

Android Bootloader Consolidation?

Sam Protsenko <semen.protsenko@linaro.org>

John Stultz <john.stultz@linaro.org>



Premise

- Almost every release, the Android platform changes the bootloader/OS interaction
- Each SoC vendor has its own bootloader
 - Each SoC vendor has to implement the new Android requirements each release
 - Non-trivial amount of work
 - **Slows rolling out new Android releases**
- **No one buys a phone based on the bootloader**
- There is little value in “differentiation” of Android bootloader functionality
- There is risk: Security issues
- There is cost: More complex support matrix
 - Different bootloader versions for one SoC (Android 9 vs Android 10)
- **It's not great for the ecosystem**

Android Specific Functionality

- [abootimg](#) support (v1/v2/v3!)
- [DTBO partition handling](#)
- Reboot reason (kernel -> bootloader)
- [Boot reason](#) (bootloader -> kernel)
- Native Fastboot
- Boot to recovery / userland [fastbootd](#)
- [A/B boot](#) & rollback
- [Verified boot](#) (AVB v1/v2/?)

Sam covered more detail in this talk:

<https://connect.linaro.org/resources/san19/san19-217/>

Consolidation would help

Goal: Lower barriers and let vendors enable their hardware faster

- Let vendors focus on what really makes their SoC/Device different
- Instead of whatever Android's new requirements of the year are

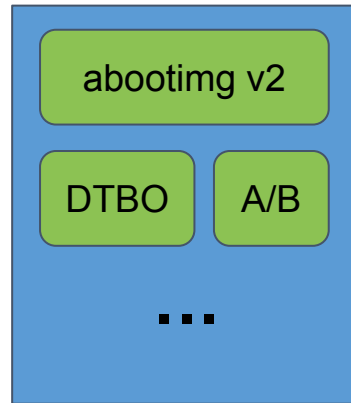
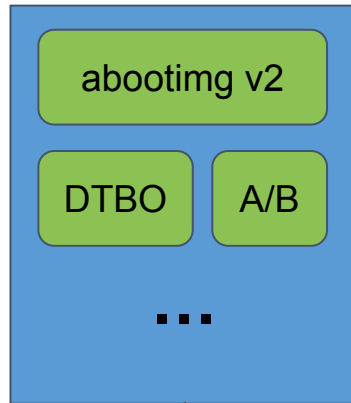
Solutions/Approaches

- Standardization: Move platforms to a common bootloader implementation?
 - Extending [EBBR](#) ?
 - Align with ChromeOS/SBBA/other platforms?
 - FIT image adoption?
- Partition: Let Google own & implement the required Android bits?
 - (Treble/GSI for bootloaders?)
 - Android UEFI app that handles everything?
 - Android mega-library that bootloaders link-in
 - Like [libaby](#), but comprehensive
 - Google is already working on U-Boot's [boot_android](#) cmd; extract to lib?
 - Dynamically linked in? (independently updatable?)
 - Testing (VTS), licensing, versioning?

Case study: Android 10 new features in BL

Vendor 1 bootloader

Vendor 2 bootloader



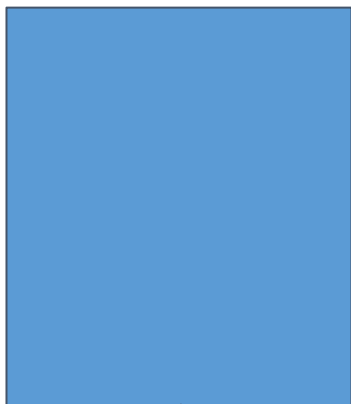
...



