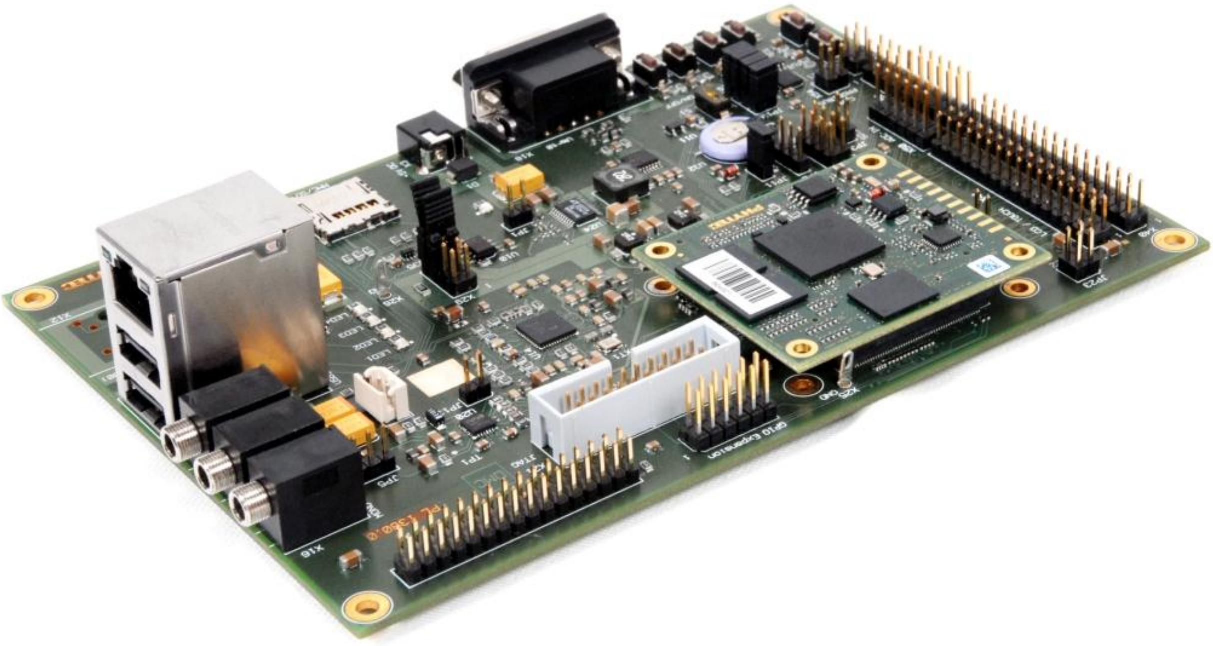


OpenBoard-AM3359 Software Development kit for Linux

Release 1.0
January, 2013

In this manual copyrighted products are not explicitly indicated. The absence of the trademark (™) and copyright (©) symbols does not imply that a product is not protected. Additionally, registered patents and trademarks are similarly not expressly indicated in this manual.

The information in this document has been carefully checked and is believed to be entirely reliable. However, PHYTEC Embedded Pvt. Ltd. assumes no responsibility for any inaccuracies. PHYTEC Embedded Pvt. Ltd neither gives any guarantee nor accepts any liability whatsoever for consequential damages resulting from the use of this manual or its associated product. PHYTEC Embedded Pvt. Ltd reserves the right to alter the information contained herein without prior notification and accepts no responsibility for any damages that might result.

Additionally, PHYTEC Embedded Pvt. Ltd offers no guarantee nor accepts any liability for damages arising from the improper usage or improper installation of the hardware or software. PHYTEC Embedded Pvt. Ltd further reserves the right to alter the layout and/or design of the hardware without prior notification and accepts no liability for doing so.

	India	Europe	North America
Address:	PHYTEC Embedded Pvt. Ltd. Koramangala 8th block, Bangalore -95 India	PHYTEC Technologie Holding AG Robert-Koch-Str. 39 55129 Mainz GERMANY	PHYTEC America LLC 203 Parfitt Way SW, Suite G100 Bainbridge Island, WA 98110 USA
Ordering Information:	+91-80-41307589 *Sales@phytec.in*	+49 (800) 0749832 *order@phytec.de*	1 (800) 278-9913 *sales@phytec.com*
Web Site:	*http://www.phytec.in*	*http://www.phytec.de*	*http://www.phytec.com*

Table of Contents

1. Prepare Host Environment:

In Order to follow this manual the host requires 32-bit Ubuntu, better go with LTS versions of Ubuntu. To install Ubuntu follow the link below.

https://help.ubuntu.com/community/GraphicalInstall

Ubuntu doesn't come with all the pre-requisite archives for the development, to prepare the host for development download the bash shell script which can be available from the link below. Download and execution steps of the script are as follows.

```
$wget ftp://ftp.phytec.de/pub/Products/India/OpenBoard-AM335x/Linux/latest/tools/scripts/elinux_pkg.sh
$ chmod +x elinux_pkg.sh
$ sh elinux_pkg.sh
```

1.1 Configuring Services (TFTP, NFS)

1.1.1 Configuring TFTP

Installation of the TFTP Server is as follows; execute the command on Linux shell.

```
$ sudo apt-get -y  install xinetd tftpd tftp
$ sudo vim /etc/xinetd.d/tftp
```

Add the following structure in /etc/xinetd.d/tftp file if it is not there or edit if it exists.

```
service tftp
        {       protocol = udp
                port = 69
                socket_type = dgram
                wait = yes
                user =  nobody
                server = /usr/sbin/in.tftpd
                server_args = /var/lib/tftpboot -s
                disable = no
        }
```

```
$ sudo service xinetd restart
```

Testing TFTP Server:

```
$ cp <file_name> /var/lib/tftpboot
$ sudo chmod +wr –R /var/lib/tftpboot
$ tftp localhost
$ get <file_name>
```

If the same file is received without any error from the tftpserver then the tftp service is configured correctly.

