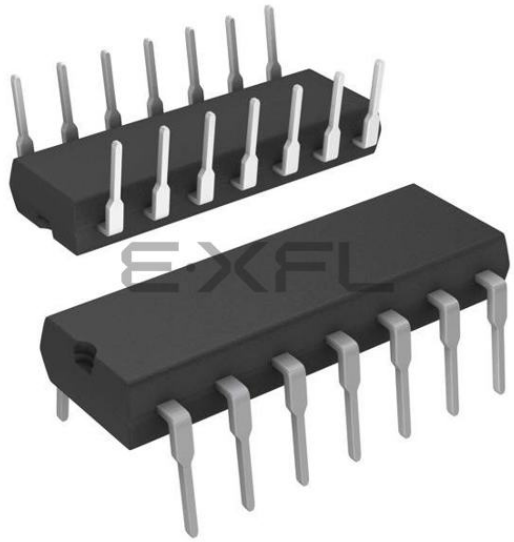


**Welcome to E-XFL.COM**

## What is "Embedded - Microcontrollers"?

"Embedded - Microcontrollers" refer to small, integrated circuits designed to perform specific tasks within larger systems. These microcontrollers are essentially compact computers on a single chip, containing a processor core, memory, and programmable input/output peripherals. They are called "embedded" because they are embedded within electronic devices to control various functions, rather than serving as standalone computers. Microcontrollers are crucial in modern electronics, providing the intelligence and control needed for a wide range of applications.

## Applications of "Embedded - Microcontrollers"

| Details | |
|---|---|
| Product Status | Active |
| Core Processor | PIC |
| Core Size | 8-Bit |
| Speed | 32MHz |
| Connectivity | I²C, LINbus, SPI, UART/USART |
| Peripherals | Brown-out Detect/Reset, POR, PWM, WDT |
| Number of I/O | 12 |
| Program Memory Size | 28KB (16K x 14) |
| Program Memory Type | FLASH |
| EEPROM Size | 256 x 8 |
| RAM Size | 2K x 8 |
| Voltage - Supply (Vcc/Vdd) | 1.8V ~ 3.6V |
| Data Converters | A/D 11x10b; D/A 1x5b |
| Oscillator Type | Internal |
| Operating Temperature | -40°C ~ 85°C (TA) |
| Mounting Type | Through Hole |
| Package / Case | 14-DIP (0.300", 7.62mm) |
| Supplier Device Package | 14-PDIP |
| Purchase URL | https://www.e-xfl.com/product-detail/microchip-technology/pic16lf18326-i-p |

**TABLE 1-3: PIC16(L)F18346 PINOUT DESCRIPTION**

| Name | Function | Input Type | Output Type | Description |
|---|---|---|---|---|
| RA0/ANA0/C1IN0+/DAC1OUT/ ICDDAT/ICSPDAT | RA0 | TTL/ST | CMOS | General purpose I/O. |
| | ANA0 | AN | — | ADC Channel A0 input. |
| | C1IN0+ | AN | — | Comparator C1 positive input. |
| | DAC1OUT | — | AN | Digital-to-Analog Converter output. |
| | ICDDAT | TTL/ST | CMOS | In-Circuit Debug Data I/O. |
| | ICSPDAT | TTL/ST | CMOS | ICSP™ Data I/O. |
| RA1/ANA1/VREF+/C1IN0-/ C2IN0-/ DAC1REF+/$\overline{SS2}$(1)/ ICDCLK/ ICSPCLK | RA1 | TTL/ST | CMOS | General purpose I/O. |
| | ANA1 | AN | — | ADC Channel A1 input. |
| | VREF+ | AN | — | ADC positive voltage reference input. |
| | C1IN0- | AN | — | Comparator C1 negative input. |
| | C2IN0- | AN | — | Comparator C2 negative input. |
| | DAC1REF+ | AN | — | Digital-to-Analog Converter positive reference input. |
| | $\overline{SS2}$ | TTL/ST | — | Slave Select 2 input. |
| | ICDCLK | TTL/ST | CMOS | In-Circuit Debug Clock I/O. |
| | ICSPCLK | TTL/ST | CMOS | ICSP Clock I/O. |
| RA2/ANA2/VREF-/ DAC1REF-/ T0CKI(1)/ CCP3(1)/CWG1IN(1)/ CWG2IN(1)/CLCIN0(1)/ INT(1) | RA2 | TTL/ST | CMOS | General purpose I/O. |
| | ANA2 | AN | — | ADC Channel A2 input. |
| | VREF- | AN | — | ADC negative voltage reference input. |
| | DAC1REF- | AN | — | Digital-to-Analog Converter negative reference input. |
| | T0CKI | TTL/ST | — | TMR0 Clock input. |
| | CCP3 | TTL/ST | CMOS | Capture/Compare/PWM 3 input. |
| | CWG1IN | TTL/ST | — | Complementary Waveform Generator 1 input. |
| | CWG2IN | TTL/ST | — | Complementary Waveform Generator 2 input. |
| | CLCIN0 | TTL/ST | — | Configurable Logic Cell 0 input. |
| | INT | TTL/ST | — | External interrupt input. |
| RA3/$\overline{MCLR}$/VPP | RA3 | TTL/ST | CMOS | General purpose I/O. |
| | $\overline{MCLR}$ | TTL/ST | — | Master Clear with internal pull-up. |
| | VPP | HV | — | Programming voltage. |
| RA4/ANA4/T1G(1)/T3G(1)/ T5G(1)/SOSCO/CCP4(1)/ CLKOUT/OSC2 | RA4 | TTL/ST | CMOS | General purpose I/O. |
| | ANA4 | AN | — | ADC Channel A4 input. |
| | T1G | TTL/ST | — | TMR1 gate input. |
| | T3G | TTL/ST | — | TMR3 gate input. |
| | T5G | TTL/ST | — | TMR5 gate input. |
| | SOSCO | — | XTAL | Secondary Oscillator connection. |
| | CCP4 | TTL/ST | CMOS | Capture/Compare/PWM 4 input. |
| | CLKOUT | — | CMOS | FOSC/4 output. |
| | OSC2 | — | XTAL | Crystal/Resonator (LP, XT, HS modes). |

**Legend:** AN = Analog input or output CMOS= CMOS compatible input or output OD = Open-Drain
TTL = TTL compatible input ST = Schmitt Trigger input with CMOS levels $I^2C$ = Schmitt Trigger input with $I^2C$
HV = High Voltage XTAL = Crystal levels

**Note 1:** Default peripheral input. Input can be moved to any other pin with the PPS input selection registers. See Register 13-2.

**2:** All pin outputs default to PORT latch data. Any pin can be selected as a digital peripheral output with the PPS output selection registers. See Register 13-2.

**3:** These $I^2C$ functions are bidirectional. The output pin selections must be the same as the input pin selections.

## 3.0 ENHANCED MID-RANGE CPU

This family of devices contains an enhanced mid-range 8-bit CPU core. The CPU has 48 instructions. Interrupt capability includes automatic context saving. The hardware stack is 16-levels deep and has Overflow and Underflow Reset capability. Direct, Indirect, and Relative Addressing modes are available. Two File Select Registers (FSRs) provide the ability to read program and data memory.

