UNIVERSITY OF SOUTHAMPTON

Project Special Brew:
Super Computing on a Budget

by

Steve R. Gunn

Technical Report

Faculty of Engineering and Applied Science
Department of Electronics and Computer Science

Draft v0.5
November 10, 2001
# Contents

1 Introduction 1

2 Hardware 3
   2.1 Processors 3
   2.2 Motherboards 3
   2.3 Network Card 3
   2.4 Memory 4
   2.5 Power Supply 4
   2.6 Cases 4
   2.7 Displays (Optional) 4

3 Software 7
   3.1 MOSIX 7
   3.2 ClusterNFS 8

4 Network Booting 9
   4.1 Etherboot, NetBoot, NILO 9
   4.2 DHCP 9
   4.3 TFTP 9

5 Hardware Configuration 11
   5.1 Hacking the Motherboard BIOS 11
      5.1.1 AMI BIOS 11
      5.1.2 Award BIOS 13
   5.2 BIOS settings 14
   5.3 Alphanumeric LCD 14
   5.4 Case Layout 14
   5.5 Cooling 14

6 Software Configuration 19
   6.1 Linux 19
   6.2 MOSIX 19
   6.3 lm_sensors 19
      6.3.1 mknbi 20
   6.4 Building the Server Kernel 20
   6.5 Building the Client Kernel 21
   6.6 ClusterNFS 21
   6.7 lcd4linux 23
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>Example</td>
<td>25</td>
</tr>
<tr>
<td>7.1</td>
<td>16 + 1 Node Example</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>Performance and Testing</td>
<td>27</td>
</tr>
<tr>
<td>9</td>
<td>Conclusions</td>
<td>29</td>
</tr>
<tr>
<td>9.1</td>
<td>Open Questions</td>
<td>29</td>
</tr>
<tr>
<td>A</td>
<td>PC Chips 810LMR Motherboard</td>
<td>31</td>
</tr>
<tr>
<td>B</td>
<td>MSI 6378 Motherboard</td>
<td>33</td>
</tr>
<tr>
<td>B.1</td>
<td>Etherboot BIOS creation procedure</td>
<td>34</td>
</tr>
<tr>
<td>C</td>
<td>Parallel Port</td>
<td>35</td>
</tr>
<tr>
<td>D</td>
<td>Alphanumeric LCD</td>
<td>37</td>
</tr>
</tbody>
</table>
Listings

6.1 ClusterNFS Client configuration script ........................................... 21
6.2 ClusterNFS Server configuration script ............................................ 22
Chapter 1

Introduction

With the advent of cheaper processors, memory and integrated motherboards, the ability to build a cheap ”super computer” using diskless nodes is becoming increasingly attractive. Add to this the support from the Linux community for managing such systems and the case becomes stronger.

**Cost**  Maximise performance vs. cost.

**Reliability**  Maximise, by reducing the number of mechanical components.

**Maintenance**  Minimise system maintenance by having only one disk.

Such machines are ideal for coarser scale parallel processing. The configuration of the system consists of one disk based server node and many additional diskless processing nodes. The diskless nodes boot their operating system kernel from the server node using **DHCP** and **TFTP**.
Chapter 2

Hardware

2.1 Processors

The choice of cheap processors is limited to the x86 architecture, and hence an AMD vs. Intel decision. Considering that motherboards for both architectures are similarly priced and contain similar features, the decision can be made on processing speed/unit cost alone. It is important to consider the kind of processing that these nodes will perform. In our work the primary purpose is number crunching, and as such floating point (FP) performance is of primary importance. With this in mind the AMD Athlon processor wins easily as of today.

2.2 Motherboards

Since the processor has been decided, a Socket A motherboard is required. Since reliability is one consideration, it would be better and cheaper to obtain an integrated motherboard containing all the required controllers built in. Of primary importance is an integrated LAN. Secondary importance is built in VGA support. Finally, the board should be as small as possible. These requirements are best fulfilled by µATX motherboards, which typically have onboard VGA and some have optional onboard LAN.

2.3 Network Card

Having decided that the LAN should be built in to the motherboard, it is worth considering the form of the network controller. Since this network controller must be able to boot from the network, we require that the network bootrom should support the controller and also that the operating system has the relevant drivers. Additionally if
the controller supports Magic Packet wake up that is an advantage since the node can be booted from the server.

