NutsBoard
Almond SoM with
Walnut Carrier Board

Linux 4.4 SDK Release Notes

Rev 1.0 20170120

Copyright

Trademark

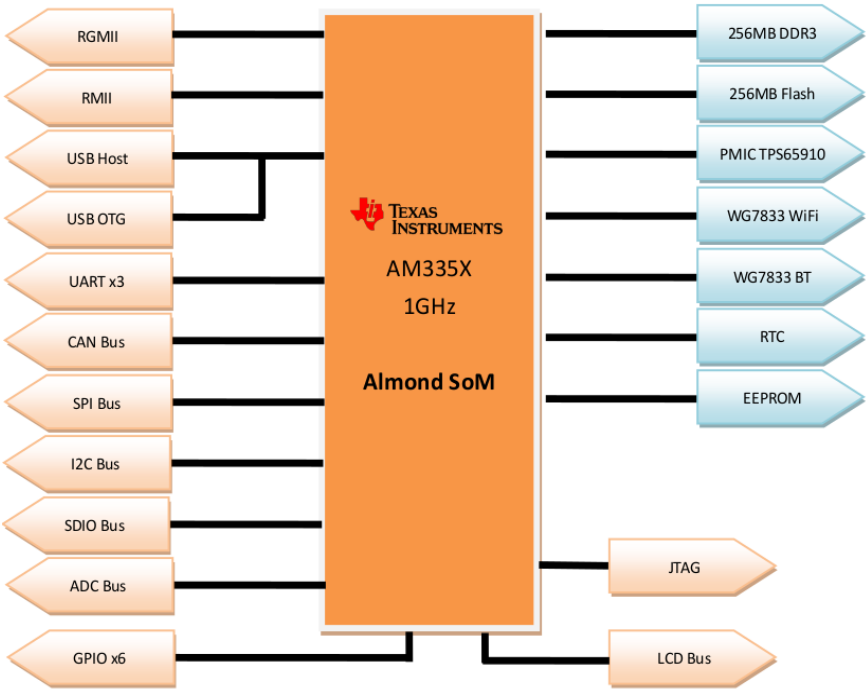
Revision History

Revision	Date	Author	Description
V1.0	01/20/2017	Wig	First Release

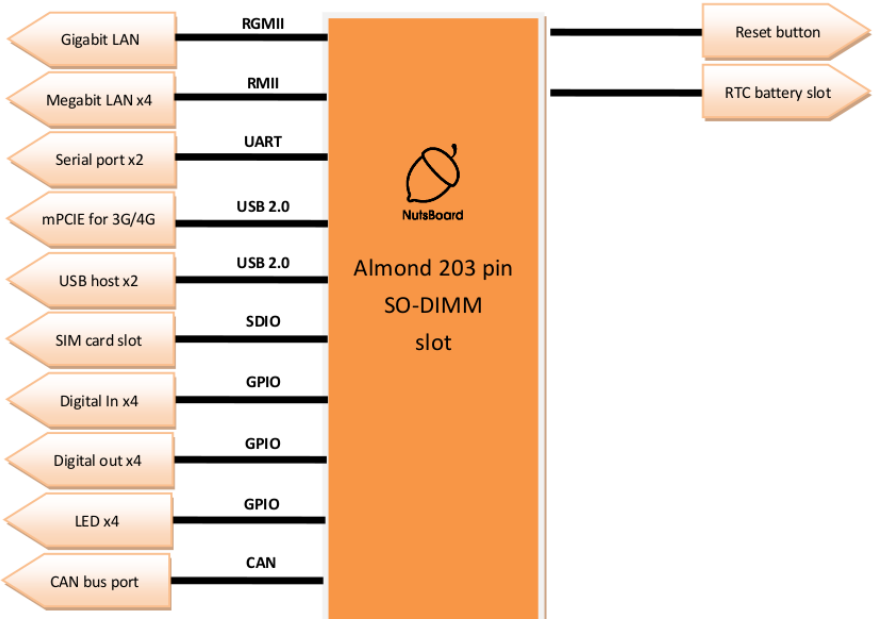
Contents

1. NutsBoards Overview

NutsBoard – Almond SoM block diagram



NutsBoard – Walnt Carrier block diagram



2. SDK Overviw

2.1 Contains

This release contains:

- u-boot bootloader source code (version 2016.05)
- Real-Time Linux kernel source code (version 4.4.32)
- Cookers source code (build script)
- User manual PDF (this document)

2.2 Partition Layout of Software Image

Section	Description
MBR	Partition information
MLO	First stage u-boot image
u-boot.img	
Partition 1 (FAT32) Under /boot directory ● uEnv.txt ● zImage ● dtb	● u-boot.img: Second stage u-boot image ● uEnv.txt: U-boot environment, you can set boot arguments in this file. ● dtb: linux device tree file, it's platform-specific.
Partition 2 (EXT4) rootfs	OS rootfs

3. Software Instructions

3.1 Setting the Develop Environment

Step by step to build the environment as follows:

- Download the cross compiler toolchain (we prefer linaro gcc 5.3 as default toolchain).
- Source the cookers build script, for example NutsBoard Almond SoM, remember rename the folder of cross-compiler to “toolchain”:

```
$ cd <your source folder>
$ source cookers/env.bash.am335x.almond.walnut
```

- For Peanut (with IMX6 platform)

```
$ cd <your source folder>
```

```
$ source cookers/env.bash.imx6.peanut.walnut
```

See inside the cookers folder for additional options.

