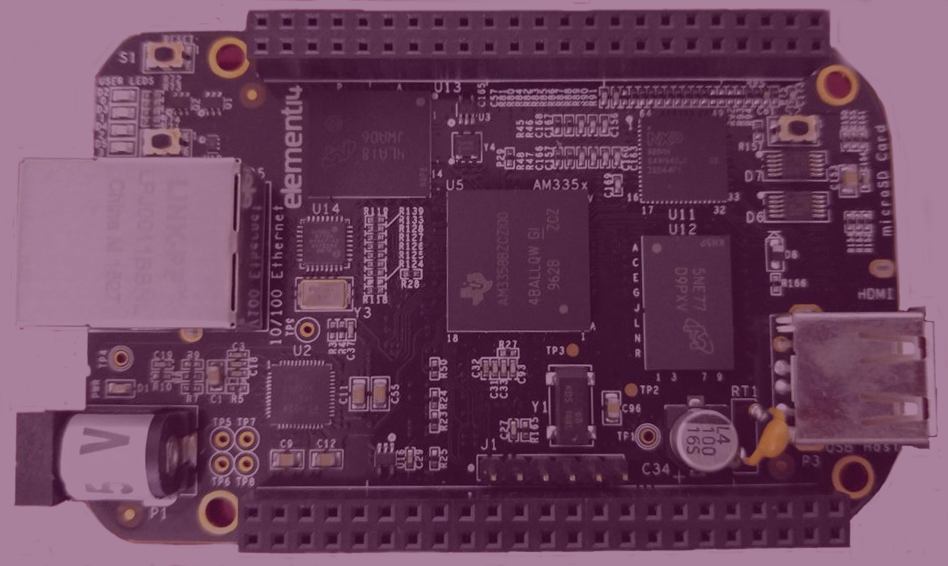
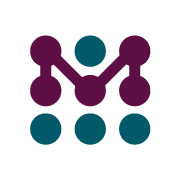
****

**YOCTO:**

**BEAGLEBONE BLACK**

**MINI-HOW**

****

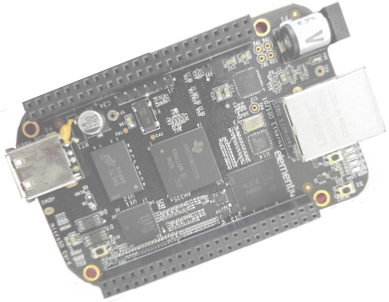


**Before You Begin**

**Beagleboard Black Meets Yocto:**

Beagleboard Black

The Beaglebone Black is a low cost, community supported computing model for developers and hobbyists. This features a AM335x 1GHz ARM[®](https://www.ti.com/product/am3358) Cortex-A8 processor,512MB DDR3 RAM, 4GB 8-bit eMMC on-board flash storage, 3D graphics accelerator, NEON floating-point accelerator and 2x PRU 32-bit microcontrollers. This also uses USB, Ethernet, HDMI and 2x 46 pin headers. The Beaglebone Black is compatible with Debian, Android, Ubuntu, Cloud 9 and much more. For more information please visit: <http://beagleboard.org/black>.



What is Yocto?

Yocto is an open source project that enables users to create custom GNU Linux systems on embedded. This project began in 2010 as a collaboration of hardware vendors. The development environment includes debuggers, Application Toolkit Generator and emulation environments. The core system recipies are available from the OpenEmbedded project. The Yocto kernel and build profiles have been community test and include architectures for:

ARM PPC

MIPS x86

x86-64 An Eclipse plugin is also available.

Covered in this Document

Setting up the Workstation

Building Images

Partition and Copy Image to SD Card

Boot from SD Card

**What You'll Need**

Beaglebone Black

Ubuntu 14.04

microSD card and adapter

Beyond that you may need a microHDMI to HDMI cable

Mini USB Cable

The Beaglebone board can be powered and controlled over USB, and uses at least one mini USB connector.

A computer with an available SD card slot and an available USB slot.

**Setting Up the Workstation**

**Ubuntu**

You will need at least 50 Gb of free disk space that is running Ubuntu 14.04 or greater

**Build Host**

BUILD HOST is based on the OpenEmbedded Project. This project uses bitbake to construct images.

Reference Build Host POKY = Bitbake + OpenEmbedded

The OpenEmbedded build system is able to run on Ubuntu that has the following versions for Git, tar, and Python.

**Dependencies – OpenEmbedded**

Git 1.7.8 or greater

tar 1.24 or greater

Python 2.7.3 or greater excluding Python 3.x, which is not supported.

**Dependencies - Build Host Packages¶**

These packages installed on the build host using an **Ubuntu System**

|  |  |
| --- | --- |
| bc | libsdl1.2-dev |
| build-essential | pkg-config |
| chrpath | socat |
| diffstat | subversion |
| gawk | texi2html |
| git | texinfo |
| libncurses5-dev | u-boot-tools |

$ sudo apt-get install bc build-essential chrpath diffstat gawk git libncurses6-dev pkg-config socat subversion texi2html texinfo u-boot-tools

**Repositories**

**YOCTO RELEASE**

Clone the Yocto Project repository poky jethro. Below is an example from an Ubuntu build host that clones the poky repository and then checks out the latest Yocto Project Release:

$ git clone -b jethro git://git.yoctoproject.org/poky.git poky-jethro

**META-OPENEMBEDDED**

Clone the OpenEmbedded repo.

