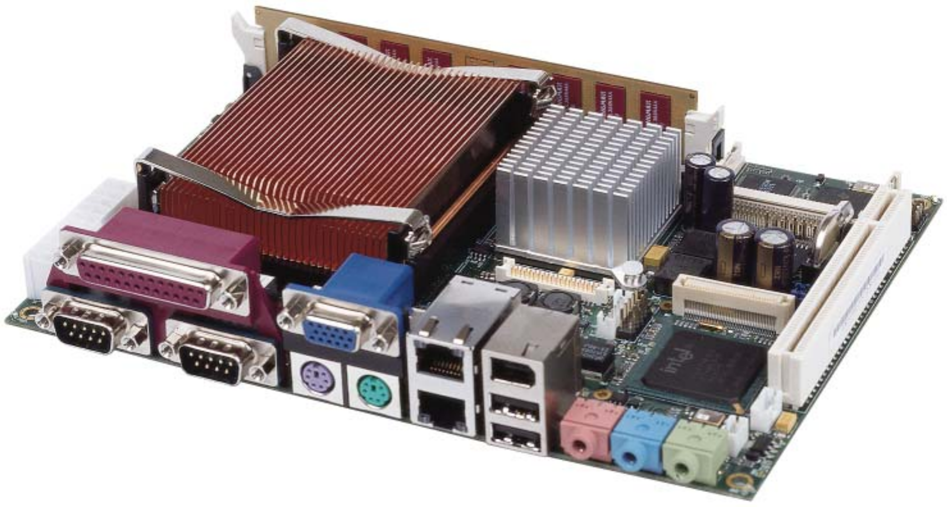
APOLLO

Linux Quickstart Manual



EuroTech Ltd

A MEMBER OF THE EUROTECH GROUP

Warranty

This product is supplied with a 3 year limited warranty. The product warranty covers failure of any Eurotech Ltd manufactured product caused by manufacturing defects. The warranty on all third party manufactured products utilised by Eurotech Ltd is limited to 1 year. Eurotech Ltd will make all reasonable effort to repair the product or replace it with an identical variant. Eurotech Ltd reserves the right to replace the returned product with an alternative variant or an equivalent fit, form and functional product. Delivery charges will apply to all returned products. Please check www.eurotech-ltd.co.uk for information about Product Return Forms.

Revision History

Manual	*PCB*	*Date*	*Comments*
Issue B		15 March 2006	Major updates to unpacking and connecting information.
Issue C		17 August 2006	Minor updates.
Issue D		11 May 2007	Updated for V2 Apollo
Issue E		16th October 2007	Minor updates, Eurotech rebranding.

For contact details, see page 46.

ISO 9001
FM12961

Contents

Introduction

The APOLLO is an EBX format, high-performance, high-functionality PC-compatible processor board designed for embedding into OEM equipment. The board is based on the Intel 855GME/ICH4 chipset and supports a range of Intel Pentium M and Celeron M mobile processors to offer a combination of high performance computing features with low power dissipation.

It offers all standard features and connectors found on a PC motherboard including:

- Multiple video ports.
- Audio.
- Two Ethernet ports.
- CF+ CompactFlash Type II socket.
- Four serial ports, parallel port, IrDA port.
- Primary IDE interface.
- Six USB 2.0 compliant ports.
- Two IEEE1394a-2000 compliant (Firewire) ports.
- General purpose IO and user defined jumpers.

The V2 Apollo includes a Gigabit Ethernet as standard. (This replaces one of the 10/100 Ethernet controllers).

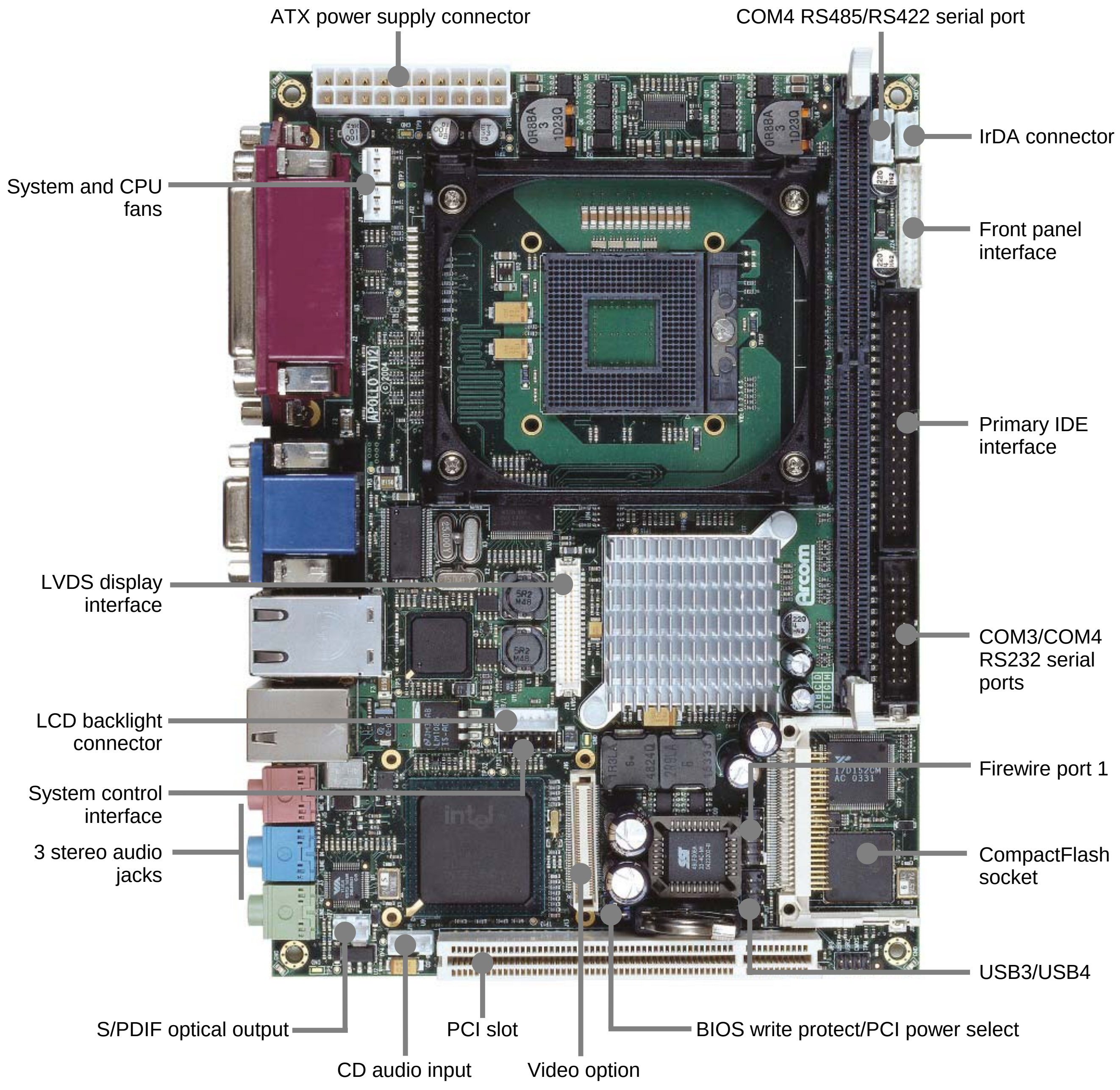
The board is able to run all popular operating systems including Windows XP/XPe and Linux.

Typical applications for the APOLLO include:

- Low power, high density server racks.
- 1U or 2U systems with passive cooling for fan-less operation.
- Systems requiring high levels of hardware/software security.
- Server/client systems using Trusted Computing.
- Compact kiosk systems.

