Handhelds Mojo: Building and running Ubuntu distributions on ARM

Andrew Christian  Brian Avery  George France
Handhelds Mojo: Building and running Ubuntu distributions on ARM

Andrew Christian  Brian Avery  George France
The Historical Approach

Development computer
- Ubuntu/RedHat/Debian…
- >12,000 packages
- Native build environment

Mobile device
- Maemo/OpenEmbedded…
- ~700-2500 packages
- Cross-build environment

The mismatch between development and mobile device is a nuisance…
What we’d like...

Precompiled software for mobile & embedded devices with:
- Large numbers of up-to-date packages
- Well-defined releases with security and bug fixes
- Easy interoperability with the developer’s desktop
- Code compiled and optimized for our specific device

What’s the quickest way to get this?
- Compile the desktop distribution for the mobile devices!
Common concerns

Aren’t mobile devices too small to run desktop software?

- 256 MB of RAM and an SD flash card is larger than the laptops we used just 9 or 10 years ago

The graphical user interface for the desktop doesn’t make sense on a mobile device!

- GUI applications represent only a small number of the applications in a distribution.
- Many desktop distributions contain small-device applications (e.g., Mobile Ubuntu)
Quick Summary

- The Mojo project has been rebuilding Ubuntu distributions for different flavors of ARM processors

<table>
<thead>
<tr>
<th>Mojo</th>
<th>Ubuntu</th>
<th>Released</th>
<th>v5</th>
<th>v5+VFP</th>
<th>v6+VFP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Frisky</td>
<td>7.04</td>
<td>March 2008</td>
<td>✔</td>
<td></td>
<td>✔</td>
</tr>
<tr>
<td>Grumpy</td>
<td>7.10</td>
<td>June 2008</td>
<td>✔</td>
<td></td>
<td>✔</td>
</tr>
<tr>
<td>Hasty</td>
<td>8.04</td>
<td>July 2008</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>Icy</td>
<td>8.10</td>
<td>Nov. 2008</td>
<td>✔</td>
<td>O</td>
<td>O</td>
</tr>
</tbody>
</table>

http://mojo.handhelds.org
Outline

- The challenges in building the distributions
  - Compilers, libraries, and toolchains
  - Native machine clusters
- The current state of the distributions
  - What works, what has been patched, what is missing
- How to use a Mojo distribution
  - Sample installation
  - Examples of systems that use the distribution
  - Performance
- Future work
Desktop distribution build process

Key points
- The build system is running its own packages. Iteration required!
- The build system runs on native hardware
- The toolchain is intrinsic to the distribution and gets compiled along with all of the other packages
Challenge: Toolchains

A toolchain is the combination of:

- C compiler (gcc)
- Linking and object tools (binutils)
- Standard C libraries (glibc)

You can’t build a distribution without a good, stable toolchain. But you can’t build the toolchain without a matching distribution….so you iterate.
Ubuntu’s toolchains

- The quality of ARM code produced and the number of architectures supported have generally improved over time.

<table>
<thead>
<tr>
<th></th>
<th>gcc</th>
<th>binutils</th>
<th>glibc</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dapper</td>
<td>4.0.3-1</td>
<td>2.16.1.cvs2006…</td>
<td>2.3.6-0ubuntu20</td>
</tr>
<tr>
<td>Edgy</td>
<td>4.1.1-6ubuntu3</td>
<td>2.17-1ubuntu1</td>
<td>2.4-1ubuntu12</td>
</tr>
<tr>
<td>Feisty</td>
<td>4.1.2-1ubuntu1</td>
<td>2.17.20070103…</td>
<td>2.5-0ubuntu4</td>
</tr>
<tr>
<td>Gutsy</td>
<td>4.1.2-9ubuntu2</td>
<td>2.18-0ubuntu3</td>
<td>2.6.1-1ubuntu9</td>
</tr>
<tr>
<td>Hardy</td>
<td>4.2.3-1ubuntu3</td>
<td>2.18.1~cvs2008…</td>
<td>2.7-10ubuntu3</td>
</tr>
<tr>
<td>Intrepid</td>
<td>4.3.1-1ubuntu2</td>
<td>2.18.93.2008…</td>
<td>2.8~20080505-…</td>
</tr>
</tbody>
</table>
Verifying you have a good toolchain

A “good” toolchain is one that passes most of its test suites.

- ARM is not the most popular architecture: building a “good” ARM toolchain requires a fair bit of testing and patching.
- Toolchains depend in surprising ways on all sorts of other packages (e.g. Perl, bash, …)
- Number of errors from test suite decreases as you iterate; for example, for gcc 4.1.2, we went from 11 to 5 to 0 with each iteration.

