Applied
Microsystems
Corporation

NetROM™ 500 Series

Installation Notes

May 1997

Contents

NetROM Installation Notes

Introduction

Installing NetROM on your network is similar to installing a new workstation. Like a workstation, NetROM is a multitasking computer. Also like a workstation, NetROM requires some configuration before it can operate effectively with the other machines on your network.

Consequently, we suggest that a network system administrator set up NetROM for your site. After NetROM is connected to the network, the only subsequent configuration necessary is to edit the NetROM startup file. Editing the startup file needs to be done in the early stages of every project in which you plan to use NetROM to debug a new target system. See Chapter 4 in the *NetROM User's Manual* for how to do this.

NetROM uses the network services BOOTP (or RARP) and TFTP for address resolution and file transfer. Chapter 2 of the *NetROM Installation Notes* contains tips on setting up address resolution and file transfer servers on your network.

Support Services

Applied Microsystems provides a full range of support services. New products are covered by a 90-day warranty. Additional support agreements are available to provide additional services.

If you have trouble installing or using your software, consult your manuals to verify that you are following the correct procedures.

If the problem persists, call Customer Support. Customers outside the United States should contact their sales representative or local Applied Microsystems office. When you contact Customer Support, have your ASI number available.

Telephone

800-ASK-4AMC (800-275-4262)

Internet address

If you have access to the Internet, you can contact Applied Microsystems Customer Support using the following address:

support@amc.com

You can also browse the Applied Microsystems World Wide Web page using the following URL:

http://www.amc.com

See the Applied web page for NetROM Application Notes.

FAX

(425) 883-3049

Chapter 1
NetROM Software Setup

To use NetROM most effectively, it helps to have address resolution and file serving software running on a host system. NetROM's software setup essentially consists of configuring one or more host systems to provide these two basic services.

At power-up, NetROM knows its Ethernet address, but not its IP address. While it is possible to configure NetROM's network address directly, using NetROM's serial console, it is much simpler to have NetROM resolve its network address automatically.

NetROM uses TFTP as a file transfer protocol. NetROM requires a TFTP server on your network to download images into emulation memory. If you are not familiar with TFTP, consult "File transfer servers" on page 2-6.

Once address resolution and file server software is running in your development environment, NetROM can be used anywhere there is an Ethernet connection. Serial lines and "dumb" terminals will not be necessary to debug your target system.

Address resolution

At power-up a NetROM unit does not know its own IP address, but it does know its hardware (Ethernet) address. Determining one's own network address is a common problem for diskless computers, such as diskless workstations. Such computers require a mechanism to determine their IP address, knowing only their hardware address.

Protocols

Two software protocols are commonly used to resolve IP addresses: **BOOTP** and **RARP**. When NetROM powers up, it sends out simultaneous RARP (Reverse Address Resolution Protocol) and BOOTP requests. These requests are broadcasts, so all systems on the Ethernet "hear" them. BOOTP and RARP "servers" listen for these requests, look up the requestor's IP address based on the hardware address in the request frame, and send a response. If NetROM receives a response to either its BOOTP or RARP request, it uses the information in the response to configure its IP address. If no response is received, it will re-broadcast the requests 10 times and finally give up. The NetROM unit should then be either power cycled or configured from the NetROM console.

BOOTP and RARP are simple protocols; often, the most complicated part of configuring a host computer as a "server" is setting up configuration files and making sure the server daemon is running. Once the server is responding to NetROM's requests, NetROM should have no problem determining its IP address, regardless of how often it is reset, power cycled, or moved. Consult the "System configuration files" on page 2-11 for details on configuration file setup.

If both BOOTP and RARP are available in your development environment, the choice of which to use should be based on convenience. If NetROM receives both a BOOTP and a RARP response simultaneously, it will use the BOOTP response to configure its address. If you are not familiar with BOOTP or RARP, consult "Address resolution servers" on page 2-1.

BOOTP address resolution

BOOTP, like TFTP, is a protocol which uses the connectionless UDP transport protocol. There are two "flavors" of BOOTP servers, the "CMU" and the "Stanford" versions. These differ only slightly; the CMU version allows the server to specify the client's IP netmask and broadcast address, as well as its IP address. Both versions provide a TFTP server address and a download filename to request from the server.

NetROM understands BOOTP responses from either of the server types. However, the configuration file formats for the two types of servers are completely different, even though the configuration files have the same names. This often results in great confusion among people who are trying to set up BOOTP on their Ethernet, especially since errors in the configuration file cause the BOOTP server to quietly ignore requests!

If a configuration file is not explicitly specified in a BOOTP response, NetROM will not try to download one.

RARP address resolution

RARP is another way to establish a mapping between an Ethernet address and an IP address. Unlike BOOTP, RARP does not use UDP as a transport. In fact, it is a variant of ARP, the Address Resolution Protocol in common use throughout TCP/IP networks, which does not even use IP as a network protocol. Also unlike BOOTP, the RARP protocol does not provide a mechanism to specify a download file. However, there is a convention which allows the RARP client to determine what configuration file it should download, as well as determine its IP address.

NetROM will map RARP responses to startup file names as follows. The RARP response consists of an IP address, which is a 32-bit value. NetROM converts the individual numbers in its dotted-decimal Internet address to two-digit hex numbers. For example, given the address 192.0.0.210, 192 becomes 0xC0, 0 remains 0x00, and 210 becomes 0xD2. NetROM will then concatenate these numbers to produce the filename (in uppercase letters) *C00000D2*. NetROM will then attempt to download one of the following startup files: "C00000D2," then "/tftpboot/C00000D2," and finally "tftpboot/C00000D2." After the first successful download, it will proceed with its boot sequence and execute the commands in the startup file. It will *not* attempt to download other startup files.

Since the RARP protocol cannot specify a TFTP server, NetROM assumes that the TFTP server for the configuration file is the same as that which responded to the RARP request. See "File transfer servers" on page 2-6 for further information on TFTP.

If this configuration file does not exist, or if there is not a TFTP server running on the RARP server host, NetROM will display appropriate error messages on its serial console and use its default configuration values (see Appendix E of the *NetROM User's Manual*).

Chapter 2 UNIX Environment Tips

This chapter provides tips on various aspects of configuring a UNIX environment to support NetROM. It is not a comprehensive "how to" manual because of the variety of possible development environments. Rather, it provides background information and pointers to additional information.

If you have difficulties in installing your NetROM unit, consult your local documentation on related servers and files.

Address resolution servers

At boot time, NetROM. is capable of using either of two address resolution protocols to determine its IP address: RARP and BOOTP. Both are described in the following sections. Servers listen for broadcast Ethernet frames containing protocol requests. When such a request is received, the server looks up the requester's Ethernet address in a configuration file. If it finds a match, it sends a reply containing the requester's IP address. BOOTP replies may contain other configuration information as well.

After you have decided which address resolution protocol to use, you need to update the configuration file for the server. If you are not sure which servers you have available on your network, consult "Finding servers on your network" on page 2-14. Note that part of NetROM's address resolution process involves downloading a configuration file. This download is done using TFTP. If you are not familiar with TFTP, or are not sure whether your server is running in "secure" mode or not, consult "File transfer servers" on page 2-6.

This section describes how to configure standard UNIX servers. If you have problems, you should consult your own documentation on the server. Local documentation may reveal discrepancies such as nonstandard configuration file locations, and provides much more detail than can be given in this section.

BOOTP

The BOOTP address resolution protocol is quite flexible. The protocol allows the server to specify the client's IP address, as well as the name of a configuration file and a TFTP server to contact to download the file. BOOTP servers for UNIX systems generally come in one of two varieties: the "Stanford" server and the "CMU" server. The CMU server is able to specify IP netmask and broadcast addresses for the client, in addition to filename, server, and IP address. NetROM is capable of understanding responses from both server types; however, because the servers have different configuration file formats, so it is important to determine which type you have on your system.