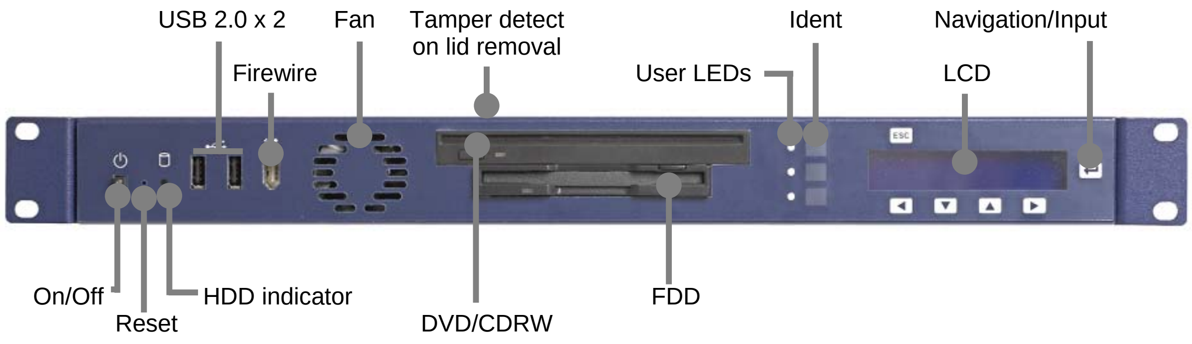
If you are running from a CompactFlash card, please note that the CompactFlash card is intended for application deployment rather than development. As such, a desktop system running the same distribution is required for development.

APOLLO ‘at a glance’

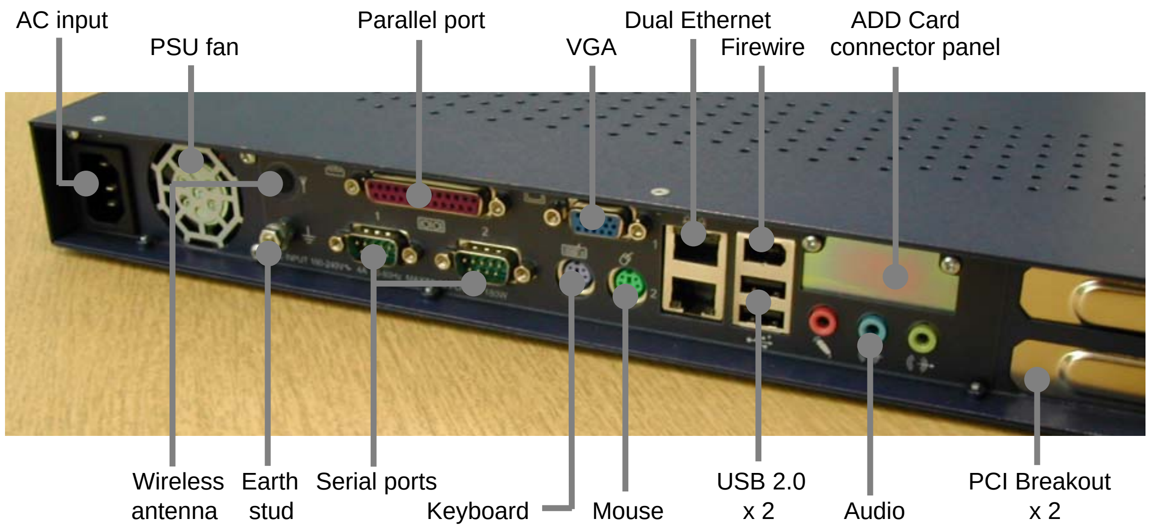


APOLLO 1U ICE 'at a glance'

Front panel view



Rear panel view



Handling your board safely

Anti-static handling

The APOLLO board, and other boards fitted inside an Industrial Compact Enclosure (ICE), contain CMOS devices. These could be damaged in the event of static electricity being discharged through them. Please observe anti-static precautions at all times when handling circuit boards. This includes storing boards in appropriate anti-static packaging and wearing a wrist strap when handling them.

Battery

The board contains a lithium non-rechargeable battery. Do not short circuit the battery or place on a metal surface where the battery terminals could be shorted. During shipment the battery is isolated from the board’s circuitry and should be connected before using the board.

When disposing of the board or battery, take appropriate care. Do not incinerate, crush or otherwise damage the battery.

Packaging

Should a board need to be returned to Eurotech Ltd, please ensure that it is adequately packed, preferably in the original packing material.

Electromagnetic compatibility (EMC)

The APOLLO is classified as a component under the European Community EMC regulations and it is the user’s responsibility to ensure that systems using the board are compliant with the appropriate EMC standards.

About this manual

This manual covers the APOLLO Linux development kit, which provides a complete environment for the development of Linux applications.

This manual does not cover the basics of administering a Linux distribution in general. Please consult your Linux distribution documentation for this information.

Related documents

Further manuals are provided on the CD-ROM that accompanies your development kit, in the /manuals/ folder. These include:

- APOLLO Technical Manual: provides details about the APOLLO hardware.
- APOLLO ICE Technical Manual: covers the enclosure solution for the APOLLO.

The software supplied in the APOLLO development kit was developed and tested using Fedora Core 3 (FC3) and the supplied binaries were compiled in an FC3 environment. Although it should be possible to compile the source code for any common Linux distribution, this has not been tested.

More information about Fedora Core can be found on the Fedora website at fedora.redhat.com.

Conventions

Symbols

The following symbols are used in this guide:

Symbol	Explanation
📌	Note - information that requires your attention.
💡	Tip - a handy hint that may provide a useful alternative or save time.
💣	Caution - proceeding with a course of action may damage your equipment or result in loss of data.

Typographical conventions

This manual contains examples of commands that you can enter. These are shown as follows:

$ **make install DESTDIR=/tmp/target-install**

The initial symbol ($ in this case) indicates the prompt that the command is for, and should not be typed.

The prompts used are explained in the following table:

Prompt	Explanation
$	Linux (bash shell) as a regular user.
#	Linux (bash shell) as root.

Different fonts are used throughout the manual to identify different types of information, as follows:

Font	Explanation
Italics	Parts of a command that should be substituted with appropriate values.
Bold	Information that you enter yourself.
`Screen text`	Information that is displayed on screen.

Getting started

What's in the kit?

APOLLO development kit

- APOLLO board:
  - 1.6GHz Pentium M processor.
  - 512MB RAM.
- 180W Switchmode ATX power supply (100-240V AC input).
- APOLLO ICE breakout board and cable set.
- 1GB CompactFlash pre-loaded with Fedora Core 3.
- Power supply cable with US, UK-, or European-style plug.
- Crossed CAT5e Ethernet cable.
- USB2 CompactFlash Card Reader.
- Null modem cable.
- APOLLO Linux development kit DVD.
- Fedora Core 3 installation DVD.
- Optional flat panel kit including:
  - 15" colour TFT flat panel display.
  - Backlight inverter module.
  - Flat panel cable assembly.
  - Eight-wire resistive touchscreen.
  - Eurotech Ltd Touchscreen Controller (TSC1).

What else do I need?

- Keyboard and mouse (PS/2 or USB).
- Monitor or optional flat panel display.

