BeagleBoardUbuntu

(For BeagleBoardAngstrom, click here.) (Should Beagleboard:Ubuntu On BeagleBone Black be merged into this page?)

This page is about running a Linux distribution (ARM EABI (https://wiki.debian.org/ArmEabiPort)) Ubuntu (http://www.ubuntu.com/) on the BeagleBoard. BeagleBoard will boot the (ARM EABI) Ubuntu distribution from the SD card. Since much of this page is generic, it has also been extended to help support devices such as the PandaBoard and BeagleBone.

- For the best experience, make sure you have an LCD/HDMI monitor attached to the BeagleBoard's HDMI port, 2 GB/4 GB/8 GB SD card, and a known good USB 2.0 hub with mouse and keyboard.

Contents

/etc/network/interfaces

Lightweight window managers

Web Apps

Midori

Surveillance

Motion

Robotics

ROS

Help

If you need any help:

- Kernel related help:
  - Email Beagleboard user group (https://groups.google.com/group/beagleboard) *Recommended method
  - *#beagle*: Beagle IRC on Freenode, accessible also by web interface (http://beagleboard.org/discuss) (logs (http://www.beagleboard.org/irclogs/index.php))
  - Kernel Trees
    - v4.17.x kernel branch (https://github.com/RobertCNelson/armv7-multiplatform/)
    - Development Kernel source code (https://github.com/RobertCNelson/linux-dev)
- Ubuntu related help:
  - *#ubuntu-arm*: Ubuntu's ARM IRC on Freenode (logs (http://irclogs.ubuntu.com/) -> year -> month -> day -> #ubuntu-arm.html)
- When requesting help, please provide some debugging information:
  - U-Boot Version installed on board
  - Kernel Version: uname -a
  - pastebin dmesg
    - Copy from serial port or use "dmesg | pastebinit" (sudo apt-get install pastebinit)

Required Beagle Software

Mainline U-Boot:

- All older BeagleBoard (classic) Ax, Bx, Cx and Dx boards are required to upgrade to at least these U-Boot versions
- XM Boards have no NAND, so MLO/u-boot.img is always required on the first partition
- Directions: Upgrade X-loader and U-Boot (http://elinux.org/BeagleBoardUbuntu#Upgrade_X-loader_and_U-boot)

Omap Serial Changes

boot.scr/boot.cmd changes:

With 2.6.35:

```
console=ttyS2,115200n8
```

With 2.6.36/37+:

```
console=ttyO2,115200n8
```

Serial console login: /etc/init/ttyO2.conf

```
start on stopped rc RUNLEVEL=[2345]
stop on runlevel [!2345]

respawn
exec /sbin/getty 115200 ttyO2
```

Method 1: Download a Complete Pre-Configured Image

Demo Image

- **Advanced Users only**: BeagleBoard xM: Kernel source, used in these demo images: https://github.com/RobertCNelson/armv7-multiplatform

```
git clone https://github.com/RobertCNelson/armv7-multiplatform.git
cd armv7-multiplatform
git checkout origin/v5.4.x -b tmp
./build_kernel.sh
```

- **Advanced Users only**: BeagleBone/BeagleBone Black/PocketBeagle: Kernel v4.19.x source, used in these demo images: https://github.com/RobertCNelson/ti-linux-kernel-dev/tree/ti-linux-4.19.y

```
git clone https://github.com/RobertCNelson/ti-linux-kernel-dev.git
cd ti-linux-kernel-dev
git checkout origin/ti-linux-4.19.y -b tmp
./build_kernel.sh
```

Ubuntu (18.04.4)

Default username/password:

- username: ubuntu
- password: temppwd

Image Updated:

- 2020-03-12
  - BeagleBoard xM: v5.4.24-armv7-x20 kernel
  - All BeagleBone Variants and PocketBeagle: v4.19.94-ti-r36 kernel
  - BeagleBoard-X15 (and BeagleBone AI): v4.19.94-ti-r36 kernel
- 2019-04-10
  - BeagleBoard xM: v4.19.31-armv7-x31 kernel
  - All BeagleBone Variants and PocketBeagle: v4.14.108-ti-r104 kernel
  - BeagleBoard-X15: v4.14.108-ti-r104 kernel
- 2018-12-10
  - BeagleBoard xM: v4.19.8-armv7-x11 kernel
  - All BeagleBone Variants and PocketBeagle: v4.14.79-ti-r84 kernel
  - BeagleBoard-X15: v4.14.79-ti-r84 kernel