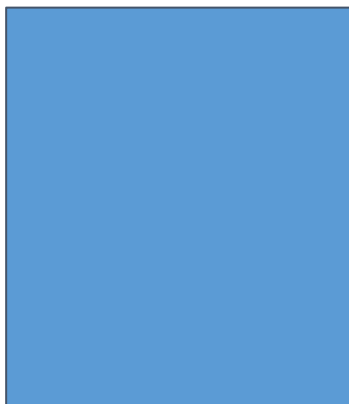
libavb

Proposal: mega-lib

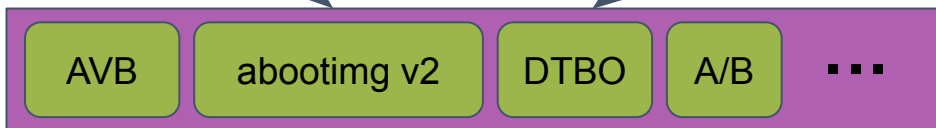
Vendor 1 bootloader



Vendor 2 bootloader



...



libandroidbl

libandroidbl: Approx Architecture

API (all-in-one):

- Boot Android (multiple versions?)

API (separate features):

- BCB
- A/B
- AVB (merge the libavb)
- booting support (v1/v2/v3)
- DTBO image format

(some stuff can be optional)

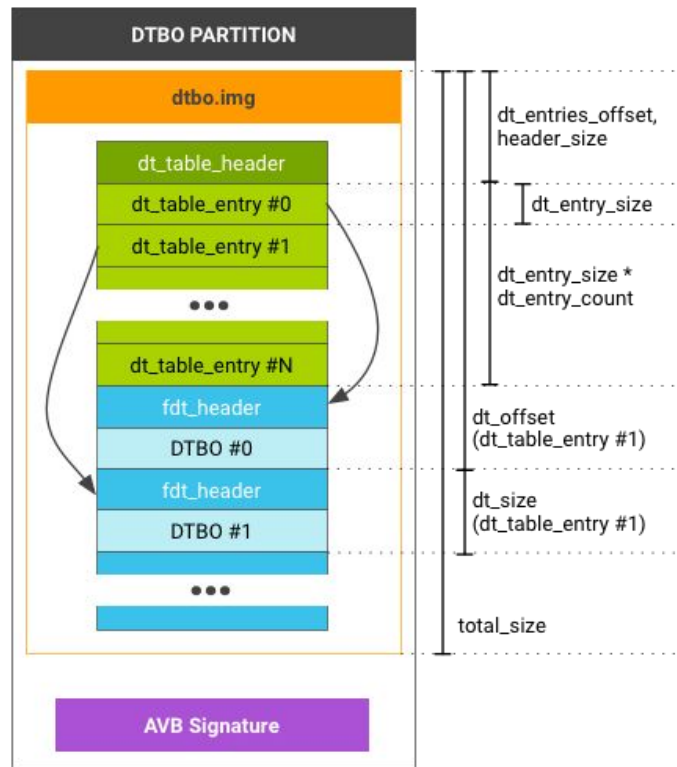
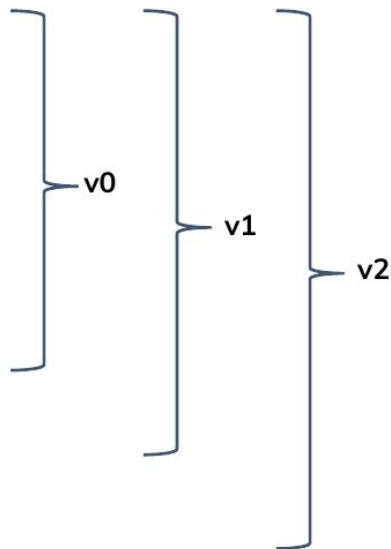
```
struct hal {
    int (*read_from_partition)(...);
    int (*write_to_partition)(...);
    ...
};

void boot_android(struct hal *hal,
                  int ver);
```

