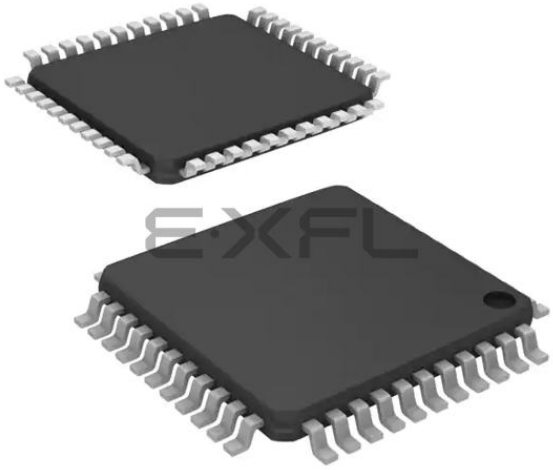


Welcome to E-XFL.COM

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.

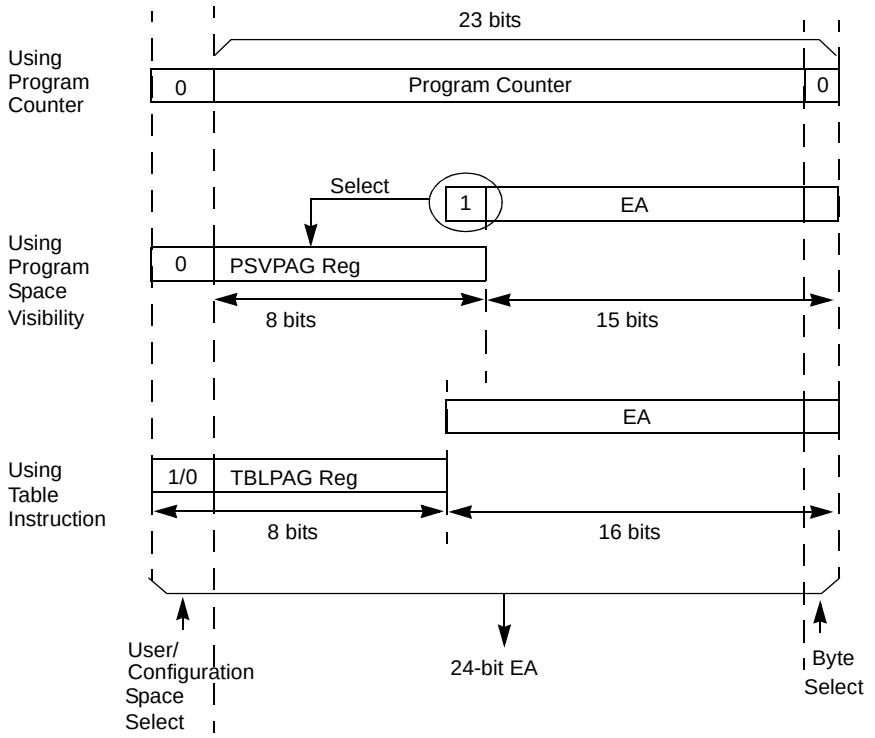
Applications of "Embedded - Microcontrollers"

Details	
Product Status	Active
Core Processor	dsPIC
Core Size	16-Bit
Speed	30 MIPs
Connectivity	I²C, IrDA, LINbus, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, POR, PWM, WDT
Number of I/O	35
Program Memory Size	12KB (4K x 24)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	512 x 8
Voltage - Supply (Vcc/Vdd)	3V ~ 5.5V
Data Converters	A/D 12x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	44-TQFP
Supplier Device Package	44-TQFP (10x10)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/dspic30f2023-30i-pt

TABLE 3-1: PROGRAM SPACE ADDRESS CONSTRUCTION

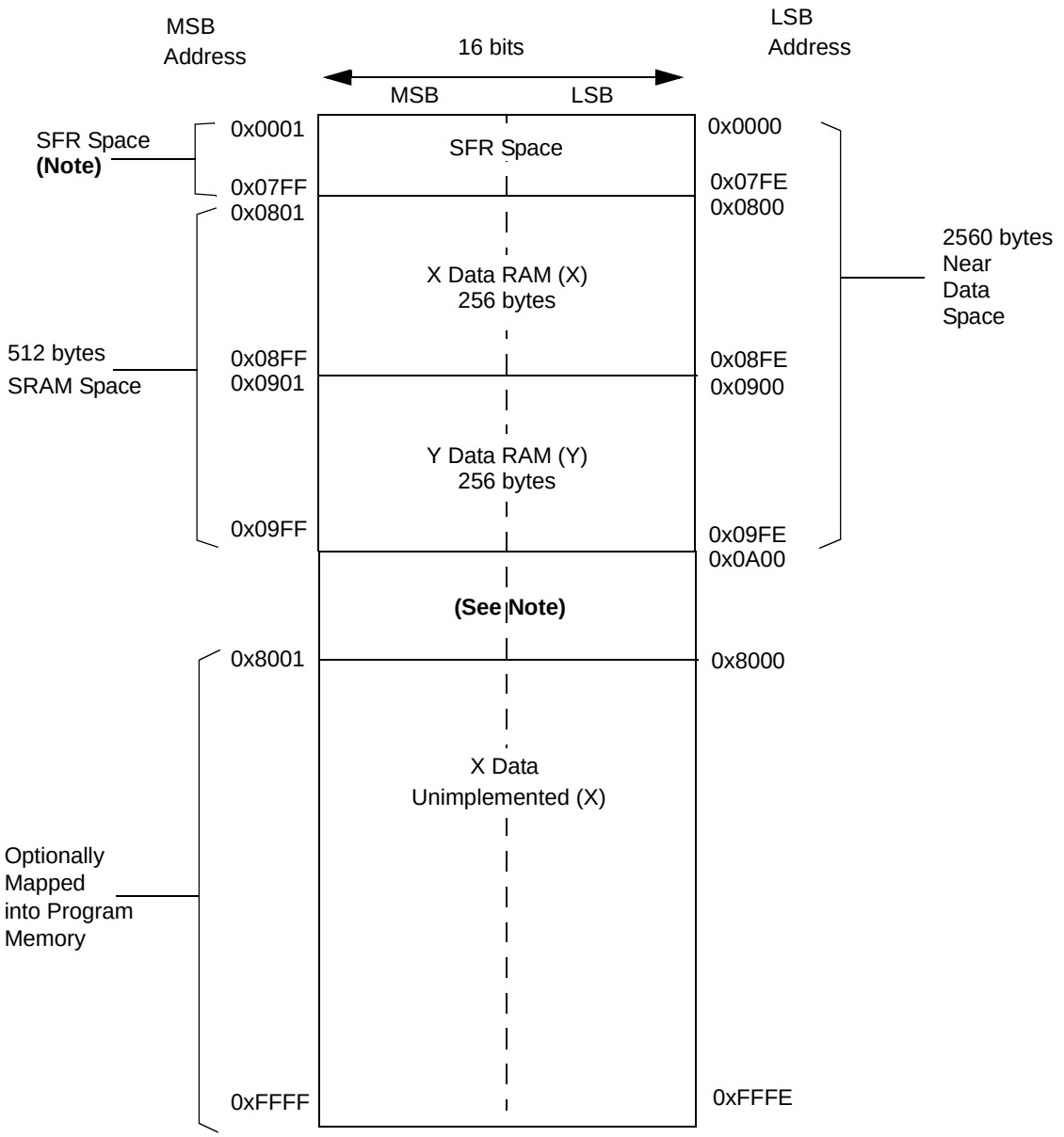
Access Type	Access Space	Program Space Address				
		<23>	<22:16>	<15>	<14:1>	<0>
Instruction Access	User	0	PC<22:1>			0
TBLRD/TBLWT	User (TBLPAG<7> = 0)	TBLPAG<7:0>		Data EA <15:0>		
TBLRD/TBLWT	Configuration (TBLPAG<7> = 1)	TBLPAG<7:0>		Data EA <15:0>		
Program Space Visibility	User	0	PSVPAG<7:0>	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

FIGURE 3-2: DATA ACCESS FROM PROGRAM SPACE ADDRESS GENERATION



Note: Program Space Visibility cannot be used to access bits <23:16> of a word in program memory.

