JEDEC STANDARD

Universal Flash Storage (UFS) File Based Optimizations (FBO) Extension

Version 1.0

JESD231

AUGUST 2022

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.

Special Legal Disclaimer: JEDEC has received information that certain patents or patent applications may be essential to this standard. However, as of the publication date of this standard, no statements regarding an assurance to license such patents or patent applications have been provided. JEDEC does not make any determination as to the validity or relevancy of such patents or patent applications. Anyone making use of the standard assumes all liability resulting from such use. JEDEC and its members disclaim any representation or warranty, express or implied, relating to the standard and its use.

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 2022
3103 North 10th Street
Suite 240 South
Arlington, VA 22201-2107

PRICE: Contact JEDEC

UFS HOST FILE BASED OPTIMIZATION (FBO) EXTENSION, VERSION 1.0

CONTENTS

FIGURES

TABLES

UFS FBO (FILE BASED OPTIMIZATION) EXTENSION, VERSION 1.0

Foreword

This standard has been prepared by JEDEC. The purpose of this standard is to define a UFS File Based Optimization (FBO) Extension specification. This document is extension of the UFS Standard, JESDxxx.

Introduction

The UFS device is a universal data storage and communication media. It is designed to cover a wide area of applications as smart phones, VR(virtual reality) device, AR(augmented reality) device, Drone, 3D games, surveillance system, cameras, organizers, PDAs, digital recorders, MP3 players, internet tablets, electronic toys, etc.

This extension specification describes the File Based Optimization (FBO) feature for UFS to improve device performance by utilizing file base information which is transferred to the device.

UFS FBO (FILE BASED OPTIMIZATION), VERSION 1.0

(From JEDEC Board Ballot JCB-22-10, formulated under the cognizance of the JC-64.1 Subcommittee on Electrical Specifications and Command Protocols (Item 139.54).)

1 SCOPE

This standard specifies the extension specification of the UFS electrical interface and the memory device. This document describes the extended feature, called File Based Optimization (FBO), in UFS specification. It also provides some details in how to utilize the FBO for gaining higher performance in UFS devices.

2 NORMATIVE REFERENCE

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

[UFS], JEDEC JESD220D, Universal Flash Storage (UFS), Version 3.1

3 TERMS, AND DEFINITIONS

For the purpose of this standard, the terms and definitions given in the documents included in section 2 “Normative Reference” and the following apply.

3.1 Acronyms

FBO	File Based Optimization
HCI	Host Controller Interface
UFS	Universal Flash Storage
MIPI	Mobile Industry Processor Interface
RPMB	Replay Protected Memory Block
SBC	SCSI Block Commands
SPC	SCSI Primary Commands
LUN	Logical Unit Number
NA	Not applicable
KB	Kilobyte
eUFS	Embedded Universal Flash Storage
LBA	Logical Block Address

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: An addressable device on the UFS bus usually a target that contains at least one LUN
Host: An addressable device on the UFS bus which is usually the main CPU that hosts the UFS bus

3.3 Keywords

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

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

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

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

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

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

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

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

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

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

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

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

3.4 Abbreviations

etc. - And so forth (Latin: et cetera)
e.g. - For example (Latin: exempli gratia)
i.e. - That is (Latin: id est.)

3.5 Conventions

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

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

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

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

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

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

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

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

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

4 INTRODUCTION

4.1 Overview

Stoage devices have a long lifespan. Device performance over its lifespan is not constant and may deteriorate over time.

This specification describes a method in which the host provide the storage device File System information in order to improve the performance regression. Based on that information device analyzes the file system data and provides the host the level of performance regression. The host then may instruct the device to execute optimization procedure to improve the regression level.

This feature should not interfere with other device operations. Other device operations are not interfered at any time and be seamless to the host.

