presented by

arm

Integrate Arm SystemReady Band – UEFI and ACPI Compliance for Better Quality and Faster Debug

UEFI 2025 Developers Conference & Plugfest

October 10, 2025

Sunny Wang (Arm)

Meet the Presenters

Sunny Wang

Principal Firmware Engineer

Sunny is a Principal Firmware Engineer in Arm's Architecture and Technology Group, working as the technical lead on SystemReady compliance program for SystemReady band, driving the creation and adoption of the UEFI and ACPI technologies across the Arm ecosystem. He is also a contributor to UEFI/APCI related open-source projects including UEFI EDK2, FWTS, EDK2-test (UEFI SCT), and industry standards. Prior to joining Arm, Sunny held UEFI/BIOS firmware lead and engineering roles at HPE/HP and ASUS, accumulating over 12 years of experience in system firmware development.

Related UEFI DevCon Sessions

- Arm System Firmware Architecture
- Evolving ACPI Standards for Arm Systems
- Advancing Firmware Compliance in the Arm Ecosystem

Agenda

- Why – Challenges and requirements for generic Operating System (OS) support on Arm systems
- What - Arm system architecture specifications and Arm SystemReady program
- What - Compliance test suites (SystemReady Band Architecture Compliance Suite (ACS))
- How - Integrate ACS and self-declare SystemReady band compliance

Why – Challenges and Requirements for Generic OS Support on Arm Systems

Issues with Generic OS Boot, Install, and Run (Linux kernel panic, Windows BSOD, etc.)

A **generic Operating System (OS)** refers to a non-customized, off-the-shelf operating system. It expects the platform to comply with **industry standards** such as UEFI, ACPI, and SMBIOS, without requiring vendor-specific workarounds.

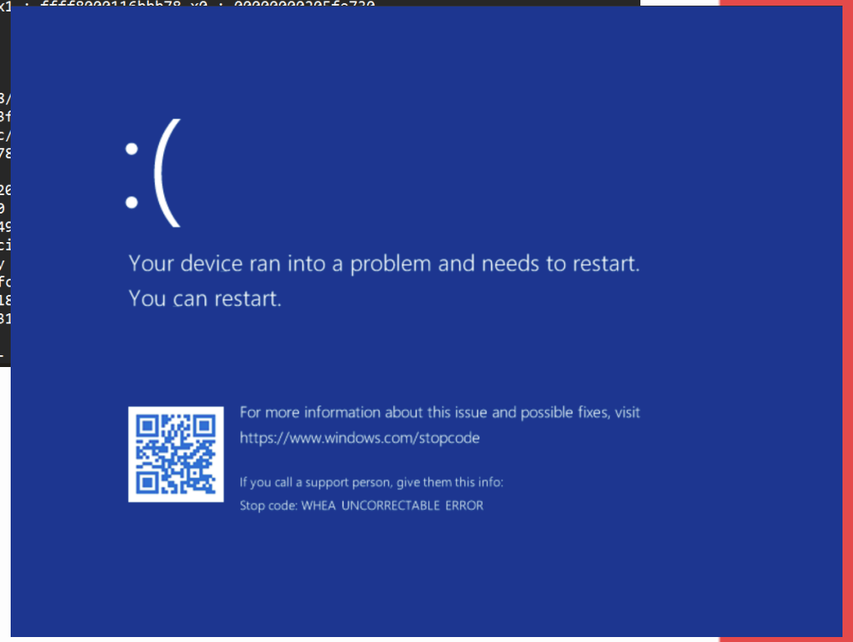
Firmware issues

- Firmware is not compliant with industry standards and doesn't meet the requirements in OSV docs.
- In most cases, the issue is caused by incorrect or incomplete implementation of UEFI, ACPI, or SMBIOS interfaces..

Hardware issues

- The hardware is implemented differently than the OS expects, or hardware is not compliant with industry standards.

```
[   10.094867] Unable to handle kernel execute from non-executable memory at virtual address 0000000020b90440
[   10.104520] Mem abort info:
[   10.107301]   ESR = 0x8600000e
[   10.110346]   EC = 0x21: IABT (current EL), IL = 32 bits
[   10.115648]   SET = 0, FnV = 0
[   10.118693]   EA = 0, S1PTW = 0
[   10.121824]   FSC = 0x0e: level 2 permission fault
[   10.126606] user pgtable: 4k pages, 48-bit VAs, pgdp=0000002080a76000
[   10.133037] [0000000020b90440] pgd=0000000000000000, p4d=0000000000000000
[   10.139819] Internal error: Oops: 8600000e [#1] PREEMPT SMP
[   10.145380] Modules linked in:
[   10.148424] CPU: 3 PID: 190 Comm: kworker/u32:3 Not tainted 5.15.0-00001-gaa68a61f9253 #1
[   10.164839] Workqueue: efi_rts_wq efi_call_rts
[   10.169282] pstate: 40000005 (nZcv daif -PAN -UAO -TCO -DIT -SSBS BTYPE=--)
[   10.176232] pc : 0x20b90440
[   10.179015] lr : 0x201b09ac
[   10.181796] sp : ffff8000116bbb10
[   10.185098] x29: ffff8000116bbb10 x28: 0000000000000000 x27: 0000000000000000
[   10.192224] x26: 000000002003037c x25: 00000000205fe730 x24: ffff8000116bbc17
[   10.199349] x23: 00000000201c0000 x22: 00000000205fe730 x21: 0000000000000000
[   10.206474] x20: 0000000000000001 x19: 0000000020d20432 x18: 0000000000000000
[   10.213598] x17: 0000000000000000 x16: 0000000000000000 x15: 000000002003037c
[   10.220723] x14: 00000000201b5000 x13: 00000000201b5000 x12: 0000000000000020
[   10.227848] x11: 0000000000000000 x10: ffff8000116bbcc0 x9 : 11d31402eb704011
[   10.234973] x8 : 0000000000000000 x7 : 0000000000000004 x6 : 0000000020120448
[   10.242098] x5 : ffff8000116bbc17 x4 : 0000000000000001 x3 : 0000000020d20432
[   10.249222] x2 : 0000000020b90440 x1 : ffff8000116bbb78 x0 : 000000002005fe730
[   10.256347] Call trace:
[   10.258782]  0x20b90440
[   10.261216]  0x201b3704
[   10.263650]  0x201b4f40
[   10.266084]  efirt_asm_wrapper+0x28/
[   10.270258]  efi_call_rts+0x120/0x3f
[   10.273910]  process_one_work+0x1ec/
[   10.277908]  workerthread+0x44/0x478
[   10.281558]  kthread+0x154/0x160
[   10.284775]  ret_from_fork+0x10/0x20
[   10.288345] Code: 00000000 00000000
[   10.294427] ---[ end trace 8fe28a849
[   10.309505] Kernel panic - not synci
[   10.316366] SMP: stopping secondary
[   10.320297] Kernel Offset: 0x5347cfc
[   10.326377] PHYS_OFFSET: 0xffffab718
[   10.330547] CPU features: 0x1000023
[   10.334892] Memory Limit: none
[   10.348298] ---[ end Kernel panic -
```

