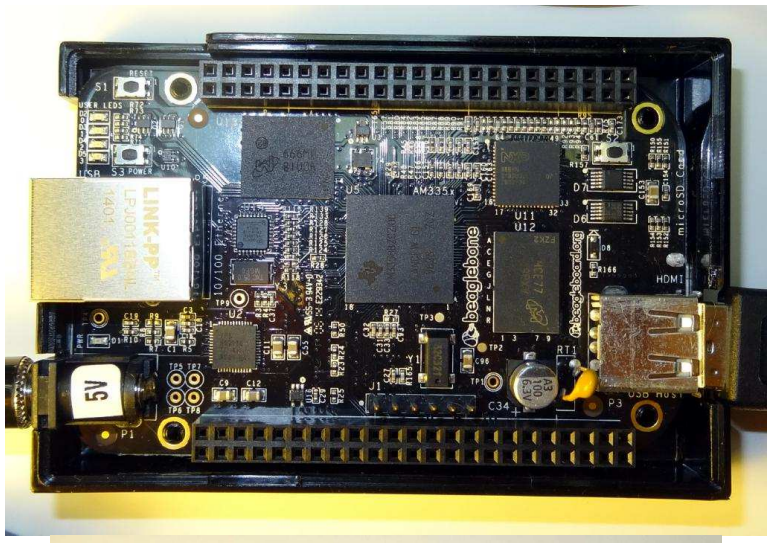
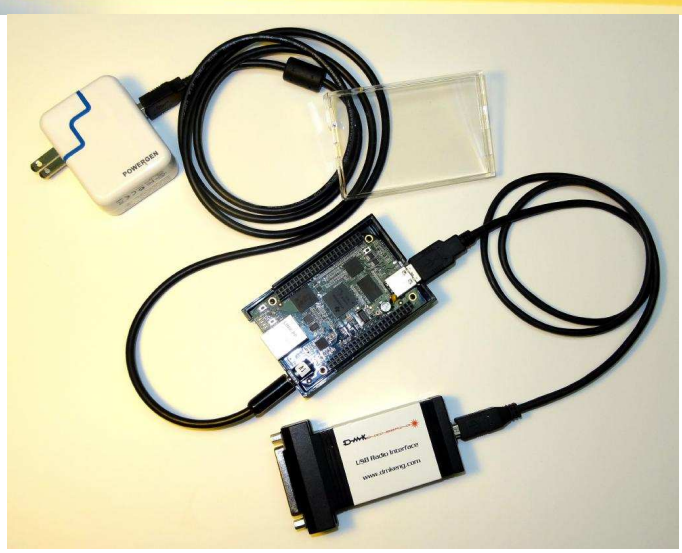
Asterisk Allstar on the BeagleBone Black





History

Ever since the Raspbery Pi was introduced over two years ago it has been the goal of many to make Asterisk Allstar work on that platform. Over the years there have been several attempts which fell short. In January, 2014 I decided to give it another try. I thought I had it but unfortunately it did not provide the kind of performance you would expect for Allstar. It worked fairly well audio wise in USB 1.1 mode but suffers from extreme audio delays that would be unacceptable in a production environment. I was using the stock Raspbien (Debian on the RPi) kernel with Dahdi and the latest Asterisk Allstar SVN.

Thinking that it might be a Debian issue I went on to compile the Allstar package in both Debian Wheezy and in Ubuntu Saucy on a PC. Both versions worked fine but Saucy required a kernel recompile to include the OSS sound code. This, I thought, proved that Debian should work on the ARM processors assuming the IO and CPU could handle it.

So I moved to the BeagleBone Black. The Black is a competitor to the RPi with a faster processor, a better IO structure, and a version 7 arm processor verses Version 6 in the RPi. It is the same physical size as the RPi and costs about $10 more.

My first attempt used Debian and the results were better than on the RPi but the DAC (the side that converts what you hear on your end) audio was not good. I finally decided to try Archlinux. Archlinux is a leading edge barebones linux that comes with very little add-on stuff. No graphics or packages that would not ordinarily be needed are part of the virgin installation. While it is not a "hold your hand" linux it was surprisingly easy to compile and get Allstar running. Maybe it was all the experience I had working at this for months before that made it seem easier but in any event I was tremendously surprised that it worked. I had almost given up on ARM Allstar working and I could hardly believe my ears that it was working so well. It is hard to explain why as the basic kernel is the same. It has to be in the ARM interface or in the less bloated Archlinux OS somewhere. CPU utilization was at least half what it had been on the Debian releases.

Initial tests were with simpleusb which worked flawlessly. Dave, KB4FXC also brought up a BBB compile and we connected to each other using our BBB's. His initial test of usbradio showed surprisingly good results but testing is still under way. It might work in usbradio mode but I would suggest using simpleusb as a starting point. I suspect that Multiple simpleusb nodes would also work. Local or Remote iaxrpt also works well. I even have my lsnodes monitor and control program working both locally and remotely with Apache. lsnodes is included on the image.

Hardware

The BBB arrives in a box with a microUSB to USB cable. This allows you to power it from your PC and also in some cases use the USB as a console connection. Archlinux is not setup to do that but you can initially power your BBB from the microUSB if you desire. The BBB has only one standard USB port which connects

directly to a DMK URI or modified USB sound FOB for interface to your radio. Eventually you will also need a power supply preferably supplying 1-2A @ 5V inserted at the barrel power connector on the BBB board. I used a USB charger type supply and I modified a USB cable cutting off one end and wiring it to a compatible male barrel connector. Red (+) to the center conductor and Black (-)- to the shell. Only use ONE power source - the microUSB or the barrel connector!