BOOTP servers, like TFTP servers, are generally invoked in one of two ways. They may run continuously, as a daemon process that is started when their host machine is booted, or they may be invoked by the *inetd*. The *inetd* is a daemon process that listens for requests for a variety of services and invokes the appropriate servers automatically.

Consult "inetd" on page 2-8 for more information. Both types of BOOTP servers read the bootptab file for configuration information. Generally this file is located in the /usr/etc or the /etc directory. The location of the bootptab file can be specified when the server is invoked, so you may have to do some detective work to find it (look for bootp in /etc/rclocal).

CMU BOOTP Server

The format of the bootptab file for CMU BOOTP is a set of "stanzas" separated by newlines. Each stanza is broken up into fields specifying what a BOOTP response to a particular client should contain. Fields within a stanza are separated by colons, and stanzas can be continued on subsequent lines using escaped newlines. For example, the stanza

```
netrom1:hd=/work/project/build:bf=startup.bat:\
:sm=255.255.255.0:ht=1:ha=0002f4000024:ip=192.0.0.210
```

specifies the following information:

- Symbolic name of the client is **netrom1.**
- Client's startup batch file is in directory **/work/project/build.**
- Name of the client's startup batch file is **startup.bat.**
- Client's subnet mask is **255.255.255.0.**
- Client's "hardware type" is **1** (which stands for Ethernet).
- Client's hardware address is **00:02:f4:00:00:24** (note that colons are not used as separators on the stanza).
- Client's IP address is **192.0.0.210.**

Note that the "hd" and "bf" fields are concatenated to give the full path to the configuration file. In this case, /work/project/build/startup.bat is the full path. If the TFTP server is running in "secure" mode, the file will actually reside in /tftpboot/work/project/build/startup.bat. Consult documentation on your server for other stanza fields.

➤ **To use a CMU BOOTP server to provide address resolution for NetROM:**

You must add a line similar to the above example to your bootptab file. If your BOOTP server is running continuously as a daemon, you need to notify it that you have changed the bootptab file by sending it a SIGHUP signal. See "Daemon conventions" on page 2-10 for more information.

Stanford BOOTP server

The format of the bootptab file for the Stanford server is less complex than that of the CMU server. To configure a Stanford server for a hypothetical NetROM unit, we would add a line like the following:

```
netrom1 1 00:02:f4:00:00:24 192.0.0.210 /work/project/
build/startup.bat
```

It specifies the following information:

- Symbolic name of the client is **netrom1.**
- Client's "hardware type" is **1** (which stands for Ethernet).
- Client's hardware address is **00:02:f4:00:00:24.**
- Client's IP address is **192.0.0.210.**
- Client's startup batch file is **/work/project/build/startup.bat.**

NetROM will use the IP address of the BOOTP server as the IP address of the TFTP server to contact for the startup batch file.

The Stanford server is capable of using defaults for the startup file, and indeed for the whole record. Such defaults are beyond the scope of this document.

➤ **To use a Stanford BOOTP server to provide address resolution for NetROM:**

You must add a line similar to the above example to your bootptab file. If your BOOTP server is running continuously as a daemon, you need to notify it that you have changed the bootptab file by sending it a SIGHUP signal. See "Daemon conventions" on page 2-10 for more information.

RARP

RARP is another way to establish a mapping between an Ethernet address and an IP address. Unlike BOOTP, the RARP protocol does not provide a mechanism to specify a download file. However, there is a convention that allows the RARP client to determine what configuration file it should download, as well as what its IP address is.

Also unlike BOOTP, RARP cannot be invoked by the *inetd*; if it is running at all it is running as a daemon.

The ethers file configures RARP servers. The ethers file is similar to the hosts file, except that the first field is an Ethernet hardware address, not an IP address. For example, the line

```
00:02:f4:00:00:24        netrom1
```

tells the RARP server that requests coming from hardware address "00:02:f4:00:00:24" should be responded to with the IP address of "netrom1". The mapping between the symbolic name "netrom1" and the IP address "192.0.0.210" is accomplished through the /etc/hosts file.

NetROM maps RARP responses to startup file names as follows. First it converts the individual numbers in its dotted-decimal Internet address to hex. For netrom1, 192 becomes 0xC0, 0 remains 0x00, and 210 becomes 0xD2. NetROM then concatenates these numbers to produce the filename (in uppercase letters) C00000D2 and will attempt to download this file from the same network host that responded to the RARP request. Since NetROM cannot know whether or not the TFTP server is running in secure mode, it actually requests /tftpboot/C00000D2. If your server is running in secure mode, the batch file should actually reside in /tftpboot/tftpboot/C00000D2.

➤ **To use RARP to provide address resolution to NetROM:** You must add a line similar to the one above to your ethers file. The ethers file is usually located in the /etc directory; however, if you are using the NIS (also called the Yellow Pages) to maintain the ethers file, you need to update the database on the NIS server and do a "yppush" operation. If NIS is running, the local ethers file may be silently ignored. Consult "Using NIS (Yellow Pages)" on page 2-13 for more information on NIS.

File transfer servers

TFTP is a standard file transfer protocol often used to provide boot images to standalone devices. The TFTP server can be run on any number of nodes on a network. TFTP servers generally have a “home” directory that they use as the default place to look for files when they receive requests. For example, if a server receives a request for startup.bat, it looks in its home directory for the file. The default home directory is /tftpboot, so by default the server looks for the file /tftpboot/startup.bat when it receives such a request.

This section describes standard UNIX servers. If you have problems, you should consult your own documentation on the server. Local documentation may reveal discrepancies such as nonstandard home directories, and provides much more detail than can be given in this section.

Server modes

The server can be run in either of two modes: “secure” mode or “normal” mode. These two modes differ primarily in the way they handle root-specific filenames in requests. Root-specific requests are those that specify directory location, starting at the root of the directory tree. For example, the client may request the file /startup.bat. Servicing this kind or request could lead to a security risk; the client could request any file on the server system, and since TFTP performs no authentication, the client would get the file. Secure TFTP was developed to get around this problem. Secure TFTP servers respond to root-specific requests as if their home directory (usually /tftpboot) were the root directory of their machine. Thus, a request for /startup.bat results in downloading /tftpboot/startup.bat from a secure server. Secure servers cannot be circumvented using symbolic links or similar tricks.

The problem with secure servers is they prevent legitimate clients from downloading files outside of the home directory subtree. For example, NetROM may want to download a file from an engineer's development directory, but secure-mode TFTP can make this impossible. There are at least two ways to handle this problem:

- do not use secure TFTP, or
- copy images into the server's home directory subtree after each modification

To determine what the home directory is for your server, and to find out whether it is running in secure mode, it is necessary to find out how the server is invoked. TFTP servers can be run as daemons that are invoked at the server system's boot time, or they can be started by the *inetd*. Consult "Daemons on UNIX systems" on page 2-7 for information on daemons on UNIX systems. After you have determined how the server is invoked, it is relatively simple to find out what its home directory and security mode are. Generally a -s flag indicates a secure server, and any kind of pathname indicates a non-default home directory. Consult local documentation for further information on your server.

Daemons on UNIX systems

On UNIX systems, "daemon" is a term used to refer to processes that run constantly in the background waiting for requests. Network servers are usually run as daemons because they are invoked more or less at random by clients. Daemons are usually started up at system boot time. There are two shell script files which are executed at system boot; these are /etc/rc.boot and /etc/rc. These shell scripts can call other shell scripts, such as /etc/rc.local or /etc/rc.ip; daemon processes are usually started by rc or rc.local.