5 FILE BASED OPTIMIZATION (FBO) DESCRIPTION

5.1 Overall FBO behavior

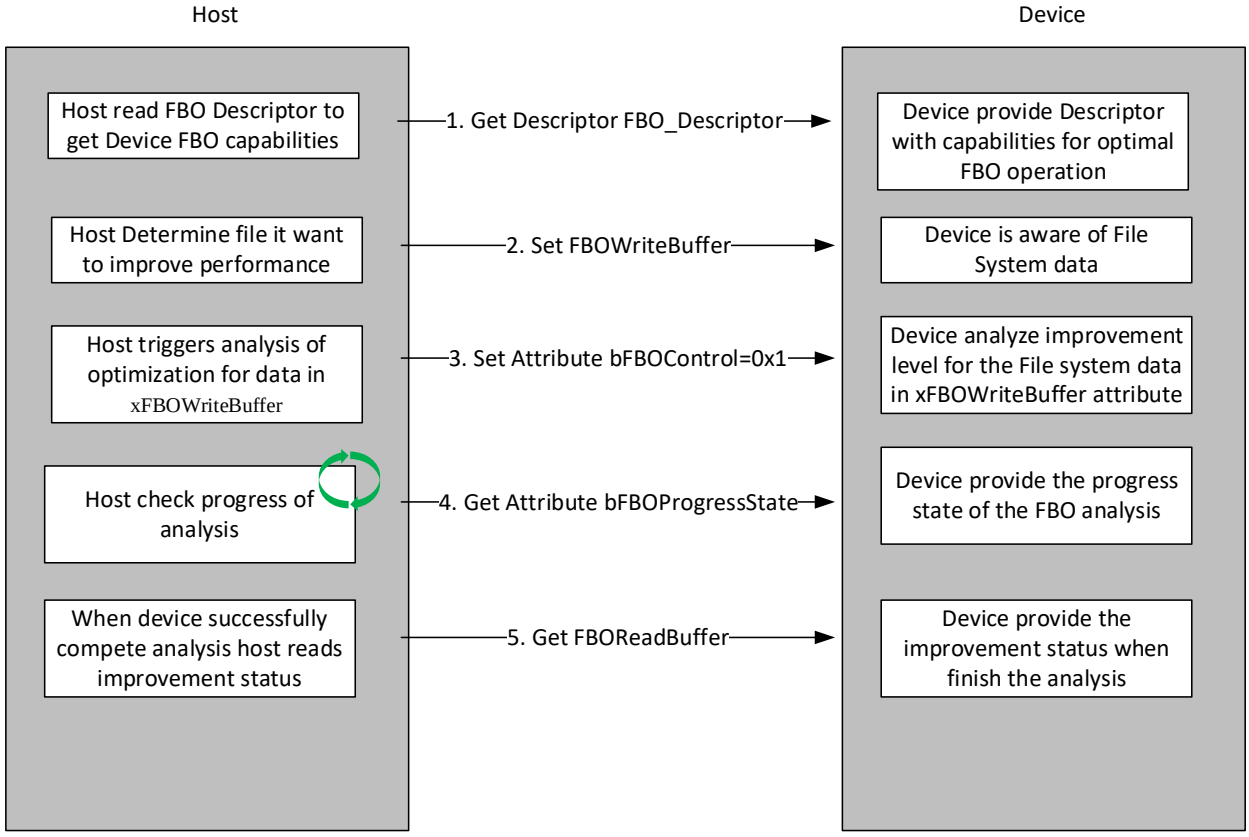

Figure 5-1 — FBO Analysis Flow

The analysis flow starts after the host software reads the device FBO descriptor to get the device FBO capabilities (**Figure 5-1** (1)). The SW then determines which file it would like to improve performance and provide the File system data to the device (**Figure 5-1** (2)) by issuing a WRITE BUFFER command (see 6.1). The File system data is composed of the File LBA ranges, as described in Table 6.2 — FBOWriteBuffer Type Specific Informaiton Structure.

The host write to the FBOWriteBuffer to restart a new FBO flow. The host then sets bFBOControl attribute to 0x1 (**Figure 5-1** (3)) to start analysis of the LBA ranges in the FBOWriteBuffer. The host may query the bFBOProgressState (**Figure 5-1** (4)), waiting for the analysis to complete (change from 01h (On-going FBO operation) to 02h (Complete FBO Analysis)).

Once the analysis is completed, the device shall update FBOReadBuffer with the improvement level for each LBA range provided in the FBOWriteBuffer and set the bFBOProgressState to 02h (Complete FBO Analysis).

