Sound and Music with the Raspberry Pi Computer

Andy Farnell

United Kingdom

January 13, 2013

Outline I

Aims

Overview of aims

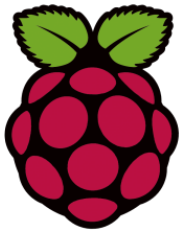
Steps to understand in this workshop

- The Debian OS on the Raspberry Pi (Raspbian)
- Basic Linux networking
- Package management, installing and configuring software
- Basics of embedded development
- Basics of audio on Linux (tools, capabilities)
- Specific use of Pure Data as an audio tool

Use cases

- experimentation system
- stand alone musical instrument synth/sample player
  - studio
  - performance
- embedded sound generator
  - installation
  - toy
- embedded media/communication device
  - skype node
  - mp3/media player

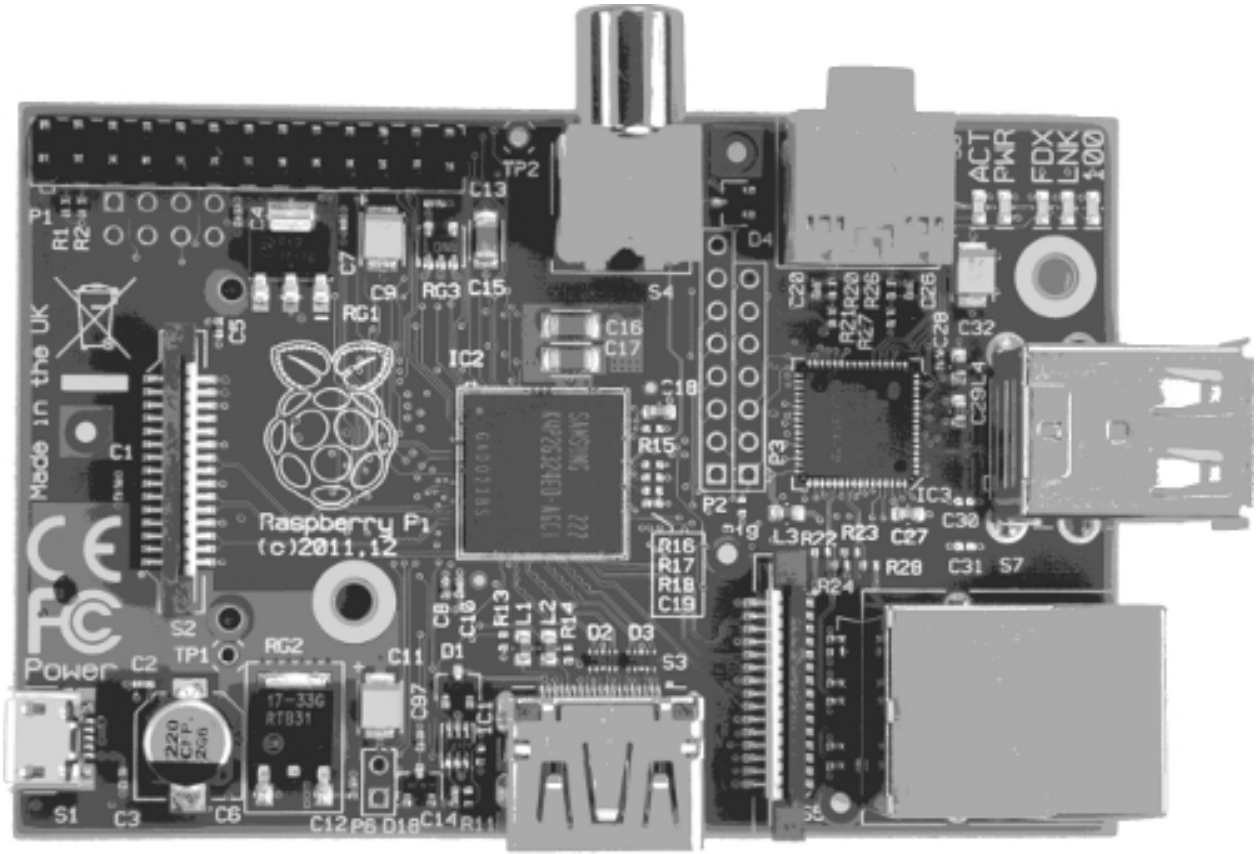
Some Resources



- *http://elinux.org/RPi_Community*
- *http://www.csounds.com/*
- *http://puredata.info/docs/faq/debian*
- *http://puredata.info/*
- *http://www.icecast.org/*
- *http://www.mplayerhq.hu/*

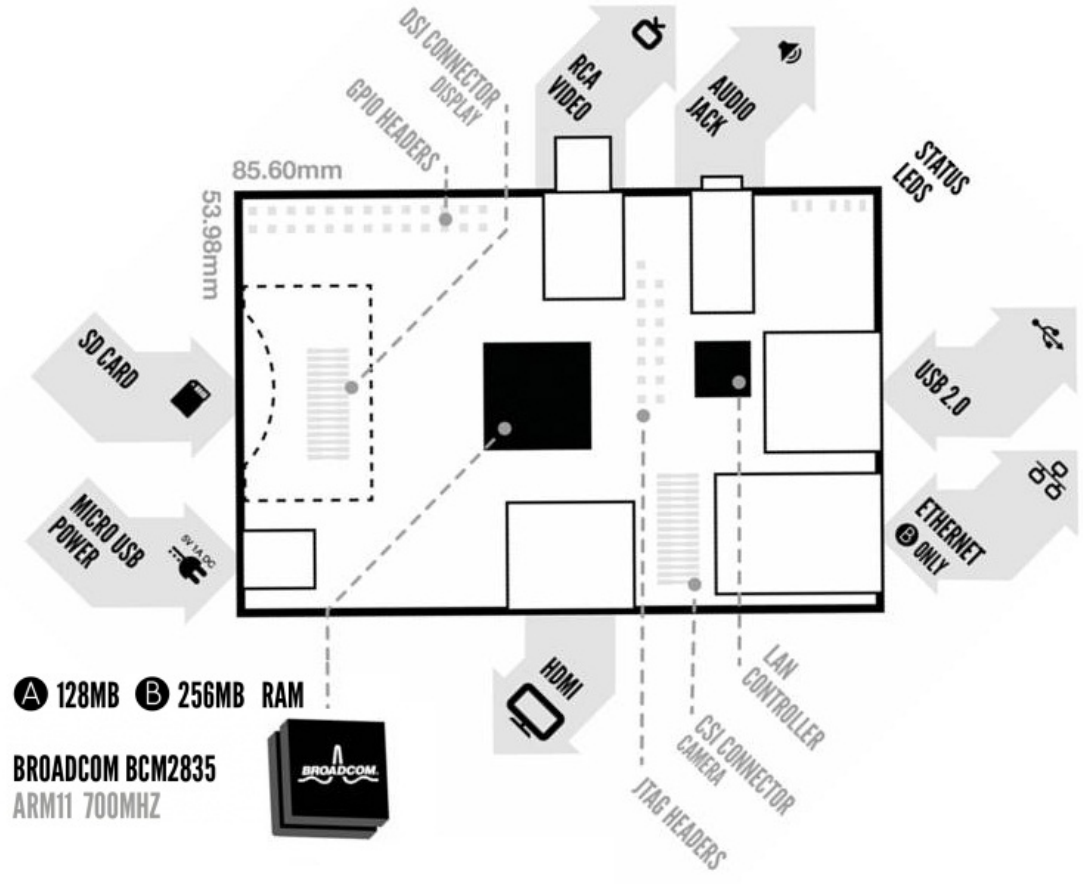
System

Pi Board Hardware



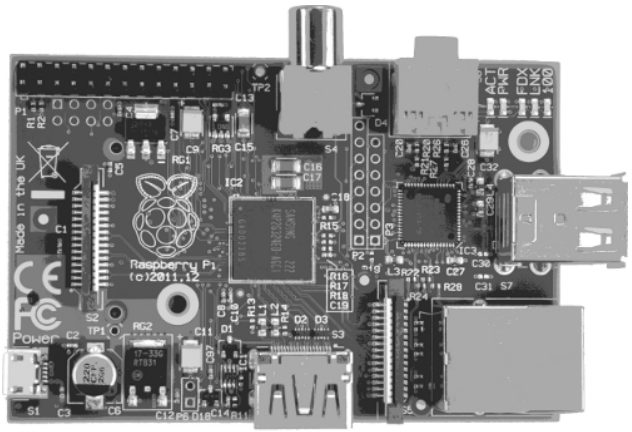
- Features
- Operational Parameters
- Limitations

Pi Board IO



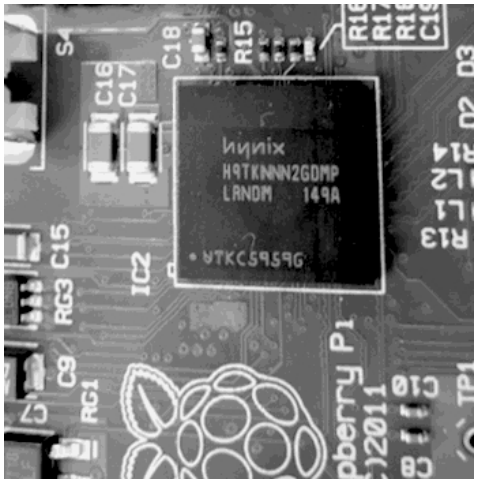
- Power
- Network
- USB Audio

RPi Physical Features



- 86mm x 54mm, 4 layer baord
- 45g weight
- Unsprung SD card slot on rear
- Protusion beyond card footprint
- MicroUSB fragile point
- Plug and unplug at other USB end