To transfer file through tftp between host and target follow steps listed below, configure the IP address for host by selecting the option Network Connection and modify the related option. The same can be done through command line interface by issuing the following command

```
$ ifconfig eth0 192.168.1.12 up
```

This will be used as server IP address. Note that IP for server and target should not be same

On Target side issue the same command with a different IP as below

```
$ ifconfig eth0 192.168.1.11 up
```

Configure the gateway address for target

```
$ route add default gw 192.168.1.1
```

Issue the tftp command to transfer the file

```
$ tftp  -r <file_name > -g <server_ip>
```

Check the default gateway by command

```
$ route
or
$ /sbin/route -n
```

1.1.2 Configuring NFS Service

First step in NFS service configuration is to decide the path of a directory which is going to share the RFS. Here in this manual it has been taken as /nfsroot which is to be created by user by issuing the following command.

```
$ sudo mkdir /nfsroot
```

Edit the /etc/exports file and add the below line at the end of the file, save and close it

```
/nfsroot  *(rw, sync, no_subtree_check, no_root_squash)
```

```
$ sudo gedit  /etc/exports
```

To start the service with made changes issue the following commands on the terminal.

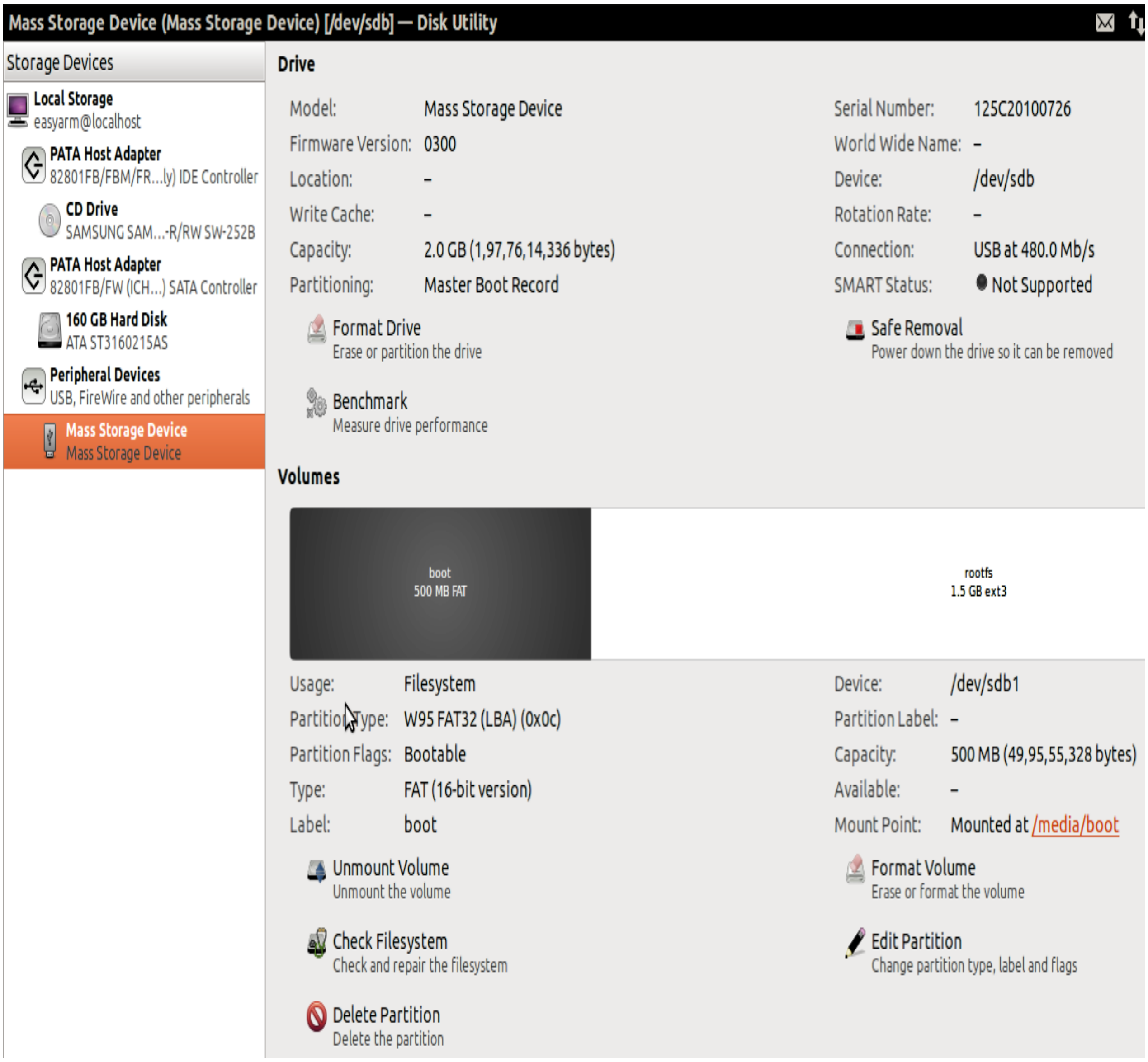
```
$ sudo exportfs  –ra

$ sudo /etc/init.d/nfs-kernel-server restart
```

1.2 Partitioning Micro SD card in Ubuntu

Method 1: Using Graphical User Interface

Use the application Disk Utility in Ubuntu.

- Select Mass storage device
- Umount the volume
- Delete the partition
- Create the two partition boot and rootfs with *fat* and *ext3* support respectively
- Select the first *fat* partition and then select "edit partition" and mark it as bootable.