The host read the bFBOProgressState attribute, and when it is set to 02h (Complete FBO Analysis), it should read the FBOReadBuffer to get the analysis results (**Figure 5-1** (5)).

At this point the host has the information to determine if device will perform the optimization or not. If the host does not want to perform optimization it will set the attribute bFBOControl to value 0x00 to abort the FBO operation. In case the host would like to continue with Optimization phase it will follow the flow in **Figure 5-2**.

5.1 Overall FBO behavior (cont'd)

In case device encountered an error in the analysis or optimization phases, it would set bFBOProgressState as 0xFF (General Error).

The host may stop the current flow by setting the bFBOControl attribute to 0x0 at any progress state (bFBOProgressState is 0x01, 0x02, 0x03 or 0xFF). Host may then set the FBOWriteBuffer with new parameters to re-initial the write buffer. Setting bFBOControl to 0x0 clears the the bFBOProgressState attribute to Idle state (0x0).

In case the flow is completed (bFBOProgressState is 0x2, 0x3 or 0xFF), the host may re-initiate the FBO flow by setting bFBOControl attribute to 0x1, the device will start the flow using the content of the FBOWriteBuffer.

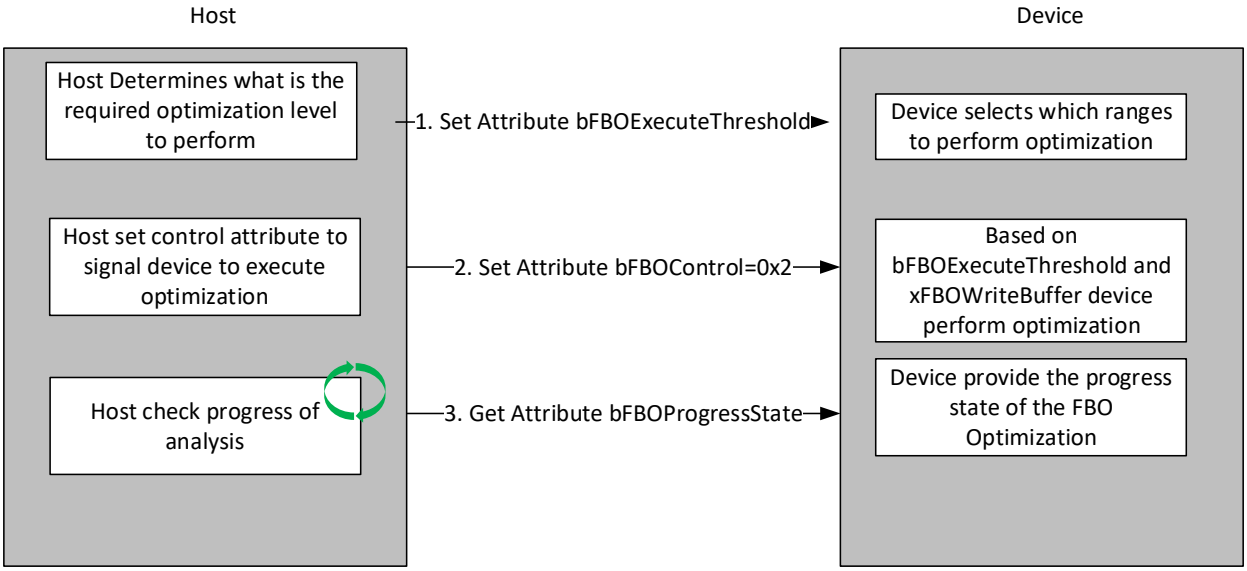

Figure 5-2 — FBO Optimization Flow

5.1.1 Controlling FBO Operations

FBO operations are performed while bCurrentPowerMode is Active and host has to keep device power rails. Host may set device power state to Sleep, Deep-Sleep or Power-Down and device shall resume FBO operation once returned to Active state. However, in case device power (i.e., VCCQ) is turned off or device reset occurs the device lose all volatile information and host would have to restart the FBO process.

Depicted in Figure 5-3 is the state machine which represents the value of bFBOProgressState attribute.

