# Working normally with BBB

## Booting normally

Boot BBB with LAN cable inserted (before powering on). Wait 1-2 minutes.

Use “beaglebone.local” as hostname to access BBB.   
(alternative, use “advanced IP scanner” to locate it. 23/5-16 IP is 192.168.**0**.21

## Editing files

We use the Adafruit system. It’s Python based, and stores the files locally on the BBB on /usr/share/adafruit/webide/repositories/my-pi-projects/Vanding

Anyway – you’ll have to log on to the Bitbucket server on <https://bitbucket.org/> to use it.  
*(user/pwd is nnohrras/rasmussen)*

* From browser <http://beaglebone.local:8080/editor>
* Log in to bitbucket
* Go to the folder my-pi-projects/vanding

## Execution & Directories

Source code in /vanding/src/

LOGFILE\_NAME = "/vanding/log/log\_vanding.txt"

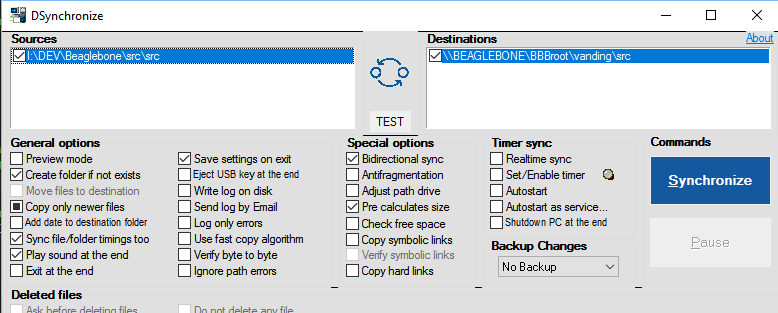
CSV\_FILE\_NAME = "/vanding/log/vanding.csv"

HISTORY\_FILE\_NAME = "/vanding/log/history\_vanding.log"

## The program WinSCP

To use a windows based file explorer on local PC and BBB jointly.

Use this for backup from BBB to PC.



## TTY terminal (PuTTY program on PC)

Select “BBB Lan (root/rasmussen)” profile

## Debugging via Wing IDE

* Start the PC program DSynchronize to enable sync between BBB and PC
* Start PuTTY “debug” session to enable the callback to the Window PC.
* Include “import wingdbstub” in the python file to debug.  
  NOTE: Put the include statement behind a flag, so it won’t be called when script is running non-debug mode.
* Set a breakpoint in the src file (from the debugger)
* From PuTTY terminal:  
   cd /vanding/src  
   /root/wingIDE/wing-debugger-raspbian-5.1.12-1/wingdb <file>

## Running a python pgm from command line

* In one shell do:

cd /usr/share/adafruit/webide/repositories/my-pi-projects/Vanding

python manual\_test.py0

* in another shell do

tail –f ~/log\_vanding.txt

## List of common Linux commands

*pwd* - show current directory

*cd* - change current directory

*ls* - list directory contents

*chmod* - change file permissions

*chown* - change file ownership

*cp* - copy files

*mv* - move files

*rm* - remove files

*mkdir* - make directory

*rmdir* - remove directory

*cat* - dump file contents

*less* - progressively dump file

*vi* - edit file (complex)

*nano* - edit file (simple)

*head* - trim dump to top

*tail* - trim dump to bottom

Ps -ef – show running processes

[systemctl](https://wiki.archlinux.org/index.php/Systemd) – start/stop processes

*echo* - print/dump value

*env* - dump environment variables

*export* - set environment variable

*history* - dump command history

*grep* - search dump for strings

*man* - get help on command

*apropos* - show list of man pages

*find* - search for files

*tar* - create/extract file archives

*gzip* - compress a file

*gunzip* - decompress a file

*du* - show disk usage

*df* - show disk free space

*mount* - mount disks

*tee* - write dump to file in parallel

*hexdump* - readable binary dumps

## Network commands

Ifconfig (view network devices)

Available wireless networks:

iwlist wlan0 scan

/usr/lib/connman/test/test-connman services

dmesg | grep wlan0 (status of the wlan)

iwconfig

/usr/lib/connman/test/get-services

nano /var/lib/connman/wifi.config

### SSH (cannot find beaglebone.local)

* Did this:  
  Also "journalctl | grep avahi" returned a single message stating something like "Daemon already runnin gon PID NNN".  
  So, I "fixed" the problem by adding the line "ExecStartPre=/bin/rm -f /var/run/avahi-daemon/pid" to the [Service] section of /lib/systemd/system/avahi-daemon.service. With this addition, "beaglebone.local" now appears on the network 100% of reboots.

### DHCP for wlan0 & eth0

**Disable connmand ad wpa\_supplicant. Use static address. No encryption.**

**Only change /etc/network/interfaces**

**Start from boot:**

ps -ef|grep conn

ps -ef|grep wpa

Kill processes (connmand, wpa\_supplicant)

ifdown wlan0

sleep 2

ip link set wlan0 up

sleep 2

iw wlan0 connect nohr54linksys

sleep 2

ifup wlan0

sleep 2

ifconfig wlan0

**Check:**

iw dev wlan0 link

ip link show wlan0 🡺 (…. state UP)

