# Arm® SBSA Architecture Compliance

**Revision: r3p1**

## User Guide

arm


101547_0301_01_en

# Arm® SBSA Architecture Compliance

## User Guide



**Release Information**

**Document History**

| Issue | Date | Confidentiality | Change |
|---|---|---|---|
| A | 30 November 2016 | Non-Confidential | Alpha release |
| B | 31 March 2017 | Non-Confidential | Beta release |
| C | 13 July 2017 | Non-Confidential | REL 1.0 |
| D | 11 May 2018 | Non-Confidential | REL 2.0 |
| 0200-01 | 27 December 2018 | Non-Confidential | REL 2.1. The document now follows a new numbering format. |
| 0200-02 | 26 April 2019 | Non-Confidential | REL 2.2 |
| 0200-03 | 18 September 2019 | Non-Confidential | REL 2.3 |
| 0200-04 | 20 March 2020 | Non-Confidential | REL 2.4 |
| 0300-01 | 30 September 2020 | Non-Confidential | REL 3.0 |
| 0301-01 | 27 September 2021 | Non-Confidential | REL 3.1 |

Arm Limited. Company 02557590 registered in England.

110 Fulbourn Road, Cambridge, England CB1 9NJ.

(LES-PRE-20349)

**Confidentiality Status**

This document is Non-Confidential. The right to use, copy and disclose this document may be subject to license restrictions in accordance with the terms of the agreement entered into by Arm and the party that Arm delivered this document to.

Unrestricted Access is an Arm internal classification.

**Product Status**

The information in this document is Final, that is for a developed product.

**Web Address**

*developer.arm.com*

**Inclusive language commitment**

Arm values inclusive communities. Arm recognizes that we and our industry have used language that can be offensive. Arm strives to lead the industry and create change.

This document includes language that can be offensive. We will replace this language in a future issue of this document.

To report offensive language in this document, email *terms@arm.com*.

# Contents
# Arm® SBSA Architecture Compliance User Guide

# Preface

This preface introduces the *Arm® SBSA Architecture Compliance User Guide*.

It contains the following:

# About this book

This book is the user guide for Arm® SBSA architecture compliance.

## Using this book

This book is organized into the following chapters:

### Chapter 1 Introduction

This chapter provides an overview of the SBSA tests and the test IDs.

### Chapter 2 UEFI shell application

This chapter provides information on executing tests from the UEFI Shell application and its PAL API implementation.

### Chapter 3 Linux application

This chapter provides information on executing tests from the Linux application.

### Appendix A Revisions

This appendix describes the technical changes between released issues of this book.

## Glossary

The Arm® Glossary is a list of terms used in Arm documentation, together with definitions for those terms. The Arm Glossary does not contain terms that are industry standard unless the Arm meaning differs from the generally accepted meaning.

See the *Arm Glossary* for more information.

## Typographic conventions

*italic*

Introduces special terminology, denotes cross-references, and citations.

**bold**

Highlights interface elements, such as menu names. Denotes signal names. Also used for terms in descriptive lists, where appropriate.

`monospace`

Denotes text that you can enter at the keyboard, such as commands, file and program names, and source code.

`monospace`

Denotes a permitted abbreviation for a command or option. You can enter the underlined text instead of the full command or option name.

*`monospace italic`*

Denotes arguments to monospace text where the argument is to be replaced by a specific value.

**`monospace bold`**

Denotes language keywords when used outside example code.

`<and>`

Encloses replaceable terms for assembler syntax where they appear in code or code fragments. For example:

```
MRC p15, 0, <Rd>, <CRn>, <CRm>, <Opcode_2>
```

SMALL CAPITALS

Used in body text for a few terms that have specific technical meanings, that are defined in the *Arm® Glossary*. For example, IMPLEMENTATION DEFINED, IMPLEMENTATION SPECIFIC, UNKNOWN, and UNPREDICTABLE.

## Additional reading

This book contains information that is specific to this product. See the following documents for other relevant information.

