

JEDEC STANDARD

Universal Flash Storage (UFS) Unified Memory Extension

Version 1.1

JESD220-1A

(Revision of JESD220-1, September 2013)

MARCH 2016

JEDEC SOLID STATE TECHNOLOGY ASSOCIATION



NOTICE

JEDEC standards and publications contain material that has been prepared, reviewed, and approved through the JEDEC Board of Directors level and subsequently reviewed and approved by the JEDEC legal counsel.

JEDEC standards and publications are designed to serve the public interest through eliminating misunderstandings between manufacturers and purchasers, facilitating interchangeability and improvement of products, and assisting the purchaser in selecting and obtaining with minimum delay the proper product for use by those other than JEDEC members, whether the standard is to be used either domestically or internationally.

JEDEC standards and publications are adopted without regard to whether or not their adoption may involve patents or articles, materials, or processes. By such action JEDEC does not assume any liability to any patent owner, nor does it assume any obligation whatever to parties adopting the JEDEC standards or publications.

The information included in JEDEC standards and publications represents a sound approach to product specification and application, principally from the solid state device manufacturer viewpoint. Within the JEDEC organization there are procedures whereby a JEDEC standard or publication may be further processed and ultimately become an ANSI standard.

No claims to be in conformance with this standard may be made unless all requirements stated in the standard are met.

Inquiries, comments, and suggestions relative to the content of this JEDEC standard or publication should be addressed to JEDEC at the address below, or refer to www.jedec.org under Standards and Documents for alternative contact information.

Published by
©JEDEC Solid State Technology Association 2016
3103 North 10th Street
Suite 240 South
Arlington, VA 22201-2107

This document may be downloaded free of charge; however JEDEC retains the copyright on this material. By downloading this file the individual agrees not to charge for or resell the resulting material.

PRICE: Contact JEDEC

Printed in the U.S.A.
All rights reserved

PLEASE!

DON'T VIOLATE
THE
LAW!

This document is copyrighted by JEDEC and may not be
reproduced without permission.

For information, contact:

JEDEC Solid State Technology Association
3103 North 10th Street
Suite 240 South
Arlington, VA 22201-2107

or refer to www.jedec.org under Standards-Documents/Copyright Information.

UNIVERSAL FLASH STORAGE (UFS) UNIFIED MEMORY EXTENSION, VERSION 1.1**Contents**

	Page
1 Scope	1
2 Normative Reference.....	1
3 Terms and Definitions	2
4 Introduction	6
4.1 Overview	6
4.2 Purpose of Unified Memory Support	7
4.3 Framework of Unified Memory	8
4.3.1 Host Memory Map	8
4.3.2 Configuration of Unified Memory	9
4.4 Speed Estimation.....	9
5 Modified UFS Architecture Overview	11
5.1 Modified UFS Top level Architecture.....	11
5.2 UFS Bus Master Transport Protocol (UBMTP) Layer.....	12
6 UFS UIC Layer: MIPI M-PHY and MIPI UniPro.....	13
6.1 Overview	13
6.2 HIBERNATE	13
7 UFS Bus Master Transport Protocol (UBMTP) Layer.....	15
7.1 Overview	15
7.2 Unified Memory Protocol Information Unit	15
7.2.1 UMPIU Transaction Codes	15
7.3 General Unified Memory Protocol Information Unit Format	17
7.3.1 Overview	17
7.3.2 Basic Header Format	18
7.3.3 COPY DATA UMPIU	19
7.3.4 UM COPY DATA UMPIU	21
7.3.5 ACKNOWLEDGE COPY UMPIU	22
7.3.6 ACCESS UM BUFFER UMPIU	23
7.3.7 WRITE UM BUFFER UMPIU	25
7.3.8 ACKNOWLEDGE UM BUFFER UMPIU.....	26
7.3.9 UM DATA OUT UMPIU	27
7.3.10 UM DATA IN UMPIU	29
7.3.11 Unified Memory Basic Operations.....	31
7.3.12 SCSI Read / Write and Unified Memory Operations	34
8 UFS Functional Descriptions Related to Unified Memory	41
8.1 UFS Boot with Unified Memory Support	41
8.1.1 Introduction	41
8.1.2 Initialization of Unified Memory	41
8.2 UFS Cache in Unified Memory.....	42
8.2.1 Example of How to Use Write Buffer Cache in Unified Memory	43
8.3 Host Device Interaction.....	43
8.3.1 Background Operation Mode with Unified Memory	43
9 UFS Descriptors, Flags and Attributes.....	44
9.1 Device Descriptor.....	44
9.2 Flags	45
9.3 Attributes.....	46
Annex A Differences between JESD220-1A and JESD220-1.....	47

