Networking Guide for BeagleBone

by Brian Fraser
Last update: Sept 19, 2017

This document guides the user through:
1. Setting up and configuring networking for the BeagleBone with the host.

Table of Contents
1. Network Setup..............................................................................................................................2
2. Wired Ethernet Network..................................................................................................................3
3. Ethernet over USB..........................................................................................................................6
   3.1 Initial Connection between Host and Target..............................................................................6
   3.2 Connecting Target to Internet..................................................................................................7
4. Direct Ethernet Connection (Optional)..........................................................................................10
5. Linux Network Commands............................................................................................................13
   5.1 General Commands....................................................................................................................13
   5.2 Set a static IP Address...............................................................................................................13
      5.2.1 Temporarily Set IP Address.................................................................................................13
      5.2.2 Permanently Set IP Address via Ubuntu’s GUI Tools.............................................................13
      5.2.3 Permanently Set IP Address via Configuration File.............................................................14

Note: Guide not yet tested in the SFU Surrey Linux Lab (SUR4080). Some changes may be needed.

Formatting:
1. Commands starting with $ are Linux console commands on the host PC:
   $ echo "Hello world!"
2. Commands starting with # are Linux commands on the target (BeagleBone):
   # echo "Hello embedded world!"
3. Almost all commands are case sensitive.

Revision History:
• Sept 19: Initial version for 2017
1. Network Setup

You need to have an internet connection to both your Linux development OS (host PC) and the BeagleBone. Plus, you need to have your development OS be able to connect to your BeagleBone (we'll use SSH), and your BeagleBone be able to connect to your development OS (we'll use NFS). There are a couple ways you can connect your devices.

1. Have your development computer and your BeagleBone plugged into the same wired network, such as a home Ethernet network (see section 2).

2. Have your BeagleBone connect to your development computer using Ethernet over USB, and your development computer connect to the internet via WiFi or wired Ethernet (see section 3).

3. Have your BeagleBone directly connected to your development computer via a single Ethernet cable (not recommended) (see section 4).

See the necessary section below for the option you are using. Section 5 covers some useful Linux networking commands.
2. **Wired Ethernet Network**

This section assumes you are connecting your BeagleBone to the same wired network that your development PC is connected to.

1. If using a virtual machine on the development computer, completely shutdown (power down, not just close/suspend) your virtual machine and change the network adapter type to “Bridged” instead of “Network Address Translation (NAT)”.
   - Under VMWare Player, select the VM and then “Edit Virtual Machine Settings” → Network Adapter → Bridged. OK to “Replicate physical network connection state”. Restart the VM.

2. Connect an Ethernet cable to the target board and have it get an IP address from your network's DHCP server (likely your home router).
   - The board should automatically acquire a IP address in a few seconds (~30), but you may need to re-run either `dhclient` or `udhcpc` (depending on which is installed) to force it.
   - You may need to reboot the target to have it acquire an IP address at startup:
     ```bash
     # reboot
     ```

3. Check the IP address of the target board using the serial connection:
   ```bash
   # ifconfig
   [root@bfraser-beagle ~]# ifconfig
   eth0: Link encap:Ethernet  HWaddr c8:a0:30:aa:dd:a0
       inet addr:192.168.0.145 Bcast:192.168.0.255  Mask:255.255.255.0
       inet6 addr: fe80::caa0:30ff:feaa:dda0/64 Scope:Link
       UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
       RX packets:22 errors:0 dropped:0 overruns:0 frame:0
       TX packets:53 errors:0 dropped:0 overruns:0 carrier:0
       collisions:0 txqueuelen:1000
       RX bytes:4176 (4.0 KiB)  TX bytes:9945 (9.7 KiB)
       Interrupt:40
   lo:    Link encap:Local Loopback
       inet addr:127.0.0.1  Mask:255.0.0.0
       inet6 addr: ::1/128 Scope:Host
       UP LOOPBACK RUNNING  MTU:65536  Metric:1
       RX packets:0 errors:0 dropped:0 overruns:0 frame:0
       TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
       collisions:0 txqueuelen:0
       RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)
   usb0:  Link encap:Ethernet  HWaddr 6a:41:80:a7:bf:db
       inet addr:192.168.7.2  Bcast:192.168.7.3  Mask:255.255.255.252
       UP BROADCAST MULTICAST  MTU:1500  Metric:1
       RX packets:0 errors:0 dropped:0 overruns:0 frame:0
       TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
       collisions:0 txqueuelen:1000
       RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)
   ```
   - Look for `eth0`’s “inet addr:”, such as:
     ```
     inet addr:192.168.0.145
     ```
     (Highlighted in above example)
   - If you do not see `eth0`, it means the board has likely not yet found an IP address. Wait up to a minute longer for it to get an IP address. If it does not, double check all Ethernet cable connections.
connections and reboot the target.