**Arm publications**

- *Arm® Server Base System Architecture Specification* (DEN-0029)
- *Arm® Server Base Boot Requirements* (DEN-0044B)
- *Arm® Architecture Reference Manual ARMv8, for Armv8-A architecture profile* (DDI 0487F.a ID021920)
- *Arm® Generic Interrupt Controller Architecture Specification for GIC architecture version 3.0 and version 4.0* (IHI 0069C (ID070116))
- *GICv3 and GICv4 Software Overview* (DAI 0492)

**Other publications**

None.

## Feedback

### Feedback on this product

If you have any comments or suggestions about this product, contact your supplier and give:

- The product name.
- The product revision or version.
- An explanation with as much information as you can provide. Include symptoms and diagnostic procedures if appropriate.

### Feedback on content

If you have comments on content then send an e-mail to *support-enterprise-acs@arm.com*. Give:

- The title *Arm SBSA Architecture Compliance User Guide*.
- The number 101547_0301_01_en.
- If applicable, the page number(s) to which your comments refer.
- A concise explanation of your comments.

Arm also welcomes general suggestions for additions and improvements.

**Note**

Arm tests the PDF only in Adobe Acrobat and Acrobat Reader, and cannot guarantee the quality of the represented document when used with any other PDF reader.

## Other information

- *Arm® Developer*.
- *Arm® Documentation*.
- *Technical Support*.
- *Arm® Glossary*.

# Chapter 1
# **Introduction**

This chapter provides an overview of the SBSA tests and the test IDs.

It contains the following sections:

## 1.1 Abbreviations

This section lists the abbreviations used in this document.

**Table 1-1 Abbreviations and expansions**

| Abbreviation | Expansion |
|---|---|
| ACPI | Advanced Configuration and Power Interface |
| DT | Device Tree |
| GIC | Generic Interrupt Controller |
| IOMMU | Input-Output Memory Management Unit |
| PAL | Platform Abstraction Layer |
| PCIe | Peripheral Component Interconnect express |
| PE | Processing Element |
| RCiEP | Root Complex integrated End Point |
| SATA | Serial Advanced Technology Attachment |
| SBSA | Server Base System Architecture |
| SMC | Secure Monitor Call |
| SMMU | System Memory Management Unit |
| UEFI | Unified Extensible Firmware Interface |

## 1.2 Overview of tests

The following table describes the general divisions of Server Base System Architecture (SBSA) tests between Unified Extensible Firmware Interface (UEFI) shell application, Linux application, and Bare-metal.

**Table 1-2 Test environment and modules**

| Test environment | Modules |
|---|---|
| UEFI Shell | PE, GIC, Timers, Watchdog, Wakeup, PCIe, NIST, Peripherals, SMMU |
| Linux command line | PCIe, SMMU, Exerciser |
| Bare-metal | Exerciser |

## 1.3 Test IDs

Each test ID is generated as an addition of module ID and unit test ID. For a given module, unit test ID begins from 1.

The following table lists the module names and their IDs.

**Table 1-3 Module names and module IDs**

| Module name | Module ID |
|---|---|
| PE | 0 |
| GIC | 100 |
| Timer | 200 |
| Watchdog | 300 |
| PCIe | 400 |
| Power and Wakeup | 500 |
| Peripheral | 600 |
| SMMU | 700 |
| Exerciser | 800 |
| NIST | 1000 |

# Chapter 2
# UEFI shell application

This chapter provides information on executing tests from the UEFI Shell application and its PAL API implementation.

It contains the following sections:

## 2.1 UEFI application arguments

Run the UEFI Shell application with the following set of arguments:

```
uefi shell> sbsa.efi [-v <n>] [-l <n>] [-skip <x,y,z>] [-f <file name>] [-s] [-p <n>] [-
nist] [-mmio]
```

The following table provides descriptions to the arguments.

