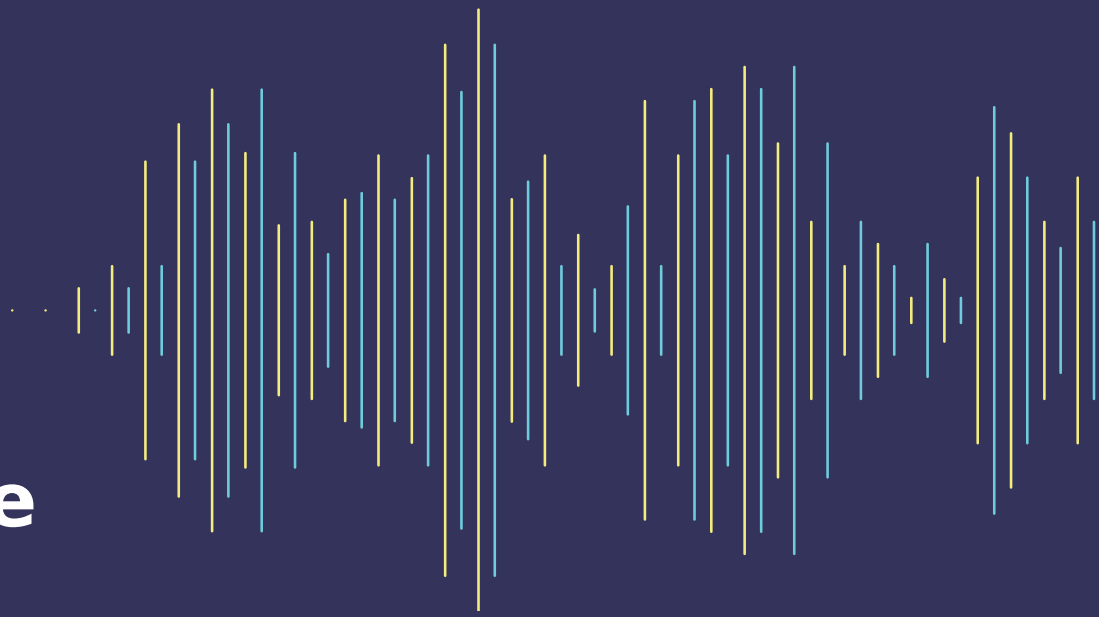
SOUND
OPEN
FIRMWARE

Journey from Closed to Open:

Lessons Learned from Open Sourcing Sound Open Firmware

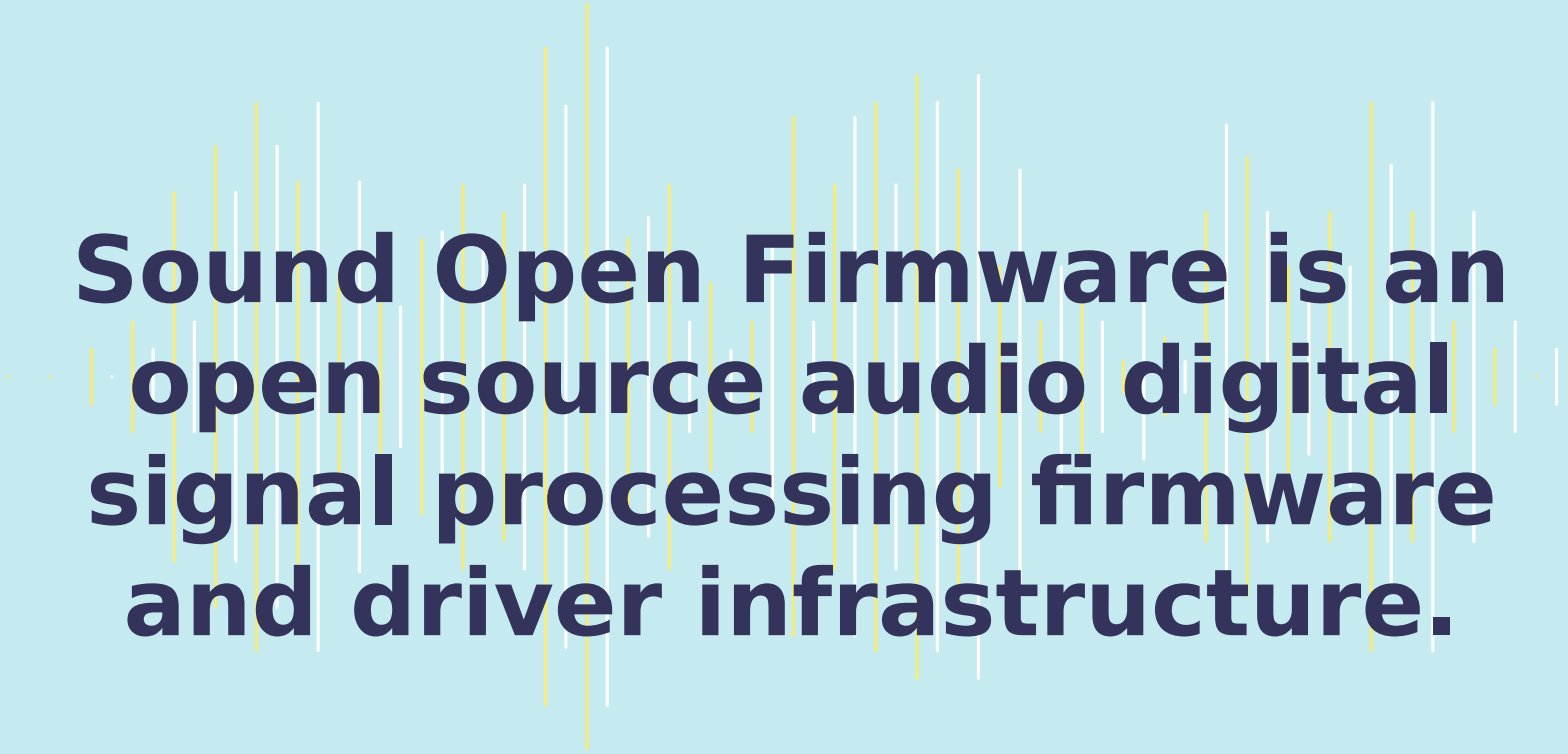


Mission

1. Inspire others to open source firmware.
2. Show how firmware can be open sourced.
3. Discuss common challenges open sourcing firmware and how to overcome them.
4. Introduce Sound Open Firmware (SOF) architecture.

About Me

- Employed by Intel as a software architect.
- Linux user and engineer since 1994.
- Developed AsoC and PMIC abstraction layers.
- Linux audio engineer since 2003.
- Working on audio DSP’s since 2008.
- Working on SOF since 2015.



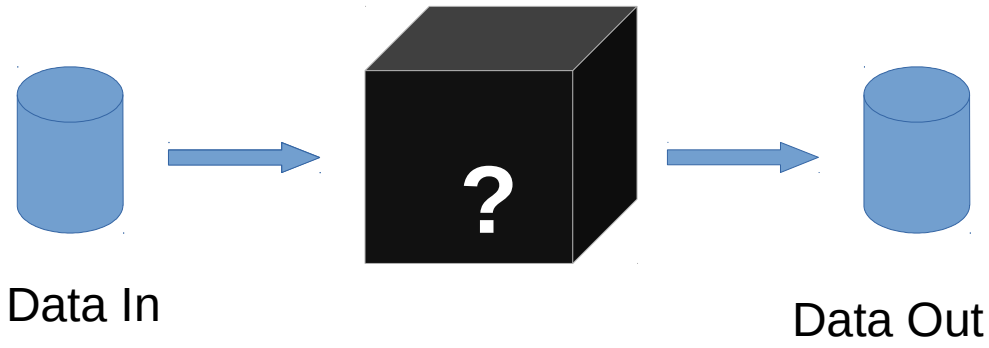
Sound Open Firmware is an open source audio digital signal processing firmware and driver infrastructure.

The Dark Ages

**Darkness due to lack of source code*

Closed Source Firmware

- Historically the standard practice amongst firmware development teams with a few exceptions.
- Often seen as a black box by integrators and driver & user space developers.
  - Makes it more difficult to debug problems in other areas of the stack.
  - Impossible for upper stack developers to debug hardware problems.
- Usually developed alongside a single OS specific driver.
- Documentation never truly matches or keeps up with firmware source code.



Changing Course

Factors influencing open sourcing.

