Optimizing SoC Boot flow using Open source Firmware - *status & challenges*

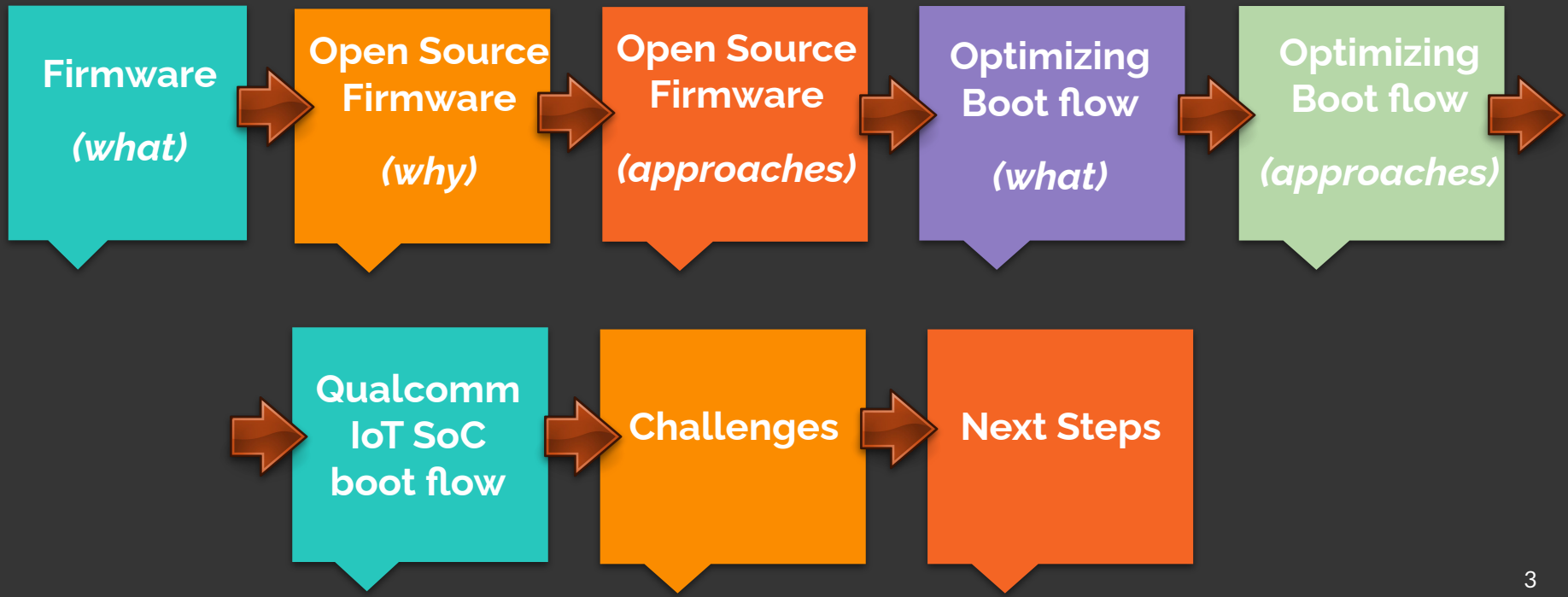
Bhupesh Sharma

 *@bhupesh_sharma*

$ whoami

- Currently @ *ARM*
- Been hacking on boot loaders & kernel since past 18 years.
- Contribute to:
  - Linux,
  - EFI/u-boot bootloader & Secure FW
- User-space utilities like:
  - kexec-tools, and
  - Makedumpfile.
- Co-maintainer hat(s):
  - U-boot UFS, Qcom Ethernet & crash-utility tool.
- Area of Interest: *0-day Silicon bringup* using open-source software.