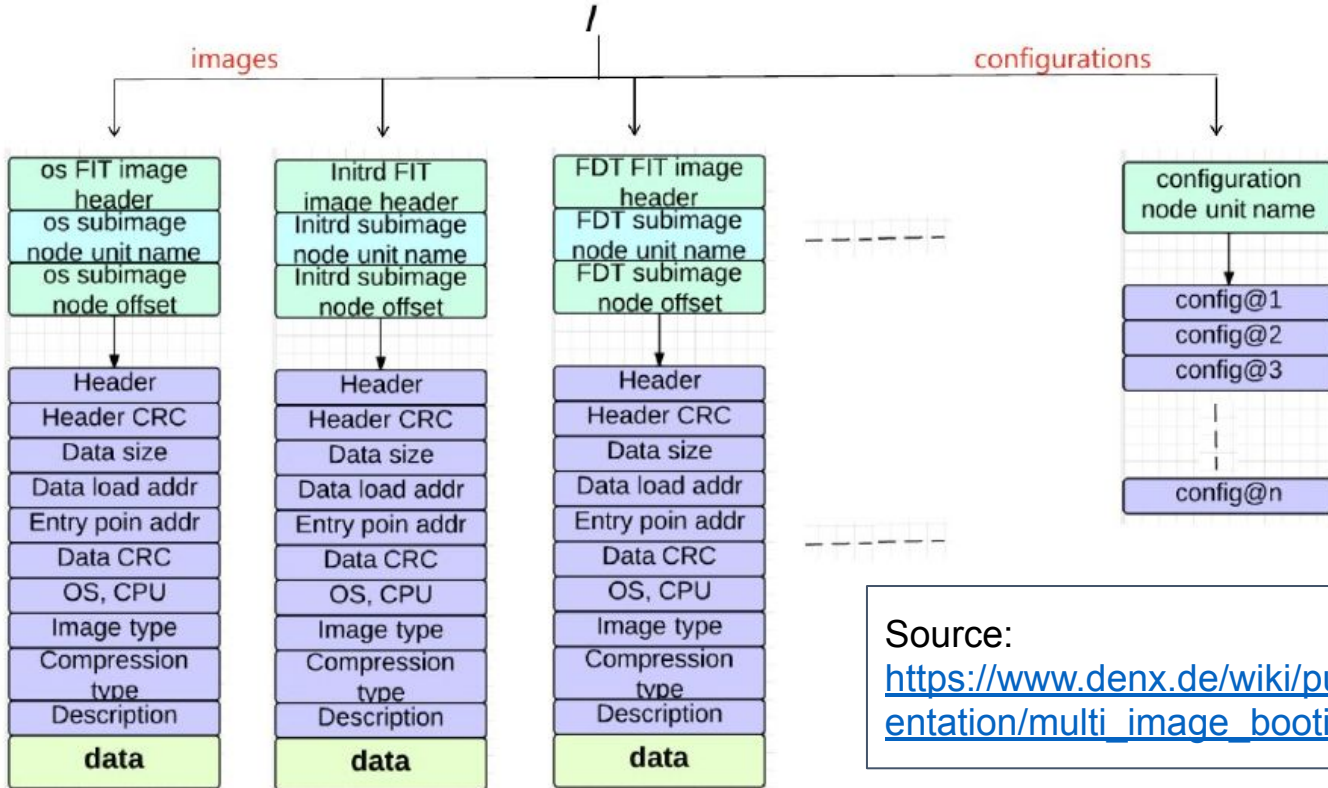
...Just fill in the structure and run!

Case study: Android images formats

+-----+	
boot header	1 page
+-----+	
kernel	n pages
+-----+	
ramdisk	m pages
+-----+	
second stage	o pages
+-----+	
recovery dtbo/acpio	p pages
+-----+	
dtb	q pages
+-----+	