Unsupported hardware features

The following hardware features are not currently supported:

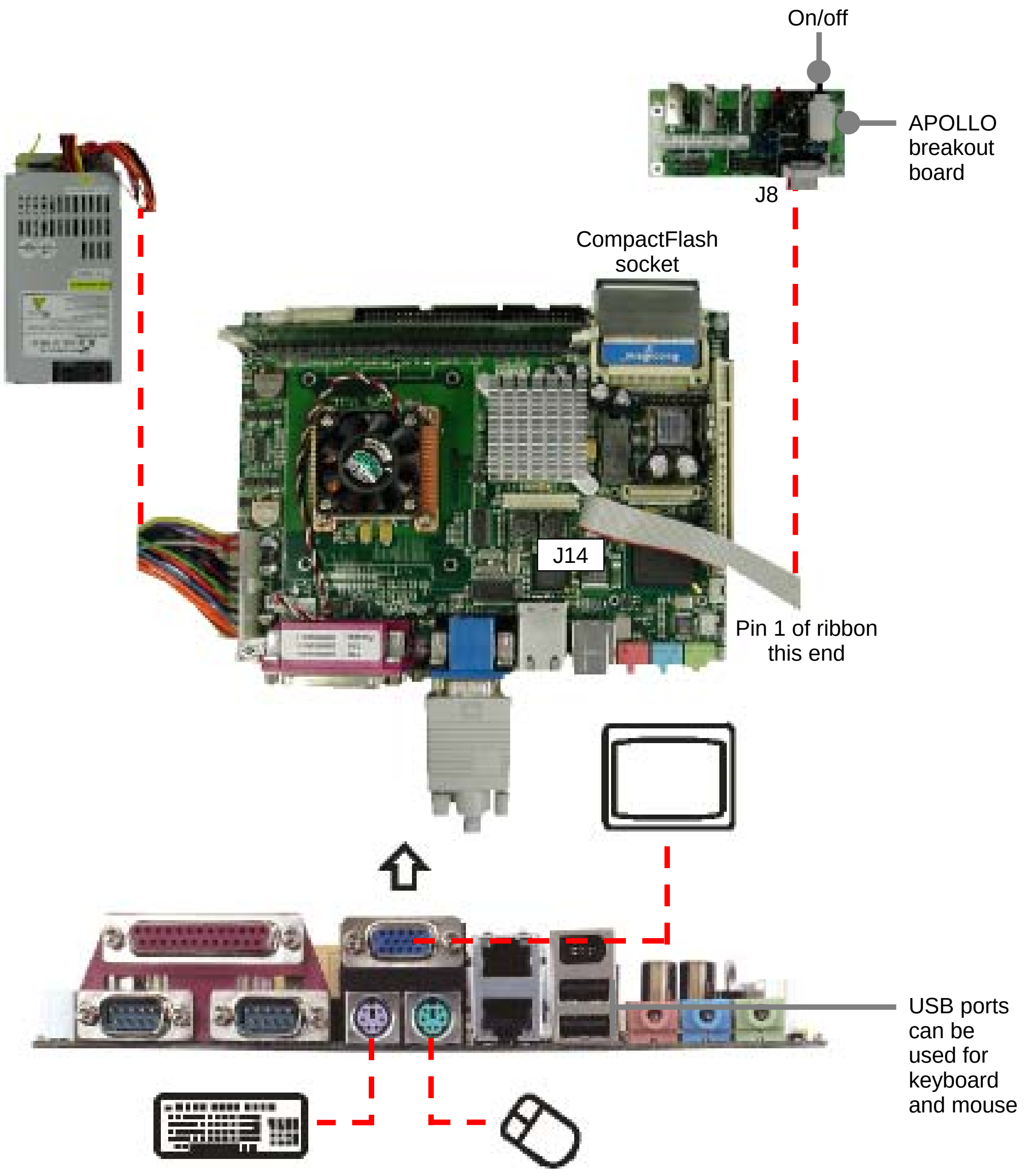
- Suspend and resume.
- TouchScreen Controller (TSC1) board.
- Trusted Platform Module (TPM).

Unpacking and connecting up

Eurotech Ltd’s APOLLO Linux development kit is shipped ‘ready to run’. You simply remove the various items from their packaging and connect them up.

Using a VGA monitor

If you are using a VGA monitor, you can get started quickly and ensure everything is connected properly by referring to the diagram and procedure below.



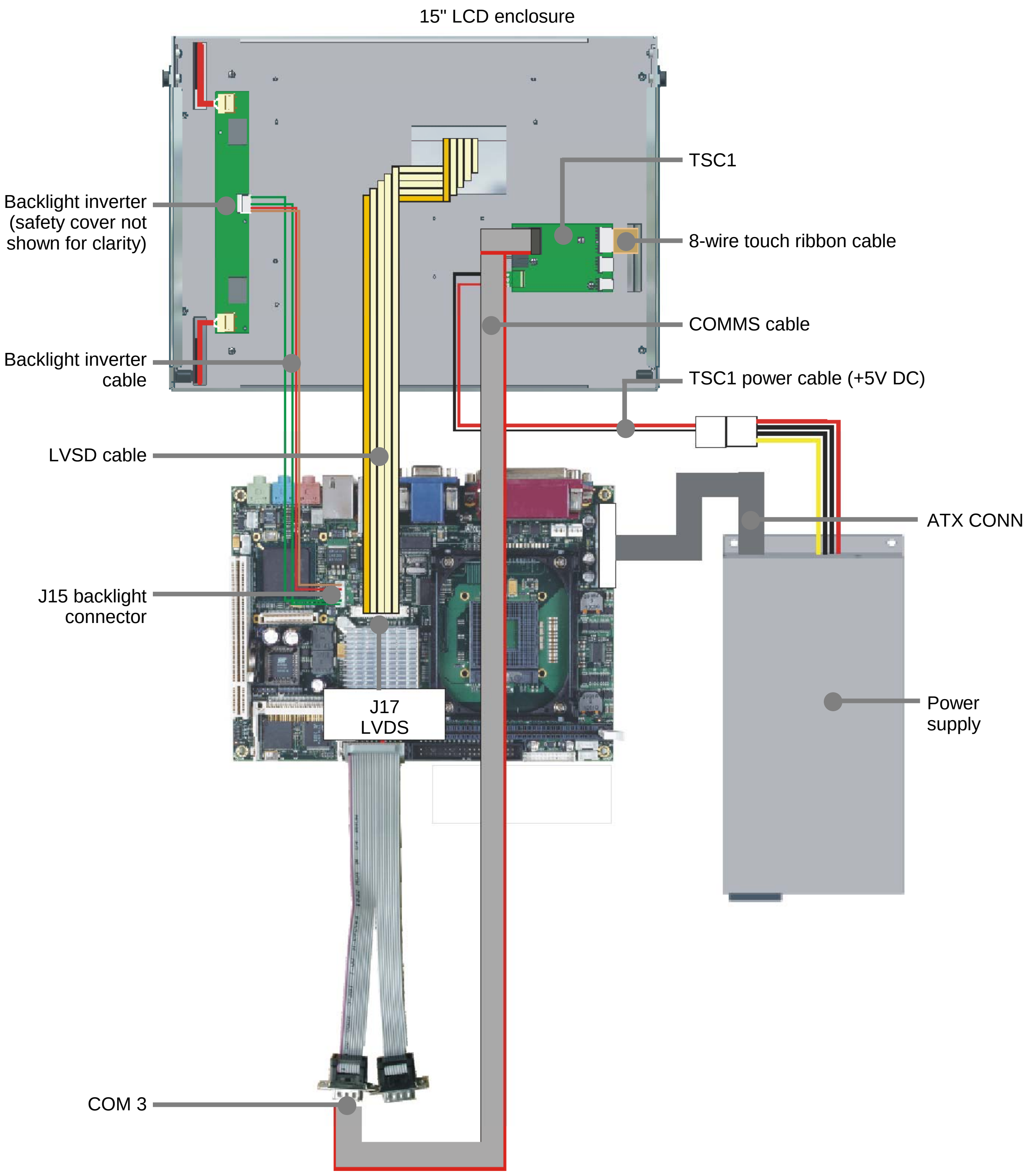
To unpack and connect up the APOLLO if you are using a VGA monitor, follow these steps:

1. Remove the APOLLO CPU board from its packaging and place it on a static-free work surface.
2. Plug a mouse into the green socket or one of the USB headers on the APOLLO.
3. Plug a keyboard into the purple socket or one of the USB headers on the APOLLO.
4. Plug a VGA monitor into the rear socket, as shown in the diagram on the preceding page.
5. Plug the ATX Power Supply Unit (PSU) cable into the socket, as shown in the diagram on the preceding page.
6. Connect up the breakout board, as shown in the diagram on the preceding page.
7. Plug the CompactFlash into the socket on the APOLLO.
8. Fit the appropriate power lead to the PSU.

The power switch is located on the APOLLO breakout board.

Using a flat panel monitor

If you are using a flat panel monitor, you can get started quickly by referring to the following diagram and procedure. For details of connections for the mouse, keyboard and breakout board, see the diagram on page 12.