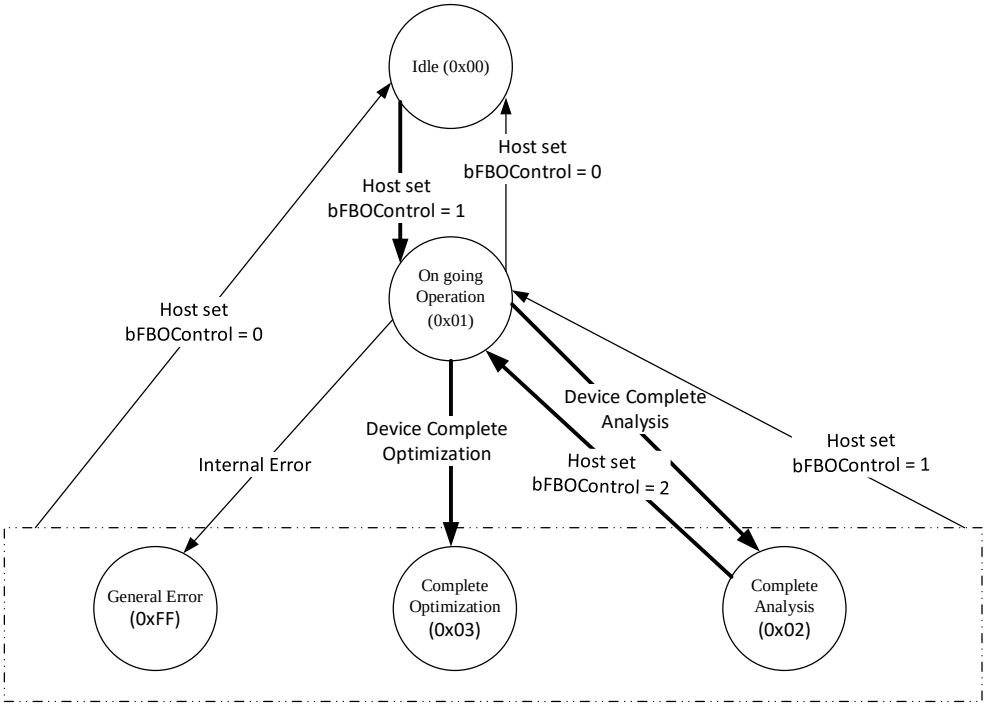

Figure 5-3 – FBO Flow state machine

File Based Optimization has 2 operations: FBO Analysis and FBO Optimization.

The host may start the FBO Analysis while bFBOProgressState is in any state except 01h (On-going FBO Operation) by setting bFBOControl to 01h. The device shall use the content of the FBOWriteBuffer to perform the analysis. While the analysis is on-goging, device shall update bFBOProgressState to 01h (FBO On-going).

The host may termindate the FBO operation (FBO Analysis and FBO Optimization) by setting bFBOControl to 00h at any time. As a result, device would set bFBOProgressState back to Idle (00h).

While the device is in Idle state, the host may write the FBOWriteBufer multiple time before setting bFBOControl to 01h (Start FBO Analysis), but only the last buffer will be stored and used for the analysis. However, after bFBOProgressState is updated to FBO On-onging any command to access the FBOWriteBuffer or FBOReadBuffer shall terminate with an error as device may be reading or writing the buffers for on-going FBO operaitons.

The host may start FBO Optimization by setting bFBOControl to 02h only once device finished FBO Analysis and bFBOProgressState is 02h (Complete FBO Analysis). The device will perform FBO Optimization based on the values of FBOWriteBuffer and bFBOExecuteThreshold. Setting bFBOControl to 02h, while bFBOProgressState is not 02h shall result in QUERY RESPONSE UPIU with Query Response Code of Parameter not writeable..

6 FBO BUFFERS

Two volitle buffers are used to transfer data from the host SW to the device and vice verse.

6.1 FBOWriteBuffer

Ths FBOWriteBuffer is a write-only volatile buffer. The host use this buffer in order to provide the device information on the LBA ranges device would be required to perform the FBO Analysis.

The size of this buffer is 4KB.