To maximize distribution quality, we iteratively compile each distribution at least 3 times.
# What, exactly, do you compile for?

<table>
<thead>
<tr>
<th>Compiler option</th>
<th>Choices</th>
<th>What it affects</th>
</tr>
</thead>
<tbody>
<tr>
<td>Application Binary Interface</td>
<td>Old ABI, EABI (1-5)</td>
<td>Data structure alignment, how parameters are passed to/from functions, kernel interface</td>
</tr>
<tr>
<td>Floating point</td>
<td>Hardware, Software, Vector (VFP)</td>
<td>Format of floating point numbers and execution speed</td>
</tr>
<tr>
<td>Endian</td>
<td>Little/Big</td>
<td>How words are stored in memory</td>
</tr>
<tr>
<td>ARM Architecture</td>
<td>v3, v4, v5, v6, v7…</td>
<td>The instruction set</td>
</tr>
<tr>
<td>Thumb</td>
<td>Non-thumb/Thumb (&amp; version)</td>
<td>Code size, execution speed, &amp; interoperability</td>
</tr>
<tr>
<td>Target processor</td>
<td>E.g. Xscale</td>
<td>Optimization for a specific processor or family</td>
</tr>
</tbody>
</table>

ARMv5EL = EABI, soft FP, little endian, v5, non-thumb
Challenge: Handling the “native” problem

Desktop distributions are not cross-built: you need an ARM-based machine to build an ARM-based distribution

- Option #1: Fundamentally change the build system using something like Scratchbox.
  - We couldn’t find a good way to do this without a lot of source package modifications
- Option #2: Create a build cluster of ARM-based machines...
Options for “native” build machines

<table>
<thead>
<tr>
<th>Pure ARM</th>
<th>QEMU-SYSTEM-ARM</th>
</tr>
</thead>
<tbody>
<tr>
<td>ARM Distribution</td>
<td>ARM Distribution</td>
</tr>
<tr>
<td>ARM Kernel</td>
<td>ARM Kernel</td>
</tr>
<tr>
<td>ARM Hardware</td>
<td>Virtual ARM Hardware</td>
</tr>
<tr>
<td></td>
<td>QEMU-SYSTEM-ARM</td>
</tr>
<tr>
<td></td>
<td>x86 Distribution</td>
</tr>
<tr>
<td></td>
<td>x86 Kernel</td>
</tr>
<tr>
<td></td>
<td>x86 Hardware</td>
</tr>
</tbody>
</table>

In 2007 we looked at the time and cost to build a sufficiently fast cluster.
2007 cluster: Native ARM build machines

20 home-built 1U ARM boxes
- 600 MHz ARMv5 processors
- 32 hours to compile and run the test suite for gcc-4.1 (one box)
- 4 days to compile the main Ubuntu packages (about 3000)
2008 cluster: Virtual ARM build machines

17 Dell x86 workstations
   (34 virtual ARMv6/7 machines)
   - Emulated ARMv6 processor (2 on each workstation)
   - 25 hours to compile and run test suites for gcc-4.1
   - 2 days to compile main Ubuntu packages (about 3000)

About a 25% performance increase and a 60% cost decrease
Outline

- The challenges in building the distributions
  - Compilers, libraries, and toolchains
  - Native machine clusters
- The current state of the distributions
  - What works, what has been patched, what is missing
- How to use a Mojo distribution
  - Sample installation
  - Examples of systems that use the distribution
  - Performance
- Future work
Mojo Releases

<table>
<thead>
<tr>
<th>Mojo</th>
<th>Ubuntu</th>
<th>Released</th>
<th>v5</th>
<th>v5+VFP</th>
<th>v6+VFP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Frisky</td>
<td>7.04</td>
<td>March 2008</td>
<td>✔</td>
<td></td>
<td>✔</td>
</tr>
<tr>
<td>Grumpy</td>
<td>7.10</td>
<td>June 2008</td>
<td>✔</td>
<td></td>
<td>✔</td>
</tr>
<tr>
<td>Hasty</td>
<td>8.04</td>
<td>July 2008</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>Icy</td>
<td>8.10</td>
<td>Nov. 2008</td>
<td>✔</td>
<td>○</td>
<td>○</td>
</tr>
</tbody>
</table>

- Frisky was our test case
- Grumpy only exists as a bridge to Hasty
- Hasty is in very good shape and has updates
- Icy is very new
Statistics from a sample build

<table>
<thead>
<tr>
<th>Hasty ARMv5 EL</th>
<th>Main</th>
<th>Contributed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ubuntu source packages</td>
<td>3114</td>
<td>11188</td>
</tr>
<tr>
<td>Ubuntu binary packages</td>
<td>6151</td>
<td>18955</td>
</tr>
<tr>
<td>Source packages we modified</td>
<td>55</td>
<td>6</td>
</tr>
<tr>
<td>Completely built source packages</td>
<td>2921 (94%)</td>
<td>9591 (86%)</td>
</tr>
</tbody>
</table>

The majority of packages just build without modification
Why didn’t some packages build?

