

APPLICATION NOTE 3552

Low-Cost Controller Includes Integrated Web Access

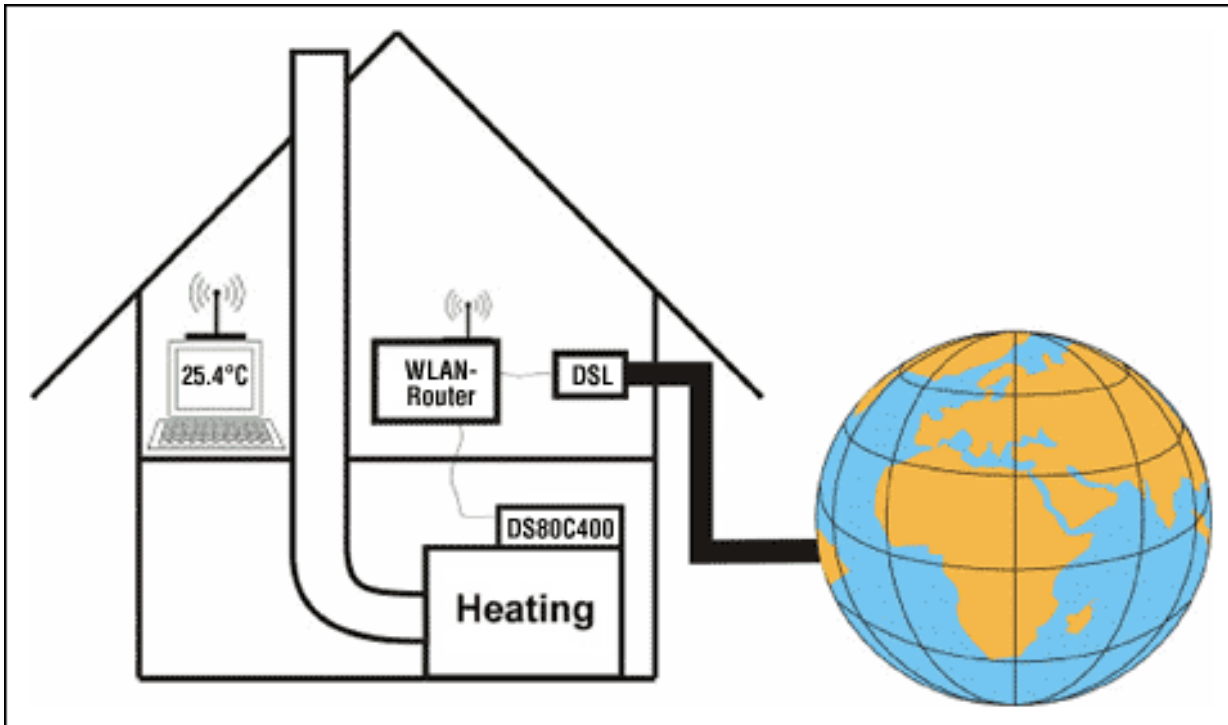
Abstract: Designing control, regulation, and data-communications systems without a network or Internet connection is like driving a car on secondary roads instead of the highways. This article presents a detailed and web-connected design for an inexpensive industrial application, using an innovative 8-bit network microcontroller (the DS80C400 from Maxim).

Introduction

Today, more and more functions are implemented over the Ethernet because that network greatly simplifies the system service and cabling. For simple tasks especially, we see an increasing number of microcontroller applications deployed over the Ethernet: wireless access points, monitoring cameras for security supervision, printer-servers, and routers.

It is now evident that a further reduction in software burden and cost requires integrating the Ethernet interface with the TCP/IP stack in the microcontroller core chip. That integration is already available in the DS80C400 8-bit, networked microcontroller.

The DS80C400 microcontroller is suitable for all the applications mentioned above, which operate around the clock and should ideally consume very little energy. The DS80C400 easily enables applications like Voice-over-IP and Internet-Phone. It can also control home air conditioning or heating systems, either over the internet or through the home network. By using the DS80C400, there is no need to install expensive single-block circuits at each unit of the system. Instead, the DS80C400 enables any PC in the system to view performance data and exercise control. Units connect only to the existing network (home, company, or internet), with service and installation performed by any PC on the network.

