

---

## Section 5. Data EEPROM

---

### HIGHLIGHTS

This section of the manual contains the following major topics:

5.1	Introduction .....	5-2
5.2	Control Registers .....	5-2
5.3	Data EEPROM Operations .....	5-4
5.4	Register Map.....	5-9
5.5	Design Tips .....	5-10
5.6	Related Application Notes.....	5-11
5.7	Revision History .....	5-12

## 5.1 INTRODUCTION

In addition to standard Flash-based program memory and volatile data RAM, certain PIC24F devices also include on-chip data EEPROM. This memory block allows users to store program or application information (such as identification, calibration constants, etc.) in a nonvolatile location, and easily rewrite the information when needed. Data EEPROM memory is based on the same Flash technology as program memory, and has been optimized for both long retention and a higher number of erase/write cycles.

The data EEPROM is mapped to the top of the user program memory space, with the top address at program memory address, 7FFFFFFh. The size of the EEPROM is device dependent. Refer to the specific device data sheet for further information.

The EEPROM is organized as 16-bit wide memory. Each word is directly addressable; unlike program memory, there are no “phantom bytes” in odd-numbered addresses.

The programming techniques used for the data EEPROM are similar to those used for Flash program memory RTSP, discussed in previous sections. The key differences between Flash and data EEPROM programming operations are the amount of data that can be programmed or erased, during each program/erase cycle, and that table writes to the data EEPROM do not suspend operation of the CPU.

## 5.2 CONTROL REGISTERS

Like Flash program memory, data EEPROM programming operations are controlled using the three Nonvolatile Memory (NVM) Control registers:

- NVMCON: Nonvolatile Memory Control Register
- NVMKEY: Nonvolatile Memory Key Register
- NVMADR: Nonvolatile Memory Address Register

### 5.2.1 NVMCON Register

The NVMCON register ([Register 5-1](#)) is the primary control register for data EEPROM program/erase operations. This upper byte contains the control bits used to start the program or erase cycle, and the flag bit to indicate if the operation was successfully performed. The lower byte of NVMCON configures the type of NVM operation that will be performed.

The NVMCON register also controls programming/erase operations for the program memory, as described in [Section 4. “Program Memory”](#).

### 5.2.2 NVMKEY Register

NVMKEY is a write-only register that is used to prevent accidental writes or erasures of data EEPROM locations. To start any programming or erase sequence, these two instructions must be executed first, in the exact order shown:

1. Write 55h to NVMKEY.
2. Write AAh to NVMKEY.

After this sequence, a write will be allowed to the NVMCON register for one instruction cycle. In most cases, the user will simply need to set the WR bit in the NVMCON register to start the program or erase cycle. Interrupts should be disabled during the unlock sequence.

The MPLAB® C30 C compiler provides a defined library procedure (`builtin_write_NVM`) to perform the unlock sequence. [Example 5-1](#) shows how the unlock sequence can be performed with in-line assembly.

#### Example 5-1: Data EEPROM Unlock Sequence

```
//Disable Interrupts For 5 instructions
asm volatile("disi #5");
//Issue Unlock Sequence
asm volatile("mov #0x55, W0    \n"
             "mov W0, NVMKEY    \n"
             "mov #0xAA, W1     \n"
             "mov W1, NVMKEY    \n");
```

**Register 5-1: NVMCON: Nonvolatile Memory Control Register (Data EEPROM Operations)**

R/S-0	R/W-0	R/W-0	R/W-0	U-0	U-0	U-0	U-0
WR	WREN	WRERR	PGMONLY	r	r	r	r
bit 15				bit 8			
U-0	R/W-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
r	ERASE	r	NVMOP4 <sup>(1)</sup>	NVMOP3 <sup>(1)</sup>	NVMOP2 <sup>(1)</sup>	NVMOP1 <sup>(1)</sup>	NVMOP0 <sup>(1)</sup>
bit 7				bit 0			

<b>Legend:</b>	r = Reserved	U = Unimplemented bit, read as '0'
R = Readable bit	W = Writable bit	S = Settable bit
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared
		x = Bit is unknown

