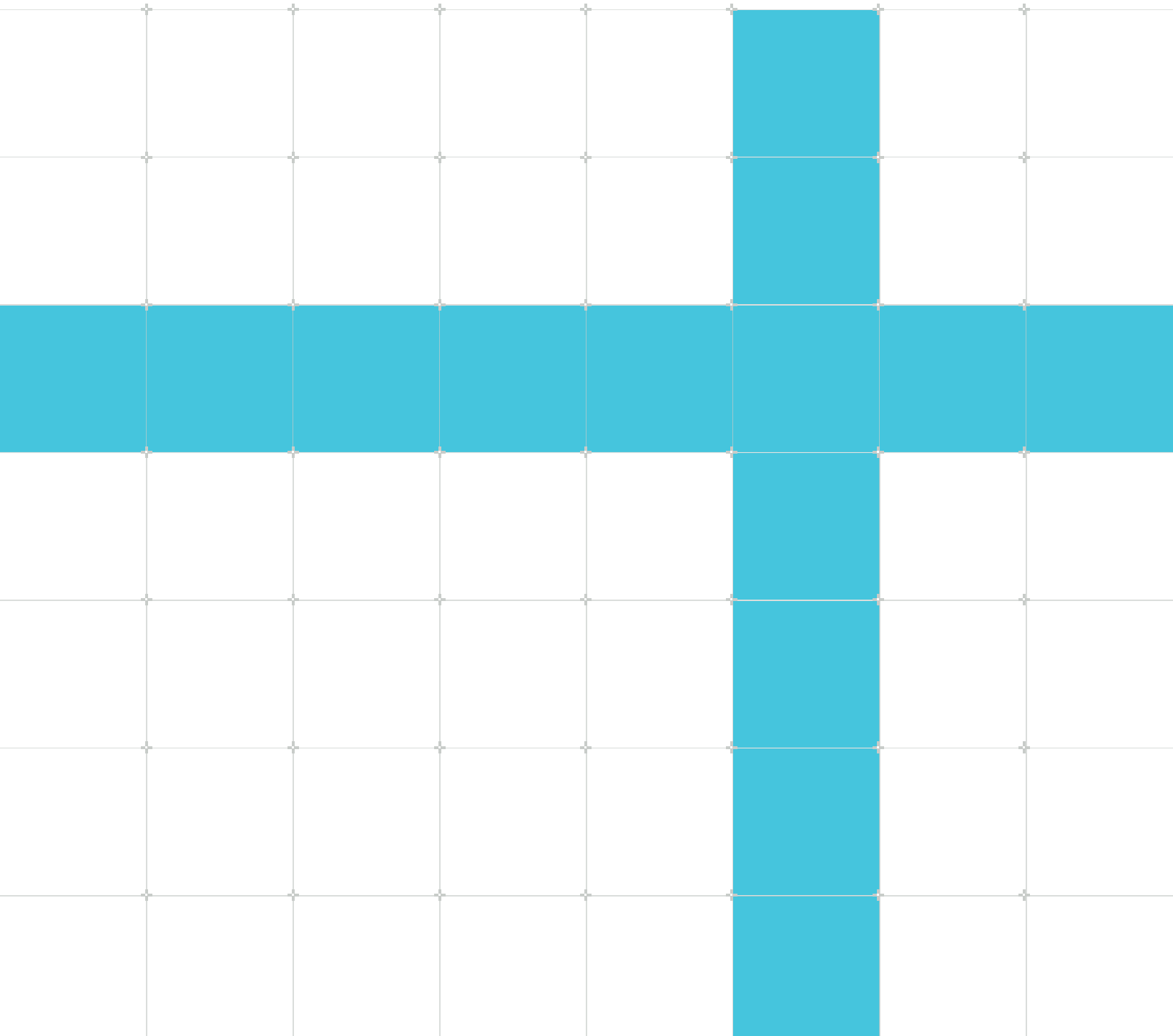
Arm® SBSA Architecture Compliance

Version 8.0.0

User Guide

Non-Confidential

Issue 01
101547_0800_01_en



Arm® SBSA Architecture Compliance User Guide

This document (101547_0800_01_en) was issued on 2025-12-10. There might be a later issue at https://developer.arm.com/documentation/101547

The product version is 8.0.0.

See also: Proprietary Notice | Product and document information | Useful resources

Start reading

If you prefer, you can skip to the start of the content.