Acquiring a Black

When I started this project the BBB hardware was at rev B and cost $45. As of the first week in May 2014 rev C is out and the cost has gone up $10. Both boards work equally well. Sources for the boards are:

- https://www.adafruit.com/products/1876
- http://www.newark.com/element14/bbone-black-4g/beaglebone-black-rev-c-cortex/dp/52X5548
- http://www.element14.com/community/community/knode/single-board_computers/next-gen_beaglebone
- http://electronics.mcmelectronics.com/electronics/Beaglebone-Black
- http://www.logicsupply.com/components/motherboards/arm/bb-bblk-000/

I used a Belkin F4U040 four port hub with a single DMK URI and a wireless keyboard FOB without problems. This is a powered hub but I did not connect the power as these two devices draw far less than the maximum 500ma. There are so many hubs out there it is impossible to predict how any particular one will perform. I can only say that my limited test proves that it can work.

The RPi wiki has an extensive list of hubs tested on the RPi. I suspect the same would hold true for the BBB. It would be good to reference it before buying a hub. A powered hub would not always be necessary but it would not hurt to have one.

RPi list of acceptable and not acceptable powered USB hubs

I have done limited testing of two ports using DMK URI's and the Belkin USB hub. It seems to work fairly well. I ran it for several days on two busy nodes. At times both were operational and I noticed no problems. Further testing is necessary. I think two is the limit though.

Image File and SD cards

The image file is sized to go on a 4GB microSD card as a minimum. The compressed image (zip file) is just over 500M. It is strongly suggested to use an 8 or 16G card and optionally resize the second root partition to use the full card AFTER you configure your node. I suggest a Sandisk Ultra or Ultra+ class 10 8G or 16G card. The larger the usable area on the card the more space there is for

wear levelling. This will reduce the number of writes to any one location on the card and greatly extend its life. I have instituted as much write to RAM in this package as possible so there are very few writes to the SD card. I expect that using a larger card the life would not be an issue based on tests I have done with other systems running here. However they can eventually fail so a backup is always very important. Backing up an SD card image once you have it configured the way you want it is as easy as reading the image into your computer using Linux 'dd' or Windows 'win32diskimager'. You MUST use an image copy method. You cannot just copy files to make an image backup of the SD card. Keep a second SD card burned with the image near your BBB and in the event of a failure it could be swapped in seconds. This is not something that would happen very often but it is always good to be prepared.

The image uses Archlinux. Archlinux uses systemd for loading modules, maintaining logs, and lots of other things. It is totally different than the initscrpt you have used in other Linux distros. I just want you to be aware of this when you are looking for things that are not there like /etc/init.d! I am including some links that will help you with that but for the most part after you get your image configured it is plug and play. You do not have to be a Linux guru but a little knowledge of Linux and systemd would be helpful in maintaining your BBB Allstar system.

Downloading the Image

The image file must be downloaded and written to a microSD card. This can be done on a Windows or Linux PC. If you are using Windows and win32diskimager the procedure is to first unzip the downloaded file. You should then have the image (.img) file in your directory. Then insert the card in your reader, start the win32diskimager program, select the file you downloaded and then select write.

Download win32diskimager - http://sourceforge.net/projects/win32diskimager

Note that after installing win32diskimager and starting for the first time you may get an error about elevating permission levels or something like that. If you do, exit out and left click once to select the win32diskimager icon, then right click and select "run as administrator" It should then run properly.

See details on how to use win32diskimager here or at this site

One of the sites gives an example using the Raspberry Pi. The procedure would be the same for the BBB just substitute the BBB Allstar image. It takes anywhere from 5-15 minutes to write the card depending on the image size and your systems speed. Once the card is successfully written you can remove it from the PC reader/writer and install it in the BBB with the power disconnected. Note that this one image written to your SD card contains the complete Archlinux/Allstar package.

If you are using Linux to write the image, insert the card in your card reader/writer. Then do a 'df' to determine the device name. On my system it was /dev/sdb but yours may be different. It is important to identify the device

properly. You do not want to write to your hard drive!! Once you have identified the device use 'dd' to write the image.

dd if=BBB_Allstar_v1.1.img of=/dev/sdb << Change the filename and device for your system

Remember dd is not going to ask if it is OK or check if you entered something wrong so be very careful to enter the proper output device!!!

The latest image is available here - http://hamvoip.org/BBB/BBB_Allstar_v1.1.zip

See Image Version Information

Booting the Image

Now that you have the microSD card with image inserted in the BBB apply power and it should boot to a login prompt. The next section will give options on how to initially login.

First communications with the BBB

Initially you have to communicate with the BBB to setup your local IP address and other things in the BBB. The easiest way to do that is to use a terminal connected to the HDMI port of the BBB and a keyboard connected to the USB port. You do not need a mouse. There is no graphics mode, you login in standard Linux text mode. The BBB has a micro HDMI connector so you would need to acquire a micro HDMI to standard HDMI cable or adapter. This does not come with the BBB but could be purchased locally at a Walmart, Best Buy, etc. or on the Internet. Most newer monitors now have HDMI inputs. If not there is an HDMI to DVI adapter available if your monitor has a DVI input. If you do not have an HDMI or DVI input on a computer monitor you can use any digital broadcast television receiver as they all have HDMI inputs. This connection will only be needed for initial setup.

I have a wireless keyboard and it works fine with its wireless FOB plugged into the BBB USB port directly or into a hub along with the DMK URI.

