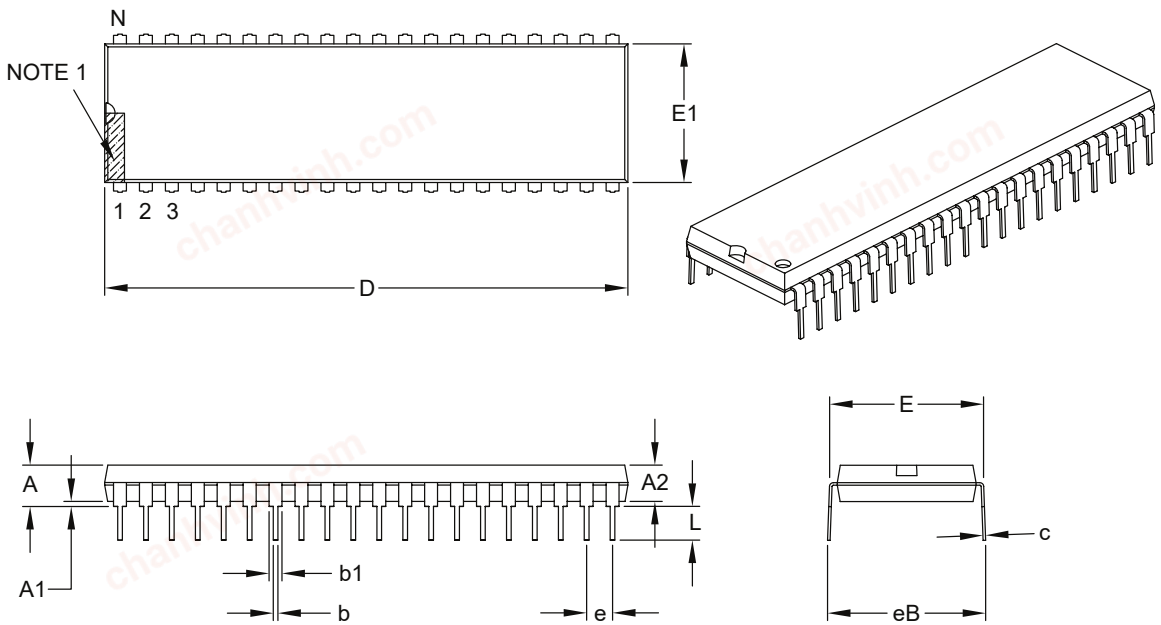
Microchip Technology Drawing C04-2456 Rev C

chanhvinh.com



### 40-Lead Plastic Dual In-Line (P) – 600 mil Body [PDIP]

**Note:** For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging




| Units | | INCHES | | |
|---|---|---|---|---|
| Dimension Limits | | MIN | NOM | MAX |
| Number of Pins | N | | 40 | |
| Pitch | e | | .100 BSC | |
| Top to Seating Plane | A | – | – | .250 |
| Molded Package Thickness | A2 | .125 | – | .195 |
| Base to Seating Plane | A1 | .015 | – | – |
| Shoulder to Shoulder Width | E | .590 | – | .625 |
| Molded Package Width | E1 | .485 | – | .580 |
| Overall Length | D | 1.980 | – | 2.095 |
| Tip to Seating Plane | L | .115 | – | .200 |
| Lead Thickness | c | .008 | – | .015 |
| Upper Lead Width | b1 | .030 | – | .070 |
| Lower Lead Width | b | .014 | – | .023 |
| Overall Row Spacing § | eB | – | – | .700 |

**Notes:**

1. Pin 1 visual index feature may vary, but must be located within 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.