Services Active:

```
Note: Depending on your internal network these may work out of the box
Apache, Port 80: http://arm.local/ (Bone: via usb) (Windows/Linux) http://192.168.7.2, (Mac/Linux) http://192.168.6.2
SSH, Port 22: ssh ubuntu@arm.local (Bone: via usb) (Windows/Linux) ubuntu@192.168.7.2, (Mac/Linux) ubuntu@192.168.6.2
Getty, Serial Port
```

Default user: ubuntu pass: temppwd

Get prebuilt image:

```
wget https://rcn-ee.com/rootfs/2020-03-12/elinux/ubuntu-18.04.4-console-armhf-2020-03-12.tar.xz
```

Verify Image with:

```
sha256sum ubuntu-18.04.4-console-armhf-2020-03-12.tar.xz
abe086f9132dfe8e8b9df8d14da225e0ce89a082abc92515de8a2ac63fc54ae2  ubuntu-18.04.4-console-armhf-2020-03-12.tar.xz
```

Unpack Image:

```
tar xf ubuntu-18.04.4-console-armhf-2020-03-12.tar.xz
cd ubuntu-18.04.4-console-armhf-2020-03-12
```

If you don't know the location of your SD card:

```
sudo ./setup_sdcard.sh --probe-mmc
```

You should see something like:

```
Are you sure? I don't see [/dev/idontknow], here is what I do see...

fdisk -l:
Disk /dev/sda: 500.1 GB, 500107862016 bytes <- x86 Root Drive
Disk /dev/sdd: 3957 MB, 3957325824 bytes <- MMC/SD card

lsblk:
NAME   MAJ:MIN RM   SIZE RO TYPE MOUNTPOINT
sda      8:0    0 465.8G  0 disk
├─sda1   8:1    0 446.9G  0 part /  <- x86 Root Partition
├─sda2   8:2    0     1K  0 part
└─sda5   8:5    0  18.9G  0 part [SWAP]
sdd      8:48   1   3.7G  0 disk
├─sdd1   8:49   1    64M  0 part
└─sdd2   8:50   1   3.6G  0 part
```

- In this example, we can see via mount, **/dev/sda1** is the x86 rootfs, therefore **/dev/sdd** is the other drive in the system, which is the MMC/SD card that was inserted and should be used by ./setup_sdcard.sh...

Install Image:

Quick install script for [board]

```
sudo ./setup_sdcard.sh --mmc /dev/sdX --dtb board
```

board options:

- BeagleBoard Ax/Bx/Cx/Dx - omap3-beagle
- BeagleBoard xM - omap3-beagle-xm
- All BeagleBone Variants - beaglebone
- OMAP5432 uEVM - omap5-uevm
- BeagleBoard-X15 (BeagleBone AI) - am57xx-beagle-x15

So for the BeagleBoard xM:

```
sudo ./setup_sdcard.sh --mmc /dev/sdX --dtb omap3-beagle-xm
```

Advanced: Build Image:

```
git clone https://github.com/RobertCNelson/omap-image-builder.git
cd omap-image-builder
git checkout v2020.03 -b tmp
```

Stable:

```
./RootStock-NG.sh -c rcn-ee_console_ubuntu_bionic_armhf
```

Flasher

eMMC: All BeagleBone Variants with eMMC

This image can be written to a 2GB (or larger) microSD card, via 'dd' on Linux or on Windows/Mac/Linux: https://etcher.io First press and hold the boot select button (next to the microSD card), then apply power. On boot-up the board should indicate it has started the flashing procedure visually via a Cylon Sweep pattern shown on the 4 LED's next to the Ethernet jack. Progress is reported on both the serial debug and HDMI connectors, once completed all 4 LED's should be full ON. Simply remove power, remove the microSD card and Ubuntu will now boot directly from eMMC.

Script for reference: (this is the script that writes to the eMMC)

```
https://github.com/RobertCNelson/boot-scripts/blob/master/tools/eMMC/init-eMMC-flasher-v3.sh
```

This script will only take about 5-6 Minutes after power on.

Notes:

- If only two LED's stay lit and nothing happens, the board has crashed due to lack of power. Retry with a 5Volt DC power supply connected.
- If the 4 LED's blink a constant pattern, the eMMC write has failed. First REMOVE ALL capes, then retry again.