Contents (cont'd)

	Page
Figures	
Figure 4.1 – Conventional Memory Structure	6
Figure 4.2 – UMA Memory Structure	6
Figure 4.3 – Inter-Memory Communication of Device-Integrated RAM Buffering	7
Figure 4.4 – Inter-Memory Communication of UM Buffering.....	7
Figure 4.5 – Host Memory Map (Physical)	8
Figure 4.6 – Definition of Latency	9
Figure 4.7 – Definition of Multiple Outstanding	10
Figure 5.1 – Modified UFS Top Level Architecture	11
Figure 6.1 – Flag Manipulation before HIBERNATE.....	14
Figure 6.2 – Flag Manipulation after HIBERNATE	14
Figure 7.1 – Unified Memory Read.....	31
Figure 7.2 – Unified Memory Write.....	32
Figure 7.3 – Copy between System Memory and Unified Memory	33
Figure 7.4 – Generic Copy in the Unified Memory	33
Figure 7.5 – SCSI Read (Non-Cached) and Subsequent Unified Memory Operations	35
Figure 7.6 – SCSI Write (Non-Cached) and Subsequent Unified Memory Operations	35
Figure 7.7 – SCSI Write (Cached) and Subsequent Unified Memory Operations.....	36
Figure 7.8 – SCSI Read (Cached) and subsequent Unified Memory Operations (Simple Single Read)	37
Figure 7.9 – SCSI Read (Cached) and subsequent Unified Memory Operations (Large Data Read with Filling Missing Write Buffer)	38
Figure 7.10 – SCSI Read (Cached) and subsequent Unified Memory Operations (Large Data Read with Flushing Dirty Write Buffer).....	39
Figure 7.11 – L2P Table Operation	40
Figure 8.1 – Unified Memory Initialization Sequence Diagram.....	42
Tables	
Table 7.1 – UMPIU Transaction Codes.....	15
Table 7.2 – UMPIU Transaction Code Definitions	16
Table 7.3 – General Format of the Unified Memory Protocol Information Unit.....	17
Table 7.4 – Basic Header Format	18
Table 7.5 – Transaction Type Format.....	18
Table 7.6 – UMPIU Flags.....	18
Table 7.7 – COPY DATA UMPIU.....	19
Table 7.8 – Flags Definition of ACKNOWLEDGE COPY UMPIU.....	20
Table 7.9 – UM COPY DATA UMPIU	21
Table 7.10 – ACKNOWLEDGE COPY UMPIU.....	22
Table 7.11 – Flags Definition of ACKNOWLEDGE COPY UMPIU.....	22
Table 7.12 – ACCESS UM BUFFER UMPIU	23
Table 7.13 – Flags Definition for ACCESS UM BUFFER UMPIU	24
Table 7.14 – WRITE UM BUFFER UMPIU	25
Table 7.15 – ACKNOWLEDGE UM BUFFER UMPIU	26
Table 7.16 – Flags Definition for ACKNOWLEDGE UM BUFFER UMPIU	26
Table 7.17 – UM DATA OUT UMPIU.....	27
Table 7.18 – Flags Definition for UM DATA OUT UMPIU	28
Table 7.19 – UM DATA IN UMPIU.....	29
Table 9.1 – Device Descriptor	44
Table 9.2 – Flags	45
Table 9.3 – Attributes	46

Foreword

This Universal Flash Storage (UFS) Unified Memory Extension standard is an extension to JESD220, Universal Flash Storage (UFS) Standard.