2.4 Memory

Since the nodes will be diskless the memory should be chosen so that programs will fit in to memory. There is an additional possibility whereby swap space can be mounted over the network. However, this will drastically reduce performance. Currently the price of memory suggests that 256MB should be an absolute minimum.

2.5 Power Supply

The Athlon processors are quite power hungry and as such they typically recommend a 300W power supply. However, since no additional hard disks, etc. are connected it should be possible to get by with a 235W power supply. An ATX power supply unit (PSU) has the advantage that it will support wake-up via ethernet provided that the LAN controller and motherboard support this.

2.6 Cases

So if you are going to build a lot of these you don’t want them to be very big. The commercial solution to this problem is to use a 19” rack. However, currently rack cases for ATX motherboards are typically selling for 200 quid upwards. A cheap solution is to buy a bare rack case and fix everything in it yourself. This is reasonably straightforward, since we only have a motherboard and PSU to fix. Rack cases typically come in 1U (44mm), 2U (88mm), 3U (132mm) and 4U (176mm) heights. The athlon processor, heatsink and fan make it extremely difficult to fit this into a 1U case. Additionally low profile PSUs are very expensive. Hence, using a 2U is the best option, since we can just squeeze in a standard ATX power supply (typically 85mm high), and have enough room for processor/heatsink/fan clearance above the motherboard. Combined with the \( \mu \text{ATX} \) form factor motherboard, there is enough room to fit everything in. Bare 2U rack cases can be obtained from Maplin Electronics.

2.7 Displays (Optional)

Whilst the nodes are diskless and all logging is redirected back to the server, it may be advantageous to have a small display to monitor CPU performance, network traffic
for diagnostic purposes. This task is best fulfilled using a cheap Alphanumeric LCD. The cheapest alphanumeric LCDs are typically based around the HD44780 controller and interface directly to the parallel port. They typically come in sizes: $2 \times 16$, $2 \times 20$, $2 \times 40$, $4 \times 20$ and $4 \times 40$, and can be purchased for around £10-£40 depending on size. A good compromise is a $4 \times 20$ display which gives you enough space to display standard information.
Chapter 3

Software

The choice of operating system has to be Linux. It is free and it supports various extensions detailed below which make it well suited to the task. The main task is to make the machine behave like one big computer. Users should only ever need to see the main server node, and the operating system will take care of assigning jobs to make the best use of the available resources.

3.1 MOSIX

MOSIX is a software package that was specifically designed to enhance the Linux kernel with cluster computing capabilities. The core of MOSIX are adaptive (on-line) load-balancing, memory ushering and file I/O optimization algorithms that respond to variations in the use of the cluster resources, e.g., uneven load distribution or excessive disk swapping due to lack of free memory in one of the nodes. In such cases, MOSIX initiates process migration from one node to another, to balance the load, or to move a process to a node that has sufficient free memory or to reduce the number of remote file I/O operations.

MOSIX operates silently and its operations are transparent to the applications. This means that users can execute sequential and parallel applications just like in an SMP. Users need not care about where processes are running, nor be concerned what other users are doing. Shortly after the creation of a new process, MOSIX attempts to assign it to the best available node at that time. MOSIX monitors all the processes, and if necessary migrates processes among the nodes to maximize the overall performance. All this is done without changing the Linux interface. This means that you continue to see (and control) all your processes as if they run on your login node.

The algorithms of MOSIX are decentralized - each node is both a master for processes that were created locally, and a server for (remote) processes, that migrated from other
nodes. This means that nodes can be added or removed from the cluster at any time, with minimal disturbances to the running processes. Another useful property of MOSIX is its monitoring algorithms which detect the speed of each node, its load and free memory, as well as the IPC and I/O rates of each process. This information is used to make near optimal process allocation decisions.

So far MOSIX has been developed for different versions of UNIX and architectures. It has been used as a production system for many years. The first PC version was developed for BSD/OS. The latest version is for Linux on X86/Pentium/AMD platforms.

3.2 ClusterNFS

Since the nodes will need to run slightly different services (typically much less than the server node) it would be nice to modify the files that the nodes see in the cleanest possible way. One way to achieve this is to use ClusterNFS.

