JEDEC STANDARD

Universal Flash Storage (UFS) Host Controller Interface

JESD223

AUGUST 2011

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 2011
3103 North 10th Street
Suite 240 South
Arlington, VA 22201-2107

PRICE: Contact JEDEC

Printed in the U.S.A.

PLEASE!

DON'T VIOLATE
THE
LAW!

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

UNIVERSAL FLASH STORAGE UFS HOST CONTROLLER INTERFACE

Contents

1 Scope 1
2 Normative Reference 1
3 Acronyms, Terms and Definitions, Keywords, and Conventions 2
4 Architectural Overview 5
4.1 Outside of Scope 5
4.2 Interface Architecture 6
4.2 Interface Architecture (cont'd) 7
4.3 Transfer Request Interface 8
4.3 Transfer Request Interface (cont'd) 9
4.4 Limitations 9
5 UFS Host Controller Register Interface 9
5.1 Register Map 10
5.2 Host Controller Capabilities Registers 11
5.2.1 Offset 00h: CAP – Controller Capabilities 11
5.2.2 Offset 08h: VER – UFS Version 11
5.2.3 Offset 10h: HCDDID – Host Controller Identification Descriptor – Device ID and Device Class 12
5.2.4 Offset 14h: HCPMID – Host Controller Identification Descriptor – Product ID and Manufacturer ID 12
5.3 Operation and Runtime Registers 13
5.3.1 Offset 20h: IS – Interrupt Status 13
5.3.2 Offset 24h: IE – Interrupt Enable 14
5.3.3 Offset 30h: HCS – Host Controller Status 15
5.3.4 Offset 34h: HCE – Host Controller Enable 16
5.3.5 Offset 38h: UECPA – Host UIC Error Code PHY Adapter Layer 16
5.3.6 Offset 3Ch: UECDL – Host UIC Error Code Data Link Layer 17
5.3.7 Offset 40h: UECN – Host UIC Error Code Network Layer 17
5.3.8 Offset 44h: UECT – Host UIC Error Code Transport Layer 18
5.3.9 Offset 48h: UECDME – Host UIC Error Code 18
5.3.10 Offset 4Ch: UTRIACR – UTP Transfer Request Interrupt Aggregation Control Register 19
5.4 UTP Transfer Request Registers 20
5.4.1 Offset 50h: UTRLBA – UTP Transfer Request List Base Address 20
5.4.2 Offset 54h: UTRLBAU – UTP Transfer Request List Base Address Upper 32-bits 20
5.4.3 Offset 58h: UTRLDBR – UTP Transfer Request List Door Bell Register 21
5.4.4 Offset 5Ch: UTRLCLR – UTP Transfer Request List CLear Register 21
5.4.5 Offset 60h: UTRLRSR – UTP Transfer Request List Run Stop Register 21
5.5 UTP Task Management Registers 22
5.5.1 Offset 70h: UTMRLBA – UTP Task Management Request List Base Address 22
5.5.2 Offset 74h: UTMRLBAU – UTP Task Management Request List Base Address Upper 32-bits 22
5.5.3 Offset 78h: UTMRLDBR – UTP Task Management Request List Door Bell Register 22
5.5.4 Offset 7Ch: UTMRLCLR – UTP Task Management Request List CLear Register 23
5.5.5 Offset 80h: UTMRLRSR – UTP Task Management Request List Run Stop Register 23
5.6 UIC Command Registers 24
5.6.1 Offset 90h: UICCMD – UIC Command 24
5.6.2 Offset 94h: UICCMDARG1 – UIC Command Argument 1 25
5.6.3 Offset 98h: UICCMDARG2 – UIC Command Argument 2 26
5.6.4 Offset 9Ch: UICCMDARG3 – UIC Command Argument 3 27
5.6.5 UFS-Specific Attributes for Local L2 Timers 27
5.7 Vendor Specific Registers 29
5.7.1 Offset F0h to FFh: VS – Vendor Specific 29

UNIVERSAL FLASH STORAGE UFS HOST CONTROLLER INTERFACE

Contents (cont'd)

6 Data structures 29
6.2 UTP Transfer Request List 29
6.2.1 UTP Transfer Request Descriptor 30
6.2.2 UTP Command Descriptor 32
6.3 UTP Task Management Request List 34
6.3.1 UTP Task Management Request Descriptor 34
7 Theory of Operation 35
7.2 Host Controller Configuration and Control 36
7.2.1 Host Controller Initialization 36
7.2.2 Host controller configuration and control 38
7.3 Data Transfer Operation 38
7.3.1 Basic Steps when Building a UTP Transfer Request 38
7.3.2 UPIU Processing 39
7.3.3 Processing UTP Transfer Request Completion 41
7.4 Task Management Function 42
7.4.1 Basic Steps when Building a UTP Task Management Request 42
7.4.2 Processing UTP Task Management Completion 43
7.5 UIC Power Mode Change 43
7.6 UFSHCI Internal Rules 45
7.6.1 Command Processing Order 45
8 Error reporting and handling 46
8.2 Error Types 46
8.2.1 System Bus Error 46
8.2.2 UIC Error 46
8.2.3 UIC Command Error 46
8.2.4 UTP Error 47
8.2.5 Host controller Fatal Error 47
8.2.6 Device Error 47
8.3 Error Handling 48
8.3.1 System Bus Error Handling 48
8.3.2 UIC Error Handling 49
8.3.3 UIC Command Error Handling 49
8.3.4 UTP Error Handling 49
8.3.5 Host Controller Error Handling 50
8.3.6 Device Error Handling 50

Figures

Figure 1 UFS Architecture Overview 5
Figure 2 General architecture of UFS Host Controller Interface. 6
Figure 3 A conceptual block diagram of UFS host system 8
Figure 4 UTP Transfer Request Descriptor 30
Figure 5 UTP Command Descriptor (UCD) 32
Figure 6 UTP Task Management Request Descriptor. 34
Figure 7 Host controller link startup sequence 37
Figure 8 UIC Power mode change 44
Figure 9 Command processing order 45

UFS HOST CONTROLLER INTERFACE

(From JEDEC Board ballot JCB-11-37, formulated under the cognizance of the JC-64.3 Subcommittee on Host Controller.)

1 Scope

This standard describes a functional specification of the Host Controller Interface (HCI) for Universal Flash Storage (UFS). The objective of UFSHCI is to provide a uniform interface method of accessing the UFS hardware capabilities so that a standard/common Driver can be provided for the Host Controller. The common Driver would work with UFS host controller from any vendor. This standard includes a description of the hardware/software interface between system software and the host controller hardware. It is intended for hardware designers, system builders and software developers. This standard is a companion document to JESD220, Universal Flash Storage (UFS). The reader is assumed to be familiar with JESD220, MIPI UniPro specification version 1.4, and MIPI M-PHY specification 1.0.

Clause 4 provides a brief overview of the architectural overview of UFS. Clause 5 describes the register interface of UFSHCI. Clause 6 describes the data structure used by UFSHCI. Clause 7 provides a theory of operation for UFSHCI. Clause 8 describes the error recovery process for UFSHCI.

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.

MIPI Alliance Specification for M-PHY℠,Version 1.00.00

MIPI Alliance Specification for Unified Protocol (UniPro℠), Version 1.40.00

INCITS/T10, 2104-D, *SCSI Architecture Model – 5 (SAM-5)*

INCITS/T10, 1799-D, *SCSI Block Commands - 3 (SBC-3)*

INCITS/T10, 1240-D, *Reduced Block Commands (RBC)*

INCITS/T10, 1731-D, *SCSI Primary Commands (SPC-4)*

JEDEC JESD220, *Universal Flash Storage (UFS)*

3 Acronyms, Terms and Definitions, Keywords, and Conventions

3.1 Acronyms

DID	Device ID
GB	Gigabyte
HCI	Host Controller Interface
KB	Kilobyte
LUN	Logical Unit Number
MIPI	Mobile Industry Processor Interface
MB	Megabyte
NA	Not applicable
PRDT	Physical Region Description Table
UCD	UTP Command Descriptor
UFS	Universal Flash Storage
UPIU	UFS Protocol Information Unit
UTP	UFS Transport Protocol
UTRD	UTP Transfer Request Descriptor
UTMRD	UTP Task Management Request Descriptor

3.2 Terms and Definitions

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

Device ID: The bus address of a UFS device

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

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

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

Host: An addressable device on the UFS bus which is usually the main CPU that hosts the UFS bus

Kilobyte(KB): 1024 or 2^{10}bytes.

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

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

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

3.2 Terms and Definitions (cont'd)

UFS Protocol Information Unit: Information transfer (communication) between a UFS host and device is done through messages which are called UFS Protocol Information Units.

NOTE The messages are UFS defined data structures that contain a number of sequentially addressed bytes arranged as various information fields.

UTP Transfer Request Descriptor: A data structure in system memory that contains a UTP command and additional contextual information needed to carry out the command operation.

NOTE The command is limited to USF adopted SCSI command sets, UFS native command set and Device Management function that uses UTP protocol. A UTP Transfer Request Descriptor is built by the host and is targeted at the attached UFS device.

UTP Task Management Request Descriptor: A data structure in system memory that contains a UTP Task Management Function and the additional contextual information needed to execute the function.

NOTE A UTP Transfer Request Descriptor is built by the host and is executed by the attached UFS device.

Unit: A bus device

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

zero-based value: A numeric value N (N > 0) represented by N-1

3.3 Keywords

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

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 flexibility of choice with no implied preference.

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.

3.3 Keywords (cont'd)

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. Designers are required to implement all such mandatory requirements to assure interoperability with other products conforming to this standard.

3.4 Conventions

The conventions used for registers in this standard are defined in the sections that follow.

Hardware shall return '0' for all bits and registers that are marked as reserved, and host software shall write all reserved bits and registers with the value of '0'.

Inside the register section, the following abbreviations are used:

RO Read Only

ROC Read Only and Read to clear

RW Read Write

R/W Read Write. The value read may not be the last value written.

RWC Read/Write '1' to clear

RWS Read/Write '1' to set

Impl Spec Implementation Specific – the controller has the freedom to choose its implementation.

HwInit The default state is dependent on device and system configuration. The value is initialized at reset, either by an expansion ROM, or in the case of integrated devices, by a platform BIOS.

4 Architectural Overview

UFS is a simple, high performance, serial interface. It is primarily for use in mobile systems, between host processing and NVM mass storage devices.

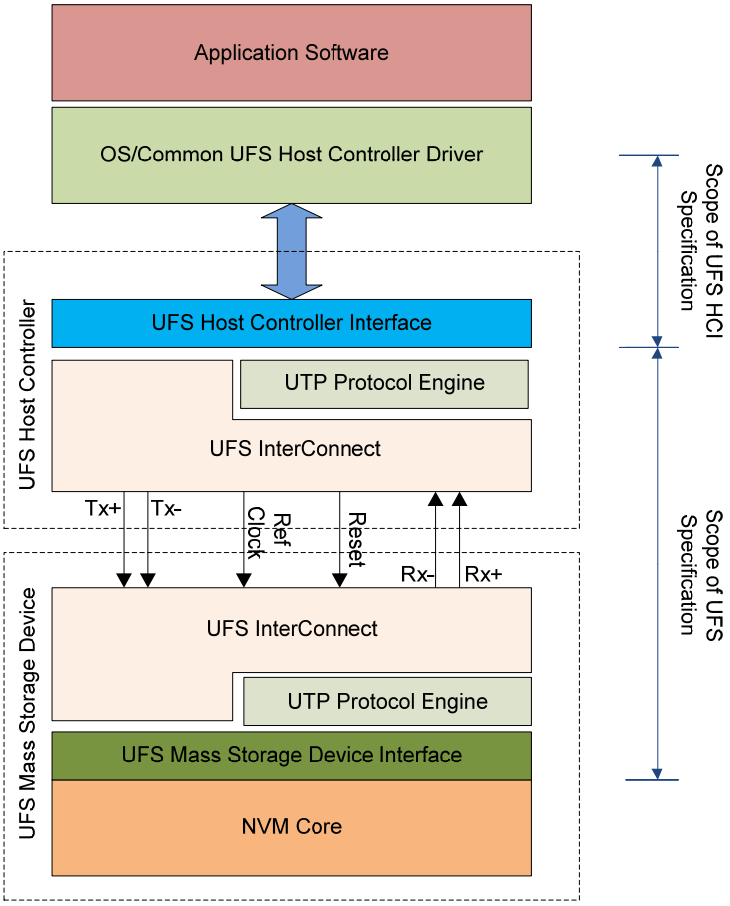


Figure 1 — UFS Architecture Overview

This standard defines the register-level host controller interface for JESD220, Universal Flash Storage (UFS). The UFS Host Controller is responsible for managing the interface between host SW and USF device and the data transfer. This includes interface management, power management, and control. Also included in this standard is the data transfer & programming model.

4.1 Outside of Scope

This standard does not contain information relevant to implementation of the UFS Interconnect as this is wholly described in MIPI UniPro Specification 1.4. This standard can be applied to any system bus interface. However this standard does not define a system bus that may be used in an UFS Host Controller implementation.

4.2 Interface Architecture

UFS host software uses a combination of a host register set and Transfer Request Descriptors in system memory to communicate with host controller hardware. Figure 2 illustrates a conceptual block diagram of UFS Host Controller Interface.

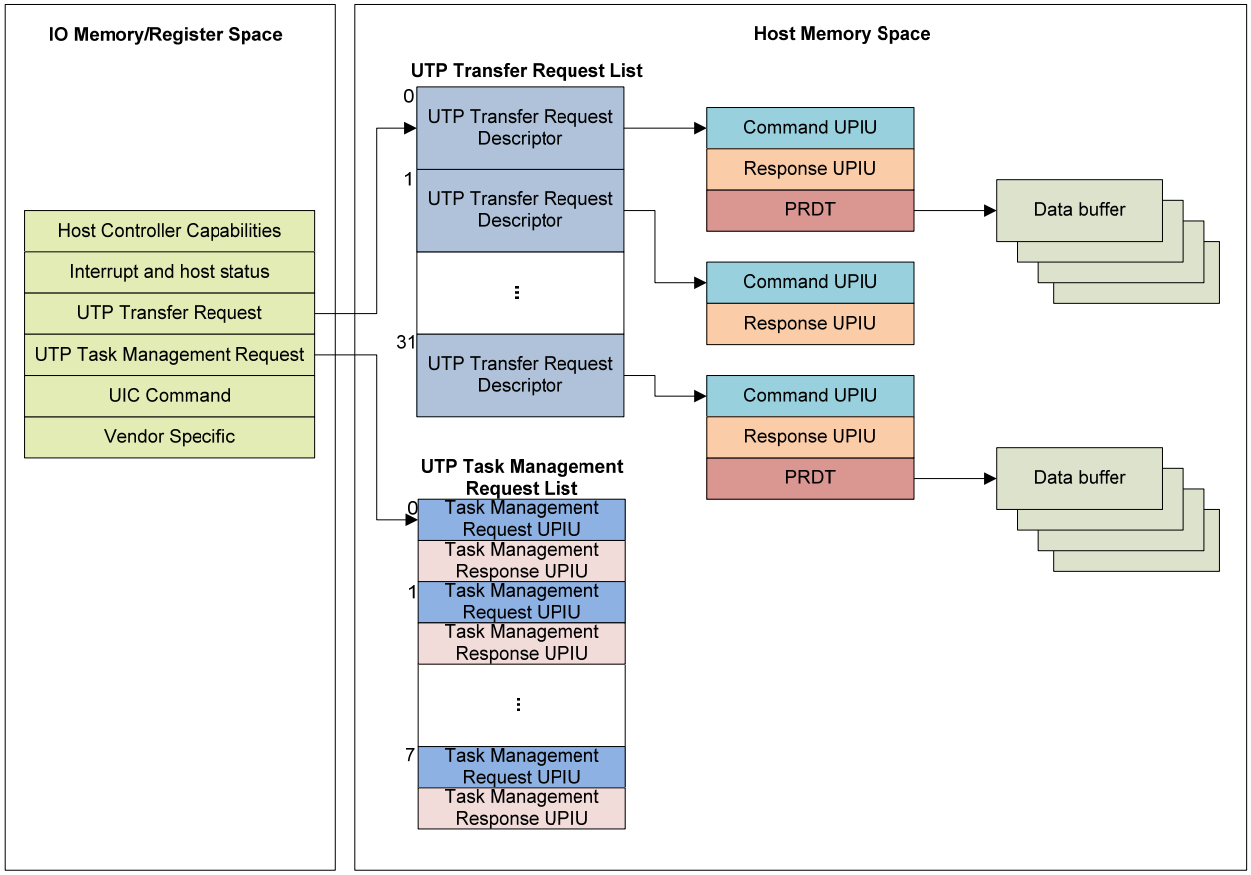


Figure 2 — General architecture of UFS Host Controller Interface.

