AN1111

An FTP Server Using BSD Socket API

Author: *Sean Justice*
Microchip Technology Inc.

INTRODUCTION

An embedded FTP (File Transfer Protocol) server is an excellent addition to any network-enabled device. FTP server capability facilitates the uploading of files to, and downloading of files from, an embedded device. Almost all computers have, at the very least, a command line FTP client that will allow a user to connect to an embedded FTP server.

This Microchip FTP server application note and the included FAT16 module, supplemented by the TCP/IP application note AN1108, "*Microchip TCP/IP Stack with BSD Socket API*", provide an FTP Server module that can be integrated with almost any application on a Microchip 32-bit microcontroller product.

The TCP/IP application note and the FAT16 module are required to compile and run the FTP server module. All notes and files mentioned in this document are available for download from www.microchip.com.

The software in the installation files includes a sample application that demonstrates all of the features offered by this FTP server module.

Questions and answers about the FTP server module are provided at the end of this document in **"Answers to Common Questions"** on page 16.

ASSUMPTION

The author assumes that the reader is familiar with the following Microchip development tools: MPLAB® IDE and MPLAB® REAL ICE™ in-circuit emulator. It is also assumed that the reader is familiar with C programming language, as well as TCP/IP stack, FAT16 file system, and FTP server concepts. Terminology from these technologies is used in this document, and only brief overviews of the concepts are provided. Advanced users are encouraged to read the associated specifications.

FEATURES

The FTP server provided here does not implement all FTP functionality, it is a minimal server that is targeted for an embedded system. You can easily add new functionality as required.

The FTP server presented here incorporates the following features:

- Provides portability across the 32-bit family of PIC® microcontrollers
- FTP Server APIs are compatible with PIC18 and PIC24 Microchip FTP Server APIs
- FTP connection is authenticated by your application
- Automatic interaction with the FAT16 file system
- Upload files to the server using the `PUT` command
- Download file to the client using the `GET` command
- Supports the FTP `NOOP` command
- Supports the `PORT` command, allowing you to change the data port
- FTP Server APIs compatible with older Microchip FTP Server APIs
- Portable across all 32-bit family of PIC® microcontrollers

LIMITATIONS

The FTP server only supports one client connection at a time, i.e., unable to support multiple client connections simultaneously.

Some of the FTP commands that you may want to use are not implemented in this presentation. The official FTP specification "*RFC 959*" is available for your use at http://www.faqs.org/rfcs/rfc959.html. It supplies information about other FTP commands and responses that provide additional functionality.

TYPICAL HARDWARE

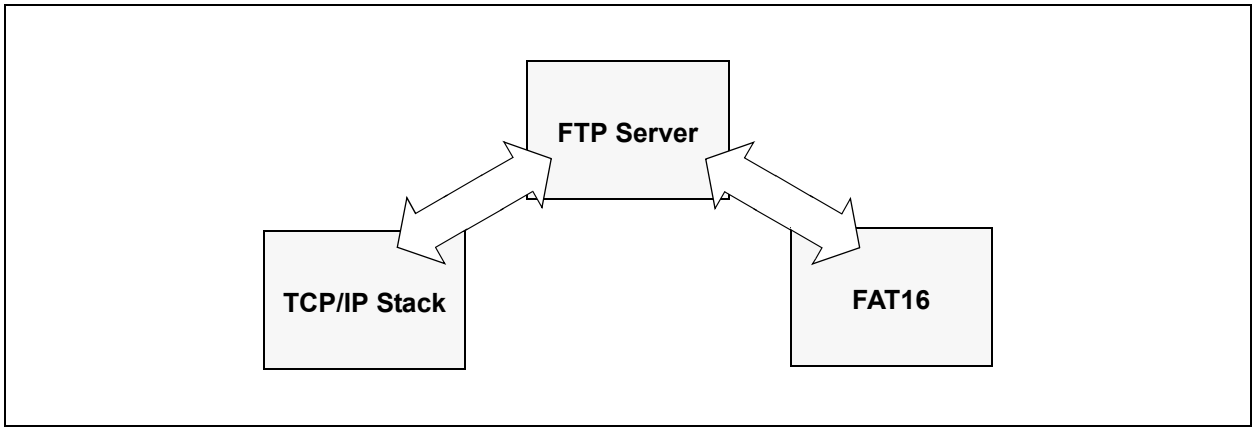
The FTP server demonstration application requires use of the hardware and software presented in AN1108, "*Microchip TCP/IP Stack with BSD Socket API*", FAT16 hardware and software.

