Marvell® ARMADA™ 16x Plug Computer Development Kit

User Manual

Doc. No. MV-S400320-00, Rev. -

December 2010

Document Conventions

	**Note:** Provides related information or information of special importance.
	**Caution:** Indicates potential damage to hardware or software, or loss of data.
	**Warning:** Indicates a risk of personal injury.

Document Status

Doc Status: Preliminary	Technical Publication: x.xx

For more information, visit our website at: www.marvell.com

Table of Contents

List of Figures

1 Introduction

1.1 GuruPlug Display (gplugD) Overview

The Marvell® ARMADA™ 16x Plug Computer Development Kit User Manual describes procedures for operating the GuruPlug[1] Display Development Kit, referred to throughout this document as the "gplugD." The gplugD device is an evaluation platform for the Marvell® ARMADA™ 168 Applications Processor. It takes the shape of a plug computer, a small form-factor computing platform created by Marvell that integrates a high-performance, power-efficient system-on-a-chip processor with network connectivity, peripheral expansion, and the Linux™ operating system. The entire system is housed in an AC-to-DC adapter that plugs into a wall outlet, thus the term "plug computer."

The gplugD expands upon existing plug computers by incorporating a display controller with an HDMI output for a LCD TV or monitor, extra USB ports for connecting a keyboard and mouse, and extra flash memory storage via internal and external microSDHC card slots. Quite simply, the gplugD is a complete computer that is smaller than a traditional desktop PC power supply.

1.2 gplugD Functionality

Unlike other plug computers, which are limited to running applications that either are non-interactive or deliver content via terminal console or web browser, the gplugD can drive a HDMI-capable display with both bit-mapped graphics and sound. This makes it an ideal platform for developing energy-efficient interactive kiosks and smart displays.

Without any optimization, the gplugD running MPlayer on Linux and X11 can render 30 fps H.264-encoded CIF (352 x 240) video with a 48 kHz AAC LC stereo audio track. While higher performance is possible using Marvell's Integrated Performance Primitives-based (IPP) multimedia codecs, the gplugD is not intended to be a high-performance video playback device for home theatre applications.

The gplugD runs Debian Linux. It can access the complete range of Debian repositories with more than 18,000 precompiled packages for armel (ARM EABI) targets. Add in projects from other Open Source collaborative efforts and end user-developed software, and the gplugD becomes a flexible and capable device for a wide range of applications.

1.3 Hardware Overview

Though about the same size as a typical notebook computer AC-to-DC power supply, the gplugD is equipped with a full set of memories and peripherals:

- Marvell ARMADA 168 processor (88AP168) running at 800 MHz
- 512 Mbytes DDR2-800 SDRAM
- 4-Mbit SPI boot flash holding the pre-boot loader and U-Boot
- Two Class 6 microSDHC cards
  - 4 GB in external slot with Debian Linux 5.0
  - 8 GB on CPU and memory board
- 4 high-speed USB ports
- 10/100 Mbps Ethernet
- HDMI 1.3 video and stereo audio output

1. GuruPlug is a registered trademark of Globalscale Technologies, Inc.

- High-speed USB On-the-Go (OTG) micro AB port
- USB mini-B connector for serial port console access

1.4 Marvell® ARMADA™ 168 Features

The gplugD is powered by the high-performance, low-power ARMADA 168 Applications Processor that integrates a comprehensive set of peripherals for smart, interactive products. Flexible hardware interfaces support a wide range of memory and I/O standards, and permit system designers to fine-tune individual subsystems for cost and performance. Features of the ARMADA 168 include:

- Marvell® Sheeva™ PJ1 ARMv5TE-compatible CPU core capable of operation at up to 1 GHz
  - 32-Kbyte direct-mapped L1 instruction cache
  - 32-Kbyte 4-way set associative L1 data cache
  - 128-Kbyte unified L2 cache
  - Intel® Wireless MMX2™ compliant SIMD coprocessor
- 32-channel DMA Controller with 92 peripheral device request sources
- NAND Flash Controller for 1.8V and 3V SLC and MLC devices
- 2-chip select static memory and VLIO Bus Controller
- LPDDR1/DDR2/LVDDR2/DDR3 Dynamic Memory Controller
- Flexible LCD Controller with hardware color conversion capable of driving up to WUXGA (1920 x 1200) class smart or dumb panels
- Programmable 2D graphics accelerator with intelligent hardware color management unit
- Up to 4 (e)MMC/SD/SDIO/CE-ATA Controller ports
- XD, MS/MS Pro, and CompactFlash Card Controllers
- CMOS video (up to 640 x 480) and still (up to 2 Mpixel) Camera Interface Controller
- High-speed USB OTG and Host Controllers with integrated PHYs
- 10/100 Mbps fast Ethernet Controller
- Dual x1 lane PCI Express Controller and PHY
- Up to 3 UARTs and 2 two-wire serial interfaces
- Up to 5 synchronous serial ports with I^2S and SPI modes
- AC '97 Audio Controller
- Up to 4 independent PWM channels
- Three 32-bit hardware timers
- Up to 122 GPIO pins

1.5 Operating System Support

The gplugD ships with Debian Linux 5.0 which is based on the Linux 2.6.29 kernel. While not yet released, the next version of Debian Linux presumably will be version 6.0. A kernel and root file system for that can be installed on the gplugD and will be available for download at some point in the future. The gplugD also supports Android, with a port of version 2.2 currently in development.

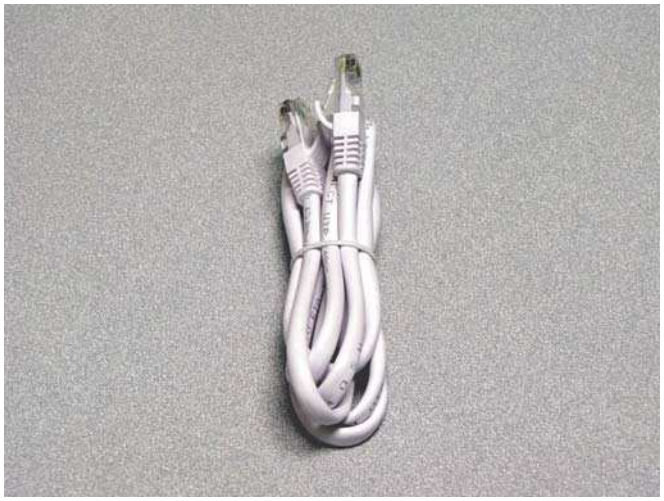
2 Unpacking the gplugD

2.1 Kit Contents

The GuruPlug Display (gplugD) Development Kit comes in a compact white box from Marvell partner Globalscale Technologies, Inc.

A full complement of accessories comes with the kit, including all of the cables needed to connect to the various I/O ports, such as HDMI…

Ethernet...

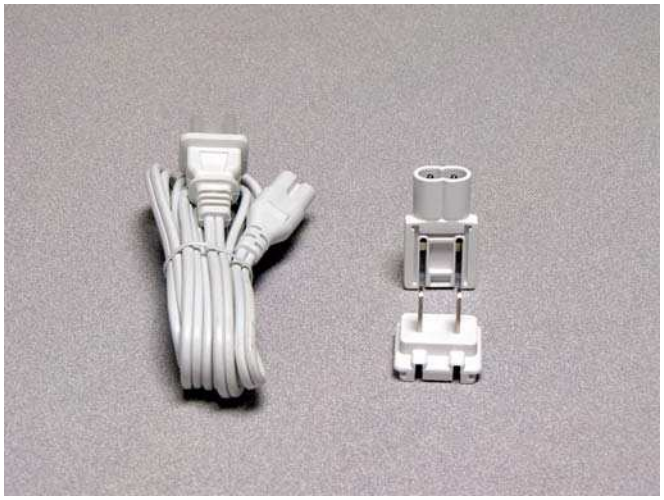


A bendable USB extension gooseneck for a Wi-Fi or Bluetooth dongle…

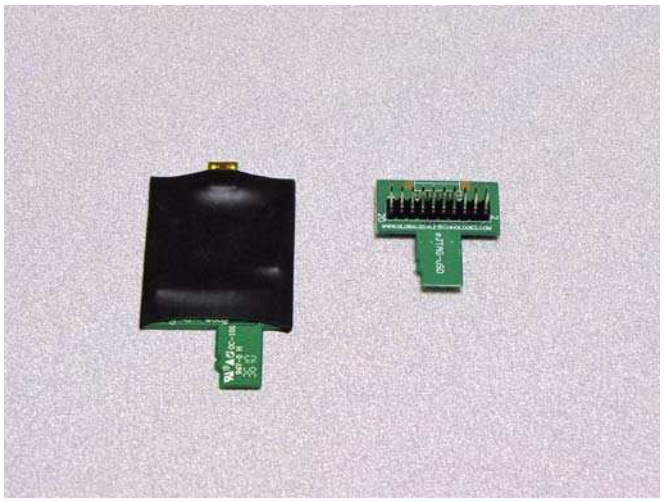
A pair for USB OTG (micro AB plug to standard A receptacle and micro AB plug to standard A plug)…

And a pair of USB mini B (one for the console serial port and the other for the JTAG debug adapter).

Just like other plug computers, the gplugD can be plugged directly into a wall outlet, operating discretely and out of the way. The desktop adapter and extension cord keep the gplugD within easy reach of a developer's workstation and nearby peripherals.

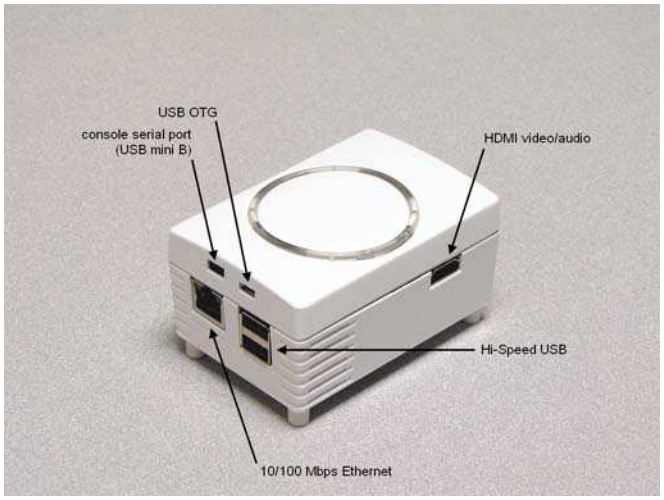


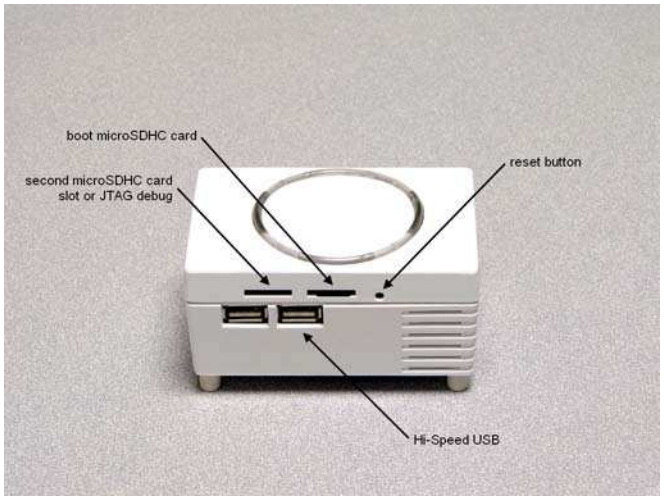
Debug access using JTAG is provided through the gplugD second microSDHC card slot using one of two enclosed adapters. The first is for use with existing JTAG hardware and is a passive circuit board that simply maps the appropriate signals to the standard 20-pin debug header. The second is a miniature USB debug module that is based on the Future Technology Devices International FT2232H multi-purpose high-speed USB converter chip. Software support for this hardware is in development.



2.2 External Connectors

I/O connectors occupy three of the four sides that are perpendicular to the gplugD internal printed circuit-board assemblies as shown in the following figures:





While not immediately obvious from these photographs, the printed circuit boards inside the gplugD are mounted in such a way that the HDMI, high-speed USB, and microSDHC connectors are inverted. Cables must be plugged in accordingly, and microSDHC cards must be inserted with contacts up.