Intended audience

This book is written for engineers who are designing or verifying an implementation of the Arm® Server Base System Architecture.

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.

Feedback

Arm welcomes feedback on this product and its documentation. To provide feedback on the product, create a ticket on https://support.developer.arm.com.

To provide feedback on the document, fill the following survey: https://developer.arm.com/documentation-feedback-survey.

Contents

1. Overview of the SBSA test suites

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

1.1 Abbreviations

The following table lists the abbreviations used in this document.

Table 1-1: 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
ETE	Embedded Trace Extension
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

Abbreviation	Expansion
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
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

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, PMU, MPAM, RAS, Memory, and ETE
Linux command line	PCIe, SMMU, PMU
Bare-metal	Exerciser

2. UEFI shell application

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

2.1 UEFI application arguments

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

```
uefi shell>  Sbsa.efi [-v <n>] [-f <filename>] [-l <n>] [-only <n>] [-r <RuleIDs|
rules.txt>] [-skip <RuleIDs>] [-m <modules>][-skipmodule <modules>] [-mmio] [-
no_crypto_ext] [-p2p] [-cache] [-el1physkip] [-skip-dp-nic-ms] [-slc <n>] [-timeout
 <n>][-h|-help] [-fr]
```

The following table provides descriptions to the arguments.

Table 2-1: Descriptions of UEFI application arguments

Argument	Description
`-cache`	Pass this flag to indicate that the system supports PCIe Address Translation Cache (ATC).
`-el1physkip`	Skips EL1 register checks. Use this option for debug purposes in VE or EL1 only scenarios.
`-f <filename>`	File name to which the output log is written.
`-fr`	Runs rules up to the Future Requirements (FR) level.
`-h, -help`	Prints the help message and exits.
`-l`	Level of compliance to be tested for. The default value is 1.
`-m <modules>`	Runs only the specified modules. Provide a comma-separated list. Accepted values: PE, GIC, PERIPHERAL, MEM_MAP, PMU, RAS, SMMU, TIMER, WATCHDOG, NIST, PCIE, MPAM, ETE, TPM, and POWER_WAKEUP.
`-mmio`	To enable pal_mmio_read or write prints, use with -v 1.
`-no_crypto_ext`	Use this when cryptography extensions are unavailable due to export restrictions.
`-only <n>`	Runs only the tests for level n.
`-p2p`	Pass this flag to indicate system support PCIe p2p.
`-r <RuleIDs|rules.txt>`	Runs only the specified rules. You can provide a comma-separated list of Rule IDs (for example: `B_PE_01`, `B_GIC_01`) or a rules file. The rules file may contain comma-separated or newline-separated Rule IDs. Lines that begin with # are treated as comments and ignored. **Note:** Rule ID string passed must match the BSA specification.
`-skip <RuleIDs>`	Skips the specified rules. Provide a comma-separated list of Rule IDs, using the same format as the `-r` option. Example: `-skip B_PE_01,B_GIC_02`

Argument	Description
`-skip-dp-nic-ms`	Skips PCIe tests for DisplayPort, network, and mass storage devices.
`-skipmodule <modules>`	Skips the specified modules. Provide a comma-separated list, using the same format as the `-m` option.
`-slc <n>-slc <n>`	System last-level cache type. **1** PPTT PE-side cache. **2** HMAT mem-side cache.
`-timeout <n>`	Timeout value for the wakeup test. **1** Minimum value. **5** Maximum value. **1** Default.
`-v`	Print level **1** INFO and above. **2** DEBUG and above. **3** TEST and above. **4** WARN and ERROR. **5** ERROR.


Note

The UEFI session becomes unusable after the SBSA tests are run and the test results are printed on the UEFI console.

Example 1

```
shell> Sbsa.efi -v 2 -l 3 -skip B_PE_01,B_SMMU_01 -skipmodule GIC,TIMER -f acs.txt
```

The set of parameters shown in the code block:

- Prints messages with verbosity of 2 and above.
- Tests for compliance against SBSA level 3.
- Skips execution of all rules belonging to Generic Interrupt Controller (GIC) and Timer modules and tests for rule IDs B_PE_01 and B_SMMU_01.

- Stores the log messages to the file `acs.txt`.

