

BeagleBoard

This is a page about TI's Cortex-A8 based; BeagleBoard.

- Availability
- Basic Requirements
- ARM Cross Compiler: GCC
- Bootloader: U-Boot
- Linux Kernel
- Root File System
 - Debian 9
 - Ubuntu 18.04 LTS
- Setup microSD card
- Install Bootloader
- Install Kernel and Root File System
 - Copy Root File System
 - Set `uname_r` in `/boot/uEnv.txt`
 - Copy Kernel Image
 - Copy Kernel Device Tree Binaries
 - Copy Kernel Modules
 - File Systems Table (`/etc/fstab`)
 - Remove microSD/SD card
- Comments

Availability

Boards:

BeagleBoard xM at Digi-Key

Basic Requirements

- Running a recent release of Debian, Fedora or Ubuntu; without OS Virtualization Software.
- ARM Cross Compiler – Linaro: <http://www.linaro.org>
 - Linaro Toolchain Binaries: <http://www.linaro.org/downloads/>
- Bootloader
 - Das U-Boot – the Universal Boot Loader: <http://www.denx.de/wiki/U-Boot>
 - Source: <http://git.denx.de/?p=u-boot.git;a=summary>
- Linux Kernel
 - Linus's Mainline tree: <https://git.kernel.org/cgit/linux/kernel/git/torvalds/linux.git>
- ARM based rootfs
 - Debian: <https://www.debian.org>
 - Ubuntu: <http://www.ubuntu.com>

ARM Cross Compiler: GCC

This is a pre-built (64bit) version of Linaro GCC that runs on generic linux, sorry (32bit) x86 users, it's time to upgrade...

Download/Extract:

```
~/  
  
wget -c https://releases.linaro.org/components/toolchain/binaries/6.4-  
2018.05/arm-linux-gnueabi/gcc-linaro-6.4.1-2018.05-x86_64_arm-linux-  
gnueabi.tar.xz  
tar xf gcc-linaro-6.4.1-2018.05-x86_64_arm-linux-gnueabi.tar.xz  
export CC=`pwd`/gcc-linaro-6.4.1-2018.05-x86_64_arm-linux-gnueabi/bin  
/arm-linux-gnueabi-
```

Test Cross Compiler:

```
~/
```

```
/${CC}gcc --version
arm-linux-gnueabi-gcc (Linaro GCC 6.4-2018.05) 6.4.1 20180425 [linaro-
6.4-2018.05 revision 7b15d0869c096fe39603ad63dc19ab7cf035eb70]
Copyright (C) 2017 Free Software Foundation, Inc.
This is free software; see the source for copying conditions. There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.
```

Bootloader: U-Boot

Das U-Boot – the Universal Boot Loader: <http://www.denx.de/wiki/U-Boot>
eewiki.net patch archive: <https://github.com/eewiki/u-boot-patches>

Download:

```
~/
```

```
git clone https://github.com/u-boot/u-boot
cd u-boot/
git checkout v2018.09-rc2 -b tmp
```

Patches:

```
~/u-boot
```

```
wget -c https://rcn-ee.com/repos/git/u-boot-patches/v2018.09-rc2/0001-
omap3_beagle-uEnv.txt-bootz-n-fixes.patch

patch -p1 < 0001-omap3_beagle-uEnv.txt-bootz-n-fixes.patch
```

Configure and Build:

```
~/u-boot
```

```
make ARCH=arm CROSS_COMPILE=${CC} distclean
make ARCH=arm CROSS_COMPILE=${CC} omap3_beagle_defconfig
make ARCH=arm CROSS_COMPILE=${CC}
```

Linux Kernel

Spectre: <https://meltdownattack.com> and <https://developer.arm.com/support/arm-security-updates/speculative-processor-vulnerability>
Spectre v2 Migration requires minimal: U-Boot: v2018.07 and Kernel: v4.18.x

```
[ 0.047363] CPU0: Spectre v2: using BPIALL workaround
```

This script will build the kernel, modules, device tree binaries and copy them to the deploy directory.
Download:

```
~/
```

```
git clone https://github.com/RobertCNelson/armv7-multiplatform  
cd armv7-multiplatform/
```

For v4.9.x (Longterm 4.9.x):

```
~/armv7-multiplatform/
```

```
git checkout origin/v4.9.x -b tmp
```

For v4.9.x-rt (Longterm 4.9.x + Real-Time Linux):

```
~/armv7-multiplatform/
```

```
git checkout origin/v4.9.x-rt -b tmp
```

For v4.14.x (Longterm 4.14.x):

```
~/armv7-multiplatform/
```

```
git checkout origin/v4.14.x -b tmp
```

For v4.14.x-rt (Longterm 4.14.x + Real-Time Linux):

```
~/armv7-multiplatform/
```

```
git checkout origin/v4.14.x-rt -b tmp
```

For v4.18.x (Stable):

```
~/armv7-multiplatform/
```

```
git checkout origin/v4.18.x -b tmp
```

Build:

```
~/armv7-multiplatform/  
  
./build_kernel.sh
```

Root File System

Debian 9

User	Password
debian	tempwd
root	root

Download:

```
~/  
  
wget -c https://rcn-ee.com/rootfs/eewiki/minfs/debian-9.5-minimal-armhf-2018-07-30.tar.xz
```