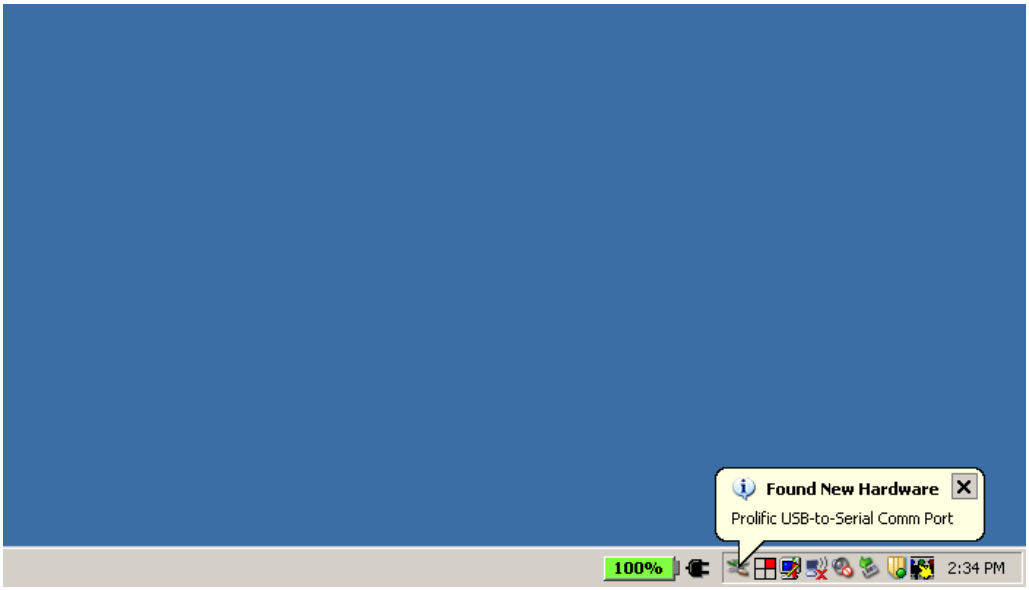
3 Getting Started with the gplugD

3.1 Console Access

Embedded Linux systems typically provide console access for interacting with a boot loader, observing start-up messages, editing configuration files, and issuing commands from a shell. A Prolific Technology PL-2303 USB-to-serial Bridge Controller is used for the console serial port on the gplugD. The following instructions assume a PC running Windows XP with appropriate user rights and a terminal program such as a recent version of TeraTerm.

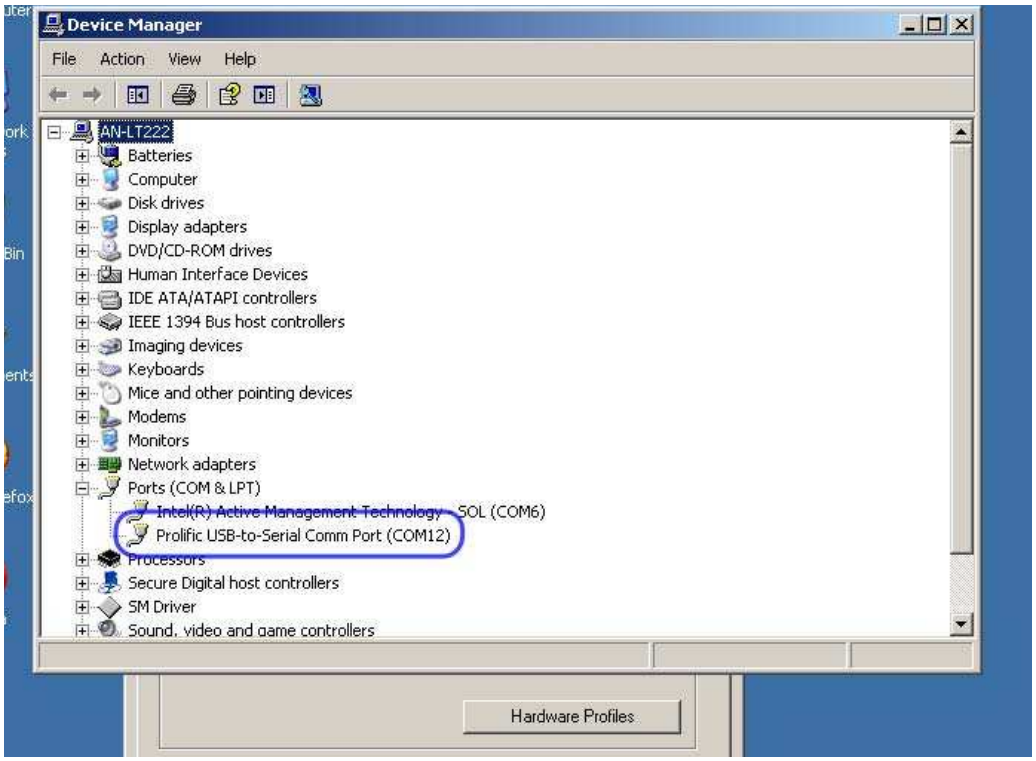
1. Visit http://www.prolific.com.tw to download the PL-2303 drivers for the, then run the installer.
2. Connect an available PC USB port to the gplugD serial port (the USB mini B receptacle above the Ethernet RJ-45 jack) using one of the enclosed USB mini B cables.
3. Power-up the plug using the desktop adapter and extension cord.
4. Wait for Windows to enumerate the PL-2303 device and load its driver as shown in Figure 1.

Figure 1: PL-2303 USB Enumeration



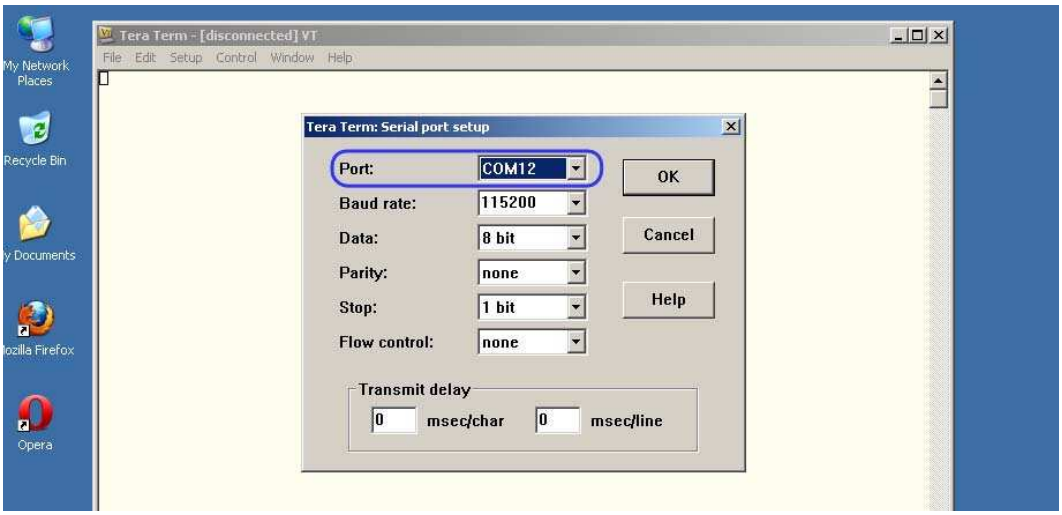
5. Go to the Device Manager (Start Menu --> Settings --> Control Panel --> System --> Hardware --> Device Manager). Expand the Ports (COM and LPT) drop-down and note the COM port assigned to the PL-2303 (see Figure 2).

Figure 2: COM Port Assignment



6. Launch TeraTerm, and select “Serial port…” from the **Setup** menu. Choose the COM port assigned to the Prolific driver in the Device Manager from the “Port” drop-down menu (Figure 3).

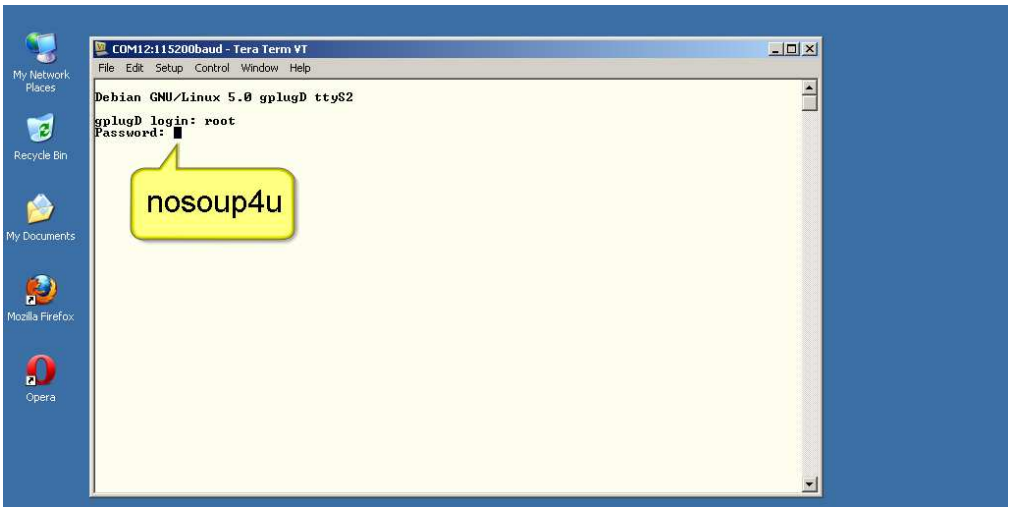
Figure 3: TeraTerm Serial Port Setup



If necessary, change the baud rate to 115200, data to 8 bit, and parity to none. Click “OK” to apply the changes.

After the serial port is configured, start-up messages scroll by as Linux boots until the login prompt is reached (see Figure 4).

Figure 4: gplugD Login Prompt

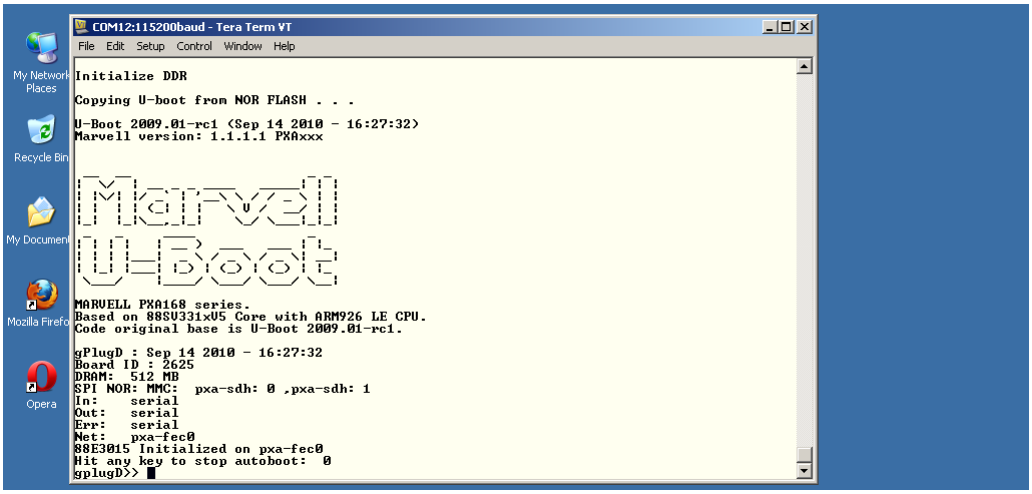


Login as “root” with “nosoup4u” as the password. From this default shell, typical command line programs are available. For example, dpkg –l | more lists the installed software packages.

U-Boot provides a short delay before loading the Linux kernel and launching the operating system. From the shell, enter “reboot” to shutdown Linux and reset the chip.

7. Press any key to halt OS loading when the U-Boot start-up message appears (Figure 5).

Figure 5: U-Boot Start-Up



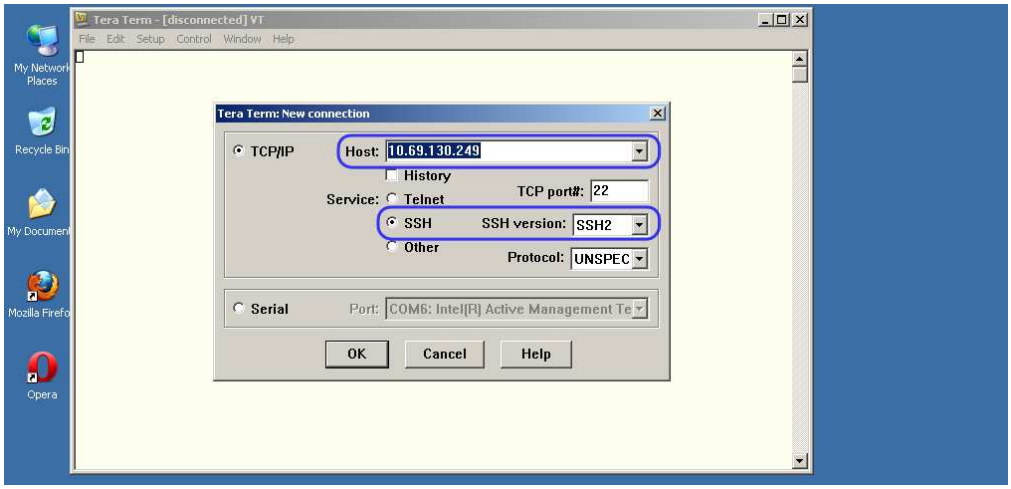
Enter “help” to get a list of U-Boot commands. Enter “boot” to load the kernel and start Linux.