The FTP demo runs on an Explorer 16 board with ENC28J60 PICtail™ Plus (Microchip Part Number AC164123) and a FAT16-type media storage device (Microchip Part Number AC164122).

The Explorer 16 board must be populated with PICtail™ Plus connections in J5 and J6. Refer to the Explorer 16 documentation for more information.

Figure 1 shows the FTP server dependencies.

FIGURE 1: FTP SERVER DEPENDENCIES



RESOURCE REQUIREMENTS

FTP server (`ftp.c` model) requires 2128 bytes of program memory.

TABLE 1: MEMORY REQUIREMENTS

Data Structure	RAM Usage (bytes)
Listening Socket	4
Data Socket	4
Connection Socket	4
Socket Address	28
Data Port	2
Command Data	16
User String	10 (user defined)
String	32
Argument Pointer	4
File Pointer	4
Tx Buffer	64 (user defined)
Other	18

Since the FTP server requires the use of TCP/IP and FAT16, it also inherits the program memory and RAM requirements of those as well. Refer to AN1108 "*Microchip TCP/IP Stack with BSD Socket API*" and FAT16 for those requirements.

The compiler used for the memory requirements was the Microchip MPLAB C32 C Compiler, version 1.00. The optimization was not used. Note that the use of compilers and optimization settings may increase or decrease the memory requirements.

INSTALLING SOURCE FILES

The complete source code for the Microchip FTP server is available for download from the Microchip web site (see **"Source Code"** on page 17).

The source code is distributed in a single Windows® installation file:
`pic32mx_bsd_tcp_ip_v1_00_00.exe`.

Perform the following steps to complete the installation:

1. Execute the file. A Windows installation wizard will guide you through the installation process.
2. Click **I Accept**, to consent to the software license agreement.
3. After the installation process is completed, the **FTP Server Using BSD Socket API** item is available under the **Microchip** program group.

   The complete source files are copied to the following directory, in your choice of installation path:
   `\pic32_solutions\microchip\bsd_ftp_server\source`

   The "include" files are copied to the following directory:
   `\pic32_solutions\microchip\include\bsd_ftp-server`

   The demonstration application for the BSD FTP server is located in the following directory:
   `\pic32_solutions\bsd_ftp_server_demo`
4. For the latest version-specific features and limitations, refer to the version HTML page, which can be accessed through `index.html` .

SOURCE FILE ORGANIZATION

The FTP server consists of multiple files. These files are organized in multiple directories.

Table 2 shows the directory structure.

Table 3 lists the server-related source files.

TABLE 2: SOURCE FILE DIRECTORY STRUCTURE

Directory	Location
pic32_solutions\bsd_ftp_server_demo	FTP server demo project files and source files
pic32_solutions\microchip\bsd_ftp_server	FTP server source files
pic32_solutions\microchip\include\bsd_ftp_server	FTP server “include” files

TABLE 3: SOURCE FILES

File	Directory	Description
bsd_ftp_server_demo.mcp	pic32_solutions\ bsd_ftp_server_demo	MPLAB® IDE FTP server demo project
bsd_ftp_server_demo.mcw	pic32_solutions\ bsd_ftp_server_demo	MPLAB IDE FTP server demo workspace
ftpex.c	pic32_solutions\ bsd_ftp_server_demo\source	User-modifiable FTP server callback functions
main.c	pic32_solutions\ bsd_ftp_server_demo\source	Main demo file
ftpex.h	pic32_solutions\ bsd_ftp_server_demo\source	User-modifiable FTP server callback functions header
eTCP.def	pic32_solutions\ bsd_ftp_server_demo\source	User-modifiable TCP/IP defines
fat.def	pic32_solutions\ bsd_ftp_server_demo\source	User-modifiable FAT16 defines
ftp.c	pic32_solutions\microchip\ bsd_ftp_server\source	User-modifiable FTP server callback functions
ftp_private.h	pic32_solutions\microchip\ bsd_ftp_server\source	Main demo file
ftpex.tmpl		User-modifiable FTP server callback functions template
ftp.h	pic32_solutions\microchip\ include\bsd_ftp_server	FTP ”include”
ftpex.tmpl		User-modifiable FTP server callback functions header template

DEMO APPLICATION

This Microchip FTP server includes a complete working application to demonstrate the FTP server running on the Microchip BSD TCP/IP stack. This application is designed to run on Microchip's Explorer 16 demonstration board. However, it can be easily modified to support any board.

