BeagleBone

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

Availability

Boards:
BeagleBone 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-gnueabihf/gcc-linaro-6.4.1-2018.05-x86_64_arm-linux-gnueabihf.tar.xz
tar xf gcc-linaro-6.4.1-2018.05-x86_64_arm-linux-gnueabihf.tar.xz
export CC=`pwd`/gcc-linaro-6.4.1-2018.05-x86_64_arm-linux-gnueabihf/bin/arm-linux-gnueabihf-
```

Test Cross Compiler:

~/

```
${CC}gcc --version
arm-linux-gnueabihf-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-am335x_evm-uEnv.txt-bootz-n-fixes.patch
wget -c https://rcn-ee.com/repos/git/u-boot-patches/v2018.09-rc2/0002-U-Boot-BeagleBone-Cape-Manager.patch

patch -p1 < 0001-am335x_evm-uEnv.txt-bootz-n-fixes.patch
patch -p1 < 0002-U-Boot-BeagleBone-Cape-Manager.patch
```

Configure and Build:

~/u-boot

```
make ARCH=arm CROSS_COMPILE=${CC} distclean
make ARCH=arm CROSS_COMPILE=${CC} am335x_evm_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 Migation 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.

Mainline

Download:

~/

```
git clone https://github.com/RobertCNelson/bb-kernel
cd bb-kernel/
```

For am33x-v4.9 (Longterm 4.9.x):

~/bb-kernel/

```
git checkout origin/am33x-v4.9 -b tmp
```

For am33x-rt-v4.9 (Longterm 4.9.x + Real-Time Linux):

~/bb-kernel/

```
git checkout origin/am33x-rt-v4.9 -b tmp
```

For am33x-v4.14 (Longterm 4.14.x):

~/bb-kernel/

```
git checkout origin/am33x-v4.14 -b tmp
```

For am33x-rt-v4.14 (Longterm 4.14.x + Real-Time Linux):

~/bb-kernel/

```
git checkout origin/am33x-rt-v4.14 -b tmp
```

For am33x-v4.18 (Stable):

~/bb-kernel/

```
git checkout origin/am33x-v4.18 -b tmp
```

Build:

~/bb-kernel/

```
./build_kernel.sh
```

TI BSP

Download:

~/

```
git clone https://github.com/RobertCNelson/ti-linux-kernel-dev.git
cd ti-linux-kernel-dev/
```

For TI v4.14.x:

~/ti-linux-kernel-dev/

```
git checkout origin/ti-linux-4.14.y -b tmp
```

For TI v4.14.x: Real-Time

~/ti-linux-kernel-dev/

```
git checkout origin/ti-linux-rt-4.14.y -b tmp
```

Build:

~/ti-linux-kernel-dev/

```
./build_kernel.sh
```

Root File System

Debian 9

User	Password
debian	temppwd
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	temppwd

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=10
```

Install Bootloader:

~/

```
sudo dd if=./u-boot/MLO of=${DISK} count=1 seek=1 bs=128k
sudo dd if=./u-boot/u-boot.img of=${DISK} count=2 seek=1 bs=384k
```

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__
4M,,L,*
__EOF__
```

sfdisk <= 2.25.x

```
sudo sfdisk --unit M ${DISK} <<-__EOF__
4,,L,*
__EOF__
```

Format Partition:

```
for: DISK=/dev/mmcblk0
sudo mkfs.ext4 -L rootfs ${DISK}p1

for: DISK=/dev/sdX
sudo mkfs.ext4 -L rootfs ${DISK}1
```

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

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

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

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

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 ./bb-kernel/deploy/${kernel_version}.zImage /media/rootfs/boot
/vmlinuz-${kernel_version}
```

Copy Kernel Device Tree Binaries

~/

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

Copy Kernel Modules

~/

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

File Systems Table (/etc/fstab)

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

Networking

Edit: /etc/network/interfaces

```
sudo nano /media/rootfs/etc/network/interfaces
```

Add:

/etc/network/interfaces

```
auto lo
iface lo inet loopback

auto eth0
iface eth0 inet dhcp
```

Networking: Using a shared SD card with Multiple BeagleBone

To always enable the Ethernet interface as eth0.
Edit: /etc/udev/rules.d/70-persistent-net.rules

```
sudo nano /media/rootfs/etc/udev/rules.d/70-persistent-net.rules
```

Add:

/etc/udev/rules.d/70-persistent-net.rules

```
# BeagleBone: net device ()
SUBSYSTEM=="net", ACTION=="add", DRIVERS=="?*", ATTR{dev_id}=="0x0", ATTR
{type}=="1", KERNEL=="eth*", NAME="eth0"
```

Remove microSD/SD card

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

U-Boot Overlays

Full Documentation: readme
Any issues:

Any issues run:

```
sudo /opt/scripts/tools/version.sh
```

Enable:

/boot/uEnv.txt

```
enable_uboot_overlays=1
```

To Disable: BB-ADC:

/boot/uEnv.txt

```
disable_uboot_overlay_adc=1
```

U-Boot: override detected capes

/boot/uEnv.txt

```
uboot_overlay_addr0=/lib/firmware/.dtbo
uboot_overlay_addr1=/lib/firmware/.dtbo
uboot_overlay_addr2=/lib/firmware/.dtbo
uboot_overlay_addr3=/lib/firmware/.dtbo
```

U-Boot: disable auto-loading of detected capes

/boot/uEnv.txt

```
disable_uboot_overlay_addr0=1
disable_uboot_overlay_addr1=1
disable_uboot_overlay_addr2=1
disable_uboot_overlay_addr3=1
```

U-Boot: load 4 more un-detected capes

/boot/uEnv.txt

```
uboot_overlay_addr4=/lib/firmware/.dtbo
uboot_overlay_addr5=/lib/firmware/.dtbo
uboot_overlay_addr6=/lib/firmware/.dtbo
uboot_overlay_addr7=/lib/firmware/.dtbo
```

U-Boot: PRU Options (v4.14.x-ti)

/boot/uEnv.txt

```
uboot_overlay_pru=/lib/firmware/AM335X-PRU-RPROC-4-14-TI-00A0.dtbo
```

U-Boot: PRU Options

/boot/uEnv.txt

```
uboot_overlay_pru=/lib/firmware/AM335X-PRU-UIO-00A0.dtbo
```

U-Boot: Cape Universal

/boot/uEnv.txt

```
enable_uboot_cape_universal=1
```

Comments

Comments, feedback, and questions can be sent to: eewiki@digikey.com
Please use the Digi-Key's TechForum: TechForum