



Welcome to [E-XFL.COM](https://www.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	16-Bit
Speed	32MHz
Connectivity	I <sup>2</sup> C, PMP, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, LVD, POR, PWM, WDT
Number of I/O	35
Program Memory Size	16KB (5.5K x 24)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	4K x 8
Voltage - Supply (Vcc/Vdd)	2V ~ 3.6V
Data Converters	A/D 13x10b
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	<a href="https://www.e-xfl.com/product-detail/microchip-technology/pic24fj16ga004-i-pt">https://www.e-xfl.com/product-detail/microchip-technology/pic24fj16ga004-i-pt</a>

# PIC24FJ64GA004 FAMILY

---

## 2.7 Configuration of Analog and Digital Pins During ICSP Operations

If an ICSP compliant emulator is selected as a debugger, it automatically initializes all of the A/D input pins (ANx) as “digital” pins. Depending on the particular device, this is done by setting all bits in the ADnPCFG register(s), or clearing all bit in the ANSx registers.

All PIC24F devices will have either one or more ADnPCFG registers or several ANSx registers (one for each port); no device will have both. Refer to **Section 21.0 “10-Bit High-Speed A/D Converter”** for more specific information.

The bits in these registers that correspond to the A/D pins that initialized the emulator must not be changed by the user application firmware; otherwise, communication errors will result between the debugger and the device.

If your application needs to use certain A/D pins as analog input pins during the debug session, the user application must modify the appropriate bits during initialization of the A/D module, as follows:

- For devices with an ADnPCFG register, clear the bits corresponding to the pin(s) to be configured as analog. Do not change any other bits, particularly those corresponding to the PGECx/PGEDx pair, at any time.
- For devices with ANSx registers, set the bits corresponding to the pin(s) to be configured as analog. Do not change any other bits, particularly those corresponding to the PGECx/PGEDx pair, at any time.

When a Microchip debugger/emulator is used as a programmer, the user application firmware must correctly configure the ADnPCFG or ANSx registers. Automatic initialization of this register is only done during debugger operation. Failure to correctly configure the register(s) will result in all A/D pins being recognized as analog input pins, resulting in the port value being read as a logic ‘0’, which may affect user application functionality.

## 2.8 Unused I/Os

Unused I/O pins should be configured as outputs and driven to a logic low state. Alternatively, connect a 1 k $\Omega$  to 10 k $\Omega$  resistor to Vss on unused pins and drive the output to logic low.

# PIC24FJ64GA004 FAMILY

## 4.2.2 DATA MEMORY ORGANIZATION AND ALIGNMENT

To maintain backward compatibility with PIC® devices and improve data space memory usage efficiency, the PIC24F instruction set supports both word and byte operations. As a consequence of byte accessibility, all Effective Address (EA) calculations are internally scaled to step through word-aligned memory. For example, the core recognizes that Post-Modified Register Indirect Addressing mode [Ws++] will result in a value of Ws + 1 for byte operations and Ws + 2 for word operations.

Data byte reads will read the complete word which contains the byte, using the LSb of any EA to determine which byte to select. The selected byte is placed onto the LSB of the data path. That is, data memory and registers are organized as two parallel, byte-wide entities with shared (word) address decode but separate write lines. Data byte writes only write to the corresponding side of the array or register which matches the byte address.

All word accesses must be aligned to an even address. Misaligned word data fetches are not supported, so care must be taken when mixing byte and word operations, or translating from 8-bit MCU code. If a misaligned read or write is attempted, an address error trap will be generated. If the error occurred on a read, the instruction underway is completed; if it occurred on a write, the instruction will be executed but the write will not occur. In either case, a trap is then executed, allowing the system and/or user to examine the machine state prior to execution of the address Fault.

All byte loads into any W register are loaded into the Least Significant Byte. The Most Significant Byte is not modified.

A Sign-Extend (SE) instruction is provided to allow users to translate 8-bit signed data to 16-bit signed values. Alternatively, for 16-bit unsigned data, users can clear the MSB of any W register by executing a Zero-Extend (ZE) instruction on the appropriate address.

Although most instructions are capable of operating on word or byte data sizes, it should be noted that some instructions operate only on words.

## 4.2.3 NEAR DATA SPACE

The 8-Kbyte area between 0000h and 1FFFh is referred to as the Near Data Space. Locations in this space are directly addressable via a 13-bit absolute address field within all memory direct instructions. The remainder of the data space is addressable indirectly. Additionally, the whole data space is addressable using MOV instructions, which support Memory Direct Addressing with a 16-bit address field.

## 4.2.4 SFR SPACE

The first 2 Kbytes of the Near Data Space, from 0000h to 07FFh, are primarily occupied with Special Function Registers (SFRs). These are used by the PIC24F core and peripheral modules for controlling the operation of the device.

SFRs are distributed among the modules that they control and are generally grouped together by module. Much of the SFR space contains unused addresses; these are read as '0'. A diagram of the SFR space, showing where SFRs are actually implemented, is shown in Table 4-2. Each implemented area indicates a 32-byte region where at least one address is implemented as an SFR. A complete listing of implemented SFRs, including their addresses, is shown in Tables 4-3 through 4-24.

**TABLE 4-2: IMPLEMENTED REGIONS OF SFR DATA SPACE**

SFR Space Address								
	xx00	xx20	xx40	xx60	xx80	xxA0	xxC0	xxE0
000h	Core			ICN	Interrupts			—
100h	Timers		Capture	—	Compare	—	—	—
200h	I <sup>2</sup> C™	UART	SPI		—	—	I/O	
300h	A/D		—	—	—	—	—	—
400h	—	—	—	—	—	—	—	—
500h	—	—	—	—	—	—	—	—
600h	PMP	RTC/Comp	CRC	—	PPS			
700h	—	—	System	NVM/PMD	—	—	—	—

**Legend:** — = No implemented SFRs in this block

# PIC24FJ64GA004 FAMILY

## REGISTER 5-1: NVMCON: FLASH MEMORY CONTROL REGISTER

R/SO-0	R/W-0	R/W-0	U-0	U-0	U-0	U-0	U-0
WR	WREN	WRERR	—	—	—	—	—
bit 15							bit 8

U-0	R/W-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0
—	ERASE	—	—	NVMOP3 <sup>(1)</sup>	NVMOP2 <sup>(1)</sup>	NVMOP1 <sup>(1)</sup>	NVMOP0 <sup>(1)</sup>
bit 7							bit 0

<b>Legend:</b>	SO = Settable Only bit		
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      **WR:** Write Control bit  
1 = Initiates a Flash memory program or erase operation; the operation is self-timed and the bit is cleared by hardware once operation is complete  
0 = Program or erase operation is complete and inactive
- bit 14      **WREN:** Write Enable bit  
1 = Enables Flash program/erase operations  
0 = Inhibits Flash program/erase operations
- bit 13      **WRERR:** Write Sequence Error Flag bit  
1 = An improper program or erase sequence attempt or termination has occurred (bit is set automatically on any set attempt of the WR bit)  
0 = The program or erase operation completed normally
- bit 12-7    **Unimplemented:** Read as '0'
- bit 6        **ERASE:** Erase/Program Enable bit  
1 = Performs the erase operation specified by the NVMOP<3:0> bits on the next WR command  
0 = Performs the program operation specified by the NVMOP<3:0> bits on the next WR command
- bit 5-4     **Unimplemented:** Read as '0'
- bit 3-0     **NVMOP<3:0>:** NVM Operation Select bits<sup>(1)</sup>  
1111 = Memory bulk erase operation (ERASE = 1) or no operation (ERASE = 0)<sup>(2)</sup>  
0011 = Memory word program operation (ERASE = 0) or no operation (ERASE = 1)  
0010 = Memory page erase operation (ERASE = 1) or no operation (ERASE = 0)  
0001 = Memory row program operation (ERASE = 0) or no operation (ERASE = 1)

- Note 1:** All other combinations of NVMOP<3:0> are unimplemented.  
**2:** Available in ICSP™ mode only. Refer to the device programming specifications.

