The end of embedded Linux

(as we know it)

Chris Simmonds

2net Ltd.

21st February 2013

Chris Simmonds

- Has been using Linux as an embedded operating system since 1999
- Has been training others how to do the same since 2002
- Blog at `http://embedded-linux.co.uk/`
- Company web site: `http://2net.co.uk/`

Overview

- How embedded hardware has changed over the last 10 years
- The traditional approach to embedded Linux
- What do we really want from embedded Linux?
- Some options for the future
  - Mainstream distribution
  - Embedded distribution
  - Android
- Are they up to the job?

Evolution of embedded hardware

10 years ago: Embedded Planet RPX Lite

- MPC823 @ 80 MHz
- 16 MiB RAM, 8 MiB NOR flash
- $500

Today: Pandaboard ES

- TI OMAP 4460 ARM Cortex A-9 dual core @ 1.2 GHz
- 1 GiB RAM, 4+ GiB on SD Card
- $160

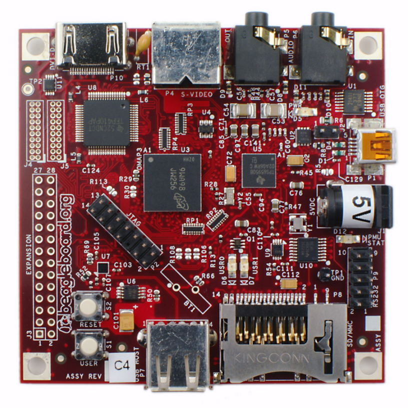
Changes

Hardware capabilities have increased many fold:

- Clock speed: 15 times
- RAM size: 64 times
- Storage: 512 times
- Cost: 0.32 times

What happened to cost?

The Beagleboard effect



A small board appearing in one place can cause huge disruption in other places

Embedded Linux - past

Embedded Linux evolved to cope with

- Non-PC architectures
- Board support packages for custom hardware
- Low RAM
- Small amount of storage
- Low clock speed
- Flash memory
- Robustness - must not fail
- Either headless, or
- User input from keypad or touch screen
    - No embedded device has keyboard or mouse

Embedded Linux - past

Engineers responded with

- Cross compilers (Crosstool(ng), etc)
- Busybox
- uClibc
- MTD, cramfs/squashfs, jffs(2), yaffs(2), ubifs
- Simplified startup scripts
- Stripped-down root file systems
- Read-only root file system (reduces wear on flash memory)
- Many, many custom BSPs and device drivers

Embedded Linux - past

The typical approach to a project used to be

- Find a toolchain
- Create a custom U-Boot
- Find a kernel for your SoC/board & build it
- Build a minimal root fs
- Cross compile libraries and applications as needed
- Tinker with it until it worked
    - Buildroot or OpenEmbedded helped a lot

Every project was different, always starting from scratch

Embedded Linux now and future?

- Clock speed (processor power), RAM and storage no longer an issue
  - Reduced need for Busybox, uClibc
  - Reduced need for custom rootfs, custom start-up scripts
- Storage moving from raw flash to eMMC and SD
  - Reduced need for mtd and flash filesystems jffs2
  - Increased need for SSD friendly software

Chance to re-think what constitutes "embedded Linux"

New problems

- Complexity
  - people want more from devices
- User interface
  - if it has a touchscreen it has to work like a smartphone
- Maintainability
  - all that software has bug fixes that need pushing to the field
- Skill level
  - doing things the old way requires a high level of skill
  - there are too many devices and not enough engineers
  - ergo, embedding Linux has to get easier

My ideal embedded Linux OS (1)

- Completely open source
- Architecture support: at least ARM, MIPS, PPC, x86
- Availability of board support packages for standard hardware
- Ability to create new BSPs for new and custom hardware
- Flexible
- Minimal base install (not all devices have 4 GiB to spare)
- Reduced system writes to preserve flash memory
  - No swap
  - Volatile /tmp, /run and parts of /var
  - Read-only root fs

My ideal embedded Linux OS (2)

- Proper system logging - that also reduces flash wear
- In-field remote update
- Fall-back boot if main boot fails (recovery mode)
- Debug and trace tools that can be used remotely
- Touchscreen support, including OSK (on screen keyboard) and fat fingered navigation
- Long term support

What are the options?

- Use a mainstream distro (e.g. Debian, Ubuntu, Fedora)?
- Use an embedded distro (e.g. Open Embedded, Yocto)?
- What about Android?