- bit 15 **WR:** Write (Program or Erase) Control bit  
1 = Initiates a data EEPROM erase or write cycle (can be set, but not cleared in software)  
0 = Write cycle is complete (cleared automatically by hardware)
- bit 14 **WREN:** Write (Erase or Program) Enable bit  
1 = Enable an erase or program operation  
0 = No operation allowed (device clears this bit on completion of the write/erase operation)
- bit 13 **WRERR:** Flash Error Flag bit  
1 = A write operation is prematurely terminated (any  $\overline{\text{MCLR}}$  or WDT Reset during programming operation)  
0 = The write operation completed successfully
- bit 12 **PGMONLY:** Program Only Enable bit  
1 = Write operation is executed without erasing target address(es) first  
0 = Automatic erase-before-write: write operations are preceded automatically by an erase of target address(es)
- bit 11-7 **Reserved:** User code should write '0's to these locations
- bit 6 **ERASE:** Erase Operation Select bit  
1 = Perform an erase operation when WR is set  
0 = Perform a write operation when WR is set
- bit 5 **Reserved:** User code should write '0's to these locations
- bit 4-0 **NVMOP<4:0>:** Programming Operation Command Byte bits<sup>(1)</sup>  
Erase Operations (when ERASE bit is '1'):  
11010 = Erase 8 words  
11001 = Erase 4 words  
11000 = Erase 1 word  
100xx = Erase entire data EEPROM  
Programming Operations (when ERASE bit is '0'):  
001xx = Write 1 word

**Note 1:** Only those bit combinations that reflect valid data EEPROM operations are shown here. Other combinations are either unimplemented, or are used for Flash program memory or device configuration operations. Please refer to the specific device data sheet for information.

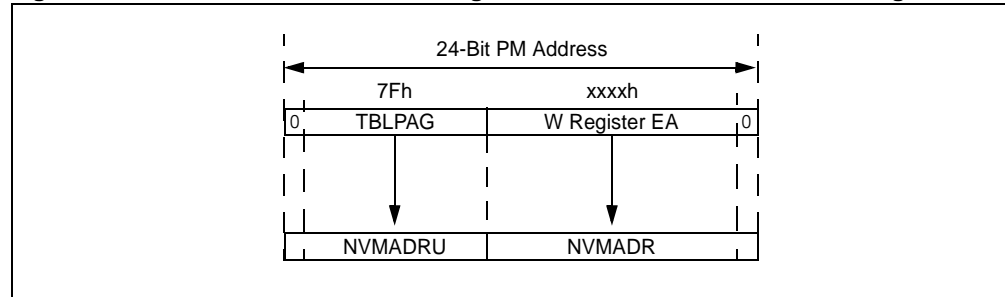
## 5.2.3 NVM Address Register

As with Flash program memory, the NVM Address Registers, NVMADRU and NVMADR, form the 24-bit Effective Address (EA) of the selected row or word for data EEPROM operations. The NVMADRU register is used to hold the upper 8 bits of the EA, while the NVMADR register is used to hold the lower 16 bits of the EA. These registers are not mapped into the SFR space; instead, they directly capture the EA<23:0> of the last table write instruction that has been executed and selects the data EEPROM row to erase. Figure 5-1 shows how the program memory EA is formed for programming and erase operations.

Like program memory operations, the Least Significant bit (LSb) of NVMADR is restricted to even addresses. This is because any given address in the data EEPROM space consists of only the lower word of the program memory width; the upper word, including the uppermost “phantom byte”, are unavailable. This means that the LSb of a data EEPROM address will always be ‘0’. The range of allowable values for NVMADR in data EEPROM operations is determined by the size of the particular device’s EEPROM.

Similarly, the Most Significant bit (MSb) of NVMADRU is always ‘0’, since all addresses lie in the user program space. Further, the value of TBLPAG (and thus, NVMADRU) can also be fixed at 7Fh for data EEPROM operations, since the data EEPROM address range will always be located in the top 64K page of the program memory space.

**Figure 5-1: Data EEPROM Addressing with TBLPAG and NVM Address Registers**



## 5.3 DATA EEPROM OPERATIONS

The EEPROM block is accessed using table read and write operations similar to those used for program memory. The TBLWTH and TBLRDH instructions are not required for EEPROM operations since the memory is only 16 bits wide. The program and erase procedures for the data EEPROM are similar to those used for the Flash program memory, except that they are optimized for fast data access. The following programming operations can be performed on the data EEPROM:

- Write one word
- Erase one, four or eight words
- Bulk erase entire data EEPROM

The data EEPROM is readable and writable during normal operation (full VDD operating range). Unlike the Flash program memory, normal program execution is not stopped during an EEPROM program or erase operation.

