Bootloaders in Limbo
Between Platform Initialization and Operating System

Daniel Maslowski
Hello, I am Daniel :-) 

Work and education
- IT security and computer science
- software engineer
- infrastructure and web
- apps, UIs, ecommerce

Open Source contributions
- hardware and firmware
- operating systems
- software distributions
- reverse engineering
Hello, I am Daniel :-)  

Work and education
- IT security and computer science
- software engineer
- infrastructure and web
- apps, UIs, ecommerce

Open Source contributions
- hardware and firmware
- operating systems
- software distributions
- reverse engineering

I joined RISC-V International as an Individual Member.
Agenda

- Bootloader = Business
  - Satisfying Customers
  - Scopes and Goals
  - Classification
- Projects and Stacks
  - Protocols, Interfaces and Features
  - Platforms, Ports and Flows
  - OS integration
- Success Stories
  - Case Studies
  - Saving Costs
Bootloader = Business
Elevator Pitch

Fast, convenient, safe and secure systems sell best. Choose your components wisely.
Elevator Pitch

Fast, convenient, safe and secure systems sell best.
Elevator Pitch

Fast, convenient, safe and secure systems sell best.

Choose your components wisely.
What is a Bootloader again?
What is a Bootloader again?

Platform Initialization aka firmware
- SoC
- clocks
- GPIOs
- DRAM controller

Bootloader today’s topic
- needs flexibility
- fetches OS kernel
- checks for integrity
- maybe interactive menu

Operating System
- Linux
- FreeBSD
- Plan 9
- Oberon
- Haiku
- …
Common Bootloader Functionalities

A bootloader is an application that loads and executes another application. 

- Target application may rely on a specific protocol often configurable via files or customizable at build time.
- Can offer an interactive menu, e.g., for switching OSs.
- GSoC 2023 project: porting GRUB to coreboot for RISC-V.
- Only UEFI at this point.

Image source: https://github.com/hartwork/grub2-theme-preview
Common Bootloader Functionalities

A bootloader is an application that loads and executes another application.
A bootloader is an application that loads and executes another application.

- Target application may rely on a specific protocol
- Often configurable via files or customizable at build time
- Can offer an interactive menu, e.g. for switching OSs
- GSoC 2023 project: porting GRUB to coreboot for RISC-V
  - Only UEFI at this point

Image source: https://github.com/hartwork/grub2-theme-preview
Satisfying Customers
Watching the Demand

Overtime, requirements develop and change. Ownership and control are important as people have different needs regarding:

- How systems boot
- Integrating OSes
- Platform security

Enterprisers need to fully control their machines.

Watching the Demand

Over time, requirements develop and change.

---

Watching the Demand

Over time, requirements develop and change.

Ownership and Control

People have different needs regarding
- how systems boot
- integrating OSes
- platform security

Enterprise customers need to fully control their machines\(^1\).

\(^1\)https://eclypsium.com/blog/supply-chain-risk-from-gigabyte-app-center-backdoor/
Rising Interest in oreboot and LinuxBoot

[Link to Reddit discussion]

[Link to RVSpace discussion]

Someone from Intel contacted me, evaluating oreboot and LinuxBoot on RISC-V.

ByteDance and many hyperscalers are using LinuxBoot.

[YouTube video link]
Rising Interest in oreboot and LinuxBoot

https://www.reddit.com/r/RISCV/comments/13ksvsz/comment/jkm63qh/
Why don’t they collaborate with Oreboot project?
Rising Interest in oreboot and LinuxBoot

https://www.reddit.com/r/RISCV/comments/13ksvsz/comment/jkm63qh/
Why don’t they collaborate with Oreboot project?

https://forum.rvspace.org/t/oreboot-visionfive-2-support-effort/2211
Rising Interest in oreboot and LinuxBoot

https://www.reddit.com/r/RISCV/comments/13ksvsz/comment/jkm63qh/
Why don’t they collaborate with Oreboot project?

https://forum.rvspace.org/t/oreboot-visionfive-2-support-effort/2211

Someone from Intel contacted me, evaluating oreboot and LinuxBoot on RISC-V.