Mass Storage Device (Mass Storage Device) [/dev/sdb] — Disk Utility

Drive

Model: Mass Storage Device
Firmware Version: 0300
Location: -
Write Cache: -
Capacity: 2.0 GB (1,97,76,14,336 bytes)
Partitioning: Master Boot Record

Serial Number: 125C20100726
World Wide Name: -
Device: /dev/sdb
Rotation Rate: -
Connection: USB at 480.0 Mb/s
SMART Status: ● Not Supported

Volumes

Usage: Filesystem
Partition Type: W95 FAT32 (LBA) (0x0c)
Partition Flags: Bootable
Type: FAT (16-bit version)
Label: boot

Device: /dev/sdb1
Partition Label: -
Capacity: 500 MB (49,95,55,328 bytes)
Available: -
Mount Point: Mounted at /media/boot

Figure 2. 11

Method 2: Using Command line interface

Download the script and follow the steps.

```
$ wget ftp://ftp.phytec.de/pub/Products/India/OpenBoard-AM335x/Linux/latest/tools/scripts/mkcard.sh
```

Change the mode of script

```
$ chmod 777 mkcard.sh
```

Insert the Micro SD card and issue the following command and check Micro SD card device interface

```
$ dmesg | tail
```

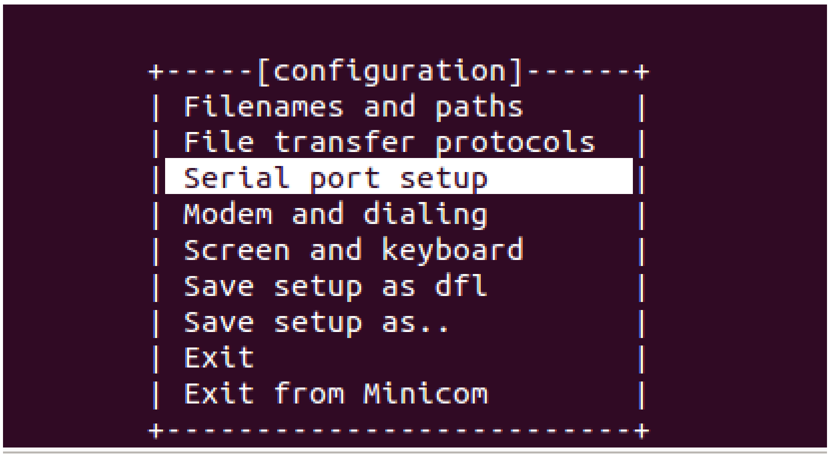
Execute the following script with that device interface like /dev/sda

```
$ sudo ./mkcard.sh /dev/sdx  /* /dev/sdx is device interface name */
```

1.3 Setting Serial Console (Minicom)

```
$ sudo minicom –s
```

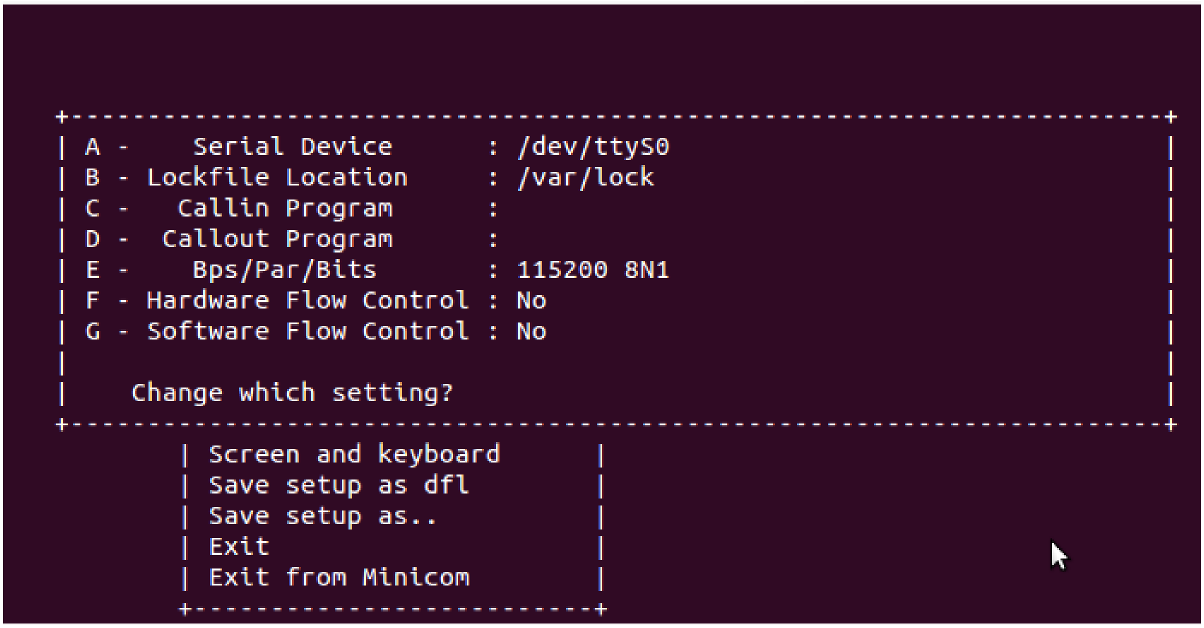
Select the option serial port setup from menu on terminal and press 'Enter'



```
+-----[configuration]------+
| Filenames and paths      |
| File transfer protocols  |
| Serial port setup        |
| Modem and dialing        |
| Screen and keyboard      |
| Save setup as dfl        |
| Save setup as..          |
| Exit                     |
| Exit from Minicom        |
+--------------------------+
```

