Keywords: network micro, microcontroller, FAQ, MxTNI, C programming, Ethernet, networked microcontroller

APPLICATION NOTE 2792

DS80C400/DS80C410/DS80C411 Network Micro: Frequently Asked Questions

By: Kevin Self
Oct 27, 2003

Abstract: This FAQ discusses answers to general questions concerning the features and use of networked microcontrollers. Please note that the microcontroller and the MxTNI™ operating system are closely linked together.

General Communication Questions

Specific Technical Questions

What are the Networked Microcontrollers? What makes them unique?

Network microcontrollers allow designers to quickly and easily add Ethernet/Internet connectivity to their embedded systems. In addition to a 10/100 Ethernet MAC, the microcontroller has three serial ports, a controller area network (CAN) 2.0B controller, and 1-Wire Master networking capabilities. To enable access to the network, a full application-accessible TCP IPv4/6 network stack and operating system are provided in ROM. The network stack supports up to 32 simultaneous TCP connections and can transfer up to 5Mbps through the Ethernet MAC.

- 4-clock cycle 8051 core
- 16MB program memory + 16MB data memory contiguous addressing
- 10/100 Base-T Ethernet MAC
- Operation up to 75MHz (instruction cycle time of 54ns)
- Low power modes
- 3 serial ports
- 4 automatic increment/decrement data pointers
- 24 general-purpose I/O pins
- 16/32-bit hardware math accelerator
- 1-Wire Master
- 64kB ROM
  - Full TCP/IPv4/v6 network stack
  - Memory manager
  - Multitasking operating system
  - Automatic network boot

The DS80C410 is a variant of the DS80C400 with a maximum clock rate of 75MHz and 64kB internal SRAM. The DS80C411 is a variant of the DS80C400 with a maximum clock rate of 75MHz and 64kB internal SRAM but with the CAN module removed.

What is MxTNI?

The Maxim Tiny Network Interface (MxTNI), more properly referred to as the MxTNI Runtime Environment, is a Java runtime environment for developing network-aware applications for Maxim IP-ready microcontrollers such as the DS80C400. As IP networks such have become more pervasive, it is now necessary to network-enable embedded systems. However, network protocols tend to be complicated to code and require a lengthy test cycle. The runtime environment provides a full TCP IPv4/6 protocol stack verified for compliance to Internet standards. The network stack is driven by a multitasking operating system (MxTNI-OS). Using the runtime environment and its built-in APIs, a developer can quickly write embedded applications that are network-aware. Currently supported network protocols include:

- PPP
- IPv4/6
- TCP
- UDP

- IGMP
- ICMP
- DAD
- SMTP
- DHCP
- FTP
- HTTP
- TELNET

How do they interface to the Internet?

An internal 10/100 Base-T Ethernet media-access control (MAC) module is the data interface between the microprocessor and the Ethernet. It converts files or data into packets that conform to the data standards for Ethernet traffic.

Physical Connection to the Internet via a physical layer interface (PHY). It converts the 0V to 3V signals of the microprocessor to the 0V for the high value and -2.05V for the low value. The PHY is composed of an integrated circuit, transformer, and associated support circuitry. A jack connects the system to a standard Cat 5E cable, which in turn plugs into an Ethernet wall jack.

What is the network stack?

The network stack is the set of TCP/IP protocol layers that work together to define communication over the Internet. The software to manipulate these layers is stored in the internal ROM for easy software access. Users can access the stack automatically when programming via MxTNI, or it can be accessed from user-written C or assembly language routines. Both local and wide area networks can be accessed using the MxTNI stack. Direct support for Ethernet allows designs that connect to LANs. PPP enables IP over serial, which provides support for networking over wireless connections or through telephone lines using analog modems.

What is a serial/CAN/1-Wire-to-Ethernet bridge?

Often a system needs to convert from one communication protocol to another. For example, a piece of factory equipment might have a serial RS-232 interface, but needs to communicate with a supervisory computer with an Ethernet interface. Networked Microcontrollers are ideal for implementing a bridge between these systems. With four serial ports, a CAN interface, and a Maxim 1-Wire interface, a DS80C400-based design can serve as a high-speed intelligent bridge between several kinds of networks. An example of an Ethernet to serial bridge can be found at MxTNI Networked Microcontrollers.