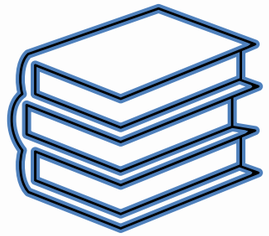


Key Challenges in Debugging Operating System Issues

- Understanding of many specs is required
  - Industry standard firmware specs (UEFI, ACPI, SMBIOS)
  - Arm firmware specs (PSCI, SMCCC, etc.)
  - Industry standard and Arm hardware specs (Arm ARM, GIC, SMMU, PCIe, etc.)
  - Documents from Operating System Vendors (OSVs)

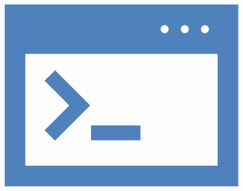
- Specialized Debugging Knowledge is Required
  - Operating System (OS) boot parameters
  - Commands to check early kernel messages and system Logs
  - Tools like kdump for Linux, Windbg for Windows

Solution to Improve System Quality and Reduce OS Boot Debug Time



Specifications - Arm System Architecture Specifications

- Contain ONLY the requirements from industry standard spec that are necessary for supporting generic Operating Systems (OSes)
- Also contain the information that are missing in industry standard specs but required for Operating System (OS) support.

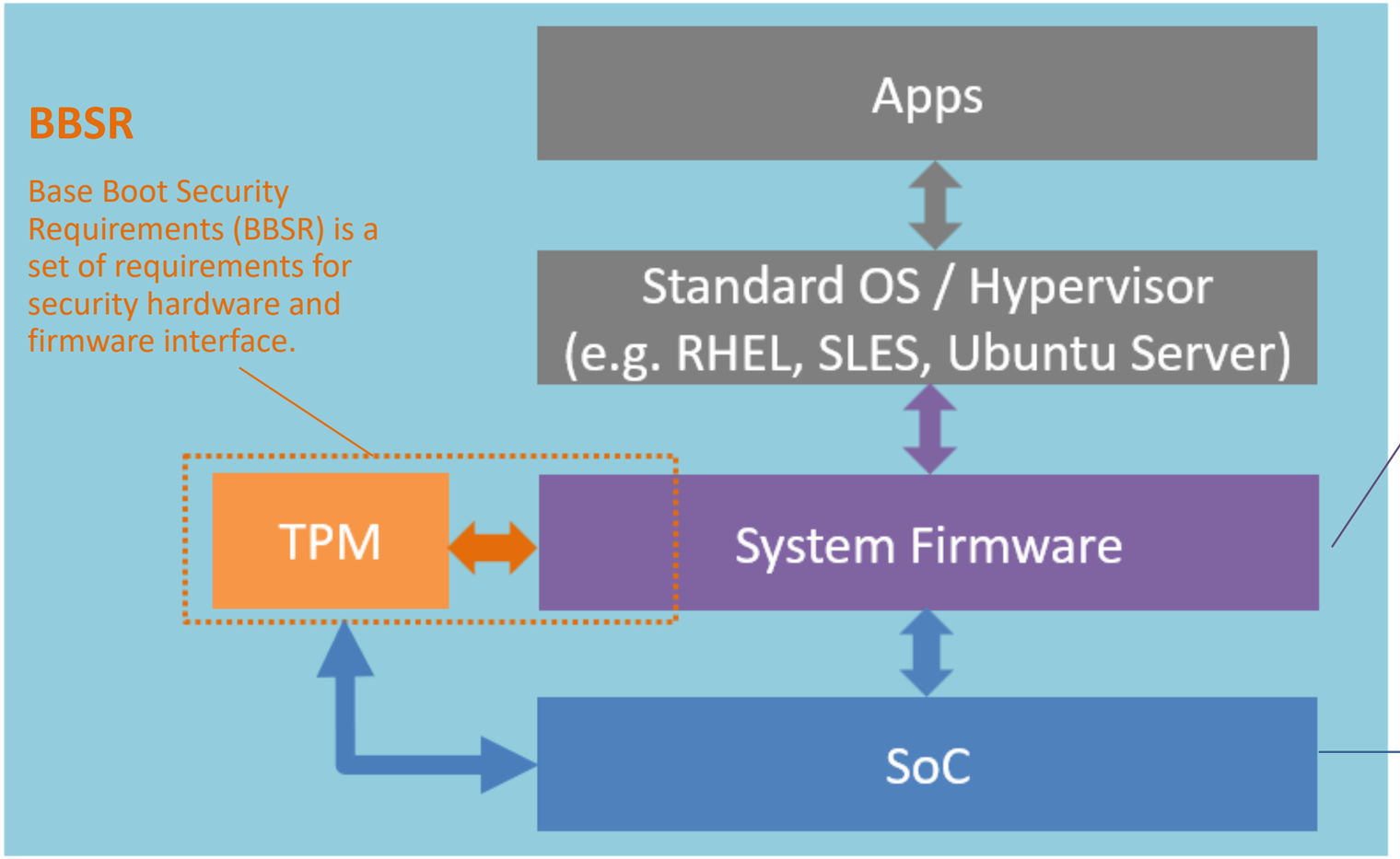


Test suites – SystemReady Architecture Compliance Suite (ACS)

- Provide a reliable and efficient way to validate implementations against the requirements in the Arm System Architecture Specifications.

What - Arm System Architecture Specifications and Arm SystemReady Program

Arm System Architecture Specifications



BBR (SBBR)

Base Boot Requirement (BBR) is foundation (minimum firmware) for generic Operating System (OS) install and boot.

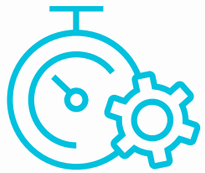
The SBBR recipe is a set of firmware interface requirements—including UEFI, ACPI, and SMBIOS

BSA + SBSA

Base System Architecture (BSA) is hardware foundation/baseline (minimum hardware) for generic OS install and boot .

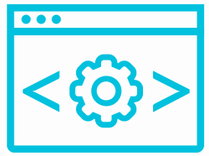
Server Base System Architecture (SBSA) Levels increase support for hypervisor features, Reliability, Availability, and Serviceability (RAS), Memory Partitioning and Monitoring (MPAM), platform security, etc.