# /etc/network/interfaces -- configuration file for ifup(8), ifdown(8)

# The loopback interface

auto lo

iface lo inet loopback

# Wireless interfaces

allow-hotplug wlan0

auto wlan0

iface wlan0 inet dhcp

wireless-ssid nohr54linksys

wireless-mode mixed

network 192.168.1.0

broadcast 192.168.1.255

dns-nameservers 192.168.1.1

#

iface atml0 inet dhcp

# Wired interfaces

auto eth0

iface eth0 inet dhcp

broadcast 192.168.1.255

gateway 192.168.1.1

dns-nameservers 192.168.1.1

iface eth1 inet dhcp

# Ethernet/RNDIS gadget (g\_ether)

# ... or on host side, usbnet and random hwaddr

iface usb0 inet static

address 192.168.7.2

netmask 255.255.255.0

network 192.168.7.0

gateway 192.168.7.1

# Bluetooth networking

iface bnep0 inet dhcp

## Wireless network commands

|  |  |
| --- | --- |
|  |  |
| wpa\_supplicant -B -Dwext -iwlan0 -c/etc/wpa\_supplicant.conf  wpa\_supplicant -d -Dwext -iwlan0 -c/etc/wpa\_supplicant.conf | runs wpa\_supplicant in thebackground |
| wpa\_cli | then call ‘status’ to see how the connection is going. |
| rm /var/run/wpa\_supplicant/wlan0 | If status of wpa\_supplicant –B ... Is  ” Delete '/var/run/wpa\_supplicant/wlan0' manually if it is not used anymore” |
| iw dev wlan0 link  iwconfig wlan0 | Getting link status. |
| iw dev wlan0 scan  iwlist wlan0 scan | Scanning for available access points. |
| iw dev wlan0 set type ibss  iwconfig wlan0 mode ad-hoc | Setting the operation mode to ad-hoc. |
| iw dev wlan0 connect your\_essid  iwconfig wlan0 essid your\_essid | Connecting to open network. |
| iw dev wlan0 connect your\_essid 2432  iwconfig wlan0 essid your\_essid freq 2432M | Connecting to open network specifying channel. |
| iw dev wlan0 connect your\_essid key 0:your\_key  iwconfig wlan0 essid your\_essid key your\_key | Connecting to WEP encrypted network using hexadecimal key. |
| iw dev wlan0 connect your\_essid key 0:your\_key  iwconfig wlan0 essid your\_essid key s:your\_key | Connecting to WEP encrypted network using ASCII key. |
| iw dev wlan0 set power\_save on  iwconfig wlan0 power on | Enabling power save. |
| ip link set dev wlan0 up |  |
| Dmesg | Check logging |
| systemctl restart connman.service |  |

## Config files

### Ntpd.conf

# This is the most basic ntp configuration file

# The driftfile must remain in a place specific to this

# machine - it records the machine specific clock error

driftfile /etc/ntp.drift

logfile /var/log/ntpd.log

# NTP Servers for Denmark from www.pool.ntp.org

server 0.dk.pool.ntp.org

server 1.dk.pool.ntp.org

server 2.dk.pool.ntp.org

server 3.dk.pool.ntp.org

# This obtains a random server which will be close

# (in IP terms) to the machine. Add other servers

# as required, or change this.

# server time.server.example.com

# Using local hardware clock as fallback

# Disable this when using ntpd -q -g -x as ntpdate or it will sync to itself

# Defining a default security setting

restrict 192.168.1.0 mask 255.255.255.0 nomodify notrap

### Wireless config files

|  |  |  |
| --- | --- | --- |
|  | **new** | **Original** |
| /etc/wpa\_supplicant.conf | ctrl\_interface=/var/run/wpa\_supplicant  ctrl\_interface\_group=0  network={  ssid=”nohrTDC"  proto=WPA2  key\_mgmt=WPA-PSK  pairwise=CCMP TKIP  group=CCMP TKIP WEP104 WEP40  psk=”RASMUSSEN"  } | ctrl\_interface=/var/run/wpa\_supplicant  ctrl\_interface\_group=0  update\_config=1  network={  key\_mgmt=NONE  } |
| /etc/network/interfaces | # Controlling ifup and ifdown  # Wireless interfaces  auto wlan0  iface wlan0 inet dhcp  wireless\_mode managed  wireless\_essid any  wpa-driver wext  wpa-conf /etc/wpa\_supplicant.conf | Not existing |
| /var/lib/connman/wifi.config | [service\_home]  Type = wifi  Name = nohr54linksys  #Security = wpa  #Passphrase = RASMUSSEN  (adafruit depends on this) |  |
| /etc/modprobe.d/blacklist.conf |  |  |
| /etc/hosts | Added line 2 (beaglebone) |  |
| /usr/share/dbus-1/system-services/fi.epitest.hostap.WPASupplicant.service  /usr/share/dbus-1/system-services/fi.w1.wpa\_supplicant1.service | Change the "Exec=" line to be "Exec=/usr/sbin/wpa\_supplicant -u -s -O /var/run/wpa\_supplicant" in both files. (initially it is the same up to ’–u’) |  |
|  |  |  |

## OBSOLETE

## Running a python program from AdaFruit

* Select statemachine3.py
* Select Run or Debug
* Output logged in files in ~/
* log\_vanding.txt and history\_vanding.log
* Use tail –f <file> to tail it

# Coding / Configuration

## Disable EMMC pins

To use all GPIO’s we will need to override some defaults.

<http://kilobaser.com/blog/2014-07-15-beaglebone-black-gpios>

For the bare-Bone user, the most interesting ones are the pre-enabled BB-BONELT-HDMI, BB-BONELT-HDMIN and BB-BONE-EMMC-2G. Disabling these in uEnv.txt can free up a lot of pins for GPIO use. Take care however, if you plan on disabling BB-BONE-EMMC-2G, which controls the pins that connect the internal MMC flash (mmc1). While it's a good idea to use an external microSD card (mmc0), you might still want to access the internal mmc1 for storage purposes.

Furthermore, no matter the MUX setting, the pins **P8\_3** to **P8\_6** and **P8\_20** to **P8\_25**will always be physically connected to the internal flash! So if you plan on preserving the data on there, steer clear of those pins.

# Installing Beaglebone

## Debian OS

### Debian 8.4 installation

#### Install Debian on SD card

BBB Memory er kun 2Gb. Dvs at hvis man skal installere på intern memory, skal man finde en file med navn der slutter på ”2Gb”. Man kan imidlertid tilsyneladende IKKE udvide memory via SD-kortet, hvis Debian ligger på intern memory. Derfor vælger jeg at lægge Debian på SD-kortet.

Følg vejledning på <https://billwaa.wordpress.com/2014/08/03/beaglebone-black-debian-setup/>, dvs:

**1.**Download the Latest Debian OS Image (without eMMC Flasher) from the Official Website: <http://beagleboard.org/latest-images>

**2.**Unzip the image using the .xz file using [7-zip](http://www.7-zip.org/), basically a free version of WinRAR. You should get a .img file after unzipping, this is the image file we are going to flash onto the MicroSD card. 

**3.** Download and Install [Win32DiskImager](http://sourceforge.net/projects/win32diskimager/)

**4.**Now, we are going to flash the image to the MicroSD card. Click the folder icon, a File Chooser Dialog Box should popup, **select the Debian OS Image**. I got a flasher version here, but it shouldn’t matter. I tried flashing it, ran out of room, so no need to try. On Device, **choose your Drive Letter for your Micro SD**

**5.**When it’s finish, stick the card into the BBB, power on and we should be set!  
Med andre ord skal man IKKE trykke på nogen knapper under power-up. Den booter automatisk fra SD-kortet.

