# Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Executive Summary</td>
<td>5</td>
</tr>
<tr>
<td>1 Introduction</td>
<td>6</td>
</tr>
<tr>
<td>2 Arm System Architecture</td>
<td>7</td>
</tr>
<tr>
<td>2.1 Arm Standards-Based Approach</td>
<td>7</td>
</tr>
<tr>
<td>2.2 Base System Architecture (BSA) and its Supplements</td>
<td>9</td>
</tr>
<tr>
<td>2.3 Base Boot Requirements (BBR)</td>
<td>10</td>
</tr>
<tr>
<td>3 Architectural Compliance Suite</td>
<td>11</td>
</tr>
<tr>
<td>4 Arm SystemReady Program</td>
<td>12</td>
</tr>
<tr>
<td>4.1 Background and History</td>
<td>12</td>
</tr>
<tr>
<td>4.2 Arm ServerReady Stakeholders and Responsibilities</td>
<td>13</td>
</tr>
<tr>
<td>4.3 Arm SystemReady Program</td>
<td>14</td>
</tr>
<tr>
<td>5 SystemReady Opt-In Extensions</td>
<td>16</td>
</tr>
<tr>
<td>5.1 Pre-silicon Certification</td>
<td>16</td>
</tr>
<tr>
<td>5.2 Security Option</td>
<td>17</td>
</tr>
<tr>
<td>6 How To Use SystemReady</td>
<td>18</td>
</tr>
<tr>
<td>6.1 BSA Compliance</td>
<td>18</td>
</tr>
<tr>
<td>6.2 BBR Compliance</td>
<td>18</td>
</tr>
<tr>
<td>7 Examples</td>
<td>20</td>
</tr>
<tr>
<td>8 Conclusion</td>
<td>21</td>
</tr>
</tbody>
</table>
Arm Non-Confidential Document Licence ("Licence")

This Licence is a legal agreement between you and Arm Limited ("Arm") for the use of Arm’s intellectual property (including, without limitation, any copyright) embodied in the document accompanying this Licence ("Document"). Arm licenses its intellectual property in the Document to you on condition that you agree to the terms of this Licence. By using or copying the Document you indicate that you agree to be bound by the terms of this Licence.

"Subsidiary" means any company the majority of whose voting shares is now or hereafter owner or controlled, directly or indirectly, by you. A company shall be a Subsidiary only for the period during which such control exists.

This Document is NON-CONFIDENTIAL and any use by you and your Subsidiaries ("Licensee") is subject to the terms of this Licence between you and Arm.

Subject to the terms and conditions of this Licence, Arm hereby grants to Licensee under the intellectual property in the Document owned or controlled by Arm, a non-exclusive, non-transferable, non-sub-licensable, royalty-free, worldwide licence to:

(i) use and copy the Document for the purpose of designing and having designed products that comply with the Document;

(ii) manufacture and have manufactured products which have been created under the licence granted in (i) above; and

(iii) sell, supply and distribute products which have been created under the licence granted in (i) above.

Licensee hereby agrees that the licences granted above shall not extend to any portion or function of a product that is not itself compliant with part of the Document.

Except as expressly licensed above, Licensee acquires no right, title or interest in any Arm technology or any intellectual property embodied therein.
THE DOCUMENT IS PROVIDED "AS IS". ARM PROVIDES NO REPRESENTATIONS AND NO WARRANTIES, EXPRESS, IMPLIED OR STATUTORY, INCLUDING, WITHOUT LIMITATION, THE IMPLIED WARRANTIES OF MERCHANTABILITY, SATISFACTORY QUALITY, NON-INFRINGEMENT OR FITNESS FOR A PARTICULAR PURPOSE WITH RESPECT TO THE DOCUMENT. Arm may make changes to the Document at any time and without notice. For the avoidance of doubt, Arm makes no representation with respect to, and has undertaken no analysis to identify or understand the scope and content of, third party patents, copyrights, trade secrets, or other rights.