Example 2

```
shell > Sbsa.efi -m PCIE -skip B_PCIe_10
```

The set of parameters shown in the code block:

- Runs only PCIE module rules.
- Skips PCIE rule B_PCIe_10.

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
`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
`aligned_alloc`	gBS->AllocatePool
`mem_free_aligned`	gBS->FreePool

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 • 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	-
`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

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

3. Linux application

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

3.1 Linux application arguments

Run the Linux application with the following set of arguments:

```
shell> sbsa [-v <n>] [-l <n>] [--only <n>] [-r <ruleids>] [--fr] [--skip <ruleids>
 [--skip-dp-nic-ms] [-h]
```

Table 3-1: Description of Linux application arguments

Argument	Description
`-h`	Prints help message and exits.
`-l`	Level of compliance to be tested for. The default value is 1.
`--only <n>`	Runs only the tests for level n.
`-r <RuleIDs>`	Runs only the specified rules. You can provide a comma-separated list of Rule IDs (for example: `B_PE_01`, `B_GIC_01`). **Note:** Rule ID string passed must match the BSA specification.
`-fr`	Runs rules up to the Future Requirements (FR) level.
`--skip <RuleIDs>`	Skips the specified rules. Provide a comma-separated list of Rule IDs, using the same format as the `-r` option. Example: `-skip B_PE_01,B_GIC_02`
`--skip-dp-nic-ms`	Skips PCIe tests for DisplayPort, network, and mass storage devices.

Example

```
shell> sbsa -v 3 -l 3 --skip PCI_IN_01
```

This set of parameters tests for compliance against SBSA level 3 with print verbosity set to 3, and skips test for rule ID PCI_IN_01.

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 Environment setup

This section details the target and runtime environment setup.

3.2.1 Test requirements

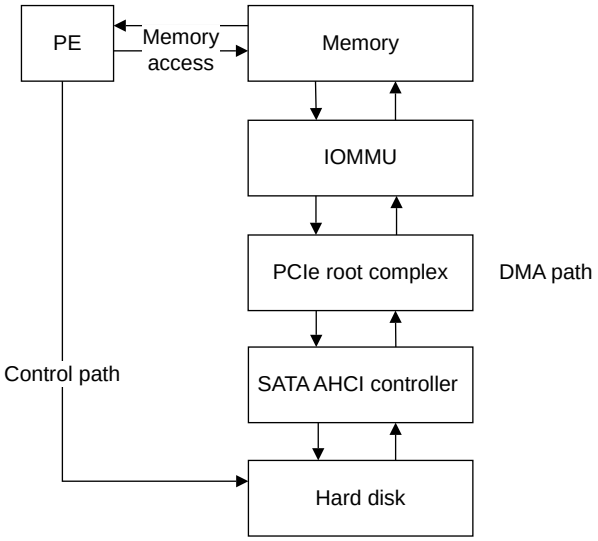
The set of tests assumes that at least one Serial Advanced Technology Attachment (SATA) controller is behind a PCIe root complex. The SATA controller may or may not be behind an Input-Output Memory Management Unit (IOMMU).

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.

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.

4. PMU Linux application

This chapter describes how to run the PMU Linux application.

4.1 Running the PMU application

The following steps describe how to run the PMU application.

To run the application, follow these steps:

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
   ```

   Option `-a` to run on all PEs.

Proprietary Notice

This document is protected by copyright and other related rights and the use 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 Limited ("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 the subject matter of this document infringes any third party patents.

The content of this document is informational only. Any solutions presented herein are subject to changing conditions, information, scope, and data. This document was produced using reasonable efforts based on information available as of the date of issue of this document. The scope of information in this document may exceed that which Arm is required to provide, and such additional information is merely intended to further assist the recipient and does not represent Arm's view of the scope of its obligations. You acknowledge and agree that you possess the necessary expertise in system security and functional safety and that you shall be solely responsible for compliance with all legal, regulatory, safety and security related requirements concerning your products, notwithstanding any information or support that may be provided by Arm herein. In addition, you are responsible for any applications which are used in conjunction with any Arm technology described in this document, and to minimize risks, adequate design and operating safeguards should be provided for by you.

This document may include technical inaccuracies or typographical errors. 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, any patents, copyrights, trade secrets, trademarks, or other rights.

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.

Reference by Arm to any third party's products or services within this document is not an express or implied approval or endorsement of the use thereof.

This document consists solely of commercial items. You shall be responsible for ensuring that any permitted 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 this document shall prevail.

The validity, construction and performance of this notice shall be governed by English Law.

Arm Limited. Company 02557590 registered in England.

110 Fulbourn Road, Cambridge, England CB1 9NJ.

PRE-1121-V1.0

Product and document information

Read the information in these sections to understand the release status of the product and documentation, and the conventions used in the Arm documents.

Product status

All products and Services provided by Arm require deliverables to be prepared and made available at different levels of completeness. The information in this document indicates the appropriate level of completeness for the associated deliverables.

Product completeness status

The information in this document is for a Beta product, that is a product under development.

Revision history

These sections can help you understand how the document has changed over time.

Document release information

The Document history table gives the issue number and the released date for each released issue of this document.

Document history

Issue	Date	Confidentiality	Change
0800-01	10 December 2025	Non-Confidential	REL 8.0.0 BETA release
0701-07	28 May 2025	Non-Confidential	REL 7.2.2 EAC release
0701-06	29 October 2024	Non-Confidential	REL 7.2.1 EAC release
0701-05	29 March 2024	Non-Confidential	REL 7.2.0 BETA release
0701-04	28 September 2023	Non-Confidential	REL 7.1.3 EAC release
0701-03	29 June 2023	Non-Confidential	REL 7.1.2 EAC release

Issue	Date	Confidentiality	Change
0701-02	28 March 2023	Non-Confidential	REL 7.1.1 BETA-1 release
0701-01	16 January 2023	Non-Confidential	REL 7.1 BETA-0 release
0700-01	15 June 2022	Non-Confidential	REL 7.0 ALPHA release
0601-01	28 October 2022	Non-Confidential	REL 6.1
0302-01	26 July 2022	Non-Confidential	REL 3.2
0301-01	27 September 2021	Non-Confidential	REL 3.1
0300-01	30 September 2020	Non-Confidential	REL 3.0
0200-04	20 March 2020	Non-Confidential	REL 2.4
0200-03	18 September 2019	Non-Confidential	REL 2.3
0200-02	26 April 2019	Non-Confidential	REL 2.2
0200-01	27 December 2018	Non-Confidential	REL 2.1. The document now follows a new numbering format.
D	11 May 2018	Non-Confidential	REL 2.0
C	13 July 2017	Non-Confidential	REL 1.0
B	31 March 2017	Non-Confidential	Beta release
A	30 November 2016	Non-Confidential	Alpha release

The Change history tables describe the technical changes between released issues of this document in reverse order. Issue numbers match the revision history in Document release information on page 17.

Table 2: Issue 0200-01

Change	Location
Information about exerciser is added.	See, Test IDs.
A new parameter `[--e]` is added to Linux application arguments.	See, 3.1 Linux application arguments on page 11.

Table 3: 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 5.
A note about additional porting for the exerciser is added.	See, 3.1 Linux application arguments on page 11.

Table 4: Differences between Issue 0200-02 and Issue 0200-03

Change	Location
No technical changes.	-

Table 5: 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, 2.1 UEFI application arguments on page 6.
NIST module ID is updated.	See, Test IDs.
Linux application arguments are updated.	See, 3.1 Linux application arguments on page 11.

Table 6: 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 6 and 3.1 Linux application arguments on page 11.

Table 7: Differences between Issue 0300-01 and Issue 0301-01

Change	Location
Removed Secure module.	See, Test IDs.
Updated the link to linux-acs.	See, 3.2 Environment setup on page 11.