To unpack and connect up the APOLLO if you are using a flat panel monitor, follow these steps:

1 Plug one end of the flat panel interface cable into the APOLLO flat panel connector J17 LVDS, and the other end into the flat panel.

2 Plug one end of the flat panel backlight cable into the APOLLO backlight connector J15 B/L, and plug the other end into the backlight inverter.

Make sure the power is off while connecting the LCD, as the backlight inverter generates high voltages.

3 If the touchscreen is to be used:

- Use the serial adapter cables provided to connect the 9-way D-type on the TSC1 to COM3 (J22) on the APOLLO.
- Connect the touchscreen to the TSC1 using the 8-way ribbon cable.

  Pin 1 is marked on the touchscreen and the cable, but it can be difficult to see.

- Connect the TSC1 to the power supply.

Logging in

The default root password on systems provided by Eurotech Ltd is ‘arcom1’. For security reasons you should change this as soon as possible. Follow these steps:

1 Log into the board as the root user.

2 Run the passwd utility:

# **passwd**

You are prompted:

```
Changing password for user root.
New UNIX password:
```

3 Enter your password.

Your password is not echoed as you enter it.

You are then prompted to retype it:

```
Retype new UNIX password:
```

4 Re-type your password.

The following prompt is displayed confirming your password change:

```
passwd: all authentication tokens updated successfully.
```

System configuration

The Fedora Core 3 image shipped by Eurotech Ltd is configured with many of the default Fedora Core 3 options, including those for keyboard mapping (US keyboard), time zone (North American EST) and network interfaces (eth0 DHCP, no eth1).

The following sections describe how to reconfigure these settings from a Linux command prompt. If you have a full Fedora Core 3 install including X Windows and the GNOME environment, you will also find several configuration utilities in the **Applications → System Settings** menu.

Configuring keyboard mapping

The default keyboard mapping is for a US keyboard. If you are using a different keyboard layout, you can edit the file /etc/sysconfig/keyboard using the nano command:

```
# nano /etc/sysconfig/keyboard
```

Change the KEYTABLE variable to the correct value for your keyboard.

The available keytables can be found in the /lib/kbd/keymaps folder. Set the KEYTABLE to the filename you require, without the .map.gz suffix.

The correct KEYTABLE for a standard British keyboard is "uk".

When you have finished, press **Ctrl-X** to save and exit.

Configuring time zone

The default time zone is North American Eastern Standard Time (EST), which is five hours behind GMT. The time zone can be configured using the tzselect command:

```
# tzselect

Please identify a location so that time zone rules can be set
correctly.
[...]
```

In the default configuration, Fedora Core expects the hardware clock to be set to GMT, and correctly takes into account the local time zone.

Configuring network interfaces

The default network configuration in the Fedora Core 3 images provided by Eurotech Ltd is to bring up eth0 automatically on boot using DHCP, and to leave eth1 down.

You can modify the network configuration by editing the files /etc/sysconfig/network-scripts/ifcfg-INTERFACE. For example:

```
# name /etc/sysconfig/network-scripts/ifcfg-eth1
```

Fedora Core 3 Linux

Each APOLLO development kit is shipped with Fedora Core 3 pre-installed on the CompactFlash or hard disk.

The sections below explain how to reinstall Fedora Core 3 Linux should this be necessary. There are two options:

- Reinstall from the pre-prepared CompactFlash image. See below.
- Reinstall from the Fedora Core 3 DVD. See Reinstalling from the Fedora Core 3 DVD on page 21.

These instructions apply specifically to the reinstallation of Fedora Core 3 Linux, but the general approach and principles relate to any Linux distribution of your choice.

Reinstalling from the pre-prepared CompactFlash image

The Fedora Core 3 operating system can be reinstalled using the pre-prepared CompactFlash image, *ApolloFC3R2_1GBCF.bin*, supplied on the development kit CD-ROM (in the /software/FedoraCore3/ folder).

You may prefer to reinstall from this image rather than installing Fedora Core 3 yourself. Details of this procedure are provided below.

To write the Fedora Core 3 Linux image to a CompactFlash card using a Linux Host system

1. Remove the card from the APOLLO system and place it into the USB to CompactFlash carrier supplied with the development kit.
2. Insert the carrier into a USB socket on your host system.
3. Use the dd utility to write the image to the device:

   ```
   # dd if=CDROM/Software/FedoraCore3/ApolloFC3R2_1GBCF.bin of=/dev/sda
   ```

   where ***CDROM*** is the CD mount point and **/dev/sda** is the correct device node.

   Once complete, the following messages are displayed on screen:

   ```
   2001888+0 records in
   2001888+0 records out
   ```

   The CompactFlash can now be removed and reinstalled into the APOLLO system.

When using the dd utility, you must specify the correct device node.

Under Linux, USB storage devices traditionally appear as SCSI disks. In a system which has no other SCSI disks this means that the CompactFlash card is available via the /dev/sda device node. If your host system has other SCSI disks then the CompactFlash card will be available via the next available SCSI device.

Alternatively, on newer Linux systems, a new USB Block (ub) device driver has been introduced. On systems which are configured to use this device the CompactFlash card will be available via /dev/uba (assuming it is the only ub device).

To write the Fedora Core 3 Linux image to a CompactFlash card using a Microsoft Windows Host system

1 Remove the card from the APOLLO system and place it into the USB to CompactFlash carrier supplied with the development kit.

2 Insert the carrier into a USB socket on your host system.

3 Open an explorer window and determine the letters of your DVD drive (e.g. D:) and the removable CompactFlash device (e.g. E:).

4 Display a command prompt window by:

   i Clicking on the **Start** menu and selecting **Run...** .

   ii Typing **cmd** and clicking on **OK**.

5 Change to the DVD drive and select the /software/ folder:

```
C:\> d:
D:\> cd software
D:\Software>
```

6 Run the cfclone utility, passing the name of the compressed CompactFlash image and the CompactFlash drive letter:

```
D:\Software> cfclone /r FedoraCore3\ApolloFC3R2_1GBCF.bin e:
```

CFCLONE is a utility developed by Eurotech Ltd that is available on the development kit CD-ROM. It can be used to write an image to a CompactFlash card under Windows 2000, Windows XP and Windows 2003 Server.

Reinstalling from the Fedora Core 3 DVD

The Fedora Core 3 operating system can also be installed to either hard disk or a CompactFlash card from the DVD supplied in the development kit. After doing this, some configuration is required to return the board to a state similar to when it was shipped by Eurotech Ltd.

If you are using a CompactFlash-based system then you also have the option to install from the supplied CompactFlash image as described in the previous section.

The Fedora Core Installation Guide was pending at the time of writing this manual. However it may now be available at fedora.redhat.com.

BIOS configuration

Before you begin reinstalling the Fedora Core 3 operating system you must make a change to the BIOS configuration. To do this, follow these steps:

1. On bootup, press **F2** to access the BIOS configuration screens.
2. Use the ← and → arrow keys to select the **boot** configuration screen.
3. Use the ↑ and ↓ arrow keys and the **+** and **-** keys to move the *CD-ROM Drive* entry above both *Hard Drive* and *Removable Devices*.

   If you are installing to a CompactFlash card, Eurotech Ltd recommends that you place the card under the *Hard Drive* entry. You can expand an entry marked with a plus sign by selecting it and pressing the **Enter** key. If the CompactFlash card appears under *Removable Devices*, select it and press the **N** key to move it to *Hard Drive*.
4. Exit the BIOS, saving your changes. To do this:
   - i Use the ← and → arrow keys to display the **Exit** configuration screen.
   - ii Use the ↑ and ↓ arrow keys to highlight **Exit Saving Changes**.
   - iii Press **Enter**.

Installing to hard disk

To install to hard disk, follow these steps:

1. Insert the Fedora Core installation DVD in to the DVD drive and boot the APOLLO board. You are presented with the Fedora installation disc boot loader.
2. Press the **Enter** key to start the installer.
3. Follow the on-screen prompts to configure your installation.

No APOLLO specific configuration is required during installation to support Fedora Core 3 installed to a hard disc on an APOLLO board. See Post installation configuration on page 23 for details of additional configuration required after installation.

