OCP Open Platform Firmware Charter

The Open Platform Firmware Project aims to create an open firmware stack for all OCP components, beginning with but not limited to servers, including but not limited to the system firmware. We believe this to be an opportunity for OCP members to collaborate and accelerate innovation in OCP hardware design and support while improving testing capabilities and increasing quality of OCP products. We will work with vendors to enable Open Platform Firmware (OPF) as a standard configuration. In the longer term, we want the “OCP Accepted™” label to require OPF.

OPF, as its name implies, is intended to be released as open source. This is not, and may never be, possible for some OCP systems. We therefore define an exception process for binary blobs that allows their incorporation into OPF (see Appendix A).

Overview

The overall mission of the OCP Open Platform Firmware is to lead a sea change in the firmware used on OCP hardware, to one in which most of the code is available as open source; easily downloaded; and easily built. The benefits from this change are many: we can build servers which boot quickly; are much easier to customize; much more amenable to use in the circular economy; and, not least, more secure. In the case full open source implementation couldn’t be provided, at least an Open API to interface to the firmware will be a strong requirement, as to ensure interoperability. .
The Open Platform Firmware project aims to facilitate and enable OCP vendors in the creation of OPF-based hardware. The goal is to define a comprehensive architecture and guidance to result in the extension of the “OCP Accepted” label to *require* that the systems be delivered with OPF as the default firmware. The OPF group will work with vendors and OCP on this issue.

This work covers a wide range from technologies and solutions, including:

- **Chip architecture**: Chip vendors will need to ensure that the owner can program their firmware of choice. This might impact how they use one-time programmable fuses, for example.
- **Board vendors**: Board vendors will need to realize that more of their board designs are visible because firmware is now open. For one example, they will need to correctly document which GPIOs perform which function and, further, rigorously manage their SKU numbers so that boards with different connections are different SKUs.
- **Silicon vendors who supply firmware**: These include, e.g., Intel, IBM Power, AMD, and silicon providers using Arm and RISC-V CPU architectures. These companies should make a practice of opening as much of their chip-specific firmware as possible.
- **Third party independent Firmware vendors (IFV)**: We would like the IFVs to engage in this effort. Their expertise is valuable and they are capable of contributing much to this effort. *We believe IFVs have an important place in OPF.*

- **BMC vendors and solution providers**: We would like the BMC vendors and solution providers to make a practice of opening as many of their solutions as possible.
- **Peripheral device vendors**: We would like the vendors to make a practice of opening as much as their device firmware as possible.

Sub-Projects

Because of the breadth of activities that the OCP Open Platform Firmware covers, there may be formal sub-projects for the OCP Open Platform Firmware. These will most typically anchor around a particular contribution, focus area, or common firmware/software packages. These sub-projects will have formal sub-project leads, periodic meetings and notes, and report back to the overall project periodically for status.

In-Scope Activities

The OCP principles of Efficiency, Scale, Openness, Impact and Sustainability provide guidance for which projects will be in scope for the Open Platform Firmware.

Example areas to consider in-scope include, but are not limited to:

- Server host/SoC system firmware
- Hardware authentication devices
- Switchgear firmware
- Storage, GPU, DPU or ASIC Appliance firmware
- Utilities (e.g. remote FW update)
- Security and trust of of system firmware
- Support multi operating systems (Linux and Windows)
- Testing frameworks and unit tests

Out-of-Scope Activities

By definition, any activity that doesn't meet the principles of efficiency, scale, openness, impact and sustainability will be considered out-of-scope.

We recognize that OCP members have built unique infrastructures with different requirements, automation, tooling, etc. Therefore this project does not intend to dictate how an OCP device must boot or support hardware architecture beyond what is necessary to enable customer choice. Instead we will enable hardware from all OCP vendors and provide users with buildable and usable firmware code that they can curate for their needs. We believe in providing options and mechanisms, not dictating policies, and enabling users to decide what is best for their infrastructure.

Collaborative Activities

Further, there are areas that overlap with other projects within or outside of OCP, and we will collaborate with those projects as needed to reduce overlap.

- RMC software - an emerging effort to open source the Rack Manager Controller (e.g. RMC) applications. The RMC is a hardware management aggregator that may also use or share features with BMC software and OPF software.
- Security - we will work very closely with the security project as there are so many overlapping areas of concern.
- BMC software - there is an open source effort for BMC in Linux Foundation (OpenBMC). BMC software communicates with the server system firmware to manage the servers. There may be overlapping areas of concern that we may need to collaborate with the OpenBMC project and the BMC vendors and solution providers.

Appendix A: Exception Process for Binary Blobs

While it is our goal to have fully open-source firmware, we plan to replace binary blobs with open source code wherever possible. We recognize the present essential nature of binary blobs, especially during the earliest stages of boot. Therefore we must define a controlled process to include binary blobs through an explicit exceptions process. This allows us to maintain control of binary blobs while not hindering our efforts to strive for open replacements. Binary blobs will be provided by a web site or repo with no limit on redistribution (as, e.g., microcode blobs are today).