$ git clone -b jethro git://git.openembedded.org/meta-openembedded

**META-BEAGLEBOARD repository**

Clone the official Beagleboard repo. Before cloning, create a sub directory and change to that directory.

mkdir bbb

cd bbb

$ git clone –b jethro git://github.com/beagleboard/meta-beagleboard

**Building Images¶**

Now that you have your system requirements in order, you can give the Yocto Project a try. This section presents steps that let you do the following:

**Initialize the Build Directory**

The build directory can be either manually created or created using the oe-init-build-env script.

1. Manually - Configure these paths to the meta layers per your standards.

2. With the script oe-init-build-env

**Customize the config files**

When using oe-init-build-env to create the build directory, some configuration files are generated in the build/conf directory.

Additionally, there are some configuration files are located in the conf directory of meta-bbb.

Copy these to the build/conf directory removing the -sample

$ cp meta-bbb/conf/local.conf-sample build/conf/local.conf

$ cp meta-bbb/conf/bblyaers.conf-sample build/conf/bblayers.conf

Editing bblayers.conf

The directory stryuction in bblayers.conf should look like this

~/poky-jethro/

meta-openembedded

~/bbb/

meta-beagleboard

build/

conf/

Editing local.conf

There are 3 variables to edit in local.conf

DL\_DIR

SSTATE\_DIR

TMP\_DIR

**Initialize the Build Environment:**

Run the oe-init-build-env environment setup script to use bitbake. This defines the OpenEmbedded build environment for the build host and will not overwrite the customized conf files.

=$ source poky-jethro/oe-init-build-env ~/bbb/build

By default the TMPDIR = OEMTMP

**Run the Build**

console-image

To build the console-image, run the command:

$bitbake console-image

Note: Depending on your workstation specifications and internet connect, this could take awhile.

Installer-image

This is an image that will run from an SD card to perform an eMMC installation.

To build the installer-image, run the command:

$bitbake installer-image

Build Errors

If you have build errors due to package download failure, clean the failed package and rerun the build.

$bitbake -c cleanstate.file

$bitbake file

Resume the full or long build

$bitbake console-image

The cleansstate command works for image recipes also.

**Copy to the SD card**

The core or long build can take some time, depending on your internet speed and hardware. There is a difference when copying a long build or a minimal build to the SD card, as well.

Partition the SD card

Partition the SD card with at least 2 partitions using gparted, fdisk, or another tool.

$ sudo apt-get install gparted

Insert the SD card into the workstation.

$ sudo gparted

Use lsblk or df -h to find the microSD card.

**Warning:** This script will format any disk on your workstation. Select the specified SD card listed when using lsblk or df -h.

**Partition 1:**

Set the “boot” flag for the first partition.

type: FAT32

size: around 30MB (that is way too much, but most SD-cards are 4GB+ anyways)

label: BOOT

flags: boot

**Partition2:**

type: ext4

size: around 200MB, or rest of SD-card

label: ROOT

After partitioning is completed, we are ready to copy the files to the SD-card.

Need to commands for copying?

Long Build

After completing the build, the following images are the most important found in the <TMPDIR>/deploy/images/beaglebone:

bootloader kernel rootfs

* **MLO-beaglebone**  
  The second stage bootloader (the first stage bootloader is implemented in ROM code on the AM335x chip and can not be altered in software)
* **u-boot-beaglebone.img**  
  The third stage u-boot bootloader (the “main” bootloader)

Install the boot loaders # cp MLO-beaglebone /media/boot/MLO # cp u-boot-beaglebone.img /media/boot/u-boot.img

* **zImage**

The Linux kernel image (zImage is a special format used with u-boot bootloader)

* **core-image-base-begalebone.tar.bz2 or something similar**  
  This archive contains the root file system.

Minimal RootFS Instructions

If you build a minimal rootfs:

* Install modules-beaglebone.tgz.
* If you used core-image-base, rootfs already includes the kernel, modules and Device tree files needed to work with The u-boot default configuration
* If using core-image-minimal rootfs, install the modules # tar x -C /media/root -f modules-beaglebone.tgz. Install the kernal uImage to /boot in rootfs and install the device tree files into rootfs /boot. Include zImage-4.4-r4am335x-boneblack-xxx.dtb
* Unmount the SD card partitions, insert the SD card into the Beaglebone, and boot the Beaglebone

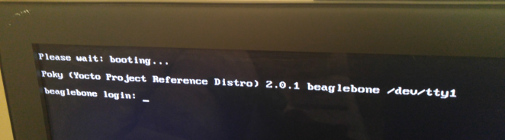
**Boot from the SD card**

The Beaglebone Black has Debian preloaded. You can flash the image to the onboard eMMC or use the SD boot. There are 4 different ways of booting up: eMMC boot, SD boot, Serial boot, and USB boot.

For SD boot, hold the S2 switch on the Beaglebone Black.

**Insert IMAGE?**

**EXIT\_Success – Conclusion**

****