NOTWITHSTANDING ANYTHING TO THE CONTRARY CONTAINED IN THIS LICENCE, TO THE FULLEST EXTENT PERMITTED BY LAW, IN NO EVENT WILL ARM BE LIABLE FOR ANY DAMAGES, IN CONTRACT, TORT OR OTHERWISE, IN CONNECTION WITH THE SUBJECT MATTER OF THIS LICENCE (INCLUDING WITHOUT LIMITATION) (I) LICENSEE'S USE OF THE DOCUMENT; AND (II) THE IMPLEMENTATION OF THE DOCUMENT IN ANY PRODUCT CREATED BY LICENSEE UNDER THIS LICENCE. THE EXISTENCE OF MORE THAN ONE CLAIM OR SUIT WILL NOT ENLARGE OR EXTEND THE LIMIT. LICENSEE RELEASES ARM FROM ALL OBLIGATIONS, LIABILITY, CLAIMS OR DEMANDS IN EXCESS OF THIS LIMITATION.

This Licence shall remain in force until terminated by Licensee or by Arm. Without prejudice to any of its other rights, if Licensee is in breach of any of the terms and conditions of this Licence then Arm may terminate this Licence immediately upon giving written notice to Licensee. Licensee may terminate this Licence at any time. Upon termination of this Licence by Licensee or by Arm, Licensee shall stop using the Document and destroy all copies of the Document in its possession. Upon termination of this Licence, all terms shall survive except for the licence grants.

Any breach of this Licence by a Subsidiary shall entitle Arm to terminate this Licence as if you were the party in breach. Any termination of this Licence shall be effective in respect of all Subsidiaries. Any rights granted to any Subsidiary hereunder shall automatically terminate upon such Subsidiary ceasing to be a Subsidiary.

The Document consists solely of commercial items. Licensee shall be responsible for ensuring that any use, duplication or disclosure of the Document complies fully with any relevant export laws and regulations to assure that the Document or any portion thereof is not exported, directly or indirectly, in violation of such export laws.

This Licence may be translated into other languages for convenience, and Licensee agrees that if there is any conflict between the English version of this Licence and any translation, the terms of the English version of this Licence shall prevail.
Executive Summary

This white paper describes the Arm standards-based approach to system development, and the Arm SystemReady program with its certification process. Arm SystemReady is a set of standards and a compliance certification program that enables interoperability with generic off-the-shelf operating systems and hypervisors. You will see the term Just Works used throughout this white paper and in other Arm SystemReady materials. The term means enabling operating systems to work out of the box on Arm-based devices, through the adoption of standards and tools. Arm ServerReady offers a foundational framework that enables an ecosystem to focus on innovation and differentiation.

The intended audience for this paper includes SoC designers, platform software engineers, OEM and ODM system designers, OS vendors and end customer procurement teams.

By reading this paper, you will learn about the Arm standards-based approach for development, how to design and certify systems with the Arm SystemReady program, and how SystemReady can benefit your business and your customers.

The Arm SystemReady program is a set of tools that Arm is providing to the Arm ecosystem. Depending on the needs of the market, these tools can be used to:

- Enable the Just Works deployment of OS distributions to standards-compliant Arm-based platforms.
- Open-up markets for Arm-based silicon where OS portability and choice is a barrier to entry including the infrastructure and IoT edge.
- Reduce the cost for deploying software to an Arm-based platform, and the cost of maintaining that software over the device lifetime, by using common OS distributions that support industry standards.

SystemReady is a language that describes the requirements of an OS distribution, and a certification scheme for demonstrating whether a platform meets those requirements. We do not prescribe how SystemReady is used. Our partners determine what they need, which certification is appropriate to opt in to, and how to engage with the standards process.

Arm SystemReady targets the cloud and data center, embedded-server, and high-performance IoT devices.
Introduction

Systems that are designed to Just Work for the end user – with the ability to install and run generic off-the-shelf operating systems out of the box – must follow a set of minimum hardware and firmware requirements.

For the Arm ecosystem, this need first surfaced in the server segment. The Arm ServerReady compliance certification program provides this Just Works solution for servers, allowing partners to deploy Arm servers with confidence. The program is based on industry standards and is accompanied by a compliance test suite, and a process for certification.