Demo FTP Server Application Features

Version 1.0 of the demo FTP server application implements the following functions:

- Downloads files to a FTP client
- Uploads files from a FTP client
- Supports user authentication

However, refer to version log, through `index.html`, for version-specific feature additions or improvements.

The main source file for this application is `main.c`. Refer to the source code as a starting point for creating your own application, utilizing different aspects of the Microchip FTP server.

In order to compile the project you must have the source code from the supplementary application notes for TCP/IP stack and the FAT16 file system.

Programming the FTP Demo Application

To program a target board with the demo application, you must have access to a PIC microcontroller programmer. The following procedure assumes that you will be using MPLAB REAL ICE in-circuit emulator as a programmer. If not, refer to the instructions for your specific programmer.

1. Connect the MPLAB REAL ICE in-circuit emulator to the Explorer 16 board or to your target board.
2. Apply power to the target board.
3. Launch MPLAB IDE.
4. Select the PIC device of your choice (this step is required only if you are importing a hex file that was previously built).
5. Enable the MPLAB REAL ICE in-circuit emulator as your programming tool.
6. If you want to use a previously built hex file, import:
   `bsd_ftp_server_demo\release\bsd_ftp_server_demo.hex`
   If you are rebuilding the hex file, open the project file:
   `bsd_ftp_server_demo\bsd_ftp_server_demo.mcp`, and follow the build procedure to create the application hex file.
7. The demo application contains necessary configuration options required for the Explorer 16 board. If you are programming another type of board, make sure that you select the appropriate oscillator mode from the MPLAB REAL ICE in-circuit emulator configuration settings menu.
8. Select the Program menu option from the MPLAB REAL ICE in-circuit emulator menu to begin programming the target.
9. After a few seconds, you should see the message "Programming successful". If not, check your board and your MPLAB REAL ICE connection. Click Help on the menu bar for further assistance.
10. Remove power from the board and disconnect the MPLAB cable from the target board.

Setting Up Demo Application Hardware

In order to run the FTP server demo correctly, you must prepare the hardware on the Explorer 16 board to use the TCP/IP stack and FAT16. Refer to AN1108 "*Microchip TCP/IP Stack with BSD Socket API*" for the proper hardware setup.

The demo requires that the TCP/IP connection (Microchip Part Number AC164123) uses SPI 1 and the FAT16-type media storage device (Microchip Part Number AC164122) uses SPI 2.

Executing the Demo Application

When the programmed microcontroller is installed on the Explorer 16 board and powered up, the system two-line LCD display shows the following information:

PIC32 BSD FTP

<Current IP address>

Building the Demo FTP Server

The demo FTP server application included in this application note can be built using Microchip's 32-bit MPLAB C32 C Compiler. However, you can port the source to the compiler that you routinely use with Microchip microcontroller products.

The demo application also includes a predefined FTP server project file, `bsd_ftp_server_demo.mcp`, to be used with the Microchip MPLAB IDE. The project was created using a PIC32 device. If you are using a different device, you must select the appropriate device by using the MPLAB IDE menu command. In addition, the demo application project uses additional "include" paths as defined in the **Build Options** of MPLAB IDE:

```
.\source
..\microchip\include
```

Table 4 lists the source files that are needed to build the demo FTP server application, and their respective locations.

The following instructions describe a high-level procedure for building the demo application. This procedure assumes that you are familiar with MPLAB IDE and will be using MPLAB IDE to build the application. If not, refer to the instructions for your in-circuit development environment to create and build the project.

1. Make sure that source files for the Microchip FTP server are installed. If not, refer to **"Installing Source Files"**on page 3.
2. Launch MPLAB IDE and open the project file, `bsd_ftp_server_demo.mcp`.
3. Use the appropriate MPLAB IDE menu commands to build the project. Note that the demo project was created to compile properly when the source files are located in the directory structure that is suggested by the installation wizard. If you installed the source files to other locations, you must recreate or modify existing project settings to accomplish the build.
4. The build process should finish successfully.

   If not, make sure that your MPLAB IDE and compiler are set up correctly.