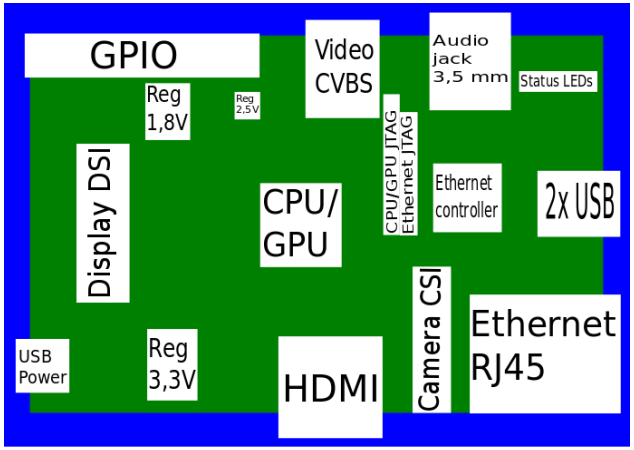
RPi Processor



- Broadcom SoC 2835
- UART, DMA, PCMi2S AUDIO, PWM, Timers, USB
- Open datasheet *resources/pdfs/RPi/BCM2835.pdf*

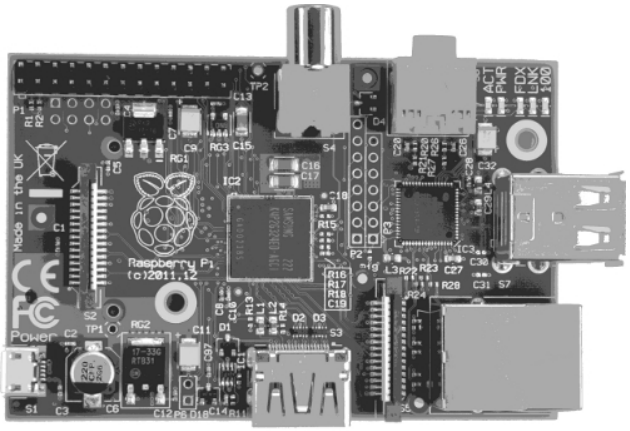
- ARM-11 (series 6) CPU
- Typical smartphone CPU
- Reduced heat
- 64 bit data path
- SIMD and longer (8) pipeline

RPi System Components



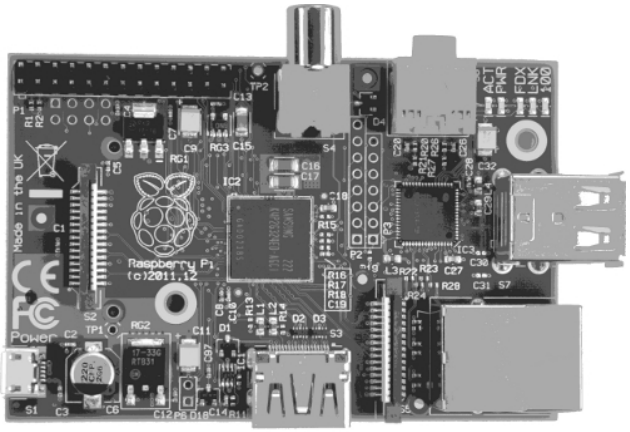
- Audio via 3.5mm PCM
- Audio from HDMI
- HDMI and RCA composite video
- DSI bus for LCD
- SD / MMC / SDIO card slot
- 10/100 RJ45 Ethernet
- 8 GPIO
- UART
- I^2C bus
- SPI bus

RPi Operational Parameters



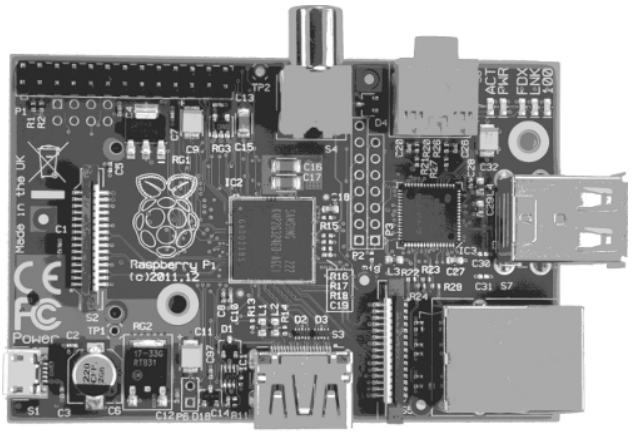
- 5V (via micro-USB or GPIO)
- Current limited USB to 140mA
- 700mA (3.5W)
- 700MHz (overclocking by 200MHz maybe)
- Dynamic overclock (rarely useful for audio processing)
- Low line level audio (low for headphones)
- Schematics *resources/pdfs/RPi/Schematic.pdf*

RPi Practical Performance



- 2 x iPhone4S (?)
- Cannot run heavy webpage (Flash and HTML5)
- YouTube 1 minute to load
- Video acceleration useful
- As low as 5ms latency? Just about OK for guitar FX
- Think about computers in different way (single function)

RPi Limitations

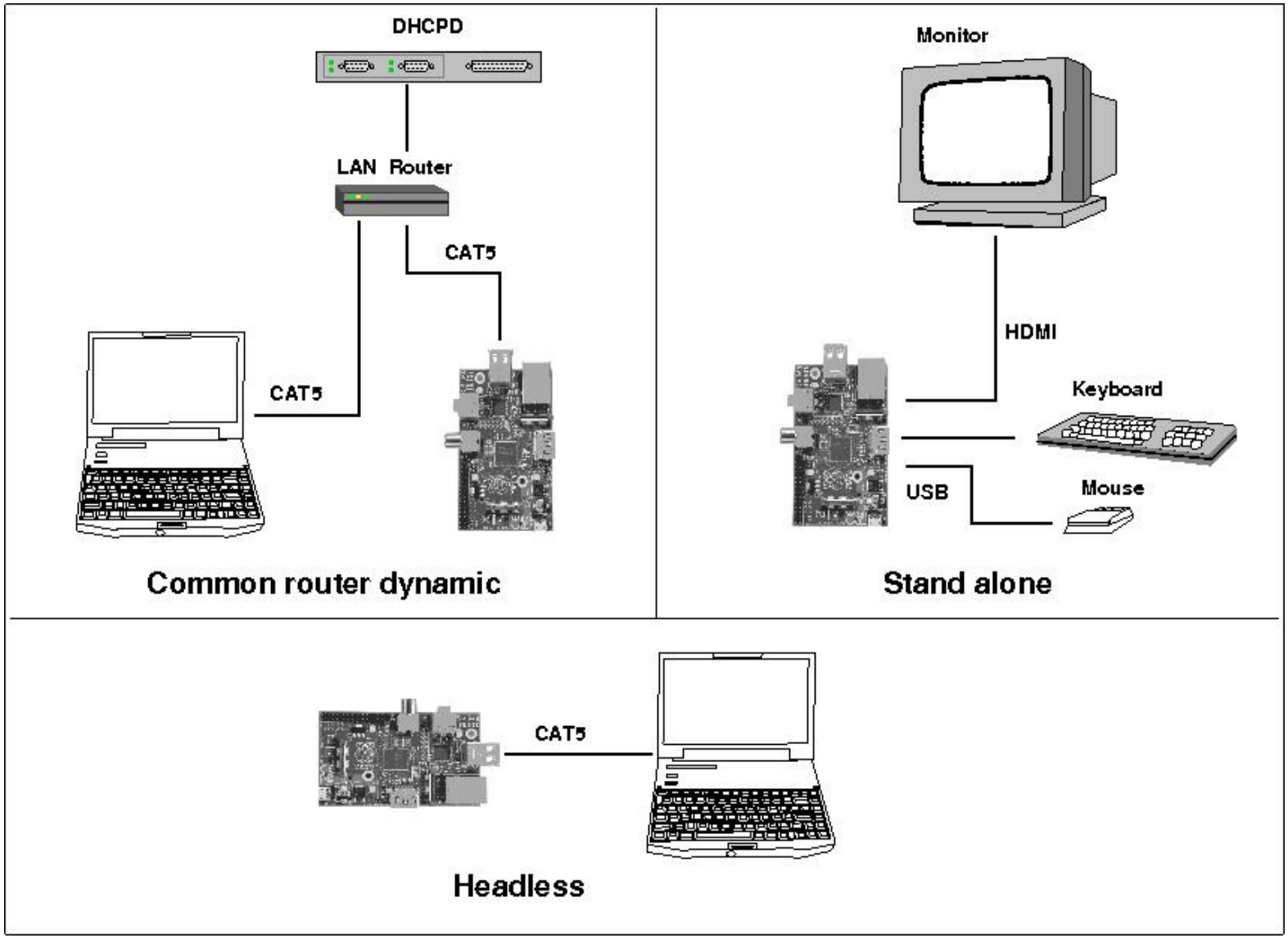


- Power supply is fussy
- Hungry for current (for battery embedded)
- No PoE yet
- Audio drivers for PCM 3.5mm not good
- 512MB is quite small for some applications
- Availability and openness issues

Development Task and Choices

Control Interface	**MIDI** **OSC** **UDP/TCP–IP** **GPIO/USB**
User Application Code	**Synthesiser** **Guitar FX** **Communication** **Music Player**
Application API	**Pure Data** **Csound** **SuperCollider** **C/Python/Perl**
Operating System	**Debian Wheezy** **Bare Metal** **(Windows?)**
Boot medium and Bootloader	**IsoLinux SD** **(GRUB)**
Hardware Host	**Raspberry Pi** **BeagleBoard** **SoC Module**

RPi development configurations



Booting to OS

Development Environment

We will use a *live* Linux distro to work with.

- Simple, easy, familiar desktop
- Same command line (Bash) as Raspberry Pi
- Can configure network in clever ways
- Learn more Linux as part of workshop
- Ask for USB stick if you dont have one

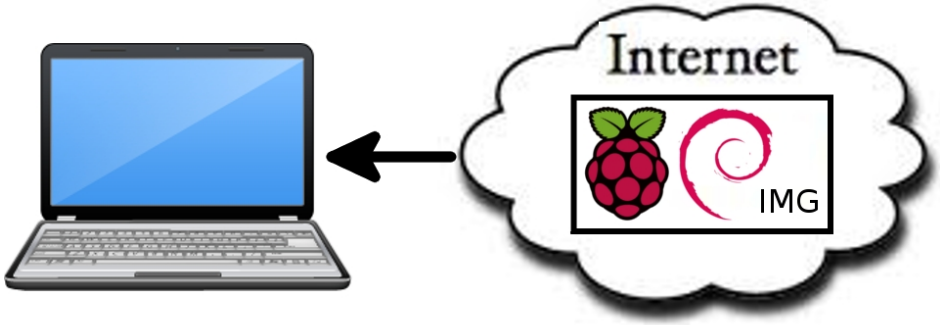
Command line wireless

On MacOSx you may have *less* trouble creating multiple network connections and sharing them, but on Linux (Ubuntu etc) most of the wireless and network managers are *badly broken and immature*. It is much more reliable (though more complex) to disable them and configure wireless by hand.

