Will you boot Haiku, on a non intel platform, no BIOS winter?

Booting Haiku on non-x86, a never-ending story.

François Revol
revol@free.fr

Haiku?

- Free Software Operating System
- Inspired by the BeOS
- Our own kernel
- Our on GUI

BeOS: Always on the run

- Hobbit BeBox prototype…
  - AT&T EOLed Hobbit in 1994
- PPC BeBox (2 × 603e)
  - Be stopped making hardware
- PPC Mac (pre-G3)
  - Then Steve said “you won’t get the specs” ☹
- Intel PC
  - “He Who Controls the Bootloader” (2001)

Jean-Louis Gassée quote

“I once preached peaceful coexistence
with Windows.
You may laugh at my expense -- I deserve it.”

Booting on PC

- BIOS → MBR {Bootman,GRUB chainload}
- MBR → partition boot sector (stage1)
  - Needs partition offset (`makebootable`)
    - Shouldn't be required
- stage1 → `haiku_loader`
- haiku_loader → `kernel_x86`

haiku_loader

- Now in haiku_loader.hpkg (uncompressed)
- Sets graphics mode (for boot splash)
- Loads kernel, modules… from BFS
  - … or initrd-like tar.gz 😉
- Sets up MMU, FPU…
- And calls the BIOS for many things…
- Calls the kernel with `struct *kernel_args`
  - Which contains `platform_args` and `arch_args`

Challenges

- Since R1/beta1: Packaging
  - Almost reproducible build
  - But requires strict dependencies
- Haiku needs Haiku to build
  - Easy on x86
- Bootstrap builds = easy to break
- C++ everywhere
  - C++ issue currently on ARM bootstrap

PowerPC

- Started long long ago…
  - Pegasos 1 … buggy OF
- (some years passed)
- Sam460ex & other AmigaOS-compatibles
  - U-Boot (heavily modded)
- QEMU Mac PPC always had issues
- BeBox
  - Very dumb bootrom; needs PEF binary

OpenFirmware

- Nice, even cleaner than BIOS
- Except for ACPI-like things
  - Clean power-off = keep OF mappings
  - Maybe use an emulator?
    - We do this for VESA BIOS already
- Standardized bindings
- Framebuffer calls too high-level
  - Get phys addr?

AmigaOne X-1000

- You read the specs.
- The specs says “It uses CFE”.
- You implement CFE support in your loader.
- It doesn’t work.
- You notice it runs an OF payload to run Linux.
- 6 years later, you remove CFE support.

* As an homage to “Adventures in Graphics Drivers” (Be Newsletter vol.4 1999)

ACube Sam460ex (PPC)

- Embedded board
- Book-E CPU
  - No page tables baby, TLBs and that's it®
  - PAE (including for I/O…) + I/O on
  - "Bringing PowerPC Book E to Linux" (2003) (3 tries)

> **5.3 Keep It Simple Stupid**
>
> Sometimes one has to travel a long road to eventually come back to the simple solution.

- U-Boot fork form Acube
  - Custom 😑 API for AmigaOS loader (Parthenope)
    - Not what we need anyway
  - How do I get the framebuffer? Ended up hardcoding hw @ 😩
- (Start of) QEMU target mostly? upstreamed by Zoltan Balaton

Booting on Sam460ex...