The embedded-server and high-performance IoT segments are different from cloud and data center. This is because the operating system is often provided with the device and different standards may be used. However, a standard hardware and firmware platform, with a certification scheme, also brings benefits to the embedded-server and IoT ecosystem. Platforms that are certified under this program are guaranteed to implement a minimum set of hardware and firmware features that an operating system can depend on for deploying the OS image. If you choose SystemReady-compliant platforms, you can:

- Significantly reduce the cost associated with adopting a new platform by eliminating custom engineering associated with firmware
- Use industry standard management tools
- Enable OS vendors to provide direct support without the need for a custom kernel build

With this program, an OEM can amortize their OS engineering work across multiple platforms. ISVs can build new businesses around a standard distribution with support and security updates. System integrators can buy certified platforms with confidence knowing that they can run their chosen OS.

Arm SystemReady takes what the Arm ecosystem has learned from Arm ServerReady and applies it to a broader set of devices in the embedded-server and high-performance IoT ecosystem.

This whitepaper describes the Arm SystemReady program and the Arm standards-based approach to system development. The paper also explains the certification process for partners who want to participate in this program.
Arm System Architecture

2.1 Arm standards-based approach

The Arm architecture supports a diverse ecosystem of devices. The infrastructure and IoT space is also diverse, covering IoT gateways, embedded servers, and cloud and enterprise data centers. While diversity is evidence of a healthy ecosystem, unnecessary divergence on Arm-based silicon creates barriers to OS and application portability. Open systems must be deployed, configured, and managed in heterogenous environments. These markets are typically horizontal, where no single vendor owns the entire stack from hardware to operating system. These open systems require support for generic off-the-shelf operating systems. Customers are used to an out-of-the-box experience, where they can take the system out of the shipping box and deploy the OS easily. Or, in the IoT and edge context, can use a zero touch method to provision an OS and application remotely to a machine that is shipped directly from manufacturer to installation point. This zero touch method often includes OSes that pre-date the silicon, like supporting current and previous versions of an OS. In this space, customizing the OS to work with specific silicon is not usually a viable option.

To unlock value from the embedded-server and high-performance IoT segments, a seamless software experience, which includes an extension of a cloud-native model, becomes essential.

Beyond the operational expense savings that are associated with early enterprise IoT deployments, the focus is expected to shift towards designing more revenue streams, based on application service monetization models.
To support these IoT models, the experience of working with systems is frictionless: from landing the end-user choice of operating system or distribution, all the way to working with cloud native stacks managing hundreds of thousands of IoT devices and low-power gateways needs to be frictionless.

In the cloud and data center markets, both the industry and the Arm ecosystem have come together in support of the Arm ServerReady program, ensuring server-grade silicon and systems adhere to the standards that make these platforms general purpose in nature. With the introduction of the Arm SystemReady program, we extend this vision beyond cloud data centers, bringing about a similar working experience to the spectrum of infrastructure for the embedded-servers and high-performance IoT. This includes, for example, systems that are deployed outside of cooled environments, on factory floors, smart buildings and transport infrastructure, and wireline and wireless networking security and storage gear.

Industry standards offer common rules for hardware and firmware, to enable supporting these generic off-the-shelf operating systems. These rules 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 innovating on top of it to differentiate and add value in their products for their customers.

This standards-based foundation for systems design is critical to the success of the data center and embedded-server and IoT gateway deployment.

The Arm standards-based approach focuses on four aspects:

- **Collaboration:** Arm uses a collaborative process, involving companies across the Arm ecosystem. Collectively, these companies form Advisory Committees, with members from OEMs, ODMs, silicon providers (SiPs), OS vendors (OSVs), and software and firmware vendors (ISVs and IBVs), cloud service providers (CSPs) and other hardware and IP vendors.

- **Creation of Arm standards:** Including the Base System Architecture (BSA) and its market-segment supplements as well as BBR (Base Boot Requirements), in addition to other standards such as the Power State Coordination Interface (PSCI), and the Secure Monitor Call Convention SMCCC.

- **Participation 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, for example, the UEFI Forum, PCI SIG, DMTF, OCP, TCG, CCIX Consortium and CXL Consortium.