4.2 Interface Architecture (cont'd)

UFSHCI defines two interface spaces.

- **MMIO Space**. In this space, a set of hardware registers are defined as the host controller interface to system software. It is normally implemented as Memory-Mapped I/O (MMIO) space, consisting of three types of registers:
  - **Host Controller Capability Registers.** These registers provide description of host controller capabilities. They include UFS standard version, the size of the command queue the host controller supported, and host controller identification data.
  - **Runtime and Operation Registers.** These include support for the following:
    - **Interrupt configuration**. These registers provide an interface for host SW to enable/disable interrupt and status of the interrupts.
    - **Host controller status**. This register shows the status of the host controller and allows host software to initialize/deactivate the host controller.
    - **UTP transfer Request List management**. These registers provide an interface to UTP Transfer Request List.
    - **UTP Task Management request lists management**. These registers provide an interface to UTP Transfer Request List.
    - **UIC Command Registers**. These registers provide an interface for UniPro configuration and control.
  - **Vendor Specific Registers.** These registers are defined by vendors.
- **Host Memory Space**. This space includes data structures that provide description of the commands for execution and the data buffers which are a part of each command. The data structures and data buffers are application protocol specific (UTP).

4.3 Transfer Request Interface

A UFS Host System is composed of a number of hardware and software layers. Figure 3 illustrates a conceptual block diagram of the building block layers in a host system.

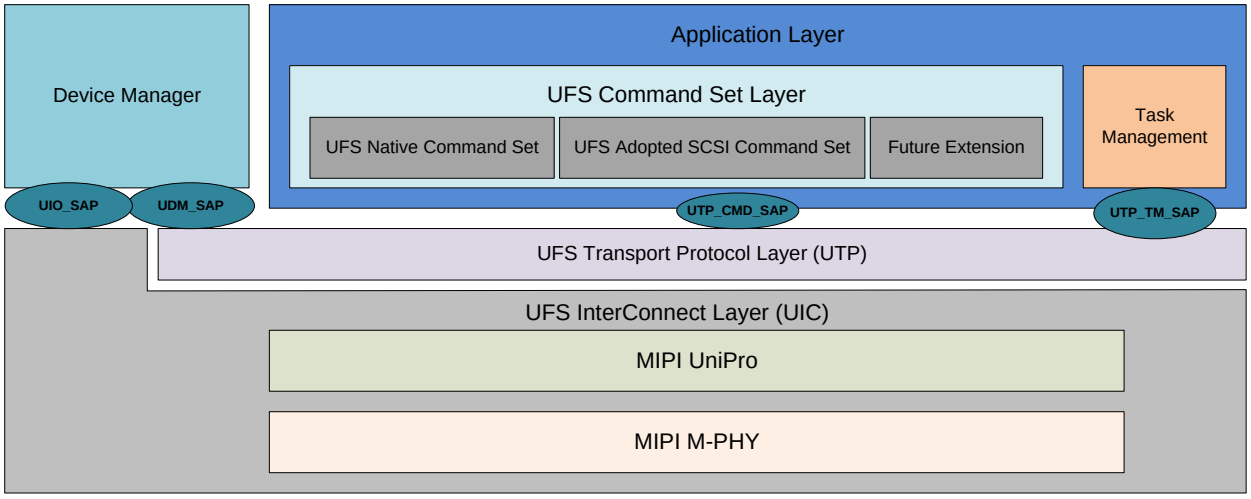


Figure 3 — A conceptual block diagram of UFS host system

UFSHCI defines a set of registers and data structures that work in concert with UFS host SW to implement the four service access points (SAPs) as described in the Figure 3: UIO_SAP, UDM_SAP, UTP_CMD_SAP and UTP_TM_SAP. Refer to JESD220, UFS Standard for the definition of the service access points.

To manage the communication between host SW and the UFS devices attached, the host controller provides three independent interfaces that host software uses to send a transfer request.

- **UTP Transfer Request List**. This list is used by host software to implement UTP_CMD_SAP and UDM_SAP. UTP_CMD_SAP includes support for the following command types:
  - All SCSI command sets adopted by UFS.
  - Native UFS command set.
  - Device management function via query request UPIU/query response UPIU.

  The List consists of a data structure called UFS Transfer Request Descriptor (UTRD). UTRD describes a command to be executed and the data associated it. UFS host SW issues a command to the host controller by placing a UTRD on the List then rings the Host Controller doorbell for the list. Commands are dispatched for execution by the UFSHCI in the order that they are placed on the List even though they may be completed out of the order. The host controller acts on behalf of host processor to manage all the data transfer operations associated with the command. All commands could result in a Command Completion interrupt or status field of the UTRD for the command in the List being updated. UFS SW may add commands to the List while it is running. The host controller supports interrupt aggregation, such that a single command completion interrupt is generated for a pre-defined number of command completions.

4.3 Transfer Request Interface (cont'd)

- **UTP Task Management Request List**. This list is used by host software to implement UTP_TM_SAP. The List consists of a data structure called UFS Task Management Request Descriptor (UTMRD). UTMRD describes a task management function that host software wants the attached device to execute. All the Task Management Requests will be prioritized over the transfer requests listed in UTP Transfer Request List as described above. UFS host SW issues a task management function to the host controller by placing a UTMRD on the List then rings the Host Controller doorbell for the list. Functions are dispatched for execution by the UFSHCI in the order that they are placed on the List even though they may be completed out of the order. All task management functions could result in a Request Completion interrupt or status field of the UTMRD for the function in the List being updated. UFS SW may add task management function to the List while it is running. Interrupt aggregation is not supported for this list.

- **UIC Command Register**. This register set is used by host software to execute a UIC command directly.

4.4 Limitations

The phase collapse feature, currently specified in JESD220, UFS Standard, will be removed from future UFS standards. As phase collapse also adds complexity to UFS Host Controller, the phase collapse feature is not supported by UFS HCI.

5 UFS Host Controller Register Interface

The host controller registers are memory mapped and exist in MMIO space. Registers access shall have a maximum size of 64-bits; 64-bit access shall not cross an 8-byte alignment boundary.

UFSHCI registers are used to control the operation of the Host Controller and also to read the status and interrupt information from the Host controller**.**

5.1 Register Map

The following is a standard register map for UFSHCI.

	Start	End	Symbol	Description
Host Capabilities	00h	03h	CAP	Host Controller **Cap**abiities
	04h	07h	Reserved	Reserved
	08h	0Bh	VER	UFS **Ver**sion
	0Ch	0Fh	Reserved	Reserved
	10h	13h	HCDDID	**H**ost **C**ontroller **I**dentification **D**escriptor – Device ID and Device Class
	14h	17h	HCPMID	**H**ost **C**ontroller **I**dentification **D**escriptor – Product ID and Manufacturer ID
	18h	1Fh	Reserved	Reserved
Operation and Runtime	20h	23h	IS	**I**nterrupt **S**tatus
	24h	27h	IE	**I**nterrupt **E**nable
	28h	2Fh	Reserved	Reserved
	30h	33h	HCS	**H**ost **C**ontroller **S**tatus
	34h	37h	HCE	**H**ost **C**ontroller **E**nable
	38h	3Bh	UECPA	Host **U**IC **E**rror **C**ode **P**HY **A**dapter Layer
	3Ch	3Fh	UECDL	Host **U**IC **E**rror **C**ode **D**ata Link Layer
	40h	43h	UECN	Host **U**IC **E**rror **C**ode **N**etwork Layer
	44h	47h	UECT	Host **U**IC **E**rror **C**ode **T**ransport Layer
	48h	4Bh	UECDME	Host **U**IC **E**rror **C**ode **DME**
	4Ch	4Fh	UTRIACR	**U**TP **T**ransfer **R**equest **I**nterrupt **A**ggregation **C**ontrol **R**egister
UTP Transfer	50h	53h	UTRLBA	**U**TP **T**ransfer **R**equest **L**ist **B**ase **A**ddress
	54h	57h	UTRLBAU	**U**TP **T**ransfer **R**equest **L**ist **B**ase **A**ddress **U**pper 32-Bits
	58h	5Bh	UTRLDBR	**U**TP **T**ransfer **R**equest **L**ist **D**oor **B**ell **R**egister
	5Ch	5Fh	UTRLCLR	**U**TP **T**ransfer **R**equest **L**ist **CL**ear **R**egister
	60h	64h	UTRLRSR	**U**TP **T**ransfer **R**equest **R**un-**S**top **R**egister
	68h	6Fh	Reserved	Reserved
UTP Task Managemeng	70h	73h	UTMRLBA	**U**TP **T**ask **M**anagement **R**equest **L**ist **B**ase **A**ddress
	74h	77h	UTMRLBAU	**U**TP **T**ask **M**anagement **R**equest **L**ist **B**ase **A**ddress **U**pper 32-Bits
	78h	7Bh	UTMRLDBR	**U**TP **T**ask **M**anagement **R**equest **L**ist **D**oor **B**ell **R**egister
	7Ch	7Fh	UTMRLCLR	**U**TP **T**ask **M**anagement **R**equest **L**ist **CL**ear **R**egister
	80h	83h	UTMRLRSR	**U**TP **T**ask **M**anagement **R**un-**S**top **R**egister
	84h	8Fh	Reserved	Reserved
UIC Command	90h	93h	UICCMDR	**U**IC **C**om**m**an**d** **R**egister
	94h	97h	UCMDARG1	**U**IC **C**om**m**an**d** **Arg**ument 1
	98h	9Bh	UCMDARG2	**U**IC **C**om**m**an**d** **Arg**ument 2
	9Ch	9Fh	UCMDARG3	**U**IC **C**om**m**an**d** **Arg**ument 3
	A0h	AFh	Reserved	Reserved
	B0h	EFh	Reserved	Reserved
Vendor Specific	F0h	FFh		Vendor Specific Registers

5.2 Host Controller Capabilities Registers

This section specifies the limits and capabilities of the host controller implementation. All Capability Registers are Read-Only (RO) or hardware Initialized. The offsets for these registers are all relative to the beginning of the host controller's MMIO address space.

5.2.1 Offset 00h: CAP – Controller Capabilities

Bit	Type	Reset	Description
31:27	RO	0	Reserved.
26	RO	Impl Spec	**UIC DME_TEST_MODE command supported (UICDMETMS):** Indicates whether the host controller supports the UniPro DME_TEST_MODE.req SAP primitive.
25	RO	Impl Spec	**Out of order data delivery supported (OODDS):** Indicates whether the host controller supports out of order data delivery for UTP data transfer. When set to '1', the host controller shall support of out of order data delivery from the target device. When set to '0', the host controller will not support out of order data delivery from the target device.
24	RO	Impl Spec	**64-bit addressing supported (64AS):** Indicates whether the host controller can access 64-bit data structures. When set to '1', the host controller shall make the 32-bit upper bits of the UTP Transfer Request List Base Address Upper 32-bit and UTP Task Management Request List Base Address upper 32-bit, the PRD Base, and each PRD entry read/write. When cleared to '0', these are read-only and treated as '0' by the host controller.
23:19	RO	0h	Reserved
18:16	RO	Impl Spec	**Number of UTP Task Management Request Slots (NUTMRS):** 0's based value indicating the number of slots provided by the UTP Task Management Request List. A minimum of 1 and maximum of 8 slots may be supported.
15:05	RO	0h	Reserved
04:00	RO	Impl Spec	**Number of UTP Transfer Request Slots (NUTRS):** 0's based value indicating the number of slots provided by the UTP Transfer Request List. A minimum of 1 and maximum of 32 slots may be supported.

5.2.2 Offset 08h: VER – UFS Version

This register indicates the major and minor versions of the UFSHCI specification that the controller implementation supports. The upper two bytes represent the major version number, and the lower two bytes represent the minor version number. Example: Version 3.12 would be represented as 00030102h. Valid versions of the standard are: 1.0.

Bit	Type	Reset	Description
31:16	RO	0001h	**Major Version Number (MJR):** Indicates the major version is "1"
15:00	RO	0000h	**Minor Version Number (MNR):** Indicates the minor version is "0".

5.2.3 Offset 10h: HCDDID – Host Controller Identification Descriptor – Device ID and Device Class

This register indicates the Device ID and Device Class identification information for host controller.

Bit	Type	Reset	Description
31:24	RO	Impl spec	**Device ID for Host Controller (HCDID):** UniPro Device ID assigned for host controller.
23:16	RO	0h	Reserved
15:00	RO	Impl spec	**Device Class (DC):** Device class for host controller (MIPI assigned).

5.2.4 Offset 14h: HCPMID – Host Controller Identification Descriptor – Product ID and Manufacturer ID

This register indicates the Product ID and Manufacturer ID identification information for host controller.

Bit	Type	Reset	Description
31:16	RO	Impl spec	**Product ID (PID):** Product ID for host controller maker
15:00	RO	Impl spec	**Manufacturer ID (MID):** Manufacturer ID (MIPI assigned) of host controller maker

5.3 Operation and Runtime Registers

This section defines the operation and runtime registers exposed by the host controller.

5.3.1 Offset 20h: IS – Interrupt Status

This register indicates pending interrupts that require service.

Bit	Type	Reset	Description
31:18	RO	0h	Reserved
17	RWC	0	**System Bus Fatal Error Status (SBFES):** Indicates that the host controller encountered a system bus error that it cannot recover from, such as a bad software pointer. When the error occurs, the host controller is stopped and both **UTRLRSR** and **UTMRLRSR** registers will be cleared to "0" by host controller.
16	RWC	0	**Host Controller Fatal Error Status (HCFES)**: Indicates that the host controller encountered a fatal error that it cannot recover from. When the error occurs, the host controller is stopped and both **UTRLRSR** and **UTMRLRSR** will be cleared to "0" by host controller. If the error occurs, host SW should reset the host controller.
15:12	RO		Reserved.
11	RWC	0	**Device Fatal Error Status (DFES):)**: Indicates that the host controller encountered a fatal error from device that it cannot recover. When the error occurs, the host controller is stopped and both **UTRLRSR** and **UTMRLRSR** will be cleared to "0" by host controller. If the error occurs, host SW should reset the host controller.
10	RWC	0	**UIC Command Completion Status (UCCS):** This bit is set to '1' by the host controller upon completion of a UIC command.
09	RWC	0	**UTP Task Management Request Completion Status (UTMRCS):** This bit is set to '1' by the host controller upon completion of a task management function.
08	RWC	0	**UIC Link Startup Status (ULSS):** indication that Link start-up process has been initiated by the remote end of the Link. This bit corresponds to the UniPro DME_LINKSTARTUP.ind SAP primitive.
07	RWC	0	**UIC Link Lost Status (ULLS):** This indicates a condition where remote end is trying to re-establish a link and the link is lost. This bit corresponds to the UniPro DME_LINKLOST.ind SAP primitive.
06	RWC	0	**UIC Hibernate Enter Status (UHES):** indicates that UniPro hibernate entering process has been completed. If the process was successful, the Link state is changed to the Hibernate state. Register HCS.UPMCRS indicates the status of the hibernation entering process. This bit corresponds to the UniPro DME_HIBERNATE_ENTER.ind SAP primitive.
05	RWC	0	**UIC Hibernate Exit Status (UHXS):** indicates that the Link has exited UniPro Hibernate state. Register **HCS.UPMCRS** indicates if exit process was unsuccessful the failure. This bit corresponds to the UniPro DME_HIBERNATE_EXIT.ind SAP primitive.
04	RWC	0	**UIC Power Mode Status (UPMS):** indicate that the *Unipro/PA/DL* part of the power mode change has been completed. Register **HCS.UPMCRS** contains the power mode change status. This bit corresponds to the UniPro DME_POWERMODE.ind SAP primitive..
03	RWC	0	**UIC Test Mode Status (UTMS):** Indicate that the peer UniPro stack has been set to a given UniPro test mode. This bit corresponds to the UniPro DME_TEST_MODE.ind SAP primitive.
02	RWC	0	**UIC Error (UE):** Indicate that a layer in the UniPro stack has encountered an error condition. Register **HCS.UEC** contains the error code for the condition. This bit corresponds to the UniPro DME_ERROR.ind SAP primitive.
01	RWC	0	**UIC DME_ENDPOINTRESET Indication(UDEPRI):** Indicate that the attached device has issued an *DME_ENDPOINTRESET* indication which is not allowed.
00	RWC	0	**UTP Transfer Request Completion Status (UTRCS):** This bit is set to '1' by the host controller upon one of the following: • Completion of a UTP transfer request with its UTRD Interrupt bit set to '1'. • Interrupt caused by the UTR interrupt aggregation logic.