Market Drivers

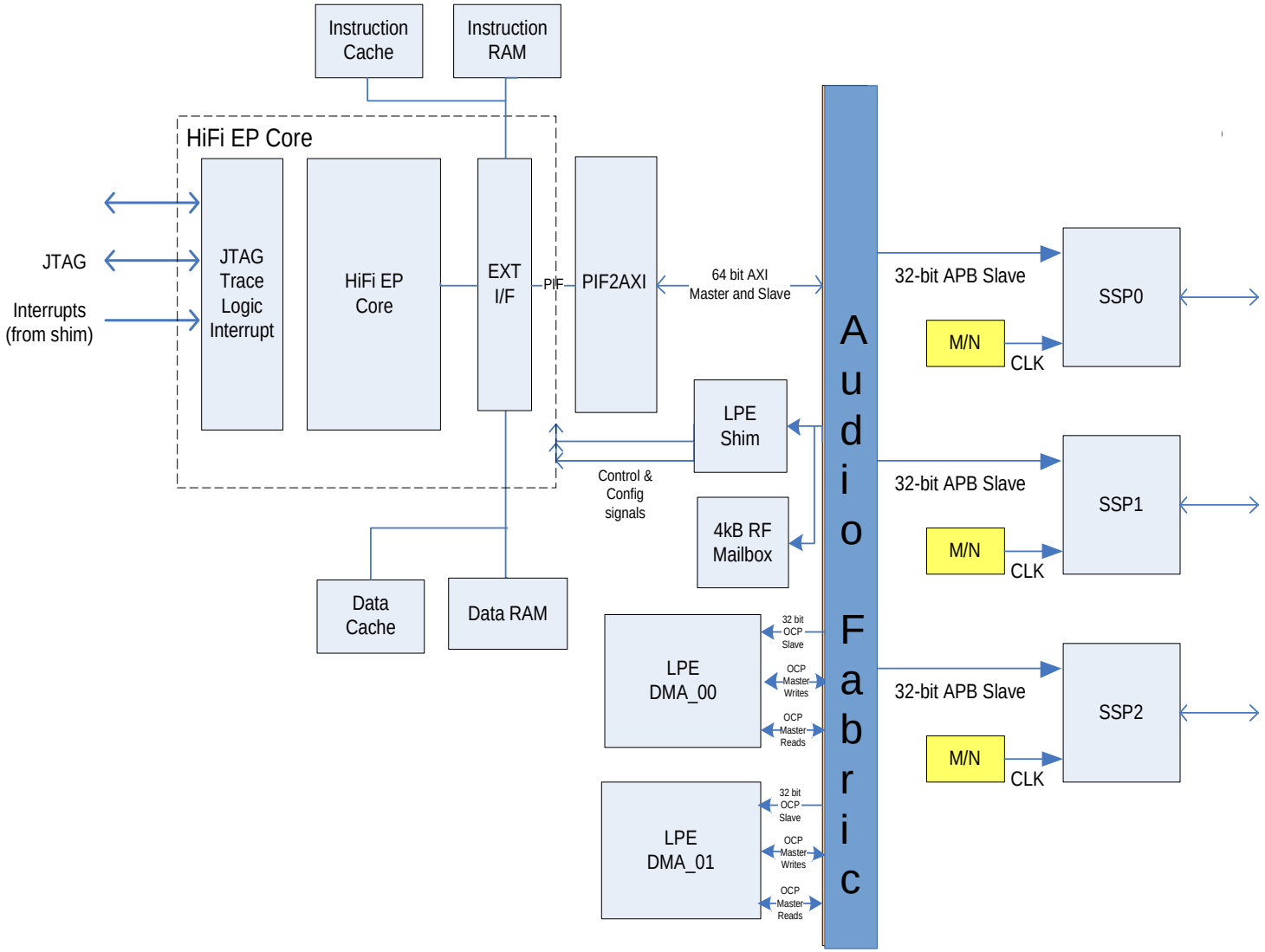
- Demand for speech and voice authentication and recognition applications and technologies
- Impact of AI on accuracy of speech and voice recognitions
- Growth in voice control-based devices in consumer and enterprise markets

Audio, speech and voice has become ubiquitous

Open Source Hardware

- Minnowboard Project
- Open Source Hardware.
  - Baytrail CPU, dual core @1.33GHz, 2GB DDR
  - Tensilica Xtensa audio DSP @ 400MHz
  - Open schematics, PCB layout, BOM
- Open source Software
  - Full open source Linux software stack
- Open source Firmware
  - Open source coreboot BIOS
  - NO open source audio DSP firmware

Minnowboard DSP Architecture



- Xtensa HiFi 2EP core.
- 96kB Instruction RAM
- 168kB Data RAM
- 2 * DMACs
- 3 * I2S ports
- PCI device from host OS

Challenges Ahead

Be prepared !

Political Challenges

Challenges

- Company policy for audio DSP firmware was historically closed source.
- Most colleagues were initially either strongly in favour or strongly against.
  - Fears around disclosing IP.
  - How do we add value?

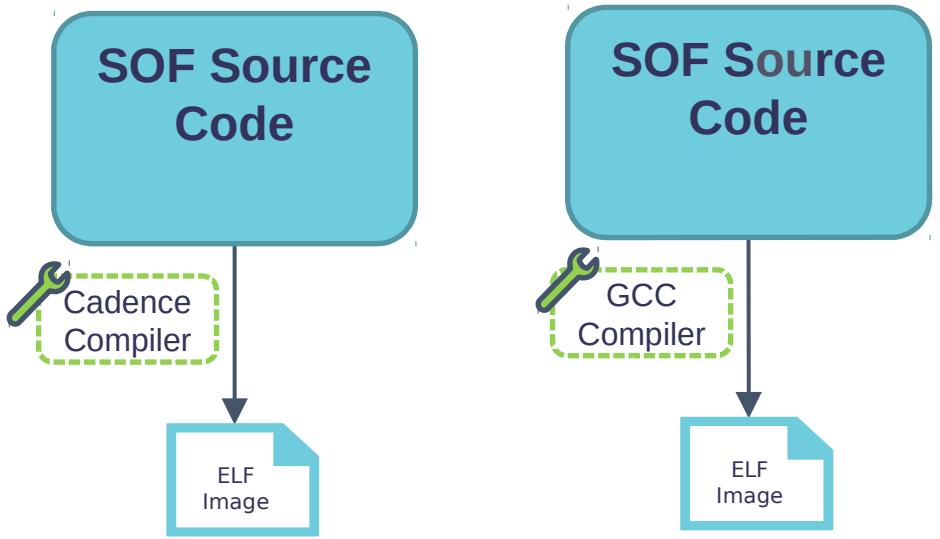
Solutions

- Be prepared to fight the same battle more than once.
- Build ground swell of opinion and facts behind open source.
- Helps if you have proof of concept code - “skunkworks”!
- Tell the world about your project!