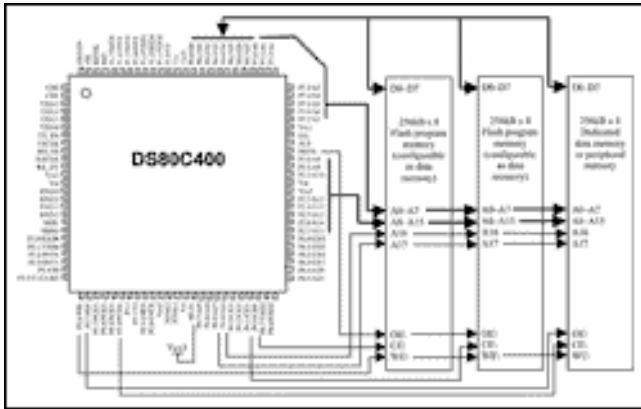


This microcontroller enables many new ways to visualize and monitor network performance. You can acquire and display temperature data from several locations in real time, or present a color-graphic display of different

temperature and conditioning cycles. Research and Development engineers can exploit several other monitoring and controlling possibilities, without adding a converter and expensive servo-actuators at each unit. Because the DS80C400 can connect to the public Internet through an Ethernet link, the operator's actual location becomes irrelevant. Remote and long-distance maintenance (or both) is easily done.

Low Power

The DS80C400 (**Figure 1**) was designed to implement simple but effective applications featuring low current consumption and minimal external circuitry. The reference design (in which the DSTINI400 and DSTINI400 together evaluate the DS80C400) includes 1MB RAM and 1MB flash memory in addition to the Ethernet and two serial ports. The total power dissipation, while running both the TINI-OS and Web-server operating systems at a clock frequency of 29.4912MHz, is only 0.5W.



[For Larger Image](#)

Figure 1. Address and data bus for the DS80C400 network microcontroller.

The microcontroller has a power-management mode (PMM) for which the internal clock is divided by 256. PMM therefore allows the microcontroller to operate at a very low speed, and with an ultra-low power-supply current (even lower than that of Idle Mode) which allows it to continue executing instructions. The microcontroller automatically reverts to normal mode when an interrupt command is issued or data is received through one of the serial ports. The power-hungry internal timers continue running in PMM, but the clock rate for the timers is divided by 1024 rather than 4! The microcontroller achieves this low energy consumption while supplying the 8051 core processor at 1.8V and the internal I/O drivers (which are 5V-tolerant) at 3.3V.

Maximum of 18.75 Million Instructions per Second (MIPS)

The DS80C400 executes code at rates to 18.75MIPS, at the maximum clock frequency of 75MHz. That rate is sufficient to receive a data stream of uncompressed audio over the Ethernet and to send it out again with a digital-to-analog converter (DAC). With this MIPS rate, the DS80C400 is also suitable in surveillance cameras with low resolution or lower refresh rate. The DS80C400 is not suitable, however, for use in applications that require higher and continuous throughputs, such as file servers and network DVD burners.

Interface and Peripherals

The DS80C400 (**Figure 2**) offers several interface choices, including a CAN 2.0B controller, three serial 1-Wire, full-duplex hardware ports, and eight bidirectional 8-bit ports, for a total of 64 digital I/O ports. An on-chip math accelerator allows the DS80C400 to perform 16- and 32-bit multiplication, division, shifting, and normalization. Featuring a 16MB address space, 22 address enables, and four internally encoded chip-select enables, the device integrates a 10/100Mb Ethernet-MAC for the half- and full-duplex-supported Internet Protocol version 4 (IPv4) and Internet Protocol version 6 (IPv6). To reduce the CPU load during transmit and receive, the microcontroller includes an 8kB packet-data memory with buffer.