FIGURE 3-6: DATA SPACE MEMORY MAP



Note: Unimplemented SFR or SRAM locations read as '0'.

5.0 INTERRUPTS

Note: This data sheet summarizes features of this group of dsPIC30F devices and is not intended to be a complete reference source. For more information on the CPU, peripherals, register descriptions and general device functionality, refer to the "*dsPIC30F Family Reference Manual*" (DS70046). For more information on the device instruction set and programming, refer to the "*dsPIC30F/33F Programmer's Reference Manual*" (DS70157).

The dsPIC30F1010/202X device has up to 35 interrupt sources and 4 processor exceptions (traps), which must be arbitrated based on a priority scheme.

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

The Interrupt Vector Table and Alternate Interrupt Vector Table (AIVT) are placed near the beginning of program memory (0x000004). The IVT and AIVT are shown in Figure 5-1.

The interrupt controller is responsible for preprocessing the interrupts and processor exceptions, prior to their being presented to the processor core. The peripheral interrupts and traps are enabled, prioritized and controlled using centralized special function registers:

- IFS0<15:0>, IFS1<15:0>, IFS2<15:0>
  All interrupt request flags are maintained in these three registers. The flags are set by their respective peripherals or external signals, and they are cleared via software.
- IEC0<15:0>, IEC1<15:0>, IEC2<15:0>
  All interrupt enable control bits are maintained in these three registers. These control bits are used to individually enable interrupts from the peripherals or external signals.
- IPC0<15:0>... IPC11<7:0>
  The user-assignable priority level associated with each of these interrupts is held centrally in these twelve registers.
- IPL<3:0> The current CPU priority level is explicitly stored in the IPL bits. IPL<3> is present in the CORCON register, whereas IPL<2:0> are present in the STATUS Register (SR) in the processor core.
- INTCON1<15:0>, INTCON2<15:0>
  Global interrupt control functions are derived from these two registers. INTCON1 contains the control and status flags for the processor exceptions. The INTCON2 register controls the external interrupt request signal behavior and the use of the alternate vector table.
- The INTTREG register contains the associated interrupt vector number and the new CPU interrupt priority level, which are latched into vector number (VECNUM<6:0>) and Interrupt level (ILR<3:0>) bit fields in the INTTREG register. The new interrupt priority level is the priority of the pending interrupt.

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

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

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

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

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

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

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

When an interrupt is serviced, the PC is loaded with the address stored in the vector location in Program Memory that corresponds to the interrupt. There are 63 different vectors within the IVT (refer to Figure 5-1). These vectors are contained in locations 0x000004 through 0x0000FE of program memory (refer to Figure 5-1). These locations contain 24-bit addresses, and, in order to preserve robustness, an address error trap will take place should the PC attempt to fetch any of these words during normal execution. This prevents execution of random data as a result of accidentally decrementing a PC into vector space, accidentally mapping a data space address into vector space, or the PC rolling over to 0x000000 after reaching the end of implemented program memory space. Execution of a `GOTO` instruction to this vector space will also generate an address error trap.

REGISTER 5-7: IEC1: INTERRUPT ENABLE CONTROL REGISTER 1

R/W-0	R/W-0	R/W-0	U-0	R/W-0	U-0	U-0	U-0
AC3IE	AC2IE	AC1IE	—	CNIE	—	—	—
bit 15							bit 8

U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	PWM4IE	PWM3IE	PWM2IE	PWM1IE	PSEMIE	INT2IE	INT1IE
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 15 **AC3IE:** Analog Comparator #3 Interrupt Enable bit
1 = Interrupt request enabled
0 = Interrupt request not enabled

bit 14 **AC2IE:** Analog Comparator #2 Interrupt Enable bit
1 = Interrupt request enabled
0 = Interrupt request not enabled

bit 13 **AC1IE:** Analog Comparator #1 Interrupt Enable bit
1 = Interrupt request enabled
0 = Interrupt request not enabled

bit 12 **Unimplemented:** Read as '0'

bit 11 **CNIE:** Input Change Notification Interrupt Enable bit
1 = Interrupt request enabled
0 = Interrupt request not enabled

bit 10-7 **Unimplemented:** Read as '0'

bit 6 **PWM4IE:** Pulse Width Modulation Generator #4 Interrupt Enable bit
1 = Interrupt request enabled
0 = Interrupt request not enabled

bit 5 **PWM3IE:** Pulse Width Modulation Generator #3 Interrupt Enable bit
1 = Interrupt request enabled
0 = Interrupt request not enabled

bit 4 **PWM2IE:** Pulse Width Modulation Generator #2 Interrupt Enable bit
1 = Interrupt request enabled
0 = Interrupt request not enabled

bit 3 **PWM1IE:** Pulse Width Modulation Generator #1 Interrupt Enable bit
1 = Interrupt request enabled
0 = Interrupt request not enabled

bit 2 **PSEMIE:** PWM Special Event Match Interrupt Enable bit
1 = Interrupt request enabled
0 = Interrupt request not enabled

bit 1 **INT2IE:** External Interrupt 2 Enable bit
1 = Interrupt request enabled
0 = Interrupt request not enabled

bit 0 **INT1IE:** External Interrupt 1 Enable bit
1 = Interrupt request enabled
0 = Interrupt request not enabled

REGISTER 5-8: IEC2: INTERRUPT ENABLE CONTROL REGISTER 2

U-0	U-0	U-0	U-0	U-0	R/W-0	R/W-0	R/W-0
—	—	—	—	—	ADCP5IE	ADCP4IE	ADCP3IE
bit 15							bit 8

R/W-0	R/W-0	R/W-0	U-0	U-0	U-0	U-0	R/W-0
ADCP2IE	ADCP1IE	ADCP0IE	—	—	—	—	AC4IE
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 15-11 **Unimplemented:** Read as '0'

bit 10 **ADCP5IE:** ADC Pair 5 Conversion done Interrupt Enable bit
1 = Interrupt request enabled
0 = Interrupt request not enabled

bit 9 **ADCP4IE:** ADC Pair 4 Conversion done Interrupt Enable bit
1 = Interrupt request enabled
0 = Interrupt request not enabled

bit 8 **ADCP3IE:** ADC Pair 3 Conversion done Interrupt Enable bit
1 = Interrupt request enabled
0 = Interrupt request not enabled

bit 7 **ADCP2IE:** ADC Pair 2 Conversion done Interrupt Enable bit
1 = Interrupt request enabled
0 = Interrupt request not enabled

bit 6 **ADCP1IE:** ADC Pair 1 Conversion done Interrupt Enable bit
1 = Interrupt request enabled
0 = Interrupt request not enabled

bit 5 **ADCP0IE:** ADC Pair 0 Conversion done Interrupt Enable bit
1 = Interrupt request enabled
0 = Interrupt request not enabled

bit 4-1 **Unimplemented:** Read as '0'

bit 0 **AC4IE:** Analog Comparator #4 Interrupt Enable bit
1 = Interrupt request enabled
0 = Interrupt request not enabled

7.6.3 LOADING WRITE LATCHES

Example 7-2 shows a sequence of instructions that can be used to load the 96 bytes of write latches. 32 TBLWTL and 32 TBLWTH instructions are needed to load the write latches selected by the table pointer.