ClusterNFS is a set of patches for the Universal NFS Daemon (UNFSD) server to allow multiple diskless clients to NFS mount the same root filesystem by providing interpreted file names. When a client requests the file /path/filename, the ClusterNFS server checks for the existence of files of the form /path/filename$$KEY=value$$. If such a file exists and the clients has a matching value for KEY, this file is returned. If the client does not have the matching value or no such file exists, the file request proceeds as normal. Currently supported keys include, HOST (hostname), IP (IP number), UID (user id), GID (group id), and CLIENT (matches any NFS client).

By naming all machine-specific files filename$$IP=aaa.bbb.ccc.ddd$$ and naming files which are the same for all clients filename$$CLIENT$$, the server and all clients can share the same root partition. This makes it easy to set up and maintain a pool of diskless machines.
Chapter 4

Network Booting

To enable network booting the client must have some network boot code that contains a driver for the ethernet card, and tells it how to download the operating system. The procedure adopted here consists of using a network boot ROM which downloads the operating system.

4.1 Etherboot, NetBoot, NILO

Etherboot is a software package for creating ROM images that can download code over an Ethernet network to be executed on an x86 computer. Many network adapters have a socket where a ROM chip can be installed. Etherboot is code that can be put in such a ROM. Typically motherboards with integrated LAN controllers do not have a ROM socket. However, it is possible to put this code into the motherboard BIOS.

4.2 DHCP

A DHCP server is configured to allocate IP addresses automatically. On startup a client can search for a DHCP server, and this server can allocate an appropriate IP address based upon the network cards MAC address, which is a six byte identifier.

4.3 TFTP

Once an IP address has been negotiated a Linux kernel can be downloaded by the boot ROM.
Chapter 5

Hardware Configuration

The hardware configuration consists of:

**BIOS configuration** The motherboard BIOS has to be modified to include the appropriate *Etherboot* code for the network controller. Additionally, the default BIOS settings can be modified to minimise the setup of multiple clients.

**Case Assembly** Power Supply and motherboard fixing plus optional display mounting.

**Display Connection (Optional)** Wiring up the LCD display.

5.1 Hacking the Motherboard BIOS

Motherboards with integrated LAN typically contain some form of code within the BIOS to boot from the network. However, there are many standards, e.g. RPL (netware), PXE etc., which provide many problems in configuring a Linux server to download the operating system. The simplest approach is to provide an additional floppy with the *Etherboot* code applicable to the network chip, and boot from that. However, this introduces extra cost and reliability problems with floppy disks. To overcome this it is actually quite straightforward to remove the manufacturers network boot code from the motherboard BIOS and replace it with the appropriate *Etherboot* code. Download the appropriate *Etherboot* code for your network controller from [http://rom-o-matic.net](http://rom-o-matic.net).

Determine whether you have an AMI BIOS or an Award BIOS and follow the instructions below:

5.1.1 AMI BIOS

- Get hold of *AMIBCP.EXE*.
Get hold of the latest BIOS for your motherboard.

Load BIOS file into AMI program by typing AMIBCP mybios.bin. You should then see something like Figure 5.1,

Select "Edit BIOS Modules", and you should see a list of all the modules, for an example see Figure 5.2

Find and extract the relevant network ROM module. It will possibly be listed as a "PCI AddOn ROM". If you are unsure, extract the module to a file (by pressing "space"), and load it into a text editor where you should be able to find enough
clues as to its purpose.

- When you are sure that you have found it, remove it from the BIOS by pressing "del".

- Press ”ins” and specify the filename of the Etherboot ROM image that you downloaded.

- Your BIOS is now modified, however you may wish to change some of the default values, e.g. disable IDE disks, etc. You can do this through the ”Configure SETUP Data” on the main menu.

- Save the BIOS file.

5.1.2 Award BIOS

- Get hold of CBROM.EXE.

- Get hold of the latest BIOS for your motherboard

- List the modules in the BIOS by typing CBROM mybios.bin /D. You should then see something like Figure 5.3,

- Find and extract the relevant network ROM module. It will possibly be listed as a ”PCI Driver[?]” or ”Other (xxxx:xxxx)”. If you are unsure, extract the module to a file (by typing CBROM mybios.bin /pci extract), and load it into a text editor where you should be able to find enough clues as to its purpose.