6.1.1 Access to FBOWriteBuffer

The host access the FBOWriteBuffer using WRITE BUFFER command with the following parameters:

- MODE = 02h (DATA)
- BUFFER ID = 01h (FBOWriteBuffer)
- BUFFER OFFSET = 0

A WRITE BUFFER command (with MODE 02h, BUFFER ID 01h) shall be terminated with CHECK CONDITION status, with the sense key set to DATA PROTECT in case sent while bFBOProgressState is 01h (On-going FBO Operation).

In case the buffer structure does not meet the FBOWriteBuffer structure or violates the File Based Optimization (FBO) Descriptor the WRITE BUFFER command (with MODE 02h, BUFFER ID 01h) shall fail and a status response of CHECK CONDITION will be returned along the sense key set to ILLEGAL REQUEST.

A READ BUFFER command (with MODE 02h, BUFFER ID 01h) shall be terminated with CHECK CONDITION status, with the sense key set to DATA PROTECT (as the FBOWriteBuffer is write-only).

6.1.2 FBOWriteBuffer Generic Structure

Table 6.1 — FBOWriteBuffer General Structure

Bit / Byte	7	6	5	4	3	2	1	0
FBOWriteBuffer Header								
0	FBO Type							
1	Reserved							
2	Reserved							
3	Reserved							
FBOWriteBuffer Body								
4	Type Specific Informaiton							
…								
4095								

6.1.2.1 FBO Type

0x0 – File Based Sequential Read improvement
0x1 – 0xFF – Reserved

6.1.2.2

Blank at time of publication..

6.1.2.3 Type Specific Informaiton

This field is specific per FBO Type. For more information see 6.1.3

6.1.3 FBOWriteBuffer Type Specific Informaiton Structure for FBO Type = 0x0

Table 6.2 provide the data structure of the FBOWriteBuffer FBO Type = 0x0.

Table 6.2 — FBOWriteBuffer Type Specific Informaiton Structure FBO Type = 0x0

Bit Byte	7	6	5	4	3	2	1	0
0	Version							
1	Number of FBOWriteBufferEntries							
2								CAR
3	(MSB)							
…		Reserved						
7								(LSB)
8	(MSB)							
…		FBO WRITEBUFFER ENTRY 1						
15								(LSB)
	….							
8*N	(MSB)							
…		FBO WRITEBUFFER ENTRY N						
8*N+7								(LSB)

6.1.3.1 Number of FBOWriteBufferEntries

This field specifies the number of entries provided in the FBOWriteBuffer.

Value in this field should be up to bFBOMaxLBARangeCount.

6.1.3.2 FBO WRITEBUFFER ENTRY

Bit Byte	7	6	5	4	3	2	1	0
0	(MSB)							
…				Start LBA				
3								(LSB)
4	(MSB)							
…				Length				
6								(LSB)
7	Reserved							

6.1.3.2.1 Start LBA

This field specifies the start LBA of the LBA range to be FBO Analysed.

6.1.3.2.2 Length

This field specifies the Length in Bytes of the LBA range to be FBO Analysed.

6.1.3.3 Calculate All Ranges (CAR)

When this bit is set to 1b, the UFS device shall provide the regression level for all provided ranges in the field All Ranges Regression Level when reading the FBOReadBuffer.

When this bit is set to 0b, the UFS device shall not provide All Ranges Regression Level when reading the FBOReadBuffer.

6.2 FBOReadBuffer

Ths FBOReadBuffer is a read-only volatile buffer. The host use this buffer in order to receive from the device information about the analysis provided for the data in FBOWriteBuffer.

The size of this buffer is 4KB.

6.2.1 Access to FBOReadBuffer

The host access the FBOReadBuffer using READ BUFFER command with the following parameters:

- MODE = 02h (DATA)
- BUFFER ID = 02h (FBOReadBuffer)
- BUFFER OFFSET = 0

A READ BUFFER command (with MODE 02h, BUFFER ID 02h) shall be terminated with CHECK CONDITION status, with the sense key set to DATA PROTECT in case sent while bFBOProgressState is 01h (On-going FBO Operation).

A WRITE BUFFER command (with MODE 02h, BUFFER ID 02h) shall be terminated with CHECK CONDITION status, with the sense key set to DATA PROTECT (as the FBOReadBuffer is read-only).