If you know that a daemon, such as the TFTP server, is running, and you want to know what its invocation parameters were, you should examine the various "rc" scripts in /etc. It is often useful to know what the invocation parameters for a daemon are; in the case of the TFTP server, they will help you find out what the server's "home" directory is, and whether or not it is running in "secure" mode. The UNIX command:

```
unixprompt> grep -i rarp /etc/rc*
```

will look through all of the rc scripts and prints all lines which have "rarp" in them, regardless of case. Generally UNIX daemons have descriptive names; the RARP server is usually called "rarpd" for example. However, the TFTP server is often called "in.tftpd," so it is not safe to assume that daemon names are a protocol name followed by a "d."

If you find something interesting with the above "grep" command, such as a line that looks like

```
rarpd -a
```

remember that a leading pound sign ("#") comments out the line, and that the invocation may be inside of a conditional statement in the shell script. You will need to look at the file more closely to see what is going on in more detail. If you are pretty sure a daemon is being invoked, consult "Finding servers on your network" on page 2-14 for tips on how to verify that it is actually running.

inetd

Since UNIX systems are generally network oriented, they provide many network services. Many of these services are implemented using daemons that listen for connections to well-known ("advertised") TCP or UDP ports. A detailed discussion of TCP and UDP is beyond the scope of this document, but, in brief, "ports" are simply numbers that allow the TCP or UDP protocols to determine to which (of possibly many) protocol connections data should be delivered. Common UNIX daemons provide TELNET, rlogin, rsh, ftp, uucp, and other services, not to mention TFTP, BOOTP, and RARP.

To help reduce the number of daemon processes to a manageable level, most UNIX systems use a special daemon, the *inetd*. The *inetd* is a server of servers; that is, it listens for requests destined for some particular server, such as TFTP, and invokes the TFTP server to respond to the request. The *inetd* is configured by the *inetd.conf* file, usually found in /etc. A typical line in the *inetd.conf* file might be

```
tufty dram up wait root /usr/etc/in.tftpd in.tftpd /
tftpboot
```

This line configures *inetd* to monitor requests for the TFTP server. The first field is the symbolic name of the port to listen on; the second and third fields, "dgram" and "udp," tell *inetd* that the server uses UDP; the fourth field, "wait," has to do with *inetd*'s behavior when there are multiple simultaneous requests; the fifth field indicates that the TFTP server will execute with root (super-user) permissions; the sixth field is the location of the server program's executable code; and the remainder of the line is the command line used in the invocation of the TFTP server by *inetd*. Note that the invocation includes the name of the daemon. As far as "in.tftpd" is concerned, "/tftpboot" is its first parameter.

The mapping between the symbolic name "tftp" and the actual port number of the TFTP protocol is done using the services file, usually located in /etc. Note that services may be mapped using the NIS as well. Consult "System configuration files" on page 2-11 and "Finding servers on your network" on page 2-14 for information about the services file and about NIS.

As with most UNIX configuration files, a leading pound sign ("#") comments out the line. If your BOOTP or TFTP server is supposed to be started by *inetd*, but doesn't seem to be working, check for commented out lines in *inetd.conf*. For example,

```
# bootpd dgram udp wait root /etc/bootpd bootpd
```

does not start the BOOTP server, though it might look like it does.

If you edit the *inetd.conf* file, for example to remove a pound sign, make sure that you tell *inetd* to re-read its configuration file. This is done by sending it a SIGHUP signal. See "Daemon conventions" on page 2-10 for details.

As a final note on the *inetd*, note that it cannot be used to start the RARP server. This is because RARP does not use TCP or UDP as a transport protocol. If RARP is running, it is running as its own daemon on your system.

Daemon conventions

Since daemons are invoked at system boot time, they are not generally killed and restarted. However, if a change is made to a daemon configuration file, such as *inetd.conf* or bootptab, the daemon must be notified. This is usually done by sending the daemon a SIGHUP signal. This can be done using the UNIX kill command. For example, to reconfigure the *inetd*, you must first edit its configuration file. Then, determine the process number of the daemon, by executing

```
unixprompt> ps -aux | grep inetd
```

Then send it the signal, by executing

```
unixprompt> kill -HUP process-number
```

where "process-number" is the process number obtained with "ps." Note that you must be the super-user to send signals to daemons running with root permissions.

If a server is started by *inetd*, there is no need to send it a signal after editing its configuration file.

System configuration files

The UNIX system has a number of system configuration files. A few of these may need to be modified to make using NetROM more convenient. You must be super-user to modify the system files, and you should be aware that many of these files are silently ignored when the Network Information Service (also called Yellow Pages) is running on your system. See "Downloading images and emulation" on page 3-11 for more information on NIS.

/etc/hosts

The /etc/hosts file is used to maintain a mapping between the names of hosts on a network and their network address. The network address is the standard Internet dotted-decimal notation IP address. For example, the line

```
192.0.0.210        netrom1
```

allows networking programs to use the symbolic name "netrom1" to refer to the network node "192.0.0.210."

/etc/ethers

The /etc/ethers file is used to maintain a mapping between the Ethernet addresses of hosts on the network and their IP addresses. This file is used by the RARP server. For example, the line

```
00:02:f4:00:00:24    netrom1
```

indicates that the host "netrom1" has the Ethernet address "00:02:f4:00:00:24." Note that "netrom1" is a symbolic name; the mapping between symbolic names and IP addresses is made using the /etc/hosts file.

/etc/services

The /etc/services file establishes a relationship between the symbolic name for a service, its "port" number, and the protocol used to implement it. For example, the lines

```
telnet 23/tcp
tftp   69/udp
bootp 67/udp
```

indicate that "tftp" uses UDP port 69 and "bootps," the BOOTP server, uses UDP port 67. You may need to add these lines to the services file on your server for the *inetd.conf* lines given in "Daemons on UNIX systems" on page 2-7 to work.

Using NIS (Yellow Pages)

The Network Information Service, also called the Yellow Pages, is a distributed database used to automate updates to UNIX system configuration files. For a large network, updating /etc/hosts for every host, every time a new machine is added to the network, is a major chore. NIS allows a single machine to be the repository of the "real" files. This machine is called the master NIS server. There also may be several "slave" servers to reduce the load on the master.

A complete discussion of NIS is beyond the scope of this document. If you are using NIS in your development environment, consult your local documentation for information on how to add entries for your NetROM unit into the NIS database. This section provides only an overview of how NIS works and is not a "how-to" guide.

When a system is using the NIS as a client, it starts up a daemon process that determines the location of the nearest NIS server. This daemon, ypbind, acts as an interface to the remote server. When a program on the client machine, such as rarpd, wishes to access a system file, such as the hosts file, it checks to see if ypbind is running. If it is, the program uses ypbind to get the hosts file data. If ypbind is not running, it will read the local copy of the hosts file. Thus, if the NIS is being used in your system, you must be sure to update the master copies of the hosts and/or ethers files with NetROM's addresses. Changes to the local copies of these files will be quietly ignored.

You can determine whether NIS is running on your system with either of the two commands:

```
unixprompt> ps -aux | grep ypbind
unixprompt> ypmatch -x
```

The first command checks to see if the ypbind process is running, and the second makes contact with it and asks what its "map nickname table" is.

Finding servers on your network

Use the following commands to find which machines on your network, if any, are running servers that will interact with NetROM. These commands look for daemon processes running, check the *inetd.conf* file for BOOTP and TFTP servers, and check the "rc" script files for invocations of servers.

```
unixprompt> ps -aux | grep rarp
unixprompt> ps -aux | grep tftp
unixprompt> ps -aux | grep bootp
unixprompt> ps -aux | grep ypbind
unixprompt> grep rarp /etc/rc*
unixprompt> grep bootp /etc/rc* /etc/inetd.conf
unixprompt> grep tftp /etc/rc* /etc/inetd.conf
```

For convenience, you may want to collect these commands in a shell script.

If you find servers running on the system, you need only configure them for your NetROM unit. If they are not actually running, and you find invocations in the "rc" files or in *inetd.conf*, you need to make sure they are actually being invoked. Remember that if ypbind is running you need to update the master NIS database and do a "yppush" operation, rather than update local copies of UNIX system files.