5.3.2 Offset 24h: IE – Interrupt Enable

This register enables and disables the reporting of the corresponding interrupt to host software. When a bit is set (‘1’) and the corresponding interrupt condition is active, then an interrupt is generated. Interrupt sources that are disabled (‘0’) are still indicated in the IS register. This register is symmetrical with the IS register.

Bit	Type	Reset	Description
31:18	RO	0h	Reserved
17	RW	0	**System Bus Fatal Error Enable (SBFEE):** When set and **IS.SBFES** is set, the controller shall generate an interrupt.
16	RW	0	**Host Controller Fatal Error Enable (HCFEE):** When set and **IS.HCFES** is set, the controller shall generate an interrupt.
15:12	RO	0	Reserved
11	RWC	0	**Device Fatal Error Enable (DFEE):** When set and **IS.DFES** is set, the host controller shall generate an interrupt.
10	RWC	0	**UIC COMMAND Completion Enable (UCCE):** When set and **IS.UCCS** is set, the host controller shall generate an interrupt.
09	RWC	0	**UTP Task Management Request Completion Enable (UTMRCE):** When set and **IS.UTMRCS** is set, the host controller shall generate an interrupt.
08	RWC	0	**UIC Link Startup Status Enable (ULSSE):** When set and **IS.ULSS** is set, the controller shall generate an interrupt.
07	RWC	0	**UIC Link Lost Status Enable (ULLSE):** When set and **IS.ULLS** is set, the controller shall generate an interrupt.
06	RWC	0	**UIC Hibernate Enter Status Enable (UHESE):** When set and **IS.UHES** is set, the controller shall generate an interrupt.
05	RWC	0	**UIC Hibernate Exit Status Enable (UHXSE):** When set and **IS.UHXS** is set, the controller shall generate an interrupt.
04	RWC	0	**UIC Power Mode Status Enable (UPMSE):** When set and **IS.UPMS** is set, the controller shall generate an interrupt.
03	RWC	0	**UIC Test Mode Status Enable (UTMSE):** When set and **IS.UTMS** is set, the controller shall generate an interrupt.
02	RWC	0	**UIC Error Enable (UEE):** When set and **IS.UEE** is set, the controller shall generate an interrupt.
01	RWC	0	**UIC DME_ENDPOINTRESET (UDEPRIE):** When set and **IS. UDEPRI** is set, the controller shall generate an interrupt.
00	RWC	0	**UTP Transfer Request Completion Enable (UTRCE):** When set and **IS.UTRCS** is set, the host controller shall generate an interrupt.

5.3.3 Offset 30h: HCS – Host Controller Status

<table>
<tr><th>Bit</th><th>Type</th><th>Reset</th><th>Description</th></tr>
<tr><td>31:11</td><td>RO</td><td>0</td><td>Reserved</td></tr>
<tr><td>10:08</td><td>RW</td><td>0</td><td>UIC Power Mode Change Request Status (UPMCRS): Indicate that the status of a UIC layer request for power mode change.
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>0h</td><td>PWR_OK. The request was accepted.</td></tr>
<tr><td>1h</td><td>PWR_LOCAL. The local request was successfully applied.</td></tr>
<tr><td>2h</td><td>PWR_REMOTE.The remote request was successfully applied.</td></tr>
<tr><td>3h</td><td>PWR_BUSY.The request was aborted due to concurrent requests.</td></tr>
<tr><td>4h</td><td>PWR_ERROR_CAP.The request was rejected because the requested configuration exceeded the Link’s capabilities.</td></tr>
<tr><td>5h</td><td>PWR_FATAL_ERROR. The request was aborted due to a communication problem. The Link may be inoperable.</td></tr>
<tr><td>6h - 7h</td><td>Reserved</td></tr>
</table></td></tr>
<tr><td>07:06</td><td>RO</td><td>0</td><td>Reserved.</td></tr>
<tr><td>05</td><td>RWC</td><td>0</td><td>Device Error Indicator (DEI): If set to ‘1’ then there is a critical error in device. If cleared to ‘0’, there is not a critical error in device. This is reset when SW writes “1” to this bit. Host software should check this bit prior to setting UTRLRSR and UTMRLRSR to ‘1’.</td></tr>
<tr><td>04</td><td>RWC</td><td>0</td><td>Host Error Indicator (HEI): If set to ‘1’ then there is a critical error in host controller. If cleared to ‘0’, there is not a critical error in host controller. This is reset when SW writes “1” to this bit.</td></tr>
<tr><td>03</td><td>RO</td><td>0</td><td>UIC COMMAND Ready (UCRDY): This field indicates whether the host controller controller is ready to process UIC COMMAND. Host software shall only set the UICCMD if HCS.UCRDY is set to ‘1’.</td></tr>
<tr><td>02</td><td>RO</td><td>0</td><td>UTP Task Management Request List Ready (UTMRLRDY): This field is set to ‘1’ when the host controller controller is ready to Task Management requests. This field is cleared to ‘0’ by host controller when one of the following conditions occur:
The device presence is not detected;
UTP Task Management Request List is full;
There is an error with host controller or device.
Host software set UTMRLRSR ‘0’.
Host software shall only set the UTMRLRSR register if HCS.UTMRLRDY is set to ‘1’.</td></tr>
<tr><td>01</td><td>RO</td><td>0</td><td>UTP Transfer Request List Ready (UTRLRDY): This field indicates whether the host controller controller is ready to process UTP Transfer Request.
This field is cleared to ‘0’ by host controller when one of the following conditions occur:
The device presence is not detected;
UTP Transfer Request List is full;
There is an error with host controller or device.
Host software set UTRLRSR ‘0’.
Host software shall only set the UTRLRSR bit to ‘1’ if HCS. UTRLRDY is set to ‘1’.</td></tr>
<tr><td>00</td><td>RO</td><td>HwInit</td><td>Device Present (DP): This field is set to ‘1’ when an UFS device is attached to the controller. This field is cleared to ‘0’ when no UFS device is attached to this controller.</td></tr>
</table>

5.3.4 Offset 34h: HCE – Host Controller Enable

<table>
<tr><th>Bit</th><th>Type</th><th>Reset</th><th>Description</th></tr>
<tr><td>31:01</td><td>RO</td><td>0</td><td>Reserved</td></tr>
<tr><td>0</td><td>RW</td><td>0</td><td>Host Controller Enable (HCE): When set to ‘0’ by host software, this bit cause host controller in reset. When set ‘1’, host controller will go through sequence of initialization steps to take itself out of reset. The initialization process is completed, host controller will set the register to ‘1’. See the theory of operation section for more details on the host controller initialization.
<table>
<tr><th>Bit Value</th><th>Description</th></tr>
<tr><td>0</td><td>Disable</td></tr>
<tr><td>1</td><td>Enable</td></tr>
</table>
</td></tr>
</table>

5.3.5 Offset 38h: UECPA – Host UIC Error Code PHY Adapter Layer

<table>
<tr><th>Bit</th><th>Type</th><th>Reset</th><th>Description</th></tr>
<tr><td>31:31</td><td>ROC</td><td>0</td><td>UIC PHY AdapterA Layer Error (ERR): Indicates whether an error was generated by the PHY Adapter Layer</td></tr>
<tr><td>30:05</td><td>RO</td><td>0</td><td>Reserved</td></tr>
<tr><td>04:00</td><td>ROC</td><td>0</td><td>UIC PHY Adapter Layer Error Code (EC): error code generated when IS.UEE and UECPA.ERR are set to ‘1’.
<table>
<tr><th>Bit</th><th>Description</th></tr>
<tr><td>00</td><td>Error reported by the M-PHY layer: PHY error on Lane 0.</td></tr>
<tr><td>01</td><td>Error reported by the M-PHY layer: PHY error on Lane 1</td></tr>
<tr><td>02</td><td>Error reported by the M-PHY layer: PHY error on Lane 2</td></tr>
<tr><td>03</td><td>Error reported by the M-PHY layer: PHY error on Lane 3</td></tr>
<tr><td>04</td><td>Generic PHY Adapter Error. This should be the LINERESET indication. Categorized as "ERROR" (Not FATAL). SW is informed that M-PHY has been reset and all M-PHY Attributes (that are not handled by UniPro) need to be restored in order to keep the link optimized.</td></tr>
</table>
</td></tr>
</table>

5.3.6 Offset 3Ch: UECDL – Host UIC Error Code Data Link Layer

<table>
<tr><th>Bit</th><th>Type</th><th>Reset</th><th>Description</th></tr>
<tr><td>31:31</td><td>ROC</td><td>0</td><td>UIC Data Link Layer Error (ERR): Indicates whether an error was generated by the Data Link Layer</td></tr>
<tr><td>30:15</td><td>RO</td><td>0</td><td>Reserved</td></tr>
<tr><td>14:00</td><td>ROC</td><td>0</td><td>UIC Data Link Layer Error Code (EC): error code generated when IS.UE and UECDL.ERR are set to ‘1’. Refer to UniPro Specification Version 1.4 for the definition of the error codes.
<table>
<tr><th>Bit</th><th>Description</th></tr>
<tr><td>00</td><td>NAC_RECEIVED</td></tr>
<tr><td>01</td><td>TCx_REPLAY_TIMER_EXPIRED</td></tr>
<tr><td>02</td><td>AFCx_REQUEST_TIMER_EXPIRED</td></tr>
<tr><td>03</td><td>FCx_PROTECTION_TIMER_EXPIRED</td></tr>
<tr><td>04</td><td>CRC_ERROR</td></tr>
<tr><td>05</td><td>RX_BUFFER_OVERFLOW</td></tr>
<tr><td>06</td><td>MAX_FRAME_LENGTH_EXCEEDED</td></tr>
<tr><td>07</td><td>WRONG_SEQUENCE_NUMBER</td></tr>
<tr><td>08</td><td>AFC_FRAME_SYNTAX_ERROR</td></tr>
<tr><td>09</td><td>NAC_FRAME_SYNTAX_ERROR</td></tr>
<tr><td>10</td><td>EOF_SYNTAX_ERROR</td></tr>
<tr><td>11</td><td>FRAME_SYNTAX_ERROR</td></tr>
<tr><td>12</td><td>BAD_CTRL_SYMBOL_TYPE</td></tr>
<tr><td>13</td><td>FATAL_ERROR.</td></tr>
<tr><td>14</td><td>PA_ERROR_IND_RECEIVED</td></tr>
</table></td></tr>
</table>

5.3.7 Offset 40h: UECN – Host UIC Error Code Network Layer

<table>
<tr><th>Bit</th><th>Type</th><th>Reset</th><th>Description</th></tr>
<tr><td>31:31</td><td>ROC</td><td>0</td><td>UIC Network Layer Error (ERR): Indicates whether an error was generated by the Network Layer</td></tr>
<tr><td>30:03</td><td>RO</td><td>0</td><td>Reserved</td></tr>
<tr><td>02:00</td><td>ROC</td><td>0</td><td>UIC Network Layer Error Code (EC): error code generated when IS.UE and UECN.ERR are set to ‘1’. Refer to UniPro Specification Version 1.4 for the definition of the error codes.
<table>
<tr><th>Bit</th><th>Description</th></tr>
<tr><td>00</td><td>UNSUPPORTED_HEADER_TYPE</td></tr>
<tr><td>01</td><td>BAD_DEVICEID_ENC</td></tr>
<tr><td>02</td><td>LHDR_TRAP_PACKET_DROPPING</td></tr>
</table></td></tr>
</table>

5.3.8 Offset 44h: UECT – Host UIC Error Code Transport Layer

<table>
<tr><th>Bit</th><th>Type</th><th>Reset</th><th>Description</th></tr>
<tr><td>31:31</td><td>ROC</td><td>0</td><td>UIC Transport Layer Error (ERR): Indicates whether an error was generated by the Transport Layer</td></tr>
<tr><td>30:07</td><td>RO</td><td>0</td><td>Reserved</td></tr>
<tr><td>06:00</td><td>ROC</td><td>0</td><td>UIC Transport Layer Error Code (EC): error code generated when IS.UE and UECT.ERR are set to ‘1’. Refer to UniPro Specification Version 1.4 for the definition of the error codes.
<table>
<tr><th>Bit</th><th>Description</th></tr>
<tr><td>00</td><td>UNSUPPORTED_HEADER_TYPE</td></tr>
<tr><td>01</td><td>UNKNOWN_CPORTID</td></tr>
<tr><td>02</td><td>NO_CONNECTION_RX</td></tr>
<tr><td>03</td><td>CONTROLLED_SEGMENT_DROPPING</td></tr>
<tr><td>04</td><td>BAD_TC</td></tr>
<tr><td>05</td><td>E2E_CREDIT_OVERFLOW</td></tr>
<tr><td>06</td><td>SAFETY_VALVE_DROPPING</td></tr>
</table></td></tr>
</table>

5.3.9 Offset 48h: UECDME – Host UIC Error Code

<table>
<tr><th>Bit</th><th>Type</th><th>Reset</th><th>Description</th></tr>
<tr><td>31:31</td><td>ROC</td><td>0</td><td>UIC DME Error (ERR): Indicates whether an error was generated by the DME</td></tr>
<tr><td>30:01</td><td>RO</td><td>0</td><td>Reserved</td></tr>
<tr><td>00:00</td><td>ROC</td><td>0</td><td>UIC DME Error Code (EC): error code generated when IS.UE and UECDME.ERR are set to ‘1’.
<table>
<tr><th>Bit</th><th>Description</th></tr>
<tr><td>00</td><td>Generic DME error.</td></tr>
</table></td></tr>
</table>

5.3.10 Offset 4Ch: UTRIACR – UTP Transfer Request Interrupt Aggregation Control Register

<table>
<tr><th>Bit</th><th>Type</th><th>Reset</th><th>Description</th></tr>
<tr><td>31</td><td>RW</td><td>0</td><td>Interrupt Aggregation Enable/Disable (IAEN): When set to ‘0’ by host software, command reponses are neither counted nor timed. Interrupts are still triggerred by responses to Interrupt Commands.

When set to ‘1’, the interrupt aggregation mechanism is enabled and aggregation-based interrupts are generated
<table>
<tr><th>Bit Value</th><th>Description</th></tr>
<tr><td>0</td><td>Disable</td></tr>
<tr><td>1</td><td>Enable</td></tr>
</table></td></tr>
<tr><td>30:25</td><td>RO</td><td>0</td><td>Reserved</td></tr>
<tr><td>24</td><td>WO</td><td>0</td><td>Interrupt aggregation parameter write enable (IAPWEN): When host SW writes ‘1’, the values in IACTH and IATOVAL are updated with the contents written at the same cycle.

When host SW writes ‘0’, the values in IACTH and IATOVAL are not updated.

NOTE Write operations to IACTH and IATOVAL are only allowed when no commands are outstanding.</td></tr>
<tr><td>23:21</td><td>RO</td><td>0</td><td>Reserved</td></tr>
<tr><td>20</td><td>RO</td><td>0</td><td>Interrupt aggregation status bit (IASB):
This bit indicates to Host SW whether any responses have been received and counted towards interrupt aggregation (i.e., IASB is set iff IA counter > 0).
<table>
<tr><th>Bit Value</th><th>Description</th></tr>
<tr><td>0</td><td>No commands has been received since last counter reset (IA counter =0)</td></tr>
<tr><td>1</td><td>At least one command has been received and counted (IA counter >0)</td></tr>
</table></td></tr>
<tr><td>19:17</td><td>RO</td><td>0</td><td>Reserved</td></tr>
<tr><td>16</td><td>W</td><td>0</td><td>Counter and Timer Reset(CTR): When host SW writes ‘1’, the interrupt aggregation timer and counter are reset.
It is recommended that host software use this field to reset the timer and counter every time it services newly received UTP responses.</td></tr>
<tr><td>15:13</td><td>RO</td><td>0</td><td>Reserved</td></tr>
<tr><td>12:8</td><td>RW</td><td>0</td><td>Interrupt aggregation counter threshold (IACTH): Host SW uses this field to configure the number of responses that are required to generate an interrupt.

