Design a System that "Just Works"

arm

The Arm ServerReady Program and Beyond

November 2019

White Paper

Contents

Executive Summary

This whitepaper describes the ServerReady program, certification process, and Arm's standards-based approach to system development. This approach is not limited to servers, it is suitable for other use cases, including edge and client devices, if standard operating systems are used. The intended audience is Arm-based processors and systems developers and integrators. By reading this paper, you will learn about Arm's standards-based approach for development, and how to design and certify systems with the Arm ServerReady program.

1. Introduction

In order to design a system that "just works" for the end user, with the ability to install and run standard operating systems (e.g. Windows, RHEL, VMWare, etc...) out-of-the-box, a set of minimum hardware and firmware requirements must be followed.
For the Arm Ecosystem, such a need first surfaced in the server segment. The Arm ServerReady compliance program provides this "just works" solution for servers, enabling partners to deploy Arm servers with confidence. The program is based on industry standards and is accompanied by a compliance test suite, as well as a process for certification.

Not all servers need to support standard operating systems. For example, some cloud service providers only support their own Linux operating systems. However, a server that is compliant to the Arm ServerReady program would require less customization done to the operating systems.

The same is true for other segments. When standard operating systems support is needed for edge or client PC systems, the systems need to follow the standards-based approach similar to (if not exactly the same as) the Arm ServerReady program.

This whitepaper describes the ServerReady program and Arm's standards-based approach to system development. This paper also explains the certification process for partners to participate in this program.

2. Arm ServerReady Program

2.1 Background and History

The Arm ServerReady v1.0 certification was launched at Arm TechCon 2018, as part of the official launch of Neoverse CPU for Arm-based servers. Arm ServerReady is a compliance program that is based on standards, and that enables partners to deploy Arm servers with confidence. The program complements the earlier release of the Server Base System Architecture (SBSA) and Server Base Boot Requirement (SBBR) specifications, alongside Arm's Enterprise Architectural Compliance Suite (ACS). Partners can run a test suite based on the ACS that enables them to check their systems are ServerReady.

Arm ServerReady establishes the minimum hardware and firmware requirements

described in SBSA and SBBR. It depends on the ACS, which tests the system against these requirements. This enables the value chain to buy compliant products that are compatible and interoperable and have a wide range of choice from the global Arm partner ecosystem.

Arm ServerReady ensures that Arm-based servers work out-of-the-box, offering seamless interoperability with operating systems. Compliant systems that adhere to the Arm ServerReady Terms and Conditions will be issued with a compliance certificate.

The program involves:

A set of compliance tests

- These cover the Arm standard specifications, namely the SBSA and SBBR.
- It also includes booting standard operating systems and collecting boot logs.

A certification process

- Partners run the compliance tests themselves or with help from the Arm support team.
- Arm helps with debugging issues discovered.
- Certification is provided after successful completion of the tests.

Marketing material

- Partners can use the ServerReady logo (Figure 1) as part of their product promotion, once they pass the certification process. We also host partner logos on the Arm Developer website at https://developer.arm.com/architectures/platform-design/server-systems/serverready-partners-and-supporters

Figure 1: Arm ServerReady 1.0 Logo

For more information on the Arm ServerReady program, and a current list of partners and supporters, visit the Arm ServerReady webpage at https://developer.arm.com/architectures/platform-design/server-systems

2.2 Arm Standards-Based Approach

Arm architecture supports a diverse ecosystem of devices, ranging from embedded and mobile devices, to automotive and industrial components, and infrastructure servers. The infrastructure space is also diverse, covering IoT gateways, edge servers, and cloud and enterprise datacenters. While diversity is good, uncontrolled diversity, particularly for servers, is problematic.

Servers are different from embedded devices. They are open systems that need to be deployed, configured, and managed in heterogeneous environments along with other servers and IT equipment. The server market is typically horizontal, where no single vendor

owns the entire stack from hardware to operating system. Servers require support for off-the-shelf enterprise operating systems. Server customers are used to an 'out-of-the-box' experience, where they can take the server out of the shipping box and deploy an OS easily. This often includes OSes that pre-date the silicon (such as support for current and previous versions of an OS). In this space, customizing the OS to work with a specific silicon is typically not a viable option.