Summary of procedure

```
ifconfig -a
sudo apt-get install wpasupplicant
wpa_passphrase myrouteressid mypassphrase > connection.conf
iwlist scan
wpa_supplicant -Dwext -iwlan0 -c/connection.conf (background)
iwconfig wlan0
dhclient wlan0
```

Downloading the minimal Rasbian image



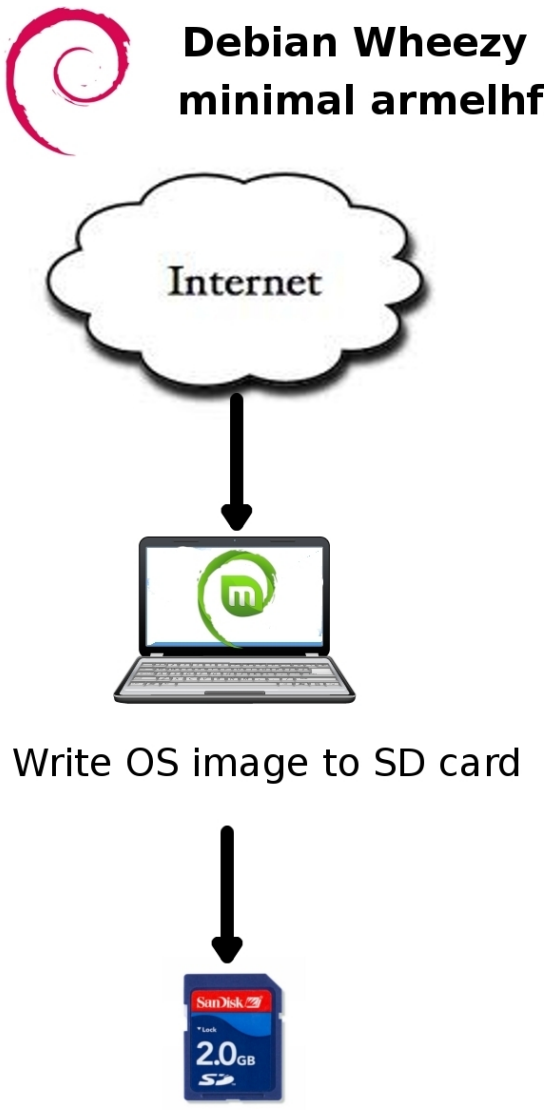
Download, examine and uncompress RPi OS from the *internet* to the *host* machine first. In a terminal window

```
wget http://files2.linuxsystems.it/raspbian_wheezy_20120608.img.7z
ls -alh
```

To uncompress it we need to first get the 7z software

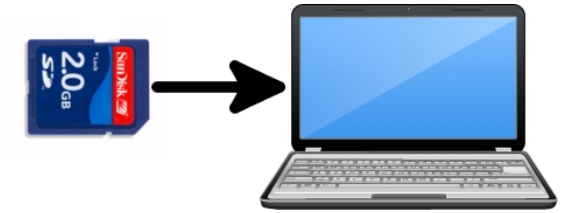
```
sudo apt-get install p7zip-full
man 7za
We will use these unarchiver settings shortly ...
```

Making minimal Debian image



- Minimal Wheezy $< 100MB$
- Hardware float
- Download and uncompress
- Use **dd** from console

Insert card and determine device ID



Determine device using dmesg

MAKE SURE WE KNOW WHICH DEVICE IS CARD

It is *very important* to know the device that matches the card (otherwise we risk destroying laptop hard disk). Run the command

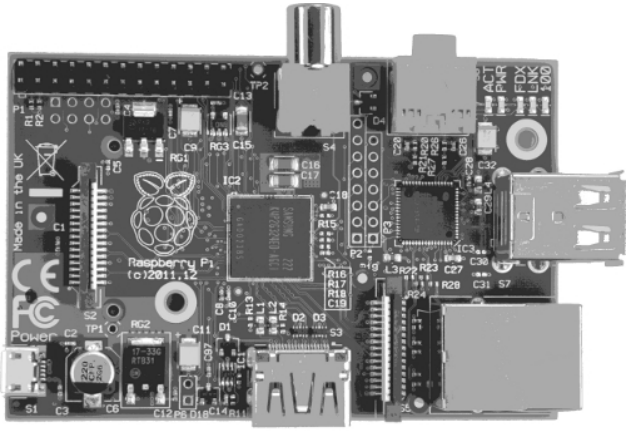
```
dmesg
mount
```

before and **after** inserting the card. Compare the change to identify the newly inserted device. For example:

```
dmesg
[324.000] sd 3.0 [sdd] Attached SCSI removable disk
mount
/dev/sdd on /media/0F7AE3
```

The device new will be `/dev/sdd`

Uncompress OS image to SD card

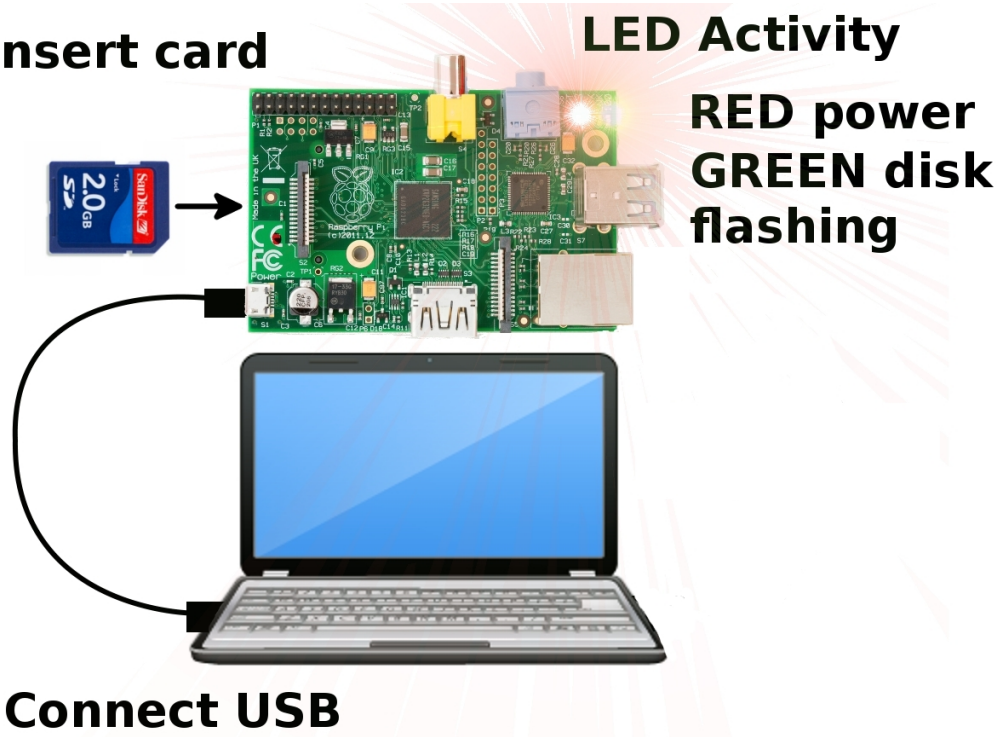


- Usually we uncompress to a file, then copy the file to medium.
- But our *live* host has limited memory, and the RPi OS image expands to **2GB**
- Uncompressing and copying can be done in *one step* using a pipeline

```
mv raspbian_wheezy_20120608.img.7z image
7za e -so image | dd of=/dev/sdd bs=1M
```

Remove card from host when done

Booting Raspberry Pi

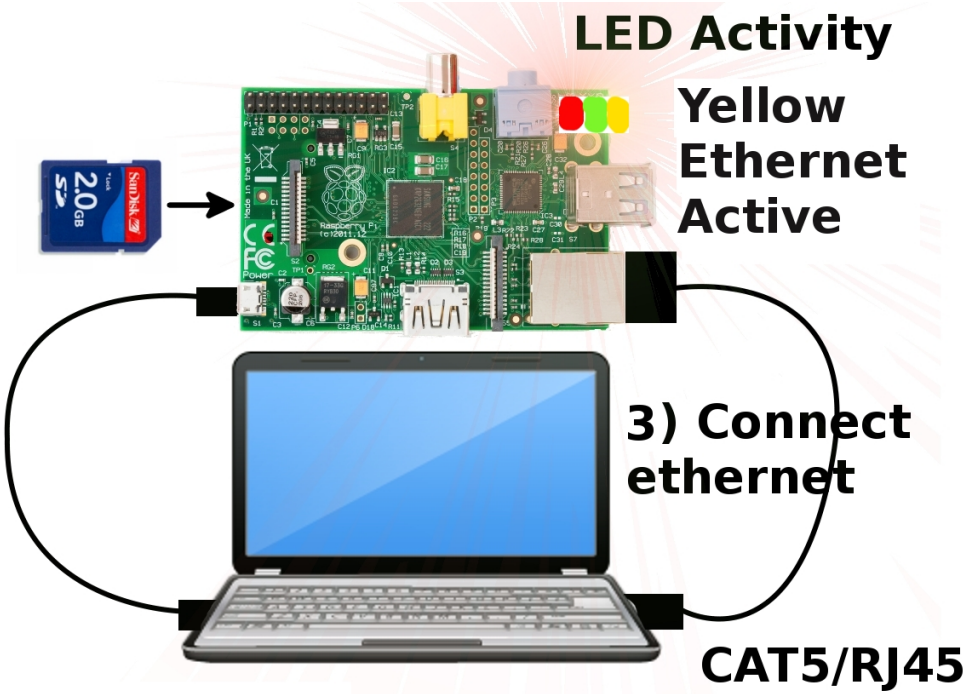


Time to power up the board

- Insert card and make connections
- Board will be powered via USB
- Boot takes about 20 seconds
- Done when green LED stops flashing