- **Supporting open source software:** Arm supports open-source software and firmware projects that help accelerate the development and adoption of Arm standards. This includes Trusted Firmware, TianoCore, UBoot, LinuxBoot, and Linux kernel.
2.2 Base System Architecture (BSA) and its Supplements

The Base System Architecture (BSA) document specifies a minimal set of CPU and system architecture requirements that are necessary for an OS to boot and run, regardless of the market segments. It includes baseline 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

The requirements are intended to ensure that standard software, or operating systems, operate correctly on machines that are compliant with the BSA.

The server supplement of the BSA is the Server BSA (SBSA). SBSA offers several compliance levels. The different levels represent the advancement of the specification over time. When 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, and Level 6 corresponds to changes in Armv8.5-A architecture.

**Note:** The BSA Specification and its SBSA supplement together can be considered equivalent to the previous monolithic SBSA Specification.

Other supplements of the BSA (xBSAs) may be developed for different market segments. In the server segment, the xBSA is the SBSA supplement. These are deliberately modular to allow partners to use the relevant specifications for the market segment that they are designing for.

The following diagram shows the relationships between the BSA Specification and the market-specific supplements:
2.3 Base Boot Requirements (BBR)

The Base Boot Requirements (BBR) document specifies firmware interface requirements that system software, like operating systems and hypervisors, can rely on. Firmware interface requirements include:

- UEFI specification
- ACPI specification
- SMBIOS specification
- DeviceTree specification
- Other Arm specifications (PSCI, SMCCC)

BBR also provides recipes tailored to the various generic operating systems:

- SBBR recipe
  - Requirements based on the previous SBBR specification
  - PSCI, SMCCC, UEFI, ACPI, SMBIOS
- EBBR recipe
  - PSCI, SMCCC, UEFI, SMBIOS
- LBBR recipe
  - PSCI, SMCCC, LinuxBoot, ACPI, SMBIOS

Other recipes may be developed.
Architectural Compliance Suites

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

- **SBSA (and its supplements) hardware requirements**
  - BSA CPU properties
  - BSA-defined system components
  - BSA rules for PCIe integration
  - Based on the PCIe specification
  - Based on standard OS drivers with no quirks enabled

- **SBBR defined firmware requirements**
  - UEFI testing that is based on the UEFI [Self Certification Test (SCT)](https://www.uefi.org/self-certification)
  - ACPI and SMBIOS testing that is based on [Firmware Test Suite (FWTS)](https://www.uefi.org/firmware-test-suite)

The test suites are hosted in GitHub and are open source, with a permissive Apache v2 license.

**SBSA Architecture Compliance Suite**

The [SBSA Architecture Compliance Suite](https://github.com/ARM-software/aro-sbsa-acs) (ACS) is a collection of self-checking, portable C-based tests. This suite includes a set of examples of the invariant behaviors that are provided by the SBSA specification, so that implementers can verify if these behaviors 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**

Arm [Enterprise ACS](https://github.com/ARM-software/aro-enterprise-acs) includes a set of examples of the invariant behaviors that are provided by a set of specifications for enterprise systems (for example, SBSA or SBBR), so that implementers can verify if these behaviors have been interpreted correctly. ACS is delivered with tests in source form along with a build script. The output of the build is a bootable Linux UEFI Validation (LUV) OS image that can run all tests that are required by these specifications.

The current ACS is designed for the previous SBSA and SBBR specifications-compliance. Arm is restructuring the ACS for the Arm SystemReady compliance testing, to be used for future compliance of the BSA and its supplements, as well as BBR recipes.
Arm SystemReady Program

4.1 Background and History

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

Arm ServerReady established the minimum hardware and firmware requirements in SBSA and SBBR and provides a test suite in ACS to test the system against the SBSA and SBBR standards. It allowed 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.

Compliant systems that adhere to the Arm ServerReady Terms and Conditions were issued with a compliance certificate.

The program involved:

❖ A set of compliance tests
  — These covered the Arm standards specifications, namely the SBSA, and SBBR.
  — The tests also included booting standard operating systems and collecting boot logs.

❖ A certification process
  — Partners ran the compliance tests themselves or with help from the Arm support team.
  — Arm helped with debugging issues discovered.
  — Certification was provided after successful completion of the tests.

