Freescale Semiconductor
Application Note

AN2748
Rev. 0, 08/2004

Genesi Pegasos II Kernel and NFS Facility

by *Maurie Ommerman and Jacob Pan*
CPD Applications
Freescale Semiconductor, Inc.

This is the eighth application note in a series that describes the Genesi Pegasos II system and its various associated applications. This document describes how to build a network file system (NFS), download kernels to a Sandpoint or ADS board target, and use NFS on Genesi as a root file system for a target.

Contents

1 Introduction

Using NFS gives great flexibility to the developer for controlling, debugging, and analyzing the target system from the host. This application note describes how to build an NFS for a Linux target on a Linux host system, how to download kernels from a host to a target, and how to start the target using the host NFS system.

2 Terminology

The following terms are used in this document.

Linux OS	Linux Operating System
Debian	One of the distributions of Linux
Yellow Dog	One of the distributions of Linux
NFS	Network file system, a complete root file system for Linux, that is self-contained on a host file system. Used to boot diskless Linux systems remotely.

freescale™ semiconductor

Host	Computer used for compiling and linking target programs
Target	Computer that is the source for programs compiled and linked on a Host.
Daemon	Small program that is always running in the background, usually waiting for activity. For example, a network daemon is waiting for activity on the network.
TFTP	Trivial file transfer protocol, a method of sending files across the ethernet interface.
ssh	Secure shell Linux command for connecting to other Linux systems using an encrypted transfer mechanism.

3 General Theory

General theory behind NFS on a native PowerPC™ system.

3.1 Tools

It is not necessary to have a host and target system that are the same architecture. Using cross-tools, any architecture can be used to develop PowerPC tools. The disadvantages to such an arrangement are never serious, but in general, inconvenient.

- Programs compiled natively on the host can not run on the target
- Programs cross compiled on the host can not run on the host.
- Cross tools are usually more difficult to obtain and use.
- Cross compiled programs may need to be downloaded to the target for testing, a sometimes arduous task. However, if an NFS mount is active, then the programs can just be placed on the hard drive.

A host that is a native PowerPC system, on the other hand, uses the same architecture as the target. Thus a native tool set can be used. These are the disadvantages.

- none

This leads to a whole host (pardon the pun) of advantages.

- Programs compiled natively on the host can run on the target.
- Programs need never be cross compiled.
- Native tools are easy to obtain and are usually supplied as part of the system package, for example GCC on any Linux system.
- Programs can be tested on the host system before the need to download them for final testing.
- The host and target both have the same touch and feel, i.e. both are the same architecture, and both can run the same Linux OS.
- The number of times downloading to the target is reduced.

3.2 Network File System (NFS)

There are four methods for supplying a root file system for a Linux target. (And possibly more)

- Local hard drive
  - — Subject to corruption by a renegade kernel
  - — Difficult to build or recover
  - — Requires an IDE or SCSI driver

- RAM disk
  - Very small disk space
  - Does not remember changes to the disk space at reboot time. That is, it always boots up with the same file system, changes in previous runs are lost.
  - Difficult to build, compress, and pack into a kernel.
- NFS
  - Requires a working ethernet connection
  - Requires a host machine
  - Protection from hard disk corruption
  - Potentially, allows for large disk space
- Flash based cramfs
  - Does not need to be loaded, always ready as soon as the kernel boots
  - Changes are permanent

NFS has the following advantages

- The target can run disk less, it only needs an ethernet connection
- The root file system is only a piece of the host file system, therefore, if the target file system is corrupted, it can be rebuild, or just recopied on the host.
- Targets can share an NFS root file system, or each target can have its own.
- Since the NFS is on the host, the same program can be run either from the host or from the target.
- The host can be a full size Linux, while the target can be a reduced version of Linux. For example, the target needs a reduced file system for an embedded target.

3.3 Benefits for Genesi Pegasos II PowerPC Native Host

So what are the benefits of a native host supplying a native NFS to a target, all using the same hardware processor architecture? In particular, the Genesi Pegasos II native PowerPC host development system with any PowerPC target, such as the Sandpoint or the ADS PowerQUICC™ boards, has many benefits.

- The host can be a fully featured Linux with multiple disk less PowerPC targets.
  - MPC74xx
  - PowerQUICC II, III
- The host supplies a complete native environment.
  - native PowerPC tool chain, for example gcc, as, ls, binutils, gdb and so forth.
  - Both host and target have the same JTAG/COP support.
  - No need for cross architecture compiler, libraries, or other tools.
- Host support
  - Coding, analyzing and verifying can be done on the host before downloading to the target.
  - Host tools supplied by CPD.
  - Performance monitoring, PMON facility
    - AN2743, *Software Analysis on Genesi Pegasos II Using PMON and AltiVec*
    - AN2744, *PMON Module—An Example of Writing Kernel Module Code for Debian 2.6 on Genesi Pegasos II*

  - sim_G4plus simulation
    - AN2749, Genesi Pegasos II Using sim_G4plus
    - AN2750, Genesi Pegasos II Analysis and Optimization of Code with sim_G4plus
- Suitable for kernel or device driver development.
  - Allows for remote access to the target
  - crash proof and easy recovery compared to a local hard drive on the target
  - More disk space available compared to a RAM disk.

3.4 Development Environment

Figure 1 diagrams the configuration discussed in this paper.

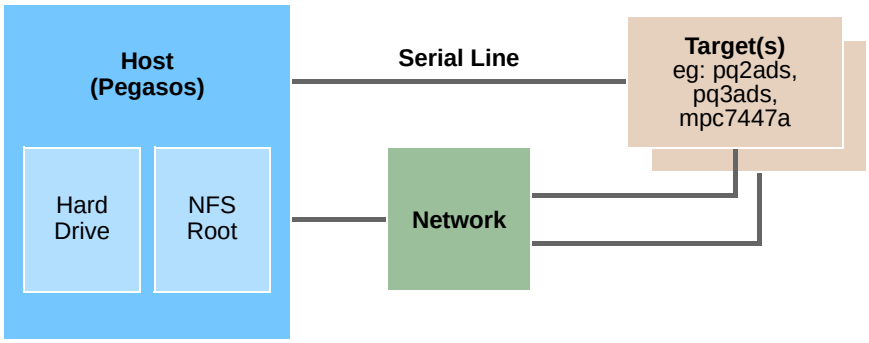

Figure 1. Host to Target Configuration

The host system is a Genesi Pegasos II with the following configuration, obtained with the `cat /proc/cpuinfo` and `uname -a` command.

- CPU : 7457, altivec supported
- clock : 999MHz
- revision : 1.1 (pvr 8002 0101)
- bogomips : 997.37
- machine : CHRP Pegasos2
- Linux debian 2.6.4-pegasos #1 Mon Mar 22 12:47:08 CET 2004 ppc GNU/Linux

The hard drive is a 40 Gig drive, the host Debian Linux system is using a single partition (/dev/hda5) of approximately 50% of the drive, or about 21 Gigs, as can be seen from this display of parted /dev/hda

```
root@debian:parted /dev/hda

Using /dev/hda

(parted) p

Pralloc = 0, Reserved = 2, blocksize = 1, root block at 114660

Disk geometry for /dev/hda: 0.000-38166.679 megabytes

Disk label type: amiga
```

```
Minor    Start       End     Filesystem  Name                  Flags
1          3.999    107.973  affs1       boot
2        107.974    611.850  asfs        MOS                   boot
3        611.851   3615.117  asfs        MOS-DATA
4       3615.117   4618.872  linux-swap  swap                  hidden
5       4618.872  25621.743  ext3        debian                hidden
6      25621.743  38166.679  ext3        YDL                   hidden
(parted)
```

Currently, there is about 69% space left on this partition as can be seen from the `df -k` command.

```
guest@debian:~$ df -k
Filesystem           1K-blocks      Used Available Use% Mounted on
/dev/hda5             21170868   6509324  14661544  31% /
```

The NFS virtual partition used for the target is the small piece of the disk shown in the figure, it resides at /opt, it will be further discussed in Section 4, “Setting up an NFS Root File System on a Host.”

Several targets can access each of these NFS root file systems via the ethernet. More than one target can share an NFS root file system, or each target can have only one. For this paper, each target will have only one NFS root file system.