User: ubuntu pass: temppwd

Image Updated:

- 2020-03-12
  - All BeagleBone Variants with eMMC: v4.19.94-ti-r36 kernel
- 2019-04-10
  - All BeagleBone Variants with eMMC: v4.14.108-ti-r104 kernel
- 2018-12-10
  - All BeagleBone Variants with eMMC: v4.14.79-ti-r84 kernel

Get prebuilt image:

```
wget https://rcn-ee.com/rootfs/2020-03-12/flasher/bone-eMMC-flasher-ubuntu-18.04.4-console-armhf-2020-03-12-2gb.img.xz
```

Verify Image with:

```
sha256sum bone-eMMC-flasher-ubuntu-18.04.4-console-armhf-2020-03-12-2gb*
2bc7f92df84dbd89c1cdd790ec794926a5d6b0f0b891143085d77141170518e2  bone-eMMC-flasher-ubuntu-18.04.4-console-armhf-2020-03-12-2gb.img.xz
```

Windows/Mac/Linux gui

```
http://etcher.io
```

Linux: (dd)

```
xzcat bone-eMMC-flasher-ubuntu-18.04.4-console-armhf-2020-03-12-2gb.img.xz | sudo dd of=/dev/sdX
```

eMMC: BeagleBoard-X15 and BeagleBone AI

This image can be written to a 2GB (or larger) microSD card, via 'dd' on Linux or on Windows/Mac/Linux: https://etcher.io First press and hold the boot select button (next to the microSD card), then apply power. On boot-up the board should indicate it has started the flashing procedure visually via a Cylon Sweep pattern shown on the 4 LED's next to the Ethernet jack. Progress is reported on both the serial debug and HDMI connectors, once completed all 4 LED's should be full ON. Simply remove power, remove the microSD card and Ubuntu will now boot directly from eMMC.

Script for reference: (this is the script that writes to the eMMC)

```
https://github.com/RobertCNelson/boot-scripts/blob/master/tools/eMMC/init-eMMC-flasher-v3.sh
```

This script will only take about 5-6 Minutes after power on.

Notes:

- If only two LED's stay lit and nothing happens, the board has crashed due to lack of power. Retry with a 5Volt DC power supply connected.
- If the 4 LED's blink a constant pattern, the eMMC write has failed. First REMOVE ALL capes, then retry again.

User: ubuntu pass: temppwd

Image Updated:

- 2020-03-12
  - BeagleBoard-X15 (and BeagleBone AI): v4.19.94-ti-r36 kernel
- 2019-04-10
  - BeagleBoard-X15 (and BeagleBone AI): v4.14.108-ti-r104 kernel
- 2018-12-10
  - BeagleBoard-X15 (and BeagleBone AI): v4.14.79-ti-r84 kernel

Get prebuilt image:

```
wget https://rcn-ee.com/rootfs/2020-03-12/flasher/am57xx-eMMC-flasher-ubuntu-18.04.4-console-armhf-2020-03-12-2gb.img.xz
```

Verify Image with:

```
sha256sum am57xx-eMMC-flasher-ubuntu-18.04.4-console-armhf-2020-03-12-2gb*
0c359f2b5e9d27d167f55877d3abfea0af2fe721463136f354dfd5d72be8a541  am57xx-eMMC-flasher-ubuntu-18.04.4-console-armhf-
2020-03-12-2gb.img.xz
```

Windows/Mac/Linux gui

```
http://etcher.io
```

Linux: (dd)

```
xzcat am57xx-eMMC-flasher-ubuntu-18.04.4-console-armhf-2020-03-12-2gb.img.xz | sudo dd of=/dev/sdX
```

raw microSD img

BeagleBoard xM

This image can be written to a 2GB (or larger) microSD card, via 'dd' on Linux or on Windows/Mac/Linux: https://etcher.io

User: ubuntu pass: temppwd

Auto partition resize:

```
cd /opt/scripts/tools
git pull
./grow_partition.sh
sudo reboot
```

Image Updated:

- 2020-03-12
  - BeagleBoard xM: v5.4.24-armv7-x20 kernel
- 2019-04-10
  - BeagleBoard xM: v4.19.31-armv7-x31 kernel
- 2018-12-10
  - BeagleBoard xM: v4.19.8-armv7-x11 kernel

Get prebuilt image:

```
wget https://rcn-ee.com/rootfs/2020-03-12/microsd/bbxm-ubuntu-18.04.4-console-armhf-2020-03-12-2gb.img.xz
```

