2009

# Enabling Musical Applications On A Linux Phone

Greg Schiemer
*University of Wollongong*, schiemer@uow.edu.au

E. Chen
*Royal Melbourne Institute of Technology*

## Publication Details

This conference paper was originally published as Schiemer, G and Chen, E, Enabling Musical Applications On A Linux Phone, in Proceedings of ACMC09, Improvise, The Australasian Computer Music Conference, Queensland University of Technology, 2-4 July 2009.

# ENABLING MUSICAL APPLICATIONS ON A LINUX PHONE

*Greg Schiemer*
Sonic Arts Research Network
Faculty of Creative Arts
University of Wollongong
2522

*Eva Cheng*
School of Electrical and Computer Engineering
RMIT Melbourne
3000

**ABSTRACT**

Over the past decade the mobile phone has evolved to become a hardware platform for musical interaction and is increasingly being taken seriously by composers and instrument designers alike. Its gradual evolution has seen improvements in hardware architecture that require alternative methods of programming. Dedicated I/O instruction sets for dealing with the idiosyncracies of various embedded peripheral devices are gradually being overtaken by I/O control using generic software that behaves more like operating systems developed for mainframe computers over three decades ago. This paper looks at the Neo FreeRunner, an open source mobile phone programmed using Linux. Its attraction as a platform for musical instrument development is that many musical applications created using open source cross platform software that once ran only on desktop computers can be now run in an embedded environment. The paper documents procedures we used in order to run musical applications effectively in the Neo FreeRunner. Musical motivations for using this platform can also be found in musical instrument development with j2me phones that provided a foundation for the creative work of the first author over the past 4 years.

## 1. INTRODUCTION

The Neo FreeRunner is a Linux phone that includes a range of hardware features such as wireless network connectivity, embedded peripherals and increased processing power offered by an advanced RISC processor. It offers developers an open source embedded platform with the scope to create a new genre of electronic music involving interaction with live performers.

Its operating system is not the only aspect of the Neo FreeRunner that is open source. Its circuit schematics and component layout can be downloaded making it possible for developers to customise hardware design. CAD files are even available for the casing of the Neo FreeRunner under a ShareAlike Creative Commons license.

Our approach has been to develop the musical application for the Neo FreeRunner, emulate this in a desktop environment and run it on the mobile platform without modifying the code emulated on the desktop.

The prospect of using compiled Arm9 native code offers a way to synthesise music using generic music software such as Pure data and Csound rather than interpretive languages like java and python which have been used in mobile devices [1, 2]. A similar approach to mobile synthesis has been adopted using the Symbian operating system [3].

The Linux environment is more suited to the development of new applications in embedded hardware than the j2me environment which had previously been used by the first author for developing musical applications [4, 5, 6].


**Figure 1**. The Neo FreeRunner Linux Phone.

## 2. BACKGROUND

In many respects the Linux tool set resembles the firmware library developed for microcontroller hardware such as the MIDI Tool Box - hereafter called MTB [7]. However, the capabilities of the MTB and Neo FreeRunner differ in three significant ways.

Firstly, the Neo FreeRunner has a 400 Mhz 32-bit RISC processor, an engine theoretically capable of real-time synthesis; the MTB is based on an 8MHz microcontroller and is more suited to DIY musical applications using physical computing devices.

Secondly, the Neo FreeRunner RAM address space is large enough to run and store musical applications on a handheld device; moreover, it is possible to download

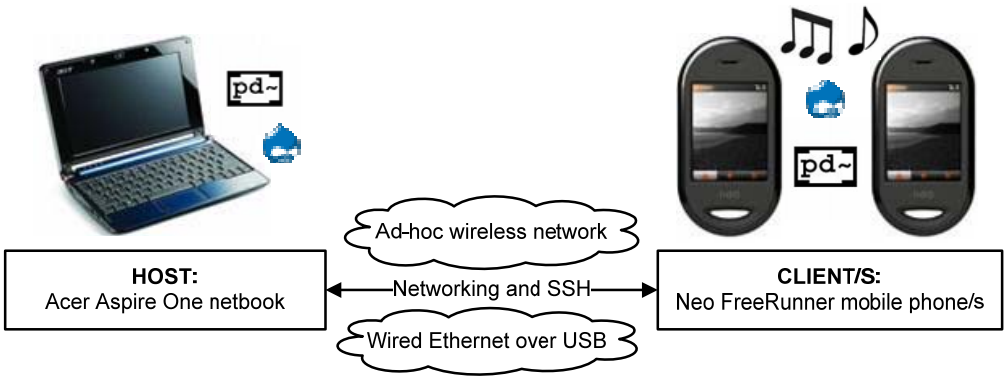


**Figure 2**. The Acer netbook host and FreeRunner client system paradigm

and run musical applications developed on a variety of standard PC platforms making it unnecessary to create an embedded library of firmware objects that allowed MTB users to program MIDI applications.