- You may see (and ignore) some messages on the target such as:
  ```
  [   24.172268] net eth0: phy 4a101000.mdio:01 not found on slave 1
  [   29.810445] libphy: PHY 4a101000.mdio:01 not found
  ```

4. Check the IP address of the Host PC:
   ```
   $ ifconfig
   $ ifconfig
   ```
   ```
   eth0  Link encap:Ethernet  HWaddr 00:0c:29:7f:47:bc
      inet addr:192.168.0.114  Bcast:192.168.0.255  Mask:255.255.255.0
      inet6 addr: fe80::20c:29ff:fe7f:47bc/64 Scope:Link
      UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
      RX packets:196832 errors:0 dropped:0 overruns:0 frame:0
      TX packets:5079 errors:0 dropped:0 overruns:0 carrier:0
      collisions:0 txqueuelen:1000
      RX bytes:41734380 (41.7 MB)  TX bytes:1109534 (1.1 MB)
   ```
   ```
   lo    Link encap:Local Loopback
      inet addr:127.0.0.1  Mask:255.0.0.0
      inet6 addr: ::1/128 Scope:Host
      UP LOOPBACK RUNNING  MTU:65536  Metric:1
      RX packets:747 errors:0 dropped:0 overruns:0 frame:0
      TX packets:747 errors:0 dropped:0 overruns:0 carrier:0
      collisions:0 txqueuelen:0
      RX bytes:277191 (277.1 KB)  TX bytes:277191 (277.1 KB)
   ```
   - Find eth0’s IP address, such as:
     ```
     inet addr:192.168.0.114
     ```
   - If running in a virtual machine, check the host OS’s IP address as well: under Windows run `ipconfig` from the command line.

5. The addresses of the host and target (and host OS if running a VM) should be on the same subnet (have the first couple octets the same). If they are not the same then check the troubleshooting section below.
   - Note that if your target device is plugged into a different network than your host PC they will be on different subnets. This may be the case in the SFU labs.

6. From the **host PC**, ping the **Target**. Command and output shown below. Press Ctrl-C to stop the pinging as needed. Replace the IP address with whatever address your target has.
   ```
   $ ping 192.168.0.145
   ```
   ```
   PING 192.168.0.145 (192.168.0.145) 56(84) bytes of data.
   64 bytes from 192.168.0.145: icmp_seq=1 ttl=64 time=1.10 ms
   64 bytes from 192.168.0.145: icmp_seq=2 ttl=64 time=0.585 ms
   64 bytes from 192.168.0.145: icmp_seq=3 ttl=64 time=0.492 ms
   ^C
   --- 192.168.0.145 ping statistics ---
   3 packets transmitted, 3 received, 0% packet loss, time 2001ms
   rtt min/avg/max/mdev = 0.492/0.727/1.105/0.270 ms
   ```