The host can access each target via a serial connection, or via ethernet, once the target is started. In this figure, we show two targets, each can be either an ADS or a Sandpoint board, each is connected to its unique NFS root file system via a network, and the host can communicate with either target.

Figure 2 shows that the host can edit files on the host, such as kernel code and build new kernels or application executable files. The host can communicate with either target via a serial line using a terminal emulator, like minicom, or through an ethernet network via telnet or ssh.

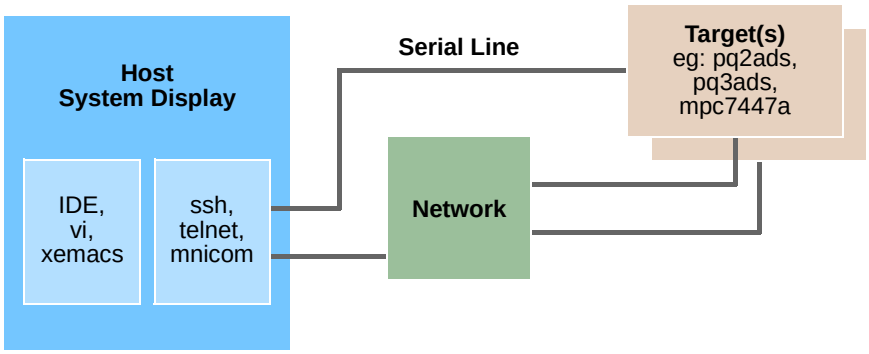

Figure 2. Typical Usage

Some typical usage is

- Beginning a session
  - Login to host system
  - Boot Linux on target
- Development, repeat as needed
  - Host: edit source
  - Host: test/verify executables on PowerPC host, which is a stable distribution.
  - Target: Load binaries via NFS from host
  - Target: test binaries on target, which may be unstable
  - Target: debug tools on host hard drive
- Logoff and go home
  - Remote access from home if desired

3.5 Concept of Host and Target

The host, Genesi Pegasos II has the following characteristics:

- Desktop fully featured system running Linux
- Large hard disk
- Complete tool chain
  - Compilers
  - JTAG software
- Graphic console
- Serial and network capabilities

and the following requirements

- Debian Linux distribution pre-installed packages
  - GNU tool chain

  - TFTP server
  - NFS Portmap server
  - DHCP server
  - Minicom, a serial console emulator
  - Target root file system
- Hardware requirements
  - Large hard drive (40GB+)
  - Lots of RAM (256MB+)
  - Fast processor, e.g. MPC744x at 1GHz+
  - Fast or GigE network interface

The target, on the other hand, has the following characteristics

- Evaluation/engineering sample boards
- Little initial software
- Serial and/or network console

and the following requirements

- Firmware/BootROM (such as DINK32 or U-boot)
  - Download capability: TFTP client
  - Serial console
  - Execute a user program
  - Pass platform information to the OS
- Hardware
  - Ethernet interface
  - Serial port
  - RAM
  - No hard drive

Figure 3., “Putting It All Together”, shows how all the parts interact.

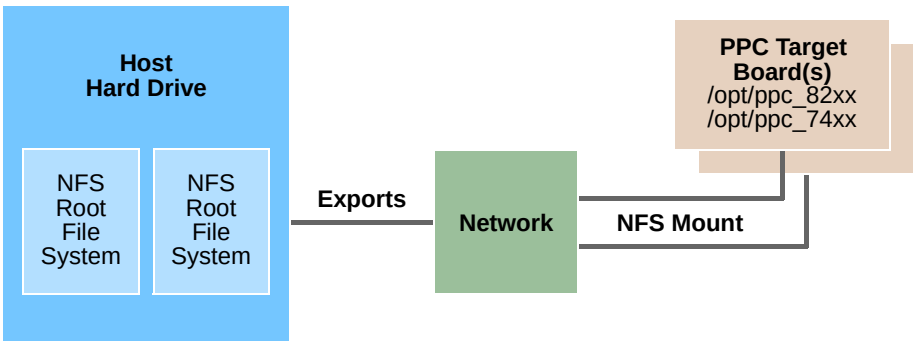

Figure 3. Putting It All Together

The target will mount the NFS root file system with an NFS mount command. The host will export the NFS root file systems to the target boards, each board has its own NFS root file system, shown here we have two targets, and two

NFS root file systems, /opt/ppc_82xx and /opt/ppc_74xx. The NFS root files systems are directories on the host hard drive, which we can visualize as a piece of the hard drive.

In summary:

- The host is exporting a small part of its hard drive, the NFS root file systems.
- The target is mounting this corresponding small part of the host hard drive via an NFS mount.
- The host has access to both NFS root file systems.
- Each target can only access its NFS root file system.
- All this activity is implemented via an ethernet network.
  - The targets are isolated from each other

4 Setting up an NFS Root File System on a Host

This section discusses setting up an NFS root file system and turning on the remote access facility. Since we are using the NFS facility that specifies the fixed IP address for all remote access, i.e. the targets, we must have a fixed IP address for all machines on this subnetwork.Therefore, it is not a DHCP/BOOTP server, it can exist on any network as long as the fixed IP addresses do not conflict with any previously assigned IP addresses or IP addresses in the range of the DHCP server on this network.

For this method of NFS, each target will query only the NFS server, by its known IP address, and each target will tell the NFS server what its IP address is. Therefore, the NFS server will only NFS mount to targets that are defined in its /etc./exports file.

The assumed IP addresses for this exercise are:

Host, which is the server: 10.82.117.52

Target 82xx: 10.82.0.105

Target 74xx: 10.82.118.204

Section 3, “General Theory” described how all this works in theory. This section will describe how to implement such a system.

4.1 NFS Root File Systems

The directory, /opt, is used to contain two NFS root file systems.

```
root@debian:/opt# ls /opt

ppc_74xx  ppc_82xx  ppc_82xx-rfs-fae.tgz  ppc_82xx-rfs.tgz
```

There are two subdirectories and two compressed tar files. The ppc_74xx and ppc_82xx root files systems are identical, however, to ensure that each target has its own NFS root system, ppc_74xx is the NFS root file system for our Sandpoint target using an MPC7447A or MPC7457 processor. ppc_82xx is the NFS root file system for our ADS PQIII board using an MPC82xx processor. As can be seen from listing these directories.

```
root@debian:/opt# ls ppc_74xx

bin  dhry.res  images  lib  proc  root  tmp  var

dev  etc       jacob   mnt  proj  sbin  usr

root@debian:/opt# ls ppc_82xx
```

```
bin  dhry.res  images  lib  proc  root  tmp  var

dev  etc       jacob   mnt  proj  sbin  usr

root@debian:/opt#
```

Uncompressing the tgz, compressed tar files, generates the ppc_82xx directory. There is only a minor difference between the two tar files. These root file system directories are minimal root file systems developed for an embedded target, that is, they have only a minimal set of commands, so that they are small. These embedded root file systems were downloaded from the ELDK, web site, http://www.denx.de/ELDK/, developed by Wolfgang Denk.

4.2 Host Mechanisms

The host supports these NFS root file systems as part of its root file system. They are just directories on the host. Linux security prevents any network access to these directories and in fact to any directories on the hard drive. Hence, the host must specify which, if any, directories can be accessed by remote users, in this case the targets. This is accomplished with the export file in the /etc directory.

4.2.1 The /etc/exports file

This file indicates to the network daemon which directories can be exported, that is, used remotely, and what the permission set is, that is, what type of access is allowed. Look at an example of this file with the `cat -n /etc/exports` command. The line numbers are generated by the `cat -n` command and are not part of the file.

```
root@debian:/# cat -n /etc/exports

     1  # /etc/exports: the access control list for filesystems which may be exported

     2  #               to NFS clients.  See exports(5).

     3  /home/pegasos 10.82.117.93(rw,sync)

     4  /opt/ppc_82xx 10.82.124.205/255.255.252.0(rw,no_root_squash)

     5  /opt/ppc_74xx 10.82.116.254/255.255.252.0(rw,no_root_squash)

root@debian:/#
```

Lines 1 and 2 are just a comment.

Lines 3, 4, and 5 are the lines defining remote access. The syntax for each line is

```
root directory, target IP address/target netmask permissions
```

Line 3 is incorrect and access to this mount point, /home/pegasos, will always be denied because there is no target netmask specified. The network daemon can not resolve the access on this line.