Counter Operation: As UTP responses are received by the host controller, they are counted. The counter is reset by software during the interrupt service routine. It increments with every response to a Regular Command received at the host controller. The counter stops counting when it reaches the value configured in IACTH, and sets the IS.UTRCS bit.

The maximum allowed value is 31

NOTE When IACTH is 0, responses are not counted, and counting-based interrupts are not generated.

In order to write to this field, the IAPWEN bit must be set at the same write operation.</td></tr>
</table>

Bit	Type	Reset	Description
7:0	RW	0	**Interrupt aggregation timeout value (IATOVAL):** Host SW uses this field to configure the maximum time allowed between a response arrival to the host controller and the generation of an interrupt. **Timer Operation:** The timer is reset by software during the interrupt service routine. It starts running when the host controller receives the first response to a Regular Command, after the timer was reset. The timer stops when it reaches the value configured in **IATOVAL** field, and **IS.UTRCS** bit is set. **NOTE** When **IATOVAL** is 0, the timer is not running, and timer-based interrupts are not generated. The Time units in this field are 40 us. Therefore, writing 0x01 represents a time-out value of 40 us, and writing 0xFF represents a time-out value of 10.2 ms

5.4 UTP Transfer Request Registers

5.4.1 Offset 50h: UTRLBA – UTP Transfer Request List Base Address

Bit	Type	Reset	Description
31:10	RW	Impl Spec	**UTP Transfer Request List Base Address (UTRLBA):** Indicates the 32-bit base physical address for the UTP Transfer Request list. This base is used when fetching commands for execution. The structure pointed to by this address range is 1KB in length. This address shall be 1KB aligned as indicated by bits 09:00 being read only.
09:00	RO	0	Reserved

5.4.2 Offset 54h: UTRLBAU – UTP Transfer Request List Base Address Upper 32-bits

Bit	Type	Reset	Description
31:00	RW	Impl Spec	**UTP Transfer Request List Base Address Upper (UTRLBAU):** Indicates the upper 32-bits for the UTP Transfer Request list base physical address. This base is used when fetching commands for execution.

5.4.3 Offset 58h: UTRLDBR – UTP Transfer Request List Door Bell Register

Bit	Type	Reset	Description
31:0	RWS	0	**UTP Transfer Request List Door bell Register(UTRLDBR):** This field is bit significant. Each bit corresponds to a slot in the UTP Transfer Request List, where bit 0 corresponds to request slot 0. A bit in this field is set to '1' by host software to indicate to the host controller that a transfer request has been built in system memory for the associated transfer request slot and may be ready for execution. The host software indicates no change to request slots by setting the associated bits in this filed to '0'. Bits in this field shall only be set '1' or '0' by host software when **UTRLRSR** is set to '1'. When a transfer request is completed (with success or error), the corresponding bit is cleared to '0' by the host controller. The host controller always process transfer requests in-order according to the order submitted to the list. In case of multiple commands with single door bell register ringing (batch mode), The dispatch order for these transfer requests by host controller will base on their index in the List. A transfer request with lower index value will be executed before a transfer request with higher index value. This field is also cleared when **UTRLRSR** is written from a '1' to a '0' by host software.

5.4.4 Offset 5Ch: UTRLCLR – UTP Transfer Request List CLear Register

Bit	Type	Reset	Description
31:0	W	0	**UTP Transfer Request List CLear Register(UTRLCLR):** This field is bit significant. Each bit corresponds to a slot in the UTP Transfer Request List, where bit 0 corresponds to request slot 0. A bit in this field is set to '0' by host software to indicate to the host controller that a transfer request slot is cleared. The host controller shall free up any resources associated to the request slot immediately, and shall set the associated bit in UTRLDBR to '0'. The host software indicates no change to request slots by setting the associated bits in this filed to '1'. Bits in this field shall only be set '1' or '0' by host software when **UTRLRSR** is set to '1'. The host software shall use this field only when a UTP Transfer Request is expected to not be completed, e.g., when a Transfer Request was aborted, or in case of a system bus error, such as an invalid UTRD.

5.4.5 Offset 60h: UTRLRSR – UTP Transfer Request List Run Stop Register

Bit	Type	Reset	Description
31:01	RO	0	Reserved
0	RW	0	**UTP Transfer Request List Run-Stop Register (UTRLRSR):** When set to '1', the host controller may process the list. Host controller starts processing the list at entry '0'. The host controller continues process the list as long as this bit is set to a '1'. When cleared to '0', the host controller shall continue to complete all the outstanding transfer requests in the list and then stop. When cleared to '0', the host controller will clear **UTRLRDY** to '0'. This bit shall only be set to '1' when **HCS.UCRDY** and **HCS.UTRLRDY** and **HCS.UTMRLRDY** is set to '1'.

5.5 UTP Task Management Registers

5.5.1 Offset 70h: UTMRLBA – UTP Task Management Request List Base Address

Bit	Type	Reset	Description
31:10	RW	Impl Spec	**UTP Task Management Request List Base Address (UTMRLBA):** Indicates the 32-bit base physical address for the list. This base is used when fetching Task Management Functions for execution. The structure pointed to by this address range is 1KB in length. This address shall be 1KB aligned as indicated by bits 09:00 being read only.
09:00	RO	0	Reserved

5.5.2 Offset 74h: UTMRLBAU – UTP Task Management Request List Base Address Upper 32-bits

Bit	Type	Reset	Description
31:00	RW	Impl Spec	**UTP Task Management Request List Base Address (UTMRLBAU):** Indicates the upper 32-bits for the list base physical address. This base is used when fetching task management functions for execution.

5.5.3 Offset 78h: UTMRLDBR – UTP Task Management Request List Door Bell Register

Bit	Type	Reset	Description
31:0	RWS	0	**UTP Task Management Request List Door bell Register(UTMRLDBR):** This field is bit significant. Each bit corresponds to a slot in the task management request List, where bit 0 corresponds to slot 0. A bit in this field is set by host software to indicate to the host controller that a task management request has been built in system memory for the associated task management request slot, and may be ready for execution. The host software indicates no change to request slots by setting the associated bits in this filed to '0'. Bits in this field shall only be set to '1' by host software when **UTMRLRSR** is set to '1'. When a task management request is completed (with success or error), the corresponding bit is cleared to '0' by the host controller. The host controller always process task management request in-order according to the order submitted to the list. In case of multiple requests with single door bell register ringing (batch mode), The dispatch order for these requests by host controller will base on their index in the List. A task management with lower index value will be executed before a task management request with higher index value. This field is also cleared when **UTMRLRSR** is written from a '1' to a '0' by host software..

5.5.4 Offset 7Ch: UTMRLCLR – UTP Task Management Request List CLear Register

Bit	Type	Reset	Description
31:0	W	0	**UTP** Task Management **List CLear Register(UTRLCLR):** This field is bit significant. Each bit corresponds to a slot in the task management request List, where bit 0 corresponds to slot 0. A bit in this field is set to '0' by host software to indicate to the host controller that a task management request slot is cleared. The host controller shall free up any resources associated to the task management request slot immediately, and shall set the associated bit in UTMRLDBR to '0'. The host software indicates no change to task management request slots by setting the associated bits in this filed to '1'. Bits in this field shall only be set '1' or '0' by host software when **UTRLRSR** is set to '1'. The host software shall use this field only when a UTP Task Management Request is expected to not be completed, e.g., in case of a system bus error, such as an invalid UTMRD.

5.5.5 Offset 80h: UTMRLRSR – UTP Task Management Request List Run Stop Register

Bit	Type	Reset	Description
31:01	RO	0	Reserved
0	RW	0	**UTP Task Management Request List Run-Stop Register (UTMRLRSR):** When set to '1', the host controller may process the list. Host controller starts processing the list at entry '0'. The host controller continues process the list as long as this bit is set to a '1'. When cleared to '0', the host controller shall continue to complete all the outstanding task management requests in the list and then stop. When cleared to '0', the host controller will clear **UTMRLRDY** to '0'. This bit shall only be set to '1' when **HCS.UTMRLRDY** is set to '1'. This bit shall only be set to '1' when **HCS.UTMRLRDY** is set to '1'.

5.6 UIC Command Registers

5.6.1 Offset 90h: UICCMD – UIC Command

<table>
<tr><th>Bit</th><th>Type</th><th>Reset</th><th>Description</th></tr>
<tr><td>31:08</td><td>RO</td><td>0</td><td>Reserved.</td></tr>
<tr><td>07:00</td><td>RW</td><td>0h</td><td>Command Opcode(CMDOP): Indicate the Opcode of a UIC Command to be dispatched to local UIC layer. When this register is set, the host host controller shall take the values of UICCMDARGx as the corresponding parameters (input and output) that are a part of the UIC Command.
<table>
<tr><th>Opcode</th><th>O/M</th><th>UIC Command</th></tr>
<tr><td colspan="3">Configuration</td></tr>
<tr><td>01h</td><td>M</td><td>DME_GET</td></tr>
<tr><td>02h</td><td>M</td><td>DME_SET</td></tr>
<tr><td>03h</td><td>M</td><td>DME_PEER_GET</td></tr>
<tr><td>04h</td><td>M</td><td>DME_PEER_SET</td></tr>
<tr><td>05h – 0Fh</td><td></td><td>Reserved</td></tr>
<tr><td colspan="3">Control</td></tr>
<tr><td>10h</td><td>O</td><td>DME_POWERON</td></tr>
<tr><td>11h</td><td>O</td><td>DME_POWEROFF</td></tr>
<tr><td>12h</td><td>M</td><td>DME_ENABLE</td></tr>
<tr><td>13h</td><td></td><td>Reserved</td></tr>
<tr><td>14h</td><td>M</td><td>DME_RESET</td></tr>
<tr><td>15h</td><td>M</td><td>DME_ENDPOINTRESET</td></tr>
<tr><td>16h</td><td>M</td><td>DME_LINKSTARTUP</td></tr>
<tr><td>17h</td><td>M</td><td>DME_HIBERNATE_ENTER</td></tr>
<tr><td>18h</td><td>M</td><td>DME_HIBERNATE_EXIT</td></tr>
<tr><td>19h</td><td></td><td>Reserved</td></tr>
<tr><td>1Ah</td><td>O</td><td>DME_TEST_MODE</td></tr>
<tr><td>1Bh - FFh</td><td></td><td>Reserved</td></tr>
</table>
O/M: O = Optional, M = Mandatory.</td></tr>
</table>

5.6.2 Offset 94h: UICCMDARG1 – UIC Command Argument 1

<table>
<tr><th>Bit</th><th>Type</th><th>Reset</th><th>Description</th></tr>
<tr><td>31:00</td><td>RW</td><td>0</td><td>
Argument 1 (ARG1): This register contains the value for 1st argument of the UIC command if applicable. The content of this field varies with the UIC Command (UICCMD).
<table>
<tr><th rowspan="2">UIC Command</th><th colspan="4">Value</th></tr>
<tr><th>Bit[31:24]</th><th>Bit[23:16]</th><th>Bit[15:08]</th><th>Bit[07:00]</th></tr>
<tr><td>DME_GET</td><td colspan="2">MIBattribute</td><td colspan="2">GenSelectorIndex</td></tr>
<tr><td>DME_SET</td><td colspan="2">MIBattribute</td><td colspan="2">GenSelectorIndex</td></tr>
<tr><td>DME_PEER_GET</td><td colspan="2">MIBattribute</td><td colspan="2">GenSelectorIndex</td></tr>
<tr><td>DME_PEER_SET</td><td colspan="2">MIBattribute</td><td colspan="2">GenSelectorIndex</td></tr>
<tr><td>DME_POWERON</td><td colspan="4">Reserved</td></tr>
<tr><td>DME_POWEROFF</td><td colspan="4">Reserved</td></tr>
<tr><td>DME_ENABLE</td><td colspan="4">Reserved</td></tr>
<tr><td>DME_RESET</td><td colspan="3">Reserved</td><td>ResetLevel</td></tr>
<tr><td>DME_ENDPOINTRESET</td><td colspan="4">Reserved</td></tr>
<tr><td>DME_LINKSTARTUP</td><td colspan="4">Reserved</td></tr>
<tr><td>DME_HIBERNATE_ENTER</td><td colspan="4">Reserved</td></tr>
<tr><td>DME_HIBERNATE_EXIT</td><td colspan="4">Reserved</td></tr>
<tr><td>DME_TEST_MODE</td><td colspan="4">Reserved</td></tr>
</table>
</td></tr>
</table>

MIBattribute: Indicates the ID of the attribute of the requested. See MIPI UniPro Specification version 1.40.00 for the details of the MIBattribute parameter.

GenSelectorIndex: Indicates the targeted M-PHY data lane or CPort or Test Feature when relevant. See MIPI UniPro Specification version 1.40.00 for the details of the GenSelectorIndex parameter.

Layer	Valid Range
L1.5	0 to 2*PA_MaxDataLanes – 1
L4 / CPort	0 to T_NumCPorts – 1
L4 / Test Feature	0 to T_NumTestFeatures – 1

ResetLevel: Indicates the reset type. See MIPI UniPro Specification version 1.40.00 for the details of the ResetLevel parameter.

Value	Definition
00h	Cold Reset
01h	Warm Reset
02h-FFh	Reserved

GenericErrorCode: Indicates the result of the UIC control command request. It is valid after host controller has set the **IS.UCCS** bit to ’1’. See MIPI UniPro Specification version 1.40.00 for the details of the GenericErrorCode parameter.

Value	Definition
0h	SUCCESS
1h	FAILURE
2h-FFh	Reserved

.

5.6.3 Offset 98h: UICCMDARG2 – UIC Command Argument 2

<table>
<tr><th>Bit</th><th>Type</th><th>Reset</th><th>Description</th></tr>
<tr><td>31:00</td><td>RW</td><td>0</td><td>Argument 2 (ARG2): This register contains the value for 2nd argument of the UIC command if applicable. The content of this field vary with UIC Command.
<table>
<tr><th rowspan="2">UIC Command</th><th colspan="4">Value</th></tr>
<tr><th>Bit[31:24]</th><th>Bit[23:16]</th><th>Bit[15:08]</th><th>Bit[07:00]</th></tr>
<tr><td>DME_GET</td><td>Reserved</td><td>Reserved</td><td>Reserved</td><td>ConfigResultCode</td></tr>
<tr><td>DME_SET</td><td>Reserved</td><td>AttrSetType</td><td>Reserved</td><td>ConfigResultCode</td></tr>
<tr><td>DME_PEER_GET</td><td>Reserved</td><td>Reserved</td><td>Reserved</td><td>ConfigResultCode</td></tr>
<tr><td>DME_PEER_SET</td><td>Reserved</td><td>AttrSetType</td><td>Reserved</td><td>ConfigResultCode</td></tr>
<tr><td>DME_POWERON</td><td colspan="3">Reserved</td><td>GenericErrorCode</td></tr>
<tr><td>DME_POWEROFF</td><td colspan="3">Reserved</td><td>GenericErrorCode</td></tr>
<tr><td>DME_ENABLE</td><td colspan="3">Reserved</td><td>GenericErrorCode</td></tr>
<tr><td>DME_RESET</td><td colspan="4">Reserved</td></tr>
<tr><td>DME_ENDPOINTRESET</td><td colspan="3">Reserved</td><td>GenericErrorCode</td></tr>
<tr><td>DME_LINKSTARTUP</td><td colspan="3">Reserved</td><td>GenericErrorCode</td></tr>
<tr><td>DME_HIBERNATE_ENTER</td><td colspan="3">Reserved</td><td>GenericErrorCode</td></tr>
<tr><td>DME_HIBERNATE_EXIT</td><td colspan="3">Reserved</td><td>GenericErrorCode</td></tr>
<tr><td>DME_TEST_MODE</td><td colspan="3">Reserved</td><td>GenericErrorCode</td></tr>
</table></td></tr>
</table>

AttrSetType: Indicates whether the attribute value (AttrSet = NORMAL) or the attribute non-volatile reset value (STATIC) setting is requested. See MIPI UniPro Specification version 1.40.00 for the details of the AttrSetType parameter.

ConfigResultCode: Indicates the result of the UIC configuration command request. It is valid after host controller has set the **IS.UCCS** bit to ’1’. See MIPI UniPro Specification version 1.40.00 for the details of the ConfigResultCode parameter.