In the next few slides I will look at the pros and cons and give marks out of 10 for MS (Mainstream Distro), ED (Embedded Distro) and AN (Android)

Completely open source

- None require closed source components to build or run
  - Although, silicon vendors cause problems with closed source drivers (especially GPU)

MS 10	ED 10	AD 10

Architecture support

Points awarded for architecture dependant repositories and cross tool chains

- Mainstream distros
  - All support x86 and most support ARM v7
  - Some (e.g Debian) also support older ARM, PPC and MIPS
- Embedded distros
  - All the main archs
- Android
  - ARM v5te and v7, x86, MIPS

MS 7	ED 10	AD 8

Board support packages

- Mainstream distros
  - Most have binary images for standard hardware (e.g Beagleboard)
  - Only Linaro has a tool for creating a BSP:
    - linaro-media-create joins a generic binary with hardware pack
    - But of course, Linaro is not a distro
- Embedded distros
  - Yes, of course
- Android
  - "lunch combo" does the job

MS 2	ED 10	AD 10

Flexibility

- Mainstream distros
  - Very flexible: many 1000's of packages to choose from
- Embedded distros
  - Quite flexible
- Android
  - 100,000's of apps, but that it not relevant here
  - Base OS very inflexible

MS 10	ED 7	AD 0

Minimal base install

- Mainstream distros
  - Most have a minimal configuration of a few 100's MiB
  - Examples: Emdebian Grip, Ubuntu Core
- Embedded distros
  - Minimal rootfs about 16 MiB
- Android
  - A default build of AOSP: 160 MiB
  - Note this is the whole stack but since Android is monolithic you can't have anything less

MS 7	ED 10	AD 8

Reduced writes/flash friendly

Points awarded for

- No swap
- Volatile /tmp, /run and parts of /var
- Read-only root fs

- Mainstream distros
  - Disable sway - easy
  - Volatile /tmp - quite easy
  - Read-only root - hard or very hard
- Embedded distros
  - All have no swap and volatile /tmp
  - Little support for read-only root
- Android
  - Yes, has all of these features

MS 2	ED 7	AN 10

Proper system logging

Minimal requirement - avoid many small writes to /var/log

- Mainstream distros
  - Have Busybox syslogd as an optional package
- Embedded distros
  - Use Busybox syslogd -C by default
- Android
  - has logcat: multiple ring buffers for various sub systems

MS 3	ED 5	AD 5

Logging in Linux is a mess

Remote update and fall-back boot

The problem

- In-field update of kernel and packages is a must
- Problems occur at the boundary between BSP and distro
  - BSP includes kernel, modules and user space config files
  - Introduces dependencies that are difficult to resolve separately
- Mainstream distros
  - In my experience, they don't handle updates well
- Embedded distros
  - More flexible definition of arch-dependant feeds
- Android
  - With a custom build, it's up to you to build and distribute updates
  - Does have a well-defined recovery mode though

MS 0	ED 5	AD 5

User interaction

Points here for touch screen navigation:

- OSK - on screen keyboard that pops up when needed (and goes away after)
- simplified navigation for fat fingers
- full-screen apps
- ideally, multitouch gestures

- Mainstream distros
  - Unity, Gnome-shell, KDE Plasma Active are all headed in that direction
- Embedded distros
  - Not that I am aware of...
- Android
  - Yes, its mostly what Android is about

MS 5	ED 0	AD 10

Debug and trace tools

- Mainstream and embedded distros
    - All the standard packages
    - gbdserver, perf, oprofile, LTTng
- Android
    - gdbserver
    - Eclipse ADT plugin
    - adb is very handy
    - But lacks trace tools such as perf and LTTng

MS 9	ED 9	AD 9

Long term support

- Mainstream distros
  - Up to 5 years support for bug fixes and updates
- Embedded distros
  - No clear statement of update policy
  - Bug fixes rather slow to come through
- Android
  - No clear statement of update policy
  - Bug fixes rather slow to come through

MS 10	ED 5	AD 5

Final scores

- Points out of 110

MS 64	ED 78	AD 80

Note: this is totally unscientific, I admit that

Android is the winner?

- Kudos to the Android developers for creating a really good embedded OS
- But...
  - It is monolithic - hard to take apart and re-purpose
  - It is inflexible - hard to add bits on (e.g. normal FLOSS packages)
  - It is not a community project - doesn't have a life beyond Google
- Android is only good for devices that look like a smartphone or tablet

What then?

- Each can learn from the others, especially
- Mainstream distros
    - Better tools to create custom BSPs
    - Support flash memory properly!
- Embedded distros
    - Reduce complexity
    - Support in-field updates better
- All - create a logging solution that actually works

Conclusion

- This is not the end of embedded Linux, but ...
- ... it is the end of embedded Linux as a cottage industry
- Future devices will take more from mainstream distros (and be better for it)
- But, there is work to be done
- Hopefully the community (and industry) will fill in the gaps
- Or, there is Android for some classes of device