I have included a script I wrote that uses the blue LED furthest from the Ethernet connector to send the assigned Ethernet IP address in Morse code at boot-up. At boot it starts the heartbeat blink and then sends a Morse "HI" twice followed by the IP address. The address is in four groups. The groups separated by periods. So 192.168.0.132 would be sent as - HI HI 192 168 0 123 This sequence is repeated three times. If you miss it you can always reboot. It comes up approximately 15-20 seconds after power is applied. It is too bad the BBB does not have a piezo sounder as it would really work nice that way. If none of the options so far work for you there are other methods to determine the initial IP address.

An Ethernet connection can be established with the BBB via ssh from Putty, WINscp, or any ssh terminal program. The image is setup to acquire an IP

address via DHCP from your local network and the ssh port is set to 222. You will need to determine what address the BBB has acquired. One way to do that is to access your router and look at the status for DHCP connections. Most routers have a start address for DHCP connections and if you cannot determine the correct address from the list you can always just try each address starting from the first dhcp IP address your router assigns. Only one will respond to port 222 and you should then first get a message warning about the authenticity of the connection. Answer yes to connect and you should get the BBB login prompt. Here is what my Buffalo DD-WRT routers dhcp list looks like -

Active Clients

Hostname	IP Address	MAC Address	Conn. Count	Ratio [4096]
*	192.168.0.204	cc:6d:a0:b1:a3:26	1	0%
*	192.168.0.120	b8:27:eb:1a:d3:bc	3	0%
BBB-Allstar	192.168.0.212	1c:ba:8c:a0:c3:5f	16	0%
*	192.168.0.8	00:14:bf:43:91:1a	2	0%
*	192.168.0.57	70:71:bc:4b:ae:f4	95	2%
*	192.168.0.124	00:03:2d:05:fc:b0	83	2%
*	192.168.0.2	00:19:d1:02:17:8f	5	0%
*	192.168.0.132	1c:ba:8c:a0:c3:5f	19	0%
Chromecast	192.168.0.203	6c:ad:f8:14:bd:d5	3	0%

The hostname you are looking for is BBB-Allstar. My dhcp starts at 191.168.0.200 and it shows that 192.168.0.212 was assigned to the BBB. So I would tell the terminal program to connect to that address at port 222. Your address will most certainly be different.

The login username is root and the PW is root All operations are as the root user so be careful when you do things in the system you don't have the protection of being a common user but you do have the flexibility of being able to do anything. This is no different then the Acid release on a PC that you are probably already using. Once you login you should change the root password by typing passwd at the linux prompt. Enter the new password twice and write your password down somewhere so you remember it!

Setting up your IP address

A "First Time" script has been added that runs only on the first boot of the system. It asks you to change your password and host name then asks if you want an address assigned by dhcp (the default) or if you want to assign your own static IP. You also have the option of aborting the script and doing this manually by following the directions that follow. If you set your IP address with the "first boot" script you can skip down to the time and timezone settings. The script will not run again unless you specifically run it. If you have a need to run it again type:

/usr/local/sbin/firsttime.sh

If you are like a many of us who like doing things manually and you have aborted the "first time" script you can change your password by typing:

passwd

and the system (host) name by typing:

hostnamectl set-hostname myhostname

When you log in the first time and you are setting things manually the next thing you will probably want to do is change the BBB to a static IP address. The other option is to find out what dhcp IP address was assigned at boot to the BBB and then make it a permanent address on your router. If you prefer this option you will need to configure your router to assign the same address each time based on the mac address of the BBB. To find the DHCP address that was assigned to the BBB type -

ifconfig

at the linux prompt. You will see a display similar to this -

```
eth0: flags=4163  mtu 1500
        inet 192.168.0.94  netmask 255.255.255.0  broadcast 192.168.0.255
        inet6 fe80::1eba:8cff:fea0:c35f  prefixlen 64  scopeid 0x20
        ether 1c:ba:8c:a0:c3:5f  txqueuelen 1000  (Ethernet)
        RX packets 64  bytes 17918 (17.4 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 15  bytes 2015 (1.9 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
        device interrupt 40

lo: flags=73  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10
        loop  txqueuelen 0  (Local Loopback)
        RX packets 96  bytes 7104 (6.9 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 96  bytes 7104 (6.9 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

This shows that the IP address 192.168.0.94 (inet under eth0) was assigned. Your system will undoubtedly show different values. To determine you BBB's mac address type -

ip link show eth0

Which on my system returns -

```
2: eth0:  mtu 1500 qdisc pfifo_fast state UP mode DEFAULT group default qlen 1000
   link/ether 1c:ba:8c:a0:c3:5f brd ff:ff:ff:ff:ff:ff
```

The link/ether 1c....5f is the mac address. I can provide no assistance with your router setup using this method since every router is different. I prefer the option of setting the BBB to a static IP within my LAN. If you choose that option here is the procedure to edit the IP address -

If you are a veteran to Linux then you know about editors and probably your choice will be vi. For those who have never used vi, nano is a good choice. To edit a file just type 'nano filename'

Archlinux uses netctl to control the Ethernet connection. The current configuration for the BBB wired interface is /etc/netctl/eth0 - that is a zero on the end.

I have included both the static and dhcp (default) configuration in this one file. The file includes comments on what to change when going from dhcp to static ip assignment. Edit the file -

/etc/netctl/eth0

The distribution image /etc/netctl/eth0 file looks like this -

```
Description='A basic static or dhcp Ethernet connection'

Interface=eth0
Connection=ethernet

## Uncomment either (not both) DHCP or Static IP lines
## Uncomment the following line for dhcp IPV4
IP=dhcp

## IP6 not normally used leave following lines commented

## for DHCPv6

#IP6=dhcp

## for IPv6 autoconfiguration

#IP6=stateless

## END dhcp


## Uncomment the following 4 lines for static IP

## and change to your local parameters

#AutoWired=yes

#IP=static

#Address=('192.168.0.132/24')

#Gateway='192.168.0.9'

## DO NOT change the DNS address entry unless you want to set a
## different nameserver. The default '127'0'0'1' uses bind9 to
## directly resolve a domain name from the root servers.
## This address is automatically written to /etc/resolv.conf
## at boot.

