The new arm ABI (EABI) and Debian armel port

Wookey
(Balloonz / Toby Churchill Ltd)
wookey@wookware.org

Debconf 2007,
Edinburgh

Intro

- History/Context
- Lots of gory details so you can understand what I'm on about.
- Why things need to change
- What Debian is doing about all this
- Current Status

Intro

- History/Context
- Lots of gory details so you can understand what I'm on about.
- Why things need to change
- What Debian is doing about all this
- Current Status

Intro

- History/Context
- Lots of gory details so you can understand what I'm on about.
- Why things need to change
- What Debian is doing about all this
- Current Status

Intro

- History/Context
- Lots of gory details so you can understand what I'm on about.
- Why things need to change
- What Debian is doing about all this
- Current Status

Intro

- History/Context
- Lots of gory details so you can understand what I'm on about.
- Why things need to change
- What Debian is doing about all this
- Current Status

EABI

- What is ABI?
- Not API
- C function calling convention
- All libraries need to match up
- Kernel syscall convention also changed

History

- ARM kernel port created in 1998
- Used GCC's C calling convention for arm
- Userspace to kernel syscall interface designed to pass 5 or more arguments efficiently (via registers). Similar to RISCOS conventions, without condition codes to indicate errors.
- Floating point was done with FP instruction set. Executed by FPU if present, emulated if not.

Supported machines

Debian-arm port started in 2000

- Netwinder: 2000
- RiscPC, Cats: 2001
- Lart, Bast: 2003
- Iyonix, Manga: 2004
- NSLU2, Thecus: 2005/6

Many others without debian-installer support

Floating Point

ARM FPUs have turned out to be like hens teeth

- CPUs with FPU:
  - ARM FPA11, VLSI 7500FE
- CPUs without FPU:
  - Cirrus Logic 711x, 720T, 72xx, 73xx
  - Dec/Intel SA110, 1100, 1110
  - Intel PXA250, 255, 270, IXP4xx, ixp2000
  - Samsung chips
  - Atmel AT91xx . . .
- New FPUs with different instruction sets:
  - Cirrus EP93xx has Maverick crunch
  - Intel IWMMXT
  - TI Omap 2420 (Arm11 - VFP)
  - Philips lpc3180 (arm9)

FP solutions

- Runtime Emulators (‘hard-float’):
  - Acorn FPE (binary module)
  - NWFPE, FastFPE
- Compile-time functions (‘soft-float’)
  - GCC softfloat

Softfloat and Emulators incompatible due to different calling conventions.

Real FPU/emulation: Calls use r0-r3 for 1st 4 arguments, stack for the rest.
floats can fill multiple registers, and be split across registers and stack.

Return value is put in Coprocessor register f0.

Softfloat: the return value is put in r0-r2 (depending on size)

Debian-arm uses hard-float, because it pre-dates the soft-float concept.

FP solutions

- Runtime Emulators (‘hard-float’):
  - Acorn FPE (binary module)
  - NWFPE, FastFPE
- Compile-time functions (‘soft-float’)
  - GCC softfloat

Softfloat and Emulators incompatible due to different calling conventions.

Real FPU/emulation: Calls use r0-r3 for 1st 4 arguments, stack for the rest.
floats can fill multiple registers, and be split across registers and stack.

Return value is put in Coprocessor register f0.

Softfloat: the return value is put in r0-r2 (depending on size)

Debian-arm uses hard-float, because it pre-dates the soft-float concept.

FP formats

Interesting wrinkle
Mixed endian (on LE arm). IEEE754 but is still rather 'weird'.
Pi in double format looks like this:

```
x86:   18 2d 44 54 fb 21 09 40 (little endian)
Sparc: 40 09 21 fb 54 44 2d 18 (big endian)
arm:   fb 21 09 40 18 2d 44 54 (mixed endian)
```

- BE arm is BE+BE
- Debian arm is LE
- Issue in mozilla, perl, gnumeric, maths libs, java, ...
- This 'feature' goes away with EABI.

Main EABI changes

- Structure packing
  - Old ABI had minimum structure packing size of 4 bytes
  - EABI has no minimum - packing determined by type sizes
- Argument alignment
  - 8-byte stack alignment at public function entry points (Old ABI was 4-bytes)
  - 64-bit data types (e.g. long long) are 8-byte aligned (Old ABI was 4-bytes)
- Enums
  - EABI allows enums to have variable type size (-mabi=aapcs)
  - Not used on GNU/Linux - they remain as 4-bytes. (-mabi=aapcs-linux)
- Floating point
  - Mixed-endian LE format goes away
  - Can mix GCC softfloat and FPU hardfloat/emulation

Why do we care?