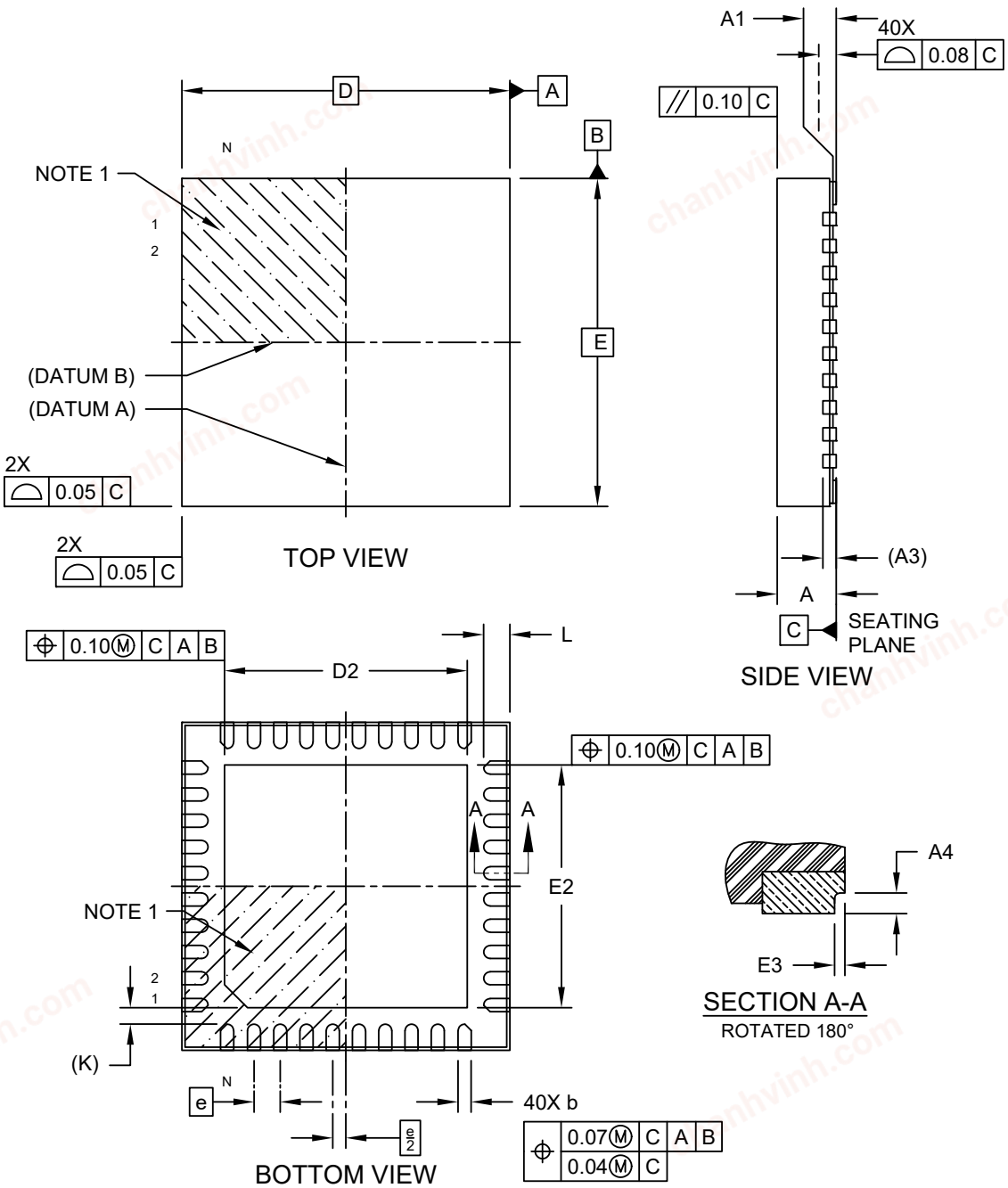
Microchip Technology Drawing C04-016B

chanhvinh.com



**40-Lead Very Thin Plastic Quad Flat, No Lead Package (NHX) - 5x5x0.9 mm Body [VQFN] With 3.7x3.7 mm Exposed Pad and Wettable Flanks**

| **Note:** | For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging |
|---|---|



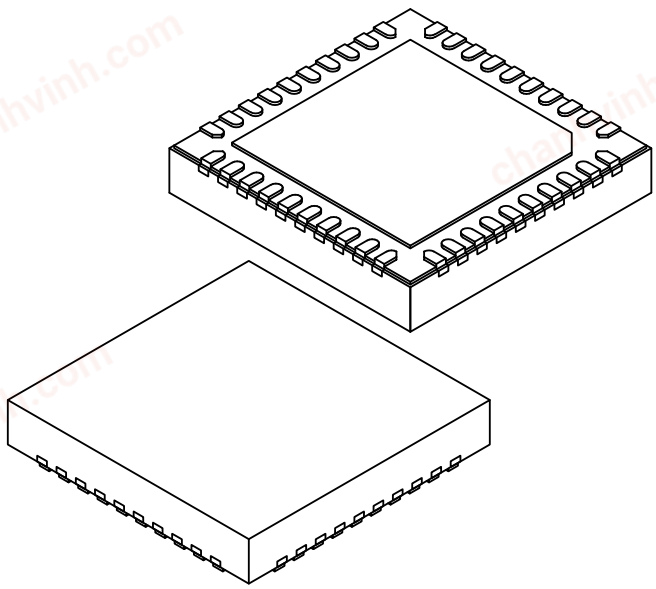

Microchip Technology Drawing C04-425 Rev D Sheet 1 of 2

chanhvinh.com



## 40-Lead Very Thin Plastic Quad Flat, No Lead Package (NHX) - 5x5x0.9 mm Body [VQFN] With 3.7x3.7 mm Exposed Pad and Wettable Flanks

