Bootstrapping the Debian and Ubuntu ARM64 Ports

Wookey

Linaro

6th November 2012

Who am I

- Free Software developer since 1990
- Unix sysadmin since 1996
- Arm Linux developer since 1999
- Debian developer since 2000
- Ubuntu development since 2010

Some things I had something to do with:
Survex, PsiLinux, ArmLinux book, Emdebian, bootfloppies, Therion, apt-cross, dpkg-cross, Debian cross-toolchains, OpenEmbedded, Netbook Project, LART, YAFFS, Balloonboard, xdeb, Multiarch, sbuild

- Currently an ARM secondee to Linaro

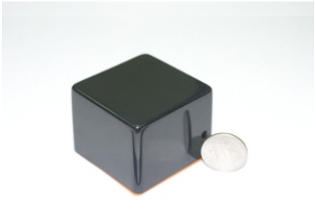
Outline

Outline

ARM desktops and servers

Acorn Risc PC (1994)

Rebel Netwinder (2000)


Solidrun Cubox (2012)

Dell/Calxeda server (2012)

Linaro

ARM laptops

Psion Netbook Pro (2003)

Toshiba AC100 (2010)

Genesi Smartbook (2010)

Samsung Chromebook (now)

Debian ports

Name	Bits	ABI	ISA	Released
arm	32	OABI	v3	2000:Potato (Discontinued 2011)
armeb	32	OABI	v3	2006:unofficial big endian
armel	32	EABI	v4t/v5	2009:Lenny, Ubuntu 9.05
armhf	32	EABI	v7	2012:Wheezy, Ubuntu 12.04
arm64	64	v8	v8	2013

Nomenclature

Simple version

arm64,aarch64,ARMv8 are all the same thing

More details:

arm64	Debian and Ubuntu architecture name
aarch64	ARM 64-bit execution mode
aarch64-linux-gnu	GNU triplet name
ARMv8	ARM CPU architecture name
A64	64 bit instruction set
A32	32 bit ARMv8 instruction set
aarch32	ARM 32-bit execution mode

Bootstrapping

20 Debian ports in 20 years

i386, 68000, Alpha, Sparc, PowerPC, ARM, IA64, PA-RISC, MIPS (Big endian), MIPS (little endian), S/390, AMD64, FreeBSD-i386, FreeBSD-amd64, armel, armhf, sh4, s390x, PowerPC64, Hurd-i386

Bootstrapping

20 Debian ports in 20 years

i386, 68000, Alpha, Sparc, PowerPC, ARM, IA64, PA-RISC, MIPS (Big endian), MIPS (little endian), S/390, AMD64, FreeBSD-i386, FreeBSD-amd64, armel, armhf, sh4, s390x, PowerPC64, Hurd-i386

Bootstrapping is normal, not exceptional
A 'Universal OS' should be able to bootstrap itself

Outline

Linaro

The Bootstrap Problem

- Build-dependency loops

The Bootstrap Problem

- Build-dependency loops

A more detailed look

The Bootstrap Problem

- Build-dependency loops
- Natively built
- Maximally configured
- Much worse for binary distros than source-based
- Lack of flexibility is in packaging

Outline

Bootstrap solutions

Traditionally

- Cheat and use something else
- Bodgery and Hackery
- No hardware yet - models are really slow

'Universal OS' solution

- Cross Building
- Build profiles
  - `Build-Depends: debhelper, byacc | bison, comerr-dev, docbook-to-man, libldap2-dev <!stage1>, libncurses5-dev`

Debian cross-build methods

Old dpkg-cross style

- Special tools for cross-dependencies: apt-cross, xapt, xdeb
- -cross packages created with dpkg-cross
- Libraries and headers in /usr/<triplet>

New Multiarch style

- Apt does cross-dependencies
- Standard library and headers paths
- Normal host architecture packages used

Multiarch

- Install libraries side-by side: i386/amd64, arm/arm64, amd64/arm64
  - /usr/lib/libfoo (amd64)→/usr/lib/x86_64-linux-gnu/libfoo
  - /usr/lib/libfoo (armel)→/usr/lib/arm-linux-gnueabi/libfoo
  - /usr/lib/libfoo (arm64)→/usr/lib/aarch64-linux-gnu/libfoo
- Packages arch-qualified: libfoo:arm64, wine:i386
- Packages marked Multi-Arch: Same, Foreign, Allowed
- Canonical file locations: Runtime is the same as build-time.
- Run foreign binaries in-place (natively or with qemu)
- 32/64 special casing goes away (/lib64, /emul/ia32-linux)
- Build/host version lockstep

Usage example

```
dpkg --add-architecture i386
apt-get install skype:i386
```

Outline

ARM internal Bootstrap

- Ubuntu Maverick
- Using xdeb, with staging support
- Equivs to fake toolchain dependencies
- Manual build order

So it's all done already?

ARM is an IP Company

So it's all done already?

ARM is an IP Company

Now I can be rude about ARM legal

So it's all done already?

ARM is an IP Company

Now I can be rude about ARM legal

- Paranoid about patent grants in FLOSS licences
- No cross-fixes, bootstrapping or arm64 support upstreamed
- Engineers disillusioned

So it's all done already?