Thirdly, the low voltage power requirements of the Neo FreeRunner are suitable for a handheld or wearable device, whereas the power requirements of the microcontroller[1] at the core of the MTB were designed with automotive applications in mind.

## 3. DEVELOPMENT ENVIRONMENT

While development in future might potentially involve a host machine running a non-Linux operating system such as Windows or MacOS, the development environment we adopted was an Acer netbook used as a host terminal. This was used to communicate with the phone client via wifi or alternatively via USB as shown in Figure 2. A remote terminal was necessary to overcome the physical constraints that the limited screen size and stylus input places on most mobile phone users. Initially, we used the netbook's default Linpus Lite system, a derivative of the Fedora Linux distribution, but found it necessary to adopt the user-friendly Debian-based Ubuntu (version 8.10) instead. Ubuntu offered a richer set of libraries from the software repositories for application development whilst also providing optimisations for the netbook hardware.

Although the Ubuntu software repositories greatly aided the installation of music composition software, there were also several known bugs with Ubuntu on the Acer notebook that are currently being addressed by the Debian/Ubuntu Linux community, including: failing to suspend when the lid is closed (instead the screen just turns off, easily draining the battery); not recovering from hibernating, requiring a hardware reset; sound sometimes not working after suspending, requiring a logout or restart; putting the laptop into suspend which can sometimes corrupt the SD card partition table on waking up.

We also decided to replace the Neo FreeRunner's default OpenMoko operating system (version OM2007.2) with Debian in order to minimise problems that may arise with different Linux distributions in both the host and remote processors; further, similar to Debian-based Ubuntu on the Acer netbook, the Debian software repositories provided comprehensive package management for installing music composition software. To maintain the original OpenMoko operating system, the phone was dual-booted by installing Debian on the phone's microSD card and simply changing the booting order.

## 4. HOST

The Acer notebook is used as the host terminal. It establishes communication with the remote client via ethernet over USB or wifi. The Acer was setup to be used by non-expert computer users, with scripts automating procedures required to setup communications with the phone.

### 4.1. Logging In

For ease of use and quick startup time, Ubuntu on the Acer automatically logs in the user. However, users must enter their password when executing sudo commands or when bringing the laptop out of suspend mode.

### 4.2. Networking

All networking scripts print out the current networking configuration on completion so the user can see at a glance whether or not the script was successful. When enabling a networking interface (wifi or USB), that interface should have an Internet Protocol (IP) address, and conversely, when disabling an interface, the interface should be listed but without an IP address.

### 4.3. Ad-Hoc Wifi to the FreeRunner

Ad-hoc wifi (i.e., peer-to-peer) was chosen as the wireless connection paradigm as it enables multiple mobile devices (computers and/or phones) to join a secure wifi network without requiring any networking infrastructure such as access points. Note that running the ad-hoc wifi scripts provided disables the Ubuntu Network Manager which controls the Ethernet connection on the Acer, thus the wired Ethernet will not work! It is necessary to dis-

[1] Motorola MC68HC811E2

able the Network Manager to stop it from controlling the wireless interface.

#### *4.3.1. WiFi On*

To turn on ad-hoc WiFi from the Acer to the FreeRunner, a shell script was written to automate the process, to be run by the user by typing into the command line (in any directory):

```
./home/greg/Documents/adhocWifi_ON.sh
```

#### *4.3.2. WiFi Off*

To turn off ad-hoc WiFi to the FreeRunner, the user simply runs the provided script by typing into the command line (in any directory):

```
./home/greg/Documents/adhocWifi_OFF.sh
```

#### *4.3.3. WiFi Properties*

Ad-hoc WiFi connection properties are:

ESSID: `p2p`

Channel: `1`

WEP encryption key: 128-bit Hex key

IP address of Acer: `192.168.2.200`

IP address of FreeRunner: `192.168.2.202`

### 4.4. USB Networking to the FreeRunner

Internet access from the Acer to the FreeRunner can be enabled over the USB Ethernet connection. In order to do this there are also some settings that may need to be tweaked, including the Domain Name Servers (DNS) of the Internet Service Provider (ISP).