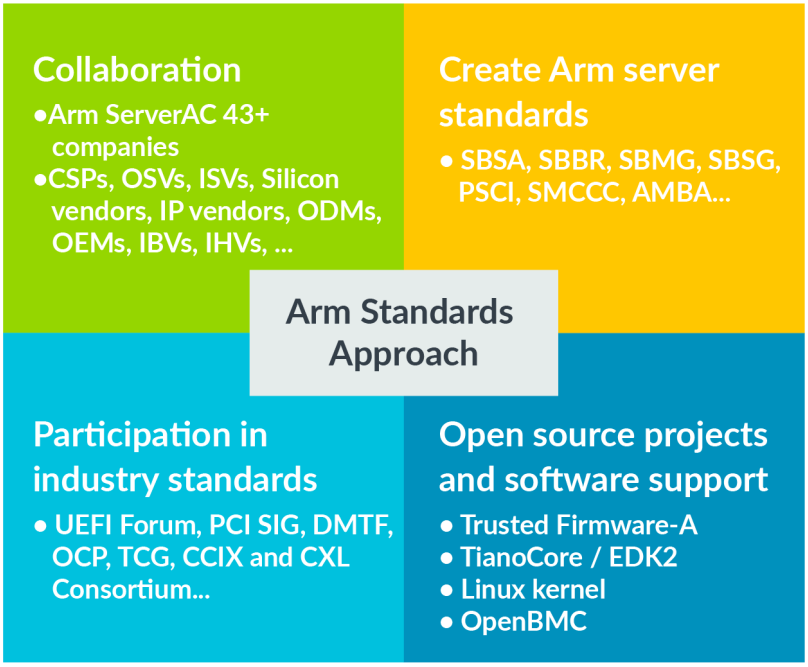
Industry standards offer common rules for hardware and firmware to enable supporting these standard operating systems. They establish a contract among the ecosystem players for a baseline framework of interoperability. Participants in the ecosystem are expected to adhere to this common baseline, while still being able to innovate on top of it to differentiate and add value in their products to their customers.

This standards-based foundation for systems design is critical to the success of the data center, edge, and cloud deployment. However, this approach is not limited to these deployment scenarios, and could be extended and adopted in other segments if supporting standard operating systems is required.

The Arm standards-based approach focuses on four aspects (Figure 2):

1. Collaboration: Arm uses a collaborative process, involving companies across the Arm server ecosystem. Collectively, these companies form the Arm Server Advisory Committee (ServerAC), with members from OEMs, ODMs, silicon providers (SiPs), OS vendors (OSVs), software and firmware vendors (ISVs and IBVs), cloud service providers (CSPs), and other hardware and IP vendors.
2. Create Arm Standards, including the SBSA and SBBR, in addition to other standards such as the Power State Coordination Interface (PSCI), the Secure Monitor Call Calling Convention (SMCCC), and AMBA. AMBA is a freely-available, open standard for on-chip connection and management of functional blocks in an SoC.
3. Participate in Industry Standards: Arm actively participates and provides leadership in various industry standards bodies and groups, ensuring the compatibility of these standards with the Arm architecture. This includes the UEFI Forum, PCI SIG, DMTF, OCP, TCG, CCIX Consortium and CXL Consortium, among others.
4. Open Source Software support: Arm supports open source software and