#DNS=('127.0.0.1')

## Routes not normally used - leave commented unless you have reason not to

#Routes=('192.168.0.0/24 via 192.168.1.2')
```

```
## IPV6 not normally used - leave following lines commented

## For IPv6 autoconfiguration

#IP6=stateless

## For IPv6 static address configuration

#IP6=static

#Address6=('1234:5678:9abc:def::1/64' '1234:3456::123/96')

#Routes6=('abcd::1234')

#Gateway6='1234:0:123::abcd'

## END static IP
```

As you can see the IPv6 configuration items are all commented out and you will probably have no reason to change that. To change to static IP comment the IP=dhcp line by adding a '#' at the beginning of the line. Then uncomment (remove the '#') from the static IP lines. You will need to change the Address, Gateway, and the DNS to the values that match your network. Note the netmask must be included in the IP address as a /xx. Netmask 255.255.255.0 = /24 the most common netmask. Also make sure the syntax is exactly as shown above. The examples are my local network. Note that in most cases your 'gateway' entry is the lan address of your router and the 'dns' is also set to the same address.

IMPORTANT NOTE - If you are setting up a static IP you also need to disable dhcpcd. To do this enter the command:

systemctl disable dhcpcd

should you need to re-enable the dhcp server just replace 'disable' with 'enable'.

For advanced users there are other configuration files located in /etc/netctl /examples. If you would like to read up on netctl visit Archlinux netctl

The image includes bind9 and DNS lookup is done from the Internet root servers. The DNS setting in the /etc/netctl/eth0 file is set to '127'0'0'1' to use the bind9 local domain server. This is the default and preferred setting. Should you have a need to change the nameserver it must be changed in the DNS entry in the static IP section of /etc/netctl/eth0. /etc/resolv.conf is dynamically written at boot by that setting.

Now you should be able to reboot and ssh to the new static IP address of the BBB. After logging into the BBB, check that all is well by pinging a local IP address in your network that normally returns pings. Also check that the DNS is working. I use 'ping ucsd.edu' which should resolve and return pings. Also if you want to be able to receive incoming Allstar connects you need to forward the Allstar port in your Internet router to the static IP address you have established on your BBB. The port is typically udp 4569 unless you changed it in your server setup at allstarlink.org

Setting the time and timezone

The image file has nttpd installed and running so your system should show the correct time assuming you are in the US eastern time zone. Setting the timezone in Archlinux with systemd is a breeze and it is also easy to check the current timezone status. Here is the configuration information.

Time zone

To check the current zone:

$ timedatectl status

My system gave the following output which is very informative -

```
[root@BBBdoug asterisk]# timedatectl status
      Local time: Thu 2014-05-01 22:55:57 EDT
  Universal time: Fri 2014-05-02 02:55:57 UTC
        RTC time: Fri 2014-05-02 02:56:00
       Time zone: America/New_York (EDT, -0400)
     NTP enabled: yes
NTP synchronized: yes
 RTC in local TZ: no
      DST active: yes
 Last DST change: DST began at
                  Sun 2014-03-09 01:59:59 EST
                  Sun 2014-03-09 03:00:00 EDT
 Next DST change: DST ends (the clock jumps one hour backwards) at
                  Sun 2014-11-02 01:59:59 EDT
                  Sun 2014-11-02 01:00:00 EST
```

To list available zones:

$ timedatectl list-timezones

To change your time zone:

timedatectl set-timezone Zone/SubZone

Example:

timedatectl set-timezone Canada/Eastern

This will create an /etc/localtime symlink that points to a zoneinfo file under /usr/share/zoneinfo/

In my system for US EST it looks like this -

[root@BBBdoug asterisk]# ls -als /etc/localtime
0 lrwxrwxrwx 1 root root 38 Apr 12 02:28 /etc/localtime -> ../usr/share/zoneinfo /America/New_York

Configuring Asterisk Allstar

If everything is good so far we can proceed to configuration. The Asterisk Allstar files in /etc/asterisk included with the image are files I have used but I have taken out my node numbers and passwords. You have a number of options, you can go through and reconfigure them manually or you can copy existing files to the /etc/asterisk directory. There are a number of ways to do that including tar'ing up the files, moving them over and untar'ing them or you could use winscp on a Windows computer connecting to the source Allstar computer, copying the files, then connect to the BBB and copy them back to the /etc/asterisk directory. You must configure or overwrite several of the files in /etc/asterisk for your BBB Allstar server to work with your node assignment but if you do nothing and just run the supplied files it will create a test with private node 1998 connected to a radio via simpleusb and node 1999 as a pseudo node. Of course you will not be registered so you can only do local connects but it should work. Specific files you should edit or change for your node(s) are extensions.conf, rpt.conf, iax.conf, and simpleusb.conf or usbradio.conf depending on which radio interface you are using. Do not delete the /etc/asterisk /local directory. It contains additional files which are documented below. Also DO NOT just copy your entire asterisk directory from a previous non BBB Acid install. There are file differences. There should be no zapdata.conf instead use the supplied chan_dahdi.conf. Also if you are using simpleusb there should be no usbradio.conf. Unless you have a reason to do otherwise it would be best to just copy the four files mentioned above, extensions, iax, rpt, and simpleusb config files from your previous install and edit them appropriately. Even better would be to just edit the config files supplied with the image to reflect your node. A link to a step by step procedure to do that is shown below. If you are moving from usbradio on a previous install to simpleusb just use the supplied simpleusb files and modify them as needed. In this case make sure the rxchannel command of rpt.conf points to the simpleusb stanza label for your radio node. Actual names could vary but it would be done like this:

rxchannel = SimpleUSB/usb << in rpt.conf

