SFO15-211: UEFI on ARM

Dealing with legacy and moving on
Overview

This session is mainly about complaining.

- Part of this session is going to be complaining about ARM.
- Another part is going to be complaining about UEFI (or mainly EDK2).
- Finally, I will finish off with some complaints over the current state of things, and describe how I would like to improve them.
<table>
<thead>
<tr>
<th><strong>Glossary</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>UEFI</strong></td>
</tr>
<tr>
<td><strong>EDK2</strong></td>
</tr>
<tr>
<td><strong>linaro-edk2</strong></td>
</tr>
<tr>
<td><strong>BDS</strong></td>
</tr>
<tr>
<td><strong>OVMF</strong></td>
</tr>
<tr>
<td><strong>ArmVirt</strong></td>
</tr>
</tbody>
</table>
The Linaro “UEFI” team

We are called the UEFI team, but we work on:

<table>
<thead>
<tr>
<th>Work Area</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>EDK2 development</td>
<td>Maintaining ARM bits of Tianocore EDK2.</td>
</tr>
<tr>
<td>Linux kernel</td>
<td>Stub loader, memory map, runtime services.</td>
</tr>
<tr>
<td>UEFI specification</td>
<td>Feeding back and trying to resolve ARM-specific issues.</td>
</tr>
<tr>
<td>Linux userland</td>
<td>acpica-tools, efivars, dmidecode, capsule update</td>
</tr>
<tr>
<td>ACPI/DT Intersections</td>
<td>Console selection, ACPI init.</td>
</tr>
<tr>
<td>“stuff”</td>
<td>GRUB, SMBIOS, ARM Trusted Firmware, LuvOS/FWTS.</td>
</tr>
</tbody>
</table>
Complaining about ARM
UEFI on ARM

Actually, UEFI on ARM was fairly well handled

- Support was added into UEFI 2.3, released in 2009. Has seen updates along the way.

EDK2 on ARM was the problem

- ARM did not (on an organisational level) *understand* UEFI. So it effectively treated EDK2 as just another embedded firmware codebase.
- This led to an unfortunate series of decisions, causing a lot of effort being directed towards creating a U-Boot replacement rather than a UEFI implementation.
ArmBds

I have two major objections regarding ArmBds:

- It is called ArmBds and is ARM-specific. This goes against the entire point of having a cross-platform firmware infrastructure. There *might* have been a need for a very-lightweight line-based generic BDS for embedded/mobile, but there is no need for an architecture-specific one.

- It was done first, instead of enabling support for a BDS already in use first, and then seeing if an alternative was needed.
Built-In Linux Loader (BILL)

UEFI provides a runtime environment

- Its architecture is such that applications can be written to run under it - including operating system loaders.
- Instead of following this model, ARM had implemented special knowledge into ArmBds to recognise a Linux kernel image and load it.
- Alternatively, the GRUB bootloader could have been ported to AArch64 UEFI (ports to x86 UEFI 32/64-bit already existed).
- In December 2011, support was merged in the Linux kernel for a “stub” being added to the x86 bzImage, turning it into a valid UEFI application.
- ARM ports of EDK2 stuck with BILL.
UEFI on AArch64

UEFI on AArch64 was very different from on AArch32

- AArch64 was the first new architecture added to UEFI specification since it stopped being an Intel-internal project
  - More on this in the “complaining about UEFI section”.

EDK2 on AArch64 was done almost as an afterthought

- ARM made public statements pushing UEFI as the One Firmware for AArch64, but did not follow up engineering-wise
  - Public release of EDK2 support lagged almost a year behind public releases of Linux kernel and Foundation model.
  - Still shipped with BILL and ArmBds.
  - No effort was made to assist migration from other firmware.
Complaining about UEFI (mostly EDK2)
EDK2 evolution

The main issues are with EDK2, not UEFI:

- EDK2 is part of Tianocore, which still shows signs of having been an Intel-internal project.
- At the same time, it is an open source project without a benevolent dictator.
- The result is that strategic changes to the project - directory layout, invasive changes to core codebase, and so on tend to “appear” rather than happen as part of public discussion.
- Discussion about structural changes to project (such as moving to Git) tend to just stall.
Platform Support

Platform support and drivers have been mostly missing

- This was a major part of their commercial offering, so why would the BIOS vendors want to share it?
- Implementors would start from an augmented EDK2 and add packages from CPU vendor, chipset vendor, controller vendors, …
- Exceptions: OvmfPkg, OptionRomPkg, BeagleBoardPkg/Omap35xxPkg, ArmPlatformPkg, …

Trying to bring order to this for those who actually want to upstream drivers and platform support has not met with much success.
Oh, look, a new BDS … and another

We recently migrated to IntelBds

- Enabling functionality such as:
  - automatically detecting bootable removable media (OS installers)
  - interface for enrolling keys for UEFI Secure Boot
- After me presenting last year at the UEFI Forum event how important it was that we didn’t have architecture-specific BDSs, but instead worked together on a common generic one…
  - … I found out through patches sent to the EDK2 development mailing list that Intel were deprecating IntelBds, to be replaced by a new BDS called UiApp.
Complaining about everything else
Linaro (yeah, we suck too)

We do not seem to be doing a very good job at helping people do the right thing.