Line 4 and 5, however, are correct and will allow network access to these two directories, /opt/ppc_82xx, and /opt/ppc_74xx.

Dissecting line 4.

/opt/ppc_82xx is the directory pointer

The 10.82.124.205/255.255.252.0 line indicates that the target with an IP of 10.82.124.205 can access this directory /opt/ppc_82xx

```
rw,no_root_squash)
```

The rw indicates read/write permission, no_root_squash indicates do not squash the root user, that is, allow the target root user, root access to this partition.

Dissecting line 5.

/opt/ppc_74xx is the directory pointer

The 10.82.116.254/255.255.252.0 line indicates that a target with an IP of 10.82.116.254 can access this directory /opt/ppc_74xx

```
rw,no_root_squash)
```

The rw indicates read/write permission, no_root_squash do not squash the root user, that is, allow the target root user root access to this partition.

Thus, when the targets announce themselves as IP address 10.82.124.205, NFS will mount /opt/ppc_82xx, or as 10.82.116.254, NFS will mount /opt/ppc_74xx.

However, for the splash screens supplied, edit this file and change lines 4 and 5 to correspond to the actual IP addresses that were used. Change these IP addresses to correspond to the addresses that are actually used.

Thus lines 4 and 5 are changed to the following:

```
4  /opt/ppc_82xx 10.82.0.105/255.255.252.0(rw,no_root_squash)

5  /opt/ppc_74xx 10.82.118.204/255.255.252.0(rw,no_root_squash)
```

4.2.2 Export the New Directories

Whenever the /etc/exports file is updated the NFS portmapper daemon must be restarted.

There are three ways to accomplish this. The exportfs command is the recommended method, I will only mention the other two. Do not do all three of these or even two of them, just do the exportfs -r if it is available.

4.2.2.1 The exportfs -r Command

The command exportfs -r will export the directories declared in the /etc/exports file. The -r parameter indicates that a re-export is requested, that is, export the files again even if already exported.

```
root@debian:/etc# exportfs -r

exportfs: /etc/exports [4]: No 'sync' or 'async' option specified for export
"10.82.0.105/255.255.252.0:/opt/ppc_82xx".

  Assuming default behaviour ('sync').

  NOTE: this default has changed from previous versions

exportfs: /etc/exports [5]: No 'sync' or 'async' option specified for export
"10.82.118.204/255.255.252.0:/opt/ppc_74xx".

  Assuming default behaviour ('sync').

  NOTE: this default has changed from previous versions
```

4.2.2.2 nfs-kernel-server

Run the script /etc/init.d/nfs-kernel-server

```
root@debian:/etc/init.d# ./nfs-kernel-server restart

Stopping NFS kernel daemon: mountd nfsd.

Unexporting directories for NFS kernel daemon...done.

Exporting directories for NFS kernel daemon...exportfs: /etc/exports [4]: No 'sync'
or 'async' option specified for export "10.82.0.105/255.255.252.0:/opt/ppc_82xx".

  Assuming default behaviour ('sync').

  NOTE: this default has changed from previous versions

exportfs: /etc/exports [5]: No 'sync' or 'async' option specified for export
"10.82.118.204/255.255.252.0:/opt/ppc_74xx".

  Assuming default behaviour ('sync').

  NOTE: this default has changed from previous versions

done.

Starting NFS kernel daemon: nfsd mountd.
```

4.2.2.3 A Manual Method of Restarting the Deamons

Run the two scripts in the /etc/rc2.d directory that stop/start the nfs kernel daemon and the portmap daemon. This is not recommended and should only be used if the exportsfs or nfs-kernel-server script is not available.

```
root@debian:/etc/rc2.d# ./K20nfs-kernel-server stop

Stopping NFS kernel daemon: mountd nfsd.

Unexporting directories for NFS kernel daemon...done.

root@debian:/etc/rc2.d# ./K18portmap stop

Stopping portmap daemon: portmap.

root@debian:/etc/rc2.d# ./K18portmap start

Starting portmap daemon: portmap.

root@debian:/etc/rc2.d# ./K20nfs-kernel-server start

Exporting directories for NFS kernel daemon...exportfs: /etc/exports [4]: No 'sync'
or 'async' option specified for export "10.82.10.105/255.255.252.0:/opt/ppc_82xx".

  Assuming default behaviour ('sync').

  NOTE: this default has changed from previous versions

exportfs: /etc/exports [5]: No 'sync' or 'async' option specified for export
"10.82.118.204/255.255.252.0:/opt/ppc_74xx".

  Assuming default behaviour ('sync').
```

```
  NOTE: this default has changed from previous versions
done.
Starting NFS kernel daemon: nfsd mountd.
```

5 Downloading a Kernel from a Host

Building a Linux kernel on a native system is the same as building it on a cross architecture system, with the exception that native GCC tools are used. See AN2578, *Creating a Linux 'Out of the Box' Experience on a Sandpoint Platform* for instructions on building a kernel from the sources available on the web.

Assuming that a Linux kernel has been built with the name, uImage.ads60 for ADS and zImage.sandpoint for the Sandpoint, copy them to the /tftpboot directory.

These images are in the /tftpboot directory. However, the uImage must be for the specific board and CPU. It works on our ADS8260 board. The zImage.sandpoint works on our sandpoint with an MPC74xx with 128MB of memory or an MPC8245 with 128 MB of memory.

TFTP is a simple mechanism for transmitting files over the ethernet. It allows the sending of files from only one directory on the host, the default of which is, /tftpboot. Therefore, copy the Linux kernels to this directory.

Configure and start the TFTP server in /etc/inetd.conf

The file /etc/inetd.conf is read during boot and each service selected in this file is started. The TFTP section is shown here.

```
cat /etc/inetd.conf
```

.... lines from the top to this point are not shown here

```
#:BOOT: Tftp service is provided primarily for booting.  Most sites
# run this only on machines acting as "boot servers."
tftp            dgram   udp     wait    nobody  /usr/sbin/tcpd  /usr/sbin/in.tftpd
/tftpboot
```

.... the rest of the file is not shown here.

Figure 4 shows the Linux kernel downloading to the target via the TFTP facility.

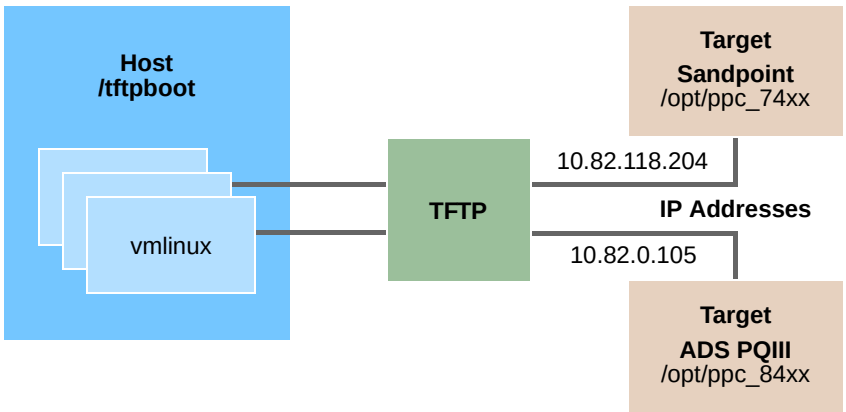

Figure 4. TFTPing a File to the Target

6 Connecting the NFS Root File System to a Target

It is arbitrary which NFS root file system we assign to either target, but for this example, we want the Sandpoint to NFS mount the /opt/ppc_74xx root file system and the ADS to NFS mount the /opt/ppc_82xx file system. This choice is made by choosing the IP address of the target, The Sandpoint will use IP address 10.82.118.204 which will NFS mount /opt/ppc_74xx. The ADS will use IP address 10.82.0.105, which will NFS mount /opt/ppc_82xx. Both choices based on the /etc/exports file designation of IP address to NFS mount.

6.1 Using a Serial Connection Terminal Emulator

Debian Linux supports the minicom terminal emulator. In order to use minicom as a user, first change to root, and set the permissions on /dev/ttyS1 to everyone read/write, that is, 666.

```
root@debian:~# chmod 666 /dev/ttyS1

root@debian:~# ls -l /dev/ttyS1

crw-rw-rw-    1 root     dialout    4,  65 Feb 24 04:50 /dev/ttyS1

root@debian:~#
```

Now, any user can use this terminal device, /dev/ttyS1, which is the serial port on the Genesi Pegasos II computer. There is only one physical serial port on the Genesi Pegasos II computer.