Introduction

The UFS standard defines a managed storage device. UFS devices are designed to offer high performance with low power consumption. The UFS device contains features that support both high throughput for large data transfers and performance for small random data accesses.

The UFS Unified Memory Extension standard describes the requirements to implement unified memory functionality in a UFS device.

UNIVERSAL FLASH STORAGE (UFS) UNIFIED MEMORY EXTENSION, Version 1.1

(From JEDEC Board Ballot, JCB-16-13, formulated under the cognizance of the JC-64.1 Subcommittee on Electrical Specifications and Command Protocols (Item 133.11).)

1 SCOPE

This document provides a comprehensive definition of the unified memory support for UFS. It also provides some details in how to utilize the unified memory for realizing high performance in UFS devices.

2 NORMATIVE REFERENCE

The following normative documents contain provisions that, through reference in this text, constitute provisions of this standard. For dated references, subsequent amendments to, or revisions of, any of these publications do not apply. However, parties to agreements based on this standard are encouraged to investigate the possibility of applying the most recent editions of the normative documents indicated below. For undated references, the latest edition of the normative document referred to applies.

[UFS], JEDEC JESD220C, Universal Flash Storage (UFS), Version 2.1.

[UFSHCI], JEDEC JESD223C, Universal Flash Storage Host Controller Interface (UFSHCI), Version 2.1.

[UFSHCI-UME], JEDEC JESD223-1A, Universal Flash Storage Host Controller Interface (UFSHCI) Unified Memory Extension Version 1.1.

3 TERMS AND DEFINITIONS

For the purposes of this standard, the terms and definitions given in the document included in section 2, Normative Reference, and the following apply.

3.1 Acronyms

CPORT	A CPort is a Service Access Point on the UniPro Transport Layer (L4) within a Device that is used for Connection-oriented data transmission
DMA	Direct Memory Access
GB	Gigabyte
HCI	Host Controller Interface
KB	Kilobyte
LUN	Logical Unit Number
MB	Megabyte
MIPI	Mobile Industry Processor Interface
TB	Terabyte
TC	Traffic Class
UFS	Universal Flash Storage
UM	Unified Memory
UMA	Unified Memory Architecture
UME	Unified Memory Extension
UMPIU	Unified Memory Protocol Information Unit
UniPro	Unified Protocol.
UPIU	UFS Protocol Information Unit
UTP	UFS Transport Protocol

3.2 Terms and Definitions

Application Client: An entity that is the source of SCSI commands and task management function requests in the host.

Byte: An 8-bit data value with the most significant bit labeled as bit 7 and the least significant bit as bit 0.

Device Server: An entity in the device that processes SCSI commands and task management functions.

Doubleword: A 32-bit data value with the most significant bit labeled as bit 31 and the least significant bit as bit 0.

Dword: A 32-bit data value, a Doubleword.

Gigabyte: 1,073,741,824 or 2^{30} bytes.

Host: An entity or a device with the characteristics of a primary computing device that includes one or more SCSI initiator devices.

Initiator Device: Within a transaction, the originator of a SCSI command request message to a target device.

Kilobyte: 1024 or 2^{10} bytes.

Logical Unit: A logical unit is an internal entity of a bus device that performs a certain function or addresses a particular space or configuration within a bus device.

Logical Unit Number: A numeric value that identifies a logical unit within a device.

Megabyte: 1,048,576 or 2^{20} bytes.

Quadword: A 64-bit data value with the most significant bit labeled as bit 63 and the least significant bit as 0.

Segment: A specified number of sequentially addressed bytes representing a data structure or section of a data structure.

Target Device: Within a transaction, the recipient of a SCSI command request message from an initiator device.

Task: A task is a SCSI command which includes all transactions to complete all data transfers and a status response that will satisfy the requirements of the requested services of the command.

Terabyte: 1.099.511.627.776 or 2^{40} bytes.

Transaction: A UFS bus primitive action which results in transmission of serial data packets between a target and initiator.