Connecting

Communication bewteen Raspberry Pi and host

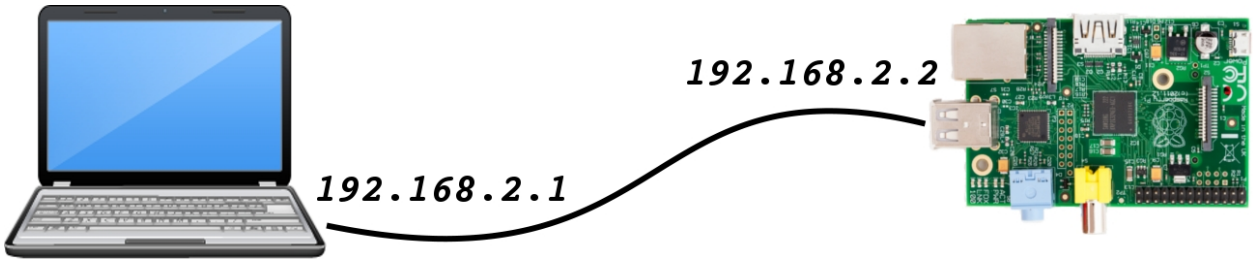


Get the RPi and host machine talking

- Connect ethernet with a cable
- Yellow LED indicates ethernet

Assign IP addresses to ethernet ports

USB power connection not shown



Ethernet ports need to have IP

- Modify `/etc/network/interface` on host to assign static IP
- Stop network manager (to stop DHCP on `eth0`)
- Bring up `eth0` and check it using `ifconfig eth0`

Modify host network interface

We have two choices: **first**
Switch to manual. Disable the `network-manager` to stop it messing with `eth0` and then modify the interface directly using `ifconfig`

```
sudo /etc/init.d/network-manager stop
ifconfig eth0 192.168.2.1 up
```

Or, **second**
Under Ubuntu, use the graphical network manager;
PREFERENCES→NETWORK CONNECTIONS

- ip address 192.168.2.1
- netmask 255.255.255.0

Check that eth0 is up and ready

```
sudo ifconfig
```

```
eth0      Link encap:Ethernet  HWaddr 20:e5:ee:47:34:7c
          inet addr:192.168.2.1 Bcast:192.168.2.255 Mask:255.255.255.0
          inet6 addr: fe80::53e5:49ee:fe42:413b/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:261403 errors:0 dropped:0 overruns:0 frame:0
          TX packets:247356 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:297062088 (283.3 MiB)  TX bytes:29499639 (28.1 MiB)
          Interrupt:40 Base address:0x6000

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:16436  Metric:1
          RX packets:868 errors:0 dropped:0 overruns:0 frame:0
          TX packets:868 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:95880 (93.6 KiB)  TX bytes:95880 (93.6 KiB)
```

Host pingsweep

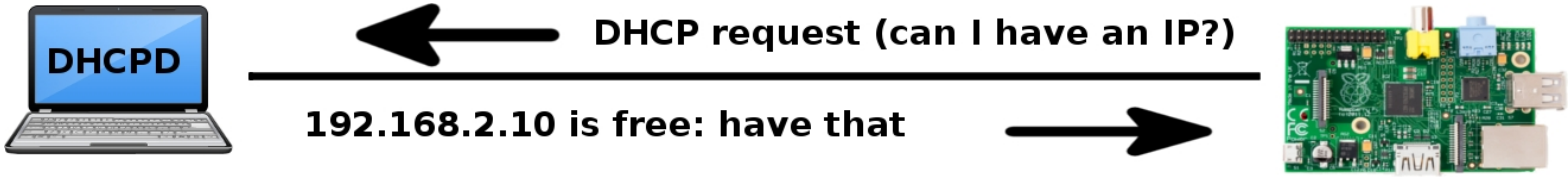
That's one side of the network ready. But does the RPi have an IP address? To detect other interfeces on the network we can use a ping sweep ...

- First need to install `fping`
- Ping entire subnet 192.158.2.1 ... 192.168.2.255 and see if other interfaces are active

```
apt-get install fping
fping -g 192.168.2.1/24 | grep alive
```

No other interfaces are alive yet, but notice the Ethernet activity light flashes on the RPi, so ping packets are travelling on the wire. How can we find the IP address if it is active or bring up the ethernet port on the RPi?

Installing DHCPD software on the host



The RPi actually *asks* for an IP using DHCP. But no response is currently given by the host. We need the *host* to be able to provide DHCP responses to the RPi. We will install and set up a very lightweight DHCP server

Search for DHCP server software

```
sudo apt-cache search dhcp server
    udhcpd - Provides busybox DHCP server implementation
sudo apt-get install udhcpd
```

Notice the instructions to activate. We will return to this later.

Configure and startup DHCPD on the host

Edit the file `/etc/udhcpd.conf`

```
sudo nano /etc/udhcpd.conf
```

```
start 192.168.2.2
end 192.168.2.3
interface eth0
```

We need to create a file

```
sudo touch /var/lib/misc/udhcpd.leases
```

Now let's start it up

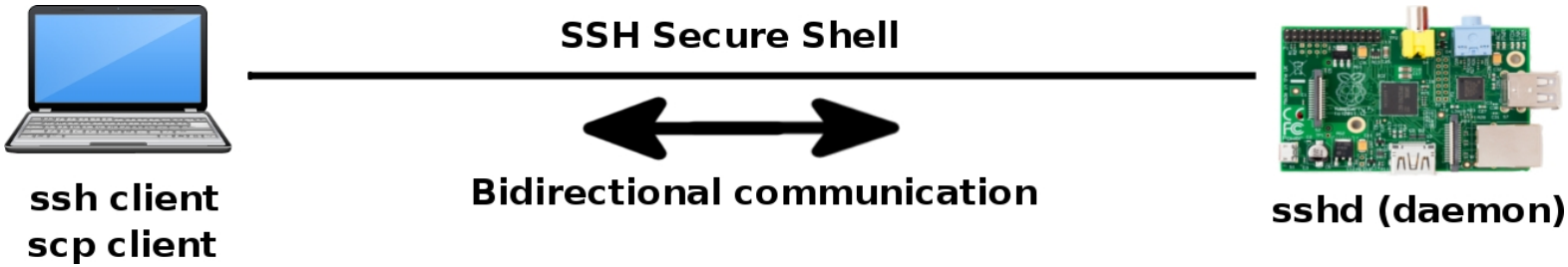
```
sudo busybox udhcpd
```

Plug out the RPi power and resume it. Wait a few seconds . . .

```
fping -g 192.168.2.1/24 2>&1 | grep alive
          192.168.2.1 is alive
          192.168.2.2 is alive
```

Logging in

Logging in using ssh



The RPi Debian OS image comes with `sshd` already set to activate on bootup. It will listen for incoming connections on port 22 of the allocated IP address.

- Can log in as **root** with password **raspberry**
- First time we need to confirm the ssh key
- Later we can use `scp` to copy files
- First let's explore the system

```
ssh -l root 192.168.2.2
Are you sure you want to continue connecting (yes/no)? yes
password:
Debian GNU/LINUX comes with ABSOLUTELY NO WARRANTY
root@raspberry-pi:~#
```

Checking out the RPi system

We will denote acivity on the client using a different background colour; Do some basic checks

Get some system information

```
uname -a
    Linux raspberry-pi 3.1.9 armv6l
```

Check on CPU

```
cat /proc /cpuinfo
    Processor:  ARMv6-compatible processor rev 7
```

Look at network interfaces

```
ifconfig
    eth0:  Link encap: Ethernet HWaddr ab:cd:ef:01:23:45
           inet addr: 192.168.2.2
```

Testing the RPi network

Have we got network connectivity back out to the internet?

Test internet connectivity

```
ping example.com
    ...
```

Hmmmm :(

Why are we not able to connect out from the RPi?

Can we even ping the host?

```
ping 192.168.2.1
    ...
```

No! What is happening?

Adding network routes

One thing we need to add is a default route for the RPi network interface, so it knows where to find other computers. Once it knows about the host machine it can find others.

Adding default route

```
route add default gw 192.168.2.1
```

- How do we know traffic is going to host?
- Can use ping ot host
- Can use `tecpdump` to view DNS request

Host masquerade

Now the RPi can see the host, but it cannot see the internet yet. We need to enable internet sharing so that wifi (`wlan0`) connection is seen on `eth0`. This is called network masquerading.

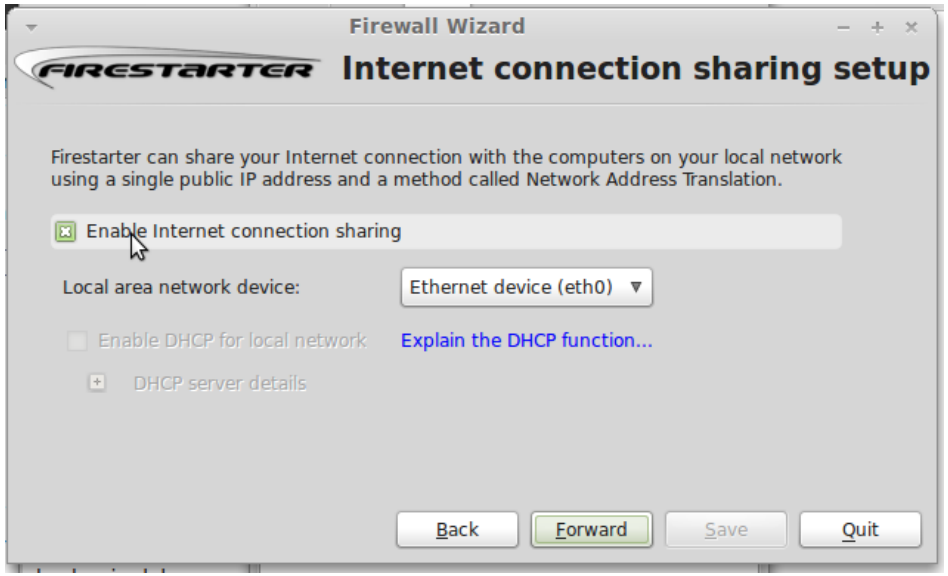
The hard way: Using `iptables` the traditional way; On the *host* machine;

```
echo 1 > /proc/sys/net/ipv4/ip_forward
iptables -t nat -A POSTROUTING -o wlan0 -j MASQUERADE
service iptables save
```

Host masquerade (The *"easy"* way)

There is a GUI tool in the Ubuntu repository for firewalling that has a simple button to enable IP forwarding to another physical port. It is called `firestarter`.

apt-get install firestarter



Testing network masquerade

Let's see if this worked. From the terminal logged in on the RPi;

Test packets are routed

```
ping 192.168.2.1
     64 bytes from 192.168.2.1: icmp_req=1...
```

Hoorah!