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

**FIGURE 3-1: CORE BLOCK DIAGRAM**

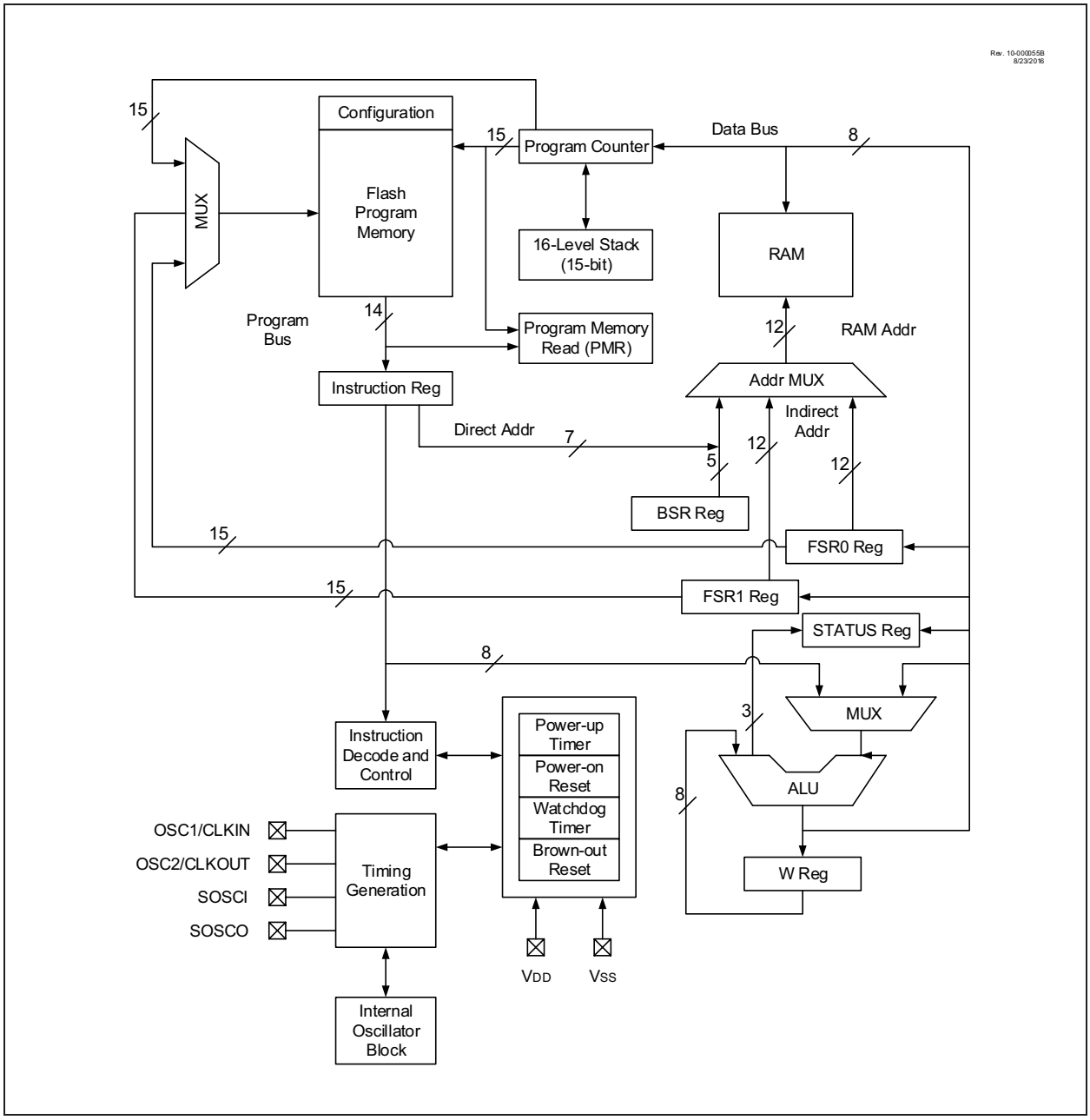


**TABLE 4-4: SPECIAL FUNCTION REGISTER SUMMARY BANKS 0-31 (CONTINUED)**

| Address | Name | PIC16(L)F18326 | PIC16(L)F18346 | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Value on: POR, BOR | Value on all other Resets |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Bank 1** | | | | | | | | | | | | | |
| | **CPU CORE REGISTERS; see Table 4-2 for specifics** | | | | | | | | | | | | |
| 08Ch | TRISA | | | — | — | TRISA5 | TRISA4 | — | TRISA2 | TRISA1 | TRISA0 | --11 -111 | --11 -111 |
| 08Dh | TRISB | X | — | Unimplemented | | | | | | | | — | — |
| | | — | X | TRISB7 | TRISB6 | TRISB5 | TRISB4 | — | — | — | — | 1111 ---- | 1111 ---- |
| 08Eh | TRISC | X | — | — | — | TRISC5 | TRISC4 | TRISC3 | TRISC2 | TRISC1 | TRISC0 | --11 1111 | --11 1111 |
| | | — | X | TRISC7 | TRISC6 | TRISC5 | TRISC4 | TRISC3 | TRISC2 | TRISC1 | TRISC0 | 1111 1111 | 1111 1111 |
| 08Fh | — | — | | Unimplemented | | | | | | | | — | — |
| 090h | PIE0 | | | — | — | TMR0IE | IOCIE | — | — | — | INTE | --00 ---0 | --00 ---0 |
| 091h | PIE1 | | | TMR1GIE | ADIE | RCIE | TXIE | SSP1IE | BCL1IE | TMR2IE | TMR1IE | 0000 0000 | 0000 0000 |
| 092h | PIE2 | | | TMR6IE | C2IE | C1IE | NVMIE | SSP2IE | BCL2IE | TMR4IE | NCO1IE | 0000 0000 | 0000 0000 |
| 093h | PIE3 | | | OSFIE | CSWIE | TMR3GIE | TMR3IE | CLC4IE | CLC3IE | CLC2IE | CLC1IE | 0000 0000 | 0000 0000 |
| 094h | PIE4 | | | CWG2IE | CWG1IE | TMR5GIE | TMR5IE | CCP4IE | CCP3IE | CCP2IE | CCP1IE | 0000 0000 | 0000 0000 |
| 095h | — | — | | Unimplemented | | | | | | | | — | — |
| 096h | — | — | | Unimplemented | | | | | | | | — | — |
| 097h | WDTCON | | | — | — | WDTPS<4:0> | | | | | SWDTEN | --01 0110 | --01 0110 |
| 098h | — | — | | Unimplemented | | | | | | | | — | — |
| 099h | — | — | | Unimplemented | | | | | | | | — | — |
| 09Ah | — | — | | Unimplemented | | | | | | | | — | — |
| 09Bh | ADRESL | | | ADRES<7:0> | | | | | | | | xxxx xxxx | uuuu uuuu |
| 09Ch | ADRESH | | | ADRES<9:8> | | | | | | | | xxxx xxxx | uuuu uuuu |
| 09Dh | ADCON0 | | | CHS<5:0> | | | | | | GO/$\overline{\text{DONE}}$ | ADON | 0000 0000 | 0000 0000 |
| 09Eh | ADCON1 | | | ADFM | ADCS<2:0> | | | — | ADNREF | ADPREF<1:0> | | 0000 -000 | 0000 -000 |
| 09Fh | ADACT | | | — | — | — | ADACT<4:0> | | | | | ---0 0000 | ---0 0000 |

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

**Note 1:** Only on PIC16F18326/18346.

**2:** Register accessible from both User and ICD Debugger.

**REGISTER 8-11: PIR4: PERIPHERAL INTERRUPT REQUEST REGISTER 4**

| R/W/HS-0/0 | R/W/HS-0/0 | R/W/HS-0/0 | R/W/HS-0/0 | R/W/HS-0/0 | R/W/HS-0/0 | R/W/HS-0/0 | R/W/HS-0/0 |
|---|---|---|---|---|---|---|---|
| CWG2IF | CWG1IF | TMR5GIF | TMR5IF | CCP4IF | CCP3IF | CCP2IF | CCP1IF |
| bit 7 | | | | | | | bit 0 |

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

bit 7 **CWG2IF:** CWG 2 Interrupt Flag bit
1 = CWG2 has gone into shutdown
0 = CWG2 is operating normally, or interrupt cleared

bit 6 **CWG1IF:** CWG1 Interrupt Flag bit
1 = CWG1 has gone into shutdown
0 = CWG1 is operating normally, or interrupt cleared

bit 5 **TMR5GIF:** Timer5 Gate Interrupt Flag bit
1 = The TMR5 gate has gone inactive (the gate is closed).
0 = The TMR5 gate has not gone inactive.

bit 4 **TMR5IF:** Timer5 Overflow Interrupt Flag bit
1 = TMR5 overflow occurred (must be cleared in software)
0 = No TMR5 overflow occurred

bit 3 **CCP4IF:** CCP4 Interrupt Flag bit

| Value | CCPM 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 2 **CCP3IF:** CCP3 Interrupt Flag bit

| Value | CCPM 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 |

## 12.7 Register Definitions: PORTC

**REGISTER 12-17: PORTC: PORTC REGISTER**

| R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u |
|---|---|---|---|---|---|---|---|
| RC7(1) | RC6(1) | RC5 | RC4 | RC3 | RC2 | RC1 | RC0 |
| bit 7 | | | | | | | bit 0 |

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

bit 7-6 **RC<7:6>**: PORTC I/O Value bits(1,2)
1 = Port pin is ≥ VIH
0 = Port pin is ≤ VIL

bit 5-0 **RC<5:0>**: PORTC General Purpose I/O Pin bits(2)
1 = Port pin is ≥ VIH
0 = Port pin is ≤ VIL

**Note 1:** PIC16(L)F18346 only; otherwise read as '0'.
**2:** Writes to PORTC are actually written to corresponding LATC register. Reads from PORTC register is return of actual I/O pin values.