Arm System Architecture Specifications



Hardware Baseline (BSA – Base System Architecture)

- Common standard architecture for 64-bit A-profile applicable to all market segments
- Defining a minimal set of CPU and System architecture necessary for an Operating System (OS) to boot and run.
- BSA v1.2 (July 2025)



Firmware baseline (BBR – Base Boot Requirements)

- Defining a minimal set of firmware requirements necessary for an OS to boot and run.
- Expands to include common firmware interfaces, but recognizes that different software stacks will require different recipes
  - SBBR (ACPI, UEFI, SMBIOS)
  - EBBR (device tree, UEFI)
- BBR v2.2 (June 2025)

Hardware Supplements (xBSA)

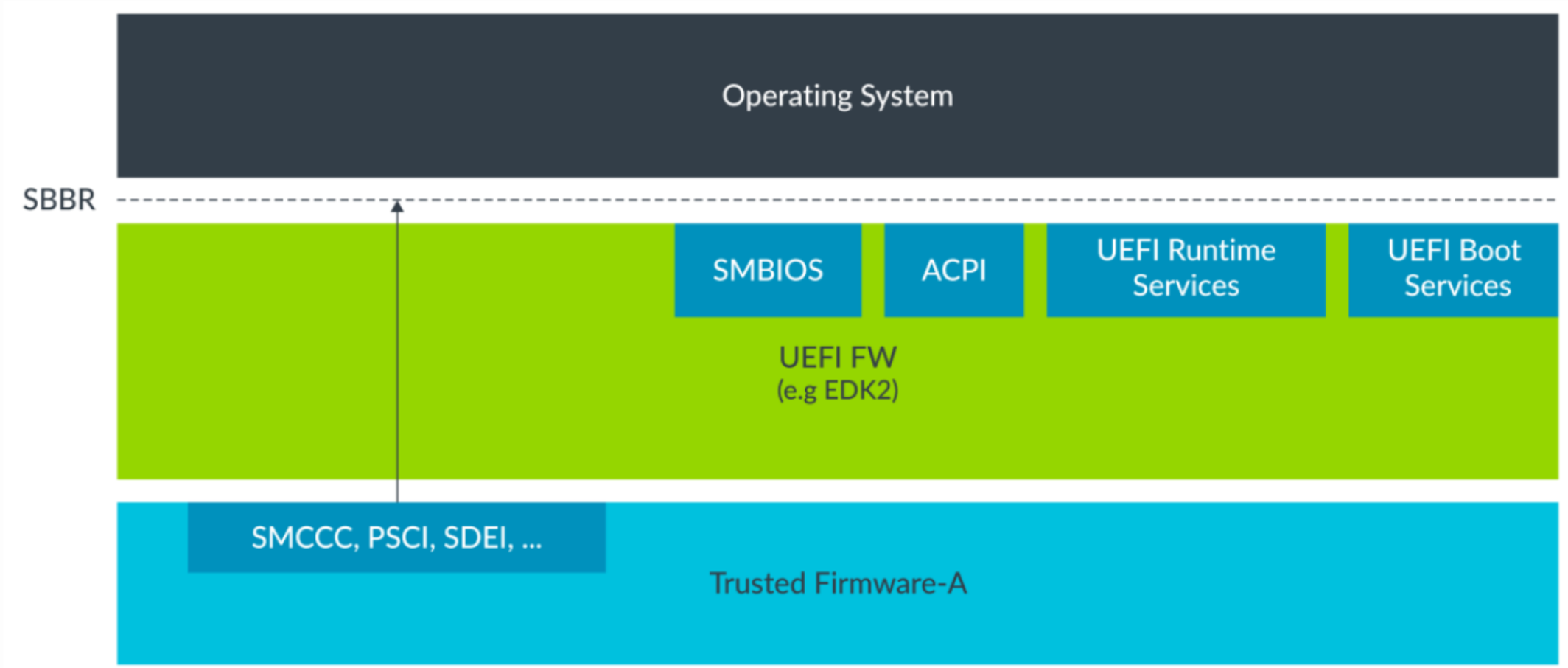
- Provides market segment specific hardware requirements
- SBSA (Server BSA) for server requirements
  - SBSA v7.2 (Nov 2024)

Arm System Architecture Specifications

Security (BBSR - Base Boot Security Requirements)

- UEFI Secure Boot, secure Firmware Update, and TPM
- V1.4 (July 2025)

BBR (SBBR recipe)



Specification for Declaring SystemReady Compliance

SystemReady Requirements Specification

- Defines the requirements for declaring SystemReady Compliance
- Two bands
  - SystemReady band
  - SystemReady Devicetree band
- SRS 3.0 (Nov 2024)

Two sets of spec compliance requirements for declaring SystemReady band compliance:

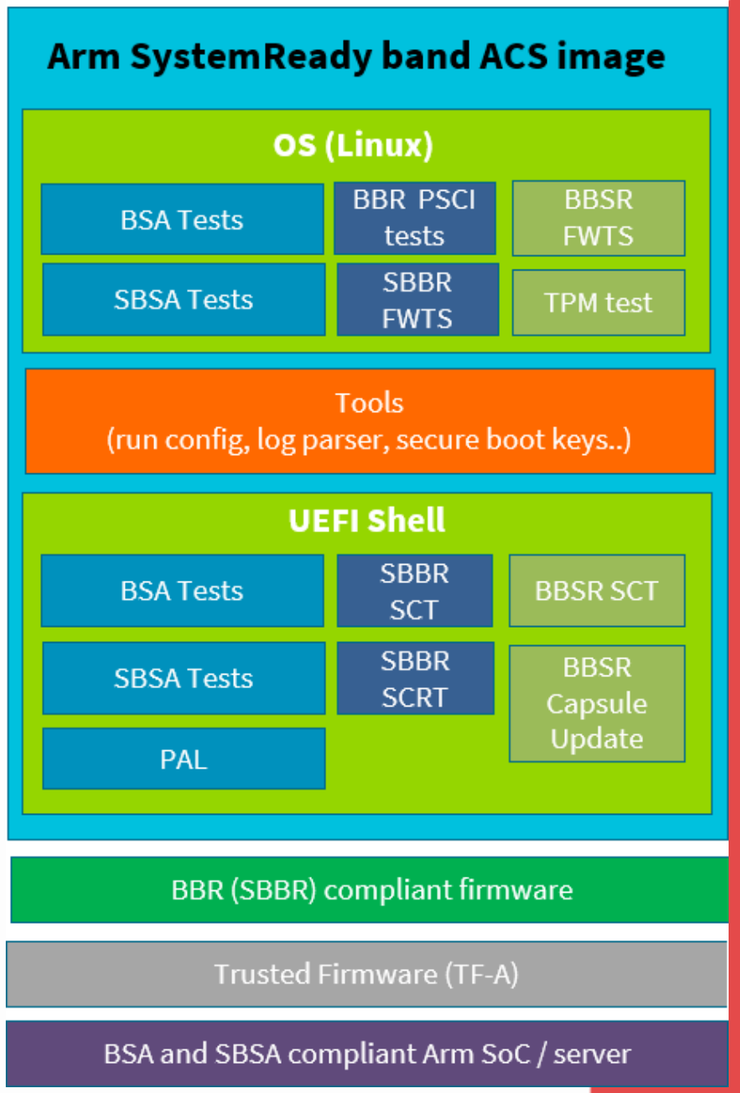
1. BSA + BBR + SBSA + BBSR
   - For server
   - BBSR is optional but strongly recommended
2. BSA + BBR
   - For non-server
   - BBSR is optional (on demand)

What - SystemReady Compliance Test Suites (SystemReady Band Architecture Compliance Suite (ACS))

ACS (Architecture Compliance Suite)

- The SystemReady band ACS is delivered through a live OS image, which enables the basic automation to run the tests for checking the compliance of BSA/SBSA, BBR (SBBR), and BBSR.