- When you are sure that you have found it, remove it from the BIOS by typing CBROM mybios.bin /pci release.

- To insert the new module type CBROM mybios.bin /other 8000:0 mynetmod.bin where mynetmod.bin the filename of the Etherboot rom image that you downloaded.
• Your BIOS is now modified, however you may wish to change some of the default values, e.g. disable IDE disks, etc. You can do this through the `MODBIN.EXE` program.

Before you upgrade the BIOS it is worth checking that the ethernet controller on the motherboard is configured correctly. To do this get hold of the relevant manufacturer's diagnostic program for your controller chip, boot to a dos prompt (press F5 to bypass start up files), and run the diagnostic program. This program should enable you to specify the size of the network bootrom and was required for the RTL8139 controller on the MSI 6378 board. You can now upgrade the BIOS on your motherboard in the usual way. Should things go horribly wrong, you should find that your bootrom gives you 3 seconds to select from network or local boot. You should then be able to select local and reprogram the BIOS, by booting to DOS from a floppy.

5.2 BIOS settings

Suggested BIOS settings are to disable most things on the motherboard that are not required: sound, gameport, IDE disks, floppy disk.

5.3 Alphanumeric LCD

The description here assumes that a parallel port interface is to be used for the LCD. Appendix C describes the pinout of the parallel connector and Appendix D describes the pinout for a typical $4 \times 20$ display. If you are using lcd4linux, wire up the unit as described in Figure 5.1. LCDproc requires a slightly different wiring for the enable signals; but it should be trivial to modify this in software.

Additionally, the spare input lines on the parallel port can be used to connect some push buttons for controlling the display; some details can be found at Push button wiring. Figure 5.4 shows the LCD mounted in the prototype case.

5.4 Case Layout

Figure 5.5 and Figure 5.6 show the case layout.

5.5 Cooling
## Chapter 5 Hardware Configuration

### Table 5.1: LCD Wiring

<table>
<thead>
<tr>
<th>Display</th>
<th>Parallel</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Pin</td>
<td>Pin</td>
</tr>
<tr>
<td>GND</td>
<td>1</td>
<td>18</td>
</tr>
<tr>
<td>+5V</td>
<td>2</td>
<td>-</td>
</tr>
<tr>
<td>LCD drive</td>
<td>3</td>
<td>-</td>
</tr>
<tr>
<td>RS</td>
<td>4</td>
<td>14</td>
</tr>
<tr>
<td>R/W</td>
<td>5</td>
<td>18</td>
</tr>
<tr>
<td>Enable</td>
<td>6</td>
<td>1</td>
</tr>
<tr>
<td>DB0</td>
<td>7</td>
<td>2</td>
</tr>
<tr>
<td>DB1</td>
<td>8</td>
<td>3</td>
</tr>
<tr>
<td>DB2</td>
<td>9</td>
<td>4</td>
</tr>
<tr>
<td>DB3</td>
<td>10</td>
<td>5</td>
</tr>
<tr>
<td>DB4</td>
<td>11</td>
<td>6</td>
</tr>
<tr>
<td>DB5</td>
<td>12</td>
<td>7</td>
</tr>
<tr>
<td>DB6</td>
<td>13</td>
<td>8</td>
</tr>
<tr>
<td>DB7</td>
<td>14</td>
<td>9</td>
</tr>
<tr>
<td>+5V</td>
<td>15</td>
<td>-</td>
</tr>
<tr>
<td>GND</td>
<td>16</td>
<td>-</td>
</tr>
</tbody>
</table>

**Figure 5.4:** LCD fitted in prototype
Chapter 5 Hardware Configuration

Figure 5.5: Prototype Case Front View

Figure 5.6: Prototype Case Back View
Figure 5.7: Prototype Case
Chapter 6

Software Configuration

6.1 Linux

The following assumes that a RedHat installation will be used. However, the configuration will be similar for other variants of Linux. The following description assumes that you are root.