Outline


Firmware
(what)
Open Source Firmware
(why)
Open Source Firmware
(approaches)
Optimizing Boot flow
(what)
Optimizing Boot flow
(approaches)
Qualcomm IoT SoC boot flow
Challenges
Next Steps

Firmware (*what*)

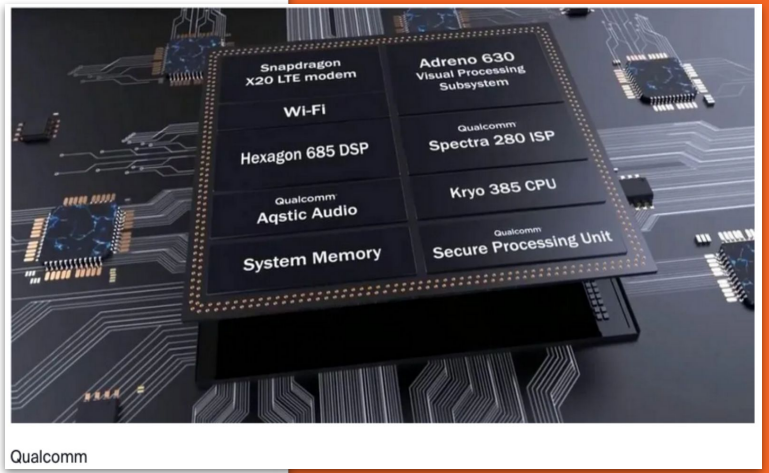
- software embedded in hardware,
- provides low-level instructions for a device to communicate with other devices or perform some basic tasks.
- without firmware, even the most basic device won't function.

Firmware (*what*)

- Various types of firmware are used in smart devices these days.
- Examples:
  - IoT devices,
  - mobile,
  - edge routers,
  - servers,

Firmware (*SoC-wide perspective*)

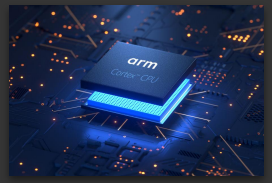
- Let's take an example of a *typical* Qualcomm (ARM based) SoC.
- Several types of firmwares involved:
  - security management,
  - power management,
  - boot flow related,
  - boot flow optimization,
  - peripheral related:
    - pcie, hdd, gps receiver, wifi, etc.



Qualcomm

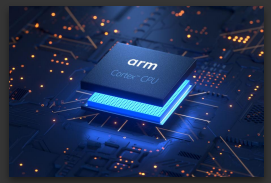
Reference: Arm® Confidential Compute Arch

Firmware (*SoC-wide perspective*)



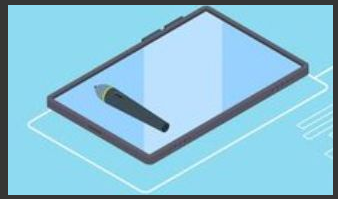
Application Processor (AP)

System Control Processor (SCP)



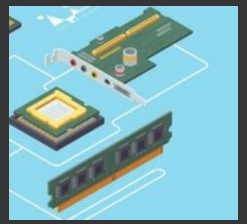
Manageability Control Processor (MCP)