Test Tool	Sys Arch Specs	Standard and Arm Specs	Issue Type
UEFI SCT (UEFI)	BBR – SBBR BBSR	UEFI ACPI SMBIOS …	FW
FWTS (Linux)			
BSA.efi and SBSA.efi	BSA and SBSA	Arm ARM, GIC… PCIe, CXL ….	HW (FW ACPI and UEFI)
Linux BSA and SBSA			HW (Linux driver)



Acronym

- FWTS: Firmware Test Suite
- UEFI SCT: UEFI Self-Certification Test

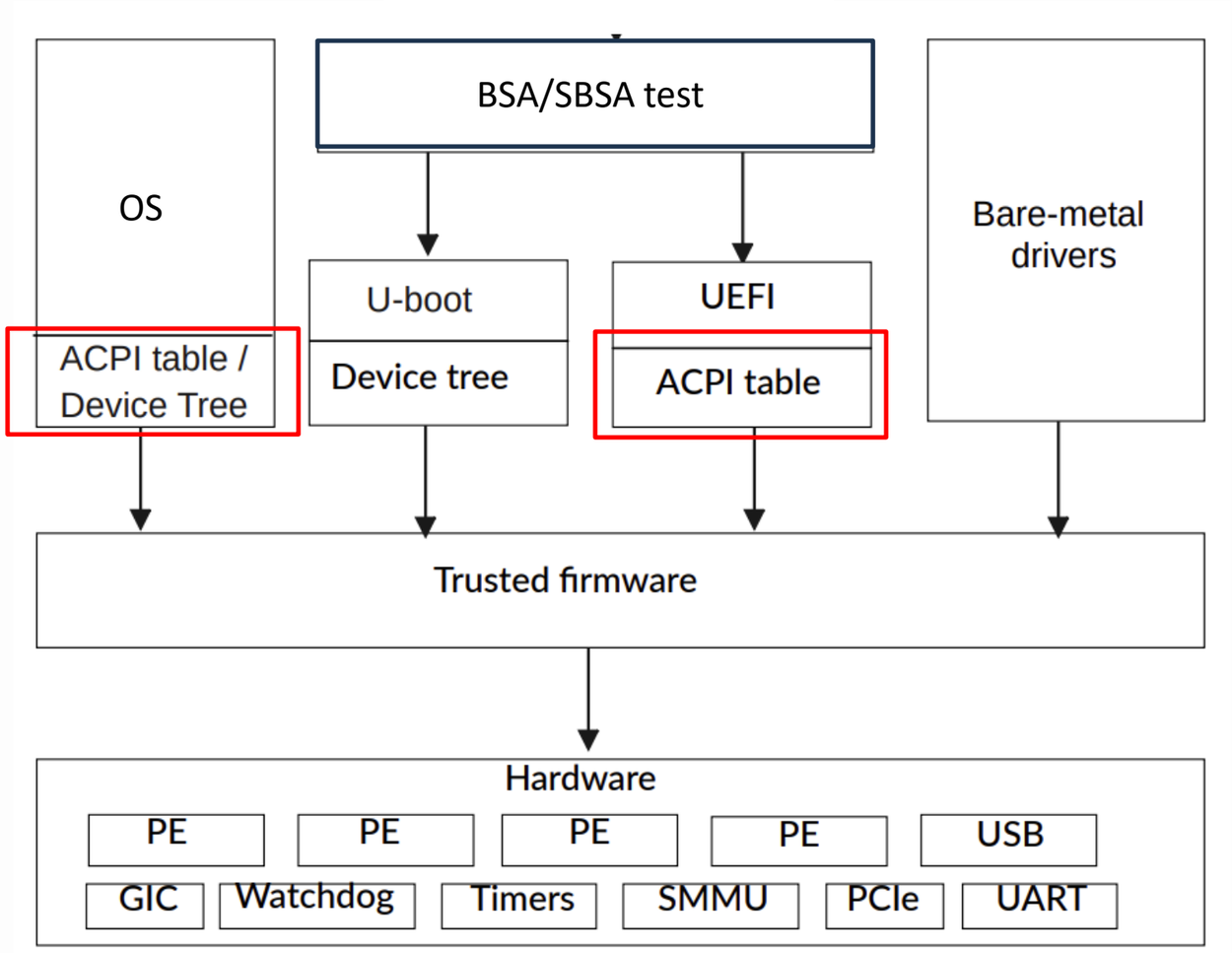
Firmware Interfaces Validated by SBBR UEFI SCT

- SBBR UEFI SCT
  - UEFI Self-Certification Test (SCT) Suite used to validate the compliance with the requirements defined in SBBR recipe of the Base Boot requirements (BBR) specification.
  - It mainly validates the Firmware interfaces that are needed to load and hand off control to the operating system during the pre-boot phase. In other words, it focuses primarily on fthe firmware boot time services.
    - BootServicesTest
    - RuntimeServicesTest
    - LoadedImageProtocolTest
    - DevicePathProcotols
    - ConsoleSupportTest
    - HIITest
    - PCIBusSupportTest
    - SecureTechTest
    - FirmwareManagementTest
    - GenericTest

Firmware Interfaces Validated by SBBR FWTS

- SBBR FWTS
  - Firmware Test Suite (FWTS) used to validate the compliance with the requirements defined in SBBR recipe of the Base Boot requirements (BBR) specification.
  - FWTS mainly validates firmware interfaces (ACPI, SMBIOS, etc.) that the operating system relies on after firmware hand-off. In other words, it focuses primarily on firmware runtime services.
    - uefirtxxxx
    - dmicheck
    - mcfg
    - spcr
    - dbg2
    - …

Firmware Interfaces Validated by BSA/SBSA Tests



For more info, please see [arm_bsa_architecture_compliance_validation_methodology.pdf]

Acronyms

- BSA test: Base System Architecture compliance test
- SBSA test: Server Base System Architecture compliance test

Firmware Interfaces Validated by BSA/SBSA Tests

BSA/SBSA Test Type	UEFI Services	ACPI Table
PE		MADT
MEM	gBS->GetMemoryMap	
GIC		MADT
SMMU		IORT
Timer, Wakeup, Watchdog		GTDT
Peripheral (USB and SATA)	gEfiPciIoProtocolGuid	
Peripheral (UART)		SPCR

Acronyms

- BSA test: Base System Architecture compliance test
- SBSA test: Server Base System Architecture compliance test

Firmware Interfaces Validated by BSA/SBSA Tests

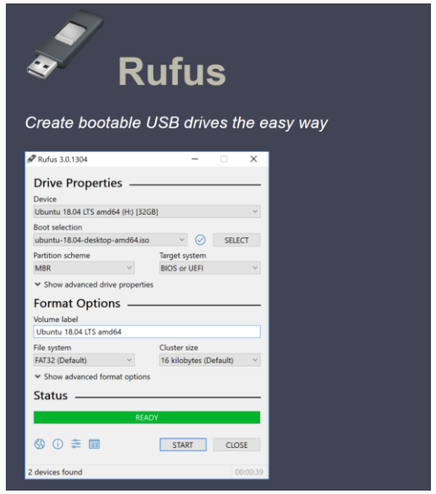
BSA/SBSA Test Type	UEFI Services	ACPI Table
PCIe	gEfiPciIoProtocolGuid gEfiPciRootBridgeIoProtocolGuid gEfiCpuArchProtocolGuid (DMA)	MCFG
MPAM		MPAM, SRAT, HMAT, PPTT
PMU		APMT
RAS		AEST, RAS2, SRAT, HMAT, PPTT, PCCT