**REGISTER 12-18: TRISC: PORTC TRI-STATE REGISTER**

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

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

bit 7-6 **TRISC<7:6>**: PORTC Tri-State Control bits(1)
1 = PORTC pin configured as an input (tri-stated)
0 = PORTC pin configured as an output

bit 5-0 **TRISC<5:0>:** PORTC Tri-State Control bits
1 = PORTC pin configured as an input (tri-stated)
0 = PORTC pin configured as an output

**Note 1:** PIC16(L)F18346 only; otherwise read as '0'.

## 20.0 COMPLEMENTARY WAVEFORM GENERATOR (CWG) MODULE

The Complementary Waveform Generator (CWGx) produces complementary waveforms with dead-band delay from a selection of input sources.

The CWGx module has the following features:

- Selectable dead-band clock source control
- Selectable input sources
- Output enable control
- Output polarity control
- Dead-band control with independent 6-bit rising and falling edge dead-band counters
- Auto-shutdown control with:
  - Selectable shutdown sources
  - Auto-restart enable
  - Auto-shutdown pin override control

## 20.1 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 **Section 20.6 "Dead-Band Control"**.

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 **Section 20.7 "Auto-Shutdown Control"**.

## 20.2 Operating Modes

The CWGx module can operate in six different modes, as specified by the MODE<2:0> bits of the CWGxCON0 register:

- 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 data input, and provide up to four outputs as described in the following sections.

All modes include auto-shutdown control as described in **Section 20.11 "Register Definitions: CWG Control"**

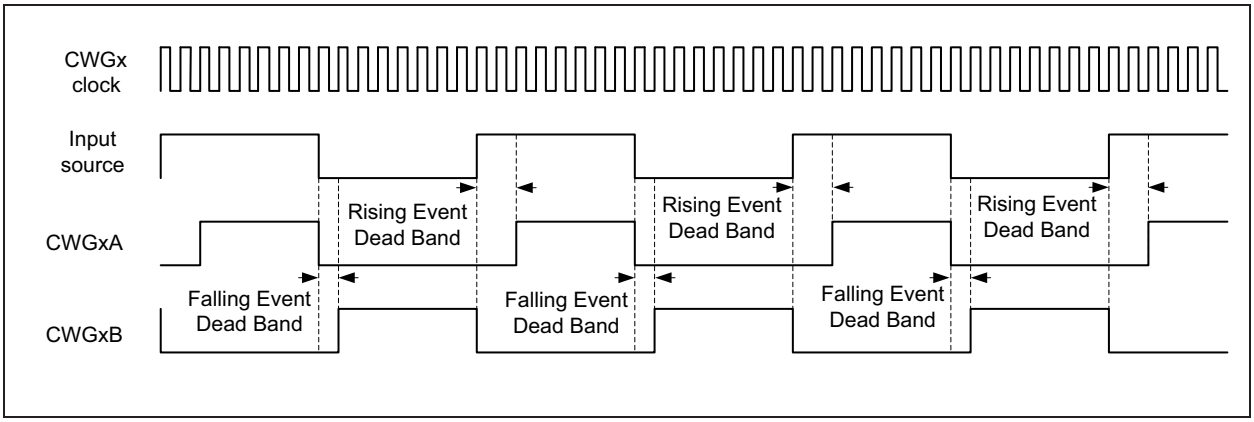
| | |
|---|---|
| **Note:** | Except as noted for Full-bridge mode (**Section 20.2.4 "Full-Bridge Modes"**), mode changes may only be performed while EN = 0 (Register 20-1). |

### 20.2.1 HALF-BRIDGE MODE

In Half-Bridge mode, two output signals are generated as true and inverted versions of the input as illustrated in Figure 20-1. A non-overlap (dead-band) time is inserted between the two outputs as described in **Section 20.6 "Dead-Band Control"**. Steering modes are not used in Half-Bridge mode.

The unused outputs, CWGxC and CWGxD, drive similar signals with polarity independently controlled by POLC and POLD, respectively.

**FIGURE 20-1: CWGx HALF-BRIDGE MODE OPERATION**



## 21.0 CONFIGURABLE LOGIC CELL (CLC)

The Configurable Logic Cell (CLCx) provides programmable logic that operates outside the speed limitations of software execution. The logic cell takes up to 36 input signals and, through the use of configurable gates, reduces the 36 inputs to four logic lines that drive one of eight selectable single-output logic functions.

Input sources are a combination of the following:

- I/O pins
- Internal clocks
- Peripherals
- Register bits

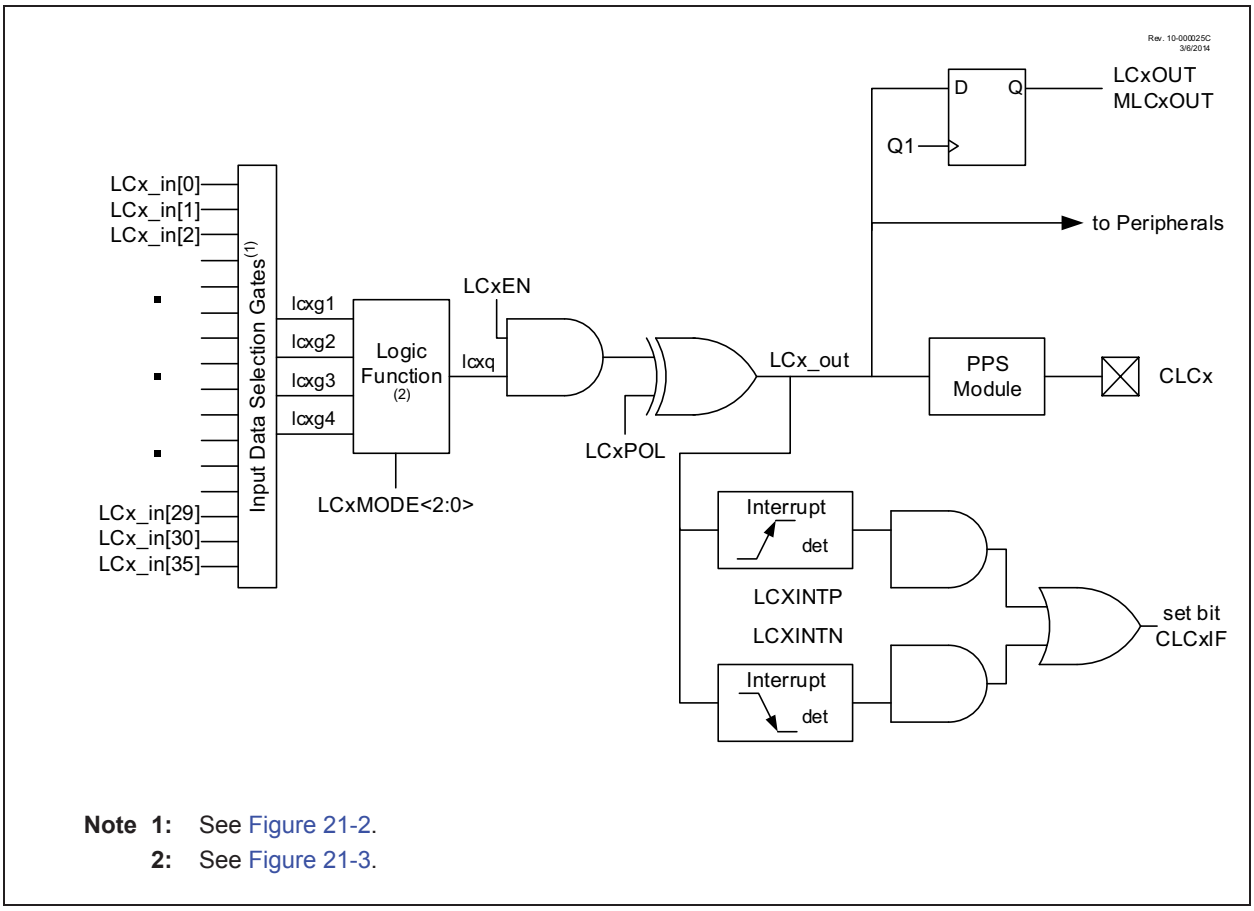
The output can be directed internally to peripherals and to an output pin.