UFS Protocol Information Unit: Information transfer (communication) between a UFS host and device is done through messages which are called UFS Protocol Information Units. These messages are UFS defined data structures that contain a number of sequentially addressed bytes arranged as various information fields.

Word: A 16-bit data value with the most significant bit labeled as bit 15 and the least significant bit as bit 0.

3.3 Keywords

Several keywords are used to differentiate levels of requirements and options, as follow:

Can - A keyword used for statements of possibility and capability, whether material, physical, or causal (*can* equals *is able to*).

Expected - A keyword used to describe the behavior of the hardware or software in the design models assumed by this standard. Other hardware and software design models may also be implemented.

Ignored - A keyword that describes bits, bytes, quadlets, or fields whose values are not checked by the recipient.

Mandatory - A keyword that indicates items required to be implemented as defined by this standard.

May - A keyword that indicates a course of action permissible within the limits of the standard (*may* equals *is permitted*).

Must - The use of the word *must* is deprecated and shall not be used when stating mandatory requirements; *must* is used only to describe unavoidable situations.

Optional - A keyword that describes features which are not required to be implemented by this standard. However, if any optional feature defined by the standard is implemented, it shall be implemented as defined by the standard.

Reserved - A keyword used to describe objects—bits, bytes, and fields—or the code values assigned to these objects in cases where either the object or the code value is set aside for future standardization. Usage and interpretation may be specified by future extensions to this or other standards. A reserved object shall be zeroed or, upon development of a future standard, set to a value specified by such a standard. The recipient of a reserved object shall not check its value. The recipient of a defined object shall check its value and reject reserved code values.

Shall - A keyword that indicates a mandatory requirement strictly to be followed in order to conform to the standard and from which no deviation is permitted (*shall* equals *is required to*). Designers are required to implement all such mandatory requirements to assure interoperability with other products conforming to this standard.

Should - A keyword used to indicate that among several possibilities one is recommended as particularly suitable, without mentioning or excluding others; or that a certain course of action is preferred but not necessarily required; or that (in the negative form) a certain course of action is deprecated but not prohibited (*should* equals *is recommended that*).

Will - The use of the word *will* is deprecated and shall not be used when stating mandatory requirements; *will* is only used in statements of fact.

3.4 Abbreviations

etc. - And so forth (Latin: et cetera)

e.g. - For example (Latin: exempli gratia)

i.e. - That is (Latin: id est)

3.5 Conventions

UFS specification follows some conventions used in SCSI documents since it adopts several SCSI standards.

A binary number is represented in this standard by any sequence of digits consisting of only the Western-Arabic numerals 0 and 1 immediately followed by a lower-case b (e.g., 0101b). Spaces may be included in binary number representations to increase readability or delineate field boundaries (e.g., 0 0101 1010b).

A hexadecimal number is represented in this standard by any sequence of digits consisting of only the Western-Arabic numerals 0 through 9 and/or the upper-case English letters A through F immediately followed by a lower-case h (e.g., FA23h). Spaces may be included in hexadecimal number representations to increase readability or delineate field boundaries (e.g., B FD8C FA23h).

A decimal number is represented in this standard by any sequence of digits consisting of only the Western-Arabic numerals 0 through 9 not immediately followed by a lower-case b or lower-case h (e.g., 25).

A range of numeric values is represented in this standard in the form "a to z", where a is the first value included in the range, all values between a and z are included in the range, and z is the last value included in the range (e.g., the representation "0h to 3h" includes the values 0h, 1h, 2h, and 3h).

When the value of the bit or field is not relevant, x or xx appears in place of a specific value.

The first letter of the name of a Flag is a lower-case f (e.g., fMyFlag).

The first letter of the name of a parameter included in a Descriptor or the first letter of the name of an Attribute is:

- a lower-case b if the parameter or the Attribute size is one byte (e.g., bMyParameter),
- a lower-case w if the parameter or the Attribute size is two bytes (e.g., wMyParameter),
- a lower-case d if the parameter or the Attribute size is four bytes (e.g., dMyParameter),
- a lower-case q if the parameter or the Attribute size is eight bytes (e.g., qMyParameter).