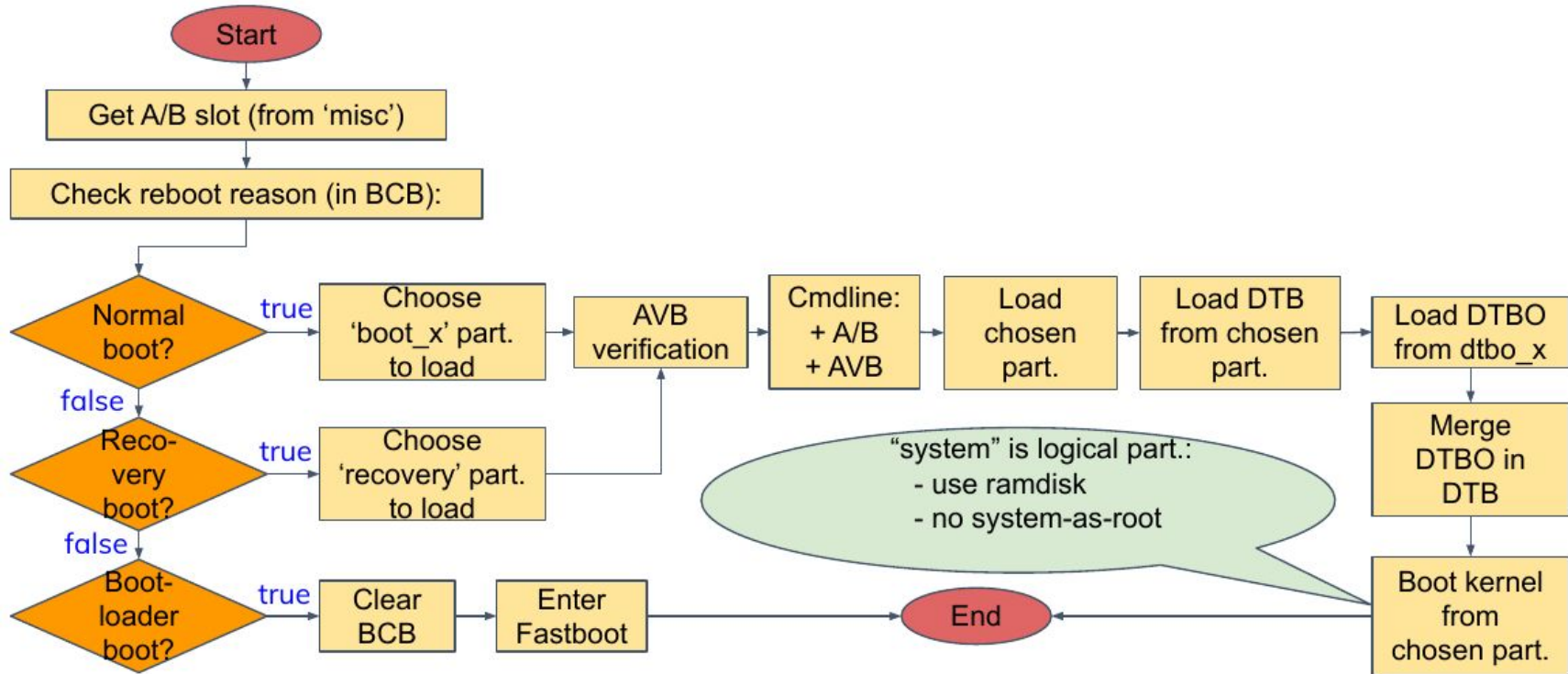
Proposal: Use U-Boot's FIT Image



Backup slides



Android 10 Boot Flow



Android Boot Image Format: v3

Boot Image v3 format:

```
+-----+
| boot header      | 1 page
+-----+
| kernel          | m pages
+-----+
| ramdisk         | n pages
+-----+
```

Vendor Boot Image format:

```
+-----+
| vendor boot header | 1 page
+-----+
| vendor ramdisk     | o pages
+-----+
| dtb                | p pages
+-----+
```

U-Boot's FIT Image: Source (.its)

```
images {
    kernel@1 {
        description = "TI kernel";
        data = /incbin/("zImage");
        type = "kernel";
        arch = "arm";
        os = "linux";
        load = <0x82000000>;
        entry = <0x82000000>;
    };
    ramdisk@1 { ... };
    fdt@1 { ... };
    ...
};
```

```
configurations {
    default = "am57xx-beagle-x15-revc.dtb";
    am57xx-beagle-x15-revc.dtb {
        description = "AM57xx beagle-x15 C";
        kernel = "kernel@1";
        ramdisk = "ramdisk@1";
        fdt = "fdt@1";
    };
    am57xx-evm-reva3.dtb {
        description = "AM57xx EVM rev A3";
        kernel = "kernel@1";
        ramdisk = "ramdisk@1";
        fdt = "fdt@1", "fdt@2", "fdt@3";
    };
};
```