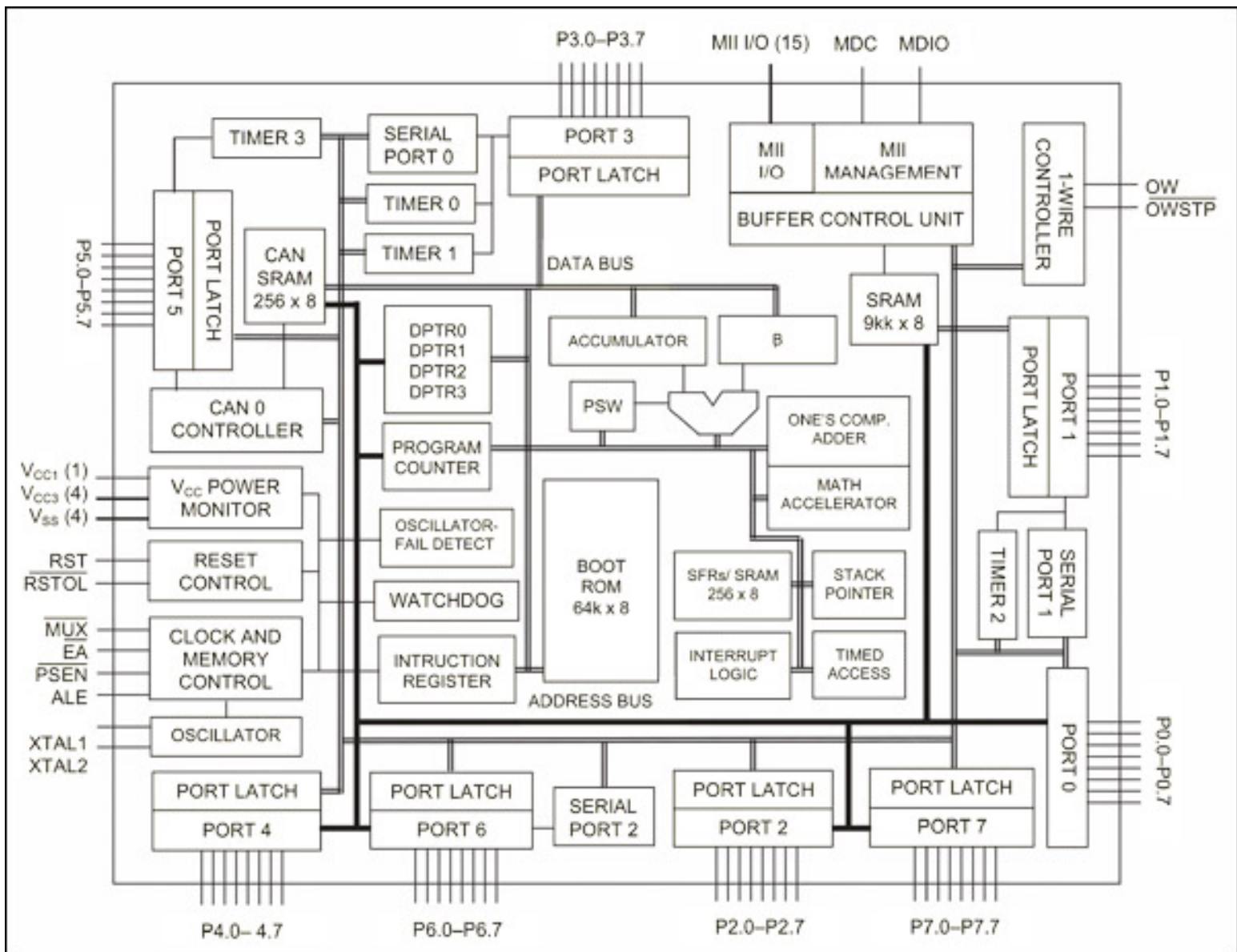


Figure 2. DS80C400 block diagram.

An embedded 64kB ROM provides instant connectivity and networking support. The ROM contains firmware that performs a network boot over an Ethernet connection using DHCP in conjunction with TFTP. The ROM firmware, which supports UDP, TCP, DHCP, ICMP, and IGMP, implements a fully application-accessible TCP/IP stack. As an option, you can acquire a MAC address from an IEEE-registered DS2502-E48, a 48-bit node address chip.

