Reference link:

https://www.yoctoproject.org/downloads/bsps/daisy16/beaglebone

http://free-electrons.com/training/yocto/

Preparations:

1. Beaglebone

http://hshop.vn/products/moy-tonh-beaglebone-black-rev-c

2. Micro SD card

http://memoryzone.com.vn/san-pham/micro-sd

3. Micro HDMI to HDMI cable

http://hshop.vn/products/cap-micro-hdmi-sang-hdmi

4. UART to USB:

http://hshop.vn/products/mach-chuyen-usb-uart-cp2102

Build Yocto:

0. Hint: For those who access to Ubuntu computer through SSH, creating multiple terminals can be done with tmux tool.

1. Install lab data:

Free-electrons team has prepared a set of data (kernel images, kernel configurations, root file systems and more). Download and extract its tarball from a terminal:

```
cd
wget http://free-electrons.com/doc/training/yocto/yocto-labs.tar.xz
tar xvf yocto-labs.tar.xz
```

2. Go to yocto-labs directory and install required package:

```
sudo apt-get install bc build-essential chrpath cpio diffstat gawk git
python texinfo wget
```

3. Download the krogoth version of Poky and apply a custom patch:

```
git clone git://git.yoctoproject.org/poky.git
cd $HOME/yocto-labs/poky
git checkout -b krogoth-15.0.1 krogoth-15.0.1
git am $HOME/yocto-labs/0001-Fix-lab.patch
```

4. Return to your project root directory (cd $HOME/yocto-labs/) and download the OpenEmbedded TI layer:

```
git clone git://git.yoctoproject.org/meta-ti.git

cd $HOME/yocto-labs/meta-ti

git checkout -b ti2016.03 ti2016.03

git am $HOME/yocto-labs/0001-Simplify-linux-ti-staging-recipe.patch
```

5. Set up the build environment

5.1. Export all needed variables and set up the build directory:

```
cd $HOME/yocto-labs/poky/
source oe-init-build-env
```

5.2. Go to conf directory and change the following variable:

```
MACHINE ?= "beaglebone"
```

Note: DL_DIR contains the path to download directory. Yocto will download packages from the internet and put them in it. Recycling this directory for different build time will reduce tremendous time. Default, DL_DIR will be in build directory, the same level with conf.

5.3 Add layer meta-ti so that Yocto can see this layer during building process.

```
POKY_BBLAYERS_CONF_VERSION = "2"

BBPATH = "${TOPDIR}"

BBFILES ?= ""

BBLAYERS ?= " \
  /home/pi/Yocto/yocto-labs/poky/meta \
  /home/pi/Yocto/yocto-labs/poky/meta-poky \
  /home/pi/Yocto/yocto-labs/poky/meta-yocto-bsp \
  /home/pi/Yocto/yocto-labs/meta-ti \
  "
```

6. Build your first image

```
bitbake core-image-minimal
```

Once the build finished, you will find the output images under $BUILDDIR/tmp/deploy/images/ beaglebone.

7. Set up your SD card

7.1 Create 2 partitions, that is BOOT (100MB) and ROOT (the rest of your micro SD card

Step 1: Insert your 16GB micro SD card to your USB adapter

Step 2: Insert the USB adapter to your Ubuntu computer.

Step 3: Find your SD card using following command "`sudo fdisk -l`". Please note that the following steps will erase all data on your SD card. Please be careful! Moreover, you can identify your SD card by looking at the size of it. In this example, I use 16GB micro SD card.

```
Disk /dev/sdd: 14,9 GiB, 15931539456 bytes, 31116288 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: dos
Disk identifier: 0x613de159
```

Step 4: Unmount any mounted partition, using the umount command:

```
umount /dev/sdd
```

Step 4: Launch the fdisk utitility and delete the previous partition(s); in our case, it is just one:

```
sudo fdisk /dev/sdd
Command (m for help): d
Selected partition 1
```

Step 5: Create new partition called BOOT of 100 MB and type primary:

```
Command (m for help): n
Partition type:
p primary (0 primary, 0 extended, 4 free)
e extended
Select (default p):
Using default response p
```

```
Partition number (1-4, default 1):
Using default value 1
First sector (2048-7774207, default 2048):
Using default value 2048
Last sector, +sectors or +size{K,M,G} (2048-7774207, default 7774207): +100M
```

Step 6: Create a second partition to hold rootfs. We will give all the remaining space to this partition:

```
Command (m for help): n
Partition type:
p primary (1 primary, 0 extended, 3 free)
e extended
Select (default p):
Using default response p
Partition number (1-4, default 2):
Using default value 2
First sector (67584-7774207, default 67584):
Using default value 67584
Last sector, +sectors or +size{K,M,G} (67584-7774207, default 7774207):
Using default value 7774207
```

Step 7: Make the frst partition bootable by setting the boot flag:

```
Command (m for help): a
Partition number (1-4): 1
```

Step 8: Set the frst partition as WIN95 FAT32 (LBA):

```
Command (m for help): t
Selected partition 1
Hex code (type L to list codes): c
```

Step 9: We are done with the file system modifcation. So, let us write it by issuing the w command:

```
Command (m for help): w
The partition table has been altered!
Calling ioctl() to re-read partition table.
Syncing disks.
```

Note: You might restart your Ubuntu computer or plug in and plug out your micro SD card. This will make your computer update new partition table.

