Arm® Virtual Base System Architecture 1.0

Platform Design Document

Non-confidential

Copyright © 2025 Arm Limited or its affiliates. All rights reserved.
Document number: DEN0150

Copyright © 2025 Arm Limited or its affiliates. All rights reserved.
Non-confidential

Contents

Copyright © 2025 Arm Limited or its affiliates. All rights reserved.
Non-confidential

Copyright © 2025 Arm Limited. All rights reserved.

Release information

Version 1 (22 Jan 2025)

- Initial public release.

Copyright © 2025 Arm Limited or its affiliates. All rights reserved.
Non-confidential

Arm Non-Confidential Document License (“License”)

This License is a legal agreement between you and Arm Limited (“**Arm**”) for the use of Arm’s intellectual property (including, without limitation, any copyright) embodied in the document accompanying this License (“**Document**”). Arm licenses its intellectual property in the Document to you on condition that you agree to the terms of this License. By using or copying the Document you indicate that you agree to be bound by the terms of this License.

“**Subsidiary**” means any company the majority of whose voting shares is now or hereafter owner or controlled, directly or indirectly, by you. A company shall be a Subsidiary only for the period during which such control exists.

This Document is **NON-CONFIDENTIAL** and any use by you and your Subsidiaries (“Licensee”) is subject to the terms of this License between you and Arm.

Subject to the terms and conditions of this License, Arm hereby grants to Licensee under the intellectual property in the Document owned or controlled by Arm, a non-exclusive, non-transferable, non-sub-licensable, royalty-free, worldwide License to:

(i) use and copy the Document for the purpose of designing and having designed products that comply with the Document;

(ii) manufacture and have manufactured products which have been created under the License granted in (i) above; and

(iii) sell, supply and distribute products which have been created under the License granted in (i) above.

Licensee hereby agrees that the Licenses granted above shall not extend to any portion or function of a product that is not itself compliant with part of the Document.

Except as expressly licensed above, Licensee acquires no right, title or interest in any Arm technology or any intellectual property embodied therein.

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.

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.

THE 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. Arm may make changes to the Document at any time and without notice. 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.

NOTWITHSTANDING ANYTHING TO THE CONTRARY CONTAINED IN THIS LICENSE, TO THE FULLEST EXTENT PERMITTED BY LAW, IN NO EVENT WILL ARM BE LIABLE FOR ANY DAMAGES, IN CONTRACT, TORT OR OTHERWISE, IN CONNECTION WITH THE SUBJECT MATTER OF THIS LICENSE (INCLUDING WITHOUT LIMITATION) (I) LICENSEE’S USE OF THE DOCUMENT; AND (II) THE IMPLEMENTATION OF

Copyright © 2025 Arm Limited or its affiliates. All rights reserved.
Non-confidential

THE DOCUMENT IN ANY PRODUCT CREATED BY LICENSEE UNDER THIS LICENSE). THE EXISTENCE OF MORE THAN ONE CLAIM OR SUIT WILL NOT ENLARGE OR EXTEND THE LIMIT. LICENSEE RELEASES ARM FROM ALL OBLIGATIONS, LIABILITY, CLAIMS OR DEMANDS IN EXCESS OF THIS LIMITATION.

This License shall remain in force until terminated by Licensee or by Arm. Without prejudice to any of its other rights, if Licensee is in breach of any of the terms and conditions of this License then Arm may terminate this License immediately upon giving written notice to Licensee. Licensee may terminate this License at any time. Upon termination of this License by Licensee or by Arm, Licensee shall stop using the Document and destroy all copies of the Document in its possession. Upon termination of this License, all terms shall survive except for the License grants.

Any breach of this License by a Subsidiary shall entitle Arm to terminate this License as if you were the party in breach. Any termination of this License shall be effective in respect of all Subsidiaries. Any rights granted to any Subsidiary hereunder shall automatically terminate upon such Subsidiary ceasing to be a Subsidiary.

The Document consists solely of commercial items. Licensee shall be responsible for ensuring that any use, duplication or disclosure of the Document complies fully with any relevant export laws and regulations to assure that the Document or any portion thereof is not exported, directly or indirectly, in violation of such export laws.

This License may be translated into other languages for convenience, and Licensee agrees that if there is any conflict between the English version of this License and any translation, the terms of the English version of this License 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. No license, express, implied or otherwise, is granted to Licensee under this License, to use the Arm trade marks in connection with the Document or any products based thereon. Visit Arm's website at http://www.arm.com/company/policies/trademarks for more information about Arm's trademarks.