**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 | | 40 | |
| Pitch | e | | 0.40 BSC | |
| Overall Height | A | 0.80 | 0.85 | 0.90 |
| Standoff | A1 | 0.00 | 0.02 | 0.05 |
| Terminal Thickness | A3 | | 0.203 REF | |
| Overall Length | D | | 5.00 BSC | |
| Exposed Pad Length | D2 | 3.60 | 3.70 | 3.80 |
| Overall Width | E | | 5.00 BSC | |
| Exposed Pad Width | E2 | 3.60 | 3.70 | 3.80 |
| Terminal Width | b | 0.15 | 0.20 | 0.25 |
| Terminal Length | L | 0.30 | 0.40 | 0.50 |
| Wettable Flank Height | A4 | 0.10 | – | 0.19 |
| Wettable Flank Width | E3 | – | – | 0.085 |
| Terminal-to-Exposed-Pad | K | | 0.25 REF | |

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 C04-425 Rev D Sheet 2 of 2

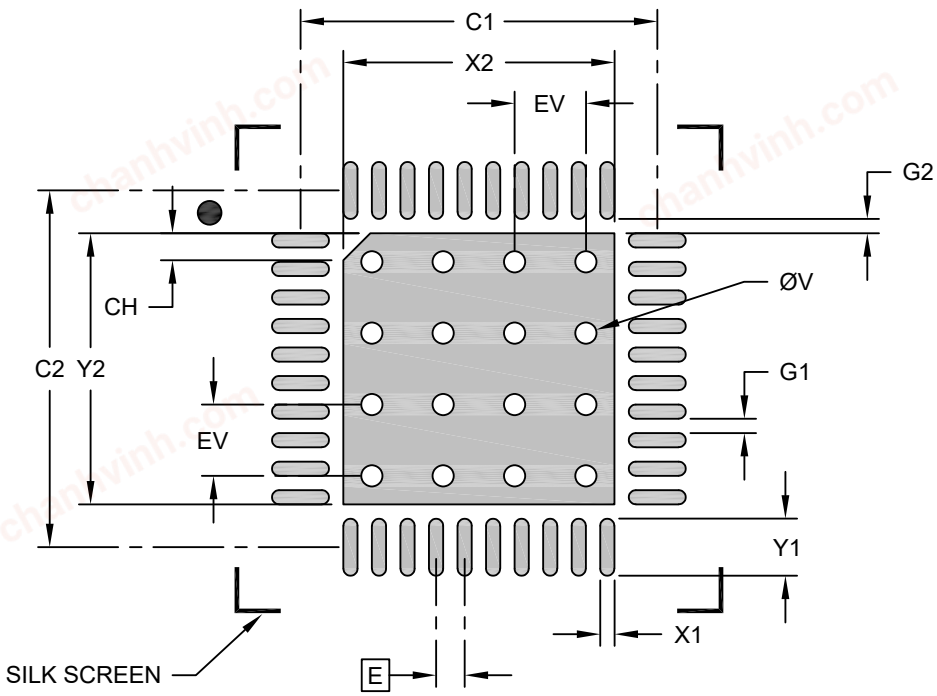
© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



## 40-Lead Very Thin Plastic Quad Flat, No Lead Package (NHX) - 5x5x0.9 mm Body [VQFN] With 3.7x3.7 mm Exposed Pad and Wettable Flanks

**Note:** For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging




RECOMMENDED LAND PATTERN

| Units | | MILLIMETERS | | |
|---|---|---|---|---|
| Dimension Limits | | MIN | NOM | MAX |
| Contact Pitch | E | | 0.40 BSC | |
| Optional Center Pad Width | X2 | | | 3.80 |
| Optional Center Pad Length | Y2 | | | 3.80 |
| Chamfer | CH | | 0.38 | |
| Contact Pad Spacing | C1 | | 5.00 | |
| Contact Pad Spacing | C2 | | 5.00 | |
| Contact Pad Width (X40) | X1 | | | 0.20 |
| Contact Pad Length (X40) | Y1 | | | 0.80 |
| Contact Pad to Pad (X36) | G1 | 0.20 | | |
| Contact Pad to Center Pad (X40) | G2 | 0.20 | | |
| Thermal Via Diameter | V | | 0.30 | |
| Thermal Via Pitch | EV | | 1.00 | |

Notes:

1. Dimensioning and tolerancing per ASME Y14.5M
   BSC: Basic Dimension. Theoretically exact value shown without tolerances.
2. For best soldering results, thermal vias, if used, should be filled or tented to avoid solder loss during reflow process