Test DNS is resolved

```
ping example.com
     64 bytes from xx-any.icann.org: icmp_req=1...
```

If not try

```
nano /etc/resolv.conf
```

```
nameserver 8.8.8.8
```

Software Management

Installing applications on the RPi

At this point we can begin customising and developing on the RPi from within. Remember that it is a *live* system with *persistence*, so new packages and configurations will survive a reboot.

Update the apt cache

```
apt-get update
    Get:1 http://archive.raspbian.org wheezy/main...
```

When this is done let's try a neat trick

Install an X application

```
apt-get install X-11 apps
    Reading package list... Done ...
```

When this has completed use `logout` or CTRL+D to log out.

Forwarded X application

Log back in again with some extra arguments to `ssh`.

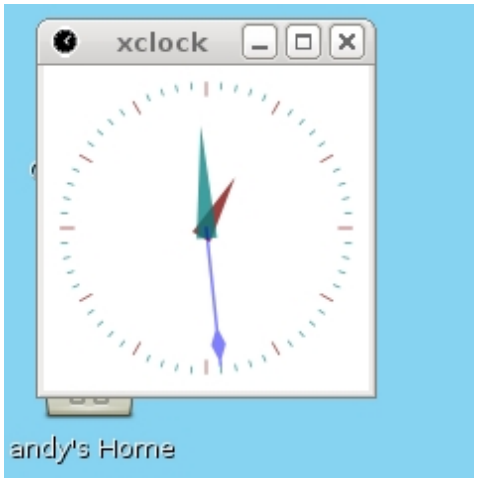
Log in with X enabled and fast cipher

```
ssh -l root -c blowfish -Y 192.168.2.2
```

An X application *does not need* X to be installed on the RPi to run from a remote host. So we can actually develop *graphically* on the RPi without needing a HDMI monitor connected. Obviously there are some limitations to this.

Test a forwarded X application

```
xclock
```



Transferring files to the board

How do we get files to abd from the RPi board? We could set up an FTP service or HTTP file server and use a browser. In Gnome file manager "Nautilus" you can Open Server and then drag and drop over an SSH link. A reliable way from the command line is to use `scp` (secure shell copy), since we already have an `ssh` connection.

The solid old fashioned way :)

```
scp file.mp3 pi@192.168.2.2:/home/pi
```

Part II - Audio Applications

Task 1 - File replay

Play back simple audio files

Find a wave file and copy it to the home directory on the board. On the host;

```
scp music1.wav   pi@192.168.2.2:~
scp music2.mp3   pi@192.168.2.2:~
```

On the RPi;

```
aplay music1.wav
```

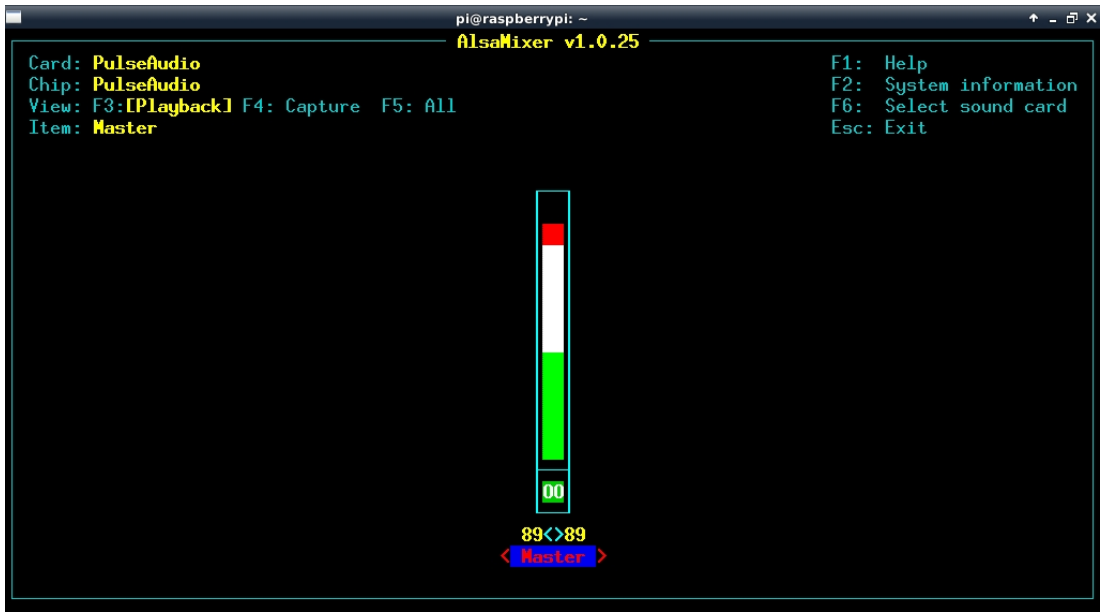
But `aplay` doesn't work for **mp3** files.

```
aplay music2.mp3
```

Setting the mixer levels

You may want to set the audio output level This is done from the `alsamixer` command. In a client terminal;

```
alsamixer
```



Set level using the up and down arrows. If available, other devices or channels can be selected using TAB and left right arrows. To exit, hit ESCAPE

Play back other sources

We need another tool that can decode mp3 files. If the search doesn't return a candidate then you may need to add *non-free* repositories to `/etc/apt/sources.list`

```
apt-cache serach mpg123
apt-get install mpg123
mpg123 music2.mp3
```

You can also decode internet radio with `mpg123`

```
mpg123 http://streaming203.radionomy.com:80/ExtraDance
```

Startup sound

```
nano /etc/rc.local
```

```
nice --10 aplay /home/pi/sounds/startup.wav
```

```
reboot
```

Periodic alarm sound

```
nano alarm.cron
```

```
*/5 * * * * mpg123 /home/pi/sounds/clock.wav
```

```
crontab alarm.cron
```

Trigger sounds over network

```
nano listener.sh
```

```
#!/bin/bash
# Ugly and insecure bash scipt
COMMAND1="SOUND1"
COMMAND2="SOUND2"
while :
do
GOT=$(nc -l 2345)
   if [ "$GOT" == "$COMMAND1" ]
      then
         echo "playing $COMMAND"
         aplay /home/pi/sounds/sound1.wav
   fi

   if [ "$GOT" == "$COMMAND2" ]
      then
         echo "playing $COMMAND"
         aplay /home/pi/sounds/sound2.wav
   fi
done
```

Beware: For demonstration only. This script is easily exploited to execute any arbitary code on the RPi remotely (That might also be useful in development!)

```
 chmod  +x listener.sh
./listener.sh
```

```
 echo "SOUND1" | nc -w1 192.168.2.2 2345
```

A streaming server

Icecast2 internet radio server

Install `icecast2`

```
apt-cache serach icecast2
     icecast2 - streaming media player
apt-get install icecast2
```

Either use autoconfig script if it launches or alter;

Configure `icecast2`

```
nano /etc/icecast2/icecast.xml
```

```
<admin-user>admin</admin-user >
<admin-password >youllneverguess</admin-password>
```

```
nano /etc/default/icecast2
```

```
ENABLE=true
```

Icecast2 administration

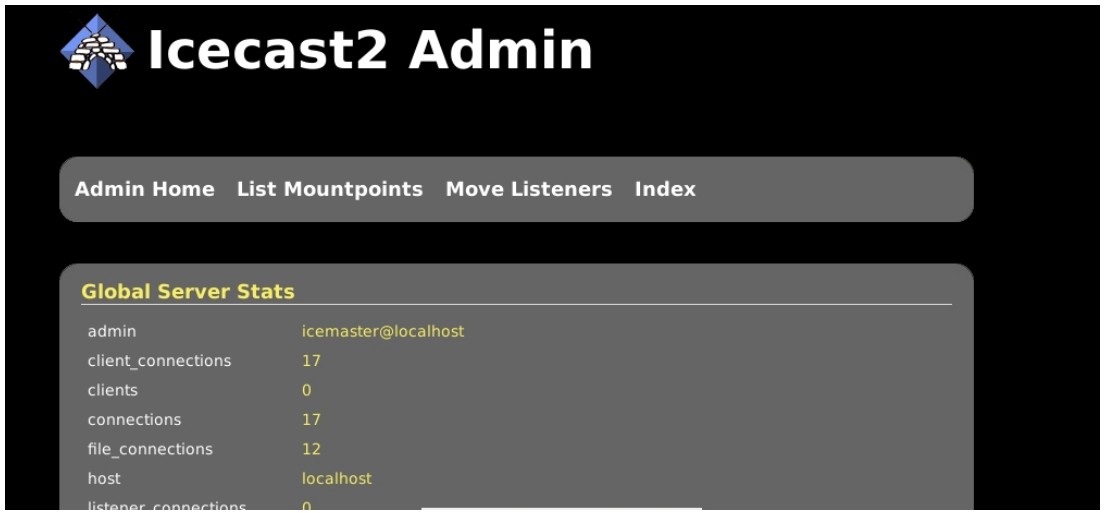
Start and stop the server

```
/etc/init.d/icecast2 start
/etc/init.d/icecast2 stop
apt-get install icecast2
```

Icecast can be remote administrated through a webmin interface;

Use browser from host or other machine

```
http://192.168.2.2:8000/admin/
```



Ices2 Ogg streamer

Another program feeds the `icecast` server with encoded audio. This can be from the audio input, or from files. We will use pre-encoded `.ogg` files

Install `ices2`

```
apt-cache serach ices2
     icest2 - Ogg Vorvis streaming source for Icecast2
apt-get install ices2 mkdir /var/log/ices
```