Figure 3.1

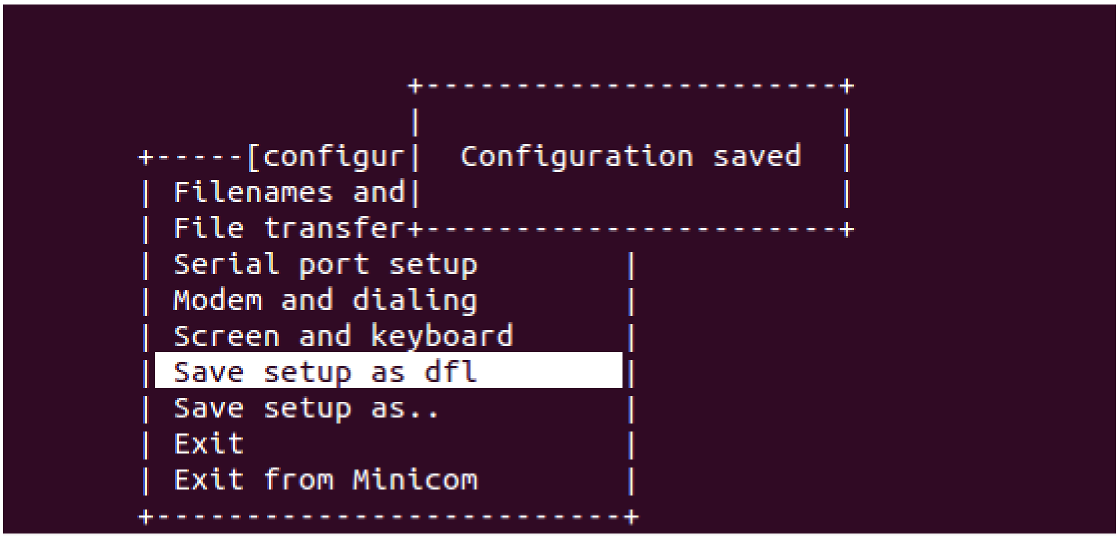
Using the options at left; make the setting as shown in the figure 3.2. For ex: Serial device - /dev/ttsyS0, first press ‘A’ the cursor blinks at the particular line then change the device name to /dev/ttyS0 and press ‘ENTER’.



```
+-----------------------------------------------------------------------+
| A -    Serial Device      : /dev/ttyS0                                 |
| B - Lockfile Location     : /var/lock                                  |
| C -   Callin Program      :                                            |
| D -  Callout Program      :                                            |
| E -    Bps/Par/Bits       : 115200 8N1                                 |
| F - Hardware Flow Control : No                                         |
| G - Software Flow Control : No                                         |
|                                                                       |
|    Change which setting?                                              |
+-----------------------------------------------------------------------+
        | Screen and keyboard      |
        | Save setup as dfl        |
        | Save setup as..          |
        | Exit                     |
        | Exit from Minicom        |
        +--------------------------+
```

Figure 3.2

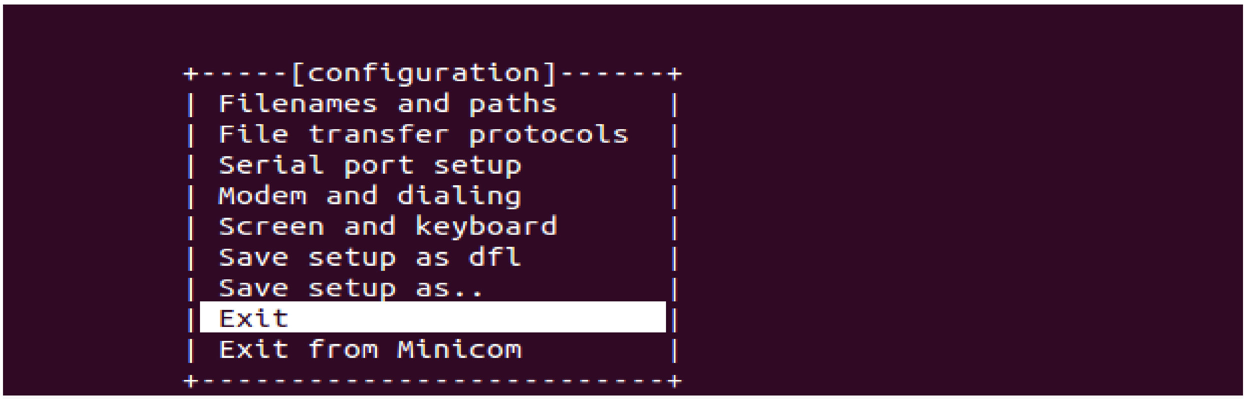
After making all changes like baud rate -115200, hardware control flow -No etc. that are required press 'ENTER' to come out from the screen, now the previous screen will appear.



```
                 +------------------------+
                 |                        |
+-----[configur|  Configuration saved    |
| Filenames and|                          |
| File transfer+------------------------+
| Serial port setup         |
| Modem and dialing         |
| Screen and keyboard       |
| Save setup as dfl         |
| Save setup as..           |
| Exit                      |
| Exit from Minicom         |
+---------------------------+
```

Figure 3.3

Now select the 'Save setup as dfl' it saves the setting as default.
Now select 'Exit'



```
+-----[configuration]------+
| Filenames and paths      |
| File transfer protocols  |
| Serial port setup        |
| Modem and dialing        |
| Screen and keyboard      |
| Save setup as dfl        |
| Save setup as..          |
| Exit                     |
| Exit from Minicom        |
+--------------------------+
```

Figure 3.4

It will take into ‘minicom’ the serial terminal.

1.4 Installing Toolchain

This section explains how to install prebuilt toolchain binaries on the Host. Whereas building of Toolchain is explained in buidsystem section.

