

Arm® Base System Architecture Compliance

Revision: r0p0

User Guide

arm

Arm® Base System Architecture Compliance

User Guide

Copyright © 2021 Arm Limited or its affiliates. All rights reserved.

Release Information

Document History

| Issue | Date | Confidentiality | Change |
|---------|-------------|------------------|---------------------------|
| 0000-01 | 12 May 2021 | Non-Confidential | Alpha release for RELv0.5 |

Non-Confidential 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, third party 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.

If any of the provisions contained in these terms conflict with any of the provisions of any click through or signed written agreement covering this document with Arm, then the click through or signed written agreement prevails over and supersedes the conflicting provisions of these terms. 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 subsidiaries) 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 © 2021 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)

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 an Alpha product, that is a product under development.

Web Address

developer.arm.com

Progressive terminology commitment

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

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

If you find offensive terms in this document, please contact terms@arm.com.

Contents

Arm® Base System Architecture Compliance User Guide

Preface

| | |
|------------------------------|---|
| <i>About this book</i> | 6 |
| <i>Feedback</i> | 8 |

Chapter 1

UEFI shell application

| | | |
|-----|--|------|
| 1.1 | <i>Abbreviations</i> | 1-10 |
| 1.2 | <i>Overview of tests</i> | 1-11 |
| 1.3 | <i>UEFI application arguments</i> | 1-12 |
| 1.4 | <i>Test IDs</i> | 1-13 |
| 1.5 | <i>UEFI shell implementation of PAL APIs</i> | 1-14 |

Chapter 2

Linux application

| | | |
|-----|--|------|
| 2.1 | <i>Linux application arguments</i> | 2-18 |
| 2.2 | <i>Build steps and environment setup</i> | 2-19 |

Appendix A

Revisions

| | | |
|-----|------------------------|-----------|
| A.1 | <i>Revisions</i> | Appx-A-21 |
|-----|------------------------|-----------|

Preface

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

It contains the following:

- [About this book](#) on page 6.
- [Feedback](#) on page 8.

About this book

This book is the user guide for Arm® BSA Architecture Compliance Suite.

Using this book

This book is organized into the following chapters:

Chapter 1 *UEFI shell application*

This chapter provides an overview on executing tests from the UEFI shell application.

Chapter 2 *Linux application*

This chapter provides an overview 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.

Other information

- *Arm® Developer.*
- *Arm® Documentation.*

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

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-systemready-accs@arm.com. Give:

- The title *Arm Base System Architecture Compliance User Guide*.
- The number 102504_0000_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.

Chapter 1

UEFI shell application

This chapter provides an overview on executing tests from the UEFI shell application.

It contains the following sections:

- [1.1 Abbreviations](#) on page 1-10.
- [1.2 Overview of tests](#) on page 1-11.
- [1.3 UEFI application arguments](#) on page 1-12.
- [1.4 Test IDs](#) on page 1-13.
- [1.5 UEFI shell implementation of PAL APIs](#) on page 1-14.

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 |
| BSA | Base System Architecture |
| DT | Device Tree |
| GIC | Generic Interrupt Controller |
| PAL | Platform Abstraction Layer |
| PCIe | Peripheral Component Interconnect express |
| PE | Processing Element |
| SMC | Secure Monitor Call |
| UEFI | Unified Extensible Firmware Interface |
| VAL | Validation Abstraction Layer |

1.2 Overview of tests

This section provides an overview of UEFI shell and Linux applications.

The general divisions of BSA tests between *Unified Extensible Firmware Interface* (UEFI) shell application and Linux application are described in the following table.

Table 1-2 Test environment and modules

| Test environment | Modules |
|--------------------|--|
| UEFI shell | PE, GIC, Timers, Watchdog, Wakeup and Power, PCIe, Memory map, Exerciser, Peripheral, and SMMU |
| Linux command line | PCIe, Memory map, and Peripheral |

1.3 UEFI application arguments

This section provides information on the UEFI shell application with its arguments.