Pros

- Most arm wierdness removed (FP formats, packing, C++ exceptions)
- Hard/soft float interworking (soft-float *much* faster)
- Standardisation across toolchains, debuggers
- Thumb interworking
- Interchangeable binaries (PalmOS, GNU/Linux, Symbian OS)
- More efficient syscall convention

Cons

- Almost total incompatibility with existing port

New kernel syscall convention

Example: long ftruncate64(unsigned int fd, loff_t length) (syscall number 194):

- legacy ABI:
  - - put fd into r0
  - - put length into r1-r2
  - - use "`swi #(0x900000 + 194)`" to call the kernel
- Better on von Neuman architecture - already in cache
- EABI:
  - - put fd into r0
  - - put length into r2-r3 (skipping over r1)
  - - put 194 into r7
  - - use "`swi 0`" to call the kernel
- Better on Harvard architecture - doesn't pollute data cache

New syscalls (2)

- Changed in kernel 2.6.15 - mainline 2.6.16
- Kernel supports old syscalls (no speed gain)
- glibc 2.3.6 used old syscalls - removed in 2.4

EABI genesis

- EABI published Dec 2003, based on:
  - external stuff:
    - ELF
    - DWARF-3
    - generic C++ ABI
  - internal stuff:
    - Procedure call standard (AAPCS = simplified, clarified ATPCS)
    - ELF processor supplement
    - anticipates thumb-2 and arm v6 (new-style BE8)
  - plus new stuff:
    - C++ exception handling, C-library, run-time helper functions

Timeline

- Code sourcery 1st cross-tools q3 2005 GCC v3.4.4
- 2005: Early Linux adopters (montavista, nokia) - shimmed glibc
- Kernel syscalls changed during 2.6.15 (early 2006)
- Debian port started q1 2006 - all new
- Aleph One and Code sourcery gcc4.1 cross-tools q1 2006
- Angstrom OE EABI Aug 2006
- ADS/Lennert Buytenhek working port Jan 2007 (v4t build)
- DD-signed (Riku Voipio) buildd announced April 2007 (v4t build)

Tools

- GCC
  - EABI support originally added to gcc3.4.4 (with -mabi=aapcs-linux)
  - From 4.1: different arch
    Old ABI is called linux-arm-none-gnu
    EABI is called linux-arm-none-gnueabi
- glibc
  - shims in 2.3.6
  - new syscalls in 2.4 and 2.3.7
  - 2.5 works *and* has new syscalls
- Kernel
  - support from 2.6.16
- Crosstool 0.42 or later

CPU choice

Arm instruction set versions:

- v3 (RiscPC)
- v4 (Strongarm)
- v4t (most arm 7)
- v5 (xscale, arm9, 10, 11)
- v6 coming soon - new BE8 mode

Issues:

- Thumb interworking (BX, LDM, etc)
- GCC only supports EABI on v4t or later

Thumb interworking

- Thumb is 16-bit opcode set (normal opcodes are 32-bit)
- Reduced code size (30% smaller) - popular for phones
- EABI allows mixing at function level granularity
  - Needs ARM v4t due to use of BX to set state
  - On arm v5 LDR/LDM does this
  - v4 doesn't have BX. StrongARM is v4. Will abort. Need to use this instead:

```
tst lr, #1
moveq pc, lr
bx lr
```

  - Patch to gcc to support this now exists (Richard K. Pixley).
- Question whether to support v4 (strongarm) with armel, or start at v4t
- If speed/size overhead is not large then we probably should.

Debian port

Worth changing to:

- Avoid obsolescence
- Fix the FP problem
- Build stuff that never worked

Lesser gains:

- Binary compatibility (can use commerical debuggers)

Incompatibility with existing port a problem.
How to make the change?...

Rename all library packages

Pros

- Can do apt-get dist-upgrade

Cons

- Every single library package needs to be renamed
- Will take a long time, during which unstable will be broken for all arches (6months for C++, so 2yrs?)
- Not popular due to large hassle for other arches
- Will lose v3, (and maybe v4) support.

New architecture

Pros

- Fits with gcc approach
- Does not affect non-arm arches
- Can keep 'arm' for v3 (and maybe v4) machines
- Can be done relatively quickly as no interaction with other arches/releases

Cons

- Current arm users don't have easy upgrade path
- Need archive space for new arch

ABI: field in control file

Suggested as part of multiarch proposal
Pros

- Reflects ABI correctly, would help other transitions too
- Technically best?

Cons

- No existing implementation
- No consensus on including it yet
- Questions over resolving dependencies and how it fits into archive
- Will lose v3 (and maybe v4) support

Conflicting libc packages

Make a libc6-eabi-dev depending on eabi and ld-linux.so.3, that conflicts with libc6.

Pros

- Only have to change glibc (and rebuild everything)
- Does not affect other arches

Cons

- Most of port will be uninstallable for a very long time
- apt-get dist-upgrade still won't work due to huge number of conflicts
- Will lose v3, (and maybe v4) support

‘New Arch’ won

- ‘New architecture’ won.
- Called armel
- Could also have armeb

Status and Remaining Issues

- Port now working and useable - 74% built yesterday
- 2 Buildds working (thecus n2100) - unofficial
- Need to qualify for Lenny (95%, etc)
- v4 support?
- How long to keep arm going?
- Transitions from arm->armel need support/testing