1.4.1 Downloading the Toolchain:

Download the toolchain from the public ftp of PHYTEC from the link below.

ftp://ftp.phytec.de/pub/Products/India/OpenBoard-AM335x/Linux/latest/tools/toolchain/arm-cortexa8-linux-gnueabi.tar.bz2

1.4.2 Export the Toolchain to Shell Path:

Extract the toolchain that has downloaded in the above section to the HOME directory. For extracting the toolchain from GUI you can ‘right click’ on the toolchain and select ‘Extract’. Same can be done from terminal issue the following

```
$ tar –xvf  arm-cortex8-linux-gnueabi.tar.bz2
```

Now write a script that export the target Architecture and toolchain prefix as below.

```
$ vim env.sh
```

In the file add the following lines

```
#!/bin/sh
export PATH=$PATH:/home/<user>/<path to toolchain bin>
export ARCH=arm
```

Save and exit from the editor. Execute the script by issuing

```
$ .  env.sh
```

Note: The command is “dot space env.sh”. Execute the above command for every new shell where you are trying to start compilation of Linux kernel or bootloaders.

Check whether the toolchain path is set or not by issuing

```
$ echo $PATH
```

Above command displays the entire paths that are added to Shell PATH variable separated by colon ':'. See if your path is printed on the terminal. If you are not able to see your toolchain path repeat the steps from the beginning of this section.

```
$ echo $PATH
```

2 Download prebuilt images:

This section gives the links to all the required binaries to boot the OpenBoard-AM335x using the simplest method using Micro SD card. Partition your Micro SD card. Follow the Partitioning Micro SD card in Ubuntu section.

All the tested binaries are placed at the public PHYTEC ftp. Follow the links below to download the binaries.

ftp://ftp.phytec.de/pub/Products/India/OpenBoard-AM335x/Linux/latest/images/MLO

ftp://ftp.phytec.de/pub/Products/India/OpenBoard-AM335x/Linux/latest/images/barebox.img

ftp://ftp.phytec.de/pub/Products/India/OpenBoard-AM335x/Linux/latest/images/u-boot.img

ftp://ftp.phytec.de/pub/Products/India/OpenBoard-AM335x/Linux/latest/images/uImage

ftp://ftp.phytec.de/pub/Products/India/OpenBoard-AM335x/Linux/latest/images/root.ubi

3 Flashing binaries:

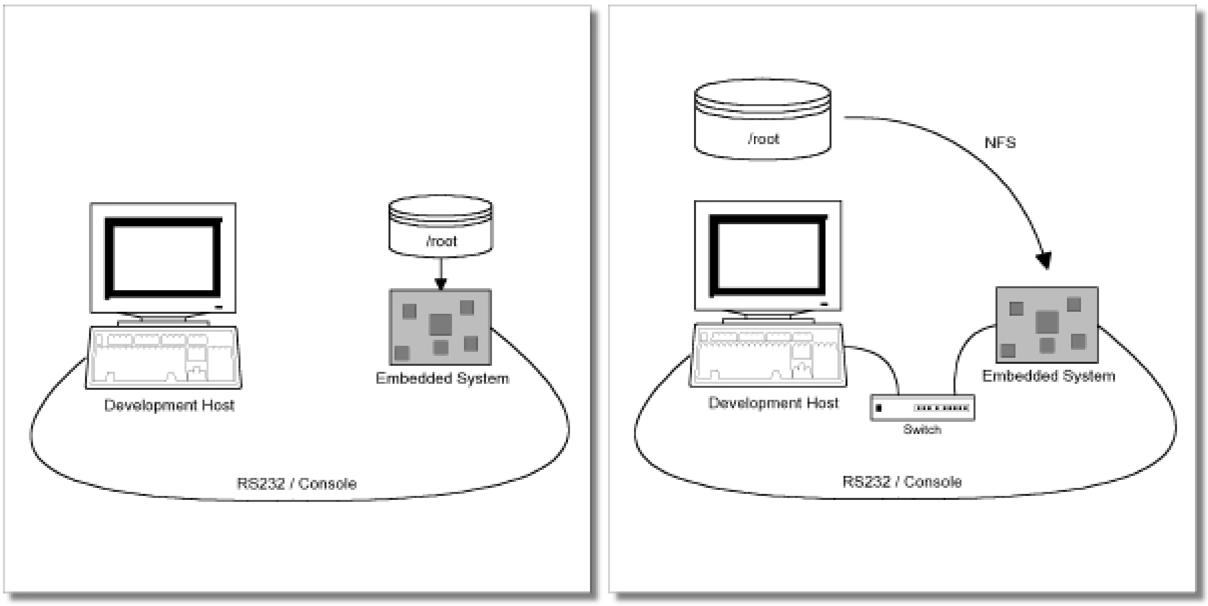


Figure 3.8

Figure shows two methods. The first method used to provide all needed components to run on the target itself. The Linux kernel image and the root filesystem image are persistent in the media the target features. This means the only connection needed is the serial cable to see what is happening on our target. This method is called standalone. This works after writing image into NAND which is explained later in Document.

The other method is to provide needed components via network. In this case the development host is connected to the OpenBoard-AM3359 with a serial cable and via Ethernet; the embedded board boots into the bootloader, then issues a TFTP request on the network and boots the kernel from the TFTP server on the host. Then, after decompressing the kernel into the RAM and starting it, the kernel mounts its root OpenBoard/root/ directory.

The latter one is especially for development purposes, as it provides a very quick turnaround while testing the kernel and the root filesystem. Other methods like Micro SD card and NAND have also been discussed.