Figure 2: Standards-based approach



3. Arm Server Advisory Committee (ServerAC)

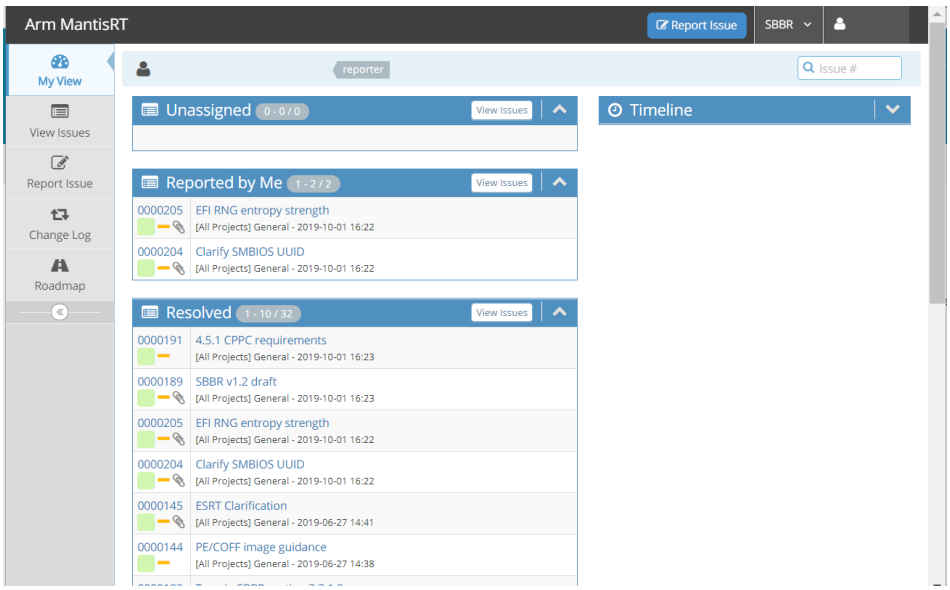
The Arm ecosystem collaborates on the creation of standards for servers through the Arm Server Advisory Committee (ServerAC). This group contains companies across the server ecosystem including OEMs, ODMs, silicon providers (SiPs), OS vendors (OSVs), software and firmware vendors (ISVs and IBVs), cloud service providers (CSPs), and other hardware and IP vendors. The committee communicates in various ways including e-mail, regular conference calls, and biannual gatherings. To communicate with this group, e-mail can be sent to:

- The community by sending a mail to armserverac-discuss@arm.com
- Privately to Arm by sending a mail armserverac-request@arm.com

3.1 ServerAC Development Process

The ServerAC uses Mantis Issue tracker (Figure 3) for developing the specifications and tracking issues, proposals, and requests. The database is accessible to ServerAC member companies at https://atg-mantis.arm.com. Any ServerAC member can raise a request. Issues are discussed in the ServerAC meetings. Once agreed upon, the changes get integrated into the specification.

Figure 3: ServerAC Mantis Database



The ServerAC development process is described below:

1. The process often starts with private discussions between a partner and Arm, raising an issue or requesting a change. However, the discussion could also start in public conversation between the Arm ServerAC community members.
2. An engineering change request (ECR) is submitted on the ServerAC mantis against one of the supported specifications. The ECR includes a problem statement, justification for change, and description of the requirement. It eventually gets updated to include a proposal for a spec change, which is driven by Arm or other ServerAC members.
3. The ECR is discussed within the ServerAC community, and changes are made based on collected feedback from all members.
4. Once the ServerAC approves the ECR, Arm integrates the change in the specification, and makes it available in the next revision.

4. Arm Server Standards

4.1 Server Base System Architecture (SBSA)

Arm architecture covers a wide range of products, across many market segments, from embedded control, to mobile, to servers. Base System Architectures (BSA) provide hardware requirements for a given type of product or market segment. The requirements are intended to ensure standard software, or operating systems, will operate correctly on machines compliant with the BSA.

The Server Base System Architecture (SBSA) is the BSA for servers. SBSA offers several compliance levels. Each level corresponds to a set of requirements on:

- Processor features
- Memory subsystem features
- GIC and SMMU revision features
- PCIe integration features
- Base levels for other peripherals (USB, SATA)
- Security features
- Power semantics

Levels represent the advancement of the specification over time - as new architectures arrive, new levels are added. Generally, each level includes all the requirements of a previous level. For example, SBSA compliance Level 5 was added to correspond to new features in Armv8.4-A architecture, while SBSA level 6 corresponds to changes in Armv8.5-A.

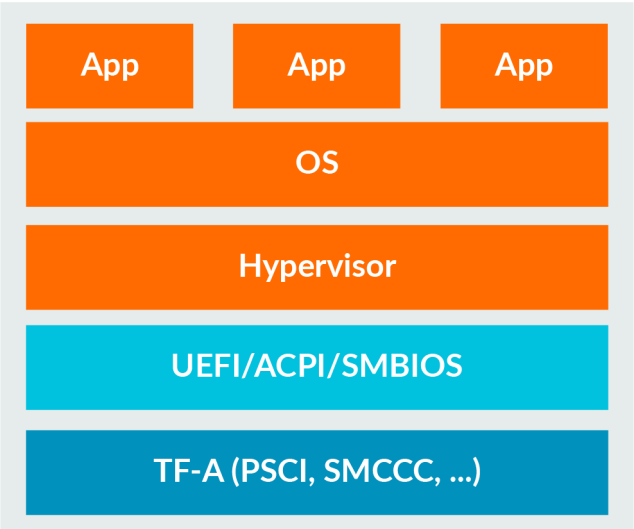
4.2 Server Base Boot Requirements (SBBR)

Operating systems running on standard server hardware require standard firmware interfaces to be present in order to boot and function correctly. The Server Base Boot Requirements (SBBR) document describes these firmware requirements for Arm based infrastructure SoCs. Together with SBSA, the SBBR provides a standards-based approach to building Arm servers and their firmware.