The first time, minicom must be set up for serial remote operations by choosing these options, an example follows

- Choose Serial port setup
- Choose /dev/ttyS1, which is the default.
- Choose appropriate baud rate, which is
  - E - Bps/Par/Bits : 115200 8N1 for the ADS and
  - E - Bps/Par/Bits : 9600 8N1 for the Sandpoint
- Save setup as dfl
- Exit from Minicom.

Invoke minicom with the -s parameter.

```
minicom -s
```

```
            +-----[configuration]------+
            ¦ Filenames and paths      ¦
            ¦ File transfer protocols  ¦
            ¦ Serial port setup        ¦
            ¦ Modem and dialing        ¦
            ¦ Screen and keyboard      ¦
            ¦ Save setup as dfl        ¦
            ¦ Save setup as..          ¦
```

```
            ¦ Exit                    ¦
            ¦ Exit from Minicom       ¦
            +-------------------------+
```

Use the down arrow to choose Serial port setup

```
+---------------------------------------------------------------------+
    ¦ A -    Serial Device      : /dev/ttyS1                          ¦
    ¦ B - Lockfile Location     : /var/lock                           ¦
    ¦ C -   Callin Program      :                                     ¦
    ¦ D -  Callout Program      :                                     ¦
    ¦ E -    Bps/Par/Bits       : 115200 8N1                          ¦
    ¦ F - Hardware Flow Control : No                                  ¦
    ¦ G - Software Flow Control : No                                  ¦
    ¦                                                                 ¦
    ¦    Change which setting?                                        ¦
    +---------------------------------------------------------------------+
```

Choose a, Serial Device by keying in the letter a. Always choose /dev/ttyS1. choose e Bps/PAR/Bits

```
                  +-----------[Comm Parameters]------------+
    +---------------¦                                      ¦--------------+
¦ Current: 115200 8N1                   ¦              ¦
¦                                       ¦              ¦
¦   Speed          Parity          Data ¦              ¦
¦                                       ¦              ¦
¦ A: 300           L: None         S: 5 ¦              ¦
¦ B: 1200          M: Even         T: 6 ¦              ¦
¦ C: 2400          N: Odd          U: 7 ¦              ¦
¦ D: 4800          O: Mark         V: 8 ¦              ¦
¦ E: 9600          P: Space             ¦              ¦
¦ F: 19200                      Stopbits ¦--------------+
¦ G: 38400                          W: 1 ¦
¦ H: 57600                          X: 2 ¦
¦ I: 115200        Q: 8-N-1             ¦
```

```
¦ J: 230400        R: 7-E-1              ¦
¦                                        ¦
¦                                        ¦
¦ Choice, or <Enter> to exit?            ¦
+----------------------------------------+
```

choose the correct baud rate for the board. For ADS choose 115200, choice I, for the Sandpoint choose 9600, choice E.

```
            +-----[configuration]------+
            ¦ Filenames and paths      ¦
            ¦ File transfer protocols  ¦
            ¦ Serial port setup        ¦
            ¦ Modem and dialing        ¦
            ¦ Screen and keyboard      ¦
            ¦ Save setup as dfl        ¦
            ¦ Save setup as..          ¦
            ¦ Exit                     ¦
            ¦ Exit from Minicom        ¦
            +--------------------------+
```

Choose Exit from Minicom.

Now, regardless of which target is being used, start minicom with the appropriate baud rate.

6.2 Sandpoint with an MPC74xx Processor

DINK32 does not use environment variables that can communicate with a download program. Therefore, the root command passed to Linux will have to be supplied during the Linux startup as shown in Section 6.2.3, “Downloading Linux” The root command is shown below:

Linux/PPC load: console=ttyS0,38400 root=/dev/nfs rw nfsroot=10.82.117.52:/opt/ppc_74xx ip=10.82.118.204:10.82.117.52:10.82.119.254:255.255.252.0:sandpoint:eth0:off

The example uses the IP address of 10.82.118.204, so that the host, Genesi Pegasos II at IP address 10.82.117.52 will NFS mount the /opt/ppc_74xx directory.

It is possible to change the DINK32 baud rate with an environment variable or the setbaud command. However, there is no advantage to running faster on the terminal, since we are downloading via the ethernet. See *DINK32 PowerPC Debugger User’s Manual,* for more information on this topic.

6.2.1 Start minicom

Use baud rate of 9600, start minicom with no parameters. This example, however, uses 38400 baud rate, so set minicom to 38400. Set the minicom baud rate to correspond to whatever baud rate the Sandpoint uses and what the Linux kernel is built to use during startup. The root command, talked about later, sets the baud rate when the kernel is running, that is, after Linux startup.

The Sandpoint board uses the DINK32 firmware package.

```
guest@debian:~$ minicom -s

Welcome to minicom 2.1


OPTIONS: History Buffer, F-key Macros, Search History Buffer, I18n

Compiled on Nov 12 2003, 20:34:49.


Press CTRL-A Z for help on special keys
```

The splash screen should now display as shown below in Section 6.2.2, “DINK32” and allow communication with the Sandpoint board.

To quit minicom, type the following command:

```
control-A

x
```

Set minicom to wrap lines rather than start a new line for lines that go past the end of the line. This is necessary for the root parameter. Turn minicom wrap on with the following commands.

```
control-a

z

w
```

Now minicom will indicate that it is using wrap mode.

6.2.2 DINK32

DINK32 will startup on the Sandpoint giving this splash screen.

```
DINK32[MPC7457] {2} >>ab
Caches Enabled: [ L1I(32K), L1D(32K), L2(512K) ]



        ## ##      ##
        ## ##      ##
        ##        ##
      ####### ## ####### ##  ##
     ##  ## ## ##  ## ## ##
     ##  ## ## ##  ## ######
     ##  ## ## ##  ## ## ##
      ######  ## ##  ## ##  ##


   (     (   ( ( (AltiVec) ) )   )     )


   Version : 13.1.1, Metaware Build
  Released : INTERIM( build 780 on Apr  2 2004 13:42:29 )
Written by : Motorola CPD/NCSG PowerPC Applications, Austin, TX
    System : Sandpoint X3 with Valis (MPMC7457)
 Processor : MPC7457 V1.2 @ 1000 MHz, 100 MHz memory
    Memory : Map B (CHRP) 256MB at CL=2


Copyright 1993-2004, by Motorola Inc.
```

6.2.3 Downloading Linux

DINK32 does not use environment variables to download and start Linux. Rather, set up the network interface and download Linux from /tftpboot on the host, with commands.

Using the `ni` command, assign the IP information to allow DINK32 to do an ethernet download.

```
DINK32[MPC7457] {5} >>ni -p

SERVER(TFTP) : [ 10. 82.118.204] : 10.82.117.52

GATEWAY      : [ 10. 82.119.252] :

NETMASK      : [255.255.252.  0] :

DHCP         : [ 10. 82.250. 25] :

CLIENT(DINK) : [ 10. 82.116.154] :

DINK32[MPC7457] {5} >>

DINK32[MPC7457] {5} >>
```

```
DINK32[MPC7457] {5} >>

DINK32[MPC7457] {5} >>
```

Now use the DINK32 download command, `dl`.