Data EEPROM operations are performed using the NVMCON and NVMKEY registers. The programming software is responsible for waiting for the operation to complete. The software may detect when the EEPROM erase or programming operation is complete by one of three methods:

- Poll the WR bit (NVMCON<15>) in software. The WR bit will be cleared when the operation is complete.
- Poll the NVMIF bit (IFS0<12>) in software. The NVMIF bit will be set when the operation is complete.
- Enable NVM interrupts. The CPU will be interrupted when the operation is complete. Further programming operations can be handled in the Interrupt Service Routine (ISR).

**Note:** Unexpected results will be obtained should the user attempt to read the EEPROM while a programming or erase operation is underway.

The C30 C compiler includes library procedures to automatically perform the table read and table write operations, manage the Table Pointer and write buffers, and unlock and initiate memory write sequences. This eliminates the need to create assembler macros or time critical routines in 'C' for each application.

The library procedures are used in the code examples detailed in the following sections. General descriptions of each process are provided for users who are not using the C30 compiler libraries.

## 5.3.1 Single-Word Write

The overall algorithm for writing to the data EEPROM is as follows:

1. Erase one data EEPROM word:
  - Set the NVMOP bits to erase one EEPROM word (NVMCON<4:0> = 11000).
  - Write address of word to be erased into the TBLPAG and WREG registers.
  - Clear NVMIF status bit and enable NVM interrupt (optional).
  - Write the key sequence to NVMKEY.
  - Set the WR bit to begin erase cycle.
  - Either poll the WR bit or wait for the NVM interrupt.
2. Write the data word into the data EEPROM latch.
3. Program the data word into the EEPROM:
  - Set up the NVMCON register to program one EEPROM word.
  - Clear NVMIF status bit and enable NVM interrupt (optional).
  - Write the key sequence to NVMKEY.
  - Set the WR bit to begin erase cycle.
  - Either poll the WR bit or wait for the NVM interrupt.

A table write instruction is used to write the data to one write latch. The TBLPAG register is loaded with the 8 MSBs of the EEPROM address. The 16 LSbs of the EEPROM address are automatically captured into the NVMADR register when the table write is executed. The LSb of the NVMADR register has no effect on the programming operation. The NVMCON register is configured to program one word of data EEPROM.

Setting the WR control bit (NVMCON<15>) initiates the programming operation. The unlock sequence must be written to the NVMKEY register before setting the WR control bit. The unlock sequence needs to be executed in the exact order shown without interruption (see [Section 5.2.2 "NVMKEY Register"](#) for details). Therefore, interrupts should be disabled prior to writing the sequence.

A typical write sequence, including the erase and key unlock sequences, is shown in [Example 5-2](#). This example uses C30 compiler library procedures to manage the Table Pointer (`builtin_tblpage` and `builtin_tbloffset`), the unlock sequence (`builtin_write_NVM`) and the actual data write (`builtin_tblwtl`). The memory unlock sequence also sets the WR bit to initiate the operation and returns control when complete.

### Example 5-2: Single-Word Write to Data EEPROM

```
// Set up NVMCON to write one word of data EEPROM
NVMCON = 0x4004;

// Set up a pointer to the EEPROM location to be written
TBLPAG = __builtin_tblpage(&ee_addr);
offset = __builtin_tbloffset(&ee_addr);

// Write Data Value To Holding Latch
__builtin_tblwtl(offset, data);

// Disable Interrupts For 5 Instructions
asm volatile ("disi #5");

// Issue Unlock Sequence & Start Write Cycle
__builtin_write_NVM();
```

## 5.3.2 Single-Word Erase

In single-word erase operations, the NVMADRU:NVMADR registers are loaded from the TBLPAG and WREG registers with the data EEPROM address to be erased. Since one word of the EEPROM is accessed, the LSb of the NVMADR has no effect on the erase operation. The NVMCON register must be configured to erase one word of EEPROM memory.

Setting the WR control bit (NVMCON<15>) initiates the erase. The unlock or key sequence must be written to the NVMKEY register before setting the WR control bit. The unlock sequence needs to be executed in the exact order shown without interruption (see [Section 5.2.2 “NVMKEY Register”](#) for details). Therefore, interrupts should be disabled prior to writing the sequence.

