Arm® Paravirtualized Time for Arm-based Systems

Platform Design Document

Non-Confidential

Version 1.0

Document number: DEN0057A.b

Contents

Release information

The Change History table lists the changes that are made to this document.

Table R.1. Change history

Date	Issue	Confidentiality	Change
Sep 2019	Issue A	Non-confidential	Version 1.0, first external release
June 2022	Issue A.b	Non-confidentiial	Version 1.0 Issue A.b. Clairfy the definition of Stolen time in Section 3.1. Add progressive terminology commitment.

Arm Non-Confidential Document Licence (“Licence”)

This Licence 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 Licence (“**Document**”). Arm licenses its intellectual property in the Document to you on condition that you agree to the terms of this Licence. By using or copying the Document you indicate that you agree to be bound by the terms of this Licence.

“**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 Licence between you and Arm.

Subject to the terms and conditions of this Licence, 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 licence 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 licence granted in (i) above; and

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

Licensee hereby agrees that the licences 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 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.

NOTWITHSTANING ANYTHING TO THE CONTRARY CONTAINED IN THIS LICENCE, TO THE FULLEST EXTENT PETMITTED 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 LICENCE (INCLUDING WITHOUT LIMITATION) (I) LICENSEE’S USE OF THE DOCUMENT; AND (II) THE IMPLEMENTATION OF THE DOCUMENT IN ANY PRODUCT CREATED BY LICENSEE UNDER THIS LICENCE). 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 Licence 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 Licence then Arm may terminate this Licence immediately upon giving written notice to Licensee. Licensee may terminate this Licence at any time. Upon termination of this Licence 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 Licence, all terms shall survive except for the licence grants.

Any breach of this Licence by a Subsidiary shall entitle Arm to terminate this Licence as if you were the party in breach. Any termination of this Licence 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 Licence may be translated into other languages for convenience, and Licensee agrees that if there is any conflict between the English version of this Licence and any translation, the terms of the English version of this Licence 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

Arm Limited. Company 02557590 registered in England.
110 Fulbourn Road, Cambridge, England CB1 9NJ.

Arm document reference: LES-PRE-21585 version 4.0

1 About this Document

This document describes a standard interface for paravirtualized time in Arm based Systems. The interface includes support for tracking stolen time.

1.1 References

This document refers to the following documents.

Reference	Document Number	Title
[Armv8]	DDI 0487	Arm Architecture Reference Manual Armv8, for Armv8-A architecture profile.
[SMCCC]	DEN0028	Arm SMC Calling Convention.
[SMCCCv1.1]	DEN0070A	Firmware interfaces for mitigating cache speculation vulnerabilities.

1.2 Terms and abbreviations

This document uses the following terms and abbreviations.

Term	Meaning
IPA	Intermediate Physical Address
PE	Processing Element
EL1	The Non-secure exception level that is used to execute operating systems.

1.3 Feedback

Arm welcomes feedback on its documentation.

1.3.1 Feedback on this manual

If you have comments on the content of this manual, send an e-mail to errata@arm.com. Give:

- The title.
- The document and version number, DEN0057A.b.
- The page numbers to which your comments apply.
- A concise explanation of your comments.

Arm also welcomes general suggestions for additions and improvements.

1.4 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.

2 Introduction

Operating systems require time stamping and timer capabilities to perform basic operations like scheduling and measuring the passage of time. The Arm architecture provides the generic timer for these purposes. See [Armv8] for more details.

Guest operating systems that run in virtual machines need time stamping and timer capabilities. Paravirtualized time and timers can provide these capabilities. This specification provides a standard mechanism for measuring stolen time on virtualized systems that are based on the Arm architecture. This specification only covers systems in which the Execution state of the hypervisor as well as EL1 of virtual machines is AArch64.

3 Paravirtualized time constructs

3.1 Terminology

This document uses the following terminology to express the different states of a virtual machine.

- **Paused:** All activity in the virtual machine has stopped to a level from which it is possible to save the context of the virtual machine. The saved context can be restored at a later time, so that the virtual machine can resume execution to a running state.
- **Running:** The virtual machine is assigned to a physical machine and can respond to user commands. When a virtual machine is running, virtual Processing Elements (PEs) can be in one of the two following states.
    - **Scheduled in:** A virtual PE is scheduled in, or running, when it is executing guest code because it is currently assigned to physical PE.
    - **Scheduled out:** A virtual PE is scheduled out if it is not currently assigned to any physical PE, and therefore is not executing code.

To describe the various views of the passage of time that can be observed by a virtual machine, or a hypervisor, this document uses the following terminology.