#### Extend partition on SD card

From <http://thethingsystem.com/dev/Bootstrapping-the-BeagleBone-Black-with-Debian.html>

Nice to know:  
SD card is named /dev/mmcblk0  
partitions on the SD card are named /dev/mmcblk0p***N (N = partition name)***

Process:

1. Format the SD card as FAT32 in windows.

2. modify the partitioning using fdisk to match the partitions in 2.1.3 “Partitioning an SD card”.   
NOTE: It is NOT the same numbers as in the link. They don’t work. Look below instead.  
Command to use in Fdisk: p, d, w, n

beaglebone login: **root**

Last login: Wed Jun 15 19:52:54 UTC 2016 on ttyGS0

Linux beaglebone 4.4.9-ti-r25 #1 SMP Thu May 5 23:08:13 UTC 2016 armv7l

The programs included with the Debian GNU/Linux system are free software;

the exact distribution terms for each program are described in the

individual files in /usr/share/doc/\*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent

permitted by applicable law.

root@beaglebone:~# **fdisk /dev/mmcblk0**

Welcome to fdisk (util-linux 2.25.2).

Changes will remain in memory only, until you decide to write them.

Be careful before using the write command.

Command (m for help): **p**

Disk /dev/mmcblk0: 15 GiB, 16039018496 bytes, 31326208 sectors

Units: sectors of 1 \* 512 = 512 bytes

Sector size (logical/physical): 512 bytes / 512 bytes

I/O size (minimum/optimal): 512 bytes / 512 bytes

Disklabel type: dos

Disk identifier: 0x6c6218b4

Device Boot Start End Sectors Size Id Type

/dev/mmcblk0p1 2048 XXXXXXXX XXXXXXXX 3.3G 83 Linux

Command (m for help):

d (delete)

n (new, same start sector)

Command (m for help): **p**

Disk /dev/mmcblk0: 15 GiB, 16039018496 bytes, 31326208 sectors

Units: sectors of 1 \* 512 = 512 bytes

Sector size (logical/physical): 512 bytes / 512 bytes

I/O size (minimum/optimal): 512 bytes / 512 bytes

Disklabel type: dos

Disk identifier: 0x6c6218b4

Device Boot Start End Sectors Size Id Type

/dev/mmcblk0p1 2048 31326207 31324160 3.3G 83 Linux

* Extend the partition

root@beaglebone:~# **df**

Filesystem 1K-blocks Used Available Use% Mounted on

udev 10240 0 10240 0% /dev

tmpfs 100784 4460 96324 5% /run

/dev/mmcblk0p1 3360336 3134840 35084 99% /

tmpfs 251956 0 251956 0% /dev/shm

tmpfs 5120 4 5116 1% /run/lock

tmpfs 251956 0 251956 0% /sys/fs/cgroup

tmpfs 50392 0 50392 0% /run/user/1000

tmpfs 50392 0 50392 0% /run/user/0

root@beaglebone:~# resize2fs /dev/mmcblk0p1

resize2fs 1.42.12 (29-Aug-2014)

Filesystem at /dev/mmcblk0p1 is mounted on /; on-line resizing required

old\_desc\_blocks = 1, new\_desc\_blocks = 1

The filesystem on /dev/mmcblk0p1 is now 3915520 (4k) blocks long.

root@beaglebone:~# **df**

Filesystem 1K-blocks Used Available Use% Mounted on

udev 10240 0 10240 0% /dev

tmpfs 100784 4460 96324 5% /run

/dev/mmcblk0p1 15353584 3137372 11540888 22% /

tmpfs 251956 4 251952 1% /dev/shm

tmpfs 5120 4 5116 1% /run/lock

tmpfs 251956 0 251956 0% /sys/fs/cgroup

tmpfs 50392 0 50392 0% /run/user/1000

tmpfs 50392 0 50392 0% /run/user/0

root@beaglebone:~#

#### Setting the time and time zone

**apt-get** update

**apt-get** install ntp

dpkg-reconfigure tzdata

<https://wiki.debian.org/TimeZoneChanges>

#### Samba server

<http://raspberrypihq.com/how-to-share-a-folder-with-a-windows-computer-from-a-raspberry-pi/>

apt-get install samba samba-common-bin

nano /etc/samba/smb.conf

Read through the file and make sure you have the following parameters set:

workgroup = WORKGROUP

wins support = yes

You can use anything as your workgroup name as long as it is alphanumerical and matches the workgroup you would like to join. The default workgroup in Windows 7 is WORKGROUP  
Scroll to the bottom and add the following:

[BBBroot]

comment=Beaglebon Black Share

path=/

browseable=Yes

writeable=Yes

only guest=no

create mask=0777

directory mask=0777

# public=no

public=yes

In this case the valid user is the user called “bbb”. To let Samba know that “bbb” is a network user run the command:

smbpasswd -a bbb

And enter bbb’s password twice (default: raspberry).

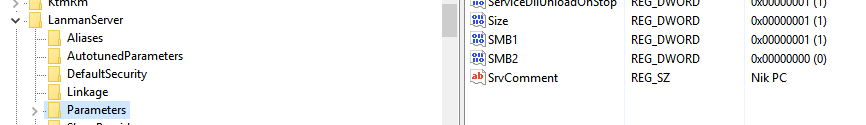
If you do not want to deal with logging in you can always make the share publicly available by changing the config file to say:

service smbd restart

#### Samba Windows 10 client

* First use the regedit

<https://support.microsoft.com/en-us/kb/2696547>



* That didn\t do the job fully so the following command was executed, BUT I am not so sure it was a good idea, since it deletes the service completely. Maybe instead use sc.exe stop lanmanworkstation.   
  But this is what I did:

sc.exe delete lanmanworkstation

* If you manage to delete the service lanmanworkstation and you cannot re/create, despair not, but reinstall the missing registry:



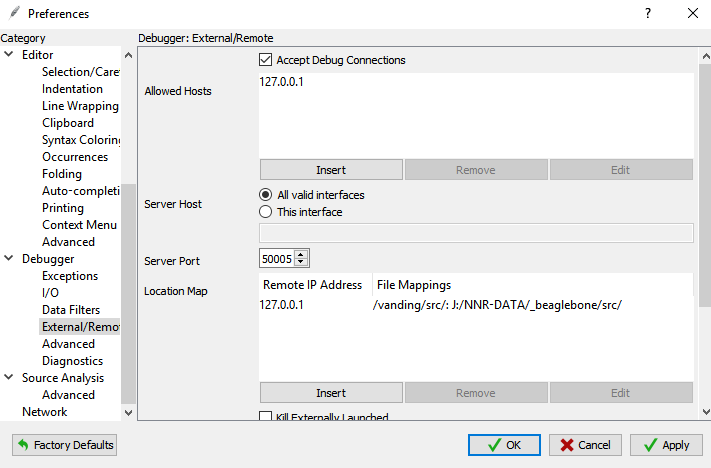
* look at <http://www.techrepublic.com/article/how-to-fix-the-missing-windows-sockets-registry-entries-error/>   
  Command Prompt 1 steps worked for me.