# PIC24FJ64GA004 FAMILY

## 5.5.1 PROGRAMMING ALGORITHM FOR FLASH PROGRAM MEMORY

The user can program one row of Flash program memory at a time. To do this, it is necessary to erase the 8-row erase block containing the desired row. The general process is:

1. Read eight rows of program memory (512 instructions) and store in data RAM.
2. Update the program data in RAM with the desired new data.
3. Erase the block (see Example 5-1):
  - a) Set the NVMOPx bits (NVMCON<3:0>) to '0010' to configure for block erase. Set the ERASE (NVMCON<6>) and WREN (NVMCON<14>) bits.
  - b) Write the starting address of the block to be erased into the TBLPAG and W registers.
  - c) Write 55h to NVMKEY.
  - d) Write AAh to NVMKEY.
  - e) Set the WR bit (NVMCON<15>). The erase cycle begins and the CPU stalls for the duration of the erase cycle. When the erase is done, the WR bit is cleared automatically.
4. Write the first 64 instructions from data RAM into the program memory buffers (see Example 5-1).
5. Write the program block to Flash memory:
  - a) Set the NVMOPx bits to '0001' to configure for row programming. Clear the ERASE bit and set the WREN bit.
  - b) Write 55h to NVMKEY.
  - c) Write AAh to NVMKEY.
  - d) Set the WR bit. The programming cycle begins and the CPU stalls for the duration of the write cycle. When the write to Flash memory is done, the WR bit is cleared automatically.
6. Repeat Steps 4 and 5, using the next available 64 instructions from the block in data RAM by incrementing the value in TBLPAG, until all 512 instructions are written back to Flash memory.

For protection against accidental operations, 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, as shown in Example 5-3.

### EXAMPLE 5-1: ERASING A PROGRAM MEMORY BLOCK

```
; Set up NVMCON for block erase operation
MOV    #0x4042, W0          ;
MOV     W0, NVMCON           ; Initialize NVMCON
; Init pointer to row to be ERASED
MOV     #tblpage(PROG_ADDR), W0 ;
MOV     W0, TBLPAG           ; Initialize PM Page Boundary SFR
MOV     #tbloffset(PROG_ADDR), W0 ; Initialize in-page EA[15:0] pointer
TBLWTL  W0, [W0]             ; Set base address of erase block
DISI    #5                   ; Block all interrupts with priority <7
                                ; for next 5 instructions

MOV     #0x55, W0
MOV     W0, NVMKEY           ; Write the 55 key
MOV     #0xAA, W1
MOV     W1, NVMKEY           ; Write the AA key
BSET    NVMCON, #WR          ; Start the erase sequence
NOP                                           ; Insert two NOPs after the erase
NOP                                           ; command is asserted
```

# PIC24FJ64GA004 FAMILY

## 8.0 OSCILLATOR CONFIGURATION

**Note:** This data sheet summarizes the features of this group of PIC24F devices. It is not intended to be a comprehensive reference source. For more information, refer to the "PIC24F Family Reference Manual", "Oscillator" (DS39700).

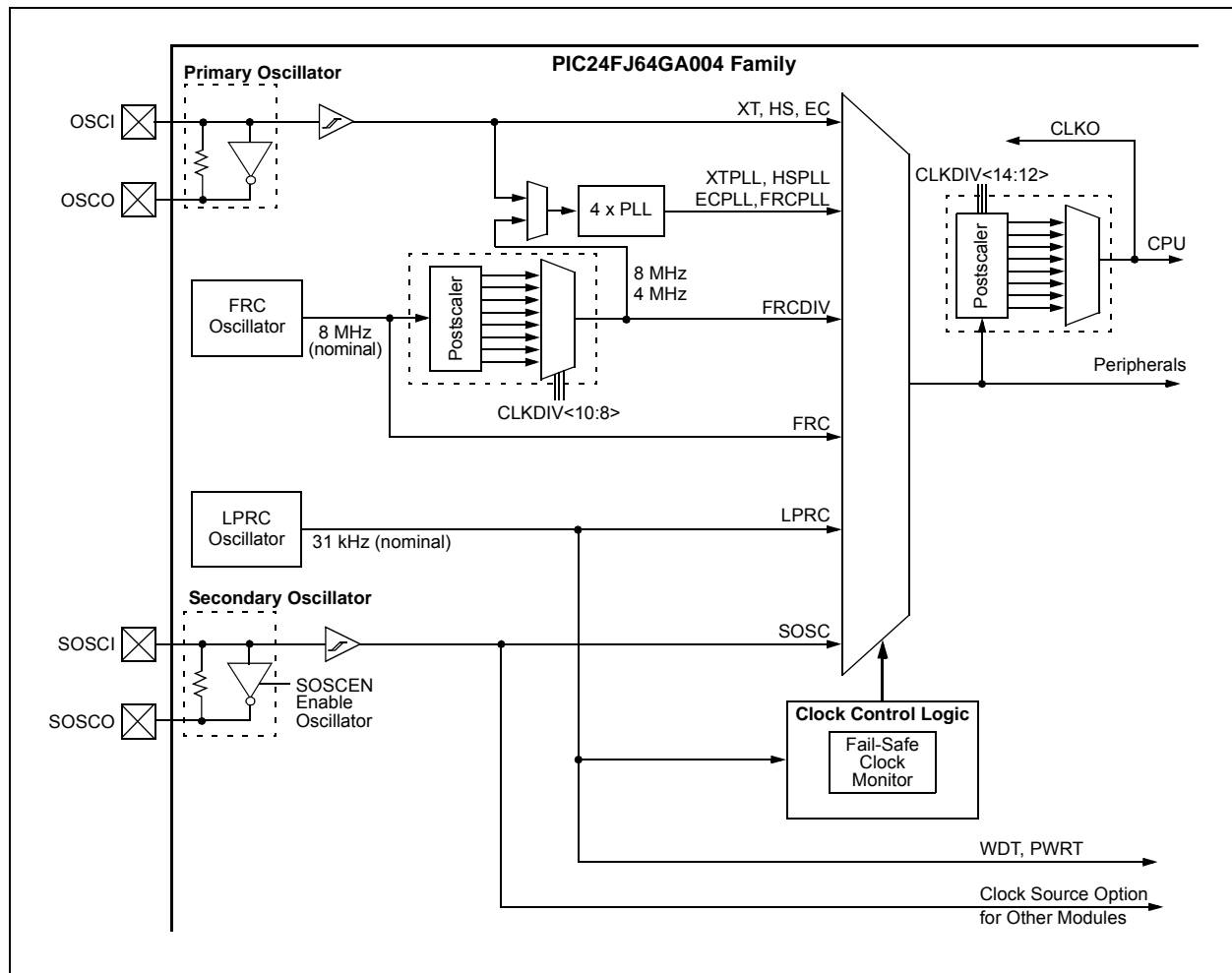
- Software-controllable switching between various clock sources
- Software-controllable postscaler for selective clocking of CPU for system power savings
- A Fail-Safe Clock Monitor (FSCM) that detects clock failure and permits safe application recovery or shutdown

A simplified diagram of the oscillator system is shown in Figure 8-1.