A typical erase sequence is shown in [Example 5-3](#). This example, and the other erase examples that follow, use C library procedures to manage the Table Pointer (`builtin_tblpage` and `builtin_tbloffset`) and the Erase Page Pointer (`builtin_tblwtl`). The memory unlock sequence (`builtin_write_NVM`) also sets the WR bit to initiate the operation and returns control when complete.

### Example 5-3: Single-Word Erase

```
// Set up NVMCON to erase one word of data EEPROM
NVMCON = 0x4058;

// Set up a pointer to the EEPROM location to be erased
TBLPAG = __builtin_tblpage(&ee_addr);
offset = __builtin_tbloffset(&ee_addr);
__builtin_tblwtl(offset, offset);

// Disable Interrupts For 5 Instructions
asm volatile ("disi #5");

// Issue Unlock Sequence & Start Write Cycle
__builtin_write_NVM();
```

## 5.3.3 Four-Word Erase

The NVMCON register is configured to erase 4 words (or one-half row) of EEPROM memory. A pointer to the data EEPROM address to be erased must first be initialized. The data EEPROM must be erased at even address boundaries. Therefore, the 3 LSbs of the Effective Address will have no effect on the memory that is erased.

Setting the WR control bit (NVMCON<15>) initiates the erase. The unlock sequence should be written to the NVMKEY register before setting the WR control bit. The unlock sequence needs to be executed in the exact order shown without interruption (see [Section 5.2.2 “NVMKEY Register”](#) for details). Therefore, interrupts should be disabled prior to writing the sequence.

### Example 5-4: Four-Word Erase Sequence

```
// Set up NVMCON to erase four words of data EEPROM
NVMCON = 0x4059;

// Set up a pointer to the EEPROM location to be erased
TBLPAG = __builtin_tblpage(&ee_addr);
offset = __builtin_tbloffset(&ee_addr);
__builtin_tblwtl(offset, offset);

// Disable Interrupts For 5 Instructions
asm volatile ("disi #5");

// Issue Unlock Sequence & Start Erase Cycle
__builtin_write_NVM();
```

### 5.3.4 Eight-Word Erase

The NVMCON register is configured to erase one row of EEPROM memory. The NVMADRU and NVMADR registers must point to the row to be erased. The data EEPROM must be erased at even address boundaries. Therefore, the 5 LSBs of the address will have no effect on the row that is erased.

Setting the WR control bit (NVMCON<15>) initiates the erase. The unlock sequence must be written to the NVMKEY register before setting the WR control bit. The unlock sequence needs to be executed in the exact order shown without interruption (see [Section 5.2.2 “NVMKEY Register”](#) for details). Therefore, interrupts should be disabled prior to writing the sequence.

#### Example 5-5: Eight-Word Erase

```
// Set up NVMCON to erase eight words of data EEPROM
NVMCON = 0x405A;

// Set up a pointer to the EEPROM location to be erased
TBLPAG = __builtin_tblpage(&ee_addr);
offset = __builtin_tbloffset(&ee_addr);
__builtin_tblwtl(offset, offset);

// Disable Interrupts For 5 Instructions
asm volatile ("disi #5");

// Issue Unlock Sequence & Start Erase Cycle
__builtin_write_NVM();
```

### 5.3.5 Data EEPROM Bulk Erase

The NVMCON register is configured to bulk erase the entire data EEPROM memory. Because this operation affects the entire data EEPROM, the address registers do not need to be configured.

Setting the WR control bit (NVMCON<15>) initiates the erase. The unlock sequence must be written to the NVMKEY register before setting the WR control bit. The unlock sequence needs to be executed in the exact order shown without interruption (see [Section 5.2.2 “NVMKEY Register”](#) for details). Therefore, interrupts should be disabled prior to writing the sequence.

In this bulk erase example ([Example 5-6](#)), executing the unlock library procedure automatically triggers the erase process, since it also sets the WR bit.

#### Example 5-6: Data EEPROM Bulk Erase

```
// Set up NVMCON to bulk erase the data EEPROM
NVMCON = 0x4050;

// Disable Interrupts For 5 Instructions
asm volatile ("disi #5");

// Issue Unlock Sequence and Start Erase Cycle
__builtin_write_NVM();
```

## 5.3.6 Reading the Data EEPROM

As with program memory operations, the table read instruction is used to read data from the data EEPROM. Since the EEPROM array is only 16 bits wide, only the `TBLRD` instruction is needed. In [Example 5-7](#), `W0` is used as a pointer to the data EEPROM address. The result is placed into register, `W4`.