- Install this on the server.
- Install the source in, e.g. /usr/src/linux-2.4.12
- rm /usr/src/linux
- ln -s /usr/src/linux-2.4.12 /usr/src/linux

6.2 MOSIX

- Install the MOSIX source in e.g. /usr/src/mosix-2.4.5
- Patch the kernel with the MOSIX source, cd /usr/src/mosix-2.4.5; ./mosix.install

6.3 lm_sensors

If you require hardware sensor monitoring do the following:
• Download the latest i2c and lm_sensors source from [http://www.netroedge.com/~lm78/](http://www.netroedge.com/~lm78/).

• Install sources in e.g. /usr/src/i2c-2.6.0, /usr/src/lm_sensors-2.6.0

• Patch the kernel with i2c source, cd /usr/src/i2c-2.6.0; mkpatch/mkpatch.pl . /usr/src/linux | patch -p1 -E -d /usr/src/linux

• Patch the kernel with lm_sensors source, cd /usr/src/lm_sensors-2.6.0; mkpatch/mkpatch.pl . /usr/src/linux | patch -p1 -E -d /usr/src/linux

6.3.1 mknbi

To produce kernel images for downloading with the Etherboot package you need to install and build the mknbi package to create tagged images. mknbi is a program that makes network bootable images for various operating systems suitable for network loading by Etherboot or NetBoot.

• Download the latest source from [http://etherboot.sourceforge.net/distribution.html](http://etherboot.sourceforge.net/distribution.html).

• Install the source in e.g. /usr/src/mknbi-1.2

• cd /usr/src/mknbi-1.2; make

• make install

6.4 Building the Server Kernel

• cd /usr/src/linux

• make xconfig

• Select the following options:

• ...

• make bzImage

• make install

• make modules

• make modules_install
6.5 Building the Client Kernel

- cd /usr/src/linux
- make xconfig
- Select the following options:
  - ...
- make bzImage
- mknbi-linux -ip=rom -rootdir=/ -a="nopnp" /usr/src/linux/arch/i386/boot/bzImage
  > /tftpboot/vmlinuz-2.4.5-mosix.client

6.6 ClusterNFS

- Download the latest source from http://sourceforge.net/projects/clusternfs/
- Install source in /usr/src/ClusterNFS
- make
- make install

```bash
#!/bin/bash
#
# Script to configure ClusterNFS for Redhat
#
# Author: Steve Gunn (srg@ecs.soton.ac.uk)
# Version: 1.0
# Date: 02/06/01
#
# Run this script on the server to configure client files

# ----------------------------------------------
echo Creating /etc/rmtab\$IP=$1\$
# ----------------------------------------------
touch /etc/rmtab\$IP=$1\$
# ----------------------------------------------
echo Creating /etc/mtab\$IP=$1\$
# ----------------------------------------------
touch /etc/mtab\$IP=$1\$
# ----------------------------------------------
echo Creating /etc/sysconfig/hwconf\$IP=$1\$
# ----------------------------------------------
touch /etc/sysconfig/hwconf\$IP=$1\$
# ----------------------------------------------
echo Creating /etc/modules.conf\$IP=$1\$
```
touch /etc/modules.conf

# Creating /var
mkdir /var
mkdir /var/lock
mkdir /var/lock/subsys
mkdir /var/log
mkdir /var/run
mkdir /var/run/netreport
mkdir /var/spool

# Creating /tmp
mkdir /tmp

---

LISTING 6.1: ClusterNFS Client configuration script

#!/bin/bash

# Script to configure ClusterNFS for Redhat
#
# Author: Steve Gunn (srg@ecs.soton.ac.uk)
# Version: 1.0
# Date: 02/06/01
#
# Run this script on the server to configure client files

GATEWAY=192.168.0.254    # Node Gateway
SERVER='/bin/hostname'   # Server Name (Main Node)
CLIENTRUNLEVEL=3