**Table 2-1 Descriptions of UEFI application arguments**

| Argument | Description |
|---|---|
| `-v` | Print level<br>**1** INFO and above.<br>**2** DEBUG and above.<br>**3** TEST and above.<br>**4** WARN and ERROR.<br>**5** ERROR.<br>**Note**<br>For print level 1, mmio read, write, and prints can be enabled for a specefic module by using module id. For example, -v11 enables for GIC module, and -v10 enables for PE module. |
| `-l` | Level of compliance to be tested for (0-6). The default value is 4. |
| `-skip` | Overrides the suite to skip the execution of a particular test. It allows a maximum of three values (comma-separated).<br>For example, 300 skips test case with ID = 300.<br>500 skips all tests in module with ID = 500.<br>For details on module IDs, see *1.3 Test IDs* on page 1-11. |
| `-f` | File name to which the output log is written. |
| `-s` | Runs Secure tests before executing Non-secure tests. It requires Secure firmware code from SBSA ACS to be ported to EL3 FW.<br>If this option is not provided, only Non-secure tests are run. |
| `-p` | Enables or disables the execution of SBSA v6.0 PCIe compliance tests (RCiEP rules).<br>Allowed values for `<n>` are 0 and 1, where 1 enables the PCIe tests and 0 disables it.<br>**Note**<br>• If this option is not provided, SBSA v6.0 PCIe (RCiEP rules) tests are not run.<br>• If `-l` has a value of 4 and above, these tests are always run. |
| `-nist` | Runs the SBSA ACS with NIST STS. |
| `-mmio` | Enables all the `mmio` read/write prints. |

**Note**

- The UEFI session becomes unusable after the SBSA tests are run and the test results are printed on the UEFI console.
- The UEFI Shell application is enhanced to accept an additional argument `[-p <n>]` for PCIe. This is to enable optionally running SBSA v6.0 PCIe tests even when the other tests run at older levels. For example, you can optionally run SBSA v6.0 PCIe tests even when running other SBSA tests at level 3.

### Example

```
shell> sbsa.efi –v 2 –l 3 –skip 20,36 –f acs.txt -p 1
```

The set of parameters shown in the code block:

- Prints messages with verbosity of 2 and above.
- Tests for compliance against SBSA level 3 for other tests and runs SBSA v6.0 PCIe (RCiEP rules) tests.
- Skips execution of all tests belonging to GIC module and test number 36.
- Stores the log messages to the file `acs.txt`.

## 2.2 UEFI implementation of PAL APIs

This section provides information on infrastructure APIs and module-specific APIs.

### Infrastructure APIs

The following table describes the Platform Abstraction Layer (PAL) APIs and UEFI interfaces.

**Table 2-2 PAL APIs and UEFI interfaces**

| PAL API | UEFI interfaces |
|---|---|
| `pal_print` | AsciiPrint |
| `mem_alloc` | gBS->AllocatePool |
| `mem_free` | gBS->FreePool |
| `mem_alloc_shared` | gBS->AllocatePool |
| `mem_free_shared` | gBS->FreePool |
| `mem_get_shared_addr` | None |
| `mmio_read` | None |
| `mmio_write` | None |

### Module-specific APIs

The following table represents the mapping of PAL API to Advanced Configuration and Power Interface (ACPI), if the system firmware presents platform configuration through ACPI tables.

**Table 2-3 PAL APIs, UEFI interfaces, and ACPI tables consumed**

| PAL API | UEFI interfaces consumed | ACPI table consumed |
|---|---|---|
| `pe_create_info_table` | • gST->ConfigurationTable<br>• CompareGuid<br>• IndustryStandard/Acpi61.h | MADT Table |
| `call_smc` | - | - |
| `pe_execute_payload` | - | - |
| `pe_install_esr` | • gEfiCpuArchProtocolGuid<br>• Cpu->RegisterInterruptHandler | - |
| `gic_create_info_table` | • gST->ConfigurationTable<br>• CompareGuid<br>• IndustryStandard/Acpi61.h | MADT table |
| `gic_install_isr` | • gHardwareInterruptProtocolGuid<br>• RegisterInterruptSource<br>• EnableInterruptSource | - |
| `timer_create_info_table` | • gST->ConfigurationTable<br>• CompareGuid<br>• IndustryStandard/Acpi61.h | GTDT table |