What is the maximum throughput of the serial ports if they are accessed via the MxTNI OS?

For a serial port configured at a baud rate of 115,200bps and a system clock frequency of 36MHz, the maximum transmit and receive rate is approximately 10kB per second. The throughput is highly dependent on CPU loading and will vary per application.

What is the maximum transfer rate of the Ethernet port using sockets?

For a system clock frequency of 36MHz, the maximum transmit and receive rate is 266kB per second.

What documentation is necessary?

The following documents are required to make full use of the DS80C400:

- DS80C400 Data Sheet
- High-Speed Microcontroller User's Guide
- High-Speed Microcontroller User's Guide: Networked Microcontroller Supplement

What development tools are available?

The TStik evaluation boards manufactured by Systronix (www.systronix.com) allow developers to use the DS80C390 or DS80C400 in a single-board computer or evaluation board format.

The MxTNI Software Development Kit (SDK) is a royalty free development tool that incorporates the programming API and MxTNI JAVA runtime environment with examples and documentation.

The TINI™ SoM-400EM module is available from EMAC, Inc. The module is based on the DS80C400 networked microcontroller and can be ordered to be pin-compatible with the DSTINIm400.

Is there a reference design I can start from?

We have placed reference design schematics on the web site at MxTNI Board Schematics and Software.

For issues specific to the microcontroller itself, or not appropriate to the mailing list, support is also available via Tech Support.

How can I program the DS80C400? Do I have to code in Java?

Code for the Networked Microcontrollers can be written in Java, C, or 8051 assembly. The MxTNI runtime environment also supports the DS80C400.

Java: Java compilers from Sun Microsystems and Borland are compatible. Java is not required to use all the Ethernet capabilities, but it is the simplest and preferred way to program in the MxTNI environment. In addition, the largest number of support tools and libraries are available for the Java environment. Compilers are available from http://java.sun.com/downloads/, specifically a 'Java 2 Platform, Standard Edition' (J2SE) package; version 1.2.2, 1.3.1, or 1.4.1 are acceptable. The Java Communications API from http://java.sun.com/products/javacomm/ is also required.

C: SDCC (http://sdcc.sourceforge.net) and Keil Software (www.keil.com) have C compilers available. A traditional 8051 compiler can be used, but only the PK51 C compiler offered by Keil Software supports for the expanded address space of the DS80C400 and the ROM-based network stack. The full TCP/IPv4/6 network stack and a small operating system are in the ROM of the DS80C400 and can be accessed from user-written application software. The home page for the C Libraries is http://files.maximintegrated.com/microcontroller/mxtni/ that contains libraries and sample applications built with the Keil tools.

Do I have to pay royalties on the use of your TCP/IP stack?

Unlike other networking solutions, Maxim charges no royalties for the use of its ROM-based TCP/IP

stack.

Do I have to pay royalties on software derived from your MxTNI operating system or any sample/example code provided by Maxim?

No. You can use or adapt our sample/example code as needed for your design. However, when using the MxTNI Java Runtime Environment, a DS2502P-E48 is required to obtain the network MAC address. In addition, using the DS2502P-E48 is proof of acceptance of the MxTNI Runtime License. If you prefer to obtain your Ethernet MAC address from a different source, please contact software support.

Do I have to pay royalties on products that I design based on your hardware reference designs?

No.

Are there reference books about programming with C, Java, TCP, etc.?

There are many books available from bookstores and Amazon.com. Recommendations by our engineering staff include:

- The TINI Specification and Developer's Guide by Don Loomis; Addison-Wesley, 2001.This is currently out of print but used copies are available via Amazon.com.
- TCP/IP Illustrated, Volume 1: The Protocols by W. Richard Stevens; Addison-Wesley, 1994.
- Thinking in Java by Bruce Eckel; Prentice Hall PTR, 2000.

