WorkStream: Silicon Interface Design

OPEN SOURCE FIRMWARE FOUNDATION

Subrata Banik (Google)
Christian Walter (9elements)

Problem Statement

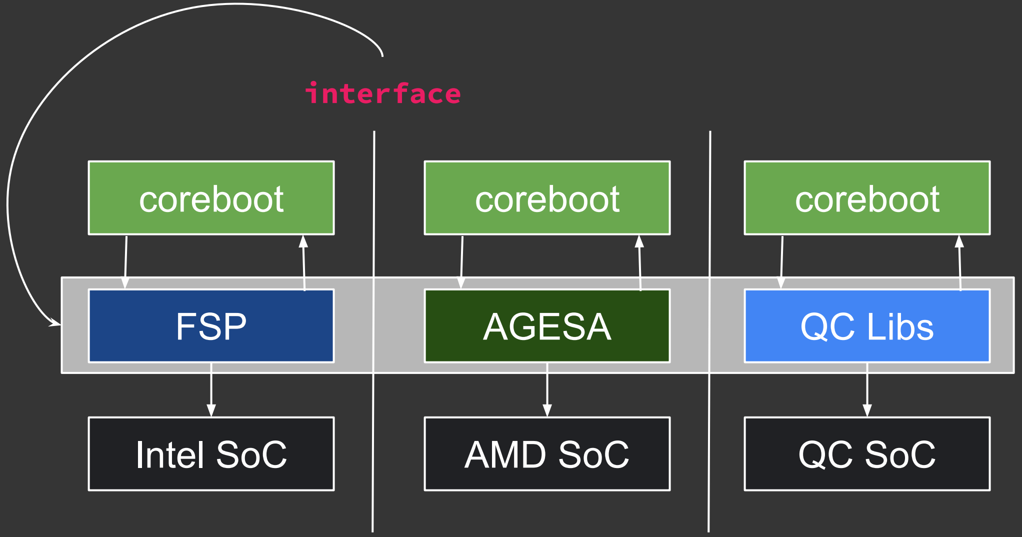
- Every SoC vendor has its own `interface`.

- Migrating the project across different SoC vendors is challenging due to lack of unified specification and reusability. Example: Great reusability of the SoC interface between Intel and AMD was using FSP.

- Even landing a new SoC platform means a lot of work. Example: UEFI is the default choice over coreboot due to lack of specification.

- Unable to provide unified user experience across different SoC platforms even with the same Operating System.

- Lack of framework that qualifies a system firmware healthy based on examining the underlying interface.



coreboot needs to keep various drivers and libraries to support different SoC interface

https://www.osfc.io/2022/talks/the-thing-around-your-system-firmware/

Objective

Create an Open Source friendly Silicon Reference Code Interface, applicable for all CPU architecture. The current model of Silicon Reference code is limited to the proprietary blobs used for the platform initialization. It increases redundancy while working with different SoCs, being a bottleneck to the SoC vendors for platform enablement, and expanding the usage of the close source increases the common mistakes and pitfalls across all silicon reference code.

Scope

To accelerate the platform enablement using Open Source Firmware (OSF) development approach, where working with different silicon reference codes is seamless, easy to leverage among existing designs without the need to start the designing/development from the scratch.

Furthermore, we want to define the trust using a transparent development approach where engagement is wider, firmware development is easy, increase reusability, being royalty free and managed by the open source community.

Agenda #1 - Design Unified Silicon Interface (USI)

- To design a **unified** `interface` for communicating with silicon reference code.

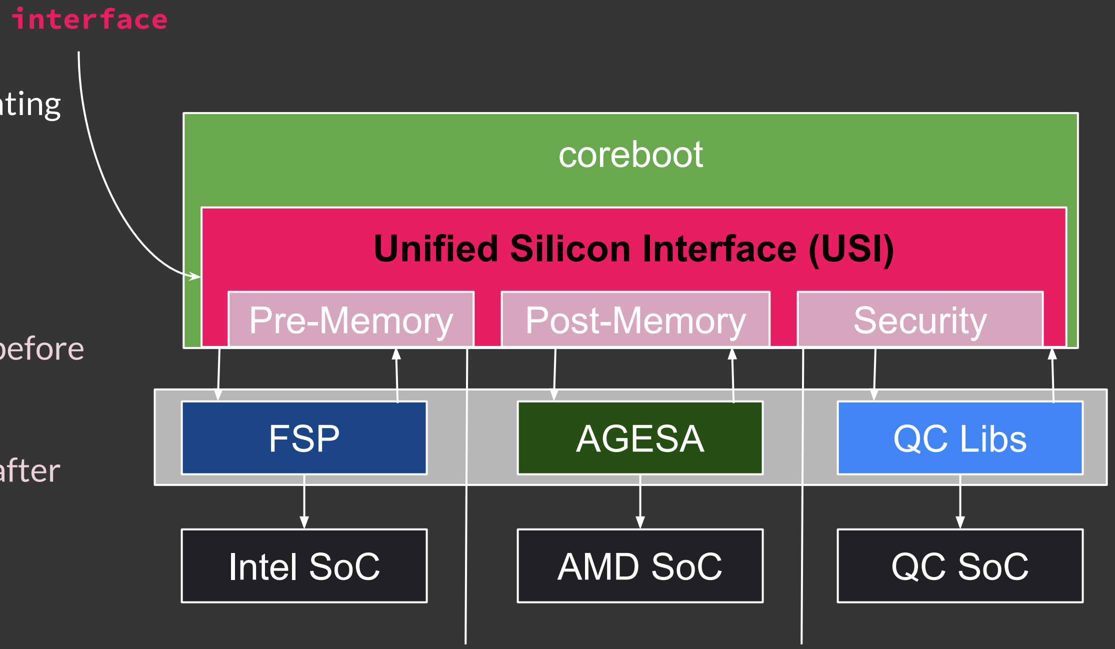
- Simple, Reduced, Generic and UEFI-neutral API-based communication model[1].