Java™

A free Java operating system, based on the OS in the 8051 core, is available for use with the DS80C400. Several example programs are available for download from the following [TINI product site](#). The Java operating system, TINI OS, is also available from the same site and offers a SLUSH-shell, an UNIX-similar shell with standard components like FTP and Telnet. Other simple example programs are available to download from the TINI website. You can, for example, find the complete Java source code for a web server which measures temperature with the DS1920 Temperature iButton (a temp sensor that provides a digital output using the 1-Wire protocol) and provides an HTML page showing the temperature.

A Practical Implementation

The following items are required to install a web server with TINI OS on the DS80C400:

- A hardware design including the DS80C400, a minimum of 1MB flash memory, and a minimum of 1MB SRAM. The TINIm400 module is a fully tested reference design which meets these requirements, and is available from Maxim. Also available is the TINIs400, a socket board for the TINIs400 which brings all interface signals out to connectors. See the [TINI page](#)
- The newest version of the [TINI OS](#).
- To compile a J2SDK 1.4.x, Linux/Unix, or Windows version, [download](#).
- To transfer the software with integrated serial loader in the DS80C400, Java-Communications-API, Linux/Unix, or Windows version, [download](#).

Step by Step at the Web

You first need to install both Java2SDK with all accessories and the JAVA-Comm API. See the file PlatformSpecific.html for the JAVA-Communication-API installation procedure. Then, unzip the downloaded Tini-Paket tini_13.tgz. Windows users should unzip the table into the main directory and use the simplest, shortest possible name, because you must often enter the full complete path manually. The following example assumes that the table is unzipped into C:\TINIOS. Next, to call the Java program, which communicates directly with the loader included in the DS80C400, you must now start the Tini JavaKit:

```
java -classpath c:\tinios\bin\tini.jar;c:\j2sdk1.4.2_05\lib\comm.jar JavaKit
```

Now that the JavaKit can communicate with the DS80C400, you must connect the PC to the DS80C400 using a Serial TxRx cable (straight through, not a null-modem) through serial port 0 (P3_B0 and P3_B1). To enable the DTR-Reset, you must use another serial transceiver to drive the reset pin of the DS80C400 low when the PC COM port DTR line is asserted.

The user must make sure that the DS80C400's pin 96 (P1_B7) is not connected to ground during a reset; otherwise, the device automatically tries to start a program on address 40000h. Also, pin 32 of the DS80C400 must not be pulled to ground; otherwise the ROM initiates the NetBoot process after a Reset from a TFTP server. After a Reset, the DS80C400 should now return the following information:

```
DS80C400 silicon software.
Welcome to the TINI DS80C400 car boat Loader 1.0.1
```

At this point, the user can copy both the TINI-OS-File with the Java-Kit over File→Load-File: C:\tinios\bin\tini_400.tbin into the FLASH on the DS80C400 circuit board, and the slush-shell over File→Load-File: C:\tinios\bin\slush_400.tbin. Before restarting, you should delete the HEAP (an area of memory allocated for data storage by user programs, similar to the stack).

```
b18: this command changes to the bank 18, where the HEAP is...
```

```
f0: this command fills the current memory bank with zeros.
```

Next, pull the DS80C400's pin 96 (P1_B7) to ground and generate a Reset. The following information should appear after the Reset:

```
[,= slush version 1.13 =,]
[System coming up. ]
[Beginning initialization... ]
[Need generating log file. ] [Info]
[Initializing shell commands... ] [Done]
[Checking system files... ] [Done]
[Initializing and parsings. startup... ]
[Initializing network... ]
[Starting DHCP client... ]
[Waiting of for DHCP IP Lease... ]
[DHCP IP of lease Successful. ]
[Network configuration] [Done]
```

```
[Starting up Telnet servers... ] [Done]
[Starting up FTP servers... ] [Done]
[System init routines] [Done]
[slush initialization complete. ]
```

Press any key to log-in.