Installing to CompactFlash

Installing Fedora Core to the APOLLO CompactFlash requires some additional, but straightforward, configuration. This is required because the CompactFlash card is visible to the operating system via both the IDE controller and via the CardBus controller. Linux begins to boot using the CompactFlash via the IDE controller. However once the CardBus controller has been found the IDE device disappears, causing the boot process to fail.

As an alternative to reinstalling from the Fedora Core 3 DVD, you can reinstall from the pre-prepared Fedora Core 3 CompactFlash image. See Fedora Core 3 Linux on pages 19 and 20 for details.

The following instructions are specific to Fedora Core 3, but the general procedure is similar for any Linux distribution.

Follow these steps:

1 Insert the Fedora Core installation DVD into the DVD drive and boot the APOLLO board. The Fedora Core installation disc boot loader message is displayed:

```
linux nopcmcia
```

2 Press **Enter** to start the installer.

The nopcmcia option prevents the Fedora Core installer from initialising PCMCIA (CardBus).

3 Follow the on-screen prompts to configure your installation. During this installation we recommend that you:

- Select the Custom installation type. This enables you to choose a minimal set of packages, or choose packages individually, later in the installation process. It is important that you only install the packages required due to space limitations on the 1G CompactFlash card.

- Select the Manual Partition with Disk Druid option, and fill the entire CompactFlash card with a single large ext2 partition. We recommend ext2 in preference to ext3 because the journaling activity of the ext3 file system dramatically increases the number of writes performed, reducing the lifespan of the CompactFlash device. For this reason we also recommend that no swap partition is created.

DO NOT reboot yet. It is essential that you complete step 4 before rebooting. Failure to do so will result in the installation failing.

4 Once the installation is complete, you are given the opportunity to reboot. At this point you must disable PCMCIA in the installed system. To do this:

i Press **Ctrl**-**Alt**-**F2** to switch to a command line console.

You may find the console is garbled by the transition from graphical to text mode. If so, use the **clear** command to reset it.

ii Change your current root to the newly installed system:

```
# chroot /mnt/sysimage
```

iii Blacklist the yenta_socket module to prevent CardBus from being initialised on boot:

```
# echo “yenta_socket” >> /etc/hotplug/blacklist
```

iv Disable the PCMCIA subsystem by editing the file /etc/sysconfig/pcmcia:

```
# nano /etc/sysconfig/pcmcia
```

Change **`PCMCIA=yes`** to read **`PCMCIA=no`**, and then press **Ctrl**-**X** to save the file and exit the editor.

v Press **Alt**-**F7** to return to the installer.

vi Select the option to reboot.

Post installation configuration

Following installation some additional configuration is required. If you are using the supplied CompactFlash, this configuration has been done for you.

The first stage is to register the Fedora Project cryptographic keys with the RPM system. This allows RPM to verify that packages from Fedora have not been tampered with. If you do not follow this step, rpm and yum may complain about missing keys.

```
# rpm
--import /usr/share/doc/fedora-release-3/RPM-GPG*
```

Secondly, there are some additional packages which are required to support some of the special features of the APOLLO hardware.

The additional packages are on the development kit CD-ROM in the /software/FedoraCore3 folder. You can install them all using the command:

```
# rpm -Uvh CDROM/software/FedoraCore3/*.rpm
```

where ***CDROM*** is the CD mount point.

You may decide to choose which packages to install, rather than installing all packages. Information about the packages you can choose from is provided in the sections that follow.

The supplied modules binary package is built against the Fedora Core 3 2.6.9-1.667 kernel. If you have a different kernel (e.g. from a Fedora Core update) then you will need to rebuild the modules. See Kernel modules on page 26 for details.

The zoneinfo data has been updated in this image to support US Daylight Saving Time changes that were introduced for 2007 and beyond. The CD contains the update file *AEL_USDST2007.tar.gz* that is used to test and fix timezone data. Copy this file to the /tmp directory on the Apollo system and proceed as follows:

```
# cd /tmp
# tar -xvzf  AEL_USDST2007.tar.gz
# ./fix_DST
```

Once you have updated the system you can test the timezone data using:

```
# ./test_DST
```

You can remove the above scripts and **.tar.gz* files once you have completed the install.

Refer to the Eurotech Ltd website for more information on DST changes and to acquire the latest version of the DST update file at www.eurotech-ltd.co.uk

If you are not using Fedora Core 3 then the sections that follow may still be useful as they outline the general procedure for rebuilding the packages for other distributions.

Additional software available in Fedora Core 3

The pre-prepared Fedora Core 3 image supplied by Eurotech Ltd contains a default selection of Fedora Core 3 packages. However, a large selection of software that is not installed is present on the Fedora DVD, in the /Fedora/RPMS/ folder.

You can query a package using the RPM tool as follows:

# **rpm -qip *PACKAGE***

where ***PACKAGE*** is the RPM package file you are interested in. This gives you a description of the package.

Once you have located a package that you want, you can install it using the following command:

# **rpm -ivh *PACKAGE***

When you do this, RPM may inform you that the package has dependencies which are not satisfied. If this is the case, then identify the required package and append the filename to the RPM command line. The Fedora Core DVD should contain a closed set of packages and so all dependencies should be available on the disk.

If you have a network connection then you may like to take advantage of the yum command. This performs automatic dependency resolution, downloads software from the Fedora Core servers and installs it for you:

# **yum install *PACKAGE_NAME***

where ***PACKAGE_NAME*** is the name of the package you want to install.

You can search for packages by keywords using yum's search facility:

# **yum search *KEYWORDS***

More information about both rpm and yum can be found in their respective man pages.

Kernel modules

If you are using the pre-installed Fedora Core 3 image from the development kit CD-ROM, the following drivers are preinstalled for the default 2.6.9-1.667 kernel:

- **apollo_wdt.ko** – kernel driver for the Maxim MAX6369 watchdog on the APOLLO board.
- **ds2401.ko** – kernel driver for the ds2401 unique identifier chip on the APOLLO board.
- **apollo**_fpi.ko – kernel driver for the APOLLO Front Panel Interface.
- apollo_fpi_**fan.ko** - kernel driver for the v1i4 APOLLO Front Panel Interface which supports a MAX6550 fan controller.

These kernel drivers were developed against the 2.6.9-1.667 kernel contained in Fedora Core 3, but they should be compatible with most 2.6 kernels.

The driver source is held in the /software/ folder on the development kit CD-ROM. The tarball is named apollo-kernel-v1i2.tar.gz. It contains a README file that provides additional detail on driver usage.

If you reinstall Fedora Core 3 to either hard disk or a CompactFlash card from the DVD supplied in the development kit, you will need to reinstall the kernel modules from the Eurotech Ltd DVD. Binaries are provided for the default Fedora Core 3 kernel (version 2.6.9-1.667).

For other kernels and/or distributions, you must build the modules yourself. The following sections explain how to do this, both using RPM packages and manually on non-RPM based systems.

Building and installing APOLLO kernel modules on RPM based systems

The first stage is to build the package. If you are using the default 2.6.9-1.667 kernel, enter:

```
# rpm -ivh CDROM/software/FedoraCore3/apollo-modules-2.6.9-1.667-v1i2-1.i386.rpm
  # depmod -a
```

where **CDROM** is the CD-ROM mount point and ***/software/FedoraCore3*** is the folder on the development kit CD-ROM where the binary RPM built against kernel version 2.6.9-1.667 can be found.

Alternatively, if you are not using this kernel, the source tarball contains the necessary instructions to build an RPM for any kernel. Run:

```
# rpmbuild -tb CDROM/software/apollo-kernel-v1i2.tar.gz
```

This command assumes that you wish to build the modules for the kernel currently running, and that the kernel headers can be found in the folder '/lib/modules/*KERN_VER/*build/', where KERN_VER is the version of the kernel currently running.