Value	Definition
00h	SUCCESS
01h	INVALID_MIB_ATTRIBUTE
02h	INVALID_MIB_ATTRIBUTE_VALUE
03h	READ_ONLY_MIB_ATTRIBUTE
04h	WRITE_ONLY_MIB_ATTRIBUTE
05h	BAD_INDEX
06h	LOCKED_MIB_ATTRIBUTE
07h	BAD_TEST_FEATURE_INDEX
08h	PEER_COMMUNICATION_FAILURE
09h	BUSY
0Ah	DME_FAILURE
0Bh-FFh	Reserved

GenericErrorCode: Indicates the result of the UIC control command request. It is valid after host controller has set the **IS.UCCS** bit to ’1’. See MIPI UniPro Specification version 1.40.00 for the details of the GenericErrorCode parameter.

Value	Definition
00h	SUCCESS
01h	FAILURE
02h-FFh	Reserved

5.6.4 Offset 9Ch: UICCMDARG3 – UIC Command Argument 3

<table>
<tr><th>Bit</th><th>Type</th><th>Reset</th><th>Description</th></tr>
<tr><td>31:00</td><td>RW</td><td>0</td><td>Argument 3 (ARG3): This register contains the value for 3rd argument of the UIC command if applicable. The content of this field vary with UIC Command.
<table>
<tr><th rowspan="2">UIC Command</th><th colspan="4">Value</th></tr>
<tr><th>Bit[31:24]</th><th>Bit[23:16]</th><th>Bit[15:08]</th><th>Bit[07:00]</th></tr>
<tr><td>DME_GET</td><td colspan="4">MIBvalue_R</td></tr>
<tr><td>DME_SET</td><td colspan="4">MIBvalue_W</td></tr>
<tr><td>DME_PEER_GET</td><td colspan="4">MIBvalue_R</td></tr>
<tr><td>DME_PEER_SET</td><td colspan="4">MIBvalue_W</td></tr>
<tr><td>DME_POWERON</td><td colspan="4">Reserved</td></tr>
<tr><td>DME_POWEROFF</td><td colspan="4">Reserved</td></tr>
<tr><td>DME_ENABLE</td><td colspan="4">Reserved</td></tr>
<tr><td>DME_RESET</td><td colspan="4">Reserved</td></tr>
<tr><td>DME_ENDPOINTRESET</td><td colspan="4">Reserved</td></tr>
<tr><td>DME_LINKSTARTUP</td><td colspan="4">Reserved</td></tr>
<tr><td>DME_HIBERNATE_ENTER</td><td colspan="4">Reserved</td></tr>
<tr><td>DME_HIBERNATE_EXIT</td><td colspan="4">Reserved</td></tr>
<tr><td>DME_TEST_MODE</td><td colspan="4">Reserved</td></tr>
</table></td></tr>
</table>

MIBvalue_R: Indicates the value of the attribute as returned by the UIC command returned. It is valid after host controller has set the **IS.UCCS** bit to ’1’. See MIPI UniPro Specification version 1.40.00 for the details of the MIBvalue parameter.

MIBvalue_W: Indicates the value of the attribute to be set. See MIPI UniPro Specification version 1.40.00 for details of the MIBvalue parameter.

5.6.5 UFS-Specific Attributes for Local L2 Timers

This section defines UFS-specific DME attributes necessary for performing the UIC power mode change using the DME_SET primitives only. These attributes are accessible via DME_SET command as any other UIC-defined attributes. In this section, these attributes are collectively named DME LocalL2TimerData attributes.

The DME LocalL2TimerData and PA_PWRModeUserData[0:11] attributes and their behavior are equivalent to the LocalL2TimerData and RemoteL2TimerData parameters in the DME_POWERMODE.req primitive, and define the local and remote Layer 2 timer values for the next UIC power mode, respectively. Neither the DME LocalL2TimerData attributes, nor the PA_PWRModeUserData[0:11] attributes have an immediate effect on the UIC stack. They are solely used to store the local and remote L2 timer values for the next UIC power mode. These values are used by UIC during the UIC power mode change triggered by setting the PA_PWRMode attribute, and are only committed to UIC if the UIC power mode change is successful. If the UIC power mode change fails, the previous L2 timer values are still used.

As a result, using the DME LocalL2TimerData, PA_PWRModeUserData and PA_PWRMode attributes for changing the UIC power mode has an identical behavior to changing the UIC power mode using the DME_POWERMODE.req primitive. These attributes are in fact a way to implement the DME_POWERMODE.req primitive.

5.6.5.1 Attribute 0xD041: DME_LocalL2TimerData_FC0ProtectionTimeOutVal

Bit	Type	Reset	Description
31:10	RO	0	Reserved
09:00	RW	511	**DME_LocalL2TimerData_FC0ProtectionTimeOutVal:** Stores the TC0 FC Protection timeout value (10 bit) for the next UIC power mode. The value has no immediate effect on UIC, and is applied to the local UIC stack with the next successful UIC power mode change. The software shall not modify this attribute between triggering the UIC power mode change by setting PA_PWRMode and until IS.UPMS is set to ‘1 to indicate the UIC power mode change has completed.

5.6.5.2 Attribute 0xD042: DME_LocalL2TimerData_TC0ReplayTimeOutVal

Bit	Type	Reset	Description
31:12	RO	0	Reserved
11:00	RW	4095	**DME_LocalL2TimerData_TC0ReplayTimeOutVal:** Stores the TC0 Replay timeout value (12 bit) for the next UIC power mode. The value has no immediate effect on UIC, and is applied to the local UIC stack with the next successful UIC power mode change. The software shall not modify this attribute between triggering the UIC power mode change by setting PA_PWRMode and until IS.UPMS is set to ‘1 to indicate the UIC power mode change has completed.

5.6.5.3 Attribute 0xD043: DME_ LocalL2TimerData_AFC0ReqTimeOutVal

Bit	Type	Reset	Description
31:12	RO	0	Reserved
11:00	RW	2047	**DME_LocalL2TimerData_AFC0ReqTimeOutVal:** Stores the TC0 AFC Request timeout value (12 bit) for the next UIC power mode. The value has no immediate effect on UIC, and is applied to the local UIC stack with the next successful UIC power mode change. The software shall not modify this attribute between triggering the UIC power mode change by setting PA_PWRMode and until IS.UPMS is set to ‘1 to indicate the UIC power mode change has completed.

5.6.5.4 Attribute 0xD044: DME_ LocalL2TimerData_FC1ProtectionTimeOutVal

Bit	Type	Reset	Description
31:10	RO	0	Reserved
09:00	RW	511	**DME_LocalL2TimerData_FC1ProtectionTimeOutVal:** Stores the TC1 FC Protection timeout value (10 bit) for the next UIC power mode. The value has no immediate effect on UIC, and is applied to the local UIC stack with the next successful UIC power mode change. The software shall not modify this attribute between triggering the UIC power mode change by setting PA_PWRMode and until IS.UPMS is set to ‘1 to indicate the UIC power mode change has completed.

5.6.5.5 Attribute 0xD045: DME_ LocalL2TimerData_TC1ReplayTimeOutVal

Bit	Type	Reset	Description
31:12	RO	0	Reserved
11:00	RW	4095	**DME_LocalL2TimerData_TC1ReplayTimeOutVal:** Stores the TC1 Replay timeout value (12 bit) for the next UIC power mode. The value has no immediate effect on UIC, and is applied to the local UIC stack with the next successful UIC power mode change. The software shall not modify this attribute between triggering the UIC power mode change by setting PA_PWRMode and until IS.UPMS is set to ‘1 to indicate the UIC power mode change has completed.

5.6.5.6 Attribute 0xD046: DME_ LocalL2TimerData_AFC1ReqTimeOutVal

Bit	Type	Reset	Description
31:12	RO	0	Reserved
11:00	RW	2047	**DME_LocalL2TimerData_AFC1ReqTimeOutVal:** Stores the TC1 AFC Request timeout value (12 bit) for the next UIC power mode. The value has no immediate effect on UIC, and is applied to the local UIC stack with the next successful UIC power mode change. The software shall not modify this attribute between triggering the UIC power mode change by setting PA_PWRMode and until IS.UPMS is set to ‘1 to indicate the UIC power mode change has completed.

5.7 Vendor Specific Registers

5.7.1 Offset F0h to FFh: VS – Vendor Specific

This block of registers is reserved for vendor specific.

6 Data structures

Most communications between host software and the UFS subsystems are via system memory descriptors. These descriptors describe commands to be executed, and data transfer operations that are part of those commands. This section defines these descriptors. The data structure definitions in this section support a 32-bit or 64-bit memory buffer address space. They are all in little endian format except as noted.

6.2 UTP Transfer Request List

The interface consists of UFS Transfer Request Descriptors that are managed in a List. The list is an array that consists of up to 32 UFS Transfer Request Descriptors (UTRD). The base of the List structure is pointed by a 64-bit pointer specified in the **UTRLBA/ UTRLBAU** registers. Except UTP Task Management, all UTP command types (SCSI/UFS commands and Device Management) utilize the same UTRD structure.

6.2.1 UTP Transfer Request Descriptor

This section defines Transfer Request Descriptor for UTP commands. The data structure support s a 32-bit or 64-bit memory buffer address space.

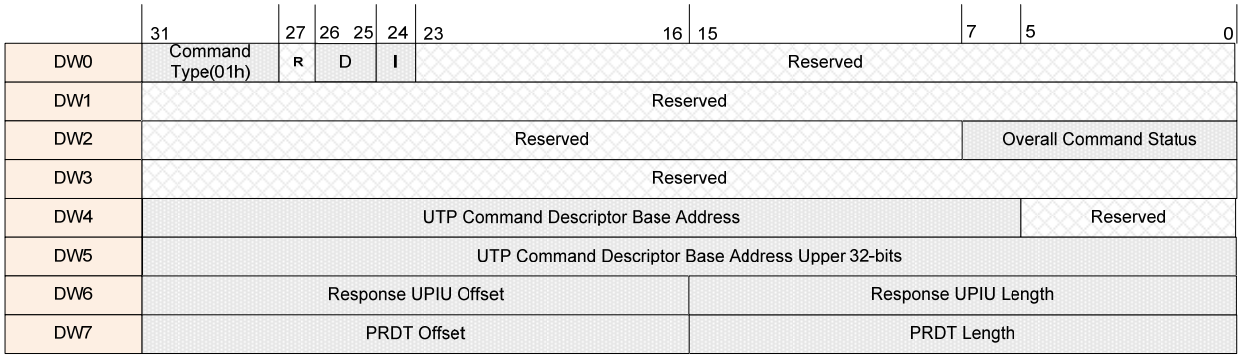


Figure 4 — UTP Transfer Request Descriptor

The following section provides the description of the data structure.

<table>
<tr><th>Bit</th><th>Description</th></tr>
<tr><td>31:28</td><td>Command Type (CT): CT=01h. Type of the command to be transferred.
<table>
<tr><th>Bits</th><th>Definition</th></tr>
<tr><td>0h</td><td>SCSI command</td></tr>
<tr><td>1h</td><td>Native UFS Command</td></tr>
<tr><td>2h</td><td>Device Management function</td></tr>
<tr><td>3h-Fh</td><td>Reserved</td></tr>
</table></td></tr>
<tr><td>27</td><td>Reserved</td></tr>
<tr><td>26:25</td><td>Data Direction (DD): This field indicates the direction of a data transfer as part of this command. When set to ‘01b’, indicates that the transfer is from system memory to the target device. The PRDT is used as the memory block descriptions for the source. When set to ‘10b’, indicates that the transfer is from the target device to system memory. The PRDT is used the memory block descriptions for the destination. This field is cleared to ‘00b’ when there is no data transfer.
<table>
<tr><th>Bits</th><th>Definition</th></tr>
<tr><td>00b</td><td>No data transfer. PRD shall have 0 entry.</td></tr>
<tr><td>01b</td><td>From system memory to target device</td></tr>
<tr><td>10b</td><td>Frome target device to system memory</td></tr>
<tr><td>11b</td><td>Reserved</td></tr>
</table></td></tr>
<tr><td>24</td><td>Interrupt (I): This field indicates the type of command with regard to interrupt generation.
<table>
<tr><th>Bit</th><th>Definition</th><th>Description</th></tr>
<tr><td>0b</td><td>Regular Command</td><td>Hardware shall count the completion of this command towards interrupt aggregation. Impact on IS.UTRCS is described in Interrupt Aggregation section.</td></tr>
<tr><td>1b</td><td>Interrupt Command</td><td>Hardware shall set IS.UTRCS to ‘1’ on completion of this command. The completion is not counted towards interrupt aggregation.</td></tr>
</table></td></tr>
<tr><td>23:00</td><td>Reserved</td></tr>
</table>

6.2.1 UTP Transfer Request Descriptor (cont'd)

<table>
<tr><th>Bit</th><th>Reset</th><th>Description</th></tr>
<tr><td>31:16</td><td>0</td><td>Reserved</td></tr>
<tr><td>07:00</td><td>Fh</td><td>Overall Command Status (OCS): Contains the command status of the associated command. The command status field is valid after host controller has cleared the corresponding UTRLDBR bit to zero.
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>0h</td><td>SUCCESS</td></tr>
<tr><td>1h</td><td>INVALID_COMMAND_TABLE_ATTRIBUTES</td></tr>
<tr><td>2h</td><td>INVALID_PRDT_ATTRIBUTES</td></tr>
<tr><td>3h</td><td>MISMATCH_DATA_BUFFER_SIZE</td></tr>
<tr><td>4h</td><td>MISMATCH_RESPONSE_UPIU_SIZE</td></tr>
<tr><td>5h</td><td>PEER_COMMUNICATION_FAILURE</td></tr>
<tr><td>6h</td><td>ABORTED</td></tr>
<tr><td>7h</td><td>FATAL ERROR</td></tr>
<tr><td>8h-Eh</td><td>Reserved</td></tr>
<tr><td>Fh</td><td>INVALID_OCS_VALUE</td></tr>
</table></td></tr>
</table>

Bit	Description
31:07	**UTP Command Descriptor Base Address (UCDBA):** Indicates the lower 32-bits of the physical address of the command descriptor, which contains the Command, Status, and PRD Table. This address shall be aligned to a 128-byte address, indicated by bits 06:00 being reserved.
06:00	Reserved

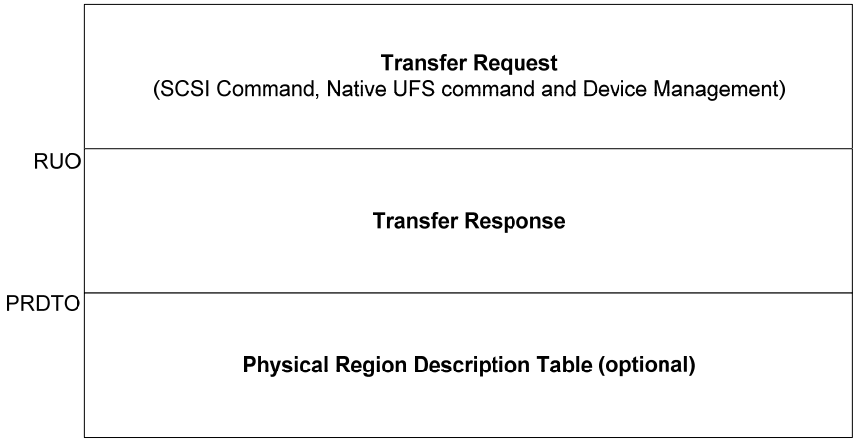
Bit	Description
31:00	**UTP Command Descriptor Base Address Upper 32-bits (UCDBAU):** This is the upper 32-bits of the Command Descriptor Base.

Bit	Description
31:16	**Response UPIU Offset (RUO):** This field contains the Dword offset of the Response UPIU within the Command Descriptor. The Response UPIU may be located at a 64-bit aligned boundary right after Command UPIU in the Command Descriptor (Bit 16 is always 0).
15:00	**Response UPIU Length (RUL): T**his field contains the length of the Response UPIU in Dword. The use and format of the Response UPIU depends on command type for this command.

Bit	Description
31:16	**PRDT Offset (PRDTO):** This field contains the Dword offset for the Physical Region Description Table within the Command Descriptor. The Physical Region Description Table may be located at a 64-bit aligned boundary right after Response UPIU in the Command Descriptor (Bit 16 is always 0). If this field is valid only when field **PRDTL** > 0.
15:00	**PRDT Length (PRDTL):** This field contains the count of the entries in PRDT. A '0' means that PRDT is empty. If this field is '0', then no data transfer shall occur with the command. The use and format of the PRDT depends on command type for this command. For non-data transfer requests and Device Management function, this field must set to '0'. Only the commands that require data transfer operation could have non-zero value. The rule is that if a command requires at least one Data-in UPIU or Data-Out UPIU in its sequence, then non-empty PRDT is required.