“You may have to fight a battle more than once to win it.”
Margaret Thatcher

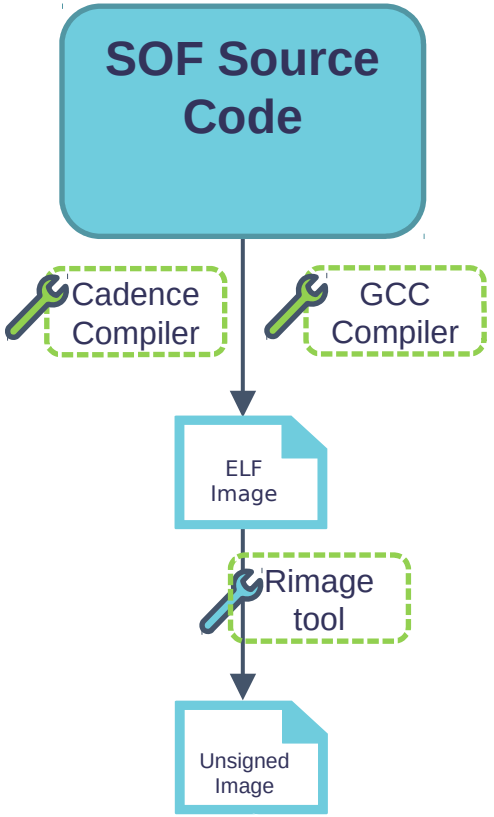
Technical Challenges - Compiler

- Xtensa ISA differs between cores adding complexity for compilers.
- Cadence provide an optimising compiler for Xtensa
  - Commercial license for some targets $$$
  - Free for some targets like Minnowboard :)
- Need open source compiler for community.
  - GCC supports xtensa base ISA
  - GCC does not support xtensa SIMD/VLIW.



Add GCC support for Minnowboard xtensa ISA.

Technical Challenges – Image Builder



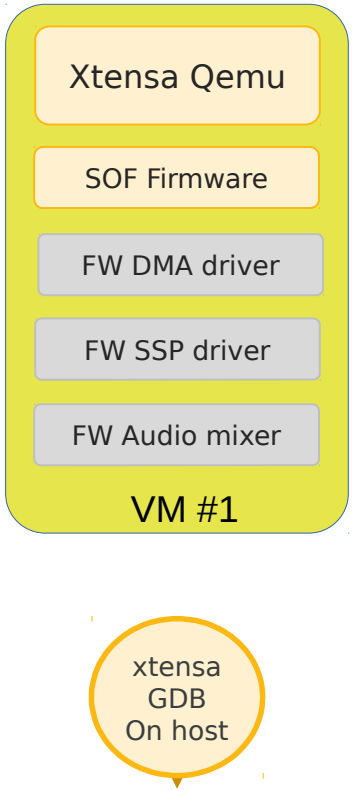
- Compilers generate ELF images.
- ELF images don’t run on bare metal.
- ELF image needs converted to binary image format for loading into DSP memories.
- Need to have tools that can convert ELF to multiple different firmware image formats.

Create tool to convert ELF images into multiple different image formats.

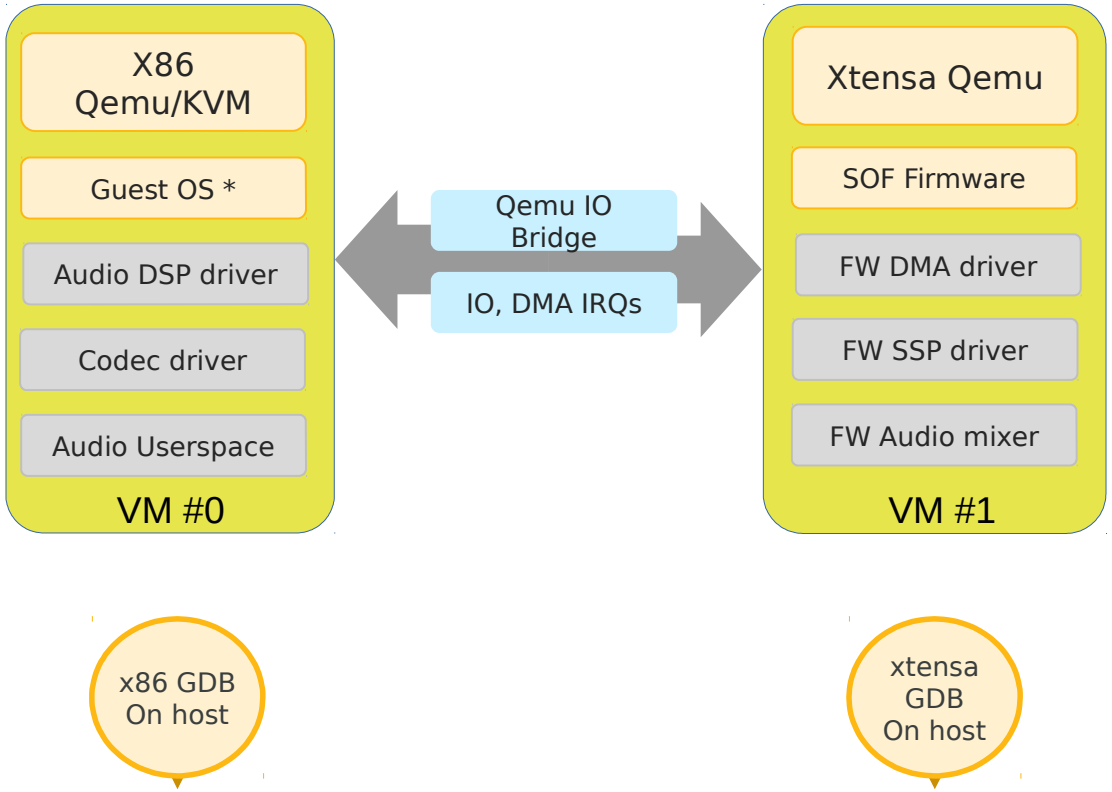
Should be able to run code now !