- **Physical Time:** Time that always progresses, regardless of whether the virtual machine is running or paused. For any virtual machine, physical time is the amount of time that machine has been in existence.
- **Live Physical Time:** Time that progresses whenever a virtual machine is running on a physical machine, regardless of whether or not it has any virtual PE scheduled in. This time does not progress while the virtual machine is paused.
- **Virtual Time:** Time that progresses only when a virtual PE in the virtual machine is scheduled in. Virtual time can be tracked individually per virtual PE, or for a whole machine. In the latter case, virtual time tracks the time that any virtual PE in the machine is scheduled in. Tracking virtual time is not covererd in this document.
- **Stolen Time:** Time during which a virtual PE is scheduled out. Stolen time does not account for any time during which:
    - the virtual machine voluntarily chooses not to execute instructions on the virtual PE, or
    - the virtual machine is paused.

Many systems also track wall-clock time (absolute date and time of day), but this is not covered in this document.

Figure 1 shows the different states of a virtual machine and the corresponding concepts of time.

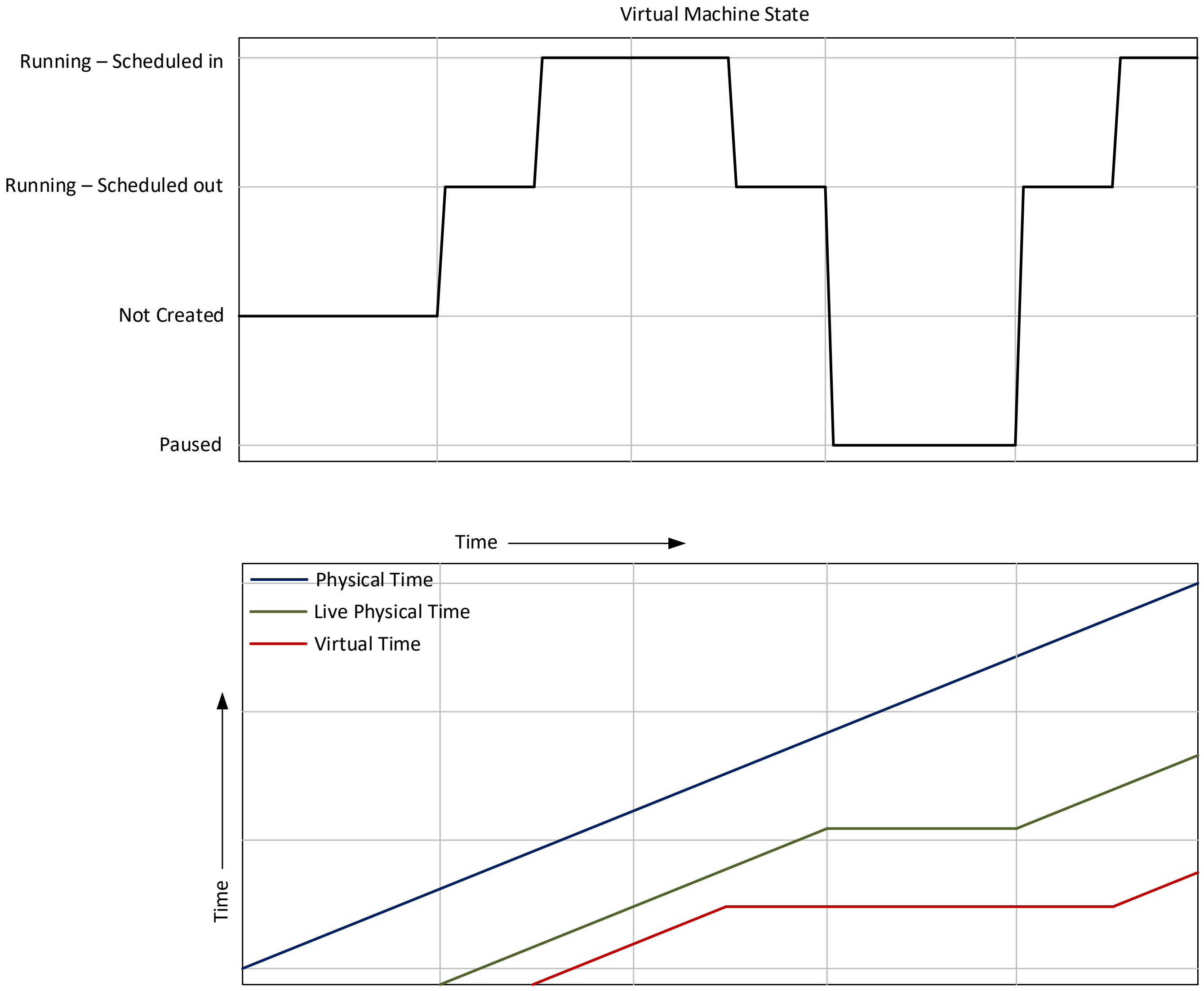

Figure 1: Virtual machine states and progress of time

3.2 Stolen time

3.2.1 Background

Guest operating systems that are running in virtual machines need time stamping and timer capabilities. A guest operating system can use the virtual counter, CNTVCT_EL0 (see [Armv8] for more details), which provides a hypervisor-controlled offset from the generic timer. The virtual counter allows the hypervisor to hide time when the guest is not running. Examples when the guest is not running includes scenarios when the guest is paused or during migration of the guest.

