

RISC-V support in GRUB2

GRUB mini-summit 2020

Piotr Król and Michał Żygowski









Piotr Król
3mdeb Founder

- coreboot contributor and maintainer
- Conference speaker and organizer
- Trainer for military, government and industrial organizations
- Former Intel BIOS SW Engineer
- 12yrs in business
- 6yrs in Open Source Firmware
- C-level positions in





Michał Żygowski
Firmware Engineer

-  [@_miczyg](https://twitter.com/_miczyg)
-  michal.zygowski@3mdeb.com
-  linkedin.com/in/miczyg
-  facebook.com/miczyg1395
- Braswell SoC, PC Engines and Protectli maintainer in coreboot
- interested in:
 - advanced hardware and firmware features
 - coreboot
 - security solutions

- Norbert Kamiński 3mdeb Junior Embedded Systems Engineer for getting through software stack debugging and integration for demo purposes

Practical demonstration GRUB2 on RISC-V and discussion about boot firmware and bootloader ecosystem

Why you should care?

RISC-V is one the most important trends in semiconductor, because of open-source, license fee and royalty payment-free RISC processor templates for a wide range of applications

Healthy open source firmware (OSF) and bootloaders ecosystem for RISC-V is key to correct™ CPU initialization and provisioning of dedicated features

- **Silicon Vendors** depend on OSF and bootloaders reference implementations to expose their product features to higher layers (OSes and applications)
- **IoT and Embedded Systems makers** depend on OSF and bootloaders to provide advanced platform security features and trusted environment for application code execution
- **Security providers** depend on OSF and bootloaders to efficiently leverage security, runtime and management capabilities in their solutions
- **AI/ML/Analytics companies** depend on OSF and bootloaders to fully utilize underlying hardware and tune it to particular workload
- Other use cases of RISC-V
 - companion processor to address market specific computation
 - fully libre computing devices

RISC-V is a free and open ISA enabling a new era of processor innovation through open standard collaboration.

The RISC-V ISA delivers a new level of free, extensible software and hardware freedom on architecture, paving the way for the next 50 years of computing design and innovation.

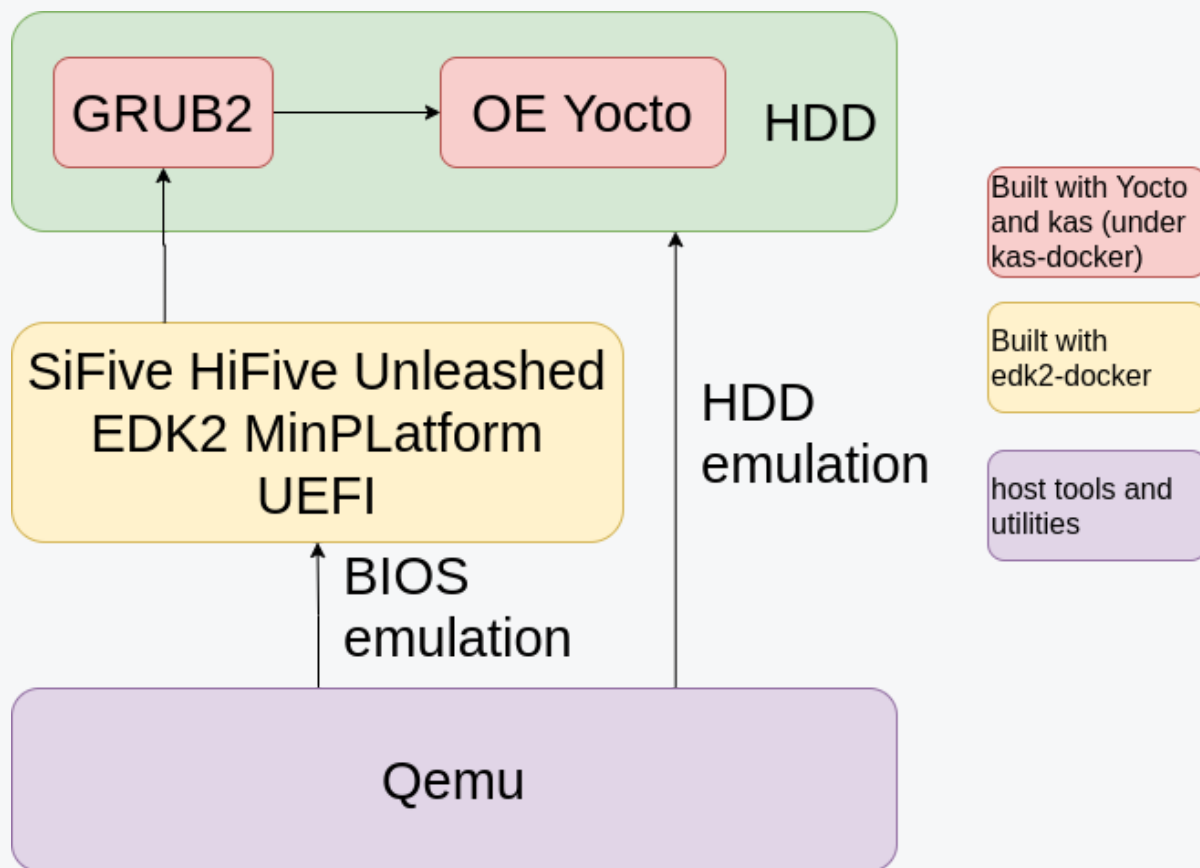
- a breath of fresh air comparing to x86 architecture
- POWER architecture is also very open (no-royalty for licensing ISA from IBM)
- openness the key to respecting the privacy and providing security