\(^2\)https://www.youtube.com/watch?v=gB3wgOuvLJQ
Rising Interest in oreboot and LinuxBoot

https://www.reddit.com/r/RISCV/comments/13ksvsz/comment/jkm63qh/
Why don’t they collaborate with Oreboot project?

https://forum.rvspace.org/t/oreboot-visionfive-2-support-effort/2211

Someone from Intel contacted me, evaluating oreboot and LinuxBoot on RISC-V.

ByteDance\(^2\) and many hyperscalers are using LinuxBoot.

\(^2\)https://www.youtube.com/watch?v=gB3wgOuvLJQ
Multiple companies, including Intel, Ventana and Rivos, are working together to define UEFI/ACPI 3 and platform runtime services 4 for RISC-V.

The RISC-V Software Ecosystem (RISE) project is a collaborative effort led by industry leaders with a mission to accelerate the development of open-source software for the RISC-V architecture.

---

3 https://riseproject.dev/
4 https://lists.riscv.org/g/tech-prs
Industry Collaboration

Multiple companies, including Intel, Ventana and Rivos, are working together to define UEFI/ACPI\(^3\) and platform runtime services\(^4\) for RISC-V.

\(^3\)https://riseproject.dev/
\(^4\)https://lists.riscv.org/g/tech-prs
Multiple companies, including Intel, Ventana and Rivos, are working together to define UEFI/ACPI\textsuperscript{3} and platform runtime services\textsuperscript{4} for RISC-V.

\textbf{RISE}

The RISC-V Software Ecosystem (RISE) project is a collaborative effort led by industry leaders with a mission to accelerate the development of open source software for the RISC-V architecture.

\textsuperscript{3}https://riseproject.dev/

\textsuperscript{4}https://lists.riscv.org/g.tech-prs
Bootloader Scopes and Goals
Drivers, Parsers, Loaders

Drivers
- Talk to hardware, e.g., graphics output
- Abstract concepts, e.g., filesystems
- Maybe provided by environment, such as UEFI or Xen

Parsers
- Understand data formats
- Translate raw data to usable form
- For configuration files and binaries

Loaders
- Potentially pickup configuration
- Load application to memory
- Place additional data in memory and/or registers
- Eventually, tell the platform (CPU) to execute from a specific memory address.

See also my talk on web boot:
- https://programm.froscon.org/2021/events/2703.html
- https://av.tib.eu/media/59579
- https://www.youtube.com/watch?v=nZgRV7gvZRw
Drivers, Parsers, Loaders

Drivers

- talk to hardware, e.g.,
  graphics output
- abstract concepts,
  e.g., file systems
- may be provided by
  environment, such as
  UEFI DXE or Linux

Parsers

- understand data
- translate raw data
  to usable form
- for configuration

Loaders

- potentially pick up
  configuration
- load application to
  memory
- place additional data
  in memory and/or
- registers

Eventually, tell the platform ("CPU") to execute from a specific memory address.

See also my talk on web boot:

- https://programm.froscon.org/2021/events/2703.html
- https://av.tib.eu/media/59579
- https://www.youtube.com/watch?v=nZgRV7gvZRw
Drivers, Parsers, Loaders

**Drivers**
- talk to hardware, e.g., graphics output
- abstract concepts, e.g., file systems
- may be provided by environment, such as UEFI DXE or Linux

**Parsers**
- understand data formats
- translate raw data to a usable form
- for configuration files and binaries

Eventually, tell the platform ("CPU") to execute from a specific memory address.

See also my talk on webboot:
- [https://programm.froscon.org/2021/events/2703.html](https://programm.froscon.org/2021/events/2703.html)
- [https://av.tib.eu/media/59579](https://av.tib.eu/media/59579)
- [https://www.youtube.com/watch?v=nZgRV7gvZRw](https://www.youtube.com/watch?v=nZgRV7gvZRw)
## Drivers, Parsers, Loaders

### Drivers
- Talk to hardware, e.g., graphics output
- Abstract concepts, e.g., file systems
- May be provided by environment, such as UEFI DXE or Linux

### Parsers
- Understand data formats
- Translate raw data to a usable form
- For configuration files and binaries

### Loaders
- Potentially pick up configuration
- Load application to memory
- Place additional data in memory and/or registers

---

Eventually, tell the platform ("CPU") to execute from a specific memory address.

