



# Learning Report on Embedded Linux



GLOBAL  
ENGINEERING  
ACADEMY

Genesis



*L&T Technology Services*



## Document History

Ver. No.	Release Date	Prepared by.	Comments
1.0	30 Oct 20	Naga Akhil ES	Template
1.1	01 Nov 20	Naga Akhil ES	Added all content
1.2	.	.	
1.3	.	.	

## Details

Name	PS No.	Email Id
Naga Akhil ES	99002623	Nagaakhil.ES@ltts.com

## Contents

Activity 1 - BBB board Configuration .....	4
Introduction.....	4
About BBB board.....	4
Part details .....	4
Communication Setup .....	5
1. Installations .....	5
2. Settings .....	5
Activity 2 - Development board Market Survey.....	6
Differences between Raspberry pie, Dragon, imx7 Sabre, BBB .....	6
Activity 3 - Different BBB Boards and its Evolution .....	7
Differences in different versions of BBB .....	7
Evolution of the Beagle bone .....	8
Release Notes .....	8
Activity 4 - Document various peripherals on pin expansion header .....	11
Activity 5 - Testing of Bootloaders .....	12
Activity 6 - u-boot Shell .....	14
1. ROM bootloader(RBL).....	14
2. Memory Loader/Secondary program loader(MLO/SPL) Job.....	14
Activity 7 - Linux boot sequence .....	17
Bootting Options.....	17
Boot Order.....	17
Activity 8 - Make uEnv.txt to Boot from MMC0 or MMC 1 .....	19
Activity 9 - Increase the AUTOLOAD timings.....	20
1. Cross Compiler(Linux Host) .....	20
2. Uboot Compilation.....	20
Activity 10 - Busybox "Dynamic" Compilation .....	22
Features.....	22
Steps.....	22

## Activity 1 - BBB board Configuration

### Introduction

#### About BBB board

- It's an open source h/w, s/w platform
- It a single piece of circuit board, comprises most of the personal Computer hardware/software components

#### **Features:**

- Processor: [AM335x 1GHz ARM® Cortex-A8](#)
- 512MB DDR3 RAM
- 4GB 8-bit eMMC on-board flash storage
- 3D graphics accelerator
- NEON floating-point accelerator
- 2x PRU 32-bit microcontrollers
- Connectivity
  - USB client for power & communications
  - USB host
  - Ethernet
  - HDMI
  - 2x 46 pin headers
- Software Compatibility
  - Debian
  - Android
  - Ubuntu
  - Cloud9 IDE on Node.js w/ BoneScript library

#### Part details

- SOC(Systems on chips) is: AM3358BZCZ100 on REV 'C' BBB Board and part number is U5
- Embedded MMC (eMMC) part No: is U13.
- DDR3 part no: is U12. During booting the boot images will get loaded to this RAM from other memories and will execute from here
- UART part no: J1- 8N1 no parity and one stop bit,115200 baudrate
- Boot loader part no : s2
- Power Button part no: is s3
- Mini port (power):part no: P6
- Memory card slot part No: P10
- MMC0 is connected to microSD card
- MMC1 is Connected to eMMC
- DC Adapter:1A current and 5V voltage.

## Communication Setup

Minicom is used as serial communicate program with BBB board using serial debug interface

\* <https://help.ubuntu.com/community/Minicom>

### 1. Installations

1. Update the package index

```
`sudo apt-get update`
```

2. Install minicom deb package

```
`sudo apt-get install minicom`
```

### 2. Settings

1. Connect USB to TTL

1. Connect the USB side of the TTL cable to your computer
2. Connect the wires to J1 headers on your BeagleBone Black
  1. Black wire to Pin 1
  2. Green wire to Pin 4
  3. White wire to Pin 5

2. Find Port Details

- ```
`dmesg | grep tty`
```

Response: 

```
`[ 1977.652735] usb 1-1.4: pl2303 converter now attached to ttyUSB0`
```

- ```
`ls /dev/tty*`
```