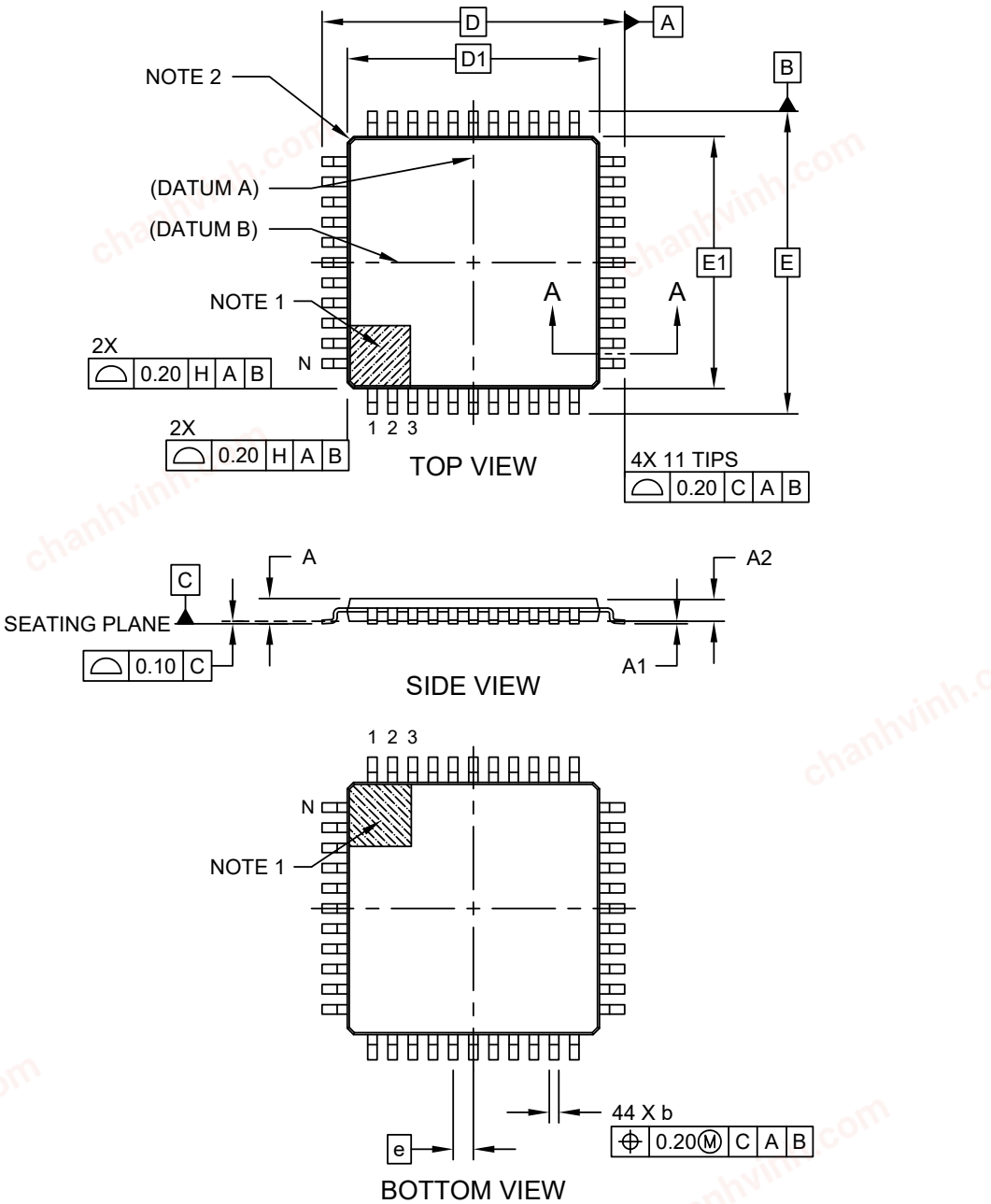
Microchip Technology Drawing C04-2425 Rev D