```
; SimpleUSB configuration
;
[general]
[usb]  << matches usb above
....rest of file
```

The 'usb' is a label and often one makes it 'usbxxxxx' where xxxxx is your node number but that is optional. It just has to match in both rpt.conf and simpleusb.conf.

Even if you change some or all of the config files in /etc/asterisk it is a good idea to keep a copy of the supplied files. There are configuration examples of an iaxrpt connection, an iphone connection, and a connection to another Asterisk server.

For step by step instructions on how to configure the BBB for the first time with your node information download this document:

http://www.crompton.com/hamradio/BeagleBoneBlackAllstar/bbb-config-

setup.pdf

The file /etc/asterisk/modules.conf has a list of Asterisk modules that are not loaded. The list is larger than normal to reduce memory usage. If you find an application in Asterisk that you want to use is not working check this file to make sure the associated module is loaded. Just comment out the noload= line for that module and restart Asterisk. In most cases you will not need to do this.

It is beyond the scope of this article but you also need to have the server and nodes you are setting up registered at the allstarlink.org website. If you are moving the server and nodes from another computer this would already be done and the only thing you would have to do is make sure your local router points to the new BBB's IP address. If you are creating a new or additional server you will first have to visit the allstarlink.org site and set up a new server and node, then wait for your new node number and password to be assigned. Then configure iax.conf and rpt.conf based on the new node and password info.

Starting Asterisk

Just for your reference here is a description of how Asterisk and the other required services are started at boot from the /etc/rc.local file.

