Universal Flash Storage

1. Overview

Universal Flash Storage (UFS) is a storage specification for flash devices. It aims to provide a universal storage interface for both embedded and removable flash memory-based storage in mobile devices such as smart phones and tablet computers. The specification is defined by JEDEC Solid State Technology Association. UFS is based on the MIPI M-PHY physical layer standard. UFS uses MIPI M-PHY as the physical layer and MIPI Unipro as the link layer.

The main goals of UFS are to provide:

- Optimized performance:

  For UFS version 1.0 and 1.1 the target performance is as follows:

  - Support for Gear1 is mandatory (rate A: 1248Mbps, rate B: 1457.6Mbps)
  - Support for Gear2 is optional (rate A: 2496Mbps, rate B: 2915.2Mbps)

  Future version of the standard,

  - Gear3 (rate A: 4992Mbps, rate B: 5830.4Mbps)
- Low power consumption
- High random IOPs and low latency

2. UFS Architecture Overview

UFS has a layered communication architecture which is based on SCSI SAM-5 architectural model.

UFS communication architecture consists of the following layers.

2.1 Application Layer

The Application layer is composed of the UFS command set layer (UCS), Task Manager and Device manager. The UFS interface is designed to be protocol agnostic, however SCSI has been selected as a baseline protocol for versions 1.0 and 1.1 of the UFS protocol layer.

UFS supports a subset of SCSI commands defined by SPC-4 and SBC-3.

- UCS:
  It handles SCSI commands supported by UFS specification.
- Task manager:
  It handles task management functions defined by the UFS which are meant for command queue control.
- Device manager:
  It handles device level operations and device configuration operations. Device level operations mainly involve device power management operations and commands to Interconnect layers. Device level configurations involve handling of query requests which are used to modify and retrieve configuration information of the device.

2.2 UFS Transport Protocol (UTP) layer

The UTP layer provides services for the higher layers through Service Access Points. UTP defines 3 service access points for higher layers.

- UDM_SAP: Device manager service access point is exposed to device manager for device level operations. These device level operations are done through query requests.
- UTP_CMD_SAP: Command service access point is exposed to UFS command set layer (UCS) to transport commands.
- UTP_TM_SAP: Task management service access point is exposed to task manager to transport task management functions.

UTP transports messages through UFS protocol information unit (UPIU).

2.3 UFS Interconnect (UIC) Layer

UIC is the lowest layer of the UFS layered architecture. It handles the connection between UFS host and UFS device. UIC consists of MIPI UniPro and MIPI M-PHY. UIC provides 2 service access points to upper layer:

- UIC_SAP: To transport UPIU between UFS host and UFS device.

- UIO_SAP: To issue commands to Unipro layers.

3. UFSHCD Overview

The UFS host controller driver is based on the Linux SCSI Framework. UFSHCD is a low-level device driver which acts as an interface between the SCSI Midlayer and PCIe-based UFS host controllers.

The current UFSHCD implementation supports the following functionality:

3.1 UFS controller initialization

The initialization module brings the UFS host controller to active state and prepares the controller to transfer commands/responses between UFSHCD and UFS device.

3.2 UTP Transfer requests

Transfer request handling module of UFSHCD receives SCSI commands from the SCSI Midlayer, forms UPIUs and issues the UPIUs to the UFS Host controller. Also, the module decodes responses received from the UFS host controller in the form of UPIUs and intimates the SCSI Midlayer of the status of the command.

3.3 UFS error handling

Error handling module handles Host controller fatal errors, Device fatal errors and UIC interconnect layer-related errors.

3.4 SCSI Error handling

This is done through UFSHCD SCSI error handling routines registered with the SCSI Midlayer. Examples of some of the error handling commands issues by the SCSI Midlayer are Abort task, LUN reset and host reset. UFSHCD Routines to perform these tasks are registered with SCSI Midlayer through .eh_abort_handler, .eh_device_reset_handler and .eh_host_reset_handler.

In this version of UFSHCD, Query requests and power management functionality are not implemented.

4. BSG Support

This transport driver supports exchanging UFS protocol information units (UPIUs) with a UFS device. Typically, user space will allocate struct ufs_bsg_request and struct ufs_bsg_reply (see ufs_bsg.h) as request_upiu and reply_upiu respectively. Filling those UPIUs should be done in accordance with JEDEC spec UFS2.1 paragraph 10.7. *Caveat emptor*: The driver makes no further input validations and sends the UPIU to the device as it is. Open the bsg device in /dev/ufs-bsg and send SG_IO with the applicable sg_io_v4:

```
io_hdr_v4.guard = 'Q';
io_hdr_v4.protocol = BSG_PROTOCOL_SCSI;
io_hdr_v4.subprotocol = BSG_SUB_PROTOCOL_SCSI_TRANSPORT;
io_hdr_v4.response = (__u64)reply_upiu;
io_hdr_v4.max_response_len = reply_len;
io_hdr_v4.request_len = request_len;
io_hdr_v4.request = (__u64)request_upiu;
if (dir == SG_DXFER_TO_DEV) {
        io_hdr_v4.dout_xfer_len = (uint32_t)byte_cnt;
        io_hdr_v4.dout_xferp = (uintptr_t)(__u64)buff;
} else {
        io_hdr_v4.din_xfer_len = (uint32_t)byte_cnt;
        io_hdr_v4.din_xferp = (uintptr_t)(__u64)buff;
}
```

If you wish to read or write a descriptor, use the appropriate xferp of sg_io_v4.

The userspace tool that interacts with the ufs-bsg endpoint and uses its UPIU-based protocol is available at:

https://github.com/westerndigitalcorporation/ufs-tool

For more detailed information about the tool and its supported features, please see the tool's README.

UFS specifications can be found at:

- UFS - http://www.jedec.org/sites/default/files/docs/JESD220.pdf
- UFSHCI - http://www.jedec.org/sites/default/files/docs/JESD223.pdf