chanhvinh.com



### 44-Lead Plastic Thin Quad Flatpack (PT) - 10x10x1.0 mm Body [TQFP]

**Note:** For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging



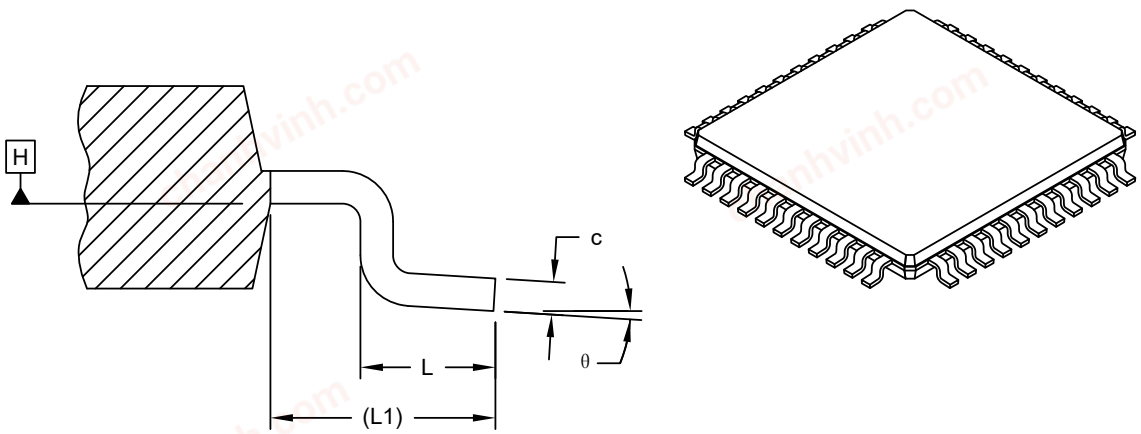

Microchip Technology Drawing C04-076C Sheet 1 of 2

chanhvinh.com



### 44-Lead Plastic Thin Quad Flatpack (PT) - 10x10x1.0 mm Body [TQFP]

**Note:** For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging




SECTION A-A

| Units | | MILLIMETERS | | |
|---|---|---|---|---|
| Dimension Limits | | MIN | NOM | MAX |
| Number of Leads | N | | 44 | |
| Lead Pitch | e | | 0.80 BSC | |
| Overall Height | A | - | - | 1.20 |
| Standoff | A1 | 0.05 | - | 0.15 |
| Molded Package Thickness | A2 | 0.95 | 1.00 | 1.05 |
| Overall Width | E | | 12.00 BSC | |
| Molded Package Width | E1 | | 10.00 BSC | |
| Overall Length | D | | 12.00 BSC | |
| Molded Package Length | D1 | | 10.00 BSC | |
| Lead Width | b | 0.30 | 0.37 | 0.45 |
| Lead Thickness | c | 0.09 | - | 0.20 |
| Lead Length | L | 0.45 | 0.60 | 0.75 |
| Footprint | L1 | | 1.00 REF | |
| Foot Angle | θ | 0° | 3.5° | 7° |

Notes:

1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. Exact shape of each corner is optional.
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 C04-076C Sheet 2 of 2

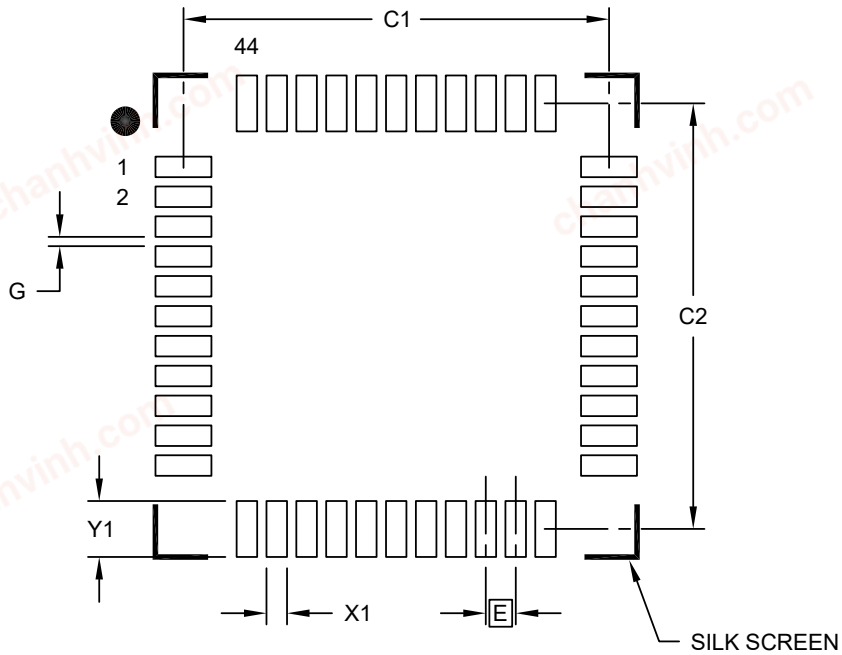
© 2023 Microchip Technology Inc.
and its subsidiaries

chanhvinh.com



### 44-Lead Plastic Thin Quad Flatpack (PT) - 10X10X1 mm Body, 2.00 mm Footprint [TQFP]

**Note:** For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging




RECOMMENDED LAND PATTERN

| Units | | MILLIMETERS | | |
|---|---|---|---|---|
| Dimension Limits | | MIN | NOM | MAX |
| Contact Pitch | E | | 0.80 BSC | |
| Contact Pad Spacing | C1 | | 11.40 | |
| Contact Pad Spacing | C2 | | 11.40 | |
| Contact Pad Width (X44) | X1 | | | 0.55 |
| Contact Pad Length (X44) | Y1 | | | 1.50 |
| Distance Between Pads | G | 0.25 | | |

Notes:

1. Dimensioning and tolerancing per ASME Y14.5M

BSC: Basic Dimension. Theoretically exact value shown without tolerances.