To change the DNS servers, edit the IP addresses in the following lines of the file `/etc/network/interfaces` on the FreeRunner (not the Acer!):

```
up echo nameserver 192.189.54.33 >
/etc/resolv.conf
```

```
up echo nameserver 203.8.183.1 >>
/etc/resolv.conf
```

It will be necessary for a user to know the IP addresses of the DNS servers on their local area network; DNS server IP addresses can be found by asking the ISP, network administrator, or ISP helpdesk. One or both IP addresses should then be edited into the `/etc/network/interfaces` file.

#### *4.4.1. USB On*

To turn on USB networking to the FreeRunner, to run the provided script the user simply types into the command line (in any directory):

```
./home/greg/Documents/usbNetworking_ON.sh
```

#### *4.4.2. USB Off*

To turn off USB networking to the FreeRunner, the user simply types into the command line (in any directory):

```
./home/greg/Documents/usbNetworking_OFF.sh
```

Note that exiting from an SSH session (see Section 4.5) with the FreeRunner will usually also automatically turn off USB networking on the Acer.

#### *4.4.3. USB Properties*

To distinguish the USB and wifi networks, different subnets are allocated in the IP addresses. USB networking connection properties are:

IP address of Acer: `192.168.0.200`

IP address of FreeRunner: `192.168.0.202`

### 4.5. Communicating with the FreeRunner over SSH

To use the terminal of the FreeRunner (as you would on the phone) over SSH, type into the command line (in any directory):

Over USB networking: `ssh root@192.168.0.202`

Over WiFi networking: `ssh root@192.168.2.202`

### 4.6. Using CSound 5.08

At the moment of writing there are occasional problems with CSound5 GUI which has crashed unexpectedly and without warning. This is a known bug with csound5gui on Ubuntu.

However, it is possible to run CSound 5.08 using a command line interface by typing into the command line (in any directory):

```
Csound test.csd
```

or

```
Csound –aiff –l test.orc test.sco
```

### 4.7. Using Pure Data 0.41-0

To run PD, click on the ‘Sound and Video’ item on the left hand menu on the Desktop, and click on the Pure Data icon. Or, type into the command line:

```
Pd
```

## 5. CLIENT

### 5.1. Installing Debian on the Neo FreeRunner

Debian was installed on the microSD card to preserve the factory install of OM2007.2. The only requirements for installing Debian on the microSD are Internet access to the FreeRunner and an existing Linux distribution in the FreeRunner’s internal flash memory; the factory in-

stall of OpenMoko is sufficient for this purpose. Installation occurs over the SSH command line to the FreeRunner. Installation requires the following steps:

#### *5.1.1. Prepare the microSD card*

Obtain a microSD card of at least 1Gb capacity. Note that not all microSD cards are compatible. We experienced problems with a SanDisk 2Gb card, with OM2007.2 claiming that it could not read the partition table or boot sector. A somewhat outdated list of supported cards can be found on the OpenMoko wiki[2].

#### *5.1.2. Prepare the networking*

Boot the FreeRunner into the Linux distribution installed in the internal flash memory (probably OM2007.2) and SSH in to get a command line terminal interface. Setup USB networking on the OpenMoko distribution with Internet access (so Debian can be downloaded and installed) – see Section 4.4, the process for setting up Internet access on OpenMoko is the same as for Debian. The USB networking interface should be enabled by default, and the USB networking part of the file `/etc/network/interfaces` should read something similar to:

```
auto usb0
iface usb0 inet static
address 192.168.0.202
netmask 255.255.255.0
network 192.168.0.0
gateway 192.168.0.200
up echo nameserver 192.189.54.33 >
/etc/resolv.conf
up echo nameserver 203.8.183.1 >>
/etc/resolv.conf
```

#### *5.1.3. Setting up WiFi*

To setup the ad-hoc wifi, the wifi part of the file `/etc/network/interfaces` should read something similar to:

```
auto eth0
```

```
iface eth0 inet static
address 192.168.2.202
netmask 255.255.255.0
network 192.168.2.0
gateway 192.168.2.200
wireless-mode ad-hoc
wireless-essid p2p
```

Note that this is an unencrypted and unsecure wireless connection. We could not get encryption to work on OM2007.2; hence, Internet access is not enabled over the wireless interface under OpenMoko.