Technical Challenges – DSP Emulator

- Code not easy to debug
  - No debugger (yet)
  - No printf();
  - JTAG can’t be used – Intel only.
- Emulation can be used to debug bring up.
- Qemu already supports base Xtensa ISA
- Qemu support added
  - Minnowboard DSP
  - Extra registers and instruction not in base ISA



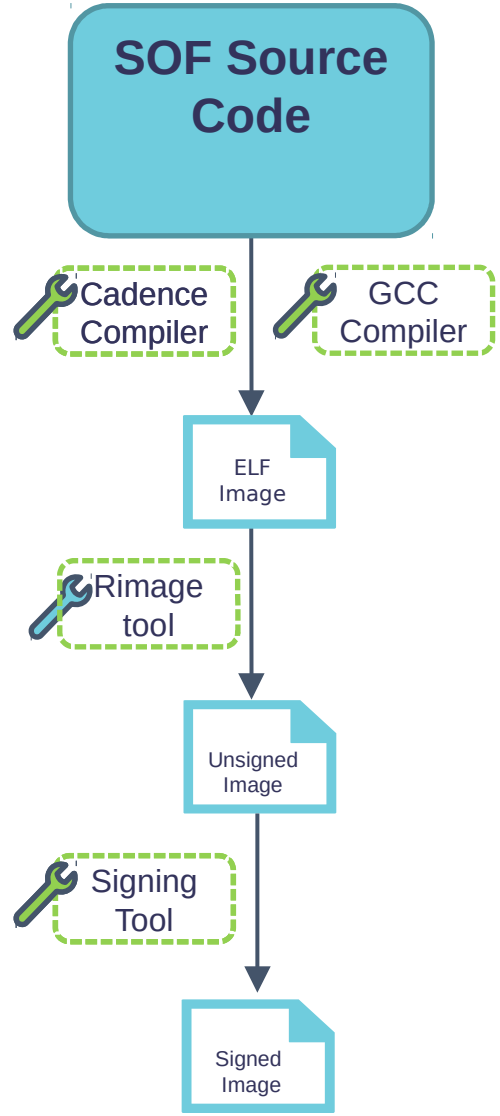
Technical Challenges – Heterogeneous Emulator



- Firmware must be debugged alongside driver.
- Qemu used to virtualise drivers and firmware together.
- Host side almost real time.
- DSP side emulated.

Technical Challenges – Code Signing

- Newer DSP hardware has security that validates firmware binaries.
- Code signing support was added to open source rimage tool.
  - PCKS #1.5
  - Openssl
- Created “public” private key to be used for developer hardware e.g. UP^2
- Implications with GPLv3 “tivoisation” clause.

