Technical Note TN2004

Debugging Open Firmware Using Telnet

CONTENTS

This Technote addresses debugging your device's firmware using Ethernet and the telnet protocol to connect a client machine running a telnet application to a target machine running Open Firmware.

This note is directed at PCI device designers, driver writers, diagnostic writers, and others. In fact, any engineer or scientist who understands how to communicate at a local bus level with a PCI peripheral may find this technical note interesting.

If you decide to use this telnet method of debugging, be aware that some of the tools used in the Technote are shareware.

Updated: [Oct 16 2000]

One machine Open Firmware debugging versus two machine debugging

One machine Mode:

Since the introduction of the PowerBook 3400 (introduced in early 1997), all Macintoshes enter the Open Firmware user interface in **one machine mode**. In one machine mode, the user can type Open Firmware commands on the keyboard and see the results on the main display. One machine mode has its advantages and disadvantages.

The primary advantage of one machine mode is that it is quick and easy to use. Just start your Macintosh while pressing the Command-Option-O-F keys. Once in Open Firmware, you can view the device tree, add a property to a node, or try a new colon definition. However, when you shut down the machine, anything that was entered at the keyboard is lost, and that's a big disadvantage.

More complex work, such as debugging your device's expansion ROM Forth code, routinely requires you to save and review your work. There are real problems with one machine debugging such as the lack of scroll bars (to let you view past history of commands and results) and the inability to save your session.

Two machine Mode:

Two machine mode refers to an editing and debugging environment in which the **target** machine runs Open Firmware and a second, or **client** machine acts as your keyboard and display. Two machine mode is preferred to one machine mode because it allows you to save all or part of your session on the client machine. This allows you to Copy and Paste commands instead of retyping them. This is especially useful for restoring your operand stack after an interpreter error (which clears the stack).

Two machine mode originally involved connecting the client machine, running a terminal application, to the target machine via a serial cable. However, beginning with the iMac, traditional serial ports have been removed from the logic board, making it difficult to use two machine mode.

This document describes a method of using Ethernet and **telnet** to connect client and target machines in order to regain the benefits of two machine mode debugging. This method requires firmware functionality first introduced in some computers announced in the summer of 2000.

Back to top

Gathering your tools

The tools needed for two machine telnet debugging are as follows:

1. A target machine that is telnet capable.

   This target is the machine on which you want to run Open Firmware in order to debug. The target will act as a **telnet server**.

2. A client machine running Mac (or any) OS with an Ethernet port and a **telnet client application.**

   This machine can be any machine with an Ethernet port capable of running a telnet client application. This machine will be used to communicate with the machine running Open Firmware just as the client terminal application communicated to the target over a serial cable.

3. An Ethernet cable to connect the two machines.

   The cable is either a normal or crossover Ethernet cable with RJ-45 connectors at each end. The type of cable depends on the kind of Ethernet network you are using. Refer to the "Using the correct Ethernet connection" section to determine the appropriate cable. It is strongly recommended that a crossover cable connection is used to simplify communications between the client and target machines.

4. A telnet client application.

   This is the telnet client application that will run on the client machine and will communicate with the Open Firmware machine's telnet server. There are several freeware and public domain telnet client applications for the Macintosh.

   NCSA Telnet can be found at: NCSA telnet
   BetterTelnet can be found at: Bettertelnet
   NiftyTelnet can be found at: NiftyTelnet

5. A functional Ethernet network or a direct connection.

   A functional Ethernet network can be anything from a Local Area Network, (DSL, T1, and so forth) connection to a single normal cable connection between two machines. It can also be two machines directly connected via a crossover cable.

   A crossover cable is recommended because the Open Firmware telnet server may see external packets with checksum errors. While these packets will not originate from your telnet client, they may disrupt your client display. Apple is looking into these error messages and there may be changes to the telnet server on future Macintoshes.

6. A valid IP address.

   This is required for both the client and server machines.