See also my talk on web boot:
- [https://programm.froscon.org/2021/events/2703.html](https://programm.froscon.org/2021/events/2703.html)
- [https://av.tib.eu/media/59579](https://av.tib.eu/media/59579)
- [https://www.youtube.com/watch?v=nZgRV7gvZRw](https://www.youtube.com/watch?v=nZgRV7gvZRw)
Drivers, Parsers, Loaders

Drivers
▶ talk to hardware, e.g., graphics output
▶ abstract concepts, e.g., file systems
▶ may be provided by environment, such as UEFI DXE or Linux

Eventually, tell the platform (“CPU”) to execute from a specific memory address.

Parsers
▶ understand data formats
▶ translate raw data to a usable form for configuration files and binaries

Loaders
▶ potentially pick up configuration
▶ load application to memory
▶ place additional data in memory and/or registers

See also my talk on web boot:
▶ https://programm.froscon.org/2021/events/2703.html
▶ https://av.tib.eu/media/59579
▶ https://www.youtube.com/watch?v=nZgRV7gvZRw
Drivers, Parsers, Loaders

**Drivers**
- talk to hardware, e.g., graphics output
- abstract concepts, e.g., file systems
- may be provided by environment, such as UEFI DXE or Linux

**Parsers**
- understand data formats
- translate raw data to a usable form for configuration files and binaries

**Loaders**
- potentially pick up configuration
- load application to memory
- place additional data in memory and/or registers

Eventually, tell the platform ("CPU") to execute from a specific memory address.

See also my talk on webboot:
- https://programm.froscon.org/2021/events/2703.html
- https://av.tib.eu/media/59579
- https://www.youtube.com/watch?v=nZgRV7gvZRw
Firmware is well known to be an attack surface. Incidents increase:

▶ OEM compromise (e.g., MSI)
▶ Vulnerabilities in firmware interfaces, such as UEFI (e.g., Option ROMs)
▶ ACPI WPBT (Windows Platform Binary Table)

5 https://uefi.org/sites/default/files/resources/UEFI‑Firmware‑Security‑Concerns‑and‑Best‑Practices.pdf
7 https://eclypsium.com/research/everyone‑gets‑a‑rootkit/
Firmware is well known to be an attack surface.

---

6https://www.binarly.io/advisories/BRLY-2021-007/index.html
7https://eclypsium.com/research/everyone-gets-a-rootkit/
Firmware is well known to be an attack surface.

Incidents increase:

- OEM compromise (e.g., MSI)
- Vulnerabilities in firmware interfaces, such as
  - UEFI, e.g., Option ROMs, parsing variables
  - ACPI WPBT (Windows Platform Binary Table)

---

Supply Chain Security

Software Bill of Materials (SBOM)

Executive Order 14028 on Improving the Nation's Cybersecurity

▶ includes a lengthy definition of SBOM

Buyers can use an SBOM to perform vulnerability or license analysis, both of which can be used to evaluate risk in a product.

Posted on May 12, 2021

This is effectively a business requirement.
Supply Chain Security

Software Bill of Materials (SBOM)

Executive Order 14028 on Improving the Nation's Cybersecurity includes a lengthy definition of SBOM. Buyers can use an SBOM to perform vulnerability or license analysis, both of which can be used to evaluate risk in a product. Posted on May 12, 2021, this is effectively a business requirement.
Supply Chain Security

Software Bill of Materials (SBOM)

**Executive Order 14028 on Improving the Nation’s Cybersecurity**
- includes a lengthy definition of SBOM
  
*Bbuyers can use an SBOM to perform vulnerability or license analysis, both of which can be used to evaluate risk in a product.***

Posted on May 12, 2021
Supply Chain Security

Software Bill of Materials (SBOM)

**Executive Order 14028 on Improving the Nation’s Cybersecurity**

- includes a lengthy definition of SBOM
  
  *Buyers can use an SBOM to perform vulnerability or license analysis, both of which can be used to evaluate risk in a product.*

Posted on May 12, 2021

This is effectively a business requirement.
Interactive vs non-interactive

Non-interactive
Simple devices need no interaction in the bootloader, e.g., wristbands. Settings and upgrade functionality may come from other devices, such as phones.

Interactive
Flexible devices are designed to run any arbitrary operating systems and software. They require a rich user interface.