3. minicom settings

1. 

```
`sudo minicom -s`
```

2. serial port setup

A. Serial Device : `/dev/ttyUSB0`

E. bps - 115200 8N1

F- Hardware Flow Control: No

G- Software Flow Control: No

3. save setup as dfl

4. exit by esc

## Activity 2 - Development board Market Survey

### Differences between Raspberry pie, Dragon, imx7 Sabre, BBB

Raspberry Pi	Dragon	IMX7	BBB
Raspberry Pi consists of single board computers	The DragonBoard 410c is the first development board based on a Qualcomm APQ8016e application processor	NXP delivers the Smart Application Blueprint for Rapid Engineering (SABRE) board based on i.MX7 dual applications processor	Beaglebone black is a low power open source single board computer
CPU- 700 MHz Low Power ARM1176JZ-F Applications Processor	CPU: Quad-core ARM® Cortex® A53 at up to 1.2 GHz per core with both 32-bit and 64-bit support	CPU - Utilizes both the Arm Cortex-A7 and Cortex-M4 cores	CPU- 1 GHz ARM Cortex-A8
RAM: 1024 MB Memory- 512MB SDRAM	Memory/storages: 1GB LPDDR3 533MHz / 8GB eMMC 4.5 / SD3.0 (UHS-I)	Memory - 1 GB DDR3, 533 MHz	DDR3 Memory: 512MB
Wifi chip and Bluetooth low energy.	It features advanced processing power, Wi-Fi, Bluetooth connectivity, and GPS,	802.11 a/b/g/n/ac Wi-Fi® on board Bluetooth V4.0 + EDR on board	Beagle bone provides the onboard storage of 4GB.
Chip enables 28 GPIO pins plus 12 power and ground pins	I/O Interfaces: HDMI Full-size Type A connector, one micro USB, two USB 2.0.	Expansion port for EPDC Port for Parallel LCD display Port for HDMI display Port for MIPI DSI display Port for MIPI CSI CMOS sensor (camera)	Provides 92 expansion pins with 46 pin female connectors on both sides of a board
Raspberry Pi does not have a micro USB cable and also does not have a pre-installed operating system.	The board can be made compatible with Arduino using an add-on mezzanine board	Dual Ethernet on board 1 SD socket for boot code 1 USB host connector 1 micro USB OTG connector	Beagle comes with a mini cable which helps to supply power to the system. It can boot from the onboard storage.
The architecture of Raspberry is ARMv6	Qualcomm IZat location technology Gen8C.	Enhanced security and power optimization	Beagle uses ARMv7