The SBBR covers industry standards as well as standards specific to Arm, as illustrated in Figure 5. Specifically, SBBR includes requirements that cover:

- UEFI Specification
- ACPI Specification
- SMBIOS Specification
- Arm Specifications (PSCI, SMCCC, TF-A, etc...)

Figure 5: SBBR firmware standards



SBBR is intended to offer interoperability with standards-based open source firmware projects for reference implementations, including:

- TianoCore EDK2 (https://www.tianocore.org/)
- Trusted Firmware-A (https://www.trustedfirmware.org/)

4.3 Server Base Security Guide (SBSG)

Platform security is increasingly important for server systems. The Server Base Security Guide (SBSG) specifically addresses the platform security of Armv8 servers based on the Arm SBSA and SBBR standards. It provides requirements and guidance that support maintaining the integrity of the platform layer of Arm servers and supports secure attesting to the platform's state of integrity. The platform forms the foundation of a system, and thus the integrity of the platform is essential to the overall integrity of the system. If any hardware or firmware (code or data) component is compromised, the security (confidentiality, integrity, availability) of the entire system might be compromised.

The SBSG specifies requirements and guidance for both firmware and hardware. Key areas of guidance that facilitate platform integrity include:

- Protection of firmware and critical data. Mutable firmware and critical data must be updatable, with updates being authorized and verified.
- Detection of corruption of firmware and critical data. All mutable firmware and critical data must be signed so that it can be verified during boot, forming a chain-of-trust rooted in an immutable hardware root-of-trust.
- First instruction integrity. The first mutable firmware executed on the host SoC or other system component must be authenticated before use by an immutable bootloader.
- TPM 2.0 integration and TPM-based measured boot, providing a means to securely attest to the state of the platform.
- UEFI security guidance.

- Hardware requirements to facilitate implementing the firmware-based security requirements.
- Hardware requirements for secure memory.

4.4 Server Base Manageability Requirements (SBMR)

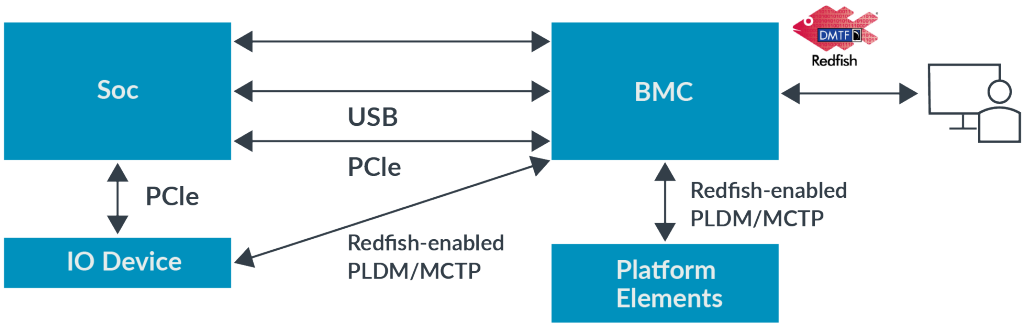
Server system management requires a foundation of standardized common capabilities that meet basic user expectations for servers. Products can still differentiate with functionality that is built on top of the standardized baseline, and that offers true value-add to the end-user. The Server Base Manageability Requirements (SBMR) is a new standard from Arm that provides a path for Arm servers to be interoperable with the prevalent industry standard system management specifications. The specification is under development in collaboration with the Arm ServerAC industry partners and community members.

The SBMR specification covers usage of manageability standards that are illustrated in Figure 6, which includes:

- Redfish™ Interface
- DMTF Platform Level Data Model (PLDM)
- Management Component Transfer Protocol (MCTP)
- Intelligent Platform Monitoring Interface (IPMI)
- Relevant OCP manageability standards

Together with SBSA, SBBR, and SBSG, the SBMR provides a standards-based approach to building Arm servers, their firmware and their server management capabilities.