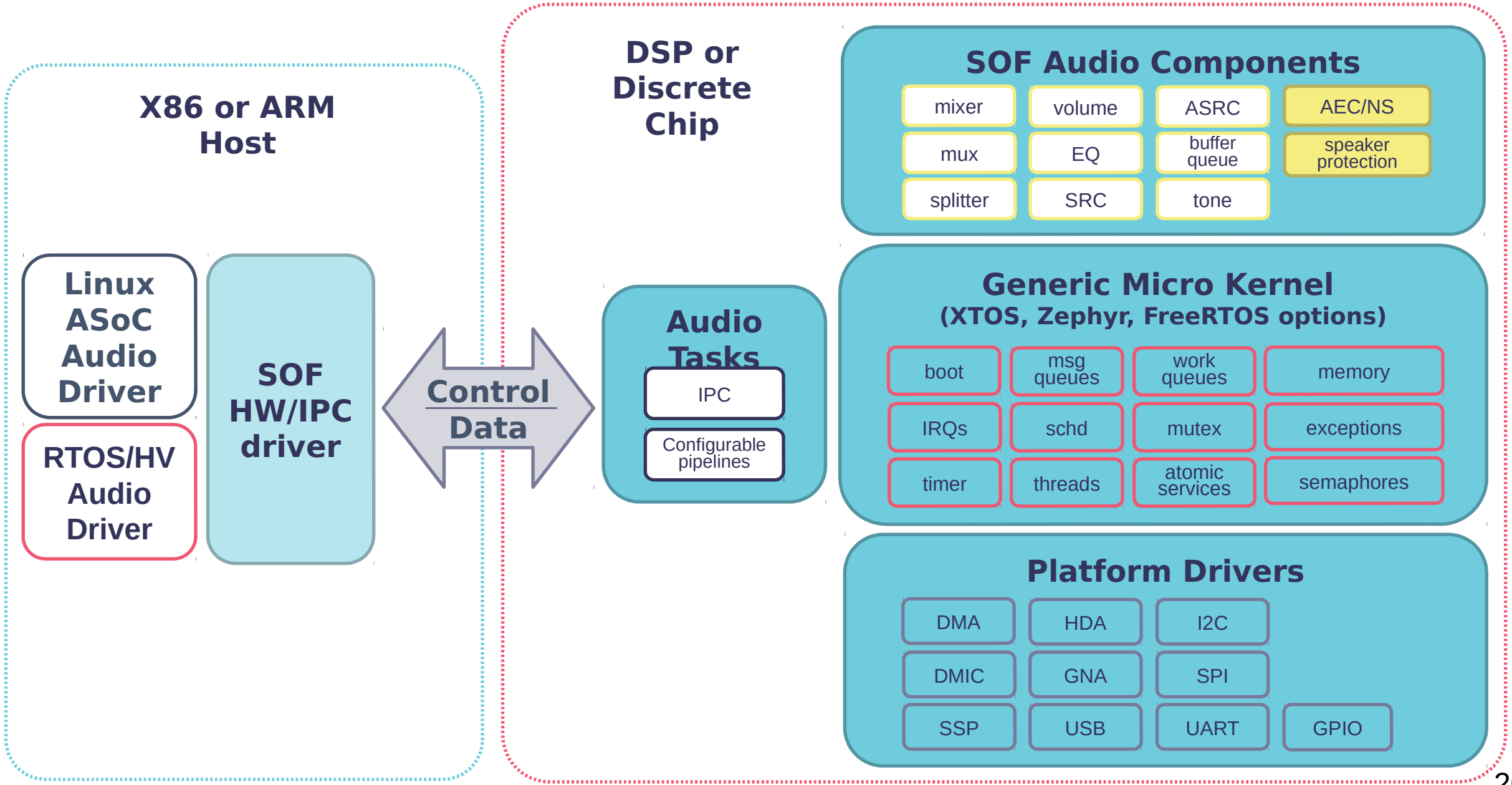


Build a Community

- A healthy open source project needs a healthy community!
- Who are the community?
  - Commercial users who deploy in products.
  - Non commercial hobbyists.
  - Academics researching audio processing algorithms.
- Use external code hosting platforms like Github, mailing lists, IRC, wiki’s etc and do development in public.
- Release patches and code “early and often”.
  - Don’t do infrequent code drops.
  - No “abandonware”.
- Accept patches from others.
- Accept bug reports from others.