Verify Image with:

```
sha256sum bbxm-ubuntu-18.04.4-console-armhf-2020-03-12-2gb*
a3d8780a69ceca6c5462c80c5a061dbba7df91d9111e08e718ecf55420302b91  bbxm-ubuntu-18.04.4-console-armhf-2020-03-12-
2gb.img.xz
```

Windows/Mac/Linux gui

```
http://etcher.io
```

Linux: (dd)

```
xzcat bbxm-ubuntu-18.04.4-console-armhf-2020-03-12-2gb.img.xz | sudo dd of=/dev/sdX
```

All BeagleBone Variants and PocketBeagle

This image can be written to a 2GB (or larger) microSD card, via 'dd' on Linux or on Windows/Mac/Linux: https://etcher.io

User: ubuntu pass: temppwd

Auto partition resize:

```
cd /opt/scripts/tools
git pull
./grow_partition.sh
sudo reboot
```

Image Updated:

- 2020-03-12
  - All BeagleBone Variants and PocketBeagle: v4.19.94-ti-r36 kernel
- 2019-04-10
  - All BeagleBone Variants and PocketBeagle: v4.14.108-ti-r104 kernel
- 2018-12-10
  - All BeagleBone Variants and PocketBeagle: v4.14.79-ti-r84 kernel

Get prebuilt image:

```
wget https://rcn-ee.com/rootfs/2020-03-12/microsd/bone-ubuntu-18.04.4-console-armhf-2020-03-12-2gb.img.xz
```

Verify Image with:

```
sha256sum bone-ubuntu-18.04.4-console-armhf-2020-03-12-2gb*
4c41976f1f574a4786002482167fe6443d8c9f1bced86a174459ffeba1f5b780  bone-ubuntu-18.04.4-console-armhf-2020-03-12-
2gb.img.xz
```

Windows/Mac/Linux gui

```
http://etcher.io
```

Linux: (dd)

```
xzcat bone-ubuntu-18.04.4-console-armhf-2020-03-12-2gb.img.xz | sudo dd of=/dev/sdX
```

BeagleBoard-X15 and BeagleBone AI

This image can be written to a 2GB (or larger) microSD card, via 'dd' on Linux or on Windows/Mac/Linux: https://etcher.io

User: ubuntu pass: temppwd

Auto partition resize:

```
cd /opt/scripts/tools
git pull
./grow_partition.sh
sudo reboot
```

Image Updated:

- 2020-03-12
  - BeagleBoard-X15 (and BeagleBone AI): v4.19.94-ti-r36 kernel
- 2019-04-10
  - BeagleBoard-X15 (and BeagleBone AI): v4.14.108-ti-r104 kernel
- 2018-12-10
  - BeagleBoard-X15 (and BeagleBone AI): v4.14.79-ti-r84 kernel

Get prebuilt image:

```
wget https://rcn-ee.com/rootfs/2020-03-12/microsd/am57xx-ubuntu-18.04.4-console-armhf-2020-03-12-2gb.img.xz
```

Verify Image with:

```
sha256sum am57xx-ubuntu-18.04.4-console-armhf-2020-03-12-2gb*
cd253167b43186c14c02e9e85ede006cd51336d5f8beae5b96765f494c24bd91  am57xx-ubuntu-18.04.4-console-armhf-2020-03-12-2gb.img.xz
```

Windows/Mac/Linux gui

```
http://etcher.io
```

Linux: (dd)

```
xzcat am57xx-ubuntu-18.04.4-console-armhf-2020-03-12-2gb.img.xz | sudo dd of=/dev/sdX
```

Method 2: Manual Install (no automatic scripts)

Note, this section used to have a lot of details, but maintenance of the two wiki's became a pain, so for now on we will just link to my other pages:

Beagle/Beagle xM

```
http://eewiki.net/display/linuxonarm/BeagleBoard
```

BeagleBone

```
http://eewiki.net/display/linuxonarm/BeagleBone
```

BeagleBone Black

```
http://eewiki.net/display/linuxonarm/BeagleBone+Black
```

Panda/Panda ES

```
http://eewiki.net/display/linuxonarm/PandaBoard
```

Advanced

Install Latest Kernel Image

General apt syntax for searching and installing a specific kernel:

```
sudo apt-get update
sudo apt-cache search linux-image | grep <branch>
sudo apt-get install linux-image-<specific version>
sudo reboot
```

Latest kernel script

```
cd /opt/scripts/tools/
git pull
sudo ./update_kernel.sh <OPTIONS>
```