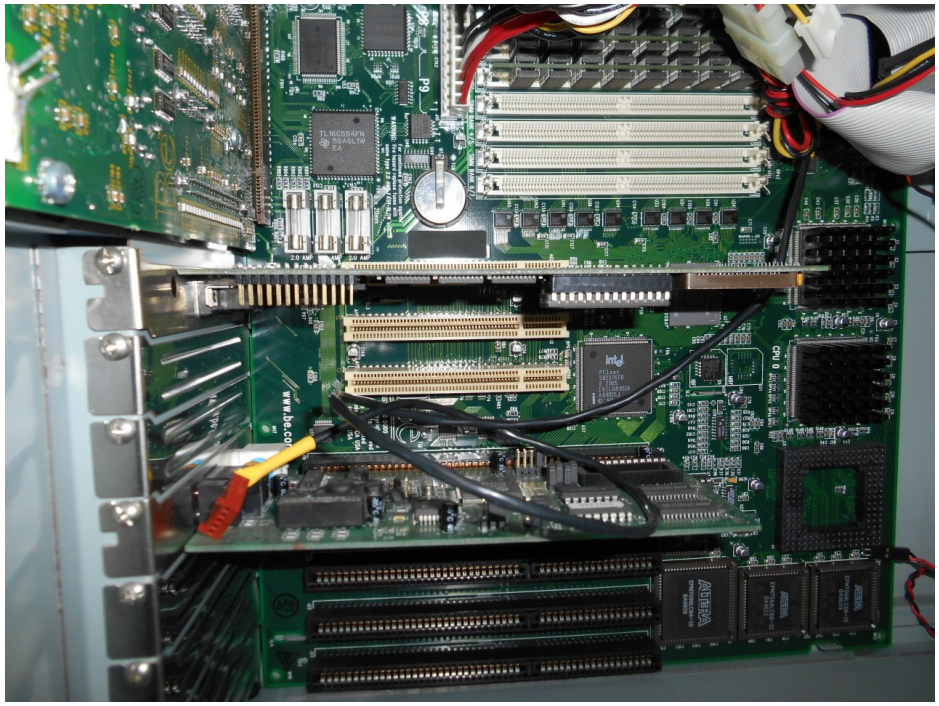
```
setenv booth1 'setenv ipaddr 192.168.4.100;
tftpboot 0x4000000
192.168.4.2:haiku_loader_linux.ub'
setenv booth2 'tftpboot 0x8000000
192.168.4.2:haiku_initrd.ub'
setenv booth3 'tftpboot 0xc000000
192.168.4.2:sam460ex.dtb'
setenv booth4 'bootm 0x4000000 0x8000000 0xc000000
plop'
setenv booth 'run booth1; run booth2; run booth3;
run booth4'
saveenv
run booth
```

PPC Macintosh (QEMU)

- Used to have OpenHackware
  - Not really Forth, just signature matching
- Replaced with OpenBIOS
- PCI bus memory at 0x80000000
  - … and no translation declared in OF tree 😱
  - Move kernel load address?

BeBox

- A blue box,
  bigger on the inside…
   No, not this one!
- Port started recently
- Loader builds and is found by the ROM
- WIP: fix PEF ld output
  - Retro68 might help



PPC TODO

- Dump OF tree to an FDT early in haiku_loader
- Cleanup sam460ex branch
- Finish PEF support in ld
- Finish bebox branch

ARM

- Started long ago (GSoC)
- “Cool there’s a BIOS-like API in U-Boot!”
  - 1 week passed… “can’t find the entry point!”
  - “Oh yeah, it’s for NetBSD, so nobody cares”
- Loads the kernel
- Broke
- Fixed
- Broke…

U-Boot

- So yeah, no API ☺
- mkimage supports -O …
- If memory size is fixed in FDT, you’re lucky.
- Doesn’t know about BFS…
- Where’s the framebuffer info in the FDT? 💻
  - But, wait, there’s simple-framebuffer binding!
    - So why nobody cares? 😁
- Let’s look at the global data…

U-Boot

```
• typedef struct uboot_gd {
    // those are the only few members that we can trust
    // others depend on compile-time config
    struct board_data *bd; // arch-dependent as well...
    uint32 flags;
    uint32 baudrate;
  #ifdef __ARM__ // !???
    uint32 have_console;
    uint32 reloc_off;
    uint32 env_addr;
    uint32 env_valid;
    uint32 fb_base; // <- THIS I WANT! But where's WxH?
  #endif
  } uboot_gd;
```

U-Boot

- `mkimage -O ==> set operating system to 'os'`
- Nice, let's add Haiku!
  - But existing boards won't support it anyway…
- Ok, let's just fake ~~NetBSD~~,
  - `start_netbsd(struct board_info *bd, struct image_header *image, const char *consdev, const char *cmdline)`
- Ok, let's just boot as ~~raw~~,
  - `start_raw(int argc, const char **argv)`
- Ok, let's just fake Linux. But which one? 😄
  - `start_linux(int argc, int archnum, void *atags)`
    `{ // newer U-Boot pass the FDT in atags`
    `return start_gen(0, NULL, NULL, atags); }`