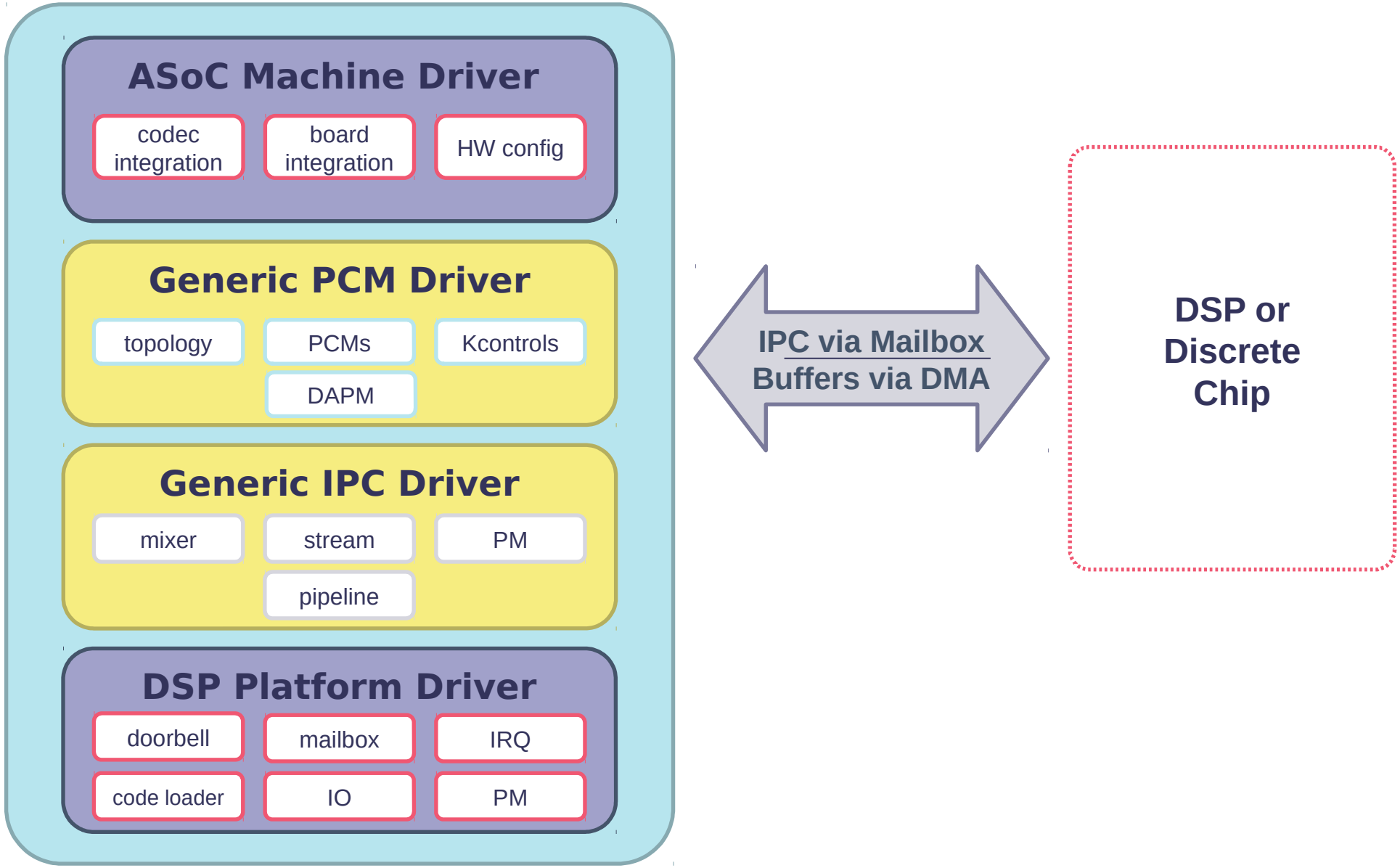
Hello World

Firmware Architecture



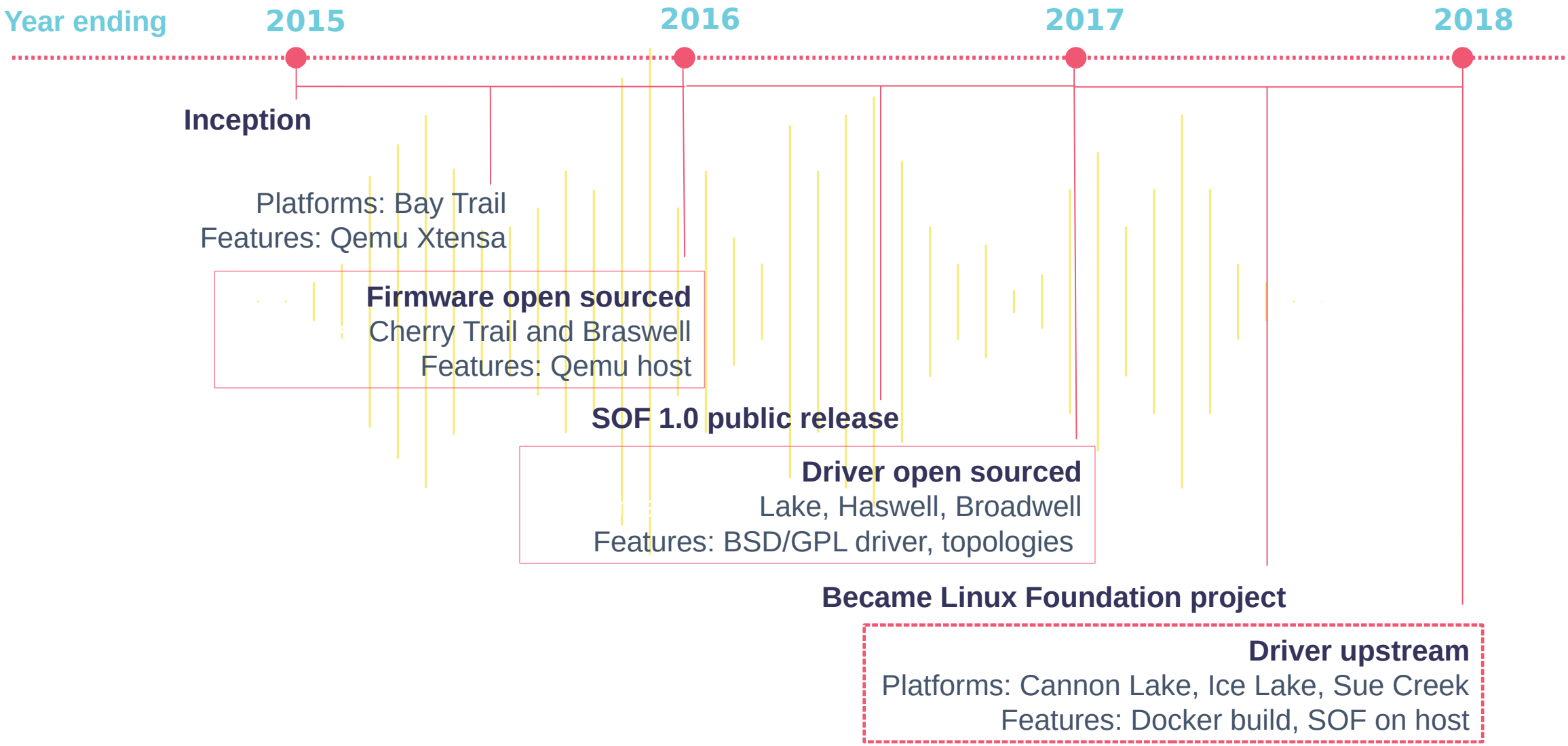
2 of 2 Firmware diagram – complete diagram