The oscillator system for PIC24FJ64GA004 family devices has the following features:

- A total of four external and internal oscillator options as clock sources, providing 11 different clock modes
- On-chip 4x PLL to boost internal operating frequency on select internal and external oscillator sources

**FIGURE 8-1: PIC24FJ64GA004 FAMILY CLOCK DIAGRAM**



# PIC24FJ64GA004 FAMILY

## REGISTER 8-3: OSCTUN: FRC OSCILLATOR TUNE REGISTER

U-0	U-0	U-0	U-0	U-0	U-0	U-0	U-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
—	—	TUN5 <sup>(1)</sup>	TUN4 <sup>(1)</sup>	TUN3 <sup>(1)</sup>	TUN2 <sup>(1)</sup>	TUN1 <sup>(1)</sup>	TUN0 <sup>(1)</sup>
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 15-6 **Unimplemented:** Read as '0'

bit 5-0 **TUN<5:0>:** FRC Oscillator Tuning bits<sup>(1)</sup>

011111 = Maximum frequency deviation

011110 =

•

•

•

000001 =

000000 = Center frequency, oscillator is running at factory calibrated frequency

111111 =

•

•

•

100001 =

100000 = Minimum frequency deviation

**Note 1:** Increments or decrements of TUN<5:0> may not change the FRC frequency in equal steps over the FRC tuning range and may not be monotonic.

## 8.4 Clock Switching Operation

With few limitations, applications are free to switch between any of the four clock sources (POSC, SOSC, FRC and LPRC) under software control and at any time. To limit the possible side effects that could result from this flexibility, PIC24F devices have a safeguard lock built into the switching process.

**Note:** The Primary Oscillator mode has three different submodes (XT, HS and EC) which are determined by the POSCMDx Configuration bits. While an application can switch to and from Primary Oscillator mode in software, it cannot switch between the different primary submodes without reprogramming the device.

### 8.4.1 ENABLING CLOCK SWITCHING

To enable clock switching, the FCKSM1 Configuration bit in Flash Configuration Word 2 must be programmed to '0'. (Refer to **Section 24.1 "Configuration Bits"** for further details.) If the FCKSM1 Configuration bit is unprogrammed ('1'), the clock switching function and Fail-Safe Clock Monitor function are disabled. This is the default setting.

The NOSCx control bits (OSCCON<10:8>) do not control the clock selection when clock switching is disabled. However, the COSCx bits (OSCCON<14:12>) will reflect the clock source selected by the FNOSCx Configuration bits.

The OSWEN control bit (OSCCON<0>) has no effect when clock switching is disabled; it is held at '0' at all times.

# PIC24FJ64GA004 FAMILY

## REGISTER 10-23: RPOR8: PERIPHERAL PIN SELECT OUTPUT REGISTER 8

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	RP17R4 <sup>(1)</sup>	RP17R3 <sup>(1)</sup>	RP17R2 <sup>(1)</sup>	RP17R1 <sup>(1)</sup>	RP17R0 <sup>(1)</sup>
bit 15							bit 8

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	RP16R4 <sup>(1)</sup>	RP16R3 <sup>(1)</sup>	RP16R2 <sup>(1)</sup>	RP16R1 <sup>(1)</sup>	RP16R0 <sup>(1)</sup>
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 **Unimplemented:** Read as '0'

bit 12-8 **RP17R<4:0>:** Peripheral Output Function is Assigned to RP17 Output Pin bits<sup>(1)</sup>  
(see Table 10-3 for peripheral function numbers)

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

bit 4-0 **RP16R<4:0>:** Peripheral Output Function is Assigned to RP16 Output Pin bits<sup>(1)</sup>  
(see Table 10-3 for peripheral function numbers)

**Note 1:** These bits are only available on the 44-pin devices; otherwise, they read as '0'.

## REGISTER 10-24: RPOR9: PERIPHERAL PIN SELECT OUTPUT REGISTER 9

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	RP19R4 <sup>(1)</sup>	RP19R3 <sup>(1)</sup>	RP19R2 <sup>(1)</sup>	RP19R1 <sup>(1)</sup>	RP19R0 <sup>(1)</sup>
bit 15							bit 8

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	RP18R4 <sup>(1)</sup>	RP18R3 <sup>(1)</sup>	RP18R2 <sup>(1)</sup>	RP18R1 <sup>(1)</sup>	RP18R0 <sup>(1)</sup>
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 **Unimplemented:** Read as '0'

bit 12-8 **RP19R<4:0>:** Peripheral Output Function is Assigned to RP19 Output Pin bits<sup>(1)</sup>  
(see Table 10-3 for peripheral function numbers)

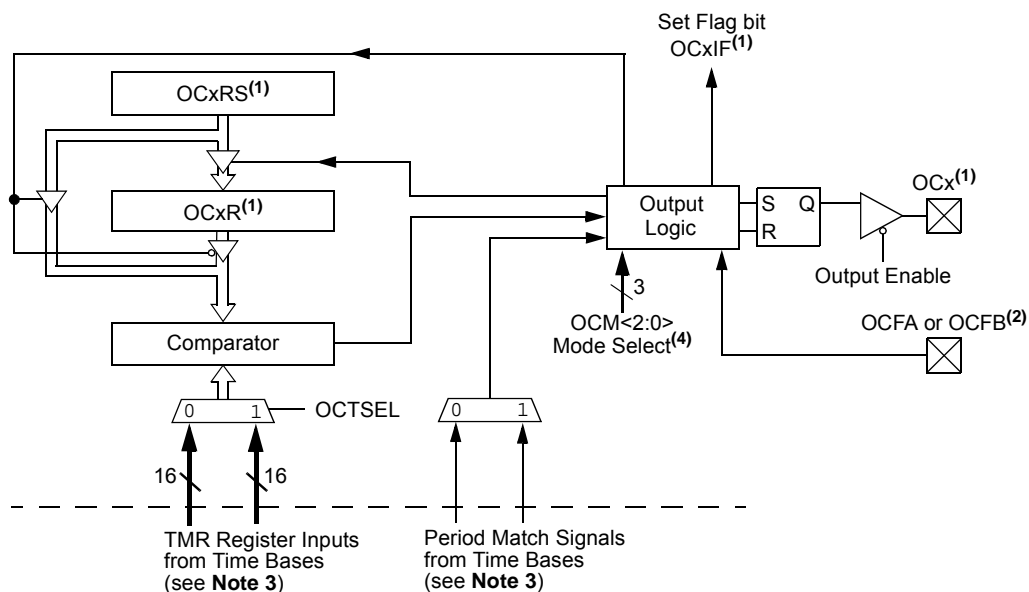
bit 7-5 **Unimplemented:** Read as '0'

bit 4-0 **RP18R<4:0>:** Peripheral Output Function is Assigned to RP18 Output Pin bits<sup>(1)</sup>  
(see Table 10-3 for peripheral function numbers)