> The `dl` command also allows the -o parameter to specify the memory address to download the kernel image. The sandpoint kernel image will download to 100000 by default, but may be downloaded to any address greater than 100000. The kernel will relocate itself to 800000 when started with the go command.

```
DINK32[MPC7457] {5} >>dl -nw -b -fzImage.sandpoint

Filename = zImage.sandpoint

File format =  Plain binary

Default Offset = 0x00100000

Received 2804 TFTP data blocks.



Successfully read 1435171 bytes via TFTP at  2299692 bytes/sec
```

6.2.4 Starting Linux on the Sandpoint

To start Linux, use the DINK32 `go` command. Then when the 'Linux/PPC load:' prompt appears, hit backspace and type the very long line of information specific to your network configuration into the root request. It is suggested that this line be typed into a file through an editor and then pasted into the minicom terminal interface when requested. Below is an example of this line:

Linux/PPC load: console=ttyS0,38400 root=/dev/nfs rw nfsroot=10.82.117.52:/opt/ppc_74xx ip=10.82.118.204:10.82.117.52:10.82.119.254:255.255.252.0:sandpoint:eth0:off

The syntax for this line is as follows:

Linux/PPC load: console=<terminal designator>,<baud rate>

space

root=/dev/nfs "indicates use nfs mount"

space

rw "indicates read write permission on the root file system"

space

nfsroot=<server IP address>:

<server directory address of NFS mount>

one or more spaces

ip=<this machines IP address>:

<server's IP address>:

<gateway IP address>:

<network IP netmask>:

<any name you choose>: "in this case we chose sandpoint, but the name is arbitrary"

eth0:off

Carefully type this line, it is one line, no carriage returns, the colons and spaces are required.

```
DINK32[MPC7457] {6} >>go 110000
loaded at:     00110000 002601DC
relocated to:  00800000 009501DC
zimage at:     0080586B 0094CF3E
avail ram:     00400000 00800000


Uncompressing Linux...done.
Now booting the kernel
Total memory = 256MB; using 512kB for hash table (at c0300000)
Linux version 2.6.3 (jacob@montero) (gcc version 3.3) #47 Fri Mar 19 00:41:59 EST 2004
Motorola SPS Sandpoint Test Platform
Port by MontaVista Software, Inc. (source@mvista.com)
On node 0 totalpages: 65536
  DMA zone: 65536 pages, LIFO batch:16
  Normal zone: 0 pages, LIFO batch:1
  HighMem zone: 0 pages, LIFO batch:1
Built 1 zonelists
Kernel command line: console=ttyS0,38400 root=/dev/hda3
```

At this point, stop the kernel load by hitting the backspace key and rub out the /dev/hda3, in its place type this root line or paste it in from the file created earlier.

Kernel command line: console=ttyS0,38400 root=/dev/nfs rw nfsroot=10.82.117.52:/opt/ppc_74xx ip=10.82.118.20:10.82.117.52:10.82.119.254:255.255.252.0:sandpoint:eth0:off

Now the kernel will continue and boot.

```
OpenPIC Version 1.2 (1 CPUs and 16 IRQ sources) at fdfd0000
OpenPIC timer frequency is 100.000000 MHz
PID hash table entries: 2048 (order 11: 16384 bytes)
time_init: decrementer frequency = 24.358929 MHz
Console: colour dummy device 80x25
Memory: 255616k available (1964k kernel code, 936k data, 100k init, 0k highmem)
Calibrating delay loop... 972.80 BogoMIPS
Dentry cache hash table entries: 32768 (order: 5, 131072 bytes)
Inode-cache hash table entries: 16384 (order: 4, 65536 bytes)
Mount-cache hash table entries: 512 (order: 0, 4096 bytes)
POSIX conformance testing by UNIFIX
NET: Registered protocol family 16
PCI: Probing PCI hardware
SCSI subsystem initialized
Registering Sandpoint 7447A CPU frequency driver
Low: 650 Mhz, High: 1300 Mhz, Boot: 1300 Mhz, switch method: CPU
INFO: enter cpufreq_cpu_init
.INFO: enter cpufreq_cpu_init
JPAN:table_verify() min=650000, max=1300000
JPAN:table_verify() min=650000, max=1300000
Installing knfsd (copyright (C) 1996 okir@monad.swb.de).
Macintosh non-volatile memory driver v1.1
Serial: 8250/16550 driver $Revision: 1.90 $ 6 ports, IRQ sharing disabled
ttyS0 at MMIO 0x0 (irq = 4) is a NS16550A
ttyS1 at MMIO 0x0 (irq = 3) is a NS16550A
RAMDISK driver initialized: 16 RAM disks of 16384K size 1024 blocksize
loop: loaded (max 8 devices)
pcnet32.c:v1.27b 01.10.2002 tsbogend@alpha.franken.de
8139too Fast Ethernet driver 0.9.27
eth0: RealTek RTL8139 at 0xd1000f00, 00:40:f4:7b:c0:09, IRQ 21
Uniform Multi-Platform E-IDE driver Revision: 7.00alpha2
```

```
ide: Assuming 33MHz system bus speed for PIO modes; override with idebus=xx
W82C105: IDE controller at PCI slot 0000:00:0b.1
W82C105: chipset revision 5
W82C105: 100% native mode on irq 16
    ide0: BM-DMA at 0xbfffd0-0xbfffd7, BIOS settings: hda:DMA, hdb:DMA
    ide1: BM-DMA at 0xbfffd8-0xbfffdf, BIOS settings: hdc:DMA, hdd:DMA
Probing IDE interface ide0...
hda: WDC WD400BB-00DEA0, ATA DISK drive
hda: selected PIO 4 (120ns) (0240)
hda: DMA enabled
Using anticipatory io scheduler
ide0 at 0xbffff8-0xbffffc,0xbffff6 on irq 14
Probing IDE interface ide1...
ide1: Wait for ready failed before probe !
Probing IDE interface ide1...
ide1: Wait for ready failed before probe !
hda: max request size: 128KiB
hda: 78165360 sectors (40020 MB) w/2048KiB Cache, CHS=65535/16/63, (U)DMA
 hda: hda1 hda2 hda3 hda4
ide-floppy driver 0.99.newide
st: Version 20040122, fixed bufsize 32768, s/g segs 256
mice: PS/2 mouse device common for all mice
serio: i8042 AUX port at 0x60,0x64 irq 12
atkbd.c: keyboard reset failed on isa0060/serio1
serio: i8042 KBD port at 0x60,0x64 irq 1
atkbd.c: keyboard reset failed on isa0060/serio0
NET: Registered protocol family 2
IP: routing cache hash table of 2048 buckets, 16Kbytes
TCP: Hash tables configured (established 16384 bind 32768)
eth0: link up, 100Mbps, full-duplex, lpa 0x41E1
IP-Config: Complete:
device=eth0, addr=10.82.118.20, mask=255.255.252.0, gw=10.82.119.254,
```

```
     host=sandpoint, domain=, nis-domain=(none),
     bootserver=10.82.117.52, rootserver=10.82.117.52, rootpath=
NET: Registered protocol family 1
NET: Registered protocol family 17
NET: Registered protocol family 8
NET: Registered protocol family 20
Looking up port of RPC 100003/2 on 10.82.117.52
Looking up port of RPC 100005/1 on 10.82.117.52
VFS: Mounted root (nfs filesystem).
Freeing unused kernel memory: 100k init 4k pmac
INIT: version 2.78 booting
                Welcome to DENX Embedded Linux Environment
                Press 'I' to enter interactive startup.
Mounting proc filesystem:  [  OK  ]
Configuring kernel parameters:  [  OK  ]
Cannot access the Hardware Clock via any known method.
Use the --debug option to see the details of our search for an access method.
Setting clock : Thu Jun 24 22:58:44 EDT 2004 [  OK  ]
Activating swap partitions:  [  OK  ]
Setting hostname sandpoint:  [  OK  ]
modprobe: Can't open dependencies file /lib/modules/2.6.3/modules.dep (No such file
or directory)
Checking filesystems
[  OK  ]
Mounting local filesystems:  [  OK  ]
Enabling swap space:  [  OK  ]
INIT: Entering runlevel: 3
Entering non-interactive startup
Setting network parameters:  [  OK  ]
Bringing up interface lo:  [  OK  ]
Starting system logger: [  OK  ]
Starting kernel logger: [  OK  ]
```

```
Starting ntpd: [  OK  ]
Starting xinetd: [  OK  ]
```

At this point Linux will ask for a login. Since the ELDK is being used, a minimal root file system via NFS, the only login user is root with no password. So log in as root. The `ifconfig` command will show that we are using IP address 10.82.118.204.

```
sandpoint login: root
Last login: Wed May 12 02:28:37 on console
bash-2.05# ls
Makefile  Makefile~  benchmarks  whichcpu  whichcpu.c
bash-2.05#
bash-2.05# ifconfig eth0
eth0      Link encap:Ethernet  HWaddr 00:40:F4:7B:C0:09
          inet addr:10.82.118.204  Bcast:10.82.119.255  Mask:255.255.252.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:6867 errors:0 dropped:0 overruns:0 frame:0
          TX packets:3751 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:6553788 (6.2 Mb)  TX bytes:607222 (592.9 Kb)
          Interrupt:21 Base address:0xf00



bash-2.05#
```

6.2.5 Logging into the Sandpoint Linux from Genesi Pegasos II

Now that Linux is running on the Sandpoint and its IP address is known, start an ethernet terminal session with the telnet command from any other computer.

```
Telnet from Genesi
guest@debian:~$ telnet 10.82.118.204
Trying 10.82.118.204...
Connected to 10.82.118.204.
Escape character is '^]'.
```

```
Linux 2.6.3 (sandpoint) (23:34 on Thursday, 24 June 2004)


login: root

Last login: Thu Jun 24 23:33:40 on console

bash-2.05# ls

Makefile  Makefile~  benchmarks  whichcpu  whichcpu.c

bash-2.05#
```

6.3 ADS with a PowerQUICC III Processor

The ADS board uses environment variables to boot and pass boot parameters to Linux. U-boot the firmware on the ADS board runs at a baud rate of 115200.

