arm

Arm® SBSA Architecture Compliance

Revision: r7p1

User Guide

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
0302-01	26 July 2022	Non-Confidential	REL 3.2
0601-01	28 October 2022	Non-Confidential	REL 6.1
0700-01	15 June 2022	Non-Confidential	REL 7.0 ALPHA release

Issue	Date	Confidentiality	Change
0701-01	16 January 2023	Non-Confidential	REL 7.1 BETA-0 release

Proprietary Notice

This document is protected by copyright and other related rights and the practice or implementation of the information contained in this document may be protected by one or more patents or pending patent applications. No part of this document may be reproduced in any form by any means without the express prior written permission of Arm. No license, express or implied, by estoppel or otherwise to any intellectual property rights is granted by this document unless specifically stated.

Your access to the information in this document is conditional upon your acceptance that you will not use or permit others to use the information for the purposes of determining whether implementations infringe any third party patents.

THIS 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. 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, patents, copyrights, trade secrets, or other rights.

This document may include technical inaccuracies or typographical errors.

TO THE EXTENT NOT PROHIBITED BY LAW, IN NO EVENT WILL ARM BE LIABLE FOR ANY DAMAGES, INCLUDING WITHOUT LIMITATION ANY DIRECT, INDIRECT, SPECIAL, INCIDENTAL, PUNITIVE, OR CONSEQUENTIAL DAMAGES, HOWEVER CAUSED AND REGARDLESS OF THE THEORY OF LIABILITY, ARISING OUT OF ANY USE OF THIS DOCUMENT, EVEN IF ARM HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

This document consists solely of commercial items. You shall be responsible for ensuring that any use, duplication or disclosure of this document complies fully with any relevant export laws and regulations to assure that this document or any portion thereof is not exported, directly or indirectly, in violation of such export laws. Use of the word "partner" in reference to Arm's customers is not intended to create or refer to any partnership relationship with any other company. Arm may make changes to this document at any time and without notice.

This document may be translated into other languages for convenience, and you agree that if there is any conflict between the English version of this document and any translation, the terms of the English version of the Agreement shall prevail.

The Arm corporate logo and words marked with ® or ™ are registered trademarks or trademarks of Arm Limited (or its affiliates) in the US and/or elsewhere. All rights reserved. Other brands and

names mentioned in this document may be the trademarks of their respective owners. Please follow Arm's trademark usage guidelines at https://www.arm.com/company/policies/trademarks.

Copyright © 2016–2023 Arm Limited (or its affiliates). All rights reserved.

Arm Limited. Company 02557590 registered in England.

110 Fulbourn Road, Cambridge, England CB1 9NJ.

(LES-PRE-20349|version 21.0)

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 for a Beta product, that is a product under development.

Feedback on content

Information about how to give feedback on the content.

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

- The title Arm® SBSA Architecture Compliance User Guide.
- The number 101547_0701_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.

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

1. Introduction

1.1 Conventions

The following subsections describe conventions used in Arm documents.

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: developer.arm.com/glossary.

Typographic conventions

Convention	Use
italic	Citations.
bold	Highlights interface elements, such as menu 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.
<u>mono</u>`space`	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.
`<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.

1.2 Useful resources

This document contains information that is specific to this product. See the following resources for other useful information.

Access to Arm documents depends on their confidentiality:

- Non-Confidential documents are available at developer.arm.com/documentation. Each document link in the following tables goes to the online version of the document.

- Confidential documents are available to licensees only through the product package.

Arm product resources	Document ID	Confidentiality
Arm® Server Base System Architecture 7.1	DEN0029H	Non-Confidential
GICv3 and GICv4 Software Overview	DAI0492	Non-Confidential

Arm architecture and specifications	Document ID	Confidentiality
Arm® Architecture Reference Manual for A-profile architecture	DDI0487I.a	Non-Confidential
Arm® Generic Interrupt Controller Architecture Specification for GIC architecture version 3.0 and version 4.0	IHI0069C	Non-Confidential

> **Note**
>
> Arm tests its PDFs only in Adobe Acrobat and Acrobat Reader. Arm cannot guarantee the quality of its documents when used with any other PDF reader.
>
> Adobe PDF reader products can be downloaded at http://www.adobe.com

1.3 Other information

See the Arm® website for other relevant information.