Make a file called `playlist1.txt` in the pi users home directory.

```
nano /home/pi/playlist.txt
```

```
/home/pi/sounds/song2.ogg
```

Configure `icecast2`

```
nano /usr/share/doc/ices2/examples/ices-playlist.xml
```

```
<password >youllneverguess</password>
<mount >/myradio.ogg</mount>
```

Connecting the radio stream

Populate the /sounds directory

```
scp song2.ogg pi@192.168.2.2: /sounds/
```

Start icecast2 and ices2

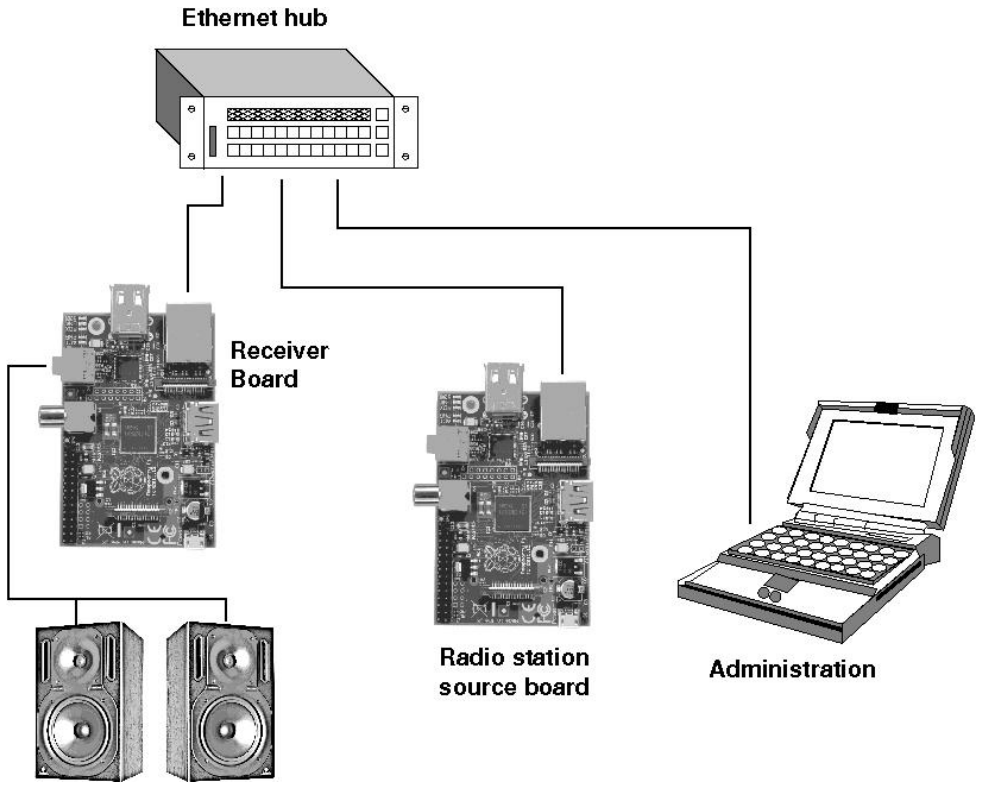
```
sudo /etc/init.d/icecast2 restart
sudo /usr/share/doc/ices2/examples/ices-playlist.xml
```

Now go to the host and open a terminal

```
apt-get install vorbis-tools
ogg123 http://192.168.2.2:8000/myradio.ogg
```

Exercise: stream pair

As a group or pair, use two RPi boards to produce an internet radio streaming source, and a reciever. Experiment logging in the webmin interface and managing the streams. Research streaming from `stdin` using a USB audio dongle and connnect a personal audio player or microphone to DJ.



Csound

Installing Csound

Csound is an extremely high quality and well established sound synthesis language.

```
apt-cache search csound
apt-get install csound
csound
```

Csound response

```
virtual_keyboard real time MIDI plugin for Csound
PortMIDI real time MIDI plugin for Csound
PortAudio real-time audio module for Csound
0dBFS level = 32768.0
Csound version 5.12 (double samples) Aug  5 2010
libsndfile-1.0.21
Usage: csound [-flags] orchfile scorefile
Legal flags are:
--help print long usage options
-U unam run utility program unam
-C use Cscore processing of scorefile
-I I-time only orch run


...
```

Installing Csound

A Csound file is just a text file, can be edited with any tool.

```
nano sine.csd
```

A Csound example

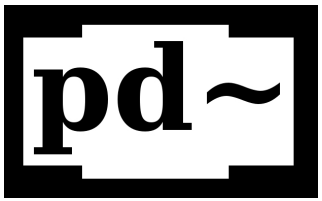
```
<CsoundSynthesizer>
<CsOptions>
-odac
</CsOptions>
<CsInstruments>
;Example by Alex Hofmann
instr 1
aSin      oscils    0dbfs/4, 440, 0
          out       aSin
endin
</CsInstruments>
<CsScore>
i 1 0 1
</CsScore>
</CsoundSynthesizer>
```

And run it thusly;

```
csound -b 2048 sine.csd
```

Pure Data

Pure Data



The most powerful and flexible music synthesis and production system. Like `MaxMSP`, but better :) (*FREE as in beer*, *Libre, as in source code*, and *Compact*, vanilla and *libpd* can be boiled down to a few hundred kilobytes in headless system.

- Flexible, many I/O objects
- Community
- Understandable visual dataflow
- Great for education

Installing Pure Data

Some choices

- Vanilla versus Extended
- Hardfloat, RPi optimised
- Monolithic or Debian components

Log in from the host using fast X forwarding

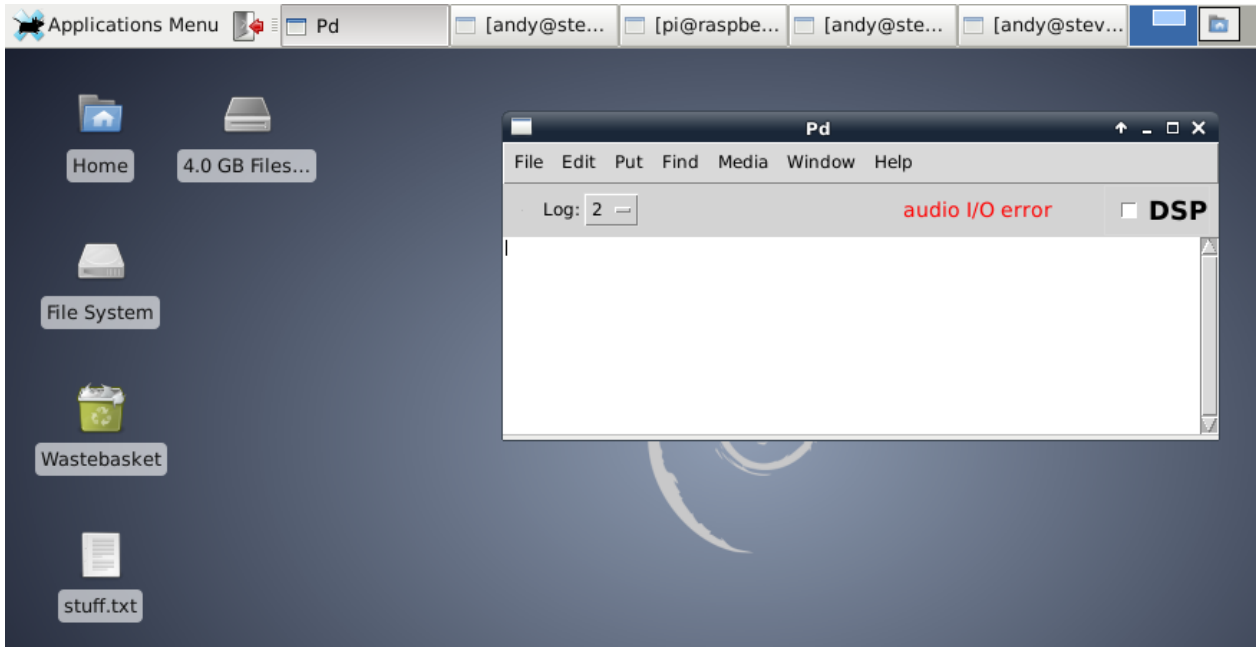
```
ssh -c blowfish -l pi -Y 192.168.2.2
```