Use IP address 10.82.0.105, which select /opt/ppc_82xx, use serial terminal at baud rate of 115200.

> Using the IP address of 10.82.0.105, so that the host, Genesi Pegasos II at IP address 10.82.117.52 will NFS mount the /opt/ppc_82xx directory.

The Linux boot parameters are as follows:

```
=> setenv ipaddr 10.82.0.105

=> setenv serverip 10.82.117.52

=> setenv gatewayip 10.82.1.254

=> setenv hostname komodo

=> setenv bootcmd $nfsboot

=> setenv bootfile uImage.ads60

=> setenv rootpath /opt/ppc_82xx

=> setenv netmask 255.255.252.0

=> setenv netdev eth2
```

The variable, $nfsboot is set to this value

```
setenv bootargs root=/dev/nfs rw nfsroot=$serverip:$rootpath
ip=$ipaddr:$serverip:$gatewayip:$netmask:$hostname:$netdev:off
console=$consoledev,$baudrate $othbootargs;tftp $loadaddr $bootfile;bootm $loadaddr
```

6.3.1 Start minicom

Use baud rate of 115200, start minicom with no parameters.

```
guest@debian:~$ minicom -s

Welcome to minicom 2.1


OPTIONS: History Buffer, F-key Macros, Search History Buffer, I18n

Compiled on Nov 12 2003, 20:34:49.


Press CTRL-A Z for help on special keys
```

The splash screen should now display as shown in Section 6.3.2, “U-boot—Setting up Environment Variables” and allow communication with the ADS board.

To quit minicom,

```
control-A

x
```

Set minicom to wrap lines, rather than start a new line for lines that go past the end of the line. This is necessary for the root parameter. Turn minicom wrap on with the following commands.

```
control-a

z

w
```

Now minicom will indicate that it is using wrap mode.

6.3.2 U-boot—Setting up Environment Variables

The ADS board uses the U-Boot firmware package. When the ADS board is powered on, u-boot will present this splash screen.

```
U-Boot 1.1.2(pq3-20040707-0) (Jul  6 2004 - 17:37:04)


Freescale PowerPC

    Core: E500, Version: 2.0, (0x80200020)

    System: 8560, Version: 2.0, (0x80700020)

    Clocks: CPU: 660 MHz, CCB: 264 MHz, DDR: 132 MHz, LBC:  66 MHz
```

```
    CPM: 264 Mhz
    L1 D-cache 32KB, L1 I-cache 32KB enabled.
Board: ADS
    PCI1: 32 bit, 66 MHz (compiled)
I2C:   ready
DRAM:  Initializing
    SDRAM: 64 MB
    DDR: 128 MB
FLASH: 16 MB
L2 cache enabled: 256KB
In:    serial
Out:   serial
Err:   serial
Net:   MOTO ENET0: PHY is Marvell 88E1011S (1410c62)
MOTO ENET1: PHY is Marvell 88E1011S (1410c62)
MOTO ENET0, MOTO ENET1
Hit any key to stop autoboot:
```

Now, change the environment variables to indicate the IP desired, the bootfile and command, and other parameters as shown below.

```
=> setenv ipaddr 10.82.0.105
=> setenv serverip 10.82.117.52
=> setenv gatewayip 10.82.1.254
=> setenv hostname komodo
=> setenv bootcmd $nfsboot
=> setenv bootfile uImage.ads60
=> setenv rootpath /opt/ppc_82xx
=> setenv netmask 255.255.252.0
=> setenv netdev eth2
```

Now use the command `printenv` to see the values of the environment variables.

```
=> printenv
```

```
nfsboot=setenv bootargs root=/dev/nfs rw nfsroot=$serverip:$rootpath
ip=$ipaddr:$serverip:$gatewayip:$netmask:$hostname:$netdev:off
console=$consoledev,$baudrate $othbootargs;tftp $loadaddr $bootfile;bootm $loadaddr

bootdelay=10

baudrate=115200

loads_echo=1

ethaddr=00:E0:0C:00:00:FD

eth1addr=00:E0:0C:00:01:FD

eth2addr=00:E0:0C:00:02:FD

loadaddr=200000

ethact=MOTO ENET0

ramdiskfile=afleming/ramdisk-main.u-boot

ramdiskaddr=1000000

consoledev=ttyS0

stdin=serial

stdout=serial

stderr=serial

ipaddr=10.82.0.105

serverip=10.82.117.52

gatewayip=10.82.1.254

hostname=komodo

bootcmd=setenv bootargs root=/dev/nfs rw nfsroot=$serverip:$rootpath
ip=$ipaddr:$serverip:$gatewayip:$netmask:$hostname:$netdev:off
console=$consoledev,$baudrate $othbootargs;tftp $loadaddr $bootfile;bootm $loadaddr

bootfile=uImage.ads60

rootpath=/opt/ppc_82xx

netmask=255.255.252.0

netdev=eth2


Environment size: 1034/8188 bytes
```

Optionally, save the environment for future boots with the `saveenv` command.

6.3.3 Invocation

Issue the u-boot command boot, which will invoke the argument bootcmd to download the Linux kernel to the target and start it executing.

- Send a TFTP read request (RRQ) to the host
  - — which will start downloading uImage.ads60 from the host directory, /tftpboot.
- Start executing at uImage.ads60 entry point
- The Linux kernel boot wrapper starts on the target
- The Linux kernel parses the boot options
  - — using the u-boot argument bootargs
- The Linux kernel will mount the NFS root to the host exported directory /opt/ppc_82xx
  - — because the IP address specified indicates this NFS mount via the host /etc/exports file.
- The Linux kernel will execute the /sbin/init program on the exported NFS root file system
  - — which is from the host hard disk target,
    - – i.e. the NFS mounted root directory, specifically, /opt/ppc_82xx/sbin/init on the host.