<table>
<thead>
<tr>
<th>Hasty ARMv5EL</th>
<th>Main</th>
<th>Contributed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Wrong architecture (not ARM)</td>
<td>58 (29%)</td>
<td>247 (15%)</td>
</tr>
<tr>
<td>Partially built – some of generated debs are not for ARM (e.g., Linux kernel)</td>
<td>36 (18%)</td>
<td>33 (2%)</td>
</tr>
<tr>
<td>Waiting on other packages that failed</td>
<td>81 (41%)</td>
<td>993 (61%)</td>
</tr>
<tr>
<td>Failed (e.g., Java)</td>
<td>22 (11%)</td>
<td>364 (22%)</td>
</tr>
</tbody>
</table>

Many packages fail to build completely because they aren’t for ARM
Challenges in package building

- Many Ubuntu packages come in build-dependency loops that must be manually unwound (e.g. KDE, Java)
- Some important packages simply don’t exist for ARM (e.g., Java)
- Some important packages have to be backported from later distributions (e.g. Mono, Fortran compilers)
- Some key packages have to be patched because we’re not officially part of Ubuntu (e.g., dpkg, apt, keyrings)
- Some packages just have errors (e.g., Qt float data type, minor fixes in Python)
A sample dependency loop (Java)
Outline

- The challenges in building the distributions
  - Compilers, libraries, and toolchains
  - Native machine clusters
- The current state of the distributions
  - What works, what has been patched, what is missing
- How to use a Mojo distribution
  - Sample installation
  - Examples of systems that use the distribution
  - Performance
- Future work
Debian Installer with QEMU

- Download installer components
  
  ```
  $ wget http://repository.handhelds.org/hasty-armv5el/installer-arm/images/versatilepb/ramdisk.gz
  $ wget http://repository.handhelds.org/hasty-armv5el/installer-arm/images/versatilepb/vmlinuz-926
  ```

- Create a QEMU disk image
  
  ```
  $ qemu-img create -f raw test.img 2G
  $ qemu-system-arm -M versatilepb -m 256M -kernel vmlinuz-926 \\
  -initrd ramdisk.gz -hda test.img -append “root=/dev/ram”
  ```

- Run the Debian Installer…
Debian Installer

Please choose the language used for the installation process. This language will be the default language for the final system.

This list is restricted to languages that can currently be displayed.

Choose a language:

- Albanian (Shqip)
- Arabic (عربي)
- Basque (Euskara)
- Belarusian (Беларуская)
- Bosnian (Bosanski)
- Bulgarian (Български)
- Catalan (Català)
- Chinese (Simplified) (中文 (简体))
- Chinese (Traditional) (中文 (繁體))
- Croatian (Hrvatski)
- Czech (Čeština)
- Danish (Dansk)
- Dutch (Nederlands)
- English (English)

<Go Back>

<Tab> moves between items; <Space> selects; <Enter> activates buttons
Debian Installer

[!] Configure the network

Please enter the hostname for this system.

The hostname is a single word that identifies your system to the network. If you don’t know what your hostname should be, consult your network administrator. If you are setting up your own home network, you can make something up here.

Hostname:

mojo

<Go Back> <Continue>

<Tab> moves between items; <Space> selects; <Enter> activates buttons
Debian Installer

[!!!] Choose a mirror of the Ubuntu archive

Please enter the hostname of the mirror from which Ubuntu will be downloaded.

An alternate port can be specified using the standard [hostname]:[port] format.

Ubuntu archive mirror hostname:

repository.handhelds.org

<Go Back> <Continue>

<Tab> moves between items; <Space> selects; <Enter> activates buttons
Debian Installer

请选择一个Ubuntu档案的镜像。

请输入Ubuntu档案镜像所在目录。

Ubuntu档案镜像目录：

/hasty-armv6el

<Go Back> <Continue>
Debian Installer

Installing the base system

Validating dpkg...
Debian Installer

[!] Install the base system

No installable kernel was found in the defined APT sources.

You may try to continue without a kernel, and manually install your own kernel later. This is only recommended for experts, otherwise you will likely end up with a machine that doesn't boot.

Continue without installing a kernel?

<Go Back> <Yes> <No>

<Tab> moves between items; <Space> selects; <Enter> activates buttons
Debian Installer

If you continue, the changes listed below will be written to the disks. Otherwise, you will be able to make further changes manually.

WARNING: This will destroy all data on any partitions you have removed as well as on the partitions that are going to be formatted.