USE:

- For full build.

```
$ cook -j8
```

- For a rebuild.

```
$ heat -j8
```

- To clean build area (“make distclean”).

```
$ throw
```

3.2 Compile the bootloader (u-boot)

- Fast way using cooker script, remember to source the environment first:

```
$ cd <your u-boot folder>
$ cook -j8
```

- Normal way, and PATH_UBOOT is your u-boot path, for exmaple use NUTS board:

```
$ export PATH="${PATH_UBOOT}/tools:${PATH}"
$ export ARCH=arm
$ export CROSS_COMPILE="${PWD}/toolchain/bin/arm-linux-gnueabihf-"
$ cd <your u-boot folder>
$ make nuts_defconfig
$ make -j8
```

Finally, you can get the images what you want (MLO, u-boot.img).

3.3 Compile the kernel

- Fast way using cooker script, remember to source the environment first:

```
$ cd <your kernel folder>
$ cook -j8
```

- Normal way, for exmaple use NutsBoard:

```
$ export ARCH=arm
$ export CROSS_COMPILE="${PWD}/toolchain/bin/arm-linux-gnueabihf-"
$ cd <your kernel folder>
$ make nutsboard_defconfig
```

```
$ make -j8 zImage modules
$ make am335x-nutsboard-almond.dtb
```

3.4 Flash a runtime image

- Prepare the boot images and rootfs as Chapter 1-2.
- Fast way using cooker script, remember to source the environment first.
- Insert the SD card to the host PC.
- Issue the command to starting flash:

```
$ flashcard sdx <sdx is device node of your SD card>
```

3.5 How to get the NutsBoard official rootfs or BSP.

- http://www.nutsboard.org/download

4. Setting a VM Develop Environment

The VMware Player is recommended for this operation. A step-by-step installation

Procedure is provided:

- Download the latest VMware Player from VMware Player website.
- Choose the version dedicated to the OS used.
- Install the VM on your systemOpen the installer and follow the instructions.
- For more information, check the VMware documentation:

  http://www.vmware.com/pdf/VMwarePlayerManual10.pdf
- Download NutsBoard disk image (Ubuntu 16.04 LTS 64bit), It can be downloaded from http://old-releases.ubuntu.com/releases/lucid/

Create a new “Open a New Virtual Machine” and select the downloaded ubuntu iso image.. Press Next button., choose the VM name: Ubuntu-16.04 and click next.Set harddisk size at least 100 GB. Select “split virtual disk into multiple files” option.Now it is necessary to set VM performances by clicking on “customize hardware” button.

Memory Options:

- 2 GB minimum (memory swapping especially during java compiling)

- 4 GB+ perfect (no memory swapping during build operations)

Ubuntu user name and password from VM image:

username:password=nutsboard:nutsboard

5. Functions Testing

5.1 WiFi station mode:

Step 1. Insert the kernel modules
root@nutsboard:~# modprobe wl18xx
root@nutsboard:~# modprobe wlcore_sdio
Step 2. Making wlan0 interface up
root@nutsboard:~# ifconfig wlan0 up
Step 3. Generate the WPA connected configuration file.
root@nutsboard:~# wpa_passphrase <SSID> <Password> > wifi_home.conf
Step 4. Connect via your configuration file.
root@nutsboard:~# wpa_supplicant -i wlan0 -c wifi_home.conf &
Step 5. Get an IP address from DHCP server
root@nutsboard:~# udhcpc -i wlan0
Step 4. Connect successful! Now you can ping the outside network.
root@nutsboard:~# ping 8.8.8.8

5.2 Bluetooth:

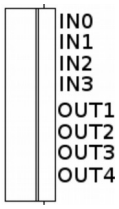
Step 1. Issue the command to initial Bluetooth function
root@nutsboard:~# uim -f /sys/devices/soc0/kim/ &
Step 2. Insert the kernel modules
root@nutsboard:~# modprobe st_drv
root@nutsboard:~# modprobe btwilink
Step 3. Making hci0 interface up
root@nutsboard:~# hciconfig hci0 up
Step 4. Scan bluetooth devices
root@nutsboard:~# hcitool scan
Step 5. Scan BLE devices
root@nutsboard:~# hcitool lescan

5.3 CAN bus:

Step 1. Issue the command to initial CAN bus function
root@nutsboard:~# ip link set can0 type can bitrate 50000; ifconfig can0 up
Step 2. Send the CAN bus data.
root@nutsboard:~# cansend can0 123#DEADBEEF
Step 3. Receive the CAN bus data
root@nutsboard:~# candump can0 &

5.4 GPIO framework:

It's NutsBoard apapted EDM GPIO framework, you can found it as following:
root@nutsboard:~# cd /sys/class/edm/gpio/
And you can control the LED lights and the Digital IN/OUT as side connector of Walnut:
root@nutsboard:~# echo "out" > GP_IN0/direction
root@nutsboard:~# echo "1" > GP_IN0/value
root@nutsboard:~# echo "out" > GP_IN0/direction
root@nutsboard:~# echo "1" > GP_IN0/value



If you want to add or rename the gpio, please modify the kernel device tree file: am335x-nutsboard-almond.dtb (please search gpio_edm).