```
ln -s /tmp/rpt_extnodes /var/lib/asterisk/rpt_extnodes
sleep 1
/usr/bin/dahdi_cfg
sleep 30
# Uncomment next line to enable openvpn
#/etc/openvpn/start-openvpn client.conf
# uncomment next line to enable firewall
#/etc/openvpn/firewall
/usr/bin/hwclock -w
/usr/bin/safe_asterisk
/etc/asterisk/rc.updatenodelist &
/etc/asterisk/local/astdb.php &
/usr/local/etc/morse_ip.sh
exit 0
```

The first line creates a log directory for the web browser. This is required because /var/log is a tmpfs and volatile - it goes away at each boot. Apache httpd expects it to be there in order to create log files.

Then we create a symbolic link for the /var/lib/asterisk/rpt_extnodes file which really resides in /tmp which is a tmpfs. The purpose of using the tmpfs is there are no writes to the SD card only internal RAM. Then there is a 30 second delay before Asterisk starts. This delay could be varied depending on your network. I found that lesser delays often resulted in lack of registration, USB detach errors, and audio problems. This is one area that needs to be investigated but for now this seems to work. You might be able to make the delay shorter but in some cases you also may need to make it longer. If you want openvpn the next line should be uncommented. You need to setup the openvpn config files before starting. The firewall, described below, can be turned on by uncommenting the next line. Then the hwclock is updated, Asterisk is started, and updatenodelist

script is run in the background. Note that rc.updatenodelist is modified to write to the /tmp directory. It is also located in /etc/asterisk. Do not delete it. Finally astdb.php is run to update the node info for use by the lsnodes script or if you load allmon and the assigned Ethernet IP address is sent to the LED in Morse code.

After Allstar is running you will need to setup your radio audio levels using simpleusb-tune-menu which will create and store your audio settings in the file /etc/asterisk/simpleusb_tune_usb.conf The file will also have your node number appended after 'usb' if you used a node number as part of the name when defining the interface in simpleusb.conf and the rxchannel in rpt.conf - example: rxchannel = SimpleUSB/usb27225 in rpt.conf and [usb27225] in simpleusb.conf

Backing up and Expanding the File System

Once you have the image the way you want it or at least reasonably complete you should make a backup. Remove the SD card insert it in your PC's card reader and save the image. Give it a unique name so you can find it like 'BBB Allstar 27225.img' inserting YOUR node number. Every time you make any significant changes to the configuration or code you should make a backup. You can keep as many backups as you want to allow going back to prior versions.

The release image has a file system size of a little less than 4G so it will fit on a 4G microSD card but the recommended size is 8G or 16G. When you write the image to a larger card the file system will still be limited to the 4G size and must be expanded to utilize the entire card. Here are instructions on how to expand the file system. Make a backup before expanding the file system just in case. The jury is out on this but utilizing the entire size of the card may extend its life. It depends on how wear levelling is done and if it is done at all on the SD card. Writes to the SD card are kept to a minimum and it is likely the card will last a long time whether or not it is expanded. Expanding the file system has the benefit of more file space if you should need it but also the disadvantage of a very large image to backup (8M or 16M vs 4M). If you do expand the file system be sure to make another backup both before in case you need to recover and after to save the changes.

How to Expand the File System

Firewall

A firewall has been added but NOT enabled by default with the following rules:

- Ports 4560-4590 UDP - use for iax if needed
- Ports 5198,5199 UDP - Echolink
- Port 5060 - SIP - Remove comment in file to allow
- Port 80 - HTTP - Remove comment in file to allow

- Port 222 - SSH - Allowed
- Port 1194 - openvpn - configure to your requirements
- All other incoming ports are blocked

The firewall default when enabled is to allow ports 4560-4590 which can be used for iax if needed and ports for echolink. SIP, HTTP, and SSH other than port 222 are not allowed outside of the local network by default. Simply uncomment those rules in the firewall file if you want to allow Internet traffic to those ports. The local LAN is not limited by these rules and can access any port. The firewall definitions are in /etc/openvpn/firewall If you change the settings in this file or comment/uncomment a rule rerun the firewall file. To rerun type:

/etc/openvpn/firewall

To view the current firewall settings type:

iptables -L -v -n

By default the firewall is disabled. You can enable it by uncommenting the firewall line in /etc/rc.local In most cases if you are using a good gateway router to the Internet the firewall is redundant. You would forward ports in your router to your BBB as needed and only the ports that were needed. If you don't currently have any firewall protection ahead of your BBB system it is recommended that the firewall be enabled. This would be the case if you were DMZ'ing to the BBB or using it in a situation with unknown external firewall protection.

Additional Notes

The openvpn package is now installed for your convenience. Most will probably not use it but it can be a valuable asset in connecting your distant nodes securely. By default it is not enabled. To enable it uncomment the openvpn line in /etc/rc.local. You must setup the configuration files. There are example files in /etc/openvpn and further information on its use can be found at openvpn howto and documentation

This release uses dahdi which is the replacement for zaptel. There should be NO zaptel.conf file or any other zaptel related files brought over to the BBB system. The file chan_dahdi.conf should remain in the /etc/asterisk directory.

There are some additional sound files provided on the image to facilitate the 'The time is' message script running outside of app_rpt. I created a script to do that so the volume could be changed on the fly. All additional scripts are located in /etc/asterisk/local. The saytime.pl script is called by a cron job every hour on the hour and sent to the local nodes on the BBB server as specified in the /etc/asterisk/local/saytime.pl file. Modify this file to include your nodes if you would to make this operational. The file uses sox to alter the volume transmitted over the local radio. The volume can be changed in the file. Delete or change the cron file to turn this off or change the time(s) that it plays. More information is

available as text in the saytime.pl file.

My script lsnodes to display and control a node locally or remotely from a web page is included. Instructions on how to use it are located at lsnodes Allstar Display and Control There is no page authentication (.htaccess) installed so if you are running lsnodes public on the web and want security you would have to create .htacess in the /srv/http/lsnodes directory and assign a password. This is described in the documentation.

To connect to your node from your local LAN enter the following address in your browser. Note that this does not work well with MS Explorer! Use Firefox, waterfox, or Chrome which all should work.

http://your BBB local IP address/lsnodes/lsnodes_form.html

This connects at the standard port 80. See the lsnodes docs for info on how to change the port if you desire. You can access the BBB from anywhere on the internet if you forward port 80 to the BBB's local IP address in your Internet router. Note that you would also have to open the BBB's firewall to port 80 or whatever port you are using in /etc/openvpn/firewall.

Disregard the lsnodes installation procedure since it is already installed on the BBB image. I did not implement echolink on the BBB image but you could certainly add it by setting up the echolink.conf file and forwarding the correct ports from your router. The file is currently named echolink.xxx

I am leaving a voice ID file and scripts to create id's as an example. They are created in combine.sh in the /etc/asterisk/local directory. Obviously you need to change it to your call etc. but this shows how to create ID scripts from 'Allison' based voice files already on the system. The combine.sh file combines asterisk sound files to form the ID. Edit it to use the letters of your call and whatever you want to say. Check the /var/lib/asterisk/sounds directory for the names of available files. When combining all of the files must be the same type, typically this would be ".gsm" Once the combine.sh file is configured with your selected sound files just run the script and it will create the /etc/asterisk/local/node-id file. Then using /etc/asterisk/local/node-id as the id file in rpt.conf will play the ID file at the ID interval. Of course this is just one way to play a voice ID. You can also record your own voice file and play it.

There are also example scripts for connection and disconnection at specific times. Again these are just examples and must be tailored to use on your system. Scripts are generally called from a cron job at a specific time or times. The scripts are not limited to connect and disconnect. They can execute any command you would do manually.

The following cron entries are installed in the BBB image. The first line calls the saytime script every hour on the hour. Edit this script to the node number you want this announced on. The second line updates the node information data daily. This is used by lsnodes or allmon.