************* Using U-boot bootloader **************

3.1 Using U-boot bootloader:

3.1.1 Auto flash:

Auto flash is a procedure provided in OpenBoard-AM335X U-Boot by which user can easily flash all the binaries into the Nand flash using user buttons BTN1 & BTN2. Do to so first copy MLO, u-boot.img, uImage, rootfs.ubi into SD card and insert it into the target board. Switch the board to SD boot mode by shorting Jumper JP3.

Usage:

BTN1: If you want to flash all binaries (MLO, u-boot.img, uImage, rootfs.ubi) into NAND just hold BTN1 and Power ON/RESET.

BTN2: If you want to flash only uImage & rootfs.ubi into NAND just hold BTN2 and Power ON/RESET.

3.1.2 Flash binaries using fastboot:

For writing all images into NAND using Fastboot command, insert OTG cable and open the JP12 jumper and power up the board and stop at auto boot by pressing any key to get u-boot prompt & then type the following command to enter into fastboot mode.

On Target:

```
$  fastboot
```

On HOST:

Install the required Tools.

```
$  sudo add-apt-repository ppa:nilarimogard/webupd8
$  sudo apt-get update
$  sudo apt-get install android-tools-adb android-tools-fastboot
```

Flashing MLO (x-loader):

```
$  sudo fastboot flash xloader MLO
```

Flashing u-boot.img (bootloader):

```
$  sudo fastboot flash bootloader u-boot.img
```

Flashing uImage (Kernel):

```
$  sudo fastboot flash kernel uImage
```

Flashing rootfs.ubi (RFS):

```
$  sudo fastboot flash filesystem rootfs.ubi
```

Remove the power and OTG cable (remove the jumper JP3) & Power up the board.

3.1.3 Booting from Network:

Now flash Kernel image and rootfs with tftp & nfs. Before using this method make sure that tftp & nfs-server is working fine. To test and configure refer section “Configuring Services (TFTP, NFS)” Copy the uImage (kernel image) and root filesystem into /var/lib/tftpboot & /nfsroot respectively.

On Host:

```
$ sudo cp uImage /var/lib/tftpboot
$ sudo  tar -xvf  rootfs.tar  -C /nfsroot
```

Now switch to minicom terminal and change certain environment variables of u-boot bootloader using commands listed below

On Target:

```
$ setenv ipaddr  <ip_address_of_target>                /*setting target ip */
```

```
                Ex: setenv ipaddr  192.168.1.11

                $ setenv serverip  <ip_address_of_server>          /* setting server ip */
                Ex. setenv serverip  192.168.1.12

                $ setenv netmask 255.255.255.0                     /* setting the netmask */

                $ setenv gatewayip <ip_address_for_gateway>        /* setting the gateway ip */
                Ex.  setenv gatewayip192.168.1.1


                /* setting the boot parameter for booting via tftp & nfs server  */

                $ setenv bootcmd 'tftp 0x82000000 uImage ;bootm 0x82000000'
                $ setenv bootargs 'console=ttyO0,  115200n8 noinitrd root=/dev/nfs
                        nfsroot=(serverip):/nfsroot rw init=/sbin/init rootwait
                        ip=(ipaddr):(serverip):(gateway):(netmask)::eth0:'

                Ex. setenv bootargs 'console=ttyO0,115200n8 noinitrd root=/dev/nfs
nfsroot=192.168.1.12:/nfsroot rw init=/sbin/init rootwait
ip=192.168.1.11:192.168.1.12:192.168.1.1:255.255.255.0::eth0:'
```

3.1.4 Booting from Micro SD card:

For partitioning the Micro SD card refer section *“Partitioning a Micro SD card in linux”*
Copy the files u-boot.img, MLO, uImage in first partition (boot) and extract root filesystem in (rootfs) second partition.

On HOST:

Insert the Micro SD card

```
                $ cp u-boot.img /media/boot
                $ cp MLO /media/boot
                $ cp uImage /media/boot
                $ sudo  tar -xvf  root.tgz  -C /media/rootfs
```

On Target:

```
                $ setenv bootcmd 'mmc rescan;fatload mmc 0:1 0x82000000  uImage; bootm
0x82000000 '
                /* <0:1> when uImage in 1st partition */

                $ setenv bootargs 'console=ttyO0,115200n8 root=/dev/mmcblk0p2 rw
rootfstype=ext3 rootwait'
                /* mmcblk0p2 when rootfilesystem in 2nd partition.  */
```

```
$ saveenv

/* save the setting in nand so it's extremely important otherwise
efforts will go in vain */
```

3.1.5 Booting from NAND:

For writing all images into NAND using Micro SD card, insert Micro SD card and short the JP3 jumper and power up the board. Press any to stop auto-boot to get u-boot prompt.

Flashing MLO (x-loader):

```
$ mmc rescan
$ mw.b 0x82000000 0xFF 0x20000
$ fatload mmc 0 0x82000000 MLO
$ nandecc hw 2
$ nand erase 0x0 0x20000
$ nand write.i 0x82000000 0x0 0x20000
```

Flashing u-boot.img (u-boot):

```
$ mmc rescan
$ mw.b 0x82000000 0xFF 0x40000
$ fatload mmc 0 0x82000000 u-boot.img
$ nandecc hw 2
$ nand erase 0x80000 0x40000
$ nand write.i 0x82000000 0x80000 0x40000
```

Flashing uImage (Kernel):

```
$ mmc rescan
$ mw.b 0x82000000 0xFF 0x400000
$ fatload mmc 0 0x82000000 uImage
$ nandecc hw 2
$ nand erase 0x120000 0x400000
$ nand write.i 0x82000000 0x120000 0x400000
```

Flashing rootfs.ubi (RFS):

```
$ mmc rescan
$ mw.b 0x82000000 0xFF 0x1E90000
$ fatload mmc 0 0x82000000 root.ubi
$ nandecc hw 2
$ nand erase 0x520000 0x1E90000
$ nand write.i 0x82000000 0x520000 0x1E90000
```