6.2.2 FBOReadBuffer Generic Structure

FBOReadBuffer Generic Structure

Bit / Byte	7	6	5	4	3	2	1	0
FBOReadBuffer Header								
0	FBO Type							
1	Reserved							
2	Reserved							
3	Reserved							
FBOReadBuffer Body								
4	Type Specific Informaiton							
…								
4095								

6.2.2.1 FBO Type

0x0 – File Based Sequential Read improvement

0x1 – 0xFF – Reserved

6.2.2.2 THIS SUBCLAUSE LEFT BLANK AT TIME OF PUBLICATION

6.2.2.3 Type Specific Informaiton

This field is specific per FBO Type. For more information see 6.2.3

6.2.3 FBOReadBuffer Type Specific Informaiton Structure for FBO Type = 0x0

Table 6.3 provide the data structure of the FBOReadBuffer for FBO Type = 0x0.

Table 6.3 — FBOReadBuffer Type Specific Informaiton Structure FBO Type = 0x0

Bit / Byte	7	6	5	4	3	2	1	0
0	Version							
1	Number of FBOReadBufferEntries							
2								CAR
3	All Ranges Regression Level							
4	(MSB)							
…				Reserved				
7								(LSB)
8	(MSB)							
…				FBO READBUFFER ENTRY 1				
15								(LSB)
	….							
8*N	(MSB)							
…				FBO READBUFFER ENTRY N				
8*N+7								(LSB)

6.2.3.1 Number of FBOReadBufferEntries

This field specifies the number of entries provided in the FBOReadBuffer.

This number shall be equal to the number of FBOWriteBufferEntries provided in the FBOWriteBuffer.

6.2.3.2 Calculated All Ranges (CAR)

When set to 1b: The UFS device provides the regression level for all provided ranges in the field All Ranges Regression Level, in addition to the calculation per region.When set to 0b: The UFS device does not provide the regression level for all provided ranges in the field All Ranges Regression Level, the host should ignore All Ranges Regression Level field.

6.2.3.3 All Ranges Regression Level

Overall regression level for all ranges in the Buffer calculated by the device.

The values of this fields are:

- 0x00 –no Regression
- 0x01 – 1%-10% Regression Level
- 0x02 – 11%-20% Regression Level
- …
- 0x0A – 91%-100% Regression Level

Other values: reserved

6.2.3.4 FBO READBUFFER ENTRY

Bit / Byte	7	6	5	4	3	2	1	0
0	(MSB)							
…				Start LBA				
3								(LSB)
4	(MSB)							
…				Length				
6								(LSB)
7	Regression Level							

6.2.3.4.1 Start LBA

This field specifies the start LBA of the LBA range to be FBO optimized.

6.2.3.4.2 Length

This field specifies the Length in Bytes of the LBA range to be FBO optimized

6.2.3.4.3 Regression Level

This field specifies the Regression Level for that entry.

The values of this fields are:

- 0x00 –no Regression
- 0x01 – 1%-10% Regression Level
- 0x02 – 11%-20% Regression Level
- …
- 0x0A – 91%-100% Regression Level
- Other values: reserved

7 DESCRIPTORS, FLAGS AND ATTRIBUTES

7.1 UFS Descriptors

A descriptor is a data structure with a defined format. Descriptors are accessed via QUERY REQUEST UPIU packets. Descriptors are independently addressable data structures. Descriptors may be stand alone and unique per device or they may be interrelated and linked in a hierarchical fashion to other descriptors by parameters defined within the top-level descriptor. Descriptors can range in size from 2 bytes through 255 bytes. A 2 byte descriptor is an empty descriptor. All descriptors have a length value as their first element. This length represents the entire length of the descriptor, including the length byte. All descriptors have a type identification as their second byte. If a parameter in a Descriptor has a size greater than one byte, the byte containing the MSB is stored at the lowest offset and the byte containing the LSB is stored at the highest offset (i.e., big-endian byte ordering). A descriptor can be partially read, but starting point is always the offset 00h.

A Descriptor is a block or page of parameters that describe something about a Device. For example, there are Device Descriptors, Configuration Descriptors, Unit Descriptors, etc.