Microchip Technology Drawing No. C04-2076B

chanhvinh.com



# 47. Appendix A: Revision History

| Doc. Rev. | Date | Comments |
|---|---|---|
| A | 04/2022 | Initial document release. |

chanhvinh.com

chanhvinh.com



# Microchip Information

## The Microchip Website

Microchip provides online support via our website at www.microchip.com/. This website is used to make files and information easily available to customers. Some of the content available includes:

- **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 (FAQs), technical support requests, online discussion groups, Microchip design partner 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

## Product Change Notification Service

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

To register, go to www.microchip.com/pcn and follow the registration instructions.

## Customer Support

Users of Microchip products can receive assistance through several channels:

- Distributor or Representative
- Local Sales Office
- Embedded Solutions Engineer (ESE)
- Technical Support

Customers should contact their distributor, representative or ESE for support. Local sales offices are also available to help customers. A listing of sales offices and locations is included in this document.

Technical support is available through the website at: www.microchip.com/support

chanhvinh.com

chanhvinh.com



## Product Identification System

To order or obtain information, e.g., on pricing or delivery, refer to the factory or the listed sales office.

PART NO. [X](1) –X /XX

Device — Tape and Reel — Temperature Range — Package

| | | |
|---|---|---|
| Device: | PIC16F17156, PIC16F17176 | |
| Tape & Reel Option: | Blank | = Tube |
| | T | = Tape & Reel |
| Temperature Range: | I | = -40°C to +85°C (Industrial) |
| | E | = -40°C to +125°C (Extended) |
| Package: | SP | = 28-lead SPDIP |
| | SO | = 28-lead SOIC |
| | SS | = 28-lead SSOP |
| | STX | = 28-lead VQFN |
| | P | = 40-lead PDIP |
| | NHX | = 40-lead VQFN |
| | PT | = 44-lead TQFP |

Examples:

- PIC16F17156 T-E/SP: Tape and Reel, Extended temperature, 28-lead SPDIP
- PIC16F17176 T-I/PT: Tape and Reel, Industrial temperature, 44-lead TQFP

**Notes:**

1. Tape and Reel identifier only appears in the catalog part number description. This identifier is used for ordering purposes and is not printed on the device package. Check with your Microchip Sales Office for package availability with the Tape and Reel option.
2. Small form-factor packaging options may be available. Please check www.microchip.com/packaging for small-form factor package availability, or contact your local Sales Office.

## Microchip Devices Code Protection Feature

Note the following details of the code protection feature on Microchip products:

- Microchip products meet the specifications contained in their particular Microchip Data Sheet.
- Microchip believes that its family of products is secure when used in the intended manner, within operating specifications, and under normal conditions.
- Microchip values and aggressively protects its intellectual property rights. Attempts to breach the code protection features of Microchip product is strictly prohibited and may violate the Digital Millennium Copyright Act.
- Neither Microchip nor any other semiconductor manufacturer can guarantee the security of its code. Code protection does not mean that we are guaranteeing the product is “unbreakable”. Code protection is constantly evolving. Microchip is committed to continuously improving the code protection features of our products.

## Legal Notice

This publication and the information herein may be used only with Microchip products, including to design, test, and integrate Microchip products with your application. Use of this information in any other manner violates these

© 2023 Microchip Technology Inc. and its subsidiaries

chanhvinh.com



terms. Information regarding device applications 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. Contact your local Microchip sales office for additional support or, obtain additional support at www.microchip.com/en-us/support/design-help/client-support-services.

THIS INFORMATION IS PROVIDED BY MICROCHIP "AS IS". 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 ANY IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY, AND FITNESS FOR A PARTICULAR PURPOSE, OR WARRANTIES RELATED TO ITS CONDITION, QUALITY, OR PERFORMANCE.

IN NO EVENT WILL MICROCHIP BE LIABLE FOR ANY INDIRECT, SPECIAL, PUNITIVE, INCIDENTAL, OR CONSEQUENTIAL LOSS, DAMAGE, COST, OR EXPENSE OF ANY KIND WHATSOEVER RELATED TO THE INFORMATION OR ITS USE, HOWEVER CAUSED, EVEN IF MICROCHIP HAS BEEN ADVISED OF THE POSSIBILITY OR THE DAMAGES ARE FORESEEABLE. TO THE FULLEST EXTENT ALLOWED BY LAW, MICROCHIP'S TOTAL LIABILITY ON ALL CLAIMS IN ANY WAY RELATED TO THE INFORMATION OR ITS USE WILL NOT EXCEED THE AMOUNT OF FEES, IF ANY, THAT YOU HAVE PAID DIRECTLY TO MICROCHIP FOR THE INFORMATION.