**Note 1:** These bits are only available on the 44-pin devices; otherwise, they read as '0'.

# PIC24FJ64GA004 FAMILY

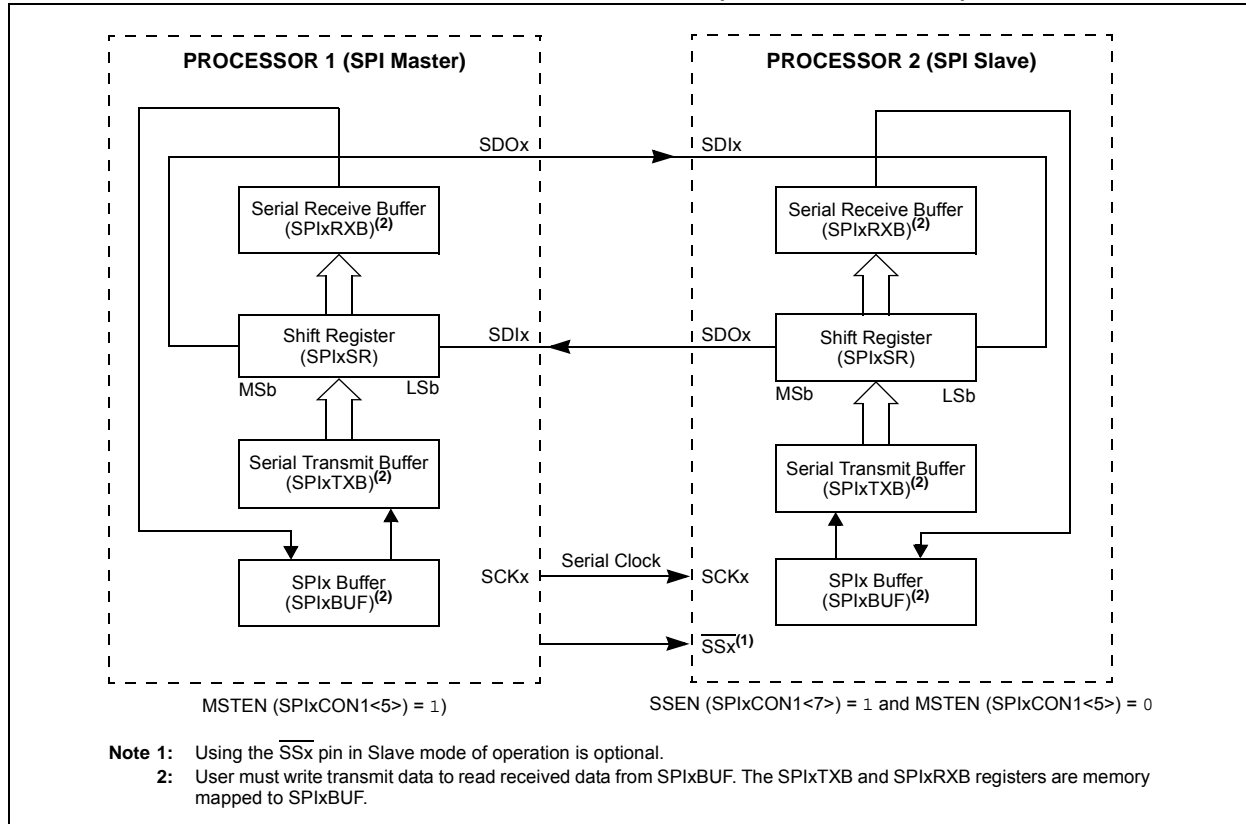
**FIGURE 14-1: OUTPUT COMPARE x MODULE BLOCK DIAGRAM**



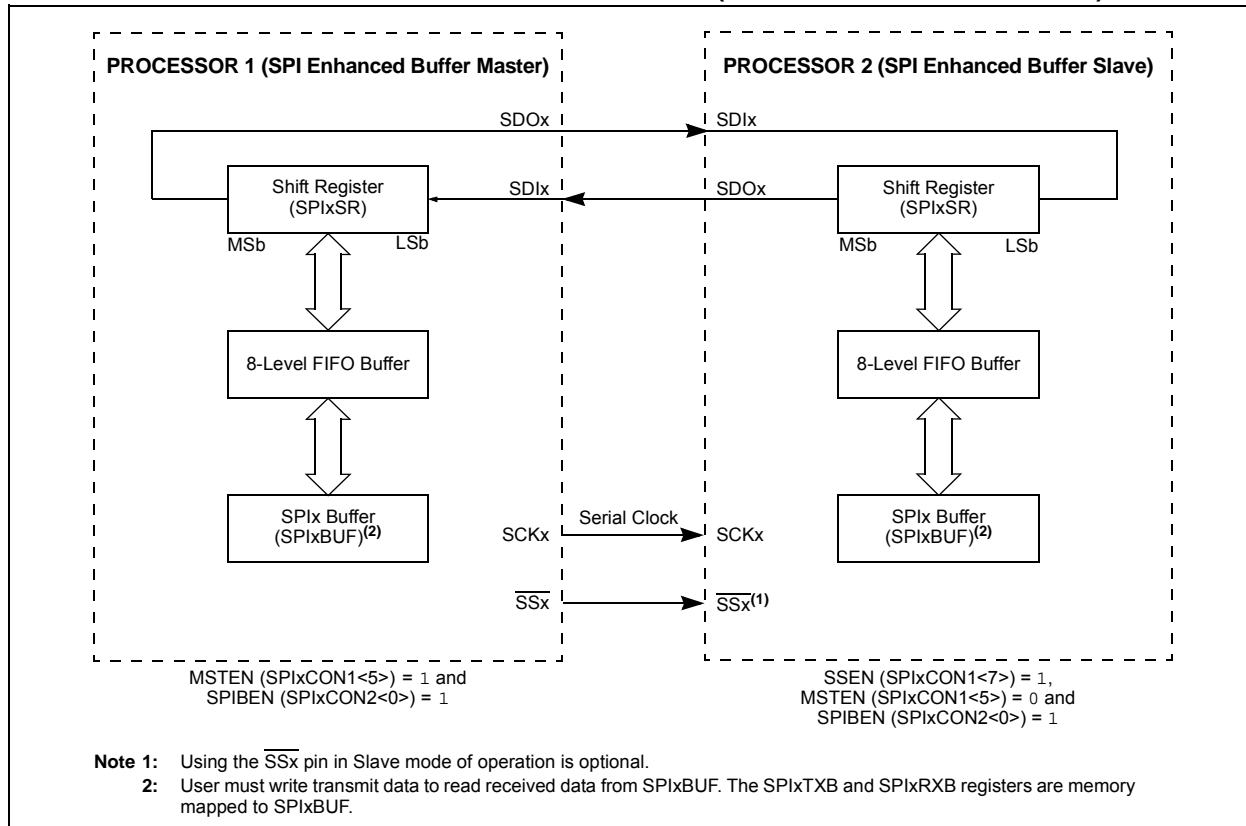
- Note 1:** Where 'x' is shown, reference is made to the registers associated with the respective Output Compare Channels 1 through 5.
- Note 2:** The OCFA pin controls the OC1-OC4 channels. The OCFB pin controls the OC5 channel.
- Note 3:** Each output compare channel can use one of two selectable time bases. Refer to the device data sheet for the time bases associated with the module.
- Note 4:** This peripheral's inputs and outputs must be assigned to an available RPN pin before use. Please see **Section 10.4 "Peripheral Pin Select (PPS)"** for more information.