The partition tables of the following devices are changed:
  SCSI1 (0,0,0) (sda)

The following partitions are going to be formatted:
  partition #1 of SCSI1 (0,0,0) (sda) as ext3
  partition #5 of SCSI1 (0,0,0) (sda) as swap

Write the changes to disks?

<Go Back> <Yes> <No>

<Tab> moves between items; <Space> selects; <Enter> activates buttons
Debian Installer

At the moment, only the core of the system is installed. To tune the system to your needs, you can choose to install one or more of the following predefined collections of software.

Choose software to install:

- DNS server
- LAMP server
- Mail server
- OpenSSH server
- PostgreSQL database
- Print server
- Samba File server
- Ubuntu desktop

<Continue>

<Tab> moves between items; <Space> selects; <Enter> activates buttons
Debian Installer

[!!!] Finish the installation

Installation complete

Installation is complete, so it is time to boot into your new system. Make sure to remove the installation media (CD-ROM, floppies), so that you boot into the new system rather than restarting the installation.

<Go Back> <Continue>

<Tab> moves between items; <Space> selects; <Enter> activates buttons
Moving beyond the installer

- Once the installer has finished, you can boot the image in QEMU with:

```
$ qemu-system-arm -M versatilepb -m 256M -kernel vmlinuz-926 -hda test.img \
   -append “root=/dev/sda1”
```

- If you’d like a graphical environment, try:

```
$ apt-get install xorg xfce4 gdm
# Edit /etc/X11/xorg.conf to include Driver “fbdev”
```

Instructions are on the website and the Mojo wiki
Running system

* Checking root file system...
  fsck 1.40.8 (13-Mar-2008)
  /dev/sda1: clean, 16393/121920 files, 96903/485958 blocks [ OK ]

  modprobe: FATAL: Could not load /lib/modules/2.6.25.10/modules.dep: No such file or directory

  * Checking file systems...
  fsck 1.40.8 (13-Mar-2008)
  [ OK ]
  * Mounting local filesystems... [ OK ]
  * Activating swapfile swap... [ OK ]

ls: cannot access /sys/module/apparmor: No such file or directory
ls: cannot access /sys/module/apparmor: No such file or directory
FATAL: Could not load /lib/modules/2.6.25.10/modules.dep: No such file or directory

Loading AppArmor module: Failed.
  * Checking minimum space in /tmp... [ OK ]
  * Skipping firewall: ufw (not enabled)... [ OK ]
  * Configuring network interfaces... [ OK ]
  * Setting up console font and keymap... [ OK ]
  * Starting system log daemon... [ OK ]
  * Starting kernel log daemon... [ OK ]
  * Starting deferred execution scheduler atd [ OK ]
  * Starting periodic command scheduler crond [ OK ]
  * Running local boot scripts (/etc/rc.local) [ OK ]

Handhilds 8.04 mojo tty1

mojo login:
After running “startx”
Examples of what you can do

- Robert Nelson has Beagle Board instructions
- Cortez has been working on the Sharp Zaurus
- Rabeeh Khoury (Marvell) has good stuff for the Marvell 78100 board (wicked fast ARM...)

![Beagle Board](image1)
![Sharp Zaurus](image2)
![Marvell 78100](image3)
Testing Hasty ARMv5EL & v6EL-VFP

XFCE4 + gdm

Beagleboard
Firefox on BeagleBoard (Hasty v5)

Handhelds Mojo | Feel the power! - Mozilla Firefox

Welcome to Firefox

User login

Navigation

Handhelds Mojo

Bringing the power of desktop distributions to mobile and embedded devices

Hasty ARMv6EL-VFP looking pretty good

Submitted by anstd on 2 October 2008 - 0:59pm

The ARMv6EL-VFP build of Hasty is available. The build form is running over the last thousand or so packages and will be done soon. Security and update feeds will follow once the build finishes them for the ARMv6EL-VFP build.

- Add new comment

Hasty ARMv5SEL with VFP support

Submitted by anstd on 24 September 2008 - 0:29am

The Hasty distribution compiled for ARMv5SEL with VFP support is now available. This distribution has been targeted to the latest ARMv5SEL processors. Security and updates will follow shortly after we receive them.
Does architecture matter?

- A quick performance test using Cairo to draw falling, spinning PNG and SVG files
- Tested on a TI BeagleBoard, we saw a 15-20% speedup from the Hasty ARMv5EL distribution to the Hasty ARMv6EL+VFP
Future work

- Building the Icy (8.10) release
- Submitting patches back to Debian and Ubuntu
- Considering adding a new architecture or two
- Considering building Debian
  …and using these distributions, of course…

http://mojo.handhelds.org