6.2.2 UTP Command Descriptor

UTRD contains a pointer for a data structure called UTP Command Descriptor (UCD). The data structure consists of the UPIU for the command, the offset and length of the Sense data buffer associated with the command, the offset and length for PRDT (scatter-gather list that is a part of the command). The format of the UTRD is defined as in 6.2.1.



NOTE Both Command UPIU (Transfer Request) and Response UPIU (Transfer Response) are in big endian format and PRDT is in little endian format.

Figure 5 — UTP Command Descriptor (UCD)

The Transfer Request region in the UCD provides the description of the requested command: NOP Out, Command, or Query Request. Given that phase collapse is not supported, the Command UPIU's Data Segment Length field shall be 0. The NOP UPIU and Query Request UPIU do not have restrictions.

The Transfer Response region in the UCD will store the content of the incoming UPIU that completes the Transfer Request: NOP In, Response, or Query Response. Since phase collapse feature will be removed from JESD220, UFSHCI will not support that feature. Consequently, receiving a Data in UPIU with Status phase collapse from a UFS 1.0 device will result in an unspecified behavior.

PRD Table supports scatter/gather operations for the command. PRDT is required only for a subset of SCSI commands that requires data transfer operation. UFSHCI supports three command types: SCSI, Native UFS Commands, and Device Management Functions. Refer to JESD220 for the definition of UTP Command UPIU and UTP Response UPIU.

6.2.2 UTP Command Descriptor (cont'd)

To provide description of data buffers that are associated with a UTP Transfer Request, this standard provides a data structure called Physical Region Description Table. For the UTP Transfer Request that does not require data transfer operation, this table is empty.

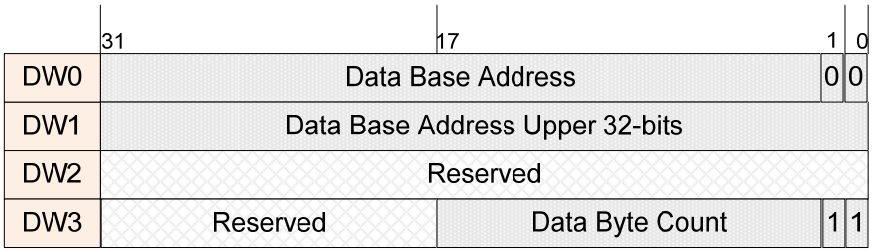


Figure 6 — Data structure for Physical Region Description Table

A breakdown of each field in a PRD table is shown below.

Bit	Description
31:02	**Data Base Address (DBA):** Indicates the lower 32-bits of the physical address of the data block. The block shall be Dword aligned, indicated by bits 01:00 being reserved.
01:00	Reserved

Bit	Description
31:00	**Data Base Address Upper 32-bits (DBAU):** This is the upper 32-bits of the data block physical address.

Bit	Description
31:00	Reserved

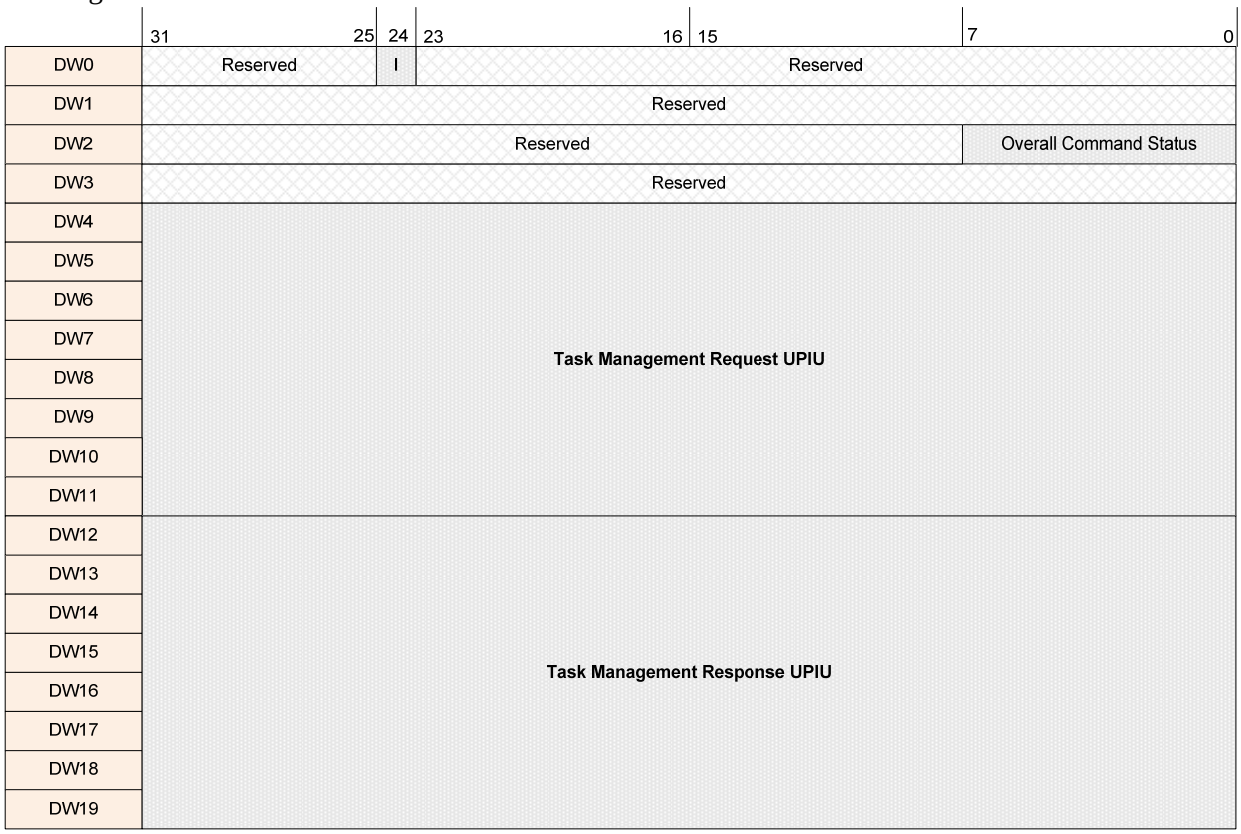
Bit	Description
31:18	Reserved
17:00	**Data Byte Count (DBC):** A '0' based value that indicates the length, in bytes, of the data block. A maximum of length of 256KB may exist for any entry. Bits 1:0 of this field shall be 11b to indicate Dword granularity. A value of '3' indicates 4 bytes, '7' indicates 8 bytes, etc.

6.3 UTP Task Management Request List

The list consists of a list of UTP Task Management Request Descriptors. The Maximum of the list size is 8. Host software is responsible to process the Task Management Response UPIU directly.

6.3.1 UTP Task Management Request Descriptor

Host controller acts as a pass through for UTP Task Management. The format of the UTMRD is defined as in Figure 6.



NOTE Both Task Management Request UPIU and Task Management Response UPIU are in big endian format.

Figure 7 — UTP Task Management Request Descriptor.

The following section provides the description of the 1st DWORD (DW0) of the data structure.

Bit	Reset	Description
31:25	0	Reserved.
24	Impl Spec	**Interrupt (I):** When set to ‘1’, hardware shall set **IS.UTMRCS** to ‘1’ on completion of this command. When cleared to ‘0’, hardware shall not set **IS.UTMRCS** to ‘1’ on completion of this command.
23:00	0	Reserved

6.3.1 UTP Task Management Request Descriptor (cont'd)

The following section provides the description of the 3rd DWORD (DW2) of the data structure.

<table>
<tr><th>Bit</th><th>Reset</th><th>Description</th></tr>
<tr><td>31:08</td><td>0</td><td>Reserved.</td></tr>
<tr><td>07:00</td><td>Fh</td><td>Overall Command Status (OCS): Contains the status of the Task Management Request. The status field is valid after host controller has cleared the corresponding UTMRLDBR bit to zero.
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>0h</td><td>SUCCESS</td></tr>
<tr><td>1h</td><td>INVALID_TASK MANAGEMENT FUNCTION_ATTRIBUTES</td></tr>
<tr><td>2h</td><td>MISMATCH_TASK_MANAGEMENT_REQUEST_SIZE</td></tr>
<tr><td>3h</td><td>MISMATCH_TASK_MANAGEMENT_RESPONSE_SIZE</td></tr>
<tr><td>4h</td><td>PEER_COMMUNICATION_FAILURE</td></tr>
<tr><td>5h</td><td>ABORTED</td></tr>
<tr><td>6h</td><td>FATAL ERROR</td></tr>
<tr><td>7h-Eh</td><td>Reserved</td></tr>
<tr><td>Fh</td><td>INVALID_OCS_VALUE</td></tr>
</table></td></tr>
</table>

Both Task Management Request UPIU and Task Management Response UPIU are 36-Byte in length. Refer to JESD220 for the definition of Task Management Request UPIU and UTP Task Management Response UPIU.

7 Theory of Operation

While the register interface provides the concise description of the software interface to UFSHCI, the implementation and interpretation of these various bits is not always clear from the definition of the bit(s). This section is a supplement to the register definitions, and provides narrative and interpretation guidance for the behaviors of the bits. The software operation of the UFS HCI is divided into three categories: Host Controller Configuration and Control, Data Transfer Operation, and Task Management. These three categories are discussion in detail in the following sections.

7.2 Host Controller Configuration and Control

7.2.1 Host Controller Initialization

When the host controller comes out of power up reset, all MMIO registers will be in their power-on default state, and the link will be inactive. Following is a sequence of the operations that host software would perform to initialize the host controller:

1) The first step in starting the controller is properly programming system bus interface. Because this operation is specific to the system bus used by the controller implementation, the documentation for the specific system bus should be followed. At the conclusion of this programming, the controller should be ready to transfer data on the system bus.
2) Write a 1 to the **HCE** register in order to enable the host controller. This triggers an autonomous basic initialization of the local UIC layer. The initialization sequence shall consist of a DME_RESET and a DME_ENABLE command. Further commands, such as DME_SET commands may be added, depending on the implementation needs. During the basic initialization sequence, the **HCE** is read as 0.
3) Wait until **HCE** is read as '1' before continuing. This indicates that the basic initialization sequence is completed.
4) Additional commands, such as DME_SET commands may be sent from the system host to the UFS host controller to provide configuration flexibility.
5) Optionally set **IE.UCCE** to 1 in order to enable the **IS.UCCS** interrupt
6) Sent DME_LINKSTARTUP command to start the link startup procedure.
7) Completion of the DME_LINKSTARTUP command sets the **IS.UCCS** bit and may flag an interrupt to the system host if the **IE.UCCE** is set. This interrupt will be flagged independently from the *GenericErrorCode*.
8) In case the *GenericErrorCode* of the completed DME_LINKSTARTUP command is SUCCESS, the **HCS.DP** is set in addition to the **IS.UCCS** bit .
9) Check value of **HCS.DP** and make sure that there is a device attached to the Link. If presence of a device is detected, go to step 10; otherwise, resend the DME_LINKSTARTUP command after **IS.ULLS** has been set to 1 (Go to step 6). **IS.ULLS** equal 1 indicates that the UFS Device is ready for a link startup.
10) Enable additional interrupts by programming the **IE** register.
11) Initialize the Interrupt Aggregation Control Register (**UTRIACR**) with the desired values for the threshold (IACTH) and timeout (IATOVAL).
    For example, write value 0x81010664 to initialize with the following parameters: bits 31 (enable), 24 (IAPWEN), and 16 (timer/counter reset) are set, IACTH =6, IATOVAL =0x64 (=4.0 ms).
    NOTE UTRIACR initialization may be executed at any time when the Run/Stop register (**UTRLRSR**) is not enabled or when no requests are outstanding.
12) Complete the host controller configuration via UIC command interface if required.
13) Allocate and initialize UTP Task Management Request List.
14) Program the *UTP Task Management Request List Base Address* and *UTP Task Management Request List Base Address* with a 64-bit address pointed to the starting address of the *UTP Task Management Request List* created at the step 7.
15) Allocate and initialize UTP Transfer Request List.
16) Program the *UTP Transfer Request List Base Address* and *UTP Transfer Request List Base Address* with a 64-bit address pointed to the starting address of the *UTP Transfer Request List* created at the step 9.
17) Enable the *UTP Task Management Request List* by setting the *UTP Task Management Request List Run-Stop Register (***UTMRLRSR***)* to '1'. This operation allows the host controller to begin accepting UTP Task Management Request via the UTP Task Management Request Door Bell mechanism.
18) Enable the *UTP Transfer Request List* by setting the *UTP Transfer Request List Run-Stop Register* (**UTRLRSR**) to '1'. This operation allows the host controller to begin accepting UTP Transfer Request via the UTP Transfer Request Door Bell mechanism.

7.2.1 Host Controller Initialization (cont'd)

At this point, the host controller is up and running.

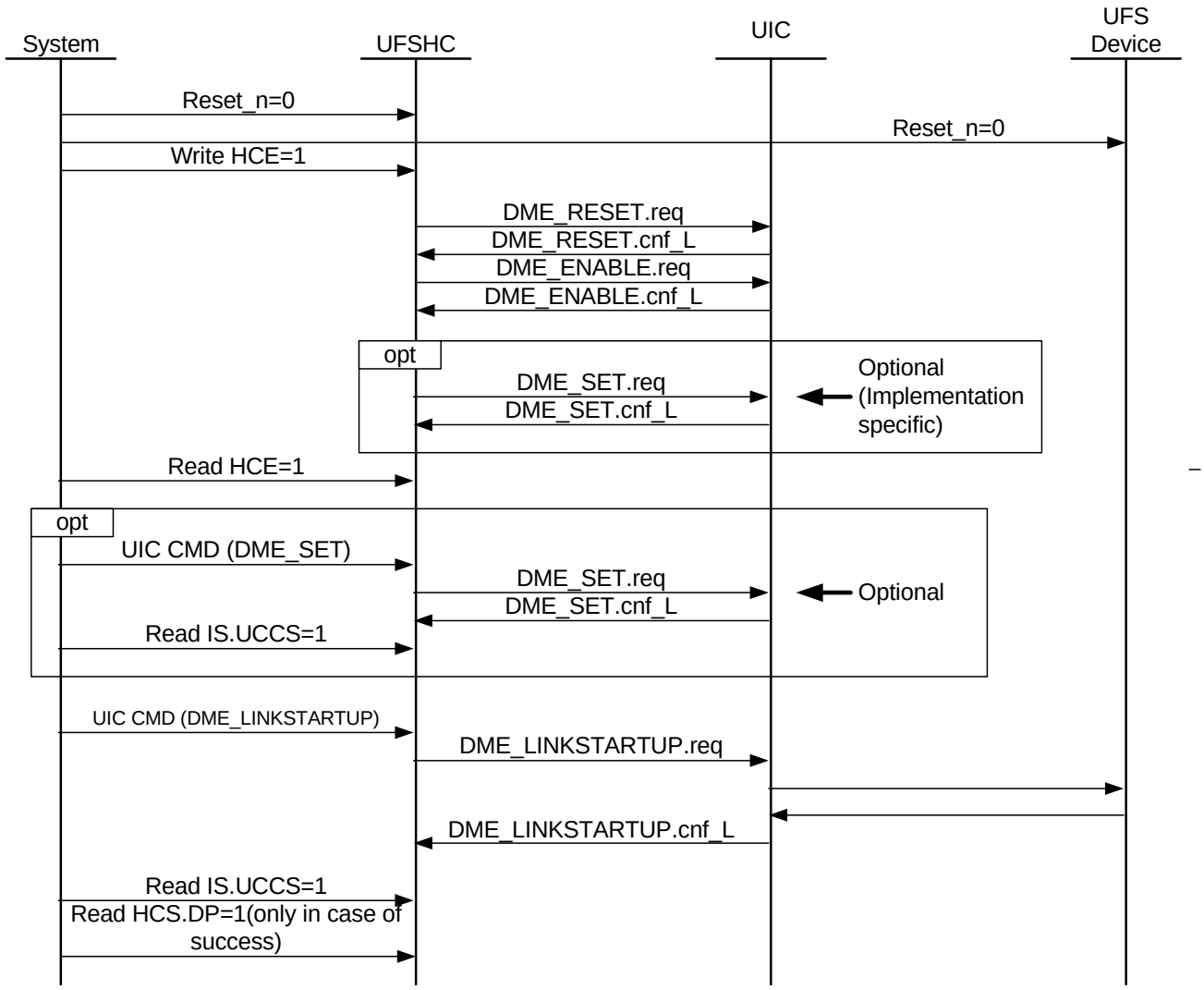


Figure 8 — Host controller link startup sequence

7.2.2 Host controller configuration and control

Once the host controller is out of reset, host software may use UIC Command Registers to configure and control the link and the attached device. Host software is responsible to configuring the UFS Interconnect stack and the attached device. When programming UIC command registers, host software shall set the register **UICCMD** only after all the UIC command argument registers (**UICCMDARG1**, **UICCMDARG2** and **UICCMDARG3**) are set. There are two options to check the status of the completion of execution of a UIC command:

- Via interrupt mechanism. Before a UIC command is sent to the host controller for execution, software enables UIC command completion interrupt by setting *UIC COMMAND Completion Enable* register **IE.UCCE**. Once the command execution is completed, the host shall generate an interrupt and set the register to *UIC COMMAND Completion Status* register '1'.
- Software Pulling. After a UIC command is sent to the host controller for execution, software keeps pulling the register *UIC COMMAND Completion Status* until it is returned '1'.