❖ Marketing material
  — Partners could use the ServerReady logo as part of their product promotion, if they passed the certification process. We also hosted partner logos on our Developer website.

For more information on the Arm ServerReady program, visit the Arm ServerReady website.
4.2 Arm ServerReady Stakeholders and Responsibilities

The server ecosystem is complex and contains multiple vendors.

- 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 test every possible system
- Compliance tools can help one vendor check the input they receive from another
  - OEMs and ODMs can check if silicon vendor hardware is compliant with SBSA hardware requirements
  - OS vendors can check if OEM and ODM systems are compliant with SBSA hardware requirements and SBBR firmware requirements

Arm engaged with silicon vendors, ODMs, OEMs, and BIOS vendors to run the ServerReady compliance suite tests. Tests were developed by our architecture team, which also developed the specifications in collaboration with the Arm ServerAC community (see Appendix A). Arm has a support team that helps in the running of tests and the debugging of issues. Arm also has a certification team that reviews test results and runs the certification process. Once tests were passed, Arm marketing team granted the certificates and helped with communication.

Arm plans to use the same certification flow for the Arm SystemReady program.
4.3 Arm SystemReady Program

The Arm SystemReady Program is a natural extension of the Arm ServerReady Program with several important changes that are described in this section. Different market segments may target different sets of operating systems and hypervisors with different hardware and firmware requirements. The term band is used to identify these differences with a shorthand notation for each band -- SR (ServerReady), LS (LinuxBoot ServerReady), ES (Embedded ServerReady) and IR (IoT Ready).

SystemReady SR is technically identical to ServerReady and continues to bring the exact same benefits to the Arm server ecosystem. The additional bands in SystemReady (LS, ES, and IR) are designed to serve the needs of a broader silicon and software ecosystem. We are defining the bands in consultation with our partners and expect that all operating system distributions find a band that adequately captures their basic requirements for a standards-based Arm platform.

There are two key changes in SystemReady that enable the additional bands:

First, the hardware requirements specification – the SBSA – has been refactored into a common BSA and a server market-specific supplement. The common BSA contains only the bare minimum requirement to deploy an operating system. The BSA is a baseline and, as such, there is no limit on differentiation and special features that can be built atop the base platform, per the market need.

Second, we have added more boot recipes to accommodate the different standards and implementations that are used in a broader ecosystem, for example LinuxBoot, U-Boot and Devicetree. These recipes are described in the BBR specification.

All SystemReady bands are supported by a common ACS that is modular, to support testing against different combinations of specifications required by a SystemReady band.

### Certification Description Specifications

<table>
<thead>
<tr>
<th>Certification</th>
<th>Description</th>
<th>Specifications</th>
</tr>
</thead>
<tbody>
<tr>
<td>SystemReady SR</td>
<td>ServerReady</td>
<td>BSA SBSA BBR(SBBR): UEFI+ACPI</td>
</tr>
<tr>
<td>SystemReady LS</td>
<td>LinuxBoot Server Ready</td>
<td>BSA SBSA BBR(LBBR)</td>
</tr>
<tr>
<td>SystemReady ES</td>
<td>Embedded Server Ready</td>
<td>BSA — BBR(SBBR): UEFI+ACPI</td>
</tr>
<tr>
<td>SystemReady IR</td>
<td>IoT Ready</td>
<td>BSA — BBR(EBBR): reduced set of UEFI</td>
</tr>
</tbody>
</table>
SystemReady ES and IR (for 64-bit) have the same hardware requirements. SystemReady IR requires the reduced set of UEFI interfaces specified in the EBBR specification. SystemReady ES requires ACPI interfaces in addition to the UEFI interfaces. Systems that are certified as SystemReady ES meet the requirements for SystemReady IR. There is no need for these systems to be certified as SystemReady IR. Similarly, SystemReady SR requires SBSA compliance and more stringent UEFI and ACPI requirements. Systems that are certified as SystemReady SR meet the requirements for SystemReady ES and IR. There is no need for these systems to be certified as SystemReady ES or IR.