- SiFive HiFive Unleashed EDK2 MinPlatform:
 - <https://github.com/riscv/riscv-edk2>
 - <https://github.com/riscv/riscv-edk2-platforms>
- QEMU emulation target
- GRUB2 for RISC-V with patches for booting Linux in UEFI:
 - <https://www.mail-archive.com/grub-devel@gnu.org/msg30107.html>
- OE Yocto for RISC-V built with kas:
 - <https://github.com/3mdeb/meta-riscv/tree/kas-support>

- Initial patches supporting RISC-V in GRUB2 were published in 2018

"As part of the plan for total world domination, we would like to make sure that booting on RISC-V is in a sane state before anyone goes and does quick hacks "just because", Alexander Graf.

- Work was merged in 2019 as part of GRUB2 2.04 release
- How it can be tested?



- MinPlatform UEFI implementation for various targets
- <https://github.com/riscv/riscv-uefi-edk2-docs#supported-risc-v-platforms>
- Supported targets:
 - SiFive HiFive Unleashed real hardware and Qemu,
 - VirtIO machine,
 - Freedom U500 VC707 FPGA,
 - Andes AX25 + AE350 FPGA

- Looks like there is very little effort needed to boot Linux
- <https://www.mail-archive.com/grub-devel@gnu.org/msg30107.html>
- In fact, it was not so obvious

- It was not so easy to find proper branches/commits to build the working EDK2
- Clone repositories and their modules

```
$ git clone --recursive git@github.com:riscv/riscv-edk2-platforms.git - devel-boot-kernel  
$ git clone --recursive git@github.com:riscv/riscv-edk2.git - d89fe9bc7d edk  
$ git clone --recursive git@github.com:riscv/riscv-gnu-toolchain.git
```

- Install toolchain dependencies:

```
$ sudo apt-get install autoconf automake autotools-dev curl python3 \  
libmpc-dev libmpfr-dev libgmp-dev gawk build-essential bison flex \  
texinfo gperf libtool patchutils bc zlib1g-dev libexpat-dev
```

- This process should be containerized

- Make the elf toolchain (It took ~1 hour):

```
$ cd riscv-gnu-toolchain  
$ ./configure --prefix=/home/nkaminski/projects/RiscV/riscv-gnu-toolchain/output  
$ make
```

- Add cross-compiler to the PATH:

```
$ export PATH="/home/nkaminski/projects/RiscV/riscv-gnu-toolchain/output/bin:$PATH"
```

- Export workspace and package directories:

```
$ export WORKSPACE=$HOME/projects/RiscV/edk2  
$ export PACKAGES_PATH=$PWD/edk2:$PWD/riscv-edk2-platforms
```