Clientside

```
apt-get install puredata
puredata
```

Running Pure Data

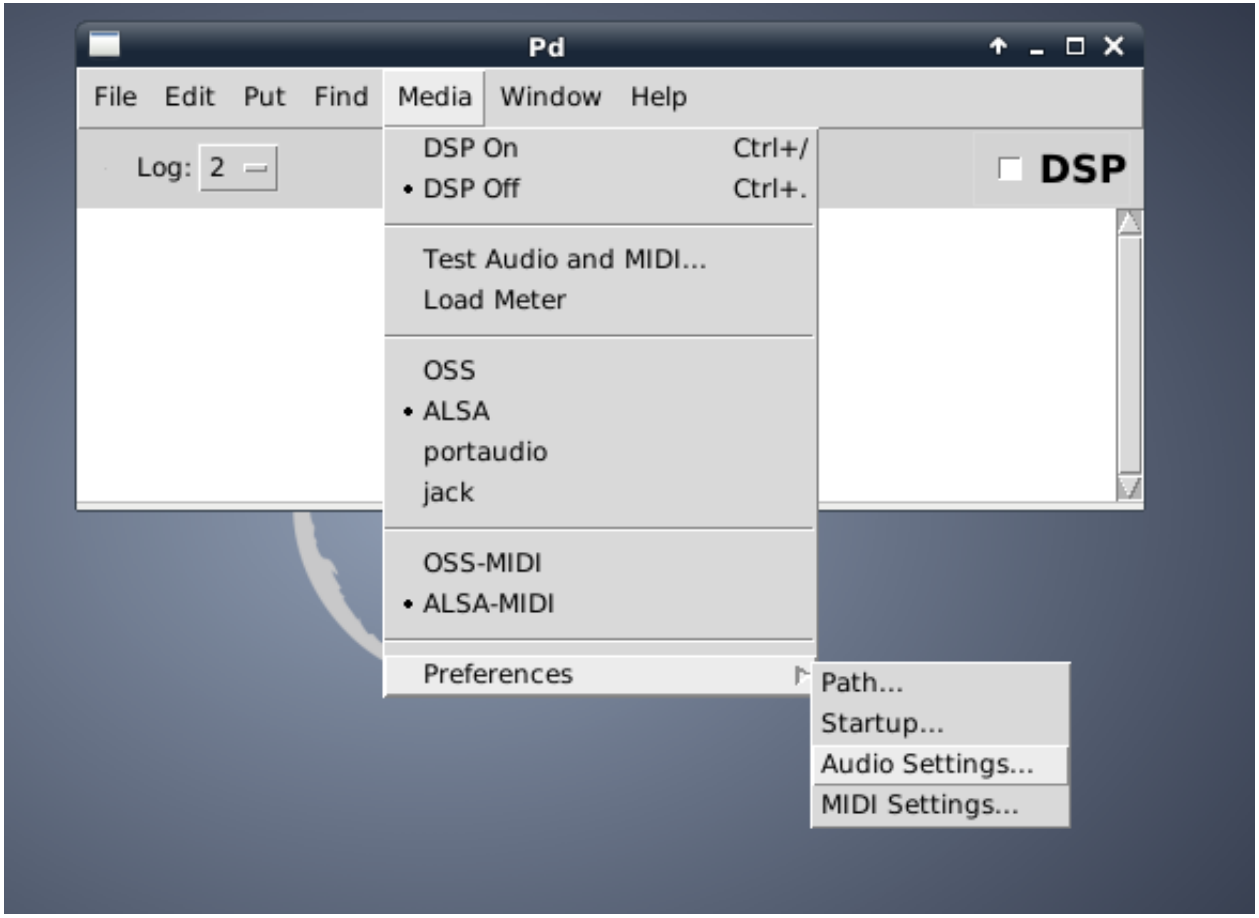
Running the GUI on host side



We will need to do some setup of the audio before it can be used

Configuring Pure Data Device

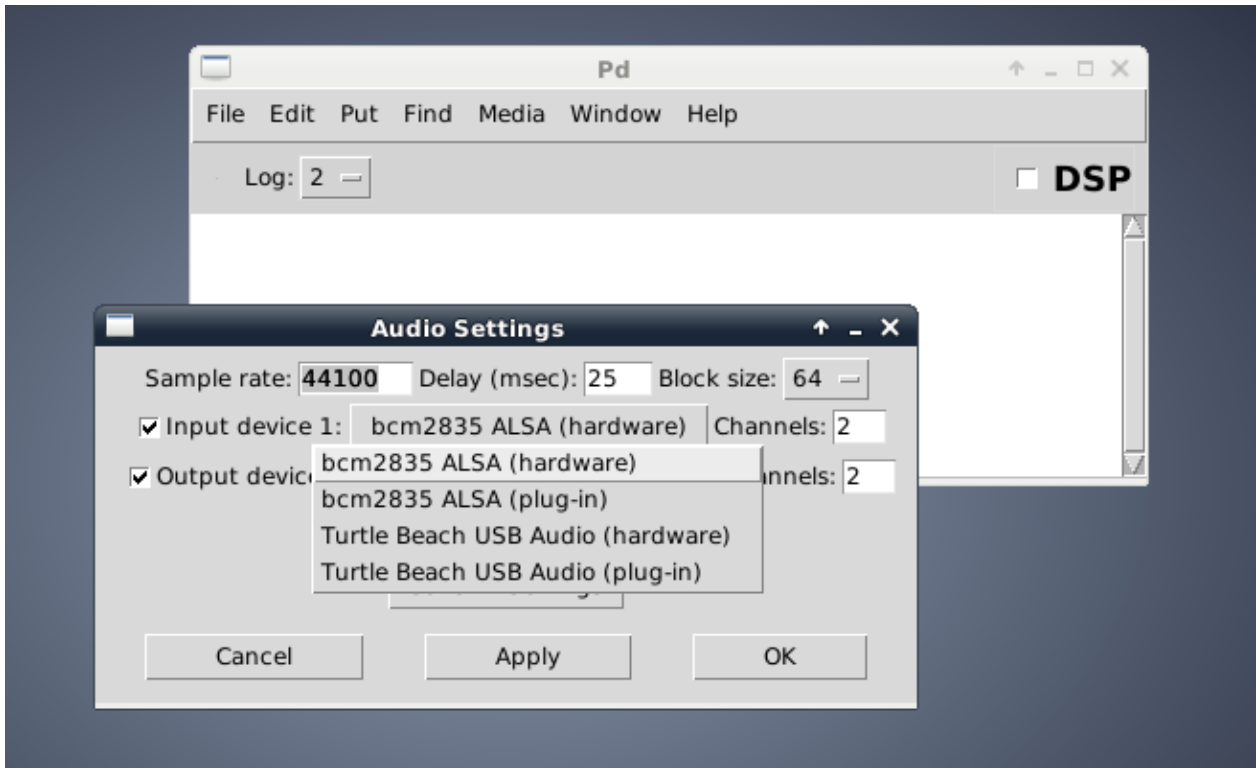
Set up Audio device



Configuring Pure Data Buffer

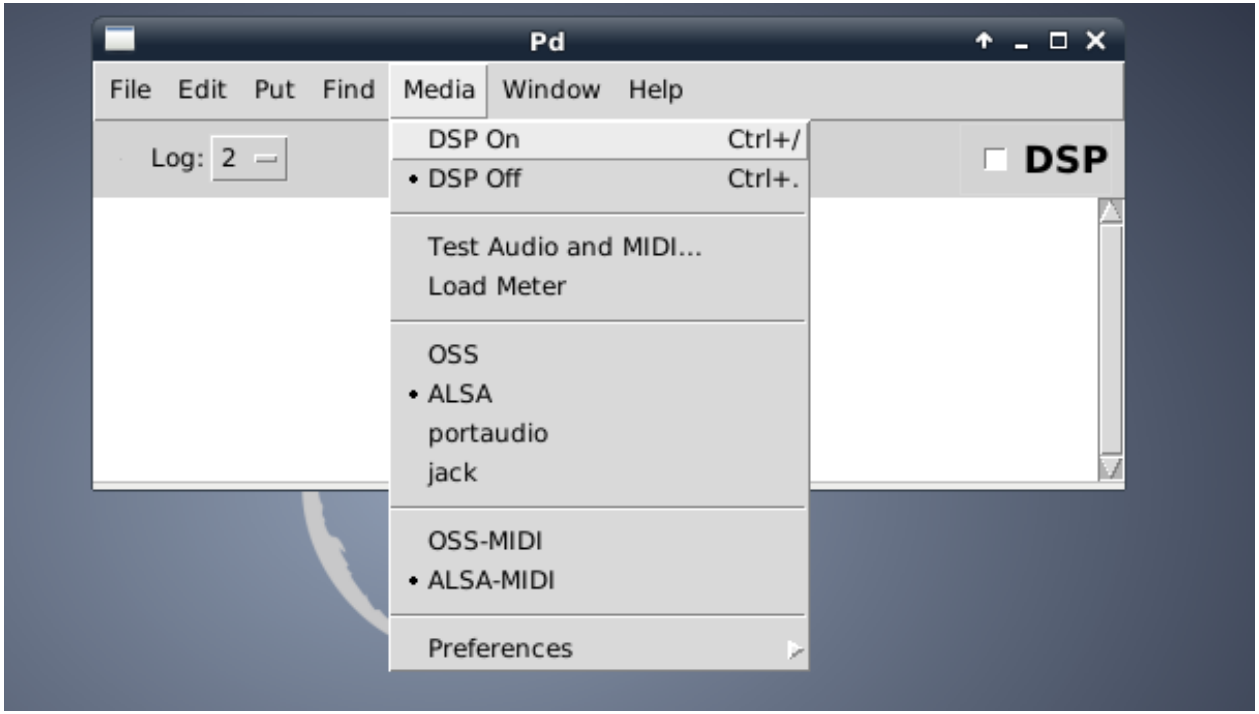
Select the appropriate device.

Adjust latency to a large amount (>about 100mS) while usng the GUI over X forwarding.