HDD



HDMI Display

Compute Subsystem

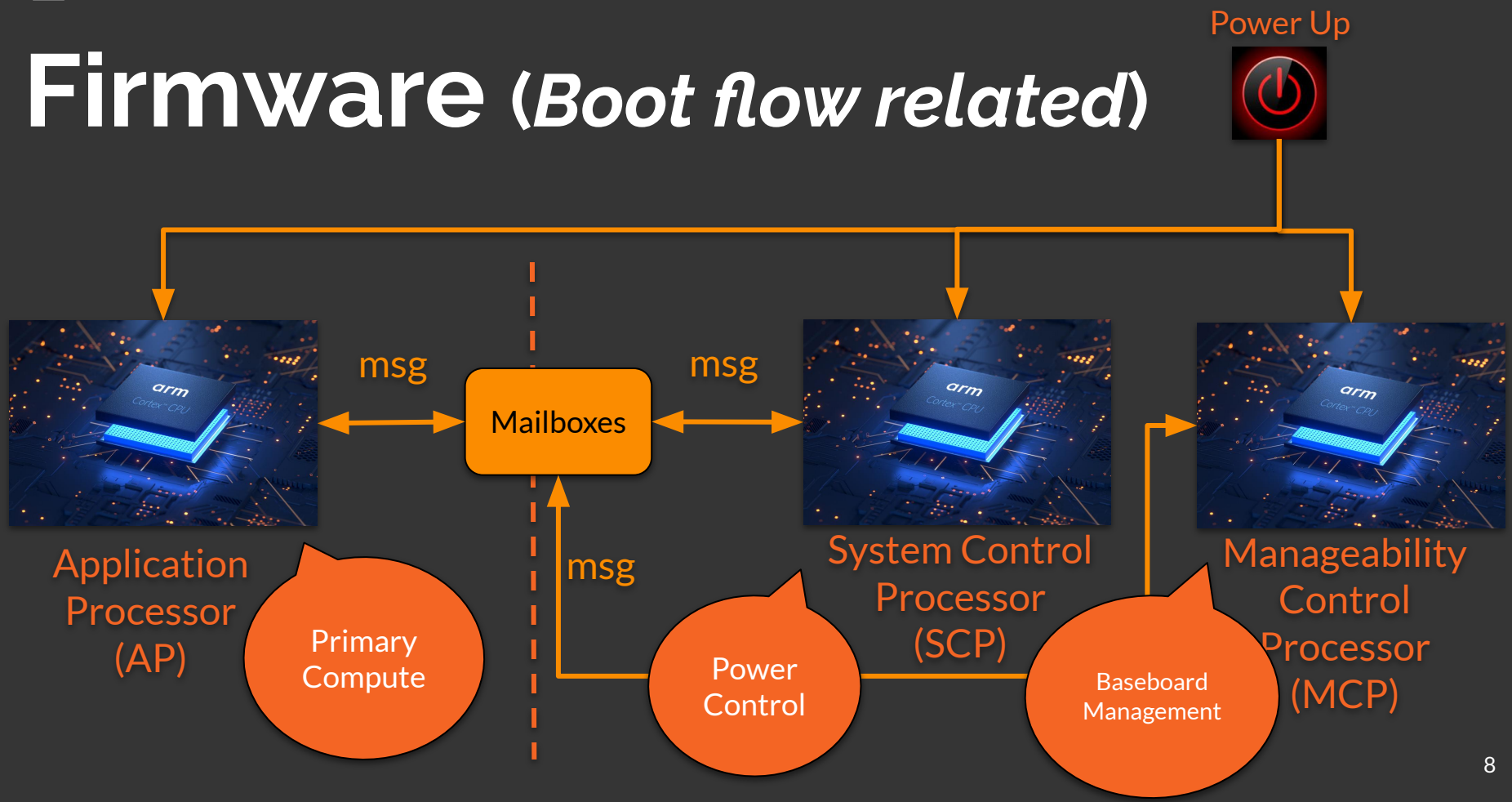


PCIe cards / WIFI adapter

Peripheral I/O Subsystem

Reference: Arm® Confidential Compute Arch

Firmware (*Boot flow related*)