A 32-bit system can be certified as SystemReady IR if it supports the EBBR specification. However, since BSA specification does not cover 32-bit systems, we list the 32-bit systems separately from the 64-bit systems on the Arm SystemReady System Compatibility List.

The detailed requirements for these bands are defined in the Arm SystemReady Requirements Specification.
SystemReady Opt-In Extensions

5.1 Pre-silicon Certification

The Arm SystemReady Program aims at making software Just Work on Arm-based devices. For systems that support technologies, such as PCI Express, this typically means the integration with third-party IP. Nearly all SoCs were or still are non-compliant with PCI Express requirements, as identified in the BSA specification. The lack of ECAM support is a good example. Until now the non-compliance is only identified post-silicon. It is very costly to respin the SoC or wait for the next generation. Providing a pre-silicon compliance verification for SoC designs, before tape-out, is critical. In addition to compliance, performance is also part of the effort.

Arm SystemReady Pre-silicon Certification is an option for the silicon partners. Arm has been working with Synopsys and Cadence on this certification. For silicon partners new to Arm SystemReady, Pre-silicon certification is the first step that would make the final Arm SystemReady certifications a much simpler task.
5.2 Security Option

The Arm SystemReady Program aims at making software Just Work on Arm-based devices. Arm adds a "Security Option" in the Arm SystemReady Program for devices that are compliant to the UEFI Secure Boot and Firmware Update through Capsule Update services. The requirements are specified in the *Base Boot Security Requirements (BBSR)* specification. Arm is adding the compliance testing in the ACS, to be used for future compliance of the BBSR specification.

SystemReady is aligned to the separate PSA Certified framework, so that it is possible for a device to be both PSA Certified and SystemReady. There are no dependencies, so it is up to the partners to decide whether either or both certifications are needed.

The SystemReady “Security Option” certification gives assurance that if a platform implements a security feature, it is implemented according to standards for the benefit of software. SystemReady itself does not tell you the features that must be implemented to make a particular platform secure, nor does it check that an implementation has the required security properties. Instead, SystemReady aligns with PSA Certified to address security needs.

The PSA Certified framework establishes a common baseline for security, based on 10 security goals at PSA Certified level 1 and focuses on the implementation of a Root of Trust (RoT) at PSA Certified level 2 and PSA Certified level 3. Devices that are Arm SystemReady certified must also obtain the “Security Option” certification, prior to certification by PSA Certified.

PSA Certified requires certain platform security features to mitigate threats and provides assurance that the security features have been implemented correctly. For those features that are visible to system software, the SystemReady “Security Option” prescribes standard implementations for those features, to enable deployment of an OS that relies on those standards.
How To Use SystemReady

6.1 BSA Compliance
To make software Just Work on Arm-based devices, the most fundamental step is for the silicon providers to make sure the SoCs are designed to be BSA-compliant, regardless of the market segment.

If the hardware is BSA-compliant, the OS vendor can target the BSA as a common Arm hardware platform vs. having to do platform-specific engineering. For open-source operating systems, a large community is working on support for BSA-compliant platforms and therefore upstreamed-support is automatic.

If PCIe is supported, PCIe-compliance is critical. PCIe enumeration by software is a complex and challenging area and so Arm has worked with third-party IP providers to enable IP integration compliance verification before tape-out. Arm SystemReady includes a Pre-silicon Certification option.

If the SoCs are targeting specific market segments, extra compliance to the segment-specific xBSA supplements are needed. For example, compliance to SBSA Supplement is important for the server systems. It is up to the end customer and OS vendor whether they require support for a segment-specific supplement.

6.2 BBR Compliance
Once the SoCs are compliant to these hardware specifications, the task for the firmware development (whether done by silicon providers, independent firmware vendors or OEM and ODMs) can be flexible and straightforward for devices based on these SoCs. Different BBR recipes can be supported based on the target OSes.