Figure 6: SBMG compliant server management



SBMR is developed with the intention to align with open source projects for reference implementations, including:

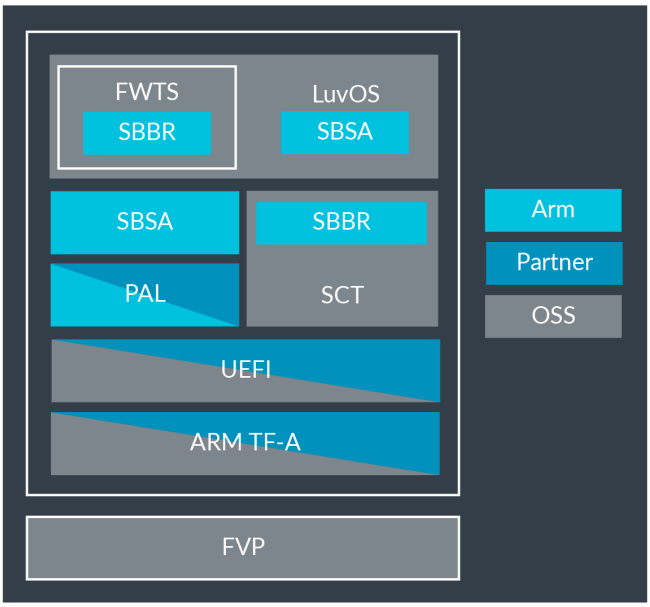
- OpenBMC (https://www.openbmc.org/)
- OpenRMC (https://www.opencompute.org/wiki/Hardware_Management/Open_RMC)

5. Architectural Compliance Suites

At Arm, we live by the statement “there is no specification without verification”. This is realised by the various Arm compliance programs that help developers to ensure their hardware is fully compliant with the Arm architecture specifications. In the same spirit, Arm has created the Architecture Compliance Suite (ACS) for the SBSA and SBBR. This compliance suite is an essential component of the Arm ServerReady program, and it covers:

- SBSA hardware requirements
  - SBSA CPU properties
  - SBSA defined system components
  - SBSA rules for PCIe integration
  - Rules based on PCIe specification
- SBBR defined FW requirements
  - Based on standard OS drivers with no quirks enabled
  - UEFI testing based on the UEFI SCT
  - ACPI testing based on FWTS
  - SMBIOS testing is intended to ensure standard software

Figure 7: ACS Test Suite Components



The test suites are hosted on GitHub and are open source, with a permissive Apachev2 license.

SBSA Architecture Compliance Suite (https://github.com/ARM-software/sbsa-acs)

SBSA Architecture Compliance Suite (ACS) is a collection of self-checking, portable C-based tests. This suite includes a set of examples of the invariant behaviours that are provided by the SBSA specification, so that implementers can verify if these behaviours have been interpreted correctly. Most of the tests are executed from UEFI Shell by executing the SBSA UEFI shell application. A few tests are executed by running the SBSA ACS Linux application which in turn depends on the SBSA ACS Linux kernel module.

Enterprise Architectural Compliance Suite (https://github.com/ARM-software/arm-enterprise-acs)

Arm Enterprise ACS includes a set of examples of the invariant behaviours that are provided by a set of specifications for enterprise systems (e.g. SBSA, SBBR, etc.), so that implementers can verify if these behaviours have been interpreted correctly.
ACS is delivered with tests in source form along with a build script, the output of the build being a bootable Linux UEFI Validation (LUV) OS image that can run all tests required by these specifications.

The latest ACS release is v2.3. ACS v1.6 corresponds to ServerReady v1.0. It is testing the compliance against SBSA version 3.1 and SBBR version 1.0. Since then, Arm has worked with partners to move SBSA to version 6.0 and SBBR to version 1.2. Arm also published Server Based Security Guide (SBSG) specification v1.0. In addition, Arm is working on Server Base Manageability Requirements (SBMR). It is expected that the future versions of the Arm ServerReady program will make use of these newer standards for the future server systems.