# Creating /etc/exports

# Creating /etc/fstab

# Creating /etc/syslog.conf

# Creating /etc/modules.conf

# Creating /etc/inittab

#
Listing 6.2: ClusterNFS Server configuration script

6.7 lcd4linux

- Download the latest source from http://lcd4linux.sourceforge.net/

http://lcdproc.onnipotent.net/
Chapter 7

Example

<table>
<thead>
<tr>
<th>Description</th>
<th>Part No.</th>
<th>Cost</th>
<th>Supplier</th>
</tr>
</thead>
<tbody>
<tr>
<td>Motherboard</td>
<td>MSI K7T-TURBO-R</td>
<td>£104.00</td>
<td>Stak Trading</td>
</tr>
<tr>
<td>Processor</td>
<td>Athlon 1.33GHz (266FSB)</td>
<td>£78.33</td>
<td>Scan International</td>
</tr>
<tr>
<td>Heatsink/fan</td>
<td>&gt;1.2GHz compatible</td>
<td>£14.50</td>
<td>Scan International</td>
</tr>
<tr>
<td>Memory</td>
<td>2Mb DIMM</td>
<td>£68.00</td>
<td>Scan International</td>
</tr>
<tr>
<td>Power Supply</td>
<td>Enermax 350W ATX</td>
<td>£42.00</td>
<td>Scan International</td>
</tr>
<tr>
<td>CD rom drive</td>
<td>52×</td>
<td>£24.50</td>
<td>Scan International</td>
</tr>
<tr>
<td>Floppy drive</td>
<td>1.44Mb Panasonic</td>
<td>£8.25</td>
<td>Scan International</td>
</tr>
<tr>
<td>Hard Disk</td>
<td>4× 41Gb IBM 60GXP</td>
<td>£336.00</td>
<td>Scan International</td>
</tr>
<tr>
<td>Disk Caddies</td>
<td>4× Removable IDE Rack</td>
<td>£36.59</td>
<td>Scan International</td>
</tr>
<tr>
<td>Network Card</td>
<td>2× 10/100 PCI</td>
<td>£24.00</td>
<td>Scan International</td>
</tr>
<tr>
<td>Case</td>
<td>SuperMicro SC-760</td>
<td>£89.00</td>
<td>Maplin Electronics</td>
</tr>
<tr>
<td>Display</td>
<td>4 × 20 LCD</td>
<td>£25.00</td>
<td>Milford Instruments</td>
</tr>
<tr>
<td>Total (Ex. VAT)</td>
<td></td>
<td>£850.17</td>
<td></td>
</tr>
<tr>
<td>Total (Inc. VAT)</td>
<td></td>
<td>£998.94</td>
<td></td>
</tr>
</tbody>
</table>

Table 7.1: RAID 0+1 Server Node Configuration

<table>
<thead>
<tr>
<th>Description</th>
<th>Part No.</th>
<th>Cost</th>
<th>Supplier</th>
</tr>
</thead>
<tbody>
<tr>
<td>Motherboard</td>
<td>PC Chips M810LMR</td>
<td>£53.00</td>
<td>Stak Trading</td>
</tr>
<tr>
<td>Processor</td>
<td>Athlon 1.33GHz (266FSB)</td>
<td>£78.33</td>
<td>Scan International</td>
</tr>
<tr>
<td>Heatsink/fan</td>
<td>&gt;1.33GHz compatible</td>
<td>£14.50</td>
<td>Scan International</td>
</tr>
<tr>
<td>Memory</td>
<td>512Mb DIMM</td>
<td>£34.00</td>
<td>Scan International</td>
</tr>
<tr>
<td>Power Supply</td>
<td>235W ATX</td>
<td>£12.00</td>
<td>Scan International</td>
</tr>
<tr>
<td>Case</td>
<td>2U bare rackmount</td>
<td>£29.78</td>
<td>Maplin Electronics</td>
</tr>
<tr>
<td>Total (Ex. VAT)</td>
<td></td>
<td>£221.61</td>
<td></td>
</tr>
<tr>
<td>Total (Inc. VAT)</td>
<td></td>
<td>£260.39</td>
<td></td>
</tr>
</tbody>
</table>

Table 7.2: Client Node Configuration

7.1 16 + 1 Node Example

The bottom line, a 20 GHz computer for £5k.
<table>
<thead>
<tr>
<th>Quantity</th>
<th>Description</th>
<th>Cost</th>
<th>Supplier</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Network Switch 16+1 port 10/100 switch</td>
<td>£86.00</td>
<td>Scan International</td>
</tr>
<tr>
<td>1</td>
<td>Server Node, RAID 0+1</td>
<td>£850.17</td>
<td>(See above)</td>
</tr>
<tr>
<td>16</td>
<td>Diskless Nodes</td>
<td>£3545.76</td>
<td>(See above)</td>
</tr>
<tr>
<td></td>
<td>Total (Ex. VAT)</td>
<td>£4888.92</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Total (Inc. VAT)</td>
<td>£5266.26</td>
<td></td>
</tr>
</tbody>
</table>

Table 7.3: Example 16 + 1 Node System Cost

(If you want an LCD display on each node, it will cost an extra £650.)
Chapter 8

Performance and Testing
Chapter 9

Conclusions

9.1 Open Questions

Memory exceeded What happens when a node runs out of memory?