Refer to Figure 21-1 for a simplified diagram showing signal flow through the CLCx.

Possible configurations include:

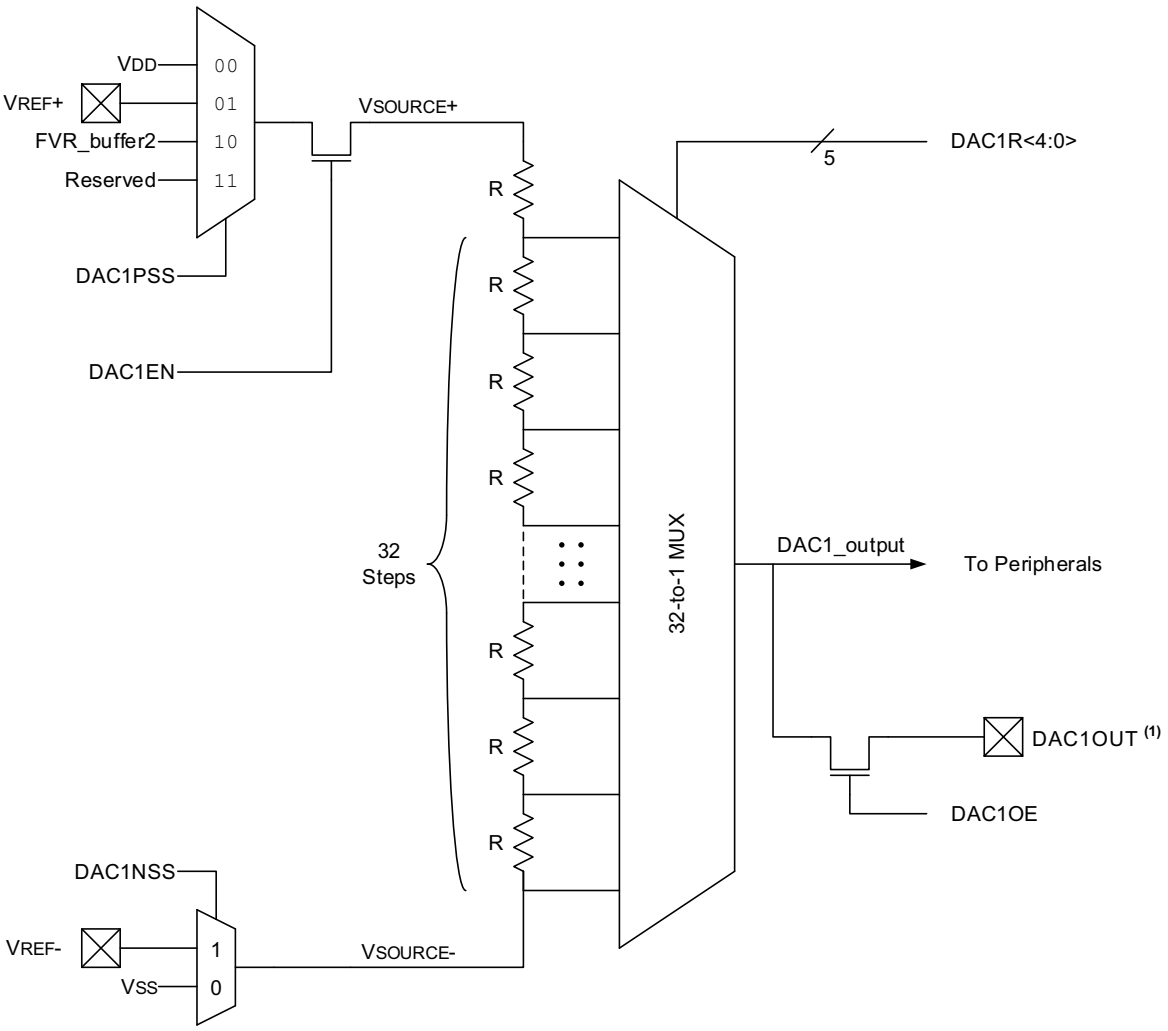
- Combinatorial Logic
  - AND
  - NAND
  - AND-OR
  - AND-OR-INVERT
  - OR-XOR
  - OR-XNOR
- Latches
  - S-R
  - Clocked D with Set and Reset
  - Transparent D with Set and Reset
  - Clocked J-K with Reset

**FIGURE 21-1: CLCx SIMPLIFIED BLOCK DIAGRAM**



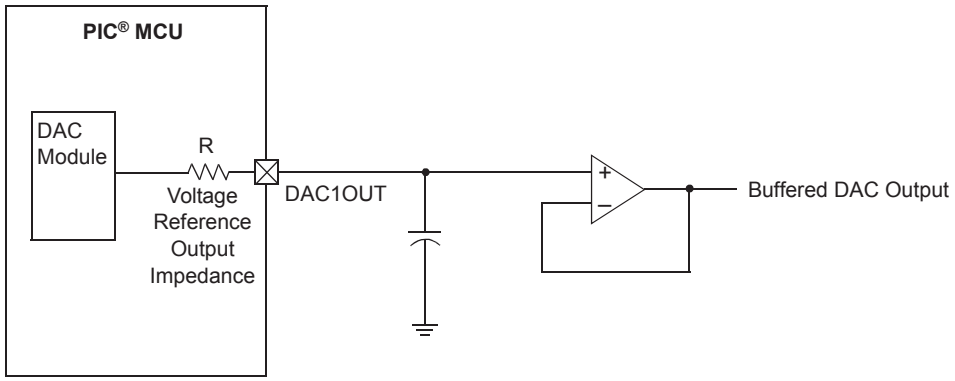

**Note 1:** See Figure 21-2.
**2:** See Figure 21-3.

**FIGURE 24-1: DIGITAL-TO-ANALOG CONVERTER BLOCK DIAGRAM**




**Note 1:** The unbuffered DAC1_output is provided on the DAC1OUT pin(s).

**FIGURE 24-2: VOLTAGE REFERENCE OUTPUT BUFFER EXAMPLE**



## 25.5 Carrier Source Polarity Select

The signal provided from any selected input source for the carrier high and carrier low signals can be inverted. Inverting the signal for the carrier high source is enabled by setting the MDCHPOL bit of the MDCARH register. Inverting the signal for the carrier low source is enabled by setting the MDCLPOL bit of the MDCARL register.

## 25.6 Programmable Modulator Data

The MDBIT of the MDCON register can be selected as the source for the modulator signal. This gives the user the ability to program the value used for modulation.

## 25.7 Modulated Output Polarity

The modulated output signal provided on the DSM pin can also be inverted. Inverting the modulated output signal is enabled by setting the MDOPOL bit of the MDCON register.