- What little documentation we have in the wiki is spectacularly dated.
- Most new platform ports I come across start by cloning the linaro-edk2 release repository.
- Most new platform ports I come across use the ArmBds (hey, it’s called ArmBds, and it’s in Linaro’s tree).
- We have way too many EDK2 repositories on git.linaro.org - and LEG and ARM landing team have been doing separate binary releases.
  ○ Oh, and there is a separate 96boards EDK2 repository on github.
Please

Never, **ever**, clone linaro-edk2 and use it as the start of a new platform port!
Validation

Because ArmBds was fairly simple to hook into expect-style scripts, the validation jobs were set up doing just that.

- This will be of little use going forward
Bad habits

Some people refer to UEFI, and das U-Boot, as bootloaders - distinguishing them from ARM Trusted Firmware which runs before.

- This is bad, because language guides our thinking, and when it is not firmware it must be software and software can be easily upgraded frequently…
- As a favour to me, can you please slap people making that mistake (lightly) on the wrist?
Fixing it
Linaro

We have restructured how we do releases.

Ryan Harkin did a stellar job setting up a maintainer/release process for Linaro’s EDK2 work...

… but it didn’t scale.

And we learned more about Git :)
Linaro - OpenPlatformPkg

To work around the issues with EDK2 platform/driver support, we created OpenPlatformPkg:

- Non-rebasing git repository held on git.linaro.org
- Contains platform support and drivers
- Imported into linaro-edk2 as a submodule when making releases.
- Presented at UEFI plugfest in Seattle May 2015
- Long-term, I would prefer for it to be part of Tianocore. But I would be happy with any other solution that brings multiple platform and driver support into a common tree.
Traditional UEFI layout

- CPU support
- Chipset/SoC support
- Tianocore EDK2
- Vendor additions
- Platform code & Drivers

UEFI BIOS
EDK2 with OpenPlatformPkg

- UEFI BIOS
  - Tianocore EDK2
  - Vendor additions

OpenPlatformPkg
EDK2

Since earlier this year, myself and Ard Biesheuvel are now maintainers for ARM bits of EDK2.

- Since we work in same team, and always in the open, we can move a lot faster than was previously possible.
- And Laszlo Ersek (Red Hat) is being extremely helpful as the maintainer for the new ArmVirtPkg
- We’re trying to bring some positive influence to the project - all patches are reviewed (in public) before being committed.
ARM

ARM needs to lead by example, and is starting to.

- Participating more actively in SCT discussions (UTWG)
- ARM-internal UEFI development shifting its focus more towards server
- Linaro’s ARM Landing Team is now working more closely with ARM (were being kept a bit at ARM’s length previously)
BDS

We need to make sure the new UiApp BDS runs on ARM - even if it will never be used in a product.

It is of vital importance that we are testing the same things as the people testing x86 platforms.

UiApp does look less messy than IntelBds (which also contained bits related to the “legacy BIOS support” - CSM).
Kill BILL

BILL is now a standalone UEFI application rather than integrated into ArmBds

- This covers most of the pre-existing use-cases (mainly ARMv7-A platforms and existing validation jobs).
- Any AArch64 platform using a 3.16+ Linux kernel should just use the kernel stub loader instead.
UEFI-tools

https://git.linaro.org/uefi/uefi-tools.git

A set of scripts put together for use in the automated builds, but also handy for development use.

- Picks the correct EDK2 toolchain profile based on compiler version
  - Automatically configures cross compilation where needed
- Reads build-time options from a configuration file
- Also builds ARM Trusted Firmware and wraps UEFI image in .fip
  - (Thanks to Victor Chong) also builds and wraps OP-TEE
- Parallellizes the build across the number of available cores
- Supports GCC and CLANG
Validation

AArch64 is now a fully supported platform in Intel’s LuvOS.

- But LuvOS does not slot well into existing LAVA setup.
- Work required, and likely more than the LAVA team can carry on their own.

Need to look into testsuites for other components

- GRUB
- efivars
Userland

We are still working in an ecosystem that managed to get used to a single, de-facto standard, hw platform

- Need to strip away users of /dev/mem
  - dmidecode and acpidump now both fixed upstream, using proper kernel interfaces instead
- Still find the occasional tool expecting signed chars or otherwise breaking on non-x86
Documentation

UEFI documentation on wiki.linaro.org is in bad need of being overhauled.

We did a decent job initially documenting what we were doing, but at the time we were working around a lot of incompleteness.

Reality is a lot more palatable these days, but the wiki still suggests differently,
You

Come talk to us

- linaro-uefi@lists.linaro.org
- #linaro-enterprise on Freenode
- edk2-devel@lists.01.org