Remove the power and Micro SD card (remove the jumper JP3).
Power up the board and set the environment variables in u-boot prompt.

```
$ setenv bootcmd 'nandecc hw 2; nand read.i 0x82000000 0x120000 \
0x400000; bootm 0x82000000'
$ setenv bootargs 'console=ttyO0,115200 \
da8xx_fb.panel_name=HT_HT800070I  root=ubi0:root ubi.mtd=12,2048 \
rootfstype=ubifs noinitrd mtdparts=omap2-nand.0:128k(xload),128k(xload_backup1),128k(xload_backup2),128k(xload_backup3),512k(barebox)ro,128k(bareboxenv),4M(kernel),-(root);spi_flash:128k(xload),512k(barebox),128k(bareboxenv),4M(kernel),-(root)'
$ saveenv
```

Important: *Save the setting in nand so it's extremely important otherwise efforts will go in vain*

3.2 Using Barebox bootloader:

3.2.1 Booting from Micro SD card:

For partitioning the Micro SD card refer section *“Partitioning a Micro SD card in linux”*

Copy barebox.img, MLO, uImage in first partition (boot) and Root file system in second partition (rootfs).

On HOST:

Insert the Micro SD card.

```
$ cp barebox.img /media/boot
$ cp MLO /media/boot
$ cp uImage /media/boot
$ sudo  tar -xvf  rootfs.tar  -C /media/rootfs
```

On Target:

Insert the Micro SD card and make sure that the Jumper JP3 is shorted, Power On the board, interrupt the auto boot by pressing any key to get barebox prompt. Edit the config file do the modifications as below without quotes.

```
$ edit /env/config
“ kernel_loc=disk ”
“ rootfs_loc=disk ”
“ rootfs_type=ext3 ”
“ kernelimage=uImage “
```

Ctrl +d to save the file.

```
$ saveenv
```

3.2.2 Booting from NAND:

If booting from SD Card start from here

```
barebox@Phytec phyCORE-AM335x:/ mkdir  /mnt
barebox@Phytec phyCORE-AM335x:/ mount  /dev/disk0.0 fat /mnt/
barebox@Phytec phyCORE-AM335x:/ erase /dev/nand0
barebox@Phytec phyCORE-AM335x:/ cd /mnt/
barebox@Phytec phyCORE-AM335x:/mnt/ cp MLO /dev/nand0.xload.bb
barebox@Phytec phyCORE-AM335x:/mnt/ cp u-boot.img /dev/nand0.barebox.bb
barebox@Phytec phyCORE-AM335x:/mnt/ cp uImage /dev/nand0.kernel.bb
barebox@Phytec phyCORE-AM335x:/mnt/ cp root.ubifs /dev/nand0.root.bb
```

Edit the file edit /env/config

```
$ edit /env/config
“ kernel_loc=nand ”
“ rootfs_loc=nand ”
“ rootfs_type=ubifs ”
```

Ctrl +d to save the file.

```
$ saveenv
```

4 Preparing OpenBoard-AM335x Bootloaders

This is required if board is not flashed with correct bootloader. Even you can skip to the flashing section if you want to use prebuilt images.

4.1 U-Boot bootloader:

Note: Refer section 1.4 Installing Toolchain before going ahead.

Download the source of u-boot version 2011.09 from the following link.

http://ftp.denx.de/pub/u-boot/u-boot-2011.09.tar.bz2

After getting the source copy it to working directory

```
$ cp u-boot-2011.09.tar.bz2  ~/work
$ cd ~/work
```

Extract the source here by following command

```
$ tar -xvf u-boot-2011.09.tar.bz2
```

Go to extracted folder by issuing the following command

```
$ cd ~/work/u-boot-2011.09
```

Download the patches for OpenBoard-AM335x from the public ftp from the link below. Apply the patches.

```
$ wget ftp://ftp.phytec.de/pub/Products/India/OpenBoard-AM335x/Linux/latest/src/patches/u-boot-2011.09/0001-u-boot-2011.09-openboard-phyCORE-AM3359.patch
```

Apply the patches one by one using following command. File name indicates the series how to apply the patches.

```
$ patch -p1 <  0001-u-boot-2011.09-openboard-phyCORE-AM3359.patch
```

Cleaning all the binaries and other configuration files

```
$ make clean
$ make distclean
$ make pcm051_config        /* configuring the U-Boot for target */
$ make
```

If the compilation goes well you will find the files u-boot, u-boot.bin and MLO in the source directory. If you experience any error act accordingly or post the errors at our forums you will get help accordingly.

4.2 Barebox bootloader:

Note: Refer section 1.4 Installing Toolchain before going ahead.

Download the source of barebox version 2012.11.0 from the following link.

http://barebox.org/download/barebox-2012.11.0.tar.bz2

After getting the source copy it to working directory

```
$ cp barebox-2012.11.0.tar.bz2  ~/work
$ cd ~/work
```

Extract the source here by following command

```
$ tar -xvf barebox-2012.11.0.tar.bz2
```

Go to extracted folder and issue the following command

```
$ cd ~/work/<directory path to source>
```

Download the patches for OpenBoard-AM335x from the public ftp from the link below. Apply the patches.

```
$ wget ftp://ftp.phytec.de/pub/Products/India/OpenBoard-AM335x/Linux/latest/src/patches/barebox-2012.11.0/0001-barebox-2012.11.0-openboard-AM3359.patch
```

Apply the patches one by one using following command. File name indicates the series how to apply the patches.

```
$ patch -p1 <0001-barebox-2012.11.0-openboard-AM3359.patch
```

Cleaning all the binaries and other configuration files

```
$ make clean
$ make distclean
$ make pcm051lb_defconfig /* configuring the Barebox for target */
$ make barebox.img
```

If the compilation goes well you will find the files barebox.bin and barebox.img in the source directory. If you experience any error act accordingly or post the errors at our forums you will get help accordingly.