## Activity 3 - Different BBB Boards and its Evolution

### Differences in different versions of BBB

Parameter	PocketBeagle	BeagleBoard-X15	BeagleBone Black	BeagleBone	BeagleBoard-xM	BeagleBoard
Release Date:	September 21, 2017	23 September 2016	April 23, 2013	October 31, 2011	September 14, 2010	July 28, 2008
SoC	OSD3358-SM	Sitara AM5728	AM3358/9	AM3358/9	DM3730	<a href="#">OMAP3530</a>
CPU	Sitara AM3358 ARM Cortex-A8	<a href="#">Dual ARM Cortex-A15 + Dual ARM M4 (212 MHz)</a> + Quad PRU (200 MHz)	<a href="#">Cortex-A8</a> + Dual PRU (200 MHz)	<a href="#">Cortex-A8</a> + Dual PRU (200 MHz)	<a href="#">Cortex-A8</a> + Dual PRU (200 MHz)	<a href="#">Cortex-A8</a> + Dual PRU (200 MHz)
Frequency(MHz)	1000	1500	1000	720	1000	720
GPU	<a href="#">PowerVR SGX530</a>	<a href="#">Dual PowerVR SGX544</a>	<a href="#">PowerVR SGX530</a> (200 MHz)	<a href="#">PowerVR SGX530</a> (200 MHz)	<a href="#">PowerVR SGX530</a> (200 MHz)	<a href="#">PowerVR SGX530</a> (200 MHz)
DSP	N/A	Dual <a href="#">TMS320C66x[41]</a> (700 MHz)	N/A	N/A	<a href="#">TMS320C64x+[44]</a> (800 MHz)	<a href="#">TMS320C64x+[41]</a> (520 MHz)
Onboard storage:	4KB of EEPROM, <a href="#">microSD</a> card slot	8-bit <a href="#">eMMC</a> 4 GB, <a href="#">microSD</a> card	8-bit <a href="#">eMMC</a> <a href="#">microSD</a> card 3.3 V Supported	<a href="#">microSD</a> card 3.3 V Supported	<a href="#">microSD</a> card Supported	256MB NAND Flash, SD/MMC card
Onboard network:	N/A	Dual <a href="#">Gigabit Ethernet</a>	<a href="#">Fast Ethernet</a> (MII based)	<a href="#">Fast Ethernet</a> (MII based)	<a href="#">Fast Ethernet</a> (via USB hub with Ethernet)	N/A
USB ports:	1 x Micro USB Type B	3 x USB 3.0 Type A Host 4 x USB 2.0 Host 1 x Micro USB Type B	1 x Standard A host port 1x mini B device port	1 x Standard A host port (direct). 1x mini B device port (via hub)	4 x Standard A host port (via hub with Ethernet). 1x mini AB OTG port (direct)	1 x Standard A host port (direct). 1x mini AB OTG port (direct)
Memory (SDRAM):	512 <a href="#">MiB DDR3</a>	2048 <a href="#">MiB DDR3L</a>	512 <a href="#">MiB DDR3</a>	256 <a href="#">MiB DDR2</a>	512 <a href="#">MiB DDR2</a>	128 <a href="#">MiB</a> (rev B) <a href="#">DDR</a> 256 <a href="#">MiB</a> (rev C+) <a href="#">DDR</a>
Video outputs:	none	<a href="#">HDMI</a> , LCD via Expansion	<a href="#">Micro-HDMI</a> , cape add-ons	cape add-ons	<a href="#">DVI-D,S-Video</a>	<a href="#">DVI-D,S-Video</a>
Audio outputs:	none	<a href="#">HDMI</a> , AIC3104 (Stereo In/Out)	<a href="#">Micro-HDMI</a> , cape add-ons	cape add-ons	<a href="#">3.5mm audio jack</a>	<a href="#">3.5mm audio jack</a>
Power ratings:	150 mA @ 5 V	210-460 mA @5 V	210-460 mA @5 V	300-500 mA @5 V		
Power source:	micro USB port or I/O pins	2.5 mm x 5.5 mm 12 V jack	Mini USB or 2.1 mm x 5.5 mm 5 V jack	Mini USB or 2.1 mm x 5.5 mm 5 V jack	Mini USB or 2.1 mm x 5.5 mm 5 V jack	Mini USB or 2.1 mm x 5.5 mm 5 V jack
Low-level peripherals:	3x <a href="#">UART</a> , 4x PWM, 2x <a href="#">SPI</a> , 2x <a href="#">I<sup>2</sup>C</a> , 2x <a href="#">CAN bus</a>	7x <a href="#">UART</a> , LCD, GPMC, 1x <a href="#">SPI</a> , 1x <a href="#">I<sup>2</sup>C</a> , 1x <a href="#">CAN bus</a>	4x <a href="#">UART</a> , 8x PWM, LCD, GPMC, MMC1, 2x <a href="#">SPI</a> , 2x <a href="#">I<sup>2</sup>C</a> , A/D Converter, 2x <a href="#">CAN bus</a> , 4 Timers	4x <a href="#">UART</a> , 8x PWM, LCD, GPMC, MMC1, 2x <a href="#">SPI</a> , 2x <a href="#">I<sup>2</sup>C</a> , A/D Converter, 2x <a href="#">CAN bus</a> , 4 Timers, FTDI USB to serial, JTAG via USB	McBSP, DSS, <a href="#">I<sup>2</sup>C</a> , <a href="#">UART</a> , LCD, McSPI, PWM, JTAG, camera interface	McBSP, DSS, <a href="#">I<sup>2</sup>C</a> , <a href="#">UART</a> , McSPI, PWM, <a href="#">JTAG</a>

## Evolution of the Beagle bone

### Release Notes

#### **Revision C (Production Version)**

This revision increases the eMMC from 2GB to 4GB. We are making this change for several reasons:

- 1) Complaints from the community about lack of space left in the eMMC.
- 2) For those worried about their eMMC wearing out, the added space will help in the area of moving the data around to prevent wear out. Assuming of course you don't try and use it all.
- 3) Concerns over the long-term availability of the 2GB device. 4GB is currently the low end of the offering. This also gives us two sources.

We are planning a price increase for the Rev C somewhere between \$5 and \$15. We are working to figure out where it needs to be. This is for several reasons:

- 1) To cover the increased cost of the 4G devices.
- 2) Currently there is \$0 margin on these boards which limits our ability to bring more manufacturing capacity on line. Added some margin allows us to find more capacity.
- 3) Without margin, we cannot respond to component price increases due to market forces. This is of particular concern in the area of NAND and DDR3. We have been successful in fighting back some increases, but we don't know if that will continue.