EXAMPLE 7-2: LOADING WRITE LATCHES

```
; Set up a pointer to the first program memory location to be written
; program memory selected, and writes enabled
       MOV     #0x0000,W0                      ;
       MOV     W0,TBLPAG                       ; Initialize PM Page Boundary SFR
       MOV     #0x6000,W0                      ; An example program memory address
; Perform the TBLWT instructions to write the latches
; 0th_program_word
       MOV     #LOW_WORD_0,W2                  ;
       MOV     #HIGH_BYTE_0,W3                 ;
       TBLWTL  W2,[W0]                         ; Write PM low word into program latch
       TBLWTH  W3,[W0++]                       ; Write PM high byte into program latch
; 1st_program_word
       MOV     #LOW_WORD_1,W2                  ;
       MOV     #HIGH_BYTE_1,W3                 ;
       TBLWTL  W2,[W0]                         ; Write PM low word into program latch
       TBLWTH  W3,[W0++]                       ; Write PM high byte into program latch
;  2nd_program_word
       MOV     #LOW_WORD_2,W2                  ;
       MOV     #HIGH_BYTE_2,W3                 ;
       TBLWTL  W2, [W0]                        ; Write PM low word into program latch
       TBLWTH  W3, [W0++]                      ; Write PM high byte into program latch
       •
       •
       •
; 31st_program_word
       MOV     #LOW_WORD_31,W2                 ;
       MOV     #HIGH_BYTE_31,W3                ;
       TBLWTL  W2, [W0]                        ; Write PM low word into program latch
       TBLWTH  W3, [W0++]                      ; Write PM high byte into program latch
```

Note: In Example 7-2, the contents of the upper byte of W3 have no effect.

7.6.4 INITIATING THE PROGRAMMING SEQUENCE

For protection, the write initiate sequence for NVMKEY must be used to allow any erase or program operation to proceed. After the programming command has been executed, the user must wait for the programming time until programming is complete. The two instructions following the start of the programming sequence should be NOPs.

EXAMPLE 7-3: INITIATING A PROGRAMMING SEQUENCE

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

REGISTER 12-9: ALTDTRx: PWM ALTERNATE DEAD-TIME REGISTER

U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	ALTDTRx<13:8>					
bit 15							bit 8

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	U-0	U-0
ALTDTR <7:2>						—	—
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’	
-n = Value at POR	‘1’ = Bit is set	‘0’ = Bit is cleared	x = Bit is unknown

bit 15-14 **Unimplemented:** Read as ‘0’

bit 13-2 **ALTDTRx<13:2>:** Unsigned 12-bit Dead-Time Value bits for PWMx Dead-Time Unit bits

bit 1-0 **Unimplemented:** Read as ‘0’

REGISTER 12-10: TRGCONx: PWM TRIGGER CONTROL REGISTER

R/W-0	R/W-0	R/W-0	U-0	U-0	U-0	U-0	U-0
TRGDIV<2:0>			—	—	—	—	—
bit 15							bit 8

U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	TRGSTRT<5:0>					
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’	
-n = Value at POR	‘1’ = Bit is set	‘0’ = Bit is cleared	x = Bit is unknown

bit 15-13 **TRGDIV<2:0>:** Trigger Output Divider bits
000 = Trigger output for every trigger event
001 = Trigger output for every 2nd trigger event
010 = Trigger output for every 3rd trigger event
011 = Trigger output for every 4th trigger event
100 = Trigger output for every 5th trigger event
101 = Trigger output for every 6th trigger event
110 = Trigger output for every 7th trigger event
111 = Trigger output for every 8th trigger event

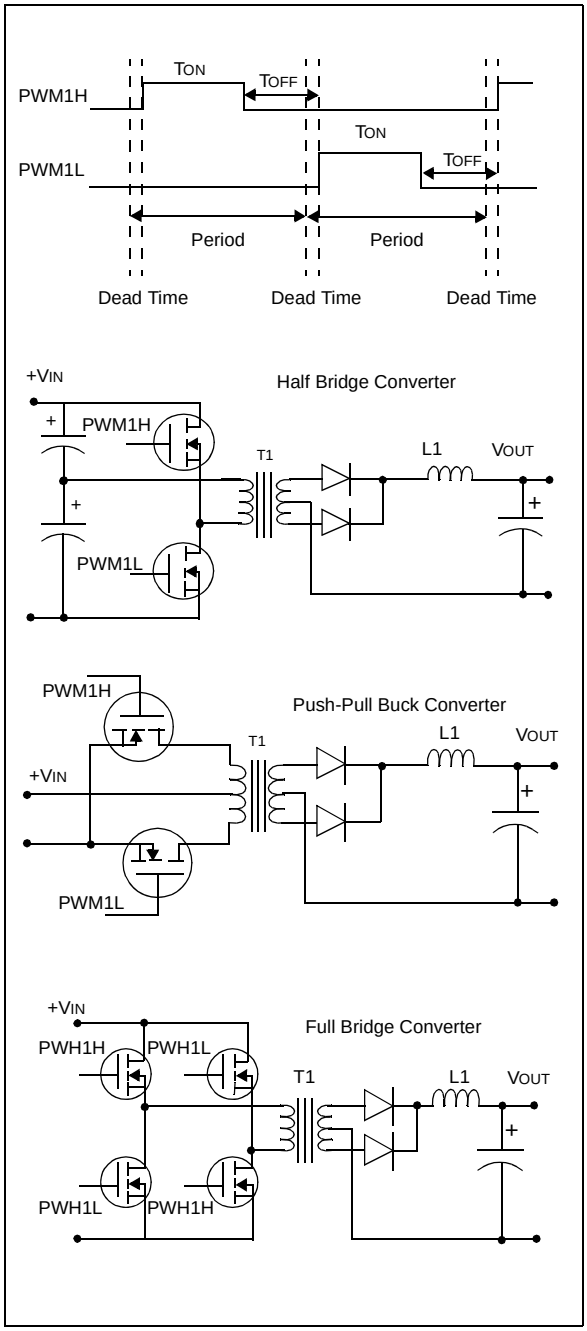
bit 12-6 **Unimplemented:** Read as ‘0’

bit 5-0 **TRGSTRT<5:0>:** Trigger Postscaler Start Enable Select bits
This value specifies the ROLL counter value needed for a match that will then enable the trigger postscaler logic to begin counting trigger events.

12.34.3 APPLICATION OF PUSH-PULL PWM MODE

Push-Pull PWM mode is typically used in transformer coupled circuits to ensure that no net DC currents flow through the transformer. Push-Pull mode ensures that the same duty cycle PWM pulse is applied to the transformer windings in alternate directions, as shown in Figure 12-24.

FIGURE 12-24: APPLICATIONS OF PUSH-PULL PWM MODE



12.34.4 APPLICATION OF MULTI-PHASE PWM MODE

Multi-Phase PWM mode is often used in DC/DC converters that must handle very fast load current transients and fit into tight spaces. A multi-phase converter is essentially a parallel array of buck converters that are operated slightly out of phase of each other, as shown in Figure 12-25. The multiple phases create an effective switching speed equal to the sum of the individual converters. If a single phase is operating with a 333 KHz PWM frequency, then the effective switching frequency for the circuit is 1 MHz. This high switching frequency greatly reduces output capacitor size requirements and improves load transient response.