Driver Architecture

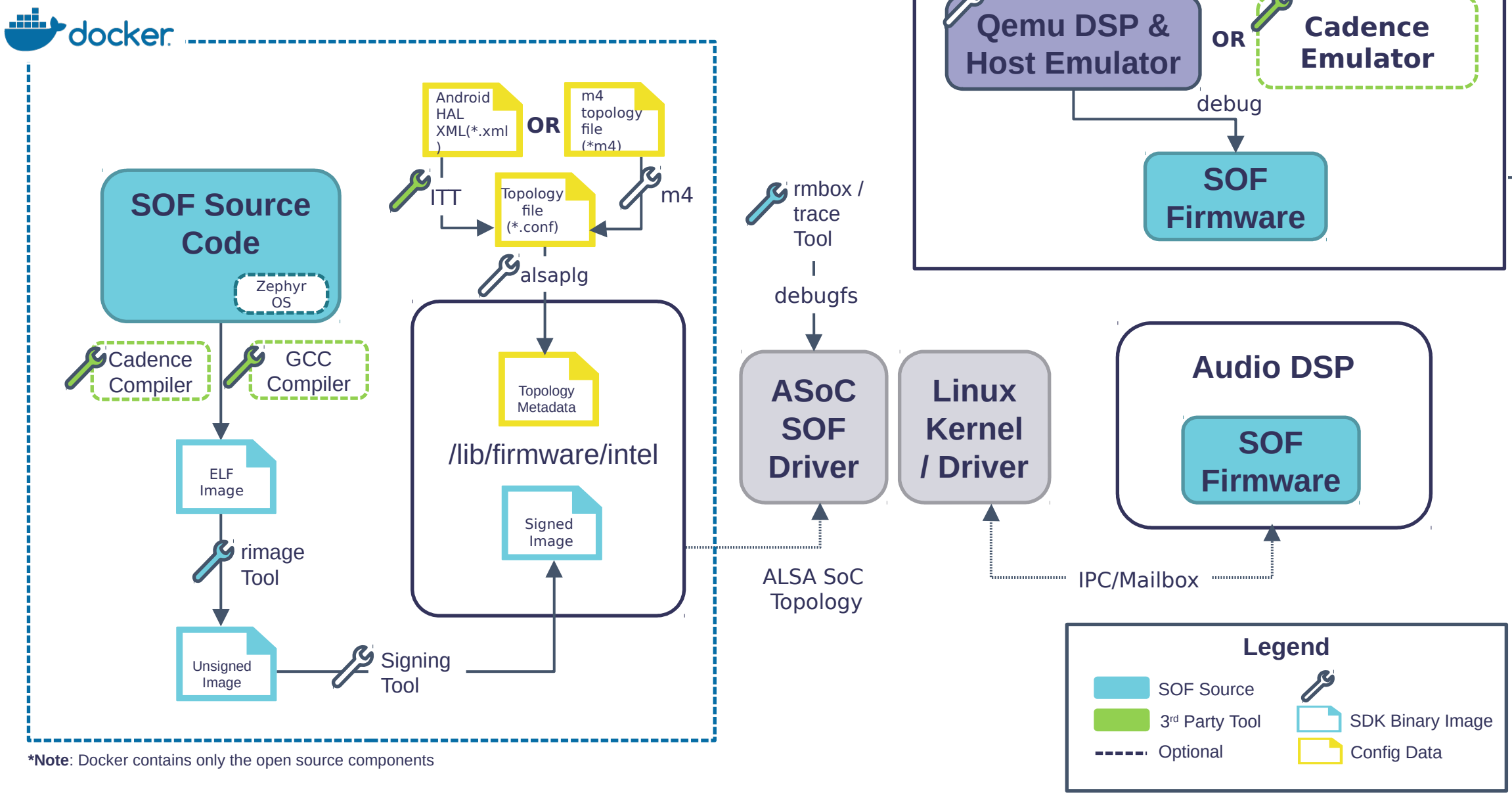


2 of 2 Driver diagram – complete diagram

History


Year ending
2015
2016
2017
2018
Inception
Platforms: Bay Trail
Features: Qemu Xtensa
Firmware open sourced
Cherry Trail and Braswell
Features: Qemu host
SOF 1.0 public release
Driver open sourced
Lake, Haswell, Broadwell
Features: BSD/GPL driver, topologies
Became Linux Foundation project
Driver upstream
Platforms: Cannon Lake, Ice Lake, Sue Creek
Features: Docker build, SOF on host

Development Kit



***Note**: Docker contains only the open source components

SDK diagram – complete SDK

Core Pillars

Open source Community driven

Permissive[1]

- BSD/MIT licensed firmware and topologies
- BSD/GPL dual licensed driver
- Firmware code changes can be private or up-streamed

Modular

- Configurable topology allows for flexible customizations
- Extendable using existing APIs to add custom or 3rd party binary modules
- Allows for custom ABIs between modules
- Develop and test out new modules before deploying onto target HW

Portable

- Platform, firmware OS, DSP architecture agnostic
- Portable to other host platform and DSP architectures
- Extendable to different real-time DSP OSes
- Modifiable for any custom integrations

Tool-rich

- BSD/GPL licensed and proprietary tools available
- Includes toolchain, debugger, emulator, scripts, firmware creation tools
- Configurable builds using supported GNU GCC or proprietary toolchains
- Virtualize both DSP and host OS via QEMU
- Virtualize trace data in real time

[1]: SDK tools are a mixture of BSD, GPL and proprietary licenses.

Thanks!

Q & A

Drop by the booth.

www.sofproject.com
https://github.com/thesofproject

Thank you