TABLE 4: DEMO FTP SERVER APPLICATION PROJECT FILES

File	Directory
main.c	\pic32_solutions\bsd_http_server_demo\source
ftpex.c	\pic32_solutions\bsd_http_server_demo\source
eTCP.def	\pic32_solutions\bsd_http_server_demo\source
fat.def	\pic32_solutions\bsd_http_server_demo\source
ftp.c	\pic32_solutions\microchip\bsd_ftp_server\source
dchp.c	\pic32_solutions\microchip\bsd_dhcp_client
block_mdr.c	\pic32_solutions\microchip\bsd_tcp_ip\source
earp.c	\pic32_solutions\microchip\bsd_tcp_ip\source
eicmp.c	\pic32_solutions\microchip\bsd_tcp_ip\source
eip.c	\pic32_solutions\microchip\bsd_tcp_ip\source
ENC28J60.c	\pic32_solutions\microchip\bsd_tcp_ip\source
etcp.c	\pic32_solutions\microchip\bsd_tcp_ip\source
ether.c	\pic32_solutions\microchip\bsd_tcp_ip\source
eudp.c	\pic32_solutions\microchip\bsd_tcp_ip\source
gpfunc.c	\pic32_solutions\microchip\bsd_tcp_ip\source
pkt_queue.c	\pic32_solutions\microchip\bsd_tcp_ip\source
route.c	\pic32_solutions\microchip\bsd_tcp_ip\source
socket.c	\pic32_solutions\microchip\bsd_tcp_ip\source
tick.c	\pic32_solutions\microchip\bsd_tcp_ip\source
fat.c	\pic32_solutions\microchip\fat16\source
fileio.c	\pic32_solutions\microchip\fat16\source
mediasd.c	\pic32_solutions\microchip\fat16\source
mstimer.c	\pic32_solutions\microchip\common
exlcd.c	\pic32_solutions\microchip\common

USING THE FTP SERVER

The installation file that accompanies this application note contains the full source code for the Microchip FTP server (see **"Source Code"** on page 17). A demo application is included in these source files. However the demo requires source code from AN1108, "*Microchip TCP/IP Stack with BSD Socket API*".

All applications based on the Microchip FTP server must be written in a cooperative multi-tasking manner. Cooperative multitasking architecture consists of a number of tasks executing in sequence. A cooperative task would quickly perform its required operation and return so that the next task would be able to execute.

Because of this requirement, a task that needs to wait for some external input, or performs a long operation, should be divided into subtasks using a state machine approach. Further discussion of cooperative multitasking and state machine programming is beyond the scope of this document. You should refer to software engineering literature for more detail.

The Microchip FTP server is written to support MPLAB C32 C Compiler without any changes. FTP server is written in standard ANSI C.

To simplify file management and application development, all source files are located in subdirectories under the `source` directory. See **"Source File Organization"** on page 4 for more information.

When you develop your application for the FTP server, use the directory structure of the demo as a reference to create your own application-specific subdirectory.

The following steps are typical of those you would use to develop an application based on the Microchip Stack. Note that these steps assume that you are using MPLAB IDE and are familiar with the interface.

1. Install the Microchip Stack source as described in **"Installing Source Files"** on page 3.
2. Create your application-specific directory in the `pic32_solutions` directory.
3. Use MPLAB IDE to create your application project and add the Stack source files as per your FTP node functionality.
4. If you are using the MPLAB C32 C compiler, add your device-specific linker script file.
5. Use the MPLAB **Build Option** dialog box to set two additional include search paths:

   ```
   .\source
   ..\microchip\include
   ```

6. Add your application-specific source files. Now your application project is ready for the build.

Integrating Your Application

The FTP server included with this application note is implemented as a cooperative task that co-exists with the Microchip BSD TCP/IP Stack and the your main application. The server, itself, is implemented in the source file, `ftp.c`, with a user application implementing seven callback functions. The demo application source file, `ftpex.c`, should be used as a template application to create the necessary callback functions.

The main component of the server consists of the FTP server task..

In order to integrate the FTP server into a user application, do the following:

1. Include the files `ftp.c` and `ftpex.c` in the project.
2. Include files to support the TCP/IP stack and FAT16.
3. Copy and modify the TCP/IP and FAT16 template files (`.tmpl`).
4. Modify the `main()` function of the application to include the FTP server (see the code example in Example 1 on page 9).

The server actually consists of two components: the FTP server, and the FTP authentication callbacks implemented by a user application. The user must implement a callback with the `FTPVerify` function that verifies both the FTP user name and password. See the `FTPVerify` function on page 14, for more detail.

FTP server uses a default time-out of approximately 180 seconds for both uploads and downloads. If a remote FTP connection stays IDLE for more than 180 seconds, it is automatically disconnected. This ensures that an orphan connection or a problem during a file upload does not tie up the FTP server indefinitely.