[2] http://wiki.openmoko.org/wiki/Supported_microSD_cards

#### *5.1.4. Download and Install Debian*

Installing Debian to run from the microSD card can all be done from an installation script consisting of the following step-by-step instructions.

Download the Debian install script on the SSH terminal command line:

```
wget -O install.sh http://pkg-
fso.alioth.debian.org/freerunner/install.s
h
```

Then make the script executable by typing

```
chmod +x install.sh
```

The install we did on the FreeRunner did not create a swap partition; fdisk failed to create the partition table. We also set the boot partition to be FAT so it became unnecessary to modify the FreeRunner's boot environment.

The command used to install Debian was:

```
SD_PART1_FS=vfat ./install.sh all
```

However readers are advised to consult the following URL for more up to date instructions:

```
http://wiki.debian.org/DebianOnFreeRunner
```

#### *5.1.5. Boot into Debian*

If Debian installs without errors, it's time to reboot from the NOR. Turn off FreeRunner and then press the Aux and Power buttons simultaneously to boot to the NOR boot menu, choose the second menu option with the Aux button to boot from MicroSD (FAT+ext2), then press the Power button to execute, and Debian should boot!

#### *5.1.6. Login to Debian*

At this point USB networking should be enabled by default (settings from OpenMoko are ported over during the Debian installation process) and upon booting into Debian the user may expect to see either a command line interface or a phone GUI interface (Zhone). The user can then connect to the FreeRunner using SSH: USB and Internet access is enabled by default. The default password for the root user is empty. Once the user is logged in this should be changed using the `passwd` command.

#### *5.1.7. Install software from the Internet*

Use Debian's `apt` package manager to add/remove software packages, as this will ensure that package dependencies are always resolved.

Useful commands are:

```
apt-get update
```

This updates the list of available packages from Debian repositories online – the list of repositories is stored in `/etc/apt/sources.list`

apt-cache search package/s

This searches the cache of available packages, though packages are not always named in an obvious manner!

```
apt-get install package/s
```

This installs the package

```
apt-get remove package/s
```

This removes the package

```
apt-get upgrade
```

This updates Debian with the latest versions of installed software, patches, etc.

The default installation is lean so users are advised to install some useful packages such as:

```
build-essential
gcc
g++
libc6-dev
make: useful packages for compiling code
joe: an easy-to-use text editor (as Debian
only comes with vi)
man: to read manual pages for Linux com-
mands
mplayer: command-line media player
```

When installing new packages, `apt` will check for dependencies and request confirmation if additional packages are required. Users should always check dependencies to ensure that no essential packages are unintentionally removed.

#### *5.1.8. Install CSound*

Install CSound; the latest Debian packaged version available is 5.08:

```
apt-get install csound
```

Check and agree to the dependencies; for extra functionality search for and install additional CSound packages by typing:

```
run apt-cache search csound
```

#### *5.1.9. Install PD*

Install PD; the latest Debian packaged version available is 0.41.4-1:

```
apt-get install puredata
```

Check and agree to the dependencies; for extra functionality search for and install additional PD packages by typing:

```
apt-cache search puredata
```

#### *5.1.10. Enable Wifi Hardware*

The latest version of the Debian kernel[3] (at the time or writing) does not enable the wifi hardware by default when the FreeRunner boots up. The script below was created to wake up the wifi hardware; this will need to be run every time the FreeRunner starts up.

```
joe turnOnWifiHardware.sh
   export
   sys_pm_wlan=/sys/bus/platform/drivers/gt
   a02-pm-wlan/gta02-pm-wlan.0
   export
   sys_wlan_driver=/sys/bus/platform/driver
   s/s3c2440-sdi
   echo 1 | tee $sys_pm_wlan/power_on
   echo s3c2440-sdi | tee
   $sys_wlan_driver/unbind 2> /dev/null >
   /dev/null
   echo s3c2440-sdi | tee
   $sys_wlan_driver/bind
chmod +x turnOnWifiHardware.sh
```

To run the script, type

```
./turnOnWifiHardware.sh
```

To do this the script must be in the current directory otherwise the full filename path should be used; the current directory pathname can be retrieved with the command `pwd`.