## 25.8 Slew Rate Control

The slew rate limitation on the output port pin can be disabled. The slew rate limitation can be removed by clearing the SLR bit of the SLRCON register associated with that pin. For example, clearing the slew rate limitation for pin RA5 would require clearing the SLRA5 bit of the SLRCONA register.

## 25.9 Operation in Sleep Mode

The DSM module is not affected by Sleep mode. The DSM can still operate during Sleep, if the Carrier and Modulator input sources are also still operable during Sleep.

## 25.10 Effects of a Reset

Upon any device Reset, the DSM module is disabled. The user's firmware is responsible for initializing the module before enabling the output. The registers are reset to their default values.

## 27.0 TIMER1/3/5 MODULE WITH GATE CONTROL

Timer1/3/5 modules are 16-bit timers/counters, each with the following features:

- 16-bit timer/counter register pair (TMR1H:TMR1L)
- 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)
- Time base for the Capture/Compare function with the CCP modules
- Auto-Conversion Trigger (with CCP)
- Selectable Gate Source Polarity
- Gate Toggle mode
- Gate Single-Pulse mode
- Gate Value Status
- Gate Event Interrupt

Figure 27-1 is a block diagram of the Timer1 module.

| | |
|---|---|
| **Note 1:** | In devices with more than one Timer module, it is very important to pay close attention to the register names used. A number placed after the module acronym is used to distinguish between separate modules. For example, the T1CON and T3CON control the same operational aspects of two completely different Timer modules. |
| **2:** | Throughout this section, generic references to Timer1 module in any of its operating modes may be interpreted as being equally applicable to Timerx module. Register names, module signals, I/O pins and bit names may use the generic designator ‘x’ to indicate the use of a numeral to distinguish a particular module, when required. |

## 27.6 Timer1 Interrupt

The Timer1 register pair (TMR1H:TMR1L) increments to FFFFh and rolls over to 0000h. When Timer1 rolls over, the Timer1 interrupt flag bit of the PIR1 register is set. To enable the interrupt on rollover, you must set these bits:

- TMR1ON bit of the T1CON register
- TMR1IE bit of the PIE1 register
- PEIE bit of the INTCON register
- GIE bit of the INTCON register

The interrupt is cleared by clearing the TMR1IF bit in the Interrupt Service Routine.

| | |
|---|---|
| **Note:** | The TMR1H:TMR1L register pair and the TMR1IF bit should be cleared before enabling interrupts. |

### 27.6.1 TIMER1 GATE EVENT INTERRUPT

When Timer1 Gate Event Interrupt is enabled, it is possible to generate an interrupt upon the completion of a gate event. When the falling edge of T1GVAL occurs, the TMR1GIF flag bit in the PIR1 register will be set. If the TMR1GIE bit in the PIE1 register is set, then an interrupt will be recognized.

The TMR1GIF flag bit operates even when the Timer1 gate is not enabled (TMR1GE bit is cleared).

## 27.7 Timer1 Operation During Sleep

Timer1 can only operate during Sleep when setup in Asynchronous mode. In this mode, an external crystal or clock source can be used to increment the timer. To set up the timer to wake the device:

- TMR1ON bit of the T1CON register must be set
- TMR1IE bit of the PIE1 register must be set
- PEIE bit of the INTCON register must be set
- $\overline{\text{T1SYNC}}$ bit of the T1CON register must be set
- TMR1CS bits of the T1CON register must be configured
- T1SOSC bit of the T1CON register must be configured

The device will wake-up on an overflow and execute the next instructions. If the GIE bit of the INTCON register is set, the device will call the Interrupt Service Routine.

Secondary oscillator will continue to operate in Sleep regardless of the $\overline{\text{T1SYNC}}$ bit setting.

## 27.8 CCP Capture/Compare Time Base

The CCP modules use the TMR1H:TMR1L register pair as the time base when operating in Capture or Compare mode.

In Capture mode, the value in the TMR1H:TMR1L register pair is copied into the CCPR1H:CCPR1L register pair on a configured event.

In Compare mode, an event is triggered when the value CCPR1H:CCPR1L register pair matches the value in the TMR1H:TMR1L register pair. This event can be an Auto-conversion Trigger.

For more information, see **Section 29.0 "Capture/Compare/PWM Modules"**.

## 27.9 CCP Auto-Conversion Trigger

When any of the CCP's are configured to trigger an auto-conversion, the trigger will clear the TMR1H:TMR1L register pair. This auto-conversion does not cause a Timer1 interrupt. The CCP module may still be configured to generate a CCP interrupt.

In this mode of operation, the CCPR1H:CCPR1L register pair becomes the period register for Timer1.

Timer1 should be synchronized and FOSC/4 should be selected as the clock source in order to utilize the Auto-conversion Trigger. Asynchronous operation of Timer1 can cause an Auto-conversion Trigger to be missed.

In the event that a write to TMR1H or TMR1L coincides with an Auto-conversion Trigger from the CCP, the write will take precedence.

For more information, see **Section 29.3.3 "Auto-Conversion Trigger"**.

**TABLE 27-5: SUMMARY OF REGISTERS ASSOCIATED WITH TIMER1/3/5**

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Register on Page |
|---|---|---|---|---|---|---|---|---|---|
| TRISA | — | — | TRISA5 | TRISA4 | —(2) | TRISA2 | TRISA1 | TRISA0 | 143 |
| ANSELA | — | — | ANSA5 | ANSA4 | — | ANSA2 | ANSA1 | ANSA0 | 144 |
| TRISB(1) | TRISB7 | TRISB6 | TRISB5 | TRISB4 | — | — | — | — | 149 |
| ANSELB(1) | ANSB7 | ANSB6 | ANSB5 | ANSB4 | — | — | — | — | 150 |
| TRISC | TRISC7(1) | TRISC6(1) | TRISC5 | TRISC4 | TRISC3 | TRISC2 | TRISC1 | TRISC0 | 155 |
| ANSELC | ANSC7(1) | ANSC6(1) | ANSC5 | ANSC4 | ANSC3 | ANSC2 | ANSC1 | ANSC0 | 157 |
| INTCON | GIE | PEIE | — | — | — | — | — | INTEDG | 100 |
| PIR1 | TMR1GIF | ADIF | RCIF | TXIF | SSP1IF | BCL1IF | TMR2IF | TMR1IF | 107 |
| PIE1 | TMR1GIE | ADIE | RCIE | TXIE | SSP1IE | BCL1IE | TMR2IE | TMR1IE | 102 |
| PIR3 | OSFIF | CSWIF | TMR3GIF | TMR3IF | CLC4IF | CLC3IF | CLC2IF | CLC1IF | 109 |
| PIE3 | OSFIE | CSWIE | TMR3GIE | TMR3IE | CLC4IE | CLC3IE | CLC2IE | CLC1IE | 104 |
| PIR4 | CWG2IF | CWG1IF | TMR5GIF | TMR5IF | CCP4IF | CCP3IF | CCP2IF | CCP1IF | 110 |
| PIE4 | CWG2IE | CWG1IE | TMR5GIE | TMR5IE | CCP4IE | CCP3IE | CCP2IE | CCP1IE | 105 |
| T1CON | TMR1CS<1:0> | | T1CKPS<1:0> | | T1SOSC | $\overline{\text{T1SYNC}}$ | — | TMR1ON | 292 |
| T1GCON | TMR1GE | T1GPOL | T1GTM | T1GSPM | T1GGO/$\overline{\text{DONE}}$ | T1GVAL | T1GSS<1:0> | | 293 |
| TMR1L | TMR1L<7:0> | | | | | | | | 294 |
| TMR1H | TMR1H<7:0> | | | | | | | | 294 |
| T1CKIPPS | — | — | — | T1CKIPPS<4:0> | | | | | 162 |
| T1GPPS | — | — | — | T1GPPS<4:0> | | | | | 162 |
| T3CON | TMR3CS<1:0> | | T3CKPS<1:0> | | T3SOSC | $\overline{\text{T3SYNC}}$ | — | TMR3ON | 292 |
| T3GCON | TMR3GE | T3GPOL | T3GTM | T3GSPM | T3GGO/$\overline{\text{DONE}}$ | T3GVAL | T3GSS<1:0> | | 293 |
| TMR3L | TMR3L<7:0> | | | | | | | | 294 |
| TMR3H | TMR3H<7:0> | | | | | | | | 294 |
| T3CKIPPS | — | — | — | T3CKIPPS<4:0> | | | | | 162 |
| T3GPPS | — | — | — | T3GPPS<4:0> | | | | | 162 |
| T5CON | TMR5CS<1:0> | | T5CKPS<1:0> | | T5SOSC | $\overline{\text{T5SYNC}}$ | — | TMR5ON | 292 |
| T5GCON | TMR5GE | T5GPOL | T5GTM | T5GSPM | T5GGO/$\overline{\text{DONE}}$ | T5GVAL | T5GSS<1:0> | | 293 |
| TMR5L | TMR5L<7:0> | | | | | | | | 294 |
| TMR5H | TMR5H<7:0> | | | | | | | | 294 |
| T5CKIPPS | — | — | — | T5CKIPPS<4:0> | | | | | 162 |
| T5GPPS | — | — | — | T5GPPS<4:0> | | | | | 162 |
| T0CON0 | T0EN | — | T0OUT | T016BIT | T0OUTPS<3:0> | | | | 280 |
| CMxCON0 | CxON | CxOUT | — | CxPOL | — | CxSP | CxHYS | CxSYNC | 190 |
| CCPTMRS | C4TSEL<1:0> | | C3TSEL<1:0> | | C2TSEL<1:0> | | C1TSEL<1:0> | | 311 |
| CCPxCON | CCPxEN | — | CCPxOUT | CCPxFMT | CCPxMODE<3:0> | | | | 308 |
| CLCxSELy | — | — | LCxDyS<5:0> | | | | | | 229 |
| ADACT | — | — | — | ADACT<4:0> | | | | | 246 |