7. From the **target**, ping the **host PC** using your serial terminal. Command and output shown below. Press Ctrl-C to stop the pinging as needed. Replace the IP address with whatever address your host has.
# ping 192.168.0.114
PING 192.168.0.114 (192.168.0.114) 56(84) bytes of data.
64 bytes from 192.168.0.114: icmp_req=1 ttl=64 time=1.49 ms
64 bytes from 192.168.0.114: icmp_req=2 ttl=64 time=0.619 ms
64 bytes from 192.168.0.114: icmp_req=3 ttl=64 time=0.620 ms
64 bytes from 192.168.0.114: icmp_req=4 ttl=64 time=0.707 ms
^C
--- 192.168.0.114 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3004ms
rtt min/avg/max/mdev = 0.619/0.861/1.499/0.370 ms

8. Ensure your target and host can both access the internet:
$ ping google.ca
# ping google.ca

- If the ping fails, double check your network setup.

9. Troubleshooting:

- If the VM is not on same subnet as host OS, then make sure the VM's network adapter is configured for Bridged mode, instead of NAT. NAT uses the host OS's IP address for access to the network which will work fine for outgoing connections (such as browsing to Google), but not work for incoming connections (such as running an NFS server). You may need to restart the VM if you make any changes.

- If your target board (BeagleBone) is on a different subnet than your host PC, double check that your host PC and target board are plugged into the same network. For example, you could have a problem if the host PC is connecting to a wireless network but the target device is connecting to a completely separate wired network. Reboot the target device to have it acquire a new IP address.
3. Ethernet over USB

Expected setup is your host PC (in a VM or not) is connected to the Internet via WiFi or Ethernet; host connected to target via a serial cable, and using a USB cable between host and target for emulated Ethernet. No Ethernet cable is required.

3.1 Initial Connection between Host and Target

1. If using a VM:
   • Configure your VM's network connection to use NAT (Network Address Translation).
   • Launch the VM.
   • Ensure that inside the VM you are able to access the Internet by pinging Google (for example).
     $ ping google.ca

2. Connect the USB mini cable from the host to the target (connection is on the BeagleBone, to the left of the Ethernet port). This cable was included with the BeagleBone.
   • Note that USB mini port will also power the target so to physically cycle the power you may need to disconnect both the USB and the power connector. You can run with just the USB power for most operations; however, the full power plug is needed to power the device when it is reflashing the system image or powering any external devices (such as USB devices).
   • The USB cable required is a USB-A (host side) to USB-Mini (target side). It is not the USB-Micro cable which connects to the Zen Cape, nor a TTL232-to-USB cable if you have one.

3. If using a VM, map the USB connection to the VM.
   • In VMware, use the “Player” button in the top left, go to Removable Devices → Linux Foundation BeagleBoneGreen and select to Connect.
   • In Virtual Box, via VM's menu at the very top of the window select Devices → USB Devices → Circuitco BeagleBoneGreen

4. Check the network connection with ifconfig. It should show eth1 with an IP address 192.168.7.1.
   $ ifconfig
   • The target will default to have a fixed IP address of 192.168.7.2 via the emulated USB network connection (usb0).

5. Ping the host from the target, and the target from the host (via your serial connection) to ensure a working connection:
   $ ping 192.168.7.2
   # ping 192.168.7.1
   • See previous section for more details on the commands.
6. Troubleshooting

- Ensure you have connected the BeagleBone’s micro USB connector to the computer (instead of the Zen cape’s; it’s OK to have them both connected).
- If using a virtual machine, ensure the BeagleBone has been mapped to the Linux VM.
- Ensure that the host is detecting the BeagleBone and creating an Ethernet-over-USB connection for it. To do so, use `ifconfig` as follows:
  - Disconnect the BeagleBone from the host and run
    ```
    $ ifconfig
    ```
  - Reconnect the BeagleBone to the host, wait until boot-up has finished, and rerun
    ```
    $ ifconfig
    ```
  - Compare the two to ensure that a new networking adapter appears. Also, note name of the adapter (such as `eth7`, `usb0`, or something more cryptic like `enx8030dc9359be`), and the HWaddr (MAC address) on this connection for later use.
  - If the target reboots, the host will lose the Ethernet-over-USB connection `eth1`. The connection should come back when the board finishes rebooting.
  - If the host does not get the 192.168.7.2 IP address when the BeagleBone is connected, try assigning a static IP to the Ethernet-over-USB on the host. See Section 5.2 for directions.
  - If software on the target stops working, the Ethernet connection on the host may disappear.
  - If unable to ping between the devices, run `ifconfig` on each to ensure correct IP addresses.
    - Target's `usb0` connection should be IP address 192.168.7.2.
    - Host's `eth1` connection should be IP address 192.168.7.1.