Once the command is completed, software can check the return/status code if applicable to the command.

The description of the attributes associated with host controller configuration and control are provided in MIPI UniPro specification. Referred to the MIPI UniPro specification for the definition of the attributes.

7.3 Data Transfer Operation

After the host controller reset and configuration is completed, software can utilizes the *UTP Transfer Request List* (UTRL) to pass UTP commands to the UFS device connected to the link. The UTRL is a list buffer located in system memory that is used to pass commands from software to the device. Software is responsible for choosing a UTRL size based on the value of **CAP.NUTRS.** In general, the software should choose the 32 entries option unless the system capabilities dictate a smaller memory footprint.

7.3.1 Basic Steps when Building a UTP Transfer Request

When host software needs to send a UTP command to host controller, it utilizes *UTP Transfer Request List*. The following is the steps for host software to build a UTP Transfer Request.

1. Find an empty transfer request slot by reading the **UTRLDBR**. An empty transfer request slot has its respective bit cleared to '0' in the **UTRLDBR**.
2. Host software builds a UTRD at the empty slot.
    1. Program field **CT** (command type) to indicate the command type: SCSI, native UFS command or device management function.
    2. Program field **DD** (data direction) that contains the direction of the data operations that are a part of the command if any.
    3. **I** (interrupt) bit set if software requests to mark the command as an Interrupt Command (**IS.UTRCS** to be set on command completion). The bit is cleared if software requests to mark the command as a Regular Command.
    4. Initialize **OCS** with 'Fh'.
    5. Allocate and initialize a UCD.
    6. Program filed *Command UPIU* in UCD with a UTP command excluding task management function.
    7. Initialize field *Response UPIU* in UCD with '0'.
    8. Fill PRD table with the pointers and sizes for all the data buffers associated with the data transfer of the command if required.

7.3.1 Basic Steps when Building a UTP Transfer Request (cont'd)

3. Program field **UCDBA** and **UCDBAU** with the starting address of UCD.
4. Program field **RUO** with the offset (from the starting address of UCD) of Response UPIU within UCD.
5. Program field **RUL** with the length of *Response UPIU*.
6. Program field **PRDTO** with the offset (from the starting address of UCD) of PRDT within UCD if required.
7. Program field **PRDTL** with the length of PRDT if required.Repeat the step 1 to step 7 for each command to be sent to host controller for execution.
9. Check register **UTRLRSR** and make sure it is read '1' before continuing.
10. Set *UTP Transfer Request Interrupt Aggregation Control Register* (UTRIACR) enable bit to '1' to enable the interrupt.
11. Set *Counter and Timer Reset*(CTR) bit to '1' to reset the counter and timer associated with the interrupt.
12. Program field *Interrupt aggregation counter threshold* (IACTH) with the number of command completions that are required to generate an interrupt.
13. Program field *Interrupt aggregation timeout value* (IATOVAL) with the maximum time allowed between a response arrival to the host controller and the generation of an interrupt.
14. Set **UTRLDBR** to ring the doorbell register to indicate to the host controller that one or more transfer requests are ready to be sent to the attached device. Host software shall only write a '1' to the bit position that corresponding to the new command; All other bit positions within **UTRLDBR** should be written with a '0', which indicates no change to their current values.

7.3.2 UPIU Processing

7.3.2.1 Outbound UPIUs generated by Software

Except for the DATA OUT UPIU all other UFS defined outbound UPIUs have to be composed and stored in the Host's memory by software. The Host controller then uses DMA to fetch the outbound UPIUs from the memory and dispatches them to the UFS Device using the local UniPro stack. Only the LUN and the Task Tag fields of outbound UPIUs will be analyzed by the UTP Engine in order to enable matching future corresponding inbound UPIUs. The specific UTP Engine behavior is detailed in Table 1.

Table 1 — Outbound UPIUs generated by software

UPIU TYpe	Relevant UPIU Fields	UTP Engine actions on UPIU fields
Command	Transaction Type, LUN, Task Tag	To match future incoming Response
Task Management Request		To match future incoming TM Response
Query Request		To match future incoming Query Response
NOP Out		To match future incoming NOP In

7.3.2.2 Outbound UPIU generated by Host Controller/UTP Engine

Only Data Out UPIUs are automatically generated by the UTP Engine without any involvement of software. A Data Out UPIU is created in response to incoming Ready To Transfer (RTT) UPIU from the Device which in turn resulted from a prior transmission of Command UPIU towards the Device. The UTP engine uses information from the corresponding RTT UPIU and from the PRD Table associated with the original Command UPIU. The specific UTP Engine behavior is detailed in Table 2.

Table 2 — Outbound UPIU generated by UTP Engine

UPIU TYpe	Relevant UPIU Fields	UTP Engine actions on UPIU fields
Data Out	Transaction Type	Matched against xx00 0010b
	LUN, Task Tag	Filled with LUN and Task Tag information from the corresponding to Ready To Transfer UPIU.
	Total EHS Length	Always set to ‘0’
	Data Segment Length	Same value as Data Transfer Count (see below)
	Data Buffer Offset, Data Transfer Count	Filled with Data Segment Offset and Data Buffer Count information from the corresponding to Ready To Transfer UPIU
	Data Segment	Fetched from PRDT buffers based on the DMA context information supplied in the corresponding RTT UPIU

7.3.2.3 Inbound UPIUs interpreted by Software

The UTP Engine shall analyze the LUN and TAG fields of all UPIUs the UFS Host receives so that they can be matched to UPIUs previously sent from the UFS Host towards the UFS Device (Request/Response matching).

UPIUs received from the UFS Device with LUN and TAG fields matching those of a UPIU previously sent by the UFS Host will be transferred into the respective Descriptor in Host memory. Only Ready To Transfer (RTT) and Data In UPIUs are handled differently and shall be analyzed further by the UTP Engine to enable autonomous data transfer operations as described in 7.2.2.4.

For the Host Controller the reception of an inbound UPIU of the types listed in

Table 3 will finally close a UTP transaction that was started when the corresponding outbound UPIU was sent. Ultimately, it results in clearing the corresponding bit in the Door Bell register UTRLDBR for NOP, Transfer Request or Query transactions and UTMRLDBR for Task Management transactions. Further analysis of the content of the listed inbound UPIUs will be only done by software and is transparent to the Host Controller.

Table 3 — Inbound UPIUs consumed by software

UPIU TYpe	Relevant UPIU Fields	UTP Engine actions on UPIU fields
Response	Transaction Type, LUN, Task Tag	Identify corresponding Command
Task Management Response		Identify corresponding TM Request
Query Response		Identify corresponding Query Request
NOP In		Identify corresponding NOP Out

7.3.2.4 Inbound UPIUs interpreted by Host Controller/UTP Engine

The Data In and Ready To Transfer UPIUs are handled entirely by the Host Controller/UTP Engine and software is not involved when processing them. Data In UPIUs carry data retrieved from the UFS Device and their header information is parsed to allow the Host Controller to transfer the contained data to the correct location in Host Memory.

Table 4 — Inbound Data In UPIU handled by UTP Engine

UPIU TYpe	Relevant UPIU Fields	UTP Engine actions on UPIU fields
Data In	Transaction Type	Matched against xx10 0010b
	LUN, Task Tag	Used to identify the corresponding Command which created the current transaction this Data In UPIU belongs to.
	Data Segment Length	Same value as Data Buffer Count value (see below).
	Data Buffer Offset, Data Transfer Count	Used to identify the correct Host memory location and DMA write transfer length (in conjunction with PRD Table info)
	Data Segment	Data to be transferred to Host memory by the Host Controller.

Ready To Transfer UPIUs are analyzed by the Host Controller/UTP Engine to extract the necessary information provided by the UFS Device about the next Data Out UPIU the UTP Engine is expected to generate.

Table 5 — Inbound RTT UPIU handled by UTP Engine

UPIU TYpe	Relevant UPIU Fields	UTP Engine actions on UPIU fields
Ready To Transfer	Transaction Type	Matched against xx11 0001b.
	LUN, Task Tag	Used to identify the corresponding Command which created the current transaction this RTT UPIU belongs to.
	Total EHS Length	Always ‘0’
	Data Segment Length	Always ‘0’
	Data Buffer Offset, Data Transfer Count	Used to identify the correct Host memory location and DMA read transfer length (in conjunction with PRD Table info) to compose the requested Data Out UPIU.

7.3.3 Processing UTP Transfer Request Completion

When a UTP Transfer Request Completion is received in the host controller, it is handled as follows:
If the completion is for a Regular Command (UTRD.I=0):

a) The IA interrupt counter is incremented.
b) If the counter was 0 before incrementing, the IA timer starts running

IS.UTRCS bit is set when at least one of the following 3 conditions is met:

a) The UTRD.I bit is set (Interrupt Command)
b) The counter, after incrementing, reaches the value configured in IACTH
c) The IA timer reaches the value configured in IATOVAL (this event may occur at any time, not necessarily coupled with request completion).

7.3.3 Processing UTP Transfer Request Completion (cont'd)

An interrupt is generated by the write operation if the completion interrupt is not masked (disabled) by the IE.UTRCE bit.

Host software processes the interrupt generated by host controller for command completion. In the interrupt service routine, host software checks register **IS** to determine if there is an interrupt pending. If the UFSHCI has an interrupt pending:

1. Host software determines the cause of the interrupt by reading the **IS** register. It is possible for multiple bits to be set.
2. Host software clears appropriate bits in the **IS** register corresponding to the cause of the interrupt.
3. Host software reads the **UTRLDBR** register, and compares the current value to the list of commands previously issued by host software that are still outstanding. Host software completes any outstanding command whose corresponding bit has been cleared in the respective register. **UTRLDBR** is a volatile register; software should only use its value to determine commands that have completed, not to determine which commands have previously been issued.
4. If there were errors, noted in the **IS** register, host software performs error recovery actions.

7.4 Task Management Function

UTMRL is used to send a UTP Task Management Function to the attached device. Host controller shall place higher priority on request for task management function over a UTP Transfer Request. When a task management request is submitted, host controller is required to suspend any active UTP transfer requests and switch to dispatch the task management to the attached device. The granularity of the context switching should happen at the transfer of the boundary of UPIU.

7.4.1 Basic Steps when Building a UTP Task Management Request

When host software needs to send a UTP Task Management function to host controller, it utilizes *UTP Task Management Request List*. The following is the steps for host software to build a UTP Task Management Request.

1. Find an empty transfer request slot by reading the **UTMRLDBR**. An empty transfer request slot has its respective bit cleared to '0' in the **UTMRLDBR**.
2. Host software builds a UTMRD at the empty slot.
3. **I** (interrupt) bit set if software requests **HCS.UTMRCE** to be set on command completion.
4. Set **OCS** to 'Fh'.
5. Program filed *Task Management Request UPIU*.
6. Initialize field *Task Management Response UPIU* with '0'.
7. Repeat the step 1 to step 7 for each task management function to be sent to host controller for execution.
8. Check register **UTMRLRSR** and make sure it is read '1' before continuing.
9. Set *UTP Task Management Request Completion Enable* (UTMRCE) enable bit to '1' to enable the interrupt.
10. Set **UTMRLDBR** to ring the doorbell register to indicate to the host controller that multiple transfer requests are ready to be sent to the attached device. Host software shall only write a '1' to the bit position that corresponding to the new command; All other bit positions within **UTMRLDBR** should be written with a '0', which indicates no change to their current values.

7.4.2 Processing UTP Task Management Completion

Host software processes the interrupt generated by host controller for command completion. In the interrupt service routine, host software checks **IS** to determine if there is an interrupt pending. If the UFSHCI has an interrupt pending:

1. Host software determines the cause of the interrupt by reading the **IS** register. It is possible for multiple bits to be set.
2. Host software clears appropriate bits in the **IS** register corresponding to the cause of the interrupt.
3. Host software reads the **UTMRLDBR** register, and compares the current value to the list of commands previously issued by host software that are still outstanding. Host software completes with success any outstanding command whose corresponding bit has been cleared in the respective register. **UTMRLDBR** is a volatile register; software should only use its value to determine commands that have completed, not to determine which commands have previously been issued.
4. If there were errors, noted in the **IS** register, host software performs error recovery actions.

7.5 UIC Power Mode Change

The UIC power mode change is performed using the DME_SET command to set UIC attributes. The UIC power mode change is an atomic operation, which means that these attributes need to be set according to certain rules as described subsequently in this section.

One or more of the following attributes can be set as necessary:

- PA_ActiveTxDataLanes
- PA_ActiveRxDataLanes
- PA_TxGear
- PA_RxGear
- PA_TxTermination
- PA_RxTermination
- PA_HSSeries
- PA_PWRModeUserData[0:11]
- DME_ LocalL2TimerData_FC0ProtectionTimeOutVal
- DME_ LocalL2TimerData_TC0ReplayTimeOutVal
- DME_ LocalL2TimerData_ AFC0ReqTimeOutVal
- DME_ LocalL2TimerData_FC1ProtectionTimeOutVal
- DME_ LocalL2TimerData_TC1ReplayTimeOutVal
- DME_ LocalL2TimerData_ AFC1ReqTimeOutVal
- PA_PWRMode

Currently, UFS uses TC0 only. Therefore, setting the following values are not needed:

- DME_ LocalL2TimerData_FC1ProtectionTimeOutVal
- DME_ LocalL2TimerData_TC1ReplayTimeOutVal
- DME_ LocalL2TimerData_ AFC1ReqTimeOutVal

Setting the PA_PWRMode attribute triggers the UIC power mode change. Therefore, PA_PWRMode shall be the last attribute to be set. The other attributes can be set in any order before PA_PWRMode.

7.5 UIC Power Mode Change (cont'd)

The UIC power mode change operation completes when IS.UPMS is set to '1'. If the IE.UPMSE is set, the UIC power mode change completion is also signaled by an interrupt. The status of the UIC power mode change is given by the value of HCS.UPMCRS. If HCS.UPMCRS is set to PWR_LOCAL, the UIC power mode change has completed with success. If HCS.UPMCRS is set to PWR_ERROR_CAP or PWR_FATAL_ERROR, the UIC power mode change has failed. HCS.UPMCRS cannot be set to PWR_REMOTE or PWR_BUSY, as these are associated to UIC power mode change requests from the UFS Device, which are not permitted in UFS 1.0.

To guarantee the atomicity of the UIC power mode change operation, between the PA_PWRMode attribute is set with DME_SET and the Host Controller Interface sets IS.UPMS, the software shall not set the attributes listed above. Otherwise, the behavior is undefined as the Host Controller Interface is not required to prevent these attributes from being set during the UIC power mode change.