In general, all Descriptors are readable, some may be write once, others may have a write protection mechanism.

The Configuration Descriptor is writeable and allows modification of the device configuration set by the manufacturer.

Table 7.1 specifies the descriptor identification values.

Table 7.1 — Descriptor identification values

Descriptor IDN	Descriptor Type
00h	Defined by UFS standard
01h	Defined by UFS standard
02h	Defined by UFS standard
03h	Defined by UFS standard
04h	Defined by UFS standard
05h	Defined by UFS standard
06h	Defined by UFS standard
07h	Defined by UFS standard
08h	Defined by UFS standard
09h	Defined by UFS standard
0Ah	FILE BASED OPTIMIZATION
0Bh ... FFh	Reserved

7.2 Device Descriptor for FBO

Table 7.2 — Device Descriptor for FBO

DEVICE DESCRIPTOR					
Offset	**Size**	**Name**	**MDV (1)**	**User Conf.**	**Description**
4Fh	4	dExtendedUFSFeaturesSupport	Device specific	No	Extended UFS Features Support. This field indicates which features are supported by the device. A feature is supported if the related bit is set to one. bit[17] : File Based Optimizations Other bits: See UFS specification [UFS]

NOTE 1 The column "MDV" (Manufacturer Default Value) specifies parameter values after device manufacturing. Some parameters may be configured by the user writing the Configuration Descriptor.

NOTE 2 "User Conf." column specifies which fields can be configured by the user writing the Configuration Descriptor: "Yes" means that the field can be configured, "No" means that the field is a capability of the device and cannot be changed by the user. The desired value shall be set in the equivalent parameter of the Configuration Descriptor.

7.3 File Based Optimization (FBO) Descriptor

The detailed format of the File Based Optimization (FBO) Descriptor is defined in Table 7.3.

In a QUERY REQUEST UPIU, the File Based Optimization (FBO) Descriptor is addressed by setting: DESCRIPTOR IDN = 0Ah. The value of INDEX and SELECTOR are set to 00h.

Table 7.3 — 1.1 File Based Optimization (FBO) Descriptor

FBO DESCRIPTOR				
Offset	**Size**	**Name**	**Value**	**Description**
00h	1	bLength	12h	Size of this descriptor
01h	2	wFBOVersion	0100h	FBO Specification Version Bits[15:8] = Major Version in BCD format Bits[7:4] = Minor Version in BCD format Bits[3:0] = Version suffix in BCD format Example: version 1.2.3 = 0123h
03h	4	dFBORecommendedLBARangeSize	Device specific	The reccomended LBA Range size in bytes to be used by the host. This field is in units of 4KB. 0x0 – no recommendation.
07h	4	dFBOMaxLBARangeSize	Device specific	The MAX LBA Range size to be used by the host, This field is in units of 4KB.
0Bh	4	dFBOMinLBARangeSize	Device specific	The MIN LBA Range size to be used by the host. This field is in units of 4KB.
0xFh	1	bFBOMaxLBARangeCount	Device specific	The maximum number of LBA ranges supported by Read/Write Buffer command. Shall be 32 or greater.
10h	2	wFBOLBARangeAlignment		Alignment requirement. 0 means no align requirement. This field is in units of 4KB.

7.4 Flags

Table 7.4 — Flags

FLAGS					
IDN	**Name**	**Type**	**Type[1]** **# Ind.[2]** **# Sel.[3]**	**Default**	**Description**