If you are sure that servers are available on a particular machine, but they don't seem to be responding to NetROM, it is possible that they are not running. This may happen because the invocation of daemon servers in "rc" files or similar scripts are commented out or occur conditionally at system boot time. Servers started by *inetd* may have their line in *inetd.conf* commented out. You may have to do some detective work in the "rc" files or in *inetd.conf* to determine if either of these situations apply.

If your server is definitely running but still does not seem to be responding to NetROM, your server configuration files may not be set up properly. Consult Chapter 3 for additional help.

Chapter 3

NetROM Installation Troubleshooting

The following sections contain possible categories of problems you may encounter when installing your NetROM. Attempt to match your problem with one of the categories, then check the matrix for troubleshooting tips.

Serial communications

The following matrix lists possible problems you may encounter with serial communications. Check the symptom column, try to identify the problem, and attempt the solution.

Symptom	Problem	Things to try
No response on NetROM console.	Serial line parameters are misconfigured.	Check baud rate, parity and stop bits on console terminal against NetROM defaults given in the *NetROM User's Manual*, Appendix E.
	Serial line is connected to the wrong socket.	Make sure that the RJ-45 connector is plugged into the socket labeled "Console" on NetROM's back panel.
	RS-232 connection is DCE-to-DCE or DTE-to-DTE.	Insert a null modem between NetROM and the console terminal.
No response on target console.	See NetROM console problems above.	See "things to try" for NetROM console problems above. Note that serial line parameters can be changed with the stty command.

Address resolution—BOOTP

The following matrix lists possible problems you may encounter with BOOTP. Check the symptom column, try to identify the problem, and attempt the solution.

Symptom	Problem	Things to try
NetROM doesn't get a response from BOOTP server.	Server is not running.	If server is a daemon, make sure it is running. If started by *inetd*, make sure the "bootps" line in *inetd.conf* is not commented out.
	Server's *bootptab* file has not been configured for NetROM.	Make an entry in the *bootptab* file for your NetROM unit.
	Server does not know *bootptab* file has been modified.	Send the BOOTP daemon a SIGHUP signal using the *kill* command, *if* it is running as a daemon. Make sure it is not killed by the signal.
	Server's *bootptab* file is in the wrong format.	Make sure you have not created a Stanford-format entry for a CMU BOOTP server, or vice versa.
	Syntax error in the *bootptab* entry.	Make sure that your *bootptab* entry is exactly correct. Many BOOTP servers will silently ignore entries with syntax errors.
	The Ethernet address in the *bootptab* entry for NetROM is incorrect.	Check the address against the label on the bottom of the NetROM unit, or given by the *di lanceha* command on the NetROM console.

Symptom	Problem	Things to try
NetROM doesn't get a response from BOOTP server. (continued)	The *bootptab* entry for NetROM uses a symbolic name for the unit's IP address, but the name hasn't been entered into the *hosts* file.	Make sure that there is an entry in the local *hosts* file for the NetROM unit. If using NIS, check the master database.
	Your BOOTP server is running on a machine which uses NIS, but the NIS *hosts* database does not have an entry for the NetROM unit.	Make sure that the master NIS database has an entry for the NetROM unit. Run the *ypmatch* command on the BOOTP server's machine to verify that the change has propagated to all NIS servers.
	Startup batch file doesn't exist, or the path to it is wrong.	Make sure that the startup file specified by your *bootptab* entry exists in the directory specified by the entry. Remember that the path will be different if your TFTP server is running in "secure" mode.
NetROM configures itself with the wrong IP address and/or downloads the wrong startup batch file.	More than one BOOTP server is responding to NetROM's request, or a RARP server is responding to the request.	Make sure that all BOOTP servers in your environment are using the same configuration information. NetROM prints a message on its console indicating which BOOTP server responded to it. Make sure that no RARP servers are responding to NetROM. NetROM prints a message indicating which RARP server responded to it.

Symptom	Problem	Things to try
NetROM configures its IP address but doesn't execute commands in the startup batch file.	Startup batch file doesn't exist, or the path to it is wrong.	See above.
	A RARP server is responding to NetROM's address resolution request, and the startup file calculated from NetROM's IP address does not exist.	Either create a startup file named according to NetROM's IP address, or configure the RARP server not to respond to NetROM.
	Another machine on the network has the same IP address as the NetROM unit, and this is confusing the TFTP download.	Make sure that NetROM's IP address is unique. Turn the unit off and attempt to *ping* it. If a response comes back, the IP address is in use.

Address resolution—RARP

The following matrix lists possible problems you may encounter with RARP. Check the symptom column, try to identify the problem, and attempt the solution.

Symptom	Problem	Things to try
NetROM doesn't get a response from RARP server.	Server is not running.	Make sure that the server is running, using the *ps* command.
	Server's *ethers* file has not been configured for NetROM.	Make an entry in the *ethers* file for your NetROM unit.
	Server does not know *ethers* file has been modified.	Send the RARP daemon a SIGHUP signal using the *kill* command. Make sure it is not killed by the signal.
	The Ethernet address in the *ethers* entry for NetROM is incorrect.	Check the address against the label on the bottom of the NetROM unit, or given by the *di lanceha* command on the NetROM console.
	The *ethers* file entry for NetROM uses a symbolic name for the unit's IP address, but the name hasn't been entered into the *hosts* file.	Make sure that there is an entry in the local *hosts* file for the NetROM unit. If using NIS, check the master database.

Symptom	Problem	Things to try
NetROM doesn't get a response from RARP server. (continued)	NIS is running on the server's machine, but the master NIS *ethers* database does not have an entry for NetROM.	Make sure that the master NIS *ethers* database has an entry for the NetROM unit. Run the *ypmatch* command on the RARP server's machine to verify that the change has propagated to all NIS servers.
	Your RARP server is running on a machine which uses NIS, but the NIS *hosts* database does not have an entry for the NetROM unit.	Make sure that the master NIS *hosts* database has an entry for the NetROM unit. Run the *ypmatch* command on the RARP server's machine to verify that the change has propagated to all NIS servers.
NetROM configures its IP address but doesn't execute commands in the startup batch file.	Startup batch file doesn't exist, or the path to it is wrong.	Make sure that the startup file calculated from NetROM's IP address exists in the / *tftpboot* directory. If your TFTP server is in "secure" mode, the file must be in a *tftpboot* subdirectory of the server's home directory.
NetROM configures itself with the wrong IP address and/or downloads the wrong startup batch file.	More than one RARP server is responding to NetROM's request, or a BOOTP server is responding to the request.	Make sure that all RARP servers in your environment are using the same configuration information. NetROM prints a message on its console indicating which RARP server responded to it. Make sure that no BOOTP servers are responding to NetROM. NetROM prints a message indicating which BOOTP server responded to it.

Symptom	Problem	Things to try
NetROM configures its IP address but doesn't execute commands in the startup batch file.	A BOOTP server is responding to NetROM's address resolution request, and the startup file named in the response does not exist.	Either create a startup file named according to the *bootptab* entry, or configure the BOOTP server not to respond to NetROM.
	Another machine on the network has the same IP address as the NetROM unit, and this is confusing the TFTP download.	Make sure that NetROM's IP address is unique. Turn the unit off and attempt to *ping* it. If a response comes back, the IP address is in use.

Startup batch file

The following matrix lists possible problems you may encounter with the startup batch file. Check the symptom column, try to identify the problem, and attempt the solution.