FIGURE 12-25: APPLICATIONS OF MULTI-PHASE PWM MODE

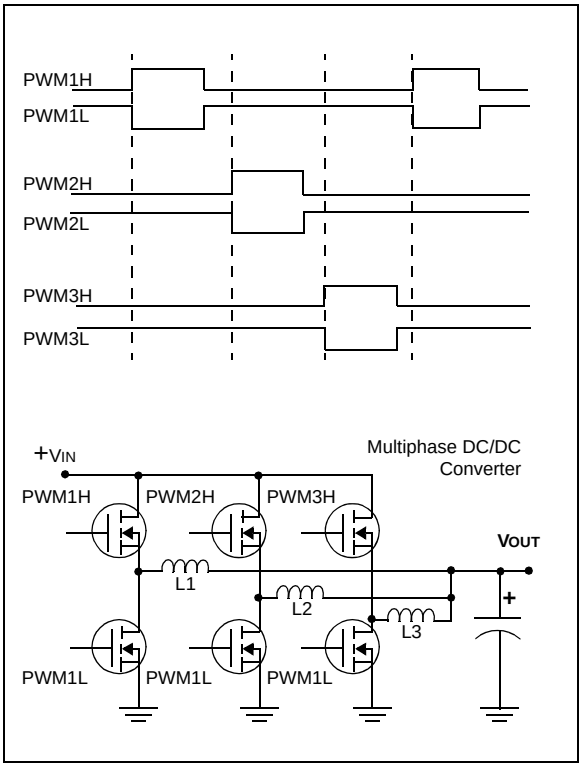


TABLE 15-1: UART1 REGISTER MAP

SFR Name	SFR Addr	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	All Resets
U1MODE	0220	UARTEN	—	USIDL	IREN	—	ALTIO	—	—	WAKE	LPBACK	ABAUD	RXINV	BRGH	PDSEL<1:0>		STSEL	0000
U1STA	0222	UTXISEL1	UTXINV	UTXISEL0	—	UTXBRK	UTXEN	UTXBF	TRMT	URXISEL<1:0>		ADDEN	RIDLE	PERR	FERR	OERR	URXDA	0110
U1TXREG	0224	—	—	—	—	—	—	—	UART Transmit Register									xxxx
U1RXREG	0226	—	—	—	—	—	—	—	UART Receive Register									0000
U1BRG	0228	Baud Rate Generator Prescaler																0000

Legend: x = unknown value on Reset, — = unimplemented, read as '0'. Reset values are shown in hexadecimal.

REGISTER 16-3: A/D BASE REGISTER (ADBASE)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
ADBASE<15:8>							
bit 15							bit 8

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	U-0
ADBASE<7:1>							—
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 15-1 **ADC Base Register:** This register contains the base address of the user's ADC Interrupt Service Routine jump table. This register, when read, contains the sum of the ADBASE register contents and the encoded value of the PxRDY Status bits.

The encoder logic provides the bit number of the highest priority PxRDY bits where P0RDY is the highest priority, and P5RDY is lowest priority.

Note: The encoding results are shifted left two bits so bits 1-0 of the result are always zero.

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

Note: As an alternative to using the ADBASE Register, the ADCP0-5 ADC Pair Conversion Complete Interrupts (Interrupts 37-42) can be used to invoke A to D conversion completion routines for individual ADC input pairs. Refer to **Section 16.9 "Individual Pair Interrupts"**.

REGISTER 16-4: A/D PORT CONFIGURATION REGISTER (ADPCFG)

U-0	U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	—	PCFG11	PCFG10	PCFG9	PCFG8
bit 15							bit 8

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
PCFG7	PCFG6	PCFG5	PCFG4	PCFG3	PCFG2	PCFG1	PCFG0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 15-12 **Unimplemented:** Read as '0'

bit 11-0 **PCFG<11:0>:** A/D Port Configuration Control bits

1 = Port pin in Digital mode, port read input enabled, A/D input multiplexor connected to AVSS

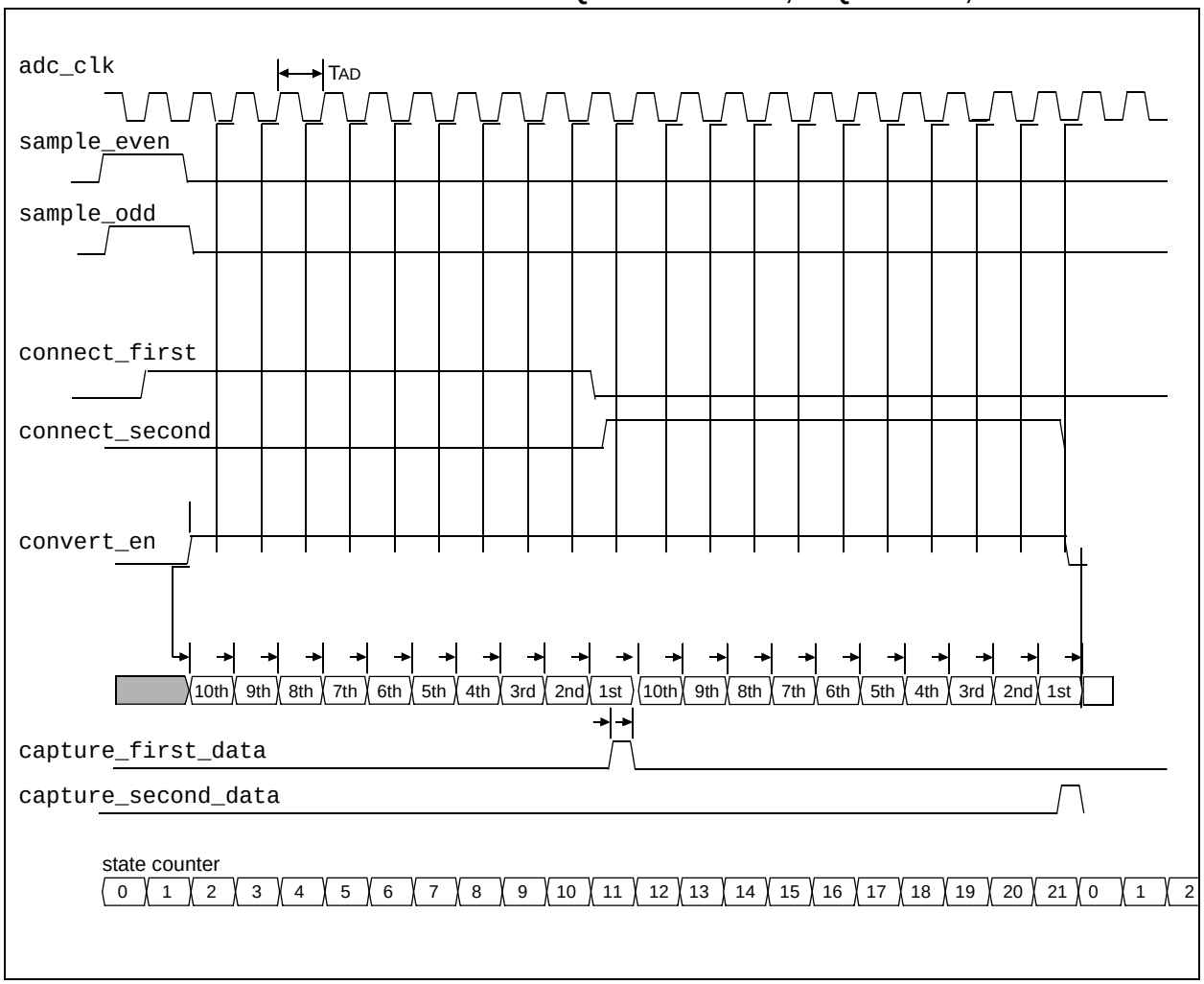
0 = Port pin in Analog mode, port read input disabled, A/D samples pin voltage

16.17 A/D Sample and Convert Timing

The sample and hold circuits assigned to the input pins have their own timing logic that is triggered when an external sample and convert request (from PWM or TMR) is made. The sample and hold circuits have a fixed two clock data sample period. When the sample has been acquired, then the ADC control logic is notified of a pending request, then the conversion is performed as the conversion resources become available.