#### Install Wing IDE

Follow <https://wingware.com/doc/howtos/raspberry>

No need to install Python on the PC.

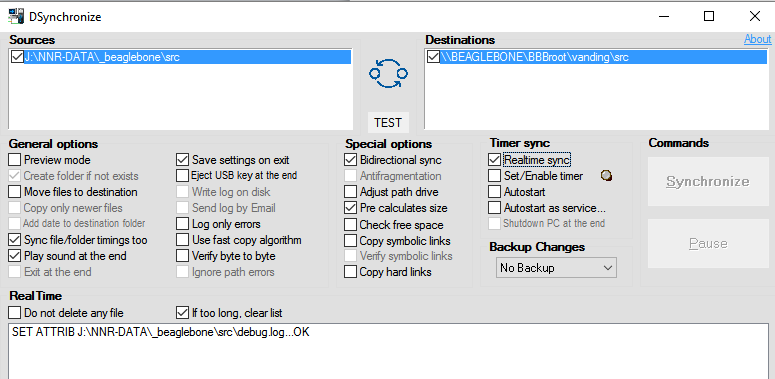
Pay attention to this setup. There’s a difference between “/” and “\”. Also notice the “/” at the end.



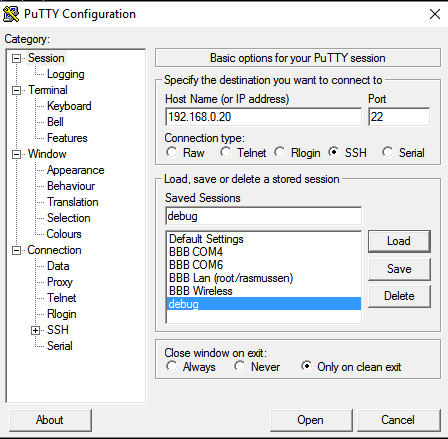
#### Install DSynchronize

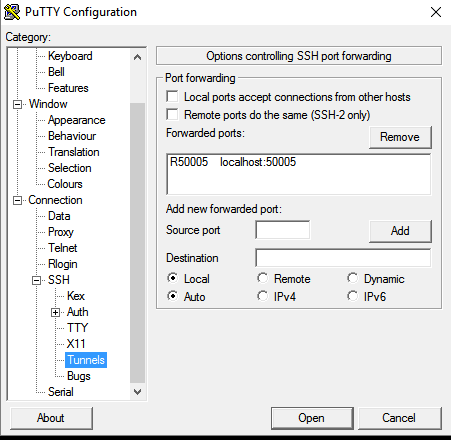
To make up for the fact that Wing IDE shows the source file from the PC during debugging but actually executes the file on the BBB, we need to synchronize the two folders.

Use DSynchronize to do that:



#### PuTTY setup (stored in “debug” profile)





#### Adafruit on Debian

NOTE> I never got this to work probably due to poor configuration.

<https://learn.adafruit.com/setting-up-io-python-library-on-beaglebone-black/installation-on-ubuntu>

sudo apt-get update

sudo apt-get install build-essential python-dev python-setuptools python-pip python-smbus

(don’t install any dtc patch – whatever that is)

sudo pip install Adafruit\_BBIO

Requirement already satisfied (use --upgrade to upgrade): Adafruit\_BBIO in /usr/local/lib/python2.7/dist-packages/Adafruit\_BBIO-0.0.30-py2.7-linux-armv7l.egg

#### Installing more Python libs

For ADC reading, install <https://github.com/alexanderhiam/PyBBIO/wiki/Installing-PyBBIO>

apt-get update && apt-get install python-serial python-setuptools python-dev python-smbus python-pip

pip install --upgrade PyBBIO

If you are using a recent Debian image with a 3.14 kernel you will need to downgrade to the 3.8 kernel to use PyBBIO. First check you're kernel version:

root@beaglebone:/# uname -r

4.4.9-ti-r25

Seems like the version is too new (!!). We try anyway.

#### Installing patches to Adafruit\_BBIO

If you get an error during ADC.setup(): <https://github.com/PeteLawler/adafruit-beaglebone-io-python>

git clone git://github.com/adafruit/adafruit-beaglebone-io-python.git

cd adafruit-beaglebone-io-python

sudo python setup.py install

cd ..

sudo rm -rf adafruit-beaglebone-io-python

Test: refer to <https://gist.github.com/pdp7/cf5551d2e925ac3d8f703d0f0e0f9f43#file-adafruit_bbio-adc-test-md>   
Look at the correct Linux version.

#### Start vanding from boot

Files:

/lib/systemd/system/vanding.service

[Unit]

Description=Start vanding fra boot

[Service]

Type=simple

WorkingDirectory=/vanding/src

ExecStart=/usr/bin/python statemachine3.py

SyslogIdentifier=vanding

[Install]

WantedBy=multi-user.target

Create /lib/systemd/system/vanding.timer to ensure a delayed start of vanding.service.  
The ADC must have time to initialize internally before calling it.

[Unit]

Description=Start vanding after boot

[Timer]

# Time to wait after booting before we run first time

OnBootSec=1min

Unit=vanding.service

[Install]vanding

WantedBy=multi-user.target

Commands to initialize:

systemctl enable vanding.service

systemctl enable vanding.timer

Check and debug:

<http://mattrichardson.com/BeagleBone-System-Services/>

systemctl start vanding

systemctl status vanding

systemctl stop vanding

systemctl disable vanding

#### Install user “devuser” and group “dev\_users” for development

Link: <http://www.metasoarous.com/setting-up-bbb-for-development/>

groupadd dev\_users

mkdir /home/devuser

useradd -G dev\_users,admin -d /home/devuser devuser

chown -R devuser /home/devuser

passwd devuser

Enter new UNIX password: x

Retype new UNIX password: x

wget https://gist.githubusercontent.com/metasoarous/a7308779837f9dcba662/raw/b6717752469da4184c4aaa8bd88407290df15e19/80-pinusers-permissions.rules -O /etc/udev/rules.d/80-pinusers-permissions.rules

wget https://gist.githubusercontent.com/metasoarous/a7308779837f9dcba662/raw/8b7dfc236514caeca0274c6144afe195cd074a0c/set-pinusers-permissions.sh -O /usr/local/bin/set-pinusers-permissions.sh

chmod +x /usr/local/bin/set-pinusers-permissions.sh

nano /usr/local/bin/set-pinusers-permissions.sh (change “pinusers” to “dev\_users”)

/usr/local/bin/set-pinusers-permissions.sh

#### NOT WORKING: WEBide on Debian

**See:** [**https://github.com/adafruit/Adafruit-WebIDE/issues/306**](https://github.com/adafruit/Adafruit-WebIDE/issues/306)

This is tricky but here we go:

* Try to install WEDide (this will fail)

sudo apt-get update && sudo apt-get -y install build-essential nodejs nodejs-legacy npm redis-server git

…

Some packages could not be installed. This may mean that you have

requested an impossible situation or if you are using the unstable

distribution that some required packages have not yet been created

or been moved out of Incoming.