3.2 Connecting Target to Internet

By following the above steps, you'll be able to ping between the host and the target. However, the target will not yet be able to access the internet. This section configures the host to allow the target to access the internet through its network connection.

1. Ensure your host PC (the Linux VM, likely) has access to the internet:
   ```
   $ ping google.ca
   ```
2. On the host PC, you should have two network adapters, likely named `eth0` (actual network connection) and `eth1` (virtual connection to target). Find them with:

---

1  Directions from Molloy (2014) Exploring BeagleBone, Ch2.
```bash
$ ifconfig

eth0
Link encap:Ethernet  HWaddr 08:00:27:a3:83:38
inet addr:10.0.2.15  Bcast:10.0.2.255  Mask:255.255.255.0
inet6 addr: fe80::a00:27ff:fea3:8338/64 Scope:Link
UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
RX packets:2391 errors:0 dropped:0 overruns:0 frame:0
TX packets:707 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:1000
RX bytes:2670933 (2.6 MB)  TX bytes:85463 (85.4 KB)

eth1
Link encap:Ethernet  HWaddr c8:a0:30:aa:dd:a2
inet addr:192.168.7.1  Bcast:192.168.7.3  Mask:255.255.255.252
inet6 addr: fe80::ca80:a0ff:feaa:dda2/64 Scope:Link
UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
RX packets:46 errors:0 dropped:0 overruns:0 frame:0
TX packets:46 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:1000
RX bytes:626 (626.0 B)  TX bytes:10416 (10.4 KB)

lo
Link encap:Local Loopback
inet addr:127.0.0.1  Mask:255.0.0.0
inet6 addr: ::1/128 Scope:Host
UP LOOPBACK RUNNING  MTU:65536  Metric:1
RX packets:352 errors:0 dropped:0 overruns:0 frame:0
TX packets:352 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:0
RX bytes:27358 (27.3 KB)  TX bytes:27358 (27.3 KB)
```

- Note: Connection to internet may come up as something like ens33, and the target’s Ethernet over USB connection may come up as something like enxb0d5cc4700d7.

3. On the host, use the `iptables` program to configure the Linux kernel firewall rules:
   ```bash
   $ sudo iptables --table nat --append POSTROUTING --out-interface eth0 -j MASQUERADE
   $ sudo iptables --append FORWARD --in-interface eth1 -j ACCEPT
   ```

4. Turn on IP forwarding:
   ```bash
   $ sudo sh -c "echo 1 > /proc/sys/net/ipv4/ip_forward"
   ```

5. Configure the target to use the host (via Ethernet over USB) as the default gateway:
   ```bash
   # route add default gw 192.168.7.1
   ```
   - You can check this succeeded by viewing the routing tables on the target using `route`:
     ```bash
     # route
     Kernel IP routing table
     Destination Gateway Genmask Flags Metric Ref Use Iface
default brian-VirtualBo 0.0.0.0 UG 0 0 0 0 usb0
     192.168.7.0 * 255.255.255.252 U 0 0 0 0 usb0
     ```

6. Check the target has an internet connection:
   ```bash
   # ping 8.8.8.8
   ```
   - This should successfully ping the server. Note, you won't yet be able to ping google.ca!