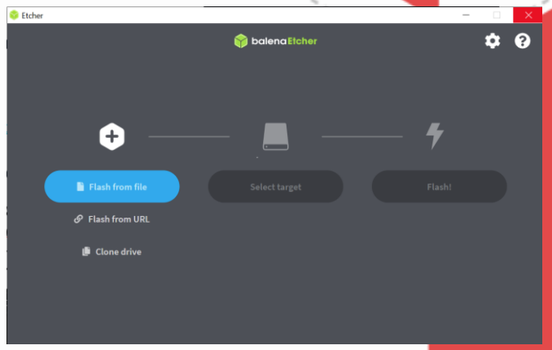
Acronyms

- BSA test: Base System Architecture compliance test
- SBSA test: Server Base System Architecture compliance test

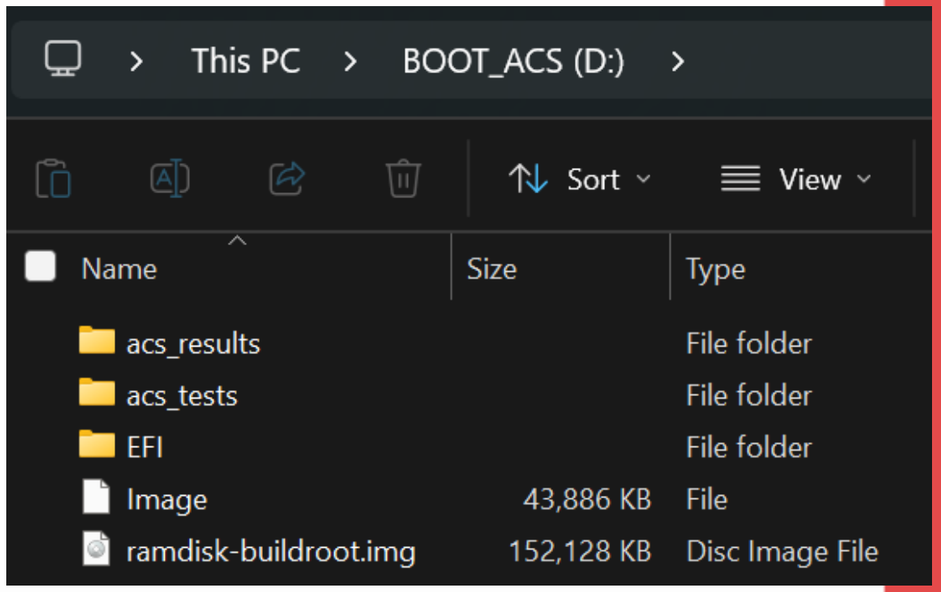
Run Tests - Download and Write the ACS Image

- **Download SystemReady band Architecture Compliance Suite (ACS) image**
  - Compliance test suite (pre-built image) - SytemReady band ACS live image
- **Write the image to a media device**
  - Linux tool – dd
    - *dd if=‘test image’ of=‘usb disk’ status=progress*
      - *dd if=systemready_acs_live_image.img of=/dev/nvme1n1 status=progress*
  - Windows tool
    - Rufus
    - balenaEtcher
  - Don’t use USB stick to run ACS test because SCT test may break USB stick due to too many writes, and using USB stick may take longer than a day to complete SCT test. Therefore, Using NVMe drive, SATA drive, or USB 3.0 external HDD as ACS media device is recommended .



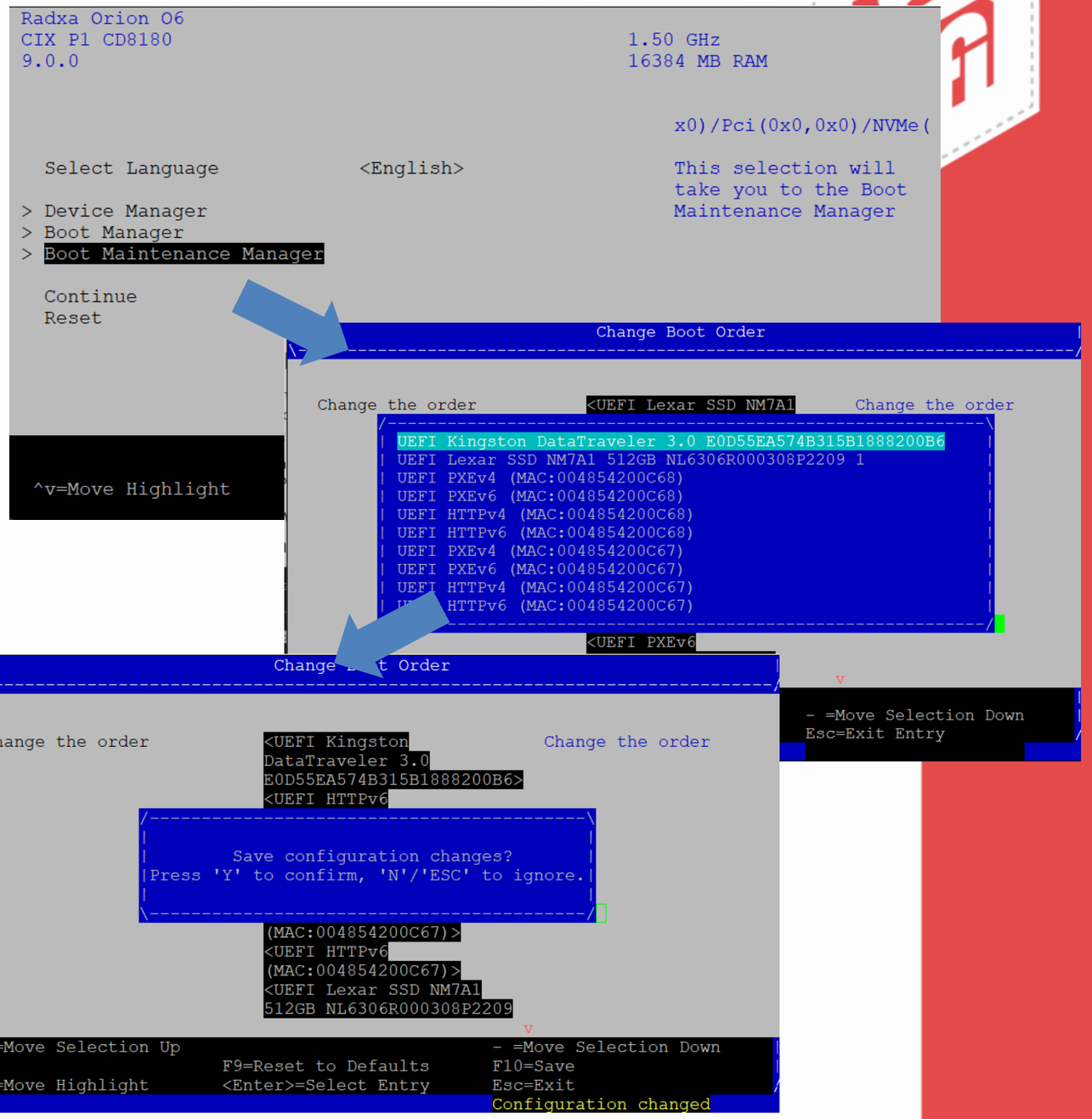


Media device (after writing the ACS image)