#### **Revision B**

This version moves to the AM3358BZCZ100 processor as we are no longer able to get the limited production version of the AM3359AZCZ100. No changes in features or operation of the board resulted from this change.

#### **Revision A6A**

No changes in features or operation of the board.

- 1) Added optional zero ohm resistor to tie GND\_OSC1 to system ground.
- 2) Changed C106 to a 1uF capacitor.
- 3) Changed C24 to a 2.2uF capacitor. This extends the reset signal to solve an issue where some boards would not boot on power up.
- 4) Removed R9 and installed R8. This change was based on an alert we just received from TI that there is a power sequencing issue with the TPS65217C power management IC and the power sequencing is incorrect. This change connects the VDDS rail to the VRTC rail. We do not believe at this time, that the issue is causing any issues with the boards, but we want to comply with the directive.



**Revision A6**

No changes in features or operation of the board.

- 1) Based on notification from TI, in random instances there could be a glitch in the SYS\_RESETh signal from the processor where the SYS\_RESETh signal was taken high for a momentary amount of time before it was supposed to. To prevent this, the signal was ORed with the PORZh (Power On reset).
- 2) Noise issues were observed in other designs where the clock oscillator was getting hit due to a suspected issue in ground bounce. A zero ohm resistor was added to connect the OSC\_GND to the system ground.
- 3) Moved the enable for the VDD\_3V3B regulator to VDD\_3V3A rail. Change was made to reduce the delay between the ramp up of the 3.3V rails. No evidence of this being an issue, but it really needs to be as close to the same as possible.

**Revision A5C**

Production had some fallout of boards when running the HDMI tests in the previous production run. Resistor values were tweaked to improve the test results.

No changes in features or operation of the board.

- 1) Changed R46, R47, R48 to a 0 ohm.
- 2) Changed R45 to a 22 Ohm.

**Revision A5B**

- 1) Updated the PCB to incorporate the modification that was being done on Rev A5A. There is NO DIFFERENCE AT ALL in functionality between REV A5A and REV A5B.
- 2) Made the LEDs dimmer for those that could not sleep due to the brightness of the LEDs.

**Revision A5A**

- 1) Boards are built using the XAM3359AZCZ100 processor.
- 2) PCB Change...LCD noise issue was resolved by adding 47pf bypass caps on some of the LCD signals.
- 3) PCB Change...Added access to four battery charger signals on the TPS65217 (TS=Temperature Sense, BAT=Battery connection, BATT\_SENSE=Battery voltage pin, GND=Ground). Pins are not populated but the four signals are in a 2x2 .1x.1 spacing.
- 4) PCB Change...Added a power button which allows for wake up, power down, and sleep options. It also provides the ability to alert the processor before powering down to provide an orderly shutdown. It is expected that SW will be used in conjunction with the switch to control the various power modes and transitions from one to the other. By holding the button down for 8 seconds, it will force a power down of the board.
- 5) Added a 100K pull down resistor from J1 pin 1 to J1 pin 4 to fix the unterminated serial port issue.

**Revision A4B**

Added a 100K pull down resistor between pins 1 and 4 of J1 to fix the serial port issue.

**Revision A4A**

Incorporated the capacitors to fix the noise issue on the display

**Revision A4**

First prototype release version of the board. Limited distribution. One notable issue here is that the board has an AM3352 processor instead of an AM3359, despite how the part is marked. Part was mismarked as an AM3359. The SGX and PRU are not operational.

## Activity 4 - Document various peripherals on pin expansion header

Github link :

Excel for Pin expansion header of BBB and the various peripherals of Bone

[https://github.com/99002623/Notebook/tree/main/Embedded\\_Linux/Pin\\_Extension\\_header\\_of\\_BBB](https://github.com/99002623/Notebook/tree/main/Embedded_Linux/Pin_Extension_header_of_BBB)

## Activity 5 - Testing of Bootloaders

Bootloaders were tested using SD card booting and the following steps was followed:

1. Make SD Card Bootable
  1. Install Gparted
    1. Update the package index `sudo apt-get update` `sudo apt-get upgrade`
    2. Install software to make pendrive bootable `sudo apt-get install gparted`
  2. Format SD Card
  3. Create New partition for boot files
    - Label: BOOT
    - file system: fat16 (minimum 2gb)
  4. Create New Partition for linux root files
    - Label: ROOTFS
    - file system: ext3/ext4
  5. Apply all operations
  6. set boot flag for BOOT partition
2. copy Image files to BOOT partition
  1. MLO
  2. u-boot.img
  3. uEnv.tsr
3. Copy ROOT files to ROOTFS partition
  1. open ROOT files directory in terminal
  2. `sudo cp -r * /media/user/ROOTFS`
4. Insert SD card to BBB board and
5. connect serial ttl cable to pc and open terminal -> run minicom
6. boot the board from SD card using S2 switch

### *The Serial Console Output from the Stage-1 Bootloader*

```
U-Boot SPL 2013.04-rc1-14237-g90639fe-dirty (Apr 13 2013 - 13:57:11)
musb-hdrc: ConfigData=0xde (UTMI-8, dyn FIFOs, HB-ISO Rx,
↳HB-ISO Tx, SoftConn)
musb-hdrc: MHDRC RTL version 2.0
musb-hdrc: setup fifo_mode 4
musb-hdrc: 28/31 max ep, 16384/16384 memory
USB Peripheral mode controller at 47401000 using PIO, IRQ 0
musb-hdrc: ConfigData=0xde (UTMI-8, dyn FIFOs, HB-ISO Rx,
↳HB-ISO Tx, SoftConn)
musb-hdrc: MHDRC RTL version 2.0
musb-hdrc: setup fifo_mode 4
musb-hdrc: 28/31 max ep, 16384/16384 memory
USB Host mode controller at 47401800 using PIO, IRQ 0
OMAP SD/MMC: 0
mmc_send_cmd : timeout: No status update
reading u-boot.img
reading u-boot.img
```

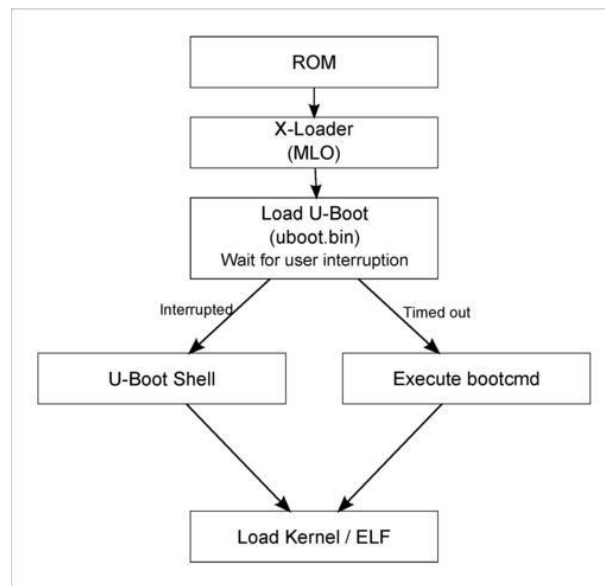
## ***The Serial Console Output from the Stage-2 Bootloader***

U-Boot 2013.04-rc1-14237-g90639fe-dirty (Apr 13 2013 - 13:57:11)

```
I2C:   ready
DRAM:  512 MiB
WARNING: Caches not enabled
NAND:  No NAND device found!!!
0 MiB
MMC:   OMAP SD/MMC: 0, OMAP SD/MMC: 1
*** Warning - readenv() failed, using default environment

musb-hdrc: ConfigData=0xde (UTMI-8, dyn FIFOs, HB-ISO Rx,
↳HB-ISO Tx, SoftConn)
musb-hdrc: MHDRC RTL version 2.0
musb-hdrc: setup fifo_mode 4
musb-hdrc: 28/31 max ep, 16384/16384 memory
USB Peripheral mode controller at 47401000 using PIO, IRQ 0
musb-hdrc: ConfigData=0xde (UTMI-8, dyn FIFOs, HB-ISO Rx,
↳HB-ISO Tx, SoftConn)
musb-hdrc: MHDRC RTL version 2.0
musb-hdrc: setup fifo_mode 4
musb-hdrc: 28/31 max ep, 16384/16384 memory
USB Host mode controller at 47401800 using PIO, IRQ 0
Net:   <ethaddr> not set. Validating first E-fuse MAC
cpsw, usb_ether
Hit any key to stop autoboot:  0
```

## Activity 6 - u-boot Shell



### 1. ROM bootloader(RBL)

1. stack setup
2. watchdog timer configuration(3minutes)
3. system clock configuration using pll
4. Booting
  1. checks boot devices list based on SYSBOOT reg
  2. configure booting device
  3. search & loads SPL/MLO to internal SOC Ram
  4. executes SPL/MLO from internal SOC Ram