3.8.x

This is the first beagleboard.org long term kernel tree with capemanager support, it's been the default install for Debian Wheezy

```
beagleboard.org patchset: https://github.com/beagleboard/linux/tree/3.8
```

```
3.8.x BeagleBone/BeagleBone Black FULL Cape Support
--bone-channel --stable
```

```
3.8.x BeagleBone/BeagleBone Black FULL Cape Support + Xenomai
```

```
--bone-xenomai-channel --stable
```

4.9.x-ti

```
beagleboard.org patchset: https://github.com/beagleboard/linux/tree/4.9
Based on: http://git.ti.com/gitweb/?p=ti-linux-kernel/ti-linux-kernel.git;a=shortlog;h=refs/heads/ti-linux-4.9.y
```

```
4.9.x-ti BeagleBone/BeagleBone Black/BeagleBoard-X15
--ti-channel --lts-4_9
```

```
4.9.x-ti BeagleBone/BeagleBone Black/BeagleBoard-X15 + RT
--ti-rt-channel --lts-4_9
```

```
4.9.x-ti BeagleBone/BeagleBone Black/BeagleBoard-X15 + Xenomai
--ti-xenomai-channel --lts-4_9
```

4.14.x-ti

```
beagleboard.org patchset: https://github.com/beagleboard/linux/tree/4.14
Based on: http://git.ti.com/gitweb/?p=ti-linux-kernel/ti-linux-kernel.git;a=shortlog;h=refs/heads/ti-linux-4.14.y
```

```
4.14.x-ti BeagleBone/BeagleBone Black/BeagleBoard-X15
--ti-channel --lts-4_14
```

```
4.14.x-ti BeagleBone/BeagleBone Black/BeagleBoard-X15 + RT
--ti-rt-channel --lts-4_14
```

Mainline (4.9.x lts)

```
4.9.x BeagleBone/BeagleBone Black
--bone-kernel --lts-4_9
```

```
4.9.x BeagleBone/BeagleBone Black + RT
--bone-rt-kernel --lts-4_9
```

Mainline (4.14.x lts)

```
4.14.x BeagleBone/BeagleBone Black
--bone-kernel --lts-4_14
```

```
4.14.x BeagleBone/BeagleBone Black + RT
--bone-rt-kernel --lts-4_14
```

Reboot with your new Kernel Image.

Xorg Drivers

Script:

```
cd /opt/scripts/tools/
git pull
```

BeagleBoard/PandaBoard:

```
cd /opt/scripts/tools/graphics/
./ti-omapdrm.sh
```

BeagleBone/BeagleBone Black:

```
cd /opt/scripts/tools/graphics/
./ti-tilcdc.sh
```

SGX Drivers

SGX BeagleBone/BeagleBone Black

Note, these are FBDEV only, no xorg/x11/etc...

Install the "4.1.x" lts/bone kernel: http://elinux.org/BeagleBoardUbuntu#Mainline_.28lts.29

Build SGX userspace for 4.1.x (must be done on an x86, due to the TI 5.01.01.02 blob extractor)

```
git clone https://github.com/RobertCNelson/bb-kernel.git
cd bb-kernel/
git checkout origin/am33x-v4.1 -b tmp-sgx
./sgx_create_package.sh
```

Copy ./deploy/GFX_5.01.01.02.tar.gz to BeagleBone/BeagleBone Black and install

```
sudo tar xfv GFX_5.01.01.02.tar.gz -C /
cd /opt/gfxinstall/
sudo ./sgx-install.sh
sudo reboot
```

Verify omaplfb & pvrsrvkm loaded

```
debian@arm:~$ lsmod | grep omaplfb
omaplfb                12065  0
pvrsrvkm              178782  1 omaplfb
```

Xorg Drivers

Script:

```
cd /opt/scripts/tools/
git pull
```

BeagleBoard/PandaBoard:

```
cd /opt/scripts/tools/graphics/
./ti-omapdrm.sh
```

BeagleBone/BeagleBone Black:

```
cd /opt/scripts/tools/graphics/
./ti-tilcdc.sh
```

Swapfile

Using a File for Swap Instead of a Partition

On the Beagleboard you should expect to require a swap file given the limitation of how little RAM is available (between 256 MB and 512 MB). Some system programs like apt-get will only run properly when some swap space is present (due to 256 MB not being enough RAM).

Some images (such as those from Linaro.org) do not come with a swap partition or any swap space allocated.