3.2 Remote Access with Secure Shell (SSH)

Secure Shell (SSH) can be used to access the gplugD when it is connected to a network. The steps that follow assume either (a) that the Plug has been reconfigured to use a static IP address or (b) that its dynamically assigned IP address is known. Any client that supports Version 2 of the SSH protocol can be used, which makes TeraTerm a particularly convenient choice for this example.

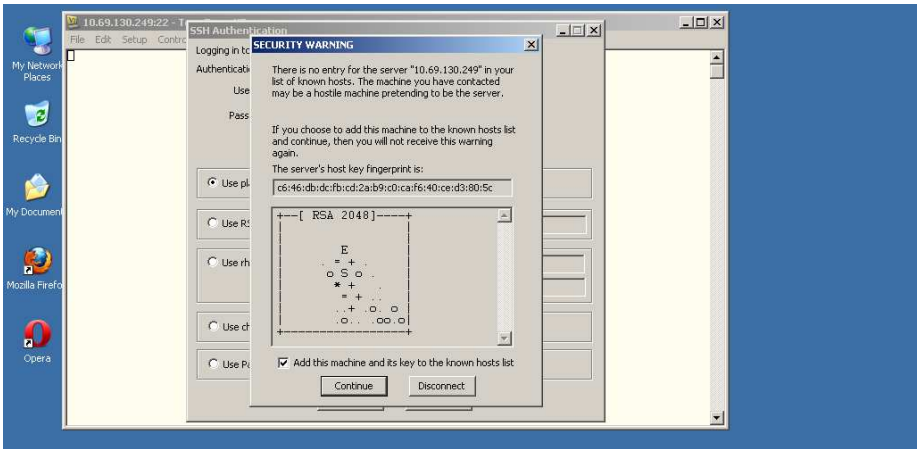
1. Select “New connection…” from the **File** menu, enter the Plug’s IP address, select the SSH radio button, chose “SSH2” from the SSH version drop-down menu, and click “OK” (see Figure 6).

Figure 6: TeraTerm SSH Setup



2. When connecting to any machine not explicitly listed in its “hosts” file, the SSH client displays a warning message as shown in Figure 7:

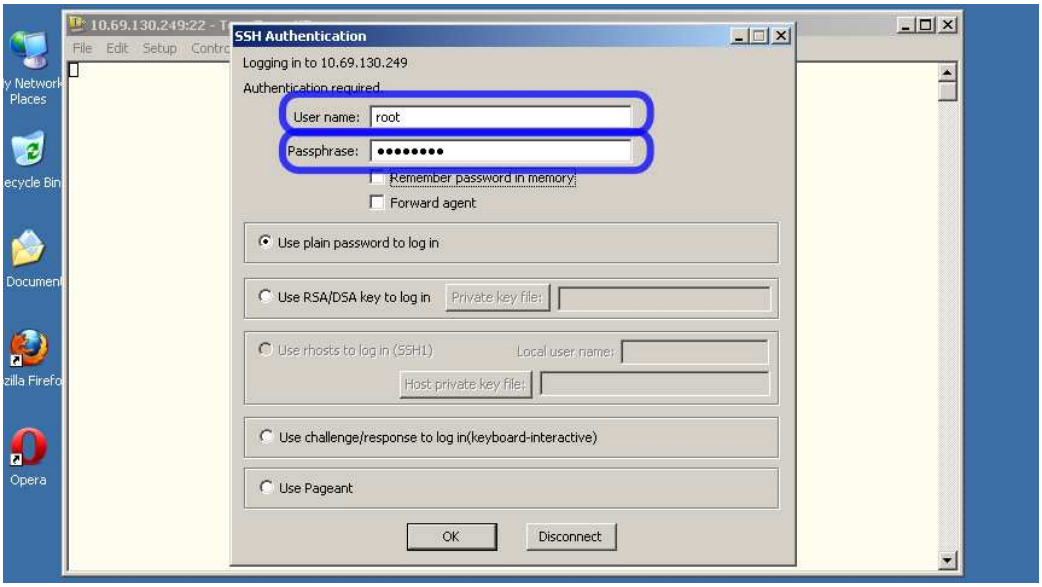
Figure 7: SSH Unrecognized Host Warning Message



Allow TeraTerm to add the gplugD IP address to its list of known hosts, if preferred. This warning message is displayed any time TeraTerm connects to a machine that is not listed (either via IP address or fully-qualified domain name) in its host file. Click “Continue.”

3. Enter “root” and “nosoup4u” for the user name and password, respectively, at the next prompt, and click “OK” (see Figure 8).

Figure 8: SSH Authentication



4. At the shell prompt, type “exit” or “logout” to disconnect.

3.3 Using a Keyboard, Mouse, and HDMI Display

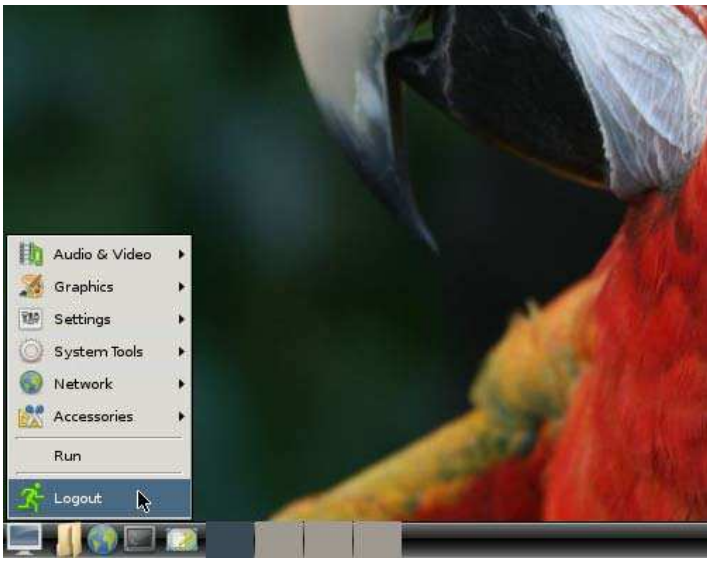
With a network connection, keyboard, mouse, and flat panel display with HDMI, the gplugD can be used as a standalone Linux machine. Follow these steps to boot the gplugD and use the Linux desktop:

1. With the gplugD powered down, plug in a USB mouse, USB keyboard, and a flat panel display with a free HDMI input using the provided cable. Connect the RJ-45 jack to a wired Ethernet network. Do not connect power to the Plug yet.
2. Turn on the display and select the HDMI input to which the gplugD is connected.
3. Connect power to the Plug **NOW**.

Boot time is about 40 seconds with X initializing the display around the 30-second mark. On some flat-panel monitors and TVs, the desktop toolbar may extend below the visible screen area. Depending on the particular display make and model, this viewing area can be adjusted by navigating to the picture settings or adjustment menu and selecting “just scan” or disabling overscan.

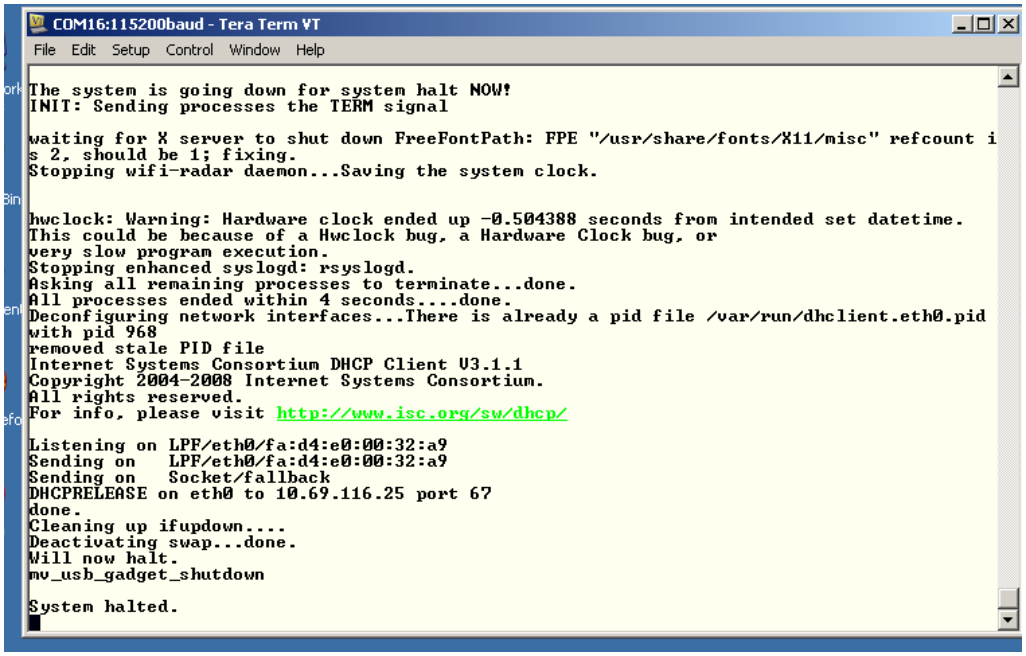
To shutdown the gplugD from the desktop, click the monitor icon in the lower left corner, select “Logout”, and then “Shutdown” (see Figure 9).

Figure 9: gplugD Shutdown



Although the desktop icons and toolbar disappear, nothing else on the display indicates that the Plug has shut down. On the console, however, status messages continue to be output, with a final "System halted" message appearing after about 20 seconds (Figure 10).

Figure 10: System Halted Message

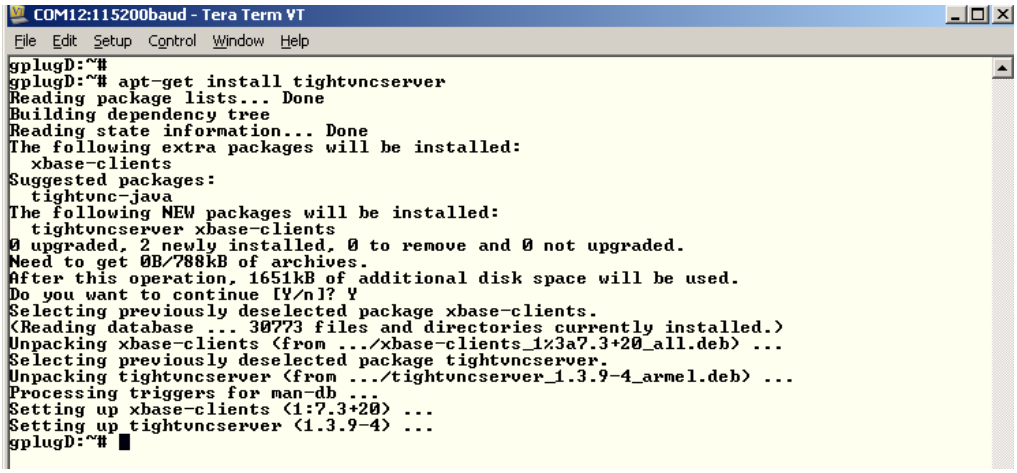


3.4 Remote Access with VNC

If there is no convenient way to connect a display to the gplugD, use the following procedure to remotely access the desktop with VNC:

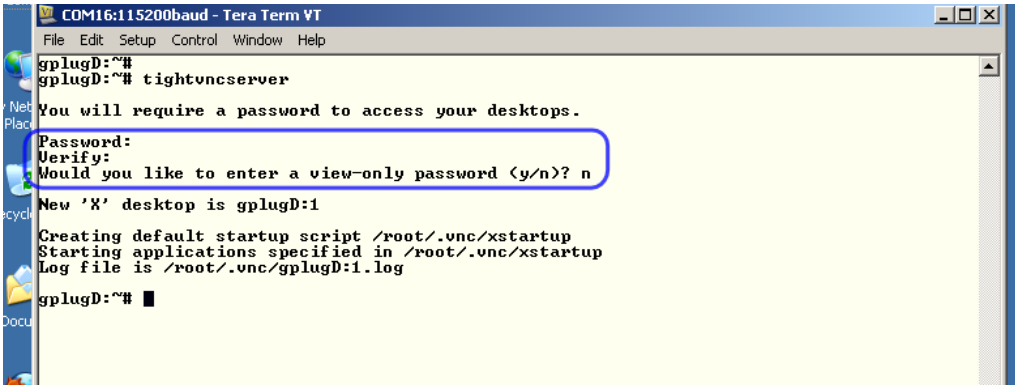
1. Visit http://www.tightvnc.com/ and download the Windows installer for TightVNC. Run the installer, and at the “Choose Components” screen, uncheck “TightVNC Server” before continuing.
2. With the gplugD powered up and connected to a wired Ethernet network with Internet access, get to the shell command prompt using the console serial port or SSH. Install TightVNC Server by entering “apt-get install tightvncserver” at the command prompt, and approve the installation of the “xbase-clients” and “tightvncserver packages” (see Figure 11).