### 2. Memory Loader/Secondary program loader(MLO/SPL) Job

rbl + mlo(bootloader - process of loading boot files). This is also known as X-loader

1. UART Initialization for printing debug messages
2. ppl modification to desired value
3. Configures communication between the SOC and DRAM
4. search and Load u-boot.img to DRAM
5. runs u-boot.img

### 3. u-boot jobs

booting - process of loading kernel to main memory

1. initialize peripherals
2. loads Linux kernel Image(ROOTFS/boot/uBoot) to DRAM
3. Passing boot arguments

## Manual Booting from SD Card

0. Enter manual booting by clicking space
1. Write uEnv.txt file console=ttyO0,115200n8 ipaddr=192.168.7.2 serverip=192.168.7.1  
loadaddr=0x82000000 fdtaddr=0x88000000
  1. Write Commands
    1. Load ulmage file loadfromsd=load mmc 0:2 \${loadaddr} /boot/ulmage
    2. Load .dtb file load mmc 0:2 \${fdtaddr} /boot/am335x-boneblack.dtb
    3. get Logs linuxbootargs=setenv bootargs console=\${console}  
root=/dev/mmcblk0p2 rw
  2. Run Commands uenvcmd=setenv autoload no; run loadfromsd; run linuxbootargs;  
bootm \${loadaddr} - \${fdtaddr}
2. Load uEnv file loady
3. Import env variable env import -t loaded location other Cmds printenv iminfo help
4. Boot to kernel run bootcmd or boot

## The Serial Console Output from the Stage-2 Bootloader and Kernel

```

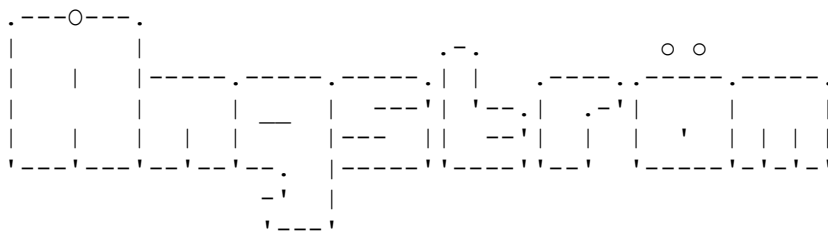
gpio: pin 53 (gpio 53) value is 1
Card did not respond to voltage select!
.
.
.
gpio: pin 54 (gpio 54) value is 1
SD/MMC found on device 1
reading uEnv.txt
58 bytes read in 4 ms (13.7 KiB/s)
Loaded environment from uEnv.txt
Importing environment from mmc ...
Running uenvcmd ...
Booting the bone from emmc...
gpio: pin 55 (gpio 55) value is 1
4215264 bytes read in 778 ms (5.2 MiB/s)
gpio: pin 56 (gpio 56) value is 1
22780 bytes read in 40 ms (555.7 KiB/s)
Booting from mmc ...
## Booting kernel from Legacy Image at 80007fc0 ...
   Image Name:   Angstrom/3.8.6/beaglebone
   Image Type:   ARM Linux Kernel Image (uncompressed)
   Data Size:    4215200 Bytes = 4 MiB
   Load Address: 80008000
   Entry Point:  80008000
   Verifying Checksum ... OK
## Flattened Device Tree blob at 80f80000
   Booting using the fdt blob at 0x80f80000
   XIP Kernel Image ... OK
OK
   Using Device Tree in place at 80f80000, end 80f888fb

Starting kernel ...

Uncompressing Linux... done, booting the kernel.
```



```
[ 0.106033] pinctrl-single 44e10800.pinmux: prop pinctrl-0
↳index 0 invalid phandle
.
.
.
[ 9.638448] net eth0: phy 4a101000.mdio:01 not found on slave 1
```



The Angstrom Distribution beaglebone tty00

Angstrom v2012.12 - Kernel 3.8.6

beaglebone login:

## Activity 7 - Linux boot sequence

### Booting Options

The AM335x SOC boots from the following sources

- 1) NAND Flash
- 2) NOR Flash (eXecute In place, XIP)
- 3) USB
- 4) eMMC
- 5) SD card
- 6) Ethernet(TFTT protocol)
- 6) UART
- 7) SPI

That means, you can keep the boot images in any of the above memory or peripheral and you can able to boot this SOC.