7.5 Attributes

Table 7.5 — Attributes

ATTRIBUTES							
IDN	**Name**	**Access Property**	**Size**	**Type[1]** / **# Ind.[2]** / **# Sel.[3]**	**MDV[4]**	**Description**	**Notes**
31h	bFBOControl	Volatile (Write-only)	1	D	0	Control FBO operation 0x00 – Device shall Stop FBO Analysis and FBO Optimization operation 0x01 – Start FBO Analysis based on the current value of FBOWriteBuffer. In case FBOWriteBuffer value is invalid, bFBOProgressState shall be set to 0xFF (General Error) 0x02 - Start FBO Optimization based on the current values of FBOWriteBuffer and bFBOExecuteThreshold. In case FBOWriteBuffer value is invalid, bFBOProgressState shall be set to 0xFF (General Error). NOTE Starting FBO Optimization is allowed only once Analysis is completed. Device shall return an error if host set bFBOControl to 0x02 and bFBOProgressState was not 02h – Complete FBO Analysis.	
32h	bFBOExecuteThreshold	Read/Volatile	1	D	Device Specific	Specifies from which FBO level of the range the device should execute the Optimization operation. 0x00 - Do optimization operation for the ranges which regression-level equal to or greater than 0x0 0x01 - Do optimization operation for the ranges which regression-level equal or greater than 0x1 … … 0x0A Do optimization operation for the ranges which regression-level is: equal 0x0A Other values are illegal.	
33h	bFBOProgressState	Read Only	1	D	0	Report FBO State. 00h – Idle 01h – On-going FBO Operation 02h – Complete FBO Analysis 03h – Complete FBO Optimization FFh – General Error Other values: reserved. Device will set bFBOProgressState to 00h (Idle) upon setting bFBOControl to 0x00.	

ATTRIBUTES							
IDN	**Name**	**Access Property**	**Size**	**Type[1]** **# Ind.[2]** **# Sel.[3]**	**MDV[4]**	**Description**	**Notes**

NOTE 1 The type "D" identifies a device level attribute, while the type "A" identifies an array of attributes. If Type = "D", the attribute is addressed setting INDEX = 00h and SELECTOR = 00h.

NOTE 2 For array of attributes, "# Ind." specifies the amount of valid values for the INDEX field in QUERY REQUEST/RESPONSE UPIU. If # Ind = 0, the attribute is addressed setting INDEX = 00h.

NOTE 3 For array of attributes, "# Sel." specifies the amount of valid values for the SELECTOR field in QUERY REQUEST/RESPONSE UPIU. If # Sel = 0, the attribute is addressed setting SELECTOR = 00h.

NOTE 4 The column "MDV" (Manufacturer Default Value) specifies attribute values after device manufacturing.

NOTE 5 bCurrentPowerMode value after device initialization may be: 20h (Pre-Sleep mode) or 22h (UFS-Sleep mode) if bInitPowerMode = 00h, or 11h (Active Mode) if bInitPowerMode = 01h.

NOTE 6 After power on or reset, bActiveICCLevel is equal to bInitActiveICCLevel parameter value included in the Device Descriptor. bInitActiveICCLevel is equal to 00h after device manufacturing and it can be configured by writing the Configuration Descriptor.

NOTE 7 bMaxDataInSize = bMaxInBufferSize when the UFS device is shipped.

NOTE 8 If the host attempts to write this Attribute when there is at least one logical unit with command queue not empty, the operation shall fail, and Response field in the QUERY RESPONSE UPIU shall be set to FFh ("General failure").

NOTE 9 dDynCapNeeded is composed by eight elements, one for each logical unit. The desired element shall be selected assigning the LUN to INDEX field of QUERY REQUEST UPIU.

NOTE 10 bRefClkFreq field had "Write once" Access Property up to UFS 2.0.

NOTE 11 bDeviceFFUStatus value is kept after power cycle, hardware reset or any other type of reset.This attribute may change value when a microcode activation event occurs.

NOTE 12 UFS Host may start a timer when DME_POWERMODE.ind is received for HS-MODE to LS-MODE transition or DME_HIBERNATE_ENTER.ind is received for HS-MODE to HIBERN8 transition. In addition to bRefClkGatingWaitTime, Device PA_MinRxTrailingClocks and Host PA_MinRxTrailingClocks should be considered to determine when the reference clock may be stopped.

NOTE 13 If this attribute is set to value '00h', it means the minimum wait time for reference clock removal is not specified by the device.

NOTE 14 If the host attempts to write bRefreshMethod when dRefreshProgress is not zero, the operation shall fail, and Response field in the QUERYRESPONSE UPIU shall be set to FFh ("General failure").

NOTE 15 If the bWriteBoosterBufferType is configured as 01h (shared type), the WriteBooster Buffer is configured as a single shared buffer for the whole device. In this case, the value of LU does not matter.

Standard Improvement Form JEDEC _____

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®