Figure 11: TightVNC Server Installation



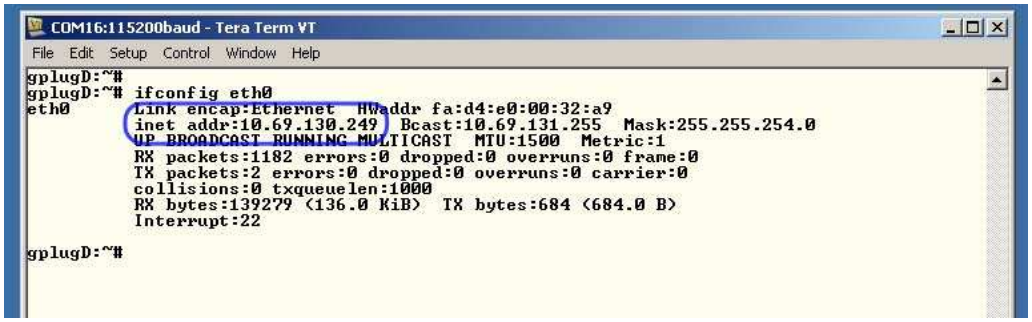
3. Launch TightVNC Server by issuing the command “tightvncserver.” Passwords are limited in length to 8 characters by the VNC protocol, so use the gplugD root password “nosoup4u” when prompted (see Figure 12). Also, do not create a view-only password. Take note of the X desktop name, which should be “gplugD:1” the first time TightVNC Server is run.

Figure 12: TightVNC Server Password Setup



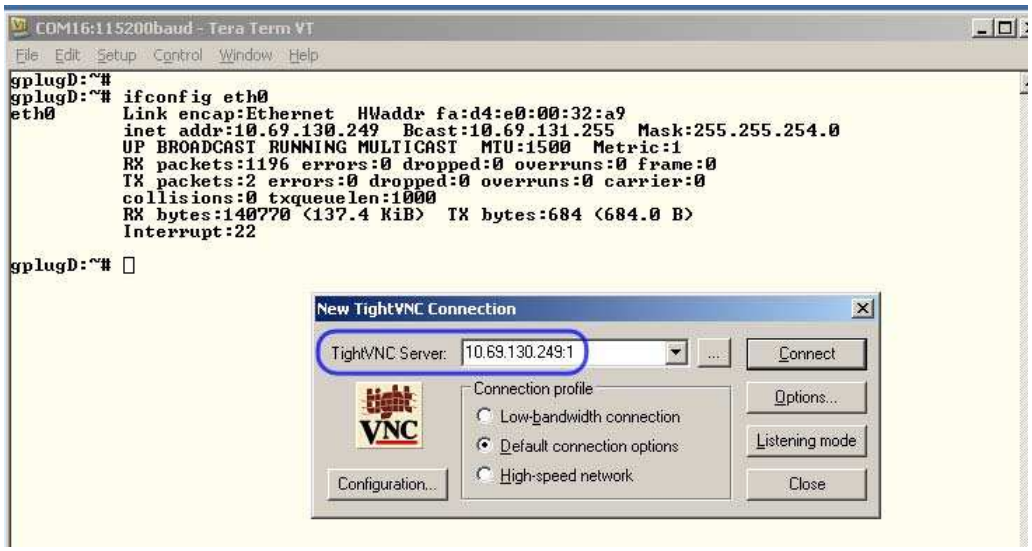
4. The Plug IP address must be known to connect with VNC. If using the console serial port, enter “ifconfig eth0” at the command prompt and note the IP address (see Figure 13).

Figure 13: Plug IP Address



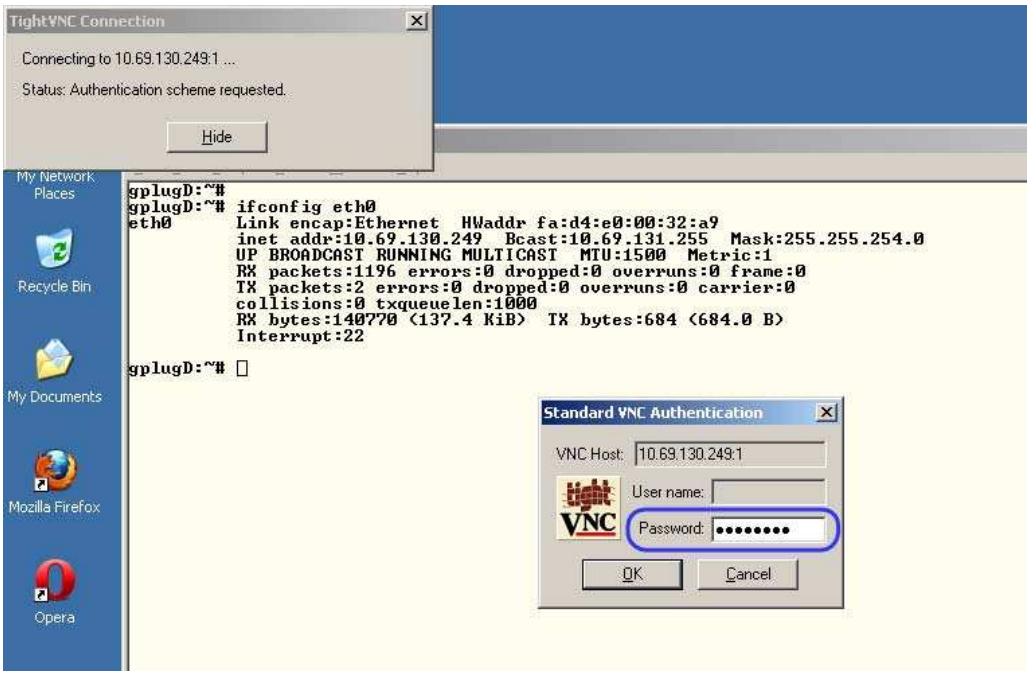
5. Now run TightVNC Viewer (**Start Menu** --> **Programs** --> **TightVNC** --> **TightVNC Viewer**). Enter the IP address of the gplugD followed by “:1” in the “TightVNC Server” box when prompted (see Figure 14).

Figure 14: TightVNC View New Connection Window



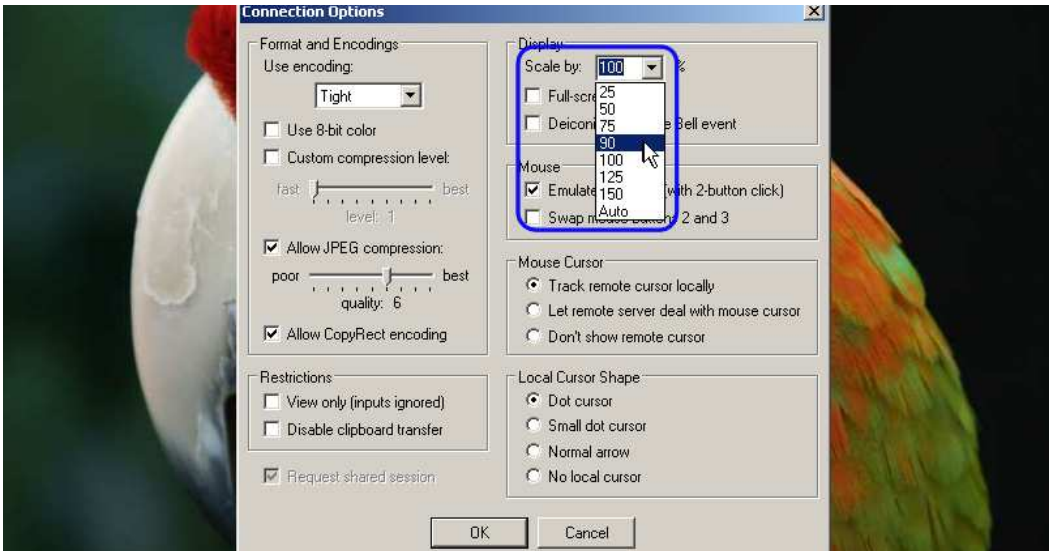
6. When prompted for the password, enter “nosoup4u” just as was done in Step 3 when launching TightVNC Server (Figure 15).

Figure 15: TightVNC Password Entry



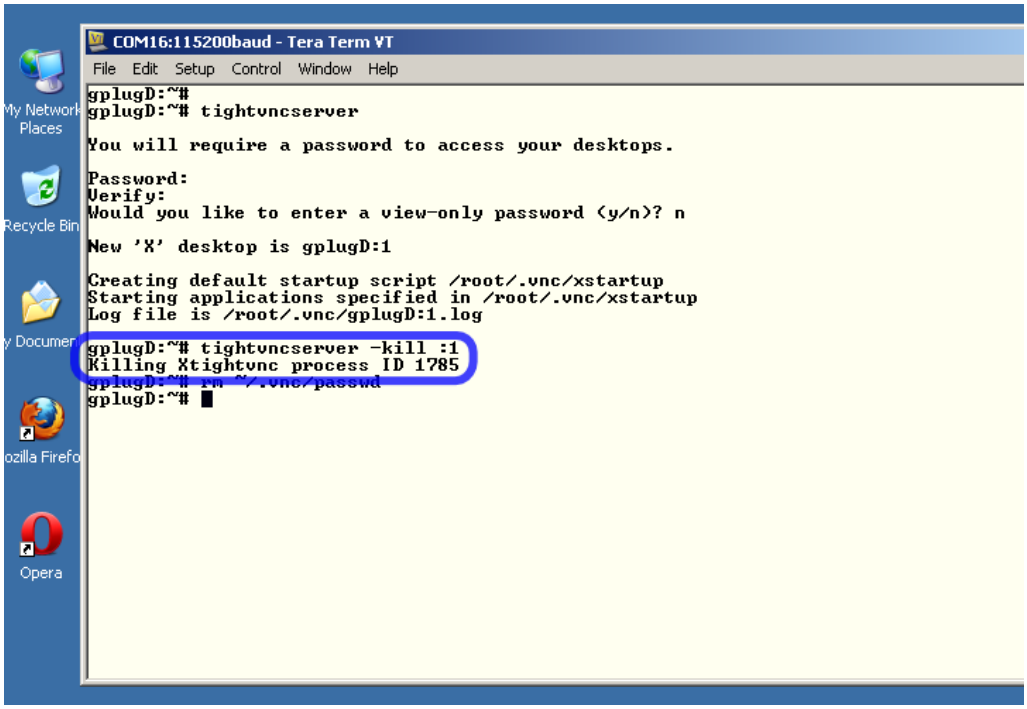
7. If the remote desktop is too large to fit in the window, click the “Connection options...” button at the left of the TightVNC Viewer toolbar and choose a suitable level from the **Scale by** dropdown (see Figure 16).

Figure 16: TightVNC Viewer Screen Scaling



8. Shutdown VNC by entering “tightvncserver –kill :1” at the shell command prompt. When TightVNC Server is next launched, it uses the password created in Step 3. If this is not preferred, remove the local password file with “rm ~/.vnc/passwd” (see Figure 17).