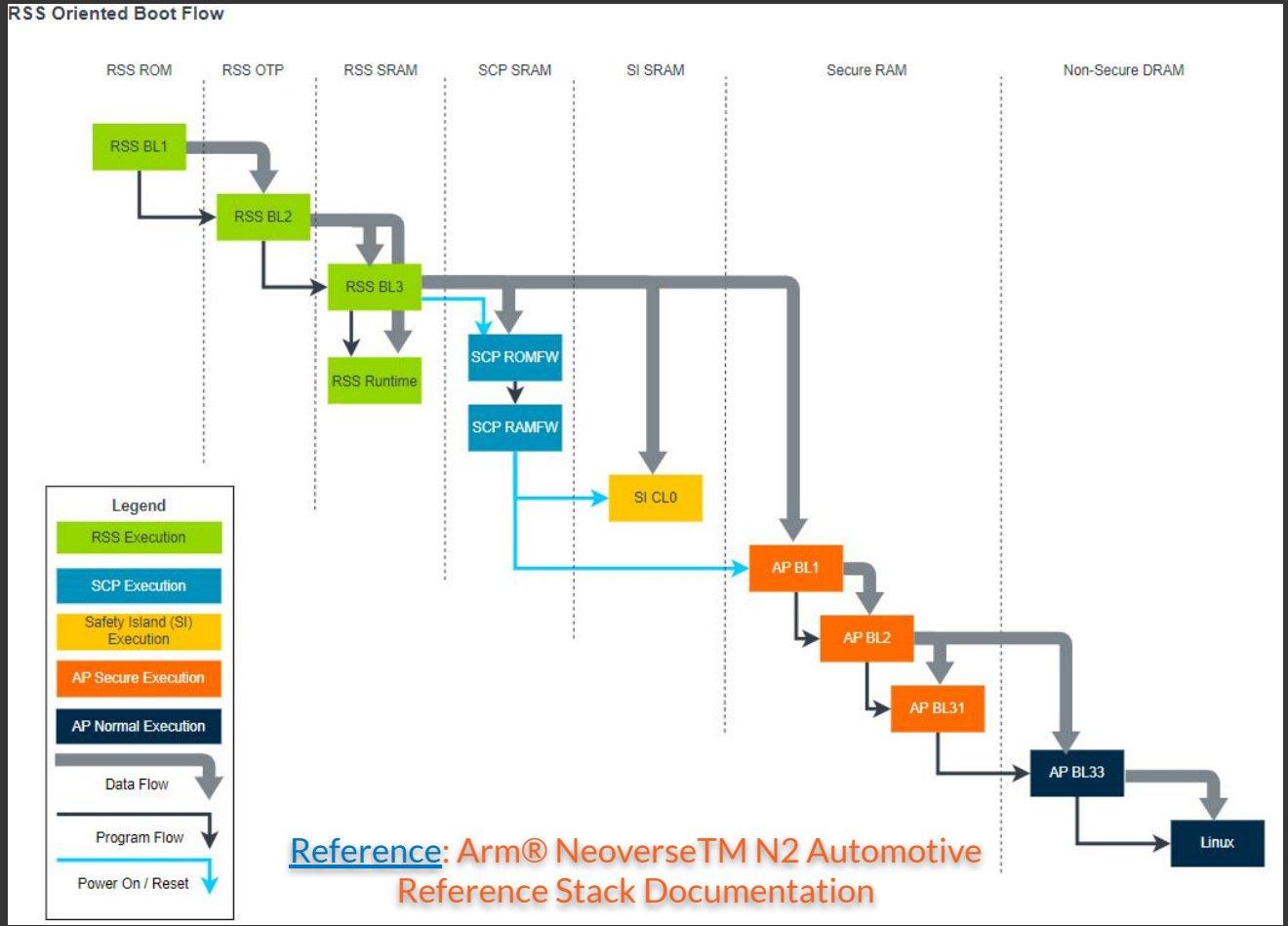


—

- Root of trust / Security,
- Power Management,
- BMC,
- Bootloader,
- Operating system loader

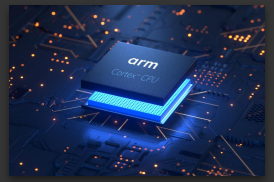
Features 🔺

Complexity 🔺

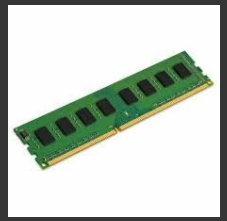


[Reference](): Arm® NeoverseTM N2 Automotive Reference Stack Documentation

Firmware (*Design Considerations*)

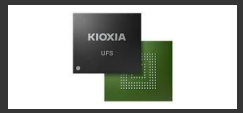


Processor Type



RAM Size

Peripherals



Flash Size

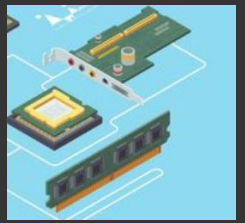
HW considerations

- Power / Battery Life
- Root of Trust
- Security
- Connectivity Options
- Encryption

Ease of use

Bare Metal / Operating System / RTOS

Firmware Update



Firmware footprint

SW considerations

Firmware (*recap..*)

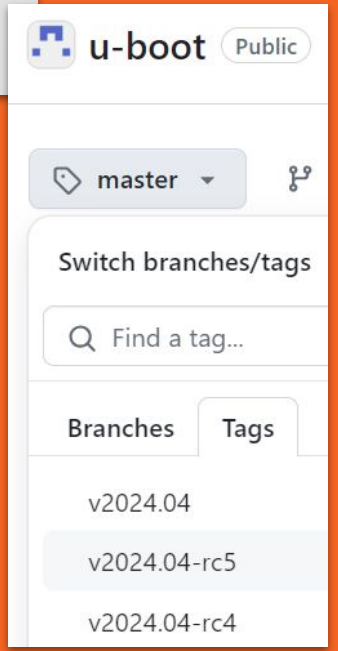
- Firmware *a.k.a* embedded software - used to initialize & **setup** smart devices around us.

- **Key** design aspects for firmware:
  - System *security*,
  - Secure FW *updates*,
  - *Compliance* with specifications,
  - Available *open-source* options.