If this is not the case you can pass in the *KERN_VER* that you wish to build against. For example, to build against 2.6.11-1.14_FC3 you would enter:

# **rpmbuild -tb --define 'kvers 2.6.11-1.14_FC3'** ***CDROM/*****software/apollo-kernel-v1i2.tar.gz**

where ***CDROM*** is the CD mount point.

This looks in '/lib/modules/2.6.11-1.14_FC3/build' and builds against that kernel. If this is still not correct then kdir can be defined in addition to kvers. (Ensure that the version you give matches the kernel source being used as no checks are made at this stage.) For example:

# **rpmbuild -tb --define 'kvers** ***[...]*****' --define 'kdir /path/to/kernel'** ***CDROM/*****software/apollo-kernel-v1i2.tar.gz**

Enter this command on a single line, with a space between 'kdir /path/to/kernel' and *CDROM.*

Once the package has been built it can be found in /usr/src/redhat/RPMS/i386. You can then install it by entering:

# **rpm -ivh /usr/src/redhat/RPMS/i386/apollo-modules-*****VERSION*****.rpm**
# **depmod -a**

/usr/src/redhat is the default under RedHat and Fedora systems, but may differ on other distributions. For example, SuSE use /usr/src/suse.

Building and installing APOLLO kernel modules on non-RPM based systems

Follow these steps:

1 Unpack the apollo-kernel source tarball by entering:

# **tar xvzf** ***CDROM/*****software/apollo-kernel-v1i2.tar.gz**

where ***CDROM*** is the CD mount point.

The contents of the tarball are unpacked to a new folder called 'apollo-kernel'.

2 Change to the new folder and build the kernel modules:

```
# cd apollo-kernel
# make
```

This assumes that you wish to build the modules for the kernel currently running, and that the kernel headers can be found in the following folder:

/lib/modules/*KERN_VER*/build/

where *KERN_VER* is the version of the kernel currently running.

If this is not the case, set the KDIR variable to point to a suitable set of kernel headers:

```
# make KDIR=/pathtokernelheaders
```

3 Install the modules, including KDIR as necessary:

```
# make [KDIR=/pathtokernelheaders] modules_install
# depmod -a
```

Loading Kernel modules

Kernel modules are loaded manually using the modprobe command, for example:

```
# modprobe evdev
```

On Fedora Core these commands can be added to the /etc/rc.sysinit file to be executed upon boot.

Utilities

The development kit CD-ROM contains a precompiled RPM package that provides the following APOLLO utilities and examples:

- **apollo-fpi-keys** – an example demonstrating how to read keypresses from the APOLLO front panel interface.
- **ich4regs** – a utility for viewing and manipulating the registers in the ICH4 chipset.
- **apollo-userlink** – an example demonstrating how to read the userlinks.
- **10-apollo.rules** – sample rules for use with a /dev/ directory dynamically managed with udev.
- **apollo-fpi-gpio-test**– an example demonstrating how to read the GPIO pins GPIO1 & GPIO2 on the APOLLO FPI header.
- **apollo-fpi-fan-test**– an example demonstrating how to read some fan information when apollo-fpi-fan.ko is loaded.
- **apollo-hwmon-test** - an example demonstrating how to read on-board Apollo hardware monitor information such as CPU & SYSTEM fans and temperatures.

This package can be found in the /software/FedoraCore3/ folder. It is installed using the following command:

```
# rpm -ivh CDROM/software/FedoraCore3/apollo-utils-v1i2-1.i386.rpm
```

where ***CDROM*** is the CD mount point.

The precompiled RPM may also be suitable for other systems. If it is not and changes are required, you can rebuild the RPM for your system. If you need to do this, the apollo-utils source is provided on the development kit CD-ROM in the /software/ folder. The tarball is named *apollo-utils-v1i2.tar.gz*. It contains a simple README that describes the function of the utilities provided.

To build the APOLLO utilities you must have a standard Linux development environment including gcc and libc, with headers etc. as well as the pciutils library. You may need to install the development packages for these libraries. On Fedora Core this can be done using the yum command:

```
# yum install glibc-devel pciutils-devel
```

(The command and package name may differ on other distributions.)

The sections on the next page explain how to build and install the APOLLO utilities, using both RPM packages and manually on non-RPM based systems.

Building and installing the APOLLO utilities on RPM-based systems

To build and install the APOLLO utilities on an RPM-based system:

1 Build the utilities by entering the command:

# **rpmbuild -tb *CDROM*/software/apollo-utils-v1i2.tar.gz**

where ***CDROM*** is the CD mount point.

2 Install the utilities:

# **rpm -ivh /usr/src/redhat/RPMS/i386/apollo-utils-v1i2-1.i386.rpm**

/usr/src/redhat is the default under RedHat and Fedora systems, but may differ on other distributions. For example, SuSE use /usr/src/suse.

Building and installing the APOLLO utilities on non-RPM systems

If you are not using an RPM-based system you can build the APOLLO utilities by hand for your distribution. To do this:

1 Unpack the source tarball by entering:

# **tar -xvzf *CDROM*/software/apollo-utils-v1i2.tar.gz**
# **cd apollo-utils-v1i2**

where ***CDROM*** is the CD mount point.

2 Configure the utilities as required using the following command:

# **./configure**

3 Build and install the utilities:

# **make**
# **make install**

The utilities are now installed in the /usr/local/bin/ folder.

Unique ID

The APOLLO unique ID is provided via the ds2401 kernel module which is built as described in the section Kernel modules, page 26. It is loaded using the following command:

```
# modprobe ds2401
```

Once the module has been loaded, the unique ID can be read from the special file /proc/driver/id as a string containing a 12 digit hexadecimal number. For example:

```
# cat /proc/driver/id
0123456789ab
```

Ethernet controllers

Under Fedora Core, the secondary Ethernet controller (100baseT from the southbridge) is detected first and hence is assigned the name eth0.

The primary Ethernet controller (Gigabit controller) is therefore eth1.

Watchdog

The APOLLO watchdog driver is provided by the kernel module apollo_wdt which is built as described in the section Kernel modules on page 26. It is loaded using the following command:

```
# modprobe apollo_wdt
```

Once the driver has been loaded the watchdog is available via the /dev/watchdog device node.

On a system such as Fedora Core 3 which uses udev to manage device nodes dynamically, /dev/watchdog is created automatically. On other systems it may be necessary to create the correct device node as a character device with major number 10 and minor number 130. To do this, enter:

```
# mknod /dev/watchdog c 10 130
```

Once the device node has been created the watchdog is activated by opening the device and kicked by writing a single character to the device. Closing the device however does not stop the watchdog. To stop the watchdog a single character 'V' must be written before closing the device.

Take care not to write any additional characters after the 'V', such as a carriage return or line-feed, as these will re-arm the watchdog. In particular, if you are using echo to trigger the watchdog from a script, pass the -n switch to prevent a carriage return from being added.

The watchdog timeout can be set by passing the heartbeat parameter when loading the module:

```
# modprobe apollo_wdt heartbeat=10
```

The heartbeat is given as a number of seconds, valid values are 1, 10 and 60 seconds with 60 being the default.

Front Panel Interface (APOLLO-FPI)

The APOLLO Front Panel Interface (FPI) is part of the APOLLO 1U Industrial Compact Enclosure (ICE). It contains three status LEDs, an LCD controller and several input buttons.

Kernel module

The APOLLO-FPI kernel driver is responsible for driving the FPI keypad and the three FPI status LEDs. The character display itself is controlled using the LCDproc daemon (see LCD controller on page 39 for details).

The driver is built as described in the section Kernel modules on page 26. An i2C bus driver i2c-1801 is also required. This module is provided by the standard Fedora kernel packages. The modules are loaded using the following commands:

```
# modprobe i2c-i801
# modprobe apollo_fpi
```

Status LEDs

Once the apollo_fpi kernel module has been loaded the status LEDs can be controlled by reading and writing the following special files:

```
/sys/bus/i2c/drivers/apollo_fpi/0-0020/led1
/sys/bus/i2c/drivers/apollo_fpi/0-0020/led2
/sys/bus/i2c/drivers/apollo_fpi/0-0020/led3
```

Writing a value of 1 turns on the LED, while writing a value of 0 turns it off.