Figure 17: TightVNC Server Shutdown

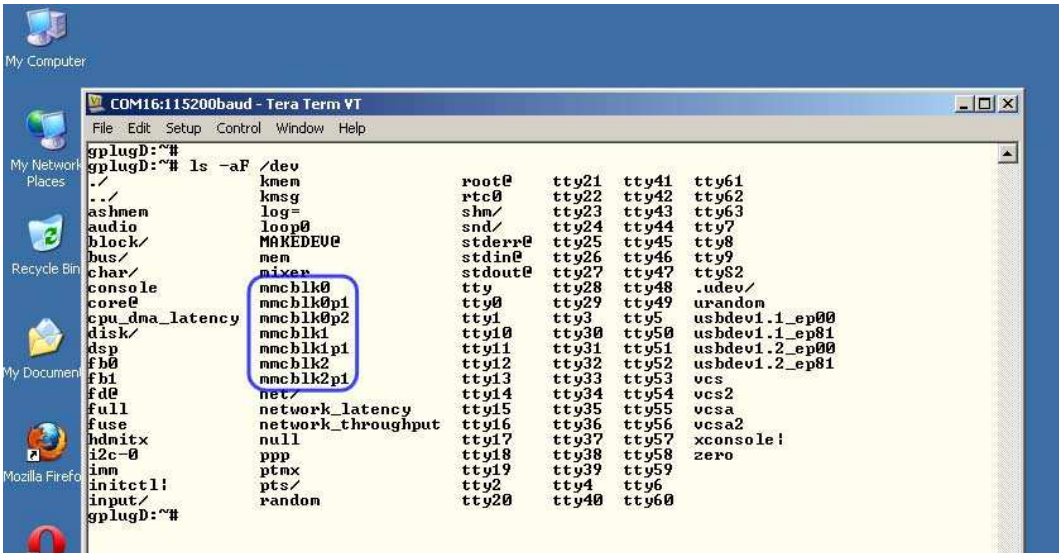


4 File Systems and Partitions

4.1 Where are the microSDHC cards in /dev?

Linux device drivers are accessed via the /dev directory. This screenshot shows a listing of the /dev directory on the gplugD with cards present in both external microSDHC slots and the internal slot (Figure 18).

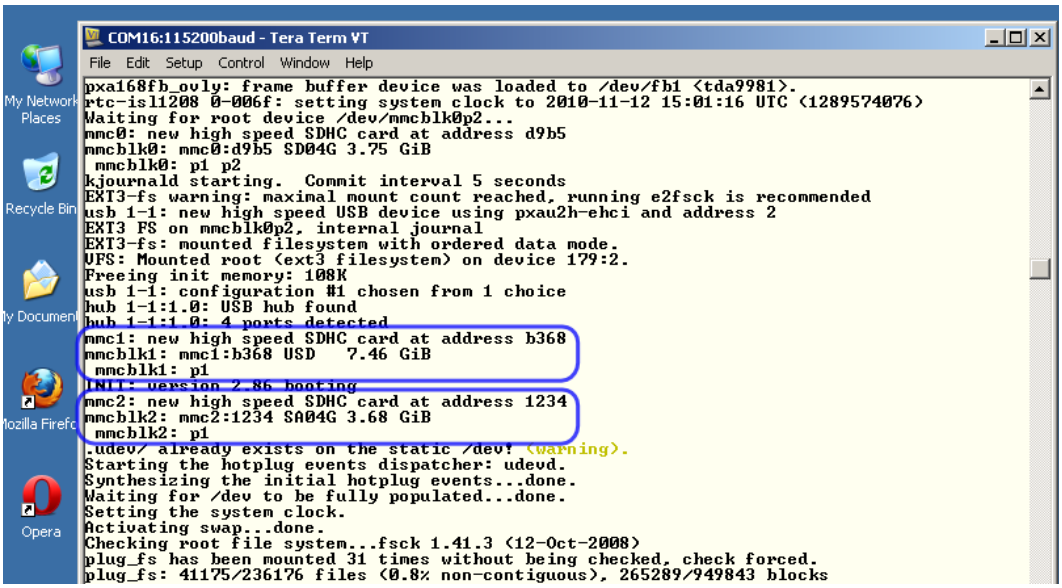
Figure 18: /dev Directory Listing



Each partition on a microSDHC card also has its own /dev node. For example, the card from which Linux boots, which is always instantiated as /dev/mmcblk0, has two partitions: /dev/mmcblk0p1 and /dev/mmcblk0p2.

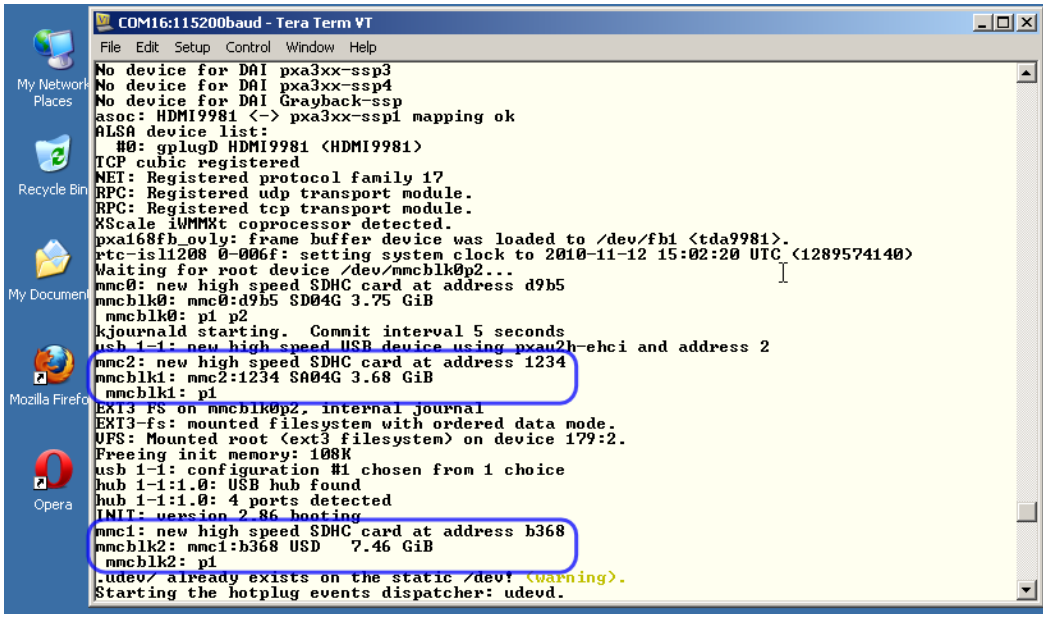
The /dev assignment of the internal and second external microSDHC card slots is not consistent each time the gplugD boots. In Figure 19, the start-up messages that follow power-on show assignment of the internal slot to /dev/mmcblk1 and the external slot to /dev/mmcblk2. As noted in the introduction chapter, the gplugD comes with an 8 GB Class 6 microSDHC card installed in the internal slot.

Figure 19: Power-On microSDHC Slot Assignment



Upon warm boot (for example, pressing the reset button or issuing the shell “reboot” command), however, the assignment is reversed: internal = /dev/mmcblk2 and external = /dev/mmcblk1. See Figure 20

Figure 20: Warm Boot microSDHC Assignment

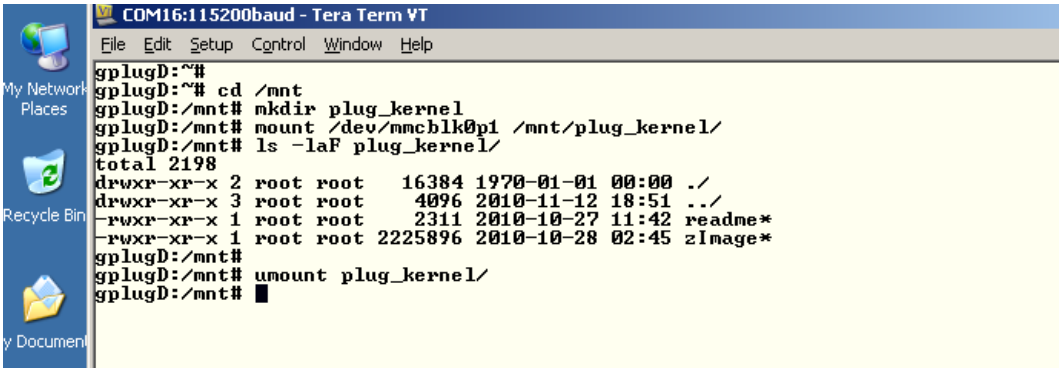


If no card is installed in the external slot or a card is not installed until after Linux boots, the internal microSDHC card is always assigned /dev/mmcblk1.

4.2 Mounting Partitions

Mounting storage partitions on the gplugD is performed the same way as it is on other Linux platforms. For example, to mount the kernel partition on the boot microSDHC device (see Figure 21).

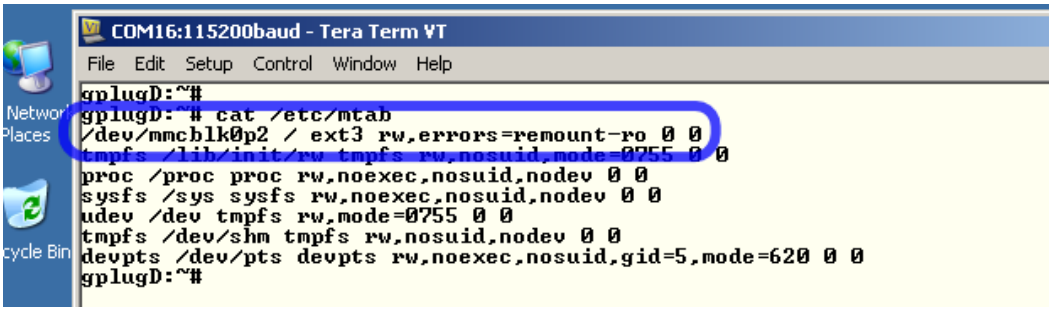
Figure 21: Mounting Kernel Partition on Boot microSDHC Device



1. Create a mount point in a suitable directory. In the screen shot above, this task is performed by issuing mkdir /mnt/plug_kernel. Placing storage mount points in the /mnt directory is standard Linux practice.
2. Use “mount /dev/mmcblk0p1 /mnt/plug_kernel” to mount the kernel partition. Listing the /mnt/plug_kernel directory shows that it holds a readme file and the compressed kernel zImage file.
3. To unmount the kernel partition, simply use “umount /mnt/plug_kernel.”

The boot microSDHC card cannot be removed because the gplugD root file system / resides on the /dev/mmcblk0p1 partition as shown in the /etc/mtab file (see Figure 22).

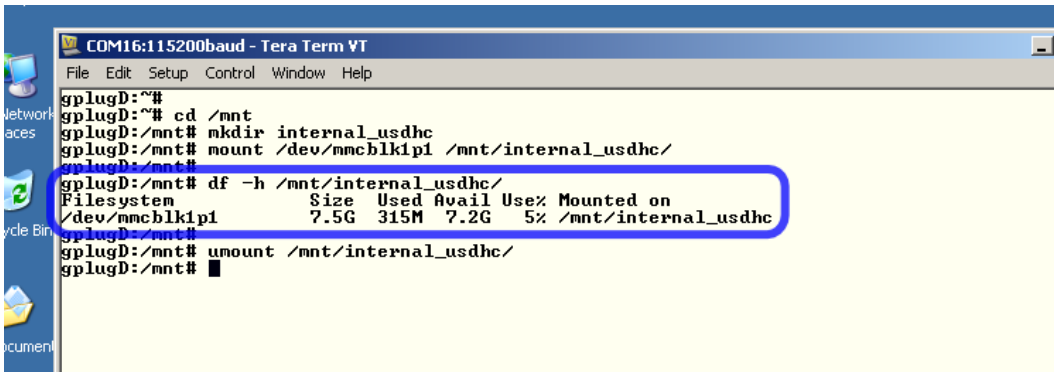
Figure 22: Root File System Mounted on /dev/mmcblk0p2 in /etc/mtab



Use the same procedure for the internal 8 GB microSDHC card.