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

2. Overview of the SBSA tests

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

2.1 Abbreviations

The following table lists the abbreviations used in this document.

Table 2-1: Abbreviations and expansions

Abbreviation	Expansion
ACPI	Advanced Configuration and Power Interface
ACS	Architecture Compliance Suite
AEST	Arm Error Source Table
BDF	Bus, Device, and Function
CATU	CoreSight Address Translation Unit
ELx	Exception Level x (where x can be 0 to 3)
ETR	Embedded Trace Router
GIC	Generic Interrupt Controller
GCD	Grand Central Dispatch
HMAT	Heterogeneous Memory Attribute Table
HVC	HyperVisor Call
IOMMU	Input-Output Memory Management Unit
ITS	Interrupt Translation Service
LPI	Locality-specific Peripheral Interrupt
MPAM	Memory System Resource Partitioning and Monitoring
MSI	Message-Signaled Interrupt
MTE	Memory Tagging Extension
NUMA	Non-Uniform Memory Access
PAL	Platform Abstraction Layer
PCIe	Peripheral Component Interconnect express
PCCT	Platform Communications Channel Table
APMT	Performance Monitoring Unit Table
PE	Processing Element
PMU	Performance Monitor Unit
PPTT	Processor Properties Topology Table
PSCI	Power State Coordination Interface
RAS	Reliability, Availability, and Serviceability
RAS2	Reliability, Availability, and Serviceability 2
RCiEP	Root Complex integrated End Point

Abbreviation	Expansion
SATA	Serial Advanced Technology Attachment
SBSA	Server Base System Architecture
SMC	Secure Monitor Call
SMMU	System Memory Management Unit
SRAT	System Resource Affinity Table
STS	Statistical Test Suite
SoC	System on Chip
UEFI	Unified Extensible Firmware Interface
UART	Universal Asynchronous Receiver and Transmitter
VAL	Validation Abstraction Layer

2.2 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 2-2: Test environment and modules

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

2.3 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 2-3: 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

Module name	Module ID
SMMU	700
Exerciser	800
NIST	900
MPAM	1000
PMU	1100
RAS	1200
Memory	1300

3. UEFI shell application

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

3.1 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>] [-p <n>] [-
nist] [-mmio] [-p2p] [-cache] [-t <test id>] [-m <module id>] [-timeout <wakeup test
 timeout multiple>]
```

The following table provides descriptions to the arguments.

Table 3-1: Descriptions of UEFI application arguments

Argument	Description
-v	Print level **1** INFO and above. **2** DEBUG and above. **3** TEST and above. **4** WARN and ERROR. **5** ERROR.
-l	Level of compliance to be tested for (0-7). 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). For example, 300 skips test case with ID = 300. 500 skips all tests in module with ID = 500. For more information on module IDs, see 2.3 Test IDs on page 10.
-f	File name to which the output log is written.
-p	Enables or disables the execution of SBSA v7.0 PCIe compliance tests (RCiEP rules). Allowed values for <n> are 0 and 1, where 1 enables the PCIe tests and 0 disables it. **Note:** • If this option is not provided, SBSA v7.0 PCIe (RCiEP rules) tests are not run. • If -l has a value of 4 and above, these tests are always run.
-nist	Runs the SBSA ACS with NIST STS.

Argument	Description
`-mmio`	Enables all the mmio read/write prints. **Note:** • To enable pal_mmio_read or write prints, use with `-v 1`. • Enables prints from specefic module by using module id. For example, `-mmio 100`, enables for GIC module, and `-mmio 0`, enables for PE module.
`-p2p`	Enables p2p feature support.
`-cache`	Enables cache support.
`-t`	To run only single selected test. **Note:** `-m` will override `-t` if used on the same module.
`-m`	To run only single selected module
`-timeout`	Timeout value for wakeup test.

> **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 v7.0 PCIe tests even when the other tests run at older levels. For example, you can optionally run SBSA v7.0 PCIe tests even when running other SBSA tests at level 3.

Example 1

```
shell> sbsa.efi -v 2 -l 3 -skip 100,36 -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 v7.0 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`.

Example 2

```
shell > sbsa.efi -m 100 -skip 102
```

The set of parameters shown in the code block:

• Runs only the GIC module.