Back to top

Determining telnet capability

To determine if your target machine is telnet ready, do the following:

At the Open Firmware user interface, go to the packages node in the device tree by typing

```
dev /packages ls
```

followed by a carriage return. The packages node is a support node that can be used by device nodes. In our case the `/packages/telnet` node is used to support the connection between the two machines. If your machine has the `/packages/telnet` node, then you can debug over Ethernet.

```
0 > dev /packages ls
ff83ebf8: /deblocker
ff83f518: /disk-label
ff83ff18: /obp-tftp
ff847048: /telnet        -----look for this node----
ff8478c8: /mac-parts
ff8489a0: /mac-files
ff84b7b0: /hfs-plus-files
ff850520: /fat-files
ff852250: /iso-9660-files
ff852e58: /bootinfo-loader
ff854af8: /xcoff-loader
ff855510: /pe-loader
ff855ee8: /elf-loader
ff857518: /usb-hid-class
ff859858: /usb-ms-class
ff85bbf8: /sbp2-disk
ff85e1c0: /ata-disk
ff85f420: /atapi-disk
ff860af8: /bootpath-search
ff8673e8: /terminal-emulator
ok
0 >
```

Listing 1. Open Firmware packages listing.

Back to top

Using the correct Ethernet connection

Occasionally, sending Open Firmware commands over an Ethernet network can be slowed down by Internet traffic and unwanted broadcast messages. To avoid Internet traffic and allow a point to point connection between two 10BaseT Ethernet devices without using a hub, a crossover cable is recommended. Crossover cables can be either constructed or purchased.

A crossover cable reroutes the transmit lines from one RJ45 Ethernet connector to the receiving lines of another RJ45 Ethernet connect. Information about constructing and connecting a crossover cable between two Macintosh computers is summarized in the Tech Info Library Article:

Macintosh: Transferring Information Between Computers Using Ethernet. Article ID: 43015

Other Web sites describing the construction of crossover cables can be found at the following URLs.

1. Ethernet Crossover Cable Pin Out
2. Ethernet Crossover Cable How To Guide
3. RJ45 Connector

Crossover cables can be purchased at almost any outlet that sells or supplies computers or computer peripherals. The following web site is one of those sites.

We Sell Cables

Back to top

Setting up your machines

You will need to connect the client and target machines to a functional Ethernet network using the appropriate cables. As discussed above, Internet traffic can complicate communications between the client and target machines and it is therefore recommended that you use a crossover cable between the two machines. The rest of this discussion assumes the use of the crossover cable.

Having obtained the items in the Gathering your tools section above, take the following steps to establish communications between the client and target machines:

1. Connect the client machine to the target machine using the crossover Ethernet cable.

The client machine will run a telnet application while the target machine will run the Open Firmware user interface.

2. Choose an IP address to be assigned to the target machine.

   There are reserved IP Addresses for setting up a standalone Ethernet network. A standalone Ethernet network can be just the target and client machines connected via a crossover cable, with no connection to anything else. Addresses of the form `10.x.y.z` can be used for this kind of Ethernet network. In this example, we'll use the address `10.1.2.3`.

3. In the target machine's Open Firmware user interface type:

   ```
   " enet:telnet,10.1.2.3" io
   ```

   followed by a carriage return. Note that the spaces after each quote mark are required. If you chose a different IP address in the step above, substitute it accordingly.

   This invokes a telnet server on the target machine, waiting for some other system to ask to connect to that address.

4. On the client machine with the telnet client application, initiate a connection to the target, using the same IP address that was chosen above (in our example, `10.1.2.3`).

   While the telnet client application must use the same IP address to connect to the target, make sure that the client machine itself (running the telnet client application) has a different IP address in its TCP/IP control panel than the target machine's IP address you entered when invoking the telnet server.

Once the telnet client application has established communication with the target machine, the Open Firmware user interface is moved to the client machine running the telnet client application. This means you have scrolling, editing, and saving capabilities. Editing and debugging on the target machine can be continued on the client machine without losing information or data.

Back to top

Summary

Debugging any code during the boot sequence can be difficult. This is due to system resources not being available while the sequence is in progress. This is especially true for any Forth code embedded in a peripheral device. The Open Firmware user interface in one machine mode can limit a developer's ability to save information and therefore, slow the debugging process.

Using telnet as a two machine mode connection is just one more tool that may be of help during the development cycle. It adds the ability to scroll, copy and paste, and save your session.

Back to top

References

Apple's PCI Development Web Page

Apple's Hardware Technical Note Collection

Apple's Hardware Technical Q&A Collection

Back to top

Downloadables

Acrobat version of this Note (K) Download

Back to top