4.3 Preparing OpenBoard-AM335x Linux kernel:

Note: Refer section 1.4 Installing Toolchain before going ahead.

Download the source of Linux kernel version 3.2 from the following link.

http://www.kernel.org/pub/Linux/latest/kernel/v3.x/linux-3.2.tar.bz2

After getting the source copy it to working directory

```
$ cp linux-3.2.tar.bz2 ~/work
$ cd ~/work
```

Extract the source here by following command

```
$ tar -xvf linux-3.2.tar.bz2
```

Go to extracted folder and issue the following command

```
$ cd ~/work/linux-3.2/
```

Download the patches for OpenBoard-AM335x from the public ftp from the link below. Apply the patches.

```
$ wget ftp://ftp.phytec.de/pub/Products/India/OpenBoard-AM335x/Linux/latest/src/patches/linux-3.2/
```

Apply the patches one by one using following command. File name indicates the series how to apply the patches.

```
$ patch -p1 <  <patch_name.patch> /* apply all patches one by one */
```

Cleaning all the binaries and other configuration files

```
$ make clean
$ make distclean
$ make pcm051lb_defconfig /* configuring the Kernel for target */
$ make uImage
```

If the compilation goes well you will find the files zImage and uImage in arch/arm/boot directory. If you experience any error act accordingly or post the errors at our forums you will get help accordingly.

5 Compiling User-application:

Note: Refer section 1.4 Installing Toolchain before going ahead.

Make a helloworld program for target. Open file helloworld.c via command

```
$ gedit helloworld.c
```

Compile it with toolchain

```
$ arm-cortexa8-linux-gnueabi-gcc helloworld.c -o helloworld
```

Transfer it on target by tftp-server or by Micro SD card or USB

```
On target for tftp $ tftp -r  helloworld -g <server_ip>and switch to minicom terminal and run
For Micro SD/USB on Host $ cp helloworld /media/<partion name>
```

Insert the USB or MMC to target and switch to minicom terminal
It will display the device like sdb:sdb1 .Use that device for mounting.

```
$ mount /dev/sdb1 /mnt                    (sdb1) is USB-device interface
$ cd /mnt
$ ./helloworld                  /* it is dot slash
```

6 Compiling RFS with PTX-dist:

Download the PTX-dist packet from the following link:

ftp://ftp.phytec.de/pub/Products/India/OpenBoard-AM335x/Linux/latest/buildsystems/ptxdist/ptxdist-2012.03.0.tar.bz2

Extract the above downloaded source

```
$ tar -xvf ptxdist-2012.03.0.tar.bz2
```

Go to downloaded folder

```
$ cd ptxdist-2012.03
```

Configure the packet

```
$ ./configure
```

When the configure script is finished successfully, to compile run

```
$ make
$ sudo make install
```

Download the toolchain from the public ftp of PHYTEC from the link below.

ftp://ftp.phytec.de/pub/Products/India/OpenBoard-AM335x/Linux/latest/tools/toolchain/arm-cortexa8-linux-gnueabi.tar.bz2

Extract the toolchain that has been downloaded. Issue following

```
$ tar –xvf  arm-cortexa8-linux-gnueabi.tar.bz2
```

Download the PTX-dist BSP source from following link

ftp://ftp.phytec.de/pub/Products/India/OpenBoard-AM335x/Linux/latest/buildsystems/ptxdist/OpenBoard-AM335x.PD13.0.0.tar.gz

Extract the source

```
$ tar -xvf OpenBoard-AM335x.PD13.0.0.tar.gz
```

Go to extracted folder

```
$ cd OpenBoard-AM335x.PD13.0.0
```

Select the toolchain

```
$ ptxdist toolchain <path_of_toolchain_bin>
```

Start the build

```
$ ptxdist go
```

Check the images at platform-phyCORE-AM335x/images/

```
$ ls platform-phyCORE-AM335x/images/
```

7 Compiling RFS with Buildroot:

Download the Buildroot source from following link

ftp://ftp.phytec.de/pub/Products/India/OpenBoard-AM335x/Linux/latest/buildsystems/buildroot/buildroot-2012.11.1.tar.bz2

Extract the source

```
$ tar -xvf buildroot-2012.11.1.tar.bz2
```

Go to extracted folder

```
$ cd buildroot-2012.11.1
```

Configure the buildroot by issuing following command

```
$ make pcm051lb_defconfig
```

Compile the buildroot

```
$ make
```

Check the output files at output/images

```
$ ls output/images
```

8 Compiling RFS with Yocto:

Download the Yocto buidsystem source from following link

ftp://ftp.phytec.de/pub/Products/India/OpenBoard-AM335x/Linux/latest/buildsystems/yocto/poky-danny-8.0.tar.bz2

Before starting the build process install the necessary packages.

```
$ sudo apt-get install gawk wget git-core diffstat unzip texinfo build-essential chrpath libsdl1.2-dev xterm
```

Extract the source

```
$ tar –xvf  poky-danny-8.0.tar.bz2
```

Go to extracted folder

```
$ cd poky-danny-8.0
```

Configure poky-danny-8.0 by issuing following command

```
$ .   ./oe-init-build-env
```

Then type the following command to get hob GUI interface.

```
$ hob
```

Then under “select machine” option select pcm051-bsp to build images for Openboard-AM335x board.

To use hob refer the link given below.

https://www.yoctoproject.org/documentation/hob-manual

Check the output files at tmp/deploy/images

```
$ ls  tmp/deploy/images
```