Activate Pure Data DSP

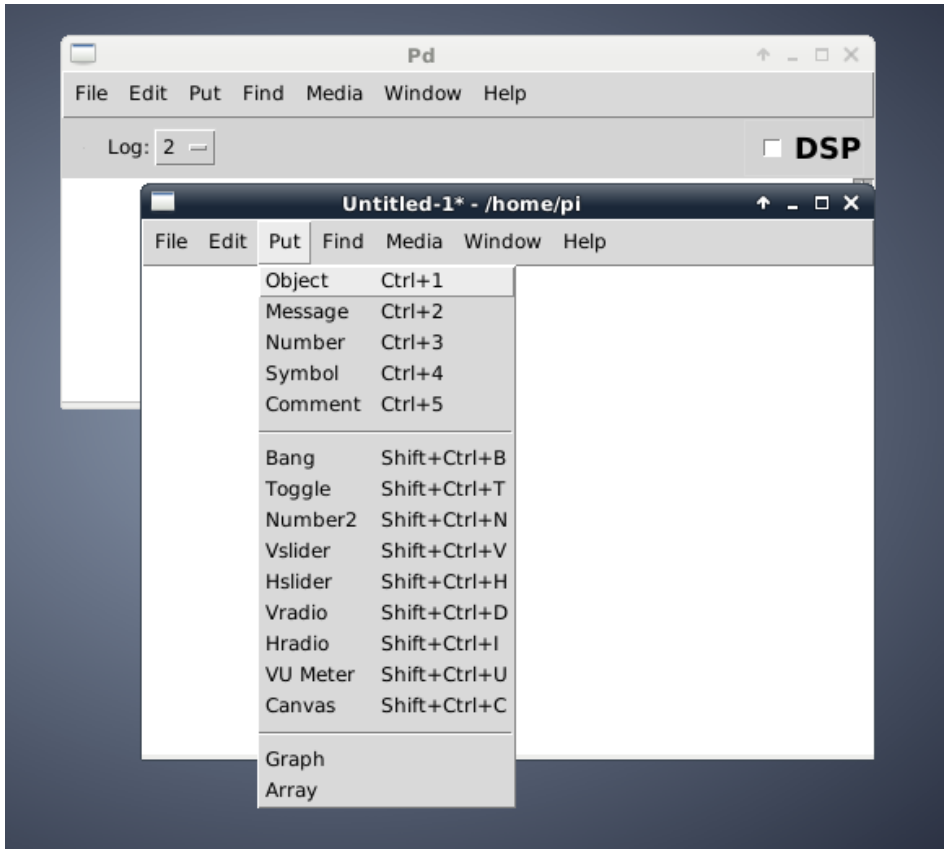
Switch on the DSP



Then `Test Audio and MIDI`

Patching with Pure Data

Create a new canvas

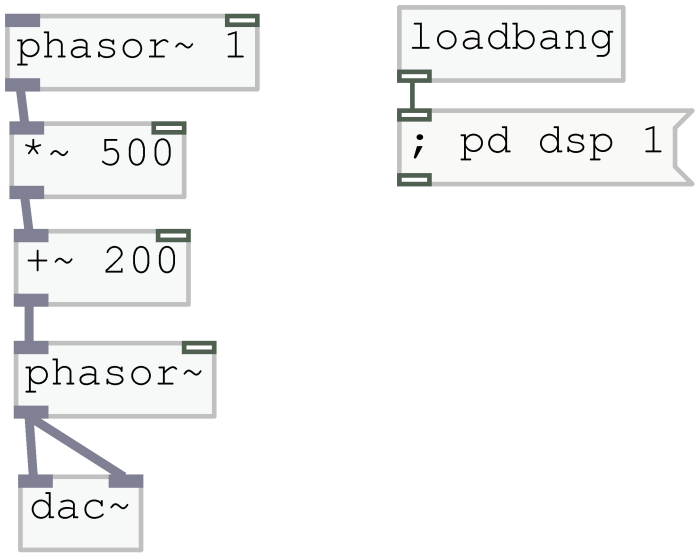


Some light reading to help :)

- *resources/pdfs/AudioProgramming/puckette.pdf*
- *resources/pdfs/AudioProgramming/farnell.pdf*

Patching a simple siren

Let's try a simple patch



Test it out and save as `siren.pd`

Headless siren

Running Pd without the GUI

- GUI takes up processing power
- X Communication over Ethernet
- Better latency and performance
- Step towards full embedding (libPd)

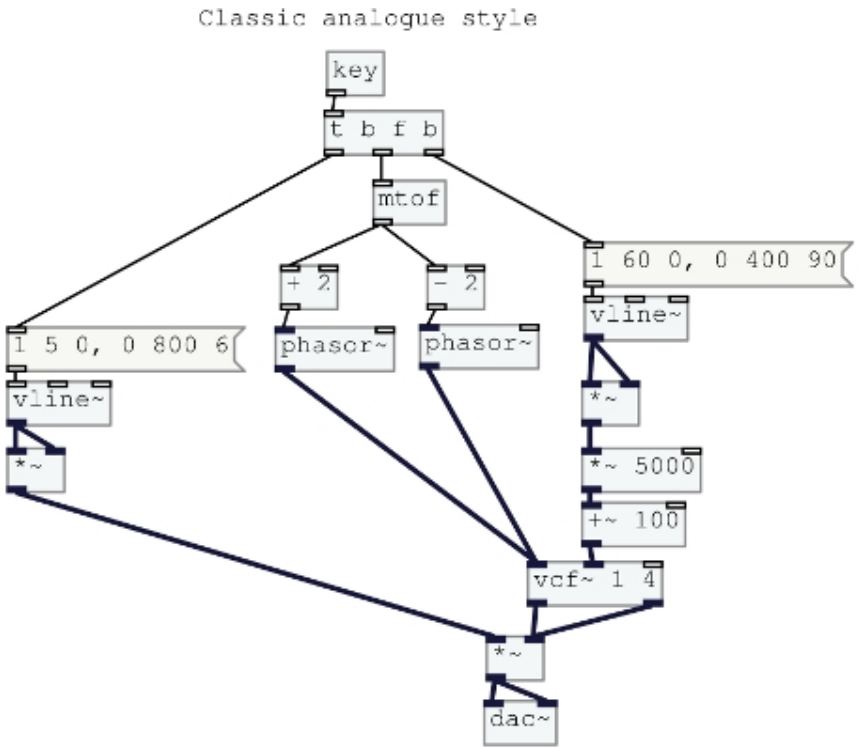
We need to know some command line options for Pd. You can browse these by typing `puredata --help`.

- `-nogui` stops GUI from launching
- `-noadc` disable audio in (more efficicent)
- `-alsa` say which audio layer to use
- `-audiobuf` set buffer size
- `-r` sampling rate

```
pd -nogui -noadc -alsa -r 48000 -audiobuf 100 siren.pd
```

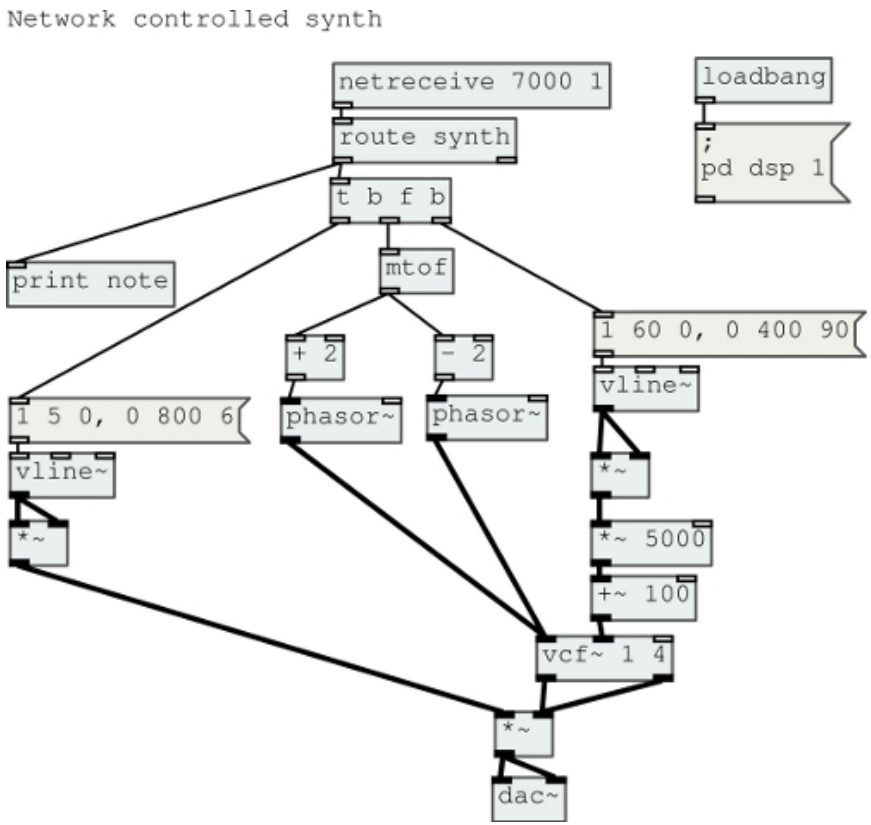
Synthesiser

Using the GUI again



Network controlled synthesiser

Edit, and run this headless



Can talk to it from the host

```
nc -u localhost 7000
synth 42
```

More Advanced Topics

USB sound devices

The audio hardware on the RPi is a bit poor, and there is no audio input, so we cannot use the RPI as a processor. For better *output* quality an option is to use the *HDMI* with a device that can act as a DAC. Better is to use am external USB sound device. These have low power consumption and even the cheap ones are super quality now (24bit/48kHz). For example;

It is likely that ALSA will detect it on boot up. Check recognised ALSA devices;

```
aplay --list-devices
```

Card and subdevice are identified. To refer to them we use a comma separated pair after a colon for the `-Dhw` option.

```
aplay -Dhw:1,0 -r 48000 -c 2 -fS16_LE song1.wav
```

ALSA sound

As you may hear, there are some issues. When using low level audio there is no automatic configuration. Here there is a smple rate mismatch. In ALSA sampling rate is managed by the *plug* resampling option. But this requires some more detailed setup than we will do here. See *http://alsa.opensrc.org* An important feature to understand is part of *alsa-lib* called *Dmix* that allows software mixing of multiple sources (so you can crossfade music or overdub with a microphone. Many ALSA features are set up by using the `.asoundrc` file in your home directory. See *http://alsa.opensrc.org/.asoundrc* for more.

USB sound recording

At present the audio recording using arecord (and audio generally on the RPi is extraordinarily poor). There are many reasons for this. High quality audio requires devices to be extensively tuned, more so than video. For example other daemons should not be running. USB disk access needs tuning if long files are to be recorded. However, to experiment, begin with:

Recording

```
arecord -N -Dhw:1,0 -B480000 -fS16_LE -c2 -r 48000 -t wav out.wav
```

Playback

```
aplay -Dhw:1,0 -fS16_LE -c2 -r 48000 out.wav
```

A more professional audio recorder could probably be built arouns *ecasound*. See *http://www.eca.cx*

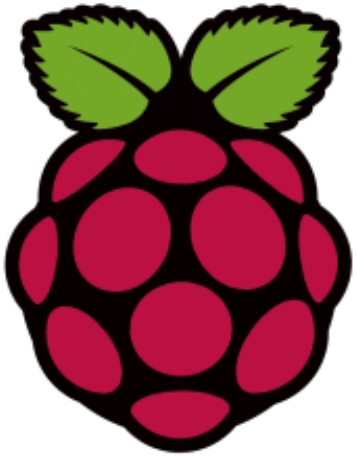
Speech synthesis


```
apt-cache search festival
apt-get install festival
```

```
echo "My goal is simple.  It is a complete understanding of
the universe, why it is as it is and why it exists at all." |
festival --tts
```

Conclusion

Voila, c'est fini. Merci!



In general

- Very cute
- Flexible
- Revolutionary price
- Thousands of applications
- Revolutionary price
- Will inspire technology growth

Audio specific

- Audio needs work to use limited power
- For experimental and educational aims
- Performance possibilty, portable