▶ changes settings
▶ setup a trust anchor
▶ enjoy colorful graphics

For more, see my talk on firmware settings and menus 8.

https://archive.fosdem.org/2022/schedule/event/fw_settings_and_menus/
Interactive vs non-interactive

Non-interactive
Simple devices need no interaction in the bootloader, e.g., wristbands. Settings and upgrade functionality may come from other devices, such as phones.
Interactive vs non-interactive

Non-interactive
Simple devices need no interaction in the bootloader, e.g., wristbands. Settings and upgrade functionality may come from other devices, such as phones.

Interactive
Flexible devices are designed to run arbitrary operating systems and software. They require a rich user interface.

---

8[https://archive.fosdem.org/2022/schedule/event/fw_settings_and_menus/]
Interactive vs non-interactive

Non-interactive
Simple devices need no interaction in the bootloader, e.g., wristbands. Settings and upgrade functionality may come from other devices, such as phones.

Interactive
Flexible devices are designed to run arbitrary operating systems and software. They require a rich user interface.
- change settings
- set up a trust anchor
- enjoy colorful graphics
For more, see my talk on firmware settings and menus⁸.

⁸https://archive.fosdem.org/2022/schedule/event/fw_settings_and_menus/
Applications

General-purpose bootloaders can be hard to customize. We will look at possible solutions.

Special-purpose bootloaders often need to be tailored toward a single use case.

https://danielmangum.com/posts/risc-v-bytes-exploring-custom-esp32-bootloader/
Applications

**General purpose**

General purpose bootloaders can be hard to customize. We will look at possible solutions.

Applications

General purpose
General purpose bootloaders can be hard to customize. We will look at possible solutions.

Special purpose
Special purpose bootloaders often need to be tailored\(^9\) toward a single use case.

\(^9\)https://danielmangum.com/posts/risc-v-bytes-exploring-custom-esp32-bootloader/
Projects and Stacks
Protocols, Interfaces and Features
Typical SoC have early code in their mask ROM, sometimes also called BootROM (BROM) or ZSBL (Zero Stage Boot Loader). Boot ROMs may offer protocols for loading over UART or USB, which is great for development, e.g., Allwinner FEL, JH71x0X modem. Depending on the platform design, multiple further stages are necessary. Developers need documentation: https://github.com/sipeed/LicheePi4A/issues/12.

I want to know how BROM loads Uboot image (emmc) to RAM, because I want to upstream Uboot. :) This is only vendor can know.
Stages / Phases

Typical SoCs have early code in their mask ROM, sometimes also called BROM (boot ROM) or ZSBL (Zero Stage Boot Loader).
Stages / Phases

Typical SoCs have early code in their mask ROM, sometimes also called BROM (boot ROM) or ZSBL (Zero Stage Boot Loader).

Boot ROMs may offer protocols for loading over UART or USB, which is great for development, e.g., Allwinner FEL, JH71x0 Xmodem.
Stages / Phases

Typical SoCs have early code in their mask ROM, sometimes also called BROM (boot ROM) or ZSBL (Zero Stage Boot Loader).

Boot ROMs may offer protocols for loading over UART or USB, which is great for development, e.g., Allwinner FEL, JH71x0 Xmodem.

Depending on the platform design, multiple further stages are necessary.
Stages / Phases

Typical SoCs have early code in their mask ROM, sometimes also called BROM (boot ROM) or ZSBL (Zero Stage Boot Loader).

Boot ROMs may offer protocols for loading over UART or USB, which is great for development, e.g., Allwinner FEL, JH71x0 Xmodem.

Depending on the platform design, multiple further stages are necessary.

Developers need documentation:

https://github.com/sipeed/LicheePi4A/issues/12
   I want to know how brom load uboot image(emmc) to ram, because I’m try to upstream uboot. :) This is only vendor can know.
provide understanding of the platform

clocks
peripherals
registers
how to program them
SDK, HAL, SVD

SVD is an XML based, structured format to describe an SoC, including memory maps, peripherals, and registers, as well as elaborating text.
SDK, HAL, SVD

SVD is an XML based, structured format to describe an SoC, including memory maps, peripherals, and registers, as well as elaborating text.

Rust offers tooling to convert SVD into usable code.
SDK, HAL, SVD

SVD is an XML based, structured format to describe an SoC, including memory maps, peripherals, and registers, as well as elaborating text.