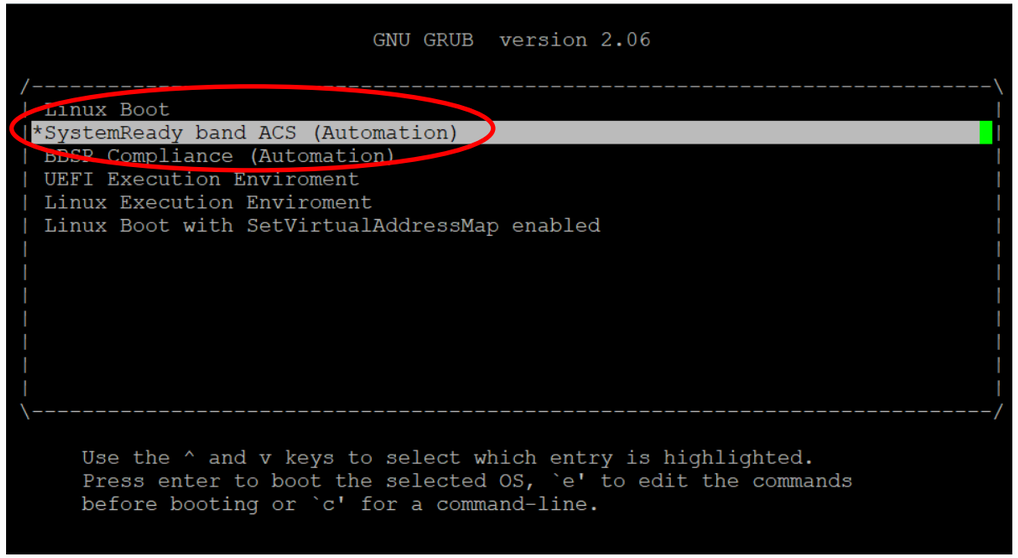


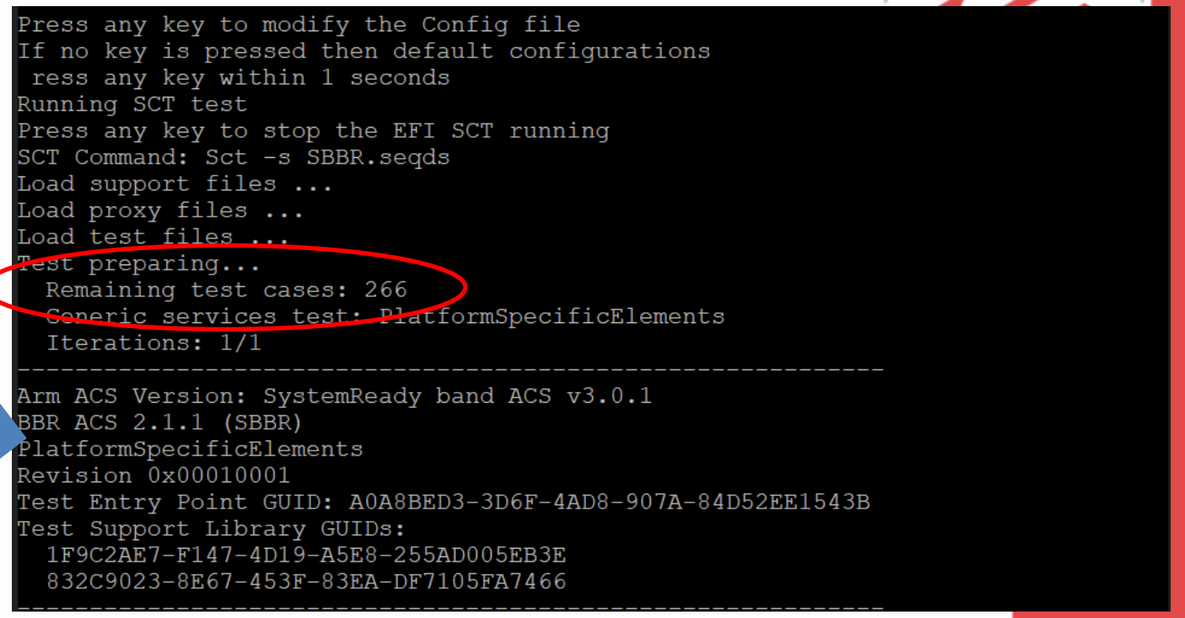
Run Test - Change the Boot Order for Automation Test

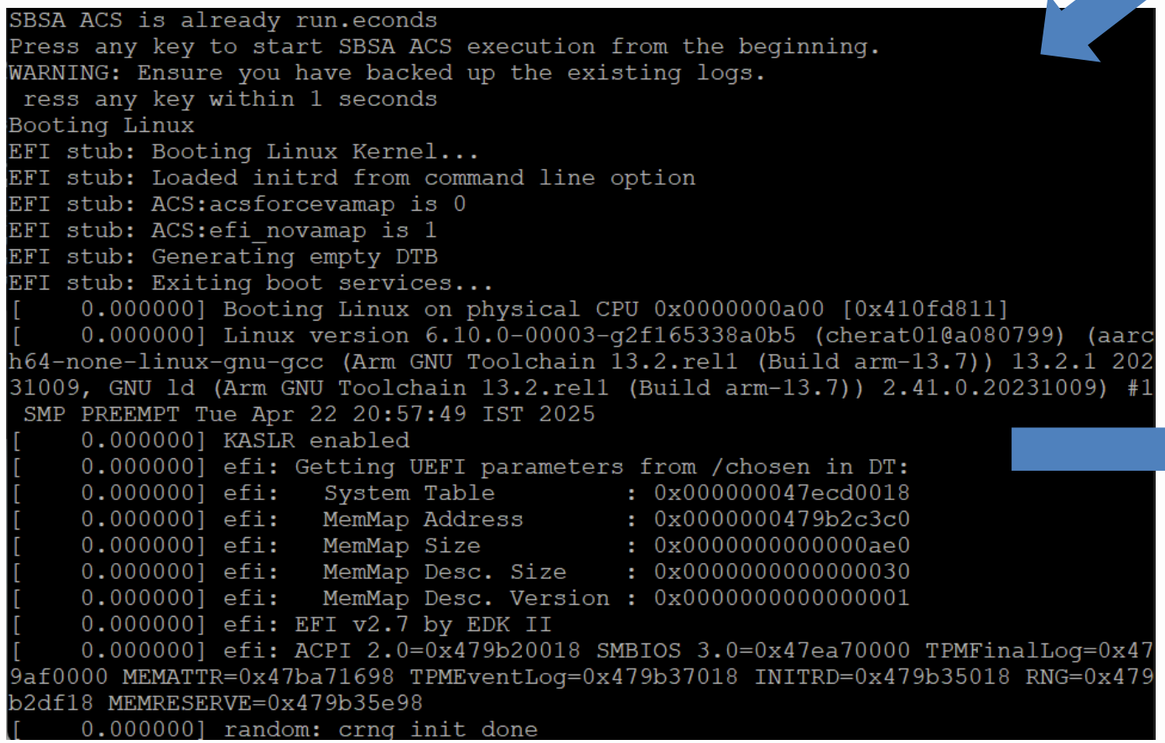
1. Insert the Architecture Compliance Suite (ACS) media device and power on the system
2. Go to Boot menu and move the ACS drive boot option to the top of the boot order and then reboot
3. After reboot, the GRand Unified Bootloader (GRUB) menu will be automatically launched and load the “SystemReady Band ACS (Automation)” option