#### *5.1.11. Setup Ad-Hoc Wifi Networking*

Create the following script to setup an ad-hoc wireless peer-to-peer connection; the WEP key - the string of 26 hex characters – and the ESSID - the ad-hoc wireless network name - can both be changed:

```
joe adhocWifi_ON.sh
   ifconfig eth0 down
   iwconfig eth0 channel 1
   iwconfig eth0 key <128-bit hex> essid
   “p2p”
   iwconfig eth0 mode ad-hoc
   ifconfig eth0 up
   ifconfig eth0 192.168.2.202
   ifconfig
chmod +x adhocWifi_ON.sh
```

Create the following script to take down the ad-hoc wireless peer-to-peer connection:

```
joe adhocWifi_OFF.sh
   ifconfig eth0 down
   ifconfig
chmod +x adhocoWifi_OFF.sh
```

[3] 2.6.28-20090105.git69b2aa26

To run either script, just type

```
./adhocWifi_ON.sh
```

or

```
./adhocWifi_OFF.sh
```

To do this the script must be in the current directory otherwise the full filename path should be used. The scripts will need to be run every time the FreeRunner's ad-hoc wireless network connection is set up or taken down.

### 5.2. Booting Neo FreeRunner

The FreeRunner can boot two systems: the original OpenMoko 2007.2 distribution (factory installed) from the internal flash memory, and Debian, which is stored on the microSD card.

To boot into the original OM2007.2 system, press the power button until the handset vibrates a little; this interface can be used for phone and SSH terminal functionality. We have not modified the OM2007.2 install aside from updating it and enabling WiFi and USB networking.

To boot into Debian for SSH terminal and CSound/PD functionality, press the Aux button and then the Power button, and keep both buttons depressed. The phone should vibrate a little until a boot menu appears; select the second option (Boot from microSD (FAT+ext2)) using the Aux button and press the Power button to execute. Debian should load and present the user with a command line.

The Debian interface can only really be used by accessing the terminal through SSH; the phone GUI (Zhone/Illume) doesn't work at the moment and we have yet to address and test X-Windows.

### 5.3. MUSIC SOFTWARE IMPLEMENTATION

At the moment of writing, Csound and Pure Data run successfully on both the Acer host platform and the Neo FreeRunner client; both programs produce sound on the Acer host platform but currently not on the Neo FreeRunner client. The special qualities of microtonally tuned sound produced at low power levels using Csound running on an untethered battery-powered processing device have already been demonstrated in a recent concert workshop[4] and will be demonstrated as part of this paper using either the host platform or, hopefully, the Neo FreeRunner client. We are also currently working on a demonstration to communicate control information via UDP between Pure Data applications running on the host and the client using Pure Data commands Netsend and Netreceive. This will enable the musical gestures produced by the motion of the untethered client device to interact with synthesis software running on the host platform. Alternatively, the client may become a mobile sound source that is modified while it is in motion. And unlike the Nokia phones used for the pocket gamelan, no purpose-built pouch is required for swinging the Neo FreeRunner (a gesture used with the pocket gamelan for creating Doppler shift) as it comes with a ready-made hole that is ideal for attaching a cord.

[4] *Butterfly Dekany* performed at GAUNG 21st Century Global Music Education Bali, 29th April 2009

## 6. REFERENCES


[1] Wyse, L. and Mitani, N. "Bridges for Networked Musical Ensembles", in *Proceedings of the International Computer Music Conference*, Montreal, CA, 2009, forthcoming

[2] William Duckworth and Nora Farrell 2007 'iOrpheus' August 31, http://www.iorpheus.com/

[3] Essl, G. Rohs, M. 2006."Mobile STK for Symbian OS in *Proceedings of the International Computer Music Conference*, New Orleans, USA, 2006

[4] Schiemer, G., and Havryliv, M. 2005 "Pocket Gamelan: a Pure Data interface for mobile phones" in *Proceedings of the 2005 International Conference on New Interfaces for Musical Expression (NIME05)*, Vancouver, BC, Canada pp. 51-56.

[5] Schiemer, G., and Havryliv, M. 2006 "Pocket Gamelan: tuneable trajectories for flying sources in Mandala 3 and Mandala 4" in *Proceedings of the 2006 International Conference on New Interfaces for Musical Expression (NIME06)*, Paris pp 156-161

[6] Schiemer, G. and Op de Coul, M. 2007 'Pocket Gamelan: tuning microtonal applications in Pd using Scala' in *Conference Proceedings of the Australasian Computer Music Conference* June 19th-21st pp. 81-86.

[7] Schiemer, G. *MIDI Tool Box: interactive tools for music composition*. (Phd thesis) Sydney: Macquarie University, March 2009