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

Copyright © 2025 Arm Limited (or its affiliates). All rights reserved.

Arm Limited. Company 02557590 registered in England.

110 Fulbourn Road, Cambridge, England CB1 9NJ.

Arm document reference: PRE-21585

version 5.0, March 2024

Copyright © 2025 Arm Limited or its affiliates. All rights reserved.
Non-confidential

About this document

Terms and abbreviations

Term	Meaning
Arm ARM	Arm Architecture Reference Manual. See [1].
BBR	Base Boot Requirements. See [2].
BSA	Base System Architecture. See [3].
DMA	Direct Memory Access.
GIC	Generic Interrupt Controller.
PE	Processing Element, as defined in the Arm ARM, and as viewed from within the virtual environment.
PMU	Performance Monitor Unit.
PPI	Private Peripheral Interrupt.
SBSA	Server Base System Architecture. See [4].
SGI	Software-Generated Interrupt.
System firmware data	System description data structures, for example ACPI or Flattened Device Tree.
VMID	Virtual Machine Identifier.

References

This section lists publications by Arm and by third parties.

See Arm Developer (http://developer.arm.com) for access to Arm documentation.

[1] *DDI 0487 Arm® Architecture Reference Manual for A-profile architecture*. Arm Ltd.

[2] *DEN 0044 Arm® Base Boot Requirements*. Arm Ltd.

[3] *DEN 0094 Arm® Base System Architecture*. Arm Ltd.

[4] *DEN 0029 Arm® Server Base System Architecture*. Arm Ltd.

Rules-based writing

This specification consists of a set of individual *content items*. A content item is classified as one of the following:

- Declaration
- Rule
- Goal
- Information
- Rationale
- Implementation note
- Software usage

Copyright © 2025 Arm Limited or its affiliates. All rights reserved.
Non-confidential

Declarations and Rules are normative statements. An implementation that is compliant with this specification must conform to all Declarations and Rules in this specification that apply to that implementation.

Declarations and Rules must not be read in isolation. Where a particular feature is specified by multiple Declarations and Rules, these are generally grouped into sections and subsections that provide context. Where appropriate, these sections begin with a short introduction.

Arm strongly recommends that implementers read *all* chapters and sections of this document to ensure that an implementation is compliant.

Content items other than Declarations and Rules are informative statements. These are provided as an aid to understanding this specification.

Content item identifiers

A content item may have an associated identifier which is unique among content items in this specification.

After this specification reaches beta status, a given content item has the same identifier across subsequent versions of the specification.

Content item rendering

In this document, a content item is rendered with a token of the following format in the left margin: L_{iiiii}

- *L* is a label that indicates the content class of the content item.
- *iiiii* is the identifier of the content item.

Content item classes

Declaration

A Declaration is a statement that does one or more of the following:

- Introduces a concept
- Introduces a term
- Describes the structure of data
- Describes the encoding of data

A Declaration does not describe behavior.

A Declaration is rendered with the label *D*.

Rule

A Rule is a statement that describes the behavior of a compliant implementation.

A Rule explains what happens in a particular situation.

A Rule does not define concepts or terminology.

A Rule is rendered with the label *R*.

Goal

A Goal is a statement about the purpose of a set of rules.

A Goal explains why a particular feature has been included in the specification.

A Goal is comparable to a “business requirement” or an “emergent property.”

A Goal is intended to be upheld by the logical conjunction of a set of rules.

A Goal is rendered with the label *G*.

Copyright © 2025 Arm Limited or its affiliates. All rights reserved.
Non-confidential

Information

An Information statement provides information and guidance as an aid to understanding the specification.

An Information statement is rendered with the label *I*.

Rationale

A Rationale statement explains why the specification was specified in the way it was.

A Rationale statement is rendered with the label *X*.

Implementation note

An Implementation note provides guidance on implementation of the specification.

An Implementation note is rendered with the label *U*.

Software usage

A Software usage statement provides guidance on how software can make use of the features defined by the specification.

A Software usage statement is rendered with the label *S*.

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.

We believe that this document contains no offensive terms. If you find offensive terms in this document, please contact terms@arm.com.

Feedback

Arm welcomes feedback on its documentation.

If you have any comments or suggestions for additions and improvements create a ticket at

https://support.developer.arm.com.

As part of the ticket include:

- The title (Virtual Base System Architecture).
- The document ID and version (DEN0150 1.0).
- The page numbers to which your comments apply.
- A concise explanation of your comments.

Arm also welcomes general suggestions for additions and improvements.