The following information may help to resolve the situation:

The following packages have unmet dependencies:

nodejs : Conflicts: nodejs-legacy

Conflicts: npm

E: Unable to correct problems, you have held broken packages.

* Install correct versions of nodejs (<https://github.com/nodesource/distributions#deb> )  
  (this takes some 5-10 minutes. Use “top” if in doubt. You will see apt-get running)

curl -sL https://deb.nodesource.com/setup\_6.x | bash -

apt-get install -y nodejs

apt-get install -y build-essential

curl -sL https://deb.nodesource.com/test | bash -

(must end with “SUCCESS”)

* Try to install WEDide again

sudo apt-get update && sudo apt-get -y install build-essential nodejs nodejs-legacy npm redis-server git

* And then things went beserk and trial-and-error got into play:  
  <http://askubuntu.com/questions/493460/how-to-install-add-apt-repository-using-the-terminal>   
  <http://stackoverflow.com/questions/21168141/cannot-install-packages-using-node-package-manager-in-ubuntu>

apt-get install build-essential libssl-dev

sudo apt-get purge node-\*

sudo apt-get autoremove

apt-get install python-software-properties

apt-get install software-properties-common

add-apt-repository ppa:chris-lea/node.js

apt-get update

apt-get install nodejs

curl https://raw.githubusercontent.com/adafruit/Adafruit-WebIDE/alpha/scripts/install.sh | sudo sh (didn’t work)

curl -O http://adafruit-download.s3.amazonaws.com/adafruitwebide-0.3.12-Linux.deb

sudo dpkg -i adafruitwebide-0.3.12-Linux.deb

sudo apt-get -f install

apt-get update && sudo apt-get -y install build-essential

apt-get -f install

apt-get install -y node-abbrev node-ansi node-ansi-color-table

apt-get install -y node-archy node-block-stream node-fstream node-fstream-ignore

apt-get install -y node-github-url-from-git node-glob node-graceful-fs node-inherits

apt-get install -y node-ini node-lockfile node-lru-cache node-minimatch node-mkdirp

apt-get install -y node-gyp node-nopt node-npmlog node-once node-osenv node-read

apt-get install -y node-read-package-json node-request node-retry

apt-get install -y node-rimraf node-semver node-sha node-slide node-tar

apt-get install -y node-underscore node-which

apt-get install npm

dpkg --configure –a

apt-get install redis-server git

<https://learn.adafruit.com/webide/installation>

Follow procedure for manual installation:

sudo apt-get update && sudo apt-get -y install build-essential nodejs nodejs-legacy npm redis-server git

…

Some packages could not be installed. This may mean that you have

requested an impossible situation or if you are using the unstable

distribution that some required packages have not yet been created

or been moved out of Incoming.

The following information may help to resolve the situation:

The following packages have unmet dependencies:

nodejs : Conflicts: nodejs-legacy

Conflicts: npm

E: Unable to correct problems, you have held broken packages.

git clone git://github.com/adafruit/Adafruit-WebIDE.git

echo "nameserver 8.8.8.8" >> /etc/resolv.conf

cd Adafruit-WebIDE

mkdir tmp

npm config set tmp tmp

npm install

(there will coma a warning on jade replaced by pug. I did not do anything based on that)

nano config/config.js *(change port 80 to the port of your choice => port 92)*

nodejs server

curl https://raw.githubusercontent.com/adafruit/Adafruit-WebIDE/alpha/scripts/install.sh | sudo sh

To uninstall:

curl https://raw.githubusercontent.com/adafruit/Adafruit-WebIDE/alpha/scripts/uninstall.sh | sudo sh

### Debian 7.5 installation

#### Flashing the eMMC (internal memory)

Husk at Debian distributionen kun må være 2Gb. De fleste er 4Gb, og så kan de ikke være på boardet.

Fil: BBB-eMMC-flasher-debian-7.5-2014-05-14-2gb.img

* Fjern ethernet kabel og USB kabel
* Boot mens knap ved memory card holdes nede til de fire LED’er lyser op
* vent til BBB selv slukker
* fjern SD card
* isæt ethernet kabel og USB kabel
* power on – ingen tryk på knapper
* check file system:

The IP Address for usb0 is: 192.168.7.2

beaglebone login: root

Linux beaglebone 3.8.13-bone50 #1 SMP Tue May 13 13:24:52 UTC 2014 armv7l

The programs included with the Debian GNU/Linux system are free software;

the exact distribution terms for each program are described in the

individual files in /usr/share/doc/\*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent

permitted by applicable law.

root@beaglebone:~# dfshutdown -rFilesystem 1K-blocks Used Available Use% Mounted on

rootfs 1715936 1503088 124016 93% /

udev 10240 0 10240 0% /dev

tmpfs 101700 576 101124 1% /run

/dev/disk/by-uuid/7f2fde16-060e-4c85-be46-ff45658025ea 1715936 1503088 124016 93% /

tmpfs 254248 0 254248 0% /dev/shm

tmpfs 254248 0 254248 0% /sys/fs/cgroup

tmpfs 5120 0 5120 0% /run/lock

tmpfs 102400 0 102400 0% /run/user

/dev/mmcblk0p1 98094 73880 24214 76% /boot/uboot

root@beaglebone:~#

root@beaglebone:~#

root@beaglebone:~# fdisk -l

Disk /dev/mmcblk0: 1920 MB, 1920991232 bytes

4 heads, 16 sectors/track, 58624 cylinders, total 3751936 sectors

Units = sectors of 1 \* 512 = 512 bytes

Sector size (logical/physical): 512 bytes / 512 bytes

I/O size (minimum/optimal): 512 bytes / 512 bytes

Disk identifier: 0x00000000

Device Boot Start End Blocks Id System

/dev/mmcblk0p1 \* 2048 198655 98304 e W95 FAT16 (LBA)

