

OpenMoko

Free Software Phone Architecture

by

Harald Welte <laforge@openmoko.org>

Introduction

Who is speaking to you?

- an independent Free Software developer, consultant and trainer
- who is a member of the free software community for 10 years
- who has worked a lot on the Linux kernel
- who had originally started OpenEZX for Motorola phones
- and who's been lead hardware + system software architect for OpenMoko until recently

OpenMoko

WARNING

I have quit working for OpenMoko, Inc. or the FIC group.

Thus, I do not officially represent either of these entities!

What is OpenMoko

The commercial side

- First International Computer, Inc.
 - A large Taiwanese hardware vendor
 - Has a FIC Mobility business unit
 - Hardware R&D and production of Neo1973 GTA01 and GTA02 handsets

- OpenMoko, Inc., ("OpenMoko, the Company")
 - Part of First International Computer (FIC) Group
 - Funding the OpenMoko software R&D
 - Responsible for product definition, sales, marketing, PR, ...

What is OpenMoko

The community side

- OpenMoko, the overall Free Software project
 - A FOSS project working on
 - ▷ OpenMoko kernel/u-boot patches (hardware support)
 - ▷ OpenMoko GNU/Linux distribution
 - ▷ OpenMoko UI / framework
 - Funded by OpenMoko, Inc.

- OpenMoko, the embedded GNU/Linux distribution
 - An OE-built embedded GNU/Linux distribution for mobile communications devices
 - Primarily targetted at OpenMoko/FIC handsets
 - Is being ported to other devices by the community
 - Maintained by OE coreteam member employed by OpenMoko, Inc.

What is OpenMoko about?

- Open**
 - Opening up the formerly-closed mobile world
 - on any achievable level

- Mobile**
 - Mobile devices are the future

- Free**
 - 100% Free Software from driver through UI

What is OpenMoko about?

- FIC provides
 - experience in mass production of consumer electronics
 - experience in production of GSM handsets
 - experience in hardware development of GSM handsets

- OpenMoko provides
 - good contacts within the FOSS communities
 - strong technical knowledge on GNU/Linux
 - software development

Neo1973 GTA01 hardware

Neo1973 GTA01 hardware (07/2007)

- S3C2410 SoC @ 266MHz
- 2.8" 480x640 LCM, 262k colors
- 128MB SDRAM
- 64MB SLC NAND (512/16k)
- USB 1.1 device and host (unpowered)
- A-GPS (without processor)
- GSM+GPRS chipset (ARM7 based)
- Wolfson
- 2 stereo speakers (1.2W)
- 2.5mm headset jack
- CSR4 based Bluetooth
- NXP PCF50606 power management unit

Application Processor

Closer look at Application Processor

- SC2410 SoC @ 266MHz

- three UART's
- 133MHz SDRAM interface
- 66MHz external bus
- Two channels SPI
- IIS
- I2C
- SDIO
- TFT controller
- NAND controller

GSM Modem

Closer look at the GSM Modem

- ❑ Ti Calypso/Iota based chipset
- ❑ As proprietary as any other phone
 - runs proprietary nucleus OS
 - runs proprietary GSM stack
- ❑ Supports GSM voice/data/fax and GPRS
- ❑ Tri-Band GSM
- ❑ Very good TS 07.05 / 07.07 / 07.10 compliance
 - everyone can download the protocol docs from ETSI.org
 - no user/hacker needs access to NDA'd documents

Free Software stack

Free Software stack

- bootloader: u-boot current git (post-1.3)
- kernel: linux 2.6.22.5/2.6.24-rc4
- xserver: kdrive
- glibc
- glib
- gtk+
- pulseaudio
- gsmd / libgsmd

Development Model

Development Model

- "do embedded GNU/Linux the right way"
 - use and track current mainline code
 - actively contribute our code upstream
 - e.g. kernel goal: make vanilla 2.6.25+ kernel have all drivers
 - all code is immediately committed to public svn repository
 - development discussions happen on public mailinglists
 - all code developed by OpenMoko is FOSS licensed
 - everyone can contribute
 - no copyright assignments to OpenMoko

Build System

- We build
 - an embedded linux distribution
 - split in ipk packages (just like dpkg/rpm)
 - ipk feeds (just like apt-get/yum)

- We release
 - full source code in svn
 - all patches to all packages
 - the entire build system (built with OE)

- Our build system is public
 - Everyone can rebuild everything
 - ▷ cross-toolchain
 - ▷ u-boot / kernel image
 - ▷ application/library packages

Hackable Device

Hackable Device

- The device shall be under full user control
- Everyone should be able to hack it, at any level
 - hardware hacking (i2c, spi, gpio on test pads / connector)
 - system-level hacking (bootloader, OS)
 - UI level hacking
- Make entry barrier for development as easy as possible
- bootloader prompt via USB serial emulation
- Serial console
- JTAG for the people
- Provide Debug Board with embedded USB JTAG + serial adapter

Standards compliance

Standards compliance

- We use open/documented/available standards wherever possible
- Use official USB device firmware upgrade protocol
- Have charger behave 100% to USB spec (100/500mA)
- Use GSM chipset that follows GSM 07.07/07.10 closely