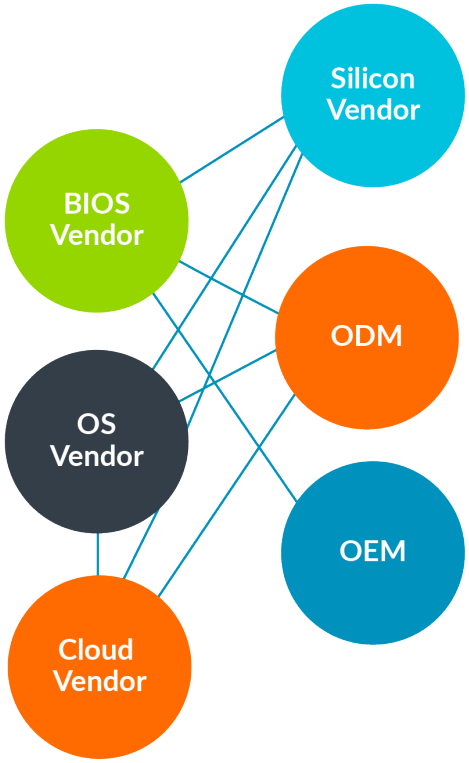
5.1 Arm ServerReady Players and Responsibilities

Like other segments, the server ecosystem is complex and contains multiple vendors (Figure 8)

- Not all players interact directly with each other
- An OS vendor maintains the OS to be compliant and up to date with standards. But an OS vendor cannot check every possible system
- Compliance tools can help one vendor check the input they receive from another
  - OEMs/ODMs can check if silicon vendor hardware is compliant with SBSA hardware requirements
  - OS vendors can check if OEMs/ODMs system is compliant with SBSA hardware requirements and SBBR firmware requirements.

Refer to Figures 9 and 10 for illustration of the Arm ServerReady roles and responsibilities and certification flow.

Figure 8: Arm Server Ecosystem Relationships



Arm is engaging with silicon vendors, ODMs, OEMs and BIOS vendors to run the ServerReady compliance suite tests. Tests are developed by our architecture team, which also develops the specifications in collaboration with the Arm ServerAC community. Arm has a support team in Taipei that helps in the running of tests and debugging issues. Arm also has a certification team that reviews test results and runs the certification process. Once tests are passed, Arm marketing teams grant the certificate and helps with communication.