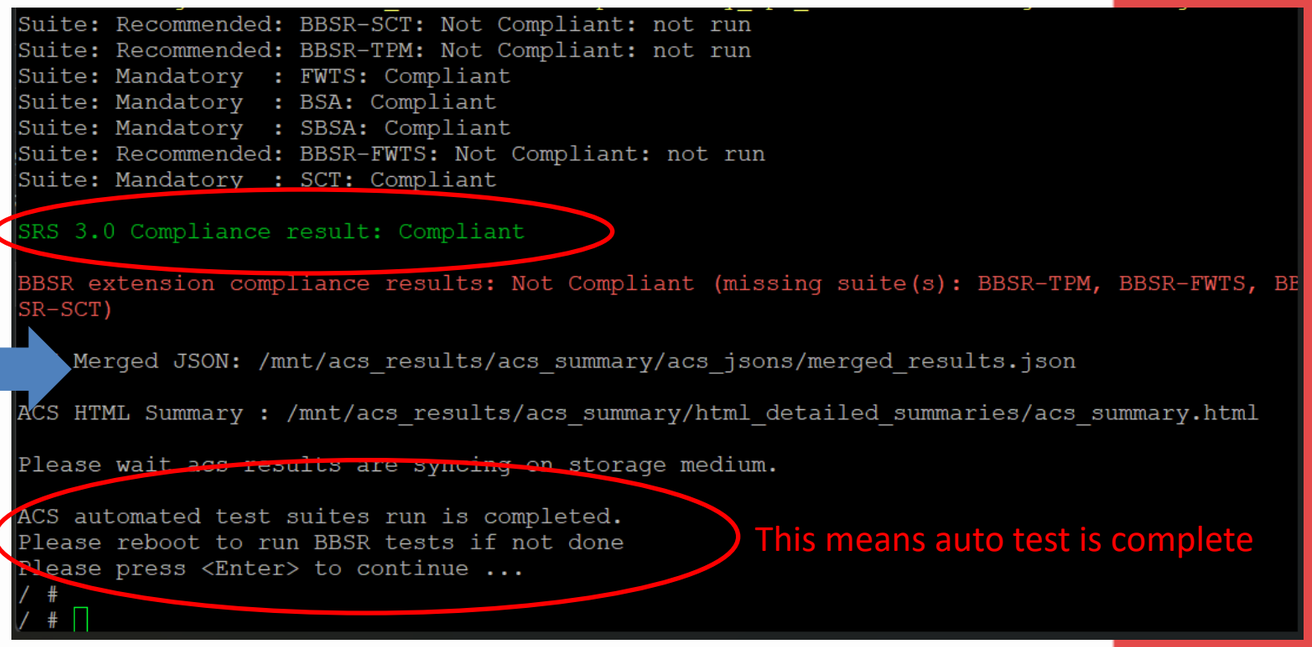


Run Tests – Automation Test









This means auto test is complete

Reference - Guidelines and Open Platforms

- SystemReady FAQ -> SystemReady band – General FAQ
- SystemReady Band Integration and Testing Guide
- SystemReady workshop slide deck
- Arm SystemReady Band Policy Guidelines

Platform	Documentation	Firmware Source Code	Report Issue
Arm Neoverse reference design FVP	User guides	Source code Repos Software Stack	support@arm.com
NVIDIA Grace system	edk2-nvidia wiki	edk2-nvidia repo	edk2-nvidia repo issues
Ampere Altra system	build and update firmware guideline	Ampere edk2-platforms	
CIX P1 system	Orion O6 ducumentation	edk2-cix repo	edk2-cix repo issues
RPi4 (non-compliant)	Raspberry Pi Documentation	edk2-platforms repo - RPi4	Pi Firmware Task Force - RPi4 issues

How - Integrate Architecture Compliance Suite and Self-Declare SystemReady Band Compliance

Integrate ACS Tests Into Production Processes

- **Method 1 - Use the ACS Live Image**
  - Download the Architecture Compliance Suite (ACS) live image and write it a media device on the System Under Test (SUT).
  - **Pros:** Easier to integrate and manage (Only one file);
  - **Cons:** Requires an additional storage device to deploy the ACS image.

- **Method 2 - Use Individual Binaries/Test Suites**
  - Integrate specific pre-built test components into your test automation.
  - **Pros:** Doesn't require additional storage device.
  - **Cons:**
    - Integration complexity - It might take a significant amount of time to develop the run scripts and log/results parser.
    - Higher maintenance effort to manage updates and dependencies. (More than 10 files)

Method 1 - Use the ACS Live Image

- Boot System Under Test (SUT) to a Linux OS, run a script with the commands below:
  - wget - download Architecture Compliance Suite (ACS) live image
  - xz - extract ACS image
  - dd – write image to a media device
  - efibootmgr – change boot order
  - Reboot – reboot the system to start automation test
  - Mount - mount the ACS media device to review the test result and logs

For details, please refer to the section "Integrate compliance tests into production processes" in SystemReady Band Integration and Testing Guide