Uploading a File Using the FTP Client

One of the purposes of the FTP server in the Microchip Stack is to remotely upload files into a FAT16 media storage device. A typical exchange between a user and a node running the Microchip Stack with FTP server is shown in Example 1. In this instance, a file is being uploaded from a computer to the node.

For the sake of illustration, this is what you would see if using a command window from a computer running Microsoft Windows; other systems and terminal emulations may vary slightly. The actual FTP user name and password depends on the user application. The FTP Client actions (i.e., manual input from the user) are shown in **bold**. System prompts and FTP server responses are in plain face.

EXAMPLE 1: UPLOADING A FILE USING FTP

```
c:\ ftp 10.10.5.15
220 ready
User (10.10.5.15: (none)): ftp
331 Password required
Password: microchip
230 Logged in
ftp> put.txt
200 ok
150 Transferring data...
226 Transfer Complete
ftp> 16212 bytes transferred in 0.01Seconds 16212000.00Kbytes/sec.
ftp> quit
221 Bye
```

Note: The FTP server does NOT echo back the password as the user types it in. In the example above, it is shown to illustrate what the user would enter.

Downloading a File Using the FTP Client

One of the purposes of the FTP server in the Microchip Stack is to remotely download files to the FTP client. A typical exchange between a user and a node running the Microchip Stack with FTP server is shown in Example 2. In this instance, a file is being downloaded from the node to a computer.

For the sake of illustration, this is what you would see if using a command window from a computer running Microsoft Windows; other systems and terminal emulations may vary slightly. The actual FTP user name and password depends on the user application. FTP Client actions (i.e., manual input from the user) are shown in **bold**. System prompts and FTP server responses are shown in plain face.

EXAMPLE 2: DOWNLOADING A FILE USING FTP

```
c:\ ftp 10.10.5.15
220 ready
User (10.10.5.15: (none)): ftp
331 Password required
Password: microchip
230 Logged in
ftp> get.txt
200 ok
150 Transferring data...
226 Transfer Complete
ftp> 16212 bytes transferred in 0.01Seconds 16212000.00Kbytes/sec.
ftp> quit
221 Bye
```

Note: The FTP server does NOT echo back the password as the user types it in. In the example above, it is shown to illustrate what the user would enter.

FTP SERVER TASK

The FTP server task contains three functions that the main application can call to set up and run FTP protocol.

- `FTPInit`
- `FTPServer`
- `FTPVerify`

FTPInit

FTPInit initializes the TCP/IP connection to handle incoming connections by a FTP client.

Syntax

```
BOOL FTPInit(void)
```

Parameters

None.

Return Value

TRUE: If the listening FTP socket was successfully created and binded.

FALSE: If the FTP socket creation and binding failed.

Precondition

The TCP/IP socket must be initialized.

Side Effects

None.

Remarks

FTPInit should be called after FAT16 and TCP/IP have been initialized.

Example

```
TCPIPSetDefaultAddr();
InitStackMgr();
TickInit();


// First of all initialize the FAT16 library.
if ( !FSInit() )
{
    // If failed to initialize the FAT16, set an error LED
    // Primary reasons for failure would be no SD card detected
    // Or badly formatted SD card.
    ERROR_LED = 1;
    return -1;
}


FTPInit();
```

FTPServer

FTPServer handles the incoming connections and processes FTP client requests.

Syntax

```
BOOL FTPServer(void)
```

Parameters

None.

Return Values

TRUE: If an FTP client request is being processed.

FALSE: If there is not a pending FTP client connection or a request from a FTP client was not handled.

Precondition

FTPInit must be called before FTPServer. TCP/IP service routine must be called before or after this function.

Side Effects

None.

Remarks

FTPServer is able to service only one connection at a time.

Example

```
while(1)
{

StackMgrProcess();
DHCPTask();
FTPServer();


}
```

FTPVerify

FTPVerify is a callback from the FTP server. The server calls FTPVerify when it receives a connect request from a remote FTP client. FTPVerify is implemented by the main user application, and is used to authenticate the remote FTP user.

Syntax

```
BOOL FTPVerify(char *login, char *password)
```

Parameters

login [in]

Character string that contains the user name.

password [in]

Character string that contains the password.

Return Values

TRUE: If *login* and *password* match the login and password variable values defined in the user application.

FALSE: If either the *login* or *password* do not match.

FTP server uses the return value from FTPVerify to allow or disallow access by the remote FTP user.

Precondition

None.

Side Effects

None.