/dev/mmcblk0p2 198656 3751935 1776640 83 Linux

Disk /dev/mmcblk0boot1: 1 MB, 1048576 bytes

4 heads, 16 sectors/track, 32 cylinders, total 2048 sectors

Units = sectors of 1 \* 512 = 512 bytes

Sector size (logical/physical): 512 bytes / 512 bytes

I/O size (minimum/optimal): 512 bytes / 512 bytes

Disk identifier: 0x00000000

Disk /dev/mmcblk0boot1 doesn't contain a valid partition table

Disk /dev/mmcblk0boot0: 1 MB, 1048576 bytes

4 heads, 16 sectors/track, 32 cylinders, total 2048 sectors

Units = sectors of 1 \* 512 = 512 bytes

Sector size (logical/physical): 512 bytes / 512 bytes

I/O size (minimum/optimal): 512 bytes / 512 bytes

Disk identifier: 0x00000000

Disk /dev/mmcblk0boot0 doesn't contain a valid partition table

root@beaglebone:~#

#### Setting the time

**apt-get** update

**apt-get** install ntp

#### Partitioning an SD card

Nice to know:

SD card is named /dev/mmcblk0  
partitions on the SD card are named /dev/mmcblk0p***N (N = partition name)***

root@beaglebone:~# fdisk /dev/mmcblk0

Command (m for help): p

Disk /dev/mmcblk0: 4075 MB, 4075290624 bytes

4 heads, 16 sectors/track, 124368 cylinders, total 7959552 sectors

Units = sectors of 1 \* 512 = 512 bytes

Sector size (logical/physical): 512 bytes / 512 bytes

I/O size (minimum/optimal): 512 bytes / 512 bytes

Disk identifier: 0x0f642ec7

Device Boot Start End Blocks Id System

/dev/mmcblk0p1 2048 16064 7008+ 1 FAT12

/dev/mmcblk0p2 16065 7959551 3971743+ 83 Linux

### Installation of original Ångstrom OS

#### Image (1 hour, 7:30-8:35)

1. Follow <http://beagleboard.org/Getting%20Started#update>
   1. (for same image, start off #8)
2. Pay attention to also following <http://elinux.org/Beagleboard:BeagleBoneBlack_Debian#Flashing_eMMC> (as mentioned in Step 8.
3. Log in via SSH client (e.g. PuTTY) to beaglebone.local. User: root / no pwd
   1. Use COM4 if attaching the BBB to the 3rd external USB on the Dell T1500. Baud rate 115200, rest as default.
   2. Notice that during first login, a new key is generated (pop-up in Putty) => maybe this key is stored on the PC? (this was not a problem on the installation done in june 2016)

Ved 2. Boot (der hvor den flasher fra eMMC, er der løbelysfrem og tilbage fra 16:40)

#### Automatic date setting (15 minutes)

Follow “Install the NTP Software” from <http://derekmolloy.ie/automatically-setting-the-beaglebone-black-time-using-ntp/> , i.e. the commands:

DEBIAN: Use apt-get instead of opkg ???

**apt-get** update

**apt-get** install ntp

nano /etc/ntp.conf

Use these servers in the ntp.conf file:

server 0.dk.pool.ntp.org  
server 1.dk.pool.ntp.org  
server 2.dk.pool.ntp.org  
server 3.dk.pool.ntp.org

The file then looks like:

# This is the most basic ntp configuration file

# The driftfile must remain in a place specific to this

# machine - it records the machine specific clock error

driftfile /etc/ntp.drift

logfile /var/log/ntpd.log

# This obtains a random server which will be close

# (in IP terms) to the machine. Add other servers

# as required, or change this.

# server time.server.example.com

server 0.dk.pool.ntp.org

server 1.dk.pool.ntp.org

server 2.dk.pool.ntp.org

server 3.dk.pool.ntp.org

# Using local hardware clock as fallback

# Disable this when using ntpd -q -g -x as ntpdate or it will sync to itself

# server 127.127.1.0

# fudge 127.127.1.0 stratum 14

# Defining a default security setting

restrict default

And then the commands:

systemctl enable ntpdate.service

systemctl enable ntpd.service

shutdown -r

(replace the line**“ExecStart=/usr/bin/ntpdate-sync silent”**with the line**“ExecStart=/usr/bin/ntpd -q -g -x”**.)

#### Wireless (8:35-9:30 + 12:05-) 🡺 not done in 2016 installation

Device: Sigma WD 150N USB Dongle. Hardware: RT2870

Check <http://www.orbit-lab.org/kernel/compat-wireless-3.0-stable/v3.6/> for latest driver version.

Do as in <https://groups.google.com/forum/#!msg/beagleboard/9KCIs7yqsa8/GyT0FwzmJzkJ> (cut-n-pasted below with a few modification **in bold**)

For what it's worth, here are steps I documented for reliably using an 'ath9k\_htc' usb wifi dongle with Angstrom (systemd-image):

Building and Installing ‘compat-wireless’ Drivers (on the Beaglebone itself):

* 1. Prepare local environment for building drivers. Following instructions here: ‘<http://nomel.org/post/30357133735/beaglebone-tutorial-how-to-compile-kernel-module>’):
     1. **opkg update** (or kernel-dev cannot be found)
     2. opkg install task-native-sdk
     3. opkg install kernel-headers
     4. opkg install kernel-dev
     5. opkg install rfkill
     6. ln -s /usr/src/kernel /lib/modules/3.**8.13**/build  
        (Replace 3.**8.13** with appropriate kernel version… Make sure the headers downloaded to ‘/usr/src/linux-3.2….’ match the actual kernel version ‘uname -r’, otherwise the modules built in this way will not work)
     7. cd /usr/src/kernel
     8. make scripts
  2. Download and build the ‘compat-wireless’ drivers. Following instructions here: (<http://wireless.kernel.org/en/users/Download/stable/#compat-wireless_3.6_stable_releases>):
     1. mkdir ~/compatwireless
     2. cd ~/compatwireless
     3. wget <http://www.orbit-lab.org/kernel/compat-wireless-3-stable/v3.6/compat-wireless-3.6.2-1-snp.tar.bz2>
     4. bunzip2 compat-wireless-3.6.8-1-snp.tar.bz2
     5. copy file to windows machine (use e.g. WinSCP)
     6. unzip compat-wireless-3.6.8-1-snp.tar (use 7-zip)
     7. copy files back to BBB
     8. cd ~/compatwireless/compat-wireless-3.6.8-1-snp
     9. chmod o+x \* (7-zip apparently screws up the permissions)
     10. chmod o+x \*/\*
     11. chmod o+x \*/\*/\*
     12. ./scripts/driver-select ath9k\_htc
     13. Make (ends with 10 lines of errors in some “rf-kill”
     14. make install-modules (Automatically copies the modules to the correct place)
     15. Run ‘depmod -a' to update the ‘known module dependencies’ list with the ones just made/copied. 🡺2 errors: “*WARNING: could not open /lib/modules/3.8.13/modules.order: No such file or directory*”
  3. (skip this section)
  4. **NOT DONE (files already exists)**  
     Download and copy the two firmware files 'htc\_7010.fw' and 'htc\_9271.fw' from <http://wireless.kernel.org/download/htc_fw/1.3/>  to '/lib/firmware':
     1. cd /lib
     2. mkdir firmware (it may exist)
     3. cd firmware
     4. wget <http://wireless.kernel.org/download/htc_fw/1.3/htc_7010.fw>
     5. wget <http://wireless.kernel.org/download/htc_fw/1.3/htc_9271.fw>
  5. Make sure to check the file ‘/var/lib/connman/settings’ and ensure that line after ‘[WiFi]’ reads ‘Enable=true’  
     cat /var/lib/connman/settings
  6. Plug in the usb-wireless dongle. Check ‘dmesg | tail’ to see that it is recognized and that the firmware files are properly transferred. Check ‘lsmod’ to see that all the ‘ath…’ modules have installed correctly.
     1. Remove encryption on nohr54linksys network (browser address: <http://192.168.1.245/>)
  7. Reboot and unplug wired Ethernet cable simultaneously
  8. Test on unencrypted network. (you need wpa-supplicatn to deal with encrypted networks)
     1. ip link set dev wlan0 down
     2. iw dev wlan0 set type managed
     3. ip link set dev wlan0 up
     4. iw dev wlan0 connect nohr54linksys
     5. iw dev wlan0 scan (check that we can read the networks in the air)
  9. Follow Steps here to verify wifi works (in client mode…): (<http://support.criticallink.com/redmine/projects/arm9-platforms/wiki/USB_WiFi_Configuration?version=6>)
     1. opkg install wpa-supplicant
     2. opkg install wireless-tools
     3. opkg install kernel-module-rfkill (annoyingly, needs to update the kernel image – not required for kernel build versions 3.2.30 onwards)
     4. Change /etc/network/interfaces to below. No more, no less:

# Wireless interfaces  
auto wlan0  
iface wlan0 inet static  
 address 192.168.1.30  
 netmask 255.255.255.0  
 gateway 192.168.1.1  
 network 192.168.1.0  
 broadcast 192.168.1.255  
 dns-nameservers 192.168.1.1  
 wireless\_mode managed  
 wireless\_essid any  
 wpa-driver wext  
 wpa-conf /etc/wpa\_supplicant.conf

Note: Alternative config which **might** work:

# Wireless interfaces  
auto wlan0  
iface wlan0 inet dhcp  
 wireless\_mode managed  
 wireless\_essid any  
 wpa-driver wext  
 wpa-conf /etc/wpa\_supplicant.conf

and /etc/wpa\_supplicant.conf (again – exactly as written):

ctrl\_interface=/var/run/wpa\_supplicant  
ctrl\_interface\_group=0  
update\_config=1  
eapol\_version=1  
ap\_scan=2  
fast\_reauth=1  
network={  
 scan\_ssid=0  
 ssid="nohrTDC"  
 psk="RASMUSSEN"  
 key\_mgmt=WPA-PSK  
 pairwise=TKIP  
 group=TKIP  
}

* + 1. Change the following files:
       1. /usr/share/dbus-1/system-services/fi.epitest.hostap.WPASupplicant.service
       2. /usr/share/dbus-1/system-services/fi.w1.wpa\_supplicant1.service
          1. Change the "Exec=" line to be "Exec=/usr/sbin/wpa\_supplicant -u -s -O /var/run/wpa\_supplicant" in both files. (initially it is the same up to ’–u’)
    2. kill the existing processes
    3. ifup wlan0
    4. wpa\_supplicant -B -Dwext -iwlan0 -c/etc/wpa\_supplicant.conf
    5. call ‘wpa\_cli’,
       1. call ‘status’ to see how the connection is going.
       2. Sometimes it can take a long time to connect – perhaps removing connman service helps.
    6. NOT DONE: What definite helps is setting ‘ap\_scan=…’ to ‘ap\_scan=1’ in the ‘wpa\_supplicant.conf’ configuration file.
  1. Reboot + unplug Ethernet + USB
  2. …. And then all the rfkill problems began
  3. opkg upgrade (takes a couple of hours)

#### Configuration files

root@beaglebone:~# cat /etc/wpa\_supplicant.conf

ctrl\_interface=DIR=/run/wpa\_supplicant

#ctrl\_interface=/var/run/wpa\_supplicant

update\_config=1

fast\_reauth=1

ap\_scan=1

network={

ssid="nohrTDC"

psk="RASMUSSEN"

}

root@beaglebone:~# cat **/etc/network/interfaces**

# Wireless interfaces

auto wlan0

#iface wlan0 inet dhcp

# dns-nameservers 192.168.1.1

iface wlan0 inet static

address 192.168.1.30

netmask 255.255.255.0

gateway 192.168.1.1

network 192.168.1.0

broadcast 192.168.1.255

dns-nameservers 193.162.153.164

wireless\_mode managed

wireless\_essid any

wpa-driver wext

wpa-conf /etc/wpa\_supplicant.conf

root@beaglebone:~# cat /etc/resolv.conf

# Generated by Connection Manager

nameserver 127.0.0.1

#### Command sequence

ps -ef|grep wpa

kill <>

wpa\_supplicant -B -i interface -c /etc/wpa\_supplicant.conf

sleep 2

ifdown wlan0

sleep 2

ifup wlan0

Successfully initialized wpa\_supplicant

Sleep 2

ifconfig

wlan0 Link encap:Ethernet HWaddr 00:1F:1F:CD:CE:97

inet addr:192.168.1.30 Bcast:192.168.1.255 Mask:255.255.255.0

UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1

RX packets:28 errors:0 dropped:0 overruns:0 frame:0

TX packets:20 errors:0 dropped:0 overruns:0 carrier:0

collisions:0 txqueuelen:1000

RX bytes:3722 (3.6 KiB) TX bytes:3337 (3.2 KiB)

root@beaglebone:~# iwconfig

wlan0 IEEE 802.11bgn ESSID:"nohrTDC"

Mode:Managed Frequency:2.422 GHz Access Point: 00:1F:C4:8B:84:F0

Bit Rate=11 Mb/s Tx-Power=20 dBm

Retry long limit:7 RTS thr:off Fragment thr:off

Encryption key:off

Power Management:on

Link Quality=70/70 Signal level=-39 dBm

Rx invalid nwid:0 Rx invalid crypt:0 Rx invalid frag:0

Tx excessive retries:0 Invalid misc:13 Missed beacon:0

#### Wlan from boot

**systemd** is the feature used to control boot and background processes in Angstrom.

It’s managed through **systemctl** and extensive logging is through **journalctl**.

Each service is described in a **.service** file, which in turn may call other scripts.