Run the UEFI shell application with the following set of arguments.

```
uefi shell> bsa.efi [-v <n>] [-skip <x,y,z>] [-f <file name>] [-os] [-hyp] [-ps]
```

Note

After all the BSA tests are run and the test results are printed on the UEFI console, the UEFI session becomes unusable.

The argument descriptions are available in the following table.

Table 1-3 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. |
| -skip | Overrides the suite to skip the execution of a particular test. It allows a maximum of nine values (comma-separated). For example, 302 skips test case with ID = 302. 200 skips all tests in module with ID = 200. For more information on module IDs, see the 1.4 Test IDs on page 1-13 . |
| -f | File name to which the output log is written. |
| -os -hyp -ps | By default, all the Operating System, Hypervisor, Platform Security view tests are run. To run specific tests, add the following options: <ul style="list-style-type: none"> -os: Run Operating System view tests. -hyp: Run Hypervisor view tests. -ps: Run Platform Security view tests. |

Example

```
shell > bsa.efi -v 2 -skip 200,302 -f acs.txt -os
```

The set of parameters shown in the code block:

- Prints messages with verbosity of 2 and above.
- Tests for compliance against BSAv1.0 Operating System view tests.
- Skips execution of all tests belonging to GIC module and test number 302
- Stores the log messages to the file acs.txt.

1.4 Test IDs

This section provides information on module names and test IDs.

The test ID of each test is generated as an addition to module ID and unit test ID. For a given module, unit test ID begins from 1. Module name and module IDs are described in the following table.

Table 1-4 Module name and Module ID

| Module name | Module ID |
|------------------|-----------|
| PE | 0 |
| Memory Map | 100 |
| GIC | 200 |
| SMMU | 300 |
| Timer | 400 |
| Wakeup and Power | 500 |
| Peripheral | 600 |
| Watchdog | 700 |
| PCIe | 800 |
| Exerciser | 900 |

Note

Each module has tests classified as Operating System, Hypervisor, and Platform Security as defined by the BSA v1.0 specification.

1.5 UEFI shell implementation of PAL APIs

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

Booting to a UEFI shell is a prerequisite for running a BSA test.

Infrastructure APIs

The following table describes the PAL APIs and UEFI interfaces.

Table 1-5 PAL APIs and UEFI interfaces

| PAL API | UEFI interface |
|---------------------|-------------------|
| 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 PAL API to ACPI table mapping, if the system firmware presents platform configuration through ACPI tables.

Table 1-6 PAL APIs, UEFI interfaces, and ACPI tables consumed

| PAL APIs | UEFI interfaces | ACPI tables consumed |
|------------------------------|--|----------------------|
| pe_create_info_table | <ul style="list-style-type: none"> gST->ConfigurationTable CompareGuid IndustryStandard/Acpi61.h | MADT Table |
| call_smc | None | - |
| pe_execute_payload | None | - |
| pe_install_esr | <ul style="list-style-type: none"> gEfiCpuArchProtocolGuid Cpu->RegisterInterruptHandler | - |
| gic_create_info_table | <ul style="list-style-type: none"> gST->ConfigurationTable CompareGuid IndustryStandard/Acpi61.h | MADT table |
| gic_install_isr | <ul style="list-style-type: none"> gHardwareInterruptProtocolGuid RegisterInterruptSource EnableInterruptSource | - |
| timer_create_info_table | <ul style="list-style-type: none"> gST->ConfigurationTable CompareGuid IndustryStandard/Acpi61.h | GTDT table |
| wd_create_info_table | <ul style="list-style-type: none"> gST->ConfigurationTable CompareGuid IndustryStandard/Acpi61.h | GTDT table |
| pcie_create_info_table | <ul style="list-style-type: none"> gST->ConfigurationTable CompareGuid IndustryStandard/Acpi61.h | MCFG table |
| pcie_get_mcfg_ecam | <ul style="list-style-type: none"> gST->ConfigurationTable CompareGuid, IndustryStandard/Acpi61.h IndustryStandard/MemoryMappedConfigurationSpaceAccessTable.h | MCFG table |
| iovirt_create_info_table | <ul style="list-style-type: none"> gST->ConfigurationTable CompareGuid IndustryStandard/Acpi61.h | IORT table |
| peripheral_create_info_table | <ul style="list-style-type: none"> gEfiPciIoProtocolGuid Pci->GetLocation Pci->Pci.Read | - |
| memory_create_info_table | gBS->GetMemoryMap | - |

The following table represents the PAL API to DT node mapping, if the system firmware presents platform configuration through DT nodes.