# PIC24FJ64GA004 FAMILY

**FIGURE 15-3: SPIx MASTER/SLAVE CONNECTION (STANDARD MODE)**

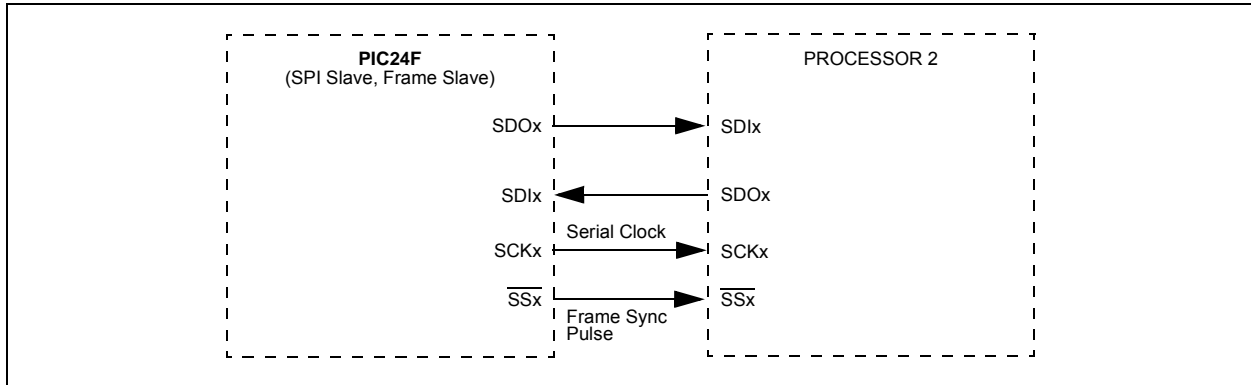


**FIGURE 15-4: SPIx MASTER/SLAVE CONNECTION (ENHANCED BUFFER MODES)**

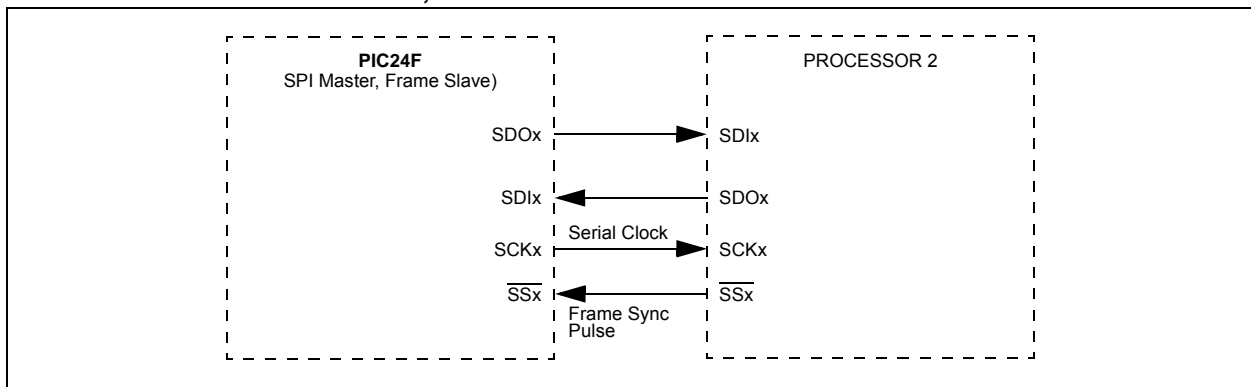


# PIC24FJ64GA004 FAMILY

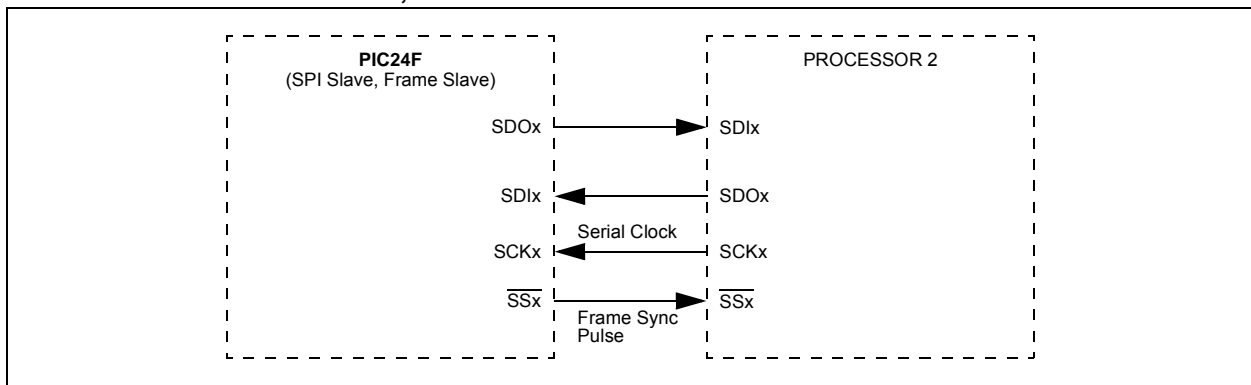
**FIGURE 15-5: SPIx MASTER, FRAME MASTER CONNECTION DIAGRAM**



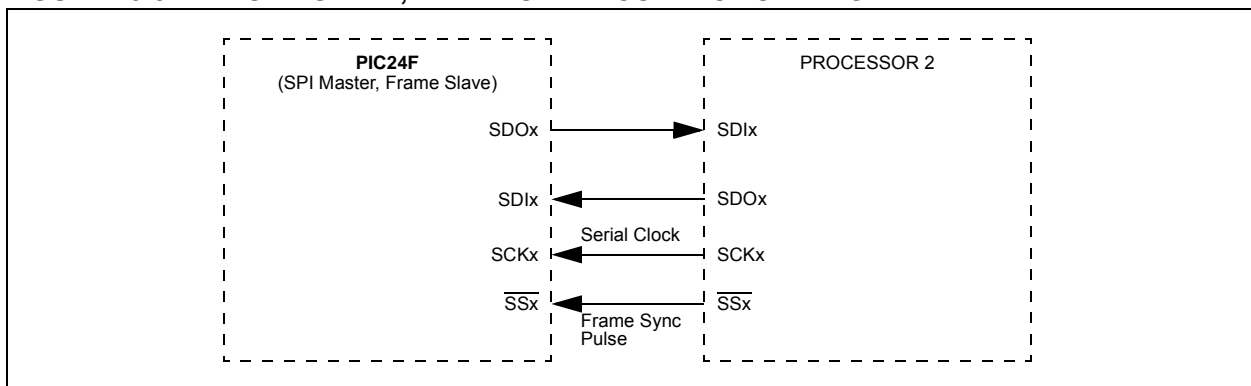
**FIGURE 15-6: SPIx MASTER, FRAME SLAVE CONNECTION DIAGRAM**



**FIGURE 15-7: SPIx SLAVE, FRAME MASTER CONNECTION DIAGRAM**



**FIGURE 15-8: SPIx SLAVE, FRAME SLAVE CONNECTION DIAGRAM**



# PIC24FJ64GA004 FAMILY

## 17.1 UARTx Baud Rate Generator (BRG)

The UARTx module includes a dedicated 16-bit Baud Rate Generator. The UxBRG register controls the period of a free-running, 16-bit timer. Equation 17-1 shows the formula for computation of the baud rate with BRGH = 0.

### EQUATION 17-1: UARTx BAUD RATE WITH BRGH = 0<sup>(1)</sup>

$$\text{Baud Rate} = \frac{\text{FCY}}{16 \cdot (\text{UxBRG} + 1)}$$

$$\text{UxBRG} = \frac{\text{FCY}}{16 \cdot \text{Baud Rate}} - 1$$

**Note 1:** Based on FCY = FOSC/2; Doze mode and PLL are disabled.

Example 17-1 shows the calculation of the baud rate error for the following conditions:

- FCY = 4 MHz
- Desired Baud Rate = 9600

The maximum baud rate (BRGH = 0) possible is FCY/16 (for UxBRG = 0) and the minimum baud rate possible is FCY/(16 \* 65536).

Equation 17-2 shows the formula for computation of the baud rate with BRGH = 1.

### EQUATION 17-2: UARTx BAUD RATE WITH BRGH = 1<sup>(1)</sup>

$$\text{Baud Rate} = \frac{\text{FCY}}{4 \cdot (\text{UxBRG} + 1)}$$

$$\text{UxBRG} = \frac{\text{FCY}}{4 \cdot \text{Baud Rate}} - 1$$

**Note 1:** Based on FCY = FOSC/2; Doze mode and PLL are disabled.

The maximum baud rate (BRGH = 1) possible is FCY/4 (for UxBRG = 0) and the minimum baud rate possible is FCY/(4 \* 65536).

Writing a new value to the UxBRG register causes the BRG timer to be reset (cleared). This ensures the BRG does not wait for a timer overflow before generating the new baud rate.