Rust offers tooling to convert SVD into usable code.
SDK, HAL, SVD

SVD is an XML based, structured format to describe an SoC, including memory maps, peripherals, and registers, as well as elaborating text.

Rust offers tooling to convert SVD into usable code.

https://docs.rs/d1-pac/latest/d1_pac
Tools for Development and Flashing

During development, or to set up a custom environment, tools are necessary to reprogram a device.

BootROM Tools
- sunxi-fel/xfeltools
- vf2-loader
- snagboot

Provided by Bootloader
- U-Boots f command
- Linux MTD (memory technology device) drivers
Tools for Development and Flashing

During development, or to set up a custom environment, tools are necessary to reprogram a device.
Tools for Development and Flashing

During development, or to set up a custom environment, tools are necessary to reprogram a device.

Boot ROM Tools

- sunxi-fel/xfel tools
- vf2-loader
- snagboot
Tools for Development and Flashing

During development, or to set up a custom environment, tools are necessary to reprogram a device.

Boot ROM Tools
- sunxi-fel/xfel tools
- vf2-loader
- snagboot

Provided by Bootloader
- U-Boot sf command
- Linux MTD (memory technology device) drivers
Silicon and DRAM init
A bootloader for a rich OS relies on DRAM being initialized.
Silicon and DRAM init

A bootloader for a rich OS relies on DRAM being initialized.

- coreboot\textsuperscript{10}
  - supported RISC-V from very early on
- oreboot
  - note: started with RISC-V right away
- UEFI SEC+PEI
  - e.g., Project Mu, Tianocore EDK2
- U-Boot TPL/SPL

\textsuperscript{10} coreboot on RISC-V 2017 https://www.youtube.com/watch?v=CDNIWuf1jAk
Platforms, Ports and Flows
DXE and BDS are effectively the UEFI bootloader. They could also be replaced.
U-Boot offers a rich environment with an interactive shell and many boot options.
U-Boot offers a rich environment with an interactive shell and many boot options.

- supports multiple architectures
- more than 1000 boards, such as SBCs and routers
- can directly boot Linux and many other payloads

See also:
- State of the U-Boot, 2017 - Thomas Rini
  https://www.youtube.com/watch?v=dKBUSMa6oZI
  https://www.youtube.com/watch?v=rJtlAi8rxgs
U-Boot offers a rich environment with an interactive shell and many boot options.

- supports multiple architectures
- more than 1000 boards, such as SBCs and routers
- can directly boot Linux and many other payloads

See also:

- State of the U-Boot, 2017 - Thomas Rini
  https://www.youtube.com/watch?v=dKBUSMa6oZI
  https://www.youtube.com/watch?v=rJtlAi8rxgs
Linux is a well-known environment, so finding fitting engineers is easy.
LinuxBoot Integration with oreboot

<table>
<thead>
<tr>
<th>MMIO mapped</th>
<th>mask ROM</th>
<th>mask ROM</th>
<th>mask ROM</th>
<th>mask ROM</th>
</tr>
</thead>
<tbody>
<tr>
<td>128K</td>
<td>second boot</td>
<td>oreboot bt0</td>
<td>oreboot bt0 with DRAM init</td>
<td>oreboot bt0 with DRAM init and RustSBI</td>
</tr>
<tr>
<td>SRAM 1</td>
<td>DDR init</td>
<td>DDR init</td>
<td>DDR init</td>
<td>DDR init</td>
</tr>
<tr>
<td>128K</td>
<td>OpenSBI</td>
<td>OpenSBI</td>
<td>OpenSBI</td>
<td>OpenSBI</td>
</tr>
<tr>
<td>SRAM 2</td>
<td>U-Boot proper</td>
<td>U-Boot proper</td>
<td>U-Boot proper</td>
<td>U-Boot proper</td>
</tr>
<tr>
<td>8G</td>
<td>DRAM</td>
<td>DRAM</td>
<td>DRAM</td>
<td>DRAM</td>
</tr>
<tr>
<td>DRAM</td>
<td>LinuxBoot</td>
<td>LinuxBoot</td>
<td>LinuxBoot</td>
<td>LinuxBoot</td>
</tr>
</tbody>
</table>

Porting oreboot to the VisionFive1 board / JH7100 SoC
LinuxBoot Environments

Any Linux userland can be used, depending on needs. A common environment is u-root, the universal root filesystem.