7. **These settings will not persist between reboots**, so you must re-do all of these steps each time you reboot in order to re-enable the settings.
   - You may be able to edit the `/etc/network/interfaces` file to make these settings permanent. Consult guides online for more information.
8. **Configure a DNS:**

   - Append a new domain nameserver (DNS) to the end of `/etc/resolv.conf`:
     ```bash
     # echo "nameserver 8.8.8.8" >> /etc/resolv.conf
     ```
   - Where 8.8.8.8 is a free public DNS owned by Google.
   - View the `/etc/resolv.conf` file to ensure it looks reasonable:
     ```bash
     # cat /etc/resolv.conf
     nameserver 127.0.0.1
     nameserver ::1
     nameserver 8.8.8.8
     ```
   - Test with:
     ```bash
     # ping google.ca
     ```
   - Note that this step won't need to be repeated after a reboot as the `/etc/resolv.conf` file is persistent.

9. **Trouble shooting:**

   - If you rebooted either your target or host, you'll need to redo the settings on that device as the settings are not persistent.
   - If you get the error message: "ping: unknown host google.ca" when trying to ping google.ca, then you don't have a DNS server setup correctly. Double check you have completed the DNS steps above.
4. Direct Ethernet Connection (Optional)

You can directly connect the target device to a host PC using an Ethernet cable (or crossover). You may skip this section if you are connecting the BeagleBone through either a wired Ethernet connection to a switch/router which supports DHCP, or using Ethernet over USB.

Expected connection is your PC (or virtual machine) is connected to the web via either WiFi or Ethernet, and you are going to connect an Ethernet cable directly from your PC to the target. (i.e., the target is connected to the host only, not to any network such as a router).

1. On the target, via a serial connection, set it to a static IP address:
   # ifconfig eth0 192.168.2.2
   • Note that this will revert back to DHCP when the target is rebooted. You can make the change permanent by editing /etc/network/interfaces and adding:
     auto eth0
     iface eth0 inet static
     address 192.168.2.2
     netmask 255.255.255.0
     network 192.168.2.0
     broadcast 192.168.2.255
   • After changing the file, restart networking on the board (or reboot)
     # /etc/init.d/networking restart

2. Edit the settings for your virtual machine.
   **Virtual Box**:
   • You must have your VM powered down, not just suspended.
   • Right click the VM in your Oracle VM VirtualBox Manager and select Settings.
   • Under Network, enable Adapter 1 and set to NAT.
     This adapter will give your VM access to the internet through your host OS.
   • Under Networking, enable Adapter 2 and set to Bridged.
     • In the “Name” drop-down select your Ethernet adapter.
     • This adapter will be the connection to the target.
     • Launch your virtual machine.

   **VM Ware**
   • Click the “Player” button in the top left, and select Manage, then Virtual Machine Settings.
   • On the Hardware tab, select “Add…”, and add a new “Network Adapter”
   • Change new network adapter to:
     • Network Connection: “Bridged”
     • Check “Replicate physical network connection state”
     • Under “Configure Adapters”, **only check the wired Ethernet adapter on your computer** (ensure it's not a wireless one, and not a BlueTooth one)
• Press OK.

3. On the host (inside your VM), edit the /etc/network/interfaces file and add:

```bash
auto eth1
iface eth1 inet static
  address 192.168.2.1
  netmask 255.255.255.0
  network 192.168.2.0
  broadcast 192.168.2.255
```

• Note that eth1 may need to be changed to another number if you are running Linux natively (likely eth0), or have a different number of network adapters connected to your VM. Use `ifconfig` to list Ethernet adapters and pick the correct number.

4. Restart the networking:

```bash
$ sudo /etc/init.d/networking restart
```

5. Test your host's internet settings on the host:

• Run `ifconfig` will likely have an IP address in the range of 10.0.2.15 and eth1 should have an IP address of 192.168.2.1.

• Ping Google to prove you have a good internet connection.

```bash
$ ping google.ca
```

• See troubleshooting below if there are any problems.

6. Ping the target from the host, and the host from the target.

```bash
# ping 192.168.2.1
$ ping 192.168.2.2
```

7. Troubleshooting

• Run `ifconfig` on the target and host to ensure they have the correct addresses.