> **Pre-Memory:** To perform any operation(s) before DIMM initialization.

> **Post-Memory:** To perform any operation(s) after DIMM initialization.

> **Security:** To perform any SoC recommended operation(s) before loading payload.

- *Shared Memory Type* IPC for exchanging[2] the information between USI and silicon reference layer.



coreboot to have only one generic driver to communicate with SoC interface

[1] Need to have more granular details about this API communication

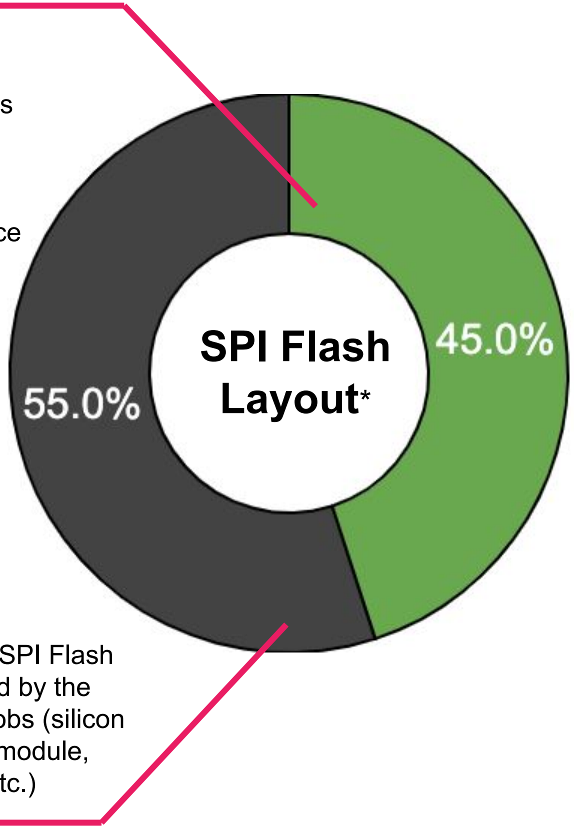
Agenda #2 - Open Source Silicon Reference Code

- Due to the restricted nature of Silicon Reference Code, the code visibility is zero, the growth in it is untraceable and as expected it's beyond the community control.

- Improve the state of OSF Development using Open Source Silicon Reference Code[1].

- Open Source development provides visibility where the community can even contribute into product development and bug fixes (without any additional cost).

- Provide great transparency where the goodness reaches to everyone without any discrimination. It might help the future product development.



Open Source

Remaining ~45% is open source boot firmware aka coreboot. None of the silicon reference code has open source visibility.

Closed Source

Around ~55% of SPI Flash layout is occupied by the closed source blobs (silicon ref code aka init module, pre-reset blobs etc.)

[1] https://blog.osfw.foundation/osf-intel-reduce-fsp-boundary/

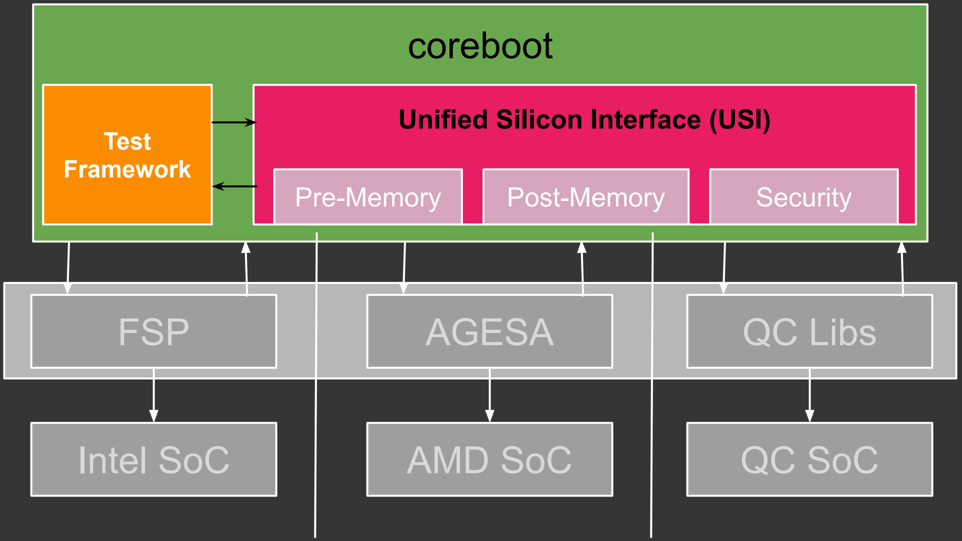
Agenda #3 - Design Test Framework around USI