Specific Technical Questions

What interface devices are required to connect to the Ethernet?

To connect the microcontroller (and associated memory) to the Internet you will need a physical layer interface (PHY) device for connecting to a network interface like 10/100 BASE-T or fiber. Our reference design uses the Intel LXT972ALC, but any Media Independent Interface (MII)-compatible PHY can be used. In addition, our reference design uses the Belfuse S558-5999-T7 as its transformer.

How does the device get its Internet MAC address?

On boot, the DS80C400 automatically searches its external 1-Wire bus for an external DS2502-E48 device (sold separately). If found, the DS2502-E48 supplies a unique IEEE Ethernet MAC address to the DS80C400. It is also possible to program an Ethernet MAC physical address via the user-application software.

What voltages does the DS80C400 require?

The DS80C400 requires both a 1.8V supply and a 3.3V supply. The I/O pads of the device are powered by the 3.3V supply, allowing the device to interface to 3.3V logic. The 5V tolerant I/O of the microprocessor can be interfaced to 5V peripherals. The order of sequencing of V_{CC1} and V_{CC3} is not important.

If I write in C or assembly language, how do I access the stack?

The network stack and scheduler are located in the internal 64kB ROM. The functions inside the ROM are accessed as BSD socket layers or APIs. Maxim provides a BSD socket interface for those programming in C.

The network stack is also accessible and can be called from assembly language. We provide an assembler as part of the MxTNI SDK.

How do I port code written for DS80C390 to the DS80C400?

The MxTNI runtime environment supports both microcontrollers. The most significant differences between the two devices are that the DS80C400 includes the Ethernet MAC and a Maxim 1-Wire interface, and has only one CAN module. Unless you are using both CAN controllers in the 390, there should not be any porting issues. The code base is entirely compatible.

What is the minimum memory configuration?

A minimal network-enabled system requires 64kB of SRAM. Application code can be downloaded via the network to the SRAM. Network initialization (Netboot) can be performed on a blank unit connected to the network. External flash/EPROM can be used if nonvolatile program memory is desired, but is not necessary.

The access speed of the memory is dependent on the operating frequency and the board design. For an example, however, we often cite that 70ns RAM and flash are required for a system running at 36MHz. To run at full speed, you need to use 15ns RAM or faster.

How is the program memory loaded into a DS80C400-based design?

The microcontroller is equipped with a ROM (bootstrap) loader that can configure certain features of the microcontroller. It can also be used to load software into NV SRAM, which will be used as the program memory. It supports the loading of any Advanced Micro Devices flash memory device that meets the speed and size/format requirements of your specific design. Details of enabling the ROM loader in your design can be found in the High-Speed Microcontroller User's Guide: Network Microcontroller Supplement.

The ROM loader attempts to autobaud to the incoming serial stream using an internal counter clocked by the external clock source (crystal or oscillator). Because the autobaud feature is dependent on the external clock source, we suggest using an 18.432MHz crystal with X4 mode and run at about 73MHz. This frequency generates allows the autobaud routine to synchronize to a wide range of standard baud rates.

Which external CAN transceivers are recommended for the DS80C390/DS80C400/DS80C410?

The following Maxim CAN transceivers are recommended:

- MAX3050: Fault-Protected/Tolerant CAN Transceiver
- MAX3053: Fault-Protected, 2Mbps, Low Current CAN Transceiver
- MAX3054: Fault-Protected/Tolerant CAN Transceiver
- MAX3055: Fault-Protected/Tolerant CAN Transceiver

- MAX3057: ±80V Fault-Protected, 2Mbps, Low Current CAN Transceiver

Related Parts		
DS80C400	Network Microcontroller	Free Samples

More Information
For Technical Support: http://www.maximintegrated.com/support
For Samples: http://www.maximintegrated.com/samples
Other Questions and Comments: http://www.maximintegrated.com/contact

Application Note 2792: http://www.maximintegrated.com/an2792
APPLICATION NOTE 2792, AN2792, AN 2792, APP2792, Appnote2792, Appnote 2792