Updated the build steps and environment setup.	See, 3.2 Environment setup on page 11.

Table 8: Differences between Issue 0301-01 and Issue 0302-01

Change	Location
Arguments for p2p and cache are added.	See, 2.1 UEFI application arguments on page 6.

Table 9: Differences between Issue 0302-01 and Issue 0601-01

Change	Location
Added an abbreviation for HVC.	See, 1.1 Abbreviations on page 4.
Removed Exerciser module from the Linux command line.	See, 1.2 Overview of tests on page 5.
Added new argument options for test id, module id, and timeout with more examples.	See, 2.1 UEFI application arguments on page 6.
Section on Build steps and environment setup is moved to the README file.	-

Table 10: Differences between Issue 0601-01 and Issue 0700-01

Change	Location
Added information on PMU Linux application	See, 4. PMU Linux application on page 14.
Added new terms in Abbreviations and new APIs in UEFI implementation of PAL APIs.	See, 1.1 Abbreviations on page 4, 2.2 UEFI implementation of PAL APIs on page 8.
Added details for PMU, MPAM, and RAS; removed Exerciser module.	See, 1.2 Overview of tests on page 5.
Updated the arguments and description for UEFI application.	See, 2.1 UEFI application arguments on page 6.
Added new module names and IDs.	See, Test IDs.
Added new PAL APIs.	See, 2.2 UEFI implementation of PAL APIs on page 8.
Updated the commands and versions.	See, 3.2 Environment setup on page 11.

Table 11: Differences between Issue 0700-01 and Issue 0701-01

Change	Location
Added details for Memory module.	See, 1.2 Overview of tests on page 5.
Updated the arguments and description for UEFI application.	See, 2.1 UEFI application arguments on page 6.
Added a new module Memory.	See, Test IDs.
Added RAS and SRAT APIs.	See, 2.2 UEFI implementation of PAL APIs on page 8.
Added new terms in Abbreviations.	See, 1.1 Abbreviations on page 4.
Updated the steps to run the PMU application.	See, 4.1 Running the PMU application on page 14.

Table 12: Differences between Issue 0701-01 and Issue 0701-02

Change	Location
Changed the order of modules executed in the Table 2-3: Test environment and modules.	See, 1.2 Overview of tests on page 5
Updated the UEFI application arguments.	See, 2.1 UEFI application arguments on page 6
Added new PAL APIs in Table 3-2: PAL APIs and UEFI interfaces.	See, 2.2 UEFI implementation of PAL APIs on page 8

Table 13: Differences between Issue 0701-02 and Issue 0701-03

Change	Location
Updated the arguments and descriptions for UEFI Shell application and examples.	See, 2.1 UEFI application arguments on page 6
Updated the Linux application arguments.	See, 3.1 Linux application arguments on page 11

Table 14: Differences between Issue 0701-03 and Issue 0701-04

Change	Location
No technical changes	-

Table 15: Differences between Issue 0701-04 and Issue 0701-05

Change	Location
Updated the arguments and descriptions for UEFI Shell application and examples.	See, 2.1 UEFI application arguments on page 6
Added ETE module name	See, 1.2 Overview of tests on page 5
Added ETE module name and ID.	See,Test IDs.

Table 16: Differences between Issue 0701-05 and Issue 0701-06

Change	Location
Updated the UEFI application arguments.	See, 2.1 UEFI application arguments on page 6
Added a new PAL API in Table 3-3: PAL APIs and UEFI interfaces, and ACPI tables consumed.	See, 2.2 UEFI implementation of PAL APIs on page 8

Table 17: Differences between Issue 0701-06 and Issue 0701-07

Change	Location
Updated the UEFI application arguments.	See, 2.1 UEFI application arguments on page 6

Table 18: Differences between Issue 0701-07 and Issue 0800-01

Change	Location
Updated the UEFI application arguments section.	See, 2.1 UEFI application arguments on page 6
Updated the Linux application arguments section.	See, 3.1 Linux application arguments on page 11
Removed the test ids section.	-

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

Arm documentation uses typographical conventions to convey specific meaning.

Convention	Use
italic	Citations.
bold	Interface elements, such as menu names. Terms in descriptive lists, where appropriate.
`monospace`	Text that you can enter at the keyboard, such as commands, file and program names, and source code.
`monospace underline`	A permitted abbreviation for a command or option. You can enter the underlined text instead of the full command or option name.
`<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>`

Convention	Use
SMALL CAPITALS	Terms that have specific technical meanings as defined in the *Arm® Glossary*. For example, **IMPLEMENTATION DEFINED**, **IMPLEMENTATION SPECIFIC**, **UNKNOWN**, and **UNPREDICTABLE**.

We recommend the following. If you do not follow these recommendations your system might not work.

Your system requires the following. If you do not follow these requirements your system will not work.

You are at risk of causing permanent damage to your system or your equipment, or of harming yourself.

This information is important and needs your attention.

This information might help you perform a task in an easier, better, or faster way.

This information reminds you of something important relating to the current content.

Useful resources

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

Arm documents are available on developer.arm.com/documentation.

Confidential documents are only available to licensees, when logged in. Each document link in the following tables provides direct access to the online version of the document.

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	IHI0069H	Non-Confidential