**Legend:** — = unimplemented location, read as '0'. Shaded cells are not used by the Timer1 module.

**Note 1:** PIC16(L)F18346 only.

**2:** Unimplemented, read as '1'.

### 31.3.5 RECEIVING A BREAK CHARACTER

The Enhanced EUSART1 module can receive a Break character in two ways.

The first method to detect a Break character uses the FERR bit of the RC1STA register and the received data as indicated by RC1REG. The Baud Rate Generator is assumed to have been initialized to the expected baud rate.

A Break character has been received when:

- RCIF bit is set
- FERR bit is set
- RC1REG = 00h

The second method uses the Auto-Wake-up feature described in **Section 31.3.3 "Auto-Wake-up on Break"**. By enabling this feature, the EUSART1 will sample the next two transitions on RX/DT, cause an RCIF interrupt, and receive the next data byte followed by another interrupt.

Note that following a Break character, the user will typically want to enable the Auto-Baud Detect feature. For both methods, the user can set the ABDEN bit of the BAUD1CON register before placing the EUSART1 in Sleep mode.

**FIGURE 31-9: SEND BREAK CHARACTER SEQUENCE**

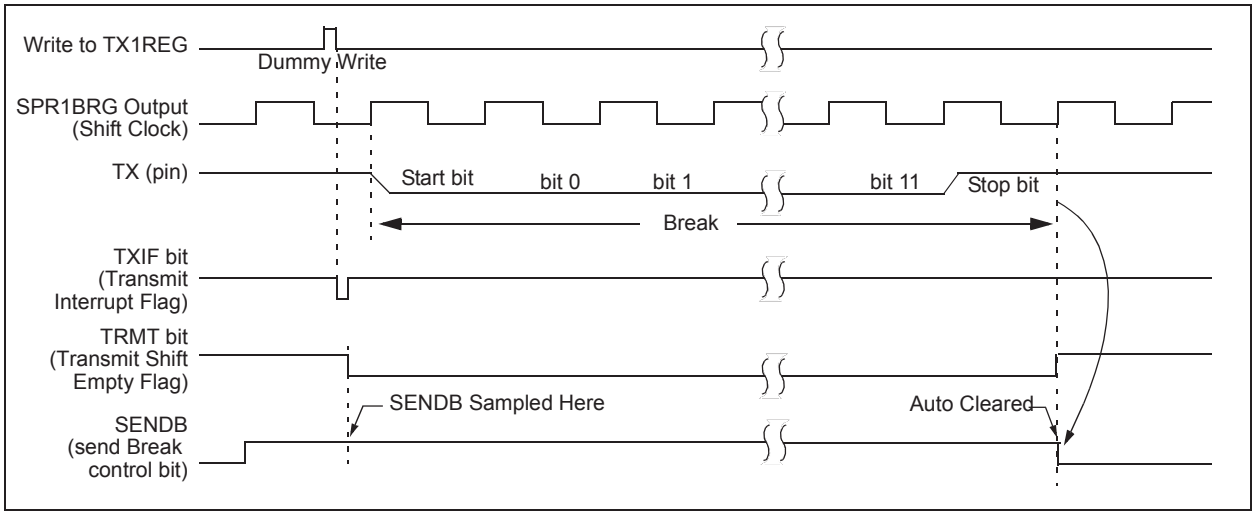


**TABLE 31-3: BAUD RATE FORMULAS**

| Configuration Bits | | | BRG/EUSART1 Mode | Baud Rate Formula |
|---|---|---|---|---|
| SYNC | BRG16 | BRGH | | |
| 0 | 0 | 0 | 8-bit/Asynchronous | FOSC/[64 (n+1)] |
| 0 | 0 | 1 | 8-bit/Asynchronous | FOSC/[16 (n+1)] |
| 0 | 1 | 0 | 16-bit/Asynchronous | |
| 0 | 1 | 1 | 16-bit/Asynchronous | FOSC/[4 (n+1)] |
| 1 | 0 | x | 8-bit/Synchronous | |
| 1 | 1 | x | 16-bit/Synchronous | |

**Legend:** x = Don't care, n = value of SP1BRGH, SP1BRGL register pair.

**TABLE 31-4: BAUD RATE FOR ASYNCHRONOUS MODES**

| BAUD RATE | SYNC = 0, BRGH = 0, BRG16 = 0 | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | FOSC = 32.000 MHz | | | FOSC = 20.000 MHz | | | FOSC = 18.432 MHz | | | FOSC = 11.0592 MHz | | |
| | Actual Rate | % Error | SPBRG value (decimal) | Actual Rate | % Error | SPBRG value (decimal) | Actual Rate | % Error | SPBRG value (decimal) | Actual Rate | % Error | SPBRG value (decimal) |
| 300 | — | — | — | — | — | — | — | — | — | — | — | — |
| 1200 | — | — | — | 1221 | 1.73 | 255 | 1200 | 0.00 | 239 | 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 | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | FOSC = 8.000 MHz | | | FOSC = 4.000 MHz | | | FOSC = 3.6864 MHz | | | FOSC = 1.000 MHz | | |
| | Actual Rate | % Error | SPBRG value (decimal) | Actual Rate | % Error | SPBRG value (decimal) | Actual Rate | % Error | SPBRG value (decimal) | Actual Rate | % Error | SPBRG value (decimal) |
| 300 | — | — | — | 300 | 0.16 | 207 | 300 | 0.00 | 191 | 300 | 0.16 | 51 |
| 1200 | 1202 | 0.16 | 103 | 1202 | 0.16 | 51 | 1200 | 0.00 | 47 | 1202 | 0.16 | 12 |
| 2400 | 2404 | 0.16 | 51 | 2404 | 0.16 | 25 | 2400 | 0.00 | 23 | — | — | — |
| 9600 | 9615 | 0.16 | 12 | — | — | — | 9600 | 0.00 | 5 | — | — | — |
| 10417 | 10417 | 0.00 | 11 | 10417 | 0.00 | 5 | — | — | — | — | — | — |
| 19.2k | — | — | — | — | — | — | 19.20k | 0.00 | 2 | — | — | — |
| 57.6k | — | — | — | — | — | — | 57.60k | 0.00 | 0 | — | — | — |
| 115.2k | — | — | — | — | — | — | — | — | — | — | — | — |