Open Source Firmware (*why*)

- No need to reinvent the wheel.

- Solid code review by maintainers & contributors.
- Release version management.

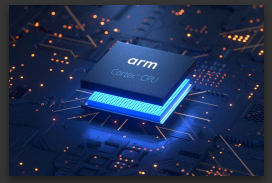


- Adherence to specifications - example UEFI EDK2
- Interoperability checks.

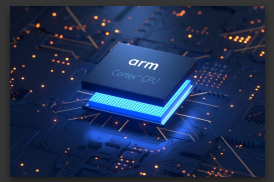
- Easy to setup CI tests.

Reference: Arm® Confidential Compute Arch

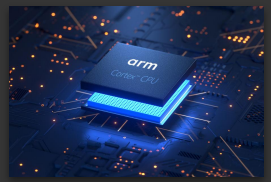
Open Source Firmware (*approaches*)



Application Processor (AP)



System Control Processor (SCP)



Manageability Control Processor (MCP)

Compute Subsystem

- Root of trust / Security,
- Power Management,
- BMC,
- Bootloader,
- Operating system loader

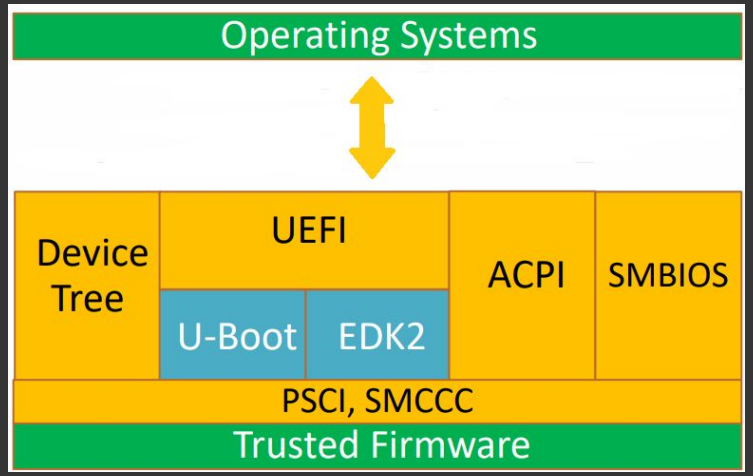
Open Source Firmware (*approaches*)

Application Processor (AP)

System Control Processor (SCP)

Manageability Control Processor (MCP)

Compute Subsystem

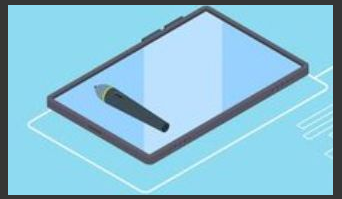


Reference: Arm® Confidential Compute Arch

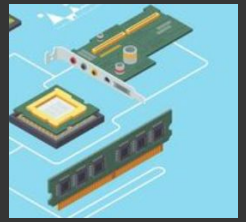
Open Source Firmware (*approaches*)

- Linux Vendor Firmware Service (LVFS)
  - secure portal which allows hardware vendors to upload firmware updates
- Linux firmware
  - *package* distributed alongside Linux kernel,
  - contains firmware *binary blobs* necessary for functionality of hardware devices.

HDD



HDMI Display



PCIe cards / WIFI adapter

Peripheral I/O Subsystem

Open Source Firmware *(available implementations)*

TrustedFirmware	• Open source for Secure World firmware
TianoCore / EDK2	• Open source for UEFI, ACPI, SMBIOS standard system firmware
U-Boot	• Open source for embedded systems firmware
LinuxBoot	• Open source for cloud providers Linux-based firmware
OpenBMC	• Open source BMC firmware

Available open-source FW implementations for ARM cores

Optimizing Boot flow (*what*)

- Boot-to-prompt / userspace (*KPI*):
  - critical for IoT and hand-held devices,

- Secondary compute cores sit idle while the boot firmware runs on the primary application core.
  - Boot timings can be optimized using available multiple-cores in the SoC,