Method 2 - Use Individual Binaries/ Test Suites

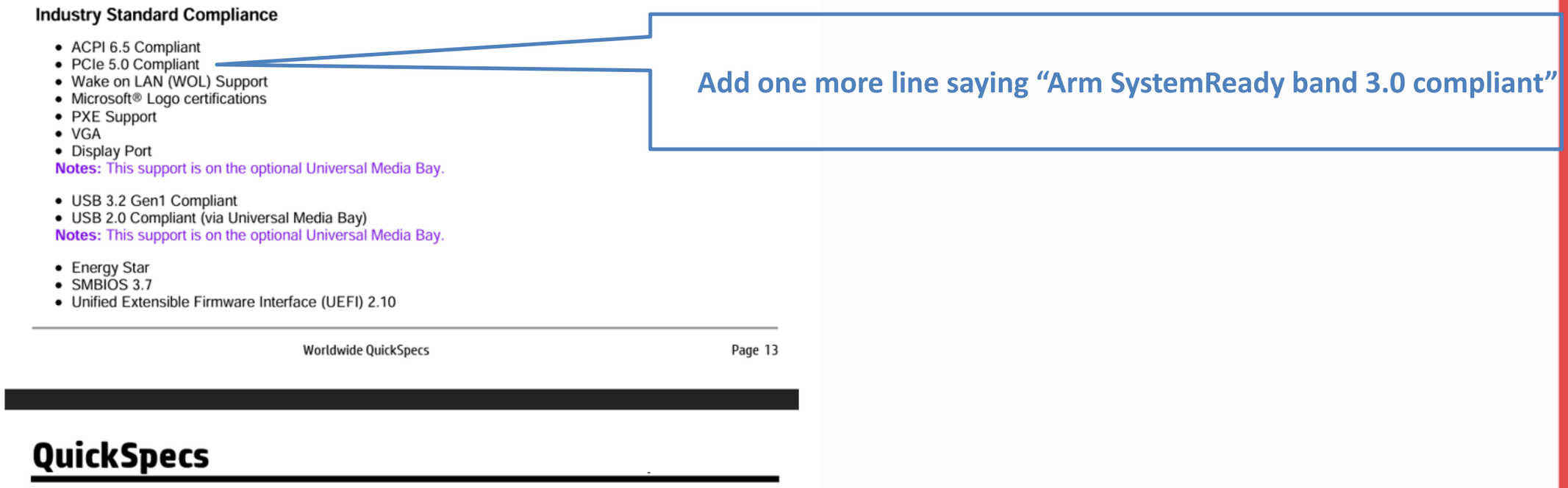
- Download or copy Individual test suite pre-built images (BSA, SBSA, SCT) along with their scripts (.nsh and .sh) from sysarch-acs and arm-systemready repos or from the ACS media device, and integrate them into your test processes.
- Install FWTS on the SUT's OS and download the run script(fwts.sh)
- Run scripts individually
- Parse the test result in /mnt/acs_results/ in ACS media device. For details about parsing, please see ACS log parser main script

For details, please refer to the section "Integrate compliance tests into production processes" in SystemReady Band Integration and Testing Guide

Partner’s Self-Declared Compliance

1. In the **product’s specification or product webpage,** mention the SystemReady band compliance in the section like “Feature” or “Industry Standard Compliance”.
2. In the **product launch blog or Press Release,** mention the compliance with wordmark like “our device XYZ is SystemReady Band v3.0 compliant” or “ABC company announced that its XYZ device has achieved compliance with SystemReady band v3.0”.

Note that if you need to use SystemReady logo (that is, not just the wordmark), please submit your request using the Contact Form



Industry Standard Compliance

- ACPI 6.5 Compliant
- PCIe 5.0 Compliant
- Wake on LAN (WOL) Support
- Microsoft® Logo certifications
- PXE Support
- VGA
- Display Port

Notes: This support is on the optional Universal Media Bay.

- USB 3.2 Gen1 Compliant
- USB 2.0 Compliant (via Universal Media Bay)

Notes: This support is on the optional Universal Media Bay.

- Energy Star
- SMBIOS 3.7
- Unified Extensible Firmware Interface (UEFI) 2.10

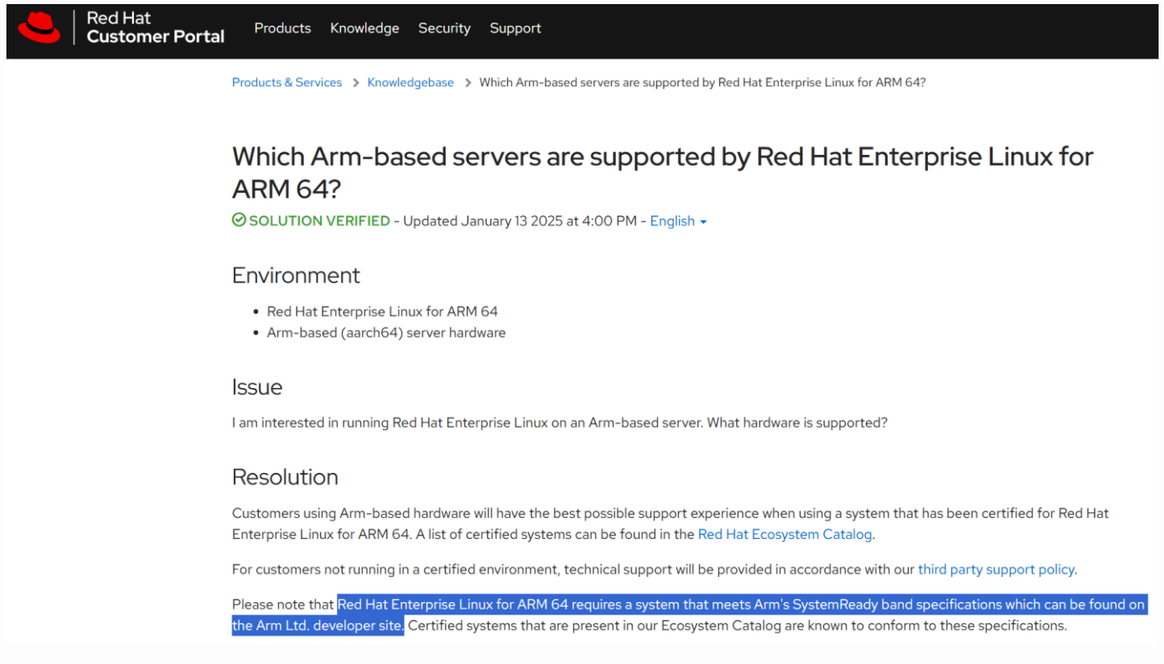
Worldwide QuickSpecs Page 13

QuickSpecs

Add one more line saying “Arm SystemReady band 3.0 compliant”

OSV & SiP Certification Transition

- Previously, some partners' certifications were based on SystemReady SR/ES certification. Arm is now working with OSVs and SiPs to shift that basis to self-declared compliance under the SystemReady band.
- Red Hat Enterprise Linux for ARM 64 requires a system that meets Arm's SystemReady band specifications. For details, please refer to https://access.redhat.com/solutions/6673691



Acronyms
- OSV : Operating System vendors
- SiP : Silicon Providers

Ongoing Integrations

- Silicon Providers' (SiPs') certification
  - Integration in progress
- OS vendors' (OSVs') certification
  - Integration under discussion
  - Red Hat Enterprise Linux for ARM 64 requires SystemReady band compliance
- Some ODMs/OEMs' release test process
  - Integration under discussion

Acronyms
- Original Equipment Manufacturer (OEM)
- Original Design Manufacturer (ODM)

Summary

- **By Integrating ACS tests, you will achieve the following benefits:**
  - Improve the quality of your implementation
  - Reduce development time for OS enablement and debugging
  - Meet SiP and OSV certification requirements more quickly
  - Easily turn test results into a SystemReady compliance declaration to promote your product

Call to Action

- **Integrate ACS tests** into your test suites or production processes
- **Share your blockers** – let us know what prevents you from Integrating
- **Contribute back** - report issues and improvements

Questions?