Use of Microchip devices in life support and/or safety applications is entirely at the buyer's risk, and the buyer agrees to defend, indemnify and hold harmless Microchip from any and all damages, claims, suits, or expenses resulting from such use. No licenses are conveyed, implicitly or otherwise, under any Microchip intellectual property rights unless otherwise stated.

## Trademarks

The Microchip name and logo, the Microchip logo, Adaptec, AVR, AVR logo, AVR Freaks, BesTime, BitCloud, CryptoMemory, CryptoRF, dsPIC, flexPWR, HELDO, IGLOO, JukeBlox, KeeLoq, Kleer, LANCheck, LinkMD, maXStylus, maXTouch, MediaLB, megaAVR, Microsemi, Microsemi logo, MOST, MOST logo, MPLAB, OptoLyzer, PIC, picoPower, PICSTART, PIC32 logo, PolarFire, Prochip Designer, QTouch, SAM-BA, SenGenuity, SpyNIC, SST, SST Logo, SuperFlash, Symmetricom, SyncServer, Tachyon, TimeSource, tinyAVR, UNI/O, Vectron, and XMEGA are registered trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

AgileSwitch, APT, ClockWorks, The Embedded Control Solutions Company, EtherSynch, Flashtec, Hyper Speed Control, HyperLight Load, Libero, motorBench, mTouch, Powermite 3, Precision Edge, ProASIC, ProASIC Plus, ProASIC Plus logo, Quiet- Wire, SmartFusion, SyncWorld, Temux, TimeCesium, TimeHub, TimePictra, TimeProvider, TrueTime, and ZL 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, Augmented Switching, BlueSky, BodyCom, Clockstudio, CodeGuard, CryptoAuthentication, CryptoAutomotive, CryptoCompanion, CryptoController, dsPICDEM, dsPICDEM.net, Dynamic Average Matching, DAM, ECAN, Espresso T1S, EtherGREEN, GridTime, IdealBridge, In-Circuit Serial Programming, ICSP, INICnet, Intelligent Paralleling, IntelliMOS, Inter-Chip Connectivity, JitterBlocker, Knob-on-Display, KoD, maxCrypto, maxView, memBrain, Mindi, MiWi, MPASM, MPF, MPLAB Certified logo, MPLIB, MPLINK, MultiTRAK, NetDetach, Omniscient Code Generation, PICDEM, PICDEM.net, PICkit, PICtail, PowerSmart, PureSilicon, QMatrix, REAL ICE, Ripple Blocker, RTAX, RTG4, SAM-ICE, Serial Quad I/O, simpleMAP, SimpliPHY, SmartBuffer, SmartHLS, SMART-I.S., storClad, SQI, SuperSwitcher, SuperSwitcher II, Switchtec, SynchroPHY, Total Endurance, Trusted Time, TSHARC, USBCheck, VariSense, VectorBlox, VeriPHY, ViewSpan, WiperLock, XpressConnect, 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.

The Adaptec logo, Frequency on Demand, Silicon Storage Technology, and Symmcom are registered trademarks 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.

© 2023, Microchip Technology Incorporated and its subsidiaries. All Rights Reserved.

ISBN: 978-1-6683-2186-7

chanhvinh.com



## Quality Management System

For information regarding Microchip's Quality Management Systems, please visit www.microchip.com/quality.