The ADC module always converts pairs of analog input channels, so a typical conversion process requires 24 clock cycles.

FIGURE 16-3: DETAILED CONVERSION SEQUENCE TIMINGS, SEQSAMP = 0, NOT BUSY



17.0 SMPS COMPARATOR MODULE

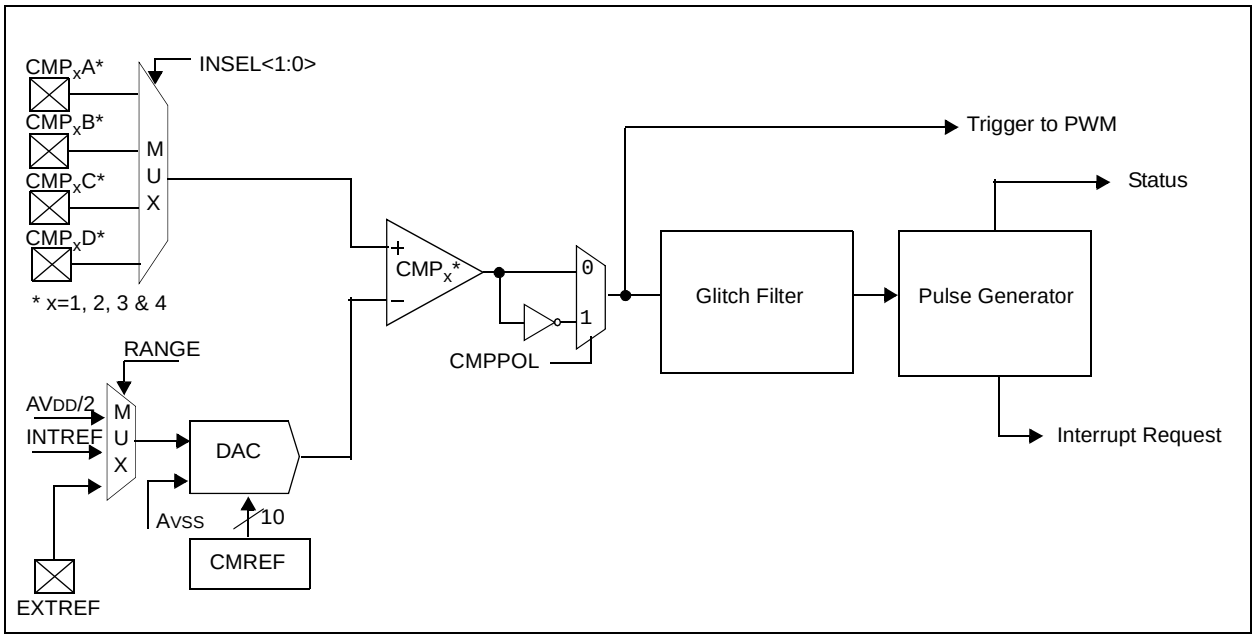
Note: This data sheet summarizes features of this group of dsPIC30F devices and is not intended to be a complete reference source. For more information on the CPU, peripherals, register descriptions and general device functionality, refer to the "*dsPIC30F Family Reference Manual*" (DS70046).

The dsPIC30F SMPS Comparator module monitors current and/or voltage transients that may be too fast for the CPU and ADC to capture.

17.1 Features Overview

- 16 comparator inputs
- 10-bit DAC provides reference
- Programmable output polarity
- Interrupt generation capability
- Selectable Input sources
- DAC has three ranges of operation:
  - AVDD/2
  - Internal Reference 1.2V 1%
  - External Reference < (AVDD - 1.6V)
- ADC sample and convert trigger capability
- Can be disabled to reduce power consumption
- Functional support for PWM Module:
  - PWM Duty Cycle Control
  - PWM Period Control
  - PWM Fault Detect

FIGURE 17-1: COMPARATOR MODULE BLOCK DIAGRAM



17.2 Module Applications

This module provides a means for the SMPS dsPIC DSC devices to monitor voltage and currents in a power conversion application. The ability to detect transient conditions and stimulate the dsPIC DSC processor and/or peripherals without requiring the processor and ADC to constantly monitor voltages or currents frees the dsPIC DSC to perform other tasks.

The Comparator module has a high-speed comparator and an associated 10-bit DAC that provides a programmable reference voltage to one input of the comparator. The polarity of the comparator output is user programmable. The output of the module can be used in the following modes:

- Generate an interrupt
- Trigger an ADC sample and convert process
- Truncate the PWM signal (current limit)
- Truncate the PWM period (current minimum)
- Disable the PWM outputs (Fault-latch)

The output of the Comparator module may be used in multiple modes at the same time, such as: (1) generate an interrupt, (2) have the ADC take a sample and convert it and (3) truncate the PWM output in response to a voltage being detected beyond its expected value.

The Comparator module can also be used to wake-up the system from Sleep or Idle mode when the analog input voltage exceeds the programmed threshold voltage.

REGISTER 18-5: FOSCSEL: OSCILLATOR SELECTION CONFIGURATION BITS

U-0	U-0	U-0	U-0	U-0	U-0	U-0	U-0
—	—	—	—	—	—	—	—
bit 23							bit 16

U-0	U-0	U-0	U-0	U-0	U-0	U-0	U-0
—	—	—	—	—	—	—	—
bit 15							bit 8

U-0	U-0	U-0	U-0	U-0	U-0	R/P	R/P
—	—	—	—	—	—	FNOSC1	FNOSC0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’	
-n = Value at POR	‘1’ = Bit is set	‘0’ = Bit is cleared	x = Bit is unknown

bit 23-2 **Unimplemented:** Read as ‘0’

bit 1-0 **FNOSC<1:0>:** Initial Oscillator Group Selection on POR bits
00 = Fast RC Oscillator (FRC)
01 = Fast RC Oscillator (FRC) divided by N, with PLL module
10 = Primary Oscillator (HS,EC)
11 = Primary Oscillator (HS,EC) with PLL module

REGISTER 18-6: FOSC: OSCILLATOR SELECTION CONFIGURATION BITS

U-0	U-0	U-0	U-0	U-0	U-0	U-0	U-0
—	—	—	—	—	—	—	—
bit 23							bit 16

U-0	U-0	U-0	U-0	U-0	U-0	U-0	U-0
—	—	—	—	—	—	—	—
bit 15							bit 8

R/P	R/P	R/P	U-0	U-0	R/P	R/P	R/P
FCKSM<1:0>		FRANGE	—	—	OSCIOFNC	POSCMD<1:0>	
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’	
-n = Value at POR	‘1’ = Bit is set	‘0’ = Bit is cleared	x = Bit is unknown

bit 23-8 **Unimplemented:** Read as ‘0’

bit 7-6 **FCKSM<1:0>:** Clock Switching and Monitor Selection Configuration bits

1x = Clock switching is disabled, fail-safe clock monitor is disabled
01 = Clock switching is enabled, fail-safe clock monitor is disabled
00 = Clock switching is enabled, fail-safe clock monitor is enabled

bit 5 **FRANGE:** Frequency Range Select for FRC and PLL bit
Acts like a “Gear Shift” feature that enables the dsPIC DSC device to operate at reduced MIPS at a reduced supply voltage (3.3V)