• Skips the GIC test 102.

3.2 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 3-2: 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
`mem_alloc_cachebale`	gBS->AllocatePages
`mem_free_cacheable`	gBS->FreePages
`time_delay_ms`	gBS->Stall
`mem_alloc_pages`	gBS->AllocatePages
`mem_free_pages`	gBS->FreePages
`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 3-3: PAL APIs, UEFI interfaces, and ACPI tables consumed

PAL API	UEFI interfaces consumed	ACPI table consumed
`pe_create_info_table`	• gST->ConfigurationTable • CompareGuid • IndustryStandard/Acpi.h	MADT Table
`pe_execute_payload`	-	-
`pe_install_esr`	• gEfiCpuArchProtocolGuid • Cpu->RegisterInterruptHandler	-
`gic_create_info_table`	• gST->ConfigurationTable • CompareGuid • IndustryStandard/Acpi.h	MADT table
`gic_install_isr`	• gHardwareInterruptProtocolGuid • RegisterInterruptSource • EnableInterruptSource	-

PAL API	UEFI interfaces consumed	ACPI table consumed
timer_create_info_table	• gST->ConfigurationTable • CompareGuid • IndustryStandard/Acpi.h	GTDT table
wd_create_info_table	• gST->ConfigurationTable • CompareGuid • IndustryStandard/Acpi.h	GTDT table
pcie_create_info_table	• gST->ConfigurationTable • CompareGuid • IndustryStandard/Acpi.h	MCFG table
pcie_get_mcfg_ecam	• gST->ConfigurationTable • CompareGuid, IndustryStandard/Acpi.h • IndustryStandard/MemoryMappedConfigurationSpaceAccessTable.h	MCFG table
iovirt_create_info_table	• gST->ConfigurationTable • CompareGuid • IndustryStandard/Acpi.h	IORT table
peripheral_create_info_table	• gEfiPciIoProtocolGuid • Pci->GetLocation • Pci->Pci.Read	-
memory_create_info_table	gBS->GetMemoryMap	-
cache_create_info_table	• IndustryStandard/Acpi.h • gST->ConfigurationTable • CompareGuid	PPTT table
hmat_create_info_table	• IndustryStandard/Acpi.h • gST->ConfigurationTable • CompareGuid	HMAT table
ras_create_info_table	• gST->ConfigurationTable • CompareGuid • IndustryStandard/ArmErrorSourceTable.h	AEST table
mpam_create_info_table	• gST->ConfigurationTable • CompareGuid • IndustryStandard/Acpi.h	MPAM table
pmu_create_info_table	• gST->ConfigurationTable • CompareGuid • IndustryStandard/Acpi.h	APMT table
ras2_create_info_table	• gST->ConfigurationTable • CompareGuid • IndustryStandard/Acpi.h	RAS2 and PCCT table

PAL API	UEFI interfaces consumed	ACPI table consumed
`srat_create_info_table`	• gST->ConfigurationTable • CompareGuid • IndustryStandard/Acpi.h	SRAT table

4. Linux application

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

4.1 Linux application arguments

Run the Linux application with the following set of arguments:

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

Table 4-1: Description of Linux application arguments

Argument	Description
v	Print level 1 INFO and above 2 DEBUG and above 3 TEST and above 4 WARN and ERROR 5 ERROR
l	Level of compliance to be tested for. (0 to 7)
skip	Overrides the suite to skip the execution of a particular test. 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
```

4.2 Environment setup

This section details the target and runtime environment setup.

4.2.1 Test requirements

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.

4.2.2 Runtime environment

The following figure describes the hardware functional blocks.

Figure 4-1: Hardware functional blocks
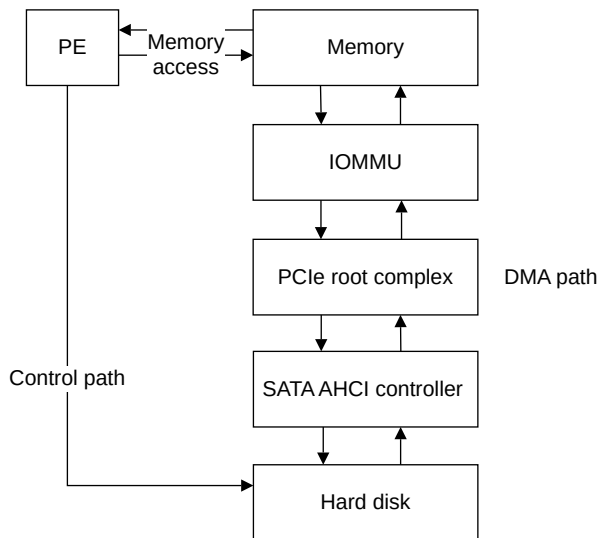
The PCIe-DMA tests initiate data transfers from a DMA requester. By default, the test searches for a SATA controller which is part of the PCIe subsystem.

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.

5. PMU Linux application

This chapter describes on how to run the PMU Linux application.

5.1 Running the PMU application

The following steps describe on how to run the PMU application.

Steps to run the application

1. Export the path from which the Python modules are present.

    ```
    export PYTHONPATH=/lib/python3.10/site-packages/
    ```

2. Navigate to the directory where the scripts are present.

    ```
    cd /bin/pmuval
    ```

3. Run script

    ```
    python sbsa_acs_pmu.py -a
    ```

    ```
    Options -: -a -- All PEs
    ```

Note	For more information on how to compile the PMU application, see the *README* document.

Appendix A Revisions

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

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 2.3 Test IDs on page 10.
A new parameter [--e] is added to Linux application arguments.	See 4.1 Linux application arguments on page 17.

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

Change	Location
Bare-metal test environment is added to the table.	See 2.2 Overview of tests on page 10.
A note about additional porting for the exerciser is added.	See 4.1 Linux application arguments on page 17.

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. • A note about UEFI session is added.	See 3.1 UEFI application arguments on page 12.
NIST module ID is updated.	See 2.3 Test IDs on page 10.
Linux application arguments are updated.	See 4.1 Linux application arguments on page 17.

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 3.1 UEFI application arguments on page 12 and 4.1 Linux application arguments on page 17.

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

Change	Location
Removed Secure module.	See 2.3 Test IDs on page 10.
Updated the link to linux-acs.	See 4.2 Environment setup on page 17.
Updated the build steps and environment setup.	See 4.2 Environment setup on page 17.

Table A-7: Differences between Issue 0301-01 and Issue 0302-01

Change	Location
Arguments for p2p and cache are added.	See 3.1 UEFI application arguments on page 12.

Table A-8: Differences between Issue 0302-01 and Issue 0601-01

Change	Location
Added an abbreviation for HVC.	See 2.1 Abbreviations on page 9.
Removed Exerciser module from the Linux command line.	See 2.2 Overview of tests on page 10.
Added new argument options for test id, module id, and timeout with more examples.	See 3.1 UEFI application arguments on page 12.
Section on Build steps and environment setup is moved to the README file.	-

Table A-9: Differences between Issue 0300-01 and Issue 0700-01

Change	Location
Added information on PMU Linux application	See 5. PMU Linux application on page 20.
Added new terms in Abbreviations and new APIs in UEFI implementation of PAL APIs.	See 2.1 Abbreviations on page 9, 3.2 UEFI implementation of PAL APIs on page 14.
Added details for PMU, MPAM, and RAS; removed Exerciser module.	See 2.2 Overview of tests on page 10.
Updated the arguments and description for UEFI application.	See 3.1 UEFI application arguments on page 12.
Added new module names and IDs.	See 2.3 Test IDs on page 10.
Added new PAL APIs.	See 3.2 UEFI implementation of PAL APIs on page 14.
Updated the commands and versions.	See 4.2 Environment setup on page 17.

Table A-10: Differences between Issue 0700-01 and Issue 0701-01

Change	Location
Added details for Memory module.	See 2.2 Overview of tests on page 10.
Updated the arguments and description for UEFI application.	See 3.1 UEFI application arguments on page 12.
Added a new module Memory.	See 2.3 Test IDs on page 10.
Added RAS and SRAT APIs.	See 3.2 UEFI implementation of PAL APIs on page 14.
Added new terms in Abbreviations.	See 2.1 Abbreviations on page 9.
Updated the steps to run the PMU application.	See 5.1 Running the PMU application on page 20.