**Table 2-3 PAL APIs, UEFI interfaces, and ACPI tables consumed (continued)**

| PAL API | UEFI interfaces consumed | ACPI table consumed |
|---|---|---|
| `wd_create_info_table` | • gST->ConfigurationTable<br>• CompareGuid<br>• IndustryStandard/Acpi61.h | GTDT table |
| `pcie_create_info_table` | • gST->ConfigurationTable<br>• CompareGuid<br>• IndustryStandard/Acpi61.h | MCFG table |
| `pcie_get_mcfg_ecam` | • gST->ConfigurationTable<br>• CompareGuid, IndustryStandard/Acpi61.h<br>• IndustryStandard/ MemoryMappedConfigurationSpaceAccessTable.h | MCFG table |
| `iovirt_create_info_table` | • gST->ConfigurationTable<br>• CompareGuid<br>• IndustryStandard/Acpi61.h | IORT table |
| `peripheral_create_info_table` | • gEfiPciIoProtocolGuid<br>• Pci->GetLocation<br>• Pci->Pci.Read | - |
| `memory_create_info_table` | gBS->GetMemoryMap | - |

# Chapter 3
# **Linux application**

This chapter provides information on executing tests from the Linux application.

It contains the following sections:

## 3.1 Linux application arguments

Run the Linux application with the following set of arguments:

```
shell> sbsa [--v <n>] [--l <n>] [--skip <x,y,z>]
```

**Table 3-1 Description of Linux application arguments**

| Argument | Description |
|---|---|
| v | Print level<br>**1** INFO and above<br>**2** DEBUG and above<br>**3** TEST and above<br>**4** WARN and ERROR<br>**5** ERROR |
| l | Level of compliance to be tested for. (0 to 6) |
| skip | Overrides the suite to skip the execution of a particular test.<br>For example, 53 skips test case with ID 53. |

### Example

```
shell> sbsa --v 3 --l 3 --skip 57
```

This set of parameters tests for compliance against SBSA level 3 with print verbosity set to 3, and skips test number 57.

### Loading the kernel module

Before the SBSA ACS Linux application is run, load the SBSA ACS kernel module using the `insmod` command.

```
shell> insmod sbsa_acs.ko
```

## 3.2 Build steps and environment setup

This section lists the porting and build steps for the kernel module.

The patch for the kernel tree and the Linux PAL are hosted separately on *linux-acs*.

### Building the kernel module

**Prerequisites**

- Linux kernel source version 4.18.

```
git clone --branch v4.18 git@github.com:torvalds/linux.git
```

- Linaro GCC tool chain 7.5 or above.

```
export CROSS_COMPILE=<local_dir>/gcc-linaro-5.3-2016.02/bin/aarch64-linux-gnu-
```

- Build environment for AArch64 Linux kernel.

**Porting steps for Linux kernel**

1. `git clone https://git.gitlab.arm.com/linux-arm/linux-acs <local_dir/linux-acs>`
2. `git clone https://github.com/ARM-software/sbsa-acs.git <local_dir/sbsa-acs>`
3. Apply the `<local_dir/linux-acs>/kernel/src/0001-Enterprise-acs-linux-v4.18.patch` to your kernel source tree.
4. Build the kernel.

**Build steps for SBSA kernel module**

1. `cd <local_dir/linux-acs>/sbsa-acs-drv/files`
2. `export KERNEL_SRC=<linux kernel path>`
3. `./setup.sh <local_dir/sbsa-acs>`
4. `./linux_sbsa_acs.sh`

`sbsa_acs.ko` file is generated.

**SBSA Linux application build**

1. `cd <sbsa-acs path>/linux_app/sbsa-acs-app`
2. `make`

The executable file `sbsa` is generated.

This section contains the following subsections:

- *3.2.1 Target environment setup* on page 3-19.
- *3.2.2 Runtime environment* on page 3-19.

### 3.2.1 Target environment setup

The set of tests assumes that at least one Serial Advanced Technology Attachment (SATA) controller is behind a PCIe root complex. The SATA controller may or may not be behind an Input-Output Memory Management Unit (IOMMU).

Before running these tests, at least one SATA hard disk must be connected to the SATA controller. The test performs read and write operations to the SATA hard disk. Therefore, the data on the HDD is overwritten. The SATA drive must not be the boot device for the OS.