Under Linux, swap space can be either a dedicated partition or a swap file. Both can be mounted as swap which the OS can access.

Creating a Swapfile

The following commands will create a 1 GB file, limit access only to root, format it as swap and then make it available to the OS:

```
sudo mkdir -p /var/cache/swap/
sudo dd if=/dev/zero of=/var/cache/swap/swapfile bs=1M count=1024
sudo chmod 0600 /var/cache/swap/swapfile
sudo mkswap /var/cache/swap/swapfile
sudo swapon /var/cache/swap/swapfile
```

To tell the OS to load this swapfile on each start up, edit the /etc/fstab file to include the following additional line:

```
/var/cache/swap/swapfile    none    swap    sw    0   0
```

To verify that the swapfile is accessilble as swap to the OS, run "top" or "htop" at a console.

Ubuntu Software

Wi-Fi Networking (command line)

/etc/network/interfaces

It is relatively easy to configure a Wi-Fi card from the command line.

You will need to edit the /etc/network/interfaces file. There are several guides available via Google.

This is a particularly useful guide https://ubuntuforums.org/showthread.php?t=202834

A sample /etc/network/interfaces file for a WPA2 encrypted access point is:

```
auto lo
iface lo inet loopback
auto wlan0
iface wlan0 inet dhcp
wpa-driver wext
wpa-ssid <NAME OF AP>
```

```
wpa-ap-scan 1
wpa-proto RSN
wpa-pairwise CCMP
wpa-group CCMP
wpa-key-mgmt WPA-PSK
wpa-psk <INSERT KEY XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX>
```

Your Wi-Fi card will automatically load these settings upon startup and initialize wireless network access.

Lightweight window managers

If you intend to use Ubuntu on the BeagleBoard you can install JWM or IceWM to improve performance.

JWM in particular uses little RAM. On a BeagleBoard with 256 MB, using JWM will leave about 60 MB free in which to run applications.

Web Apps

Midori

Given that the BeagleBoard has fewer resources than a desktop a lightweight browser is more responsive. Midori is a lightweight browser that still supports flash, etc. It is available from the standard repositories: http://en.wikipedia.org/wiki/Midori_%28web_browser%29

Surveillance

Motion

If you have a video source (webcam, IP cam, etc.) which appears as /dev/video0, etc. then you can use the Linux surveillance software "motion" to monitor the video stream and record periods of activity.

Motion is also available from the standard repositories: http://www.debian-administration.org/article/An_Introduction_to_Video_Surveillance_with_%27Motion%27 Using a 960x720 resolution webcam with a 15 fps rate under the UVC driver the Rev C BeagleBoard under Xubuntu reports ~60% CPU utilisation.

To make the BeagleBoard automatically start recording on boot, do the following:

- Auto Login - run "gdmsetup" from a terminal and select a user to automatically login
- Sessions - make sure you don't save any previous X Windows sessions so that it doesn't prompt you for which one you want
- motion.conf - edit /etc/motion/motion.conf to use the settings you want (that is, video output directory, record only video, record in MPEG-4, set frame rate, etc). Do this with "sudo medit /etc/motion/motion.conf" at a prompt.
- Boot script - create a new script in /etc/rc2.d called "S65motion_client" and set permissions appropriately ("sudo chmod 777 /etc/rc2.d/S65motion_client"). Then edit the file so it contains the following lines:

```
#! /bin/sh
/usr/bin/motion -c /etc/motion/motion.conf
```

This will now launch the motion client as root when you boot up.

Also note that unless your BeagleBoard can remember the time (battery backed up clock installed), the timestamps will not be correct until you update the time. If your BeagleBoard has an Internet connection this can be achieved using the ntpdate application.

Robotics

ROS

ROS (Robot Operating System) provides libraries and tools to help software developers create robot applications. It provides hardware abstraction, device drivers, libraries, visualizers, message-passing, package management, and more. ROS is licensed under an open source, BSD license.

There are currently builds of ROS for Ubuntu Bionic armhf. These builds include most but not all packages, and save a considerable amount of time compared to doing a full source-based installation:

http://wiki.ros.org/melodic/Installation/Ubuntu

Alternatively ROS can be installed from source and is generally easy to do so (although slow).

For more information about ROS, see www.ros.org.

Retrieved from "https://elinux.org/index.php?title=BeagleBoardUbuntu&oldid=510151"

This page was last edited on 13 March 2020, at 07:47.