4 INTRODUCTION

4.1 Overview

Unified Memory (UM) architecture is a very versatile memory architecture which enables sharing the Host memory with Device. A typical example is today’s graphics memory used in popular CPUs, graphics workstations, etc.

Unified Memory Architecture (UMA) enables using Host memory as Device working memory, e.g., as L2P table cache, write buffer, etc. The UM area is physically located on the Host side, but logically belongs to the Device as if it would be physical RAM in the Device. Thus the UM area is logically a property of the Device, and it is a replacement of the Device-integrated RAM.

Figure 4.1 shows conventional memory structure of Host and Device, and Figure 4.2 shows that of UMA.

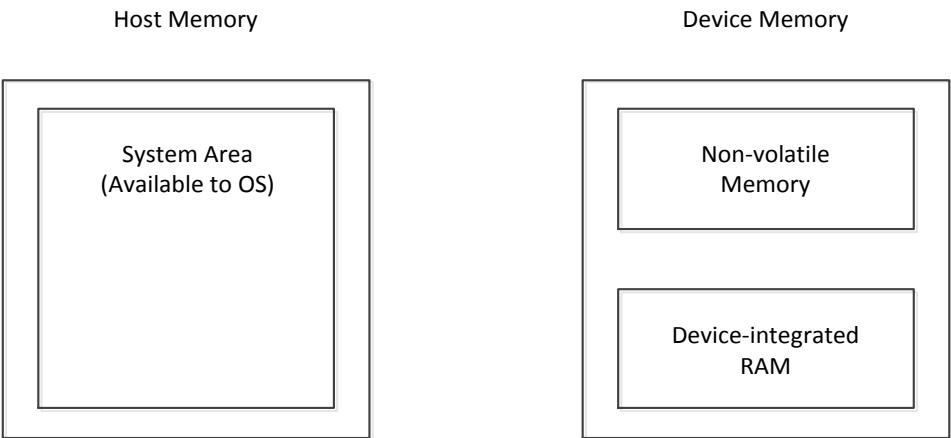


Figure 4.1 — Conventional Memory Structure

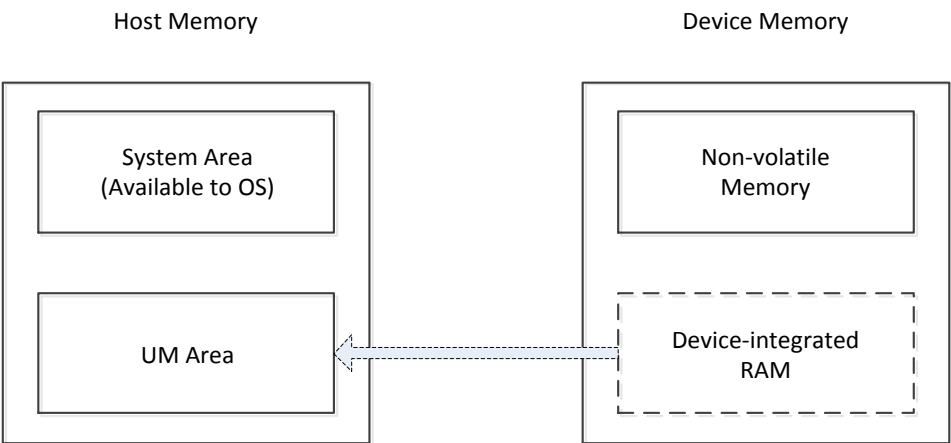


Figure 4.2 — UMA Memory Structure

4.2 Purpose of Unified Memory Support

The purpose of UM is to realize high performance at a reasonable cost. UM provides large working memory (e.g., several Mbytes), which is larger than that of Device-integrated RAM, to the Device.

L2P table access speed is important for high performance operation of non-volatile memory. Device-integrated RAM or single-level cell (SLC) area of non-volatile memory is usually used to speedup L2P table access. Buffer RAM is effective to realize high performance for small random data accesses. Device-integrated RAM is usually used for high performance storage like Solid State Drive (SSD). However such a Device-embedded RAM is costly, and it is sometimes difficult to use for embedded storage. UMA gives an appropriate solution to this problem.