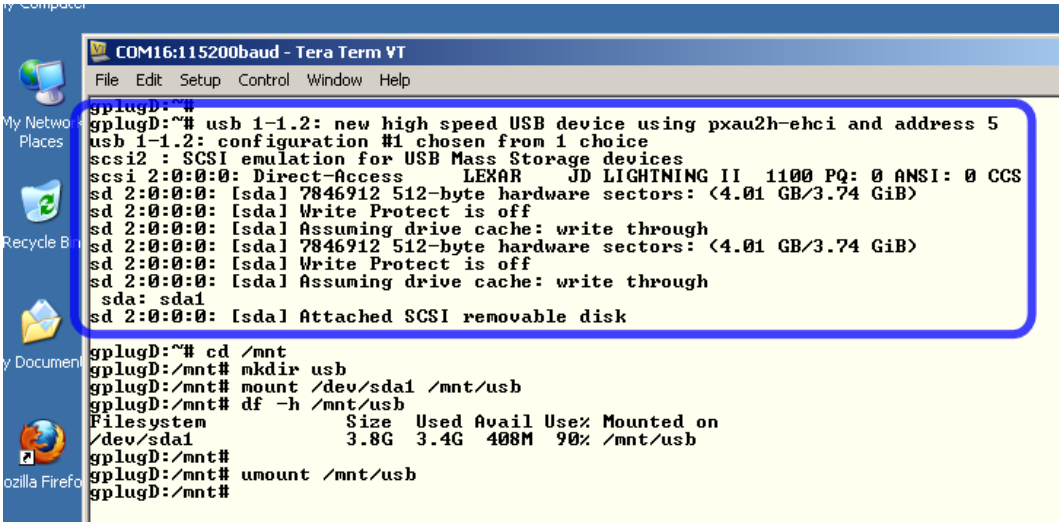
In Figure 6, /mnt/internal_usdhc is used for the mount point. The “df -h” command shows that this gplugD internal microSDHC card has approximately 7.2 GB of free space (Figure 23).

Figure 23: Free Space Available on Internal 8 GB microSDHC Card



External USB mass storage devices (hard drives, flash drives, and memory card readers) are mounted in a similar fashion, but they use the Linux SCSI driver and use the /dev/sdX naming convention (/dev/sda, /dev/sdb, etc). When using the console serial port, information about USB hot plug events is shown (these same messages can be read with “dmesg | tail -20” when logged into the gplugD via SSH). In Figure 24, /dev/sda is created for the USB flash drive, and its single partition is mounted with “mount /dev/sda1 /mnt/usb.”

Figure 24: USB Flash Drive on /dev/sda



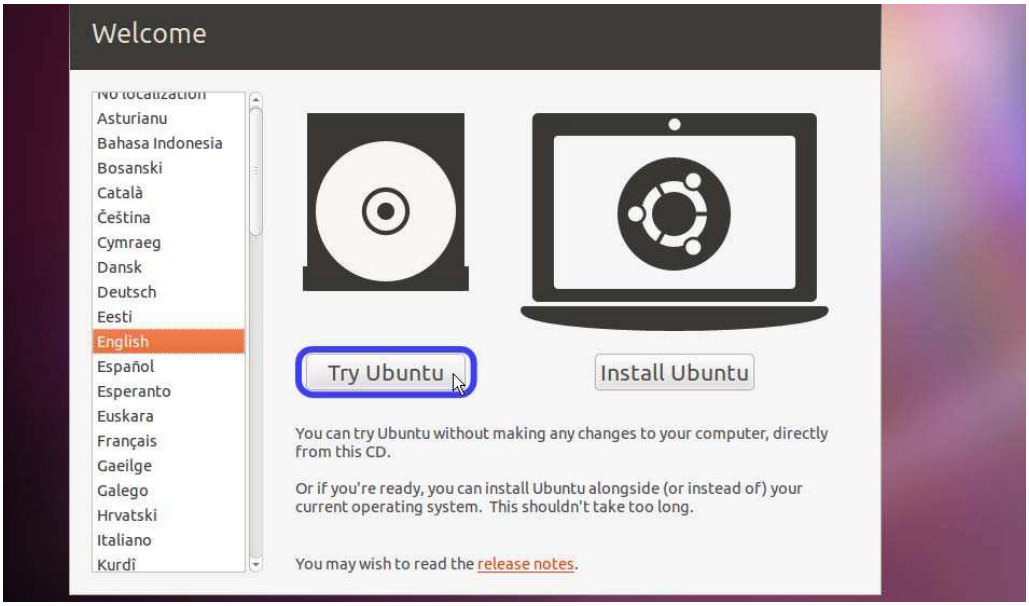
4.3 New Boot Card Setup

The gplugD boot microSDHC card has two partitions — one with a DOS file system and one with a Linux file system — and cannot be recreated under any Windows operating system. In the procedure that follows, a bootable SD card with a “live” Ubuntu Linux 10.10 distribution is used. The new kernel image and file system archive for the gplugD are placed in a folder named “gplugD” in the root directory of the SD card. Any recent version of Linux can also be used regardless of the boot media.

The only additional requirement is some means of reading and writing a microSDHC card. In the steps that follow, a USB card reader is used. Integrated SD card readers are found on many notebook computers and may be used along with a microSD-to-SD adapter. The gplugD boot card is a 4 GB microSDHC card, so when rebuilding it, the card reader must be SDHC-compatible. However, the gplugD kernel needs only a 4 MB partition, and the uncompressed root file system is on the order of 900 MB, so even a 1 GB card can be used, eliminating the need for a SDHC-compatible reader.

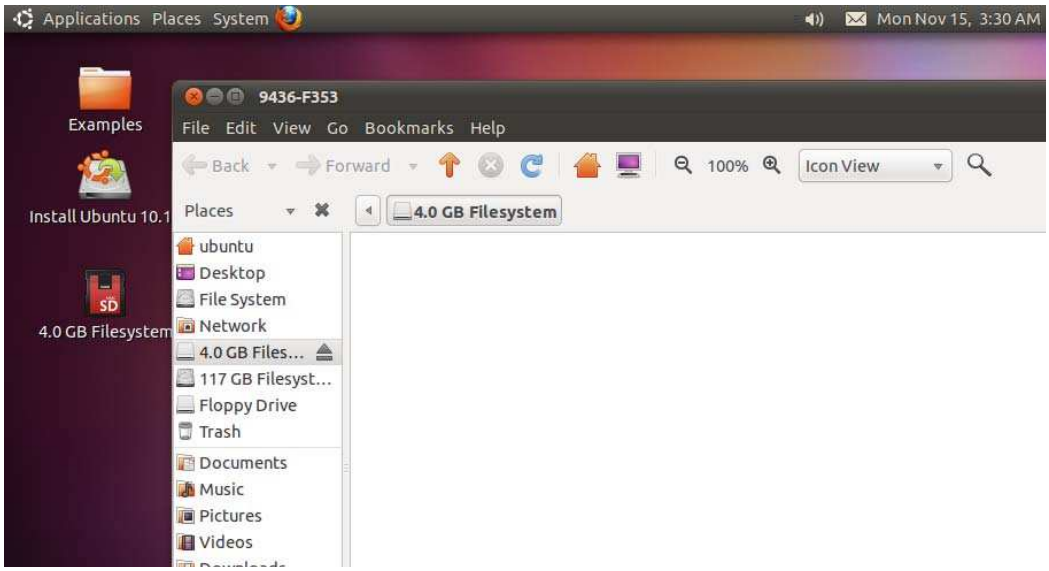
Boot the Linux operating system. When using the Ubuntu Linux 10.10 “live” trial, select the “Try Ubuntu” option when prompted. There is no need to install the operating system when following these instructions. See Figure 25.

Figure 25: Ubuntu “Live” Trial



1. Insert the new microSDHC card into the card reader. Ubuntu automounts the card and opens it in a file manager window. Close this window. See Figure 26.

Figure 26: Ubuntu Automounts the New microSDHC Card

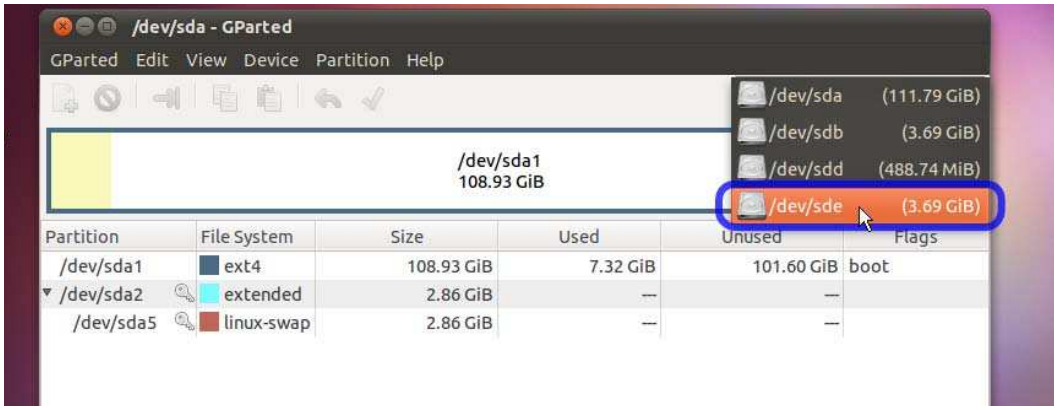


2. 3. Navigate to **System --> Administration** and select “GParted Partition Editor” (Figure 27).

Figure 27: Run the GParted Partition Editor

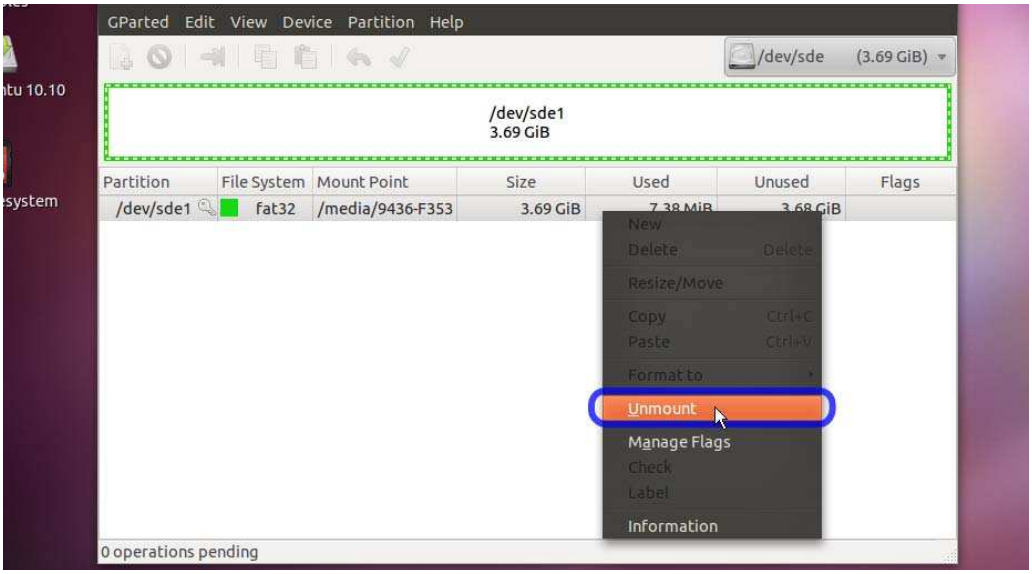
After GParted scans the available storage devices, select last entry from the drop-down menu in the upper right corner of the GParted window. Linux assigns storage drivers dynamically and in order. The last entry in the devices drop-down menu corresponds to the new microSDHC card as long as it was inserted after Linux boots. (On notebook computers with integrated SD card readers, look for **/dev/mmcblk0** on the drop-down menu). See Figure 28.

Figure 28: Select the New microSDHC Card Device



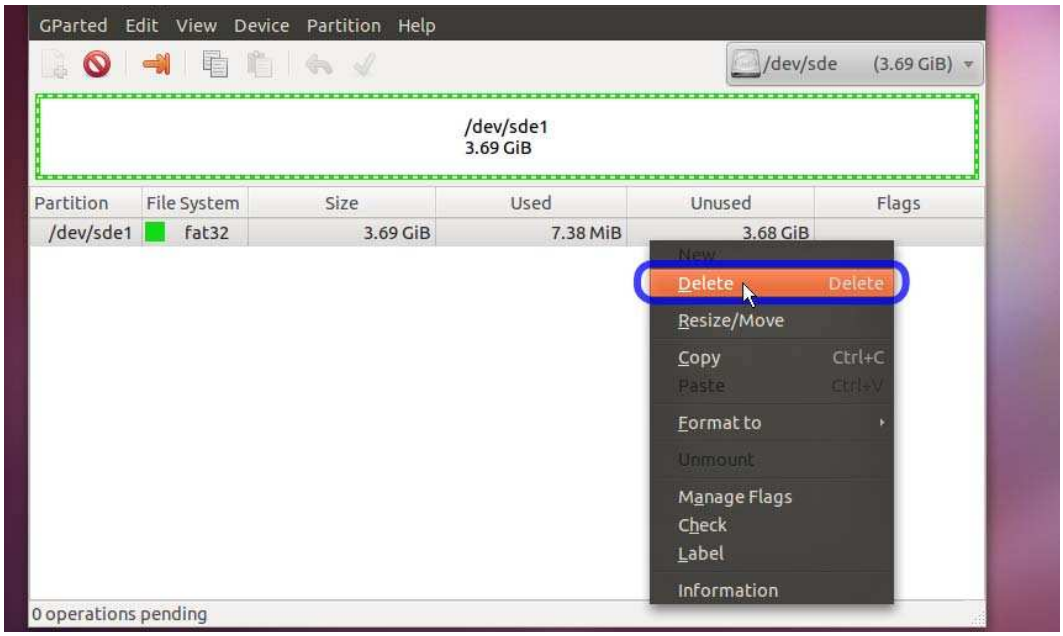
3. Before the partition table on the SD card can be modified, any mounted partitions must be unmounted. Right click on each partition listed (new cards and those used in digital cameras, MP3 players, and other electronic devices likely have only a single partition), and select **Unmount**. See Figure 29.