FRANGE Bit Value	Temperature Rating	FRC Frequency (Nominal)	PLL VCO (Nominal)
1 = High Range	Industrial Extended	14.55 MHz 9.7 MHz	466 MHz (480 MHz max.) 310 MHz (320 MHz max.)
0 = Low Range	Industrial Extended	9.7 MHz 6.4 MHz	310 MHz (320 MHz max.) 205 MHz (211 MHz max.)

bit 4-3 **Unimplemented:** Read as ‘0’

bit 3 **OSCIOFNC:** OSC2 Pin I/O Enable bit

1 = CLKO output signal active on the OSCO pin
0 = CLKO output disabled

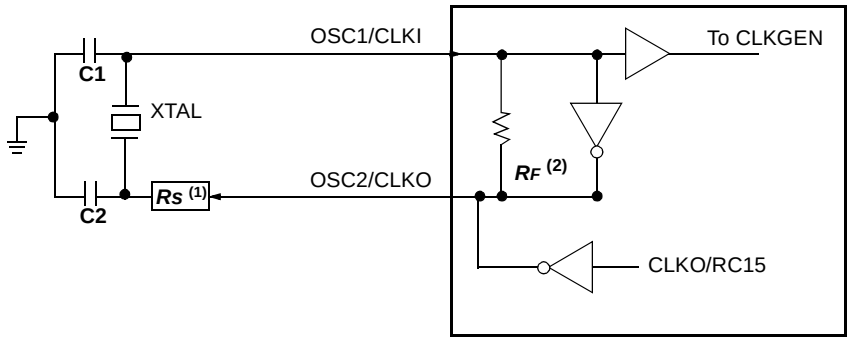
bit 1-0 **POSCMD<1:0>:** Primary Oscillator Mode

11 = Primary Oscillator Disabled
10 = HS oscillator mode selected
01 = Reserved
00 = External clock mode selected

18.4 PRIMARY OSCILLATOR ON OSC1/ OSC2 PINS:

The primary oscillator uses is shown in Figure 18-3.

FIGURE 18-3: PRIMARY OSCILLATOR



Note 1: A series resistor, Rs, may be required for AT strip cut crystals.

2: The feedback resistor, RF, is typically in the range of 2 to 10 MΩ.

18.5 EXTERNAL CLOCK INPUT

Two of the primary Oscillator modes use an external clock. These modes are EC and EC with IO.

In the EC mode (Figure 18-4), the OSC1 pin can be driven by CMOS drivers. In this mode, the OSC1 pin is high-impedance and the OSC2 pin is the clock output (FOSC/2). This output clock is useful for testing or synchronization purposes.

In the EC with IO mode (Figure 18-5), the OSC1 pin can be driven by CMOS drivers. In this mode, the OSC1 pin is high-impedance and the OSC2 pin becomes a general purpose I/O pin. The feedback device between OSC1 and OSC2 is turned off to save current.

FIGURE 18-4: EXTERNAL CLOCK INPUT OPERATION (EC OSCILLATOR CONFIGURATION)

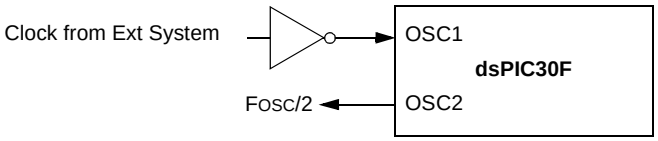


FIGURE 18-5: EXTERNAL CLOCK INPUT OPERATION (ECIO OSCILLATOR CONFIGURATION)

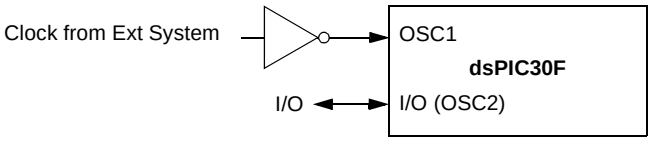


TABLE 18-8: SYSTEM INTEGRATION REGISTER MAP FOR dsPIC30F202X

SFR Name	Addr.	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset State
RCON	0740	TRAPR	IOPUWR	—	—	—	—	—	—	EXTR	SWR	SWDTEN	WDTO	SLEEP	IDLE	—	POR	Depends on type of Reset.
OSCCON	0742	—	COSC<2:0>			—	NOSC<2:0>			CLKLOCK	—	LOCK	PRCDEN	CF	TSEQEN	—	OSWEN	Depends on Configuration bits.
OSCTUN	0748	TSEQ3<3:0>				TSEQ2<3:0>				TSEQ1<3:0>				TUN<3:0>				0000 0000 0000 0000
OSCTUN2	074A	TSEQ7<3:0>				TSEQ6<3:0>				TSEQ5<3:0>				TSEQ4<3:0>				0000 0000 0000 0000
LFSR	074C	—	LFSR<14:0>															0000 0000 0000 0000
PMD1	0770	—	—	T3MD	T2MD	T1MD	—	PWMMD	—	I2CMD	—	U1MD	—	SPI1MD	—	—	ADCMD	0000 0000 0000 0000
PMD2	0772	—	—	—	—	—	—	—	IC1MD	—	—	—	—	—	—	OC2MD	OC1MD	0000 0000 0000 0000
PMD3	0774	—	—	—	—	CMP_PSMD	—	—	—	—	—	—	—	—	—	—	—	0000 0000 0000 0000

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

TABLE 18-9: DEVICE CONFIGURATION REGISTER MAP

File Name	Addr.	Bits 23-16	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
FBS	F80000	—	—	—	—	—	—	—	—	—	—	—	—	—	BSS<2:0>			BWRP
FGS	F80004	—	—	—	—	—	—	—	—	—	—	—	—	—	—	GSS1	GSS0	GWRP
FOSCSEL	F80006	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	FNOSC<1:0>	
FOSC	F80008	—	—	—	—	—	—	—	—	—	FCKSM<1:0>		FRANGE	—	—	OSCIOFNC	POSCMD<1:0>	
FWDT	F8000A	—	—	—	—	—	—	—	—	—	FWDTEN	WWDTEN	—	WDTPRE	WDTPOST<3:0>			
FPOR	F8000C	—	—	—	—	—	—	—	—	—	—	—	—	—	—	FPWRT<2:0>		