Program Space Visibility (PSV) can also be used to read locations in the program memory address space. See [Section 4. “Program Memory”](#) for further information about PSV.

The data EEPROM read example ([Example 5-7](#)) uses the Table Pointer management (`builtin_tblpage` and `builtin_tbloffset`) and table read (`builtin_tblrld`) procedures from the C30 compiler library.

### Example 5-7: Reading the Data EEPROM Using the `TBLRD` Command

```
// Set up a pointer to the EEPROM location to be read
TBLPAG = __builtin_tblpage(&ee_addr);
offset = __builtin_tbloffset(&ee_addr);

// Read the EEPROM data
data = __builtin_tblrld(offset);
```



## 5.4 REGISTER MAP

A summary of the Special Function Registers (SFRs) associated with the PIC24F Data EEPROM is provided in [Table 5-1](#).

**Table 5-1: Registers Associated with Data EEPROM Operation**

File Name	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
TBLPAG	—	—	—	—	—	—	—	—	Table Memory Page Address Register								0000
NVMCON	WR	WREN	WRERR	PGONLY	r	r	r	r	r	ERASE	r	NVMOP4	NVMOP3	NVMOP2	NVMOP1	NVMOP0	0000 <sup>(1)</sup>
NVMKEY	—	—	—	—	—	—	—	—	NVMKEY<7:0>								0000

**Legend:** — = unimplemented, read as '0'; r = reserved, user code should write '0's to these locations

**Note 1:** Reset value shown is for Power-on Reset (POR) only. Value on other Reset states is dependent on the state of memory write or erase operations at the time of Reset.

## 5.5 DESIGN TIPS

**Question 1:** *I cannot get the data EEPROM to program or erase properly. My code appears to be correct. What could be the cause?*

**Answer:** Interrupts should be disabled when a program or erase cycle is initiated to ensure that the key sequence executes without interruption. Interrupts can be disabled by using the `DISI` instruction, or by raising the current CPU priority to Level 7.

The code examples shown in this chapter disable interrupts for a specified number of instruction cycles with the `DISI` instruction. An alternate method to temporarily disable interrupts is by saving the current SR register value on the stack, then ORing the value, `00E0h`, with SR to force `IPL<2:0> = 111`.

**Question 2:** *What is an easy way to read data EEPROM without using table instructions?*

**Answer:** The data EEPROM is mapped into the program memory space. PSV can be used to map the EEPROM region into data memory space. See [Section 4. “Program Memory”](#) for further information about PSV.

### 5.6 RELATED APPLICATION NOTES

This section lists application notes that are related to this section of the manual. These application notes may not be written specifically for the PIC24F device family, but the concepts are pertinent and could be used with modification and possible limitations. The current application notes related to the Data EEPROM are:

Title	Application Note #
Emulating Data EEPROM for PIC18 and PIC24 Microcontrollers and dsPIC® Digital Signal Controllers	AN1095

**Note:** Please visit the Microchip web site ([www.microchip.com](http://www.microchip.com)) for additional application notes and code examples for the PIC24F family of devices.

## 5.7 REVISION HISTORY

### **Revision A (October 2007)**

This is the initial released revision of this document.

### **Revision B (June 2011)**

Corrects code examples in [Example 5-2](#) through [Example 5-5](#) and [Example 5-7](#) for TBLPAG statement and NVMCOM register data.

---

**Note the following details of the code protection feature on Microchip devices:**

- Microchip products meet the specification contained in their particular Microchip Data Sheet.
- Microchip believes that its family of products is one of the most secure families of its kind on the market today, when used in the intended manner and under normal conditions.
- There are dishonest and possibly illegal methods used to breach the code protection feature. All of these methods, to our knowledge, require using the Microchip products in a manner outside the operating specifications contained in Microchip's Data Sheets. Most likely, the person doing so is engaged in theft of intellectual property.
- Microchip is willing to work with the customer who is concerned about the integrity of their code.
- Neither Microchip nor any other semiconductor manufacturer can guarantee the security of their code. Code protection does not mean that we are guaranteeing the product as "unbreakable."

Code protection is constantly evolving. We at Microchip are committed to continuously improving the code protection features of our products. Attempts to break Microchip's code protection feature may be a violation of the Digital Millennium Copyright Act. If such acts allow unauthorized access to your software or other copyrighted work, you may have a right to sue for relief under that Act.