Table 1-7 PAL APIs, UEFI interfaces, and DT nodes consumed

| PAL APIs | UEFI interfaces | DT nodes consumed |
|------------------------------|---|---|
| pe_create_info_table | <ul style="list-style-type: none"> gST->ConfigurationTable CompareGuid | cpu, pmu, interrupt-controller node |
| gic_create_info_table | <ul style="list-style-type: none"> gST->ConfigurationTable CompareGuid | interrupt-controller, v2m and its nodes |
| timer_create_info_table | <ul style="list-style-type: none"> gST->ConfigurationTable CompareGuid | systemtimer and memory mapped timer nodes |
| wd_create_info_table | <ul style="list-style-type: none"> gST->ConfigurationTable CompareGuid | watchdog nodes |
| pcie_create_info_table | <ul style="list-style-type: none"> gST->ConfigurationTable CompareGuid | pcie node |
| iovirt_create_info_table | <ul style="list-style-type: none"> gST->ConfigurationTable CompareGuid | smmu node |
| peripheral_create_info_table | <ul style="list-style-type: none"> gST->ConfigurationTable CompareGuid | usb, sata, and uart node |
| memory_create_info_table | gBS->GetMemoryMap | - |

Chapter 2

Linux application

This chapter provides an overview on executing tests from the Linux application.

It contains the following sections:

- [2.1 Linux application arguments](#) on page 2-18.
- [2.2 Build steps and environment setup](#) on page 2-19.

2.1 Linux application arguments

This section provides information on the Linux application with its set of arguments.

Run the Linux application with the following set of arguments.

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

Table 2-1 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 |
| skip | Overrides the suite to skip the execution of a particular test. For example, 53 skips test case with ID 53. |

Example

```
shell> bsa --v 3 --skip 57
```

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

Loading the kernel module

Before the BSA ACS Linux application can be run, load the BSA ACS kernel module using the `insmod` command.

```
shell> insmod bsa_acs.ko
```

2.2 Build steps and environment setup

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

The patch for the kernel tree and the Linux PAL are hosted separately on <https://gitlab.arm.com/linux-arm/linux-acs>.

Building the kernel module

Prerequisites

- Linux kernel source version 5.10.
- Linaro GCC tool chain 5.3 or above.
- Build environment for AArch64 Linux kernel.

Porting steps for Linux kernel

1. `git clone https://git.gitlab.arm.com/linux-arm/linux-acs.git <local_dir/linux-acs>`
2. `git clone https://github.com/ARM-software/bsa-accs.git <local_dir/bsa-accs>`
3. Apply the `<local_dir/linux-accs/kernel/src/0001-BSA-SBSA-ACS-Linux-5.10.patch>` patch to your kernel source tree.
4. Build the kernel.

Build steps for BSA kernel module

1. `cd <local_dir/linux-accs/bsa-accs-drv/files>`
2. Set `CROSS_COMPILE` to the ARM64 toolchain path.
3. `export KERNEL_SRC=<linux kernel path>`
4. `./setup.sh <local_dir/bsa-accs>`
5. `./linux_bsa_acs.sh`

`bsa_acs.ko` file is generated.

BSA Linux application build

1. `cd <bsa-accs path>/linux_app/bsa-accs-app`
2. Set `CROSS_COMPILE` to the ARM64 toolchain path.
`export CROSS_COMPILE=<local_dir>/gcc-linaro-5.3-2016.02/bin/aarch64-linux-gnu-`
3. `make`

The executable file `bsa` is generated.

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-21.](#)

A.1 Revisions

This section consists of all the technical changes between different versions of this document.

Table A-1 Issue 0000-01

| Change | Location |
|---------------|----------|
| First release | - |