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

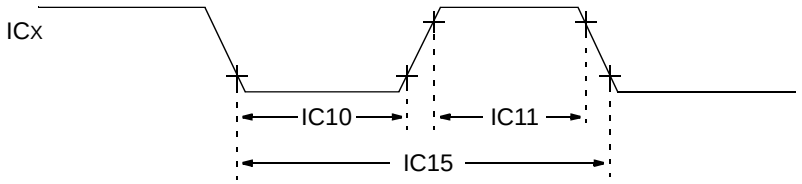
TABLE 19-2: INSTRUCTION SET OVERVIEW

Base Instr #	Assembly Mnemonic	Assembly Syntax		Description	# of words	# of cycles	Status Flags Affected
1	ADD	ADD	Acc	Add Accumulators	1	1	OA,OB,SA,SB
		ADD	f	f = f + WREG	1	1	C,DC,N,OV,Z
		ADD	f,WREG	WREG = f + WREG	1	1	C,DC,N,OV,Z
		ADD	#lit10,Wn	Wd = lit10 + Wd	1	1	C,DC,N,OV,Z
		ADD	Wb,Ws,Wd	Wd = Wb + Ws	1	1	C,DC,N,OV,Z
		ADD	Wb,#lit5,Wd	Wd = Wb + lit5	1	1	C,DC,N,OV,Z
		ADD	Wso,#Slit4,Acc	16-bit Signed Add to Accumulator	1	1	OA,OB,SA,SB
2	ADDC	ADDC	f	f = f + WREG + (C)	1	1	C,DC,N,OV,Z
		ADDC	f,WREG	WREG = f + WREG + (C)	1	1	C,DC,N,OV,Z
		ADDC	#lit10,Wn	Wd = lit10 + Wd + (C)	1	1	C,DC,N,OV,Z
		ADDC	Wb,Ws,Wd	Wd = Wb + Ws + (C)	1	1	C,DC,N,OV,Z
		ADDC	Wb,#lit5,Wd	Wd = Wb + lit5 + (C)	1	1	C,DC,N,OV,Z
3	AND	AND	f	f = f .AND. WREG	1	1	N,Z
		AND	f,WREG	WREG = f .AND. WREG	1	1	N,Z
		AND	#lit10,Wn	Wd = lit10 .AND. Wd	1	1	N,Z
		AND	Wb,Ws,Wd	Wd = Wb .AND. Ws	1	1	N,Z
		AND	Wb,#lit5,Wd	Wd = Wb .AND. lit5	1	1	N,Z
4	ASR	ASR	f	f = Arithmetic Right Shift f	1	1	C,N,OV,Z
		ASR	f,WREG	WREG = Arithmetic Right Shift f	1	1	C,N,OV,Z
		ASR	Ws,Wd	Wd = Arithmetic Right Shift Ws	1	1	C,N,OV,Z
		ASR	Wb,Wns,Wnd	Wnd = Arithmetic Right Shift Wb by Wns	1	1	N,Z
		ASR	Wb,#lit5,Wnd	Wnd = Arithmetic Right Shift Wb by lit5	1	1	N,Z
5	BCLR	BCLR	f,#bit4	Bit Clear f	1	1	None
		BCLR	Ws,#bit4	Bit Clear Ws	1	1	None
6	BRA	BRA	C,Expr	Branch if Carry	1	1 (2)	None
		BRA	GE,Expr	Branch if greater than or equal	1	1 (2)	None
		BRA	GEU,Expr	Branch if unsigned greater than or equal	1	1 (2)	None
		BRA	GT,Expr	Branch if greater than	1	1 (2)	None
		BRA	GTU,Expr	Branch if unsigned greater than	1	1 (2)	None
		BRA	LE,Expr	Branch if less than or equal	1	1 (2)	None
		BRA	LEU,Expr	Branch if unsigned less than or equal	1	1 (2)	None
		BRA	LT,Expr	Branch if less than	1	1 (2)	None
		BRA	LTU,Expr	Branch if unsigned less than	1	1 (2)	None
		BRA	N,Expr	Branch if Negative	1	1 (2)	None
		BRA	NC,Expr	Branch if Not Carry	1	1 (2)	None
		BRA	NN,Expr	Branch if Not Negative	1	1 (2)	None
		BRA	NOV,Expr	Branch if Not Overflow	1	1 (2)	None
		BRA	NZ,Expr	Branch if Not Zero	1	1 (2)	None
		BRA	OA,Expr	Branch if accumulator A overflow	1	1 (2)	None
		BRA	OB,Expr	Branch if accumulator B overflow	1	1 (2)	None
		BRA	OV,Expr	Branch if Overflow	1	1 (2)	None
		BRA	SA,Expr	Branch if accumulator A saturated	1	1 (2)	None
		BRA	SB,Expr	Branch if accumulator B saturated	1	1 (2)	None
		BRA	Expr	Branch Unconditionally	1	2	None
		BRA	Z,Expr	Branch if Zero	1	1 (2)	None
		BRA	Wn	Computed Branch	1	2	None
7	BSET	BSET	f,#bit4	Bit Set f	1	1	None
		BSET	Ws,#bit4	Bit Set Ws	1	1	None
8	BSW	BSW.C	Ws,Wb	Write C bit to Ws<Wb>	1	1	None
		BSW.Z	Ws,Wb	Write Z bit to Ws<Wb>	1	1	None
9	BTG	BTG	f,#bit4	Bit Toggle f	1	1	None
		BTG	Ws,#bit4	Bit Toggle Ws	1	1	None
10	BTSC	BTSC	f,#bit4	Bit Test f, Skip if Clear	1	1 (2 or 3)	None
		BTSC	Ws,#bit4	Bit Test Ws, Skip if Clear	1	1 (2 or 3)	None

TABLE 19-2: INSTRUCTION SET OVERVIEW (CONTINUED)

Base Instr #	Assembly Mnemonic	Assembly Syntax	Description	# of words	# of cycles	Status Flags Affected
11	BTSS	BTSS f,#bit4	Bit Test f, Skip if Set	1	1 (2 or 3)	None
		BTSS Ws,#bit4	Bit Test Ws, Skip if Set	1	1 (2 or 3)	None
12	BTST	BTST f,#bit4	Bit Test f	1	1	Z
		BTST.C Ws,#bit4	Bit Test Ws to C	1	1	C
		BTST.Z Ws,#bit4	Bit Test Ws to Z	1	1	Z
		BTST.C Ws,Wb	Bit Test Ws<Wb> to C	1	1	C
		BTST.Z Ws,Wb	Bit Test Ws<Wb> to Z	1	1	Z
13	BTSTS	BTSTS f,#bit4	Bit Test then Set f	1	1	Z
		BTSTS.C Ws,#bit4	Bit Test Ws to C, then Set	1	1	C
		BTSTS.Z Ws,#bit4	Bit Test Ws to Z, then Set	1	1	Z
14	CALL	CALL lit23	Call subroutine	2	2	None
		CALL Wn	Call indirect subroutine	1	2	None
15	CLR	CLR f	f = 0x0000	1	1	None
		CLR WREG	WREG = 0x0000	1	1	None
		CLR Ws	Ws = 0x0000	1	1	None
		CLR Acc,Wx,Wxd,Wy,Wyd,AWB	Clear Accumulator	1	1	OA,OB,SA,SB
16	CLRWDT	CLRWDT	Clear Watchdog Timer	1	1	WDTO,Sleep
17	COM	COM f	$f = \overline{f}$	1	1	N,Z
		COM f,WREG	$WREG = \overline{f}$	1	1	N,Z
		COM Ws,Wd	$Wd = \overline{Ws}$	1	1	N,Z
18	CP	CP f	Compare f with WREG	1	1	C,DC,N,OV,Z
		CP Wb,#lit5	Compare Wb with lit5	1	1	C,DC,N,OV,Z
		CP Wb,Ws	Compare Wb with Ws (Wb – Ws)	1	1	C,DC,N,OV,Z
19	CP0	CP0 f	Compare f with 0x0000	1	1	C,DC,N,OV,Z
		CP0 Ws	Compare Ws with 0x0000	1	1	C,DC,N,OV,Z
20	CPB	CPB f	Compare f with WREG, with Borrow	1	1	C,DC,N,OV,Z
		CPB Wb,#lit5	Compare Wb with lit5, with Borrow	1	1	C,DC,N,OV,Z
		CPB Wb,Ws	Compare Wb with Ws, with Borrow ($Wb – Ws – \overline{C}$)	1	1	C,DC,N,OV,Z
21	CPSEQ	CPSEQ Wb, Wn	Compare Wb with Wn, skip if =	1	1 (2 or 3)	None
22	CPSGT	CPSGT Wb, Wn	Compare Wb with Wn, skip if >	1	1 (2 or 3)	None
23	CPSLT	CPSLT Wb, Wn	Compare Wb with Wn, skip if <	1	1 (2 or 3)	None
24	CPSNE	CPSNE Wb, Wn	Compare Wb with Wn, skip if ≠	1	1 (2 or 3)	None
25	DAW	DAW Wn	Wn = decimal adjust Wn	1	1	C
26	DEC	DEC f	f = f –1	1	1	C,DC,N,OV,Z
		DEC f,WREG	WREG = f –1	1	1	C,DC,N,OV,Z
		DEC Ws,Wd	Wd = Ws – 1	1	1	C,DC,N,OV,Z
27	DEC2	DEC2 f	f = f –2	1	1	C,DC,N,OV,Z
		DEC2 f,WREG	WREG = f – 2	1	1	C,DC,N,OV,Z
		DEC2 Ws,Wd	Wd = Ws – 2	1	1	C,DC,N,OV,Z
28	DISI	DISI #lit14	Disable Interrupts for k instruction cycles	1	1	None
29	DIV	DIV.S Wm,Wn	Signed 16/16-bit Integer Divide	1	18	N,Z,C, OV
		DIV.SD Wm,Wn	Signed 32/16-bit Integer Divide	1	18	N,Z,C, OV
		DIV.U Wm,Wn	Unsigned 16/16-bit Integer Divide	1	18	N,Z,C, OV
		DIV.UD Wm,Wn	Unsigned 32/16-bit Integer Divide	1	18	N,Z,C, OV
30	DIVF	DIVF Wm,Wn	Signed 16/16-bit Fractional Divide	1	18	N,Z,C, OV
31	DO	DO #lit14,Expr	Do code to PC + Expr, lit14 + 1 times	2	2	None
		DO Wn,Expr	Do code to PC + Expr, (Wn) + 1 times	2	2	None
32	ED	ED Wm * Wm,Acc,Wx,Wy,Wxd	Euclidean Distance (no accumulate)	1	1	OA,OB,OAB, SA,SB,SAB
33	EDAC	EDAC Wm * Wm,Acc,Wx,Wy,Wxd	Euclidean Distance	1	1	OA,OB,OAB, SA,SB,SAB