The basic concept of UMA is supported by high-speed communication of UFS. The speed is slightly slower than that of internal bus, but still much faster than that of non-volatile memory access. Thus UM can work as working memory of Device, and it is a best use of hi-speed feature of UFS.

Figure 4.3 shows Inter-Memory Communication of Device-Integrated RAM Buffering in SCSI WRITE, and Figure 4.4 shows Inter-Memory Communication of UM Buffering. The difference in SCSI data phase is that the UFS link is used for the former and the Host internal bus is used for the latter. In write buffer flush, to the contrary, the Device internal bus and the UFS link are used, respectively.

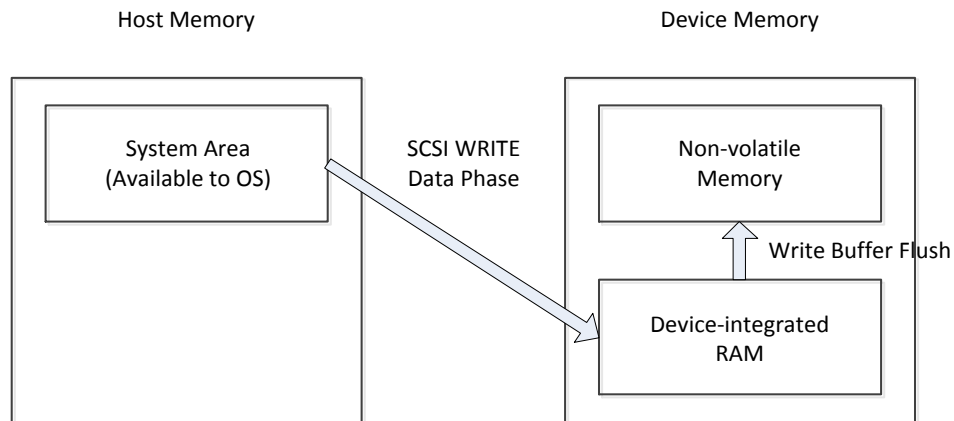


Figure 4.3 — Inter-Memory Communication of Device-Integrated RAM Buffering

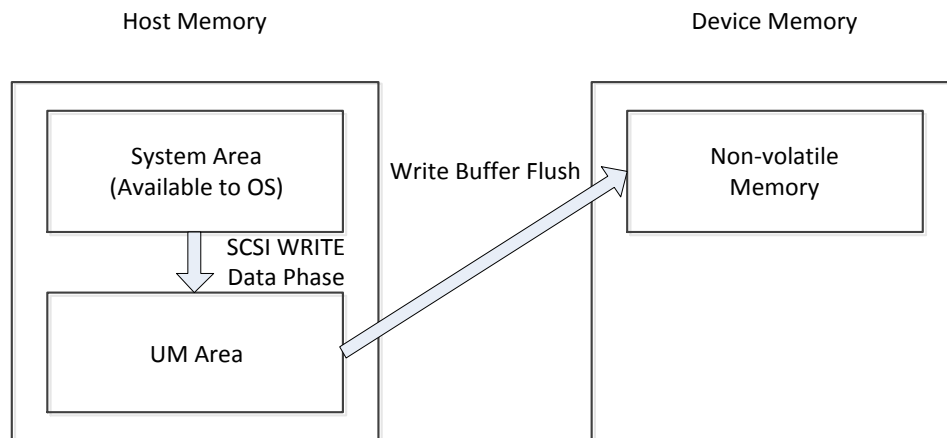


Figure 4.4 — Inter-Memory Communication of UM Buffering

4.3 Framework of Unified Memory

Today's computer usually has the virtual memory architecture, where CPU accesses the system memory using virtual address. However devices connected to the internal bus access the memory using either virtual address or physical address, depending on the position of the memory management unit. Using physical address is more common, and it is assumed in this section.

4.3.1 Host Memory Map

UM is a part of Host memory, more precisely a part of Host physical memory. The Host memory has at least the system memory that is managed by the operating system. Some parts of the Host memory may be used for other dedicated purposes, e.g., buffering data of devices, and UM is included as one of them. The UM Base Address Register points to the start address of UM (UM Base Address) as shown in Figure 4.5.