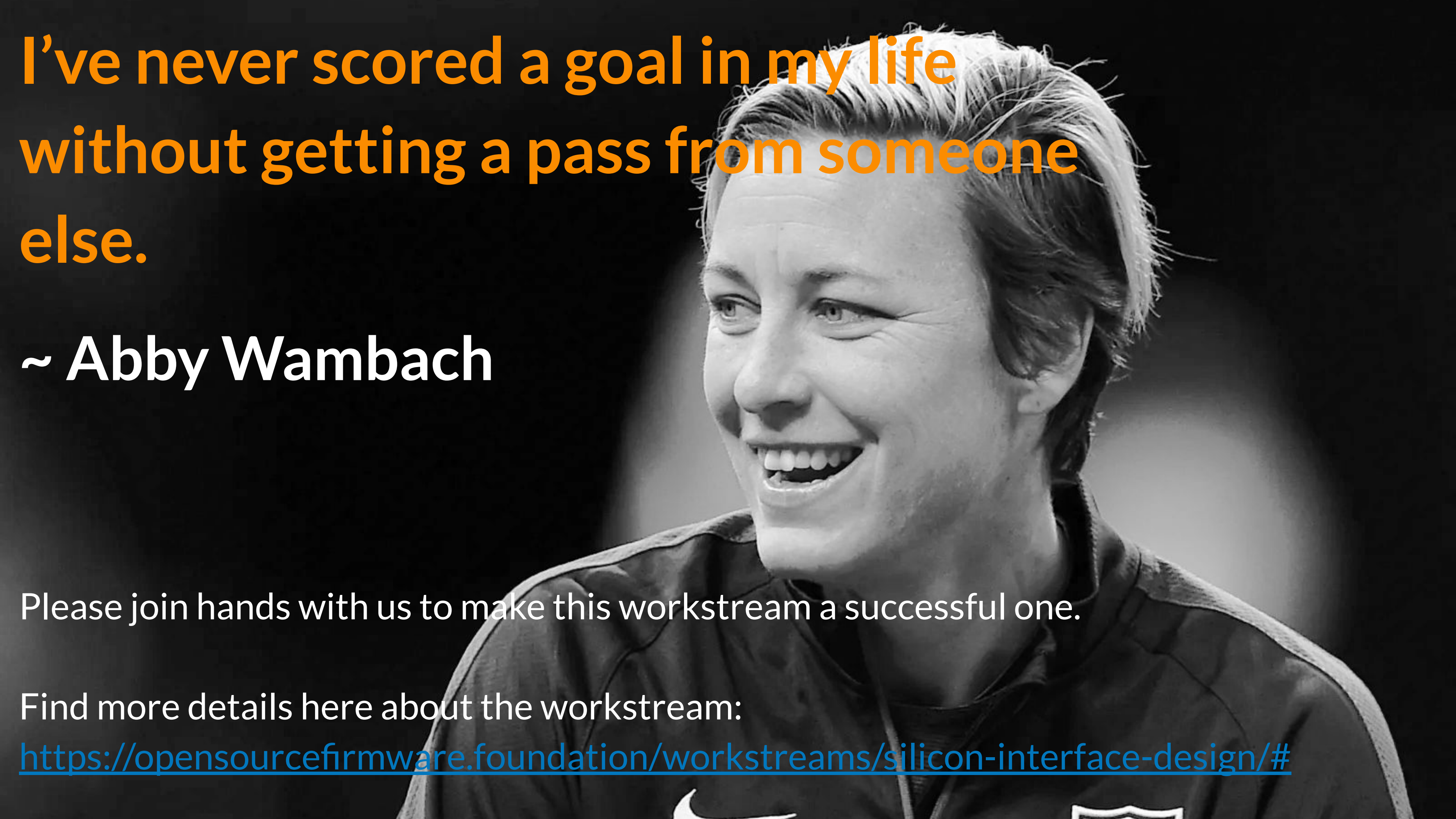
- The lack of a test framework that defines the completeness of the silicon programming can be costly if vulnerabilities are found during post production.

- Difficult for the product team to pay attention towards meeting firmware compliance . A defined standard test procedure can be used to certify the firmware completeness and robustness. For example: CTS for Android, WHCK for Windows, SCT for UEFI are meant to define that trust in the product quality.

- Due to the lack of a unified test framework, each product designer needs to define their own which results in redundancy and increased product cost.

A Test Framework around USI would help to bridge all these gaps and define a generic methodology to quality the open source system firmware.





I’ve never scored a goal in my life without getting a pass from someone else.

~ Abby Wambach

Please join hands with us to make this workstream a successful one.

Find more details here about the workstream: https://opensourcefirmware.foundation/workstreams/silicon-interface-design/#