### Boot Order

When S2 Released (SYSBOOT[4:0] = 11100)

1. MMC1 (eMMC)
2. MMC0 (SD card)
3. UART0
4. USB0

When S2 pressed (SYSBOOT[4:0] = 11000)

1. SPI0
2. MMC0 (SD card)
3. USB0
4. UART0

### *eMMC Boot(MMC1)*

- eMMC is connected over MMC1 interface,
- This is the fastest boot mode possible,
- This is the default boot mode.
- As soon as you reset the board, the board start booting from loading the images stored in the eMMC.
- If no proper boot image is found in the eMMC, then Processor will automatically try to boot from the next device on the list.

### *SD boot(MMC0)*

- sd card connector at MMC0 interface
- If the default (eMMC) boot mode fails, then it will try to boot from the SD card
- If you press S2 and then apply the power, then
  1. the board will try to boot from the SPI
  2. if nothing is connected to SPI, it will try to boot from the MMC0(SD card)
  3. we can use SD card boot to flash boot images on the eMMC
  4. boot through sd card
  5. write new images to eMMC
  6. reset the board, so that your board can boot using new images stored in the eMMC.

**Serial Port**

In this mode, the ROM code of the SOC will try to download the boot images from the serial port

**USB Port**

- This is booting through usb stick!
- You would have booted your PC through the usb stick. What you do is, you restart the PC, then press bios button to put the PC in to bios mode, there you select boot from usb, right? so It is very similar, when you reset the board, you can make your board to boot from the USB stick.

**UART Port**

- used on UART0(115200 8n1)
- Any serial communication program will work
- uses x-modem protocol to load the boot image
  7. `loadx DRAMaddress` optional only for specific address loading
  8. `(cntrl + A) + S`
  9. select `x-modem`
  10. select file by navigation
  11. double space to go in and single space + enter to select

## Activity 8 - Make uEnv.txt to Boot from MMC0 or MMC 1

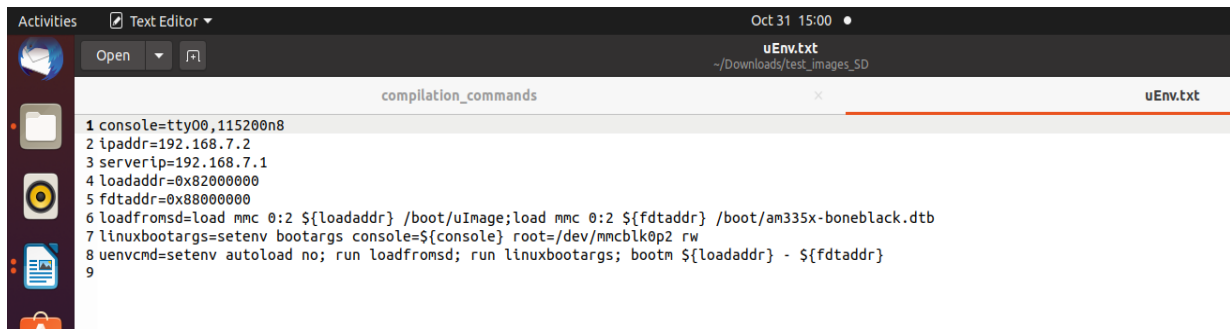
### 1. Env variable Defining

1. console=ttyO0,115200n8
2. ipaddr=192.168.7.2
3. serverip=192.168.7.1
4. loadaddr=0x82000000
5. fdtaddr=0x88000000

### 2. Write Commands

1. Load ulmage file loadfromsd=load mmc 0:2 \${loadaddr} /boot/ulmage
2. Load .dtb file load mmc 0:2 \${fdtaddr} /boot/am335x-boneblack.dtb
3. get Logs linuxbootargs=setenv bootargs console=\${console} root=/dev/mmcblk0p2 rw

### 3. Run Commands uenvcmd=setenv autoload no; run loadfromsd; run linuxbootargs; bootm \${loadaddr} - \${fdtaddr}



The screenshot shows a text editor window titled "uEnv.txt" with the following content:

```
1 console=ttyO0,115200n8
2 ipaddr=192.168.7.2
3 serverip=192.168.7.1
4 loadaddr=0x82000000
5 fdtaddr=0x88000000
6 loadfromsd=load mmc 0:2 ${loadaddr} /boot/uImage;load mmc 0:2 ${fdtaddr} /boot/am335x-boneblack.dtb
7 linuxbootargs=setenv bootargs console=${console} root=/dev/mmcblk0p2 rw
8 uenvcmd=setenv autoload no; run loadfromsd; run linuxbootargs; bootm ${loadaddr} - ${fdtaddr}
9
```