### 3.2.2 Runtime environment

The following figure describes the hardware functional blocks.

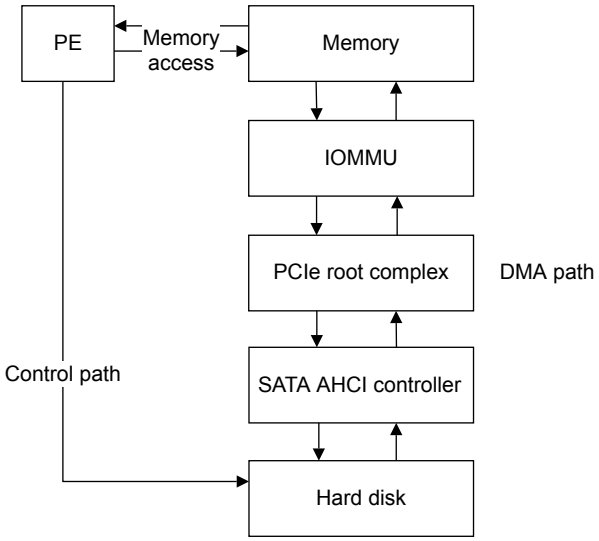


**Figure 3-1 Hardware functional blocks**

The PCIe-DMA tests initiate data transfers from a DMA requester. By default, the test searches for a SATA controller which is part of the PCIe subsystem.

1. The test programs the known data from the PE to main memory.
2. The test programs the DMA requester to transfer this known data to its end-point device.
3. The test programs the DMA requester to transfer the data back to a different location in the main memory.
4. The test compares the data at both the locations.

If the SATA controller is not placed before an IOMMU, then during this data transfer, the address that is used by the SATA controller is retrieved and compared with the DMA address that is seen by the PE.

If the DMA requester is placed before an IOMMU, then the address that is used by the SATA AHCI controller is compared with the address that is seen by the IOMMU. Both these addresses must match.

To enable the export of the addresses that are seen by the SATA AHCI controller and IOMMU, the kernel drivers for these two modules must be patched.

# Appendix A
# **Revisions**

This appendix describes the technical changes between released issues of this book.

It contains the following section:

- *A.1 Revisions* on page Appx-A-22.

## A.1 Revisions

The following tables describe the changes between different issues of this document.

**Table A-1 Issue 0200-01**

| Change | Location |
|---|---|
| Information about exerciser is added. | See *1.3 Test IDs* on page 1-11. |
| A new parameter `[--e]` is added to Linux application arguments. | See *3.1 Linux application arguments* on page 3-18. |

**Table A-2 Differences between Issue 0200-01 and Issue 0200-02**

| Change | Location |
|---|---|
| Bare-metal test environment is added to the table. | See *1.2 Overview of tests* on page 1-10. |
| A note about additional porting for the exerciser is added. | See *3.1 Linux application arguments* on page 3-18. |

**Table A-3 Differences between Issue 0200-02 and Issue 0200-03**

| Change | Location |
|---|---|
| No technical changes. | - |

**Table A-4 Differences between Issue 0200-03 and Issue 0200-04**

| Change | Location |
|---|---|
| • Arguments for NIST and PCIe tests are added.<br>• A note about UEFI session is added. | See *2.1 UEFI application arguments* on page 2-13. |
| NIST module ID is updated. | See *1.3 Test IDs* on page 1-11. |
| Linux application arguments are updated. | See *3.1 Linux application arguments* on page 3-18. |

**Table A-5 Differences between Issue 0200-04 and Issue 0300-01**

| Change | Location |
|---|---|
| Additional level of compliance to be tested is added. | See table in *2.1 UEFI application arguments* on page 2-13 and *3.1 Linux application arguments* on page 3-18. |

**Table A-6 Differences between Issue 0300-01 and Issue 0301-01**

| Change | Location |
|---|---|
| Removed Secure module. | See *1.3 Test IDs* on page 1-11. |
| Updated the link to linux-acs. | See *3.2 Build steps and environment setup* on page 3-19. |
| Updated the build steps and environment setup. | See *3.2 Build steps and environment setup* on page 3-19. |