- Setup build variables and build base tools:

```
$ . edk2/edksetup.sh  
$ make -C edk2/BaseTools
```

- Export cross compiler prefix:

```
$ GCC5_RISCV64_PREFIX=riscv64-unknown-elf-
```

- Now you are ready to build the RISC-V EDK2:

```
$ build - RISC64 -t GCC5 - Platform/SiFive/U5SeriesPkg/FreedomU540HiFiveUnleashedBoard/U540.dsc
```

- To prepare the grub-efi binary and proper initramfs image we have used the Yocto framework
- The Yocto Project (YP) is an open source collaboration project that helps developers create custom Linux-based systems regardless of the hardware architecture.
- At first you need to clone the 3mdeb fork of meta-riscv:

```
$ git clone git@github.com:3mdeb/meta-riscv.git - kas-support
```


- If you do not have installed kas tools look at our blog post:

<https://blog.3mdeb.com/2019/2019-02-07-kas/>

- Build yocto image (it takes about 4 hours):

```
$ SHELL=bash kas-docker --ssh-dir ~/.ssh shell meta-riscv/kas-U540.yml
```

- When build is finished go to the deploy directory

```
$ cd build/tmp/deploy/images/freedom-u540
```

- Now you need to generate boot image

```
$ mkfs.msdos -C linux.iso 64000
$ sudo losetup -P -f --show linux.iso
$ sudo mount /dev/loopX /mnt

$ sudo mkdir /mnt/EFI
$ sudo mkdir /mnt/EFI/BOOT
$ sudo cp grub-efi-bootINVALID.efi /mnt/EFI/BOOT

# Add grub.cfg config
$ sudo touch /mnt/EFI/BOOT/grub.cfg
$ sudo vim /mnt/EFI/BOOT/grub.cfg

$ sudo cp fitImage /mnt/fitImage.efi
$ sudo cp initramfs.cpio /mnt
$ sudo umount /mnt
$ sudo losetup -d /dev/loopX
```

- Copy linux.iso to the EDK and device tree blob:

```
$ cp linux.iso ~/projects/RiscV/riscv-edk2-platforms/Silicon/RISC-V/ProcessorPkg/Universal\
/EspRamdisk/linux.iso
$ cp hifive-unleashed-a00--5.4.x+git0+fc83346191-r0-freedom-u540-20201123174948.dtb \
~/projects/RiscV/riscv-edk2-platforms/Platform/RISC-V/PlatformPkg/Universal/Sec/Riscv64/U540.dtb
```

- Rebuild EDK

- Build the QEMU as it is shown in the documentation:

<https://risc-v-getting-started-guide.readthedocs.io/en/latest/linux-qemu.html>

- Go to the EDK2 BIOS directory:

```
$ cd ~/projects/RiscV/edk2/Build/FreedomU540HiFiveUnleashed/DEBUG_GCC5/FV
```

- Run the QEMU:

```
$ qemu-          -riscv64 -cpu sifive-u54 -machine sifive_u -bios U540.fd -m 2048 \  
-nographic -smp cpus=5,maxcpu
```

- EDK2:

<https://asciinema.org/a/HjePenslDo9ru2JuDPR5fUR9d?speed=0.03&t=6&cols=100&rows=30&size=big>

- EDK2 + grub-efi:

<https://asciinema.org/a/NdDnYlfTLI5bOkOeAgJ5h1DzX?cols=100&rows=30&size=big>

- It is interesting that Amazon seem to actively sponsor RISC-V development, especially that they are not in foundation and that Alibaba Cloud seem to invest deeply in it.
- This is not new since Amazon acquired FreeRTOS supports RISC-V since Feb 2019
- Maybe now they building something competing with Alibaba?
- It is also worth to mention that WD provided support for EFI stub in Linux
- HPE contributed edk2 support
- From our own research it looks like some companies look for RISC-V as trusted S-CRTM and security offload/accelerator
- Even if it may take time to provide competing performance/\$ by RISC-V, then it is worth to consider it as companion chip
- We also watching closely libre and freedom path RISC-V takes, despite concerns about SBI we hope for trustworthy computing alternatives to OpenPOWER

Q&A