U-Boot TODO

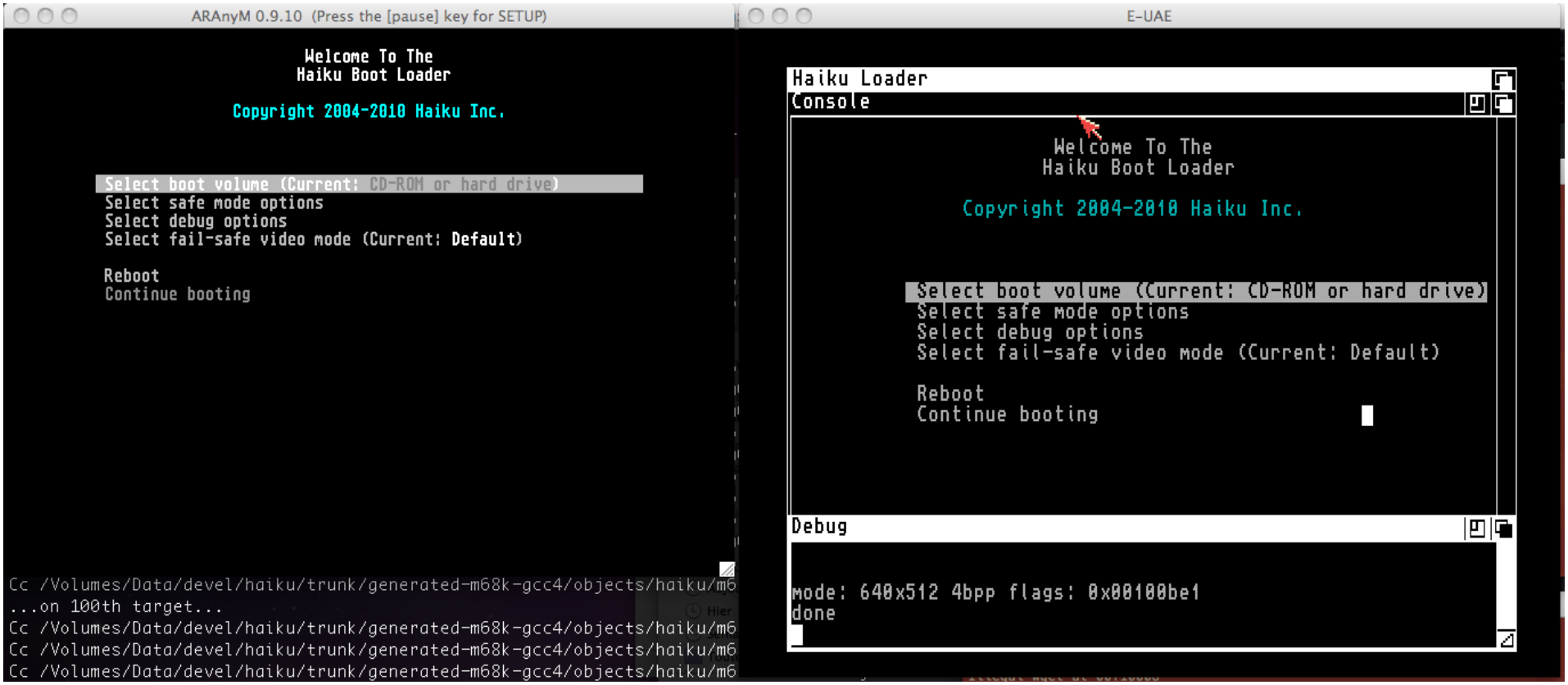
- Separate firmware repository (☑ kallisti5)
- MMC image tool (☑ rune by kallisti5)
- Clean up loader gfx code…
- Assume FDT /chosen/framabuffer
- Write board-specific helper cmds to patch FDT
  - We can link that to specific U-Boot builds…
    - When we have the source
  - Or patch FDT in haiku_loader

M68K (≥ 68030)

- Mostly for fun™
- Targets Atari Falcon & Amiga (with lots of RAM)
  - DOS-like boot floppy with checksum variations
  - Weird video modes, custom chips…
- Some hardware still produced
  - Firebee (ColdFire Atari compatible)
  - Apollo Vampire 68080 cards for Amiga & Atari
- TOS & AmigaDOS usable from haiku_loader 😎

Demo

M68K (as of 2010)



Sparc, MIPS…

- Nothing to see here, move along
  - (barely started, and removed)
  - (but if you make it work, please send patches)

(U)EFI

- GPT support ☑
- Bulk of the work by JessicaH since 2014
- De-x86zation by kallisti5 for ARM support
- EFI doesn’t know about BFS…
  - Manual copy of loader to the FAT
  - Not yet automatically done in R1/beta1

RISC-V

- 2018-05-02: elf: Add aarch64 and riscv defines
- 2018-11-04: build: Add riscv architecture
- 2018-12-05: Finally some stubs \o/
- Please send dev boards our way ☺

I want to help! Where do I start?

- www.haiku-os.org/ …/getting-started
- cgit. …/docs/develop/kernel/ports
- Pick your target…
- dev. …/SubmittingPatches !