Symptom	Problem	Things to try
NetROM doesn't execute commands in the startup batch file	The batch file is not being executed at address resolution time.	See the above sections on troubleshooting BOOTP and RARP.
NetROM doesn't execute commands in the startup batch file. (continued)	The batch file is not delimited by *begin* and *end* statements.	Put *begin* and *end* statements, on lines by themselves, at the beginning and end of the batch file.
	The batch file contains non-printable characters.	Make sure that the file has only printable characters. Files created with the *vi* editor will not have hidden formatting characters.
	There are syntax errors in the commands.	Errors during execution of the startup batch file are displayed on NetROM's serial console. If you do not have a terminal on the NetROM console, execute the batch file manually from a TELNET console and observe the status of the commands as they execute. Use the *batch* command to execute the commands.
	The file is too large.	NetROM batch files cannot be larger than 2048 bytes. Make sure that your file is smaller than this.

Symptom	Problem	Things to try
NetROM displays a message about TFTP errors such as "Access Violation" or "File Not Found"	The file does not reside at the named directory.	Make sure that the file exists at the named directory, and that the directory path leading to the file exists.
	The file exists at the named directory, but the TFTP server is attempting to control access to the file.	Make sure that the file has global read permissions. Consult documentation on your local TFTP server.

Downloading images and emulation

The matrix on the following page lists possible problems you may encounter with the downloading images and emulation. Check the symptom column, try to identify the problem, and attempt the solution.

Symptom	Problem	Things to try
NetROM doesn't execute commands in the startup batch file	The batch file is not being executed at address resolution time.	See the sections on troubleshooting BOOTP and RARP.
	The batch file is not delimited by *begin* and *end* statements.	Put *begin* and *end* statements, on lines by themselves, at the beginning and end of the batch file.
	The batch file contains non-printable characters.	Make sure that the file has only printable characters. Files created with the *vi* editor will not have hidden formatting characters.
	There are syntax errors in the commands.	Errors during execution of the startup batch file are displayed on NetROM's serial console. If you do not have a terminal on the NetROM console, execute the batch file manually from a TELNET console and observe the status of the commands as they execute. Use the *batch* command to execute the commands.
	The file is too large.	NetROM batch files cannot be larger than 2048 bytes. Make sure that your file is smaller than this.

Symptom	Problem	Things to try
NetROM displays a message about TFTP errors such as "Access Violation" or "File Not Found"	The file does not reside at the named directory.	Make sure that the file exists at the named directory, and that the directory path leading to the file exists.
	The file exists at the named directory, but the TFTP server is attempting to control access to the file.	Make sure that the file has global read permissions. Consult documentation on your local TFTP server.

Miscellaneous problems

The following matrix lists other possible problems you may encounter. Check the symptom column, try to identify the problem, and attempt the solution.

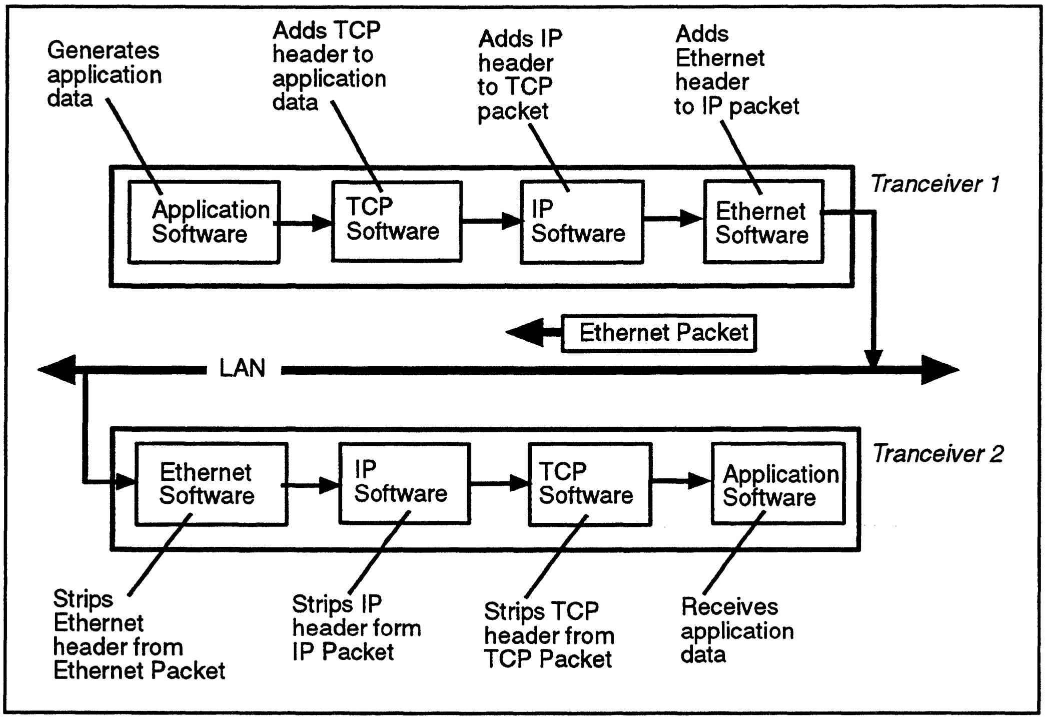
Symptom	Problem	Things to try
NetROM responds to network connections from some hosts but not from others	The hosts it does not respond to are on different "subnets" from the ones to which it does respond.	Add a default route to NetROM's routing table, using the *route* command.
You can't get out of your TELNET session with NetROM.	Your *telnet* program is waiting for you to tell it to exit.	Type the TELNET escape character (often (^]'), then type "quit" on the prompt that appears. Consult local documentation on TELNET for more information.
Even after you change it with the *stty* command, your target console exits when it sees the *old* "eof" character.	The "eof" character is interpreted by the *conspathd* process, whose "eof" character was not changed by the *stty* command.	Set the *default* "eof" character in your startup.bat file using *stty* -d, and reset your NetROM unit.

Chapter 4
Network Basics

This appendix provides a simplified description of network operation. "TCP/IP network protocol" on page 4-1 outlines Transmission Control Protocol/Internet Protocol (TCP/IP), which assembles message packets for network transmission. "Addressing" on page 4-4 explains network and subnet addressing.

TCP/IP network protocol

On Ethernet, Transmission Control Protocol / Internet Protocol (TCP/IP) software allows communication between different networks. A basic system consists of two transceivers connected through a network; see Figure 4-1. From the transceiver 1, the application software generates the application data. Next, the TCP software adds the TCP header to the application data forming the TCP packet. Next, the IP software forms the IP packet by adding the IP header to the TCP packet. Finally, Ethernet software creates the Ethernet packet by adding an Ethernet header to the IP packet. The transceiver sends the Ethernet packet across the network to transceiver 2 that receives the packet and successively strips the headers leaving the application data.



Generates application data
Adds TCP header to application data
Adds IP header to TCP packet
Adds Ethernet header to IP packet
Application Software
TCP Software
IP Software
Ethernet Software
Tranceiver 1
Ethernet Packet
LAN
Ethernet Software
IP Software
TCP Software
Application Software
Tranceiver 2
Strips Ethernet header from Ethernet Packet
Strips IP header form IP Packet
Strips TCP header from TCP Packet
Receives application data

Figure 4-1 TCP/IP network protocol

Figure 4-2 illustrates the Ethernet packet, which consists of a series of embedded data portions with headers. The TCP packet is the data portion of the IP packet, and the IP packet becomes the data portion of the Ethernet packet. Because of this structure, the system can transmit the data within the packet to other networks regardless of the packet type carrying it.