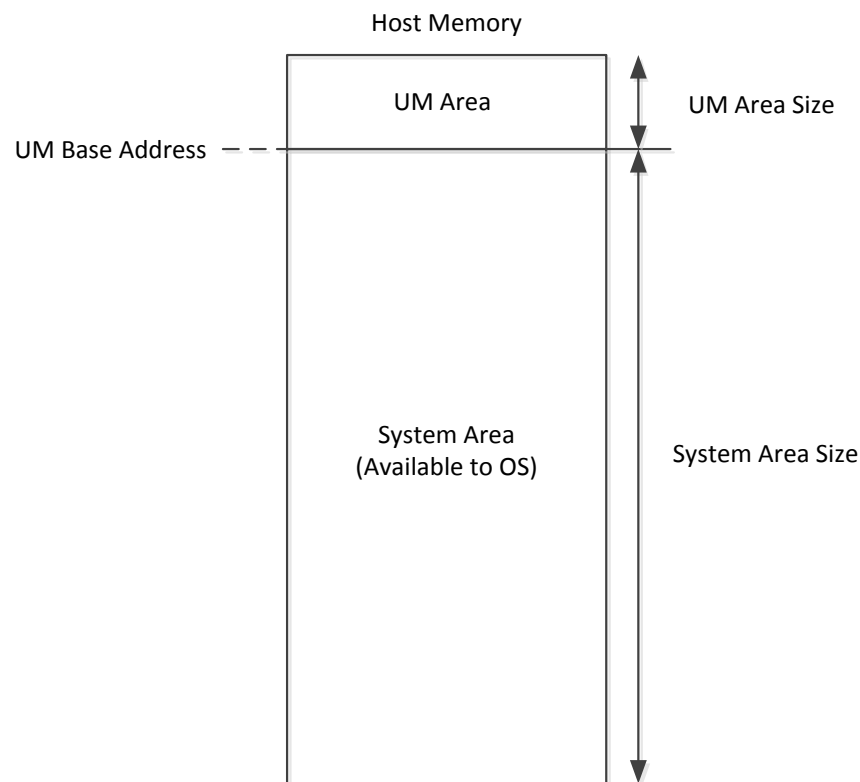


Figure 4.5 — Host Memory Map (Physical)

The UM Area Size is decided by the Host, after referring to `dMinUMAreaSize` in the Device Descriptor (See Table 9.1 for detail.), and is told to the Device in `dUMAreaSize` in the Attributes (See Table 9.3 for detail.).

4.3.2 Configuration of Unified Memory

UM is used as a working memory of Device. The UM may contain the following items:

- Logical-to-Physical address translation table (L2P Table) cache,
- Write Buffer (WB) cache,
- L2P Table cache tag,
- WB cache tag.

Some of the above may exist in the Device, or more items may be contained in the UM; it is an implementation matter, and the standard is UM content agnostic. The UM accesses are performed only using physical addresses. Only offsets from the UM Base Address are specified in the command from the Device to the Host, and the Device is responsible for all of the UM operations and all contents of the UM.

4.4 Speed Estimation

It is difficult to estimate operation speed correctly because it varies according to running applications. Thus benchmark tests are adopted to estimate the speed.

The following three parameters give hints of speed estimation:

- Bandwidth,
- Latency,
- Number of outstanding commands.

Bandwidth depends on the Gear of UFS, e.g., 2.9 Gbps with Gear 2 (single lane) and 5.8 Gbps with Gear 3 (single lane).

The definition of latency is shown in Figure 4.6.