Step 10: Format the frst partition as FAT, using the following command. We will set the label as BOOT so that we know what directory it will be mounted to by udisks:

```
sudo mkfs.vfat -n "BOOT" /dev/sdd1
```

Step 11: Format the second partition as an ext4 flesystem, using the following command. The label for this is set to ROOT, as it will contain the extracted image of rootfs.

```
sudo mkfs.ext4 -L "ROOT" /dev/sdd2
```

8. Copying images to the SD card

The following assumes the SD card partitions 1 and 2 are mounted at /media/boot and /media/root respectively. Removing the card and reinserting it will do just that on most modern Linux desktop environments.

8.1 Install the boot loaders

```
cp MLO-beaglebone /media/boot/MLO
cp u-boot-beaglebone.img /media/boot/u-boot.img
```

```
pi@raspberrypi04:~/boot$ ls
MLO  u-boot.img
```

8.2 Install the root filesystem

```
tar xvf core-image-minimal-beaglebone.tar.gz -C /media/root
```

8.3 Install the modules

```
tar xvf modules-beaglebone.tgz -C /media/root
```

8.4 Install the kernel zImage into /boot directory of rootfs

```
cp zImage-beaglebone.bin /media/root/boot/zImage
```

8.5 Install device tree (DTB) files into /boot directory of rootfs

```
cp uImage-am335x-bone.dtb /media/root/boot/am335x-bone.dtb
cp uImage-am335x-boneblack.dtb /media/root/boot/am335x-boneblack.dtb
```

```
root@beaglebone:/boot# ls
am335x-bone.dtb       am335x-boneblack.dtb  zImage
```

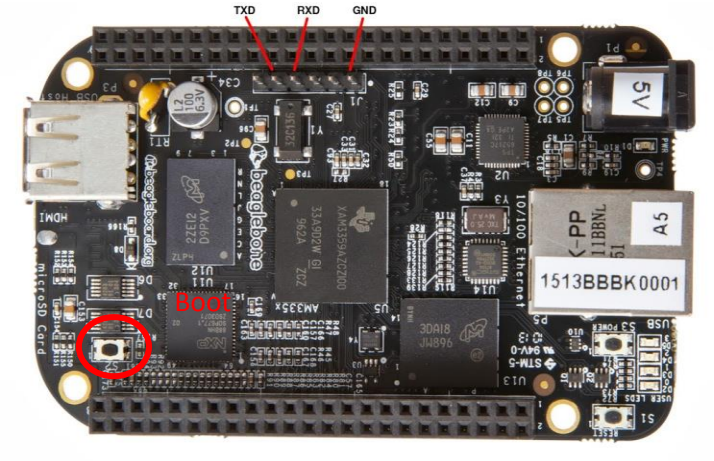
8.6 Unmount the SD partitions, insert the SD card into the Beaglebone, and boot the Beaglebone

9. Setting up UART

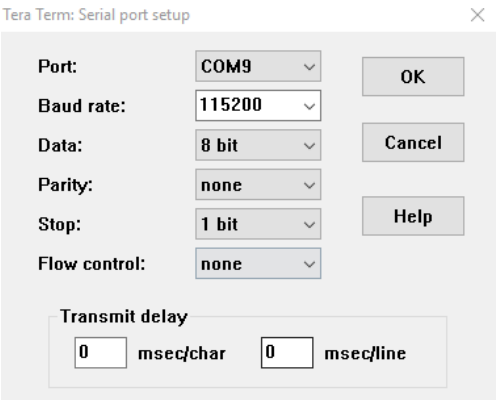
Connect your “USB to UART” device to Windows machine. Then, connect its TX, RX and GND to beagle bone as the below format. Note: TX connects to RX and RX connects to RX.

USB-TTL is connected to the J1 connector of BeagleBone in the following formation:

J1 Pin	USB TTL Function
1	GNG Ground
4	RXL
5	TXL



10. On Windows, install TeraTerm software. Then, plug your “USB to UART” device. Configure as following format:



11. Hold boot button, then connect it to your Windows computer. Then release this button. You should see these messages:

```
Poky (Yocto Project Reference Distro) 2.1.1

beaglebone login:
Poky (Yocto Project Reference Distro) 2.1.1

beaglebone login:
```