The above output indicates that the operating system already runs on the DS80C400. Users can now log in by entering the user name, "root," and password, "tini." If your network does not support the Dynamic Host Configuration Protocol (DHCP), you must use a static IP address. To set the IP address to a static value, use ipconfig followed by:

```
-a 192.168.1.50 -m 255.255.255.0.
```

The resulting configuration is stored in the HEAP. If the SRAM is persistent like the one in the Reference Design (i.e., has a battery backup), the network parameters and system data remain intact even in the absence of primary power. Now, the JAVA-Kit and serial connection are no longer necessary. Users can copy their own software into the system file over the FTP server. Further configurations can be made over Telnet.

An example of a simple JAVA web-server application is visible on the table: c:\tinios\examples\TINIWebserver. You simply compile it over the batch file buildWebServer.bat. Users can copy the acquired file TINIWebServer.tini from the FTP into the table /bin in the DS80C400 file-system; start it from Telnet with the command java /bin/TINIWebServer.tini. To ensure that the web-server application starts when the system starts up, you can also write that command into the startup file (/etc/.startup, a file somewhat like autoexec.bat in your PC).

At this time you can run Webserver, which runs on the DS80C400 through Internet Explorer. Type the address <http://192.168.1.50>, as shown in **Figure 3**.

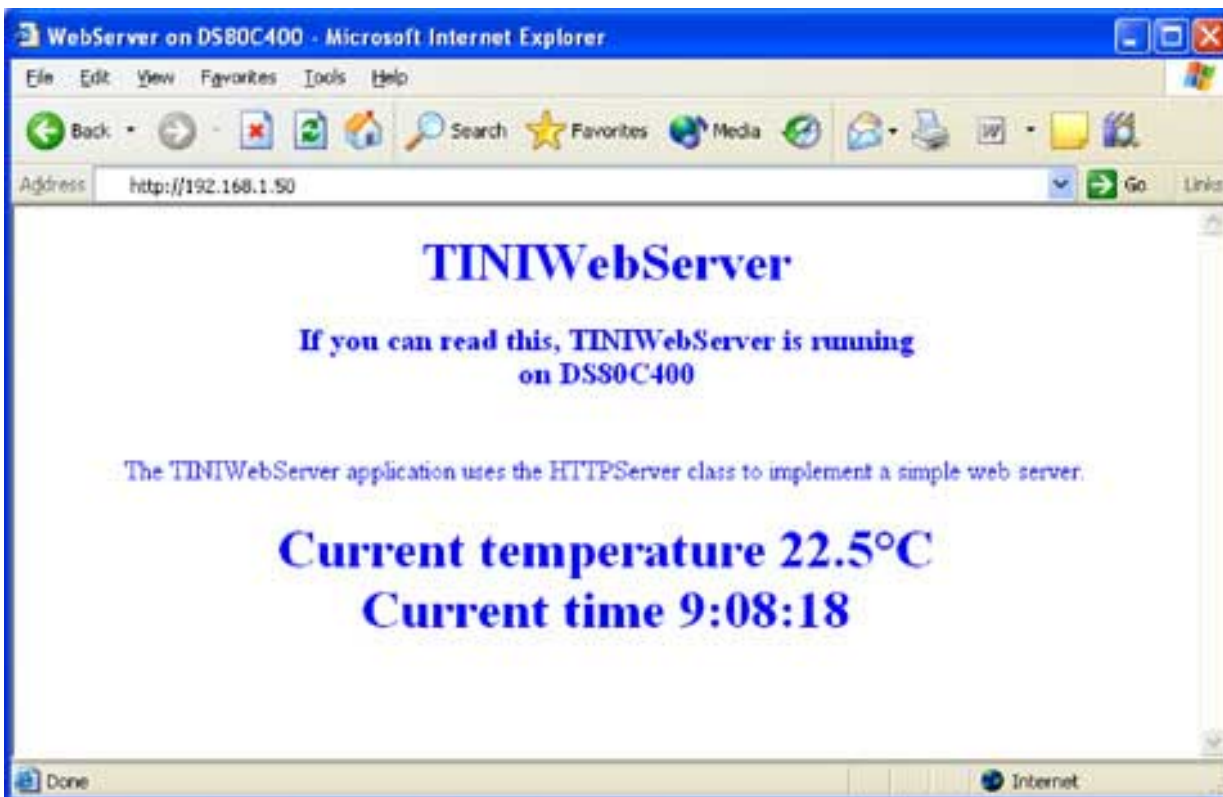


Figure 3. This screen gives you access to the web server running on the DS80C400. In the reference design, temperature is monitored by the DS1920 temp sensor connected to pin PIN99 (OW), while time information comes from the I2C bus on the DS1672 real-time clock (RTC).