5.5 UART RS232/RS422:

Step 1. Check the UART mode first (default is RS232), Example - UART1:
root@nutsboard:~# cat /sys/class/edm/gpio/MODEA0/value
Step 2. Check the A0-A3 value:
1,0,0,0 → RS232 mode
1,1,1,0 → RS422 mode
0,1,0,0 → RS485 mode
Step 3. Use minicom or other tools to communicate with other UART of devices.
root@nutsboard:~# minicom

5.6 UART RS485:

Same as RS232 and RS422 testing procedure, but the device tree file need to be modified:

```
uart1_pins: pinmux_uart1_pins {
    pinctrl-single,pins = <
    //0x178 (PIN_INPUT | MUX_MODE0)          /* uart1_ctsn.uart1_ctsn */
    0x17C (PIN_OUTPUT_PULLDOWN | MUX_MODE7) /* uart1_rtsn.uart1_rtsn */
    0x180 (PIN_INPUT_PULLUP | MUX_MODE7)  /* uart1_rxd.uart1_rxd */
    0x184 (PIN_OUTPUT_PULLDOWN | MUX_MODE7)  /* uart1_txd.uart1_txd */
    >;
};

&uart1 {
    pinctrl-names = "default";
    pinctrl-0 = <&uart1_pins>;
    status = "okay";
    rts-gpio = <&gpio1 19 GPIO_ACTIVE_HIGH>;
    rs485-rts-active-high;
    rs485-rts-delay = <0 0>;
    linux,rs485-enabled-at-boot-time;
};
```

5.7 Ethernet WAN port:

Get IP address from DHCP server (WAN port interface is eth1)
root@nutsboard:~# udhcpc -i eth1
A simple ping test. You can increase the size of the packet press -s switch
root@nutsboard:~# ping -s 500 192.168.0.100
PING 192.168.0.100 (192.168.0.100): 1000 data bytes
1008 bytes from 192.168.0.100: seq=0 ttl=64 time=1.980 ms
1008 bytes from 192.168.0.100: seq=1 ttl=64 time=0.459 ms
1008 bytes from 192.168.0.100: seq=2 ttl=64 time=0.461 ms
1008 bytes from 192.168.0.100: seq=3 ttl=64 time=0.475 ms
--- 192.168.0.100 ping statistics ---
4 packets transmitted, 4 packets received, 0% packet loss

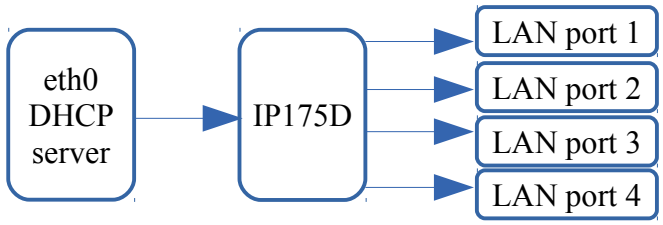
5.8 Switch LAN port:

Change to NAT mode as a DHCP server (LAN port interface is eth0)
root@nutsboard:~# eth1_nat.sh
Connect a device, then device will get a IP address automatically from the board, now, you can do the ping test to your device.
root@nutsboard:~# ping -s 500 192.168.30.25
PING 192.168.0.100 (192.168.0.100): 1000 data bytes
1008 bytes from 192.168.30.25: seq=0 ttl=64 time=1.980 ms
1008 bytes from 192.168.30.25: seq=1 ttl=64 time=0.459 ms
1008 bytes from 192.168.30.25: seq=2 ttl=64 time=0.461 ms
1008 bytes from 192.168.30.25: seq=3 ttl=64 time=0.475 ms
--- 192.168.30.25 ping statistics ---
4 packets transmitted, 4 packets received, 0% packet loss

6. Q & A

6.1 Why eth0 has no response using DHCP from Almond with Walnut?

- 4 LAN ports (eth0) is connected to a switch chip: IP175D, so do a DHCP server by itself already, and it always has cable inserted status. Please connect a device (PC, NB, etc.) to a LAN port from Almond with Walnut, and you will auto get a IP address.



6.2 WiFi problem: No wlan0 interface?

- Please check your firmwares not loaded or not. the firmwares are in /lib/firmware/ti-connectivity/

6.3 WiFi problem: No nearby wireless network?

- Please check your antenna already connected or not.
- Please check your antenna is for 2.4GHz or 5GHz, Almond supports 2.4GHz only.