Remarks

The length user name may range from 0 through 16. If a longer user name is required, modify the value of FTP_USER_NAME_LEN in the header file ftp.h.

The maximum password string length and total FTP command length is predefined to be 31. If a longer password and/or command is required, modify the value of MAX_FTP_CMD_STRING_LEN in FTP.c.

Example

This example demonstrates how a FTP callback will be implemented.

```
char FTPUserName[] = "ftp";
#define FTP_USER_NAME_LEN   (sizeof(FTP_USER_NAME)-1)
char FTPPassword[] = "microchip";
#define FTP_USER_PASS_LEN   (sizeof(FTP_USER_PASS)-1)

BOOL FTPVerify(char *login, char *password)
{
    if ( memcmp(FTP_USER_NAME, login, FTP_USER_NAME_LEN) )
    {
        if ( memcmp(FTP_USER_PASS, password, FTP_USER_PASS_LEN) )
            return TRUE;
    }
    return FALSE;
}
```

For more detail, refer to the Webpages*.cgi files and the corresponding callback in the Websrvr.c source file.

USER-MODIFIABLE DEFINES

`FTP_TX_BUF_SIZE`

Location

`ftp.h`

Recommended Values

16 - 128

Default Value

64

Remarks

This is the size of the buffer used for sending information over the TCP/IP connection. The buffer should be at least as large as the largest FTP server response.

`FTP_USER_NAME_LEN`

Location

`ftp.h`

Recommend Values

10 - 16

Default Value

10

Remarks

This is the size of the user name of the FTP client..

ANSWERS TO COMMON QUESTIONS

Q: Why am I unable to log into the FTP server?

A: Check to make sure that the user name and password are correct. Remember that the FTP client will not echo back the password and it could be case sensitive depending on how you defined it in `FTPVerify` callback.

CONCLUSION

The Microchip BSD FTP Server is not the first application of its type for the Microchip PIC32 family of microcontrollers. However, it does provide is a space-efficient and modular version of an application integrating a TCP/IP stack with a FAT16 file system, implementing cooperative multitasking (without an operating system) in a small footprint. With its adaptability to many firmware applications and its availability as a no-cost software solution, this stack should prove to be useful in developing applications where embedded control requires network connectivity.

PARTIAL LIST OF RFC DOCUMENTS

RFC Document	Description
RFC 826	Ethernet Address Resolution Protocol (ARP)
RFC 791	Internet Protocol (IP)
RFC 792	Internet Control Message Protocol (ICMP)
RFC 793	Transmission Control Protocol (TCP)
RFC 768	User Datagram Protocol (UDP)
RFC 821	Simple Mail Transfer Protocol (SMTP)
RFC 1055	Serial Line Internet Protocol (SLIP)
RFC 1866	Hypertext Markup Language (HTML 2.0)
RFC 2616	Hypertext Transfer Protocol (HTTP) 1.1
RFC 1541	Dynamic Host Configuration Protocol (DHCP)
RFC 1533	DHCP Options
RFC 959	File Transfer Protocol (FTP)

The complete list of Internet RFCs and the associated documents is available on many Internet web sites. Interested readers are referred to **www.faqs.org/rfcs** and **www.rfc-editor.org** as starting points.

APPENDIX A: SOURCE CODE

The complete source code for the Microchip BSD FTP Server, including the demo applications and necessary support files, is available under a no-cost license agreement. It is available for download as a single archive file from the Microchip corporate Web site at:

www.microchip.com

After downloading the archive, always check the `version.log` file for the current revision level and a history of changes to the software.

REVISION HISTORY

Rev. A Document (10/2007)

This is the initial released version of this document.

NOTES:

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.

Trademarks

QUALITY MANAGEMENT SYSTEM
CERTIFIED BY DNV
ISO/TS 16949:2002

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://support.microchip.com
Web Address:
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

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

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

Kokomo
Kokomo, IN
Tel: 765-864-8360
Fax: 765-864-8387

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: 86-10-8528-2100
Fax: 86-10-8528-2104

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

China - Fuzhou
Tel: 86-591-8750-3506
Fax: 86-591-8750-3521

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 - Shunde
Tel: 86-757-2839-5507
Fax: 86-757-2839-5571

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

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

ASIA/PACIFIC

India - Bangalore
Tel: 91-80-4182-8400
Fax: 91-80-4182-8422

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: 886-3-572-9526
Fax: 886-3-572-6459

Taiwan - Kaohsiung
Tel: 886-7-536-4818
Fax: 886-7-536-4803

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

10/05/07