FIGURE 21-7: INPUT CAPTURE x (ICx) TIMING CHARACTERISTICS



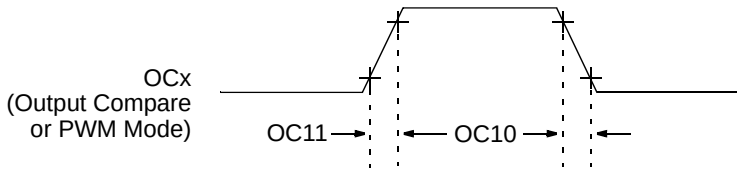
Note: Refer to Figure 21-1 for load conditions.

TABLE 21-23: INPUT CAPTURE x TIMING REQUIREMENTS

AC CHARACTERISTICS				Standard Operating Conditions: 3.3V and 5.0V (±10%) (unless otherwise stated) Operating temperature -40°C ≤ TA ≤ +85°C for Industrial -40°C ≤ TA ≤ +125°C for Extended			
Param No.	**Symbol**	**Characteristic(1)**		**Min**	**Max**	**Units**	**Conditions**
IC10	TccL	ICx Input Low Time	No Prescaler	0.5 TCY + 20	—	ns	
			With Prescaler	10	—	ns	
IC11	TccH	ICx Input High Time	No Prescaler	0.5 TCY + 20	—	ns	
			With Prescaler	10	—	ns	
IC15	TccP	ICx Input Period		(2 TCY + 40)/N	—	ns	N = Prescale value (1, 4, 16)

Note 1: These parameters are characterized but not tested in manufacturing.

FIGURE 21-8: OUTPUT COMPARE x (OCx) MODULE TIMING CHARACTERISTICS



Note: Refer to Figure 21-1 for load conditions.

TABLE 21-24: OUTPUT COMPARE x MODULE TIMING REQUIREMENTS

AC CHARACTERISTICS			Standard Operating Conditions: 3.3V and 5.0V (±10%) (unless otherwise stated) Operating temperature -40°C ≤ TA ≤ +85°C for Industrial -40°C ≤ TA ≤ +125°C for Extended				
Param No.	**Symbol**	**Characteristic(1)**	**Min**	**Typ(2)**	**Max**	**Units**	**Conditions**
OC10	TccF	OCx Output Fall Time	—	—	—	ns	See Parameter D032
OC11	TccR	OCx Output Rise Time	—	—	—	ns	See Parameter D031

Note 1: These parameters are characterized but not tested in manufacturing.

2: Data in "Typ" column is at 5V, +25°C unless otherwise stated. Parameters are for design guidance only and are not tested.

TABLE 21-33: 10-BIT HIGH-SPEED A/D MODULE SPECIFICATIONS

AC CHARACTERISTICS			Standard Operating Conditions: 3.3V and 5.0V (±10%) (unless otherwise stated) Operating temperature -40°C ≤ TA ≤ +85°C for Industrial -40°C ≤ TA ≤ +125°C for Extended				
Param No.	**Symbol**	**Characteristic**	**Min.**	**Typ**	**Max.**	**Units**	**Conditions**
Device Supply							
AD01	AVDD	Module VDD Supply	Greater of: VDD – 0.3 or 2.7		Lesser of: VDD + 0.3 or 5.5	V	
AD02	AVSS	Module VSS Supply	Vss – 0.3		VSS + 0.3	V	
Analog Input							
AD10	VINH-VINL	Full-Scale Input Span	VSS		VDD	V	
AD11	VIN	Absolute Input Voltage	AVSS – 0.3		AVDD + 0.3	V	
AD12	—	Leakage Current	—	±0.001	±0.244	µA	VINL = AVSS = 0V, AVDD = 5V, Source Impedance = 1 kΩ
AD13	—	Leakage Current	—	±0.001	±0.244	µA	VINL = AVSS = 0V, AVDD = 3.3V, Source Impedance = 1 kΩ
AD17	RIN	Recommended Impedance of Analog Voltage Source	—		1K	Ω	
DC Accuracy							
AD20	Nr	Resolution	10 data bits			bits	
AD21	INL	Integral Nonlinearity	—	±0.5	< ±1	LSb	VINL = AVSS = 0V, AVDD = 5V
AD21A	INL	Integral Nonlinearity	—	±0.5	< ±1	LSb	VINL = AVSS = 0V, AVDD = 3.3V
AD22	DNL	Differential Nonlinearity	—	±0.5	< ±1	LSb	VINL = AVSS = 0V, AVDD = 5V
AD22A	DNL	Differential Nonlinearity	—	±0.5	< ±1	LSb	VINL = AVSS = 0V, AVDD = 3.3V
AD23	GERR	Gain Error	—	±0.75	<±4.0	LSb	VINL = AVSS = 0V, AVDD = 5V
AD23A	GERR	Gain Error	—	±0.75	<±3.0	LSb	VINL = AVSS = 0V, AVDD = 3.3V

Note 1: Because the sample caps will eventually lose charge, clock rates below 10 kHz can affect linearity performance, especially at elevated temperatures.

2: The A/D conversion result never decreases with an increase in the input voltage, and has no missing codes.