```
00 0-23 * * * cd /etc/asterisk/local; perl saytime.pl > /dev/null
03 03 * * * cd /etc/asterisk/local; ./astdb.php
```

USB Woes. If you have used the USB interface much you are most certainly aware that it sometimes can be the cause of all kinds of problems. The BBB is especially susceptible because unlike a PC it is not directly grounded. Sneak paths of especially AC current on the ground leads can totally confuse the USB interface. One example of how this actually happened was a wall wart type supply used to power the BBB had a high level - 45 volts of AC - floating on the ground and +5 output. Since this was equal on both polarities (common mode) it did not effect the performance of the BBB but when a device such as the DMK URI was connected this AC was grounded through the USB interface to the URI and then to the radio to ground inducing AC onto the USB power and data lines. This is just one example of how this can happen so beware if you have USB issues. Check your Power supply and grounds!!! The small switching wall-wart type supplies do not have a AC ground lug. I checked a Powergen supply here and sure enough it had a fairly high level of AC riding on the DC ground measured to the line AC ground. I have been using them on a number of boards without any noticeable problems but my thought is that these things are just plain trouble waiting to happen. A conventional transformer coupled three terminal regulator supply or higher quality switching supply might be a better idea. Another thing to be especially careful of is the ground on your radio. On TX a 50 watt radio could draw 10 amps or more. If the radio ground came loose it would try to seek a ground back through the URI and USB and the high current flow could destroy the URI and/or the BBB. The USB interface and the DMK URI in particular are also very susceptible to RF interference especially in the HF range. Ferrites on the USB and radio leads usually solves this problem. For really tough problems a USB isolator might be the answer. The ADUM4160 USB isolator is available from a number of sources and is in the $30 range.

Power woes. As mentioned in the previous paragraph power supply issues are often the cause of problems with the BBB, One area that is particularly weak on the BBB is the 5V power connector. I don't like barrel connectors as I often find them to be intermittent and a poor choice to run a computer. An intermittent power connector can cause reboots and all sorts of problems. Check your connector by wiggling it after applying power without the SD card in place and see if the power light blinks. I have one board that has a bad connector and reboots sporadically. For best reliability I would either replace or better yet hard wire the power to the connector. It is also probably acceptable to use the mini USB as the power source as long as you are not running more than the BBB and a USB sound FOB.

While technically it is possible to use a usb hub and an additional sound FOB for a second node it is not recommended. A far better choice would be to use a second BBB setup as another server. You would need to select a different iax upd port like, 4568, configure it at allstarlink.org in your sever setup, in your local iax.conf, and add an entry in your router port forwarding. In this case you would have two entries in your port forwarding for iax, one to the IP address and port 4569 of the first server and one to the IP address and port 4568 of the second server. To be able to connect between local servers you most likely would need to add the local addresses in each servers rpt.conf. It might look something like this:

At node/server 40638 -

40638 = radio@127.0.0.1/40638,NONE
40689 = radio@192.168.1.100/40689,NONE

At node/server 40689 -

40689 = radio@127.0.0.1/40689,NONE
40638 = radio@192.168.0.101/40638

Good audio quality is a pet peeve of mine. In particular usbradio even on PC systems often suffers from audio distortion. I hear it all the time on the air. Often it is on transmit so the node owner has no knowledge of it happening unless someone says something. Please consider this when using your node. We are looking into ways to improve this but for now whether you are using a BBB or PC this distortion can and does happen. Consider using simpleusb especially with the BBB. Also the BBB image has "radio relax" turned off. The acid release has this turned on by default. When it is on voice signals can false the DTMF decoder and each time this happens there is a hole in the audio when it mutes thinking there is a real DTMF signal. You can see this in the client when it logs the false DTMF character. Leaving it off has no significant impact on real DTMF detection and eliminates the falsing.

The iLBC codec has been added to the BBB image. iLBC was developed by Global IP solutions and was included in Asterisk for many years until it was removed due to possible licensing issues. Google bought GIPS in 2011 and made the codec open source. It uses a 20ms frame rate and a fixed 15.2kbit/s bitrate in Allstar. The use of this is optional and would require another system with this codec installed for its use. It has very good quality at low data rates, about one third of the standard g726aal2 that is typically used. It is particularly useful in limited bandwidth situations and degrades much more gracefully than g726aal2 on poor circuits. Note however that the transcoding of iLBC to and from other formats takes considerably more CPU. It does work well on the BBB with several nodes connected but you need to be aware of possible issues when larger numbers of nodes are connected of differing formats. For more information on iLBC see http://en.wikipedia.org/wiki/Internet_Low_Bitrate_Codec

iLBC is enabled by default as a codec option in conf If you want to disable it you can comment the:

allow ilbc

line in the general stanza (for outgoing connections) and radio stanza (for incoming connections) of iax.conf You can also change the preference order of the codecs by changing its position relative to the other allow entries.

You can check the format of connected channels in the Asterisk client with the command:

iax2 show channels

One of the biggest complaints Allstar users have besides the lack of coherent documentation is the fact that if you are connected to a large number of nodes you get constant and annoying connect and disconnect messages. There is an easy way around this with a couple of configuration lines in rpt.conf. To disable

the message except your own put the following lines in rpt.conf

holdofftelem=1
telemdefault=2

These lines already exist in the default example rpt.conf in the image. If you copy over another rpt.conf you will have to recreate them.