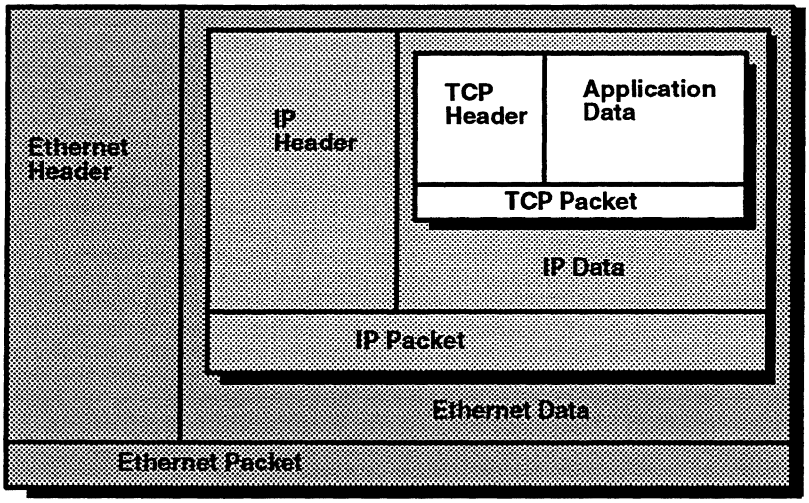


Figure 4-2 Ethernet message packet

Ethernet packets

The transceiver routes data across the Ethernet in variable-length “frame” or “packet” format (see Figure 4-3). Ethernet packets are self identifying with each packet containing the source address, the destination address, and the information type. Including packet type information in the structure allows the use of multiple protocols on a single machine or on the same physical network without interference.

Preamble	Destination Address	Source Address	Packet Type	Data	CRC
64 bits	48 bits	48 bits	16 bits	368-12,000 bits	32 bits

Figure 4-3 Ethernet packet format

Besides the destination and source addresses, Ethernet packets contain this additional information.

Preamble	Consists of alternating 0s and 1s to aid the receiving transceiver in signal synchronization. The last two digits are 11.
Packet Type	Informs the receiving transceiver protocol software module used to process the packet. In our case this is Internet Protocol.
CRC	Used to identify data errors within the packet.

Addressing

A host is any end-user computer connected to the network. Each host under TCP/IP has a physical (Ethernet) address and an Internet (IP) address.

Physical/Ethernet addresses

Each host's Ethernet interface contains the host unit's physical or Ethernet address stored in a ROM or a PAL. The physical address allows the computer to determine the packets the computer will receive. The physical address is a six-byte, hexadecimal number. Ethernet hardware manufacturers purchase blocks of physical address and assign the address in sequence as they manufacture the interface hardware. Physical addresses can not be changed.

Note

The physical address relates to the interface hardware rather than the host. Replacing the host's interface hardware changes the host's physical address.

Internet addresses

An Internet (IP) address is a 4-Byte number written in dotted decimal notation (XX.X.XX.XXX). This notation expresses each byte as a decimal integer between 0 and 255 with decimal points separating the bytes.

For example: 10.4.25.196

The Internet address differs from the physical address because users can assign and program the Internet address.

Caution

You must insure the Internet address is not duplicated anywhere on the network, otherwise duplicate Internet addresses can seriously interfere with network operation.

The Internet address consists of two portions: a Network ID portion and a Host ID portion. The Network ID identifies a given physical network, and the Host ID identifies a particular host on that network.

There are five defined Internet address classes. Each class is divided based on the number of networks and the number of hosts. Only three of the classes (A, B, and C) can specify individual hosts (Figure 4-4).

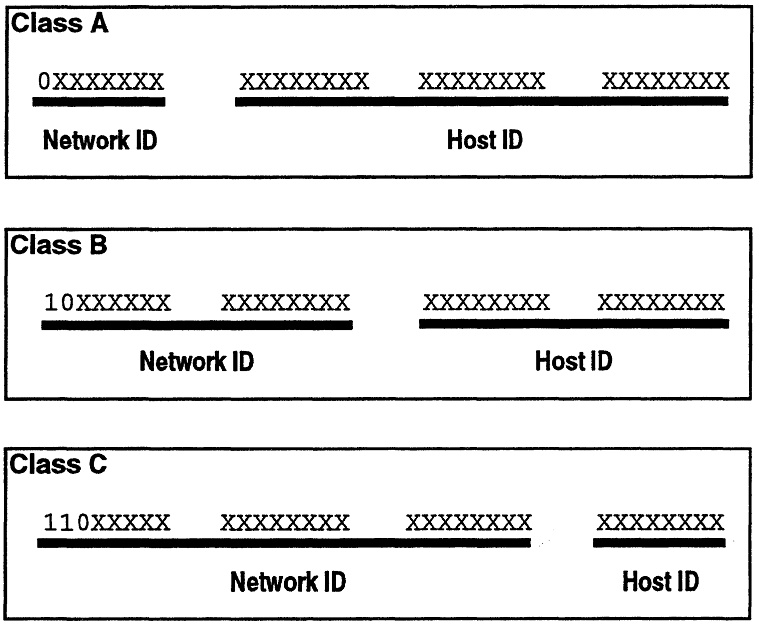

Class A
0XXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX
Network ID
Host ID
Class B
10XXXXXX XXXXXXXX XXXXXXXX XXXXXXXX
Network ID
Host ID
Class C
110XXXXX XXXXXXXX XXXXXXXX XXXXXXXX
Network ID
Host ID

Figure 4-4 Internet address classes

Class A

Networks having more than 65,536 (2^{16}) hosts use class A addresses. The first eight bits are the network ID, and 24 bits are the host ID. The high-order bit of the first byte (network ID) must always be 0, allowing network addresses from 0 to 127.

Note

Network ID 127 should not be used because it is reserved for the local and loopback device.

Class B

Networks have between 256 (2^8) and 65,536 (2^{16}) hosts use class B addresses. The first 16 bits of the address are the network ID, and the second 16 bits are the host ID. The high-order bits of the first byte (network ID) must be 10 allowing network addresses between 128.0 and 191.255.

Class C

Networks with fewer than 256 (2^8) hosts use class C addressing. The first 24 bits of the address are the network ID, and the remaining 8 bits are the host ID. The high-order bits of the first byte (network ID) must be 110 allowing network addresses from 192.0.0 to 223.255.255.

Network addresses

Convention regards the address of the network itself to have a host ID of 0. Therefore, the user should never assign others host a host ID of 0. A class B network address would have the form:

x.y.0.0

For example: 192.12.0.0.

Broadcast address

The Ethernet standard provides for a Broadcast Address that refers to all hosts on the network. Broadcast addresses allows a copy of a message packet to be sent to all hosts on the system. An Internet address having a host ID of all bits set to 1 (255) is the broadcast address. A class B broadcast address would have the form:

x.y.255.255

For example: 192.12.255.255.

Subnets

As the number of physical networks in a given class grows, the pool could run out of addresses. To reduce the number of addresses needed in a large system Subnet Addressing and Subnet Routing was developed. This allows a group of physical networks (subnets) to share a single network address.

To do this the Internet address is replaced with an Internet component and a local component. The Internet component is the network address shared by the various local networks.

The local component consists of two parts. One part identifies a particular subnet, and the second part identifies a host on that subnet. The subnet address is concealed in the host ID portion of the Internet address (Figure 4-5).

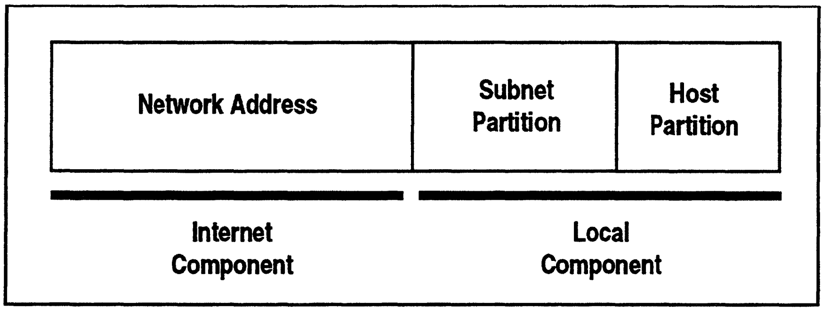


Figure 4-5 Internet address with subnet addressing

Figure 4-6 illustrates how subnet addressing works with a class B network. In this system, the third byte of each Internet address specifies the subnet (for example: 192.<u>16</u>.1.2), and the fourth byte identifies the host (for example: 192.16.<u>1.2</u>). To the rest of the Internet, the subnets and their hosts appear to be a collection of hosts on network 192.16.0.0. The local gateway accepts all packets with this network address and routes them to the appropriate subnet by examining the third byte of the address. The fourth byte of the address identifies the host on the given subnet.