The u-boot console will now display

```
=> boot

Speed: 100, full duplex

Using MOTO ENET0 device

TFTP from server 10.82.117.52; our IP address is 10.82.0.105; sending through gateway
10.82.1.254

Filename 'uImage.ads60'.

Load address: 0x200000

Loading: *#################################################################
         #################################################################
         ###########################################################
done

Bytes transferred = 943035 (e63bb hex)

## Booting image at 00200000 ...
   Image Name:   Linux-2.4.26-pre5-moto-pq3-2004_
   Image Type:   PowerPC Linux Kernel Image (gzip compressed)
   Data Size:    942971 Bytes = 920.9 kB
   Load Address: 00000000
   Entry Point:  00000000
   Verifying Checksum ... OK
   Uncompressing Kernel Image ... OK
```

```
Memory CAM mapping: CAM0=64Mb, CAM1=64Mb, CAM2=0Mb residual: 0Mb

Linux version 2.4.26-pre5-moto-pq3-2004_04_23-0 (galak@blarg.somerset.sps.mot.com)
(gcc version 3.2.2 20030217 (Yellow Dog Linux 3.0 3.2.2-2a)) #4 Fri Apr 23 19:50:02
CDT 2004

On node 0 totalpages: 32768

zone(0): 32768 pages.

zone(1): 0 pages.

zone(2): 0 pages.

Kernel command line: root=/dev/nfs rw nfsroot=10.82.117.52:/opt/ppc_82xx
ip=10.82.0.105:10.82.117.52:10.82.1.254:255.255.252.0:komodo:eth2:off
console=ttyS0,115200

OpenPIC Version 1.2 (1 CPUs and 44 IRQ sources) at e0040000

time_init: decrementer frequency = 33.000000 MHz

Calibrating delay loop... 658.63 BogoMIPS

Memory: 127168k available (1504k kernel code, 448k data, 248k init, 0k highmem)

Dentry cache hash table entries: 16384 (order: 5, 131072 bytes)

Inode cache hash table entries: 8192 (order: 4, 65536 bytes)

Mount cache hash table entries: 512 (order: 0, 4096 bytes)

Buffer cache hash table entries: 8192 (order: 3, 32768 bytes)

Page-cache hash table entries: 32768 (order: 5, 131072 bytes)

POSIX conformance testing by UNIFIX

PCI: Probing PCI hardware

PCI: moved device 00:00.1 resource 0 (200) to 80000000

PCI: Failed to allocate resource 1(0-7fffffff) for 00:00.1

Linux NET4.0 for Linux 2.4

Based upon Swansea University Computer Society NET3.039

Initializing RT netlink socket

Starting kswapd

Installing knfsd (copyright (C) 1996 okir@monad.swb.de).

i2c-core.o: i2c core module version 2.6.1 (20010830)

i2c-dev.o: i2c /dev entries driver module version 2.6.1 (20010830)

i2c-proc.o version 2.6.1 (20010830)

CPM UART driver version 0.01
```

```
ttyS0 on SCC1 at 0x8000, BRG1
ttyS1 on SCC2 at 0x8100, BRG2
pty: 256 Unix98 ptys configured
eth0: FCC ENET Version 0.3, 00:e0:0c:40:00:fd
eth0: Phy @ 0x2, type Davicom DM9161E (0x0181b881)
eth1: FCC ENET Version 0.3, 00:e0:0c:20:00:fd
eth1: Phy @ 0x3, type Davicom DM9161E (0x0181b881)
RAMDISK driver initialized: 16 RAM disks of 32768K size 1024 blocksize
loop: loaded (max 8 devices)
Intel(R) PRO/1000 Network Driver - version 5.2.30.1-k1
Copyright (c) 1999-2004 Intel Corporation.
eth2: Gianfar Ethernet Controller Version 1.0, 00:e0:0c:00:00:fd
eth2: Running with NAPI disabled
eth2: 64/64 RX/TX BD ring size
eth3: Gianfar Ethernet Controller Version 1.0, 00:e0:0c:00:01:fd
eth3: Running with NAPI disabled
eth3: 64/64 RX/TX BD ring size
PPP generic driver version 2.4.2
PPP Deflate Compression module registered
Uniform Multi-Platform E-IDE driver Revision: 7.00beta4-2.4
ide: Assuming 33MHz system bus speed for PIO modes; override with idebus=xx
mpc-i2c0: scanning bus...
 0x31 found
 0x50 found
mpc-i2c0: bus was busy
mpc-i2c0: bus was busy
mpc-i2c0: bus was busy
mpc-i2c0: 2 device(s) detected
NET4: Linux TCP/IP 1.0 for NET4.0
IP Protocols: ICMP, UDP, TCP, IGMP
IP: routing cache hash table of 1024 buckets, 8Kbytes
TCP: Hash tables configured (established 8192 bind 16384)
```

```
eth2: PHY is Marvell 88E1011S (1410c62)
eth2: Auto-negotiation done
eth2: Full Duplex
eth2: Speed 100BT
eth2: Link is up
IP-Config: Complete:
      device=eth2, addr=10.82.0.105, mask=255.255.252.0, gw=10.82.1.254,
     host=komodo, domain=, nis-domain=(none),
     bootserver=10.82.117.52, rootserver=10.82.117.52, rootpath=
NET4: Unix domain sockets 1.0/SMP for Linux NET4.0.
Looking up port of RPC 100003/2 on 10.82.117.52
Looking up port of RPC 100005/1 on 10.82.117.52
VFS: Mounted root (nfs filesystem).
Freeing unused kernel memory: 248k init
INIT: version 2.78 booting
               Welcome to DENX Embedded Linux Environment
               Press 'I' to enter interactive startup.
Mounting proc filesystem:  [  OK  ]
Configuring kernel parameters:  [  OK  ]
Cannot access the Hardware Clock via any known method.
Use the --debug option to see the details of our search for an access method.
Setting clock : Wed Dec 31 19:00:05 EST 1969 [  OK  ]
Activating swap partitions:  [  OK  ]
Setting hostname komodo:  [  OK  ]
modprobe: Can't open dependencies file
/lib/modules/2.4.26-pre5-moto-pq3-2004_04_23-0/modules.dep (No such file or
directory)
Finding module dependencies:  depmod: Can't open
/lib/modules/2.4.26-pre5-moto-pq3-2004_04_23-0/modules.dep for writing
[FAILED]
Checking filesystems
[  OK  ]
Mounting local filesystems:  [  OK  ]
```

```
Enabling swap space:  [  OK  ]

INIT: Entering runlevel: 3

Entering non-interactive startup

Setting network parameters:  [  OK  ]

Bringing up interface lo:  [  OK  ]

Starting system logger: [  OK  ]

Starting kernel logger: [  OK  ]

Starting ntpd: [  OK  ]

Starting xinetd: [  OK  ]
```

Now Linux is running on the ADS board and will accept all Linux commands available in the minimal NFS root file system that is mounted, /opt/ppc_82xx from the host.

An example of the root login is shown below. It is the only login, since this is a minimal root file system. No other users can be created since the adduser command is not available. Also, root user has no password. After the login, the current/root directory is shown, followed by an example of the program whichcpu, then /proc/cpuinfo, which shows the same information. Finally, an example using /proc/devices shows all the available devices.

```
komodo login: root

Last login: Wed Dec 31 19:00:16 on console

lbash-2.05# ls

Makefile benchmarks temp whichcpu whichcpu.c

[mbash-2.05#

bash-2.05# ./whichcpu



-----------------------------------------------------------------


2004 FAE Training test program, printing local CPU info

processor: 0

cpu             : e500

revision: 2.0 (pvr 8020 0020)

bogomips: 658.63

Vendor          : Motorola SPS

Machine         : mpc8560ads

bus freq: 660.000000 MHz
```

```
PVR             : 0x80200020
SVR             : 0x80700020
PLL setting: 0x5
Memory          : 128 MB
-----------------------------------------------------------------
bash-2.05#
bash-2.05# cat /proc/cpuinfo
processor: 0
cpu             : e500
revision: 2.0 (pvr 8020 0020)
bogomips: 658.63
Vendor          : Motorola SPS
Machine         : mpc8560ads
bus freq: 660.000000 MHz
PVR             : 0x80200020
SVR             : 0x80700020
PLL setting: 0x5
Memory          : 128 MB
bash-2.05#      cat /proc/devices
Character devices:
  1 mem
  2 pty
  3 ttyp
  4 ttyS
  5 cua
  7 vcs
 10 misc
 89 i2c
108 ppp
128 ptm
136 pts
162 raw
```

```
Block devices:
  1 ramdisk
  7 loop
bash-2.05#
```

6.4 Ethernet Connection

The host can now connect via ethernet and the telnet or ssh program to either target. For simplicity, once the targets are running, the results will be the same, with the exception of which NFS mount is being used. Therefore, these examples will only use the ADS board. Results for the Sandpoint are the same.

6.4.1 Start a Telnet Session with the ADS Target on the Host

```
telnet 10.82.0.105
[bash-2.05# login
[bash-2.05# root]$
```

6.4.2 Start a Login Session with the Local Sandpoint Host

```
[appslab12.sps.mot.com:/2004] telnet 10.82.118.204
Trying 10.82.118.204...
Connected to 10.82.118.204.
Escape character is '^]'.

Linux 2.6.3 (sandpoint) (07:47 on Friday, 25 June 2004)

login: root
Last login: Thu Jun 24 23:34:33 from 10.82.117.52
```

6.4.3 Compare the Target with the Host

By using the Linux command cat of the special file /proc/cpuinfo, compare the two CPUs, one on the target and one on the host.