The entire asterisk allstar and dahdi source code is located in the /usr/src/utils directory and the kernel header files are located in /usr/src. So if necessary you can do a recompile of dahdi or asterisk. This should only be necessary if you are making changes to the code. I would NOT recommend updating the Archlinux system (pacman -Syu) unless absolutely necessary. Always make a image backup before doing any major changes so you can return to where you were if something goes wrong. Archlinux is a bleeding edge Linux OS and has continual updates. If you do an entire system update you will most likely get a new kernel and need to recompile and install the dahdi and asterisk code. I have also had mixed results with other kernel versions. The 3.8.13-24-ARCH which is what the image is based on works. There is no guarantee for anything else. If you need a package that won't load unless there is an upgrade either use the Archlinux rollback machine and get the package for the installed kernel or get the source and compile under the current kernel version. See Archlinux Downgrading Packages

The linux locate which is handy to locate files on the system is installed but the daily update is turned off. This is to avoid unnecessary writes to the SD card. If you have a need to find a file that was added after the last update run updatedb first.

If you plan on doing any serious code work on the BBB you should consider buying a serial adapter cable. This plugs onto J1 on the BBB board and a USB port on the host computer. It works very well with putty in serial mode in Windows7 but could be used with any computer that offers a serial connection. Windows should load the FTDI driver automatically. Then use device manager to determine or change the comm port settings. The baud rate is 115200. You will see all boot messages and get a login prompt. The adapter is available from Adafruit - Standard FTDI Cable

For those of you who are tinkerers and like to write and modify code be sure the package(s) you load or write is utilizing RAM rather than the SD card for writing data on the BBB. It is OK to write occasional information that must be retained to the SD card but something that does continuous updates that can be volatile (not survive a boot) should be written to one of the tmpfs areas. Both /tmp and /var/log/ are tmpfs on this BBB image and anything written to those directories or any directory below use RAM.

We love contributions! Not money but rather things you have done to improve your BBB that we may either include in the distribution or as links to optional add-ons. This is a community effort and input is always welcome.

Talking about code work and the value of having a serial connection, one of the things that held up distribution an extra week was work on the boot process for

Archlinux on the BBB. The BBB has a complex boot process using Uboot which involves the on-board eMMC and the MicroSD card. With the stock Archlinux the BBB wants to boot from the eMMC unless you hold down the boot switch at power up. If you wholesale wipe out the on-board eMMC it will boot fine but we did not want to force users to do that. Dave, KB4FXC came up with a method that appears to work fine and involves no modification to the stock eMMC. Your Allstar image should boot without any user intervention. The rev C BBB board is also booting correctly as of v1.04.

Version information can be found in the file /etc/bbb_allstar_version

To Do List

- GPIO for PTT and COS when using a generic sound FOB. This will probably be in the next release.
- A watchdog timer to guard against system or usb crashes. Possibly next release.
- More testing of USB radio as this possibly might work for one port
- Further testing of multiple ports with a USB hub running simpleusb. A single port works fine through a hub.
- Building of scripts to make the install process more seamless.
- Testing of a wireless network connection. This would require a powered hub.
- Upgrade of the app_rpt USB sound interface code. This is a weak point in app_rpt and one of the reasons why we have to jump through so many hoops to get it to work on less powerful I/O and processors.

Final Comments

Finally let me say this is a work in progress and I will be continually updating both the image and these notes over time. I would be glad to answer any questions or try to resolve any problems you may have but please understand that at this point you need to have or gain a little knowledge about using Linux and at least be able to follow the above directions in order to get this running. I would also like to hear about your experience with Allstar and the BBB and this code. If you have ideas on how the code can be improved or you have expertise and would like to be a part of the project please contact me at my QRZ email address (WA3DSP)

Please also forward any comments you may have on this web page. I am interested in how well the instructions worked for you and any thoughts you may have on how they can be improved.

I hope you have as much fun using the BBB with Allstar as I have had implementing it. We have started a mailing list for the ARM processor Allstar and

specifically, at least for now, the BBB. To subscribe to the arm-allstar mail list visit this link:

http://lists.hamvoip.org/cgi-bin/mailman/listinfo/arm-allstar

Try to keep discussion specific to the BBB/Allstar project on the arm-allstar list rather than apt_rpt.

Doug, WA3DSP

Credits

- Mihhail Jakovlev, ES1BIS - Mihhail got me started with all this. Without his sharing of the initial compilation in RPi Pidora I would probably not have gotten started.
- Dave McGough, KB4FXC - Dave and I got together after I moved to the BBB and Archlinux. His input and testing was a great help. Dave fixed the boot problem and was always there when I needed help and he has contributed greatly to the refining of the BBB image. We will continue to work together on future projects.
- Chris Kovacs, W0ANM - Chris is responsible for getting the WIFI operational and documented and for many of the neat scripts like weather, TTS, backup and restore, etc. He has been a great help in testing and project discussions.
- Credit also to Jim Dixon, WB6NIL and Tim Sawyer, WD6AWP for initially contributing scripts for node and net setup.
- And I must give credit to the place I got the most help from - Google and the Internet! - without that resource I would not have surmounted many of the obstacles. There is a reason why "Google is your Friend"

Links

- Archlinux Beginners Guide
- Systemd Users Guide
- BeagleBone Black WIKI
- BeagleBone Black Accessories
- BeagleBone Black Reference Manual
- Linux ate my RAM
- Allsarlink.org main site
- Configuring a two node Allstar system

- simpleusb.conf
- DMK URI USB interface
- USB Allstar Adapters
- Acid System Admin Manual (old but useful)
- Mac SD card utility

How To Notes

- Wireless How-to (Version 1.2 or greater ONLY!)
- BBB Global Scripts
- BBB Text to Speech How-to
- How to setup iaxrpt
- How to setup the configuration files
- iLBC Install Instructions for Acid (NOT for BBB)
- How to use simpleusb and improve audio (for Acid installs not BBB)
- How to Backup your BBB (Important ver 1.2 update)
- How to implement the BBB watchdog timer (ONLY VERSIONS <1.2!!)
- General Allstar Howto's (not all relate to BBB)

User Tech Notes and interesting sites

(note these may or may not be suitable for the BBB, use at your own risk)

- Allstar and the GPIO Fan Script for DMK Engineering URI by KP4TR
- ARRL News on Allstar Node by KI6PSP
- A fan mount for cooling Motorola Maxtrac Radios

Download this web page in PDF - BeagleBoneBlack.pdf

If you would like to translate this page content, please use the Google Translate feature below:

Google Translate My Page