For example, the same device based on a SoC that is compliant to BSA and SBSA Supplement can either provide the firmware using:

- The SBBR recipe if the target OS requires UEFI, ACPI and SMBIOS, or
- The LBBR recipe if LinuxBoot is used instead of UEFI

Similarly, the same device based on a SoC that is compliant to BSA can either provide the firmware using:

- The SBBR recipe if the target OS requires UEFI, ACPI and SMBIOS, or
- The EBBR recipe if the target OS requires UEFI
This is the basis for the current four bands of the Arm SystemReady program: SR (ServerReady), LS (LinuxBoot ServerReady), ES (Embedded ServerReady) and IR (IoT Ready).

For the devices that are Arm SystemReady SR, ES or IR certified, there is an extra “Security Option” to certify if the device also supports the UEFI Secure Boot and firmware update through Capsule services.

SystemReady SR, ES, and IR require a firmware that implements the UEFI ABI. To enable binary portability of operating system images between platforms, there has to be an agreed on ABI between the OS and the Non-secure world boot loader. While we could have invented a new standard for the embedded world, UEFI has the benefit of a large ecosystem, including implementations (for example, in U-Boot) that have been demonstrated on very constrained devices.

If an OEM/ODM wants to build a vertically integrated platform, based on Arm, that runs its own embedded Linux distribution, or one provided by the silicon provider, Arm SystemReady certification is not required. However, the OEM/ODM partner should consider whether it is possible to reduce software engineering cost over the lifetime of the device by using an Arm SystemReady-certified SoC.
# Examples

<table>
<thead>
<tr>
<th>Use-case</th>
<th>SystemReady solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>I want to build an Arm server SoC for hyperscale/enterprise and my customer will choose and deploy a server OS.</td>
<td>SoC and firmware should be built for SystemReady SR certification.</td>
</tr>
<tr>
<td>I want to build an IoT gateway based on an Arm SoC. I want to deploy a standard Linux distro that can host containers and application software stacks.</td>
<td>SoC and firmware should be built for SystemReady IR certification.</td>
</tr>
<tr>
<td>I want to build an enterprise networking platform based on Arm silicon from my supplier. I will build my own OS based on Yocto Linux, but I want UEFI standard firmware and also UEFI Secure Boot. I want to update my firmware using UEFI capsule update.</td>
<td>Require SoC and firmware to be SystemReady IR + Security Option. Consider PSA Certified for an independent lab evaluation of the Root of Trust.</td>
</tr>
<tr>
<td>I want to land my OS distribution on Arm SoCs. My OS needs ACPI but I am prepared to work around hardware that is not quite BSA-compliant to gain access to a broad selection of silicon, originally built for the mobile, and embedded markets.</td>
<td>Require SoC and firmware to be SystemReady ES certified. Provide a sample of my OS for the hardware compatibility list.</td>
</tr>
<tr>
<td>I want to deploy an Arm server with firmware that supports LinuxBoot.</td>
<td>Require server to be SystemReady LS-certified.</td>
</tr>
<tr>
<td>I want to build a vertically integrated platform based on Arm that runs my own embedded Linux distribution, or one provided by the SIP.</td>
<td>SystemReady certification is not required, but I should consider whether it is possible to reduce my software engineering cost over the lifetime of the device by using a SystemReady device.</td>
</tr>
</tbody>
</table>
Conclusion

This whitepaper describes the Arm SystemReady program, its components and process, and the extension from the existing ServerReady compliance program. It also describes the Arm standards-based approach and Arm-specific standards that lay the foundation for the SystemReady program.

Please contact SystemReady@arm.com if there are any questions about the Arm SystemReady program.
Appendix A

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 uses Causeway and communicates in various ways including email, regular conference calls, and biannual gatherings. To communicate with this group, emails can be sent to armserverac-discuss@arm.causewaynow.com.

The ServerAC uses Mantis Issue tracker for developing the specifications and tracking issues, proposals, and requests. The database is accessible to ServerAC member companies at https://mantis.arm.causewaynow.com/. Any ServerAC member can raise a request. Issues are described in the ServerAC regular meetings. Once agreed on, the changes get integrated into the specifications.
The ServerAC development process is described here:

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, and 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 once it is published.
More Advisory Committees may be formed for various market segments on the as-needed basis. At the time of writing, BSA and its SBSA supplement, BBR, SBMR, and SBSG specifications are developed in the ServerAC.