chanhvinh.com

chanhvinh.com

# Worldwide Sales and Service


| AMERICAS | ASIA/PACIFIC | ASIA/PACIFIC | EUROPE |
|---|---|---|---|
| **Corporate Office**<br>2355 West Chandler Blvd.<br>Chandler, AZ 85224-6199<br>Tel: 480-792-7200<br>Fax: 480-792-7277<br>Technical Support:<br>www.microchip.com/support<br>Web Address:<br>www.microchip.com<br>**Atlanta**<br>Duluth, GA<br>Tel: 678-957-9614<br>Fax: 678-957-1455<br>**Austin, TX**<br>Tel: 512-257-3370<br>**Boston**<br>Westborough, MA<br>Tel: 774-760-0087<br>Fax: 774-760-0088<br>**Chicago**<br>Itasca, IL<br>Tel: 630-285-0071<br>Fax: 630-285-0075<br>**Dallas**<br>Addison, TX<br>Tel: 972-818-7423<br>Fax: 972-818-2924<br>**Detroit**<br>Novi, MI<br>Tel: 248-848-4000<br>**Houston, TX**<br>Tel: 281-894-5983<br>**Indianapolis**<br>Noblesville, IN<br>Tel: 317-773-8323<br>Fax: 317-773-5453<br>Tel: 317-536-2380<br>**Los Angeles**<br>Mission Viejo, CA<br>Tel: 949-462-9523<br>Fax: 949-462-9608<br>Tel: 951-273-7800<br>**Raleigh, NC**<br>Tel: 919-844-7510<br>**New York, NY**<br>Tel: 631-435-6000<br>**San Jose, CA**<br>Tel: 408-735-9110<br>Tel: 408-436-4270<br>**Canada - Toronto**<br>Tel: 905-695-1980<br>Fax: 905-695-2078 | **Australia - Sydney**<br>Tel: 61-2-9868-6733<br>**China - Beijing**<br>Tel: 86-10-8569-7000<br>**China - Chengdu**<br>Tel: 86-28-8665-5511<br>**China - Chongqing**<br>Tel: 86-23-8980-9588<br>**China - Dongguan**<br>Tel: 86-769-8702-9880<br>**China - Guangzhou**<br>Tel: 86-20-8755-8029<br>**China - Hangzhou**<br>Tel: 86-571-8792-8115<br>**China - Hong Kong SAR**<br>Tel: 852-2943-5100<br>**China - Nanjing**<br>Tel: 86-25-8473-2460<br>**China - Qingdao**<br>Tel: 86-532-8502-7355<br>**China - Shanghai**<br>Tel: 86-21-3326-8000<br>**China - Shenyang**<br>Tel: 86-24-2334-2829<br>**China - Shenzhen**<br>Tel: 86-755-8864-2200<br>**China - Suzhou**<br>Tel: 86-186-6233-1526<br>**China - Wuhan**<br>Tel: 86-27-5980-5300<br>**China - Xian**<br>Tel: 86-29-8833-7252<br>**China - Xiamen**<br>Tel: 86-592-2388138<br>**China - Zhuhai**<br>Tel: 86-756-3210040 | **India - Bangalore**<br>Tel: 91-80-3090-4444<br>**India - New Delhi**<br>Tel: 91-11-4160-8631<br>**India - Pune**<br>Tel: 91-20-4121-0141<br>**Japan - Osaka**<br>Tel: 81-6-6152-7160<br>**Japan - Tokyo**<br>Tel: 81-3-6880- 3770<br>**Korea - Daegu**<br>Tel: 82-53-744-4301<br>**Korea - Seoul**<br>Tel: 82-2-554-7200<br>**Malaysia - Kuala Lumpur**<br>Tel: 60-3-7651-7906<br>**Malaysia - Penang**<br>Tel: 60-4-227-8870<br>**Philippines - Manila**<br>Tel: 63-2-634-9065<br>**Singapore**<br>Tel: 65-6334-8870<br>**Taiwan - Hsin Chu**<br>Tel: 886-3-577-8366<br>**Taiwan - Kaohsiung**<br>Tel: 886-7-213-7830<br>**Taiwan - Taipei**<br>Tel: 886-2-2508-8600<br>**Thailand - Bangkok**<br>Tel: 66-2-694-1351<br>**Vietnam - Ho Chi Minh**<br>Tel: 84-28-5448-2100 | **Austria - Wels**<br>Tel: 43-7242-2244-39<br>Fax: 43-7242-2244-393<br>**Denmark - Copenhagen**<br>Tel: 45-4485-5910<br>Fax: 45-4485-2829<br>**Finland - Espoo**<br>Tel: 358-9-4520-820<br>**France - Paris**<br>Tel: 33-1-69-53-63-20<br>Fax: 33-1-69-30-90-79<br>**Germany - Garching**<br>Tel: 49-8931-9700<br>**Germany - Haan**<br>Tel: 49-2129-3766400<br>**Germany - Heilbronn**<br>Tel: 49-7131-72400<br>**Germany - Karlsruhe**<br>Tel: 49-721-625370<br>**Germany - Munich**<br>Tel: 49-89-627-144-0<br>Fax: 49-89-627-144-44<br>**Germany - Rosenheim**<br>Tel: 49-8031-354-560<br>**Israel - Ra'anana**<br>Tel: 972-9-744-7705<br>**Italy - Milan**<br>Tel: 39-0331-742611<br>Fax: 39-0331-466781<br>**Italy - Padova**<br>Tel: 39-049-7625286<br>**Netherlands - Drunen**<br>Tel: 31-416-690399<br>Fax: 31-416-690340<br>**Norway - Trondheim**<br>Tel: 47-72884388<br>**Poland - Warsaw**<br>Tel: 48-22-3325737<br>**Romania - Bucharest**<br>Tel: 40-21-407-87-50<br>**Spain - Madrid**<br>Tel: 34-91-708-08-90<br>Fax: 34-91-708-08-91<br>**Sweden - Gothenberg**<br>Tel: 46-31-704-60-40<br>**Sweden - Stockholm**<br>Tel: 46-8-5090-4654<br>**UK - Wokingham**<br>Tel: 44-118-921-5800<br>Fax: 44-118-921-5820 |


© 2023 Microchip Technology Inc.
and its subsidiaries