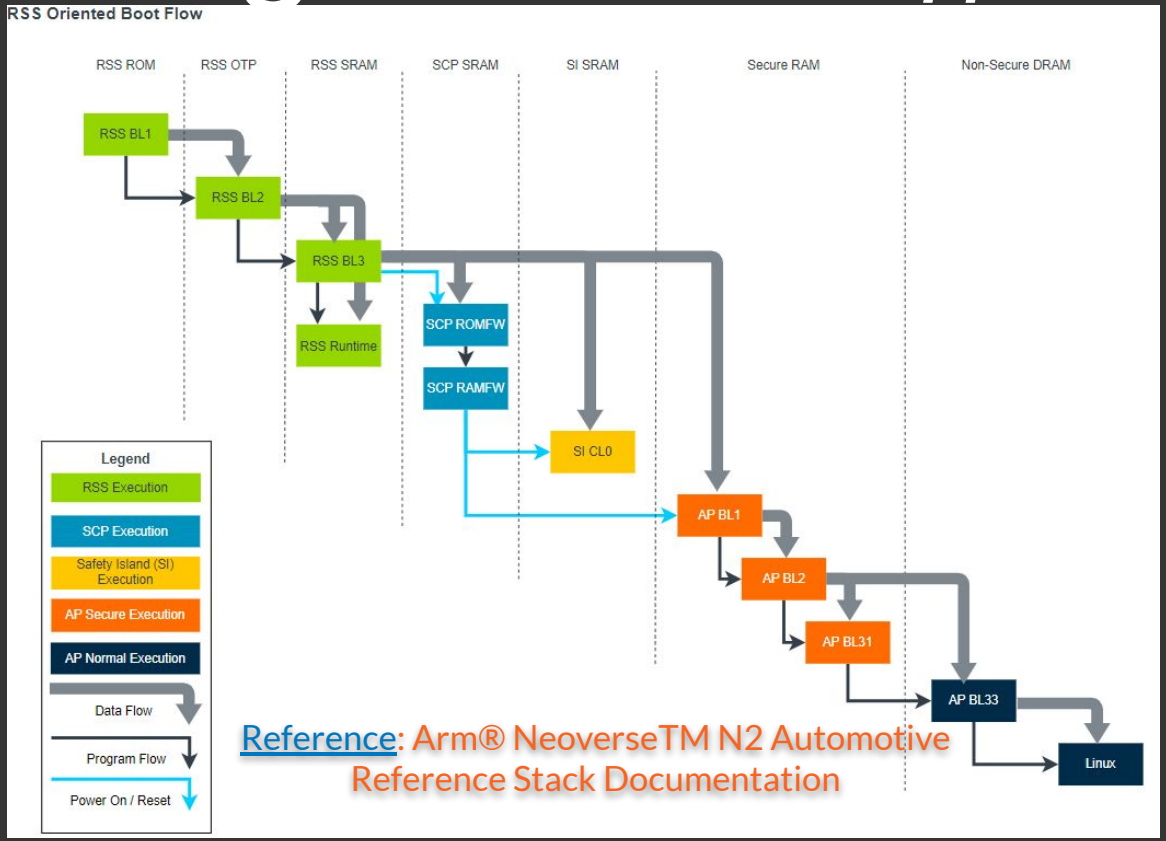
- Handing over security, power management etc to dedicated co-processors.boot flow optimization,

source

Optimizing Android boot time

Optimizing Linux Boot time

Optimizing boot flow (*approaches*)