Figure 29: Unmount Previously Mounted Partitions



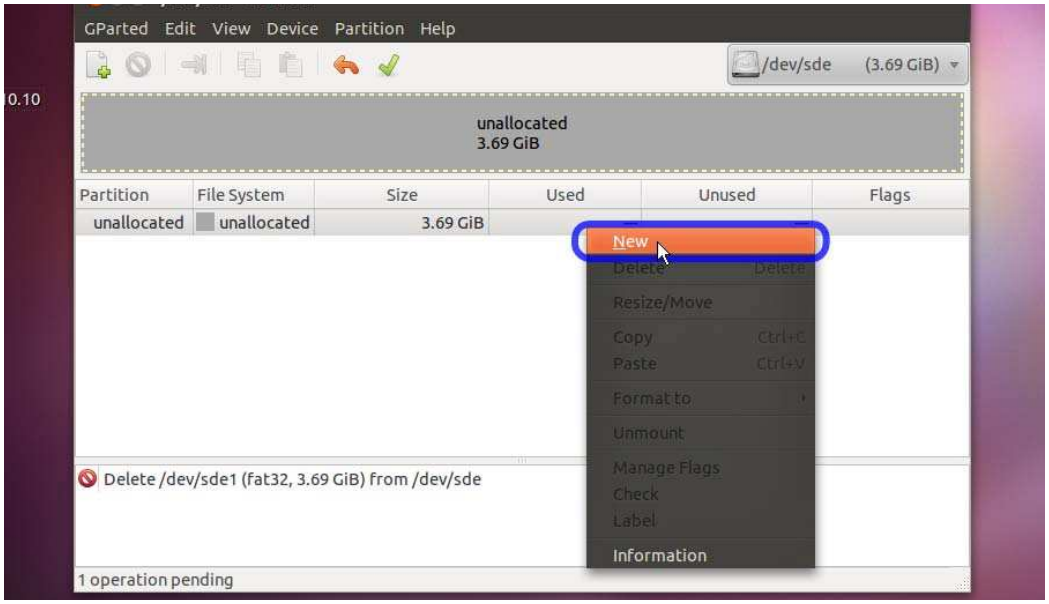
4. Next, right click on each partition and select **Delete** from the menu (Figure 30).

Figure 30: Delete Existing Partitions



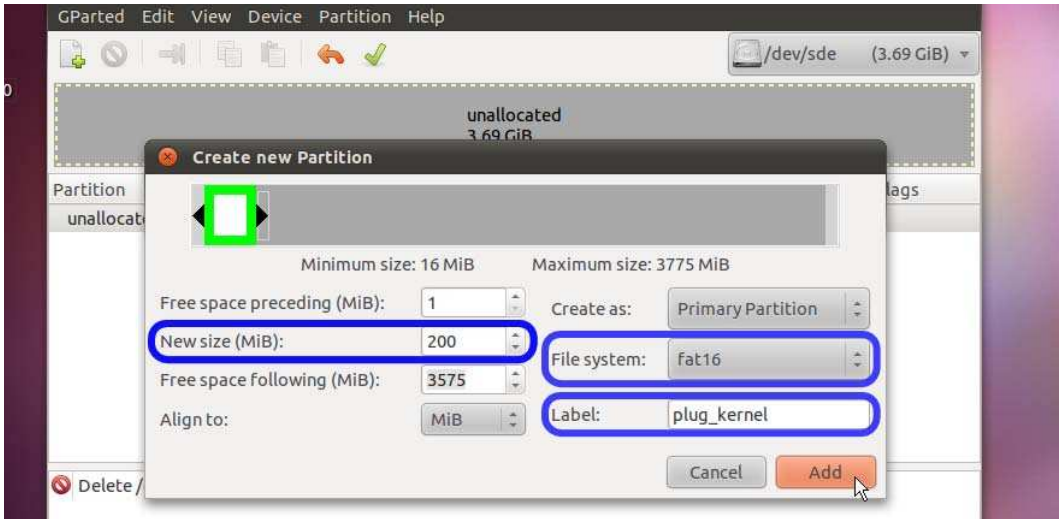
5. Create a new FAT16 partition for the kernel “zImage” file by right clicking on the unallocated space and selecting **New** from the menu (Figure 31).

Figure 31: Creating a FAT16 Partition for the Linux Kernel



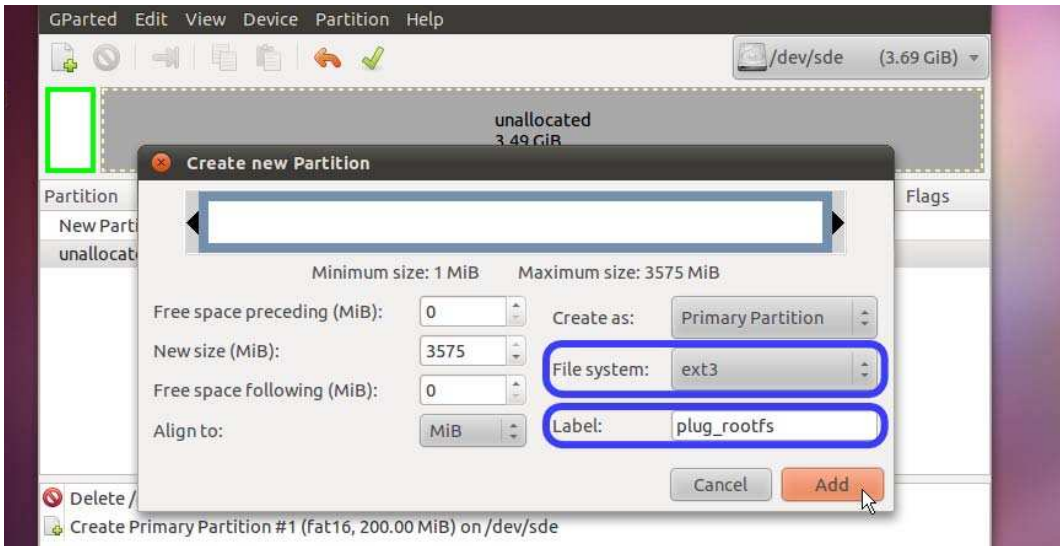
6. In the **Create new Partition** dialog box, (a) make the **New size (MiB)** big enough to hold the “zImage” file (4 MiB is sufficient for the stock gplugD kernel), (b) select “fat16” for the **File system**, (c) use “plug_kernel” for the new partition **Label,** and (d) click the **Add** button (Figure 32).

Figure 32: FAT16 Partition Parameters



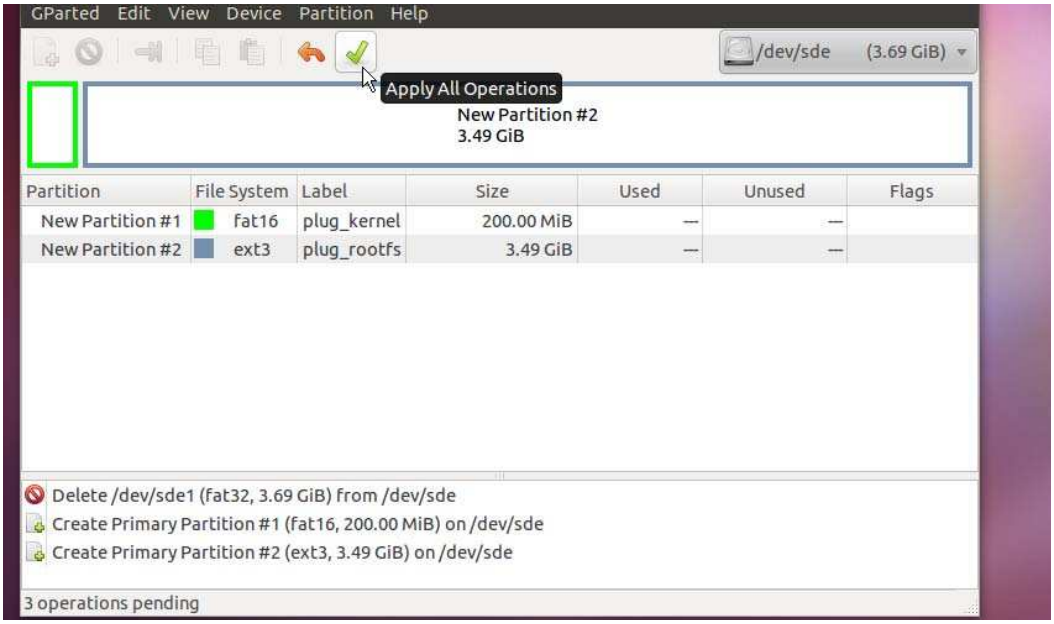
7. Repeat the same procedure for all of the remaining unallocated space. GParted will default to all of the remaining space for the new partition size. Select “ext3” for the **File system**, use “plug_rootfs” for the new partition, and then click the **Add** button (see Figure 33).

Figure 33: ext3 Partitions Parameters



8. Now click the green check mark in the toolbar to execute the chosen partition modifications (Figure 34).

Figure 34: Apply Pending Partition Modifications

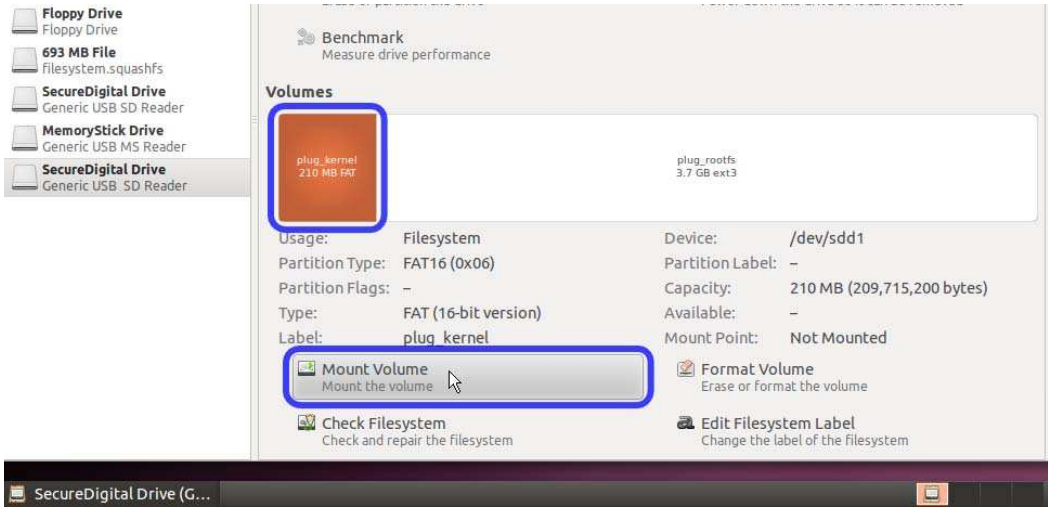


9. Once GParted finishes, the new partitions must be mounted in order to copy the kernel image and root file system onto them. Navigate to **System** --> **Administration** and select **Disk Utility** (Figure 35).