Status GPIO

The Apollo-FPI hardware v1i4 has 3 additional inputs that can be monitored. PS_A & PS_B are GPIO inputs on a 3-way header (J4). There is also an additional input called ALERT_FAN that is connected to the alarm output on the fan controller. If the fan controller triggers an alarm, the status of ALERT_FAN input will alter.

PS_A, PS_B & ALERT_FAN can be read at

```
/sys/bus/i2c/drivers/apollo_fpi/0-0020/ps_a
/sys/bus/i2c/drivers/apollo_fpi/0-0020/ps_b
/sys/bus/i2c/drivers/apollo_fpi/0-0020/alert_fan
```

Keypad

The apollo_fpi kernel module presents the keypad as a standard kernel input device via a device special file named event*N* in the /dev/input/ directory.

The /dev/input/event interface requires that the kernel is built with support for the 'evdev' interface (CONFIG_INPUT_EVDEV). This is normally the case for kernels provided by Linux distributions. If evdev support is built as a module it must be loaded using modprobe evdev.

Under a system such as Fedora Core 3 that uses udev to dynamically manage /dev/, the device node is created automatically. If the apollo-utils package is installed, then a file named /etc/udev/rules.d/10-apollo.rules is present with the following contents:

```
KERNEL="event*", SYSFS{name}="apollo_fpi", NAME="input/%k",
SYMLINK="input/apollo_fpi"
```

This file creates a symbolic link to the input event device created by the APOLLO-FPI driver (see below), making it easier for applications to locate the correct device. If you have already loaded the driver you must unload and reload for the link to be created. To do this, enter:

```
# modprobe -r apollo_fpi
# modprobe apollo_fpi
```

On systems not using udev, you may need to create a series of character device nodes with major number 13 and minor numbers starting at 64:

```
# mkdir /dev/input
# mknod /dev/input/event0 c 13 64
# mknod /dev/input/event1 c 13 65
# mknod /dev/input/event2 c 13 66
# mknod /dev/input/event3 c 13 67
```

One method of determining the correct device is to examine the files /sys/class/input/event*N*/device/name until one containing 'apollo_fpi' is found. Alternatively, the ioctl EVIOCGNAME can be issued on each device node in turn until one named 'apollo_fpi' is located.

Once the correct device has been found and opened it can be read using either blocking or non-blocking reads, or polled using select. A very simple example (apollo-fpi-keys) for reading events from the device is provided in the apollo-utils package on the development kit CD-ROM.

Each read should expect a whole number of input events. Where each input event consists of a struct input_event data structure. The structure is described in the header file linux/input.h, often found in /usr/include/linux/input.h.

```
struct input_event {
 struct timeval time;
 __u16 type;
 __u16 code;
 __s32 value;
};
```

The meanings of the fields in struct input_event are explained in the following table:

Field name	Description
struct timeval time	Timestamp containing the time of the event.
type	The type of event. This is always EV_KEY for apollo_fpi events.
code	For events of type EV_KEY this field contains the keycode as described in the table below.
value	For events of type EV_KEY, this field contains 0 for a key release, 1 for a key press and 2 for auto-repeat.

The apollo_fpi driver generates the following keycodes:

Keycode name	Keypad key
KEY_ESC	**Esc**
KEY_LEFT	←
KEY_DOWN	↓
KEY_UP	↑
KEY_RIGHT	→
KEY_ENTER	**Enter**

By default, the apollo_fpi driver acts like a normal keyboard and therefore key presses are passed to the console as well as your application. If you wish to direct keypad events to your application only, issue the EVIOCGRAB ioctl on the device with an argument of 1:

ioctl(fd,EVIOCGRAB,1);

where **fd** is an open file descriptor to the device.

The grab can be released by issuing the ioctl again with an argument of 0.

LCD backlight

Once the apollo_fpi kernel module has been loaded, the LCD backlight can be controlled by reading and writing the following special file:

/sys/bus/i2c/drivers/apollo_fpi/0-0020/backlight

Writing a 0 to this file turns the backlight off. Writing a 1 turns it on.

Apollo FPI fan control kernel module

The APOLLO-FPI (v1i4) has additional hardware support to control 3 fans via a MAX6550 fan controller. The kernel driver *apollo_fpi_fan.ko* provides the interface to this hardware.

The driver is built as described in the section Kernel modules on page 26. An i2C bus driver i2c-1801 is also required. This module is provided by the standard Fedora kernel packages. The modules are loaded using the following commands:

```
# modprobe i2c-i801
# modprobe apollo_fpi_fan
```

The APOLLO-FPI supports up to 3 fans though only 1 (a master fan) can be directly controlled. The 2 additional (slave) fans follow the changes applied to the master fan.

For this reason it is advised that any additional fans that are used be identical to the master fan, *which must be plugged into J6.* All fans can be controlled by reading and writing to the master fan at:

```
/sys/bus/i2c/drivers/apollo_fpi_fan/0-0048/percent
```

Where 'percent' is a percentage of maximum fanspeed ranging from 1% to 100%.

Most fans cannot be slowed much below ~20%. In this case, a read will report the attained fanspeed in percent.

```
# cat /sys/bus/i2c/drivers/apollo_fpi_fan/0-0048/percent
```

In addition to writing & reading percentage fanspeed, other characteristics can be observed and/or altered. Some are read/write as designated by [R/W], others are read-only [RO]

Fan0 speed in RPM [R/W]:	/sys/bus/i2c/drivers/apollo_fpi_fan/0-0048/fanspeed
MAX RPM of fan0 [RO]:	/sys/bus/i2c/drivers/apollo_fpi_fan/0-0048/fanmax
fan0 in RPM [RO]:	/sys/bus/i2c/drivers/apollo_fpi_fan/0-0048/rpmfan0
fan1 in RPM [RO]:	/sys/bus/i2c/drivers/apollo_fpi_fan/0-0048/rpmfan1
fan2 in RPM [RO]:	/sys/bus/i2c/drivers/apollo_fpi_fan/0-0048/rpmfan2
Alarm [R/W]:	/sys/bus/i2c/drivers/apollo_fpi_fan/0-0048/alarm

Both *fanspeed* & *rpmfan0* refer to the MASTER fan. *rpmfan0* reports live RPM stats whereas *fanspeed* will show the requested setting. *fanmax* reports the detected maximum speed of the master fan in RPM.

The alarm attribute has a two-fold function. In read-mode, the alarm *status* is read. When no alarms are active, this will report 0. At initialisation, the driver enables the following alarms:

Alarm Name	Bit weighting
Tach Overflow	4
Minimum Output level overflow	2
Maximum Output level overflow	1

e.g. If a read of the alarm status returns '7', then all 3 alarms have tripped.

To clear an alarm, write 0.

```
# echo "0" > /sys/bus/i2c/drivers/apollo_fpi_fan/0-0048/alarm
```

An alarm value of "0" just clears the alarms and restores the driver defaults. It is possible to set up a user specified alarm configuration. i.e. If we only want to generate an alarm for MAX overflow, then write "1". It is also possible to set up a configuration with no alarm trips by writing "8".

Any time a value >0 is written to alarm, the driver initiates a fan re-calibration. This is in case the initial fan calibration produced an erroneous 'fanmax'. An invalid fanmax could skew subsequent fan settings which could lead to alarm trips. Under normal circumstances, it should not be necessary to recalibrate the master fan or alter the alarm settings.

An example demonstrating reading fan RPM & fan PERCENT can be found in the file *apollo-utils-v1i2.tar.gz*.

The Apollo FPI v1i4 hardware supports 5 and 12 volt fans. There is a hardware link, JP1, on the Apollo FPI that needs to be set to the correct fan voltage. The default is to support 12 volt fans.

Use the correct voltage for the fans fitted in your system. All fans must run at the same voltage.

If you use 5 volt fans you must also set the fan voltage at the point of driver load.

```
# modprobe apollo_fpi_fan voltage=5
```

The fan voltage setting that the driver is expecting, can be read back using

```
# cat /sys/module/apollo_fpi_fan/voltage
```

The fan voltage setting cannot be altered after driver load.

