ARM ports status

Debconf 22

Wookey

Agenda

- 64 bit port
  - New architectural features
  - ABI breaks
- 32 bit ports
  - 64-bit time_t
  - armel lifetime
- Buildds

Overview

- Things are good
- ARM is not a backwater any more
- Just about everything ported and working
- Software parity getting close

Hardware Availability

- Cloud hardware: AWS Graviton2,3
- More hardware available:
  - Servers (Emag, Altra)
  - dev boards (Rpi4, Rpi2Bv2, PineH64)
  - Phones (mainline) (imx8, snapdragon)
- There are even some laptops
  (based on M1, Snapdragon)

X86 parity

- Arm64 strives for parity
- Debian CI and reproducible builds fixed
- Mostly achieved but there are gaps
  - Cloud ‘Plain VM’ image missing
  - Upstream optimisations

New features since 8.0

- 8.1 CRC32, PAN (Privileged Access Never), LSE (Large System Extension) atomics
- 8.2 SVE (Scalable Vector Extension), FP16/FPHP (Half-precision Floating Point)
- 8.3 PAC (Pointer Authentication)
- 8.4 Nested virtualisation
- 8.5 BTI (Branch Target Identification), MTE (Memory Tagging Extension)
- Really useful page: https://en.opensuse.org/Arm_architecture_support

8.1 & 8.2 features

PAN (memory protection for EL0, enabled, used if present)

- Optional instructions done with checks or ifuncs
  - (CRC32)
- LSE atomics gated with HWCAP check (works because LSE is slow). On by default in gcc10.
- Some things need specific upstream support
  - SVE (variable length vectors),
  - FP16 (good for ML)

Virtualisation

- arm64 can’t do nested virtualisation before 8.3/8.4.
- amd64 always could
- Also got HyperV support in kernel 5.15

PAC & BTI

- Instructions in <8.3 NOP space for ABI compatibility
- PAC (v8.3) tags/signs pointers, protects against ‘Return Oriented Programming’ attacks.
- BTI (v8.5) declares permitted ‘jump landings’. Protects against ‘Jump oriented Programming attacks.
  - (same as -fcf-protection on x86)
- PAC has kernel and userspace parts.
- PAC could do much more, but not ABI-compatibly

Enabling PAC+BTI

- Go together with -mbranch-protection=standard
- GCC 10
- I (Lucas) did an archive rebuild – 12 packages FTBFS
- Considered safe (userspace part)
  - Did break some DRM
  - Kernel switch to disable at runtime
- Should be enabled by default

Default flags

- Set by both gcc and dpkg-buildflags
- When should we use which?
- Neon example (armv7-a+nosimd+fp)
- PAC+BTI (-fbranch_protection=standard) – which should it use?
- dpkg-buildflags: hardening=branch, enables:
  - -fbranch_protection=standard on arm64
  - -fcf-protection on amd64

MTE

- Memory Tagging Extensions
- colouring of pointers and memory (4-bit tag)
- Heap and stack tagging possible. Heap done now.
- Userspace and kernel parts
  - Userspace part enabled in glibc
- KASAN (Kernel Adress Sanitizer) can use this hardware support. In kernel 5.11

Hardware variants

- Seattle, Ampere emag 8.0
- PineH64, Pi2 modelB A53 8.0
- Pi4, imx8 A72 8.0
- Ampere Altra 8.2
- Graviton2 8.2
- Graviton3 8.4
- Apple M1 8.5 (no BTI)

ABI history

- arm 2000:Potato (OABI v3) (removed 2011)
- armel 2009:Lenny (EABI v5)
- armhf 2012:Wheezy (EABI v7, fp)
- arm64 2013:Jessie (v8.0)
- 64-bit time_t ?
- armv9 ?

ABI break tradeoffs

- ABI breaks are a big deal. ARM understands and talks about ABI now
- Tradeoff between new features and compatibility
  - PAC could be used for data
  - Overhead and extra work to fit into existing ABI
  - New security features being researched
- OS projects decide when, not ARM
- Debian prefers stability/compatibility
- More public discussion would be good – ABI decisions are long-term.
- V9 is coming at some point

Neon is (still) optional

- Instructions not in all hardware
- Harris (im6) has it, abel (Marvell Armada) doesn't
- GCC changed defaults(-mfpu=auto) so -march=armv7-a now implies +simd+fp
- Our baseline is armv7-a+nosimd+fp
- GCC uses simd in early setup, before HWCAP checks
- Multiple bugs (mozilla, valgrind, etc)
- https://bugs.debian.org/cgi-bin/bugreport.cgi?bug=1014091

armhf-64bit time_t

- 2038 approaches
- We missed Debian 11. Is Debian 12 feasible?
- Arnd Bergman tried bootstap March 2020
- Time to try again.
- New arch easier, but needs a name.
  - arm, armt, armhft, arm32? (bikeshed here)

armel

- Still working, but some breakage: some upstreams dropping support: defaulting to v7, simd (neon) etc.
- How much is it still used?
- Is it time to retire to ports?
- Does adding armhf-64bit time_t affect our choice?

Buildds

- 32 bit buildds are Marvell Armada XP dev boards
- Getting old and tired – no replacements in sight
- Most things build on 64-bit hardware+kernel/32-bit userspace, but not everything.
- Some hardware has no 32-bit support (ThunderX/ThunderX2), some
- Hard to know which features a build machine supports

Alignment fixups

- 32 bit kernel fixes up unaligned access (emulate x86 behaviour)
- Added to 64-bit kernel (this month), enabling more builds with 64bit kernel/32bit userspace

armhf kernel on arm64

- 32-bit kernels on 64-bit hardware
- Requested for RPi4 – low-latency usecase
- https://bugs.debian.org/cgi-bin/bugreport.cgi?bug=971059
- We used to do this x86?
- What is the downside?

Want to do this for a living?

- Talk to me