Figure 35: Launch the Disk Utility

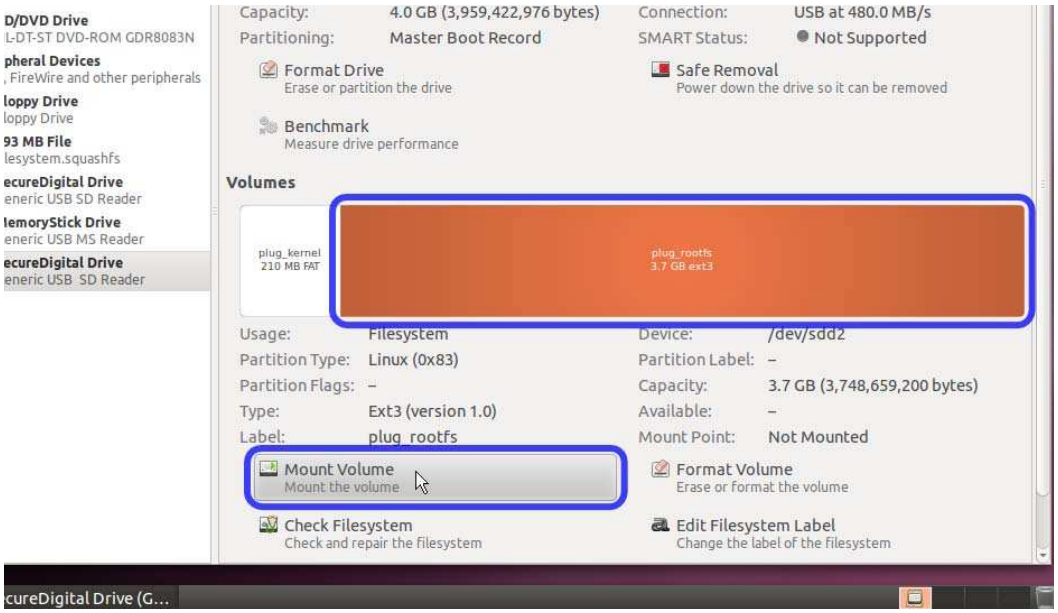
10. Select the SD card reader under peripheral devices. A graphical representation of the two partitions created on the microSDHC will be shown. Select the **plug_kernel** partition, then click the **Mount Volume** button. Notice that the FAT16 partition is mounted at /media/plug_kernel (see Figure 36).

Figure 36: Mounting the FAT16 Partition



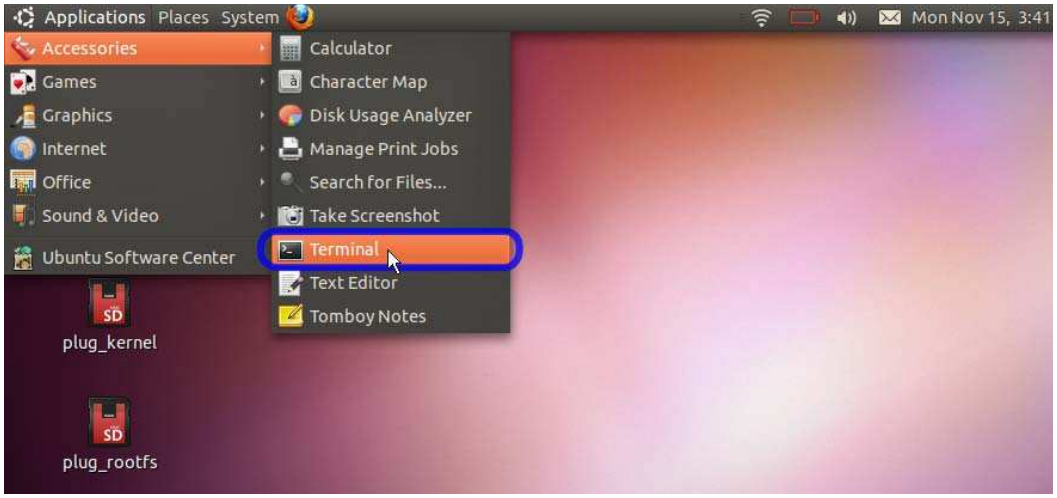
11. Now click the **plug_rootfs** partition, and mount it. Notice that the EXT3 partition is mounted at /media/plug_rootfs (see Figure 37).

Figure 37: Mount the ext3 Partition



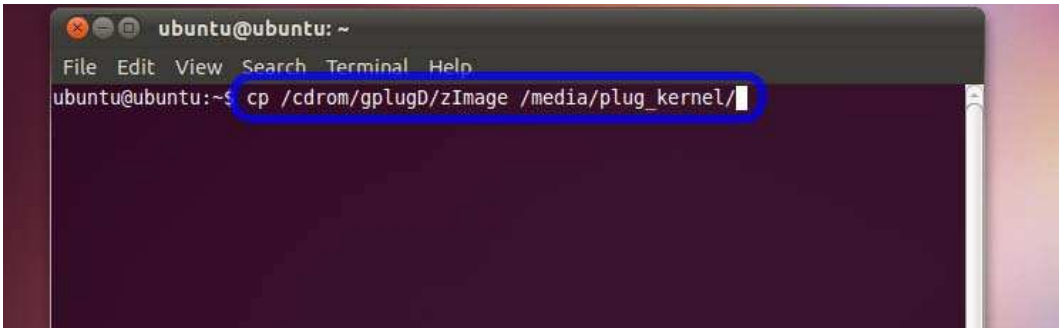
12. To copy the kernel image file and extract the root file system to their respective partitions, navigate to **Applications** --> **Accessories** and select “Terminal” to get to the command shell (Figure 38).

Figure 38: Launch the Terminal



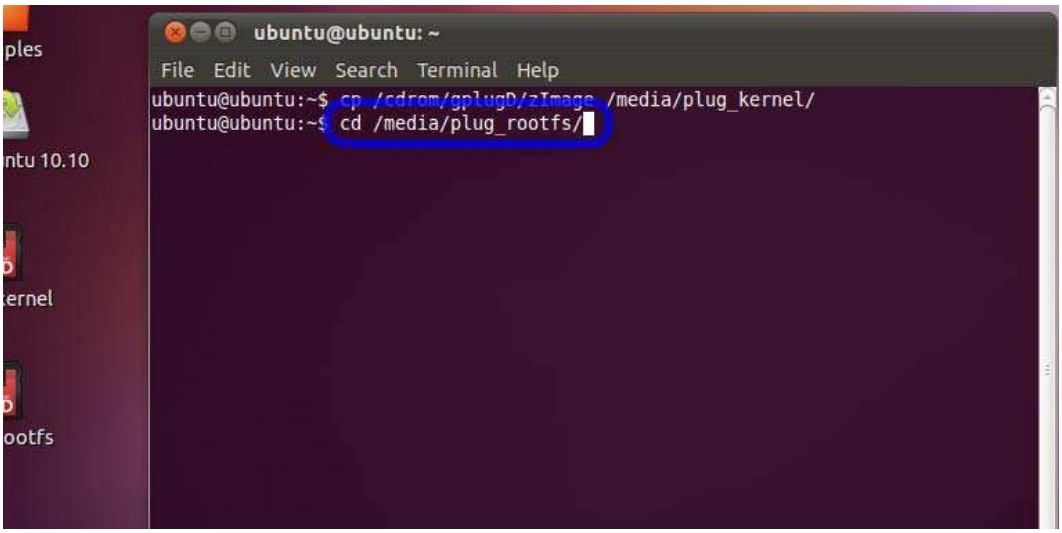
13. At the command prompt, enter “cp /cdrom/gplugD/zImage/media/plug_kernel” (see Figure 39). The “zImage” file resides in /cdrom/gplugD/ when using the Ubuntu Linux 10.10 “live” trial with the kernel image and file system archive placed in a folder named “gplugD” in the root directory of the bootable SD card or USB drive. The path to these files may be different if using either a computer that already has Linux installed or the “live” trial of a different Linux distribution.

Figure 39: Copy the Kernel zImage File to the FAT16 Partition



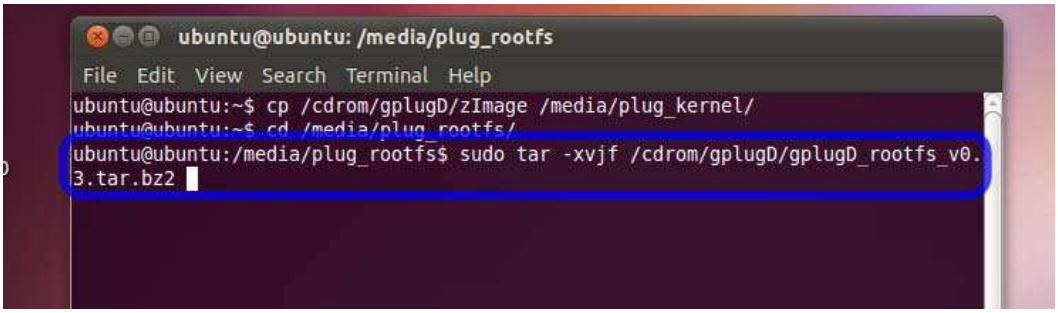
14. The root file system archive must be extracted to the EXT3 partition now mounted at /media/plug_rootfs. Make this the working directory by entering “cd/media/plug_rootfs” (Figure 40).

Figure 40: Change Directory to the ext3 Partition for the Root File System



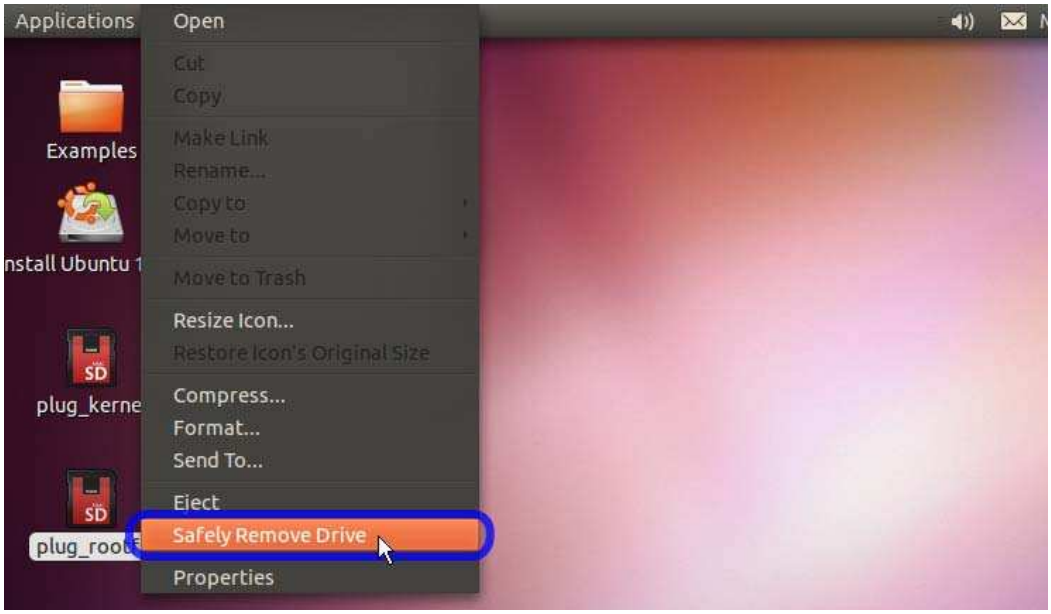
15. Extract the contents of the root file system archive with “sudo tar –xvjf /cdrom/gplugD/gplugD_rootfs_v0.3.tar.bz2.” Use the appropriate file name if a new release of the gplugD root file system has been made available. Version 0.3 of the root file system is about 470 MB and, depending upon the write speed of the microSDHC card used, it may take around 10 minutes to extract all of the files (see Figure 41).

Figure 41: Decompress and Extract the Root File System Tarball



16. When all of the files have been extracted, the SD card must be safely removed from the reader. If using an integrated SD card reader, launch the Disk Utility (see Step 11), unmount the FAT16 and EXT3 partitions, and eject the card from the reader. USB card readers can be disconnected by right-clicking on either the **plug_kernel** or the **plug_rootfs** icon and selecting **Safely Remove Drive** (see Figure 42).

Figure 42: Safely Remove the microSDHC Card



File system writes to the microSDHC card may still be pending in the disk cache, and Ubuntu Linux displays an activity message if this is the case. Do not remove the microSDHC card until this window is dismissed (see Figure 43). The Plug can now be booted with the newly created card. Be sure to insert the contacts up into the microSD socket on the right (nearest the reset button) before applying power.

Figure 43: Wait for Linux to Finish Before Removing the New microSDHC Card

Marvell Semiconductor, Inc.
5488 Marvell Lane
Santa Clara, CA 95054, USA

Tel: 1.408.222.2500
Fax: 1.408.752.9028

www.marvell.com

Marvell. Moving Forward Faster