LCD controller

The APOLLO-FPI LCD controller is compatible with the LCDproc[1] daemon. A patch is provided that adds support for the APOLLO-FPI hardware interface.

The patch does not add support for the APOLLO-FPI keypad to LCDproc. For details of accessing the keypad, see Keypad on page 35.

A precompiled RPM package for Fedora Core 3 is provided on the development kit CD-ROM, in the /software/ folder. It can be installed using the following command:

```
# rpm -ivh CDROM/software/FedoraCore3/lcdproc-0.4.5-2.i386.rpm
```

where ***CDROM*** is the CD mount point.

The precompiled RPM may also be suitable for other systems. If it is not and changes are required, you can rebuild the RPM for your system. If you need to do this, the patch is provided on the development kit CD-ROM in the /software/ folder.

To build LCDproc you must have a standard Linux development environment including gcc, libc and ncurses, with headers etc. You may need to install the development packages for these libraries. On Fedora Core this can be done using the yum command:

```
# yum install glibc-devel ncurses-devel
```

(On other distributions the command and package name may differ.)

The following sections explain how to build and install LCDproc, using both RPM packages and manually on non-RPM based systems.

[1] lcdproc.omnipotent.net/

Building and installing LCDproc on RPM-based systems

To build and install LCDproc on an RPM-based system, follow these steps:

1 Copy the sources to the RPM build area. On Red Hat Linux and Fedora systems this is the /usr/src/redhat folder. Enter the following commands:

```
# cp CDROM/software/LCDd.conf /usr/src/redhat/SOURCES
# cp CDROM/software/lcdproc-0.4.5-apolloV2-1.diff /usr/src/redhat/SOURCES
# cp CDROM/software/lcdproc-0.4.5.tar.gz /usr/src/redhat/SOURCES
# cp CDROM/software/lcdproc.spec /usr/src/redhat/SPECS
```

where ***CDROM*** is the CD mount point.

On other systems the folder may be named differently. For example on a SuSE system you might need to copy the files to the folders beneath /usr/src/suse/ instead.

2 Build the RPM package:

```
# cd /usr/src/redhat/SPECS
# rpmbuild -ba lcdproc.spec
```

3 Install the package:

```
# rpm -ivh /usr/src/redhat/RPMS/i386/lcdproc-0.4.5-2.i386.rpm
```

Building and installing LCDproc on non-RPM systems

If you are not using an RPM-based system then you may choose to build LCDproc by hand for your distribution. To do this:

1 Unpack the upstream source tarball:

```
# tar -xvzf CDROM/software/lcdproc-0.4.5.tar.gz
# cd lcdproc-0.4.5
```

where ***CDROM*** is the CD mount point.

2 Apply the patch for APOLLO-FPI support:

```
# patch -i CDROM/software/lcdproc-0.4.5-apolloV2-1.diff -p1
```

3 Configure LCDproc. The hd44780 driver supports the APOLLO-FPI LCD controller. The curses driver can be useful for testing and development; it can optionally be omitted.

```
# ./configure –enable-drivers=hd44780,curses
```

4 Build and install the daemon

```
# make
# make install
```

LCDproc is now installed into the /usr/local/ directory. The LCDd daemon is installed into /usr/local/sbin. The lcdproc utility is installed in /usr/local/bin.

5 Arrange for the LCDd daemon to be started at boot time, either by using one of the example scripts (available in the ‘scripts’ subdirectory of the lcdproc source tree, unpacked in step 1) or by some other distribution specific method. An example /etc/LCDd.conf configuration file is provided in *CDROM*/software/LCDd.conf.

Using LCDproc

The LCDproc package provides a daemon (LCDd) which controls the LCD controller. Client applications can connect to this daemon and control the display.

The package also contains a client named lcdproc which displays information about various aspects of the system on the LCD panel, including memory usage, load average etc. The lcdproc(1) manual page describes the various options. This is installed along with LCDproc and is also available in the source tree which can be unpacked as described in the previous section.

In addition the source contains a number of examples in the clients folder and documentation in the docs folder. The LCDproc website at lcdproc.omnipotent.net/ contains further documentation as well as links to other clients.

PCI bus device descriptions

Fedora Core 3 ships with a utility named lspci (part of the pciutils package) which can be used to list the devices present on the PCI bus. The database of devices shipped with Fedora Core 3 does not cover all of the devices present on the APOLLO board. This makes no functional difference to the operation of the system but it can be useful to have a complete view of the system's PCI devices when administering the system. The device database can be updated using the following commands:

```
# wget http://pciids.sourceforge.net/pci.ids.bz2
# bunzip2 pci.ids.bz2
# mv pci.ids /usr/share/hwdata/pci.ids
```

You are prompted:

```
mv: overwrite '/usr/share/hwdata/pci.ids'?
```

Enter **y** to confirm the overwrite.

These instructions also work on other distributions, although the location of the pci.ids file may differ.

User jumpers

The user jumpers are read via GPIO lines 1 and 2 on the Firmware Hub (FWH).

The state of the firmware hub GPIO lines can be read directly from memory address 0xffbc0100 via the /dev/mem special device.

The apollo-userlinks utility in the apollo-utils package is an example of a program that does this.

CPU frequency scaling

The standard Fedora Core 3 installation includes the cpuspeed daemon (provided by the kernel-utils packages). This daemon dynamically alters the processor's operating frequency based on the current workload.

X Window System

Fedora Core 3 makes use of the X.org project to provide X Windows support. In many configurations the correct packages are installed during installation. If they are not you will need to install the relevant packages yourself, including xorg-x11 and any application packages. For example to install xorg-x11 you would use the yum command:

```
# yum install xorg-x11
```

Once the X Window System has been installed it must be configured by editing the file /etc/X11/xorg.conf. You may find the system-config-display utility useful for this. An example xorg.conf is supplied on the development kit CD-ROM in the folder /software/.

The current drivers for the i852 dual pipeline graphics chipset only support multiple cloned displays (i.e. all outputs displaying the same screen). True multihead support (i.e. side-by-side or independent display) is not yet available.

The driver defaults to using whichever display configuration has been setup in the BIOS. You can, however, override this using the MonitorLayout option in the ‘Device’ section of xorg.conf. This option takes as a parameter a string ‘<PIPEA>,<PIPEB>’ where each pipe can be assigned to an output device such as CRT, LFP etc. A pipe can be configured for multiple outputs using the + operator, e.g. CRT+LFP. At this time only pipe B can be active so pipe A should be given as NONE. Further information about this option can be found in the i810(4x) manual.

Appendix A – Contacting Eurotech Ltd

Eurotech Ltd sales

Eurotech Ltd's sales team is always available to assist you in choosing the board that best meets your requirements.

Eurotech Ltd
3 Clifton Court
Cambridge
CB1 7BN
UK

Tel:	+44 (0)1223 403410
Fax:	+44 (0)1223 410457
Email:	sales@eurotech-ltd.co.uk

Comprehensive information about our products is also available at our web site: www.eurotech-ltd.co.uk.

While Eurotech Ltd's sales team can assist you in making your decision, the final choice of boards or systems is solely and wholly the responsibility of the buyer. Eurotech Ltd's entire liability in respect of the boards or systems is as set out in Eurotech Ltd's standard terms and conditions of sale. If you intend to write your own low level software, you can start with the source code on the disk supplied. This is example code only to illustrate use on Eurotech Ltd's products. It has not been commercially tested. No warranty is made in respect of this code and Eurotech Ltd shall incur no liability whatsoever or howsoever arising from any use made of the code.

Eurotech Ltd technical support

Eurotech Ltd has a team of dedicated technical support engineers available to provide a quick response to your technical queries.

Tel:	+44 (0)1223 412428
Fax:	+44 (0)1223 410457
Email:	support@eurotech-ltd.co.uk

Eurotech Ltd Group

Eurotech Ltd is a subsidiary of Eurotech Group. For further details see www.eurotech.com

Index