* Change /lib/systemd/system/wpa\_supplicant.service to:

cat /lib/systemd/system/wpa\_supplicant.service

[Unit]

Description=WPA supplicant

[Service]

Type=dbus

BusName=fi.epitest.hostap.WPASupplicant

ExecStart=/usr/sbin/wpa\_supplicant -B -i interface -c /etc/wpa\_supplicant.conf

#Restart=on-failure

Retart=no

#RestartSec=30

[Install]

WantedBy=multi-user.target

Alias=dbus-fi.epitest.hostap.WPASupplicant.service

* Create file for /lib/systemd/system/wlan-restart.service:

[Unit]

Description=Start wlan0 wireless connection at boot. NNR version.

#Before=wlan-restart.service

#Requires=connman.service

[Service]

WorkingDirectory=/home/root/at\_boot/

Type=forking

ExecStart=/bin/bash start\_wireless.sh > /home/root/at\_boot/log

SyslogIdentifier=wlan-restart

Restart=on-failure

RestartSec=60

[Install]

WantedBy=multi-user.target

* Create the file referenced in line “ExecStart” above, i.e. /home/root/at\_boot/start\_wireless.sh

#!/bin/bash

# redirect STDOUT to a file (note the single > -- this will truncate

# http://www.xensoft.com/content/use-exec-direct-all-bash-script-output-file

exec > log.txt

# redirect STDERR to STDOUT and append (note the double >)

exec 1>> log.txt 2>&1

# We can also turn off all output

# set -x ==> turn on cmd output

# set +x ==> turn it off

set -x

echo "REBOOTED:::::::::::::::::::::::::::::::::::::::"

date

sleep 5

echo "Wait for connman to start"

while !(ps aux | pgrep "connman" > /dev/null )

do

echo "Sleeping a bit to allow connman to start"

sleep 5

done

if ps aux | pgrep "connman"

then

if (ps -ef| pgrep "wpa\_")

then

echo "WPA process already running"

else

echo "start wpa\_supplicant"

/bin/bash /home/root/at\_boot/retry.sh

fi

date

echo "no pwer mgt and increased tx signal strength"

iwconfig wlan0 power off

iw reg set BO

iwconfig wlan0 txpower 30

iwconfig wlan0

else

echo "Cannot start WLAN. connman not running (check script /home/root/at\_boot/start\_wireless.sh)"

fi

# this file is reset by connman after each reset. Needed for DNS lookup

echo "nameserver 8.8.8.8" >> /etc/resolv.conf

echo "test connection A"

while !( iwconfig wlan0|grep "nohrTDC" )

do

date

iwconfig wlan0

echo "\*\*\*Not connected. Retrying to connect"

/bin/bash /home/root/at\_boot/retry.sh

sleep 6

done

echo "\*\*\* Connected"

date

/usr/bin/ntpdate -b -s -u pool.ntp.org

ping beaglebone.local -c 3

date

while true

do

date

echo "test connection B"

if !( iwconfig wlan0|grep "nohrTDC" )

then

iwconfig wlan0

echo "RECONNECTING..."

/bin/bash /home/root/at\_boot/retry.sh

echo "after retry"

fi

sleep 30

done

* chmod u+x /home/root/at\_boot/start\_wireless.sh
* Create /home/root/at\_boot/retry.sh

root@beaglebone:~/at\_boot# cat retry.sh

#!/bin/bash

exec >> log.txt

exec 1>> log.txt 2>&1

set -x

#if (ps -ef| pgrep "wpa\_")

#then

# echo "WPA process already running"

#else

wpa\_supplicant -B -i interface -c /etc/wpa\_supplicant.conf

#fi

ifdown wlan0

ifup wlan0

exit 0

* chmod u+x /home/root/at\_boot/retry.sh
* Note that only parts of the output are written to the log file. If not this is indicated, it’s written to the journalctl log instead.

#### Installing more Python libs

For ADC reading, install <https://github.com/alexanderhiam/PyBBIO/wiki/Installing-PyBBIO>

opkg update && opkg install python-distutils python-setuptools python-mmap python-pyserial python-smbus python-pip

pip install PyBBIO

#### Time and date, cont’d

* BBB looses it’s time after a reboot, and does only recover, when connected to network (through wired LAN).

#### Start vanding from boot

Files:

* /lib/systemd/system/vanding.service

[Unit]

Description=Start vanding fra boot

[Service]

Type=simple

WorkingDirectory=/usr/share/adafruit/webide/repositories/my-pi-projects/Vanding

ExecStart=/usr/bin/python statemachine2.py

SyslogIdentifier=vanding

[Install]

WantedBy=multi-user.target

* /lib/systemd/system/vanding.timer

[Unit]

Description=Start vanding after boot

[Timer]

# Time to wait after booting before we run first time

OnBootSec=1min

Unit=vanding.service

[Install]vanding

WantedBy=multi-user.target

Commands:

systemctl enable vanding.service

systemctl enable vanding.timer

## IBM IoT Cloud (BlueMIX)

Section 4 in <https://deskinhursley.wordpress.com/2014/05/20/bluemix-internet-of-things-workshop-with-texas-instruments-beaglebone-and-sensortag/>

What is BlueMIX:

<https://d3c33hcgiwev3.cloudfront.net/_af73b84085fd29941bc8c6124bc24f28_IBM-IoT-BlueMIX-Presentation-.pdf?Expires=1474416000&Signature=LSEy~3NK5bQ9XCIMmdS0la3bBd1000P6FdvnLH8LJAeSsyCUtlRIccFilXZDmDqkERatw258-7hxASi~Bwun6Zbr1yKhdSyG~rT0J89KVVCuTVA2BMON82VZ5uSE-IWHf3psxsFjUVv1DTkiCeB4mIbL2PBuyhM8bPrX-KJfDrk_&Key-Pair-Id=APKAJLTNE6QMUY6HBC5A>

What is NODE-RED:

<https://d3c33hcgiwev3.cloudfront.net/_d542a6c8e80d551491cf0fcbc0474e24_IBM-IoT-Node-RED-Presentation-.pdf?Expires=1474416000&Signature=YFJolupqOShVnkE4kalZRUZOg9qRelTG439oGLaEfBRWCi9qMnR9MM316JY2jpR~OMyBzcaA~8eGpD1dL-8OvbJsjEX3lCqg96Psm-1xtzwmy9WCaqCgtjcEZLnC6D-~SIUFYpmr3NfSpiGuGJoWEEUjUAfMZzs5PtoOKVPt5ZU_&Key-Pair-Id=APKAJLTNE6QMUY6HBC5A>

## Bluetooth Dongle

Section 5 in <https://deskinhursley.wordpress.com/2014/05/20/bluemix-internet-of-things-workshop-with-texas-instruments-beaglebone-and-sensortag/>