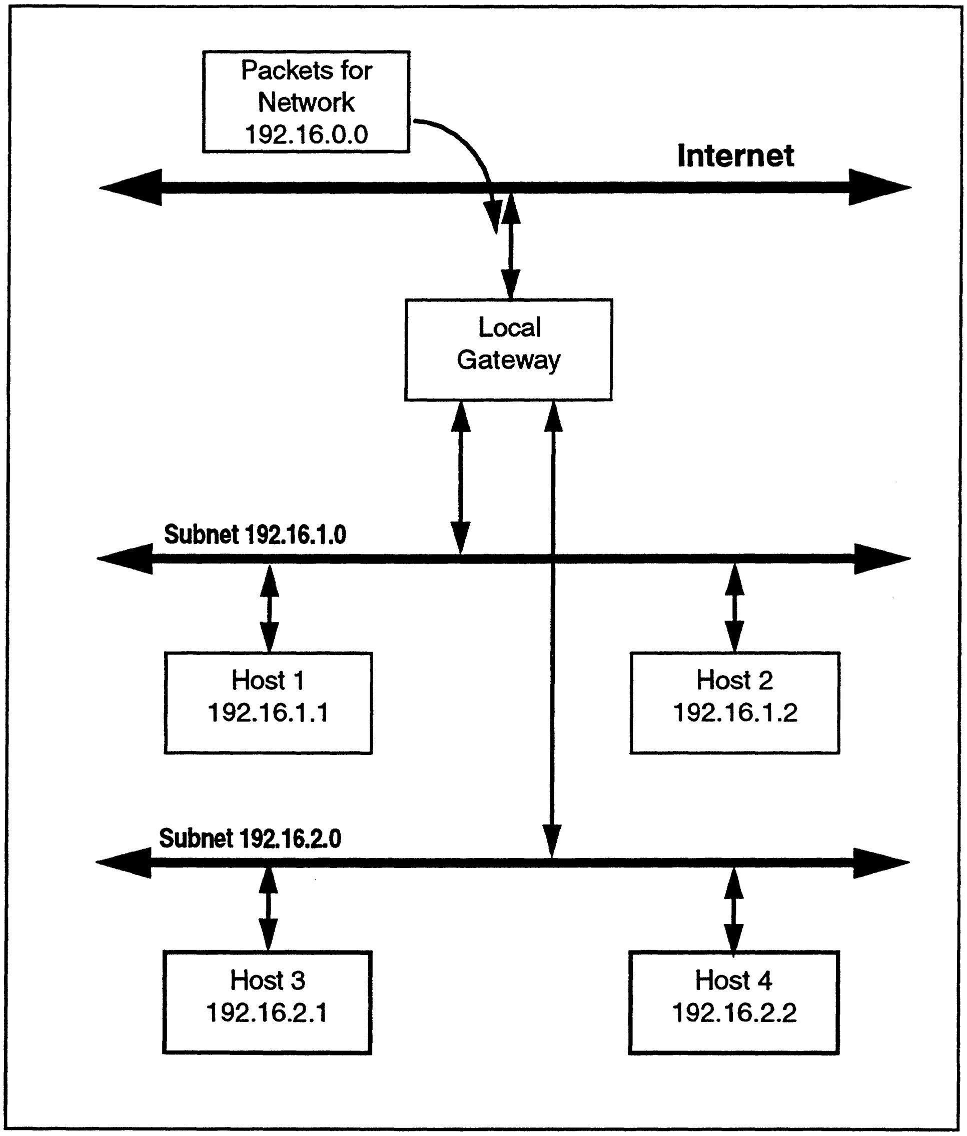


Figure 4-6 Subnet addressing

The system shown in Figure 4-6 uses the first 8 bits of the 16 bit local component to identify the subnet (for example: 192.16.1.2), and the remaining 8 bits identify the host (for example: 192.16.1.2). This system allows up to 256 subnets with 256 hosts per subnet.

This is not the only method of local component partitioning. Each network administrator can partition the local component in any manner that is usually based on expected growth of the system. For example, if a network administrator determines that a class B site requires many subnets with a few hosts on each subnet, they can allocate 12 bits to the subnet partition and 4 bits for the host partition. This example allows 4096 (2^{12}) subnets with up to 16 (2^{4}) hosts per subnet (including network and broadcast addresses).

Subnet masks

The Internet standard requires every site using subnet addressing to define a 32-bit subnet mask for each physical network at the site. A network's subnet mask defines how. Bits in the subset mask are set to 1s if the corresponding bits (including the Internet component) in the Internet address are considered to be part of the network address. The bits are set 0s if the corresponding bits in the Internet addresses are treated as part of the host identifier.

If a network administrator of a class B network wants to allocate 12 bits of the local component to the subnet partition and 4 bits to the host partition, the required subnet mask would be shown in Figure 4-7.

Internet Component				Subnet Partition			Host Partition
1111	1111	1111	1111	1111	1111	1111	0000
F	F	F	F	F	F	F	0

Figure 4-7 Subnet mask for a sample class B network

Since the Internet component and the subnet partition together identify a particular physical network, they comprise the "network address," and all bits of these two fields are set to 1s in the subnet mask. The host partition identifies a particular host on a given network; all bits of this field are set to 0s in the mask. The subnet mask for Figure 4-7 would be "255.255.255.240."

Each host on a network has a copy of the subnet mask for that network. By using the subnet mask, the host can extract the network address from an Internet address; this helps make packet-routing algorithms efficient.

Routers

Routers connect two or more networks and transfer message packets among those networks. The router uses software algorithms and routing tables to make the routing decisions necessary to ensure each packet reaches its destination. Often, the terms router and gateway will be used interchangeably.

Router tables

Both hosts and routers use routing tables to determine where to send message packets. Router tables do not contain complete information on how to reach each destination address. Instead, the tables tell the host or router where to send the packet to reach the next step along the path to the packet's destination. Essentially, the routing table contains pairs of addresses (D and R). "D" represents the destination's network address, and "R" represents the Internet address. These addresses may point to a router or directly to a host connected to the same subnet.

When a host or router needs to forward a message packet, it extracts the destination's network address by performing a logical AND operation between the network address and the local subnet mask. Next, the host or router searches its routing table for the matching destination network address (D). If a match is found, the host or router sends the message packet to the corresponding destination (R). In turn, the host or router at the destination performs the same process and routes the message packet to the next host or router. If the destination lies on the same subnet as the router or host, the router delivers the packet directly to the destination.

For example, host 4 needs to send a message packet to host 7 (see Figure 4-8). Since all the networks are class B, the subnet mask for each network is 255.255.0.0 (binary: 11111111 11111111 00000000 00000000). Also, host 4 contains the routing table shown in Table 4-1.