Copyright © 2025 Arm Limited or its affiliates. All rights reserved.
Non-confidential

1 Background

Virtual Base System Architecture (V-BSA) specifies the requirements and run-time features that a base Virtual Environment (VE) needs to install, boot, and run an operating system.

This document is a supplement to the Arm Base System Architecture document [3].

A virtual environment is a framework that exposes to software a logical view of a system that is implemented using a combination of hardware and software.

The requirements specified in the Arm BSA [3] are extended, modified, or removed in V-BSA to meet the specific needs of virtual environments.

Requirements for virtual environment can vary depending on usage and target market. The Virtual Base System Architecture uses levels of functionality. Each level adds functionality better than the previous level. Unless explicitly stated, all specification items that belong to level N apply to levels greater than N.

An implementation is consistent with a level of the Virtual Base System Architecture if it implements all of the functionality of that level at performance levels that are appropriate for the target uses of that level. This means that all functionality of a level is expected to perform well when exploited by software.

This specification refers to system firmware data which the Arm BBR [2] describes.

Copyright © 2025 Arm Limited or its affiliates. All rights reserved.
Non-confidential

2 V-BSA levels and rules

Level 1 of V-BSA is the minimum set of requirements for a virtual environment. Section 2.2 describes requirements for future levels.

This Specification describes rules for virtual environments. Some rules are new and others are revised rules or a collection of rules from the Arm BSA [3] and Arm SBSA [4]. Section 3 lists the requirements and related rules from other specifications which help map the BSA and V-BSA rules.

2.1 Level 1

Section 3.1 lists the rules to be implemented for Level 1.

2.1.1 PE architecture

$R_{V\_L1PE\_01}$ The virtual environment must implement Arm BSA [3] rules B_PE_01, B_PE_02, B_PE_03, B_PE_04, B_PE_05, B_PE_06, B_PE_07, B_PE_08, B_PE_10, B_PE_13, and B_PE_14.

$R_{V\_L1PE\_02}$ PEs must implement PMU extension version 3 (FEAT_PMUv3) and implement the PMU cycle counter (PMCCNTR_EL0).

I Each PE is recommended to implement a minimum of six breakpoints, two of which must be able to match virtual address, contextID, or VMID.

I Each PE is recommended to implement a minimum of four synchronous watchpoints.

X Breakpoints and watchpoints are important for software debugging from within the virtual environment.

2.1.2 Memory map

$R_{V\_L1MM\_01}$ The virtual environment must implement Arm BSA [3] rules B_MEM_01, B_MEM_02, B_MEM_04, B_MEM_05, and B_MEM_07.

$R_{V\_L1MM\_02}$ All DMA requesters in a base system that are expected to be under the control of the operating system must be able to address all of the Non-secure address space.

2.1.3 Interrupt controller

$R_{V\_L1GI\_01}$ The virtual environment must implement the Arm BSA [3] rules B_GIC_01, B_GIC_02, B_GIC_03, and B_GIC_05.

I The requirement for eight SGIs in B_GIC_05 is a limitation in physical systems that is adopted as a minimum requirement for virtual environments. Implementations can expose a higher number of virtual SGIs.

2.1.4 PPI assignments

$R_{V\_L1PP\_00}$ The virtual environment must map the interrupts to PPI INTIDs as listed in B_PPI_01 from the Arm BSA [3].

2.1.5 System MMU and device assignment

I Implementation of an SMMU is optional for virtual environments.

$R_{V\_L1SM\_01}$ If an SMMU is implemented in the virtual environment, it must adhere to Arm BSA [3] rules B_SMMU_01, B_SMMU_02, B_SMMU_06, B_SMMU_07, and B_SMMU_12.

$R_{V\_L1SM\_02}$ Stage 1 System MMU functionality that is made visible to an operating system must present the interface of a System MMU compatible with one of the following:

Copyright © 2025 Arm Limited or its affiliates. All rights reserved.
Non-confidential

- The SMMUv2 specification, where each context bank must present a unique physical interrupt to the GIC.
- The Arm SMMUv3 specification or higher, where the integration of the System MMUs is compliant with rules SMMU_01 and SMMU_02 as specified in “SMMUv3 integration”, Section D of the BSA [3].

$R_{V\_L1SM\_03}$ The virtual environment must guarantee that any boot device that is required for installing and booting the operating system is not blocked by any form of System MMU functionality.

2.1.6 Clock and timer subsystem

$R_{V\_L1TM\_01}$ The virtual system must include the virtual counter and the physical counter, as specified in the Arm ARM [1].