### EXAMPLE 17-1: BAUD RATE ERROR CALCULATION (BRGH = 0)<sup>(1)</sup>

$$\text{Desired Baud Rate} = \text{FCY}/(16 (\text{UxBRG} + 1))$$

Solving for UxBRG value:

$$\text{UxBRG} = ((\text{FCY}/\text{Desired Baud Rate})/16) - 1$$

$$\text{UxBRG} = ((4000000/9600)/16) - 1$$

$$\text{UxBRG} = 25$$

$$\begin{aligned} \text{Calculated Baud Rate} &= 4000000/(16 (25 + 1)) \\ &= 9615 \end{aligned}$$

$$\begin{aligned} \text{Error} &= (\text{Calculated Baud Rate} - \text{Desired Baud Rate}) / \text{Desired Baud Rate} \\ &= (9615 - 9600)/9600 \\ &= 0.16\% \end{aligned}$$

**Note 1:** Based on FCY = FOSC/2; Doze mode and PLL are disabled.

# PIC24FJ64GA004 FAMILY

## 19.1 RTCC Module Registers

The RTCC module registers are organized into three categories:

- RTCC Control Registers
- RTCC Value Registers
- Alarm Value Registers

### 19.1.1 REGISTER MAPPING

To limit the register interface, the RTCC Timer and Alarm Time registers are accessed through corresponding register pointers. The RTCC Value register window (RTCVALH and RTCVALL) uses the RTCPTR<1:0> bits (RCFGCAL<9:8>) to select the desired Timer register pair (see Table 19-1).

By writing the RTCVALH byte, the RTCC Pointer value (the RTCPTR<1:0> bits) decrements by one until the bits reach '00'. Once they reach '00', the MINUTES and SECONDS value will be accessible through RTCVALH and RTCVALL until the pointer value is manually changed.

**TABLE 19-1: RTCVAL REGISTER MAPPING**

RTCPTR <1:0>	RTCC Value Register Window	
	RTCVAL<15:8>	RTCVAL<7:0>
00	MINUTES	SECONDS
01	WEEKDAY	HOURS
10	MONTH	DAY
11	—	YEAR

The Alarm Value register window (ALRMVALH and ALRMVALL) uses the ALRMPTR bits (ALCFG RPT<9:8>) to select the desired Alarm register pair (see Table 19-2).

By writing the ALRMVALH byte, the Alarm Pointer value (the ALRMPTR<1:0> bits) decrements by one until the bits reach '00'. Once they reach '00', the ALRMMIN and ALRMSEC value will be accessible through ALRMVALH and ALRMVALL until the pointer value is manually changed.

### EXAMPLE 19-1: SETTING THE RTCWREN BIT

```
asm volatile("push w7");
asm volatile("push w8");
asm volatile("disi #5");
asm volatile("mov #0x55, w7");
asm volatile("mov w7, _NVMKEY");
asm volatile("mov #0xAA, w8");
asm volatile("mov w8, _NVMKEY");
asm volatile("bset _RCFGCAL, #13"); //set the RTCWREN bit
asm volatile("pop w8");
asm volatile("pop w7");
```

**TABLE 19-2: ALRMVAL REGISTER MAPPING**

ALRMPTR <1:0>	Alarm Value Register Window	
	ALRMVAL<15:8>	ALRMVAL<7:0>
00	ALRMMIN	ALRMSEC
01	ALRMWD	ALRMHR
10	ALRMMNTH	ALRMDAY
11	—	—

Considering that the 16-bit core does not distinguish between 8-bit and 16-bit read operations, the user must be aware that when reading either the ALRMVALH or ALRMVALL, the bytes will decrement the ALRMPTR<1:0> value. The same applies to the RTCVALH or RTCVALL bytes with the RTCPTR<1:0> being decremented.

**Note:** This only applies to read operations and not write operations.

### 19.1.2 WRITE LOCK

In order to perform a write to any of the RTCC Timer registers, the RTCWREN bit (RCFGCAL<13>) must be set (refer to Example 19-1).

**Note:** To avoid accidental writes to the timer, it is recommended that the RTCWREN bit (RCFGCAL<13>) is kept clear at any other time. For the RTCWREN bit to be set, there is only 1 instruction cycle time window allowed between the 55h/AA sequence and the setting of RTCWREN; therefore, it is recommended that code follow the procedure in Example 19-1.

# PIC24FJ64GA004 FAMILY

## REGISTER 19-3: ALCFGRPT: ALARM CONFIGURATION REGISTER

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
ALRMEN	CHIME	AMASK3	AMASK2	AMASK1	AMASK0	ALRMPTR1	ALRMPTR0
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
ARPT7	ARPT6	ARPT5	ARPT4	ARPT3	ARPT2	ARPT1	ARPT0
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      **ALRMEN:** Alarm Enable bit  
 1 = Alarm is enabled (cleared automatically after an alarm event whenever ARPT<7:0> = 00h and CHIME = 0)  
 0 = Alarm is disabled
- bit 14      **CHIME:** Chime Enable bit  
 1 = Chime is enabled; ARPT<7:0> bits are allowed to roll over from 00h to FFh  
 0 = Chime is disabled; ARPT<7:0> bits stop once they reach 00h
- bit 13-10      **AMASK<3:0>:** Alarm Mask Configuration bits  
 0000 = Every half second  
 0001 = Every second  
 0010 = Every 10 seconds  
 0011 = Every minute  
 0100 = Every 10 minutes  
 0101 = Every hour  
 0110 = Once a day  
 0111 = Once a week  
 1000 = Once a month  
 1001 = Once a year (except when configured for February 29th, once every 4 years)  
 101x = Reserved; do not use  
 11xx = Reserved; do not use
- bit 9-8      **ALRMPTR<1:0>:** Alarm Value Register Window Pointer bits  
 Points to the corresponding Alarm Value registers when reading ALRMVALH and ALRMVALL registers; the ALRMPTR<1:0> value decrements on every read or write of ALRMVALH until it reaches '00'.  
ALRMVAL<15:8>:  
 00 = ALRMMIN  
 01 = ALRMWD  
 10 = ALRMMNTH  
 11 = Unimplemented  
ALRMVAL<7:0>:  
 00 = ALRMSEC  
 01 = ALRMHR  
 10 = ALRMDAY  
 11 = Unimplemented
- bit 7-0      **ARPT<7:0>:** Alarm Repeat Counter Value bits  
 11111111 = Alarm will repeat 255 more times  
 ...  
 00000000 = Alarm will not repeat  
 The counter decrements on any alarm event. The counter is prevented from rolling over from 00h to FFh unless CHIME = 1.

## 23.0 COMPARATOR VOLTAGE REFERENCE

**Note:** This data sheet summarizes the features of this group of PIC24F devices. It is not intended to be a comprehensive reference source. For more information, refer to the “PIC24F Family Reference Manual”, “Comparator Voltage Reference Module” (DS39709).