ARM is an IP Company

Now I can be rude about ARM legal

- Paranoid about patent grants in FLOSS licences
- No cross-fixes, bootstrapping or arm64 support upstreamed
- Engineers disillusioned
- All has to be done again

```
+export DEB_BUILD_GNU_TYPE ?= $(shell dpkg-architecture -qDEB_BUILD_GNU_TYP
+
+ifeq ($(DEB_BUILD_GNU_TYPE), $(DEB_HOST_GNU_TYPE))
+  confflags += --build $(DEB_HOST_GNU_TYPE)
+  CROSS=""
+else
+  confflags += --build $(DEB_BUILD_GNU_TYPE) --host $(DEB_HOST_GNU_TYPE)
+  CROSS=$(DEB_HOST_GNU_TYPE)-
+endif
```

On the one hand great early community engagement

On the other complete failure to give back

Illustrates Linaro/corporate culture clash

Outline

Debian/Ubuntu Bootstrap Overview

Overview

- Ubuntu Quantal-based (and Debian Wheezy)
- All done in public from start - Upstreaming as we go along
- Multiarch building and cross-dependencies
- Standard tools: Sbuild, reprepro, apt, dpkg-cross
- Modified dpkg: build-profile support
- cross-build-essential

Linaro

Debian/Ubuntu Bootstrap Process

1. Prepare repository
2. Add new arch support to dpkg-architecture
3. Set up build chroot
4. Toolchain bootstrap
5. Fix support packages: dpkg-cross, cross-build-essential, autoconf
6. Build stuff. . .

How many packages do we need?

	Debian Sid	Ubuntu Precise
`base system`	62	94
`plus build-essential`	73	138
the above plus dependencies	106	140
number of source packages	55	75

Toolchain Bootstrap

'3-stage' bootstrap

1	Linux	linux-libc-dev headers
2	GCC stage1	Bare C-compiler
3	eglibc stage1	Minimal libc
4	GCC stage2	C-compiler against eglibc
5	eglibc stage2	Full libc build (without libselinux)
6	GCC stage3	All compilers

Automated by arm64-cross-toolchain-base

Toolchain Bootstrap

Bootstrap fun

- Need to build -source packages
  - linux-source
  - binutils-source
  - eglibc-source
  - gcc-4.7-source
- Kernel is 3.5, arm64 support is 3.7
- Quantal has eglibc 2.15, aarch64 support is glibc 2.16
- Package only builds with gcc-4.6, aarch64 support only for gcc-4.7
- Complicated packaging for Debian+Ubuntu, 12 architectures, Linux+BSD

Set up a chroot

`http://wiki.linaro.org/Platform/DevPlatform/CrossCompile/arm64bootstrap`

Create chroot

```
apt-get install sbuild
sudo sbuild-createchroot
  --make-sbuild-tarball=/srv/chroots/quantal-cross-arm64.tgz quantal
  /srv/chroots/quantal http://archive.ubuntu.com/ubuntu/
```

Build flags

```
STRIP CFLAGS -fstack-protector
APPEND LDFLAGS -L/usr/lib/aarch64-linux-gnu
               -L/lib/aarch64-linux-gnu -L/usr/lib
               -Wl,-rpath-link=/usr/lib/aarch64-linux-gnu:
               /lib/aarch64-linux-gnu:/usr/lib
```

Apt preferences

```
Package:  *
Pin:  release n=quantal-bootstrap
Pin-Priority:  1001
```

Building Packages

Getting Build-Deps and building is simple

Manually

```
apt-get install crossbuild-essential-arm64
apt-get build-dep -aarm64 acl
apt-get source acl
cd acl-2.2.51
dpkg-buildpackage -aarm64
```

Using sbuild

```
sbuild -c quantal-bootstrap -d quantal
  --host=arm64 acl_2.2.51
```

Linaro

Dependency analysis

Dependency analysis

```
dose-debbuildcheck --deb-native-arch=amd64
--deb-foreign-archs=arm64 --deb-host-arch=arm64 <packages
files> <source file> -f -e -s --checkonly <package>
```

Output

```
      package: src:dpkg
      version: 1.16.7ubuntu3profile1
      architecture: any,all
      essential: false
      unsat-dependency: arm64:liblzma-dev
```

Outline

Current status

- Nearly all Ubuntu so far
- Cross toolchain available to install
- Cross-support packages available
- Perl and python multiarched
- Arch all packages are easy
- 25 source Packages built
- Maybe 100 to go for base image
- No real testing beyond compiler yet

TODO: update perl cross-build

Getting involved

This is an open effort - help is welcome.

Just trying arm64

OE-based image `http://www.linaro.org/engineering/armv8`

Helping with the bootstrap

Set up build environment

`wiki.linaro.org/Platform/DevPlatform/CrossCompile/arm64bootstrap`

- Fix cross-build failures
- Fix cyclic dependency problems

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

Resources

- `http://wiki.debian.org/Arm64Port`
- `http://people.debian.org/~wookey/bootstrap.html`

Thanks

Wookey
wookey@wookware.org
http://wookware.org

http://wiki.debian.org/Arm64Port

Linaro