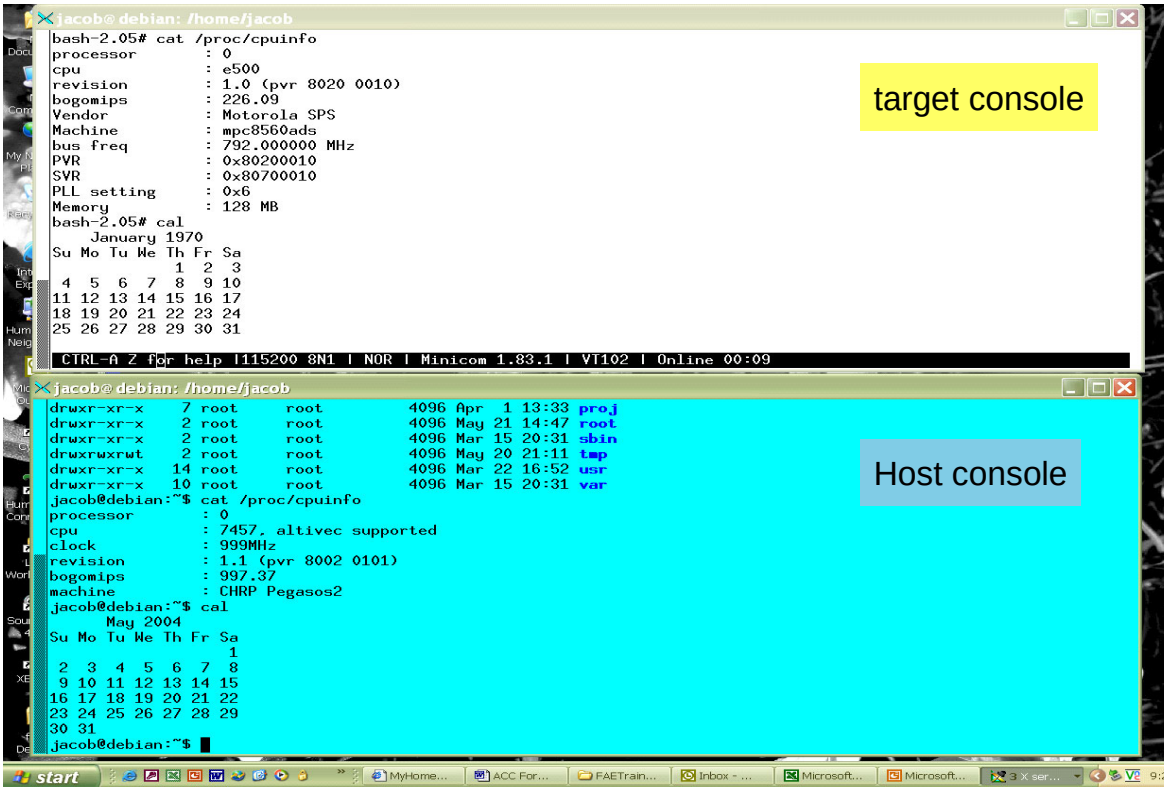

Figure 5. Target and Host CPU Information

By using the `ls` command look at the same root file system for the target, /,which is the root, and the directory for the host, /opt/ppc_82xx. These are the same directory and are shared between the target and the host.

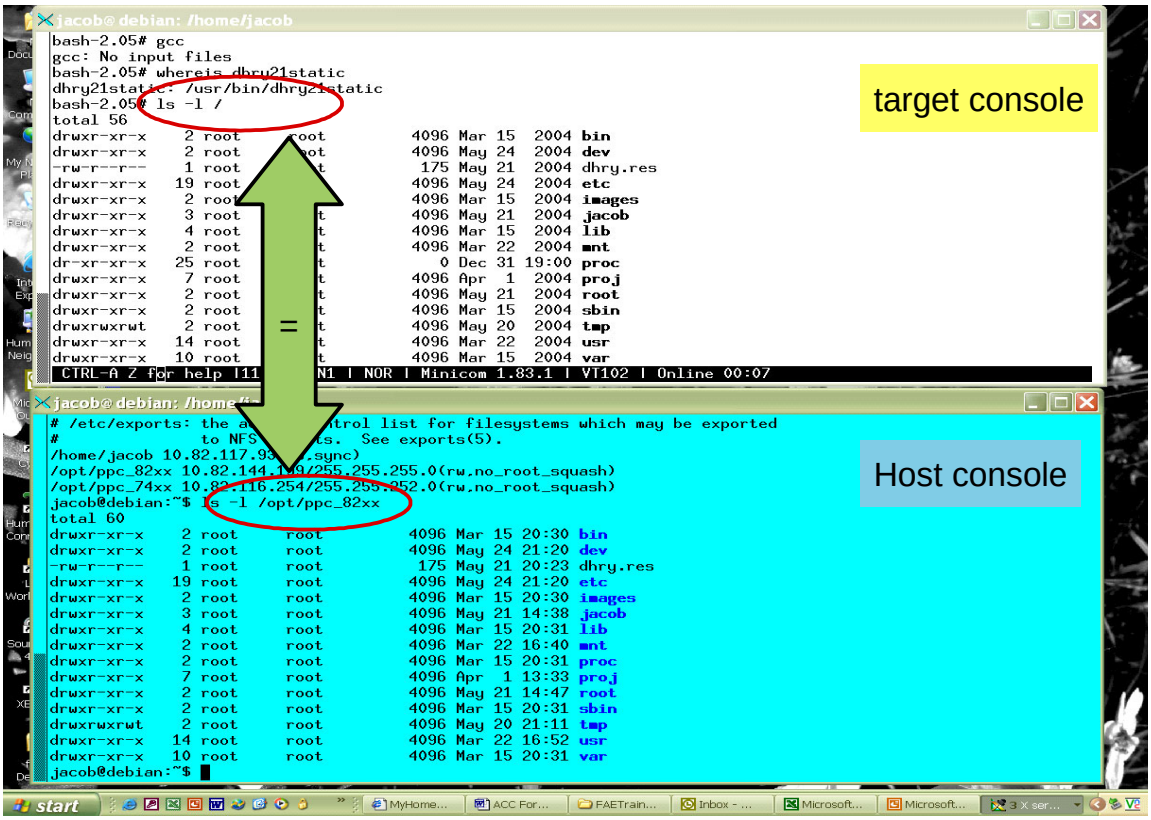

Figure 6. Looking at the Same Directory on Host and Target

It should be clear from the two screens in Figure 6 that the target has mounted an NFS root file system, which it sees as the root, /, which is in fact the same as the directory /opt/ppc_82xx on the host. Both the host and the target have access to this directory structure. Since the /etc/exports file has denoted this directory as mountable with no_root_squash, the target can write on this directory.

7 References

The following documents describe the various applications of the Genesi Pegasos II system.

1. Freescale application note AN2666, *Genesi Pegasos II Setup*
2. Freescale application note AN2736, *Genesi Pegasos II Boot Options*
3. Freescale application note AN2738, *Genesi Pegasos II Firmware*
4. Freescale application note AN2739, *Genesi Pegasos II Debian Linux*
5. Freescale application note AN2743, *Software Analysis on Genesi Pegasos II Using PMON and AltiVec*
6. Freescale application note AN2744, *PMON Module—An Example of Writing Kernel Module Code for Debian 2.6 on Genesi Pegasos II*
7. Freescale application note AN2578, *Creating a Linux 'Out of the Box' Experience on a Sandpoint Platform*, available on the Freescale Semiconductor, DINK32 web site.
8. *DINK32 PowerPC Debugger User's Manual*, available on the Freescale Semiconductor DINK32 web site.

For assistance or answers to any question on the information that is presented in this document, send an e-mail to risc10@freescale.com.

8 Document Revision History

Table 1 provides a revision history for this application note.

Table 1. Document Revision History

Rev. No.	Date	Substantive Change(s)
0	08/11/04	Initial release.

THIS PAGE INTENTIONALLY LEFT BLANK

THIS PAGE INTENTIONALLY LEFT BLANK

How to Reach Us:

USA/Europe/Locations Not Listed:
Freescale Semiconductor
Literature Distribution Center
P.O. Box 5405,
Denver, Colorado 80217
1-480-768-2130
(800) 521-6274

Japan:
Freescale Semiconductor Japan Ltd.
Technical Information Center
3-20-1, Minami-Azabu, Minato-ku
Tokyo 106-8573, Japan
81-3-3440-3569

Asia/Pacific:
Freescale Semiconductor Hong Kong Ltd.
2 Dai King Street
Tai Po Industrial Estate
Tai Po, N.T. Hong Kong
852-26668334

Home Page:
www.freescale.com

Learn More: For more information about Freescale Semiconductor products, please visit www.freescale.com

AN2748
Rev. 0
08/2004

freescale™ semiconductor