UFS Unified Memory Extension

Masafumi Takahashi
Toshiba Corporation

Mobile Forum 2014

Copyright © 2014 Toshiba Corporation
UFS Unified Memory Extension (UME) is OUT NOW!!

- **UFS UME**, an optional feature for UFS 2.0, consists of two standards.
  - Device side: JESD220-1
  - Host (HCI) side: JESD223-1

- Available on JEDEC web site!!
Outline

• UFS Unified Memory Extension (UME) Overview
  – What is Unified Memory Extension?
• Use cases
  – Example usages
• Summary
UFS Unified Memory Extension

• Introduce Unified Memory Architecture into UFS system.
  – Both host and device are involved.

• Provide means to use system memory as a working memory for UFS device.
  – Device get a large working memory in the system memory.
**Unified Memory Architecture (UMA) and UFS**

- **UMA**: Treat memories in the system as *unified*.
- **UFS**: Suitable for UMA
  - Fast communication speed
  - Rich features
    - Full duplex.
    - SCSI compliant commands.
    - Multiple outstanding.
  - For cost-sensitive mobile devices.
Scenario to Achieve Higher Performance with UM

- Store some data in UM, and reduce the number of accesses to NAND Flash.
- Access latency gets shorter.
  - Access latency to NAND Flash is relatively large compared against access latency to host RAM.
- Efficiency of NAND Flash is improved.
  - Other accesses to NAND Flash can be issued.

• Achieve HIGHER performance!!
Target Area

- SSD
- UFS w/ UME
- UFS
- eMMC
UFS and UFS UME

Host
UFS HCI
UFS IF
Device Controller
Device

UPIU: UFS Protocol Information Unit

System Memory

UM: Unified Memory

UMPIU: Unified Memory Protocol Information Unit

Global Standards for the Microelectronics Industry
Unified Memory Protocol Information Unit (UMPIU)

- Protocol to realize UME
  - Similar to the UPIU; protocol for basic data access in UFS
- Eight transaction codes are defined
  - Device and HCI use them to manipulate data in UM

<table>
<thead>
<tr>
<th>UMPIU</th>
</tr>
</thead>
<tbody>
<tr>
<td>COPY DATA</td>
</tr>
<tr>
<td>UM COPY DATA</td>
</tr>
<tr>
<td>ACKNOWLEDGE COPY</td>
</tr>
<tr>
<td>ACCESS UM BUFFER</td>
</tr>
<tr>
<td>WRITE UM BUFFER</td>
</tr>
<tr>
<td>ACKNOWLEDGE UM BUFFER</td>
</tr>
<tr>
<td>UM DATA OUT</td>
</tr>
<tr>
<td>UM DATA IN</td>
</tr>
</tbody>
</table>
Outline

• UFS Unified Memory Extension (UME) Overview
  – What is Unified Memory Extension?

• Use cases
  – Example usage

• Summary
Example usage of Unified Memory (UM)

- **Store User Data**
  - *Write Buffer*: Store write data from host
  - *Data Cache*: Cache data transferred from / to host

- **Store System Data**
  - *FTL Data Cache*: Cache FTL data and reduce latency of address translations.
  - Store data for background operations
  - and so on...

FTL: Flash (File) Translation Layer
Example Use Cases

- Write Buffer (SCSI WRITE)
- FTL Data Cache (SCSI READ)
Example Use Cases

• Write Buffer (SCSI WRITE)
  – Write data from host is once stored in UM, and write back it to the device at the timing of the device desired.
  – Reduce number of NAND accesses and make command response time shorter.

• FTL Data Cache (SCSI READ)
SCSI Write (1/3)
(Issue WRITE Command and Ready to Transfer Write Data)
SCSI Write (2/3)
(Transfer Write Data and Send Response)

UFS w/o UFS UME

- Host
- UFS
- Device
- (4) RESPONSE
- (3) DATA OUT

UFS w/ UFS UME

- Host
- UFS
- Device
- (5) RESPONSE
- (3) Data Copy
- (4) ACK. COPY
SCSI Write (3/3)
(Another Write and Write Back)

UFS w/o UFS UME

1. Host
2. Buffer is FULL
3. Need Write Back
4. Device
5. Write Back

UFS w/ UFS UME

1. Host
2. Device
3. Buffer is not FULL
4. Device
5. Write Back
SCSI Write Summary  
(Write Buffer in UM)

• Buffering write data in UM works effectively.
  – Device can make write buffer in UM larger than the one in device controller.

• It leads to higher performance!!
  – Efficiency of UFS link is improved.
    • In particular, for random accesses
  – Efficiency of NAND Flash is improved.
    • No need to write small data to NAND Flash.
    • Writing bigger data at once, and bandwidth gets wider.
Example Use Cases

• Write Buffer (SCSI WRITE)

• FTL Data Cache (SCSI READ)
  – Use cached FTL data in UM to translate address of the requested data.
  – Reduce number of accesses to NAND Flash and command response time becomes shorter than using FTL data in NAND Flash.
SCSI Read (1/3)
(Issue Read Command)
SCSI Read (2/3)
(Get FTL data and Resolve Physical Address)

**UFS w/o UFS UME**
- Host
- Device
- (2) Get FTL data from NAND (up to 100 us)

**UFS w/ UFS UME**
- Host
- Device
- (2) ACCESS UM BUFFER
- (3) UM DATA IN
- (4) ACK. UM BUFFER
- Get FTL data from UM (up to 10 us)
SCSI Read (3/3)
(Read out and Transfer User Data)
## SCSI Read Summary (1/2)

(FTL data cache in UM)

### UFS w/o UFS UME

- **Host**
- **Device**
- **FTL Data**

- Takes up to 100 us

### UFS w/ UFS UME

- **Host**
- **Device**
- **FTL Data**

- Takes up to 100 us

---

Global Standards for the Microelectronics Industry
SCSI Read Summary (2/2)
(FTL data cache in UM)

- Number of reads from NAND Flash is reduced.
  - With UM, FTL data need to address translations are read from UM.
- It leads to higher performance!!
  - Latency of reading from NAND Flash is much larger than latency of reading from UM.
  - Efficiency of NAND Flash is improved.
    - Other operations, such as an user data read, can be issued instead of the FTL data read.
- In our experimental result, over 60 kIOPS was achieved in random read with FTL data cache in UM*.

## Summary (1/2)

<table>
<thead>
<tr>
<th></th>
<th>UFS</th>
<th>UFS UME (Optional)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Work memory</td>
<td>In device controller</td>
<td>In host system</td>
</tr>
<tr>
<td>Protocol unit</td>
<td>UPIU</td>
<td>UMPIU</td>
</tr>
<tr>
<td>Initiator</td>
<td>Host</td>
<td>Device</td>
</tr>
<tr>
<td>Complexity</td>
<td>-</td>
<td>Extra communication</td>
</tr>
<tr>
<td>Performance</td>
<td>-</td>
<td>Faster random access</td>
</tr>
<tr>
<td>Power consumption</td>
<td>-</td>
<td>Lower energy consumption in device</td>
</tr>
<tr>
<td>Cost</td>
<td>-</td>
<td>Smaller device controller memory</td>
</tr>
<tr>
<td>Standard</td>
<td>UFS 2.0, UFS HCI 2.0</td>
<td>UFS UME 1.0, UFS HCI UME 1.0</td>
</tr>
</tbody>
</table>
• Again, UFS UME is now available!!
• With UFS UME, you can get UFS system with higher performance.
  – Result of the co-operation between host and device.
Thank You!