Verify:

```
~/  
  
sha256sum debian-9.5-minimal-armhf-2018-07-30.tar.xz  
9399d649d1ce9910bbfc745f59dc57ee0e1134f57e8cae01c8cd75a8bd9d1e1e  debian-9.5-minimal-armhf-2018-07-30.tar.xz
```

Extract:

```
~/  
  
tar xf debian-9.5-minimal-armhf-2018-07-30.tar.xz
```

Ubuntu 18.04 LTS

User	Password
ubuntu	tempwd

Download:

```
~/
```

```
wget -c https://rcn-ee.com/rootfs/eewiki/minfs/ubuntu-18.04.1-minimal-armhf-2018-07-30.tar.xz
```

Verify:

```
~/
```

```
sha256sum ubuntu-18.04.1-minimal-armhf-2018-07-30.tar.xz  
6b212ee7dd0d5c9c0af49c22cf78b63e6ad20cec641c303232fca9f21a18804c  ubuntu-  
18.04.1-minimal-armhf-2018-07-30.tar.xz
```

Extract:

```
~/
```

```
tar xf ubuntu-18.04.1-minimal-armhf-2018-07-30.tar.xz
```

Setup microSD card

For these instruction we are assuming, DISK=/dev/mmcblk0, lsblk is very useful for determining the device id.

```
export DISK=/dev/mmcblk0
```

Erase partition table/labels on microSD card:

```
sudo dd if=/dev/zero of=${DISK} bs=1M count=20
```

Create Partition Layout:

With util-linux v2.26, sfdisk was rewritten and is now based on libfdisk.

```
sudo sfdisk --version  
sfdisk from util-linux 2.27.1
```

sfdisk >= 2.26.x

```
sudo sfdisk ${DISK} <<-__EOF__
1M,12M,0xE,*
13M,,, -
__EOF__
```

sfdisk <= 2.25.x

```
sudo sfdisk --unit M ${DISK} <<-__EOF__
1,12,0xE,*
13,,, -
__EOF__
```

Format Partition:

```
for: DISK=/dev/mmcblk0
sudo mkfs.vfat -F 16 -n BOOT ${DISK}p1
sudo mkfs.ext4 -L rootfs ${DISK}p2

for: DISK=/dev/sdX
sudo mkfs.vfat -F 16 -n BOOT ${DISK}1
sudo mkfs.ext4 -L rootfs ${DISK}2
```

Mount Partition:

On most systems these partitions may will be auto-mounted...

```
sudo mkdir -p /media/boot/
sudo mkdir -p /media/rootfs/

for: DISK=/dev/mmcblk0
sudo mount ${DISK}p1 /media/boot/
sudo mount ${DISK}p2 /media/rootfs/

for: DISK=/dev/sdX
sudo mount ${DISK}1 /media/boot/
sudo mount ${DISK}2 /media/rootfs/
```

Install Bootloader

Copy MLO/u-boot.img to the boot partition

```
~/
```

```
sudo cp -v ./u-boot/MLO /media/boot/  
sudo cp -v ./u-boot/u-boot.img /media/boot/
```

Install Kernel and Root File System

To help new users, since the kernel version can change on a daily basis. The kernel building scripts listed on this page will now give you a hint of what kernel version was built.

```
-----  
Script Complete  
eewiki.net: [user@localhost:~$ export kernel_version=4.X.Y-Z]  
-----
```

Copy and paste that "export kernel_version=4.X.Y-Z" exactly as shown in your own build/desktop environment and hit enter to create an environment variable to be used later.

```
export kernel_version=4.X.Y-Z
```

Copy Root File System

```
~/
```

```
sudo tar xfvp ./*-*-*-armhf-*/armhf-rootfs-*.tar -C /media/rootfs/  
sync  
sudo chown root:root /media/rootfs/  
sudo chmod 755 /media/rootfs/
```

Set `uname_r` in `/boot/uEnv.txt`

```
~/
```

```
sudo sh -c "echo 'uname_r=${kernel_version}' >> /media/rootfs/boot/uEnv.  
txt"
```

Copy Kernel Image

Kernel Image:

```
~/
```

```
sudo cp -v ./armv7-multiplatform/deploy/${kernel_version}.zImage /media  
/rootfs/boot/vmlinuz-${kernel_version}
```

Copy Kernel Device Tree Binaries

```
~/
```

```
sudo mkdir -p /media/rootfs/boot/dtbs/${kernel_version}/  
sudo tar xfv ./armv7-multiplatform/deploy/${kernel_version}-dtbs.tar.gz -C  
/media/rootfs/boot/dtbs/${kernel_version}/
```

Copy Kernel Modules

```
~/
```

```
sudo tar xfv ./armv7-multiplatform/deploy/${kernel_version}-modules.tar.gz  
-C /media/rootfs/
```

File Systems Table (/etc/fstab)

```
sudo sh -c "echo '/dev/mmcblk0p2 / auto errors=remount-ro 0 1' >>  
/media/rootfs/etc/fstab"  
sudo sh -c "echo '/dev/mmcblk0p1 /boot/uboot auto defaults 0 2' >>  
/media/rootfs/etc/fstab"
```

Remove microSD/SD card

```
sync  
sudo umount /media/boot  
sudo umount /media/rootfs
```

Comments

Comments, feedback, and questions can be sent to: eewiki@digikikey.com
Please use the Digi-Key's TechForum: [TechForum](#)