$R_{V\_L1TM\_02}$ The virtual counter and physical counter must run at a minimum effective frequency of 10MHz and both counters must run at the same effective frequency.

$R_{V\_L1TM\_03}$ The virtual environment must implement Arm BSA [3] rules B_TIME_03, and B_TIME_04.

I The passage of virtual time is measured using CNTVCT_EL0. The passage of physical time is measured using CNTPCT_EL0.

$R_{V\_L1TM\_04}$ The passage of time as measured by reading a counter, provides a uniform view of virtual or physical time. This means that both of the following are true:

- It must not be possible for a sequence of reads of CNTVCT_EL0 by any PE to show virtual time going backwards.
- It must not be possible for a sequence of reads of CNTPCT_EL0 by any PE to show physical time going backwards.

I An example of a sequence of reads that shows time going backwards is as follows:

1. PE 0 reads a counter and obtains a value of 0x100.
2. PE 0 writes a flag to memory which is read by PE 1.
3. PE 1 reads the same counter and obtains a value that is less than 0x100.

I The difference between virtual and physical time is IMPLEMENTATION DEFINED.

Note

PE timers and system counters are always on, except in semantic G in Table 3 when they must be off.

2.1.7 Wakeup semantics

X Systems may implement different PE and system low power states, which are described in system firmware tables. The virtual environment must understand the relationship between these power states and the facilities a system has for waking PEs from these low power states. This enables the virtual environment to choose an optimal PE power state to enter when there is no more work to be scheduled on the PE. It also enables the correct wakeup source to be used according to the power state selected.

$R_{V\_L1WK\_01}$ A PE must wake in response to a wakeup interrupt, independent of the state of its PSTATE interrupt mask bits, which are the A, I, and F bits, and of the wakeup interrupt priority.

Note

There are some power states where a PE does not wake on an interrupt. It is the responsibility of system software to ensure there are no wakeup interrupts targeting a PE entering these states. See Table 2.

Copyright © 2025 Arm Limited or its affiliates. All rights reserved.
Non-confidential

R_V_L1WK_02 Whenever a PE is woken from a sleep or off state by a wakeup interrupt, the OS or hypervisor must be presented with an interrupt so that the PE software can determine which device requested the wakeup.

R_V_L1WK_03 The interrupt must be pending in the GIC at the point that control is handed back to the OS or hypervisor after a PE wake.

R_V_L1WK_04 The virtual environment must implement Arm BSA [3] rules B_WAK_05, B_WAK_06.

2.1.8 Power state semantics

R_V_L1WK_05 For either the OS or hypervisor, or both, to be able to reason about wakeup events and to know which timers will be available to wake the PE, all PEs must be in a state that is consistent with one of the semantics described in Table 2 and Table 3.

Note

All PEs do not need to be in the same state. Arm expects that the semantics of the power states that a system supports will be described by system firmware data. Table 4 describes the power state semantics in a set of component-specific rules.

R_V_L1WK_06 The virtual environment must implement Arm BSA [3] rule B_WAK_08.

I

Table 2: PE Power states

PE State	Description
Run	The PE is powered up and running code.
Idle_standby	The PE is in STANDBYWFI state, but remains powered up. There is full state retention, and no state saving or restoration are required. Execution automatically resumes after any interrupt or external debug request (EDBGRQ). Debug registers are accessible.
Idle_retention	The PE is in STANDBYWFI state, but remains powered up. There is full state retention, and no state saving, or restoration are required. Execution automatically resumes after any interrupt or external debug request (EDBGRQ). Debug registers are not accessible.
Sleep	The PE is powered down but hardware wakes the PE autonomously, for example, on receiving a wakeup interrupt. No PE state is retained. State must be explicitly saved and restored.
Off	The PE is powered down and is not required to be woken by interrupts. The only way to wake the PE is from system software running on another PE, or an external source such as a poweron_reset. This state can be used to support hot remove of PE. No PE state is retained.

R_V_L1WK_07 When the system is in a state where the GIC is powered down, devices must not send messaged interrupts. See Table 3.

R_V_L1WK_08

Copyright © 2025 Arm Limited or its affiliates. All rights reserved.
Non-confidential

Table 3: Power state semantics

Semantic	PE and GIC PE Interface	GIC Distributor	Note
A	Run	On	-
B	Idle	On	PE resumes execution on receipt of any interrupt.
C	Sleep	On	PE wakes on receipt of a wakeup interrupt.
E	Sleep	Off	PE wakes from system timer wakeup event or other system specific events.
F	Off	On	Some, but not all, PEs are in Off state.
G	Off	Off	All PEs in Off state.