---

Information contained in this publication regarding device applications and the like is provided only for your convenience and may be superseded by updates. It is your responsibility to ensure that your application meets with your specifications. MICROCHIP MAKES NO REPRESENTATIONS OR WARRANTIES OF ANY KIND WHETHER EXPRESS OR IMPLIED, WRITTEN OR ORAL, STATUTORY OR OTHERWISE, RELATED TO THE INFORMATION, INCLUDING BUT NOT LIMITED TO ITS CONDITION, QUALITY, PERFORMANCE, MERCHANTABILITY OR FITNESS FOR PURPOSE. Microchip disclaims all liability arising from this information and its use. Use of Microchip devices in life support and/or safety applications is entirely at the buyer's risk, and the buyer agrees to defend, indemnify and hold harmless Microchip from any and all damages, claims, suits, or expenses resulting from such use. No licenses are conveyed, implicitly or otherwise, under any Microchip intellectual property rights.

#### **Trademarks**

The Microchip name and logo, the Microchip logo, dsPIC, KEELOQ, KEELOQ logo, MPLAB, PIC, PICmicro, PICSTART, PIC<sup>32</sup> logo, rfPIC and UNI/O are registered trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.


FilterLab, Hampshire, HI-TECH C, Linear Active Thermistor, MXDEV, MXLAB, SEEVAL and The Embedded Control Solutions Company are registered trademarks of Microchip Technology Incorporated in the U.S.A.

Analog-for-the-Digital Age, Application Maestro, chipKIT, chipKIT logo, CodeGuard, dsPICDEM, dsPICDEM.net, dsPICworks, dsSPEAK, ECAN, ECONOMONITOR, FanSense, HI-TIDE, In-Circuit Serial Programming, ICSP, Mindi, MiWi, MPASM, MPLAB Certified logo, MPLIB, MPLINK, mTouch, Omniscent Code Generation, PICC, PICC-18, PICDEM, PICDEM.net, PICKit, PICtail, REAL ICE, rfLAB, Select Mode, Total Endurance, TSHARC, UniWinDriver, WiperLock and ZENA are trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

SQTP is a service mark of Microchip Technology Incorporated in the U.S.A.

All other trademarks mentioned herein are property of their respective companies.

© 2007-2011, Microchip Technology Incorporated, Printed in the U.S.A., All Rights Reserved.

 Printed on recycled paper.

ISBN: 978-1-61341-312-8

**QUALITY MANAGEMENT SYSTEM**  
**CERTIFIED BY DNV**  
**== ISO/TS 16949:2009 ==**

*Microchip received ISO/TS-16949:2002 certification for its worldwide headquarters, design and wafer fabrication facilities in Chandler and Tempe, Arizona; Gresham, Oregon and design centers in California and India. The Company's quality system processes and procedures are for its PIC® MCUs and dsPIC® DSCs, KEELOQ® code hopping devices, Serial EEPROMs, microperipherals, nonvolatile memory and analog products. In addition, Microchip's quality system for the design and manufacture of development systems is ISO 9001:2000 certified.*

## Worldwide Sales and Service

### AMERICAS

**Corporate Office**  
2355 West Chandler Blvd.  
Chandler, AZ 85224-6199  
Tel: 480-792-7200  
Fax: 480-792-7277  
Technical Support:  
<http://www.microchip.com/support>  
Web Address:  
[www.microchip.com](http://www.microchip.com)

**Atlanta**  
Duluth, GA  
Tel: 678-957-9614  
Fax: 678-957-1455

**Boston**  
Westborough, MA  
Tel: 774-760-0087  
Fax: 774-760-0088

**Chicago**  
Itasca, IL  
Tel: 630-285-0071  
Fax: 630-285-0075

**Cleveland**  
Independence, OH  
Tel: 216-447-0464  
Fax: 216-447-0643

**Dallas**  
Addison, TX  
Tel: 972-818-7423  
Fax: 972-818-2924

**Detroit**  
Farmington Hills, MI  
Tel: 248-538-2250  
Fax: 248-538-2260

**Indianapolis**  
Noblesville, IN  
Tel: 317-773-8323  
Fax: 317-773-5453

**Los Angeles**  
Mission Viejo, CA  
Tel: 949-462-9523  
Fax: 949-462-9608

**Santa Clara**  
Santa Clara, CA  
Tel: 408-961-6444  
Fax: 408-961-6445

**Toronto**  
Mississauga, Ontario,  
Canada  
Tel: 905-673-0699  
Fax: 905-673-6509

### ASIA/PACIFIC

**Asia Pacific Office**  
Suites 3707-14, 37th Floor  
Tower 6, The Gateway  
Harbour City, Kowloon  
Hong Kong  
Tel: 852-2401-1200  
Fax: 852-2401-3431