### 23.1 Configuring the Comparator Voltage Reference

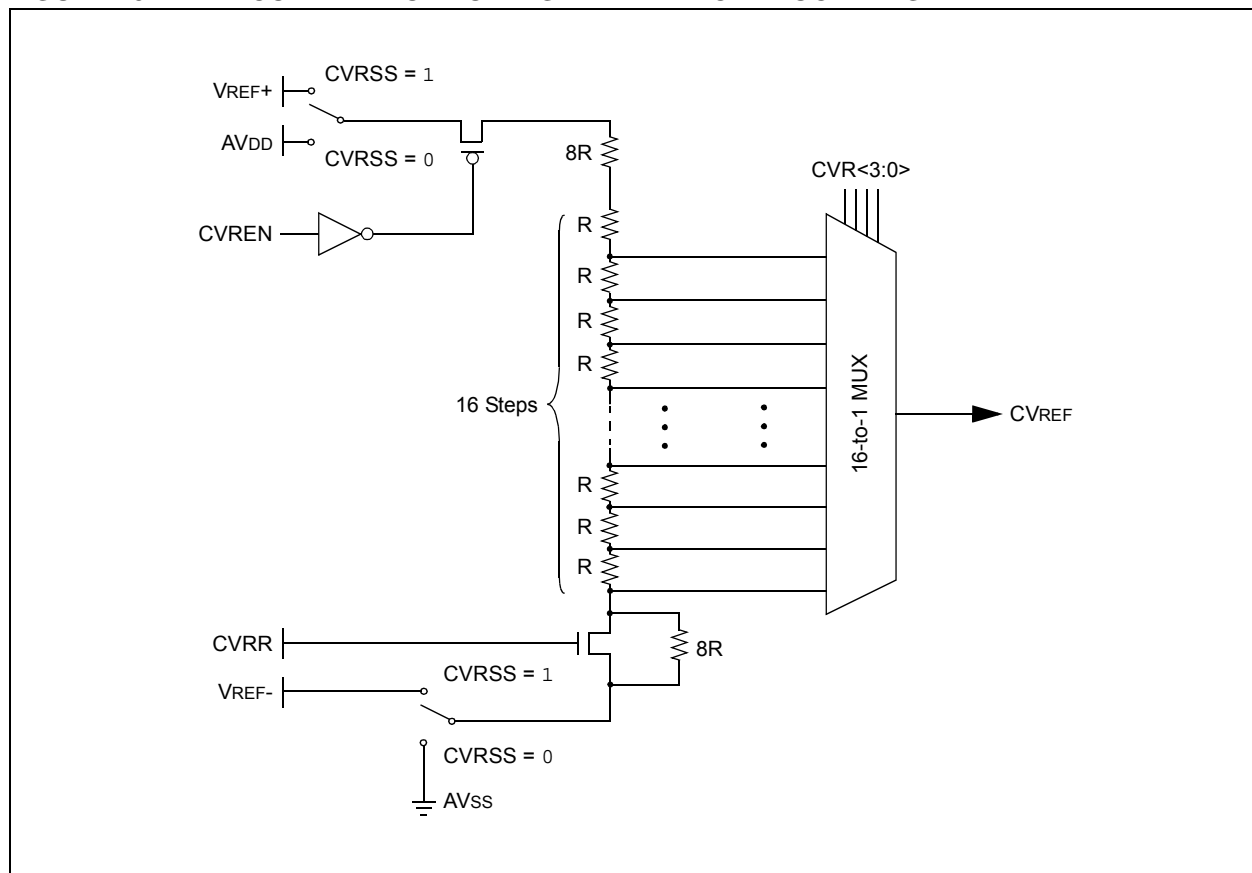
The comparator voltage reference module is controlled through the CVRCON register (Register 23-1). The comparator voltage reference provides two ranges of

output voltage, each with 16 distinct levels. The range to be used is selected by the CVRR bit (CVRCON<5>). The primary difference between the ranges is the size of the steps selected by the CVREF Selection bits (CVR<3:0>), with one range offering finer resolution.

The comparator reference supply voltage can come from either VDD and VSS, or the external VREF+ and VREF-. The voltage source is selected by the CVRSS bit (CVRCON<4>).

The settling time of the comparator voltage reference must be considered when changing the CVREF output.

**FIGURE 23-1: COMPARATOR VOLTAGE REFERENCE BLOCK DIAGRAM**



# PIC24FJ64GA004 FAMILY

## 24.0 SPECIAL FEATURES

**Note:** This data sheet summarizes the features of this group of PIC24F devices. It is not intended to be a comprehensive reference source. For more information, refer to the following sections of the “PIC24F Family Reference Manual”:

- “**Watchdog Timer (WDT)**” (DS39697)
- “**High-Level Device Integration**” (DS39719)
- “**Programming and Diagnostics**” (DS39716)

PIC24FJ64GA004 family devices include several features intended to maximize application flexibility and reliability, and minimize cost through elimination of external components. These are:

- Flexible Configuration
- Watchdog Timer (WDT)
- Code Protection
- JTAG Boundary Scan Interface
- In-Circuit Serial Programming
- In-Circuit Emulation

### 24.1 Configuration Bits

The Configuration bits can be programmed (read as ‘0’), or left unprogrammed (read as ‘1’), to select various device configurations. These bits are mapped starting at program memory location, F80000h. A complete list of locations is shown in Table 24-1. A detailed explanation of the various bit functions is provided in Register 24-1 through Register 24-4.

Note that address, F80000h, is beyond the user program memory space. In fact, it belongs to the configuration memory space (800000h-FFFFFFh), which can only be accessed using table reads and table writes.

#### 24.1.1 CONSIDERATIONS FOR CONFIGURING PIC24FJ64GA004 FAMILY DEVICES

In PIC24FJ64GA004 family devices, the configuration bytes are implemented as volatile memory. This means that configuration data must be programmed each time the device is powered up. Configuration data is stored in the two words at the top of the on-chip program memory space, known as the Flash Configuration Words. Their specific locations are shown in Table 24-1. These are packed representations of the actual device Configuration bits, whose actual locations are distributed among five locations in configuration space. The configuration data is automatically loaded from the Flash Configuration Words to the proper Configuration registers during device Resets.

**Note:** Configuration data is reloaded on all types of device Resets.

**TABLE 24-1: FLASH CONFIGURATION WORD LOCATIONS FOR PIC24FJ64GA004 FAMILY DEVICES**

Device	Configuration Word Addresses	
	1	2
PIC24FJ16GA	002BFEh	002BFCh
PIC24FJ32GA	0057FEh	0057FCh
PIC24FJ48GA	0083FEh	0083FCh
PIC24FJ64GA	00ABFEh	00ABFCh

When creating applications for these devices, users should always specifically allocate the location of the Flash Configuration Word for configuration data. This is to make certain that program code is not stored in this address when the code is compiled.

The Configuration bits are reloaded from the Flash Configuration Word on any device Reset.

The upper byte of both Flash Configuration Words in program memory should always be ‘1111 1111’. This makes them appear to be NOP instructions in the remote event that their locations are ever executed by accident. Since Configuration bits are not implemented in the corresponding locations, writing ‘1’s to these locations has no effect on device operation.

## 25.0 DEVELOPMENT SUPPORT

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

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

## 25.1 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

# PIC24FJ64GA004 FAMILY

**TABLE 27-4: DC CHARACTERISTICS: OPERATING CURRENT (IDD)**

DC CHARACTERISTICS			Standard Operating Conditions: 2.0V to 3.6V (unless otherwise stated)		
			Operating temperature -40°C ≤ TA ≤ +85°C for Industrial -40°C ≤ TA ≤ +125°C for Extended		
Parameter No.	Typical <sup>(1)</sup>	Max	Units	Conditions	
Operating Current (IDD): PMD Bits are Set <sup>(2)</sup>					
DC20	0.650	0.850	mA	-40°C	2.0V <sup>(3)</sup>  1 MIPS
DC20a	0.650	0.850	mA	+25°C	
DC20b	0.650	0.850	mA	+85°C	
DC20c	0.650	0.850	mA	+125°C	
DC20d	1.2	1.6	mA	-40°C	
DC20e	1.2	1.6	mA	+25°C	
DC20f	1.2	1.6	mA	+85°C	
DC20g	1.2	1.6	mA	+125°C	
DC23	2.6	3.4	mA	-40°C	2.0V <sup>(3)</sup>  4 MIPS
DC23a	2.6	3.4	mA	+25°C	
DC23b	2.6	3.4	mA	+85°C	
DC23c	2.6	3.4	mA	+125°C	
DC23d	4.1	5.4	mA	-40°C	
DC23e	4.1	5.4	mA	+25°C	
DC23f	4.1	5.4	mA	+85°C	
DC23g	4.1	5.4	mA	+125°C	
DC24	13.5	17.6	mA	-40°C	2.5V <sup>(3)</sup>  3.3V <sup>(4)</sup>  16 MIPS
DC24a	13.5	17.6	mA	+25°C	
DC24b	13.5	17.6	mA	+85°C	
DC24c	13.5	17.6	mA	+125°C	
DC24d	15	20	mA	-40°C	
DC24e	15	20	mA	+25°C	
DC24f	15	20	mA	+85°C	
DC24g	15	20	mA	+125°C	
DC31	13	17	μA	-40°C	2.0V <sup>(3)</sup>  3.3V <sup>(4)</sup>  LPRC (31 kHz)
DC31a	13	17	μA	+25°C	
DC31b	20	26	μA	+85°C	
DC31c	40	50	μA	+125°C	
DC31d	54	70	μA	-40°C	
DC31e	54	70	μA	+25°C	
DC31f	95	124	μA	+85°C	
DC31g	120	260	μA	+125°C	

**Note 1:** Data in "Typical" column is at 3.3V, +25°C unless otherwise stated. Parameters are for design guidance only and are not tested.