Note

- PE Timers, wake timers, and system counters are always on, except in semantic G in Table 3 when they must be off.
- Semantic D has been intentionally skipped to prevent overlap with semantic D in the 'Power State Semantics' table in Arm BSA[3].

$R_{V\_L1WK\_09}$

Table 4: Component Power state semantics

PE and GIC PE Interface	Individual PEs and their associated GIC PE interface can be in Run, Idle, Sleep, or Off state.
GIC Distributor	Must be On if any PE is in the Run or Idle state. Might be On or Off if all PEs are in either the Sleep or Off state, with at least one PE in the Sleep state. Must be Off If all PEs are in the Off state.
System wakeup timers and system counter	Must be On if any PE is not in the Off state. Must be Off if all PEs are in the Off state.

2.1.9 Peripheral subsystems

$R_{V\_L1PR\_01}$ The virtual environment must implement Arm BSA [3] rules B_PER_01, B_PER_02, B_PER_03, B_PER_04, B_PER_05, B_PER_06, B_PER_09, B_PER_10, B_PER_11, and B_PER_12.

$R_{V\_L1PR\_02}$ If the virtual environment implements PCIe, it must comply with all rules in BSA [3] Section E, except for rule PCI_MM_02.

I Support for mapping PCI Express memory space as normal non-cacheable memory is optional for virtual environments.

2.2 Future requirements

Section 3.2 lists the rules to be implemented in future levels.

Copyright © 2025 Arm Limited or its affiliates. All rights reserved.
Non-confidential

2.2.1 PE architecture

$R_{V\_L2PE\_01}$ PEs must implement a minimum of two PMU event counters in addition to the PMU cycle counter.

$R_{V\_L2PE\_02}$ The PMU overflow signal from each PE must be wired to a unique Interrupt ID with no intervening logic.

2.2.2 Watchdogs

$R_{V\_L2WD\_01}$ The virtual environment must provide an IMPLEMENTATION DEFINED watchdog.

X A watchdog timer with heartbeat is an important functionality for virtual environments. This interface might be standardized in the future.

Copyright © 2025 Arm Limited or its affiliates. All rights reserved.
Non-confidential

3 V-BSA checklist

This section lists the minimum virtual environment requirements required to boot and run a guest operating system on a virtualized environment.

Note

The list of “Related rules from other specifications” below is indicative, and provided only for reference.

3.1 V-BSA level 1 checklist

Module	Required Rule ID	Related rules from other specifications (for reference only)
PE	V_L1PE_01	
PE	V_L1PE_02	B_PE_09 (BSA)
Memory Map	V_L1MM_01	
Memory Map	V_L1MM_02	B_MEM_03 (BSA), B_MEM_06 (BSA)
Interrupts	V_L1GI_01	
Interrupts	V_L1PP_00	
SMMU	V_L1SM_01	
SMMU	V_L1SM_02	B_SMMU_08 (BSA)
SMMU	V_L1SM_03	
Timer	V_L1TM_01	B_TIME_01 (BSA)
Timer	V_L1TM_02	B_TIME_02 (BSA)
Timer	V_L1TM_03	
Timer	V_L1TM_04	
Wakeup	V_L1WK_01	B_WAK_01 (BSA)
Wakeup	V_L1WK_02	B_WAK_03 (BSA)
Wakeup	V_L1WK_03	B_WAK_04 (BSA)
Wakeup	V_L1WK_04	
Power State	V_L1WK_05	B_WAK_07 (BSA)
Power State	V_L1WK_06	
Power State	V_L1WK_07	B_WAK_09 (BSA)
Power State	V_L1WK_08	B_WAK_10 (BSA)
Power State	V_L1WK_09	B_WAK_11 (BSA)
Peripheral	V_L1PR_01	
Peripheral	V_L1PR_02	B_PER_08 (BSA)

Copyright © 2025 Arm Limited or its affiliates. All rights reserved.
Non-confidential

3.2 Future Level checklist

In addition to the V-BSA Level 1 rules in Section 3.1, the following additional rules are required.

Module	Required Rule ID	Related rules from other specifications (for reference only)
PE	V_L2PE_01	B_PE_09 (BSA)
PE	V_L2PE_02	B_PE_10 (BSA)
Watchdog	V_L2WD_01	S_L3WD_01 (SBSA)

Copyright © 2025 Arm Limited or its affiliates. All rights reserved.
Non-confidential