The relationship of the specifications and the potential Advisory Committees is shown in the following diagram:

### Appendix B

#### Glossary

<table>
<thead>
<tr>
<th>Term</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACS</td>
<td>Architectural Compliance Suite</td>
</tr>
<tr>
<td>BSA</td>
<td>Base System Architecture</td>
</tr>
<tr>
<td>BBR</td>
<td>Base Boot Requirements</td>
</tr>
<tr>
<td>SBSA</td>
<td>BSA Supplement for Servers. Also used for the previous Server Base System Architecture specification</td>
</tr>
<tr>
<td>SBBR</td>
<td>A recipe defined in the BBR specification requiring UEFI and ACPI. Also used for the previous Server Base Boot Requirements specification</td>
</tr>
<tr>
<td>EBBR</td>
<td>A recipe defined in the BBR specification requiring a subset of UEFI. Also used for the Embedded Base Boot Requirements specification</td>
</tr>
<tr>
<td>LBBR</td>
<td>A recipe defined in the BBR specification supporting LinuxBoot</td>
</tr>
<tr>
<td>SBSG</td>
<td>Server Base Security Guide</td>
</tr>
<tr>
<td>Acronym</td>
<td>Description</td>
</tr>
<tr>
<td>---------</td>
<td>-------------</td>
</tr>
<tr>
<td>SBMR</td>
<td>Server Base Manageability Requirements</td>
</tr>
<tr>
<td>ECR</td>
<td>Engineering Change Request</td>
</tr>
<tr>
<td>TF-A</td>
<td>Trusted Firmware-A</td>
</tr>
<tr>
<td>RHEL</td>
<td>Red Hat Enterprise Linux</td>
</tr>
<tr>
<td>OEM</td>
<td>Original Equipment Manufacturer</td>
</tr>
<tr>
<td>ODM</td>
<td>Original Device Manufacturer</td>
</tr>
<tr>
<td>IHV</td>
<td>Independent Hardware Vendor</td>
</tr>
<tr>
<td>ISV</td>
<td>Independent Software Vendor</td>
</tr>
<tr>
<td>IBV</td>
<td>Independent BIOS (or Firmware) Vendor</td>
</tr>
<tr>
<td>OSV</td>
<td>Operating System Vendor</td>
</tr>
<tr>
<td>SIP</td>
<td>Silicon Provider</td>
</tr>
<tr>
<td>CSP</td>
<td>Cloud Service Provider</td>
</tr>
<tr>
<td>SoC</td>
<td>System-on-chip</td>
</tr>
<tr>
<td>UEFI</td>
<td>Unified Extensible Firmware Interface</td>
</tr>
<tr>
<td>ACPI</td>
<td>Advanced Configuration and Power Interface</td>
</tr>
<tr>
<td>CXL</td>
<td>Compute Express Link</td>
</tr>
<tr>
<td>CCIX</td>
<td>Cache Coherent Interconnect for Accelerators</td>
</tr>
<tr>
<td>OCP</td>
<td>OpenCompute Project</td>
</tr>
<tr>
<td>EDK2</td>
<td>EFI Development Kit, ver 2.0</td>
</tr>
<tr>
<td>PLDM</td>
<td>Platform Level Data Model</td>
</tr>
<tr>
<td>BMC</td>
<td>Baseboard Management Controller</td>
</tr>
<tr>
<td>IPMI</td>
<td>Intelligent Platform Management Interface</td>
</tr>
<tr>
<td>MCTP</td>
<td>Management Component Transport Protocol</td>
</tr>
</tbody>
</table>

All brand names or product names are the property of their respective holders. Neither the whole nor any part of the information contained in, or the product described in, this document may be adapted or reproduced in any material form except with the prior written permission of the copyright holder. The product described in this document is subject to continuous developments and improvements. All particulars of the product and its use contained in this document are given in good faith. All warranties implied or expressed, including but not limited to implied warranties of satisfactory quality or fitness for purpose are excluded. This document is intended only to provide information to the reader about the product. To the extent permitted by local laws, Arm shall not be liable for any loss or damage arising from the use of any information in this document or any error or omission in such information.