**CALL** **Call Subroutine**

| | |
|---|---|
| Syntax: | [ *label* ] CALL k |
| Operands: | $0 \le k \le 2047$ |
| 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. The 11-bit immediate address is loaded into PC bits <10:0>. The upper bits of the PC are loaded from PCLATH. CALL is a 2-cycle instruction. |

**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. First, the return address (PC + 1) is pushed onto the return stack. Then, the contents of W is loaded into PC<7:0>, and the contents of PCLATH into PC<14:8>. CALLW is a 2-cycle instruction. |

**CLRF** **Clear f**

| | |
|---|---|
| Syntax: | [ *label* ] CLRF f |
| Operands: | $0 \le f \le 127$ |
| Operation: | 00h → (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 |
| Operation: | 00h → (W)<br>1 → Z |
| Status Affected: | Z |
| Description: | W register is cleared. Zero bit (Z) is set. |

**CLRWDT** **Clear Watchdog Timer**

| | |
|---|---|
| Syntax: | [ *label* ] CLRWDT |
| Operands: | None |
| Operation: | 00h → WDT<br>0 → WDT prescaler,<br>1 → $\overline{TO}$<br>1 → $\overline{PD}$ |
| Status Affected: | $\overline{TO}$, $\overline{PD}$ |
| Description: | CLRWDT instruction resets the Watchdog Timer. It also resets the prescaler of the WDT. Status bits $\overline{TO}$ and $\overline{PD}$ are set. |

**COMF** **Complement f**

| | |
|---|---|
| Syntax: | [ *label* ] COMF f,d |
| Operands: | $0 \le f \le 127$<br>$d \in [0,1]$ |
| Operation: | $(\bar{f}) \rightarrow$ (destination) |
| Status Affected: | Z |
| Description: | The contents of register 'f' are complemented. If 'd' is '0', the result is stored in W. If 'd' is '1', the result is stored back in register 'f'. |

**DECF** **Decrement f**

| | |
|---|---|
| Syntax: | [ *label* ] DECF f,d |
| Operands: | $0 \le f \le 127$<br>$d \in [0,1]$ |
| Operation: | (f) - 1 → (destination) |
| Status Affected: | Z |
| Description: | Decrement register 'f'. If 'd' is '0', the result is stored in the W register. If 'd' is '1', the result is stored back in register 'f'. |

**DECFSZ** **Decrement f, Skip if 0**

| | |
|---|---|
| Syntax: | [ *label* ] DECFSZ f,d |
| Operands: | 0 ≤ f ≤ 127<br>d ∈ [0,1] |
| Operation: | (f) - 1 → (destination);<br>skip if result = 0 |
| Status Affected: | None |
| Description: | The contents of register 'f' are decremented. If 'd' is '0', the result is placed in the W register. If 'd' is '1', the result is placed back in register 'f'.<br>If the result is '1', the next instruction is executed. If the result is '0', then a NOP is executed instead, making it a 2-cycle instruction. |

**GOTO** **Unconditional Branch**

| | |
|---|---|
| Syntax: | [ *label* ] GOTO k |
| Operands: | 0 ≤ k ≤ 2047 |
| Operation: | k → PC<10:0><br>PCLATH<6:3> → PC<14:11> |
| Status Affected: | None |
| Description: | GOTO is an unconditional branch. The 11-bit immediate value is loaded into PC bits <10:0>. The upper bits of PC are loaded from PCLATH<4:3>. GOTO is a 2-cycle instruction. |

**INCF** **Increment f**

| | |
|---|---|
| Syntax: | [ *label* ] INCF f,d |
| Operands: | 0 ≤ f ≤ 127<br>d ∈ [0,1] |
| Operation: | (f) + 1 → (destination) |
| Status Affected: | Z |
| Description: | The contents of register 'f' are incremented. If 'd' is '0', the result is placed in the W register. If 'd' is '1', the result is placed back in register 'f'. |

**INCFSZ** **Increment f, Skip if 0**

| | |
|---|---|
| Syntax: | [ *label* ] INCFSZ f,d |
| Operands: | 0 ≤ f ≤ 127<br>d ∈ [0,1] |
| Operation: | (f) + 1 → (destination),<br>skip if result = 0 |
| Status Affected: | None |
| Description: | The contents of register 'f' are incremented. If 'd' is '0', the result is placed in the W register. If 'd' is '1', the result is placed back in register 'f'.<br>If the result is '1', the next instruction is executed. If the result is '0', a NOP is executed instead, making it a 2-cycle instruction. |

**IORLW** **Inclusive OR literal with W**

| | |
|---|---|
| Syntax: | [ *label* ] IORLW k |
| Operands: | 0 ≤ k ≤ 255 |
| Operation: | (W) .OR. k → (W) |
| Status Affected: | Z |
| Description: | The contents of the W register are OR'ed with the 8-bit literal 'k'. The result is placed in the W register. |

**IORWF** **Inclusive OR W with f**

| | |
|---|---|
| Syntax: | [ *label* ] IORWF f,d |
| Operands: | 0 ≤ f ≤ 127<br>d ∈ [0,1] |
| Operation: | (W) .OR. (f) → (destination) |
| Status Affected: | Z |
| Description: | Inclusive OR the W register with register 'f'. If 'd' is '0', the result is placed in the W register. If 'd' is '1', the result is placed back in register 'f'. |

| TRIS | Load TRIS Register with W |
|---|---|
| Syntax: | [ *label* ] TRIS f |
| Operands: | $5 \le f \le 7$ |
| 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: | $0 \le k \le 255$ |
| Operation: | (W) .XOR. k → (W) |
| Status Affected: | Z |
| Description: | The contents of the W register are XOR’ed with the 8-bit literal ‘k’. The result is placed in the W register. |

| XORWF | Exclusive OR W with f |
|---|---|
| Syntax: | [ *label* ] XORWF f,d |
| Operands: | $0 \le f \le 127$<br>$d \in [0,1]$ |
| Operation: | (W) .XOR. (f) → (destination) |
| Status Affected: | Z |
| Description: | Exclusive OR the contents of the W register with register ‘f’. If ‘d’ is ‘0’, the result is stored in the W register. If ‘d’ is ‘1’, the result is stored back in register ‘f’. |

**TABLE 35-8: OSCILLATOR PARAMETERS(1)**

| Standard Operating Conditions (unless otherwise stated) | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Param. No.** | **Sym.** | **Characteristic** | **Min.** | **Typ.†** | **Max.** | **Units** | **Conditions** |
| OS20 | FHFOSC | Precision Calibrated HFINTOSC Frequency | 3.92 | 4 | 4.08 | MHz | 25°C |
| OS20 | FHFOSC | Precision Calibrated HFINTOSC Frequency | — | 4<br>8<br>12<br>16<br>32 | — | MHz | -40°C to 125°C (2) |
| OS21 | FHFOSCLP | Low-Power Optimized HFINTOSC Frequency | 0.93<br>1.86 | 1<br>2 | 1.07<br>2.14 | MHz<br>MHz | |
| OS23 | FLFOSC | Internal LFINTOSC Frequency | — | 31 | — | kHz | |
| OS24 | THFOSCST | HFINTOSC Wake-up from Sleep Start-up Time | — | 11<br>50 | 20<br>— | µs<br>µs | VREGPM = 0<br>VREGPM = 1 |
| OS26 | TLFOSCST | LFINTOSC Wake-up from Sleep Start-up Time | — | 0.2 | — | ms | |