The host can decide to schedule only some of the virtual PEs of the guest. This can happen if the host is oversubscribed, or for other reasons. This specification describes a standard mechanism that allows a guest to discover how much time has been forcibly ‘stolen’ from the execution of a virtual PE. Stolen time can be used by the guest to more accurately account for the execution time of the processes that the guest is running. Stolen time does not include any intervals during which the virtual machine is paused or is migrating from one physical machine to another.

3.2.2 Hypervisor and guest shared data

The guest and the hypervisor share a stolen time shared memory region for each virtual PE. The format for the stolen time shared memory region is shown in Table 1.

The hypervisor must update the `stolen_time` field in the stolen time shared memory region before scheduling the virtual PE. The value must be provided in nanoseconds. Writing or reading the `stolen_time` field in the stolen time shared memory region must only be done using 64-bit single-copy atomic memory accesses.

Table 1: Layout of the stolen time shared memory region

Field	Byte length	Byte offset	Description
`Revision`	4	0	For implementations compliant with this revision of the specification, this field must be 0.
`Attributes`	4	4	This field must be 0.
`stolen_time`	8	8	Total stolen time, in nanoseconds, measured over the lifetime of the virtual PE. This field must be accessed with 64-bit single-copy atomicity.

4 Calls

This section describes the calls that allow the hypervisor and the guest to discover and configure each other’s capability to support paravirtualized time and timers.

The calls follow the SMC64/HVC64 conventions in [SMCCC], which mandate that the immediate value of the Secure Monitor Call (SMC) or Hypervisor Call (HVC) instruction must be zero.

To support nested virtualization in the future, either SMC or HVC instructions can be used. Calls directed from a guest hypervisor to a host hypervisor should use SMC instructions. Calls directed from a guest operating system, to a guest or a host hypervisor should use HVC instructions. Therefore, a host hypervisor supporting nested virtualization must support both SMC and HVC conduits.

If EL1 Execution state of the guest operating system is AArch32 or if the Execution state of the hypervisor is AArch32, then all the calls should return NOT_SUPPORTED (See [SMCCC] for more detail on return codes).

4.1 Discovery

This specification requires [SMCCCv1.1] compliance.

A call to SMCCC_ARCH_FEATURES with PV_TIME_FEATURES returns the following:

- NOT_SUPPORTED (-1) if this specification is not implemented.
- SUCCESS (0) if the interface is supported.

The guest can then make use of PV_TIME_FEATURES to discover the other calls that are defined in this specification.

4.2 PV_TIME_FEATURES

This call determines if a specific paravirtualized time call is supported. When a virtual machine starts, it can make use of this call to discover whether stolen time is supported.

Parameters	
Name	**Description**
uint32 FunctionID	This field should be set to 0xC5000020, which is a Function Identifier in the SMC64/HVC64 Standard Hypervisor Service Call range.
uint32 PV_call_id	This field takes the value of the FunctionID that is associated with another call defined in this specification. For values of PV_call_id, please refer to the other calls that are defined in this specification.
Return values	
Name	**Description**

	This call returns: • NOT_SUPPORTED (-1) to indicate that the specified paravirtualized time function is not supported or is invalid. • SUCCESS (0) to indicate that the specified paravirtualized time function is supported.
int64 status	If PV_call_id identifies PV_TIME_FEATURES, this call returns: • NOT_SUPPORTED (-1) to indicate that all paravirtualized time functions in this specification are not supported. • SUCCESS (0) to indicate that all the paravirtualized time functions in this specification are supported. For more information on error codes see [SMCCC].

4.3 PV_TIME_ST

This call is used to retrieve the stolen time memory region for the calling virtual PE. If stolen time is supported, the guest can request access to a stolen time memory region for each virtual PE. When a guest calls this function, the hypervisor returns the Intermediate Physical Address (IPA) of the stolen time shared memory region of the calling virtual PE, as described in Table 1. The calling guest can map the IPA into normal memory with inner and outer write back caching attributes in the inner shareable domain.

Parameters	
Name	**Description**
uint32 FunctionID	This field should be set to 0xC5000021, which is a Function Identifier in the SMC64/HVC64 Standard Hypervisor Service Call range.
Return values	
Name	**Description**
int64 status	This call returns: • The 64-byte aligned IPA of the stolen time shared memory region for the calling virtual PE as described in Table 1, on success. • NOT_SUPPORTED (-1), on failure or if stolen time is not supported. For more information on error codes see [SMCCC].