User control

User control

- The phone needs to be under control of the user, and the free software he uses
- Even backdoors or rogue GSM firmware shall not be able to intrude the privacy fo the user
- So we e.g. put the Audio codec (under explicit control from the Linux-running AP) between microphone/speaker and the GSM modem
- So we enable the Linux-running AP to cut power of the GSM modem
- Thus, free software (and thus the user) remains in ultimate control

GSM Integration

Application Processor GSM integration

- kernel line discipline implementation for GSM 07.10
- userspace GSM daemon with unix domain socket
- libgsmd with API for applications
- lightweight, doesn't have any dependencies aside from glibc
- we're working on gobject integration on top
- kernel part scheduled for mainline submission
- will support different phones / gsm chipsets
 - Various HTC devices with Linux
 - Motorola EZX phones using OpenEZX

Difference

Difference from other Linux phones

- 'others' discourage third parties from writing apps
 - ▷ you need explicit permission? WTF!
- 'others' try to make customers pay for a device that's still under manufacturer / GSM operator control
- 'others' use proprietary kernel modules
 - ▷ locks you into some old kernel version
- 'others' use proprietary bootloaders
- 'others' dont give you JTAG/serial access
- 'others' use proprietary UI toolkits
 - ▷ vendor lock-in
- 'others' dont give out their build system
- 'others' dont give out their firmware update tools

Neo1973 GTA02 hardware

Neo1973 GTA02 hardware ("soon")

- S3C2442B SoC @ 400 MHz (500MHz option)
- 2.8" 480x640 LCM, 262k colors
- 128MB SDRAM
- 256MB SLC NAND (2048/128k)
- USB 1.1 device and host (with power)
- A-GPS (fully autonomous firmware-based)
- GSM+GPRS chipset (ARM7 based)
- CSR4 based Bluetooth
- Atheros AR6k based 802.11b/g WiFi
- 2 3D accelerometers
- Smedia Glamo 3362 GPU
- NXP PCF50633 power management unit

GTA02: Smedia Glamo GPU

Smedia Glamo 3362 GPU

- 8MB internal SDRAM
- 16bit local bus interface to S3C2410
- 2D acceleration
- 3D acceleration
- h.263 codec (encode/decode)
- LCM controller
- SD-Card controller
- hardware JPEG encoder/decoder
- Camera interface and image processing (unused)

OpenMoko is writing 100% FOSS drivers (GPL/MIT licensed)

- kernel driver for core and framebuffer done
- Xglamofb making good progress

Future Hardware

GTA03

not public yet :)

GTX01

not public yet :)

GTX02

not public yet :)

Always in motion, the future is!

Neo1973 GTA01 Emulator

The Neo1973 GTA01 emulator

- based on popular qemu project
- full GTA01 hardware emulation, including
 - ▷ NAND controller
 - ▷ LCM controller
 - ▷ power management unit
 - ▷ GSM modem
 - ▷ touchscreen controller
 - ▷ SD card controller
 - ▷ ...
- you can run the exact same bootloader/kernel/rootfs images
- thus, no need to buy real hardware to start hacking
- e.g. NetBSD port has been done entirely on emulator!
- http://wiki.openmoko.org/wiki/OpenMoko_under_QEMU

How to contribute

- First: get hands-on experience
 - with emulator (free, based on qemu, full GTA01 emulation)
 - with real hardware (GTA01 now, GTA02 soon)
- follow instructions on the wiki, improve it with your feedback
- start local user / developer groups
- go through bugzilla, look for bugs in your favourite components
 - try to reproduce bug with current images
 - provide feedback
 - help by providing additional debugging information

How to contribute

- write your own gtk+ applications fit for 480x640 screen size and limited CPU
 - do development on your host pc (native)
 - then cross-compile for OpenMoko
 - then test on emulator or hardware
 - then build and package with OE
- go through projects.openmoko.org and contact project teams, help them out
- hang out on mailinglists and #openmoko on freenode.net
 - start sharing your experience with others with your experience

Online Resources

- <http://www.openmoko.org/>
 - ▷ portal site, just links everywhere else
- <http://wiki.openmoko.org/>
 - ▷ everything you (n)ever wanted to know about openmoko ;)
- <http://bugzilla.openmoko.org/>
 - ▷ documents all known bugs, please add/report and debug!
- <http://lists.openmoko.org/>
 - ▷ various mailing lists for Q&A and discussions
- <http://planet.openmoko.org/>
 - ▷ planet aggregating RSS feeds of various blogs
- [#openmoko](irc.freenode.net)
 - ▷ lots of developers hanging out there
- <https://direct.openmoko.com/>
 - ▷ for buying actual hardware

Thanks

□ Thanks to

○ Free Software Foundation

- ▷ for the GNU Project
- ▷ for the GNU General Public License

○ Motorola

- ▷ for never managing to entirely lock down their EZX device

○ OpenEZX community

- ▷ for continuing the effort that I once started

○ First International Computer, Inc.

- ▷ for believing in a 100% Open Source product
- ▷ for funding OpenMoko

○ mond

- ▷ my girlfriend, whom I've had terribly neglected

○ All my friends

- ▷ for whom I didn't have time a single minute in 18 months