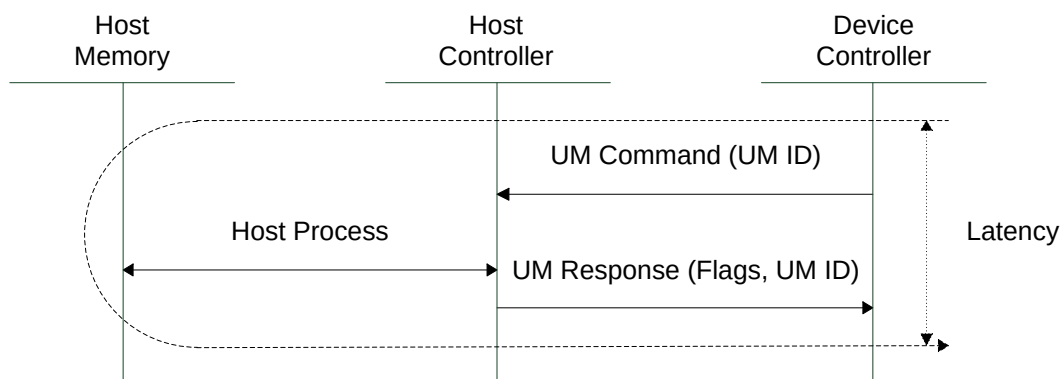


Figure 4.6 — Definition of Latency

Latency can be partly hidden by issuing UM commands in parallel, i.e., multiple outstanding commands (See Figure 4.7 for the definition of multiple outstanding commands.). Thus the number of outstanding commands is important. In this version of the standard, 32 is the default.

4.4 Speed Estimation (cont'd)

The concrete parameter values of a product should be specified in the data sheet provided by the manufacturer.

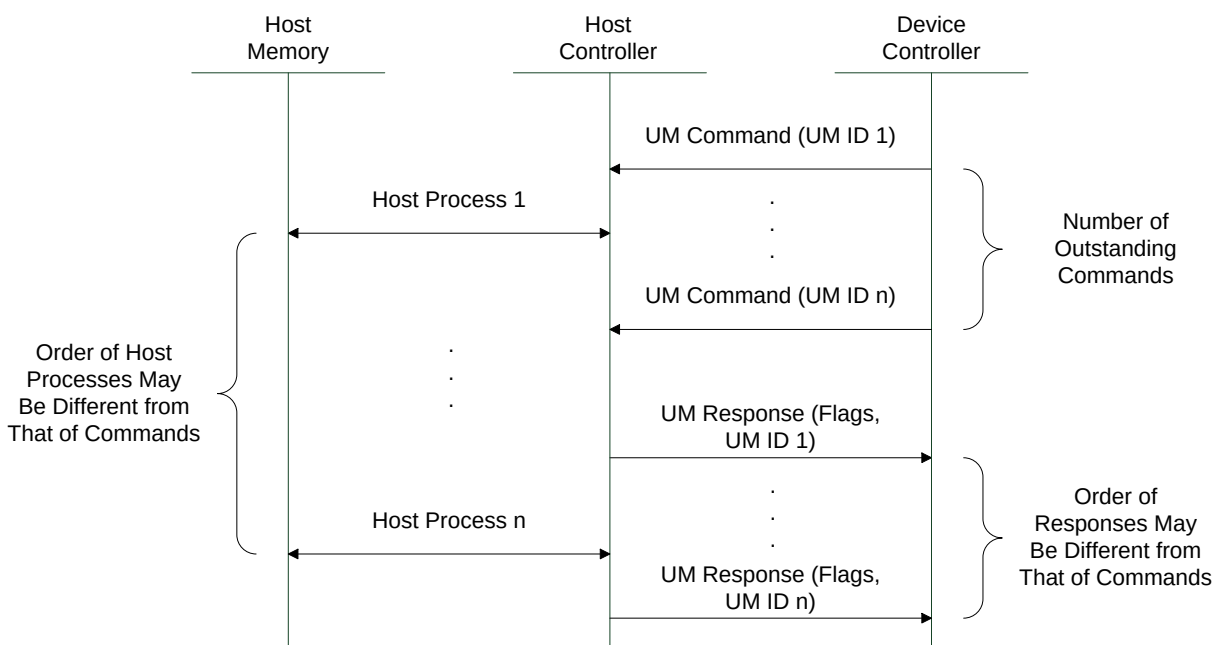


Figure 4.7 — Definition of Multiple Outstanding Commands