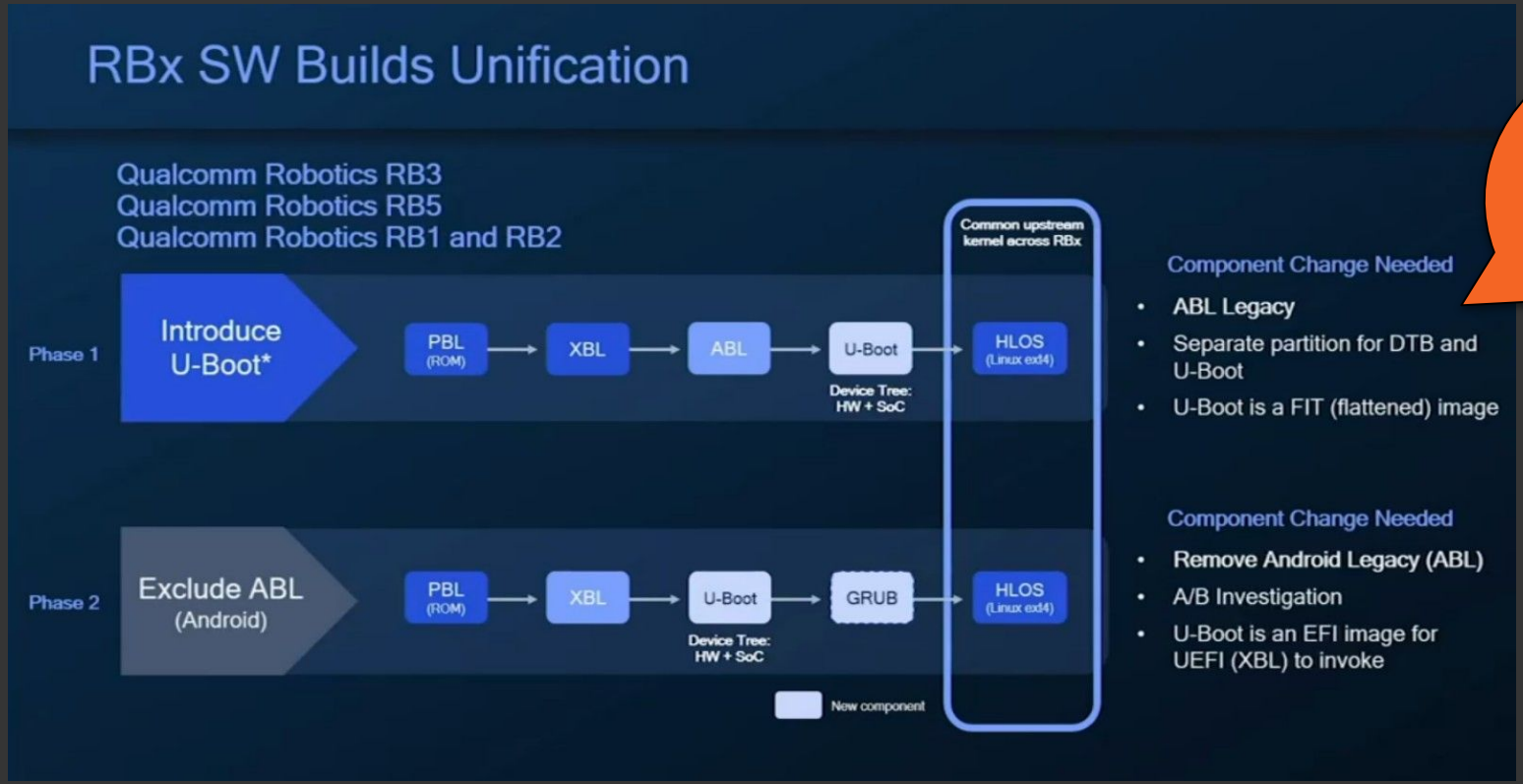


Reference: Arm® NeoverseTM N2 Automotive Reference Stack Documentation

Qualcomm IoT boot flow (*optimization*)

- I proposed an approach for optimizing boot flow on Qualcomm SoCs by replacing ABL by u-boot.
- Unified Boot on Qualcomm RB1 & RB2 boards - Demo.

Qualcomm IoT boot flow (*Optimization*)



Qualcomm Unified Boot Flow Strategy [Keynote]

Optimizing boot flow *(recap..)*

- by removing proprietary & redundant firmware layers (e.g. ABL for Qualcomm IoT platforms).
- utilizing *multiple-cores* available to achieve *parallelization*.
- *depends* on underlying use-case & end-design.

- ***Standardization***:
  - Specifications,
  - Interoperability checks.

- One size ***doesn't*** fit all:
  - End use-case,
  - Available RAM, flash resources.

- ***Open source*** first advocacy.

Challenges

- Use ***virtualized*** test platforms:
  - Qemu,
  - ARM fast models,
  - ARM FVP models.
- Use ***low-cost*** test platforms:
  - UEFI showcase on RPI4
- ***Open source*** first:
  - Use mailing lists & discussion forums.

Next Steps

- Boot Flow for RD-Fremont platforms.
- Boot Flow for Neoverse N2 automotive platform.
- UEFI plugfest talk.
- Repos:.
  - https://trustedfirmware.org/
  - https://www.tianocore.org/
  - https://www.denx.de/wiki/U-Boot

References

Questions?

Slides can be found on github

@bhupesh_sharma