▶ written in Go
▶ uses the Linux drivers
▶ offers booting from local storage
▶ can be embedded in flash
▶ easy to extend

https://u-root.org
LinuxBoot Environments

Any Linux userland can be used, depending on needs.

A common environment is u-root\(^\text{11}\), the universal root filesystem.

\(^{11}\text{https://u-root.org}\)
LinuxBoot Environments

Any Linux userland can be used, depending on needs.

A common environment is u-root\textsuperscript{11}, the universal root filesystem.

\begin{itemize}
  \item written in Go
  \item uses the Linux drivers
  \item offers booting from
    \begin{itemize}
      \item local storages
      \item network
    \end{itemize}
  \item can be embedded in flash
  \item easy to extend
\end{itemize}

\textsuperscript{11}\url{https://u-root.org}
OS Integration
Linux Distributions

OS Distributions such as openSUSE, Fedora, and Ubuntu need bootloaders. If the bootloader in flash does not suffice, they may bring their own, e.g., GRUB. For them, the best setup is standardized, such as Boot Loader Specification 12.

https://uapi-group.org/specifications/specs/boot_loader_specification/
OS Distributions such as openSUSE, Fedora and Ubuntu need bootloaders.
Linux Distributions

OS Distributions such as openSUSE, Fedora and Ubuntu need bootloaders. If the bootloader in flash does not suffice, they may bring their own, e.g., GRUB.

---

12 https://uapi-group.org/specifications/specs/boot_loader_specification/
Linux Distributions

OS Distributions such as openSUSE, Fedora and Ubuntu need bootloaders. If the bootloader in flash does not suffice, they may bring their own, e.g., GRUB.

For them, the best setup is standardized, such as Boot Loader Spec\textsuperscript{12}.

\textsuperscript{12}\url{https://uapi-group.org/specifications/specs/boot_loaderSpecification/}
Success Stories
Case Studies
Allwinner D1 with oreboot and LinuxBoot

The system bootswithin seconds. We created environments that allow for using a D1 as a USB gadget that can be used as an additional CPU for a laptop.
Allwinner D1 with oreboot and LinuxBoot

The system boots within seconds. We created environments that allow for using a D1 as a USB gadget that can be used as an additional CPU for a laptop.
Allwinner D1 with oreboot and LinuxBoot

The system boots within seconds. We created environments that allow for using a D1 as a USB gadget that can be used as an additional CPU for a laptop.
kboot: Booting FreeBSD with LinuxBoot\textsuperscript{13}

FreeBSD’s kboot is a Linux binary that loads FreeBSD’s kernel, modules, tuneables and other metadata via the kexec(2) API

\textsuperscript{13}\url{https://www.bsdcan.org/events/bsdcan_2023/schedule/session/138-kboot-booting-freebsd-with-linuxboot/}
Other Operating Systems

What is my purpose?

You boot Plan 9.

I happily will.
Saving Costs
Sharing Code

DeviceTrees describes specific boards and are shared between projects.

- Linux
- U-Boot
- FreeBSD

Note: U-Boot also shares parts of its driver model with Linux.

Rust code can be shared using crates, which can speed up driver development.

https://crates.io/
Sharing Code

Device Trees
Device Trees describe specific boards and are shared between projects.
▶ Linux
▶ U-Boot
▶ FreeBSD

Note: U-Boot also shares parts of its driver model with Linux.

14https://crates.io/
Sharing Code

Device Trees
Device Trees describe specific boards and are shared between projects.
- Linux
- U-Boot
- FreeBSD

Note: U-Boot also shares parts of its driver model with Linux.

Rust
Rust code can be shared using crates\textsuperscript{14}, which can speed up driver development.

\textsuperscript{14}https://crates.io/
Drivers

LinuxBoot requires only writing drivers once.
Less effort means lower costs and faster time to market.
Thanks! :)

Follow Me

https://github.com/orangecms
https://twitter.com/orangecms
https://twitch.tv/cyrevolt
https://youtube.com/@cyrevolt

Daniel Maslowski
https://github.com/oreboot/oreboot