Swap Currently no swap has been implemented, but it may be advisable to mount this over NFS.

Swap Why not just stick in twice as much memory and do away with swap?
Appendix A

PC Chips 810LMR Motherboard

Figure A.1: PC Chips 810LMR Motherboard Layout

Features  Cheap

Problems  Hard Disk controller has poor performance; however, we are not using it.

FSB  266MHz processor compatibility unknown. Conflicting reports.
Appendix B

MSI 6378 Motherboard

Features Magic Packet Wakeup

Problems Memory problems - currently PC133 memory only runs stable at 100MHz. Problem narrowed down to athlon optimisation for kernel, in mmx.c. Set kernel optimisation to K6 to fix this.

FSB 266MHz processor compatibility unknown. Conflicting reports. It works.
B.1 Etherboot BIOS creation procedure

Download latest BIOS file from MSI.

Download the .lzrom file from rom-o-matic.

cbrom bios.rom \ pci release

cbrom bios.rom \ pci .lzrom
# Appendix C

## Parallel Port

![Parallel Port Pinout](image)

### Pin Definition

<table>
<thead>
<tr>
<th>PIN</th>
<th>SIGNAL</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>STROBE</td>
<td>Strobe</td>
</tr>
<tr>
<td>2</td>
<td>DATA0</td>
<td>Data0</td>
</tr>
<tr>
<td>3</td>
<td>DATA1</td>
<td>Data1</td>
</tr>
<tr>
<td>4</td>
<td>DATA2</td>
<td>Data2</td>
</tr>
<tr>
<td>5</td>
<td>DATA3</td>
<td>Data3</td>
</tr>
<tr>
<td>6</td>
<td>DATA4</td>
<td>Data4</td>
</tr>
<tr>
<td>7</td>
<td>DATA5</td>
<td>Data5</td>
</tr>
<tr>
<td>8</td>
<td>DATA6</td>
<td>Data6</td>
</tr>
<tr>
<td>9</td>
<td>DATA7</td>
<td>Data7</td>
</tr>
<tr>
<td>10</td>
<td>ACK#</td>
<td>Acknowledge</td>
</tr>
<tr>
<td>11</td>
<td>BUSY</td>
<td>Busy</td>
</tr>
<tr>
<td>12</td>
<td>PE</td>
<td>Paper End</td>
</tr>
<tr>
<td>13</td>
<td>SELECT</td>
<td>Select</td>
</tr>
<tr>
<td>14</td>
<td>AUTO FEED#</td>
<td>Automatic Feed</td>
</tr>
<tr>
<td>15</td>
<td>ERR#</td>
<td>Error</td>
</tr>
<tr>
<td>16</td>
<td>INIT#</td>
<td>Initialize Printer</td>
</tr>
<tr>
<td>17</td>
<td>SLIN#</td>
<td>Select In</td>
</tr>
<tr>
<td>18</td>
<td>GND</td>
<td>Ground</td>
</tr>
<tr>
<td>19</td>
<td>GND</td>
<td>Ground</td>
</tr>
<tr>
<td>20</td>
<td>GND</td>
<td>Ground</td>
</tr>
<tr>
<td>21</td>
<td>GND</td>
<td>Ground</td>
</tr>
<tr>
<td>22</td>
<td>GND</td>
<td>Ground</td>
</tr>
<tr>
<td>23</td>
<td>GND</td>
<td>Ground</td>
</tr>
<tr>
<td>24</td>
<td>GND</td>
<td>Ground</td>
</tr>
<tr>
<td>25</td>
<td>GND</td>
<td>Ground</td>
</tr>
</tbody>
</table>

*Figure C.1: Parallel Port Pinout*
Appendix D

Alphanumeric LCD

Figure D.1: 4 × 20 LCD Display