• If your VM cannot access the Internet (say pinging Google or installing via `apt-get`), then ensure that your host OS (Windows, for example) has a good connection to the Internet.

• Ensure your VM has the correct network adapters installed. In VirtualBox, check that it has enabled the network connection:

  • In the VM's menu, select Devices → Network

  • Ensure that two network connections are listed and that both are enabled (filled background around the icon).

• Check that the network cable connected correctly.

• Check that the connection is enabled in Linux. If `ifconfig` does not show `eth0` (on the target) or `eth1` (on the host), then enable the adapter using a command such as:

```bash
$ sudo ifconfig eth0 up
```

• Double check the VM's networking settings. Ensure you select NAT and Bridged correctly.

• If using VMWare, ensure that your VM’s bridged network adapter’s “configured adapter” is set to only connect to your Ethernet (and not bluetooth, WiFi, or any other systems).

• For more information on configuring the network settings, see:

  https://help.ubuntu.com/community/NetworkConfigurationCommandLine/Automatic
5. Linux Network Commands

5.1 General Commands
Here are some useful Linux commands for working with Ethernet (may need root access):

- Display Ethernet configuration information:
  
  ```
  $ ifconfig 
  or 
  $ ip addr 
  ```

- Turn off, and turn on Ethernet connection (to cause it to redetect it):
  
  ```
  $ sudo ifconfig eth0 down 
  $ sudo ifconfig eth0 up 
  ```
  Or:
  
  ```
  $ sudo ifdown eth0 
  $ sudo ifup eth0 
  ```

- Run DHCP to acquire an IP address (only on some systems):
  
  ```
  $ dhclient 
  ```

5.2 Set a static IP Address
Usually, IP addresses are assigned automatically via DHCP (for normal Ethernet), or by the configuration of the Ethernet-over-USB system. However, when these are not working it is possible to force the connection to have an IP address manually using static IP address assignment.

5.2.1 Temporarily Set IP Address

- Force a new (static) IP address:
  
  ```
  $ ifconfig eth0 192.168.0.212 
  ```

- Note that when the system reboots, these changes will be lost. Plus, some systems seem to automatically override this change every now and then.

5.2.2 Permanently Set IP Address via Ubuntu's GUI Tools

- Click on the “Ubuntu” menu (like the Start menu) and search for and run Network.
- To know which connection to edit, click on each of the Wired network entries in the list to see their Hardware Address. Match this to the HWaddr shown by ifconfig.
  - If you are unsure which connection in ifconfig is for the connection of interest, try disconnecting the BeagleBone’s Ethernet-over-USB and see what connection disappears. This may help you identify them.
- Select the Wired network listed for the BeagleBone, and click Options.
- Go to the IPv4 Settings tab.
- Change the Method to Manual
- Click Add to add an IP Address and set desired address.
  - For the Ethernet-over-USB, use IP 192.168.7.1, netmask 255.255.255.0, gateway 0.0.0.0
- Click Save.
5.2.3 Permanently Set IP Address via Configuration File

- Make a backup of the internet configuration file:
  ```bash
  $ sudo cp /etc/network/interfaces /etc/network/interfaces.bak
  ```

- Edit configuration file to set a static IP at boot for Debian based (Ubuntu) systems:
  ```bash
  $ sudo gedit /etc/network/interfaces
  ```
  For the Ethernet connection in question (assumed to be `eth5` for this example), add the following to the end of the file (change connection name and IP address as needed):
  ```
  # some comment to describe what you are doing!
  auto eth5
  iface eth5 inet static
     address 192.168.7.1
     netmask 255.255.255.0
     network 192.168.7.0
     broadcast 192.168.7.255
  ```

- Restart networking:
  ```bash
  $ sudo service networking restart
  ```
  - If this does not work, try (replace `eth7` with your connection name):
    ```bash
    $ sudo ifdown eth7
    $ sudo ifup eth7
    ```
  - Or try:
    ```bash
    $ sudo /etc/init.d/networking restart
    ```
  - Or, if that does not work, reboot.