Figure 9: Arm ServerReady roles and responsibilities

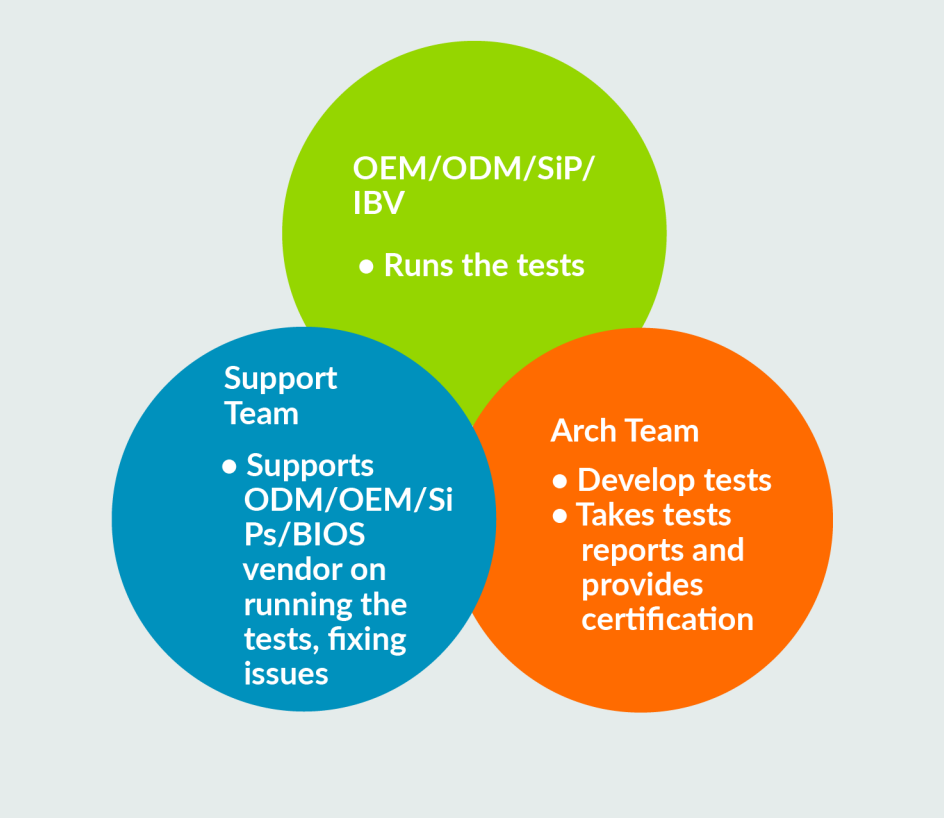
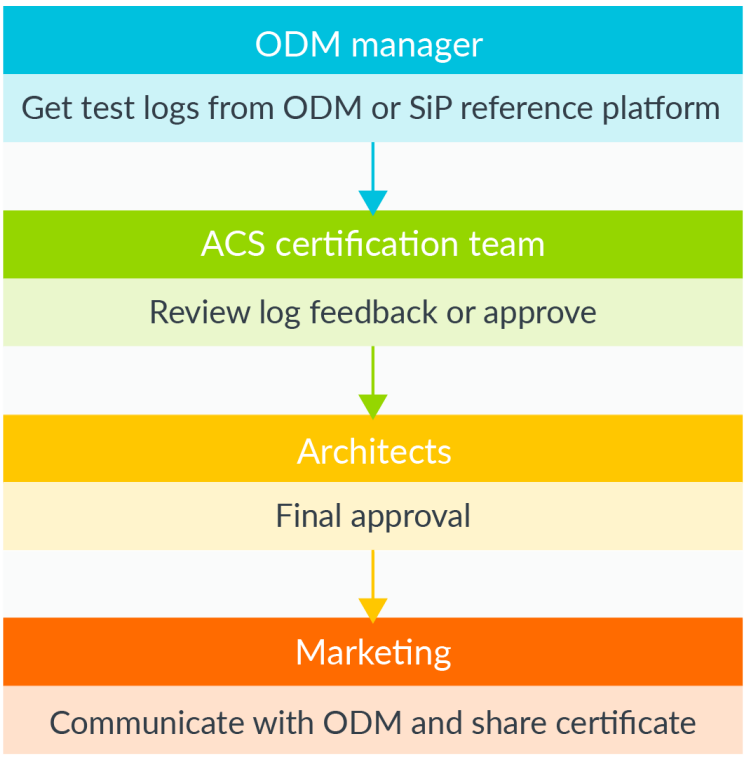


Figure 10: ServerReady certification flow



Conclusion

This whitepaper describes the Arm ServerReady program, its components and process. In addition, the paper describes the Arm standards-based approach and Arm specific standards that lay the foundation for the ServerReady program. A more detailed description of this can be found on the Arm ServerReady page at https://developer.arm.com/architectures/platform-design/server-systems.

Arm has teams in Taiwan, India, the US and the UK who can help you become ServerReady. We also collaborated with OpenGCC to launch ServerReady support in China, as explained in this blog. Please contact us at arm.serverreadyprog@arm.com for more information on joining.

What has been created in the Arm ServerReady program can be readily leveraged to other market segments, such as edge and client PC, if the systems need to "just work" with the standard operating systems. We look forward to working with the ecosystem to establish the necessary standards and certification process for these segments.

Glossary

This document uses the following terms and abbreviations.

Term	Meaning
ACS	Architectural Compliance Suite
BSA	Base System Architecture
SBBA	Server Base System Architecture
SBBR	Server Base Boot Requirements
SBSG	Server Base Security Guide
SBMR	Server Base Manageability Requirements
ECR	Engineering Change Request
TF-A	Trusted Firmware-A
RHEL	Red Hat Enterprise Linux
OEM	Original Equipment Manufacturer
ODM	Original Device Manufacturer
IHV	Indepenet Hardware Vendor
ISV	Independent Software Vendor
IBV	Indepenent BIOS (or Firmware) Vendor
OSV	Operating System Vendor
SiP	Silicon Provider
CSP	Cloud Service Provider
SoC	System-on-chip
UEFI	Unified Extensible Firmware Interface
ACPI	Advanced Configuration and Power Interface.
CXL	Compute Express Link
CCIX	Cache Coherent Interconnect for Accelerators
OCP	OpenCompute Project
EDK2	EFI Development Kit, ver 2.0
PLDM	Platform Level Data Model
BMC	Baseboard Management Controller
IPMI	Intelligent Platform Management Interface
MCTP	Management Component Transport Protocol

arm