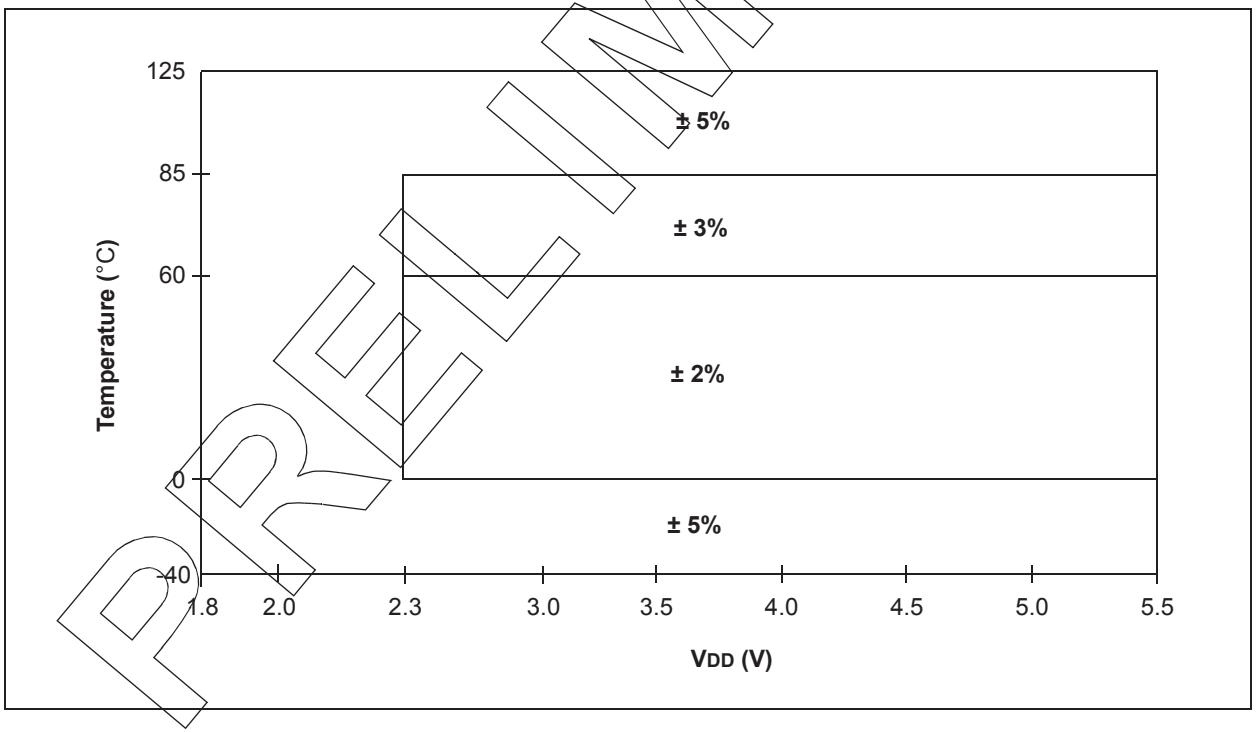
* These parameters are characterized but not tested.

† Data in "Typ." column is at 3.0V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

**Note 1:** To ensure these oscillator frequency tolerances, VDD and VSS must be capacitively decoupled as close to the device as possible. 0.1 µF and 0.01 µF values in parallel are recommended.

**2:** See Figure 35-6.

**FIGURE 35-6: PRECISION CALIBRATED HFINTOSC FREQUENCY ACCURACY OVER DEVICE VDD AND TEMPERATURE**





## 37.0 DEVELOPMENT SUPPORT

The PIC® microcontrollers (MCU) and dsPIC® digital signal controllers (DSC) are supported with a full range of software and hardware development tools:

- Integrated Development Environment
  - MPLAB® X IDE Software
  - MPLAB Xpress IDE Software
  - Microchip Code Configurator (MCC)
- Compilers/Assemblers/Linkers
  - MPLAB XC Compiler
  - MPASM™ Assembler
  - MPLINK™ Object Linker/ MPLIB™ Object Librarian
  - MPLAB Assembler/Linker/Librarian for Various Device Families
- Simulators
  - MPLAB X SIM Software Simulator
- Emulators
  - MPLAB REAL ICE™ In-Circuit Emulator
- In-Circuit Debuggers/Programmers
  - MPLAB ICD 3
  - PICkit™ 3
- Device Programmers
  - MPLAB PM3 Device Programmer
- Low-Cost Demonstration/Development Boards, Evaluation Kits and Starter Kits
- Third-party development tools

### 37.1 MPLAB X Integrated Development Environment Software

The MPLAB X IDE is a single, unified graphical user interface for Microchip and third-party software, and hardware development tool that runs on Windows®, Linux and Mac OS® X. Based on the NetBeans IDE, MPLAB X IDE is an entirely new IDE with a host of free software components and plug-ins for high-performance application development and debugging. Moving between tools and upgrading from software simulators to hardware debugging and programming tools is simple with the seamless user interface.

With complete project management, visual call graphs, a configurable watch window and a feature-rich editor that includes code completion and context menus, MPLAB X IDE is flexible and friendly enough for new users. With the ability to support multiple tools on multiple projects with simultaneous debugging, MPLAB X IDE is also suitable for the needs of experienced users.

Feature-Rich Editor:

- Color syntax highlighting
- Smart code completion makes suggestions and provides hints as you type
- Automatic code formatting based on user-defined rules
- Live parsing

User-Friendly, Customizable Interface:

- Fully customizable interface: toolbars, toolbar buttons, windows, window placement, etc.
- Call graph window

Project-Based Workspaces:

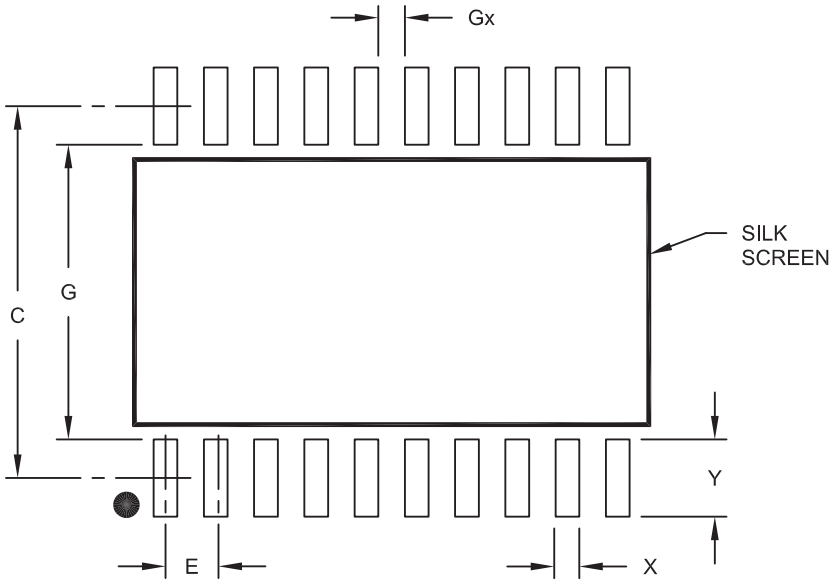
- Multiple projects
- Multiple tools
- Multiple configurations
- Simultaneous debugging sessions

File History and Bug Tracking:

- Local file history feature
- Built-in support for Bugzilla issue tracker

20-Lead Plastic Small Outline (SO) - Wide, 7.50 mm Body [SOIC]

**Note:** For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging




RECOMMENDED LAND PATTERN

| Units | | MILLIMETERS | | |
|---|---|---|---|---|
| Dimension Limits | | MIN | NOM | MAX |
| Contact Pitch | E | | 1.27 BSC | |
| Contact Pad Spacing | C | | 9.40 | |
| Contact Pad Width (X20) | X | | | 0.60 |
| Contact Pad Length (X20) | Y | | | 1.95 |
| Distance Between Pads | Gx | 0.67 | | |
| Distance Between Pads | G | 7.45 | | |

Notes:

1. Dimensioning and tolerancing per ASME Y14.5M

BSC: Basic Dimension. Theoretically exact value shown without tolerances.

Microchip Technology Drawing No. C04-2094A