JAVA

In the directory c:\tinios\examples you can see several Java programs for the DS80C400, all designed for the TINIOS operating system:

- Blinky: showing the access to the GPIOs of DS80C400
- canautobaud: showing automatic baud recognition for the CAN-Bus
- cantester: simple send-receive program for the CAN-Bus
- CommTester: showing a serial-to-Ethernet communication conversion
- Echo: send data received over Ethernet on an assigned port, then back
- I2CTest: showing the use of I2C bus with the temperature sensor DS1621
- ListOW: showing the use of 1-Wire bus, listing all existing components with 1-Wire address
- MemDisplay: showing memory statistic in graphic way
- OWDump: showing access to the 1-Wire bus
- PostExample: showing the HTTP-POST functionality
- PPPClient: showing the use of PPP-Class as PPP-Client.
- PPPServer: showing the use of the PPP-Class as PPP-Server.
- SPIExample: implementing a simple SPI interface

Programming in C

Many users would consider programming an 8-bit microcontroller only in assembly language, but very good C cross-compilers are now available. The Keil Compiler (www.keil.com) is recommended for the DS80C400. Download: [an extensive library in C](#).

The following files can be found in the library:

- 1-Wire public domain kit: access functions for the 1-wire bus.
- Crypto (SHA1, MD4): showing the SHA1 algorithms on an 8-bit array.
- Debug port: showing the debugging functions.
- DHCP client: receiving the IP address of a DHCP-Server.
- DNS: translating from host name to IP address.
- Enhanced Network Stack: enhancing the network stack.
- ROM error codes: listing the error codes returned by the internal ROM.
- File system: basic reading-writing-renaming-deleting file features.
- Flash programming: showing FLASH erasing and programming functions.
- I2C: all available library functions for writing and reading a DS1672 real-time clock.
- ISR installation: showing the use of interrupt vectors.
- Memory manager: showing the use of Malloc, MallocDirty, and Free.
- MIME encoder/decoder: used invisibly in the POP3 and SMTP sample applications.
- Netstat: getting current information on the state of the network stack.
- NTLM authentication: used in the POP3 sample application.
- POP3 client: prompting the user for e-mail account information, and attempting to retrieve information from the server for the specified e-mail account.
- Raw 1-Wire: searching the 1-wire bus for a device. If a DS2502 is found, its memory is displayed.
- RTC access: configuring the DS1672U real-time clock.
- SMTP client: showing how to send a simple canned test message to an e-mail account.
- SPI: providing a simple interface that allows communication with SPI devices using Port 5's PCEN pins.
- Task schedulers: spawning/killing/suspending/awakening a process at the user's request.
- TFTP client: allowing applications to set a TFTP server, then request files from that server.
- Utilities: giving applications access to various useful functions.

Java is a trademark of Sun Microsystems, Inc.

Application Note 3552: www.maxim-ic.com/an3552

More Information

For technical support: www.maxim-ic.com/support

For samples: www.maxim-ic.com/samples

Other questions and comments: www.maxim-ic.com/contact

Automatic Updates

Would you like to be automatically notified when new application notes are published in your areas of interest? [Sign up for EE-Mail™](#).

Related Parts

DS80C400: [QuickView](#) -- [Full \(PDF\) Data Sheet](#) -- [Free Samples](#)

DSTINIM400: [QuickView](#) -- [Full \(PDF\) Data Sheet](#)

DSTINIS400: [QuickView](#) -- [Full \(PDF\) Data Sheet](#)

AN3552, AN 3552, APP3552, Appnote3552, Appnote 3552

Copyright © by Maxim Integrated Products

Additional legal notices: www.maxim-ic.com/legal