## Activity 9 - Increase the AUTOLoad timings

### 1. Cross Compiler(Linux Host)

1. Download arm cross toolchain for your Host machine [gcc-linaro-5.5.0-2017.10-x86\\_64\\_arm-linux-gnueabi/f/tar.xz](https://gcc-linaro-5.5.0-2017.10-x86_64_arm-linux-gnueabi/f/tar.xz)
2. Install Development Tools
  - o `sudo apt install build-essential`
  - o `sudo apt install bison`
  - o `sudo apt install flex`
  - o `sudo apt-get install libncurses5-dev libncursesw5-dev`
  - o `sudo apt-get install -y u-boot-tools`
  - o `sudo apt install lzop`
3. settings export path of the cross compilation toolchain
  1. open bashrc code `~/.bashrc`
  2. update path in .bashrc file `export PATH=/home/user/Documents/gcc-linaro-5.5.0-2017.10-x86_64_arm-linux-gnueabi/f/bin:$PATH` or `export PATH=$PATH:/home/user/Documents/gcc-linaro-5.5.0-2017.10-x86_64_arm-linux-gnueabi/f/bin`
  3. reload bashrc `source /home/user/.bashrc`
  4. arm and couples of times TAB button to see the files

### 2. Uboot Compilation

Building bootloader u-boot.img and u-boot-spl.bin(MLO) 0. Download u-boot u-boot.img raw c files

1. Clean old object files `make ARCH=arm CROSS_COMPILE=arm-linux-gnueabi/ distclean`
2. Generate Default Configurations `make ARCH=arm CROSS_COMPILE=arm-linux-gnueabi/ am335x_boneblack_defconfig`
3. run menuconfig, if you want to do any settings other than default configuration . `make ARCH=arm CROSS_COMPILE=arm-linux-gnueabi/ menuconfig`
4. Change the delay time in the GUI
5. To generate bootloader `make ARCH=arm CROSS_COMPILE=arm-linux-gnueabi/ -j4` To Cores nproc(4 core machine)
6. After the compilation, start booting up BBB using serial booting method.
7. Upload the newly created U-boot.img instead of the old one.

```

Compiled on Dec 23 2019, 02:06:26.
Port /dev/ttyUSB0, 11:50:31

Press CTRL-A Z for help on special keys

CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC |      |
U-Boot SPL 2017.03-00270-g5cf618e (Mar 28 2017 - 17:14:21)
Trying to boot from UART
CCxyzModem - CRC mode, 2892(SOH)/0(STX)/0(CAN) packets, 8 retries
Loaded 369972 bytes

U-Boot 2017.05-rc2 (Oct 24 2020 - 11:33:09 +0530)

CPU   : AM335X-GP rev 2.0
I2C:   ready
DRAM:  512 MiB
MMC:   OMAP SD/MMC: 0, OMAP SD/MMC: 1
*** Warning - bad CRC, using default environment

<ethaddr> not set. Validating first E-fuse MAC
Net:   cpsw, usb_ether
Press SPACE to abort autoboot in 10 seconds
=>
```

## Activity 10 - Busybox "Dynamic" Compilation

### Features

- This is used to generate one executable file for which all other files report
- This is used to generate the minimal rootfs with kernel files ie RFS\_Static
- Compilation utility by menu gui
  - kernel optimization
  - kernel with static or dynamic libraries

### Steps

1. Install Busybox [link](#)
2. Open terminal in the busy box directory
3. Clean old files `make ARCH=arm CROSS_COMPILE=arm-linux-gnueabihf- distclean`
4. generate the default configuration file `make ARCH=arm CROSS_COMPILE=arm-linux-gnueabihf- defconfig`
5. Change menu settings
  1. `make ARCH=arm CROSS_COMPILE=arm-linux-gnueabihf- menuconfig`
  2. change busybox settings->Build Shared Libraries select and save
6. generate the ROOT File system `make ARCH=arm CROSS_COMPILE=arm-linux-gnueabihf- CONFIG_PREFIX=/home/user/software/RFS_Static install -j4`
7. Check the files generated in the path