Table 4-1 Example host 4 routing table

Destination Network (D)	Routing Address (R)
192.7.0.0	192.12.0.4
192.12.0.0	deliver directly
192.16.0.0	192.12.0.4
192.41.0.0	192.12.0.4

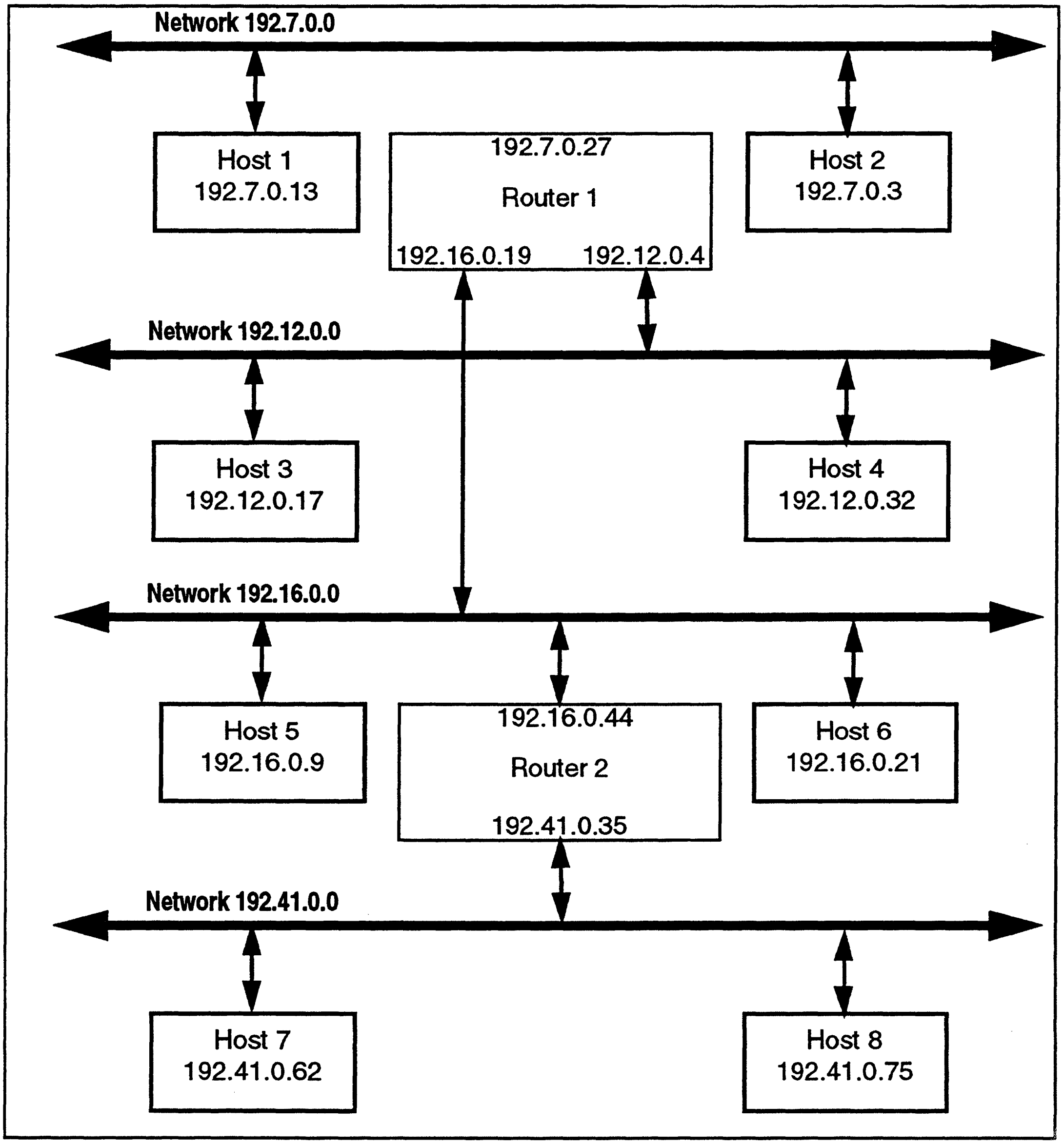


Figure 4-8 Example network with routers

Host 4 begins constructing the message packet and addressing it to host 7 (Internet address: 192.41.0.62). Host 4 extracts the network address (192.41.0.0) by ANDing the host 7 Internet address with the subnet mask.

Subnet Mask	FF FF 00 00
Internet Address =	CO 29 00 3E
Destination Address	CO 29 00 00

This address matches a destination network address in the routing table (Table 4-1) with a corresponding routing address of 192.12.0.4. Host 4 sends the message packet to 192.12.0.4 (router 1).

Router 1 receives the packet and extracts the destination network address (192.41.0.0). Router 1 searches its router table (Table 4-2) for the destination address and its corresponding routing address which tells router 1 to send the packet to 192.16.0.44 (router 2).

Table 4-2 Router 1 router table

Destination Address	Routing Address
192.7.0.0	deliver directly
192.12.0.0	deliver directly
192.16.0.0	deliver directly
192.41.0.0	192.16.0.44

Router 2 receives the packet and extracts the destination network address (192.41.0.0). Router 2 searches its router table (Table 4-3) for the destination address that tells router 2 to send the packet directly to host 7 at address 192.41.0.62.

Table 4-3 Router 2 router table

Destination Address	Routing Address
192.7.0.0	192.16.0.19
192.12.0.0	192.16.0.19
192.16.0.0	deliver directly
192.41.0.0	deliver directly

In summary, the router receives a packet, extracts the destination network address, finds its destination address in its router table. The routing table tells the router to send the packet directly to a host or to another router. In the example, router 1 was unable to send the packet directly to the host, therefore, router 1 sent the packet to router 2. Router 2 could send the packet directly to the destination host 7 (see Figure 4-9).

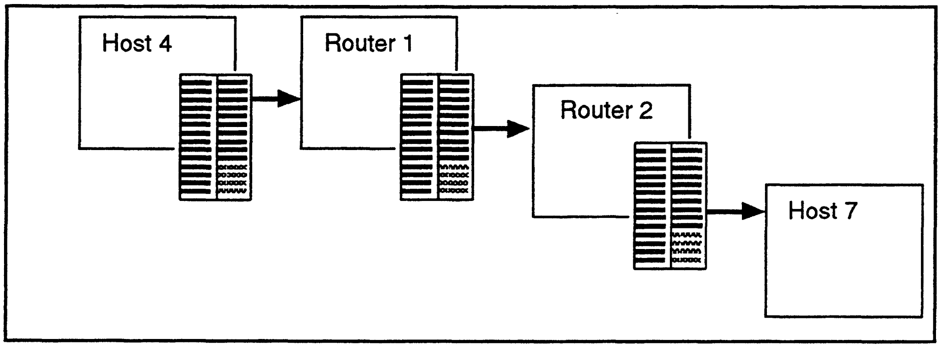


Figure 4-9 Example message packet flow

Index

Applied
Microsystems
Corporation

Applied Microsystems Corporation maintains a worldwide network of direct offices committed to quality service and support. For information on products, pricing, or delivery, please call the nearest office listed below. In the United States, for the number of the nearest local office, call 1-800-426-3925.

CORPORATE OFFICE
Applied Microsystems Corporation
5020 148th Avenue Northeast
P.O. Box 97002
Redmond, WA 98073-9702
(425) 882-2000
1-800-426-3925
Customer Support
1-800-ASK-4AMC (1-800-275-4262)
TRT TELEX 185196
FAX (425) 883-3049
Internet Home Page: http://www.amc.com

UNITED KINGDOM
Applied Microsystems Corporation Ltd.
AMC House
South Street
Wendover
Buckinghamshire
HP22 6EF United Kingdom
(0) 1296-625462
Telex 265871 REF WOT 004
FAX 44 (0) 1296-623460

JAPAN
Applied Microsystems Japan, Ltd.
Arco Tower 13 F
1-8-1 Shimomeguro
Meguro-ku
Tokyo 153, Japan
03-3493-0770
FAX 03-3493-7270

GERMANY
Applied Microsystems Gmbh
Stanhlgruberring 11a
81829 Muenchen
Germany
(089) 427403-0
FAX (089) 427403-33

FRANCE
Applied Microsystems Sarl
ZA1 de Courtaboeuf
7, Avenue des Andes
F-91952 Les Ulis Cedex
France
01-64-46-30-00
FAX 01-64-46-07-60

Part No.	Revision History	Date
924-07001-00	Release NetROM 500 series.	12/96
924-07001-01	Maintenance release of NetROM.	5/97

P/N 924-07001-01
May 1997