**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. The test conditions for all IDD measurements are as follows: OSC1 driven with external square wave from rail-to-rail. All I/O pins are configured as inputs and pulled to VDD.

MCLR = VDD; WDT and FSCM are disabled. CPU, SRAM, program memory and data memory are operational. No peripheral modules are operating and all of the Peripheral Module Disable (PMD) bits are set.

**3:** On-chip voltage regulator is disabled (DISVREG tied to VDD).

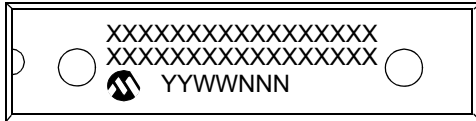
**4:** On-chip voltage regulator is enabled (DISVREG tied to VSS). Low-Voltage Detect (LVD) and Brown-out Detect (BOD) are enabled.

# PIC24FJ64GA004 FAMILY

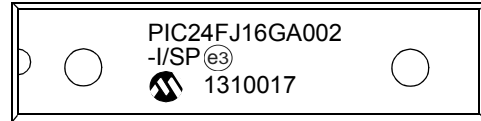
## 28.0 PACKAGING INFORMATION

### 28.1 Package Marking Information

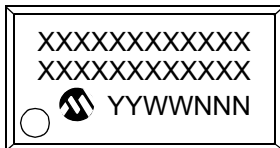
28-Lead SPDIP (.300")



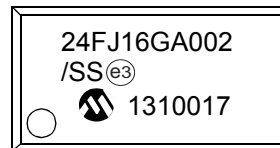
Example



28-Lead SSOP (5.30 mm)



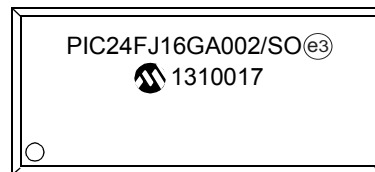
Example



28-Lead SOIC (7.50 mm)



Example



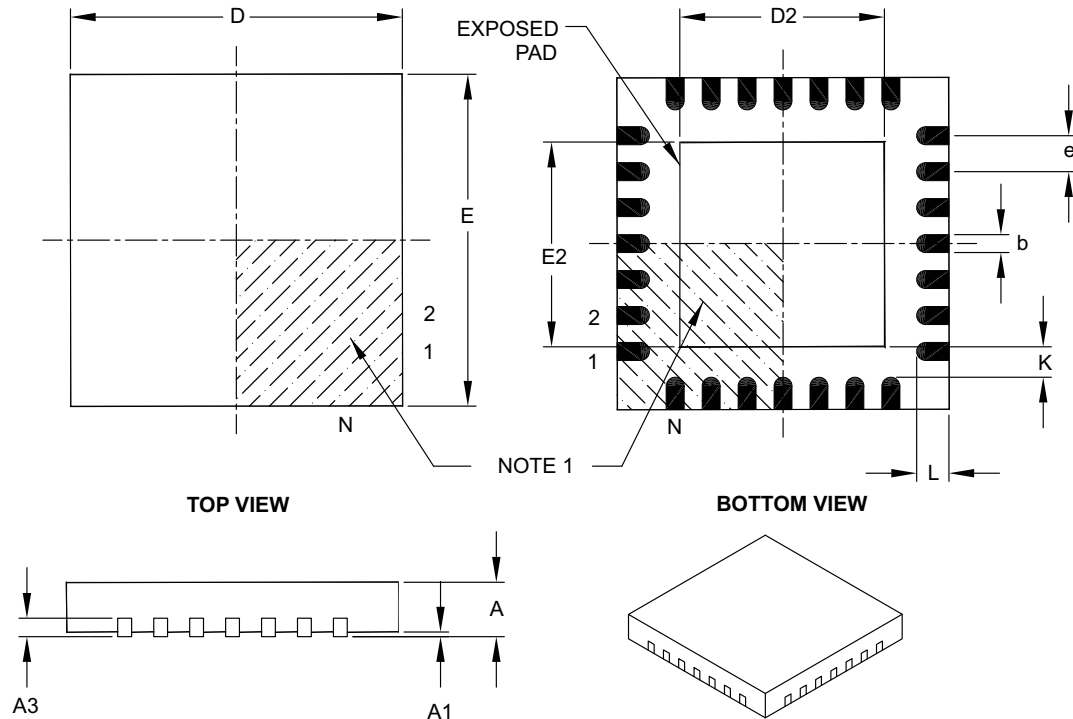
<b>Legend:</b>	XX...X	Customer-specific information
	Y	Year code (last digit of calendar year)
	YY	Year code (last 2 digits of calendar year)
	WW	Week code (week of January 1 is week '01')
	NNN	Alphanumeric traceability code
	*	Pb-free JEDEC designator for Matte Tin (Sn)
		This package is Pb-free. The Pb-free JEDEC designator (e3) can be found on the outer packaging for this package.

**Note:** In the event the full Microchip part number cannot be marked on one line, it will be carried over to the next line, thus limiting the number of available characters for customer-specific information.

# PIC24FJ64GA004 FAMILY

## 28-Lead Plastic Quad Flat, No Lead Package (ML) – 6x6 mm Body [QFN] with 0.55 mm Contact Length

**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	0.80	0.90	1.00
Standoff	A1	0.00	0.02	0.05
Contact Thickness	A3	0.20 REF		
Overall Width	E	6.00 BSC		
Exposed Pad Width	E2	3.65	3.70	4.20
Overall Length	D	6.00 BSC		
Exposed Pad Length	D2	3.65	3.70	4.20
Contact Width	b	0.23	0.30	0.35
Contact Length	L	0.50	0.55	0.70
Contact-to-Exposed Pad	K	0.20	–	–

**Notes:**

- Pin 1 visual index feature may vary, but must be located within the hatched area.
- Package is saw singulated.
- Dimensioning and tolerancing per ASME Y14.5M.

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

REF: Reference Dimension, usually without tolerance, for information purposes only.

Microchip Technology Drawing C04-105B

## THE MICROCHIP WEB SITE

Microchip provides online support via our WWW site at [www.microchip.com](http://www.microchip.com). This web site is used as a means to make files and information easily available to customers. Accessible by using your favorite Internet browser, the web site contains the following information:

- **Product Support** – Data sheets and errata, application notes and sample programs, design resources, user's guides and hardware support documents, latest software releases and archived software
- **General Technical Support** – Frequently Asked Questions (FAQ), technical support requests, online discussion groups, Microchip consultant program member listing
- **Business of Microchip** – Product selector and ordering guides, latest Microchip press releases, listing of seminars and events, listings of Microchip sales offices, distributors and factory representatives

## CUSTOMER CHANGE NOTIFICATION SERVICE

Microchip's customer notification service helps keep customers current on Microchip products. Subscribers will receive e-mail notification whenever there are changes, updates, revisions or errata related to a specified product family or development tool of interest.

To register, access the Microchip web site at [www.microchip.com](http://www.microchip.com). Under "Support", click on "Customer Change Notification" and follow the registration instructions.

## CUSTOMER SUPPORT

Users of Microchip products can receive assistance through several channels:

- Distributor or Representative
- Local Sales Office
- Field Application Engineer (FAE)
- Technical Support
- Development Systems Information Line

Customers should contact their distributor, representative or field application engineer (FAE) for support. Local sales offices are also available to help customers. A listing of sales offices and locations is included in the back of this document.

**Technical support is available through the web site at: <http://microchip.com/support>**