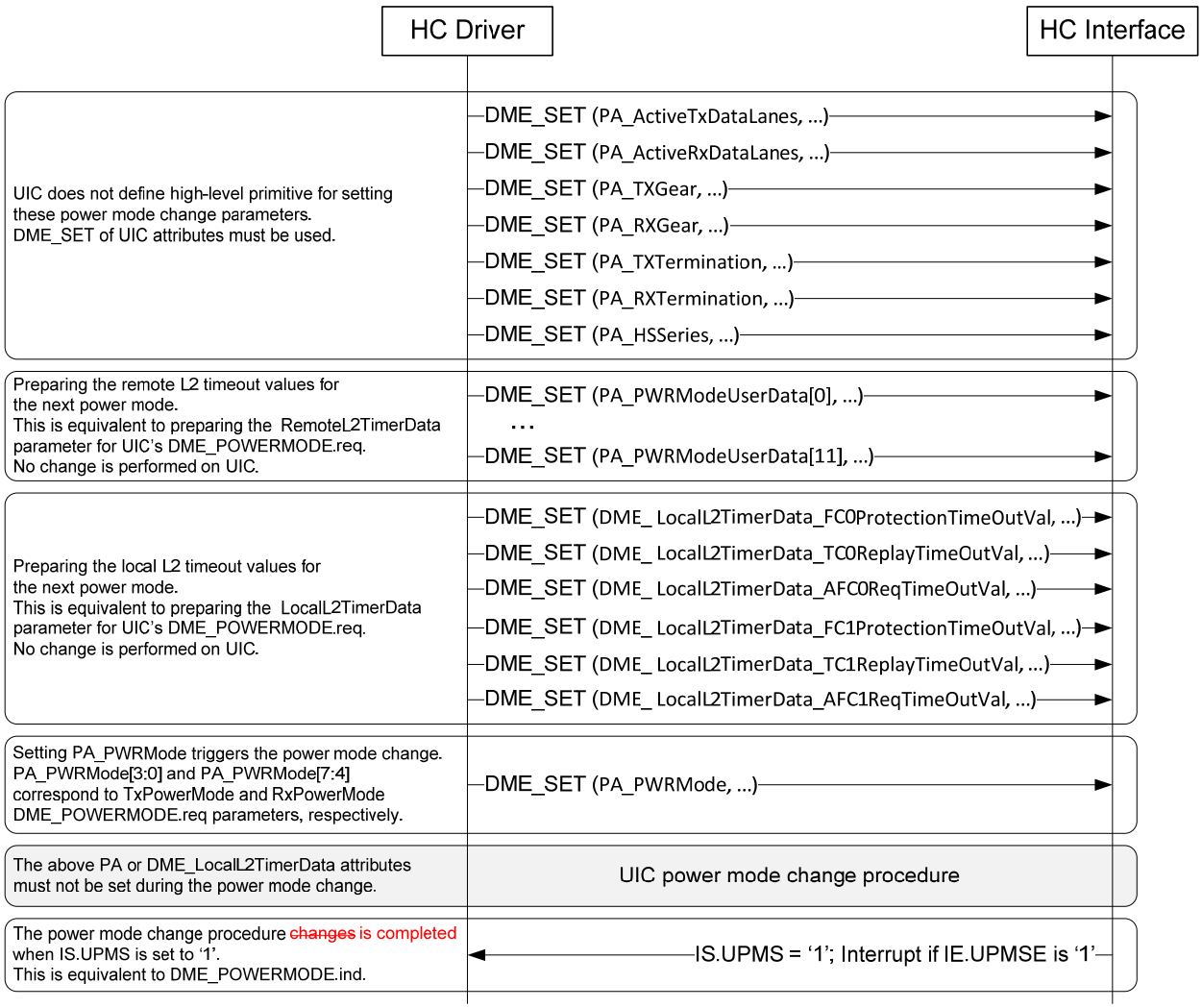


Figure 9 — UIC Power mode change

7.6 UFSHCI Internal Rules

7.6.1 Command Processing Order

UFSHCI processes three types of commands, which shall be processed using the following priority (in order of priority, highest priority first):

- UIC Commands issued via the UIC Command Registers.
- Task Management Requests issued via UTP Task Management Request List.
- Transfer Requests issued via the UTP Transfer Request List.

Software shall issue UIC Commands one at a time. For Task Management Requests and Transfer Requests, software may issue multiple commands at a time, and may issue new commands before previous commands have completed. When software sets the corresponding Door Bell register, the Task Management Requests and Transfer Requests automatically get a time stamp with their issue time. The commands within a command list (Task Management List or Transfer Request List) shall be processed in the order of their time stamps, starting from the oldest time stamp. In the case multiple commands from the same list have the same time stamp, they shall be processed in the order of their command list index, starting from the lowest index.

The UPIUs associated with Task Management Requests and Transfer Requests are sequentialized due to the use of a single UIC CPort. Therefore, the UPIUs associated with these Requests are transmitted one at a time.

In addition to the Request UPIUs provided by software (NOP Out, Command, Task Management Request and Query Request), UFS HCI also generates Data Out UPIUs. The order of the Data Out UPIUs and their interleaving with the Request UPIUs is implementation dependent, and is therefore not specified.

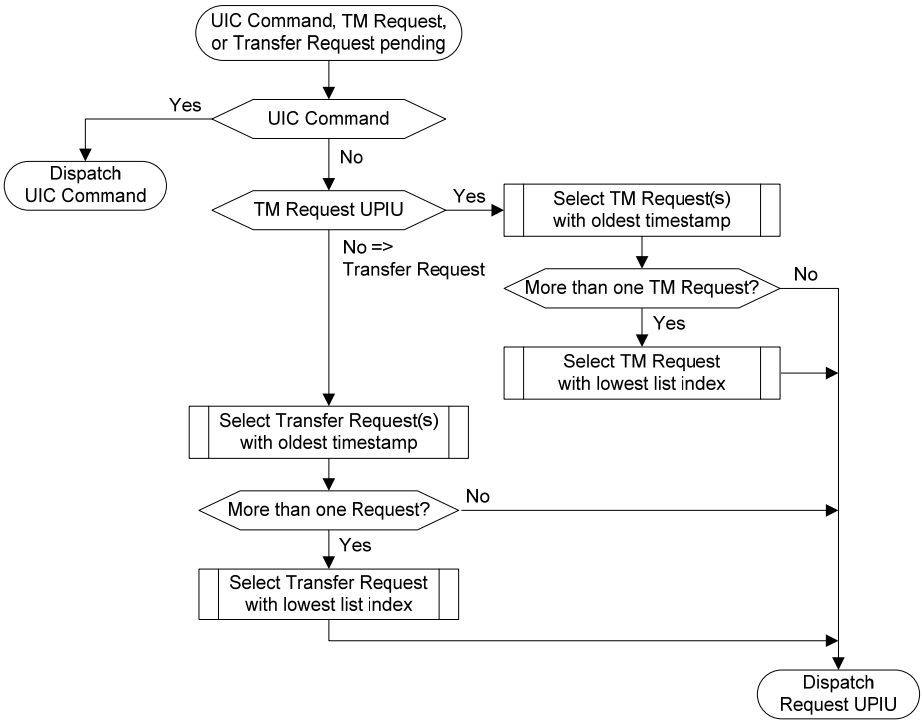


Figure 10 — Command processing order

8 Error reporting and handling

This section describes the various errors that UFS host controller can report and how they are handled.

8.2 Error Types

8.2.1 System Bus Error

There are several sources of errors that could occur on the system bus:

- Bad memory pointers. This occurs when host software provide an invalid memory pointer.
- ECC/parity error on the bus. This could result from CRC or parity checking on system bus or memory. It may or may not be transient.
- Bad operation. This occurs when host controller tries to write to read-only memory.
- Protection violation. This occurs when host controller tries to read/write to protected memory without proper privilege.

All these errors are not recoverable without software intervention. When one of them occurs, host controller shall abort the request and report the error by setting register **IS.SBFES**. Host controller optionally generates an interrupt if **IE.SBFEE** is set.

8.2.2 UIC Error

UIC errors occur within UIC layers of host controller. When a UIC error occurs, host controller shall abort the request and report the error by setting register **IS.UE**. Host controller optionally generates an interrupt if **IE.UEE** is set. Host controller shall check all of the following registers for the cause of the errors:

- **UECPA** for Host UIC Error Code within PHY Adapter Layer.
- **UECDL** for Host UIC Error Code within Data Link Layer.
- **UECN** for Host UIC Error Code within Network Layer.
- **UECT** for Host UIC Error Code within Transport Layer.
- **UECDME** for Host UIC Error Code within DME subcomponent.

Refer to 5.3.5 through 5.3.9 for the error codes of each UIC error register.

8.2.3 UIC Command Error

UIC Command errors are indicated by a non-zero ConfigResultCode or GenericErrorCode field in the **UICCMDARG2 register after the UIC Command has completed.**

These errors are generally recoverable by software.

8.2.4 UTP Error

There are two mechanisms that host controller uses to report a UTP error:

- *UTP Transfer Request*. When a UTP transfer request completes (success or failure as indicated by **IS.UTRCS**), host software shall check field **OCS** of UTRD for the request just completed. The field contains the status code for the request. Host software may need to check Transfer Response UPIU's Response, and possibly Sense Data to find out more details for the cause of the error. The host controller does not halt for non-fatal error conditions.

- *UTP Task Management Request*. When a UTP task management request completes (success or failure as indicated by **IS.UTMRCS**), host software shall check OCS of UTMRD for request status. Host software may need to check Task Management Response UPIU to find out more details for the cause of the error. The host controller does not halt for non-fatal error conditions.

8.2.5 Host controller Fatal Error

Within host controller errors could occur outside of system bus and UIC subcomponent. These errors are reported as host controller fatal errors. When a host controller fatal error is detected, the host controller shall set the **IS.HCFES** bit. If **IE.HCFEE** is set, the host controller shall generate an interrupt.

8.2.6 Device Error

Device errors are the fatal error conditions that prevent the device from completing any request. They apply the entire device. When a fatal device error is detected, the host controller shall set the **IS.DFES** bit. If **IE.DFEE** is set, the host controller shall generate an interrupt. Before setting the **IS.DFES** bit, the host controller shall perform the following steps:

1. Clear **HCS.UTRLRDY.**

2. Clear **HCS.UTMRLRDY**.

3. Abort all outstanding UTP transfer requests by clearing the **UTRLCLR** bits corresponding to the outstanding requests.

4. For each of the outstanding request, update the **OCS** field of the UTRD corresponding request to indicate the reason for the device error.

5. Abort all outstanding UTP task management requests by clearing the **UTMRLCLR** bits corresponding to the outstanding requests. For each of the outstanding request, update the **OCS** field of the UMRD corresponding request to indicate the reason for the device error.

8.3 Error Handling

8.3.1 System Bus Error Handling

This standard does not define any reporting and handling mechanism for invalid pointers to **UTRLBA**[**U**] and **UTMRLBA**[**U**]. Host software shall ensure these pointers are valid before providing to the host controller.

System Bus Errors are serious errors that cannot be recovered from without software intervention. In case of invalid pointer for a *UTRD*, host software shall use the following procedure to recover

- Identify the failed request with an invalid pointer to its UTRD
  - Reads **UTRLDBR** to determine which requests are still outstanding
  - Checks **OCS** field of the *UTRD* for each request completed to determine if there was an error with that request, If the OCS field of the UTRD for the failed request is the same value as the value for initialization of the UTRD, the pointer to the UTRD is invalid.
- Abort the request by clearing corresponding **UTRLCLR** bit to '0'.
- Host software then can re-issues these requests to the host controller.

In case of invalid pointer within a UTP Task Management Request Descriptor, host software shall use the following procedure to recover

- Identify the failed request with an invalid pointer to its UTMRD
  - Reads **UTMRLDBR** to determine which requests are still outstanding
  - Checks **OCS** field of the *UTMRD* for each request completed to determine if there was an error with that request. If the OCS field of the UTMRD for the failed request is the same value as the value for initialization of the UTMRD, the pointer to the UTMRD is invalid.
- Abort the request by clearing corresponding **UTMRLCLR** bit to '0'.
- Host software then can re-issues these requests to the host controller.

For all other system bus errors, the host controller aborts the command with either a "invalid transfer request descriptor attributes" or "invalid PRDT attributes" status code.

For the error that occurs within UTP Transfer Request List, the following procedure can be used to recover:

- Identify the failed request
  - Reads UTRLDBR to determine which requests are still outstanding
  - Checks OCS field of the *UTRD* for each request completed to determine if there was an error with that request. If the OCS field of the UTRD for the failed request is not 'Fh', the pointer to the UTRD is valid. Otherwise, there is an error with UTRD.
- Abort the request by clearing corresponding **UTRLCLR** bit to '0'.
- Host software then can re-issues these requests to the host controller.

For the error that occurs within UTP Task Management Request List, the following procedure can be used to recover:

- Identify the failed request
  - Reads **UTMRLDBR** to determine which requests are still outstanding
  - Checks **OCS** field of the *UTMRD* for each request completed to determine if there was an error with that request. If the OCS field of the UTMRD for the failed request is not 'Fh', the pointer to the UTMRD is valid. Otherwise, there is an error with UTRD
- Abort the request by clearing corresponding **UTMRLCLR** bit to '0'.
- Host software then can re-issues these requests to the host controller.

8.3.2 UIC Error Handling

The following are the rules for handling UIC errors:

- Except PA_INIT_ERROR, UECPA and UECDL are non-fatal error. UIC recovers by itself.
- PA_INIT_ERROR should lead to a UIC reset.
- UECN, UECT and UECDME need software intervention. UIC doesn't need to be reset.

When a fatal UIC error occurs, host software shall follow the steps below to recover:

- Reads **UTRLDBR** to determine which requests are completed.
- Checks **OCS** field of the *UTRD* for each request completed to determine if there was an error with that request, and if so the host software shall ignore the data in the system memory locations for requests that complete with such error conditions.
- Save to a list of outstanding and failed request so that they can be optionally re-issue them after host controller reset.
- Reset the controller by setting register HCE to ‘0’.
- Re-initialize the host controller by setting register HCE to ‘1’.
- Optionally host software can re-issue the saved requests to the host controller.

8.3.3 UIC Command Error Handling

In case the UIC Command Error is caused by DME_GET, DME_SET, DME_PEER_GET or DME_PEER_SET, the ConfigResultCode field in the **UICCMDARG2 register** carries the cause of the error. This is either an incorrect set of UIC Command parameters, or UIC state when UIC cannot be configured. In the case of DME_PEER_GET and DME_PEER_SET, the UIC link may also be in a state where it cannot transfer the UIC configuration command to the Device.

The host software should reissue the UIC Command with a correct set of parameters when the UIC is in a state where it can accept UIC Commands.

8.3.4 UTP Error Handling

8.3.4.1 UTP Transfer Request Error Handling

The following is a flow that host software shall use to recover from an error within UTP Transfer Request List:

- Reads **UTRLDBR** to determine which requests are still outstanding
- Checks **OCS** field of the *UTRD* for each request completed to determine if there was an error with that request, and if so the error condition
- Clear **IS.UTRCS** to ‘0’.
- Host software then either completes the request that had the error and requests still outstanding with error to higher level software, or re-issues these requests to the host controller.

8.3.4.2 UTP Task Management Request Error Handling

The following is a flow that host software shall use to recover from an error within UTP Task Management Request List:

- Reads **UTMRLDBR** to determine which requests are still outstanding
- Checks **OCS** field of the *UTMRD* for each request completed to determine if there was an error with that request, and if so the error condition
- Clear **IS.UTMRCS** to ‘0’.
- Host software then either completes the request that had the error and requests still outstanding with error to higher level software, or re-issues these requests to the host controller.

8.3.5 Host Controller Error Handling

Host Controller Errors are fatal errors. When this condition occurs, host software should reset by setting register HCE to ‘0’ and then re-initialize the host controller by setting register HCE to ‘1’.

8.3.6 Device Error Handling

Device Errors are fatal errors. When this condition occurs, host software shall follow the same procedure for UIC error handling as described in 8.2.2, except that in addition to resetting UIC, the host software shall reset the device too.

STANDARD IMPROVEMENT FORM **JEDEC JESD223**

The purpose of this form is to provide the Technical Committees of JEDEC with input from the industry regarding usage of the subject standard. Individuals or companies are invited to submit comments to JEDEC. All comments will be collected and dispersed to the appropriate committee(s).

If you can provide input, please complete this form and return to:

JEDEC
Attn: Publications Department
3103 North 10th Street
Suite 240 South
Arlington, VA 22201-2107

Fax: 703.907.7583

1. I recommend changes to the following:
   - ☐ Requirement, clause number _____
   - ☐ Test method number _____ Clause number _____

   The referenced clause number has proven to be:
   - ☐ Unclear ☐ Too Rigid ☐ In Error
   - ☐ Other _____

2. Recommendations for correction:

3. Other suggestions for document improvement:

Submitted by

Name: _____ Phone: _____

Company: _____ E-mail: _____

Address: _____

City/State/Zip: _____ Date: _____


JEDEC