**Australia - Sydney**  
Tel: 61-2-9868-6733  
Fax: 61-2-9868-6755

**China - Beijing**  
Tel: 86-10-8569-7000  
Fax: 86-10-8528-2104

**China - Chengdu**  
Tel: 86-28-8665-5511  
Fax: 86-28-8665-7889

**China - Chongqing**  
Tel: 86-23-8980-9588  
Fax: 86-23-8980-9500

**China - Hangzhou**  
Tel: 86-571-2819-3180  
Fax: 86-571-2819-3189

**China - Hong Kong SAR**  
Tel: 852-2401-1200  
Fax: 852-2401-3431

**China - Nanjing**  
Tel: 86-25-8473-2460  
Fax: 86-25-8473-2470

**China - Qingdao**  
Tel: 86-532-8502-7355  
Fax: 86-532-8502-7205

**China - Shanghai**  
Tel: 86-21-5407-5533  
Fax: 86-21-5407-5066

**China - Shenyang**  
Tel: 86-24-2334-2829  
Fax: 86-24-2334-2393

**China - Shenzhen**  
Tel: 86-755-8203-2660  
Fax: 86-755-8203-1760

**China - Wuhan**  
Tel: 86-27-5980-5300  
Fax: 86-27-5980-5118

**China - Xian**  
Tel: 86-29-8833-7252  
Fax: 86-29-8833-7256

**China - Xiamen**  
Tel: 86-592-2388138  
Fax: 86-592-2388130

**China - Zhuhai**  
Tel: 86-756-3210040  
Fax: 86-756-3210049

### ASIA/PACIFIC

**India - Bangalore**  
Tel: 91-80-3090-4444  
Fax: 91-80-3090-4123

**India - New Delhi**  
Tel: 91-11-4160-8631  
Fax: 91-11-4160-8632

**India - Pune**  
Tel: 91-20-2566-1512  
Fax: 91-20-2566-1513

**Japan - Yokohama**  
Tel: 81-45-471- 6166  
Fax: 81-45-471-6122

**Korea - Daegu**  
Tel: 82-53-744-4301  
Fax: 82-53-744-4302

**Korea - Seoul**  
Tel: 82-2-554-7200  
Fax: 82-2-558-5932 or  
82-2-558-5934

**Malaysia - Kuala Lumpur**  
Tel: 60-3-6201-9857  
Fax: 60-3-6201-9859

**Malaysia - Penang**  
Tel: 60-4-227-8870  
Fax: 60-4-227-4068

**Philippines - Manila**  
Tel: 63-2-634-9065  
Fax: 63-2-634-9069

**Singapore**  
Tel: 65-6334-8870  
Fax: 65-6334-8850

**Taiwan - Hsin Chu**  
Tel: 886-3-6578-300  
Fax: 886-3-6578-370

**Taiwan - Kaohsiung**  
Tel: 886-7-213-7830  
Fax: 886-7-330-9305

**Taiwan - Taipei**  
Tel: 886-2-2500-6610  
Fax: 886-2-2508-0102

**Thailand - Bangkok**  
Tel: 66-2-694-1351  
Fax: 66-2-694-1350

### EUROPE

**Austria - Wels**  
Tel: 43-7242-2244-39  
Fax: 43-7242-2244-393

**Denmark - Copenhagen**  
Tel: 45-4450-2828  
Fax: 45-4485-2829

**France - Paris**  
Tel: 33-1-69-53-63-20  
Fax: 33-1-69-30-90-79

**Germany - Munich**  
Tel: 49-89-627-144-0  
Fax: 49-89-627-144-44

**Italy - Milan**  
Tel: 39-0331-742611  
Fax: 39-0331-466781

**Netherlands - Drunen**  
Tel: 31-416-690399  
Fax: 31-416-690340

**Spain - Madrid**  
Tel: 34-91-708-08-90  
Fax: 34-91-708-08-91

**UK - Wokingham**  
Tel: 44-118-921-5869  
Fax: 44-118-921-5820